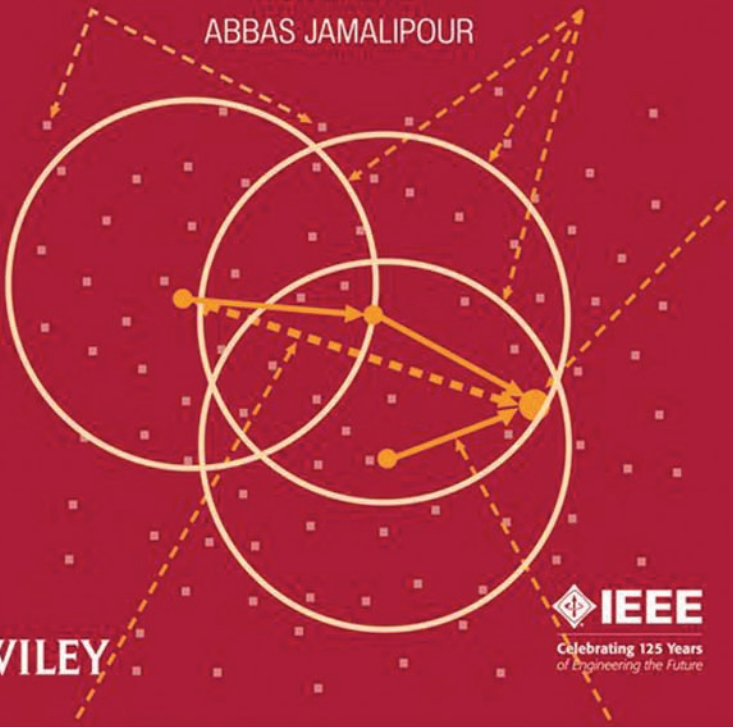
WIRELESS SENSOR NETWORKS

A Networking Perspective

JUN ZHENG
ABBAS JAMALIPOUR



WILEY

IEEE
Celebrating 125 Years
of Engineering the Future

WIRELESS SENSOR NETWORKS

A Networking Perspective

Edited by

Jun Zheng

Abbas Jamalipour

A JOHN WILEY & SONS, INC., PUBLICATION

WIRELESS SENSOR NETWORKS

IEEE Press
445 Hoes Lane
Piscataway, NJ 08854

WIRELESS SENSOR NETWORKS

A Networking Perspective

Edited by

Jun Zheng

Abbas Jamalipour

A JOHN WILEY & SONS, INC., PUBLICATION

Published by John Wiley & Sons, Inc., Hoboken, New Jersey.
Published simultaneously in Canada.

Library of Congress Cataloging-in-Publication Data is available.

ISBN: 978-0-470-16763-2

Printed in the United States of America.

10 9 8 7 6 5 4 3 2 1

To our parents and family

CONTENTS

PREFACE

Wireless sensor networking is an emerging technology that promises a wide range of potential applications in both civilian and military areas. A wireless sensor network (WSN) typically consists of a large number of low-cost, low-power, and multifunctional sensor nodes that are deployed in a region of interest. These sensor nodes are small in size but are equipped with sensors, embedded microprocessors, and radio transceivers. Therefore, they have not only sensing, but also data processing and communicating capabilities. They communicate over short distance via a wireless medium and collaborate to accomplish a common task, for example, environment monitoring, military surveillance, and industrial process control. In many WSN applications, the deployment of sensor nodes is performed in an ad hoc fashion without careful preplanning and engineering. Once deployed, the sensor nodes must be able to autonomously organize themselves into a wireless communication network. In particular, sensor nodes are typically battery-powered and should operate without attendance for a relatively long period of time. In most cases, it is very difficult and even impossible to change or recharge batteries for the sensor nodes. Distinguished from traditional wireless networks, WSNs are characterized with denser levels of node deployment, higher unreliability of sensor nodes, and severe power, computation, and memory constraints. The unique characteristics and constraints present many new challenges for the development and application of WSNs. Due to the wide range of potential applications, WSNs have received tremendous attentions from both academia and industry all over the world in recent years. A voluminous amount of research activities have been carried out to explore and solve various design and application issues, and significant advances have been made in the development and deployment of WSNs. It is envisioned that WSNs will change the way we live, work, and interact with the physical world in the near future.

The purpose of this book is to provide a comprehensive and systematical introduction of the fundamental concepts, major issues, and effective solutions in wireless sensor networking. Distinguished from other books, this book focuses on the networking aspects of WSNs and covers the most important networking issues, including network architecture design, medium access control, routing and data dissemination, node clustering, query processing and data aggregation, node localization, time synchronization, transport and quality of service, energy efficiency, network security, and sensor network standards.

This book is intended for a wide range of audience, including academic researchers, graduate students, practitioners in industry, and research engineers.

It can be an excellent source of information for academic researchers, industry practitioners, and research engineers who are working in the area of wireless ad hoc and sensor networks to learn the state-of-the-art technologies in the networking aspect of WSNs. It can be used as a textbook or supplementary reading for relevant graduate level courses in electrical engineering, computer engineering, and computer science, for example, wireless sensor networks, wireless ad hoc networks, or wireless networks. It can also be used as a textbook for self-study by professionals who are not working in the field but would like to learn more about wireless sensor networks.

Jun Zheng and Abbas Jamalipour
May 1, 2009

ACKNOWLEDGMENTS

This book would not have been possible without the contribution, support, and encouragement from many people in their different ways.

First of all, we would like to thank all chapter authors for contributing their excellent work to this book. Without their contributions, this book would not have been possible. Thanks are also given to many anonymous reviewers for carefully reviewing all chapters, and providing constructive and valuable comments.

We are grateful to the editors and production group at Wiley-IEEE Press for their enthusiastic support for this project. The publication of this book would not have been possible without the vision and foresight of our Senior Acquisitions Editors, Catherine Faduska and Steve Welch, and Wiley-IEEE Press. Many thanks go to Steve Welch for his efforts throughout the whole process of this project, to our Project Editor, Jeanne Audino, and our Senior Production Editor, Melissa Yanuzzi, for coordinating the production of the book, to our Marketing Manager, Sika Dunyoh, for managing the marketing plan for the book, and to our Copyeditor, Jeannette Stiefel, for carefully copyediting the whole manuscript.

Special thanks are given to a number of friends and colleagues for their support and encouragement during the editing of the book. We will not forget their gentle reminding messages that there are more beautiful things in life beyond working.

Last but most, we would like to express our deep gratitude to our family for their invaluable love, and continuous support and encouragement throughout the whole editing process of this book.

ABOUT THE EDITORS

Jun Zheng is a Full Professor with the School of Information Science and Engineering of Southeast University, China. He received his Ph.D. degree in Electrical and Electronic Engineering from The University of Hong Kong, China. Before joining Southeast University, he was with the School of Information Technology and Engineering of the University of Ottawa, Canada.

He is an Associate Technical Editor of IEEE Communications Magazine, an Editor of IEEE Communications Surveys & Tutorials, and an Associate Editor of IEEE/OSA Journal of Optical Communications and Networking. He is also the founding Editor-in-Chief of ICST Transactions on Mobile Communications and Applications, and an Associate Editor of several other refereed journals, including Wiley Wireless Communications and Mobile Computing, Wiley Security and Communication Networks, Inderscience International Journal of Communication Networks and Distributed Systems, and Inderscience International Journal of Autonomous and Adaptive Communications Systems. He has guest-edited eight special issues for different refereed journals and magazines, including IEEE Network, IEEE Journal on Selected Areas in Communications, Wiley Wireless Communications and Mobile Computing, Wiley International Journal of Communication Systems, and Springer Mobile Networks and Applications, all as Lead Guest Editor.

He has served as the founding General Chair of AdHoctNets'09, General Chair of AccessNets'07, TPC Co-Chair of AccessNets'08, Symposium Co-Chair of IEEE GLOBECOM'08, IEEE ICC'09, and IEEE GLOBECOM'10 respectively. He is also serving on the steering committees of AdHocNets and AccessNets, and has served on the technical program committees of a number of international conferences and symposia, including IEEE ICC and IEEE GLOBECOM.

Dr. Zheng has conducted extensive research in the field of telecommunications and computer communication networks. The scope of his research includes design and analysis of network architectures and protocols for efficient and reliable communications, and their applications to different types of communication networks, covering wireless networks, optical networks, and IP networks. His current research interests focus on wireless ad hoc and sensor networks. He has co-authored (first author) a book published by Wiley-IEEE Press, and has published a number of technical papers in refereed journals and magazines as well as peer-reviewed conference proceedings. Dr. Zheng is a senior member of IEEE.

Abbas Jamalipour holds a Ph.D. from Nagoya University, Japan. He is the author of the first book on wireless IP, two other books, and has co-authored 7 books and over 190 technical papers, all in the field of mobile communications network. He is a Fellow of IEEE (for contributions to next generation networks for traffic control), a Fellow of Institute of Engineers Australia, an IEEE Distinguished Lecturer, the Editor-in-Chief of the IEEE Wireless Communications, and a Technical Editor of several scholarly journals including IEEE Communications, Wiley International Journal of Communication Systems, Journal of Communication Network, and so on. His areas of research are wireless data communication networks, wireless IP networks, wireless sensor networks, next generation mobile networks, traffic control, network security and management, and satellite systems. He was one of the first researchers to disseminate the fundamental concepts of the next generation mobile networks and broadband convergence networks as well as the integration of wireless LAN and cellular networks; some of which are being gradually deployed by industry and included in the ITU-T standards. Dr. Jamalipour has authored several invited papers and been a keynote speaker in many prestigious conferences. He served as the Chair of the Satellite and Space Communications Technical Committee (2004–2006), and currently is the Vice Chair of Communications Switching and Routing TC, and Chair of Chapters Coordinating Committee, Asia-Pacific Board, all from the IEEE Communications Society. He is a voting member of the IEEE GITC and IEEE WCNC Steering Committee. He has been a Vice Chair of IEEE WCNC2003 to 2006, Program Chair of SPECTS2004, Chair of symposiums at IEEE GLOBECOM2005 to 2007 and IEEE ICC2005 to 2008, among many conference leadership roles. He has received several prestigious awards, for example, 2006 IEEE Distinguished Contribution to Satellite Communications Award, 2006 IEEE Communications Society Best Tutorial Paper Award, and 2005 Telstra Award for Excellence in Teaching.

CONTRIBUTORS

Dharma P. Agrawal University of Cincinnati, USA
Nayef A. Alsindi Worcester Polytechnic Institute, USA
Habib M. Ammari Hofstra University, USA
Nirwan Ansari New Jersey Institute of Technology, USA
Torsha Banerjee University of Cincinnati, USA
Stefano Chessa University of Pisa, Italy
Sajal K. Das University of Texas at Arlington, USA
Yuguang Fang University of Florida, USA
Edwin Hou New Jersey Institute of Technology, USA
Abbas Jamalipour University of Sydney, Australia
Bo Li Hong Kong University of Science and Technology, China
Tommaso Melodia State University of New York at Buffalo, USA
Kaveh Pahlavan Worcester Polytechnic Institute, USA
Nikolaos A. Pantazis Technological Educational Institute of Athens, Greece
Dario Pompili Rutgers, The State University of New Jersey, USA
Fikret Sivrikaya Rensselaer Polytechnic Institute, USA
Kazem Sohraby University of Arkansas at Fayetteville, USA
Dimitrios D. Vergados University of Piraeus, Greece
Mehmet Can Vuran University of Nebraska-Lincoln, USA
Chonggang Wang University of Arkansas at Fayetteville, USA
Guoliang Xue Arizona State University, USA
Bülent Yener Rensselaer Polytechnic Institute, USA
Baoxian Zhang Chinese Academy of Sciences, China
Chao Zhang New Jersey Institute of Technology, USA
Jun Zheng Southeast University, China
Yun Zhou University of Florida, USA

1

INTRODUCTION TO WIRELESS SENSOR NETWORKS

Jun Zheng

Southeast University, China

Abbas Jamalipour

University of Sydney, Australia

1.1 OVERVIEW OF WIRELESS SENSOR NETWORKS

Wireless Sensor Networks (WSNs) have been widely considered as one of the most important technologies for the twenty-first century [1]. Enabled by recent advances in microelectronicmechanical systems (MEMS) and wireless communication technologies, tiny, cheap, and smart sensors deployed in a physical area and networked through wireless links and the Internet provide unprecedented opportunities for a variety of civilian and military applications, for example, environmental monitoring, battle field surveillance, and industry process control [2]. Distinguished from traditional wireless communication networks, for example, cellular systems and mobile ad hoc networks (MANET), WSNs have unique characteristics, for example, denser level of node deployment, higher unreliability of sensor nodes, and severe energy, computation, and storage constraints [3], which present many new challenges in the development and application of WSNs. In the past decade, WSNs have received tremendous attention from both

Wireless Sensor Networks: A Networking Perspective, Edited by Jun Zheng and Abbas Jamalipour

academia and industry all over the world. A large amount of research activities have bee carried out to explore and solve various design and application issues, and significant advances have been made in the development and deployment of WSNs. It is envisioned that in the near future WSNs will be widely used in various civilian and military fields, and revolutionize the way we live, work, and interact with the physical world [4].

1.1.1 Network Characteristics

A WSN typically consists of a large number of low-cost, low-power, and multi-functional sensor nodes that are deployed in a region of interest. These sensor nodes are small in size, but are equipped with sensors, embedded microprocessors, and radio transceivers, and therefore have not only sensing capability, but also data processing and communicating capabilities. They communicate over a short distance via a wireless medium and collaborate to accomplish a common task, for example, environment monitoring, battlefield surveillance, and industrial process control. Compared with traditional wireless communication networks, for example, cellular systems and MANET, sensor networks have the following unique characteristics and constraints:

- *Dense Node Deployment.* Sensor nodes are usually densely deployed in a field of interest. The number of sensor nodes in a sensor network can be several orders of magnitude higher than that in a MANET.
- *Battery-Powered Sensor Nodes.* Sensor nodes are usually powered by battery. In most situations, they are deployed in a harsh or hostile environment, where it is very difficult or even impossible to change or recharge the batteries.
- *Severe Energy, Computation, and Storage Constraints.* Sensor nodes are highly limited in energy, computation, and storage capacities.
- *Self-Configurable.* Sensor nodes are usually randomly deployed without careful planning and engineering. Once deployed, sensor nodes have to autonomously configure themselves into a communication network.
- *Application Specific.* Sensor networks are application specific. A network is usually designed and deployed for a specific application. The design requirements of a network change with its application.
- *Unreliable Sensor Nodes.* Sensor nodes are usually deployed in harsh or hostile environments and operate without attendance. They are prone to physical damages or failures.
- *Frequent Topology Change.* Network topology changes frequently due to node failure, damage, addition, energy depletion, or channel fading.
- *No Global Identification.* Due to the large number of sensor nodes, it is usually not possible to build a global addressing scheme for a sensor network because it would introduce a high overhead for the identification maintenance.

- *Many-to-One Traffic Pattern.* In most sensor network applications, the data sensed by sensor nodes flow from multiple source sensor nodes to a particular sink, exhibiting a many-to-one traffic pattern.
- *Data Redundancy.* In most sensor network applications, sensor nodes are densely deployed in a region of interest and collaborate to accomplish a common sensing task. Thus, the data sensed by multiple sensor nodes typically have a certain level of correlation or redundancy.

The unique characteristics and constraints present many new challenges in the design of sensor networks.

1.1.2 Network Applications

Sensors can be used to detect or monitor a variety of physical parameters or conditions [5], for example,

- Light
- Sound
- Humility
- Pressure
- Temperature
- Soil composition
- Air or water quality
- Attributes of an object such as size, weight, position, speed, and direction.

Wireless sensors have significant advantages over conventional wired sensors [6]. They can not only reduce the cost and delay in deployment, but also be applied to any environment, especially those in which conventional wired sensor networks are impossible to be deployed, for example, inhospitable terrains, battlefields, outer space, or deep oceans. WSNs were originally motivated by military applications, which range from large-scale acoustic surveillance systems for ocean surveillance to small networks of unattended ground sensors for ground target detection [1]. However, the availability of low-cost sensors and wireless communication has promised the development of a wide range of applications in both civilian and military fields. This section introduces a few examples of sensor network applications.

1.1.2.1 Environmental Monitoring. Environmental monitoring is one of the earliest applications of sensor networks. In environmental monitoring, sensors are used to monitor a variety of environmental parameters or conditions.

- *Habitat Monitoring.* Sensors can be used to monitor the conditions of wild animals or plants in wild habitats, as well as the environmental parameters

of the habitats. For example, Mainwaring et al. [7], from the University of California at Berkeley and the college of the Atlantic in Bar Harbor, conducted an experiment to monitor the habitat of the nesting petrels on Great Duck Land in Maine by deploying 190 wireless sensors, including humidity, pressure, temperature, and radiation.

- *Air or Water Quality Monitoring.* Sensors can be deployed on the ground or under water to monitor air or water quality. For example, water quality monitoring can be used in the hydrochemistry field. Air quality monitoring can be used for air pollution control.
- *Hazard Monitoring.* Sensors can be used to monitor biological or chemical hazards in locations, for example, a chemical plant or a battlefield.
- *Disaster Monitoring.* Sensors can be densely deployed in an intended region to detect natural or non-natural disasters. For example, sensors can be scattered in forests or revivers to detect forest fires or floods. Seismic sensors can be instrumented in a building to detect the direction and magnitude of a quake and provide an assessment of the building safety.

1.1.2.2 Military Applications. WSNs are becoming an integral part of military command, control, communication, and intelligence (C3I) systems [5]. Wireless sensors can be rapidly deployed in a battlefield or hostile region without any infrastructure. Due to ease of deployment, self-configurability, untended operation, and fault tolerance, sensor networks will play more important roles in future military C3I systems and make future wars more intelligent with less human involvement.

- *Battlefield Monitoring.* Sensors can be deployed in a battlefield to monitor the presence of forces and vehicles, and track their movements, enabling close surveillance of opposing forces.
- *Object Protection.* Sensor nodes can be deployed around sensitive objects, for example, atomic plants, strategic bridges, oil and gas pipelines, communication centers, and military headquarters, for protection purpose.
- *Intelligent Guiding.* Sensors can be mounted on unmanned robotic vehicles, tanks, fighter planes, submarines, missiles, or torpedoes to guide them around obstacles to their targets and lead them to coordinate with one another to accomplish more effective attacks or defences.
- *Remote Sensing.* Sensors can be deployed for remote sensing of nuclear, biological, and chemical weapons, detection of potential terrorist attacks, and reconnaissance [5].

1.1.2.3 Health Care Applications. WSNs can be used to monitor and track elders and patients for health care purposes, which can significantly relieve the severe shortage of health care personnel and reduce the health care expenditures in the current health care systems [8].

- *Behavior Monitoring.* Sensors can be deployed in a patient's home to monitor the behaviors of the patient. For example, it can alert doctors when the patient falls and requires immediate medical attention. It can monitor what a patient is doing and provide reminders or instructions over a television or radio.
- *Medical Monitoring.* Wearable sensors can be integrated into a wireless body area network (WBAN) to monitor vital signs, environmental parameters, and geographical locations, and thus allow long-term, noninvasive, and ambulatory monitoring of patients or elderly people with instantaneous alerts to health care personal in case of emergency, immediate reports to users about their current health statuses, and real-time updates of users' medical records [9].

1.1.2.4 Industrial Process Control. In industry, WSNs can be used to monitor manufacturing processes or the condition of manufacturing equipment. For example, wireless sensors can be instrumented to production and assembly lines to monitor and control production processes. Chemical plants or oil refiners can use sensors to monitor the condition of their miles of pipelines. Tiny sensors can be embedded into the regions of a machine that are inaccessible by humans to monitor the condition of the machine and alert for any failure. Traditionally, equipment is usually maintained on a schedule basis, for example, every 3 months for a check-up, which is costly. According to related statistics, a US equipment manufacturer spends billions of dollars in maintenance every year [6]. With sensor networks, maintenance can be conducted based on the condition of equipment, which is expected to significantly reduce the cost for maintenance, increase machine lifetime, and even save lives.

1.1.2.5 Security and Surveillance. WSNs can be used in many security and surveillance applications. For example, acoustic, video, and other kinds of sensors can be deployed in buildings, airports, subways, and other critical infrastructure, for example, nuclear power plants or communication centers to identify and track intruders, and provide timely alarms and protection from potential attacks. Unlike applications that do not require a fixed infrastructure, many security applications can afford to establish an infrastructure for power supply and communications [6].

1.1.2.6 Home Intelligence. WSNs can be used to provide more convenient and intelligent living environments for human beings.

- *Smart Home.* Wireless sensors can be embedded into a home and connected to form an autonomous home network. For example, a smart refrigerator connected to a smart stove or microwave oven can prepare a menu based on the inventory of the refrigerator and send relevant cooking

parameters to the smart stove or microwave oven, which will set the desired temperature and time for cooking [10]. The contents and schedules of TV, VCR, DVD, or CD players can be monitored and controlled remotely to meet the different requirements of family members.

- *Remote Metering.* Wireless sensors can be used to remotely read utility meters in a home, for example, water, gas, or electricity, and then send the readings to a remote center through wireless communication [11].

In addition to the above applications, self-configurable WSNs can be used in many other areas, for example, disaster relief, traffic control, warehouse management, and civil engineering. However, a number of technical issues must be solved before these exciting applications become a reality.

1.1.3 Network Design Objectives

The characteristics of sensor networks and requirements of different applications have a decisive impact on the network design objectives in terms of network capabilities and network performance. The main design objectives for sensor networks include the following several aspects:

- *Small Node Size.* Reducing node size is one of the primary design objectives of sensor networks. Sensor nodes are usually deployed in a harsh or hostile environment in large numbers. Reducing node size can facilitate node deployment, and also reduce the cost and power consumption of sensor nodes.
- *Low Node Cost.* Reducing node cost is another primary design objective of sensor networks. Since sensor nodes are usually deployed in a harsh or hostile environment in large numbers and cannot be reused, it is important to reduce the cost of sensor nodes so that the cost of the whole network is reduced.
- *Low Power Consumption.* Reducing power consumption is the most important objective in the design of a sensor network. Since sensor nodes are powered by battery and it is often very difficult or even impossible to change or recharge their batteries, it is crucial to reduce the power consumption of sensor nodes so that the lifetime of the sensor nodes, as well as the whole network is prolonged.
- *Self-Configurability.* In sensor networks, sensor nodes are usually deployed in a region of interest without careful planning and engineering. Once deployed, sensor nodes should be able to autonomously organize themselves into a communication network and reconfigure their connectivity in the event of topology changes and node failures.
- *Scalability.* In sensor networks, the number of sensor nodes may be on the order of tens, hundreds, or thousands. Thus, network protocols designed for sensor networks should be scalable to different network sizes.

- *Adaptability.* In sensor networks, a node may fail, join, or move, which would result in changes in node density and network topology. Thus, network protocols designed for sensor networks should be adaptive to such density and topology changes.
- *Reliability.* For many sensor network applications, it is required that data be reliably delivered over noisy, error-prone, and time-varying wireless channels. To meet this requirement, network protocols designed for sensor networks must provide error control and correction mechanisms to ensure reliable data delivery.
- *Fault Tolerance.* Sensor nodes are prone to failures due to harsh deployment environments and unattended operations. Thus, sensor nodes should be fault tolerant and have the abilities of self-testing, self-calibrating, self-repairing, and self-recovering [12].
- *Security.* In many military applications, sensor nodes are deployed in a hostile environment and thus are vulnerable to adversaries. In such situations, a sensor network should introduce effective security mechanisms to prevent the data information in the network or a sensor node from unauthorized access or malicious attacks.
- *Channel Utilization.* Sensor networks have limited bandwidth resources. Thus, communication protocols designed for sensor networks should efficiently make use of the bandwidth to improve channel utilization.
- *QoS Support.* In sensor networks, different applications may have different quality-of-service (QoS) requirements in terms of delivery latency and packet loss. For example, some applications, for example, fire monitoring, are delay sensitive and thus require timely data delivery. Some applications, for example, data collection for scientific exploration, are delay tolerant but cannot stand packet loss. Thus, network protocol design should consider the QoS requirements of specific applications.

Most sensor networks are application specific and have different application requirements. It is not necessary and actually impractical to implement all the design objectives in a single network. Instead, only part of these objectives should be considered in the design of a specific network in order to meet its application requirements.

1.1.4 Network Design Challenges

The unique network characteristics present many challenges in the design of sensor networks, which involve the following main aspects:

- *Limited Energy Capacity.* Sensor nodes are battery powered and thus have very limited energy capacity. This constraint presents many new challenges

in the development of hardware and software, and the design of network architectures and protocols for sensor networks. To prolong the operational lifetime of a sensor network, energy efficiency should be considered in every aspect of sensor network design, not only hardware and software, but also network architectures and protocols.

- *Limited Hardware Resources.* Sensor nodes have limited processing and storage capacities, and thus can only perform limited computational functionalities. These hardware constraints present many challenges in software development and network protocol design for sensor networks, which must consider not only the energy constraint in sensor nodes, but also the processing and storage capacities of sensor nodes.
- *Massive and Random Deployment.* Most sensor networks consist of a large number of sensor nodes, from hundreds to thousands or even more. Node deployment is usually application dependent, which can be either manual or random. In most applications, sensor nodes can be scattered randomly in an intended area or dropped massively over an inaccessible or hostile region. The sensor nodes must autonomously organize themselves into a communication network before they start to perform a sensing task.
- *Dynamic and Unreliable Environment.* A sensor network usually operates in a dynamic and unreliable environment. On one hand, the topology of a sensor network may change frequently due to node failures, damages, additions, or energy depletion. On the other hand, sensor nodes are linked by a wireless medium, which is noisy, error prone, and time varying. The connectivity of the network may be frequently disrupted because of channel fading or signal attenuation.
- *Diverse Applications.* Sensor networks have a wide range of diverse applications. The requirements for different applications may vary significantly. No network protocol can meet the requirements of all applications. The design of sensor networks is application specific.

1.2 TECHNOLOGICAL BACKGROUND

The concept of WSNs was originally introduced three decades ago [2]. At that time, this concept was more a vision than a technology that could widely be exploited because of the state-of-the-art in sensor, computer, and wireless communication technologies. As a result, its application was mostly limited to large military systems. However, recent technological advances in MEMS, wireless communication, and low-cost manufacturing technologies have enabled the development of tiny, cheap, and smart sensors with sensing, processing, and communications capabilities, which has therefore stimulated the development of sensor networks and their applications.

1.2.1 MEMS Technology

MEMS is a key technology for manufacturing tiny, low-cost, and low-power sensor nodes. It is based on micromachining techniques, which have been developed to fabricate micron-scale mechanical components that are controlled electrically, resulting in MEMS. Through highly integrated processes, these electromechanical components can be fabricated with microelectronics, yielding complex systems. There are different micromachining techniques, for example, planar micromachining, bulk micromachining, and surface micromachining, which involve different fabrication processes [13,14]. Most micromachining processes begin with a substrate 100–100 μm thick, usually composed of silicon, other crystalline semiconductors, or quartz, on which a number of subsequent steps are performed, for example, thin-film deposition, photolithography, etching, oxidation, electroplating, machining, and wafer bonding. Different processes may involve different specific steps. By integrating different components together into a single process, the size of a sensor node can significantly be reduced. Of particular interest are the processes that combine CMOS transistors with micromachining capabilities. There are a number of techniques for performing post-process micromachining on foundry CMOS [15]. By using the MEMS technology, many components of sensor nodes can be miniaturized, for example, sensors, communication blocks, and power supply units, which can also lead to a significant reduction in cost through batch fabrication, as well as in power consumption. For a more detailed introduction of the MEMS technology and related techniques, the reader is referred to Refs. [13,14].

1.2.2 Wireless Communication Technology

Wireless communication is a key technology for enabling the normal operation of a WSN. Wireless communication has been extensively studied for conventional wireless networks in the last couple of decades and significant advances have been obtained in various aspects of wireless communication. At the physical layer, a variety of modulation, synchronization, and antenna techniques have been designed for different network scenarios and application requirements. At higher layers, efficient communication protocols have been developed to address various networking issues, for example, medium access control, routing, QoS, and network security. These communication techniques and protocols provide a rich technological background for the design of wireless communication in WSNs.

Today most conventional wireless networks use radio frequency (RF) for communication, including microwave and millimetre wave. The primary reason is that RF communication does not require a line of sight and provides omnidirectional links. However, RF has some limitations, for example, large radiators and low transmission efficiencies [16], which make RF not the best communication medium for tiny energy-constrained sensor nodes. Another possible medium for communication in sensor networks is free-space optical communication, which has many advantages over RF communication [16]. For example, optical

radiators, for example, mirrors and laser diodes, can be made extremely small. Optical transmission provides extremely high antenna gain, which produces higher transmission efficiencies. The high directivity of optical communication enables the use of spatial division multiple access (SDMA) [17], which requires no communication overhead and has the potential to be more energy efficient than the medium access schemes used in RF, such as time, frequency, and code division multiple access (TDMA, FDMA, and CDMA). However, optical communication requires a line of sight and accurate pointing for transmission, which also limit the use in many sensor network applications.

On the other hand, most communication protocols for conventional wireless networks, for example, cellular systems, wireless local area networks (WLANs), wireless personal area networks (WPANs), and MANETs, do not consider the unique characteristics of sensor networks, in particular, the energy constraint in sensor nodes. Therefore, they cannot be applied directly without modification. A new suite of network protocols are needed to address various networking issues, taking into account the unique characteristics of WSNs.

1.2.3 Hardware and Software Platforms

The development of WSNs largely depends on the availability of low-cost and low-power hardware and software platforms for sensor networks. With the MEMS technology, the size and cost of a sensor node have been significantly reduced. To achieve low-power consumption at the node level, it is necessary to incorporate power awareness and energy optimization in hardware design for sensor networks [18]. Low-power circuit and system design [19] has enabled the development of ultralow power hardware components, for example, microprocessors and microcontrollers. Meanwhile, power consumption can further be reduced through efficiently operating various system resources using some dynamic power management (DPM) technique [20]. For example, a commonly used DPM technique is to shutdown idle components or put them in a low-power state when there is little or no load to process, which can significantly reduce power consumption. Furthermore, additional energy savings also are possible in the active state by using a dynamic voltage scaling (DVS) technique [21]. It has been shown that DVS based power management has significantly higher energy efficiency compared to shutdown-based power management [18].

On the other hand, energy efficiency can significantly be enhanced if energy awareness is incorporated in the design of system software, including the operating system, and application and network protocols. At the core of the operating system is a task scheduler, which is responsible for scheduling a given set of tasks in the system under certain timing constraints. System lifetime can considerably be prolonged if energy awareness is incorporated into the task scheduling process [22].

The low-power circuit and system design, as well as power management techniques, have enabled the development of many low-power sensor hardware

and software platforms. The commercial availability of these platforms has significantly stimulated the further development of WSNs.

1.2.3.1 Hardware Platforms. Sensor node hardware platforms can be classified into three categories [6]: augmented general-purpose personal computers (PCs), dedicated sensor nodes, and system-on-chip (SoC) sensor nodes.

- *Augmented General-Purpose PCs.* This class of platforms include various low-power embedded PCs (e.g., PC104) and personal digital assistants (PDAs), which typically run off-the-shelf operating systems, for example, Win CE, Linux, or real-time operating systems, and use standard wireless communication protocols, for example, IEEE 802.11 or Bluetooth. Compared with dedicated and SoC sensor nodes, these PC-like platforms have higher processing capability and thus can incorporate a richer set of networking protocols, popular programming languages, middleware, application programming interfaces (APIs), and other off-the-shelf software. However, they require more power supply.
- *Dedicated Sensor Nodes.* This class of platforms include the Berkeley mote family [23], the UCLA Medusa family [24], and MIT μAMP [25], which typically use commercial off-the-shelf chips and are characterized by small form factors, low-power processing and communication, and simple sensor interfaces.
- *System-on-chip Sensor Nodes.* This class of platforms include Smart Dust [26] and the BWRC PicoNode [27], which are based on CMOS, MEMS, and RF technologies, and aims to have extremely low power and small footprint with certain sensing, computation, and communication capabilities.

Among all the above hardware platforms, the Berkeley Motes have received wide popularity in the research community of sensor networks due to their small form factor, open source software development, and commercial availability [6].

1.2.3.2 Software Platforms. A software platform can be an operating system that provides a set of services for applications, including file management, memory allocation, task scheduling, peripheral device drivers, and networking, or it can be a language platform that provides a library of components to programmers [6]. Typical software platforms for sensor networks include TinyOS [22], nesC [28], TinyGALS [29], and Moté [30]. TinyOS is one of the earliest operating systems supporting sensor network applications on resource-constrained hardware platforms, for example, the Berkeley motes. This system is event driven and uses only 178 bytes of memory, but supports communication, multitasking, and code modularity. It has no file system, supports only static memory allocation, implements a simple task scheduler, and provides minimal

device and networking abstractions. The nesC is an extension of C language to support the design of TinyOS. It provides a set of language constructs and restrictions to implement TinyOS components and applications. TinyGALS is a language for TinyOS, which provides a way of building event-triggered concurrent execution from thread-unsafe components. Unlike nesC, it addresses concurrency at the system level rather than at the component level. Moté is a virtual machine for the Berkeley motes. It defines virtual machine instructions to abstract those common operations, for example, polling sensors and accessing internal states. Therefore, software written in Moté instructions does not have to be rewritten to accommodate a new hardware platform with support for the virtual machine.

1.2.4 Wireless Sensor Network Standards

To facilitate the worldwide development and application of WSNs, there is a need for building a large low-cost market for sensor products in the field. For this purpose, it is important to specify relevant standards so that sensor products from different manufacturers may interoperate. A lot of efforts have been made and are under way in many standardization organizations in order to unify the market, leading to low-cost and interoperable devices, and avoiding the proliferation of proprietary incompatible network protocols. To a certain extent, the success of WSNs as a technology will largely rely on the success of these standardization efforts.

1.2.4.1 The IEEE 802.15.4 Standard. The IEEE 802.15.4 [31] is a standard developed by IEEE 802.15 Task Group 4, which specifies the physical and MAC layers for low-rate WPANs. As defined in its Project Authorization Request, the goal of Task Group 4 is to "provide a standard for ultralow complexity, ultralow cost, ultralow power consumption, and low-data rate wireless connectivity among inexpensive devices". The first release of the IEEE 802.15.4 standard was delivered in 2003 and is freely distributed in [32]. This release was revised in 2006, but the new release is not yet freely distributed. Its protocol stack is simple and flexible, and does not require any infrastructure. The standard has the following features [32]:

- Data rates of 250 kbps, 40 kbps, and 20 kbps.
- Two addressing modes: 16-bit short and 64-bit IEEE addressing.
- Support for critical latency devices, for example, joysticks.
- The CSMA-CA channel access.
- Automatic network establishment by the coordinator.
- Fully handshaking protocol for transfer reliability.
- Power management to ensure low-power consumption.
- Some 16 channels in the 2.4-GHz ISM band, 10 channels in the 915-MHz band, and 1 channel in the 868-MHz band.

The physical layer of the IEEE 802.15.4 standard has been specified to coexist with other IEEE standards for wireless networks, for example, IEEE 802.11 (WLAN) and IEEE 802.15.1 (Bluetooth). It features activation and deactivation of the radio transceiver and transmission of packets on the physical medium. It operates in one of the following three license-free bands:

- 868–868.6 MHz (e.g., Europe) with a data rate of 20 kbps.
- 902–928 MHz (e.g., North America) with a data rate of 40 kbps.
- 2400–2483.5 MHz (worldwide) with a data rate of 250 kbps.

The MAC layer provides data and management services to the upper layers. The data service enables transmission and reception of MAC packets over the physical layer. The management services include synchronization, timeslot management, and association and disassociation of devices to the network. Moreover, the MAC layer implements basic security mechanisms. For a comprehensive introduction of the IEEE 802.15.4 standard, the author is referred to Chapter 13.

1.2.4.2 The ZigBee Standard. The IEEE 802.15.4 standard only defines the physical and MAC layers without specifying the higher protocol layers, including the network and application layers. The ZigBee standard [33] is developed on top of the IEEE 802.15.4 standard and defines the network and application layers. The network layer provides networking functionalities for different network topologies, and the application layer provides a framework for distributed application development and communication. The two protocol stacks can be combined together to support short-range low data rate wireless communication with battery-powered wireless devices. The potential applications of these standards include sensors, interactive toys, smart badges, remote controls, and home automation.

The ZigBee protocol stack was proposed at the end of 2004 by the ZigBee Alliance [34], an association of companies working together to enable reliable, cost-effective, low-power, wirelessly networked, monitoring, and control products based on an open global standard. The first release of ZigBee was revised at the end of 2006, which introduces extensions on the standardization of application profiles and some minor improvements to the network and application layers. Both releases can be freely downloaded at Ref. [34]. For a comprehensive introduction of the IEEE 802.15.4 standard, the author is referred to Ref. [35] or Chapter 13.

1.2.4.3 The IEEE 1451 Standard. The IEEE 1451 standards are a family of Smart Transducer Interface Standards that defines a set of open, common, network-independent communication interfaces for connecting transducers (i.e., sensors or actuators) to microprocessors, instrumentation systems, and control/field networks [36]. Transducers have a wide variety of applications in industry, for example, manufacturing, industrial control, automotive, aerospace, building,

and biomedicine. Since the transducer market is very diverse, transducer manufacturers are seeking ways to build low-cost, networked, and wireless smart transducers. But one problem for transducer manufacturers is the large number of wired and wireless networks on the market today. Currently, it is too costly for transducer manufacturers to produce unique smart transducers for the large number of networks on the market. Therefore, a set of open standards that are universally accepted, for example, the suite of IEEE 1451 smart transducer interface standards, are developed to address these issues.

The key feature of these standards is the definition of transducer electronic data sheets (TEDS), which is a memory device attached to a transducer for storing transducer identification, calibration, correction data, measurement range, manufacture-related information, and so on. The objective of 1451 is to make it easier for transducer manufacturers to develop smart devices and to interface those devices to networks, systems, and instruments by incorporating existing and emerging sensor and networking technologies. In another word, it is to allow the access of transducer data through a common set of interfaces whether the transducers are connected to systems or networks via a wired or wireless medium. The family of IEEE 1451 standards is sponsored by the Sensor Technology Technical Committee of the IEEE Instrumentation and Measurement Society. The definitions of the IEEE 1451 standards [36] are briefly described below:

- IEEE P1451.0 defines a set of common commands, common operations, and TEDS for the family of IEEE 1451 smart transducer standards. Through this command set, one can access any sensors or actuators in the 1451-based wired and wireless networks.
- IEEE 1451.1 defined a common object model describing the behavior of smart transducers, a measurement model that streamlines measurement processes, and the communication models used for the standard, which includes the client-server and publish-subscribe models.
- IEEE 1451.2 defined a transducers-to-NCAP interface and TEDS for a point-to-point configuration.
- IEEE 1451.3 defined a transducer-to-NCAP interface and TEDS for multidrop transducers using a distributed communication architecture. It allowed many transducers to be arrayed as nodes, on a multidrop transducer network, sharing a common pair of wires.
- IEEE 1451.4 defined a mixed-mode interface for analog transducers with analog and digital operating modes.
- IEEE P1451.5 defines a transducer-to-NCAP interface and TEDS for wireless transducers. Protocol standards for wireless networks, for example, 802.11 (WiFi), 802.15.1 (Bluetooth), and 802.15.4 (ZigBee), are being considered as some of the physical interfaces for IEEE P1451.5.
- IEEE P1451.6 defines a transducer-to-NCAP interface and TEDS using the high-speed CANopen network interface. Both intrinsically safe and nonintrinsically safe applications are being supported.

1.3 FEATURES OF THIS BOOK

Networking is one of the most important aspects in the design of WSNs, which involves a variety of network architectural and protocol design issues. Due to the unique characteristics of sensor networks, conventional network protocols cannot be applied directly to sensor networks without modification. A new suite of network protocols must be developed to address the unique characteristics and constraints, in particular, the energy constraint, in sensor networks. This book focuses on the major networking issues in the design of WSNs, including medium access control, routing and data dissemination, node clustering, node localization, transport protocols, time synchronization, and network security. The aim of this book is to provide a comprehensive and systematic introduction of the fundamental concepts, major issues, and effective solutions in the networking aspect of WSNs. The main features of this book include the following:

- Giving an insight into wireless sensor networks from a networking perspective.
- Providing a comprehensive and systematic introduction of the fundamental concepts, major issues, and effective solutions in wireless sensor networking.
- Striking a balance between fundamental concepts and state-of-the-art technologies.
- Contributed by a group of leading researchers who are internationally recognized in the field.
- Intended for a wide range of audience, including academic researchers, graduate students, industry practitioners, and research engineers.
- Including a comprehensive up-to-date bibliography.

1.4 ORGANIZATION OF THIS BOOK

This book is organized into 14 chapters. Chapter 1 serves as an introduction to the whole book. The unique network characteristics, typical network applications, and technological background are introduced. Then the major network design objectives and challenges, and the focus and features of this book are described.

Chapter 2 presents a brief overview of network architectures and introduces a protocol stack for WSNs.

Chapter 3 is dedicated to medium access control (MAC) in WSNs. The fundamental concepts on MAC and traditional MAC protocols for wireless networks are introduced, the major challenges in MAC design for sensor networks are discussed, and an overview of MAC protocols for WSNs are presented.

Chapter 4 focuses on routing and data dissemination in WSNs. The fundamental concepts on routing and data dissemination are introduced and the major challenges in routing and data dissemination are discussed. Moreover, a taxonomy

of routing protocols for WSNS is introduced and based on this taxonomy a survey of routing and data dissemination protocols for WSNs is presented.

Chapter 5 is dedicated to broadcasting, multicasting, and geocasting in WSNs. The concepts of broadcasting, multicasting, and geocasting are introduced, the major challenges in geocasting, multicasting, and broadcasting are discussed, and an overview of typical broadcasting, multicasting, and geocasting algorithms are presented.

Chapter 6 concentrates on node clustering in WSNs. The purpose of node clustering and the fundamental concepts on node clustering are introduced, and a variety of node clustering algorithms for WSNs is presented.

Chapter 7 is dedicated to query processing and data aggregation in WSNs. The concept of query processing and the importance of data aggregation are introduced. The major challenges in query processing and data aggregation are discussed. The chapter also presents an overview of typical query processing and data aggregation techniques for WSNs.

Chapter 8 focuses on node localization in WSNs. The importance of node localization is introduced, the major challenges in node localization are discussed, and an overview of typical localization algorithms is presented.

Chapter 9 addresses time synchronization in WSNs. The importance of time synchronization is explained, the major challenges in time synchronization are introduced, and effective synchronization protocols for WSNs are presented.

Chapter 10 is dedicated to energy efficiency and power control in WSNs. The need for energy efficiency and power control is explained, the major challenges in designing efficient power conservation mechanisms are discussed, and an overview of major power conservation mechanisms for WSNs are presented.

Chapter 11 focuses on transport protocols and quality of service in WSNs. The fundamental concepts on transport protocols and QoS are introduced, the major challenges in the design of transport protocols for quality of service are discussed, and an overview of transport protocols for WSNs are presented.

Chapter 12 is dedicated to network security in WSNs. The importance of security in WSNs is described, the major challenges in designing security mechanisms are discussed, and a variety of effective security techniques for WSNs are presented.

Chapter 13 presents an overview of standardization activities and relevant standards for WSNs, focused on the IEEE 802.15.4 and ZigBee standards.

Chapter 14 presents an overview of the recent evolution of the sensor network paradigm, as well as future research directions in the networking aspect of WSNs.

REFERENCES

[1] "21 ideas for the 21st century", *Business Week*, Aug. 30 1999, pp. 78–167.

[2] C.-Y. Chong and S. P. Kumar, "Sensor networks: Evolution, opportunities, and challenges", *Proceedings of the IEEE*, vol. 91, no. 8, Aug. 2003, pp. 1247–1256.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks", *IEEE Communications Magazine*, vol. 40, no. 8, Aug. 2002, pp. 102–114.

[4] D. Estrin, D. Culler, K. Pister, and G. Sukhatme, "Connecting the physical world with pervasive networks", *IEEE Pervasive Computing*, Jan. 2002, pp. 59–69.

[5] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey", *Computer Networks*, vol. 38, no. 4, Mar. 2002, pp. 393–422.

[6] F. Zhao and L. Guibas, *Wireless Sensor Networks: An Information Processing Approach*, Morgan Kaufmann Publishers, San Francisco, CA, 2004.

[7] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless sensor networks for habitat monitoring", in *Proceedings of 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA'02)*, Atlanta, GA, Sept. 2002, pp. 88–97.

[8] R. Jafari, A. Encarnacao, A. Zahoory, F. Dabiri, H. Noshadi, and M. Sarrafzadeh, "Wireless sensor networks for health monitoring", in *Proceedings of 2nd Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'05)*, July 2005, pp. 479–481.

[9] D. Trossen and D. Pavel, "Sensor networks, wearable computing, and healthcare Applications", *IEEE Pervasive Computing*, vol. 6, no. 2, Apr.–June 2007, pp. 58–61.

[10] C. Herring and S. Kaplan, "Component-based software systems for smart environments", *IEEE Personal Communications*, vol. 7, no. 5, Oct. 2000, pp. 60–61.

[11] A. J. Goldsmith and S. B. Wicker, "Design challenges for energy-constrained ad hoc wireless networks", *IEEE Wireless Communications*, vol. 9, no. 4, Aug. 2002, pp. 8–27.

[12] F. Koushanfar, M. Potkonjak, and A. Sangiovanni-Vincentelli, "Fault-tolerance techniques for ad hoc sensor networks", *Proceedings of IEEE Sensors*, vol. 2, June 2002, pp. 1491–1496.

[13] R. F. Pierret, *Introduction to Microelectronic Fabrication*, Addison-Wesley, Menlo Park, CA, 1990.

[14] S. D. Senturia, *Microsystem Design*, Kluwer Academic Publishers, Norwell, MA, 2001.

[15] A. E. Franke, T.-J. King, and R. T. Howe, "Integrated MEMS technologies", *MRS Bull.*, vol. 26, no. 4, 2001, pp. 291–295.

[16] B. Warneke, "Miniaturizing sensor networks with MEMS", *SMART DUST: Sensor Network Applications, Architecture, and design (edited)*, CRC, Boca Raton, FL, 2006, pp. 5-1–5-9.

[17] J. M. Kahn, R. You, P. Djahani, A. G. Weisbin, Beh Kian Teik, and A. Tang, "Imaging diversity receivers for high-speed infrared wireless communication", *IEEE Communications Magazine*, vol. 36, no.12, Dec. 1998, pp. 88–94.

[18] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, "Energy-aware wireless microsensor networks", *IEEE Signal Processings Magazine*, vol. 19, no. 2, Mar. 2002, pp. 40–50.

[19] A. P. Chandrakasan and R. W. Broderson, *Low Power CMOS Digital Design*, Kluwer Academic Publishers, Norwell, MA, 1996.

[20] L. Benini and G. DeMicheli, *Dynamic Power Management: Design Techniques and CAD Tools*, Kluwer Academic Publishers, Norwell, MA, 1997.

[21] T. A. Pering, T. D. Burd, and R. W. Brodersen, "The simulation and evaluation of dynamic voltage scaling algorithms", in *Proceedings of 1998 International Symposium on Low Power Electronics and Design (ISLPED'98)*, Monterey, CA, Aug. 1998, pp. 76–81.

[22] J. Hill, R. Szewcyk, A. Woo, D. Culler, S. Hollar, and K. Pister, "System architecture directions for networked sensors", in *Proceedings of 9th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS IX)*, Cambridge, MA, Nov. 2000, pp. 93–104.

[23] A. Savvides and M. B. Srivastava, "A distributed computation platform for wireless embedded sensing", in *Proceedings of International Conference on Computer Design (ICCD'02)*, Freiburg, Germany, Sept. 2002, pp. 220–225.

[24] A. Chandrakasan, R. Min, M. Bhardwaj, S.-H. Cho, and A. Wang, "Power aware wireless microsensor systems", in *Proceedings of 32nd European Solid-State Device Research Conference (ESSDERC'02)*, Florence, Italy, Sept. 2002, pp. 47–54.

[25] J. M. Khan, R. H. Katz, and K. Pister, "Next century challenges: Mobile networking for smart dust", in *Proceedings of 5th International Conference on Mobile Computing and Networking (MobiCom'99)*, Seattle, WA, Aug. 1999, pp. 271–278.

[26] J. Rabaey, J. Ammer, J. da Silva, D. Patel, and S. Roundy, "Picoradio supports ad-hoc ultra-low power wireless networking", *IEEE Computer Magazine*, July 2002, pp. 42–48.

[27] F. Yao, A. Demers, and S. Shenker, "A scheduling model for reduced CPU energy", in *Proceedings of 36th Annual Symposium on Foundations of Computer Science (FOCS'95)*, Oct. 1995, pp. 374–382.

[28] D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, and D. Culler, "The nesC language: A holistic approach to network embedded systems", in *Proceedings of 2003 ACM SiGPLAN Conference on Programming Language Design and Implementation (PLDI'03)*, San Diego, CA, June 2003, pp. 1–11.

[29] E. Cheong, J. Liebman, J. Liu, and F. Zhao, "TinyGALS: A programming model for event-driven embedded systems", in *Proceedings of 18th Annual ACM Symposium on Applied Computing (SAC'03)*, Melbourne, FL, Mar. 2003, pp. 698–704.

[30] P. Levis and D. Culler, "Moté: A tiny virtual machine for sensor networks", in *Proceedings of 10th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS X)*, San José, CA, Oct. 2002, pp. 85–95.

[31] Institute of Electrical and Electronics Engineers, Inc., "IEEE Std. 802.15.4–2003: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (LR-WPANs)", New York, IEEE Press. Oct. 2003.

[32] Available at http://www.ieee802.org/15/pub/TG4.html

[33] ZigBee Alliance, "ZigBee Specifications", Dec. 2006.

[34] Available at http://www.ZigBee.org/en/index.asp

[35] P. Baronti, P. Pillai, V. Chook, S. Chessa, A. Gotta, and Y. F. Hu, "Wireless Sensor Networks: a Survey on the State of the Art and the 802.15.4 and ZigBee Standards", *Computer Communications*, vol. 30, no. 7, May 2007, pp. 1655–1695.

[36] Available at http://ieee1451.nist.gov/

2

NETWORK ARCHITECTURES AND PROTOCOL STACK

Jun Zheng

Southeast University, China

2.1 INTRODUCTION

Network architectures and protocols are important aspects in the design of wireless sensor networks (WSNs) [1]. Due to the severe energy constraint of sensor nodes, network architectural design has a big impact on the energy consumption and thus the operational lifetime of the whole network. On the other hand, a sensor network consists of a large number of sensor nodes that are densely deployed in a sensing region and collaborate to accomplish a sensing task. It requires a suite of network protocols to implement various network control and management functions, for example, synchronization, self-configuration, medium access control, routing, data aggregation, node localization, and network security. However, existing network protocols for traditional wireless networks, for example, cellular systems and mobile ad hoc networks (MANETs), cannot be applied directly to sensor networks because they do not consider the energy, computation, and storage constraints in sensor nodes. On the other hand, most sensor networks are application specific and have different application requirements. For these reasons, a new suite of network protocols is required, which take into account not only the resource constraints in sensor nodes, but also the

Wireless Sensor Networks: A Networking Perspective, Edited by Jun Zheng and Abbas Jamalipour

requirements of different network applications. For this purpose, it is important to define a protocol stack to facilitate the protocol design for WSNs.

This chapter introduces fundamental concepts on network architectures and protocol stack of WSNs. We will first introduce sensor node structure and typical sensor network architectures in Section 2.2, then discuss the classification of sensor networks in Section 2.3, and finally describe a protocol stack for sensor networks in Section 2.4. Section 2.5 will summarize this chapter.

2.2 NETWORK ARCHITECTURES FOR WIRELESS SENSOR NETWORKS

In this section, first we introduce the structure of a sensor node and then describe typical network architectures for WSNs.

2.2.1 Sensor Node Structure

A sensor node typically consists of four basic components: a sensing unit, a processing unit, a communication unit, and a power unit, which is shown in Fig. 2.1. The sensing unit usually consists of one or more sensors and analog-to-digital converters (ADCs). The sensors observe the physical phenomenon and generate analog signals based on the observed phenomenon. The ADCs convert the analog signals into digital signals, which are then fed to the processing unit. The processing unit usually consists of a microcontroller or microprocessor with memory (e.g., Intel's StrongARM microprocessor and Atmel's AVR microprocessor), which provides intelligent control to the sensor node. The communication unit consists of a short-range radio for performing data transmission and reception over a radio channel. The power unit consists of a battery for supplying power

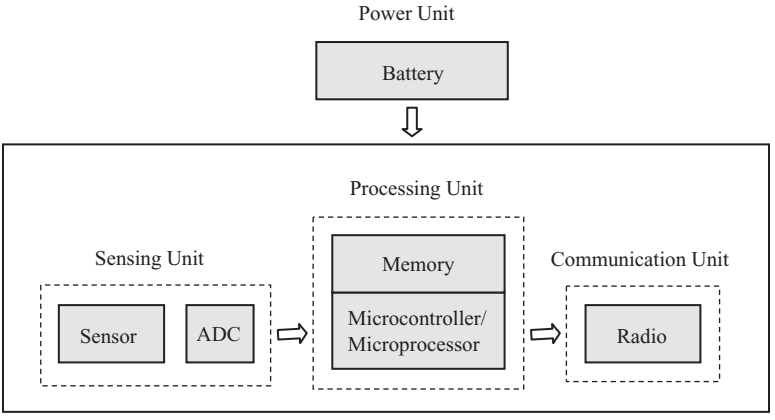


Fig. 2.1 Sensor node structure.

to drive all other components in the system. In addition, a sensor node can also be equipped with some other units, depending on specific applications. For example, a global positioning system (GPS) may be needed in some applications that require location information for network operation. A motor may be needed to move sensor nodes in some sensing tasks. All these units should be built into a small module with low power consumption and low production cost.

2.2.2 Network Architectures

A sensor network typically consists of a large number of sensor nodes densely deployed in a region of interest, and one or more data sinks or base stations that are located close to or inside the sensing region, as shown in Fig. 2.2. The sink(s) sends queries or commands to the sensor nodes in the sensing region while the sensor nodes collaborate to accomplish the sensing task and send the sensed data to the sink(s). Meanwhile, the sink(s) also serves as a gateway to outside networks, for example, the Internet. It collects data from the sensor nodes, performs simple processing on the collected data, and then sends relevant information (or the processed data) via the Internet to the users who requested it or use the information.

To send data to the sink, each sensor node can use single-hop long-distance transmission, which leads to the single-hop network architecture, as shown in Fig. 2.3. However, long-distance transmission is costly in terms of energy consumption. In sensor networks, the energy consumed for communication is much higher than that for sensing and computation. For example, the energy consumed for transferring one bit of data to a receiver at 100 m away is equal to that needed to execute 3,000 instructions [2]. The ratio of energy consumption for communicating 1 bit over the wireless medium to that for processing the same bit could be in the range of 1,000–10,000 [3,4]. Furthermore, the energy consumed for transmission dominates the total energy consumed for communication and the required transmission power grows exponentially with the increase of transmission distance. Therefore, it is desired to reduce the amount of traffic and transmission distance in order to increase energy savings and prolong network lifetime.

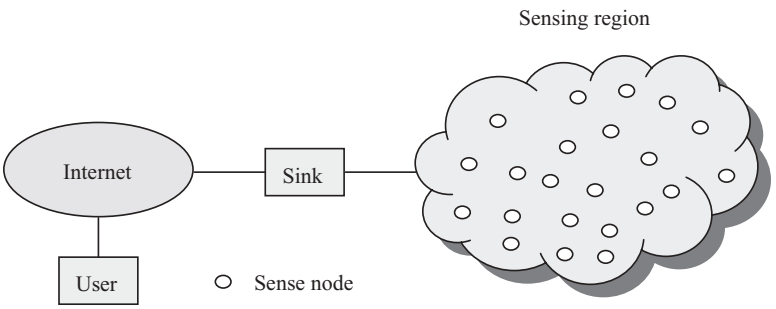


Fig. 2.2 Sensor network architecture.

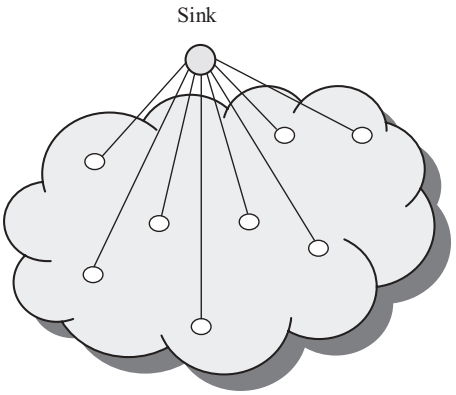


Fig. 2.3 Single-hop network architecture.

For this purpose, multihop short-distance communication is highly preferred. In most sensor networks, sensor nodes are densely deployed and neighbor nodes are close to each other, which makes it feasible to use short-distance communication. In multihop communication, a sensor node transmits its sensed data toward the sink via one or more intermediate nodes, which can reduce the energy consumption for communication. The architecture of a multihop network can be organized into two types: flat and hierarchical [5], which are described in the next two sections (Sections 2.2.2.1 and 2.2.2.2).

2.2.2.1 Flat Architecture. In a flat network, each node plays the same role in performing a sensing task and all sensor nodes are peers. Due to the large number of sensor nodes, it is not feasible to assign a global identifier to each node in a sensor network. For this reason, data gathering is usually accomplished by using data-centric routing, where the data sink transmits a query to all nodes in the sensing region via flooding and only the sensor nodes that have the data matching the query will respond to the sink. Each sensor node communicates with the sink via a multihop path and uses its peer nodes as relays. Figure 2.4 illustrates the typical architecture of a flat network.

2.2.2.2 Hierarchical Architecture. In a hierarchical network, sensor nodes are organized into clusters, where the cluster members send their data to the cluster heads while the cluster heads serve as relays for transmitting the data to the sink. A node with lower energy can be used to perform the sensing task and send the sensed data to its cluster head at short distance, while a node with higher energy can be selected as a cluster head to process the data from its cluster members and transmit the processed data to the sink. This process can not only reduce the energy consumption for communication, but also balance traffic load and improve scalability when the network size grows. Since all sensor nodes have the same transmission capability, clustering must be periodically performed in

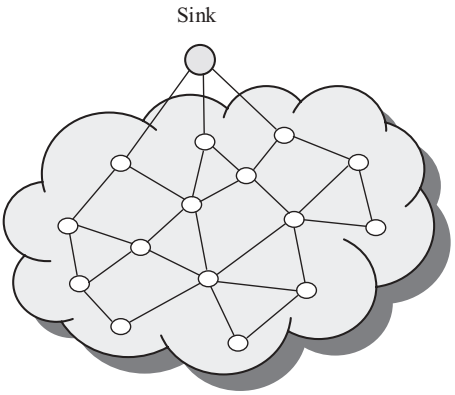


Fig. 2.4 Flat network architecture.

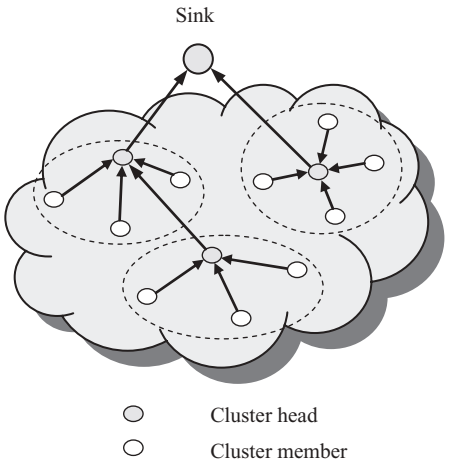


Fig. 2.5 Single-hop clustering architecture.

order to balance the traffic load among all sensor nodes. Moreover, data aggregation can be performed at cluster heads to reduce the amount of data transmitted to the sink and improve the energy efficiency of the network [6].

The major problem with clustering is how to select the cluster heads and how to organize the clusters [7]. In this context, there are many clustering strategies. According to the distance between the cluster members and their cluster heads, a sensor network can be organized into a single-hop clustering architecture or a multihop clustering architecture, as shown in Figs. 2.5 and 2.6, respectively [8]. According to the number of tiers in the clustering hierarchy, a sensor network can be organized into a single-tier clustering architecture or a multitier clustering architecture. Figure 2.7 illustrates an example of the multitier

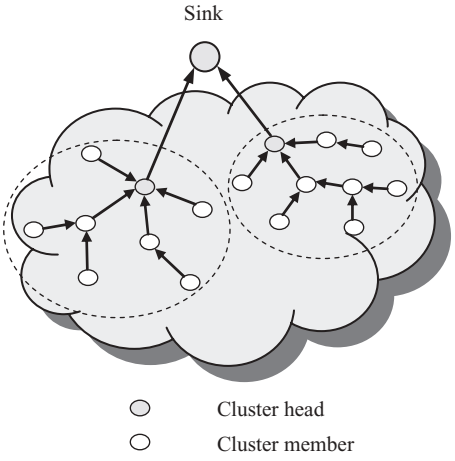


Fig. 2.6 Multihop clustering architectures.

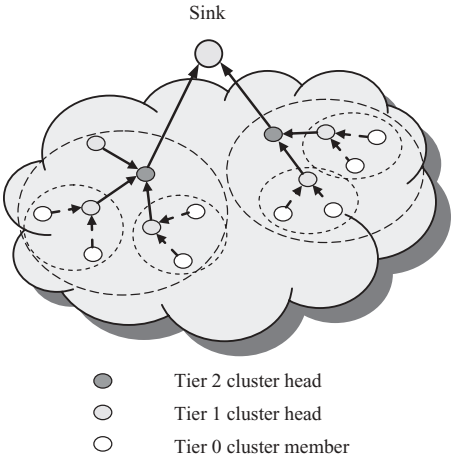


Fig. 2.7 Multitier clustering architectures.

clustering architecture [9]. To address the clustering problem, a variety of clustering algorithms have been proposed in the literature [7–13]. The reader is referred to Chapter 6 for a comprehensive introduction of these clustering algorithms.

2.3 CLASSIFICATIONS OF WIRELESS SENSOR NETWORKS

WSNs are application specific. A sensor network is usually deployed for a specific application and thus has some different characteristics. According to different criteria, WSNs can be classified into different categories.

- *Static and Mobile Network.* According to the mobility of sensor nodes, a sensor network can be static or mobile. In a static sensor network, all sensor nodes are static without movement, which is the case for many applications. However, some sensor applications require mobile nodes to accomplish a sensing task. A wireless biosensor network using autonomously controlled animals is a typical example of mobile sensor networks [14]. Compared with static sensor networks, which is simpler to control and easier to implement, the design of mobile sensor networks must consider the mobility effect, which increases the complexity of implementation.
- *Deterministic and Nondeterministic Network.* According to the deployment of sensor nodes, a sensor network can be deterministic or nondeterministic. In a deterministic sensor network, the positions of sensor nodes are preplanned and are fixed once deployed. This type of network can only be used in some limited situations, where the preplanned deployment is possible. In most situations, however, it is difficult to deploy sensor nodes in a preplanned manner because of the harsh or hostile environments. Instead, sensor nodes are randomly deployed without preplanning and engineering. Obviously, nondeterministic networks are more scalable and flexible, but require higher control complexity.
- *Static-Sink and Mobile-Sink Network.* A data sink in a sensor network can be static or mobile. In a static-sink network, the sink(s) is static with a fixed position located close to or inside a sensing region. All sensor nodes send their sensed data to the sink(s). Obviously, a static sink makes the network simpler to control, but it would cause the hotspot effect [5]. The amount of traffic that sensor nodes are required to forward increases dramatically as the distance to the data sink becomes smaller. As a result, sensor nodes closest to the data sink tend to die early, thus resulting in network partition and even disrupting normal network operation. In a mobile-sink network, the sink(s) moves around in the sensing region to collect data from sensor nodes, which can balance the traffic load of sensor nodes and alleviate the hotspot effect in the network.
- *Single-Sink and Multisink Network.* A sensor network can have a single sink or multiple sinks. In a single-sink network, there is only one sink located close to or inside the sensing region. All sensor nodes send their sensed data to this sink. In a multisink network, there may be several sinks located in different positions close to or inside the sensing region. Sensor nodes can send their data to the closest sink, which can effectively balance the traffic load of sensor nodes and alleviate the hotspot effect in the network.
- *Single-Hop and Multihop Network.* According to the number of hops between a sensor node and the data sink, a sensor network can be classified into single-hop or multihop. In a single-hop network, all sensor nodes transmit their sensed data directly to the sink, which makes network control simpler to implement. However, this requires long-range wireless communication, which is costly in terms of both energy consumption and hardware

implementation. The furthest nodes from the data sink will die much more quickly than those close to the sink. Also, the overall traffic load in the network may increase rapidly with the increase of the network size, which would cause more collisions, and thus increase energy consumption and delivery latency. In a multihop network, sensor nodes transmit their sensed data to the sink using short-range wireless communication via one or more intermediate nodes. Each intermediate node must perform routing and forward the data along a multihop path. Moreover, data aggregation can be performed at an intermediate node to eliminate data redundancy, which can reduce the total amount of traffic in the network and thus improve the energy efficiency of the network. In general, a single-hop network has simpler network architecture and thus is easier to control. It is suitable for applications in small sensing areas with sparsely deployed sensor nodes. Multihop networks have a wider range of applications at the cost of higher control complexity.

- *Self-Reconfigurable and Non-Self-Configurable Network.* According to the configurability of sensor nodes, a sensor network can be self-configurable or non-self-configurable. In a non-self-configurable network, sensor nodes have no ability to organize themselves into a network. Instead, they have to rely on a central controller to control each sensor node and collect information from them. Therefore, this type of networks is only suitable for small-scale networks. In most sensor networks, however, sensor nodes are able to autonomously organize and maintain their connectivity by themselves and collaboratively accomplish a sensing task. A network with such self-configurability is suitable for large-scale networks to perform complicated sensing tasks.
- *Homogeneous and Heterogeneous Network.* According to whether sensor nodes have the same capabilities, a sensor network can be homogeneous or heterogeneous [15]. In a homogeneous network, all sensor nodes have the same capabilities in terms of energy, computation, and storage. In contrast, a heterogeneous network has some sophisticated sensor nodes that are equipped with more processing and communicating capabilities than normal sensor nodes. In this case, the network can assign more processing and communication tasks to those sophisticated nodes in order to improve its energy efficiency and thus prolong the lifetime.

2.4 PROTOCOL STACK FOR WIRELESS SENSOR NETWORKS

The protocol stack for WSNs consists of five protocol layers: the physical layer, data link layer, network layer, transport layer, and application layer, as shown in Fig. 2.8. The application layer contains a variety of application-layer protocols to generate various sensor network applications. The transport layer is responsible for reliable data delivery required by the application layer. The network layer is

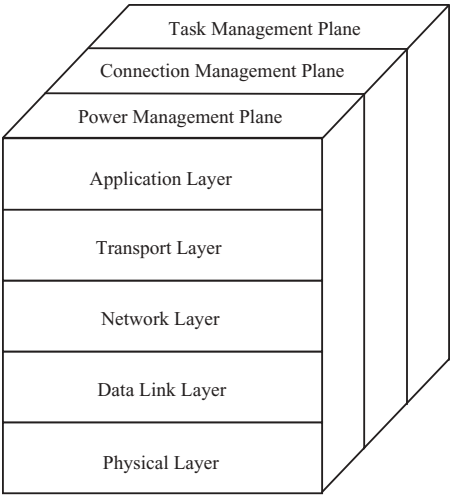


Fig. 2.8 Protocol stack for sensor networks.

responsible for routing the data from the transport layer. The data link layer is primarily responsible for data stream multiplexing, data frame transmission and reception, medium access, and error control. The physical layer is responsible for signal transmission and reception over a physical communication medium, including frequency generation, signal modulation, transmission and reception, data encryption, and so on.

On the other hand, the protocol stack can be divided into a group of management planes across each layer [16], including power, connection, and task management planes. The power management plane is responsible for managing the power level of a sensor node for sensing, processing, and transmission and reception, which can be implemented by employing efficient power management mechanisms at different protocol layers. For example, at the MAC layer, a sensor node can turn off its transceiver when there is no data to transmit and receive. At the network layer, a sensor node may select a neighbor node with the most residual energy as its next hop to the sink. The connection management plane is responsible for the configuration and reconfiguration of sensor nodes to establish and maintain the connectivity of a network in the case of node deployment and topology change due to node addition, node failure, node movement, and so on. The task management plane is responsible for task distribution among sensor nodes in a sensing region in order to improve energy efficiency and prolong network lifetime. Since sensor nodes are usually densely deployed in a sensing region and are redundant for performing a sensing task, not all sensor nodes in the sensing region are required to perform the same sensing task. Therefore, a task management mechanism can be used to perform task distribution among multiple sensors. The reader is referred to Ref. [16] for a more comprehensive introduction of sensor network management.

2.4.1 Application Layer

The application layer includes a variety of application-layer protocols that perform various sensor network applications, such as query dissemination, node localization, time synchronization, and network security. For example, the sensor management protocol (SMP) [1] is an application-layer management protocol that provides software operations to perform a variety of tasks, for example, exchanging location-related data, synchronizing sensor nodes, moving sensor nodes, scheduling sensor nodes, and querying the status of sensor nodes. The sensor query and data dissemination protocol (SQDDP) provides user applications with interfaces to issue queries, respond to queries, and collect responses [1]. The sensor query and tasking language (SQTL) provides a sensor programming language used to implement middleware in WSNs [17]. Although many sensor network applications have been proposed, their corresponding application-layer protocols still need to be developed.

2.4.2 Transport Layer

In general, the transport layer is responsible for reliable end-to-end data delivery between sensor nodes and the sink(s). Due to the energy, computation, and storage constraints of sensor nodes, traditional transport protocols cannot be applied directly to sensor networks without modification. For example, the conventional end-to-end retransmission-based error control and the window-based congestion control mechanisms used in the transport control protocol (TCP) cannot be used for sensor networks directly because they are not efficient in resource utilization.

On the other hand, sensor networks are application specific. A sensor network is usually deployed for a specific sensing application, for example, habitat monitoring, inventory control, and battlefield surveillance. Different applications may have different reliability requirements, which have a big impact on the design of transport-layer protocols. In addition, data delivery in sensor networks primarily occurs in two directions: upstream and downstream. In the upstream, the sensor nodes transmit their sensed data to the sink(s), while in the downstream the data originated from the sink(s), for example, queries, commands, and programming binaries, are sent from the sink(s) to the source sensor nodes. The data flows in the two directions may have different reliability requirements. For example, the data flows in the upstream direction are loss tolerant because the sensed data are usually correlated or redundant to a certain extent. In the downstream, however, the data flows are queries, commands, and programming binaries sent to the sensor nodes, which usually require 100% reliable delivery. Therefore, the unique characteristics of sensor networks and the specific requirements of different applications present many new challenges in the design of transport layer protocols for WSNs. Chapter 11 gives a more comprehensive introduction and discussion of the transport layer design, as well as a variety of transport layer protocols for WSNs.

2.4.3 Network Layer

The network layer is responsible for routing the data sensed by source sensor nodes to the data sink(s). In a sensor network, sensor nodes are deployed in a sensing region to observe a phenomenon of interest. The observed phenomenon or data need to be transmitted to the data sink. In general, a source node can transmit the sensed data to the sink either directly via single-hop long-range wireless communication or via multihop short-range wireless communication. However, long-range wireless communication is costly in terms of both energy consumption and implementation complexity for sensor nodes. In contrast, multihop short-range communication can not only significantly reduce the energy consumption of sensor nodes, but also effectively reduce the signal propagation and channel fading effects inherent in long-range wireless communication, and is therefore preferred. Since sensor nodes are densely deployed and neighbor nodes are close to each other, it is possible to use multihop short-range communication in sensor networks. In this case, to send the sensed data to the sink, a source node must employ a routing protocol to select an energy-efficient multihop path from the node itself to the sink. However, routing protocols for traditional wireless networks are not suitable for sensor networks because they do not consider energy efficiency as the primary concern. Also, data from the sensing region toward the sink exhibit a unique many-to-one traffic pattern in sensor networks [5]. The combination of multihop (i.e., hop-by-hop) and many-to-one communications results in a significant increase in transit traffic intensity and thus packet congestion, collision, loss, delay, and energy consumption as data move closer toward the sink. The sensor nodes closer to the sink, typically within a small number of hops, will loose a larger number of packets and consume much more energy than the nodes further away from the sink, thus largely reducing the operational lifetime of the entire network. Therefore, it is important to take into account the energy constraint of sensor nodes as well as the unique traffic pattern in the design of the network layer and routing protocols. In this context, a large amount of research has been conducted and a variety of routing protocols have been proposed to address various application scenarios of sensor networks. Chapter 4 will give a more comprehensive introduction and discussion of the routing issues and a variety of routing protocols for WSNs.

2.4.4 Data Link Layer

The data link layer is responsible for data stream multiplexing, data frame creation and detection, medium access, and error control in order to provide reliable point-to-point and point-to-multipoint transmissions. One of the most important functions of the data link layer is medium access control (MAC). The primary objective of MAC is to fairly and efficiently share the shared communication resources or medium among multiple sensor nodes in order to achieve good network performance in terms of energy consumption, network throughput, and delivery latency. However, MAC protocols for traditional wireless

networks cannot be applied directly to sensor networks without modification because they do not take into account the unique characteristics of sensor networks, in particular, the energy constraint. For example, the primary concern in a cellular system is to provide quality of service (QoS) to users. Energy efficiency is only of secondary importance because there is no power limit with the base stations and the mobile users can replenish the batteries in their handsets. In MANETs, mobile nodes are equipped with portable devices powered by battery, which is also replaceable. In contrast, the primary concern in sensor networks is energy conservation for prolonging network lifetime, which makes traditional MAC protocols unsuitable for sensor networks. Therefore, a large amount of research work has been conducted on MAC and a variety of MAC protocols have been proposed to address different application scenarios, which will be introduced in Chapter 3.

Another important function of the data link layer is error control in data transmission. In many applications, a sensor network is deployed in a harsh environment where wireless communication is error prone. In this case, error control becomes indispensable and critical for achieving link reliability or reliable data transmission. In general, there are two main error control mechanisms: Forward Error Correction (FEC) and Automatic Repeat reQuest (ARQ). ARQ achieves reliable data transmission by retransmitting lost data packets or frames. Obviously, this incurs significant retransmission overheads and additional energy consumption, and therefore is not suitable for sensor networks. FEC achieves link reliability by using error control codes in data transmission, which introduces additional encoding and decoding complexities that require additional processing resources in sensor nodes. However, FEC can significantly reduce the channel bit error rate (BER) for any given transmission power. Given the energy constraint of sensor nodes, FEC is still the most efficient solution to error control in sensor networks. To design a FEC mechanism, the choice of the error control code is very important because a well-chosen error control code can obtain a good coding gain and several orders of magnitude reduction in BER. Meanwhile, the additional processing power consumed for encoding and decoding must also be considered. Therefore, a trade-off should be optimized between the additional processing power and the corresponding coding gain in order to have a powerful, energy-efficient, and low-complexity FEC mechanism.

2.4.5 Physical Layer

The physical layer is responsible for converting bit streams from the data link layer to signals that are suitable for transmission over the communication medium. For this purpose, it must deal with various related issues, for example, transmission medium and frequency selection, carrier frequency generation, signal modulation and detection, and data encryption. In addition, it must also deal with the design of the underlying hardware, and various electrical and mechanical interfaces.

Medium and frequency selection is an important problem for communication between sensor nodes. One option is to use radio and the industrial, scientific and medical (ISM) bands that are licence-free in most countries. The main advantages of using the ISM bands include free use, large spectrum, and global availability [18]. However, the ISM bands already have been used for some communication systems, such as cordless phone systems and wireless local area networks (WLANs). On the other hand, sensor networks require a tiny, low cost, and ultralow power transceiver. For these reasons, the 433-MHz ISM band and the 917-MHz ISM band have been recommended for use in Europe and North America, respectively. Many projects have used radio frequency (RF) circuits in the hardware design for sensor nodes, such as the μAMPS project [19], where the sensor node uses a 2.4-GHz transceiver, and that in [20], where the sensor node uses a single-channel RF transceiver operating at 916 MHz. In addition to radio, optical or, infrared medium can be a possible option. For example, the Smart Dust project [21] used the optical medium for transmission. However, both require that a sender and its receiver be within the sight distance to communicate with each other, which limits their use to a certain extent [22].

2.5 SUMMARY

Network architectural design has a big impact on the energy consumption and thus the operational lifetime of a WSN. Because of the energy constraint in sensor nodes and the unique many-to-one traffic pattern, multihop short-distance communication is preferred in sensor networks. In multihope networks, a hierarchical network architecture based on clustering can not only reduce the energy consumption for communication, but also balance traffic load and improve scalability when the network size grows. On the other hand, sensor networks require a new suite of network protocols to perform various network control and management functions, which must take into account not only the resource constraints, but also the application-specific nature of sensor networks. This chapter gave an introduction of fundamental concepts on network architectures and protocol stack for sensor networks, which can help understand the subsequent chapters of this book.

REFERENCES

[1] Ian F. Akyildiz et al., "A survey on sensor networks", *IEEE Communications Magazine*, vol. 40, no. 8, Aug. 2002, pp. 102–114.

[2] G. Pottie and W. Kaiser, "Wireless integrated sensor networks (WINS)", *Communications of the ACM*, vol. 43, no. 5, May 2000, pp. 51–58.

[3] J. Hill, R. Szewcyk, A. Woo, D. Culler, S. Hollar, and K. Pister, "System architecture directions for networked sensors", in *Proceedings of 9th International Conference on*

Architectural Support for Programming Languages and Operating Systems (ASPLoS IX), Cambridge, MA, Nov. 2000, pp. 93–104.

[4] W. M. Merrill, K. Sohrabi, L. Girod, J. Elson, F. Newberg, and W. Kaiser, "Open standard development platforms for distributed sensor networks", in *Proceedings of SPIE—Unattended Ground Sensor Technologies and Applications IV (AeroSense 2002)*, vol. 4743, Orlando, FL, Apr. 2002, pp. 327–337.

[5] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: A survey", *IEEE Wireless Communications*, vol. 11, no. 6, Dec. 2004, pp. 6–28.

[6] R. Rajagopalan and P. Varshney, "Data-aggregation techniques in sensor networks: A survey", *IEEE Communications and Surveys and Tutorials*, vol. 8, no. 4, 4th Quarter 2006, pp. 48–63.

[7] A. A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor networks", *Computer Communications*, vol. 30, nos. 14–15, Oct. 2007, pp. 2826–2841.

[8] G. Gupta and M. Younis, "Load-balanced clustering of wireless sensor networks", in *Proceedings of 2003 IEEE International Conference on Communications (ICC'03)*, Anchorage, AK, May 2003, pp. 1848–1852.

[9] S. Bandyopadhyay and E. J. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks", in *Proceedings of IEEE INFOCOM'03*, vol. 3, San Francisco, Mar.–Apr. 2003, pp. 1713–1723.

[10] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks", *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, Oct. 2002, pp. 660–670.

[11] O. Younis and S. Fahmy, "Heed: A hybrid, energy-efficient, distributed clustering approach for ad-hoc sensor networks", *IEEE Transactions on Mobile Computing*, vol. 3, no. 4, Oct.–Dec. 2004, pp. 366–379.

[12] P. Wang, C. Li, and J. Zheng, "Distributed minimum-cost clustering protocol for underwater sensor networks (UWSNs)", in *Proceedings of 2007 IEEE International Conference on Communications (ICC'07*), Glasgow, UK, June 2007, pp. 3510–3515.

[13] S. Banerjee and S. Khuller, "A clustering scheme for hierarchical control in multi-hop wireless networks", in *Proceedings of IEEE INFOCOM'01*, Anchorage, AK, Apr. 2001, pp. 1028–1037.

[14] Y. Li, S. S. Panwar, and S. Mao, "A wireless biosensor network using autonomously controlled animals", *IEEE Network*, vol. 20, no. 3, May/June 2006, pp. 6–11.

[15] H. Nakayama, N. Ansari, A. Jamalipour, Y. Nemoto, and N. Kato, "Fault-resilient sensing in wireless sensor networks", *Computer Communications*, vol. 30, no. 11, Sept. 2007, pp. 2375–2384.

[16] L. B. Ruiz, J. M. Nogueira, and A. A. F. Loureira, "Sensor network management", *SMART DUST: Sensor Network Applications, Architecture, and Design (edited)*, CRC Press, Boca Raton, FL, 2006.

[17] C. Shen, C. Srisathapornphat, and C. Jaikaeo, "Sensor information networking architecture and applications", *IEEE Personal Communications*, vol. 8, no. 4, Aug. 2001, pp. 52–59.

[18] W. Su, Ö. B. Akan, and E. Cayirci, "Communication protocols for sensor networks", *Wireless Sensor Networks (edited)*, Kluwer Academic Publishers, Norwell, MA, 2004.

[19] E. Shih, S. Cho, N. Ickes, R. Min, A. Sinha, A. Wang, and A. Chandrakasan, "Physical layer driven protocol and algorithm design for energy-efficient wireless sensor networks", in *Proceedings of ACM Mobicom'01*, Rome, Italy, July 2001, pp. 272–286.

[20] A. Woo and D. Culler, "A transmission control scheme for media access in sensor networks", in *Proceedings of ACM Mobicom'01*, Rome, Italy, July 2001, pp. 221–235.

[21] J. M. Kahn, R. H. Katz, and K. S. J. Pister, "Next century challenges: Mobile networking for smart dust", in *Proceedings of ACM Mobicom'99*, Washington, DC, 1999, pp. 271–278.

[22] F. Zhao and L. Guibas, *Wireless Sensor Networks: An Information Processing Approach*, Morgan Kaufmann Publishers, San Francisco, CA, 2004.

3

MEDIUM ACCESS CONTROL

Jun Zheng

Southeast University, China

3.1 INTRODUCTION

Medium access control (MAC) is one of the critical issues in the design of wireless sensor networks (WSNs) [1]. As in most wireless networks, collision, which is caused by two nodes sending data at the same time over the same transmission medium, is a great concern in WSNs. To address this problem, a sensor network must employ a MAC protocol to arbitrate access to the shared medium in order to avoid data collision from different nodes and at the same time to fairly and efficiently share the bandwidth resources among multiple sensor nodes. Therefore, a MAC protocol plays an important role in enabling normal network operation and achieving good network performance.

Medium access control has been extensively studied for traditional wireless networks. A variety of MAC protocols have been proposed to address different network scenarios. From different perspectives, MAC protocols can be classified into different categories, for example, centralized and distributed, single-channel based and multiple-channel based, contention based and contention free, and so on. Time division multiple access (TDMA), frequency division multiple access (FDMA), code division multiple access (CDMA), and carrier sense multiple access (CSMA) are typical MAC protocols that have been widely used in

Wireless Sensor Networks: A Networking Perspective, Edited by Jun Zheng and Abbas Jamalipour

traditional wireless networks. However, these protocols do not take into account the unique characteristics of sensor networks, for example, denser levels of node deployment, higher unreliability of sensor nodes, and severe power, computation, and memory constraints. For this reason, traditional MAC protocols cannot be applied directly to sensor networks without modification. To design an efficient MAC protocol for sensor networks, the unique characteristics of sensor networks, in particular, energy efficiency and network scalability must be taken into account. Moreover, delivery latency, network throughput, bandwidth utilization, and fairness, which are the primary concerns in traditional wireless networks, should also be considered, but are of secondary importance in sensor networks [2].

The purpose of this chapter is to help understand the MAC problem in WSNs, discuss its unique characteristics, and present a survey of MAC protocols for WSNs. Section 3.2 gives a brief introduction of fundamental MAC protocols used in traditional wireless networks. Section 3.3 discusses major MAC design issues for WSNs. Section 3.4 presents a survey of MAC protocols for WSNs. Section 3.5 summarizes the chapter with a brief discussion of future research directions.

3.2 FUNDAMENTAL MAC PROTOCOLS

Medium access control is critical for enabling successful network operation in all shared-medium networks. The primary task of a MAC protocol is to arbitrate access to a shared medium or channel in order to avoid collision and at the same time to fairly and efficiently share the bandwidth resources among multiple nodes. According to the underlying control mechanism for collision avoidance, MAC protocols can be typically classified into two broad categories: contention-based and contention free. This section gives a brief introduction of fundamental MAC protocols used in traditional wireless networks.

3.2.1 Contention-Based MAC Protocols

In contention-based MAC, all nodes share a common medium and contend for the medium for transmission. Thus, collision may occur during the contention process. To avoid collision, a MAC protocol can be used to arbitrate access to the shared channel through some probabilistic coordination. Both ALOHA (Additive Link On-Line Hawaii System) and CSMA are the most typical examples of contention-based MAC protocols [3]. In pure ALOHA, a node simply transmits whenever it has a packet to send. In the event of a collision, the collided packet is discarded. The sender just waits a random period of time and then transmits the packet again. In slotted ALOHA, time is divided into discrete timeslots. Each node is allocated a timeslot. A node is not allowed to transmit until the beginning of the next timeslot. Pure ALOHA is easy to implement. However, its problem is that the channel efficiency is only ~10% [3]. Compared with pure ALOHA,

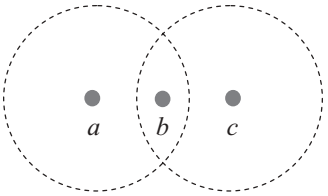


Fig. 3.1 Illustration of the hidden-node problem.

slotted ALOHA can double the channel efficiency. However, it requires global time synchronization, which complicates the system implementation.

CSMA differs from ALOHA in that it uses carrier sense; that is, it allows a node to listen to the shared medium before transmission, rather than simply transmits immediately or at the beginning of the next timeslot. CSMA has several different versions, including non-persistent, 1-persistent, and n-persistent. In non-persistent CSMA, if a node detects a busy medium, it will wait a random period of time and start to listen again. In 1-persistent CSMA, the node will continue to listen until the medium becomes idle. In n-persistent, if a node detects an idle medium, it will transmit with probability p, and back off and restart carrier sense with probability $(1\text{-}p)$.

However, CSMA cannot handle the hidden-terminal problem in multihop wireless networks [4]. Figure 3.1 illustrates the hidden-terminal problem in a 2-hop network with three nodes. Suppose that each node can only receive the signal from its immediate node. If node *a* is transmitting data to node *b*, node *c* will not be able to detect this transmission. As a result, if node *c* is also transmitting data to node *b*, the data sent by node *a* and node *c* will be collided at node *b*. To address this problem, CSMA/CA was developed and is adopted in the IEEE 802.11 wireless LAN standard [5], where CA stands for collision avoidance. In CSMA/CA, a handshake mechanism is introduced between a sender and a receiver. Before the sender transmits its data, it must establish a handshake with the receiver. The sender starts the handshake by sending a request-to-send (RTS) packet to the receiver. The receiver then acknowledges with a clear-to-send (CTS) packet. The sender starts transmitting data after it receives the CTS packet from the receiver. Through such a handshake process, the neighbors of both the sender and the receiver can know the transmission that is going on and thus back off without transmitting its own data. In the example of Fig. 3.1, node *c* cannot receive the RTS from node *a*. However, it can receive the CTS from node *b*. Therefore, if a node receives a RTS or CTS to other nodes, it should back off and does not send its own packet. In this case, collisions will mainly happen to RTS packets and can thus be reduced significantly.

To improve the performance of CSMA/CA, a MAC protocol called multiple access with collision avoidance (MACA) was developed for wireless local area networks (LANs) [6], which introduces an additional field in both RTS and CTS packets to indicate the amount of data to be transmitted so that other nodes can

know how long they should back off. To further improve the performance of MACA, another protocol called MACAW was developed in Ref. [7], which makes several enhancements to MACA. For example, after each data packet, an acknowledgment (ACK) packet is used to enable fast link-layer recovery in the event of unsuccessful transmissions. The IEEE 802.11 distributed coordination function (DCF) was mainly based on MACAW and adopted all the features of CSMA/CA, MACA, and MACAW. For more details on IEEE 802.11, the reader is referred to Ref. [5].

3.2.2 Contention-Free MAC Protocols

In contention-free MAC, a shared medium is divided into a number of subchannels in terms of time, frequency, or orthogonal pseudo-noise codes. These subchannels are allocated to individual nodes with each node occupying one subchannel. This allows different nodes to access the shared medium without interfering with each other and thus effectively avoids collision from different nodes.

The most typical examples of contention-free MAC protocols are TDMA, FDMA, and CDMA [3]. TDMA divides the shared channel into a fixed number of timeslots and configures these timeslots into a frame that repeats periodically. Each node is allocated a timeslot and is allowed to transmit only in the allocated timeslot in each frame. TDMA has been widely used in wireless cellular systems. In a typical cellular system, the base station in each cell allocates timeslots and provides information for time synchronization to all mobile nodes. The mobile nodes communicate only with the base station without direct peer-to-peer communication between each other. The major advantage of TDMA is its energy efficiency because those nodes that do not transmit can be turned off. However, TDMA has some limitations as compared with other MAC protocols. For example, TDMA usually requires nodes to form clusters like the cells in a cellular communication system. It has limited scalability and adaptability to network changes. It requires strict time synchronization for timeslots.

FDMA divides the shared channel into a number of non-overlapping frequency subbands and allocates these subbands to individual nodes. Each node can transmit at any time, but only at a different frequency to avoid interfering with the others. The major advantage of FDMA is its simplicity in implementation. However, it also has some drawbacks. For example, a guard band is needed between two adjacent subchannels. The reason is that it is not possible for a transmitter to output all its energy only in the main band and nothing in the side bands. The amount of bandwidth wasted in the guard bands can be a substantial fraction of the total bandwidth. The transmitters must be carefully power controlled. If a transmitter outputs too much power in the main band, it will also output too much power in the side bands, causing interference with adjacent channels.

CDMA divides the shared channel by using orthogonal pseudo-noise codes, rather than timeslots in TDMA and frequency bands in FDMA. All nodes can transmit in the same channel simultaneously, but with different pseudo-noise

codes. The major advantage of CDMA is that it does not require strict time synchronization and avoids the channel allocation problem in FDMA. However, it also has some disadvantages. For example, it introduces the energy consumption for coding and decoding. The capacity of a CDMA system in the presence of noise is usually lower than that of a TDMA system.

3.3 MAC DESIGN FOR WIRELESS SENSOR NETWORKS

This section introduces the characteristics of WSNs and discusses major MAC design issues in such networks.

3.3.1 Network Characteristics

To better understand WSNs, let us first take a brief look at some conventional wireless networks, for example, wireless cellular networks, mobile ad hoc networks (MANETs), and wireless LANs.

A cellular system is a wireless network consisting of both stationary and mobile nodes. The stationary nodes, or base stations, are connected by wired links, forming a fixed infrastructure. The number of mobile nodes is much larger than that of stationary nodes. Each base station usually covers a large region with little overlap and serves tens to hundreds of mobile nodes in the region. Each mobile node is only a single-hop away from its closest base station. The primary goal of a cellular system is to provide quality of service and bandwidth efficiency. The base stations have sufficient power supply and the mobile users can conveniently replace the batteries in their handsets. Accordingly, energy conservation is only of secondary importance.

A MANET is a peer-to-peer network that usually consists of tens to hundreds of mobile nodes and covers a range of up to hundreds of meters. All nodes are mobile and there is no fixed infrastructure. Hence, the network must organize the nodes to form a communication infrastructure, perform routing to enable effective communication, and maintain the organization and routing under mobile conditions. The primary goal of a MANET is to provide high quality of service in the face of high node mobility. Although each node is a portable battery-powered device, it is always attended by a person, who can replace the battery whenever needed. Hence, energy consumption is also of secondary importance in this system.

Bluetooth [8] is a short-range wireless LAN that was developed to replace the cable between electronic consumer devices with RF links. Bluetooth technology is a star network where a master node is able to have up to seven slave nodes connected to it to form a piconet. Each piconet uses a TDMA schedule and frequency hopping pattern. All slave nodes are synchronized to the master node. Multiple piconets can be interconnected to form a multihop topology. The transmission power is typically ~1 mW and the transmission range is on the order of 10 m. In Bluetooth, the primary goal is also to provide high quality of service for users.

In contrast to all the above conventional networks, WSNs have the following unique characteristics:

- A sensor network typically consists of a larger number of sensor nodes densely deployed in a geographical field. The number of sensor nodes can be several orders of magnitude larger than that of conventional wireless networks.
- Sensor nodes are usually powered by battery and thus are limited in power capacity. It is often difficult or impossible to change or recharge batteries for these nodes. The lifetime of a sensor network largely depends on the lifetime of its sensor nodes.
- Sensor nodes are often deployed in an ad hoc fashion without careful planning and engineering. Hence, they must be able to organize themselves into a communication network.
- The topology of a sensor network changes more frequently due to both node failure and mobility. Sensor nodes are prone to failures. Most sensor nodes are stationery after deployment. But in some applications, some sensor nodes can also be mobile.
- Sensor nodes have very limited computational capacity and memory.

Due to these unique characteristics, in particular, the limited energy resources, traditional MAC protocols are not suitable for being used in WSNs without modification.

3.3.2 Objectives of MAC Design

The basic function of a MAC protocol is to arbitrate access to a shared medium in order to avoid collisions from different nodes. In addition to this basic function, a MAC protocol must also take into account other factors in its design in order to improve network performance and provide good network services for different applications. In WSNs, these mainly include energy efficiency, scalability, adaptability, channel utilization, latency, throughput, and fairness [9].

- *Energy Efficiency.* Energy efficiency is one of the most important factors that must be considered in MAC design for sensor networks. It refers to the energy consumed per unit of successful communication. Since sensor nodes are usually battery powered and it is often very difficult or impossible to change or recharge batteries for sensor nodes, a MAC protocol must be energy efficient in order to maximize not only the lifetime of individual sensor nodes, but also the lifetime of the entire network.
- *Scalability.* Scalability refers to the ability to accommodate the change in network size. In sensor networks, the number of sensor nodes deployed may be on the order of tens, hundreds, or thousands. A MAC protocol must be scalable to such changes in network size.

- *Adaptability.* Adaptability refers to the ability to accommodate the changes in node density and network topology. In sensor networks, node density can be very high. A node may fail, join, or move, which would result in changes in node density and network topology. A MAC protocol must be adaptive to such changes efficiently.
- *Channel Utilization.* Channel utilization refers to the bandwidth utilization for effective communication. Due to limited bandwidth, a MAC protocol should make use of the bandwidth as efficiently as possible.
- *Latency.* Latency refers to the delay from the time a sender has a packet to send until the time the packet is successfully received by the receiver. In sensor networks, the importance of latency depends on different applications. While it is true that latency is not a critical factor for some applications (e.g., data collection for scientific exploration), many applications may have stringent latency requirements (e.g., real-time monitoring of bush fires).
- *Throughput.* Throughput refers to the amount of data successfully transferred from a sender to a receiver in a given time, usually measured in bits or bytes per second. It is affected by many factors, for example, the efficiency of collision avoidance, control overhead, channel utilization, and latency. Like latency, the importance of throughput depends on different applications.
- *Fairness.* Fairness refers to the ability of different sensor nodes to equally share a common transmission channel. In some traditional networks, it is important to achieve fairness for each user in order to ensure the quality of service for their applications. In sensor networks, however, all nodes cooperate to accomplish a single common task. What is important is not to achieve per-node fairness, but to ensure the quality of service for the whole task.

Among all these factors, energy efficiency, scalability, and adaptability are the most important for the MAC design of sensor networks. In particular, energy consumption is the primary factor affecting the operational lifetime of individual nodes and the entire network. The overall performance of a sensor network highly depends on the energy efficiency of the network. Therefore, energy efficiency is of primary importance in sensor networks. For this purpose, it is even worth trading some network performance for energy efficiency.

3.3.3 Energy Efficiency in MAC Design

As mentioned in Section 3.3.2, energy efficiency is of primary importance in WSNs. In general, energy consumption occurs in three aspects: sensing, data processing, and data communication, where data communication is a major source of energy consumption. According to Ref. [10], it consumes 3 J of energy to transmit 1-Kb data over a distance of 100 m. In contrast, a general-purpose processor with a processing capability of 100 million instructions per second can

process 300 million instructions with the same amount of energy. For this reason, it is desired to reduce data communication as much as possible in a sensor network. Thus, sensor nodes can use their processing capability to locally perform simple data processing, instead of sending all raw data to the sink(s) for processing, and then transmit partially processed data to the sink(s) for further processing. On the other hand, an efficient MAC protocol can improve energy efficiency in data communication and prolong the lifetime of a sensor network. To design an energy-efficient MAC protocol, it is important to identify the major sources of energy waste in sensor networks from the MAC perspective.

According to Ref. [9], energy waste comes from four major sources: collision, overhearing, control overhead, and idle listening.

- *Collision.* Collision occurs when two sensor nodes transmit their packets at the same time. As a result, the packets are corrupted and thus have to be discarded. Retransmissions of the packets increase both energy consumption and delivery latency.
- *Overhearing.* Overhearing occurs when a sensor node receives packets that are destined for other nodes. Overhearing such packets results in unnecessary waste of energy and such waste can be very large when traffic load is heavy and node density is high.
- *Idle Listening.* Idle listening occurs when a sensor node is listening to the radio channel to receive possible data packets while there are actually no data packets sent in the network. In this case, the node will stay in an idle state for a long time, which results in a large amount of energy waste. However, in many MAC protocols, for example, IEEE 802.11 ad hoc mode or CSMA, a node has to listen to the channel to receive possible data packets. There are reports that idle listening consumes 50–100% of the energy required for receiving data traffic [9]. For example, Stemm and Katz [11] reported that the idle: receive: send ratios are 1:1.05:1.4, while in the Digitan 2-Mbps wireless LAN module (IEEE 802.11/2 Mbps) specification the ratios are 1:2:2.5 [12].
- *Control Overhead.* A MAC protocol requires sending, receiving, and listening to a certain necessary control packets, which also consumes energy not for data communication.

3.4 MAC PROTOCOLS FOR WIRELESS SENSOR NETWORKS

In this section, we introduce various MAC protocols for WSNs, including contention-based protocols, contention-free protocols, and hybrid protocols.

3.4.1 Contention-Based Protocols

This section introduces several contention-based MAC protocols that have been proposed for WSNs.

3.4.1.1 S-MAC. The sensor-MAC (S-MAC) protocol proposed by Ye et al. [13,14] is an energy-efficient MAC protocol specifically designed for WSNs. S-MAC considers a sensor network scenario in which most communication occurs between nodes as peers, rather than to a single base station, and its applications have long idle periods and can tolerate latency on the order of network messaging time. The primary goal of the S-MAC design is to improve energy efficiency while maintaining good scalability and collision avoidance. To achieve this goal, S-MAC tries to reduce energy consumption from all the major sources that cause inefficient use of energy. In exchange, it allows some performance degradation in both per-hop fairness and latency. This is implemented by integrating several effective control mechanisms into a contention-based MAC protocol built on top of the IEEE 802.11 standard. These mechanisms include periodic listen and sleep, collision avoidance, coordinated synchronization, and message passing.

To reduce idle listening, S-MAC introduces a periodic listen and sleep mechanism to establish a low-duty-cycle operation on each node. With this mechanism, each node is periodically put into a sleep state for some time, and then wakes up and listens to see if it needs to communicate with any other node. In the sleep state, the radio is completely turned off and a timer is set to awake the node at a later time. A complete cycle of listen and sleep periods is called a frame. Each frame begins with a listen period, during which a node can communicate with the other nodes, followed by a sleep period, during which a node sleeps if it has no data to send or receive, or remains awake if it has data to send or receive, as shown in Fig. 3.2. A duty cycle is defined as the ratio of the listen duration to the whole duration of a frame. The listen period is further divided into smaller intervals for sending or receiving SYNC, RTS, and CTS packets. The duration of the listen period is normally fixed depending on physical- and MAC-layer parameters, for example, the radio bandwidth and the contention window size. The duration of the sleep period can be changed according to different application requirements, which actually changes the duty cycle.

In S-MAC, all nodes are free to choose their own listen and sleep schedules. To reduce control overhead, however, neighboring nodes coordinate their sleep schedules and try to adopt the same schedules to listen and sleep, rather than randomly sleep on their own. To establish coordinated or synchronized sleep schedules, each node exchanges its schedule with other nodes by periodically

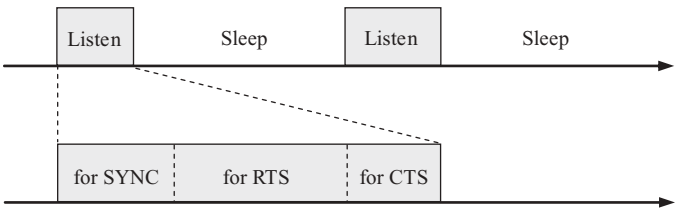


Fig. 3.2 Periodic listen and sleep in S-MAC.

broadcasting a SYNC packet to all its immediate neighbors and maintains a schedule table that stores the schedules of all its known neighbors for listening and sleeping. However, it is not always possible for all neighboring nodes to synchronize their schedules in a multihop network. In this case, S-MAC allows a node to adopt multiple schedules to enable multihop operation in the network.

On the other hand, the clock drift on each node can cause timing errors, which would affect the schedule coordination and synchronization. To address this problem, S-MAC uses relative timestamps instead of absolute ones and at the same time makes the listen period significantly longer than the clock drift. To maintain synchronization, however, each node still needs to periodically update its schedule to prevent long-term clock drift. For this purpose, each node periodically broadcasts its schedule to all its neighbors in a SYNC packet. The SYNC packet is very short and contains the address of the sender and the next sleep time of the sender. The next sleep time is relative to the time when the sender starts to send the SYNC packet. When a node receives the SYNC packet, it will use the new value of the next sleep time to adjust its timer.

In order for a node to receive both SYNC packets and data packets, its listen period is divided into two parts. The first part is for receiving SYNC packets and the second is for receiving RTS packets. Each part is further divided into many timeslots for senders to perform carrier sensing. For example, if a sender has a SYNC packet to send, it starts carrier sensing when the receiver begins listening and randomly selects a timeslot to perform carrier sensing. If it has not detected any transmission by the end of the timeslot, it wins the medium and then starts to send its SYNC packet immediately. Figure 3.3 illustrates the timing relationship between a sender and a receiver in different possible situations, where sender 1 only sends a SYNC packet, sender 2 only sends a unicast data packet, and sender 3 sends both a SYNC packet and a data packet.

The collision avoidance mechanism used in S-MAC is similar to that in the IEEE 802.11 DCF [5]. To avoid collision, S-MAC uses both virtual and physical carrier sensing and adopts the RTS/CTS mechanism to address the hidden terminal problem. In virtual carrier sensing, each transmitted packet carries a duration field that indicates the duration of the transmission. If a node receives a packet destined to another node, it knows how long it needs to keep silent. The node records this value in a variable called network allocation vector (NVA) [5] and sets a timer for it. Every time the NAV timer times out, the node decrements the NAV value until the NVA becomes zero. When a node has data to send, it first checks the NAV value. A nonzero value indicates that the medium is busy. Physical carrier sensing is performed by listening to the channel at the physical layer. The procedure is the same as that for sending SYNC packets. The medium is determined as free if both virtual and physical carrier sensing indicates it is free. All nodes perform carrier sensing before its data transmission. If a node is unable to win the medium, it goes to sleep and wakes up when the receiver becomes free, and listens again. Unicast packets are sent with an exchange of RTS/CTS/DATA/ACK packets between the sender and the receiver, while broadcast packets are sent without exchanging RTS and CTS packets.

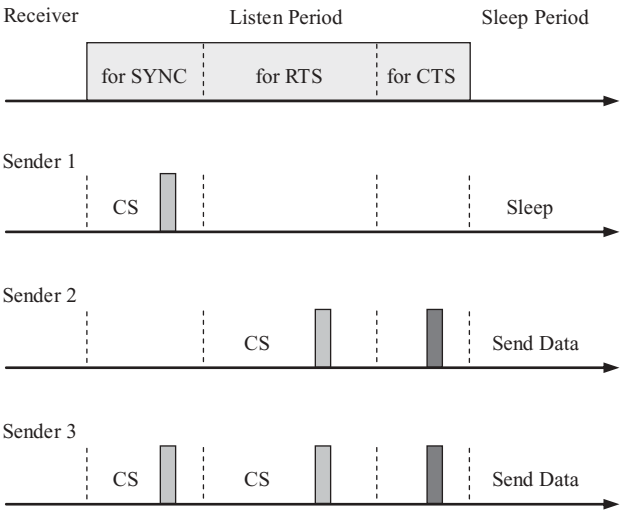


Fig. 3.3 Timing relationship between a receiver and a sender.

To avoid overhearing, S-MAC puts a node into the sleep state after it receives an RTS or CTS packet. Since DATA packets are normally much longer than control packets, this prevents neighboring nodes from overhearing long DATA packets and subsequent ACK packets. A node can wake up after the NAV value becomes zero.

In addition, S-MAC introduces a transmission mechanism called message passing to efficiently transmit a long message in terms of both energy and latency. A message is a collection of meaningful and interrelated units of data, which can be a long series of packets or a short packet. Usually, a receiver needs to obtain all the data units before it can perform in-network data processing or aggregation. In-network data processing or aggregation is an important feature of WSNs, which can greatly save energy consumption by largely reducing the amount of data to be transmitted [15]. However, if a long message is transmitted as a single packet and only a few bits are corrupted, the whole packet needs to be retransmitted, which would result in a high transmission cost. On the other hand, if the long message is segmented into many independent small fragments, it would cause larger control overhead and longer delay because RTS and CTS packets are used in contention for each independent packet. To address this problem, S-MAC segments a long message into many small fragments, and transmits them in a burst. Only one RTS and one CTS are used to reserve the medium for transmitting all fragments. Each fragment is acknowledged separately and is retransmitted if the ACK packet is not received for the fragment. If a neighboring node hears an RTS or CTS packet, it will go to sleep for the time that is needed to transmit all the fragments. Besides RTS and CTS, each fragment or ACK packet also has a duration field, which indicates the time for transmitting all the remaining data fragments and ACK packets and allows a node that wakes up in the

middle of the transmission to return to sleep. This is different from 802.11's fragmentation mode, where each fragment only indicates the presence of an additional fragment rather than all of them. If a node wakes up or a new node joins in the middle of a transmission, it can properly go to sleep no matter whether it is the neighbor of the sender or the receiver. If the sender extends the transmission time because of fragment losses or errors, the sleeping neighbors will not be aware of the extension immediately. However, they will learn it from the retransmitted fragments or ACK packets when they wake up.

S-MAC is much more energy efficient than 802.11. However, due to the fixed sleep time/awake time ratio, some portion of the bandwidth is always unusable and the delay is higher. Overhearing is avoided for unicast traffic, but for broadcast or carrier sense traffic, overhearing is still an unsolved problem. The main drawback of S-MAC is high message delivery latency as S-MAC is designed to sacrifice latency for energy savings.

3.4.1.2 DS-MAC. DS-MAC is an S-MAC protocol with a dynamic duty cycle proposed by Lin et al. [16], which aims to achieve a good tradeoff between energy consumption and latency without incurring much overhead. In DS-MAC, each sensor node assumes all functionalities defined in S-MAC and each receiver node keeps track of its own energy consumption level and average latency. To achieve the intended tradeoff, each node attempts to dynamically adjust its sleep–wakeup cycle time based on the current energy consumption level and the average latency it has experienced. The average latency is used as an approximate estimation of the current traffic condition and an indicative parameter for a receiver node.

With DS-MAC, each node uses the SYNC packets to set up and maintain clock synchronization as done similarly in S-MAC. Unlike S-MAC, which adopts a constant duty cycle, DS-MAC adopts a common initial basic duty cycle at all sensor nodes. If a receiver node finds that the latency becomes intolerable, it will double the original duty cycle by reducing the sleeping period accordingly without changing the listening period. As a result, a node with an increased duty cycle can get more chances to receive packets from other senders instead of blocking them while sleeping. Therefore, DS-MAC alleviates the high-latency problem with S-MAC under high-traffic load while still keeping high energy efficiency under low traffic load.

To implement DS-MAC, some additional protocol overhead needs to be introduced, including a "duty cycle" field and a "delay" field in each SYNC packet. Compared with the S-MAC implementation, each sensor node also needs to maintain its own average latency and energy consumption level, which requires additional storage overhead and processing overhead. However, all these overheads are negligible and can actually be compensated by the reduced queuing cost due to the decreased latency.

3.4.1.3 MS-MAC. MS-MAC is an adaptive mobility-aware MAC protocol proposed by Pham and Jha [17] to address the mobility issue in mobile sensor

applications like smart patient assistance and rare animal monitoring. In such mobile sensor applications, each sensor node could be highly mobile and the level of mobility may vary significantly during different periods of a day. Before MS-MAC, most MAC protocols proposed for WSNs only consider stationery sensor nodes, which may largely degrade the network performance if directly applied to mobile scenarios. To improve the network performance in mobile scenarios, a MAC protocol must be mobility aware and able to adapt to different levels of mobility. To this end, MS-MAC adopts the design of S-MAC and extends the protocol to support mobile sensor nodes. For a stationery scenario, MS-MAC operates similar to S-MAC in order to conserve energy. For a highly mobile scenario, it switches to an operating mode similar to IEEE 802.11. The protocol uses any change in the received signal levels of periodical SYNC messages as an indication of mobility and if necessary triggers a mobility handling mechanism, which dynamically adjusts the frequency of mobility handling actions based on the presence of mobile nodes and their moving speeds. With such a mobility estimating and handling mechanism, MS-MAC is highly energy efficient for stationery scenarios while also maintaining a certain level of network performance in scenarios with mobile sensor nodes.

3.4.1.4 D-MAC. D-MAC is an energy-efficient and low-latency MAC protocol by Lu et al. [18] for data gathering in WSNs. This protocol was proposed to address the data forwarding interruption problem in multihop data delivery and its primary goal is to achieve both energy efficiency and low latency. To deliver data from a source sensor node to the sink through a multihop path, most MAC protocols that use active–sleep duty cycles (e.g., S-MAC) suffer from a data forwarding interruption problem, where some nodes on the multihop path cannot be aware of the on-going data delivery. For example, in an implicit duty-cycle adjusting mechanism, a node remains active when it overhears ongoing transmissions in the neighborhood [2]. Since the overhearing range of a node is limited by its radio sensibility, a node that is out of the overhearing range of both the sender and the receiver of a data transmission cannot be aware of the ongoing data transmission and thus goes to sleep until the next cycle. As a result, the data forwarding process will be interrupted at a node whose next hop toward the sink is out of the overhearing range. The data packet has to wait in the queue until the next active period, resulting in sleep latency. For an explicit duty-cycle adjusting mechanism [19], it uses duty-cycle adjusting messages to directly adjust the duty cycle. Since the adjusting messages can only be forwarded a limited number of hops in an active period, a node out of the range goes to sleep after its basic duty cycle, leading to the interruption of the data forwarding as well.

To address this problem, D-MAC employs a staggered wake-up schedule to enable continuous data forwarding on a multihop path. In WSNs, the primary traffic is for data gathering from sensor nodes to a sink. The data delivery paths from multiple sources to one sink constitute a data gathering tree [15], in which data flows are unidirectional and all nodes except the sink forward the packets

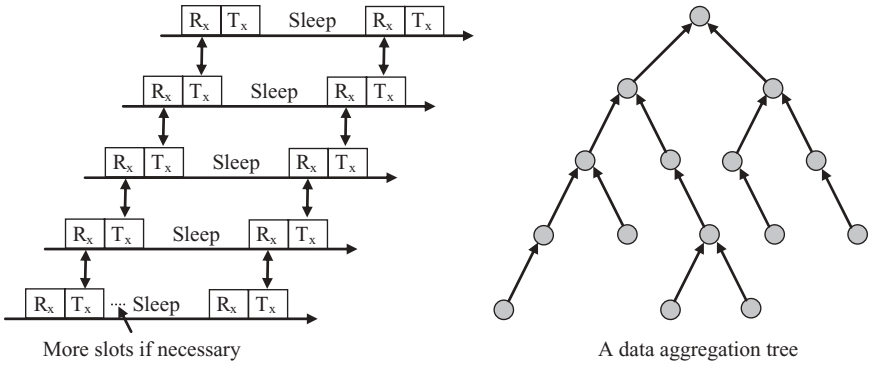


Fig. 3.4 An aggregation tree in D-MAC and its implementation.

they receive to the next hop. To enable continuous data forwarding on a multihop path, D-MAC staggers the schedule of the nodes on the multihop path and allows the nodes to wake up sequentially like a chain reaction, as shown in Fig. 3.4. In the schedule, an interval is divided into three periods (or states): receiving, sending, and sleeping. In the receiving period, a node is expected to receive a packet and send an ACK packet back to the sender. In the sending period, a node tries to send a packet to its next hop and receive an ACK packet. In the sleeping period, a node turns off its radio to save energy. The receiving and sending periods have the same length of μ, which is long enough for transmitting and receiving one packet. Depending on its depth d in the data gathering tree, a node sets its wake-up schedule $d\mu$ ahead from the schedule of the sink.

With the operation like a multihop chain, each node periodically goes into the receiving, sending, and sleeping states. As a result, when there is no collision, a packet will be forwarded sequentially along a multihop path to the sink without sleep latency. However, when a node has multiple packets to send at a sending slot, it needs to increase its own duty cycle, and has to request other nodes on the multihop path to increase their duty cycles as well. For this purpose, D-MAC employs a slot-by-slot renewal mechanism, where a *more data* flag is piggybacked in the MAC header to indicate the request for an additional active period with little overhead. Before a node transmits a packet, it first sets the *more data* flag in the packet if either its buffer is not empty or it received a packet with a *more data* flag from its previous hop. The receiver will check if the *more data* flag is set in the received packet, and if the flag is set, it will also set the *more data* flag of its ACK packet to the sender. With this slot-by-slot renewal mechanism, D-MAC can adaptively adjust the duty cycles to the traffic load.

In addition, D-MAC employs a data prediction mechanism to solve the problem when each single source has a traffic rate low enough for the basic duty cycle to handle, but the aggregated rate at an intermediate node is larger than the basic duty cycle can handle. When multiple children of a node have packets to send in the same sending slot, data prediction is used to request active

sending slots. When multiple nodes on the same level of the data gathering tree with different parents compete for the channel, the data prediction mechanism is unable to handle the interference. In that case, an explicit control packet called *More-to-Send* packet is used to adjust the duty cycle under the interference.

3.4.1.5 Sift. Sift is a CSMA based MAC protocol proposed by Jamieson et al. [20] for handling spatially correlated contention in event-driven WSNs. It is motivated by the observations that sensor networks are usually event driven and have spatially correlated contention. In most sensor networks, multiple sensors are deployed in the same geographical area and share the same radio medium. When an event of interest occurs, the sensors that observe the event will send messages to report the event. If multiple sensors have messages to send at the same time, it will cause contention for the radio medium, which is called spatially correlated contention. However, in many sensor applications, not all the sensing nodes that observe an event need to report the event and the number of contending nodes changes over time. For these reasons, a MAC protocol for sensor networks should be able to not only handle spatial correlation, but also adapt to the changes in the number of contending nodes.

The above observations lead to a problem in sensor network MAC protocol design that is different from classical MAC protocol design. For a shared medium with N nodes observing an event and contending for transmission at the same time, a MAC protocol should be designed with the objective to minimize the time taken to send R of N messages without collisions. If R = N, this problem becomes the throughput optimization problem in classical MAC protocol design. If R < N, the objective is to allow the first R winners in the contention to send their messages through as quickly as possible, with the remaining nodes backing off their transmissions. Sift is a randomized CSMA protocol designed to solve this problem. Unlike traditional MAC protocols, Sift does not use a time-varying contention window from which a node randomly picks a contention slot. Instead, to reduce the latency for delivering event reports, it uses a small and fixed contention window of 32 slots, where the duration of each slot is on the order of tens of microseconds, and a geometrically increasing non-uniform probability distribution for picking a transmission slot in the contention window. The key difference between Sift and traditional MAC protocols, for example, IEEE 802.11, is that the probability distribution for selecting a contention slot is not uniform.

With the non-uniform probability distribution, a node competes for a contention slot within the contention window with other nodes based on a shared *belief* of the current population size N, which changes after every slot with no transmission. This belief starts with some large value and a correspondingly small probability for per node transmission. If no node transmits in the first slot, each node will reduce its belief of the number of competing nodes by multiplicatively increasing its transmission probability for the second slot. This process is repeated for each slot, allowing for the competition to happen at geometrically decreasing possible values in the same small total number of contention slots. As a result, Sift enables the winner to be chosen quickly in a wide range of potential

population sizes without incurring long latency due to collisions. If exactly one node happens to select some contention slot, it will start to transmit in that slot. When its transmission is done, all other competing nodes will randomly select new contention slots, and repeat the process of backing off over the fixed contention window. The same process happens if two or more nodes happen to select the same contention slot.

The simulation results show that Sift can offer up to a sevenfold latency reduction compared to IEEE 802.11 as the size of the network scales up to 512.

3.4.1.6 T-MAC. Timeout-MAC (T-MAC) is an adaptive energy-efficient MAC protocol proposed by Dam and Langendoen [21] for WSNs. The basic idea of T-MAC is to reduce idle listening by introducing a dynamic duty cycle and transmitting all messages in bursts of variable size in active periods and sleeping between active periods. To maintain an optimal active period under variable traffic load, T-MAC dynamically determines the length of an active period by simply timing out if nothing is heard.

In T-MAC, each node periodically wakes up to communicate with its neighbors and then go to sleep until the next frame, as shown in Fig. 3.5. The nodes communicate with each other following a RTS-CTS-Data-ACK sequence, which provides both collision avoidance and reliable transmission. A node keeps listening and potentially transmitting as long as it is in an active period. If no *activation event* occurs for a threshold time T_h, an active period will end and the node will go to sleep. An activation event can be (1) the timing out of a periodic frame timer; (2) the reception of a data packet on the radio; (3) the sensing of communication on the radio; (4) the end of transmission of a node's own data packet or acknowledgment; or (5) the end of transmission of a neighbor's data packet. Obviously, T_h determines the minimum amount of idling listening per frame. As a result, all nodes transmit at the beginning of each active period. Since data packets between active periods need to be buffered, the buffer capacity determines an upper bound on the maximum frame time.

The simulation results show that T-MAC and S-MAC achieve similar energy consumption reductions (up to 98%) compared to CSMA. However, T-MAC outperforms S-MAC by a factor of 5 in a sample scenario with variable traffic load.

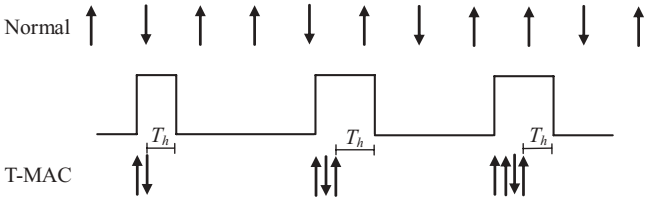


Fig. 3.5 The T-MAC frames with adaptive active periods.

3.4.1.7 WiseMAC. Wireless Sensor MAC (WiseMAC) is an energy-efficient MAC protocol proposed by Hoiydi et al. [22] for both multihop and infrastructure networks. To improve energy efficiency, it combines non-persistent CSMA with synchronized preamble sampling to mitigate idle listening. In the preamble sampling technique, all nodes in a network sample the medium with the same constant period, but their relative sampling schedule offsets are independent. If a node finds the medium busy, it will continue to listen until it receives a data packet or the medium becomes idle again. At a transmitting node, a wake-up preamble of size equal to the sampling period is transmitted ahead of each data packet to alter the receiving node. This technique provides low power consumption when traffic is low. However, the fixed-length preamble leads to high power consumption overhead in both transmission and reception. To reduce the power consumption incurred by the fixed-length preamble, WiseMAC introduces an effective scheme to dynamically reduce the length of the wake-up preamble. This scheme learns the sampling schedules of direct neighbors and exploits these schedules to reduce the length of a wake-up preamble. The nodes learn or refresh their neighbor's sampling schedule during each data communication by piggybacking the remaining time to the next sampling instant in the acknowledgment messages. Each node keeps an updated table of the sampling time offsets of its neighbors. Based on these tables, WiseMAC schedules a transmission such that the middle of the wake-up preamble coincides the sampling time of the destination, as shown in Fig. 3.6.

WiseMAC requires no setup signaling or network-wide time synchronization. The combination of preamble sampling and wake-up preamble-length minimization provides both ultra-low power consumption under low-traffic conditions and high energy efficiency under high-traffic conditions. Although WiseMAC was originally designed for multihop networks, it is also suitable for the downlink of an infrastructure network [23]. It has been shown that under low-traffic conditions, WiseMAC leads to lower power consumption than the power-save scheme

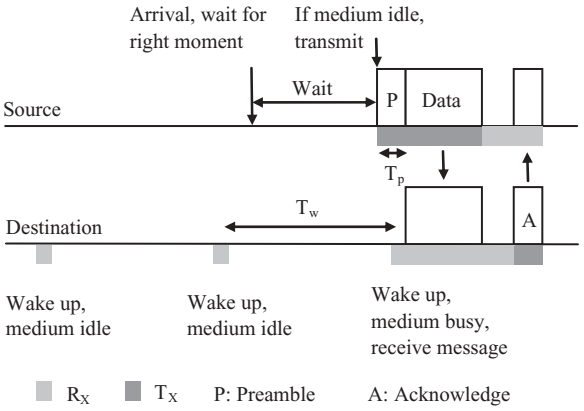


Fig. 3.6 WiseMAC preamble minimization.

in IEEE 802.11 [5] and IEEE 802.15.4 [24]. Therefore, WiseMAC can be used in a hybrid network topology to receive data from both energy-constrained sensor nodes and energy-unconstrained base stations.

3.4.1.8 CSMA Based MAC with Adaptive Rate Control. Woo and Culler [25] proposed a CSMA based MAC protocol that combines CSMA with an adaptive rate control mechanism. This protocol considers a specific network scenario in which a base station collects data from all sensors in a field of interest and the applications generate periodic and highly correlated traffic. It aims at achieving both energy efficiency and fair bandwidth allocation for all nodes in a multihop network. In such a network scenario, the contention for channel bandwidth between originating traffic and pass-through traffic at a node has a direct impact on the multihop fairness in bandwidth allocation. For this reason, a MAC protocol should be able to control the rate of originating data of a node in order to allow pass-through traffic to more easily access the channel and reach the base station. On the other hand, a progressive signaling mechanism is also needed for pass-through traffic to inform the nodes down in the network to lower their rate of originating data. This will in turn decrease the aggregate pass-through traffic and open up the channel for nodes closer to the base station to originating data.

For this purpose, Woo and Culler [25] proposed an adaptive rate control mechanism to balance the rates of originating traffic and pass-through traffic at a node. With this mechanism, a node periodically attempts to transmit a packet into the channel. If the packet is successfully transmitted, it becomes part of the pass-through traffic. As the packet is routed by the node's parent node, it signals that the channel can still accommodate more traffic and thus the node can increase its transmission rate. However, if the packet is not transmitted into the channel successfully, it signals that the channel is congested. In this case, the node decreases its rate of originating data and backoffs in order to achieve a phase change effect. In this way, the originating data rate can adapt to the pass-through traffic. Similarly, the pass-through traffic will also adapt to the originating traffic. Specifically, if a node transmits lots of originating traffic into the channel, the rate of transmitting pass-through traffic will decrease. This information is propagated down into the network, which would ultimately decrease the aggregate path-through traffic. In addition, the rate control mechanism uses a linear increase and multiplicative decrease approach to control the transmission rate. While the linear increase leads to more aggressive channel competition, the multiplicative decrease controls the penalty for a transmission failure. Since it costs more to drop pass-through traffic than to drop originating traffic, the penalty associated with a pass-through data transmission failure is smaller than that with an originating data transmission failure, which ensures that pass-through traffic is more favored over originating traffic.

It has been shown that the CSMA based MAC protocol is most effective in achieving fair bandwidth allocation while being energy efficient for both low- and high-duty cycles of network traffic. However, since it is based on CSMA, it may suffer from high control overheads and the hidden terminal problem.

3.4.2 Contention-Free Protocols

This section introduces several contention-free MAC protocols that have been proposed for WSNs.

3.4.2.1 Traffic-Adaptive Medium Access. The traffic-adaptive medium access (TRAMA) protocol is a TDMA based MAC protocol proposed by Rajendran et al. [26] to provide energy-efficient collision-free channel access in WSNs while maintaining good throughput, acceptable latency, and fairness. In TRAMA, energy efficiency is achieved by ensuring collision-free data transmissions and allowing nodes to switch to a low-power idle state when they are not transmitting or receiving. To maintain throughput and fairness, TRAMA uses a transmitter-election algorithm that is inherently fair and promotes channel reuse as a function of the competing traffic around a given source or receiver.

The TRAMA protocol assumes a single time-slotted channel for both data and signaling transmissions. Time is divided into a series of random-access periods and scheduled-access periods, which alternate with each other, as shown in Fig. 3.7. A random-access period, also referred to as a signaling slot, is further divided into smaller signaling slots and a scheduled-access period, also referred to as a transmission slot, into smaller transmission slots. Since the data rate in a sensor network is relatively low, the bit duration is much larger than typical clock drifts. For this reason, slot synchronization can be implemented by using a simple time-stamp mechanism or a technique, for example, a global positioning system (GPS). The TRAMA protocol starts with a random access period where each node randomly selects a timeslot and then transmits. A node can only join the network during a random access period. The duty cycle of random access and scheduled access depends on the type of network. In a more dynamic scenario, random access periods should occur more often while in a more static scenario the interval between random access periods can be larger because topology changes need to be accommodated only occasionally. Depending on the type of the application, there is little or no mobility in a sensor network. Accordingly, the random access

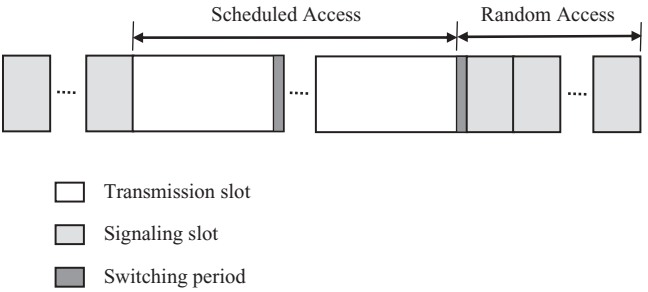


Fig. 3.7 Time-slot structure in TRAMA.

periods are mainly used to allow node addition and deletion. During a random access period, all nodes must be in either a transmitting state or a receiving state so that they can send out their neighborhood information and receive information from neighbors. Due to collisions, signaling packets may be dropped, which can lead to inconsistent neighborhood information between different nodes. To ensure consistent neighborhood information with some degree of confidence, the duration of a random access period and the number of retransmissions of a signaling packet are set accordingly. In addition, time synchronization could also be performed during this period.

The TRAMA protocol consists of three components: the neighbor protocol (NP), the schedule exchange protocol (SEP), and the adaptive election algorithm (AEA). Both the NP and the SEP allow nodes to exchange 2-hop neighborhood information and their schedules. The AEA uses the neighbor and schedule information to select transmitters and receivers for the current timeslot, allowing all other nodes to switch to a low-power mode.

The NP collects 2-hop neighborhood information by exchanging small signaling packets among neighboring nodes during the random access periods. A signaling packet carries incremental neighborhood updates. If there are no updates, it is sent as a "keep alive" beacon. Each node sends incremental updates about its 1-hop neighborhood. These signaling packets are also used to maintain connectivity between the neighbors. A node times out a neighbor if it does not hear from that neighbor for a certain period of time. The updates are retransmitted such that 99% of success is ensured. Since a node knows the 1-hop neighbors of its 1-hop neighbors, consistent 2-hop neighborhood information can eventually be obtained.

Transmission slots are used for transmitting data traffic and also for exchanging traffic-based schedule information between neighboring nodes. The schedule information is required by the transmitter (i.e., slot reuse) and receiver (i.e., sleep-state switching) selection. A node has to announce its schedule via a schedule packet using the SEP before actual data transmissions. The SEP updates the schedule information periodically during the scheduled-access periods and thus maintains consistent schedule information among neighbors.

The AEA is used to select transmitters and receivers to achieve collision-free transmissions using the information obtained by the NP and the SEP. To achieve energy efficiency in a collision-free transmission, it is necessary to select both a transmitter and a receiver(s) for a particular timeslot. Selecting a transmitter randomly may lead to collisions, while selecting a transmitter, but not a receiver(s), may lead to energy waste because all the neighbors around a selected transmitter have to listen in the timeslot even if they are not to receive any data. Moreover, selecting a transmitter without considering its traffic leads to low-channel utilization because the selected transmitter may not have data to send to the selected receiver. Therefore, the AEA uses traffic information in selecting transmitters and receivers in order to improve channel utilization.

According to the simulation results, TRAMA can achieve significant energy savings due to a higher percentage of sleep time. It can also achieve higher

throughput compared to contention-based protocols due to reduced collision probability. However, TRAMA has a higher delay than contention-based protocols due to a higher percentage of sleep time and thus is suitable for applications that are not delay sensitive, but require high delivery throughput and energy efficiency.

3.4.2.2 Self-Organizing Medium Access Control. Self-organizing medium access control for sensor networks (SMACS) is a distributed MAC protocol proposed by Sohrabi et al. [27], which enables a collection of nodes to discover their neighbors and establish schedules for communicating with them without the need for any local or global master nodes. In SMACS, each node is able to turn its radio on and off, and tune the carrier frequency to different bands. The number of available bands is relatively large. To form a flat topology, the neighbor discovery and channel assignment phases are combined. A channel is assigned to a link immediately once the existence of the link is discovered. Therefore, by the time all nodes hear from all their neighbors, they will have formed a connected network, where there is at least one multihop path between any two distinct nodes. In SMACS, only partial information about radio connectivity in the vicinity of a node is used to assign timeslots to links. Each node maintains a TDMA-like frame called superframe, in which it schedules different timeslots to communicate with its known neighbors. In each timeslot, a node only communicates with one neighbor. However, there is a potential for time collisions with slots assigned to adjacent links whose existence is unknown at the time of channel assignment. To reduce the likelihood of collisions, each link operates on a different frequency, which is chosen randomly from a large pool of frequencies when the links are established. After a link is established, a node knows when to turn on its transceiver ahead of time to communicate with another node and will turn off when there is no communication. By using such scheduling, energy savings can be achieved at the node. On the other hand, since link assignment is done without a need for collecting global connectivity information or even connectivity information that reaches farther than one hop away, significant energy savings can be achieved. The drawback of SMACS is its low bandwidth utilization. For example, if a node only has packets to be sent to one neighbor, it cannot reuse the timeslots scheduled for other neighbors.

3.4.2.3 Distributed Energy-Aware MAC. The distributed energy-aware MAC (DE-MAC) protocol is a TDMA based MAC protocol proposed by Kalidindi et al. [28] to address the energy management problem in WSNs. The DE-MAC protocol exploits the inherent features of TDMA to avoid energy waste caused by collision and control overhead, and employs a periodical listening and sleeping mechanism to avoid idle listening and overhearing. Unlike some existing MAC protocols that treat all nodes equally with respect to energy conservation, DE-MAC treats those critical nodes (i.e., with lower energy) differently by using them less frequently to achieve load balancing among all nodes. The criticality of a sensor node can be based on local state information, for

example, relative energy levels within a group of neighbor nodes. For this purpose, a group of neighbor nodes periodically perform a local election process based on their energy levels to elect the worst-off node(s) as the winner(s) and let the winner(s) sleep more than its (or their) neighbor nodes. The local election process is fully integrated with the regular TDMA schedule and thus would not cause additional throughput loss. More specifically, the protocol initially assigns the same number of transmission slots to each node in a TDM frame. A node can independently decide to initiate an election if its current energy level is below a threshold value. Once an election is initiated, each node sends its energy level to all of its neighbors, which is included to its regularly scheduled transmission packet during its scheduled timeslot. To receive the energy level information from other nodes, all nodes listen to all transmitted packets. There are no sleeping nodes when other nodes are transmitting. This is to enable the integration of leader-election with regular TDMA transmission and thus save bandwidth. At the end of the election process, the node with the minimum energy level is elected as a winner. Once one or more winners are elected, all the losers reduce the number of their timeslots by a constant factor (e.g., two) and the winners have timeslots twice the number of the losers. By performing such slot adjustment, the idling listening time of those critical nodes are reduced, leading to more energy savings in the critical nodes. The simulation results show that DE-MAC achieves a significant gain in energy savings compared to the simple TDMA based MAC protocol in Ref. [29].

3.4.2.4 Implicit Prioritized MAC. The implicit prioritized access protocol is a MAC protocol based on earliest deadline first (EDF), which was proposed by Caccamo et al. [30] to address the MAC problem in a cellular structure network. It considers the periodic nature of sensor network traffic and focuses the network performance in terms of guaranteed bounded delay. For a cellular structure network, the network is spatially divided into multiple cells. Within each cell, the sensor nodes are fully connected in peer to peer and intra-cell messages are exchanged using EDF with implicit contention in a multicast manner. Between adjacent cells, frequency division multiplexing (FDM) is used to avoid conflicts and inter-cell messages are exchanged using more capable router nodes. The inter-cell messages are first sent to the router node within the same cell and then forwarded by the router node through the network hop by hop. For intra-cell communication, the MAC protocol uses a combined deterministic scheduling and local carrier sensing mechanism, which replicates the EDF schedule at each node for data transmission. Since the schedules are identical at different nodes, each node can know which node has the message with the earliest deadline and has the right to transmit next. If a node is not listening to the channel, it is able to select the right frame to transmit simply by counting the frames and assuming that all previous messages used all their scheduled frames. Otherwise, if a node detects an early completion of the previous message by listening to the channel, the unused frames are exploited by using a proposed FRAme SHaring (FRASH) technique, which improves the network utilization.

3.4.2.5 Contention-Free Scheduling TDMA MAC. The contention-free scheduling TDMA MAC protocol proposed by Carley et al. [31] is a TDMA based MAC protocol with a contention-free message scheduler at each node. The message scheduler uses a periodic message model to construct a contention-free schedule for transmitting and receiving the messages of a node to ensure that there is no contention in the transmission medium and even in the message scheduler. Specifically, a contention-free periodic message set is first obtained through message attribute assignment and a periodic task set is then constructed from a given contention-free periodic message set by translating the attributes of each message to task attributes. Since there is no contention in the message scheduler, each node only needs to schedule the messages of its own. For this reason, the complexity of each node only grows with the number of messages transmitted and received by that node, rather than the size of all messages in the network, and therefore is often constant. This largely reduces the space and time complexity of the network scheduler and thus results in memory, processor utilization, and power consumption savings. Moreover, since the message scheduler of each node only schedules the messages that are transmitted or received by that node, it is possible to combine the message scheduler with the task scheduler in that node. Therefore, this MAC protocol is highly scalable to large sensor networks.

3.4.2.6 CDMA Sensor MAC. The CDMA Sensor MAC (CS-MAC) protocol is a self-organizing location-aware MAC protocol proposed by Liu et al. [32] for DS-CDMA based sensor networks, which is suitable for applications with high traffic and stringent latency requirements, for example, battlefield surveillance. The design objectives of the CS-MAC protocol include energy efficiency, low latency, fault tolerance, and scalability. The assumptions for the protocol design include the following: (1) each node starts up at approximately the same time; (2) each node is able to estimate its location using GPS or alternate techniques; (3) each node is static during the network lifetime, which implies that the estimation of its location only needs to be performed once and thus the energy consumption for the location estimation can be ignored.

In CS-MAC, the network formation process consists of several different phases, as shown in Fig. 3.8. In the location broadcast phase, each node broadcasts its location information to the neighbors within its radio range. To ensure that each node can get a chance for a successful transmission, CS-MAC uses a large contention window and allows each node to broadcast several times. At the end of this phase, each node should have a list of neighbors within its radio range with their locations, called *redundant neighbor list* (*RNL*).

In the TORN (turning off redundant node) phase, a node that is redundant for a sensing application is turned off to conserve energy and reduce network interference. Specifically, each node first ranks all neighbors in the RNL based on their distances to the node itself. If sensors are densely deployed, a node will have a high probability to have redundant neighbors within a radius *Sensing Resolution* (SR), which denotes the sensing accuracy required by an application. Note that SR is an application-specific criterion that is different from the sensing

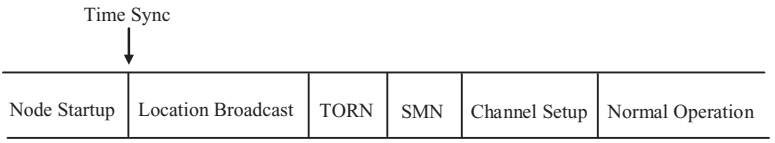


Fig. 3.8 Network formation phases.

range. Then each node uses a contention-based protocol to negotiate who should keep active. A random timer is set to avoid collisions. The first node that gets the medium to transmit will inform its redundant neighbor(s) to turn off by including their ID numbers in a request. A node will turn off itself upon receiving such a request from a neighbor, and will wake up later to check the energy level of the active node and decide whether it should take over, thus providing fault tolerance. Obviously, a node with more redundant neighbors will less likely become an active node during the TORN phase. At the end of the TORN phase, only active nodes are left in the network. The resulting neighbor list in each active node is called *non-redundant neighbor list* (*NNL*), which will be used in the SMN phase (select minimum neighbor).

In the SMN phase, each active node has a list of location information of the active neighbors within its radio range. A node will not select all of them as neighbors. Instead, it only selects a node as its neighbor if there is no other neighbor that can provide a multihop path with lower power consumption. For this purpose, an algorithm is designed for a sensor node (or seed node) to select its neighbors from the NNL. After the SMN phase, each active node only has a minimum set of neighbors called *minimum neighbor list* (MNL). This MNL will be used in the channel setup phase, where a peer-to-peer communication channel will be set up for each neighbor in the MNL and the seed.

In the channel setup phase, each node sets up connections to all its neighbors in the MNL. It first estimates the transmission power required to reach its furthermost neighbor in the MNL and then uses this power level for negotiation. This allows a node that is far enough from this node to initiate another set-up process simultaneously. CSMA/CA is used by nodes to set up connections with each other. Once a node wins the channel, it will hold the channel until it finishes the channel allocation with all its neighbors in the MNL.

CS-MAC uses a combination of DS-CDMA and frequency division in channel allocation to reduce channel interference, and consequently the message latency in the network. The simulation results in Ref. [33] have shown that CS-MAC can significantly reduce average latency and average energy consumption per message compared to traditional MAC protocols for sensor networks.

3.4.3 Hybrid Protocols

This section introduces several hybrid MAC protocols for WSNs, which combine the features of both contention-based and contention-free protocols.

3.4.3.1 Spatial TDMA and CSMA with Preamble Sampling. Spatial TDMA and CSMA with Preamble Sampling is a hybrid MAC protocol proposed by El-Hoiydi [33] for low-power sensor networks. This protocol assumes that data traffic is periodical while signaling traffic is sporadic. All sensor nodes have two communications channels: data channel and control channel. In the data channel, a spatial TDMA protocol is used to transport periodic and frequent data while in the control channel a low-power CSMA protocol is used to transport sporadic signaling traffic. In classic CSMA, a node has to listen to the channel all the time except during its transmission. Since idle listening consumes much energy, classic CSMA is not preferred for a sensor network where the channel is idle most of time. This protocol introduces a low-power CSMA protocol that is obtained by combining it with the preamble sampling technique used in paging systems [34]. With this technique, a node sends a preamble of size T_p before every message. A receiver will sleep and wake up every T_p to check whether the channel is idle or busy. When a preamble is detected, the receiver will continue to listen until the beginning of the packet is found and the packet is received. This allows a node to sleep most of the time when the channel is idle, and can thus improve energy efficiency and prolong network lifetime.

3.4.3.2 Z-MAC. Zebra-MAC (Z-MAC) is a hybrid MAC protocol proposed by Rhee et al. [35], which combines the strengths of TDMA and CSMA while offsetting their weaknesses. The main feature of Z-MAC is its adaptability to the dynamic contention level in the network. Under low contention, it behaves like CSMA and can achieve high channel utilization and low latency. Under high contention, it behaves like TDMA and can achieve high channel utilization and reduce collisions among 2-hop neighbors at a low cost. Moreover, it is also robust to time synchronization errors, slot assignment failures, time-varying channel conditions, and dynamic topology changes.

Zebra-MAC uses CSMA as the basic MAC mechanism and meanwhile uses a TDMA schedule as a "hint" to improve contention resolution. In Z-MAC, timeslot assignment is performed at the time of deployment, which incurs high initial overhead. The argument behind this is that the high initial overhead is distributed over a long period of network operation and eventually can be compensated by improved throughput and energy efficiency. The slot assignment is performed by DRAND [36], an efficient scalable scheduling algorithm, which is a distributed implementation of RAND [37], a centralized channel scheduling algorithm. After the slot assignment, each node reuses its assigned slot periodically in every predetermined period, call frame. A node assigned to a timeslot is called an owner of that slot and the others the non-owners of that slot. Since GRAND allows any two nodes beyond their 2-hop neighborhoods to own the same timeslot, there can be more than one owner per slot.

Unlike TDMA, a node may transmit during any timeslot in Z-MAC. Before a node transmits during a slot (not necessarily at the beginning of the slot), it always performs carrier sensing and transmits a packet when the channel is idle. However, an owner of that slot always has a higher priority over its non-owners

in accessing the channel. To implement the priority, Z-MAC adjusts the size of the initial contention window so that the owners are always given earlier chances to transmit than the non-owners. In this way, Z-MAC reduces the chance of collisions during the slots when the owners have data to transmit. For a slot when the owners do not have data to transmit, the non-owners can use it. Therefore, Z-MAC can dynamically adjust the behavior of MAC between CSMA and TDMA, depending on the contention level in the network.

By combining CSMA and TDMA, Z-MAC becomes more robust to time synchronization errors, slot assignment failures, time-varying channel conditions, and dynamic topology changes than a stand-alone TDMA. In the worst case, it comes back to CSMA. Since Z-MAC only requires local synchronization among sending nodes in 2-hop neighborhoods, a simple local synchronization mechanism is employed, in which each sending node adjusts its synchronization frequency based on its current data rate and resource budget. The simulation results show Z-MAC has better performance than B-MAC [38] under medium to high contention, but slightly worse performance under low contention, especially in terms of energy efficiency. Even in the case when clocks are completely unsynchronized and some degree of slot assignment failures occurs, the performance of Z-MAC is comparable to that of CSMA.

3.4.3.3 Funneling-MAC. Funneling-MAC is a hybrid TDMA and CSMA/CA MAC protocol proposed by Ahn et al. [39] for WSNs. It aims at addressing the unique funneling effect [40], where events generated in a sensor field travels hop-by-hop in a many-to-one traffic pattern toward one or more sinks, as shown in Fig. 3.9. This funneling effect results in a significant increase in transit traffic intensity and thus packet congestion, collision, loss, delay, and energy consumption as events move closer toward the sink(s). The sensor nodes closer to the sink, typically within a small number of hops, also called the intensity or funneling region, will loose a larger number of packets and consume much more energy than the nodes further away from the sink, thus largely reducing the operational

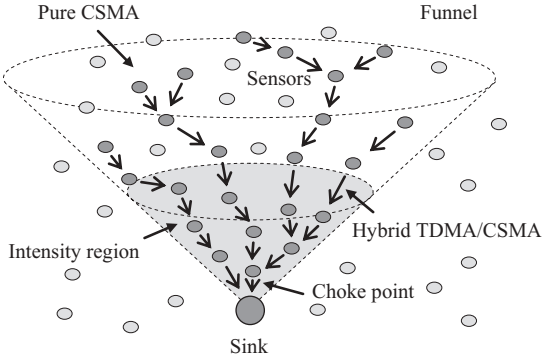


Fig. 3.9 Funneling effect in wireless sensor networks.

lifetime of the entire network. To increase network lifetime, it is desired to reduce the traffic in the intensity region and thus alleviate this funneling effect, which presents a big challenge in the network design.

The funneling MAC protocol is a localized sink-oriented hybrid TDMA and CSMA/CA MAC protocol for operating in the intensity region of the event funnel. It is based on pure CSMA/CA, which is implemented not only in the funneling region, but also network wide. Meanwhile, it uses a local TDMA scheduling in the funneling region only to provide additional scheduling opportunities to the nodes closer to the sink. It is "sink-oriented" because the TDMA scheduling of sensor events in the funneling region is performed by the sink node rather than by resources-limited sensor nodes. It is "localized" in the sense that TDMA only operates in the funneling region close to the sink rather than in the whole sensor field. Moreover, the depth of the intensity region is also computed and maintained by the sink node. By using TDMA in a localized manner, and putting more management on the sink, the scalable problem is solved for the deployment of TDMA in a sensor network. The experimental results show that the funneling MAC effectively alleviates the funneling effect, improves throughput, loss, and energy efficiency, and more importantly significantly outperforms other representative protocols, for example, B-MAC [38], a default protocol in TinyOS [41], and more recent hybrid MAC protocols, for example, Z-MAC.

3.5 SUMMARY AND FUTURE DIRECTIONS

Medium access control plays an important role in improving energy efficiency and network performance of WSNs. This chapter introduced the fundamental concepts on MAC, discussed the major challenges in MAC design, and presented a survey of MAC protocols for WSNs. Although a variety of MAC protocols have been proposed for sensor networks, no protocol has been standardized yet. The primary reason is that sensor networks are application specific and thus a MAC protocol is usually application dependent. Basically, TDMA and CSMA are the most common underlying MAC protocols that are used for sensor networks. The major advantage of TDMA is its collision-free nature, which can significantly improve energy efficiency under high traffic load. However, it has higher delay and lower throughput under low traffic load due to idle timeslots. Moreover, TDMA requires strict time synchronization between different sensor nodes, and has limited scalability and adaptability to network changes. In contrast, CSMA are contention based, which results in lower energy efficiency and higher delay under high traffic load, but can reduce delay and has higher throughput under low traffic load. Depending on specific applications, a MAC protocol can incorporate TDMA or/and CSMA with other techniques to meet different performance requirements. Due to the limitation of space, there are many other MAC protocols not included in this chapter. The reader is referred to Refs. [42–52] for further readings. To further improve the network performance, there is a trend to take into account the effects across multiple protocol layers in the design of MAC protocols, which provides many research opportunities in the future.

REFERENCES

[1] I. Demirkol, C. Ersoy, and F. Alagoz, "MAC protocols for wireless sensor networks: A survey", *IEEE Communications Magazine*, Apr. 2006, pp. 115–121.

[2] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks", *IEEE/ACM Transactions on Networking*, vol. 12, no. 3, June 2004, pp. 493–506.

[3] A. S. Tanenbaum, *Computer Networks* (3rd ed.), Prentice-Hall, Inc., Upper Saddle River, NJ, 1996.

[4] F. Tobagi and L. Kleinrock, "Packet switching in radio channels: Part II—the hidden terminal problem in carrier sense multiple access and the busy-tone solution", *IEEE Transactions on Communications*, vol. 23, no. 12, Dec. 1975, pp. 1417–1433.

[5] Wireless LAN Medium Access Control (MAC) and Physical Layer (LHY) Specification, IEEE Std. 802.11 IEEE—1999 edition.

[6] P. Karn, "MACA: A new channel access method for packet radio", in *Proceedings of 9th ARRL Computer Networking Conference*, London, Canada, Sept. 1990, pp. 134–140.

[7] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: A media access protocol for wireless LANs", in *Proceedings of ACM SIGCOMM'94*, London, UK, Sept. 1994, pp. 212–225.

[8] K. Sairam, N. Gunasekaran, and S. Reddy, "Bluetooth in wireless communication", *IEEE Communications Magazine*, June 2002, pp. 90–96.

[9] W. Ye and J. Heidemann, "Medium access control in wireless sensor networks", *Technical Report ISI-TR-580, USC/Information Sciences Institute*, Oct. 2003, pp. 1–8.

[10] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors", *Communications of the ACM*, vol. 43, no. 5, May 2000, pp. 51–58.

[11] M. Stemm and R. H. Katz, "Measuring and reducing energy consumption of network interfaces in hand-held devices", *IEICE Transactions on Communications*, vol. E80-B, no. 8, Aug. 1997, pp. 1125–1131.

[12] O. Kasten, Energy Consumption, Eldgenossische Technische Hochschule Zurich, available at http://www.inf.ethz.ch/~kasten/research/bathtub/energy_consumption.html

[13] W. Ye, J. Heidemann, and D. Estrin, "An energy efficient MAC protocol for wireless sensor networks", in *Proceedings of IEEE INFOCOM'02*, New York, NY, June 2002, pp. 1567–1576.

[14] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks", *IEEE/ACM Transactions on Networking*, vol. 12, no. 3, June 2004, pp. 493–506.

[15] R. Rajagopalan and P. Varshney, "Data-aggregation techniques in sensor networks: A survey", *IEEE Communications and Surveys and Tutorials*, vol. 8, no. 4, 4th Quarter 2006, pp. 48–63.

[16] P. Lin, C. Qiao, and X. Wang, "Medium access control with a dynamic duty cycle for sensor networks", in *Proceedings of 2004 IEEE Wireless Communications and Networking Conference (WCNC'04)*, Atlanta, GA, Mar. 2004, pp. 1534–1539.

[17] H. Pham and S. Jha, "An adaptive mobility-aware MAC protocol for sensor networks (MS-MAC)", in *Proceedings of 2004 International Conference on Mobile Ad Hoc and Sensor Systems (MASS'04)*, Fort Lauderdale, FL, Oct. 2004, pp. 558–560.

[18] G. Lu, B. Krishnamachari, and C. S. Raghavendra, "An adaptive energy-efficient and low-latency MAC for data gathering in wireless sensor networks", in *Proceedings of 18th International Parallel and Distributed Processing Symposium (IPDPS'04)*, Santa Fe, NM, Apr. 2004, pp. 224–231.

[19] R. Zheng, J. C. Hou, and L. Sha, "Asynchronous wakeup for ad hoc networks", in *Proceedings of ACM MobiHoc'03*, Minnapolis, MA, June 2003, pp. 35–45.

[20] K. Jamieson, H. Balakrishnan, and Y. C. Tay, "Sift: A MAC protocol for event-driven wireless sensor networks", LCS Technical Reports, May 1, 2003.

[21] T. V. Dam and K. Langendoen, "An adaptive energy efficient MAC protocol for wireless sensor networks", in *Proceedings of 1st ACM Conference on Embedded Networked Sensor Systems (SenSys'03)*, Los Angeles, CA, Nov. 2003, pp. 171–180.

[22] A. El-Hoiydi et al., "WiseMAC: An ultra low power MAC protocol for the WiseNET wireless sensor network", in *Proceedings of 1st ACM Conference on Embedded Networked Sensor Systems (SenSys'03)*, Los Angeles, CA, Nov. 2003, pp. 302–303.

[23] A. El-Hoiydi and J.-D. Decotignie, "WiseMAC: An ultra low power MAC protocol for the downlink of infrastructure wireless sensor networks", in *Proceedings of IEEE International Symposium on Computers and Communications (ISCC'04)*, Alexandria, Egypt, June–July 2004, pp. 244–251.

[24] Institute of Electrical and Electronics Engineers, Inc., "IEEE Std. 802.15.4–2003: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (LR-WPANs)", New York, IEEE Press. Oct. 2003.

[25] A. Woo and D. Culler, "A transmission control scheme for media access in sensor networks", in *Proceedings of 2001 ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'01)*, Rome, Italy, July 2001, pp. 221–235.

[26] V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves, "Energy-efficient, collision-free medium access control for wireless sensor networks", in *Proceedings of 1st ACM Conference on Embedded Networked Sensor Systems (SenSys'03)*, Los Angeles, CA, Nov. 2003, pp. 181–192.

[27] K. Sohrabi et al., "Protocols for self-organization of a wireless sensor network", *IEEE Personal Communications*, Oct. 2000, pp. 16–27.

[28] R. Kalidindi, L. Ray, R. Kannan, and S. Iyengar, "Distributed energy aware MAC layer protocol for wireless sensor networks", in *Proceedings of 2003 International Conference on Wireless Networks (ICWN'03)*, Las Vegas, Nevada, June 2003.

[29] S. Park, A. Savvides, and M. B. Srivastava, "SensorSim: A simulation framework for sensor networks", in *Proceedings of 5th International Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM'02)*, Boston, MA, Aug. 2000.

[30] M. Caccamo, L. Y. Zhang, L. Sha, and G. Buttazzo, "An implicit prioritized access protocol for wireless sensor networks", in *Proceedings of 23rd IEEE Real-Time Systems Symposium (RTSS'02)*, Austin, TX, Dec. 2002, pp. 39–48.

[31] T. W. Carley et al., "Contention-free periodic message scheduler medium access control in wireless sensor/actuator networks", in *Proceedings of 24th IEEE International Real-Time Systems Symposium (RTSS'03)*, Cancun, Mexico, Dec. 2003, pp. 298–307.

[32] B. Liu, N. Bulusu, H. Pham, and S. Jha, "CSMAC: A novel DS-CDMA based MAC protocol for wireless sensor networks", in *Proceedings of IEEE Global Telecommunications Conference (Globecom'04) Workshops*, Dallas, TX, Nov.–Dec. 2004, pp. 33–38.

[33] A. El-Hoiydi, "Spatial TDMA and CSMA with preamble sampling for low power ad-hoc wireless sensor networks", in *Proceedings of 2002 IEEE Symposium on Computers and Communications (ISCC'02)*, Taormina, Italy, July 2002, pp. 685–692.

[34] B. Mangione-Smith, "Low power communications protocols: paging and beyond," in *Proc. of 1995 IEEE Symposium on Low Power Electronics (ISLPED'95*), Dana Point, CA, Apr. 1995, pp. 8–11.

[35] I. Rhee, A. Warrier, M. Aia, and J. Min, "Z-MAC: a hybrid MAC for wireless sensor networks", in *Proceedings of 3rd ACM Conference on Embedded Networked Sensor Systems (SenSys'05*), San Diego, CA, Nov. 2005.

[36] I. Rhee, A. Warrier, and L. Xu, "Randomized dining philosophers to TDMA scheduling in wireless sensor networks", Technical Report, Computer Science Department, North Carolina State University, Raleigh, NC, 2004.

[37] S. Ramanathan, "A unified framework and algorithms for (T/F/C) DMA channel assignment in wireless networks", in *Proceedings of IEEE INFOCOM'07*, Anchorage, AK, May 2007, pp. 900–907.

[38] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks", in *Proceedings of 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys'04)*, Baltimore, MD, Nov. 2004, pp. 95–107.

[39] G.-S. Ahn, E. Miluzzo, A. T. Campbell, S. G. Hong, and F. Cuomo, "Funneling-MAC: A localized, sink-oriented MAC for boosting fidelity in sensor networks", in *Proceedings of ACM Conference on Embedded Networked Sensor Networks (SenSys'06)*, Boulder, Colorado, Nov. 2006, pp. 293–306.

[40] C.-Y. Wan, S. E. Eisenman, A. T. Campbell, and J. Crowcroft, "Siphon: Overload traffic management using multi-radio virtual sinks", in *Proceedings of 3rd ACM conference on Embedded Networked Sensor Systems (SenSys'05)*, Dan Diego, CA, Nov. 2005, pp. 116–129.

[41] TinyOS website: http://www.tinyos.net/

[42] C. Guo, L. C. Zhong, and J. M. Rabaey[, "Low power distributed MAC for Ad Hoc sensor radio networks", in *Proceedings of 2001 IEEE Global Telecommunications Conference (Globecom'01)*, San Antonio, TX, Nov. 2001, pp. 2944–2948.

[43] Y. C. Ray, K. Jamieson, and H. Balakrishnan, "Collision-minimizing CSMA and its applications to wireless sensor networks", *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 6, Aug. 2004, pp. 1084–1057.

[44] S. Cui et al., "Joint routing, MAC, and link layer optimization in sensor networks with energy constraints", in *Proceedings of 2005 IEEE International Conference on Communications (ICC'05)*, Seoul, Korea, May 2005, pp. 725–729.

[45] M. Zorzi, "A New contention-based MAC protocol for geographic forwarding in ad hoc and sensor networks", in *Proceedings of 2004 IEEE International Conference on Communications (ICC'04)*, Paris, France, June 2004, pp. 3481–3485.

[46] R. Rugin and G. Mazzini, "A simple and efficient MAC-routing integrated algorithm for sensor networks", in *Proceedings of 2004 IEEE International Conference on Communications (ICC'04)*, Paris, France, June 2004, pp. 3499–3503.

[47] Q. Ren and Q. Liang, "A contention-based energy-efficient MAC protocol for wireless sensor networks", in *Proceedings of 2006 IEEE Wireless Communications and Networking Conference (WCNC'06)*, Las Vegas, NV, Apr. 2006, pp. 1154–1159.

[48] M. C. Vuran and I. F. Akyildiz, "Spatial correlation-based collaborative medium access control in wireless sensor networks", *IEEE/ACM Transactions on Networking*, vol. 14, no. 2, Apr. 2006, pp. 316–329

[49] K. R. Chowdhury, N. Nandiraju, D. Cavalcanti, and D. P. Agrawal, "CMAC-A multichannel energy efficient MAC for wireless sensor networks", in *Proceedings of 2006 IEEE Wireless Communications and Networking Conference (WCNC'06)*, Las Vegas, NV, Apr. 2006, pp. 1172–1177.

[50] K. W. Tang and R. Kamoua, "Cayley pseudo-random (CPR) protocol: A novel MAC protocol for dense wireless sensor networks", in *Proceedings of 2007 IEEE Wireless Communications and Networking Conference (WCNC'07)*, Hong Kong, China, Mar. 2007, pp. 361–366.

[51] S. B. Eisenman and A. T. Campbell, "E-CSMA: Supporting Enhanced CSMA Performance in Experimental Sensor Networks using Per-neighbor Transmission Probability Thresholds", in *Proceedings of IEEE INFOCOM'07*, Anchorage, AK, May 2007, pp. 1208–1216.

[52] X. Shi and G. Stromberg, "SyncWUF: An ultra low-power MAC protocol for wireless sensor networks", *IEEE Transactions on Mobile Computing*, vol. 6, no. 1, Jan. 2007, pp. 115–125.

4

ROUTING AND DATA DISSEMINATION

Sajal K. Das

University of Texas at Arlington, USA

Habib M. Ammari

Hofstra University, USA

4.1 INTRODUCTION

Routing and data dissemination are an important issue in wireless sensor networks (WSNs). The essential function of a WSN is to monitor a phenomenon in a physical environment and report sensed data to a central node called a *sink*, where additional operations can be applied to the gathered data. This chapter focuses on routing and data dissemination in WSNs, and introduces the fundamental concepts related to routing and data dissemination, discusses the major issues and challenges in accomplishing this vital function, and surveys a variety of protocols for routing and data dissemination in WSNs. In particular, we present a taxonomy of routing and data dissemination protocols for WSNs based on different classification criteria, for example, location information, network layering and in-network processing, data centricity, multipath, network dynamics, quality-of-service requirements, and heterogeneity. The taxonomy is developed through an extensive analysis of a variety of routing and data dissemination protocols for WSNs. The objective of the taxonomy is threefold: (1) to provide a framework

Wireless Sensor Networks: A Networking Perspective, Edited by Jun Zheng and Abbas Jamalipour

in which routing and data dissemination protocols for WSNs can be examined and compared; (2) to show how the routing and data dissemination protocols can be categorized according to this taxonomy; and (3) to gain new insights into the routing and data dissemination protocols and thereby suggest avenues for future research. More specifically, the taxonomy comprises two types of classifications: one that classifies routing and data dissemination protocols with respect to sensor deployment, for example, sensor mobility, where sensors could be mobile or static, and one that classifies them with respect to trade-offs between different metrics specific to sensing applications, for example, energy efficiency, low delay, high data accuracy, and fault tolerance. Also shown are the benefits of sensor heterogeneity in routing and data dissemination for WSNs. This chapter complements other existing excellent surveys on WSNs [1,2], as well as those on routing and data dissemination protocols for WSNs [3–5].

The remainder of this chapter is organized as follows: Section 4.2 introduces the fundamentals and presents the major challenges in routing and data dissemination in WSNs. Section 4.3 overviews the ingredients of interest of the taxonomy for a variety of existing protocols. Section 4.4 surveys a sample of existing routing and data dissemination protocols in WSNs and classifies them with respect to the taxonomy. Section 4.5 concludes this chapter.

4.2 FUNDAMENTALS AND CHALLENGES

This section introduces related fundamentals and presents the major challenges in the design of routing and data dissemination protocols for WSNs.

4.2.1 Fundamentals

First, we define the terminologies that will be used in the subsequent sections. Then, we introduce a commonly used energy model [6] in most of the protocols for WSNs. Finally, we describe the *Voronoi* diagram [7], which has been widely used as a model of WSNs.

4.2.1.1 Terminology.

Sensing Range. The *sensing range* of a sensor (s_i) is a disk of radius (r_i), including its boundary, centered at ξ_i (the location of s_i) and defined by the point set, $D(\xi_i, r_i) = \{\xi \in \mathbf{IR}^2 : |\xi_i - \xi| \leq r\}$, where $|\xi_i - \xi|$ is the Euclidean distance between the locations ξ_i and ξ.

Transmission Range. The *transmission range* of a sensor s_i is a disk of radius (R_i), including its boundary, centered at ξ_i (the location of s_i), and defined by the point set, $D(R_i, \xi_i) = \{\xi \in \mathbf{IR}^2 : |\xi_i - \xi| \leq R_i\}$.

Neighbor Set. The *neighbor set* of a sensor (s_i) is given by $N(s_i) = \{s_j : |\xi_i - \xi_j| \leq R_j\}$, where R_i is the radius of the transmission range of s_i.

Coverage. Let A be an area of the field. A point $p \in A$ is said to be *covered* (or *sensed*) if and only if it belongs to the sensing range of at least one

sensor. The area A is said to be covered if and only if for every point $p \in A$ is covered.

Homogeneous versus Heterogeneous Network. A WSN is said to be *homogeneous* if all its sensors have the same storage, computation, communication, sensing, and energy capabilities. Otherwise, it is *heterogeneous*.

Communication Graph. A communication graph of a homogeneous (*heterogeneous*) WSN is an undirected (*directed*) graph, $G = (S, E)$, where S is a set of sensors and E is a set of (*directed*) edges between them such that for all $s_i, s_j \in S$, $(s_i, s_j) \in E$ if $|\xi_i - \xi_j| \leq R_i$.

Connectivity and Fault Tolerance. Let $G = (S, E)$ be a communication graph representing a network, where S is a set of sensors and E is a set of communication links between them such that for all $s_i, s_j \in S$, $(s_i, s_j) \in E$ if $|\xi_i - \xi_j| \leq R_i$. The *vertex-connectivity* (or *connectivity*) of G is equal to K if and only if G can be disconnected by the removal of at least K nodes. The *fault tolerance* of G is equal to K − 1.

Voronoi Diagram. Let $S = \{s_0, \ldots, s_{m-1}\}$ be a finite set of m sites in the plane. The *Voronoi diagram* [7] of S, denoted by $Vor(S)$, is a subdivision of the plane containing S into m *Voronoi* regions $VR(s_i)$, for $1 \leq i \leq m$, as shown in Fig. 4.1. Note that $VR(s_i)$ is possibly an unbounded open convex polygonal region that consists of all points closer to s_i than any other site in S. The edges of this region are called *Voronoi edges*. The $Vor(S)$ is the union of all *Voronoi* regions of sites $s_i \in S$. A WSN can be modeled by a *Voronoi* diagram with sites representing locations of sensors.

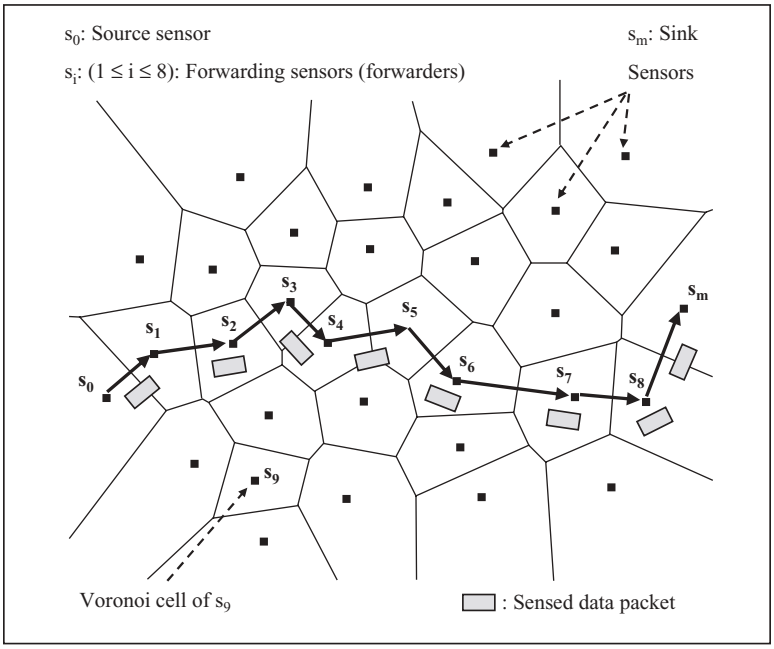


Fig. 4.1 The *Voronoi* diagram of a wireless sensor network.

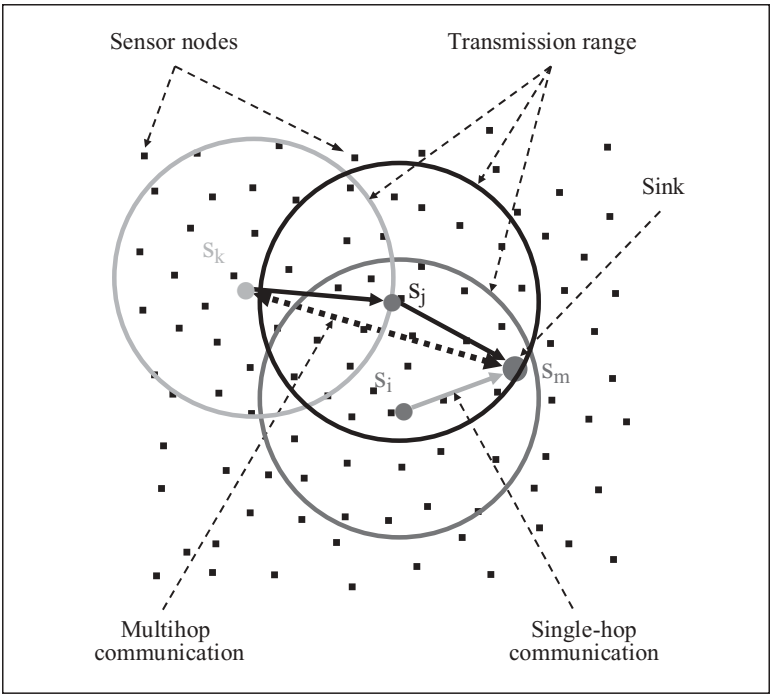


Fig. 4.2 Architecture of a wireless sensor network.

Figure 4.2 shows a network composed of a set of sensors randomly deployed in a square sensor field. The transmission range of a sensor is represented by a circle. When a sensor needs to communicate with another sensor that is inside its transmission range, the communication can be *single hop* (or direct). Otherwise, it must be *multihop* (or indirect) via other intermediate sensors that act as relays between the two communicating sensors. While the sensor s_i can communicate directly with the sink s_m, the sensor s_k can communicate with s_m only through other intermediate sensors, for example, s_j.

4.2.1.2 Energy Model. We assume that the energy consumption of the sensors is due to data transmission and reception. According to Ref. [6], the energy consumed in transmitting one message of size κ bits over a distance d called *transmission distance*, is given by $E_{tx}(d) = (\varepsilon d^{\alpha} + E_{elec})\kappa$, where E_{elec} represents the electronic energy, $\varepsilon \in \{\varepsilon_{fs}, \varepsilon_{mp}\}$ is the transmitter amplifier in the free space (ε_{fs}) or the multipath (ε_{mp}) model, and α is the path-loss exponent, $2 \le \alpha \le 4$. Also, the energy consumed in message reception is given by $E_{rx} = \kappa E_{elec}$. Hence, the total energy consumption when a sensor receives a message and forward it over a distance d is given by $E_{tot}(d) = (\varepsilon d^{\alpha} + 2E_{elec})\kappa$.

4.2.2 Challenges

The design of routing and data dissemination protocols for WSNs is challenging because of several network constraints. These constraints are imposed not only by the characteristics of individual sensors, the behavior of a network, and the nature of sensor fields, but also by the requirements of a sensing application in terms of some desirable metrics.

4.2.2.1 Sensor Characteristics. WSNs suffer from the limitations of several network resources, for example, energy, bandwidth, central processing unit (CPU), and storage [3], where energy is the most crucial resource because it determines the lifetime of a sensor. Also, energy poses a big challenge for network designers especially in hostile environments, for example, a battlefield, where it is impossible to access the sensors and recharge their batteries. Furthermore, when the energy of a sensor reaches a certain threshold, the sensor will become faulty and will not be able to function properly, which will have a major impact on the network performance. Therefore, algorithms designed for sensors should be as energy efficient as possible to extend their lifetime, and hence prolong the network lifetime while guaranteeing good performance overall.

Another challenge that faces the design of routing and data dissemination protocols is to manage the locations of the sensors. Most of the proposed protocols assume that the sensors either are equipped with *global positioning system* (GPS) receivers or use some localization technique [8] to learn about their locations. On one hand, although high sensor-location accuracy could be achieved, it is not cost effective that each sensor is equipped with a GPS receiver given that a WSN is highly dense in nature. On the other hand, the use of a localization technique may introduce certain inaccuracy in estimating the locations of the sensors.

4.2.2.2 Field Nature. As mentioned earlier, a sensor field may cause a difficulty not only in accessing the sensors for replacing and/or recharging their batteries, but also in their deployment. Thus, a deterministic sensor deployment strategy is not always possible. Such a strategy would help cover the field appropriately and minimize the total number of sensors required to achieve the specific requirements of sensing applications in terms of their expected type of coverage. In the real world, an application may require partial coverage, where only a certain percentage of the field is covered; full coverage, where the entire field is covered; or redundant coverage, where every location in the field is covered by multiple sensors simultaneously. In the case where sensors cannot be deployed deterministically because of the field nature, random deployment is the only remaining strategy. With random deployment, however, there is no guarantee that the coverage required by an application would be satisfied. There may be some areas that are not covered well or even not covered at all, which would lead to a problem known as *coverage hole*. Moreover, all deployed sensors are not guaranteed to be connected to each other or to the sink. This situation would lead to

another problem known as *connectivity hole*. These are two reasons that in most cases WSNs are designed with densely deployed sensors. Thus, the nature of a field has an influence on the network and this is a challenge for the designers and the investing party. As discussed later, one of the most widely used assumptions in the design of routing and data dissemination protocols is a high density of sensors deployed in a network. Although a highly densely deployed network needs more than necessary sensors, it helps guarantee network connectivity and achieve the coverage required by an application.

4.2.2.3 Network Characteristics. The topology of a network, which is defined by the sensors and the communication links between the sensors, changes frequently due to sensor addition and deletion. When a new sensor decides to join the network, the neighbor set of some sensors have to be updated. In many cases, it is necessary to add more sensors to maintain certain coverage properties of a sensor field and network connectivity. Similarly, when sensors deplete all their energy, they are considered faulty and no longer belong to the network. In this case, the neighbor sets of the faulty sensors should be updated as well. Also, in a mobile network, the network topology gets updated as sensors move in the sensor field. Consequently, any topology change in the network will have an influence on the communication paths (or routes) between the sensors. Therefore, routing and data dissemination paths should consider network topology dynamics due to limited energy and sensor mobility as well as increasing the size of the network to maintain specific application requirements in terms of coverage and connectivity. In particular, connectivity to the sink is very important. If the sensed data cannot reach the sink or there is no communication path between the source sensors (or data generators) and the sink, maintaining coverage would become not meaningful. Therefore, connectivity between all source sensors and the sink, either directly or indirectly, should be guaranteed for the proper operation of the network.

Another challenge is network scalability. In other words, routing and data dissemination protocols should be able to scale with the network size. Also, sensors may not necessarily have the same capabilities in terms of energy, processing, sensing, and particularly communication. Hence, communication links between sensors may not be symmetric, that is, a pair of sensors may not be able to have communication in both directions. This should be taken care of in the routing and data dissemination protocols.

4.2.2.4 Sensing Application Requirements. In most sensing applications, the sensed data should be as accurate as possible to assure better decision making by the sink. Moreover, the sensed data should reach the sink in a timely manner. Also, data redundancy is sometimes desirable in that it increases data accuracy. For example, in the intruder detection and tracking application, multiple sensors should be active in order to gather enough information about the intruder and track its motion accurately. Therefore, the routing and data dissemination protocols should guarantee data delivery and its accuracy so that the sink

can gather the required knowledge about the physical phenomenon on time. Furthermore, sensors may deplete their energy and become faulty. As discussed earlier, the sensor field may not be accessible and thus replacing those faulty sensors would be impossible. Hence, a network should tolerate the presence of faulty sensors and remain functional in spite of those faulty sensors. The degree of fault tolerance of the network depends on the underlying sensing application. Therefore, the routing and data dissemination protocols should also be fault tolerant.

4.3 TAXONOMY OF ROUTING AND DATA DISSEMINATION PROTOCOLS

This section presents a taxonomy of routing and data dissemination protocols for WSNs, as shown in Fig. 4.3. This taxonomy is based on several classification criteria, including location information, network layering and in-network processing, data centricity, path redundancy, network dynamics, quality-of-service (QoS) requirements, and network heterogeneity.

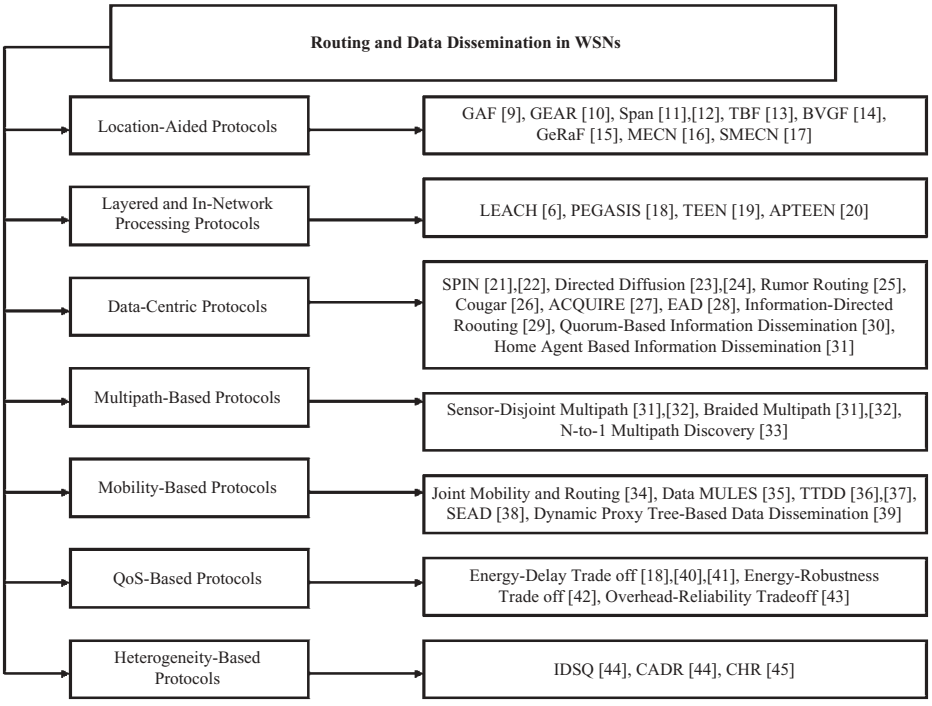


Fig. 4.3 Taxonomy of routing and data dissemination protocols for WSNs.

4.3.1 Location Information

The notion of physical location is an essential metric in several routing and data dissemination protocols in WSNs. Based on the location information of the sensors, these protocols can be short- or long range, depending on whether the distance between consecutive forwarders is minimum or maximum. The energy consumed in data forwarding depends on the distance over which data is transmitted. Note that location information was first used by routing protocols for mobile ad hoc networks (MANETs) [46]. While energy is not a metric in some MANET routing protocols, for example, location-aided routing (LAR) [46], it should be considered in the design of routing protocols for WSNs.

4.3.2 Network Layering and In-Network Processing

The architecture of a network can be flat in the sense that all sensors have the same role. In other words, all sensors forward their sensed data to the sink without necessarily passing through a particular node. A network is said to be *nonlayered* if all its sensors form only one group in which the sensors collaborate together to accomplish a common monitoring task. On the other hand, the sensors in a network can be grouped into *clusters*, each of which is managed by a specific sensor called a *cluster head*. These types of networks are said to be *layered*, where any sensed data should pass through one or more cluster heads before reaching the sink. These cluster heads are supposed to be powerful enough so that they can process the data they receive before sending them to the sink. All other sensors only need to sense the environment and send their data to the cluster heads for further processing. In some sensing applications, redundancy and correlation exist in the gathered data. Hence, it would be desirable to transmit only more representative data. For example, in monitoring the temperature of a room, the variation of the data within a given region is expected to be small. Thus, the sink is not interested in receiving all the temperature measures, but rather only some of them. This would reduce the communication overhead introduced by data forwarding significantly and improve the network performance. Also, the concept of layering makes a network more scalable and leads to more efficient usage of the energy of sensors, thus extending the network lifetime.

Extending network lifetime is an ultimate goal in the design of a WSN. Given that most energy of a sensor is mainly consumed in processing, sensing, and communication, an efficient design approach should take into account these three components of energy consumption. A question that network designers are mostly concerned about is *How can the lifetime of a network be extended?* To address this problem, several energy-efficient routing and data dissemination protocols have been proposed, which focus on how to forward the data until they reach the sink regardless of the type of data being transmitted from the source sensors to the sink. Among those protocols, one class does not update the data at intermediate sensors. That is, each intermediate sensor only acts as a pure data relay without altering any of the data it has received. Another class of protocols

introduces the concept of *in-network processing* to handle unnecessary redundancy and correlation contained in the sensed data. In many applications, the data sensed by the sensors have a certain amount of redundancy and correlation. It would be desirable if the sink can only receive relevant data for faster and better decision making. For this purpose, the sensed data should be processed at intermediate sensors before they reach the sink. The benefit of this in-network processing, such as data fusion, can be seen when vectorial data rather than scalar data are being transmitted. For example, in an application monitoring the temperature of a room, the sensed data are scalar (i.e., integer or real values). Hence, the cost of data communication is not very high, and the data fusion or aggregation is not costly as well. But sending continuously unnecessary redundant data will consume a huge amount of energy. If a sensing application has to send a large size of data, for example images, to the sink for further analysis and processing, it would consume a huge amount of energy. In this case, it would be more beneficial if those images sensed by different sensors could be aggregated and only a few of them would be sent. However, it is also true that processing those images for data fusion requires a considerable amount of energy. Moreover, there will be a delay due to the processing of those images. Therefore, there is a trade-off between data communication and fusion in this type of information intensive networks, where the sensed data are not scalar, but rather vectorial.

4.3.3 Data Centricity

A new communication paradigm has emerged in WSNs, which makes sensors capable of sensing, storage, processing, and computation to coordinate their sensing activities. This communication paradigm is *data centric* as all communications between sensors concern named data [47]. Because of its high density and mission nature, a WSN should be designed differently from IP-style networks in order to guarantee more efficient routing and data dissemination. Unlike general communication networks, a WSN is *task specific* in that a task to be performed by sensors is known at the time of sensor deployment.

4.3.4 Path Redundancy

In addition to their scalability and energy efficiency, the design of routing and data dissemination protocols for WSNs should also consider robustness, which means that a network remains functional in spite of the occurrence of sensor and link failures. Multipath routing is one technique that can make routing and data dissemination robust. This routing technique implies the existence of multiple paths between source and destination sensors (the sink is one of the destinations). These paths could be either disjoint or partially disjoint. Although maintaining alternate paths introduces some overhead and consumes more energy, multipath routing is an effective technique to improve robustness in the face of path failures that are caused by frequent topological changes due to unreliable wireless communication links and sensor failures. More specifically, multipath routing

helps recover from sensor and link failures and provide necessary resilience to the network at the cost of excessive redundancy.

4.3.5 Network Dynamics

As mentioned earlier, several factors, for example, limited energy and mobility, have an impact on the network topology. However, we consider energy a constraint, but not a goal. Any protocol designed for sensors should be as energy efficient as possible in order to address the constraint imposed by the limited energy of sensors. On the other hand, mobility is a desirable feature that can be used to tackle some problems in WSNs, for example, coverage hole and connectivity hole. In this taxonomy, we focus on mobility because it is the main source of network dynamics [48]. At the end, the sensed data will be transmitted over some established paths between the source sensors and the sink. The existence of these paths depends on whether the sensors are static or mobile. Thus, we classify the routing and data dissemination protocols based on whether a network is static or dynamic.

In a static network, there is no mobility at all; that is, both the sensors and the sink remain in their fixed locations during their collaborative mission of monitoring a physical environment. Therefore, there is not much overhead required to maintain routes between the sensors and the sink and between the sensors themselves. Actually, the locations of the sensors and the sink can be learned at the beginning of their monitoring task by exchanging some control messages. The neighbors of a given sensor are always the same unless a new sensor has joined the network or an existing sensor has left the network either by its will or because its entire energy is depleted.

In a mobile network, either the sensors are moving or the sink is moving. In any case, the routes between the sensors and the sink change frequently. A route that is currently valid might not be valid later on. This route instability would introduce an additional overhead for finding valid routes for data transmission and forwarding. As a result, the network may suffer from a delay in relaying the sensed data to the sink. In some scenarios, for example, the data MULES based architecture in [35], both the sensors and the sink are static, but there are other nodes acting as *relays*, which move in the sensor field to collect the sensed data from the source sensors and report them to the sink.

It is worth noting that whether mobility needs to be considered depends on the sensing application. For example, if we are interested in controlling the temperature, humidity, sound, or light in a room, there is no need to have mobile sensors or a mobile sink. However, for monitoring a moving object, it is necessary to introduce some degree of mobility to the network for an efficient tracking of the object. It has been proved that the use of mobile relays helps increase the lifetime of a WSN [49].

4.3.6 Quality of Service Requirements

Sensing applications may have different requirements, which can be expressed in terms of some QoS metrics, such as delay, reliability, and fault tolerance.

For example, time-critical applications have delay bounds to meet. For such applications, the sensed data must reach the sink within a certain time. Also, a desired property of sensing applications is fault tolerance by which it is meant that a network should remain functional in the event of sensor failures. Another desired property is reliability by which it is meant that the sensed data should be received by the sink as correctly as possible to ensure accurate decision making by the sink. Both fault tolerance and reliability require the deployment of more than necessary sensors so that the network can continue to function properly and deliver accurate sensed data to the sink despite some sensor failures. However, the use of redundant sensors yields additional energy consumption. Therefore, routing and data dissemination protocols should be designed in a way to trade-off between energy, fault tolerance, reliability, and delay. Recall that energy is a constraint that should be met by any routing and data dissemination protocol in order to guarantee an efficient usage of the amount of energy available at each sensor.

4.3.7 Network Heterogeneity

Most of the protocols designed for WSNs assume that the sensors have the same capabilities in terms of storage, processing, sensing, and communication. The resulting network is said to be *homogeneous*, where all communication links between the sensors are symmetric, that is, a given pair of neighboring sensors can directly communicate with each other. In these types of networks, a pair of sensors would have the same lifetime if they have the same energy consumption rate. Some sensing applications, however, use sensors with different capabilities and accordingly the resulting network is said to be *heterogeneous*. In the real world, the assumption of homogeneous sensors may not be practical because sensing applications may require heterogeneous sensors in terms of their sensing and communication capabilities in order to enhance network reliability and extend network lifetime [50]. Also, even if the sensors are equipped with identical hardware, they may not always have the same communication and sensing models. In fact, at the manufacturing stage, there is no guarantee that two sensors using the same platform have exactly the same physical properties. This taxonomy focuses on heterogeneity at the designing stage, when sensors are designed to have nonidentical capabilities to meet the specific needs of sensing applications.

4.4 OVERVIEW OF ROUTING AND DATA DISSEMINATION PROTOCOLS

Traditional routing protocols have several shortcomings when applied to WSNs, which are mainly due to the energy-constrained nature of such networks. For example, *flooding* is a technique in which a given node broadcasts data and control packets that it has received to the rest of the nodes in the network. This

process repeats until the destination node is reached. Note that this technique does not take into account the energy constraint imposed by WSNs. As a result, when used for data routing in WSNs, it leads to the following two problems, namely, *implosion* and *overlap* [4]. Given that flooding is a blind technique, duplicated packets may keep circulate in the network, and hence sensors will receive those duplicated packets, causing an implosion problem. Also, when two sensors sense the same region and broadcast their sensed data at the same time, their neighbors will receive duplicated packets. To overcome the shortcomings of flooding, another technique known as *gossiping* can be applied. In *gossiping*, upon receiving a packet, a sensor would select randomly one of its neighbors and send the packet to it. The same process repeats until all sensors receive this packet. Using gossiping, a given sensor would receive only one copy of a packet being sent. While gossiping tackles the implosion problem, there is a significant delay for a packet to reach all sensors in a network.

This section surveys a sample of existing routing and data dissemination protocols for WSNs and classifies them with respect to the taxonomy introduced in Section 4.3.

4.4.1 Location-Aided Protocols

There are several location-based routing protocols proposed for WSNs, for example, *greedy other adaptive face routing* (GOAFR) [51], *greedy perimeter stateless routing* (GPSR) [52], *most forward with fixed radius* (MFR) [53], *geographic distance routing* (GEDIR) [54], to name a few. However, those protocols were initially designed for MANETs without any energy considerations. They do not consider the specific requirements of WSNs, particularly their limited energy resources, and therefore cannot be used for such networks.

This section presents a sample of location-aware routing and data dissemination protocols proposed for WSNs, as well as some of those proposed for MANETs with energy consideration. Both types of protocols do not use flooding due to the implosion and overlap problems it can cause.

4.4.1.1 Geographic Adaptive Fidelity. *Geographical adaptive fidelity* (GAF) [9] is a routing protocol proposed for MANETs. Although it was proposed for MANETs, it favors energy conservation and thus can be used for WSNs. Hence, we will use the word *sensor* instead of *node*, which is used in GAF. The design of GAF is motivated by the results of the previous studies based on an energy model that considers energy consumption due to the reception and transmission of packets as well as idle (or listening) time when the radio of a sensor is on to detect the presence of incoming packets. These studies [48,55] showed that battery-powered nodes consume energy not only when receiving or sending packets, but also when listening or idle. Therefore, it is not enough to optimize energy consumption by only reducing packet transmission and reception. In addition, the radio should also be turned off. GAF is based on this mechanism; that is, turning off unnecessary sensors while keeping a constant

level of *routing fidelity* (or uninterrupted connectivity between communicating sensors).

GAF divides a sensor field into grid squares and every sensor uses its location information, which can be provided by GPS or other location systems [8,56,57], to associate itself with a particular grid in which it resides. This kind of association is exploited by GAF to identify the sensors that are equivalent from the perspective of packet forwarding. The size of the grid square is chosen in a way such that sensors within the same grid are equivalent with regard to routing and that sensors in adjacent grids can communicate with each other. Thus, equivalent sensors can coordinate with each other to determine an energy-efficient schedule of their activities, which specifies when and for how long the sensors stay awake or sleep.

As shown in Fig. 4.4, the state transition diagram of GAF has three states, namely, *discovery, active*, and *sleeping*. When a sensor enters the *sleeping* state, it turns off its radio for energy savings. In the *discovery* state, a sensor exchanges discovery messages to learn about other sensors in the same grid. Even in the *active* state, a sensor periodically broadcasts its discovery message to inform equivalent sensors about its state. The time spent in each of these states can be tuned by the application depending on several factors, such as its needs and sensor mobility. GAF aims to maximize the network lifetime by reaching a state where each grid has only one active sensor based on sensor ranking rules. The ranking of sensors is based on their residual energy levels. Thus, a sensor with a higher rank will be able to handle routing within their corresponding grids. For example, a sensor in the *active* state has a higher rank than a sensor in the *discovery* state. A sensor with longer expected lifetime has a higher rank.

In order to have all the sensors running for as long as possible without penalizing any one of them, GAF uses a load balancing strategy in which a sensor remains in the *active* state for only some time before switching to the *sleeping*

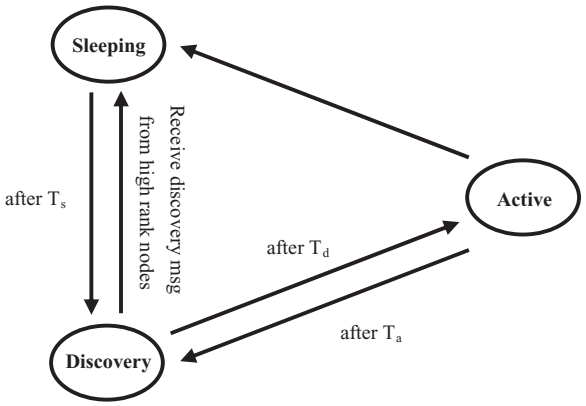


Fig. 4.4 State transition diagram of GAF.

state. This would give a chance to other sensors within the same grid to become active and handle routing. The rationale behind this rule is that sensors switching to the *discovery* state would have less residual energy than their neighbors in the *sleeping* state, where they conserve their energy. Note that sensor mobility may leave a grid with no active sensors at all. To address this problem, a sensor estimates the time it expects to leave its grid based on its GPS receiver and advertises it in its discovery message. Upon receiving this discovery message, the sensor's neighbors adjust their sleeping time so that their grid has always one active sensor to handle routing within that grid.

4.4.1.2 Geographic and Energy-Aware Routing. Yu et al. [10] proposed an energy-efficient routing protocol, called *geographic and energy aware routing* (GEAR), for routing queries to target regions in a sensor field. In GEAR, the sensors are supposed to have localization hardware equipped, for example, a GPS unit or a localization system [8] so that they know their current positions. Furthermore, the sensors are aware of their residual energy as well as the locations and residual energy of each of their neighbors. GEAR uses energy aware heuristics that are based on geographical information to select sensors to route a packet toward its destination region. Then, GEAR uses a recursive geographic forwarding algorithm to disseminate the packet inside the target region. The goal behind using energy aware data dissemination with geographical information is to help make energy-efficient routing decisions. GEAR is motivated by the fact that in several location-aware systems, such as WSNs, it is useful to disseminate information to a geographical region. For example, a user could interrogate the sensing application about the temperature in a given region within some time interval. To receive an answer, this query should be disseminated to the sensors located in the target region. The location information added to the query will help it to be sent directly to its ultimate destination area rather than flooding it in the entire sensor field.

For each of its neighbors, a sensor maintains two variables, called *estimated cost* and *learned cost*. The estimated cost of a neighbor N_i depends on the consumed energy at N_i and the distance between N_i and the centroid of the target region. If a sensor does not have the learned cost for its neighbor N_i, it computes the estimated cost as a default value for the learned cost. A sensor selects the neighbor $N_{\min}$ with minimum learned cost in order to balance the energy consumption across all its neighbors. After the selection process, a sensor sets its own learned cost to the sum of the learned cost of $N_{\min}$ and the cost of transmitting a packet to $N_{\min}$.

GEAR has mainly two phases, namely, forwarding a packet toward its destination region (phase 1) and disseminating the packet within the destination region (phase 2). During phase 1, a sensor selects a neighbor that is closer to the destination region than itself to act as the next forwarder. Otherwise, all its neighbors are farther away from the destination region than itself, and hence there is a void region between the sensor holding a packet and the target region (see Fig. 4.5). In this case, GEAR selects one of those neighbors whose learned

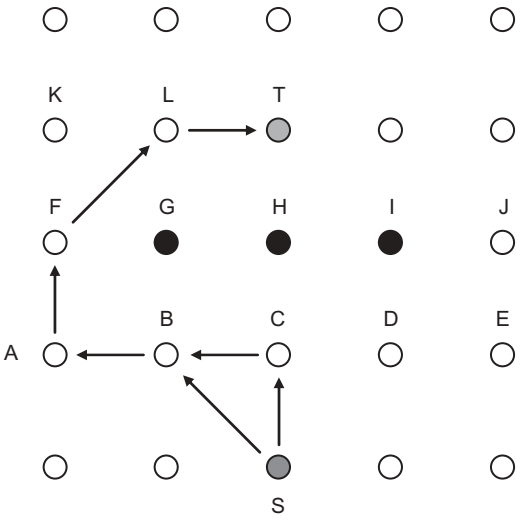


Fig. 4.5 Routing while avoiding holes.

cost is the minimum. In phase 2, GEAR uses a recursive geographic forwarding algorithm to disseminate the packet within the target region. In this case, the target region is split into four subregions and the current sensor creates four copies of the packet to be unicast to those subregions. This procedure of splitting and forwarding repeats until the current node finds itself to be the only one inside this subregion, and hence the packet is dropped. When the sensors are sparsely deployed, GEAR uses restricted flooding, which is more energy efficient than recursive geographic forwarding. In this case, a sensor sends only one broadcast message to all its neighbors.

Note that GEAR can also be classified as a data-centric data dissemination protocol, which will be discussed in Section 4.4.8. In GEAR, a query is expressed in terms of the name of the data, for example, temperature, not the sensor identifiers.

4.4.1.3 Coordination of Power Saving with Routing. *Coordination of power saving with routing* (Span) [11,12] is a routing protocol proposed for MANETs, but can be applied to WSNs as its goal is to reduce energy consumption of the nodes. In the context of WSNs, we also use a sensor to refer to a node in Span. Span is motivated by the fact that the wireless network interface of a device is often the single largest consumer of power. Hence, it would be better to turn the radio off during idle time. Although Span does not require that sensors know their location information, it runs well with a geographic forwarding protocol. Span helps sensors to join a forwarding backbone topology as coordinators that will forward packets on behalf of other sensors between any source and destination. According to Span, a sensor is eligible for becoming a coordinator if any pair of its neighbors cannot communicate either directly or via at most two

coordinators. To give a chance to other noncoordinator sensors to become coordinators, a coordinator withdraws if any pair of its neighbors can reach each other via some neighbors even if those neighbors are not currently coordinators. When used with a geographic forwarding protocol, Span's election rule requires each sensor to advertise its status (i.e., coordinator or noncoordinator), its neighbors, and its coordinators. Furthermore, when it receives a packet, a coordinator forwards the packet to a neighboring coordinator if any, which is the closest to the destination or to a noncoordinator that is closer to the destination.

4.4.1.4 Trajectory-Based Forwarding. *Trajectory-based forwarding* (TBF) [13] is another protocol that can be used in dense ad hoc networks, for example, WSNs. TBF requires a sufficiently dense network and the presence of a coordinate system, for example, a GPS, so that the sensors can position themselves and estimate distance to their neighbors. This new paradigm benefits from the characteristics of source-based routing, for example, *dynamic source routing* (DSR) and *Cartesian routing*. The source specifies the trajectory in a packet, but does not explicitly indicate the path on a hop-by-hop basis. The trajectory is expressed in the parametric form $\{X(t), Y(t)\}$ that is suitable for the purpose of forwarding, where t is a parameter, for example, the distance along the curve, indicated by the source. In such a representation of the trajectory, the parameter of the curve is a proxy for the hop count and represents a metric that measures the forward progress along the path. Moreover, a trajectory can be composed of several simple trajectories, each of which can be considered as a segment that can be represented by an appropriate interval of the parameter associated with the trajectory. Furthermore, the forwarding nodes are selected based on their proximity to the trajectory, not to the destination. Based on the location information of its neighbors, a forwarding sensor makes a greedy decision to determine the next hop that is the closest to the trajectory fixed by the source sensor. In fact, the specification and evaluation of a trajectory has certain cost in terms of complexity.

As can be seen, route maintenance in TBF is unaffected by sensor mobility given that a source route is a trajectory that does not include the names of the forwarding sensors. In order to increase the reliability and capacity of the network, it is also possible to implement multipath routing in TBF where an alternate path is just another trajectory. Note that it is not necessary to specify a final destination in the trajectory. This helps implement some networking functions, for example, flooding, discovery, and network management. To flood a packet in the network, a source sensor could specify the directions and lengths of radial lines in order to provide a satisfactory coverage of the sensors in the network. In addition to radial lines, trajectories could be specified as H-trees or fractals to achieve the required coverage. As mentioned earlier, the trajectory specification and evaluation has some complexity and it would be beneficial to the trade-off between this complexity and the required coverage. TBF can also be used for resource discovery. For example, a server could advertise its location along an arbitrary trajectory (or line) and a client could send its query along another trajectory that

will eventually intersect the server's trajectory. The sensor located at the intersection point will then inform the client about the position of the server. The client can then transmit its request along another trajectory to the server. Another interesting application of TBF is securing the perimeter of the network. For this purpose, a source sensor could specify a trajectory as a boomerang, where a packet including a challenge response token is sent along this trajectory. Any sensor that answers properly can be considered as authenticated.

4.4.1.5 Bounded Voronoi Greedy Forwarding. Bounded *Voronoi greedy forwarding* (BVGF) [14] is another location-based routing protocol for WSNs, which uses the concept of *Voronoi* diagram [5]. Therefore, the sensors should be aware of their geographical positions. In BVGF, a network is modeled by a *Voronoi* diagram with sites representing the locations of sensors. In this type of greedy geographic routing, a sensor will always forward a packet to the neighbor that has the shortest distance to the destination. The sensors eligible for acting as the next hops are the ones whose *Voronoi* regions are traversed by the segment line joining the source and the destination. The BVGF protocol chooses as the next hop the neighbor that has the shortest Euclidean distance to the destination among all eligible neighbors. It does not help the sensors deplete their battery power uniformly. Each sensor actually has only one next hop to forward its data to the sink. Therefore, any data dissemination path between a source sensor and the sink will always have the same chain of the next hops, which will severely suffer from battery power depletion. BVGF does not consider energy as a metric.

4.4.1.6 Geographic Random Forwarding. A relay sensor in data forwarding toward the sink is usually referred to as a sender. Zorzi and Rao [15] proposed a new data transmission protocol, called *geographic random forwarding* (GeRaF), which uses geographic routing where a sensor acting as relay is not known *a priori* by a sender. As will be discussed below, there is no guarantee that a sender will always be able to forward the message toward its ultimate destination, that is, the sink. This is the reason that GeRaF is said to be *best-effort* forwarding. GeRaF assumes that all sensors are aware of their physical locations, as well as that of the sink. Although GeRaF integrates a geographical routing algorithm and an awake–sleep scheduling algorithm, the sensors are not required to keep track of the locations of their neighbors and their awake–sleep schedules.

When a source sensor has sensed data to send to the sink, it first checks whether the channel is free in order to avoid collisions. If the channel remains idle for some period of time, the source sensor broadcasts a request-to-send (RTS) message to all of its active (or listening) neighbors. This message includes the location of the source and that of the sink. Note that the coverage area facing the sink, called *forwarding area*, is split into a set of N_p regions of different priorities such that all points in a region with a higher priority are closer to the sink than any point in a region with a lower priority. When active neighboring sensors

receive the RTS message, they assess their priorities based on their locations and that of the sink. The source sensor waits for a CTS message from one of the sensors located in the highest priority region. For GeRaF, the best relay sensor is the one closest to the sink, thus making the largest advancement of the data packet toward the sink. In case that the source does not receive the CTS message, it implies that the highest priority region is empty. Hence, it sends out another RTS polling sensors in the second highest priority region. This process continues until the source receives the CTS message, which means that a relay sensor has been found. Then, the source sends its data packet to the selected relay sensor, which in turn replies back with an ACK message. The relay sensor will act in the same way as the source sensor in order to find the second relay sensor. The same procedure repeats until the sink receives the sensed data packet originated from the source sensor. It may happen that the sending sensor (source sensor or relay sensor) does not receive any CTS message after sending N_p RTS messages. This means that the neighbors of the sending sensor are not active. In this case, the sending sensor backs off for some time and retries later. After a certain number of attempts, the sending sensor either finds a relay sensor or discards the data packet if the maximum allowed number of attempts is reached.

A question that can arise is *Which sensors are allowed to reply to a given RTS message*? If a sensor is in the highest priority region and receives an RTS message, it replies immediately with a CTS message. In general, a sensor in the ith priority region replies with a CTS message only if it receives the ith RTS message and the first (i-1) RTS messages were not answered. In case that there are multiple CTS messages answering a given RTS message, some collision resolution algorithm is triggered by sending a special RTS message, which results in selecting only one relay sensor among all sensors located in the priority region being considered by the original RTS message.

The interested reader can also refer to [58] for a detailed description of the collision avoidance protocol, as well as a detailed analysis of the energy and latency performance of GeRaF.

4.4.1.7 Minimum Energy Communication Network. In all previous discussed protocols, static sensors are assumed. In Ref. [16], a location-based protocol for achieving minimum energy for randomly deployed ad hoc networks was proposed. This protocol, called *minimum energy communication network* (MECN), can be used for WSNs. MECN attempts to set up and maintain a minimum energy network with mobile sensors. The motivation of MECN is based on the key premise that maximizing the total battery lifetime of a network requires minimizing the energy consumption of the entire network. MECN is a self-reconfiguring protocol that maintains network connectivity in spite of sensor mobility. It computes an optimal spanning tree rooted at the sink, called *minimum power topology*, which contains only the minimum power paths from each sensor to the sink. It is based on the positions of sensors on the plane and consists of two main phases, namely, *enclosure graph construction* and *cost distribution*.

For a stationary network, in the first phase (i.e., *enclosure graph construction*), MECN constructs a sparse graph, called an *enclosure graph*, based on the immediate locality of the sensors. For this purpose, a sensor first determines its relay region with respect to each of the sensors it can communicate with directly. A relay region contains all the points where relaying a message to any of these points through an intermediate sensor (or relay sensor) is always more energy efficient than sending the message to them directly. This determines a region around a sensor, called an *enclosure region*, beyond which it is not energy efficient to search for more neighbors. The sensors located in the enclosure region of a sensor are its neighbors to which the sensor will maintain communication links for energy-efficient transmission. As can be seen, the enclosure region of a sensor is bounded by the intersection of all relay regions with respect to all the sensors it can interact with directly. An enclosure graph is a directed graph that includes all the sensors as its vertex set and whose edge set is the union of all edges between the sensors and the neighbors located in their enclosure regions. In other words, a sensor will not consider the sensors located in its relay regions as potential candidate forwarders of its sensed data to the sink. Figure 4.6 shows the relay region of a transmit-relay pair of sensors. Only the sensors in its immediate neighborhood (i.e., *enclosure region*) will be the only potential candidate forwarders. Furthermore, this graph is sparse and strongly connected.

In the second phase (i.e., *cost distribution*), nonoptimal links of the enclosure graph are simply eliminated and the resulting graph is a *minimum power topology*. This graph has a directed path from each sensor to the sink and consumes the least total power among all graphs having directed paths from each sensor to the sink. To find optimal links on the enclosure graph, the Bellman–Ford shortest path algorithm is applied using the power consumption as the cost metric.

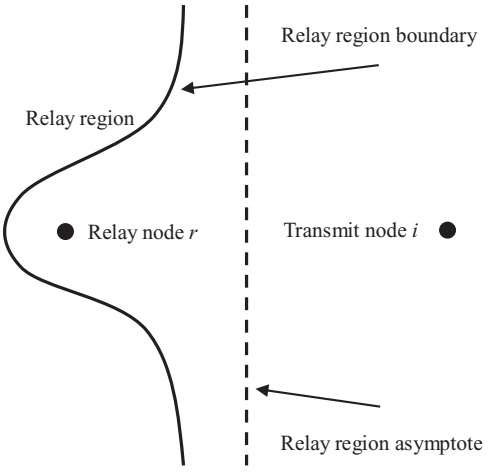


Fig. 4.6 Relay region of the transmit-relay pair (*i, r*).

Each sensor broadcasts its cost to its neighbors, where the cost of a node is the minimum power required for this sensor to establish a directed path to the sink. Specifically, a sensor first calculates

$$C_{i,n} = \text{Cost}(n) + P_{\text{tx}}(i,n) + P_{\text{rx}}(n),$$

where $P_{\text{tx}}(i, n)$ is the power needed to transmit from i to n, $P_{\text{rx}}(n)$ is the power required for n to receive from any transmitting sensor, and Cost(n) is the cost computed by the neighbor n (i.e., n is a neighbor of the sensor i). Then, the cost of sensor i is calculated as

$$\text{Cost}(i) = \min_{n \in N(i)} C_{i,n},$$

where $N(i)$ is the neighbor set of i based on the concept of the enclosure region. Figure 4.7 illustrates this concept. At the end of this phase, every sensor will have computed the minimum-cost neighbor link, which will be used to send its sensed data to the sink. All the links form the minimum power topology.

MECN applies also to synchronous mobile WSNs, in which a GPS can be used to provide absolute time information for synchronization. For energy-efficiency purposes, a sensor can move back and forth between a *listen* mode to listen for any change in the network topology due to sensor mobility and a *sleep* mode to conserve its energy. The *cycle period*, defined as the time between two successive wakeups, is very critical. A short cycle period introduces much overhead, and hence wastes energy due to computing costs that change very slowly, while a long cycle period will not reflect the exact costs to the sink. Therefore, a tradeoff between these two scenarios is required.

While MECN is a self-reconfiguring protocol, and hence is fault tolerant (in the case of mobile networks), it suffers from a severe battery depletion problem

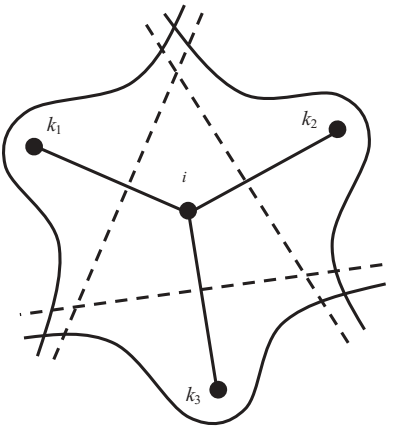


Fig. 4.7 Enclosure of sensor i.

when applied to static networks. The MECN does not take into consideration the available energy at each sensor, and hence the optimal cost links are static. In other words, a sensor will always use the same neighbor to transmit or forward sensed data to the sink. For this reason, this neighbor would die very quickly and the network thus becomes disconnected. To address this problem, the enclosure graph and thus the minimum power topology should be dynamic based on the residual energy of the sensors.

4.4.1.8 Small Minimum-Energy Communication Network. *Small minimum-energy communication network* (SMECN) is a protocol proposed to improve MECN discussed in Section 4.4.1.7. In this protocol, Li and Halpern [17] characterized a minimal graph with regard to the *minimum energy property*. This property implies that for any pair of sensors in a graph associated with a network, there is a minimum energy-efficient path between them; that is, a path that has the smallest cost in terms of energy consumption over all possible paths between this pair of sensors. Their characterization of a graph with respect to the minimum energy property is intuitive. A path between a pair of sensors u and v whose length is >1 is preferable to a direct edge (u, v) between them if the total power consumption of this path is less than that of the direct edge. A graph that satisfies this property is said to be *minimal* and is denoted by $G_{\min}$. In this case, the edge (u, v) is said to be k-redundant if the length of the energy-efficient path between u and v is equal to k.

The SMECN protocol attempts to construct a graph (i.e., a communication network) that includes $G_{\min}$ as a subgraph. For this purpose, SMECN needs to find a subset E_2 that contains all edges in the original graph that are not 2-redundant. Every sensor discovers its immediate neighbors by broadcasting a neighbor discovery message using some initial power that is updated incrementally. Specifically, the immediate neighbors of a given sensor are computed analytically. Then, a sensor starts broadcasting a neighbor discovery message with some initial power p and checks whether the theoretical set of immediate neighbors is a subset of the set of sensors that replied to that neighbor discovery message. If this is the case, the sensor will use the corresponding power p to communicate with its immediate neighbors. Otherwise, it increments p and rebroadcasts its neighbor discovery message. Li and Halpern [16] showed that E_2 is a subgraph of the enclosure graph produced by the MECN protocol.

4.4.2 Layered and In-Network Processing-Based Protocols

Traditional (or flat) routing and data dissemination protocols for WSNs may not be optimal in terms of energy consumption. Clustering is an energy-efficient communication protocol that can be used by the sensors to report their sensed data to the sink. In this section, we describe a sample of layered protocols in which a network is composed of several *clumps* (or *clusters*) of sensors. Each clump is managed by a special node, called *cluster head*, which is responsible for coordinating the data transmission activities of all sensors in its clump. All sensors

in a cluster communicate with the cluster head that acts as a local sink, which in turn transmits the sensed data to the global sink. Note that the transmission distance over which the sensors send their data to their cluster head is smaller compared to their respective distances to the global sink. Since a network is characterized by its limited wireless channel bandwidth, it would be beneficial if the amount of data transmitted to the sink can be reduced. To achieve this goal, a local collaboration between the sensors in a cluster is required in order to reduce bandwidth demands.

4.4.2.1 Low-Energy Adaptive Clustering Hierarchy. To overcome the shortcomings of conventional routing and data dissemination protocols, which run on top of nonlayered or flat network architectures, a clustering-based protocol, called *low-energy adaptive clustering hierarchy* (LEACH), was proposed in Ref. [6]. LEACH is based on an *aggregation* (or *fusion*) technique that combines or aggregates the original data into a smaller size of data that carry only meaningful information of all individual sensors. For this purpose, LEACH divides a network into several clusters of sensors, which are constructed by using localized coordination and control not only to reduce the amount of data that are transmitted to the sink, but also to make routing and data dissemination more scalable and robust. Given that the energy dissipation of the sensors depends on the distance and the data size to be transmitted, LEACH attempts to transmit data over short distances and reduce the number of transmission and reception operations.

In LEACH, the cluster heads are not selected in a static manner; otherwise, they will drain their energy and die quickly. Instead, LEACH uses a randomized rotation of the high-energy cluster-head position in order to give a chance to all sensors to act as cluster heads and avoid the battery depletion of an individual sensor. The operation of LEACH is divided into *rounds*, each of which has mainly two phases: a setup phase to organize the network into clusters and a steady-state phase for data transmission to the sink. Cluster heads use CSMA MAC protocol to advertise their status. Thus, all noncluster-head sensors must keep their receivers on during the setup phase in order to hear the advertisements sent by the cluster heads. These cluster heads are selected with some probability by themselves and broadcast their statuses to the other sensors in the network. The decision for a sensor to become a cluster head is made independently without any negotiation with the other sensors. Specifically, a sensor decides to become a cluster head based on the desired percentage P of cluster heads (determined *a priori*), the current round, and the set of sensors that have not become cluster heads in the past $1/P$ rounds. If the number of cluster heads is $<T(n)$, a sensor n becomes a cluster head for the current round, where $T(n)$ is a threshold given by

$$T(n) = \begin{cases} \dfrac{P}{1 - P(r \bmod 1/P)} & \text{if } n \in G, \\ 0 & \text{otherwise.} \end{cases}$$

The sensors that are cluster heads in round 0 cannot be a cluster for the next $1/P$-1 rounds. At round 0, each sensor has probability P to become a cluster head. Among all advertised cluster heads, a sensor selects the closest one that will incur minimum energy communication and then informs its cluster head about its decision to join the cluster using CSMA MAC protocol. Similarly, cluster heads should keep their receivers on to hear these join messages. Once the network is divided into clusters, a cluster head computes a TDMA schedule for its sensors specifying when a sensor in the cluster is allowed to send its data. Thus, a sensor will turn its radio on only when it is authorized to transmit according to the schedule established by its cluster head, thus yielding significant energy savings. Furthermore, LEACH enables data fusion in each cluster by aggregating the data in order to reduce the total amount of data before sending them to the sink. In another word, once a cluster head gathers all the data from its sensors, it aggregates them and transmits the aggregated data to the sink.

LEACH can be viewed as a hybrid approach using short- and long-range based data forwarding. The sensors within a cluster transmit their sensed data over short distances, whereas cluster heads communicate directly with the sink. While LEACH helps the sensors within their cluster dissipate their energy slowly, the cluster heads consume a larger amount of energy when they are located farther away from the sink. Sending directly to the sink is the main problem with LEACH. A better approach is to allow multihop data transmission to the sink through other cluster heads. In this case, a cluster head does not have to update the aggregated data from other cluster heads, but only forward them toward the sink. Moreover, a decision for a sensor to become a cluster head should consider the residual energy of that sensor. Table 4.1 shows that LEACH outperforms all other protocols, including a data transmission protocol called *Direct*, in which the sensors transmit directly to the sink, the minimum transmission energy (MTE) protocol, in which each data packet must go through n low-energy transmissions and n receptions, and the static clustering protocol, in which all cluster-heads are selected at once.

4.4.2.2 Power-Efficient Gathering in Sensor Information Systems. In LEACH, all cluster heads should broadcast their advertisements to all sensors in the network. In addition, all of them should transmit their aggregated data to the sink in each round. To improve LEACH, another protocol, called *power-efficient gathering in sensor information systems* (PEGASIS) [18], was proposed, which allows only one cluster head to transmit to the sink in each round. Moreover, a sensor has to transmit to its local neighbors in the data fusion phase instead of sending directly to its cluster head as in the case of LEACH. In PEGASIS, sensors are organized in a way to form a chain, which can be performed either by the sensors themselves using a greedy algorithm or by the sink, which has to broadcast the chain to all sensors in the network. The construction phase assumes that all the sensors have global knowledge about the network, particularly, the positions of the sensors, and uses a greedy approach. Specifically, it starts with the furthest sensor to the sink to guarantee that sensors farther away from the

TABLE 4.1 Lifetime with Different Sensor Initial Energy

Energy (Joule/Sensor)	Protocol	Round First Sensor Dies	Round Last Sensor Dies
0.25	Direct	55	117
	MTE	5	221
	Static clustering	41	67
	LEACH	394	665
0.5	Direct	109	234
	MTE	8	429
	Static clustering	80	110
	LEACH	932	1312
1	Direct	217	468
	MTE	15	843
	Static clustering	106	240
	LEACH	1848	2608

sink have close neighbors. When a sensor fails or dies due to low battery power, the chain is constructed using the same greedy approach by bypassing the failed sensor.

The chain has two end sensors and in each data fusion phase only one leader (i.e., a sensor responsible for transmitting the fused data to the sink) will transmit the fused data to the sink. Any other intermediate sensor will fuse the data received from its neighbor with its own data and transmit the fused data to its neighbor located closer to the sink than itself so that the fused data get forwarded toward the sink. Note that all sensors will participate in the data fusion except the end sensors unless they are leaders, which will transmit the fused data to the sink. The data fusion phase in each round requires that a leader send a control token to the end sensors of the chain, where the data transmission should start. At the end, the leader receives two fused data from both sides of the chain, fuses them with its own data, and transmits the final fused data to the sink.

Table 4.2 shows a comparison between LEACH, PEGASIS, and *Direct*, in which all sensors transmit directly to the sink. The results in the first half of the table correspond to a sensor field of size 50 m × 50 m, while those in the second half correspond to a 100 m × 100 m sensor field. We vary the initial energy of individual sensors (0.25, 0.5, and 1) and the percentage of sensors that die (1–100%). Note that the number of rounds increases with the initial energy of the sensors. Also, PEGASIS outperforms both LEACH and *Direct* for both sizes of the network.

4.4.2.3 Threshold Sensitive Energy Efficient Sensor Network Protocol.

A sensing application can be designed in a way where the sensors either sense and transmit their sensed data periodically to the sink or react immediately to any sudden change in the value of the sensed attribute. While in the first scenario, a network is said to be *proactive*, the second scenario corresponds to a *reactive*

TABLE 4.2 Comparison between PEGASIS, LEACH, and Direct

Energy (Joule/Sensor)	Protocol	1%	20%	50%	100%
0.25	Direct	54	62	76	117
	LEACH	402	480	523	635
	PEGASIS	788	1004	1041	1096
0.5	Direct	108	124	152	235
	LEACH	803	962	1036	1208
	PEGASIS	1578	2011	2082	2192
1.0	Direct	215	248	304	471
	LEACH	1610	1921	2055	2351
	PEGASIS	3159	4023	4165	4379
0.25	Direct	14	16	20	30
	LEACH	166	204	232	308
	PEGASIS	335	624	684	779
0.5	Direct	28	32	40	61
	LEACH	339	408	461	576
	PEGASIS	675	1250	1362	1544
1.0	Direct	56	64	80	122
	LEACH	690	812	911	1077
	PEGASIS	1346	2497	2720	3076

network. For time-critical applications, a reactive network is more suitable than a proactive network. In order to trade-off between energy efficiency, data accuracy, and response time dynamically, a communication protocol, called *threshold sensitive energy efficient sensor network protocol* (TEEN), was proposed in Ref. [33]. TEEN uses hierarchical clustering, which groups sensors into clusters with each led by a cluster head. The sensors within a cluster report their sensed data to their cluster head. The cluster head sends aggregated data to higher level cluster heads until the data reach the sink. Thus, TEEN is a clustering communication protocol that targets a reactive network and enables cluster heads to impose a constraint on when the sensors should report their sensed data. Each cluster head broadcasts to its members a value, called *hard threshold* (H_T), for the sensed attribute, beyond which a sensor should turn its transmitter on to report its sensed data to its cluster head. In addition, a cluster head broadcasts another value, called *soft threshold* (S_T), which indicates a small change in the value of the sensed attribute, which triggers a sensor to turn on its transmitter and send its sensed data to the cluster head.

The sensors within a cluster can be scheduled using TDMA or CDMA in order to avoid collisions in a cluster. However, this will introduce delay when reporting time-critical data to the sink. At the beginning of a sensing task, a sensor transmits its sensed data when its value is higher than the hard threshold specified by its cluster head. Moreover, a sensor stores the current value *SV* of

the sensed attribute. Based on the values of the hard and soft thresholds, a sensor transmits its sensed data only if its value is higher than H_T and the difference between this current value and the previously stored value *SV* is $\geq S_T$. When a sensor sends its sensed data, it updates *SV* with the current value of its sensed attribute.

As can be seen, the hard threshold helps the sensors to transmit only significant information while the soft threshold further reduces the number of transmissions for sensed data. Thus, the sensors will send only sensed data that are of interest to the end user based on the hard threshold value and the change with respect to the previously reported data, thus yielding more energy savings. However, both values of the hard and soft thresholds have an impact on TEEN. These values should be set very carefully to keep the sensors responsive by reporting sensed data to the sink. It may happen that for some value of the hard threshold, the sensors are not able to transmit at all. In this case, the cluster heads will not receive any data at all from their members. Also, the changes between the values of the currently sensed data and the previously reported one may not reach the soft threshold. In this case, the sensors will not report to their cluster heads. In either case, the cluster heads cannot know whether their members have died because of low energy or the above-mentioned conditions on the values of the hard and soft thresholds are not satisfied. Therefore, TEEN is not suitable for sensing applications which require sensors to report their data on a regular basis.

The simulation results reported in [19] show that TEEN performs much better than LEACH. Furthermore, TEEN using a soft threshold outperforms TEEN with a hard threshold as expected.

4.4.2.4 Adaptive Periodic TEEN. To overcome the above-mentioned shortcomings of TEEN, a new protocol, called *adaptive periodic TEEN* (APTEEN), which combines the best features of both TEEN (time-critical data) and LEACH (periodic sensed data transmission) was proposed in Ref. [34]. Therefore, APTEEN is a hybrid clustering-based routing protocol that allows the sensors to send their sensed data periodically and react to any sudden change in the value of the sensed attribute by reporting the corresponding values to their cluster heads. Similar to TEEN, APTEEN uses the concept of hierarchical clustering for energy efficient communication between source sensors and the sink. After the clusters are formed, a cluster head broadcasts the sensed attributes of interest, the hard and soft thresholds, a TDMA schedule that assigns a slot to each sensor, and a maximum time interval between two successive reports sent by a sensor, called *count time* (T_c). This count time is used when sensors have to report their sensed data periodically to the sink.

Contrary to other existing query routing protocols, APTEEN can handle three types of queries. Specifically, it can answer *historical queries* by extracting historical data associated with the events that occurred in the past. It can also respond to *one-time queries* that give a snapshot view of the network. Moreover, it can reply to *persistent queries* that allow monitoring the network within a time interval with respect to some sensed attributes.

It has been shown through extensive simulations that APTEEN guarantees lower energy dissipation and a larger number of sensors alive [20]. Compared to LEACH and TEEN, the performance of APTEEN in terms of energy consumption and network lifetime lies between those of LEACH and TEEN. While in LEACH sensors transmit their sensed data continuously to the sink, in APTEEN sensors transmit their sensed data based on the threshold values.

4.4.3 Data-Centric Protocols

In traditional routing protocols for WSNs, also known as *address-centric* protocols, when the sink sends out a query for collecting data, each source sensor that has the appropriate data responds by sending its data to the sink independently of all other sensors. Data-centric protocols differ from address-centric protocols in the manner that the data is sent from source sensors to the sink. In *data-centric* protocols, when the source sensors send their data to the sink, intermediate sensors can perform some form of aggregation on the data originating from multiple source sensors and send the aggregated data toward the sink. This process can result in energy savings because less transmissions are required to send the data from the sources to the sink. This section reviews a sample of data-centric routing and dissemination protocols for WSNs.

4.4.3.1 Sensor Protocols for Information via Negotiation. For some applications, for example, intruder detection, the design of a surveillance network that provides a way to replicate complete and global views of the physical environment across the entire network is necessary. This type of network helps disseminate critical pieces of information to all sensors in the network so that they become aware of any critical event that may occur. Moreover, it enhances the fault tolerance of the network and helps the network to continue to function normally in the presence of sensor failures. Thus, disseminating individual sensor observations to all sensors in the network should be performed as energy efficient as possible, where all sensors are considered as potential sinks. In light of this, a family of adaptive protocols, called *sensor protocols for information via negotiation* (SPIN) [21,22], is suggested. The SPIN protocols were designed in a way to improve classic flooding protocols and overcome the problems they may cause, for example, implosion and overlap, which were discussed earlier. In addition, flooding, when used, makes the sensors blindly consume their available resources. The SPIN protocols are resource aware and resource adaptive. The sensors running the SPIN protocols are able to compute the energy consumption required to compute, send, and receive data over the network. Thus, they can make informed decisions for efficient use of their own resources.

In data dissemination using SPIN, sensors are resource aware in the sense that they make their decisions based on their missions, the information they have about the environment and other sensors, and their computational, communication, and energy resources. Specifically, the SPIN protocols are based on two key mechanisms: *negotiation* and *resource adaptation*. In terms of negotiation, SPIN

enables the sensors to negotiate with each other before any data dissemination can occur in order to avoid injecting nonuseful and redundant information in the network. Therefore, the data observed by the sensors need to be named. For this purpose, SPIN uses *meta-data* as the descriptors of the data that the sensors want to disseminate. The notion of meta-data avoids the occurrence of overlap given that the sensors can name the interesting portion of the data they want to get. Note that the size of the meta-data should definitely be less than that of the corresponding sensor data. Also, meta-data have application-specific formats and introduce additional costs for their storage, retrieval, and management. For example, sensors covering disjoint areas may use their unique IDs as meta-data. Thus, the meta-data x stands for "all sensed data of sensor x". As can be observed, this negotiation process tackles the problems of implosion and overlap introduced by classic flooding protocols. Contrary to the flooding technique, each sensor is aware of its resource consumption with the help of its own *resource manager* that is probed by the application before any data processing or transmission. This helps the sensors to monitor and adapt to any change in their own energy resources. For example, based on its available energy, a sensor may not want to participate in data forwarding on behalf of other sensors, thereby extending its lifetime and the operating lifetime of the network. Thus, both negotiation and resource adaptation effectively address the above-mentioned problems caused by classic flooding protocols.

The family of SPIN protocols is motivated by the fact that routing decisions are best made using not only the knowledge about the network topology, but also the application data layout and the available resources at each sensor. Hence, it is interesting to integrate the concepts of data naming and routing in WSNs. There are two protocols in the SPIN family: SPIN-1 (or SPIN-PP) and SPIN-2 (or SPIN-EC) [22]. While SPIN-1 uses a negotiation mechanism to reduce the energy consumption of the sensors, SPIN-2 uses a resource-aware mechanism for more energy savings. Both protocols allow the sensors to exchange information about their sensed data, thus helping them to obtain the data they are interested in. SPIN-1 is a three-stage handshake protocol by which the sensors can disseminate their data. This protocol applies for those networks using point-to-point transmission media (or point-to-point networks), in which two sensors can communicate exclusively with each other without interfering with other sensors. For a sensor to communicate with all its neighbors, it has to communicate with each of them separately. Hence, the energy and time required for a sensor to communicate with n neighbors is n times the energy and time required for a sensor to communicate with one neighbor. SPIN-1 consists of two stages: advertisement, request, and data transmission. In the *advertisement* stage, a sensor advertises its data by sending an advertisement (ADV) message containing the meta-data of the data it wants to share with other sensors. In the *request* stage, if a sensor is interested in the actual data being advertised and does not possess all of the advertised data, it sends back a request (REQ) message to the source of the ADV message listing all of the data it wants to acquire. In the *data*

transmission stage, the source of the ADV message retrieves the requested data and replies to the REQ message with a DATA message that contains the actual data with a meta-data header. The sensor that has received the requested data could advertise them to its neighbors.

SPIN-BC [22] improves SPIN-PP by using one-to-many communication instead of many one-to-one communications. It is a three-stage handshake protocol for broadcast transmission media, where the sensors in a network communicate with each other using a single shared channel. A sensor can communicate with all of its neighbors using only one ADV message. However, if a sensor wants to advertise or receive data, it has to wait for the channel to be free before sending its messages.

SPIN-2 differs from SPIN-1 in that it takes into account the residual energy of sensors. If the sensors have plenty of energy, SPIN-2 is identical to SPIN-1, and hence has the same three stages. However, when a sensor has low residual energy, it controls its participation in a data dissemination process. Specifically, if a sensor finds out that it can participate in a data dissemination process without having its residual energy to become below some low-energy threshold, it sends out an REQ message for any advertised data it is interested in. Otherwise, the sensor will not send a REQ message for the advertised data because it does not have energy to send its request, receive the data, and process them. Note that a SPIN-2 sensor consumes energy when it receives ADV and REQ messages even when its residual energy is below a certain low-energy threshold. SPIN-2 does not prevent this energy consumption from occurring.

While the family of SPIN protocols applies to lossless networks, it can be slightly updated to apply to lossy or mobile networks. In addition to the conservative approach of SPIN-2, periodic readvertisement of ADV messages are necessary to address lost ADV messages. Similarly, a sensor can retransmit REQ messages to compensate for DATA messages that do not arrive because of some time out. The SPIN-RL [22] is an updated version of SPIN-BC that helps the sensors reliably disseminate their data in a lossy network.

4.4.3.2 Directed Diffusion. The main requirements of WSNs are energy efficiency, scalability, and robustness with conserving energy resources being the most crucial one. *Directed diffusion* [23,24] is a data-centric paradigm for sensor query dissemination and processing that meets the above requirements. Examples of such queries can have the following forms: *How many pedestrians observed in the geographical area X? or In what direction vehicle Y in region Z is moving?* Sensors can be used to detect pedestrians or vehicle movements, collaborate with each other to disambiguate the locations of pedestrians or the movement direction of vehicles, and report their, possibly aggregated, results to the sink. For example, sensors within region Z may coordinate to select the best estimate of the vehicle movement direction and send it back to the query originator. Specifically, a sensor transforms the sampled waveforms generated by a vehicle into *event descriptions* that include the sensor's location information, the direction of

vehicle movement, a codebook value (or event code) for the vehicle, a timestamp, the intensity of the signal, and a degree of confidence in its estimation. In directed diffusion, sensors name their generated data by attribute-value pairs. If a sensor wants to receive data, it sends *interests* for named data. When sent by a source sensor, the data can be cached or transformed by intermediate sensors, which in turn may initiate interests based on the data that were previously cached. In addition, the data sent by the source sensor may be aggregated by intermediate sensors before being forwarded to their destination. More importantly, interests, data dissemination, and data aggregation occur in directed diffusion in a *localized* manner via exchanging messages between neighboring sensors.

Directed diffusion has several key elements: *data naming, interests and gradients, data propagation*, and *reinforcement*. As mentioned earlier, a sensing task can be described by a list of attribute-value pairs. For example, consider the following task: In every I ms for the next T seconds, send me a location estimate of any four-legged animal in sub-region R of the sensor field. This animal tracking task can be described by the following data:

Type = four-legged animal // detect animal location
Interval = 20 ms //send back events every 20 ms
Duration = 10 seconds // … for the next 10 seconds
Rect = [–100, 100, 200, 400] // from sensors within rectangle

This task description is called an *interest*. We assume that an interest is injected into the network at the sink. Hence, the sink periodically broadcasts an interest for each active task specifying a low data rate, which has the following form:

Type = four-legged animalInterval = 1 s
Rect = [–100, 100, 200, 400]
Timestamp = 01:20:40
ExpiresAt = 01:30:40

To ensure reliable interest transmission, an interest has a timestamp that allows the sink to refresh the interest by resending it with a monotonically increasing timestamp value. The refresh rate of the sink is selected in a way to trade-off overhead for robustness to lost interests. An interest cache is maintained at each sensor and contains the following fields:

- Timestamp of the last received matching interest.
- A gradient per neighbor indicating the data rate requested by the neighbor (1 event per second for the above interest) and a direction in which to send events.
- Approximate lifetime of the interest (10 s for the above interest).

These fields all together help a sensor maintain only unique and active interests in its interest cache. When a sensor receives an interest, it may resend it to some subset of its neighbors or even rebroadcast it to all of its neighbors in case it does not have information about the sensors that satisfy the interest. This allows interests to be *diffused* throughout the network. For energy savings purposes, geographic routing can be used in order to limit the topological scope for interest diffusion. Also, a sensor can use cached data to direct interests to particular sensors instead of broadcasting them to all of its neighbors.

A response (or event description) to the above interest is also named and could have the following form:

Type = four-legged animal // type of animal seen
Instance = elephant // instance of this type
Location = [145,222] // sensor location
Intensity = 0.6 // signal amplitude measure
Confidence = 0.85 // confidence in the match
Timestamp = 01:20:40 // event generation time

A sensor will consult its interest cache to decide to which neighbors it has to unicast this event description (or *data* message) using the highest data rate over all its outgoing gradients. Upon receiving a data message, a sensor may drop it if it does not find a matching interest in its cache. Otherwise, it checks the *data cache*, which contains the data messages seen recently, corresponding to the matching interest entry. As a result, this data message can be cached in the data cache and resent to the sensor's neighbors, if it is not already in the data cache. Otherwise, this data message is simply dropped. If the data rate specified in all gradients is higher than that of incoming events, the receiving sensor will simply send the received data message to the corresponding neighbors. Otherwise, the receiving sensor may *downconvert* to the specified data rate for the neighbors with a lower requested data rate.

At the beginning of the directed diffusion process, the sink specifies a low data rate for incoming events. After that, the sink can *reinforce* one particular sensor to send events with a higher data rate by resending the original interest message with a smaller interval. Likewise, if a neighboring sensor receives this interest message and finds that the sender's interest has a higher data rate than before, and this data rate is higher than that of any existing gradient, it will *reinforce* one or more of its neighbors. Note that it may happen that the sink reinforces one neighboring sensor A, but then receives a new event from neighboring sensor B that sends the event before A does. Directed diffusion allows the sink to *negatively reinforce* the path through A by explicitly degrading the path through A by resending the interest message with a lower data rate. Upon receiving this interest message, sensor A degrades its gradient toward the sink. Additionally, sensor A will negatively reinforce its neighbors sending at a data rate higher than those of all its gradients.

4.4.3.3 Rumor Routing. In data-centric networks, it is necessary to design efficient protocols for routing queries to the sensors that have detected the events of interest. A query can be either a request for obtaining relevant data or an order to collect data. When a coordinate system is not available or the phenomenon of interest is not geographically correlated, geographic routing cannot apply. In this case, the use of a flooding technique can be a justifiable alternative, which depends on the ratio of the number of events to the number of queries. In fact, query flooding is a useful scheme when the number of events is very high compared to the number of queries. Otherwise, event flooding is an efficient scheme. In this context, a data-centric routing protocol, called *rumor routing*, can be used as a logical compromise between query flooding and event flooding schemes [25]. Rumor routing is an efficient protocol if the number of queries is between the two intersection points of the curve of rumor routing with those of query flooding and event flooding.

Rumor routing is based on the concept of *agent*, which is a long-lived packet that traverses a network and informs each sensor it encounters about the events that it has learned during its network traverse. It seems more reasonable to only allow the sensors that have observed events to create agents that will carry relevant information to the other sensors in the network. An agent will travel the network for a certain number of hops and then die. Each sensor, including the agent, maintains an event list that has event-distance pairs, where every entry in the list contains the event and the actual distance in the number of hops to that event from the currently visited sensor. Therefore, when the agent encounters a sensor on its path, it synchronizes its event list with that of the sensor it has encountered. Also, the sensors that hear the agent update their event lists according to that of the agent in order to maintain the shortest paths to the events that occur in the network. A sensor can also send a query for a particular event. This query can be either sent along a route to the sensor that witnessed the event or forwarded in a random direction in the network.

A query stays in the network as long as its time-to-live has not expired or has not reached the sensor that has observed the target event. The agent maintains a list of recently seen sensors. Hence, when it visits a sensor, it adds the sensor's neighbors to the list and picks its next hop that is not already in the list. This will maximize the chance of creating fairly straight dissemination paths that yield better results than random forwarding. Similarly, when a query is sent in a random direction, it acts exactly in the same way as an agent. In case that the query originator finds out that the query has not reached its destination, it simply gives up or floods it in the network to guarantee its delivery.

4.4.3.4 The Cougar Approach. A design approach that is based on preprogrammed sensors and a central entity at which data is aggregated and stored for future querying and analysis suffers from two major problems. First, it is impossible for a user to change the behavior of the system on the fly, for example, to change the sensing task of the sensors. Second, the main goal of battery power conservation in the design of a network cannot be fully met when the sensed data

communicated by the source sensors to the sink is huge and correlated. In this case, the network does not benefit from in-network processing that can reduce the amount of information to be fed in the network, thus saving significant energy. To address these problems, a database approach to tasking sensor networks, called *Cougar*, was proposed in Ref. [26]. The Cougar approach provides a user and application programs with declarative queries of the sensed data generated by the source sensors. These queries are suitable for WSNs in that they abstract the user from knowing the execution plan of its queries. In other words, the user does not know which sensors are contacted, how sensed data are processed to compute the queries, and how final results are sent to the user. The Cougar approach uses a *query layer* where every sensor is associated with a *query proxy* that lies between the network layer and application layer of the sensor. This query proxy provides higher level services through queries that can be issued from a gateway node. Furthermore, the Cougar approach employs in-network processing to reduce the total energy consumption and enhance the network lifetime. Specifically, the query proxy is responsible for in-network processing, for example, aggregating records or eliminating irrelevant records, when it processes user queries. For some sensing applications, the sensed data of individual sensors are either not important or inaccurate. Hence, it is more beneficial if a set of sensed data could be aggregated or fused into a single one that is more representative and thus significant to the user.

There are a few challenges facing any database approach, including the Cougar approach. First, a network can be viewed as a huge distributed database system, where every sensor possesses a subset of data. Hence, current distributed management approaches cannot be applied directly, but need to be modified accordingly. In particular, data in a WSN could change very frequently depending on the frequency of occurrence of new events. Therefore, if a sensing application cares about the current state of the network, sensed data have to be updated frequently. One way to keep query results up-to-date is to have long-running queries that recompute query results periodically. However, all sensed data do not have the same change rate. For example, in an object detection application, data values change rapidly and thus become outdated quickly. In a temperature sensing application, data values change slowly over time. Thus, the queries issued to the sensors in these two applications do not need to have the same execution rate. For the temperature sensing application, which requires only approximate results, previous values have to be cached and the query update rate has to be lowered in order to save energy. Moreover, *uncertainty* is an inherent property of data measurement. Actually, the sensed data, for example, temperature, generated by a sensor, is an approximate as there are always errors introduced by the sensor. Another source of errors in the sensed data is noise. If we assume that this error follows some distribution, we can compute the probability that the actual value of temperature T lies in the range $[T_1, T_2]$. Hence, the user should be provided the possibility to query the network about all temperatures whose actual values lie in the range $[T_1, T_2]$ with a given probability.

Given that local computation is much cheaper than communication, the Cougar approach is in favor of in-network processing. In other words, adding computation to a sensor and reducing communication in the network will help save a significant amount of energy. In addition to the query proxy at each sensor, there is a query optimizer in the gateway, which generates distributed query processing plans once it receives queries from the outside. A query plan is disseminated to the relevant sensors after an exact computation plan at each sensor and the data flows between those sensors are specified. The processing of a query starts once the sensor behaviors are synchronized. A query plan can also specify a leader election algorithm of this query. For the temperature monitoring application, a query could be *Notify an administrator if the temperature in the office is greater than a user-defined threshold*. In this case, the query optimizer will produce a query plan for each of the relevant sensors, indicating how to elect a leader that will compute the average temperature. These query plans are disseminated to the query proxies of the relevant sensors to control their execution and the submission of their results toward the leader sensor. The leader can be elected randomly among all the relevant sensors. However, for an energy-efficiency purpose, this leader can be the sensor with the highest residual energy. It can also be energy efficient to select a leader based on its physical location. The leader selection should consider the cost of data communication from the source sensors to the leader and the data delivery cost from the leader to the gateway. Note that both costs depend on the position of the leader. There are also two computation plans, one for the leader and the other for each of the nonleader sensors. Each of the nonleader sensors has a sensor scan that reads sensor values periodically and sends them to an in-network aggregator. The aggregator combines its local data with the partially aggregated data received from other sensors and sends the results toward the leader. The query plan for the leader is to compute the average of the partially aggregated results and compare it to the user-defined threshold. If the computed average is greater than the threshold, the leader will send the average temperature value to the gateway node.

4.4.3.5 Active Query Forwarding. Another data-centric querying mechanism, called *active query forwarding in sensor networks* (ACQUIRE), was proposed for querying named data [27]. ACQUIRE is a data-centric routing mechanism for providing superior query optimization to answer specific types of queries, called *one-shot complex queries for replicated data*. In ACQUIRE, a query (i.e., interest for named data) consists of several subqueries for which several simple responses are provided by several relevant sensors. Each subquery is answered based on the currently stored data at its relevant sensor.

ACQUIRE allows a sensor to inject an active query in a network following either a random or a specified trajectory until the query gets answered by some sensors on the path using a localized update mechanism. When a sensor receives a query, it triggers an on-demand update to obtain information from all neighbors located within a look-ahead of d hops. This query is resolved incrementally as it traverses the network from one sensor to another. When it is completely

solved, it returns a completed response to the query originator. The value of d has an impact on the trade-off between the collected information, which helps reduce the length of the overall trajectory of the query, and the cost introduced by collecting the information. ACQUIRE is a mechanism for extracting data from relevant sensors in order to respond to complex, one-shot, and nonaggregate queries for replicated data. It differs from traditional flooding-based query techniques. In those techniques, several copies of a query are flooded into the network by either a querying sensor or the sink (or simply *querier*). Any sensor with relevant data will respond to this query. Note that for a one-shot query whose answer is a single value, flooding will be very costly and dominates the querying costs. Actually, the energy costs will be higher even when aggregation is used due to duplicate responses. Moreover, the querying and answering stages are separated. Unlike such query techniques, ACQUIRE allows the querier to inject a complex query into the network to be forwarded stepwise through a sequence of sensors. A sensor receiving this active query, known as *active sensor*, will partially answer it based on its local knowledge. This knowledge can be either its fresh updates/data or the updates received from all sensors within a look-ahead of d hops as a result of a request originated from the active sensor to all sensors within d hops if its data is obsolete. This request is forwarded hop by hop and each sensor receiving the request will forward its information to the active sensor. After that, the active sensor chooses its next sensor from those d hops to which it forwards the partially resolved query (or remaining query). The next active sensor selection can be randomly done by executing a random walk or by finding an appropriate sensor that would guarantee the maximum possible further resolution of the query. Following the same process, the query becomes smaller and smaller as it is forwarded from one active sensor to another until its complete resolution; that is, becomes a *completed response*. Then, the completed response is sent back to the original querier using the reverse path or the shortest path.

4.4.3.6 Energy-Aware Data-Centric Routing. One way to save the energy of sensors is to turn their radios off from time to time. However, this cannot be done to all sensors in the network. Otherwise, a sensor cannot be used as a relay to forward data on behalf of other sensors. *Energy-aware data-centric routing* (EAD) is a novel distributed routing protocol, which builds a virtual backbone composed of active sensors that are responsible for in-network data processing and traffic relaying [28]. In this protocol, a network is represented by a broadcast tree spanning all sensors in the network and rooted at the gateway, in which all leaf nodes' radios are turned off while all other nodes correspond to active sensors forming the backbone and thus their radios are turned on. Specifically, EAD attempts to construct a broadcast tree that approximates an optimal spanning tree with a maximum number of leaves, thus reducing the size of the backbone formed by active sensors. This approach is energy aware and helps extend the network lifetime. The gateway plays the role of a *data sink* or *event sink*, whereas each sensor acts as a *data source* or *event source*.

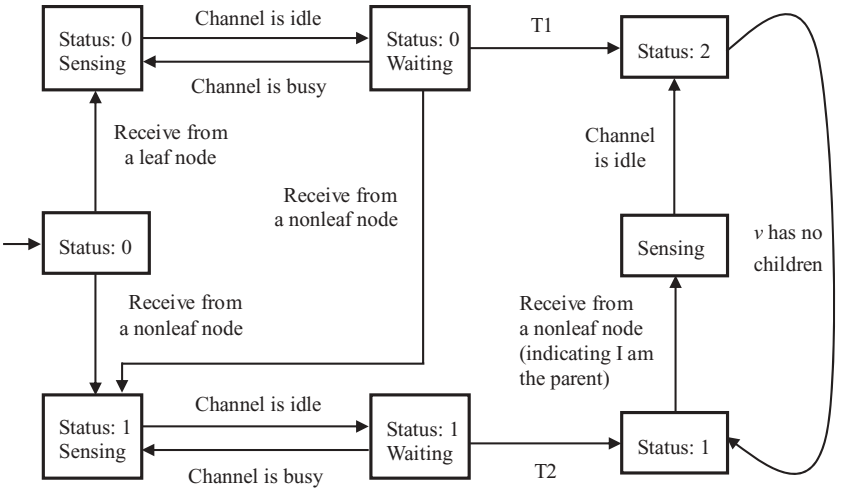


Fig. 4.8 State transition of an EAD sensor.

EAD enables the sensors and the sink to exchange messages with four fields. The state diagram of EAD is given in Fig. 4.8. If s is the sender of a message, the fields are *type* of s (sensor or sink) indicating the status of s, which can be *undefined, nonleaf node*, or *leaf node*; the *level* of s, which indicates the number of hops from s to the sink; the *parent* of s, which indicates the next hop of s in the path to the sink; and the *power* of s, which indicates the residual energy of s. A sensor switches to a specific state based on the messages it receives from other sensors. Initially, the states of all sensors are *undefined*. The sink starts by broadcasting a message $msg(2, 0, NULL, \infty)$ with ∞ indicating the sink with infinite energy. When a sensor v receives a message $msg(2, level_u, parent_u, E_u)$ from a sensor u, it becomes a leaf node, senses the channel until it is idle, and waits for time T_2^v. Then, node v broadcasts $msg(1, level_u + 1, u, E_v)$ if the channel is still idle. If sensor v receives $msg(1, level_u, parent_u, E_u)$ from sensor u, it senses the channel until it is idle, and waits for time T_1^v. Then, sensor v becomes a nonleaf node by broadcasting $msg(2, level_u + 1, u, E_v)$ if the channel is still idle. If a nonleaf node v receives $msg(2, level_w, v, E_w)$ from node w, in which it indicates that node v is its parent, after the channel is idle, node v becomes a nonleaf node by broadcasting $msg(2, level_v, parent_v, E_v)$. This message exchange takes place until the status of each sensor is either a *leaf node* or a *nonleaf node*. Obviously, any sensor with *undefined* status becomes a leaf node if it finds out that it has no children from the messages it has received.

EAD is based on the residual energy of the sensors, and provides a neighboring broadcast scheduling by T_1 and T_2, and a distributed competition among neighboring sensors to become nonleaf nodes of the broadcast tree by T_1. When a sensor v switches to a nonleaf node by broadcasting its corresponding message into the network, all of its 1-hop neighbors whose status is *undefined* become a *leaf node* and broadcast their status, and sensors with higher residual energy

broadcast first. The 2-hop neighbors of sensor v hear those broadcasts and start to compete with each other to become a nonleaf node, where the sensors with the highest residual energy win the competition. The waiting times T_1^v and T_2^v can be computed as follows:

$$T_1^v = 2t_0 + c/E_v,$$

$$T_2^v = t_0 + c/E_v,$$

where t_0 is an upper bound on the propagation time between a pair of neighboring sensors and c is an adjusting constant. This process will force neighboring sensors with higher residual energy to broadcast before those with lower residual energy.

Note that the broadcast tree grows from the sink. The identified leaf nodes of the tree will turn their radios off. However, they turn them on periodically or when they detect events. The nonleaf nodes constitute the virtual backbone that will be used by the sink to broadcast its queries to the sensors forming the backbones, which will reply by sending their data to the sink using the sensors in the backbone as relays.

A round in EAD is composed of two phases: the virtual backbone formation and data transmission to the sink. While the nonleaf nodes act as relay nodes in any communication between the sink and the source sensors, the leaf nodes are not active. Therefore, those nonleaf nodes deplete their battery power faster than the leaf nodes. Hence, when those nonleaf nodes die, their children have no parents and the broadcast tree is disconnected. Those leaf nodes will have to transmit directly to the sink, which is costly in terms of energy consumption. However, once a current round is done, the next one will start and take care of those leaf nodes, where a new virtual backbone is formed. Thus, the broadcast tree is maintained at the beginning of each round. Furthermore, the virtual backbone formation phase takes much time as messages are broadcast from the sink to all sensors in the network. To make EAD more scalable when the network size increases, a topology-based approach was proposed. In this approach, all sensors' radios are initially off. For every T_0 time, a sensor u randomly wakes up and broadcasts a hello message. An active sensor v that hears this message replies with a message containing a binary $INIT = 1$ if the number of neighbors of v is < 4; otherwise, $INIT = 0$. If u receives $INIT = 1$ or finds out that v has < 4 neighbors, it stays active; otherwise, it switches to the sleep mode. After T_0 time, EAD attempts to build a broadcast tree rooted at the sink. However, it is not guaranteed that the broadcast tree spans all active nodes. Note that the sleeping sensors wake up periodically to compute their parents. These sensors can join the broadcast tree as nonleaf nodes with the help of active neighbors that want to connect to the tree.

4.4.3.7 Information-Directed Routing. In some sensing applications, the information content of the messages exchanged by sensors is very important.

A routing protocol can successively refine the content of the exchanged messages as in a tracking application. Taking this into account, a routing protocol cannot simply be viewed as a message-forwarding mechanism. Depending on where a query is routed, two source-initiated on-demand routing protocols were proposed in Ref. [29], both of which consider a target tracking application.

In the first protocol, a user issues a query from a peripheral sensor (or entry sensor), called *query proxy*. This query is to collect information generated by sensors about a phenomenon of interest. Note that the sending sensor (or query proxy) does not know the destination that is usually known by traditional routing protocols. Therefore, the query proxy will have to find out the high information content area. In other words, the query proxy has to determine a neighboring sensor that the query should be relayed to and may have better information. In this type of routing, each relay sensor includes its own measurement to refine the mobile target estimate. The relay process looks like routing with a gradient in the information field whose content is dynamic given that every relay sensor incorporates its contribution in the form of the knowledge it has about the mobile target. In this context, a greedy approach cannot be applied to this canonical problem of target tracking due to the fact that a greedy search does not consider the choice of any information measurement. Figure 4.9 illustrates the problem that can be caused by applying a greedy routing protocol. While the target moves on a vertical line from X to Y, the relay keeps bouncing between sensors A and B because both sensors have a higher information value about the target. The relay process cannot involve any of the other sensors, for example, E, F, or G, because of the existence of a sensor hole that the mobile target went through. Assume that the mobile target keeps oscillating between X and Y for the same network configuration (Fig. 9b). To deal with the sensor hole problem, the *greedy perimeter stateless routing* (GPSR) [16] can be used. However, the relay process in this case gets away from the target that moves into and out of the hole area. Thus, the routing path should alternate between A and B. Although GPSR is well suited for static planar graphs, it is not a good choice for the mobile target

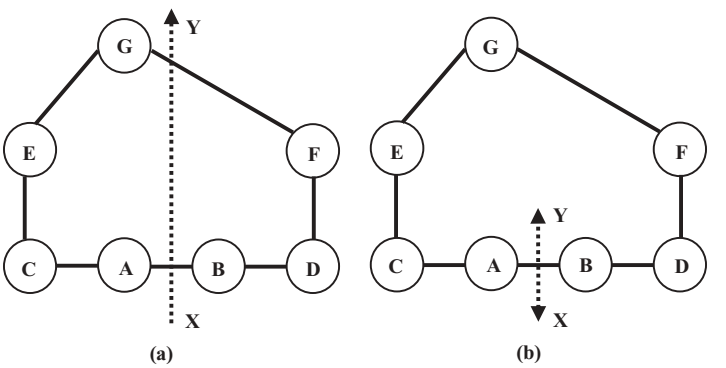


Fig. 4.9 Routing in the presence of sensor holes.

tracking scenario. To address the above problem, an information-directed multiple step look-ahead approach can be used. Specifically, a suboptimal path-finding algorithm, called *minhop* algorithm, is run by each sensor. The objective is to find a path with maximum information aggregation (or gain) among the paths with $< M$ hops. This look-ahead horizon M is selected in a way to be comparable to the diameter of the sensor hole with as low computational cost as possible. When a sensor receives a query-routing request in the form of a tuple $(t, S^{(t)}, P^{(t)})$ of time, state, and path, it wakes up and becomes active. In the case of target tracking, the state is represented by the belief $p\left(x^{(t)} \middle| \overline{z^{(t)}}\right)$, where $x^{(t)}$ is an estimate of the target location based on a set of measurements at time t denoted by $\overline{z^{(t)}} = \{z^{(0)}, z^{(1)}, \cdots, z^{(t)}\}$. Hence, the belief (or prior knowledge) that a sensor has at time t about the location of a target can be expressed by a conditional probability density function. This active sensor selects one of its M-hop neighbors with the highest information value as the destination. Given that there are several minimum hop paths from it to the selected sensor as the destination, it compares between them and chooses the path with the maximum information aggregation by maximizing the function $\Sigma_k I_k$, where I_k is the information contribution of sensor k with measurement $z_k^{(t+1)}$. I_k can be computed as follows:

$$I_k = MI\left(X^{(t+1)}; Z_k^{(t+1)} \middle| \overline{Z^{(t)}} = \overline{z^{(t)}}\right)$$

and

$$MI(U;V) = E_{p(u,v)}\left[\log \frac{p(u,v)}{p(u)\,p(v)}\right] = D(p(u|v)\|p(u)),$$

where $MI(U;V)$ is the *mutual information* [59] between two random variables U and V with a joint probability density function $p(u, v)$ and $D(\cdot\|\cdot)$ is the *Kullback–Leibler divergence* [59] between two distributions. Here mutual information is used to quantify the contribution of each sensor. Note that I_k measures the information conveyed by $z_k^{(t+1)}$ about the location $x^{(t+1)}$ of the target based on the current belief. Thus, I_k quantifies the amount of change in the posterior belief by sensor k by applying its measurement $z_k^{(t+1)}$. After that, the active sensor incorporates its measurement and sends its query to the selected destination using the selected path.

Note that the minimum hop path with the maximum accumulated information is computed after converting the graph $z_k^{(t+1)}$ representing the network. This conversion algorithm assigns to each ingoing edge a cost equal to $L - I_i$, where I_i is the information value at the incident sensor in the graph and L is a large value. Hence, the path-finding problem is turned into a shortest path problem.

In the second protocol, a query proxy will be requested through a query issued by the user to collect information from the network and report to an extraction sensor (or exit sensor). The objective is to collect as much information

as possible as the query is routed from the query proxy to the exit sensor in order to have a good estimate at the exit sensor about the state of the mobile target. Furthermore, the total communication cost should be upper bounded by a prespecified "hypothetical" cost C_0 that is considered as a *soft* constraint. This cost is used to control the trade-off between the communication cost and the information aggregation. To achieve a minimum communication cost, C_0 should be low, thus favoring the shortest path routing. Otherwise, C_0 should be high, thus leading to better information aggregation. This routing protocol is based on the basic best-fit heuristic search, denoted by A^*, in which the merit of a sensor is computed as the sum of the cost g to reach this sensor from the query sensor and the cost h (or cost-to-go) to arrive at the exit sensor. The path along which the query is routed from the query proxy to the exit sensor is guaranteed to be optimal if the estimated cost h does not exceed the true cost-to-go. For example, the Dijkstra's algorithm is a special case of A^* with estimated cost-to-go $h = 0$. For real-time path finding, there is a variation of A^*, called real-time A^* (RTA^*), which uses local information and guarantees finding a path if it exists. However, the solution may be suboptimal and the selected path may have backtracked behavior. By using RTA^*, the active sensor selects the best move based on the estimated cost-to-go h. The total cost for routing a query from the query proxy to the exit sensor is the sum of the communication cost and the information aggregation cost. The communication cost can be estimated based on the Euclidean distance metric. The information aggregation cost can be estimated based on the currently available information. Assume that we have planned the path $P^{(t)}$ and that the current node (or sensor) is v_t. In order to get to the exit sensor v_{exit}, the length of the remaining path is upper bounded by $C_0 - C_{P(t)}$ with $C_{P(t)}$ being the communication cost already paid. The region that covers all possible paths from v_t to v_{exit}, which satisfy the constraint of the communication cost, is given by an ellipse whose major and minor axes are X_1X_2 and Y_1Y_2, respectively, as shown in Fig. 4.10.

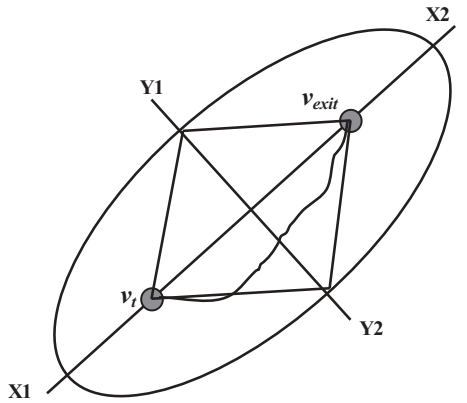


Fig. 4.10 Ellipse for all possible path coverage.

Note that there are an infinite number of paths in the region covered by the ellipse satisfying the length constraint. Thus, it would be difficult to estimate h and only four representative paths are sampled:

- Path 1: $v_t \rightarrow X_1 \rightarrow v_{\text{exit}}$
- Path 2: $v_t \rightarrow X_2 \rightarrow v_{\text{exit}}$
- Path 3: $v_t \rightarrow Y_1 \rightarrow v_{\text{exit}}$
- Path 4: $v_t \rightarrow Y_2 \rightarrow v_{\text{exit}}$

For routing the query from node v_t to the exit node v_{exit}, the information lying ahead is estimated as the maximum among the four paths. This measurement helps approximate the admissible heuristic estimate. In the case that $C_0 - C_{P(t)}$ is less than the Euclidean distance $|v_t - v_{\text{exit}}|$, the estimation of the information is equal to zero and the forward search is equivalent to a shortest path problem using the communication cost only. Likewise, the query from the query proxy to the exit sensor follows the shortest path if $C_0 = 0$.

4.4.3.8 Quorum-Based Information Dissemination. The *quorum*-based protocol [30] uses the same ADV, REQ, and DATA messages as the family of SPIN protocols. The ADV message is used to advertise some new data, the REQ message contains a request for some data, and the DATA message contains the data that were requested by a sensor. This protocol allows a sensor that wants to advertise its data to send an ADV message in both the north and south directions so that the data can reach both the north and south boundaries of the network. On the other hand, a sensor that wants to access the data sends its REQ message into both east and west directions. Note that an ADV message is sent out to a group of sensors while a REQ message is sent out to another group of sensors. Assuming that there exists a sensor that belongs to both groups, this sensor is called a *rendezvous* node that has received both the ADV and REQ messages. Recall that the ADV message contains the location of the sending sensor and the data description, whereas the REQ message contains the location of the querying sensor and the description of its interest. Thus, the rendezvous sensor first checks whether there is a match between the interest in the REQ message and the description in the ADV message. If there is a match, the rendezvous sensor sends a REQ message to the source sensor that initiated the ADV message along the path traversed by the matched ADV message, which in turn sends a DATA message back to the querying sensor that originated the REQ message along the path traversed by the REQ message.

When a neighbor receives an ADV message originally sent by a sensor s, it will record the ADV message along with the sending sensor s. For the northward direction, the sensor with the highest y coordinate is selected to further forward the ADV message to its neighbors. This kind of greedy geographical forwarding continues until the ADV message reaches the farthest sensor at the north boundary. Similarly, the same process will take place for the southward direction where

the ADV message propagates until it reaches the south boundary. Likewise, when a sensor is interested in an advertised data, it will broadcast a REQ message in both eastward and westward directions until it reaches the farthest sensors at the east and west boundaries, respectively. Therefore, the sensors located close to the intersection of the vertical line that passes through the source of the ADV message and the horizontal line that passes through the querying sensor (or sender of the REQ message) have already received both the ADV and REQ messages. These sensors are the rendezvous sensors for the matched pair (ADV, REQ). When one of these rendezvous sensors receives a new REQ message, it checks whether it has already received an ADV message matching that REQ message. If there is a match, the rendezvous sensor will forward the REQ message to the source of the ADV message using the reverse path of the one along which the matched ADV message was sent by its source. Then, the source sensor will send a DATA message to the querying sensor using the reverse path of the one along which that REQ message was sent. When the rendezvous sensor receives the DATA message, it will forward it to the sensor that initiated the REQ message.

Note that the rendezvous sensors are in charge of relaying REQ messages to the source of the matched ADV messages and forwarding DATA messages to the initiators of the matched REQ messages. Since the sensors are supposed to be location aware, the source sensor can use one of the existing geographical routing protocols to deliver the DATA messages directly to the querying sensor without passing by the rendezvous sensor, that is, without using the row and column routes created by the quorum system. This option will help balance the data dissemination load on all the sensors in the network.

4.4.3.9 Home Agent-Based Information Dissemination. In home agent-based information dissemination [30], all sensors in a network are associated with a particular sensor or a location, called *home agent*. Specifically, this home agent could be a real sensor or a geographical area containing a number of sensors. In the quorum-based protocol, the rendezvous sensors are associated with pairs of ADV and REQ, and hence all sensors are candidates to act as rendezvous sensors for data dissemination. In contrast, there is only one home agent for all sensors in the home agent-based protocol. Any ADV message is sent to the home agent and any REQ message is also sent to the home agent. Hence, a home agent acts as a point of contact between the source sensors (i.e., senders of ADV messages) and the querying sensors (i.e., senders of REQ messages).

The selection of a home agent is performed *a priori*. If it is a sensor, a home agent can be selected in a way to minimize the maximum distance between any sensor and the home agent. This distance can be expressed in terms of either the Euclidean distance between two sensors or the number of hops between them. When it is a location, a home agent is the center of the sensor field. Using a geographical routing protocol, sensors can send their ADV and REQ messages to the home agent that will act as the rendezvous sensor for ADV and REQ messages as described earlier. In case of choosing the center of the sensor field as a home agent, it may happen that there is no sensor located at the center of

the field. In this case, the ADV/REQ messages are propagated within a circle centered at the center of the field with some radius. One of the sensors inside the circle will be designated as the rendezvous sensor. In case the circle is empty, that is, a void region, a routing protocol, for example, GPSR [52], can be used to route the ADV/REQ messages around the void area and select a rendezvous sensor for ADV and REQ messages.

Note that the home agent cannot span the entire network as it is the case for the rendezvous sensors in the quorum-based protocol. All the sensors located around the home agent are heavily used in data dissemination, and hence deplete their battery power very quickly, resulting in the energy sink-hole problem.

4.4.4 Multipath-Based Protocols

Considering data transmission between source sensors and the sink, there are two routing paradigms: *single-path* routing and *multipath* routing. In single-path routing, each source sensor sends its data to the sink via the shortest path. In multipath routing, each source sensor finds the first k shortest paths to the sink and divides its load evenly among these paths. In this section, we review a sample of multipath routing protocols for WSNs.

4.4.4.1 Disjoint Paths. First, we consider the design of multipath protocols that help find a small number of alternate paths that have no sensor in common with each other and with the primary path. These protocols are said to be *sensor-disjoint* multipath routing [31,32]. In sensor-disjoint path routing, the primary path is best available whereas the alternate paths are less desirable as they have longer latency. Being disjoint makes those alternate paths independent of the primary path. Thus, if a failure occurs on the primary path, it remains local and does not affect any of those alternate paths. In general, multipath routing leads to the construction of k node-disjoint multipaths by assuming a global knowledge of the network topology. However, these k disjoint paths can also be constructed in a localized manner. Actually, the sink can determine which of its neighbors can provide it with the highest quality data characterized by the lowest loss or lowest delay after the network has been flooded with some low-rate samples. Then, the sink sends out a *primary-path reinforcement* to its best neighbor. This neighbor can apply the same mechanism as in the case of directed diffusion to locally identify its most preferred neighbor. This reinforcement process repeats until the construction of the primary path is done. After that, the sink iterates the same operation for its next most preferred neighbor by sending out to it an *alternate-path reinforcement*. If each sensor accepts only the first reinforcement, those alternate paths are guaranteed to be disjoint with each other and with the primary path. In other words, a sensor negatively reinforces all reinforcements that follow the first one. Note that there is no guarantee that this search procedure of *localized* disjoint paths will discover the same alternate paths as in the idealized version that assumes a global knowledge of the network topology.

4.4.4.2 Braided Paths. Although disjoint paths are more resilient to sensor failures, they can be potentially longer than the primary path and thus less energy efficient. Furthermore, they introduce higher delay when they replace the primary path. Relaxing the disjointedness constraint leads to partially disjoint paths from the primary one, called *braided multipath* [31,32]. To construct the braided multipath, the first step is to compute the primary path. Then, for each node (or sensor) on the primary path, the best path from a source sensor to the sink that does not include that node is computed. As can be seen, those best alternate paths are not necessarily disjoint from the primary path. This set of alternate and primary paths are called *idealized* braided multipaths. Moreover, the links of each of the alternate paths lie either on or geographically close to the primary path. Therefore, the energy consumption on the primary and alternate paths seems to be comparable as opposed to the scenario of mutually disjoint alternate and primary paths.

Similarly, the braided multipath can be constructed in a localized manner. First, the sink sends out a primary-path reinforcement to its first preferred neighbor and an alternate-path reinforcement to its second preferred neighbor. Also, each of the other nodes on the primary path sends out an alternate-path reinforcement to its next most preferred neighbor. Hence, each node on the primary path attempts to route around its immediate neighbor on the primary path toward the source. Furthermore, when a node receives an alternate-path reinforcement, it drops it or propagates it further to its most preferred neighbor depending on whether that node lies on the primary path or not.

4.4.4.3 N-to-1 Multipath Discovery. For the disjoint and braided multipaths previously discussed, the objective is to find multiple disjoint or partially disjoint paths between a source sensor and the sink. This is the case for most of the multipath routing protocols. In Ref. [33], an *N-to-1 multipath discovery* protocol is proposed, which benefits from flooding to find multiple node-disjoint paths from each sensor to the sink simultaneously. This protocol is based on the simple flooding originated from the sink and is composed of two phases, namely, *branch aware flooding* (or phase 1) and *multipath extension of flooding* (or phase 2). Both phases use the same routing messages whose format is given by $\{mtype, mid, nid, bid, cst, path\}$, where *mtype* refers to the type of a message. During phase 1, all paths are primary, which is indicated by *mtype* = *"RPRI"; mid* represents a sequence number of the current routing update; *nid* is the ID of the sender of the message; *bid* is the ID of the branch defined as the ID of the closest node (*nid*) to the sink in the branch; *path* is a sequence of nodes visited by the message; and *cst* is the cost of the path. A message $\{RPRI, mid, Sink, \emptyset, 0, (Sink)\}$ is broadcast by the sink periodically or on-demand in order to initialize the routing update. Upon hearing a message $\{RPRI, mid, nid, bid, cst, path\}$ for the first time, a sensor s determines its parent (*nid*) and rebroadcasts an updated version of the received message in the form of $\{RPRI, mid, s, (bid = \emptyset)?s : bid, cst + \text{cost}(s, nid), path + (s)\}$. Furthermore, the sensor s marks the path $p = path + (s)$ as the *primary path* back to the sink. Note that the sensor s updates the field *bid* only if it is

empty. Also, the cost is incremented by the cost from the sensor s to its parent (*nid*) and the path includes the receiving sensor s. When the sensor s receives the same message from a neighbor s', it marks s' as a *child* or *sibling* based on the content of the path if $bid = s$; otherwise, the message is coming from another branch. Thus, s marks s' as a *cousin*. When the sensor s receives a message from its cousin, it checks if the path $q = path + (s)$ is disjoint with the primary path p and with any other alternate path with a lower cost in the alternate path set, denoted by Q_s. If this is true, the new path q is added to Q_s. In addition, any path that shares some nodes with q and has a higher cost than q will be removed from Q_s. The forwarding process of RPRI messages terminates when each sensor has broadcast the message only once.

While the maximum number of node-disjoint paths from any node to the sink is upper bounded by the number of branches in the spanning tree created by flooding, the links established between the nodes belonging to different branches lead to alternate disjoint paths to the underlying nodes (i.e., the nodes being connected). Figure 4.11 gives an example showing that node w has one primary path (w, r, l, g, d, *Sink*) and an alternate path disjoint with the primary path after it has heard the broadcast by node v, thus creating a link between v and w. Similarly, the node v will also establish an alternate path to the sink through node w. Phase 2 of the protocol allows the nodes to exchange the same message format, but with the type field set to *mtype* = "*RALT*" given that the paths formed in this phase are the alternate paths. These *RALT* messages will allow a node to further propagate the alternate paths to its parent and sibling or cousin node. A sensor s composes a *RALT* message {*RALT*, *mid*, s, q, *bid*, *q.cst*, q} for each alternate path q and broadcasts it to its neighbors. When a sensor s receives a message {*RALT*, *mid*, *nid*, *bid*, *cst*, *path*}, it will learn an alternate path $q = path + (s)$ only if this message was not sent from its parent and s is not included in the path. Also, this path will be added to the alternate path set Q_s of s if q is disjoint with all the paths with a lower cost in Q_s. If this is true, sensor s

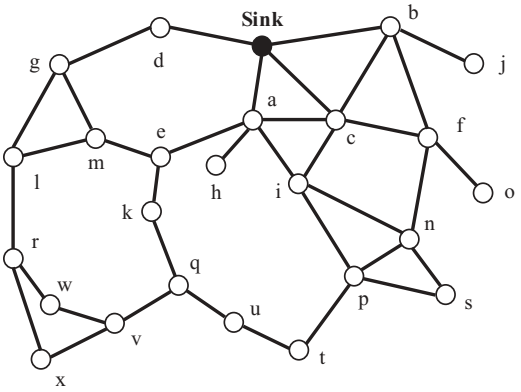


Fig. 4.11 Multipath extension of flooding.

will remove all paths in Q_s that have a higher cost and intersect with q and broadcast an updated version of q:{*RALT*, *mid*, *s*, *q.bid*, *q.cst*, *q*}. The forwarding of *RALT* messages terminates when no new disjoint alternate path can be added to the alternate path set of any node. For example, if node *w* further broadcasts the disjoint paths it learned to its neighbors, node *r* will learn a new alternate path (*r*, *w*, *v*, *q*, *k*, *e*, *a*, *Sink*) to the sink. Therefore, phase 2 helps the sensors discover more disjoint alternate paths at the cost of additional routing messages with regard to the alternate paths found during phase 1 across multiple branches.

This multipath discovery protocol generates multiple node-disjoint paths for every sensor. In multihop routing, an active per-hop packet salvaging strategy can be adopted to handle sensor failures and enhance network reliability. Assume that a network uses a reliable MAC protocol, for example, IEEE 802.11, which acknowledges the successful transmission of each frame. This would help the sensors decide whether they need to keep or remove the most recently transmitted frames from their transmission buffers. When a source sensor sends a packet toward the sink, it includes the source routing option. If at any intermediate sensor transmitting to the next hop is not successful, the current sensor will salvage the packet by randomly selecting a route to the sink and sending this packet along this route. If all next hops from the current sensor fail, the packet should be dropped. However, a routing loop can occur if the new randomly selected path consists of a sensor that the packet has already visited. In order to avoid this problem, the current node has to make sure that there is no common node between the partial route the packet has already traveled and the new candidate route used to salvage the packet. Moreover, the current node needs to update the source routing option so that the packet has the actual path it has gone through when it reaches the sink. Thus, the per-hop salvaging using alternate paths is an effective and efficient technique to handle sensor failures and enhance network reliability. Figure 4.12 illustrates a packet originally sent by sensor *t* being salvaged at sensors *p* and *f* before it reaches the sink.

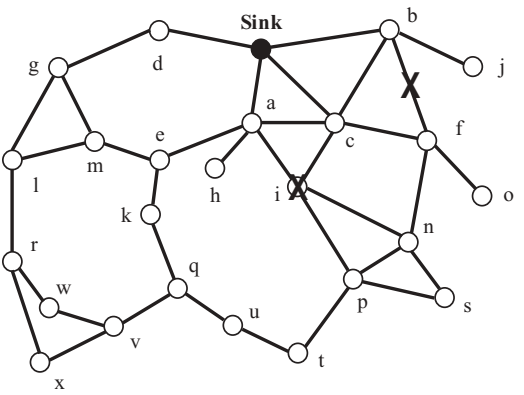


Fig. 4.12 Alternate path packet salvaging.

4.4.5 Mobility-Based Protocols

Mobility brings new challenges to routing and data dissemination in WSNs. In particular, sink mobility requires energy-efficient protocols to guarantee data delivery originated from source sensors toward mobile sinks. This section discusses several routing and data dissemination protocols for mobile WSNs.

4.4.5.1 Joint Mobility and Routing Protocol. A network with a static sink suffers from a severe problem, called *energy sink-hole problem*, where the sensors located around the static sink are heavily used for forwarding data to the sink on behalf of other sensors. As a result, those heavily loaded sensors close to the sink deplete their battery power more quickly, thus disconnecting the network. This problem exists even when the static sink is located at its optimum position corresponding to the center of the sensor field [34]. To address this problem, a mobile sink for gathering sensed data from source sensors was suggested [34]. In this case, the sensors surrounding the sink change over time, giving the chance to all sensors in the network to act as data relays to the mobile sink and thus balancing the load of data routing on all the sensors. Under the shortest-path routing strategy, the average load of data routing is reduced when the trajectories of the sink mobility correspond to concentric circles (assuming that the sensor field is a circle). Another category of mobility trajectories is to move the sink in annuli. However, such movement can be viewed as a weighted average over the movements on a set of concentric circles. In particular, the optimum mobility strategy of the sink is a symmetric strategy in which the trajectory of the sink is the periphery of the network. This result was shown by comparing mobility trajectories on concentric circles of different radii and it was proved that the maximum average load of data routing is achieved at the network center. Therefore, the trajectory with a radius equal to the radius of the sensor field maximizes the distance from the sink to the center of the network that represents the hot spot.

Another dimension of the design strategy that reduces the network load is routing. Regardless of the shape of a sensor field, it is always true that the sensors located on the network periphery are almost not used for forwarding sensed data to the sink on behalf of other sensors and thus have a longer lifetime than all other sensors in the network. Therefore, an efficient routing strategy should exploit the available energy of those sensors close to the border of the network in order to balance the data dissemination load on all the sensors. Based on the sink mobility strategy and the routing strategy discussed earlier, an energy-efficient heuristic for joint routing and mobility is to have the sink moving on a circle of radius $R_m < R$, where R stands for the radius of the sensor field, while data routing depends on the location of the source sensors. Note that the sensor field is divided into two regions: the inner circle and the annulus between the periphery of the network and the trajectory of the sink represented by a circle. The sensors within the inner circle use the shortest path routing to transmit their sensed data to the sink, whereas the sensors in the annulus send their data to the sink using two steps. First, a sensor uses *round routing* around the center of the

network O until the segment OB is reached, where B is the current position of the mobile sink. Then, the data is sent to the sink using a shortest path. This joint heuristic leads to lower network load by reducing the distance between the mobile sink and the sensors that follow the shortest path routing from R, which corresponds to the optimum mobility strategy, to R_m.

4.4.5.2 Data MULES Based Protocol. Although sensor deployment depends on the sensing application, most of the routing and data dissemination protocols for WSNs assume that sensors are very densely deployed in a network. In fact, one of the most desirable features of sensor deployment is network connectivity by which any source sensor is able to communicate directly or indirectly with the sink in order to report its sensed data. To guarantee network connectivity, two different deployment approaches can be used. First, a network can be deployed by using a large number of sensors, yielding a densely connected network in which the source sensors send their sensed data to the sink through multihop communication paths including other intermediate sensors. Second, a network can be deployed by using multiple sinks that cover the entire geographical area, where the source sensors communicate directly with the nearest sink to report their sensed data. While the first approach may not be cost-effective to build a dense and fully connected network, the second one is not cost-effective either, but reduces the communication cost that would be incurred when the sensors communicate with only one single sink. Both scenarios raise the need for developing an architecture that benefits from the two approaches and can guarantee cost-effective connectivity in a sparse network while reducing the energy consumption of the sensors.

To address this need, a three-tier architecture based on mobile entities, called *mobile ubiquitous LAN extensions* (MULEs), was proposed to collect sensed data from source sensors in sparse networks [35]. The MULEs architecture has three main components. The bottom layer contains static wireless sensors that are responsible for sensing an environment; the top layer includes WAN connected devices and access points/central repositories for analyzing the sensed data. Moreover, these access points can be positioned at locations providing network connectivity and power. They communicate with a central data warehouse enabling them to synchronize the collected data, identify redundant data, and acknowledge the receipt of the data sent by the MULEs for reliable data transmission. The middle layer has mobile entities (MULEs) that move in the sensor field and collect sensed data from the source sensors when in proximity to deliver them to those access points when in close range. These MULEs have the capabilities to communicate with both the access points and the sensors using short-range wireless communications. Hence, the MULE component can be considered as a mobile transport agent that connects heterogeneous nodes, namely, the source sensors and the access points. Furthermore, the MULEs can communicate with each other, thus forming a multihop MULE network that can be used to reduce the latency between MULEs and those access points. Because of their motion, the MULEs are able to collect and store data from the sensors, and

acknowledge them. This implies that the MULEs are equipped with larger storage capacities compared to the sensors.

Note that the MULE architecture helps the sensors save their energy as much as possible and thus extend their lifetime. Since the sensors directly communicate with the MULEs through short-range paths, they deplete their energy slowly and uniformly. Thus, the MULE architecture has low sensor energy consumption. Moreover, the MULEs move in the sensor field in a random fashion, which guarantees that all the sensors are equally visited and consume the same amount of energy during their monitoring task. In addition, the MULE architecture has low infrastructure cost. Because of the direct communication between the source sensors and the MULES, there is no routing overhead that would drain the energy of the sensors. As far as robustness and scalability are concerned, the MULE architecture is fault tolerant and scale well. If a MULE fails, it will not affect any particular sensor because no sensor is dependent on any MULE. However, it will degrade the performance of a sparse network for decreasing its data success rate and increasing its latency. Also, when the number of sensors or the number of MULEs increases, there is no need for any network reconfiguration. Note that the sensors have to wait until the MULEs come close by to report their sensed data to the MULEs. Therefore, for time-critical applications, the MULE architecture may introduce an undesirable delay in reporting the sensed data of the source sensors and thus may not be practical. One way to solve this problem is to equip the MULEs with an always-on connection so that they act as mobile sinks (i.e., MULEs and access points). Furthermore, when a MULE fails, the corresponding sensed data will never reach the sink.

4.4.5.3 Two-Tier Data Dissemination. Existing mobile sink-based routing protocols, for example, directed diffusion [36,37], require that each mobile sink propagate its location updates throughout the sensor field in order to inform all the sensors about the direction of sending future data reports. This type of information flooding increases collision in wireless transmissions and yields significant depletion of the limited battery power of the sensors. To alleviate this problem, Ye et al. [36,37] proposed a *two-tier data dissemination* (TTDD) protocol that provides scalable and efficient data delivery to multiple mobile sinks. In TTDD, while the sensors know their own locations, the sinks may or may not be aware of their own locations. Furthermore, the sensors are stationary and aware of their missions, which change infrequently. Therefore, the overhead for mission dissemination is negligible compared to that of sensed data delivery.

The TTDD protocol has three main phases: *grid construction, two-tier query, data forwarding*, and *grid maintenance*. When a source sensor has sensed data to send, it builds its own grid structure for data dissemination, in which the location of the source sensor is one of the crossing points, called *dissemination points*. Given the location of the source sensor $L_s = (x, y)$, these dissemination points are defined by the set L_p, which is given by

$$L_p = \{(x_i, y_i) | x_i = x + i\alpha;\ y_j = y + j\alpha;\ i, j = \pm 0, \pm 1, \pm 2, \ldots\}.$$

Then, the source sensor sends its data announcement message to its four adjacent crossing points that are computed by the source sensor based on its location and the size α of the grid cell. The source sensor uses greedy geographical forwarding to forward the message to the closest neighbor to the crossing points. The neighbor node will use the same forwarding technique until the data announcement message gets received by the sensors closer to the dissemination points L_p than all their neighbors. If the distance between any of these sensors and L_p is less than a threshold $\alpha / 2$, it becomes a *dissemination node* serving a dissemination point in L_p. Upon receiving a data announcement message, a dissemination node stores the source message, the dissemination point it is serving, and the location of the upstream dissemination node. Then, it further forwards the data announcement message to its neighboring dissemination points on the grid except its upstream dissemination point from which it has received the data announcement message. The announcement propagation process continues until each dissemination point is served by a dissemination node. Note that those dissemination points act as reference locations for selecting dissemination nodes. Also, the grid is built on a per-source-sensor basis, thus yielding different sets of dissemination nodes for those source sensors. This approach balances the data dissemination load among all the sensors in the network, enhances its scalability, and provides better robustness in the presence of sensor failures.

The sink can then flood its query within a cell of the grid to receive data from the source sensor. This query will eventually be received by the nearest dissemination node, called *immediate dissemination node*, from the sink. To restrict flooding of its query, the sink includes a maximum distance beyond which any sensor receiving the query will just drop it. The immediate dissemination node will forward the query to its upstream dissemination node from which it has received the data announcement message. This process repeats until the source sensor receives the query. The two-tier query forwarding process has two levels of aggregation. When an immediate dissemination node receives multiple queries from different sinks for the same data (i.e., source sensor), it sends only one query to its upstream dissemination node in the form of an *upstream update*. Likewise, when a dissemination node receives multiple upstream updates from different downstream neighbors, it further forwards only one of them. As can be seen, this two-level aggregation provides scalability with the number of sinks. A dissemination node keeps sending upstream update messages periodically in order to receive data continuously until the sink either stops sending queries or moves out of the local region. As an upstream update message traverses the grid, the dissemination nodes store soft-state timers to forward data streams back to the sinks in the reverse path. These soft-state timers are an order-of-magnitude higher than the time interval between data messages. The rationale behind this design choice is to balance the overhead caused by the forwarding of periodic upstream update messages and that introduced by sending data to the nodes where they are not useful anymore.

When a source sensor receives upstream updates from its neighbor dissemination nodes, it sends back the data to each of those dissemination nodes. These

dissemination nodes forward the data toward the nodes from which they received the upstream updates. This forwarding process continues until the data reach the immediate dissemination node of each sink. Furthermore, if a dissemination node has aggregated queries, it will send a copy of the data to each of its downstream dissemination nodes that sent those queries. In case of relaying data to a mobile sink, a forwarding technique, called *trajectory forwarding*, is used by an immediate dissemination node. With trajectory forwarding, a sink is associated with a *primary agent* and an *immediate agent*. First, a sink chooses one sensor as its primary agent and uses the location of this node in its queries. Initially, both primary and immediate agents are the same sensor. When a sink decides to move out of the range of its current immediate agent, it selects another neighboring sensor as its immediate agent by broadcasting a solicit message and sends the location of this node to its primary agent. This selection is based on the strength of the signal-to-noise ratio of the replying sensors. Any future data will be forwarded to the new immediate agent. The sink selects its immediate agent. It may happen that when a sink is about to move, data were already forwarded to its old immediate agent. To receive these data, the sink also sends the location of its new immediate agent to its old one. Therefore, the immediate dissemination node for the sink forwards the data to the primary agent of the sink, which in turn forwards them to the immediate agent of the sink. This agent is one-hop away from the sink, and hence relays the data directly to the sink. When the sink moves away from its current primary agent (e.g., the location of the new sink is one cell size from the primary agent), it selects a new primary agent by flooding a query locally. Similarly, a sink associates its primary agent with a timer that is set to the duration the sink can stay in a cell. This mechanism avoids receiving duplicate data from its old primary agent. Also, the old immediate agent is associated with a shorter timer whose value is equal to the duration a sink remains within the one-hop distance.

It is worth noting that a grid has a lifetime that is set up by a source sensor. If the grid lifetime expires before receiving any data announcement messages to extend the lifetime, all dissemination nodes clear the grid states. The grid lifetime depends on the mission of the network and the period of data availability. Moreover, to deal with sensor failures, TTDD employs a mechanism, called *upstream information duplication*, where each dissemination node selects several sensors from its neighborhood and replicates in them the location of its upstream dissemination node. If any failure of this dissemination node occurs, the upstream update message from its downstream dissemination node will be processed by one of those selected sensors. This selected sensor will emerge as a new dissemination node and will forward the update message to the upstream dissemination node based on the stored information.

4.4.5.4 Scalable Energy-Efficient Asynchronous Dissemination. To improve TTDD, a distributed self-organizing protocol, called *scalable energy-efficient asynchronous dissemination* (SEAD), was proposed in [52] to trade-off between minimizing the forwarding delay to a mobile sink and energy savings.

SEAD considers data dissemination in which a source sensor reports its sensed data to multiple mobile sinks and consists of three main components: dissemination tree (d-tree) construction, data dissemination, and maintaining linkages to mobile sinks. Also, SEAD does not assume high network density and does not use flooding to find an entry to the d-tree for a mobile sink joining d-tree. However, it assumes that the sensors are aware of their own geographic locations. Every source sensor builds its data dissemination tree rooted at itself and all the dissemination trees for all the source sensors are constructed separately. SEAD can be viewed as an overlay network that sits on top of a location-aware routing protocol, for example, geographical forwarding.

In SEAD, every mobile sink is associated with one of its neighbors, called *access node*, which will be responsible for sending a join request to a source of the d-tree on behalf of its mobile sink. Therefore, when a source sensor reports its sensed data, the access point receives the data and delivers them to its mobile sink. For this purpose, the access point keeps track of the current location of its mobile sink. While a mobile sink is not a member of the d-tree, it is represented by its access node. An access node is selected in a way such that the hop count to its mobile sink does not exceed a certain threshold used to trade-off between the energy consumed on reconfiguring the tree and the path delay. The sensed data of each source sensor is replicated at selected nodes, called *replicas*, which are located between the source sensor and the sinks. They are members of the d-tree. These replicas act as intermediate destinations for the sensed data. A d-tree is a minimum-cost weighted Steiner tree enabling the selection of replicas at intermediate points different from the source sensors and mobile sinks in order to reduce the cost of the d-tree. Figure 4.13 shows the elements of SEAD. All nodes in the d-tree collaborate to disseminate the sensed data to the mobile sinks along the tree in an asynchronous manner. For reliability purposes, each source sensor sends *idle* messages along the tree at a minimum update rate U_m in case it has no new sensed data to report. Moreover, every member of the d-tree has a pointer to each of its children and its parent as well. Thus, when a member of the d-tree does not receive any message within $1/U_m$ time units, it contacts its parent. In case of the parent's failure, the node sends out an error message to the root of the d-tree, requesting a new parent. This mechanism is used to track packet loss and node failures.

The design of the SEAD protocol includes four main phases. In the first phase, *subscription query*, a mobile sink B_i selects the closest neighbor A_i as an access node and issues a *join* query to a source sensor through its selected static access node. This join query message includes the $B_i's$ desired update rate and $A_i's$ location. Then, the access node A_i uses the routing protocol to send this message to the source sensors. Figure 4.13 shows the SEAD tree model.

The second phase, *gate replica search*, consists of determining a gate replica that acts as a grafting point on the data dissemination tree. This d-tree is extended with a new branch from this replica to the new access node. This replica will be connected to the new access node in order to feed it with the sensed data. Hence, this replica r is selected in a way such that it introduces the least additional cost

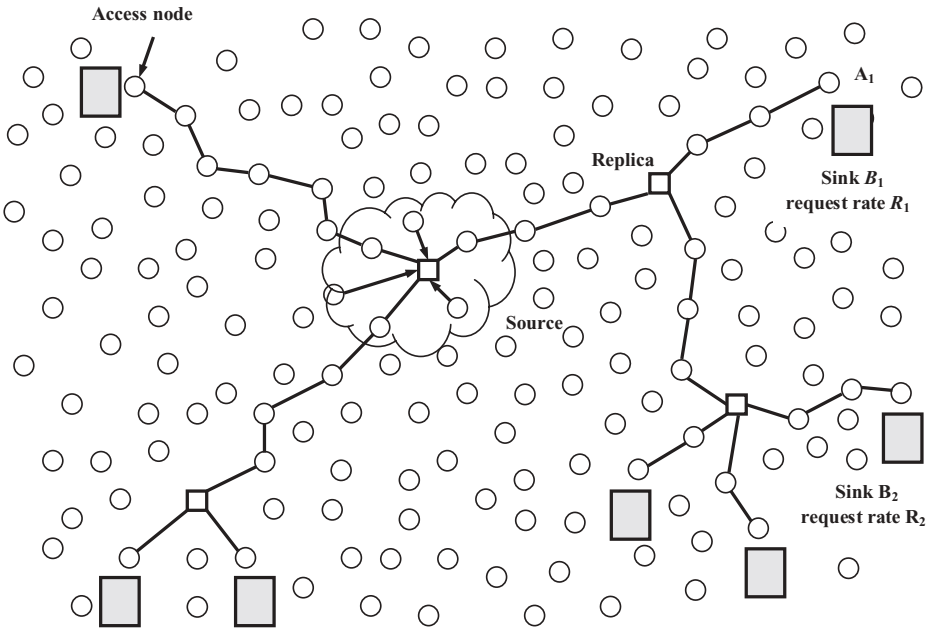


Fig. 4.13. The SEAD tree model.

$K(r)$ for connecting it to the access point. This additional cost $K(r)$ depends on the desired update rate R_i of the sink, the physical distance between the candidate replica and the access node, and the cost of the children of the candidate replica. Let E_r be a set of ancestors of r. In case the node r has a parent and the downstream rate $Q_r^{p(r)}$ of the parent $p(r)$ of r is $<R_i, Q_r^{p(r)} = R_i$. Formally, $K(r)$ is calculated as

$$K(r) = R_i d(r, A_i) + \sum_{m \in E_r} \left\| R_i - Q_m^{p(m)} \right\| d(p(m), m),$$

where

$$\|z\| = \begin{cases} z & if\ z \geq 0 \\ 0 & if\ z < 0 \end{cases}$$

and $p(m)$ is the parent of replica m. In order to calculate $K(r)$, the node r computes the incremental cost $K(r) - K(c)$ for each of its children c, that is, $r = p(c)$, as follows:

$$K(r) - K(c) = R_i d(r, A_i) - R_i d(c, A_i) - \left\| R_i - Q_c^r \right\| d(r, c),$$

where Q_c^r is the downstream rate of the child c. If $K(r)$ is $<K(c)$ for each child c in the set $C(r)$ of the children of the node r, the replica r is selected as the gate replica. Otherwise, the node r forwards the message to one of its children that maximizes $K(r) - K(c)$. It may happen that the message is recursively forwarded until it reaches a leaf node (i.e., another access node). In this case, the parent of this access node is selected as the gate replica.

The third phase, *replica placement*, is to save more communication energy by locally readjusting the dissemination tree (or d-tree) in the neighborhood of the gate replica. This will lead to an optimal d-tree from the source sensor to the access node. There are two modes for connecting the access node to the replica gate: *nonreplica mode* and *junction mode*. In the *nonreplica mode*, the access node is directly connected to the gate replica as a child. In the *junction mode*, a child for the gate replica is created and the access node is connected to the gate replica via its child replica, called junction replica, which sends sensed data to the access node as well as some of the original children of the gate replica. The selection of the appropriate mode should minimize the cost for joining the access node to the d-tree. For this purpose, the gate replica g compares between the energy cost of the nonreplica mode, that is,

$$U_{nonreplica}(c) = d(g, A_i)R_i + d(g, c)Q_c^g$$

for each child $c \in C(g)$ when the gate replica g is the parent of the access node A_i, and the energy of the junction mode, that is,

$$U_{jreplica}(c) = \min_{n \in W}\{d(g, n)\max(R_i, Q_c^g) + d(n, A_i)R_i + d(n, c)Q_c^g\},$$

where W is a set of neighbors of the gate replica g. Then, the gate replica g identifies one of its children $c \in C(g)$ that maximizes $U = U_{nonreplica}(c) - U_{jreplica}(c)$. If $U < 0$, the gate replica is directly connected to the access node. Otherwise, the child c is the sibling of the access node A_i. Then, a *jreplica_search* message that indicates node c is forwarded to the neighbor n, which in turn repeats the above process with respect to its neighbors and forwards the message recursively. This process terminates if dead ends are met, in which case a control message is sent upstream and the previous node is a junction replica. Also, if no neighbor can make $U_{jreplica}$ smaller, the current node is selected as the new junction replica J. The selected replica J stores the downstream rate Q_c^J and the desired update rate R_i of the sink. It also registers the access node A_i as its child and the gate replica g as its parent. Moreover, g sets $Q_J^g = \max\{R_i, Q_c^g\}$.

The fourth phase, *d-tree management*, is to maintain connectivity between the mobile sinks and their access nodes. The selection of access nodes depends on a threshold on the total hop count. When a mobile sink renews its access node, it sends a *disconnect* message to its old one. The old access node informs its parent in the d-tree that it has left the tree. Also, depending on the distance between the new access node and the old gate replica, the mobile sink may or may not request the source sensor for a new gate replica.

4.4.5.5 Dynamic Proxy Tree-Based Data Dissemination. The data dissemination protocols previously discussed, for example, directed diffusion and TTDD, are not efficient enough for some sensing applications, for example, mobile target detection and tracking, where the sensed data have to be disseminated from a dynamic source to multiple mobile sinks. While the directed diffusion protocol requires that source sensors flood the availability of their data throughout the entire network, thus leading to much redundancy, the TTDD protocol maintains a grid-based propagation structure over the entire network proactively irrespective of the locations of the sinks, thus causing a considerable amount of overhead. Furthermore, even the tree-based multicasting protocol is not relevant due to the frequent movement of source sensors and sinks as well as the limited communication ranges of the sensors. In the real world, paths between those sources and sinks may be disconnected frequently, which prevents much of the sensed data from reaching the sinks. Therefore, it is necessary to reconfigure the tree in order to re-stablish routes between dynamic source sensors and mobile sinks, which would cause high maintenance overhead. To address this problem, a *dynamic proxy tree-based data dissemination* framework was proposed in Ref. [39] for maintaining a tree connecting a source sensor to multiple sinks that are interested in the source. This helps the source disseminate its data directly to those mobile sinks.

In this data dissemination framework, a network is composed of stationary sensors and several mobile hosts, called *sinks*. The sensors are used to detect and continuously monitor some mobile targets, while the mobile sinks are used to collect data from specific sensors, called *sources*, which may detect the target and periodically generate detected data or aggregate detected data from a subset of sensors. Because of target mobility, a source may change and a new sensor closer to the target may become a source. Each source is represented by a *stationary source proxy* and each sink is represented by a *stationary sink proxy*. Figure 4.14 illustrates this framework. It is worth mentioning that the source and sink proxies are temporary in the sense that they change as the source sensors change and the sinks move. A source will have a new source proxy only when the distance between the source and its current proxy exceeds a certain threshold. Likewise, a sink will have a new sink proxy only when the distance between the sink and its current proxy exceeds a certain threshold. Moreover, the proxies associated with the same source sensor form a *proxy tree*. The motivation behind the design of such proxies is to reduce the cost of pushing data to and querying data from the source and sink proxies. Since the source sensors are changing and the sinks are moving, the tree should be reconfigured in order to minimize the cost of data multicasting from a source proxy to the sink proxies. For this purpose, both centralized and distributed tree reconfiguration algorithms have been proposed with an objective to minimize the cost of data dissemination and the overhead of tree reconfiguration [39].

Generating a minimum-cost proxy tree is equivalent to constructing a minimum Steiner tree connecting terminals in a graph. In this context, there are two centralized algorithms for reconfiguring a proxy tree: off-line and on-line. The

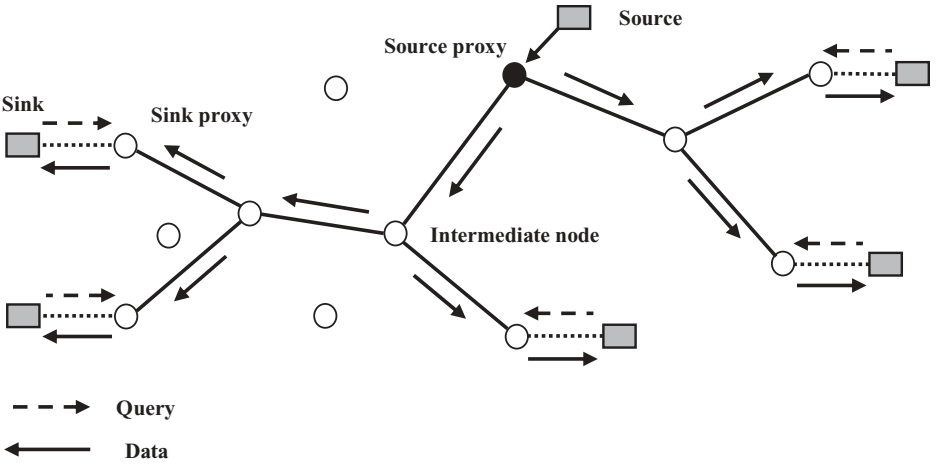


Fig. 4.14 Proxy tree for supporting dynamic multicasting.

off-line algorithm is computationally costly because a proxy tree has to be constructed after any membership change, where a sink that joins or leaves a multicasting group causes a proxy to be added or removed. In other words, the proxy set has to be recomputed for any sink addition or deletion. The two on-line algorithms, namely, the *approximated on-line minimum Steiner tree* (ONMST) and the *enhanced ONMST* (E-ONMST), on the other hand, are not convenient for WSNs due to the large amount of overhead they both introduce in order for a new proxy or its neighbor to gather necessary information to construct a *Voronoi* diagram and reconfigure the subgraph surrounding it whenever a proxy changes due to target and sink mobility. Recall that each sensor has only partial knowledge about its multicasting group, that is, its neighbors in the tree. To alleviate this problem, two distributed heuristic-based algorithms, namely, the *shortest-path* (SP) *based* algorithm and the *spanning-range* (SR) *based* algorithm, were suggested.

The shortest-path based algorithm allows a proxy to join or leave a proxy tree and has three phases: *presearching, finding the closest node*, and *node join*. Let P_n be the proxy of a sink that wants to join the proxy tree. In the *presearching* phase, the proxy identifies the location of the current source proxy using an index-based technique, in which some sensors, called *index nodes*, maintain the locations of the sources [60]. The proxy P_n queries the appropriate index node to learn about the location of the source proxy and sends a *join_req* message to the root (or source proxy) of the proxy tree. When the root receives the *join_req* message, it uses geographical routing to send back a *join_req* message to P_n. This message will be forwarded until it reaches the closest sensor, denoted by P_j, to P_n. Then, P_n determines the closest sensor P_i by flooding a discovery message within a circle of radius equal to the distance $d(P_n, P_j)$ between itself and P_j. In the *finding the closest node* phase, upon receiving the replies from the sensors within the circle, P_n identifies P_i and sends a confirm message to P_i. In the *node*

join phase, upon receiving the confirm message, P_i adds P_n to the proxy tree and reconfigures the resulting subtree including itself and its neighbors into a *full Steiner tree* (FST). As can be observed, the new sink proxy P_n joins the proxy tree by attaching itself to the closest sensor in the tree. Leaving the tree depends on whether P_n is a leaf in the tree or not. If it is a leaf, it sends a *leave_req* message to its parent. If its parent is a Steiner node and has only two neighbors, these two neighbors directly connect to each other and the Steiner node is removed. However, if P_n is not a leaf, it marks itself as a Steiner node and stays in the tree. When a sink/source becomes far away from its proxy, the tree should be reconfigured. Specifically, assume that P_n' is the new proxy that is closer to the sink/source. First, P_n' sends a *migrate_req* message to the old proxy P_n, which in turn adds a temporary edge between P_n' and its parent, denoted by X, and leaves the tree. Similar to a new proxy joining the proxy tree, P_n' identifies its closest sensor P_i. If P_i is not X, P_n' connects to P_i and disconnects from X.

The spanning-range based algorithm improves the shortest-path based algorithm by using flooding to locate itself in the proxy tree. Flooding can degrade the performance of the data dissemination protocol. Specifically, the spanning-range based algorithm assigns a certain *spanning range* to each subtree whose nodes are within the range. Moreover, each node in the tree can decide the spanning range of each of its children using a few simple rules for space decomposition based on the locations of a child and its parent. When a mobile sink decides to join the multicasting tree, its proxy P_n sends a *join_req* message to the source proxy P. The source decides whether to add P_n as its immediate child or forwards its *join_req* message to one of its children whose spanning range covers P_n. This decision is based on the location of P_n with respect to the spanning ranges of the children of the source P computed by P. If P_n cannot be added as a direct child of P, the child receiving the *join_req* message will act exactly as its parent P to decide whether to add P_n as its direct child or forward the *join_req* message to the appropriate child. This process repeats until the sink proxy P_n gets attached as a child to one of the subtrees of the proxy tree.

As a result of sink mobility, the procedure of adding a new sink proxy and deleting the old one for the mobile sink requires migration of the role of a sink proxy from one sensor to another and attaching the new proxy to the proxy tree. The new proxy, say P_n', sends a *migrate_req* to the old proxy P_n, which in turn removes itself from the tree if it is a leaf and sends an *add_req* message to its parent in order to add P_n' to the proxy tree. Similarly, when a source is far away from its source proxy, the proxy tree needs to be reconfigured as the root has changed. Thus, the new spanning ranges are computed by the new root (or source proxy) and forwarded to all of its children. Similarly, each of these children will check whether the spanning ranges of their own children need any maintenance.

4.4.6 QoS Based Protocols

In addition to minimizing energy consumption, it is also important to consider QoS requirements in terms of delay, reliability, and fault tolerance in routing and

data dissemination in WSNs. This section discusses several QoS based routing and data dissemination protocols that help find a balance between energy consumption and QoS requirements.

4.4.6.1 Trade-Off between Energy Savings and Delay. In WSNs, the delay in the transmission of sensed data depends on the transmission time because there is no queuing delay and the propagation and processing delays are negligible compared to the transmission time. On one hand, given N deployed sensors, transmitting sensed data directly to the sink requires a total delay of N units when the sensors transmit one at a time to the sink. This delay can be lowered to $\log N$ units if the sensors are allowed to transmit their sensed data simultaneously to the sink using a binary scheme. However, the energy consumed in data transmission is proportional to the square of the transmission distance between the sending and receiving sensors. For this reason, direct transmission will incur significant energy consumption. On the other hand, minimizing energy introduces longer delay if sensed data have to be sent over short distances. Hence, there is a trade-off between energy consumption and transmission delay.

To account for the delay cost per round of data gathering, Lindsey et al. [40,41] proposed an *energy* × *delay* metric and two data gathering schemes to trade-off between energy and delay. By minimizing *energy* × *delay*, it is possible to achieve acceptable delay for those time-critical applications while reducing energy consumption in sensors, thus extending the network lifetime. Note that simultaneous wireless communications among pairs of sensors is possible if the sensors are CDMA capable, but low interference between those transmissions will occur. In this case, a *binary chain-based scheme*, which is an updated version of the PEGASIS protocol, can be used. However, the *energy* × *delay* cost depends on the sensor distribution in the sensor field. Recall that PEGASIS constructs a chain of sensors in which each sensor receives sensed data from its neighbor in the chain, fuses them with its own data, and forwards them to its neighbor. Assume that the neighboring sensors are equidistant from each other and this distance is equal to d. Then, there are $N/2$ sensors transmitting their sensed data at distance d. According to PEGASIS, the receiving sensors will fuse their own sensed data with the data they have received and become active in the next level of the tree as shown in Fig. 4.15. Therefore, only $N/4$ sensors will be transmitting their sensed data, but at distance $2d$. The same process repeats until the fused data is received by the last sensor, which performs data fusion with its own data and transmits the fused data to the sink. Therefore, the total energy cost for this binary chain-based scheme is proportional to

$$N/2 \times d^2 + N/4 \times (2d)^2 + N/8 \times (4d)^2 + \ldots + 1 \times (N/2 \times d)^2,$$

which is approximated by $N^2/2 \times d^2$ provided that we consider energy consumption in data transmission to the sink.

If the sensors are not CDMA capable, the use of the binary chain-based scheme would introduce a considerable amount of interference. In order to solve

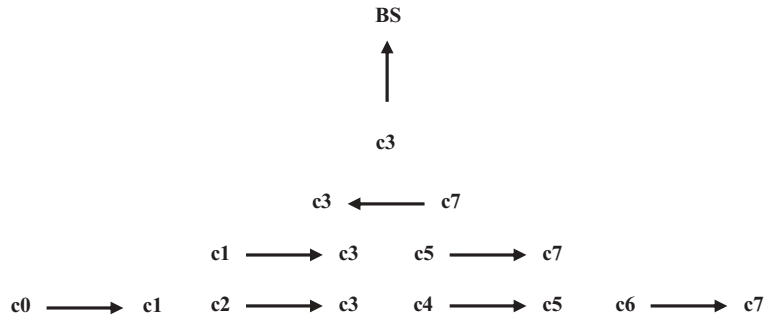


Fig. 4.15 Data gathering in a chain-based binary scheme.

this problem, Lindsey et al. [40,41] proposed a *three-level chain-based scheme*, in which simultaneous transmissions among pairs of spatially separated sensors are possible. As its name indicates, this scheme constructs a three-level hierarchy where each level contains a few groups and each group promotes one sensor to the next level. Figure 4.16 illustrates an example of this scheme. In this example, a set of N sensors is split into G groups, each of which has N/G successive sensors. The value of G is computed based on the number of sensors in the network and the size of the sensor field. Each group will promote one sensor to be active in the next level. Thus, the selected G sensors from the first level will be split into two groups in the second level and a sensor is promoted from each group to have two sensors in the third level. One of these two sensors will be promoted to the last level from which it will transmit the fused data to the sink. Figure 4.16 shows a chain of 100 non-CDMA sensors in the first level of the three-level hierarchy. It was found that for a $100\,\text{m} \times 100\,\text{m}$ sensor field and a network with 100 sensors the best balance between energy and delay is obtained for a value of $G = 10$. This means that only 10 simultaneous transmissions can take place and data fusion occurs at each sensor except the end ones in each level. Note that the leader that will transmit to the sink changes from one round to another in order to balance the load among the sensors.

4.4.6.2 Trade-Off between Energy Savings and Robustness. One of the main requirements of a network for some applications is *functionality*. In other words, the network should remain functional in spite of the occurrence of sensor failures. For this purpose, it is necessary for routing protocols to be fault tolerant (or robust) and at the same time guarantee energy efficiency. To provide robustness, conventional approaches attempt to control the transmission power while maintaining connectivity between sensors and use multipath routing, in which multiple disjoint or partially disjoint communication paths between source sensors and the sink are used. Different from these approaches, Krishnamachari et al. [42] proposed an approach that uses a single-path routing scheme with higher transmission power, but can achieve robustness against sensor failures.

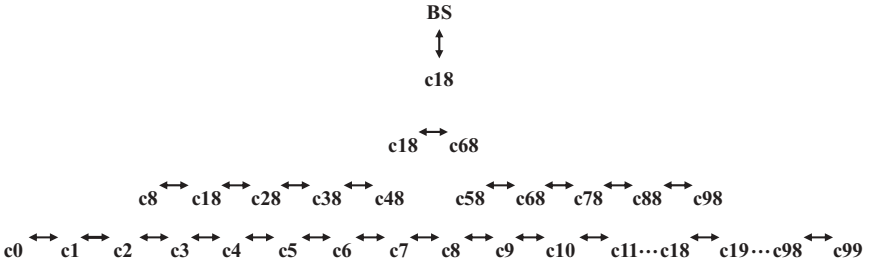


Fig. 4.16 Chain-based three-level scheme for non-CDMA sensors.

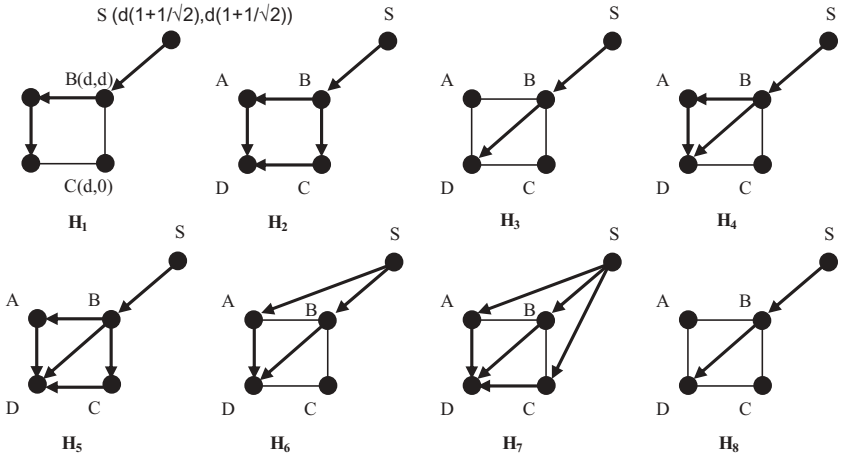


Fig. 4.17 Alternate routing configurations.

Using this approach, for each routing scheme H that routes information from a source sensor to a receiver, an energy metric equal to $m_H R_H^{\alpha}$ is assigned, where R_H is the minimum common transmission radius required for this routing scheme, m_H is the number of transmissions required for the information to reach the receiver, and α stands for the path-loss exponent. The approach assumes that any intermediate sensor can fail independently of other sensors with probability p while the source and the destination are perfectly reliable. The robustness metric Π_H associated with the routing scheme H is equal to the probability that a message initiated by a source sensor reaches the sink. Figure 4.17 shows different routing schemes for routing information from the source sensor S to the destination sensor D while Table 4.3 gives the energy and robustness measures for each of these routing schemes with $\alpha \in \{2, 4\}$.

Given the above assumption that the source and destination sensors are perfectly reliable, the routing scheme H_8 is the most robust one. However, direct

TABLE 4.3 Energy and Robustness Measures for Alternate Routing Configurations

Routing scheme H	E_H ($\alpha = 2$)	E_H ($\alpha = 4$)	Robustness Π_H
H_1	$3d^2$	$3d^4$	$(1\text{-}p)^2$
H_2	$4d^2$	$4d^4$	$(1\text{-}p)(1\text{-}p^2)$
H_3	$4d^2$	$8d^4$	1-p
H_4	$6d^2$	$12d^4$	1-p
H_5	$8d^2$	$16d^4$	1-p
H_6	$10.2d^2$	$35d^4$	$1\text{-}p^2$
H_7	$13.7d^2$	$46.6d^4$	$1\text{-}p^3$
H_8	$5.2d^2$	$34d^4$	1

transmission between the source and destination is highly costly in terms of energy consumption. Thus, there must be a trade-off between robustness, which has to be maximized, and energy consumption, which has to be minimized. Let H_i and H_j be two routing schemes. H_i *dominates* H_j if ($\prod_{H_i} \geq \prod_{H_j}$ and $E_{H_i} < E_{H_j}$) or ($E_{H_i} \leq E_{H_j}$ and $\prod_{H_i} > \prod_{H_j}$). The routing schemes that are not dominated by any other routing scheme form the *Pareto set* and are called *Pareto optimal*. According to Table 4.3, it is clear that $\{H_1, H_3, H_8\}$ is the Pareto set. Note that those Pareto optimal routing schemes provide single-path routes. Moreover, although the routing scheme H_8 uses direct transmission, it consumes less energy than multipath routing schemes H_6 and H_7. This result means that when robustness and energy efficiency are the main concerns, single-path routing outperforms multipath routing under the assumption of perfectly reliable source and destination sensors.

4.4.6.3 Trade-Off between Traffic Overhead and Reliability. While single-path routing routes a sensed data packet from a source sensor to the sink through a sequence of intermediate sensors acting as forwarders, multipath routing routes the same data packet via multiple paths between source and destination sensors. The former scheme is sensitive to the failure of intermediate sensors whereas the latter increases the reliability of data transmission, but yields much overhead. Dulman et al. [43] suggested that a variant of multipath routing be used, where a data packet is split into k subpackets of equal size with added redundancy and sent over k available disjoint paths. In order to construct the original data packet at the destination sensor, only a smaller number of subpackets are needed. The amount of redundancy that should be added for the split message transmission is determined based on the number of successful paths. Let S_k be a random variable representing the number of successfully delivering paths. It is clear that S_k is upper bounded by k, that is, $S_k \leq k$. In fact, all the generated paths are disjoint and the experiment corresponding to transmitting a data packet from a source sensor to a destination sensor can be viewed as a repeated Bernoulli experiment. For the ith path, if the transmission succeeds, this subrun

is assigned 1; otherwise, it is assigned 0. The value of S_k is the sum of the values assigned to the k subruns as there are k disjoint paths. Thus, the expected number of successful delivering paths can be calculated as

$$E(S_k) = \sum_{i=1}^{k} p_i,$$

where p_i is the probability of successfully delivering a message to a destination using the ith path. In order to compute a good estimation for the value of E_k for a given bound α representing the overall probability of successfully reconstructing the transmitted message at the destination such that $P(S_k \geq E_k) \geq \alpha$, Dulman et al. [43] approximated the repeated Bernoulli experiment by a standard distribution $N(\mu, \sigma)$, where the mean is given by

$$\mu = E(S_k) = \sum_{i=1}^{k} P_i$$

and the standard deviation is calculated as

$$\sigma^2 = \sum_{i=1}^{k} p_i (1 - p_i).$$

Given that the total number of subpackets depends on the degree k of multipath routing, a given pair $(k, \{p_1, \ldots, p_k\})$ produces a different normal distribution, $N(\mu(k), \sigma(k))$. To solve this problem, the random variable S_k is transformed into $S_k^* = (S_k - \mu)/\sigma$, which is $N(0, 1)$ distributed. However, the values of the bound x_α are known (see Table 4.1 [43]) for any given α such that $P(S_k^* \geq x_\alpha) \geq \alpha$ is satisfied. As a result, $S_k^* = \dfrac{S_k - \mu}{\sigma} \geq x_\varepsilon$ implies that $S_k \geq x_\alpha \times \sigma + \mu$, and hence we have the following probability

$$P(S_K \geq x_\alpha \times \sigma + \mu) \geq \alpha.$$

By equating this probability with $P(S_k \geq E_k) \geq \alpha$, we obtain an approximation of E_k for a given bound α; that is,

$$E_k = \max(\lfloor x_\alpha \times \sigma + \mu \rfloor, 1).$$

By using the previously computed values of μ and σ, the value of E_k is given by

$$E_k = \max\left(\left\lfloor x_\alpha \times \sqrt{\sum_{i=1}^{k} p_i (1 - p_i)} + \sum_{i=1}^{k} p_i \right\rfloor, 1\right),$$

which gives a good estimation of the number of successfully delivering paths for a given bound α.

4.4.7 Heterogeneity-Based Protocols

All the routing and data dissemination protocols discussed so far assume a homogeneous network architecture, in which all sensors have the same capabilities in terms of battery power, communication, sensing, storage, and processing. Recently, there has been an interest in heterogeneous sensor networks, especially for real deployment. For example, Intel has deployed a pilot application of heterogeneous sensor networks. The proposed architecture uses two types of sensors: the sensors attached to pumps and motors in a fabrication plant have no energy constraint (i.e., line-powered sensors), whereas the others are battery-powered sensors in order to reduce the installation cost and complexity. Those battery-powered sensors have limited lifetime, and hence should use their available energy efficiently by minimizing their potential of data communication and computation.

Another real deployment of heterogeneous sensor networks can be found in [61]. This study demonstrated that CrossBow Mica and iPAQ motes can be integrated together in the same architecture. Since Mica motes use very little power and perform complex computation, it is more efficient to deploy them for sensing only. The iPAQ motes are suitable for data fusion because they consume more power and perform computation quickly. It has been shown that network lifetime can be extended provided that an intelligent assignment of tasks on the heterogeneous sensors is guaranteed.

Heterogeneous networks are attractive as they can potentially extend network lifetime, which is defined as the *time to the first sensor death*, and improve reliability. For both energy efficiency and cost effectiveness, this type of network requires that a large number of inexpensive sensors perform sensing while a few expensive sensors provide in-network processing and data forwarding to the sink. This section discusses how heterogeneity can be used to extend network lifetime and present a few routing and data dissemination protocols for heterogeneous WSNs.

4.4.7.1 Benefits of Heterogeneity in Wireless Sensor Networks. A network consists of two main components: sensors and communication links between them. Hence, heterogeneity can be introduced at these two levels, thus leading to network deployment with *energy heterogeneity* and/or *link heterogeneity*. Yarvis et al. [50] proposed a three-layer architecture for heterogeneous WSNs. In this architecture, the top layer contains only one sink that receives sensed data and analyze them. The second layer includes sensors with no energy constraint. These sensors, called *line-powered* sensors, have unlimited energy resources by connecting them to a wall outlet. The third layer contains battery-powered sensors that are 1-hop away from line-powered sensors. The rationale behind this architecture is that the sensors closer to the sink in a multihop sensor network with many-to-one delivery consume more energy than all other sensors in the network, and thus should be line powered. As observed, this three-layer architecture forms a dominating tree, where each battery-powered sensor communicates with the sink via only line-powered sensors to transmit its sensed data.

There is no communication among battery-powered sensors in order to save their energy, and hence no battery-powered sensor can play the role of a data forwarder on behalf of other sensors. Obviously, there should be a sufficient number of line-powered sensors. If we assume that most of the energy consumption is due to data communication, it can be easily proved that this dominating tree of line-powered sensors rooted at the sink can extend network lifetime by at least $nS_{e2e}/mS_{\text{link}}$ compared to a network architecture without energy heterogeneity, where n is the network size, m is the number of sensors within the radio range of the sink, S_{e2e} is the average end-to-end delivery rate, and S_{link} is the link success rate in the vicinity of the sink [50].

In addition to heterogeneous sensors with regard to their energy, the communication links between the sensors can be heterogeneous as well. To realize link heterogeneity, some sensors need to have high-quality links to the sink; that is, long-distance highly reliable communication links, for example, 802.11 type connectivity. This link characteristic will reduce the average number of hops required for a packet transmitted by a battery-powered sensor to reach the sink. Thus, these high-quality links (or backhaul links) help decrease the end-to-end delay and energy consumption while increasing the end-to-end delivery rate. In other words, the objective of adding heterogeneous links to the network is to increase the rate of successful packet delivery to the sink. The sensors followed by a highly reliable hop to the sink are called *backhaul sensors*. For example, assume that the heterogeneous sensors are deployed in an $m \times n$ Manhattan grid. It is easy to check that the length of the shortest path from a sensor located at location (i, j) to a sink that is adjacent to the midpoint of one edge can be calculated as

$$d\{(i, j), s\} = \left| \frac{m-1}{2} - i \right| + (j+1).$$

If we consider backhaul sensors, each of which is one hop away from the sink, the length of the shortest path from a sensor located at location (i, j) to a sink via a backhaul sensor located at location (k, l), which is denoted by $b(k, l)$, is calculated as

$$d^{b(k,l)}_{((i,j),s)} = |k-i| + |l-j| + 1.$$

Thus, the length of the delivery path for a sensor located at location (i, j) is given by

$$d^{*}_{((i,j),s)} = \min\left\{ d_{((i,j),s)}, d^{b(k,l)}_{((i,j),s)} \right\}$$

over all possible backhaul sensors. Therefore, the sum of all shortest paths from each sensor to the sink is given by

$$S_{m,n} = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} d^{*}_{((i,j),s)}.$$

Hence, there is an optimal deployment of backhaul links for a given $m \times n$ Manhattan grid that minimizes $S_{m,n}$. The benefits of heterogeneity depend on the number of backhaul sensors and their locations. The maximum benefit is obtained when each sensor is one hop away from either a backhaul sensor or the sink. In this case, the end-to-end success rate gets closer to the link success rate. Furthermore, the benefit increases with the number of backhaul sensors. For an arbitrary topology, the optimal deployment of heterogeneous resources (energy and links) corresponds to a tree rooted at the sink. More specifically, the maximum benefit of heterogeneous resources, such as energy heterogeneity and link heterogeneity, depends on the shape, size, and density of the network. For more information about optimal resource deployment, the interested reader is referred to Ref. [50].

4.4.7.2 Information-Driven Sensor Query. An interesting problem in heterogeneous WSNs is how to maximize information gain and minimize detection latency and energy consumption for target localization and tracking through dynamic sensor querying and data routing, which is addressed in Ref. [41]. To improve tracking accuracy and reduce detection latency, communication between sensors is necessary and consumes significant energy. In order to conserve power, only a subset of sensors need to be active when there are interesting events to report in some parts of the network. The choice of a subset of active sensors that have the most useful information is balanced by the communication cost needed between those sensors. Useful information can be sought based on predicting the space and time interesting events would take place.

If x is the parameter representing the target position that we want to estimate, the *belief* that is defined as a representation of the current *a posteriori* distribution of x given a set of measurements $z_1, \ldots, z_N$ is calculated as

$$p(x|z_1, \ldots, z_N)$$

and the expectation value of this distribution is considered as the *estimate* and is given by

$$\bar{x} = \int xp(x|z_1, \ldots, z_N)\,dx,$$

where

$$z_i = \frac{a}{\|x_i - x\|^{\alpha/2}} + w_i$$

with a being a random variable representing the amplitude of the target, α a known attenuation coefficient, w_i a zero mean Gaussian random variable, and $\|\cdot\|$ the Euclidean norm. It is assumed that each sensor, s_i, is aware of its own location, $x_i \in \mathbf{IR}^2$. Since the belief is calculated based on measurements from several sensors, there will be a cost to collect the information. Therefore, it is

necessary to select a subset of sensor measurements providing good information to build a belief state while minimizing the cost of communicating those measurements to a single sensor. To assess the information provided by a sensor measurement to a belief state, Chu et al. [44] introduced a measure called *information content*.

Chu et al. [44] proposed an *information-driven sensor querying* (IDSQ) protocol to optimize sensor selection. In the IDSQ protocol, the first step is to select a sensor l as a leader from a cluster of N sensors. This leader will be responsible for selecting optimal sensors based on some information utility measure, for example, the *geometric measure*, that is,

$$\psi(p_X) = (x_j - \hat{x})^T \sum{}^{-1} (x_j - \hat{x})$$

and the Mahalanobis distance of the sensor under consideration to the current position estimate of the target, and requesting data from them. The Mahalanobis distance measures the distance to the center of the error ellipsoid, normalized by the covariance Σ of the distribution $p_X(x)$. The leader l is supposed to have knowledge of certain characteristics $\{\lambda_i\}_{i=1}^N$ of the sensors, for example, their locations, $\lambda_i = x_i$. Each sensor that is not a leader will wait for a query from the leader sensor. When it is queried, a sensor processes its measures and sends the queried information back to the leader. When a target is within the range of the cluster of sensors, the sensor leader l becomes activated. This activation can be done when, for example, the amplitude reading at the leader is higher than a certain threshold. Then, the leader l computes a representation of the belief state using its own measurement, $p(x|z_l)$, and keeps track of the sensors' measurements that have been incorporated into the belief state, $U = \{l\} \subset \{1, \ldots, N\}$. Based on the quality of the belief, which can be assessed using some goodness measure, the leader may finish processing or continue with sensor selection. In case the belief is not good enough, the leader runs its sensor selection algorithm based on the belief state $p(x|\{Z_i\}_{i\in U})$ and the sensor characteristics $\{\lambda_i\}_{i=1}^N$. When the leader selects a sensor j from the set $\{1, \ldots, N\}\backslash U$ to request data from j, it sends a query to j and waits for the queried data. Upon receiving the information z_j from j, the leader updates its current belief state $p(x|\{z_i\}_{i\in U})$ with z_j, thus leading to a new belief state $p(x|\{z_i\}_{i\in U} \cup z_j)$. Then, it updates the set of sensors that have been incorporated so far; that is, $U = U \cap \{z_j\}$, and check again its belief state as to whether it is good enough. Note that the leader queries only a subset of sensors to obtain the most useful information for it to build its belief state. This intelligent selection helps the sensors save their energy if they do not have pertinent information about a target to communicate to the leader. In Section 4.4.7.3, we describe how the query and the information are routed between a querying sensor (or leader) and the queried sensor using an algorithm, called *constrained anisotropic diffusion routing* (CADR) [60].

4.4.7.3 Constrained Anisotropic Diffusion Routing. By using CADR [60], the selection of the optimal routing path is dynamic and is guided by the

composite objective function that considers the information utility and the actual cost of bandwidth and latency. This composite objective function, M_c, is defined as

$$M_c(\lambda_l, \lambda_j, p(x|\{z_i\}_{i\in U})) = \gamma M_u(p(x|\{z_i\}_{i\in U}), \lambda_j) - (1-\gamma) M_a(\lambda_l, \lambda_j),$$

where $M_u(p(x|\{z_i\}_{i\in U}), \lambda_j)$ represents the information utility function. The parameter $M_a(\lambda_l, \lambda_j)$ stands for the cost of the bandwidth and latency of information communication between sensor j and sensor l, and $0 \le \gamma \le 1$ is a trade-off parameter that balances the contribution from the two terms. Now, let us discuss different cases for query and information routing using CADR.

Case 1: Routing with Global Knowledge of Sensor Positions. In this case, a querying sensor is aware of the locations of all sensors in the network. The best next sensor to select is the one closest to the optimal position x_0; that is,

$$x_0 = \arg_x [\nabla M_c = 0],$$

where ∇M_c stands for the gradient of the composite objective function M_c. It is possible to establish a routing path toward the potentially best sensor along which the measurement from the sensor closest to the optimal position is sent back to the querying sensor. Given this global knowledge of sensor positions, the routing path is optimal.

Case 2: Routing without Global Knowledge of Sensor Positions. In this case, the information query routing is based on localized decisions by individual sensors that consider the regions in the sensor field, where the constraints imposed by the composite objective function M_c are met. Furthermore, since the belief state undergoes updates along the routing path, the function M_c is also updated. The current routing sensor k that holds the information query and is located at x_k selects one of its neighbors $\hat{j}$ as the best next sensor that maximizes the objective function M_c. Formally, this local selection can be expressed as

$$\hat{j} = \arg_j \max[M_c(x_j)], \forall j \ne k.$$

Also, the current routing sensor k can choose the next best one $\hat{j}$ among its neighbors that is located in the direction of ∇M_c, thus satisfying

$$\hat{j} = \arg_j \max\left(\frac{(\nabla M_c)^T (x_k - x_j)}{|\nabla M_c||x_k - x_j|} \right).$$

Another alternative to select the next best sensor is to first determine the direction toward the minimum objective function at any routing step by solving

$$x_0 = \arg_x [\nabla M_c = 0],$$

which allows computing $x_0 - x_k$ that corresponds to the direction toward the optimal position x_0. Then, the next routing sensor can be selected based on the distance

$$d = \beta \nabla M_c + (1 - \beta)(x_0 - x_k),$$

where β is a parameter that is defined as a function of the distance between the current and optimal sensor positions, that is, $\beta = \beta(|x_0 - x_k|)$. Hence, for a large distance d, it would be better to follow the gradient of the objective function. Otherwise, it is more efficient to go toward the minimum rather than following the gradient ascent. As can be seen, this routing alternative chooses the routing direction based on the distance from the optimal position.

The evaluation of the composite objective function and its derivatives requires that a query be sent together with the information on the current belief state. This information should be enough to update the belief state incrementally based on local sensor measurement. In case the Mahalanobis distance is used to quantify the information utility, the triplet $\left\{\hat{x}, x_q, \hat{\Sigma}\right\}$ has to be sent together with the query, where $\hat{x}$ is the current state of the estimated location of the target, x_q is the location of the querying sensor, and $\hat{\Sigma}$ is the current estimate of the uncertainty covariance of the target position. Similarly, a routing path toward the potentially best sensor can be established, along which the measurement from the sensor closest to the optimal position is sent back to the querying sensor. However, this routing path is locally optimal given the greedy nature of the routing algorithm. Moreover, the information provided by the sensors along the path improves incrementally toward the global optimum given that the information utility objective function is monotonically increasing.

4.4.7.4 Cluster-Head Relay Routing. The *cluster head relay* (CHR) protocol [45] uses two types of sensors to form a *heterogeneous* network with a single sink: a large number of low-end sensors, denoted by *L-sensors*, and a small number of powerful high-end sensors, denoted by *H-sensors*. Both types of sensors are static and aware of their locations using some location service. Moreover, those L- and H-sensors are uniformly and randomly distributed in the sensor field. The CHR protocol partitions the heterogeneous network into groups of sensors (or *clusters*), each being composed of L-sensors and led by an H-sensor.

Within a cluster, the L-sensors are in charge of sensing the underlying environment and forwarding data packets originated by other L-sensors toward their cluster head in a multihop fashion. The H-sensors, on the other hand, are responsible for data fusion within their own clusters and forwarding aggregated data packets originated from other cluster heads toward the sink in a multihop fashion using only cluster heads. While L-sensors use short-range data transmission to their neighboring H-sensors within the same cluster, H-sensors perform long-range data communication to other neighboring H-sensors and the sink. Because of their different functioning modes, H-sensors have more powerful resources than L-sensors. As any cluster-based routing protocol, CHR has three phases: cluster formation, intracluster routing, and intercluster routing. At the beginning, the sink broadcasts its location to all H-sensors in the network. These H-sensors advertise their IDs and locations through Hello messages to the L-sensors with a certain random delay in order to avoid collisions between those messages. Upon receiving those Hello messages, each L-sensor selects an H-sensor as its *primary cluster head* based on the strength of the received signal. More specifically, each L-sensor chooses the closest H-sensor as its cluster head. However, in the presence of obstacles, the network is modeled by a *Voronoi* diagram where the nuclei of *Voronoi* cells are the cluster heads. In addition, each L-sensor stores the IDs and locations of the other H-sensors that will serve as backup cluster heads in case of a primary cluster head failure. Up to now, each sensor belongs to only one cluster.

Once an L-sensor selects an H-sensor, it starts sending its sensed data to the H-sensor. If an L-sensor does not have any sensed data to send after T seconds of deployment, it sends a specific location packet to its H-sensor, including its physical location. Therefore, after T seconds, each H-sensor has learned the locations of all L-sensors belonging to its cluster. For each L-sensor, the corresponding H-sensor computes two routes based on the locations of its L-sensors: one is called an *optimal* route, which can be formed based on energy consumption, hop count, or any other metric, whereas the second is called a *suboptimal* route. Then, each H-sensor informs its members of those optimal and suboptimal routes. For this purpose, each H-sensor first divides its cluster into sectors. The number of sectors depends on the number of sensors in the cluster and should be a trade-off between the number of broadcast messages and the message length. Before sending those two routes to each of the L-sensors in its cluster, the H-sensor sends a short message specifying the ID of the sector whose L-sensors should consider the routes being disseminated. Then, the H-sensor broadcasts a long message including the two routes for each L-sensor in its cluster. Note that only the L-sensors in the sector whose ID is recently advertised will be able to receive this long message. If both routes for a given sensor s are not available (e.g., the next hops failed), this sensor, s, broadcasts its packet to its neighbors. If one of the neighbors, say s', replies with an acknowledgment that it knows a route to the sink, s forwards its data to s', which in turn forwards the data to the sink. Otherwise, s becomes disconnected from the sink.

Each cluster head advertises its IDs and locations to its neighbor cluster heads. To send its data to the sink, a cluster head forwards its packet to the cluster head whose *Voronoi* cell is crossed by a straight line connecting cluster head and the sink. Such a *Voronoi* cell is called a *relay cell*. Specifically, the packet will be forwarded from the source cluster head (or simply *source*) to the sink along the cluster heads whose *Voronoi* cells are relay cells. The intercluster routing is similar to a source-initiated routing, in which a route is specified by the source of the message (a cluster head in this case). To enforce this route, the source specifies in the header of the packet the relay cell list as well as the source ID, sink ID, and session ID. The pair (source ID, session ID) uniquely identifies a data dissemination session. To guarantee data delivery, the current cluster head waits for an acknowledgment within a timeout. If it does not hear this acknowledgment, it resends the packet to the same next cluster head. In case the transmission fails again, the current cluster head uses a backup path, which is constructed using the same approach. In other words, this cluster head identifies the relay cells with respect to the sink. If the next relay cell is the one that has failed, the cluster head uses a detoured path to avoid the cell. Otherwise, the new set of relay cells will be used as the actual forwarding path to the sink.

4.4.8 Comparisons

Although we have classified a sample of routing and data dissemination protocols for WSNs according to our taxonomy, it should be mentioned that some of those protocols fit into more than one class. For example, PEGASIS [18] uses data aggregation and helps find a balance between energy and delay [40,41]. Table 4.4 shows the similarities between the protocols surveyed in this chapter with respect to the classification criteria used in the taxonomy.

It is worth mentioning that the transmission of sensed data to the sink takes place in one of the following forms: on-demand, continuous, triggered, and hybrid. In other words, there are four potential data delivery models. For some sensing applications, the source sensors send their sensed data continuously to the sink. For example, in a temperature-monitoring application, the sensors send their data to the sink in a continuous manner without looking at the values obtained. To make their task more energy efficient, the sensors can send their data only when the value of the temperature is above or below a certain threshold. In other words, data transmission is triggered by an event that is based on the threshold. Also, data transmission can be initiated in an on-demand fashion in which the sink requests data from the source sensors by sending them specific queries. These three forms of data transmission can also take place within the same sensing application, that is, in the hybrid form. Moreover, data transmission can be either *broadcast* throughout the network or *unicast* to specific sensors based on some criteria. Although *data delivery models* deal with data delivery from an application traffic perspective [2], routing protocols can also be classified based on the type of data delivery models being used by sensing applications.

TABLE 4.4 Comparison of Routing and Data Dissemination Protocols for WSNs

Classification Criteria	Protocols
Location awareness	GAF, GEAR, Span, TBF, BVGF, GeRaF, MECN, SMECN, PEGASIS, Quorum and home agent-based information dissemination, Joint mobility and routing, TTDD, SEAD, Dynamic proxy tree based data dissemination, Energy-robustness trade-off, IDSQ, CADR, CHR
Network layering	GAF, LEACH, PEGASIS, TEEN, APTEEN, Cougar, EAD, Data MULEs, TTDD, SEAD, Dynamic proxy tree based data dissemination, Energy-delay trade-off, IDSQ, CADR, CHR
In-network processing	LEACH, PEGASIS, TEEN, APTEEN, SPIN, Directed diffusion, Rumor routing, Cougar, ACQUIRE, EAD, Information directed routing, TTDD, Energy-delay trade-off, CHR
Data centricity	GEAR, TEEN, APTEEN, SPIN, Directed diffusion, Rumor routing, Cougar, ACQUIRE, EAD, Information directed routing, Quorum and home agent-based information dissemination
Multipath	TBF, SPIN, Directed diffusion, Sensor-disjoint multipath, Braided multipath, N-to-1 multipath discovery, Energy-robustness trade-off, Overhead-reliability trade-off
Mobility	GAF, TBF, MECN, SMECN, Joint mobility and routing, Data MULEs, TTDD, SEAD, Dynamic proxy tree-based data dissemination.
Quality-of-Service	PEGASIS, TEEN, APTEEN, SPIN, Directed diffusion, Information directed routing, Sensor-disjoint multipath, Braided multipath, N-to-1 multipath discovery, Data MULEs, TTDD, Energy-delay trade off, Energy-robustness trade-off, Overhead-reliability trade off, IDSQ, CADR
Heterogeneity	Data MULEs, IDSQ, CADR, CHR
Energy awareness	GAF, GEAR, Span, MECN, SMECN, LEACH, PEGASIS, TEEN, APTEEN, SPIN, Directed diffusion, Cougar, EAD, Sensor-disjoint multipath, Braided multipath, Joint mobility and routing, Data MULEs, TTDD, SEAD, Energy-delay trade off, Energy-robustness trade off, IDSQ, CADR, CHR

4.5 SUMMARY AND FUTURE DIRECTIONS

One of the main challenges in the design of protocols for WSNs is energy efficiency due to the scarce energy resources of sensors. The ultimate objective behind the protocol design is to keep the sensors operating for as long as possible, thus extending the network lifetime. In particular, routing and data dissemination

is a vital task in WSNs, and thus the design of routing and data dissemination protocols should be specially taken care of. To accomplish their monitoring operation appropriately, the sensors in a network need to collaborate with each other by acting as forwarders of data and control messages on behalf of others. Therefore, it is necessary for the sensors to get involved in the communication between themselves during their operation. However, the energy consumption of the sensors is dominated by data transmission and reception. Specifically, the energy consumed by the sensors in processing (or computation) and sensing is negligible compared to that in data communication. Therefore, routing and data dissemination protocols designed for WSNs should be as energy efficient as possible to prolong the lifetime of individual sensors, and hence the network lifetime. On the other hand, there are other QoS requirements, for example, delay and fault tolerance, to name a few, which should also be considered in the protocol design. To meet such requirements, for example, to minimize delay and increase fault tolerance, some conflicts with the goal of guaranteeing energy efficiency could be introduced. Therefore, a reasonable trade-off should be established between energy efficiency and those QoS requirements.

This chapter surveyed a sample of routing and data dissemination protocols for WSNs based on our proposed taxonomy. This taxonomy takes into account several classification criteria, including location information, network layering and in-network processing, data centricity, path redundancy, network dynamics, QoS requirements, and network heterogeneity. For each of these categories, we have discussed a few example protocols. Our objective is to help the reader get a better understanding of those protocols and gain an insight into how to design efficient protocols that best meet the requirements of a sensing application.

We believe that two important related research directions should receive much attention from the community. While the first concerns the design of routing and data dissemination protocols for duty-cycled WSNs, the second is to consider three-dimensional (3D) sensor fields when designing such protocols. Most of the existing geographic routing and data dissemination protocols assume that all sensors in a network are awake during the forwarding activity. In real-world scenarios, however, the sensors switch between on and off states in order to save their limited energy. It is not even practical to keep a sensor awake all the time while it is active for some short periods of time. Moreover, some sensor failures may have a severe impact on the performance of the network. In case of a sensor failure, the network could be disconnected and partitioned into at least two noncommunicating subnetworks, and hence the existence of the whole network may become meaningless. Therefore, it is important to duty cycle the sensors so that they deplete their energy resources uniformly and slowly. Unfortunately, duty cycling may create a problem for routing a current message to the next hop while it is asleep. To get around this problem, the message can be either buffered until the next hop becomes awake or forwarded over the currently awake sensors. In the former case, a certain delay would be introduced, whereas in the latter case the number of hops may increase significantly, thus leading to considerable energy overhead. Nath and Gibbons [62] addressed this problem by providing a

scheduling algorithm that can be tuned to achieve a certain routing performance. It is more useful that all other routing and data dissemination protocols are designed to handle highly dynamic networks that experience time-varying connectivity due to sensor duty cycling. The challenge is how to duty cycle the sensors while guaranteeing good routing performance. It is also important to extend those protocols to *k*-covered WSNs, where each location in a sensoring field is covered by at least *k* sensors.

Although most of research work on WSNs, in particular, on routing and data dissemination, considered two-dimensional (2D) settings, where sensors are deployed on a planar field, there are some situations where the 2D assumption is not reasonable and the use of a 3D design becomes a necessity. In fact, 3D settings reflect more accurate network design for real-world applications. For example, a network deployed on the trees of different heights in a forest, in a building with multiple floors, or underwater, requires design in 3D rather than 2D space. Oceanographic data collection, pollution monitoring, offshore exploration, disaster prevention, and assisted navigation are all typical applications of underwater sensor networks [63]. Pompili et al. [63] proposed different deployment strategies for 2D and 3D communication architectures for underwater acoustic sensor networks, where the sensors are anchored at the bottom of the ocean for the 2D design and float at different depths of the ocean to cover the entire 3D region. Although some efforts have been devoted to the design of routing and data dissemination protocols for 3D sensing applications, we believe that these first-step attempts are in their infancy, and more powerful and efficient protocols are required to satisfactorily address all problems that may occur, including the ones prior to routing. Perhaps the most nontrivial conceptual problem is sensor deployment. Routing and data dissemination are strictly dependent on the sensor placement in a sensing field. A first question that arises is *How should sensors be placed in a 3D space so that the required quality of monitoring is satisfied*? More importantly, *How should connectivity between the sensors in a 3D space be guaranteed in order to provide a high-quality service of routing and data dissemination*? It has been proved that connectivity depends on coverage. More specifically, a network is connected if the network is configured to provide coverage and the radius of the communication range of the sensors is at least double the radius of their sensing range. Some studies have already considered sensing coverage and network connectivity in an integrated manner. From at least the above questions, it is clear that these three components, namely, sensing coverage, network connectivity, and routing and data dissemination should be studied together. We hope that such studies will be given more attention by researchers in their future work.

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey", *Computer Networks*, vol. 38, no. 4, Mar. 2002, pp. 393–422.

[2] W. Wang, V. Srinivasan, and K.-C. Chua, "Using mobile relays to prolong the lifetime of wireless sensor networks", *Proceedings ACM/IEEE MobiCom'05*, Cologne, Germany, Sept. 2005, pp. 270–283.

[3] K. Akkaya and M. Younes, "A survey on routing protocols for wireless sensor networks", *Ad Hoc Networks*, vol. 3, no. 3, May 2005, pp. 325–349.

[4] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks", *IEEE Communications Magazine*, vol. 40, no. 8, Aug. 2002, pp. 102–114.

[5] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: A survey", *IEEE Wireless Communications*, vol. 11, no. 6, Dec. 2004, pp. 6–28.

[6] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks", *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, Oct. 2002, pp. 660–670.

[7] F. Aurenhammer, "Voronoi diagrams—A survey of a fundamental data structure", *ACM Computing Surveys*, vol. 23, no. 3, Sept. 1991, pp. 345–405.

[8] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less low cost outdoor localization for very small devices", *IEEE Personal Communication Magazine*, vol. 7, no. 5, Oct. 2000, pp. 28–34.

[9] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed energy conservation for ad-hoc routing", *Proceedings ACM/IEEE MobiCom'01*, Rome, Italy, July 2001, pp. 70–84.

[10] Y. Yu, R. Govindan, and D. Estrin, "Geographical and energy aware routing: A recursive data dissemination protocol for wireless sensor networks", *Technical Report UCLA/CSD-TR-01-0023*, UCLA Computer Science Department, May 2001.

[11] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks", *Proceedings ACM MobiCom'01*, Rome, Italy, July 2001, pp. 85–96.

[12] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks", *Wireless Networks*, vol. 8, no. 5, Sept. 2002, pp. 481–494.

[13] B. Nath and D. Niculescu, "Routing on a curve", *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 1, Jan. 2003, pp. 155–160.

[14] G. Xing, C. Lu, R. Pless, and Q. Huang, "On greedy geographic routing algorithms in sensing-covered networks", *Proceedings ACM MobiHoc'04*, Tokyo, Japan, May 2004, pp. 31–42.

[15] M. Zorzi and R. R. Rao, "Geographic random forwarding (GeRaF) for ad hoc and sensor networks: Mutlihop performance", *IEEE Transactions on mobile Computing*, vol. 2, no. 4, Oct.–Dec. 2003, pp. 337–348.

[16] V. Rodoplu and T. H. Meng, "Minimum energy mobile wireless networks", *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, Aug. 1999, pp. 1333–1344.

[17] L. Li and J. Y. Halpern, "Minimum-energy mobile wireless networks revisited", *Proceedings IEEE ICC'01*, Helsinki, Finland, June 2001, pp. 278–283.

[18] S. Lindsey and C. S. Raghavendra, "PEGASIS: Power-efficient gathering in sensor information systems", *Proceedings IEEE Aerospace Conference*, vol. 3, Big Sky, MT, Mar. 2002, pp. 1125–1130.

[19] A. Manjeshwar and D. P. Agrawal, "TEEN: A routing protocol for enhanced efficiency in wireless sensor networks", *Proceedings IPDPS'01*, San Francisco, CA, Apr. 2001, pp. 2009–2015.

[20] A. Manjeshwar and D. P. Agrawal, "APTEEN: A hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks", *Proceedings IPDPS'01*, San Francisco, CA, Apr. 2001, pp. 2009–2015.

[21] W. R. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive protocols for information dissemination in wireless sensor networks", *Proceedings ACM MobiCom'99*, Seattle, WA, Aug. 1999, pp. 174–185.

[22] J. Kulik, W. Heinzelman, and H. Balakrishnan, "Negotiation-based protocols for disseminating information in wireless sensor networks", *Wireless Networks*, vol. 8, no. 2/3, Mar.–May 2002, pp. 169–185.

[23] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks", *Proceedings ACM MobiCom'00*, Boston, MA, Aug. 2000, pp. 56–67.

[24] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking", *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, Feb. 2003, pp. 2–16.

[25] D. Braginsky and D. Estrin, "Rumor routing algorithm in sensor networks", *Proceedings ACM WSNA, in conjunction with ACM MobiCom'02*, Atlanta, GA, Sept. 2002, pp. 22–31.

[26] Y. Yao and J. Gehrke, "The Cougar approach to in-network query processing in sensor networks", *SGIMOD Record*, vol. 31, no. 3, Sept. 2002, pp. 9–18.

[27] N. Sadagopan, B. Krishnamachari, and A. Helmy, "The ACQUIRE mechanism for efficient querying in sensor networks", *Proceedings SNPA'03*, Anchorage, AK, May 2003, pp. 149–155.

[28] A. Boukerche, X. Cheng, and J. Linus, "Energy-aware data-centric routing in microsensor networks", *Proceedings ACM MSWiM, in conjunction with ACM MobiCom*, San Diego, CA, Sept. 2003, pp. 42–49.

[29] J. Liu, F. Zhao, and D. Petrovic, "Information-directed routing in ad hoc sensor networks", *Proceedings ACM WSNA'03, in conjunction with ACM MobiCom*, San Diego, CA, Sept. 2003, pp. 88–97.

[30] D. Liu, X. Hu, and X. Jia, "Energy efficient information dissemination protocols by negotiation for wireless sensor networks", *Computer Communications*, vol. 29, no. 11, July 2006, pp. 2136–2149.

[31] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly-resilient, energy-efficient multipath routing in wireless sensor networks", *Proceedings ACM MobiHoc'01*, Long Beach, CA, Oct. 2001, pp. 251–254.

[32] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin, "Highly-resilient, energy-efficient multipath routing in wireless sensor networks", Mobile Computing and *Communications Review*, vol. 5, no. 4, Oct. 2001, pp. 10–24.

[33] W. Lou, "An efficient N-to-1 multipath routing protocol in wireless sensor networks", *Proceedings IEEE MASS'05*, Washington, DC, Nov. 2005, pp. 1–8.

[34] J. Luo and J.-P. Hubaux, "Joint mobility and routing for lifetime elongation in wireless sensor networks", *Proceedings IEEE INFOCOM'05*, vol. 3, Miami, FL, Mar. 2005, pp. 1735–1746.

[35] R. C. Shah, S. Roy, S. Jain, and W. Brunette, "Data MULEs: Modeling a three-tier architecture for sparse sensor networks", *Proceedings SNPA'03*, Anchorage, AK, May 2003, pp. 30–41.

[36] H. Luo, F. Ye, J. Cheng, S. Lu, and L. Zhang, "TTDD: Two-tier data dissemination in large-scale wireless sensor networks", *Wireless Networks*, vol. 11, no. 1–2, Jan. 2005, pp. 165–175.

[37] F. Ye, H. Luo, J. Cheng, S. Lu, and L. Zhang, "A two-tier data dissemination model for large-scale wireless sensor networks", *Proceedings ACM/IEEE MobiCom'02*, Atlanta, GA, Sept. 2002, pp. 148–159.

[38] H. S. Kim, T. Abdelzaher, and W. H. Kwon, "Minimum-energy asynchronous dissemination to mobile sinks in wireless sensor networks", *Proceedings ACM SenSys'03*, Los Angeles, CA, Nov. 2003, pp. 193–204.

[39] W. Zhang, G. Cao, and T. La Porta, "Dynamic proxy tree-based data dissemination schemes for wireless sensor networks", *Proceedings IEEE MASS'04*, Fort Lauderdale, FL, Oct. 2004, pp. 21–30.

[40] S. Lindsey, C. S. Raghavendra, and K. M. Sivalingam, "Data gathering in sensor networks using the energy*delay metric", *Proceedings IPDPS'01*, San Francisco, CA, Apr. 2001, pp. 2001–2008.

[41] S. Lindsey, C. S. Raghavendra, and K. M. Sivalingam, "Data gathering algorithms in sensor networks using energy metrics", *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 9, Sept. 2002, pp. 924–935.

[42] B. Krishnamachari, Y. Mourtada, and S. Wicker, "The energy-robustness tradeoff for routing in wireless sensor networks", *Proceedings IEEE ICC'03*, Seattle, WA, May 2003, pp. 1833–1837.

[43] S. Dulman, T. Nieberg, J. Wu, and P. Havinga, "Trade-off between traffic overhead and reliability in multipath routing for wireless sensor networks", *Proceedings IEEE WCNC'03*, New Orleans, LA, Mar. 2003, pp. 1918–1922.

[44] M. Chu, H. Haussecker, and F. Zhao, "Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks", *International Journal of High Performance Computing Applications*, vol. 16, no. 3, Fall 2002, pp. 293–313.

[45] X. Du and F. Lin, "Improving routing in sensor networks with heterogeneous sensor nodes", *Proceedings IEEE VTC'05*, Dallas, TX, Sept. 2005, pp. 2528–2532.

[46] Y.-B. Ko and N. H. Vaidya, "Location-aided routing (LAR) in mobile ad hoc networks", *Wireless Networks*, vol. 6, no. 4, July 2000, pp. 307–321.

[47] B. Krishnamachari, D. Estrin, and S. Wicker, "Modeling data-centric routing in wireless sensor networks", *Proceedings IEEE INFOCOM'02*, New York, NY, June 2002, pp. 1–11.

[48] M. Stemm and R. H. Katz, "Measuring and reducing energy consumption of network interfaces in hand-held devices", *IEICE Transaction on Communications*, vol. E80-B, no. 8, Aug. 1997, pp. 1125–1131.

[49] W. Wang, V. Srinivasan, and K.-C. Chua, "Using mobile relays to prolong the lifetime of wireless sensor networks", *Proceedings ACM/IEEE MobiCom'05*, Cologne, Germany, Sept. 2005, pp. 270–283.

[50] M. Yarvis, N. Kushalnagar, H. Singh, A. Rangarajan, Y. Liu, and S. Singh, "Exploiting heterogeneity in sensor networks", *Proceedings IEEE INFOCOM'05*, vol. 2, Miami, FL, Mar. 2005, pp. 878–890.

[51] F. Kuhn, R. Wattenhofer, and A. Zollinger, "Worst-case optimal and average-case efficient geometric ad-hoc routing", *Proceedings ACM MobiHoc'03*, Annapolis, MD, June 2003, pp. 267–278.

[52] B. Karp and H. T. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks", *Proceedings ACM MobiCom'00*, Boston, MA, Aug. 2000, pp. 243–254.

[53] T.-C. Hou and V. Li, "Transmission range control in multihop packet radio networks", *IEEE Transactions on Communications*, vol. 34, no. 1, Jan. 1986, pp. 38–44.

[54] I. Stojmenovic and X. Lin, "Geographic distance routing in ad hoc wireless networks", *Technical Report TR-98-10*, Computer Science Department, SITE, University of Ottawa, Canada, 1998.

[55] O. Kasten, "Energy Consumption", *www.inf.ethz.ch/~kasten/research/bathtub/energy_consumption.html*

[56] P. Bahl and V. N. Padmanabhan, "Radar: A in-building rf-based user location and tracking system", *Proceedings IEEE INFOCOM'00*, vol. 2, Tel-Aviv, Israel, Mar. 2000, pp. 775–784.

[57] L. Doherty, K. S. Pister, and L. E. Ghaoui, "Convex position estimation in wireless sensor networks", *Proceedings IEEE INFOCOM'01*, vol. 3, Anchorage, AK, Apr. 2001, pp. 1655–1663.

[58] M. Zorzi and R. R. Rao, "Geographic random forwarding (geraf) for ad hoc and sensor networks: Energy-Latency Performance", *IEEE Transactions on Mobile Computing*, vol. 2, no. 4, Oct.–Dec. 2003, pp. 349–365.

[59] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, John Wiley & Sons, Inc., New York, 1991.

[60] W. Zhang, G. Cao, and T. La Porta, "Data dissemination with ring-based index for wireless sensor networks", *Proceedings IEEE ICNP'03*, Atlanta, GA, Nov. 2003, pp. 305–314.

[61] R. Kumar, V. Tsiatsis, and M. B. Srivastava, "Computation hierarchy for in-network processing", *Mobile Networks and Applications*, vol. 10, no. 4, Aug. 2005, pp. 505–518.

[62] S. Nath and P. B. Gibbons, "Communicating via fireflies: Geographic routing on duty-cycled sensors", *Proceedings IPSN'07*, Cambridge, MA, Apr. 2007, pp. 440–449.

[63] D. Pompili, T. Melodia, and I. F. Akyildiz, "Routing algorithms for delay-insensitive and delay-sensitive applications in underwater sensor networks", *Proceedings ACM MobiCom'06*, Los Angeles, CA, Sept. 2006, pp. 298–309.

5

BROADCASTING, MULTICASTING, AND GEOCASTING

Baoxian Zhang

Chinese Academy of Sciences, China

Guoliang Xue

Arizona State University, USA

5.1 INTRODUCTION

Broadcasting, multicasting, and geocasting are fundamental operations in wireless sensor networks (WSNs). Broadcasting is to disseminate packet(s) from one source to all other nodes in a network, which is also referred to as one-to-all communications. Broadcasting is useful for disseminating interests, signaling and sensed data, and networkwide software upgrading. Multicasting is to disseminate packet(s) from one source to multiple destinations, which is often referred to as one-to-many communications. Multicasting is useful for a sensor to disseminate its sensed data to multiple sinks, observers, or storage places, and so on. Geocasting is to disseminate packet(s) to all nodes inside a specified area, for example, a circle, rectangular, or polygon area. Geocasting is in particular useful for monitoring a specific area in a WSN. The multihop, self-organized, energy-limited, resource-limited, and broadcast nature of wireless channels and dynamic characteristics of WSNs present a big challenge for the design of efficient broadcasting,

Wireless Sensor Networks: A Networking Perspective, Edited by Jun Zheng and Abbas Jamalipour

multicasting, and geocasting mechanisms. Desirable features of such mechanisms are high efficiency, scalability, reliability, and robustness, and so on.

This chapter is dedicated to the broadcasting, multicasting, and geocasting issues in WSNs. We will introduce related fundamental concepts, discuss major challenges, and give an overview of major existing broadcasting, multicasting, and geocasting mechanisms for WSNs. Section 5.2 introduces fundamental concepts and discusses design guidelines and major challenges. Section 5.3 introduces broadcasting mechanisms. Section 5.4 introduces multicasting mechanisms. Section 5.5 introduces geocasting mechanisms. Section 5.6 summarizes the chapter with a brief discussion of open issues and future research directions.

5.2 CONCEPTS AND MAJOR CHALLENGES

This section introduces related fundamental concepts and discusses the challenges and guidelines for designing broadcasting, multicasting, and geocasting mechanisms in WSNs.

5.2.1 Basic Concepts

Wireless sensor networks is a multihop wireless network, where all nodes cooperate in order to accomplish a given communication task. The network can be represented by a graph $G = (V, E)$, where V is a set of nodes and E is a set of links connecting the nodes in V. A link (u,v) belongs to $E(G)$ if node u can send packets directly to node v. We assume that all nodes have the same maximum transmission range, denoted by R. Each node is equipped with an omnidirectional antenna. A link $(u,v) \in E(G)$ is associated with a positive cost $c(u,v)$, which can be a monetary cost, link power, hop count, and so on. The amount of residual energy associated with a node $u \in V(G)$ is denoted by E_u. The cost (or power) associated with a path is the sum of the costs or power of its constituent links. The residual energy associated with a path (or a tree) is determined by its bottleneck node (i.e., the node with the lowest residual energy). We assume that each node in the network can dynamically adjust its transmission power for communicating with a neighboring node based on the communication range. Moreover, each node is able to estimate the minimal power that it requires to communicate with any direct neighbor. The power associated with link (u,v) is denoted by $e(u,v)$.

We give a definition of fundamental concepts that will be used in the subsequent sections of this chapter.

1. *Broadcast.* Broadcast is a form of communication in which a broadcast source node $s \in V(G)$ disseminates packet(s) to all other nodes in the network.
2. *Multicast.* Multicast is a form of communication in which a multicast source node $s \in V(G)$ disseminates packet(s) to a group of multicast

destinations $S = \{d_1, d_2, \ldots, d_m\}$ in the network, where $d_x \in V(G)\backslash\{s\}$, $1 \le x \le m$.

3. *Geocast.* Geocast is a form of communication in which a source node or multiple source nodes disseminate packet(s) to all nodes inside a target area, which can be a circular, rectangular, or a certain predetermined shape of area. The target area is in general geographically specified. Usually, the geographic information related to the target area is carried in each packet to be disseminated. This information will be used for intermediate nodes to make decisions on packet forwarding in order to accomplish a geocasting task.
4. *Power-Aware Multicast.* To multicast a packet from a source to a group of multicast destinations, a multicast delivery structure is required, which typically uses a tree structure. Power-aware multicast is to multicast a packet in a way such that the total power consumed at the nodes on a multicast delivery structure is effectively reduced. The concept of power-aware broadcast can also be similarly defined.
5. *Tree Power.* Tree power is defined as the total power associated with a tree T, which can be expressed by $P(T) = \Sigma_{u \in V(T)} P_u$, where P_u represents the min-power value required for a node $u \in V(T)$ to successfully send a packet to its farthest downstream on-tree neighboring node. Note that if node u is a leaf node of T, then $P_u = 0$.
6. *Min-Power Multicast.* Min-power multicast is to multicast a packet by building a multicast structure T that covers a group of multicast destinations S so that the total tree power associated with T is the minimum among all such structures. This is the well-known Steiner tree problem (STP), which is known to be NP-Complete.
7. *Minimal Spanning Tree.* A minimal spanning tree (MST) is a tree that covers all nodes in the network and minimizes the total tree cost, where cost is equivalent to power. Hereafter, we will use "cost" and "power" interchangeably unless otherwise stated. Finding an optimal MST in a wired mesh network is known to be polynomial and can be easily solved, for example, by using Prim's method [1]. However, finding an MST in a multihop wireless network, for example, WSN, is known to be NP-Complete. In a network where all nodes operate at the same uniform transmission power, finding an MST is equivalent to finding the minimum connected dominating set of the network. In contrast, in a network where nodes can adaptively adjust their transmission power values, the problem is to find the min-power spanning tree.

5.2.2 Design Guidelines and Challenges

There are several guidelines for the design of broadcasting, multicasting, and geocasting mechanisms in WSNs, including high efficiency, localized operation, reliable transmission, and scalability.

- *Energy Efficiency.* A well-designed mechanism should be able to minimize the total power consumed for accomplishing the task of multicasting, broadcasting, or geocasting and accordingly prolong the network lifetime.
- *Localized Operation.* To forward a packet, it is preferable to perform localized operations such that each forwarding decision is made based only on the information that a packet holder locally keeps and that the packet carries (if any), and the forwarding operations (if any) just taken by the neighbors of the packet holder.
- *Reliable Transmission.* In many cases, reliable transmissions are necessary, which need to consider the broadcast and lossy features of the wireless transmission medium.
- *Scalability.* A well-designed mechanism should have low protocol overhead as the network size increases and be able to accommodate dynamic network environments.

Meanwhile, the main challenges in the design of efficient broadcasting, multicasting, and geocasting mechanisms for WSNs include the following aspects:

- *Limited Energy, Processing, Computation, and Storage Capability at Sensor Nodes.* These are generic constraints posed by WSNs and greatly affect the design of efficient broadcasting, multicasting, and geocasting mechanisms in such networks.
- *Broadcast and Time-Varying Lossy Characteristics of Wireless Links.* The broadcast nature of the wireless medium can ease broadcasting, multicasting, and geocasting operations. However, the lossy nature of wireless links makes packet transmissions, especially, broadcast operations at the MAC layer, unreliable and unpredictable. The latter makes reliable network operations in WSNs difficult. Furthermore, the time-varying characteristic of wireless links makes those issues like routing and medium access even challenging.
- *Quality of Service Requirements.* A critical event should be reported in a given period of time. Otherwise, the information would become outdated and useless. However, the unreliable, time-varying, contention-based characteristics of wireless links make quality of service (QoS) provisioning difficult to be realized.
- *Dynamic Environments.* During the lifetime of a WSN, new nodes may join, existing nodes may die, and link conditions can change over time. A well-designed mechanism should be adaptive to such dynamics while keeping its overhead low.
- *Limited and Inconsistent Local State Information.* In dynamic large WSNs, it is difficult to gather global network state information at sensor nodes or sinks. Thus, suboptimal and even inconsistent decisions on forwarding or medium access can be made at neighboring nodes. In this case, how to make the dissemination of a packet reliably reach intended destinations or areas becomes difficult.

- *Security.* How to make a broadcasting, multicasting, and geocasting operation more secure is another critical concern. However, this is not a focus of this chapter and thus will not be discussed.

5.3 BROADCASTING MECHANISMS

This section introduces major broadcasting mechanisms for WSNs, including some simple and some sophisticated mechanisms.

5.3.1 Simple Broadcasting Mechanisms

This section introduces several simple broadcasting mechanisms, which require neither *a priori* knowledge about network state nor strict coordination among neighboring nodes. Except for blind broadcast, these simple mechanisms can reduce broadcast redundancy to a certain degree without guaranteeing the full coverage of the original network.

5.3.1.1 Blind Broadcast. *Blind broadcast*, also called simple broadcast, is the simplest, but most inefficient, broadcasting mechanism, in which each node in $V(G)\backslash\{s\}$ retransmits every broadcast packet that it receives for the first time. This simple mechanism only requires each node to maintain a list of packets (including packet source ID, sequence number) that it receives recently. Ideally, every node $x \in V(G)$ will receive a broadcast packet $|N(x)|$ times, where $|N(x)|$ represents the number of one-hop neighbors of x, $x \in V(G)$. However, because of the resource blindness and implosion nature of WSNs, blind broadcast leads to a lot of duplicate packet (re-)transmissions and thus results in a huge waste in energy consumption and bandwidth resources. This is the so-called broadcast storm issue [2]. Moreover, if used with a random medium access protocol, blind retransmissions would cause excessive collisions. Therefore, blind broadcast is inefficient in dense WSNs. A simple way to alleviate the transmission collisions of broadcast packets is to insert a small random delay before a node retransmits a broadcast packet.

5.3.1.2 Probability-Based Broadcast. *Probability-based broadcast* [2] is another simple broadcasting mechanism. In probability-based broadcast, upon receiving a nonduplicate broadcast packet, an intermediate node retransmits the packet with probability p $(0 < p \leq 1)$. For a network where nodes are uniformly distributed, if p is above a certain threshold, the network still is expected to be connected with probability 1 by using such a probability-based broadcast operation. To ensure a high probability that such a broadcast source is connected with the other part of the network, all the neighbors of the broadcast source are required to forward a broadcast packet with probability 1 (see, e.g., [3]). However, for a connected network with low density, the probability-based broadcast may

result in the situation where some nodes in the network cannot receive the broadcast packet from the source even though paths exist.

5.3.1.3 Distance-Based Broadcast. *Distance-based broadcast* is a broadcasting mechanism in which a node retransmits a packet depending on distance. It is based on the observation that if all positions in the network have been fully covered by the transmission and retransmission of the broadcast source or intermediate nodes, all nodes in the network should be able to receive the broadcast packet. In this mechanism, upon receiving a nonduplicate broadcast packet from a neighbor node x, an intermediate node y ($y \neq x$) retransmits the packet further if the geometrical distance between x and y is above a certain threshold. By setting a larger threshold, each packet retransmission is expected to cover an additional area that has not been covered by existing transmission(s) due to other nodes. The distance information between neighboring nodes can be obtained via link power estimation, acoustic technique, GPS, and so on.

5.3.1.4 Area-Based Broadcast. *Area-based broadcast* is a broadcasting mechanism in which a node $x \in V(G) \backslash \{s\}$ retransmits a received nonduplicate packet only when it believes that such a retransmission will lead to an area above a certain threshold, which has not been covered by x's 1-hop neighbors' retransmissions of the packet.

5.3.1.5 Counter-Based Broadcast. *Counter-based broadcast* is a broadcasting mechanism in which a node x only retransmits a received nonduplicate packet if the detected retransmissions of the packet by its neighbors is smaller than a predefined number. This mechanism is useful when the network density is high such that each node has a large number of neighbors.

In the next few sections, we will introduce several more sophisticated broadcasting mechanisms for achieving different design objectives, for example, reducing broadcast redundancy, increasing energy efficiency, increasing broadcast reliability, which require the availability of different knowledge (location information, neighborhood knowledge, etc.) to be kept at nodes.

5.3.2 Neighborhood-Aware Broadcasting Mechanisms

Neighborhood awareness is beneficial to reducing broadcast redundancy. In general, the larger the local view each node keeps, the higher the degree to which broadcast redundancy can be reduced. This section introduces some typical neighborhood-aware broadcasting mechanisms and strategies, in which nodes are required to keep their respective 1-hop or 2-hop neighborhood knowledge, and some localized broadcasting mechanisms that can support efficient broadcasting by building a backbone structure (e.g., Connected Dominating Set and cluster-based structures). These mechanisms require neighborhood knowledge to be gathered and kept at nodes in the network. Once a backbone is built,

only the nodes in the backbone are supposed to forward broadcast packets while others do not need to. A backbone structure can be either built on-the-fly or prebuilt.

5.3.2.1 Neighbor Elimination Strategy. *Neighbor elimination strategy* (NES) is a broadcasting strategy for effectively reducing broadcast redundancy. It has been used in some localized broadcasting mechanisms due to its simplicity and effectiveness [4]. In NES, each intermediate node that receives a nonduplicate broadcast packet defers its own retransmission for a certain period of time, which is calculated based on a certain criterion or randomly chosen. At the same time, the node also keeps monitoring its neighborhood and removes those neighbors that are assumed to have received the broadcast packet (from other neighbors) from its rebroadcast list. If the list becomes empty when timed out, its retransmission is canceled. This strategy is simple to implement and effective in reducing broadcast redundancy, while guaranteeing a full coverage of the original network if an idealistic medium access protocol is available at the MAC layer and can guarantee packet delivery without loss. Its efficiency partially depends on how the deferring time at each intermediate node is determined. For an example, a node can defer its retransmission of a received broadcast packet with latency proportional to the number of its neighbors uncovered by other neighbors' transmissions yet. When energy is a big concern, the residual energy of a node can also be considered in the calculation. This neighbor elimination strategy can be embedded into many neighborhood-aware broadcasting mechanisms for improving performance.

5.3.2.2 Connected-Dominating-Set-Based Broadcasting Strategy. *Connected-dominating-set (CDS) creation* is an effective strategy for supporting broadcasting operation. A localized CDS creating algorithm builds a connected dominating set of a network and makes use of the availability of 2-hop neighborhood knowledge at nodes to enable further suppression of broadcast redundancy. A node set is a dominating one if each node in the network is either in the set or a neighbor of a node in the set. For a CDS, each pair of nodes in the set is reachable via only nodes in the set. Nodes in the CDS take the role of forwarding to carry out a broadcast task and are in the "forward" status. The problem of finding the minimum CDS is known to be NP-hard. There are basically two types of CDS-generating strategies [5]: neighbor designating [6,7] and self-pruning [4,5,8,9]. With the neighbor designating strategy, whether a node is in the CDS is determined by its neighbors. Specifically, each node chooses a subset of its direct neighbors as its forward set, through which it can reach each of its 2-hop neighbors.

Figure 5.1 gives an example illustrating how a node $u \in V(G)$ chooses its forward set. If a node is in a neighbor's forward set, then it is in the CDS. Most neighbor-designating methods use similar strategies for nodes to generate their respective forwarding sets. In the *multipoint relaying* (MPR) method [7], the forward set at each node is created in a source-independent manner such that

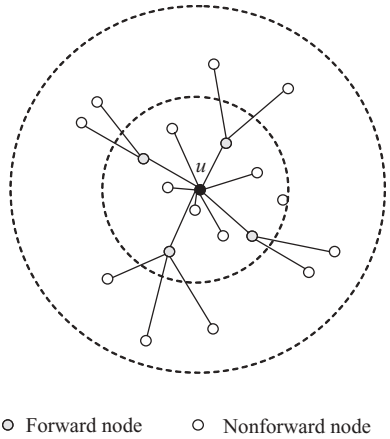


Fig. 5.1 Illustration of how a node *u* chooses its forward set.

each node creates its forward set without considering where the broadcast source is. In contrast, in the *ad hoc broadcast protocol* (AHBP) [4], those nodes that have already received a broadcast packet are excluded from consideration when generating a node's forward set so that a decision is source dependent. Simulation results have demonstrated that source-dependent broadcasting methods outperform source-independent methods in terms of broadcast efficiency at the expense of computational complexity [10]. However, one advantage of source-independent methods is that they allow those nodes not in the CDS to go into the sleep mode when they do not have data for exchanging, while source-dependent methods do not encourage this in order to achieve high broadcast efficiency.

With the self-pruning strategy, a node independently determines whether it becomes a forward node (i.e., $\in$ CDS) based on its local connectivity. For example, if a node's resigning (or self-pruning) does not affect the reachability of any pair of its 1-hop neighbors, it can choose not to be in the CDS. The criteria for determining whether a node is in a forward status can be based on its ID, residual energy, degree, and so on.

5.3.2.3 Cluster-Based Broadcasting Strategy. *Cluster structure creation* is another effective strategy for supporting broadcasting operation. In this strategy, a cluster-based structure is used to facilitate broadcasting operations. In a cluster-based network, sensor nodes are divided into cluster heads, cluster members, and optionally gateway nodes for connecting neighboring clusters, depending on the clustering strategy used. To perform a broadcast operation, only cluster heads and gateway nodes (if any) need to forward broadcast packets while cluster members do not. Figure 5.2 gives an example illustrating how a cluster-based structure is created. A cluster-based structure can be either created in a localized manner by using neighborhood knowledge at nodes or in a global manner. There are many clustering protocols for WSNs in the literature, which

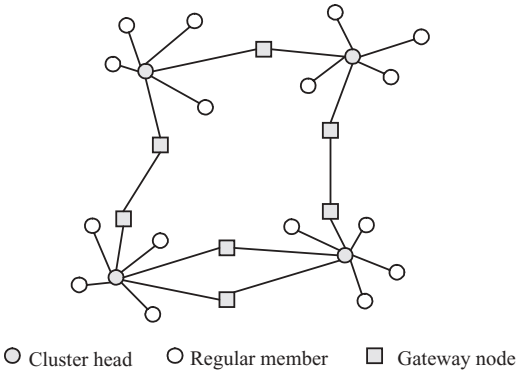


Fig. 5.2 An example of cluster-based network architecture.

are not the focus of this chapter. Readers are referred to Ref. [11] for a survey of clustering protocols.

5.3.3 Location-Aided Broadcasting Mechanisms

Location information can be used to provide efficient broadcast in WSNs. Location information can be obtained by equipping GPS receivers at nodes or using some GPS-free localization techniques. A good survey of localization techniques can be found in Ref. [12]. This section introduces several location-based broadcasting mechanisms.

5.3.3.1 Integrated Distance and Angle-Based Broadcast. *Integrated distance and angle-based broadcast* is a location-aided broadcasting mechanism [13]. In this mechanism, an intermediate node performs a distance-based deferring when receiving a nonduplicate broadcast packet, such that the deferring time is proportional to the reciprocal of the geometrical distance of the link over which the packet is received. Assume that the coverage of each node is a circular area. For a node u, the transmission of a node $v \in N(u)$ will cover a sector of the transmission area of node u, where $N(u)$ represents the one-hop neighbors of node u. Figure 5.3a shows that the transmission area of node v will cover an angle of $\angle AuB$ in the circular area covered by node u. Both A and B are the cross-points of the two circles. Figure 5.3b shows that the transmissions of v and w will cover an angle of $\angle AuD$ ($180° < \angle AuD < 270°$) in the circular area of node u. If the transmission area of node u is fully covered by the transmission areas of some of its direct neighbors, it cancels its own retransmission. Otherwise, a retransmission is triggered when timed out.

5.3.3.2 Geographic Adaptive Fidelity. *Geographic adaptive fidelity* (*GAF*) is another location-aided broadcasting mechanism [14]. In GAF, the whole deployment area of a network is divided into equally squared grids. Nodes

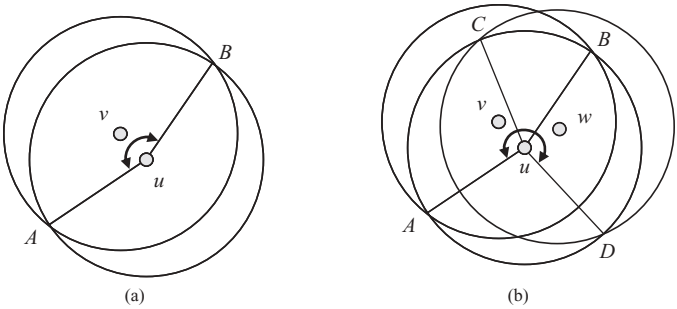


Fig. 5.3 Illustration of angle-based broadcast.

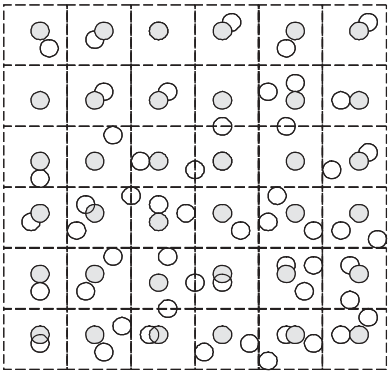


Fig. 5.4 Illustration of GAF $(r = R/\sqrt{5})$.

inside the same grid are equivalent with respect to forwarding packets and coordinate with each other to determine who will sleep and how long it will sleep. The network at the grid level is still connected if the side length of grids is set to not larger than $R/\sqrt{5}$ and if there exists at least one node in each grid, as shown in Fig. 5.4, where R represents the uniform maximum transmission range of nodes. Note that a node at any position in a grid is an immediate neighbor of a node at any position in a neighboring grid and two such grids are called grid neighbors. According to GAF, one node per grid is supposed to be active to keep the whole network connected. Accordingly, to perform a networkwide broadcast operation, the number of retransmissions equal to the number of grids in the network. In this way, broadcast redundancy can be greatly reduced. GAF is in particular useful for dense WSNs.

5.3.3.3 Grid-Based Routing Structure. *Grid-based routing structure* (*GBR*) is a routing protocol that is built on top of the network structure created by GAF and is designed for improving network operation efficiency. Zhang et al. [15] found that to keep a network connected it is unnecessary to enforce

one node per grid to be active as required in Ref. [14]. Instead, a largely reduced subset of the grids can still preserve the same degree of coverage. For example, a small connected dominating set of the network at the grid level is sufficient for this purpose. In each grid, a node can be selected and those selected nodes can create a connected backbone of the whole network. There are two types of backbone structures: cross-sectional (see Fig. 5.5) and diagonal-enabled (see Fig. 5.6). In the cross-sectional structure, only vertical or horizontal grids are grid neighbors and in this case the grid side length $r = R/\sqrt{5}$ (to achieve the minimal number of grids to support this structure). Each row of active grids can cover

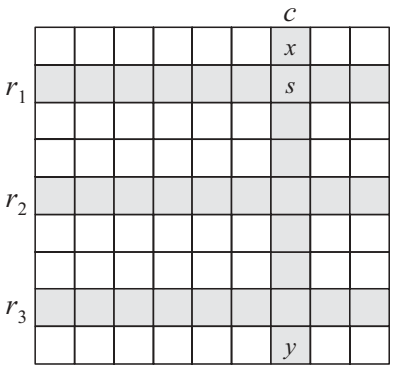


Fig. 5.5 Illustration of GBR: cross-sectional routing $(r = R/\sqrt{5})$.

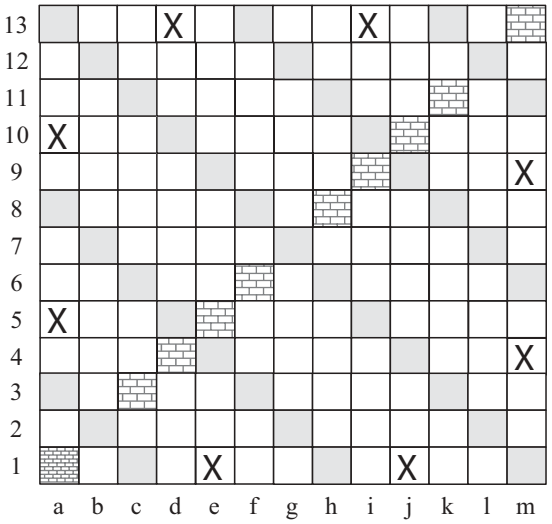


Fig. 5.6 Illustration of diagonal routing $(r = R/\sqrt{8})$.

three rows. In contrast, in the diagonal-enabled structure, a grid and any grid (directly) around it are grid neighbors. That is, each grid has eight grid neighbors, excluding the border effect. In this case, r will be set to $R/\sqrt{8}$. Each diagonal of active grids can cover five neighboring grid diagonals, including the grid itself. Analytical results have shown that the diagonal-based structure outperforms the cross-sectional-based structure in terms of path length (for packet delivery) and the number of active grids (for network connectivity). The reason is that a grid in the diagonal-based structure covers more areas than its counterpart in the cross-sectional structure due to the difference in the number of neighboring grids. Moreover, location information can also be used in neighborhood-aware broadcasting mechanisms for further improving performance. For more details regarding the grid-based routing protocol, please refer to Ref. [15]. Note that in Fig. 5.6 some extra grids are still needed to make those X-grids connected with other shaded grids. Moreover, the purpose of the a1-m13 diagonal is to make those (other) orthogonal diagonals connected.

5.3.4 Energy-Efficient Broadcasting Mechanisms

Energy-efficient broadcasting can prolong the network lifetime of a WSN because sensor nodes are typically battery operated and in many environments it is difficult, if not impossible at all, to recharge sensor batteries. In this context, a typical problem is how to minimize the total power consumed for disseminating a packet across the network, which is often referred to as the min-power broadcast problem. This problem is equivalent to finding a min-power spanning tree, which has proved to be NP-Complete [16]. Note that for a network where all nodes operate at the same maximum transmission power, this problem is equivalent to finding a minimum CDS, which is known to be NP-hard.

Before proceeding further, we first illustrate the concept of *wireless broadcast advantage*. Suppose a node $i \in V(G)$ is equipped with an omni-directional antenna and has a list of neighboring nodes $(x_1, x_2, \ldots, x_k)$. It can then send at a power level $Tx_{\text{wireless}} = \max\{e(i, x_u) | 1 \le u \le k\}$ to cover all its neighbors. In contrast, the total energy required to accomplish such a broadcast task will be $\Sigma_{1 \le u \le k}\, e(i, x_u)$ in a wired network with point-to-point links. The reduction in the energy (or cost) required to broadcast a packet from one node to multiple neighboring nodes over a wireless channel compared with its counterpart in a wired network constituent of point-to-point links is referred to as the *wireless broadcast advantage* [17,18]. This section introduces several energy-efficient broadcasting mechanisms for WSNs.

5.3.4.1 Broadcast Incremental Power. *Broadcast incremental power (BIP)* is the first power-efficient broadcast mechanism for multihop wireless networks [17]. This broadcast protocol exploits the wireless broadcast advantage in the construction of broadcast trees. In Ref. [17], two versions of BIP are discussed: min-power-path-based and minimal spanning tree based (MST-based). In the min-power-path-based BIP, a spanning tree is constructed by using Dijkstra's

algorithm, which is the union of the min-power paths from a source to every other node in the network. The MST-based BIP is similar in principle to Prim's algorithm in building a MST in the sense that new nodes are added to an existing tree one at a time on a minimum extra cost basis until all nodes are added onto the tree. The difference from Prim's algorithm is that at each step of adding a new node, BIP can incrementally increases the transmission power of an internal on-tree node (from a nonzero power), or that of a leaf node (from zero power). The min-power-path-based BIP outperforms the MST-based BIP in terms of power consumption. In the implementation of BIP, global network state information is required.

5.3.4.2 Near-Maximum Lifetime Broadcast. *Near-maximum lifetime broadcast* is a broadcasting mechanism that follows the principle of BIP for building an efficient broadcasting structure and introduces a look-ahead feature to further improve energy utilization in broadcasting [19]. In this mechanism, broadcasting spanning trees are first built on a per-packet basis instead of on a per-session basis in some other mechanisms in order to better balance energy consumption among nodes. At each step of adding a new node onto an existing tree, in addition to the individual extra cost for adding this new node to the existing tree, the extra cost (or power) from this new node to its (not-on-tree) neighbors are also considered for tree expansion in the future. In this way, the possibility of extending the tree due to a bad decision in the future is minimized. This process continues until a spanning tree is created. Simulation results have demonstrated that this broadcasting mechanism outperforms BIP in terms of network lifetime [19].

5.3.4.3 Min-Hop Maximum Residual Energy Broadcast. *Min-Hop maximum residual energy broadcast* is a broadcasting mechanism that builds a spanning tree with the maximum residual energy [20]. The maximum residual energy of a path is determined by the on-path node with the lowest residual energy. Accordingly, the maximum residual energy associated with a tree is determined by the in-tree node with the lowest residual energy. The min-hop maximal residual energy path connecting a pair of nodes u and v is such a path with the highest residual energy and if there are multiple such paths, the one with the minimum hop count among them is selected. We define the diameter of a tree T to the length of the path with the highest hop count from a source node to one of the leaf nodes in T. A maximum residual energy spanning tree with the minimum diameter is the spanning tree with the maximum residual energy tree and if there are multiple trees the one with the minimal diameter is selected. The problem of constructing such a tree is referred to as the minimum diameter maximum residual energy spanning routing problem. Low and Goh [20] studied how to bind the maximum path length (in hop count) from a source node to every other node in the process of generating such a tree, where a centralized algorithm is used. At each step, a subtree is expanded subject to the minimum diameter constraint in such a way that the minimum diameter constraint is not

violated. This process continues until a spanning tree is created. In this way, the worst-case latency for packet broadcasting can be largely reduced, which is beneficial to delivering high-priority information and delay-sensitive packets. The implementation of this mechanism requires the availability of global network state information.

5.3.4.4 Localized Power-Efficient Broadcast. The broadcasting mechanisms introduced above require global network state information. Although efficient, they have a severe scalability problem and are difficult to be deployed in large-scale WSNs. In Ref. [21], a *localized power-efficient broadcasting* mechanism was proposed, which combines a neighbor-eliminating strategy and a localized topology control mechanism for lower power broadcasting. In this mechanism, each node first builds its reduced neighbor set using a localized minimal spanning tree algorithm (LMST) [22] for distributed topology control with the assistance of its 1-hop topology that it locally keeps. Nodes in the reduced neighbor set is denoted by $NS(x)$ for $x \in V(G)$. The source node s uses a power for reaching the farthest neighbor in $NS(s)$. Upon receiving a broadcast packet, an intermediate node x first generates its list of LMST neighbors that have not received the packet based on its recent overhearing history, and schedules a timer based on the number of such neighbors. When timed out, the node retransmits the packet to the farthest neighbor that is supposed to have not received the packet yet (if any). Simulation results demonstrated that the LMST based broadcasting mechanism has competitive performance as compared with the BIP mechanism, which requires the availability of global network state information in its implementation. In Ref. [23], it is investigated to achieve a good trade-off between (1) reaching more nodes in a single-hop using higher transmission power and (2) reaching fewer nodes using lower transmission power and relaying packets through multiple hops.

In summary, a key design guideline for energy-efficient broadcasting is to balance the minimization of power consumption for broadcasting each packet across the network and the energy consuming rate among nodes in the network. Moreover, power consumed for packet reception at a receiving node can be another factor affecting the performance of a broadcasting mechanism, which was not considered in [17,21,23]. How much power in total is consumed for packet reception in a networkwide broadcast operation depends mainly on the average degree of a node. The higher degree a node has, the more listeners it has when it sends a packet to the channel and the more power will be consumed to receive the packet. It would be a waste of power for a node to receive the same packet from more than one neighboring nodes.

5.3.5 Reliable Broadcasting Mechanisms

To support reliable dissemination of packets, reliable broadcast is needed to ensure reliable delivery of packets from a source to all other nodes. Packet loss can be caused by link transmission error, collision, or buffer overflow at

nodes. To design reliable broadcasting mechanisms, the unreliable feature of wireless channels should be considered. It is important to provide highly reliable broadcast while reducing broadcast redundancy. This section introduces several approaches for supporting reliable broadcast. Due to limited space, we will not discuss how to provide reliable end-to-end broadcasting at the transport layer.

5.3.5.1 Recursive Reliable Unicast. A simple mechanism to achieve reliable broadcast in multihop wireless networks is to perform reliable unicast on each individual link in the original network or in a prebuilt delivery structure (e.g., a spanning tree). To perform a networkwide reliable broadcast on a connected graph G, a number of $|E(G)|$ (for the former) or $|V(G)|$-1 (for the latter) transmissions are required provided that no packet loss, corruption, or collusion occurs. According to this mechanism, each branching node needs to unicast packets reliably to each of its downstream on-tree nodes separately. Obviously, this mechanism is expensive because it does not consider the broadcast nature of wireless channels.

5.3.5.2 Most Reliable Spanning Tree. *Most reliable spanning tree* is a tree that can be used to achieve high reliability in packet broadcasting from a source to other nodes in the network. Let $p(u,v)$ be the link error bit rate of link (u,v). A most reliable spanning tree is the tree with the minimal value of $\Sigma_{(u,v)\in T} lg[1\text{-}p(u,v)]$ among all such trees. A most reliable broadcasting mechanism creates the (most) reliable spanning tree and an MST creating algorithm using the link bit error rate as the primary metric can be used for this purpose. Finding the most reliable spanning tree in WSNs is NP-Complete due to the broadcast nature of wireless links in such networks. In this mechanism, each branching node on the spanning tree only needs to forward a packet one time, which may be received by all its downstream receivers or at least with high probability. To build the most reliable spanning tree, various heuristics (either centralized or distributed) can be employed based on the preference on different trade-offs. This mechanism, however, cannot guarantee end-to-end reliable packet transmissions.

5.3.5.3 Integrated Round-Robin Reliable Unicast and Promiscuous Listening. Reliable broadcasting can be performed by integrating round-robin reliable unicast and promiscuous listening [24]. In this mechanism, a logical tree is first built for a broadcasting task. Each nonleaf node in the tree maintains a children list (CL), which contains all its downstream on-tree neighbors. Each node unicasts packets to each of its children using a reliable MAC protocol in a round-robin manner. Specifically, every time a node forwards a packet with sequence number x, it only unicasts the packet to one child in its CL. All other nodes in the CL list will promiscuously listen to the channel to hopefully get the packet. The intended receiver replies with an ACK, which indicates the receipt of the current packet and also those missing packets (if any) in its receiving

window. As a result, those missing packets (if any) will be retransmitted. Once the retransmission(s) (if needed) is finished, the sender will repeat the whole process by sending the next packet (with sequence number $x + 1$) to another node in its CL list, which is the so-called round-robin reliable unicast. This mechanism requires each nonleaf node maintains a cache of recent packets that it has sent.

5.3.5.4 Broadcast with Selective Acknowledgments and Double Coverage. *Broadcast with selective acknowledgments and double coverage* is a broadcasting mechanism for achieving highly reliable dissemination of broadcast packets [25]. In this mechanism, each node independently computes a forward set among its 1-hop neighbors, which can cover all its 2-hop neighbors. After a node sends out a broadcast packet, each node in its forward set will send an ACK back to the sender while each nonforward 1-hop neighbor will not send an ACK. In this way, the ACK implosion problem can be largely alleviated. Failing to receive an ACK at the sender will cause a retransmission, up to a maximum number of retries. Moreover, the mechanism guarantees that each node is covered by at least two transmissions so that a missing packet caused by a single collision can be avoided.

5.3.5.5 TDMA Based Broadcast. *TDMA based broadcast* can be used in a TDMA based network, in which a successful handshaking at the MAC layer between a sender and its intended (1-hop) receivers can ensure reliable broadcasting at the MAC layer. An example of TDMA based broadcast mechanisms is the traffic-adaptive medium access protocol (TRAMA) proposed in Ref. [26]. In TRAMA, each node maintains a list of its neighboring nodes within its 2 hops, including those that currently have packets to send. According to the list, each node in the network can independently perform a hashing operation onto each node in the list and the node itself to calculate which node in its 2-hop scope has the highest priority to send over time. If the node itself has the highest priority to send, it will send its stored packets in an appropriate slot without causing collision with others because it is the only sender within the 2-hop neighborhood. This mechanism enables nodes to adaptively listen, send, and sleep, and can realize both reliable broadcast and multicast. Simulation results have shown that it outperforms contention-based MAC protocols [26]. However, it requires a certain proactive overhead for exchanging traffic information among neighbors, which can consume a certain amount of energy when traffic load is low [27].

5.4 MULTICASTING MECHANISMS

The main design objectives of multicasting mechanisms is to achieve high efficiency in terms of minimizing the number of relaying nodes, minimizing the total transmission power, and maximizing the network lifetime. Other concerns include

robustness and scalability, which are important in large-scale WSNs where link quality changes over time, and sensor nodes can join, die, and move. Although there are many multicasting protocols for mobile ad hoc networks (MANETs), most of them do not consider the unique characteristics of WSNs and thus cannot be directly used in WSNs.

This section introduces typical multicasting mechanisms proposed particularly for WSNs. These multicasting mechanisms can be classified into two categories: tree based and location based. The former type of mechanisms in general builds a tree structure for multicasting without the assistance of location information. The latter uses location information in building a multicast structure, which can greatly facilitate the mechanism design.

5.4.1 Tree-Based Multicasting Mechanisms

Tree-based multicasting mechanisms have been widely used in MANETs and WSNs. The core assisted mesh protocol (CAMP) [28], the location-guided tree constructing algorithms [29], the overlay multicast protocol [30], and the differential destination multicast (DDM) [31] are all multicasting mechanisms for MANETs. These mechanisms assume the availability of the reachability information between each pair of nodes in the network and use the reachability information between each pair of group members as virtual links to build a cost-effective tree [28–30] or to facilitate the packet forwarding at intermediate on-tree nodes [31]. The multicasting algorithm proposed in [32] mainly addresses the issue of how to maintain high efficiency of a multicast tree in a dynamic environment, where nodes may move arbitrarily by introducing route optimization and TTL-Scoping for members to reconnect to the tree. However, all the above mechanisms are not suitable for WSNs because of two main reasons. First, they were proposed for MANETs where peer-to-peer communications is typical and an underlying unicast routing protocol is required to maintain the reachability between all pairs of nodes. In WSNs, however, sensor-to-sink communications is typical and there is in general no need for maintaining the connectivity between a pair of sensor nodes. Second, the above mechanisms are aimed at highly dynamic environments, where frequent link breaks and creations are normal due to node movements. In WSNs, however, sensor nodes are in general static and link quality changes slowly over time. Therefore, these multicasting mechanisms are not suitable for being used in WSNs. In the next few sections, we will introduce several multicasting mechanisms that are particularly proposed for WSNs.

5.4.1.1 Multicast-Enabled Ad Hoc On-Demand Distance Vector Routing. The *multicast-enabled ad hoc on-demand distance vector routing protocol* (MAODV) is *a tree*-based *multicasting* mechanism [33]. In this mechanism, a source that has data to multicast first broadcasts a route request (RREQ) packet to the network. Each intermediate node receiving the RREQ packet will further retransmits the packet if it receives the packet for the first time. Upon

receipt of a RREQ packet, each multicast destination sends a route reply (RREP) packet back to the source in a hop-by-hop manner along the reverse path. A RREP packet travels back toward the source until reaching a node that has already sent a RREP packet back to the source or the source itself. After that, the source will send data packets downstream. The structure taken by the RREP packets is a source-rooted multicast tree. Although this mechanism was originally proposed for MANETs, it is also suitable for WSNs. However, its cost efficiency, for example, in terms of bandwidth utilization, is low.

5.4.1.2 Centralized Power-Aware Multicast. *Centralized power-aware multicast* is a multicasting mechanism that multicasts data by building a low-power tree based on global network state knowledge. A min-power-path-based multicast tree is a union of the min-power paths from a source to each of its multicast destinations, which can be computed by using Dijkstra's shortest path algorithm. One way to compute a global min-power multicast tree is to use linear programming. However, this is with prohibitively high computation overhead and thus is suitable only for very small networks. Another way to build a low-power multicast tree is to first compute a min-power spanning tree by using Prim's method and then prune those links that do not lead to any multicast destinations. Note that Prim's method produces heuristic solutions due to the NP-Completeness of the min-power multicast problem. In the computation of either a min-power-path-based multicast tree or min-power spanning tree (using Prim's method), global network state information and multicast group membership information are required.

5.4.1.3 Localized Power-Aware Multicast. *Localized power-aware multicast* is one type of broadcasting mechanisms, which broadcast data packets by building a low-power multicast tree based on neighborhood knowledge that each node independently collects and stores. An example of such broadcasting mechanisms is the LMST based broadcasting mechanism [21]. In this mechanism, however, an extra pruning process is needed after a power-efficient broadcasting structure is created. In the pruning process, upon receipt of a broadcast packet, each multicast destination sends a mini-report packet back to the source, which travels toward the source until reaching a node already on the existing tree or the source. However, those branches without receiving any mini-report packets will finally time out and then be pruned, which is implemented by setting a timer for each downstream receiver at an intermediate node. The remaining structure will be the multicast delivery structure. This mechanism is similar to the "flooding-and-then-prune" strategy used in the DVMRP protocol [34].

5.4.2 Location-Based Multicasting Mechanisms

This section introduces several location-based multicasting mechanisms for supporting a source sensor node to send sensed data to multiple sinks, either static or mobile.

5.4.2.1 Scalable Energy-Efficient Asynchronous Dissemination. *Scalable energy-efficient asynchronous dissemination* (SEAD) [35] is a multicasting protocol that takes advantage of the stationary nature of sensors for multicasting data to multiple mobile sinks. In SEAD, each mobile sink associates itself to a closest sensor called access point (AP), which may change over time. An energy-efficient tree can be built by treating a data source and each of the APs as group members. For this purpose, replica points are selected in a way such that geometrical distances between group members are used as virtual costs in order to reduce the total tree cost. A sink can dynamically adjust its AP as it moves in order to maintain high delivery tree quality. SEAD requires the availability of sinks' locations for replica point selection and tree constructions.

5.4.2.2 Geographic Multicast Routing. *Geographic multicast routing* (GMR) [36] is a multicasting mechanism that extends greedy (unicast) geographical forwarding discipline to support multicasting. In GMR, the optimizing objective of greedy geographical forwarding is to reduce the bandwidth consumption such that the total number of transmissions for accomplishing a multicast task is minimized. With GMR, each packet carries the IDs of multicast destinations (or a subset of them) and is forwarded further to each of these destinations independently in a greedy manner. However, those destinations that share the same next hop will go along the same way in the hop-by-hop forwarding in GMR. In this way, the total tree cost can be greatly reduced by such path sharing for reaching different destinations. Each packet will be forwarded in a hop-by-hop manner until it reaches its intended destination(s). GMR is fully distributed and operates in a localized manner in tree formations and thus scales well.

5.4.2.3 Two-Tier Data Dissemination. *Two-tier data dissemination* (TTDD) [37] is a simple mechanism for multicasting data from a source to mobile sinks, which provides scalable and efficient data multicasting. In TTDD, each data source proactively builds a grid structure, which enables mobile sinks (or observers) to continuously receive data on the move by flooding queries within a local cell only. TTDD exploits the fact that sensor nodes are stationary and location aware to construct and maintain the grid structure with low overhead. Figure 5.7 shows the grid structure that is composed of many dissemination points for a data source to advertise the availability of its sensed data. In Fig. 5.7, only one sink is shown. To advertise the detection of an event, the source sensor that detects the event will periodically disseminate the event across the network by "posting" it to some dissemination points (or nodes). A sink interested in this event will send a local query to a nearby dissemination point, via which it can be grafted to the grid-based information delivery structure. Compared with SEAD, TTDD is simpler for delivery structure creation and management, while SEAD has better cost performance.

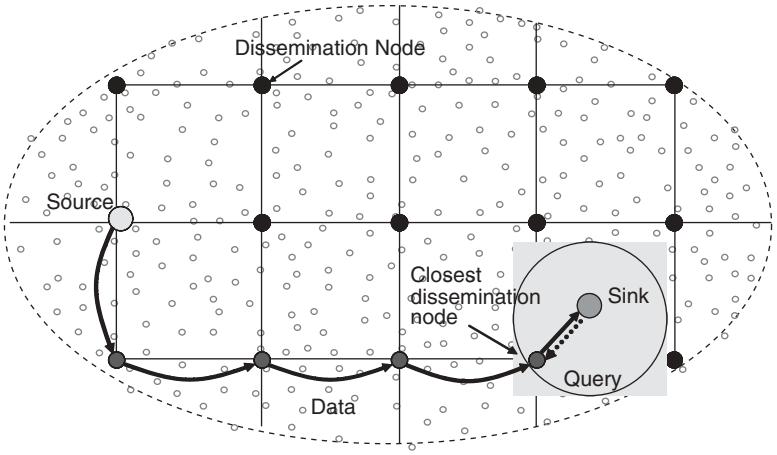


Fig. 5.7 Illustration of the TTDD protocol.

5.5 GEOCASTING MECHANISMS

The main design objectives of geocasting mechanisms are to improve delivery efficiency, network robustness, and network scalability. In general, geocasting implicitly assumes the availability of location information at nodes because geocasting operations, by definition, are geography information based. Based on whether nodes in a target geocast area can receive those geocast packets or not when paths exist, existing geocasting mechanisms can be classified into two categories: nonguaranteed and guaranteed, which will be introduced in the subsequent sections, respectively.

5.5.1 Nonguaranteed Geocasting Mechanisms

This section introduces several nonguaranteed geocasting mechanisms for WSNs and discusses their characteristics.

5.5.1.1 Unicast Routing with Area Delivery. *Unicast routing with area delivery* (URAD) [38] is a nonguaranteed geocasting mechanism. In URAD, a source first unicasts geocast data toward a point inside the geocast area (e.g., the center of the target area) via a geographical forwarding discipline (e.g., GPSR [39]) and then performs pure flooding once the packet reaches a node inside the region. When the network density is high, it can ensure that those nodes in the target geocast area are connected by using only the links connecting those nodes inside the area. However, when the network inside the geocast area is disconnected, URAD can lead to the situation where some nodes in the geocast area cannot receive the geocast packets even though there are paths existing between the geocast source and these nodes.

5.5.1.2. Directed-Flooding-Based Geocasting. *Directed-flooding-based geocasting* is a type of geocasting mechanisms based on detected flooding. The key idea is to define a packet forwarding zone based on the geocast region and the position of the data source. Ko and Vaidya [40] proposed three directed-flooding-based geocasting mechanisms based on different criteria for defining the forwarding zone. In the static forwarding zone mechanism, the forwarding zone is defined to be the smallest rectangular covering the data source and the geocast area, as shown in Fig. 5.8a. Only nodes in the forwarding zone forward data packets, while others simply discard the packets. In the adaptive forwarding zone adjusting mechanism, each intermediate node receiving a geocast packet can independently redefine (shrink) the forwarding zone based on its own position and the target geocast zone, as shown in Fig. 5.8b. Gradually, this can reduce the size of the forwarding zone for reduced overhead by avoiding a visit to those nodes that are not helpful to the geocasting task. The third mechanism adjusts (or reduces) the forwarding zone based on the geometric distance such that an intermediate node forwards a geocast packet only if it is closer to the centric point of the target geocast area than the node from which it receives the packet, as shown in Fig. 5.8c. Note that the nonshaded nodes in Fig. 5.8 are not included in the forwarding zones.

5.5.1.3 Performance Comparison. Maihofer [41] conducted computer simulations to compare the performance of the above geocasting mechanisms. The results obtained show that the simple flooding mechanism achieves the highest packet delivery ratio with the highest overhead. Surprisingly, URAD is the second best in terms of packet delivery ratio, but with the lowest overhead. The reason is that URAD benefits from the use of reliable transmission in the unicast forwarding phase, which uses RTS/CTS/ACK dialogue at the MAC layer. In contrast, in those mechanisms presented in Ref. [40], packet broadcast is

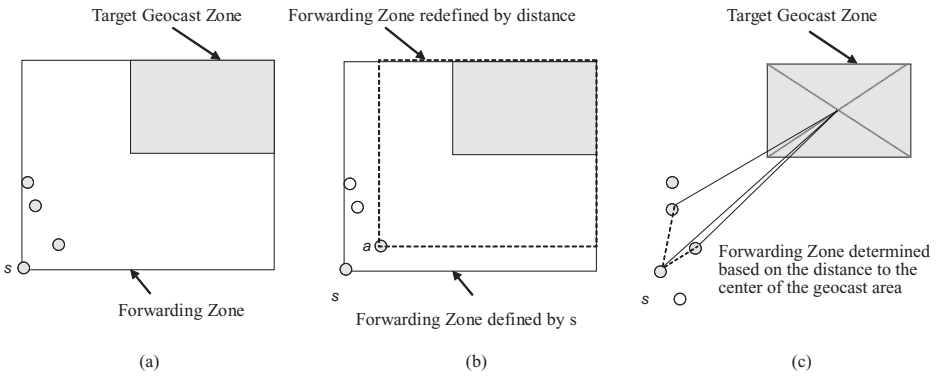


Fig. 5.8 Illustration of geocasting mechanisms: (a) static forwarding zone; (b) adaptive forwarding zone adjusting; and (c) distance-based forwarding zone adjusting.

performed for packet retransmissions inside and also outside the target geocast area, which is unreliable in nature.

All of the above geocasting mechanisms cannot guarantee the successful delivery of geocast packets to each node in the geocast area when paths exist.

5.5.2 Guaranteed Geocasting Mechanisms

This section introduces a couple of geocasting strategies for achieving guaranteed geocasting in WSNs. We assume that there exists no packet loss due to transmission errors or collisions.

5.5.2.1 Simple Flooding. *Simple flooding* (or blind broadcast) is a mechanism that performs simple flooding such that each node forwards the first geocast packet that it receives. This mechanism can ensure that each node in the target area receives a copy of a geocast packet if the transmissions over wireless links are reliable. However, it does not consider the geocast region information at all. Therefore, it is expensive to implement and is only suitable for the case when the geocast area is almost the same as the network deployment area.

5.5.2.2 Geocasting via Efficient Broadcasting. *Geocasting via efficient broadcasting* is the simplest strategy for achieving guaranteed geocasting. It can employ any (efficient) broadcasting mechanism that we introduced earlier, which can ensure that each node in a network receive a copy of a geocast packet, assuming that no packet loss occurs during packet transmission and the network is connected. This strategy can be used when energy efficiency is not a concern in the protocol design or the geocast area is almost the same as the whole network deployment area.

5.5.2.3 Geocasting via Face Routing. *Geocasting via face routing* is another strategy for achieving guaranteed geocasting. Stojmenovic [42] proposed three guaranteed geocasting mechanisms based on face routing [43,44]. Face routing only uses the edges of a planarized graph and can guarantee the delivery of packets from a source to the intended destination. Among the three mechanisms, the first two use the strategy of face traversal and are developed based on a depth-first search of a face tree. The first is based on an understanding that geocasting delivery can be guaranteed if all faces of a planar graph that are inside or intersect the geocasting region are traversed [44]. In the second, only those faces that intersect the geocast region boundary are traversed. Further, an intrageocast-region flooding is enforced from the boundary. In the third, namely, the entrance-zone-multicasting-based mechanism, an entrance ring of the geocast region is first built and this ring is then divided into nonoverlapping zones, each of which has a diameter equal to the uniform transmission range R, as shown in Fig. 5.9. In this figure, s is the geocast source and the area covered by the outer rectangular is the target geocasting area. The outer ring filled with 84 nonoverlapping $R/2 \times R/2$ squares is the entrance zone. The geocasting problem is

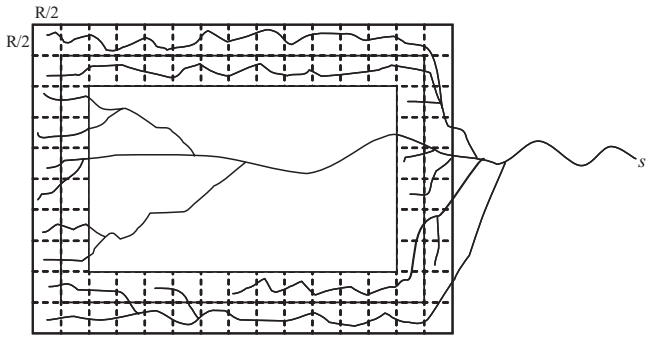


Fig. 5.9 Illustration of the entrance-zone-multicasting-based mechanism.

decomposed into multicasting from the source toward the centers of each zone, and then flooding from the nodes in these zones inwards.

5.6 SUMMARY AND FUTURE DIRECTIONS

Broadcasting, multicasting, and geocasting are fundamental and useful operations in WSNs. This chapter introduced related fundamental concepts, discussed major challenges in the design of broadcasting, multicasting, and geocasting mechanisms, and presented an overview of major broadcasting, multicasting, and geocasting mechanisms for WSNs. Although a lot of research has been carried out in this area, some open issues still need further studies.

- *Scalability.* Scalability is a concern in the design of broadcasting, multicasting, and geocasting mechanisms in WSNs. The performance of a broadcasting, multicasting, or geocasting mechanism primarily depends on the way network state information is disseminated, gathered, and maintained, as well as the accuracy of the information. In general, fundamental trade-offs in efficient broadcasting, multicasting, and geocasting exist between the communications overhead for disseminating state information and maintaining the information, as well as its inaccuracy and performance in terms of global resources utilization, for example, power utilization. The design of such mechanisms should also consider constraints posed by the applications of WSNs. In addition, how to adaptively manage a robust and efficient backbone structure in dynamic networks deserves further attention. Such a structure can greatly ease efficient broadcasting, multicasting, and geocasting.
- *Reliability.* Reliable transmission requires certain packet delivery redundancy while efficient broadcasting, multicasting, and geocasting try to maximally reduce such redundancy, which are contradictory. How to achieve a good trade-off between efficiency and redundancy is an open issue and

needs further study. This study should consider the availability of many low-quality links. Experiment results in Ref. [45] have shown that in some environments many links have a loss probability >50%.

- *QoS.* Quality of service (QoS) is a big concern for many applications of WSNs. Many existing broadcasting, multicasting, and geocasting mechanisms target on efficient resource utilization, but do not consider end-to-end QoS (e.g., delay and packet loss) at all. To support QoS in broadcasting, multicasting, and geocasting, location information can be used in the protocol design, which should address the location inaccuracy issue. In general, QoS constrained broadcasting, multicasting, and geocasting are NP-Complete. Heuristics are required to support such operations in large wireless sensor networks.

To enable efficient broadcasting, multicasting, and geocasting in WSNs, future research may consider the following couple of directions.

- *Cross-Layer Design.* The efficiency of a broadcasting, multicasting, or geocasting mechanism depends on many factors, for example, node's transmission range [46], channel characteristic, packet size, MAC [47], and routing. These factors can affect the link bit error rate, how a wireless channel is accessed, how a network is connected, and so on. In designing broadcasting, multicasting, and geocasting mechanisms, it is necessary to consider multiple different factors in an integrated manner in order to achieve high efficiency. Therefore, cross-layer design has been widely considered a promising design approach in the future. For more details regarding cross-layer design in WSNs, the reader is referred to Ref. [48].
- *Network Coding.* Network coding [49] has recently emerged as a new coding paradigm that has demonstrated a wide range of potential applications for improving network performance in multihop wireless communication networks. In traditional store-and-forward networks, an intermediate network node (or router) simply forwards data packets it receives. In contrast, network coding allows the information (or data) received from multiple links to be combined at intermediate network nodes for subsequent transmissions so that the amount of data transmitted in the network is reduced and the network performance is improved. Among its wide range of potential applications, network coding has proved to be a promising technique for providing efficient multicasting and broadcasting for improving energy efficiency and/or bandwidth utilization in WSNs [50–52].

ACKNOWLEDGMENTS

The work of Baoxian Zhang is supported partially by the National High-Tech Research and Development Plan of China under Grant No. 2006AA01Z207-1,

the Knowledge Innovation Program of Chinese Academy of Sciences under Grant Nos. KGCX2-YW-110-1, KGCX2-YW-110-6, and the National Natural Science Foundation of China under Grant No. 60702074. The work of Guoliang Xue is supported in part by NSF grants CCF-0431167 and CNS-0524736. Baoxian Zhang is with Graduate University of Chinese Academy of Sciences and Key Lab of Wireless Sensor Networks and Communications at Shanghai Institute of Microsystem and Information Technology of Chinese Academy of Sciences.

REFERENCES

[1] R. C. Prim, "Shortest connection networks and some generalizations", *Journal of Bell Systems Technology*, vol. 36, 1957, pp. 1389–1401.

[2] S. Ni, Y. Tseng, Y. Chen, and J. Sheu, "The broadcast storm problem in a mobile ad hoc network", in *Proceedings of ACM MOBICOM'99*, Seattle, WA, Aug. 1999, pp. 151–162.

[3] L. Li, J. Halpern, and Z. Haas, "Gossip-based ad hoc routing", in *Proceedings of IEEE INFOCOM'02*, New York, June 2002, pp. 1707–1716.

[4] W. Peng and X. Lu, "AHBP: An efficient broadcast protocol for mobile ad hoc networks", *Journal of Computer Science and Technology*, vol. 16, no. 2, Mar. 2001, pp. 114–125.

[5] J. Wu and F. Dai, "Broadcasting in ad hoc networks based on self-pruning", in *Proceedings of IEEE INFOCOM'03*, San Francisco, CA, Mar./Apr. 2003, pp. 2240–2250.

[6] W. Peng and X. Lu, "AHBP: An efficient broadcast protocol for mobile ad hoc networks", *Journal of Computer Science and Technology*, vol. 16, no. 2, 2001, pp. 114–125.

[7] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint relaying for flooding broadcast messages in mobile wireless networks", in *Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)*, Big Island, HI, Jan. 2002, pp. 3866–3875.

[8] J. Wu and H. Li. "On calculating connected dominating sets for efficient routing in ad hoc wireless networks", in *Proceedings of the International Workshop on Discrete Algorithms and methods for Mobile Computing and Communication (DIAL-M)*, Seattle, WA, Aug. 1999, pp. 7–14.

[9] I. Stojmenovic, M. Seddigh, and J. Zunic, "Dominating sets and neighbor elimination based broadcasting algorithms in wireless networks", *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 1, Jan. 2002, pp. 14–25.

[10] B. Williams and C. Tracy, "Comparison of broadcasting techniques for mobile ad hoc networks", in *Proceedings of ACM MOBIHOC'02*, Lausanne, Switzerland, June 2002, pp. 194–205.

[11] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: A survey", *IEEE Wireless Communications*, vol. 11, no. 6, Dec. 2004, pp. 6–28.

[12] D. Niculescu, "Positioning in ad hoc sensor networks", *IEEE Network Magazine*, vol. 18, no. 4, Aug. 2004, pp. 24–29.

[13] M.-T. Sun, W. Feng, and T.-H. Lai, "Location aided broadcast in wireless ad hoc networks", in *Proceedings of IEEE GLOBECOM'01*, San Antonio, TX, Nov. 2001, pp. 2842–2846.

[14] Y. Xu, J. Heidemann, and D. Estrin, "Geography informed energy conservation for ad hoc routing", in *Proceedings of ACM MOBICOM'01*, Rome, Italy, July 2001, pp. 70–84.

[15] B. Zhang and H. T. Mouftah, "Efficient grid-based routing for wireless multi-hop networks", in *Proceedings of IEEE ISCC'05*, Cartagena, Spain, June 2005, pp. 367–372.

[16] A. Clementi, P. Crescenzi, P. Penna, G. Rossi, and P. Vocca, "On the complexity of computing minimum energy consumption broadcast subgraphs", in *Proceedings of the 18th Annual Symposium on Theoretical Aspects of Computer Science, LNCS 2010*, Feb. 2001, pp. 121–132.

[17] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, "On the construction of energy-efficient broadcast and multicast trees in wireless networks", in *Proceedings of IEEE INFOCOM'00*, Tel Aviv, Israel, Mar. 2000, pp. 585–594.

[18] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, "Energy-efficient broadcast and multicast trees in wireless networks", *Mobile Networks (MONET)*, vol. 7, no. 6, Dec. 2002, pp. 481–492.

[19] J. Park, S. Sahni, "Maximum Lifetime Broadcasting in Wireless Networks", *IEEE Transactions on Computers*, vol. 54, no. 9, Sept. 2005, pp. 1081–1090.

[20] C. P. Low and L. W. Goh, "On the construction of maximum residual energy resource broadcast trees with minimum diameter in static ad hoc wireless networks", *Wiley International Journal of Communication Systems*, vol. 19, no. 1, Feb. 2006, pp. 39–51.

[21] J. Cartigny, F. Ingelrest, D. Simplot-Ryl, and I. Stojmenovic, "Localized LMST and RNG based minimum energy broadcast protocols in ad hoc networks", *Ad Hoc Networks*, vol. 3, no. 1, Jan 2005, pp. 1–16.

[22] N. Li, J. C. Hou, and L. Sha, "Design and Analysis of an MST-Based Topology Control Algorithm", in *Proceedings of IEEE INFOCOM* 2003, San Francisco, CA, Apr. 2003, pp. 1702–1712.

[23] N. Li and J. C. Hou, "A scalable, power-efficient broadcast algorithm for wireless networks", *Wireless Networks (WINET)*, vol. 12, no. 4, July 2006, pp. 495–509.

[24] K. Tang and M. Gerla, "Reliable multicast of the on-demand multicast routing protocol", in *Proceedings of SCI* 2001, Orlando, FL, July 2001.

[25] W. Lou and J. Wu, "A reliable broadcast algorithm with selected acknowledgements in mobile ad hoc networks", in *Proceedings of IEEE Globecom'03*, San Francisco, CA, Nov. 2003, pp. 3536–3541.

[26] V. Rajendran, K. Obraczka, and J. J. Gracia-Luna-Aceves, "Energy efficient, collision-free medium access control for wireless sensor networks", *Wireless Networks*, vol. 12, no. 1, Feb. 2006, pp. 63–78.

[27] I. Demirkol, C. Ersoy, F. Alagoz, "MAC protocols for wireless sensor networks: a survey", *IEEE Communications Magazine*, vol. 44, no. 4, Apr. 2006, pp. 115–121.

[28] J. J. Garcia-Luna-Aceves and E. L. Madruga, "The cost-assisted mesh protocol", *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, Aug. 1999, pp. 1380–1394.

[29] K. Chen and K. Nahrstedt, "Effective location-guided tree construction algorithms for small group multicast in MANET", in *Proceedings of IEEE INFOCOM'02*, New York, NY, June 2002, pp. 1180–1189.

[30] Ki-Il Kim and Sang-Ha Kim, "A novel overlay multicast protocol in mobile ad hoc networks: design and evaluation", *IEEE Transactions on Vehicular Technology*, vol. 54, no. 6, Nov. 2005, pp. 2094–2101.

[31] L. S. Ji and S. Corson, "Differential destination multicast—A MANET multicast routing protocol for small groups", in *Proceedings of IEEE INFOCOM'01*, Anchorage, AK, Apr. 2001, pp. 1192–1202.

[32] T. Ozaki, Bae Kim Jaime, and T. Suda, "Bandwidth-efficient multicast routing for multihop ad-hoc wireless networks", in *Proceedings of IEEE INFOCOM'01*, Anchorage, AK, Apr. 2001, pp. 1182–1191.

[33] E. M. Royer and C. E. Perkins, "Multicast operation of the ad-hoc on-demand distance vector routing protocol", in *Proceedings of ACM MOBICOM'99*, Seattle, WA, Aug. 1999, pp. 207–218.

[34] S. E. Deering and D. R. Cheriton, "Multicast routing in datagram internetworks and extended LANs", *ACM Transactions on Computer System*, vol. 8, no. 2, May 1990, pp. 85–110.

[35] H. S. Kim, T. F. Abdelzaher, and W. H. Kwon, "Minimum-energy asynchronous dissemination to mobile sinks in wireless sensor networks", in *Proceedings of IEEE Sensys'03*, Los Angeles, CA, Nov. 2003, pp. 193–204.

[36] J. Sanchez, P. Ruiz, X. Liu, and I. Stojmenovic, "GMR: Geographic multicast routing for wireless sensor networks", in *Proceedings of IEEE SECON'06*, Reston, VA, Sept. 2006, pp. 20–29.

[37] H. Luo, F. Ye, J. Cheng, S. Lu, and L. Zhang, "TTDD: Two-tier data dissemination in large-scale wireless sensor networks", *Wireless Networks*, vol. 11, no. 1–2, June 2005, pp. 161–175.

[38] C. Maihofer, W. Franz, and R. Eberhardt, "Stored geocast", in *Proceedings of Kommunikation in Verteilten Systemen (KiVS)*, Leipzig, Germany, Springer Verlag, Feb. 2003, pp. 257–68.

[39] B. Karp and H. T. Kung, "Greedy perimeter stateless routing for wireless networks", in *Proceedings of ACM MobiCom'00*, Boston, MA, Aug. 2000, pp. 241–54.

[40] Y. B. Ko and N. H. Vaidya, "Flooding-based geocasting protocols for mobile ad hoc networks", *Mobile Networks and Applications*, vol. 7, no. 6, Dec. 2002, pp. 471–280.

[41] C. Maihofer, D. AG, "A survey of geocast routing protocols", *IEEE Communications Surveys and Tutorials*, vol. 6, no. 2, Apr. 2004, pp. 32–42.

[42] I. Stojmenovic, "Geocasting with guaranteed delivery in sensor networks", *IEEE Wireless Communications Magazine*, vol. 11, no. 6, Dec. 2004, pp. 29–37.

[43] E. Kranakis, H. Singh, and J. Urrutia, "Compass routing on geometric networks", in *Proceedings of the 11th Canadian Conference on Computational Geometry*, Vancouver, Canada, Aug. 1999, pp. 51–54.

[44] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia, "Routing with guaranteed delivery in ad hoc wireless networks", in *Proceedings of ACM DIAL-M'99*, Seattle, WA, Aug. 1999, pp. 48–55.

[45] J. Zhao and R. Govindan. "Understanding packet delivery performance in dense wireless sensor networks", in *Proceedings of IEEE Sensys'03*, Los Angeles, CA, Nov. 2003, pp. 1–13.

[46] Z. M. Zuniga, B. Krishnamachari, "Optimal transmission radius for flooding in large scale wireless sensor networks", *Journal of Cluster Computing*, vol. 8, no. 2–3, July 2005, pp. 167–178.

[47] Y. Sankarasubramanian, I. F. Akyildiz, and S. W. McLaughlin, "Energy efficiency based packet size optimization in wireless sensor networks", in *Proceedings of the 1st IEEE International Workshop on Sensor Network Protocols and Applications (SNPA'03)*, Anchorage, AK, May 2003, pp. 1–8.

[48] Raja Jurdak, *Wireless Ad Hoc and Sensor Networks: A Cross-Layer Design Perspective (Signals and Communication Technology)*, Springer, New York, 2007.

[49] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow", *IEEE Transaction on Information Theory*, vol. 46, no. 4, July 2000, pp. 1204–1216.

[50] Y. Wu et al., "Minimum-energy multicast in mobile ad hoc networks using network coding", *IEEE Transactions on Communications*, vol. 53, no. 11, Nov. 2005, pp. 1906–1918.

[51] J. Zhang, P. Fan, and K. B. Letaief, "Network coding for efficient multicast routing in wireless ad-hoc networks", *IEEE Transactions on Communications*, vol. 56, no. 4, Apr. 2008, pp. 598–607.

[52] D. Nguyen, T. Nguyen, and B. Bose, "Wireless broadcasting using network coding", in *Proceedings of NetCod'07*, San Diego, CA, Jan. 2007.

6

NODE CLUSTERING

Chao Zhang, Edwin Hou, and Nirwan Ansari

New Jersey Institute of Technology, USA

6.1 INTRODUCTION

Wireless sensor networks (WSNs) have gained increasing importance in a variety of military and civilian applications. With recent advances in wireless communications technologies, WSNs represent a great leap forward over traditional sensor networks [1].

A WSN consists of a large number of sensor nodes, which are densely deployed over an unattended area either close to or inside the targets to be observed. These sensor nodes periodically monitor or sense the conditions of the targets, process the data, and transmit the sensed data back to a base station. All of the sensor nodes collaborate together to form a communication network for providing reliable networking service. The collaboration among sensor nodes is very important in WSNs for two reasons:

1. Data collected from multiple sensor nodes can offer valuable inference about the environment.
2. The collaboration among sensor nodes can provide trade-offs between communication cost and computation energy.

Wireless Sensor Networks: A Networking Perspective, Edited by Jun Zheng and Abbas Jamalipour

Since it is likely that the data acquired from one sensor node are highly correlated with the data from its neighbors, data aggregation can reduce the redundant information transmitted in the network. It is well known that the energy consumed for transferring one bit of data can be used to perform a large number of arithmetic operations in a sensor processor (power consumed for transferring one bit of data to a receiver 100 m away is equal to that needed to execute 3,000 instructions [2]). When the base station is far away, there are significant advantages in using local data aggregation instead of direct communication. Thus, node clustering, which aggregates nodes into groups (clusters), is critical to facilitating practical deployment and operation of WSNs.

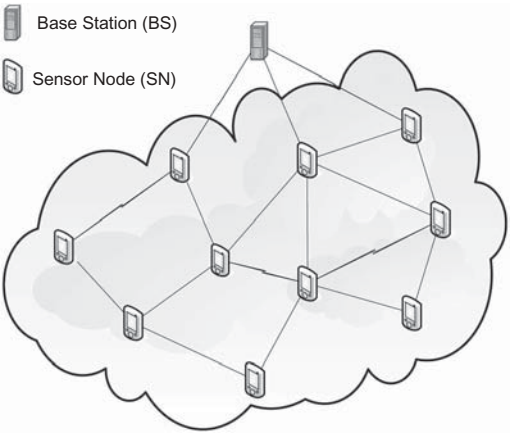
In this chapter, the major issues and challenges in node clustering for WSNs are discussed and a variety of state-of-the-art clustering algorithms are introduced. The remainder of this chapter is organized as follows. In Section 6.1, the common architectures, as well as the node clustering structures in WSNs, are introduced. Section 6.2 introduces general clustering techniques and the major node clustering algorithms for ad hoc networks. Section 6.3 describes the specialties for clustering in WSNs and introduces the major node clustering algorithms for WSNs. Section 6.4 concludes with a brief summary of the chapter and future directions.

6.1.1 Wireless Sensor Network Architectures

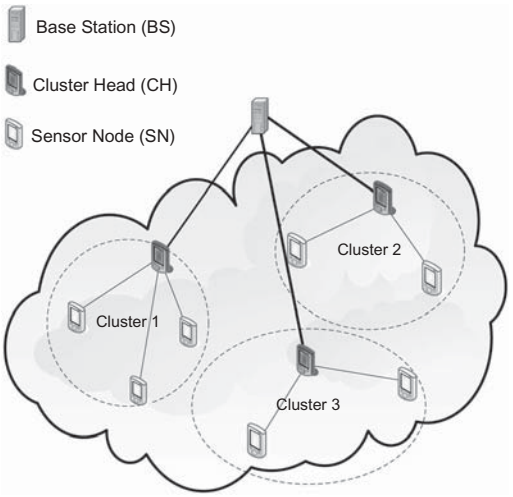
According to the way that data are collected, WSNs can be classified into three types: homogenous sensor networks, heterogeneous sensor networks, and hybrid sensor networks [3].

6.1.1.1 Homogenous Sensor Networks. A homogenous network consists of base stations and sensor nodes equipped with equal capabilities, for example, computational power and storage capacity. Data gathering in this type of networks is based on the structure of data dissemination. Flat and hierarchical topologies are two representative structures being widely studied for data dissemination and gathering in homogenous networks [4].

- In a flat network, data aggregation is accomplished by data-centric routing where the base station usually transmits a query message to the sensor nodes via flooding, and the sensor nodes that have data matching the query will send response messages back to the base station. The sensor nodes communicate with the base station via multihop routes by using peer nodes as relays. The choice of a particular communication protocol depends on the specific application [5]. Figure 6.1a illustrates the architecture of a flat network. The sensor nodes are assumed to be stationary once being distributed in the targeted area and the collected sensing information is gathered at the base station.
- In a hierarchical network, sensor nodes are organized into clusters where the cluster heads serve as simple relays for transmitting the data. Since the



(a)



(b)

Fig. 6.1 Network topology: (a) flat topology and (b) hierarchical topology.

cluster heads have the same transmission capacity as the sensor nodes, the minimum requirement on the number of clusters can be derived from the upper bound of the throughput. Higher throughput can be achieved by using clustering at the cost of having extra nodes functioned as cluster heads. Data aggregation in a hierarchical network involves data fusion at cluster heads, which reduces the number of messages transmitted to the base station, and hence improves the energy efficiency of the network. A typical structure of a hierarchical network is shown in Fig. 6.1b.

6.1.1.2 Heterogeneous Sensor Networks. A heterogeneous sensor network consists of base stations (fixed and mobile), sensor nodes, and sophisticated sensor nodes with advanced embedded processing and communicating capabilities as compared to normal sensor nodes. Data gathering can be executed at the mobile base stations [6]. In such networks, mobile base stations move randomly in the area of the deployed network, collecting data directly from normal sensor nodes, or use some surrounding sensor nodes to relay the data (see Fig. 6.2). Sometimes, sensor nodes may be distributed sparsely and the distance between any two sensor nodes can be far apart. The long distance among sensor nodes implies that more energy will be consumed for communication. Meanwhile, sensor nodes need to perform sensing and communication for as long as possible. As shown in many experimental results, data gathering with mobile sinks is able to prolong the lifetime of the system [7].

6.1.1.3 Hybrid Sensor Networks. In a hybrid sensor network, several mobile base stations work cooperatively to provide fast data gathering in a

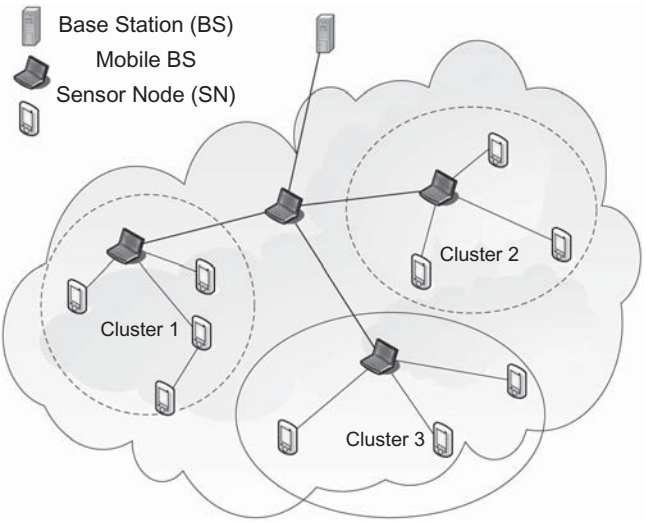


Fig. 6.2 Heterogeneous sensor network topology.

real-time manner. In the scenario shown in Fig. 6.3, collected data will be relayed by several mobile base stations. The conventional and well-studied routing algorithms for ad hoc networks can be adopted as the routing protocols among these mobile base stations. Mobile ad hoc networks (MANETs) assume that every node is able to move at their own pace. Even though WSNs are more constrained than other wireless networks, for example, MANETs, in terms of energy, processing, transmission range, and bandwidth, routing from a source base station to a destination base station can be accomplished by using MANET protocols in hybrid sensor networks [5]. Accordingly, if the location of a base station is unpredictable or in case the base stations cannot communicate with each other on their own, it is reasonable to tailor techniques originally proposed for MANETs and apply them in WSNs. Hybrid sensor networks can achieve longer lifetime and can also improve the efficiency of data gathering [6,8]. As pointed out in Ref. [9], a mobile base station prefers the hybrid architecture, by which a mobile base station can communicate with other sensor nodes by using a WSN protocol and with other base stations by using a MANET protocol.

While individual sensor nodes are not as powerful as normal computers, a large number of sensor nodes are required to provide high quality and reliable networking service, as well as easy deployment and fault tolerance in inaccessible environments where maintenance is inconvenient or impossible. Such unique operating environments and performance requirements of WSNs require fundamentally new approaches to networking design.

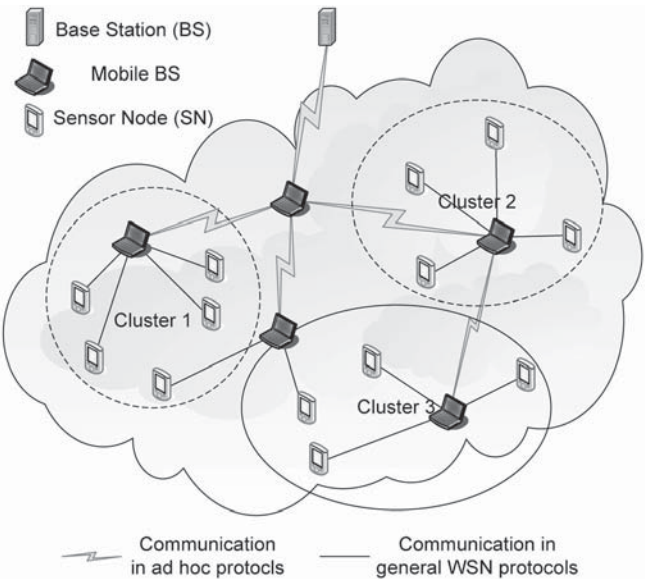


Fig. 6.3 Hybrid sensor network topology.

6.1.2 Node Clustering Structures

Clustering mechanisms have been applied to sensor networks with hierarchical structures to enhance the network performance while reducing the necessary energy consumption [10]. Clustering is a cross-cutting technique that can be used in nearly all layers of the protocol stack. The primary idea is to group nodes around a cluster head that is responsible for state maintenance and intercluster connectivity.

In conventional cellular networks, fixed base stations are connected through wired backbones. Communications between two mobile nodes that are only 1-hop away from their respective base stations can be established through the fixed base stations and the wired backbone. In this case, clustering is used to select and allocate channel groups to all the base stations within a system and to achieve efficient frequency reuse (see Fig. 6.4). In multihop wireless networks, node clustering is a technique that aggregates nodes into groups (clusters) to reduce the routing overhead and to provide a convenient framework for efficient resource (e.g., bandwidth or code) allocation, energy management, fault-tolerant routing, and high end-to-end throughput.

In clusters without any cluster head, a proactive strategy is used for intracluster routing while a reactive strategy is used for intercluster routing. However, as the network size grows, there will be heavy traffic overhead within the network [11]. Therefore, normally one node is selected as the cluster head of a cluster, and it acts as the local coordinator of transmissions within its cluster. A hierarchical routing or network management protocol can be more efficiently implemented with cluster heads. As compared to the base stations used in current cellular

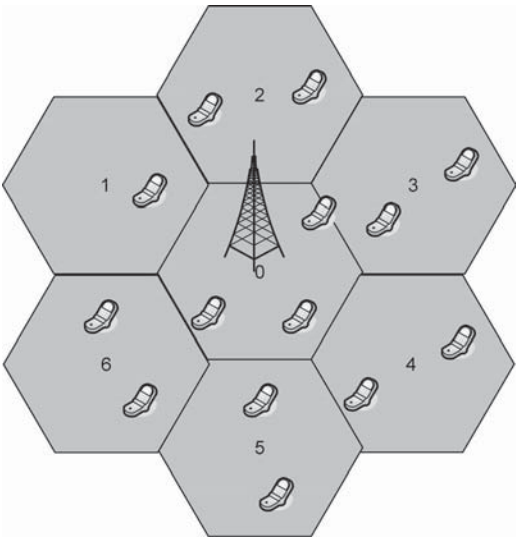


Fig. 6.4 Cluster structure in cellular system.

systems, the cluster head does not have any special hardware, and is in fact dynamically selected among the set of nodes. However, a cluster head performs additional functions as a central administration point, and a cluster-head failure would degrade the performance of the entire network; it may become the bottleneck of the cluster. An efficient node-clustering mechanism tends to preserve its structure when a few nodes are moving and the topology is slowly morphing. The objective of the node clustering procedure is to find a feasible interconnected set of clusters that covers the entire node population.

For the initial deployment of the network, the nodes could be deployed in the coverage area regularly or randomly.

6.1.2.1 Regularly Placed Nodes Deployment. If the area to be deployed is easily accessible and sensor nodes can be placed anywhere, regular placement will allow the best possible coverage and easier clustering of the sensor nodes. To cover a given area A, assuming that the distance between any adjacent nodes is given by R in all cases, the coverage area of each node and the total numbers of nodes required for three common placement types, triangular, rectangular, and hexagonal clusters are given in Table 6.1.

If each subregion can be covered by more than one sensor node, some selected sensor nodes can be allowed to go into the sleep state.

6.1.2.2 Randomly Distributed Nodes Deployment. The nodes to be deployed are randomly distributed in an unknown or inaccessible area with unmanned devices or airplanes. In this scenario, the nodes have to discover their neighbors by themselves. If N nodes are uniformly distributed over an area A, the node density can be given by $\lambda = N/A$ [12]. The probability that there are m nodes within the area S is Poisson distributed and can be given by

$$P(m) = \frac{(\lambda S)^m}{m!} e^{-\lambda S}. \tag{6.1}$$

TABLE 6.1 Coverage Area by Each Node and Total Required Number of Nodes for a Given Area A

Cluster Shape	Distance between Adjacent Nodes	Coverage Area by Each Node	Total Required Number of Nodes per Coverage Area
Triangular	R	$\frac{\sqrt{3}}{4}R^2$	$\left\lceil \frac{4A}{\sqrt{3}R^2} \right\rceil$
Rectangular	R	R^2	$\left\lceil \frac{A}{R^2} \right\rceil$
Hexagonal	R	$\frac{3\sqrt{3}}{4}R^2$	$\left\lceil \frac{4A}{3\sqrt{3}R^2} \right\rceil$

The probability that the monitored area has one node can be expressed as

$$P_{1-\text{node}} = 1 - P(0) = 1 - e^{-\lambda S}. \tag{6.2}$$

In many situations, it needs at least k nodes cooperating within an area to ensure reliable service. The probability of having k nodes in a given area S can be expressed by

$$P_{k-\text{nodes}} = 1 - \sum_{m=0}^{k-1} P(m) = 1 - \sum_{m=0}^{k-1} \frac{(\lambda S)^m}{m!} e^{-\lambda S}. \tag{6.3}$$

The sensing and communication ranges in a randomly distributed nodes deployment are determined by the maximum distance between any two adjacent nodes in the given area. Several heuristic deployment schemes have been discussed in Ref. [13].

To optimally group or cluster a large number of nodes is typically an NP-hard problem, which requires searching through a tremendously large number of possible solutions to find one approximate solution [14]. Therefore, node-clustering algorithms are all heuristic based. In the following sections, some popular clustering algorithms are introduced. Most of these algorithms are studied in the context of ad hoc wireless network architectures.

6.2 NODE CLUSTERING ALGORITHMS

A communication network can be modeled as an undirected graph $G = (V, E)$, where V is the set of vertices representing the nodes in the network, and E is the set of edges representing the communication links between two adjacent nodes [15]. The objective of a clustering algorithm is to divide the graph G into an interconnected set of subgroups.

A subset of nodes V_D is called a D-hop dominating set if every node in V is at most D ($D > 1$) hops away from a node in V_D. In a D-hop cluster, each node is at most D-hops away from the cluster-head. An example structure of a 2-hop cluster is shown in Fig. 6.5.

Node clustering algorithms are usually performed in two phases: node-clustering setup and clustering maintenance. In the node-clustering setup phase, cluster heads are chosen among the nodes in the network. After electing the cluster heads, other nodes affiliated with the cluster heads would form the clusters. Nodes that are not cluster heads are called ordinary nodes. Nodes at the fringe of a cluster are called *gateway* nodes, which typically communicate with gateway nodes of other clusters, for example, nodes 8 and 14, shown in Fig. 6.5. In the clustering maintenance phase, the clustering configuration may be changed after the initial cluster is set up due to node movements or topology changes.

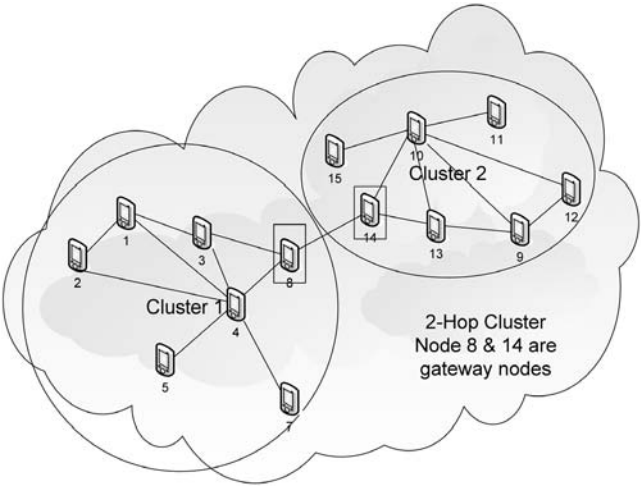


Fig. 6.5 Clustering structure of 2-hop clusters.

Conventional clustering algorithms face limitations in a real wireless ad hoc environment because of unreliable and limited link capacity, and changes of the node topology. Many heuristic clustering algorithms have been proposed to overcome these limitations by properly selecting the cluster heads and avoiding excessive computation in the cluster maintenance phase.

6.2.1 Cluster-Head Election Algorithms

Properly selecting the cluster heads can lower the rate for refreshing clusters and therefore reduce the overhead in the ad hoc environments. In this section, four common clustering algorithms for electing the cluster heads will be discussed. According to their specific situations and applications, most node clustering algorithms adopt one of these cluster-head selecting algorithms.

6.2.1.1 Lowest ID Clustering Algorithm. The lowest ID (LID) clustering algorithm is a 2-hop clustering algorithm [16]. While executing this algorithm, a node periodically broadcasts the list of nodes that it can hear (including itself). A node, which only hears the nodes with IDs higher than itself from the 1-hop neighborhood, declares itself as the cluster head. It then broadcasts its ID and cluster ID. A node that can hear two or more cluster heads is a gateway node; otherwise, it is an ordinary node or a cluster head. Figure 6.6 shows an example of the LID structure, in which nodes 1, 6, and 10 are selected as cluster heads because they have the lowest ID numbers in their respective clusters. Nodes 5 and 13 are gateway nodes. Simulation results showed that the LID algorithm is more stable in an environment in which the network topology changes frequently [17].

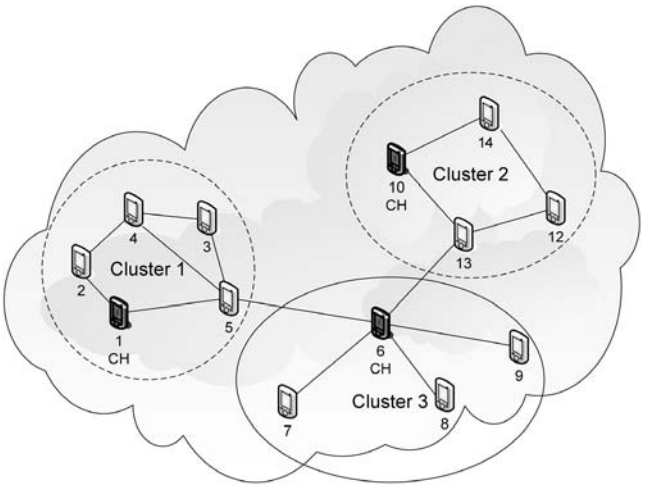


Fig. 6.6 Clustering by using the LID algorithm.

6.2.1.2 Highest Connectivity Clustering Algorithm. The highest connectivity (HCN) clustering algorithm elects the node with the highest connectivity (degree) in a neighborhood as the cluster head [18]. The connectivity of a node is the number of links to its 1-hop neighbors. Each node broadcasts the list of nodes that it can hear (including itself). In the case of a tie, the LID node is chosen as the cluster head. A node, which has already elected another node as its cluster head, gives up its role as the cluster head. Figure 6.7 shows the result of applying HCN to the same node topology used in Fig. 6.6. In cluster 1, nodes 4 and 5 both have three connections; however, node 4 is selected as the cluster head. Node 13 is selected as the cluster head in cluster 3 because it has the highest connectivity. As compared to the LID algorithm, HCN incurs a higher message overhead because more information about connectivity is exchanged. However, the cluster heads will change more frequently, and therefore the load distribution is fairer.

6.2.1.3 Least Cluster Change Algorithm. The least cluster change (LCC) algorithm is proposed to minimize the frequency of cluster-head change, where cluster stability is a major consideration under certain circumstances [19]. In the LCC algorithm, the cluster heads may change only under either one of these two conditions:

- Two cluster heads come within the transmission range of each other.
- A node loses its membership in any other cluster and forms a new cluster.

When it needs to form initial clusters or reselect cluster heads, LCC will use the LID or HCN clustering algorithm. Since the changes of cluster heads are

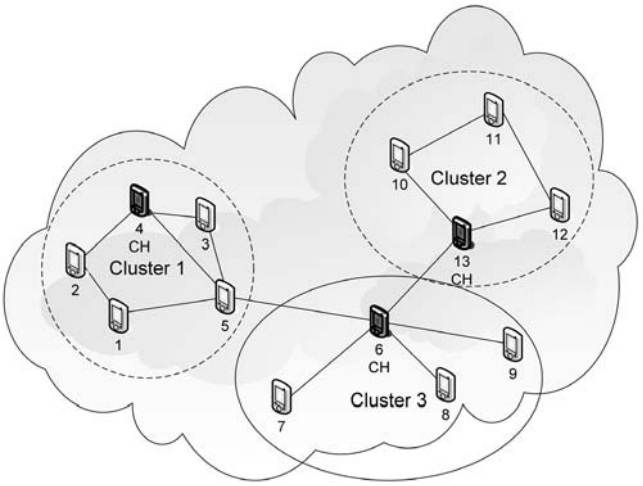


Fig. 6.7 Clustering by using the highest connectivity algorithm.

minimized, the cluster structures will not change frequently when nodes join or leave the clusters. The LCC clustering algorithm is robust in an environment in which the network topology changes frequently, and has low routing overhead and latency. However, the load distribution would be unfair for all nodes.

6.2.1.4 Weighted Clustering Algorithm. The weighted clustering algorithm (WCA) is based on a combined weight metric, which includes one or more parameters, for example, the node degree, distances with respect to a node's neighbors, node speed, and the time spent as a cluster head [20]. Each node broadcasts its weight value to all other nodes. A node is chosen to be a cluster head if its weight is the highest among its neighbors; otherwise, it joins a neighboring cluster. In the event of a tie, the LID algorithm is applied. Basically, a node has to wait for all the responses from its neighbors to make its own decision, and as a result, the latency and the overhead induced by WCA are very heavy.

None of the above heuristics algorithms leads to an optimal election of cluster heads because each deals with only a subset of the parameters that can possibly impose constraints on the network. Each of these heuristics is suitable for a specific application rather than for generic wireless mobile networks.

6.2.2 Node Clustering Algorithms in Ad Hoc Networks

Clustering mechanisms in wireless ad hoc networks have been investigated in the past in order to enhance network manageability, channel efficiency, and energy economy. Moreover, clustering is indispensable for hierarchical routing or multicasting. Many heuristic-based node clustering algorithms have been proposed and several of the more popular algorithms will be discussed in this section.

6.2.2.1 Linked Cluster Algorithm. The linked cluster algorithm (LCA) [21] was proposed to organize radio-equipped mobile nodes into a reliable network structure and maintain this structure for arbitrary topological changes. All nodes in the network are organized into a set of node clusters and each node belongs to at least one cluster. Every cluster has its own cluster head, which acts as a local controller for the other nodes in the cluster. The cluster heads are linked via gateway nodes to connect the neighboring clusters and to provide global network connectivity. LCA is a distributed algorithm and does not depend on the existence of any particular node. The algorithm consists of two steps: formation of clusters and linking of the clusters. Upon the completion of LCA, each node will become a normal node, a gateway node, or a cluster head. Each node maintains the following data structures as shown in Fig. 6.8:

- *Heads_one_hop_away* is a list recording those cluster heads that are connected to a node (1-hop away).
- *Heads_two-hops_away* is a list of the cluster heads that are not directly connected but connected to the neighbors of a node (2-hops away).
- *Nodes_heard* is a list that includes all neighbor nodes.

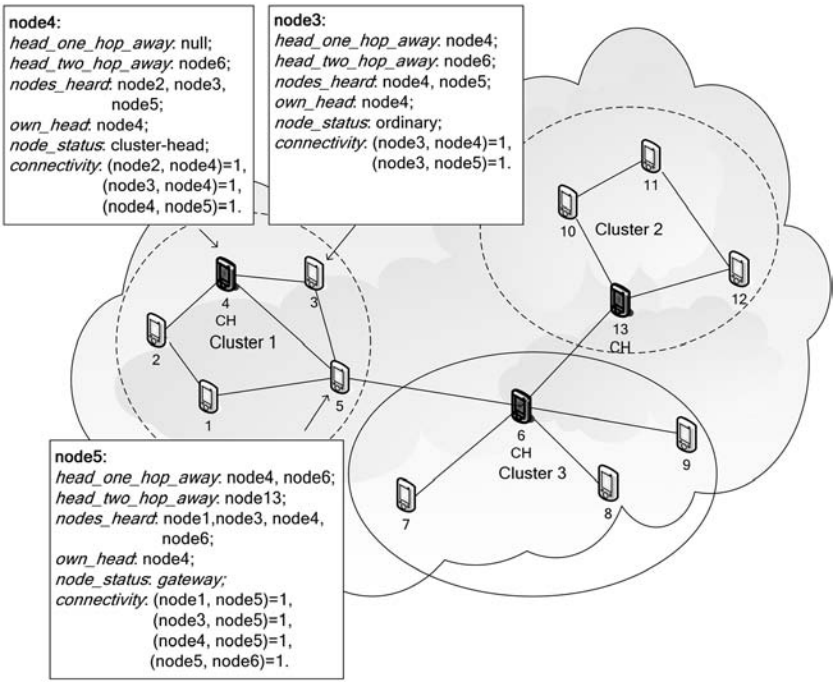


Fig. 6.8 Clustering by the LCA.

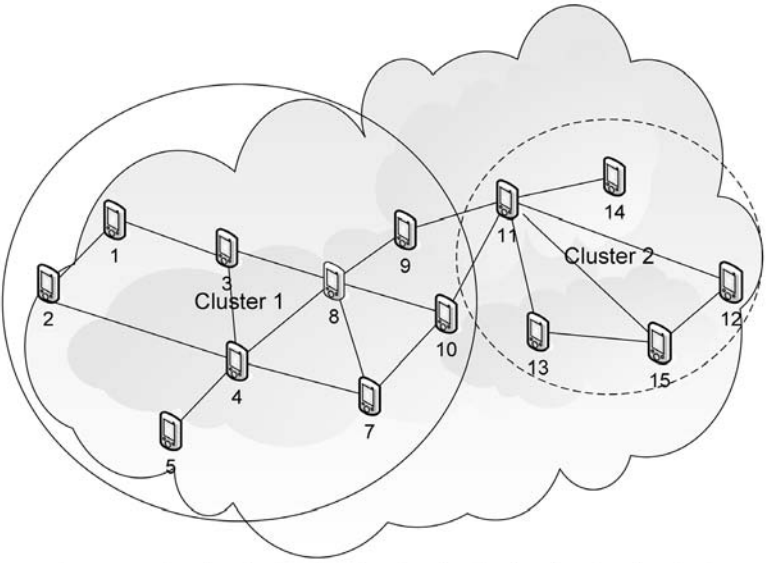
- *Node_status* indicates the status (a normal node, a gateway node, or a cluster head) of itself.
- *Own_head* is the identity of the cluster head of the given node.
- *Connectivity* is a matrix having binary entries. A value of 1 at the (*i*, *j*) position indicates that there is a link between nodes *i* and *j*, while 0 indicates no connection.

These data structures are updated regularly as control messages are received from other nodes. Each node broadcasts the *Nodes_heard* list and its full connectivity row from the *Connectivity* matrix. A node decides to become a cluster head if it has the highest ID among its neighboring nodes, or if it has the highest ID in the neighborhood of one of its neighbors. This highest ID linked cluster algorithm yields poor clustering when the nodes are arranged in the order of their identities; that is, all but one node becomes a cluster head. Therefore, this linked cluster mechanism was later revised in [22] to use the LID mechanism. The disadvantage of both of these linked cluster mechanisms is that the cluster-head load is not uniformly distributed among all the nodes. The higher (or lower in LID) the node-ID the more likely it is to become a cluster head.

Another limitation of LCA is its relatively high control message overhead because the nodes have to broadcast their *Nodes_heard* list. Also, LCA does not consider the node mobility, adaptive transmission range, and power efficiency issues.

6.2.2.2 Max–Min D-Clustering Algorithm. The max–min D-clustering (MMD) algorithm proposed in Ref. [23] uses a load-balancing (max–min) heuristic to form D-hop clusters to ensure a fairly distributed load among cluster heads. In a *D*-hop cluster, each node is at most *D*-hops away from the cluster head.

The MMD algorithm has 2**D* rounds of message exchanges, and each node needs to maintain two arrays, *WINNER* and *SENDER*, of size 2**D* for the node IDs. The *WINNER* and *SENDER* arrays, respectively, store the winning node ID, and the node ID that was sent to the winning node ID in each round, as shown in Fig. 6.9. Initially, each node sets its ID to *WINNER*. The algorithm consists of four stages: *floodmax*, *floodmin*, determining cluster heads and linking clusters. The *floodmax* stage consists of *D* rounds of message exchanges of the *WINNER* value from each node to their 1-hop neighbors. During each round, each node broadcasts its present *WINNER* value to all of its 1-hop neighbors and updates the *WINNER* value with the largest value it receives. Therefore, *floodmax* propagates the largest node ID to its *D*-hops neighborhood, and the node ID left at the end is elected as the cluster head. However, this mechanism may result in an unbalanced loading for the cluster heads. After *floodmax*, *D* rounds of *floodmin* start to propagate smaller node IDs. In contrast to *floodmax*, each node updates its WINNER value with the smallest node IDs. Any node ID that appears at least



Node	1	2	3	4	5	7	8	9	10	11	12	13	14	15
floodmax1	3	4	8	8	5	10	10	11	11	14	15	15	14	15
floodmax2	8	8	10	10	8	11	11	15	15	15	15	15	15	15
floodmin1	8	8	10	10	8	8	11	15	15	15	11	11	15	11
floodmin2	8	8	8	8	8	8	11	8	8	11	11	11	11	11
Result	**8**	**8**	**8**	**8**	**8**	**8**	**8**	**8**	**8**	**11**	**11**	**11**	**11**	**11**

Fig. 6.9 Clustering by the max–min 2-clustering algorithm.

once as a WINNER in both stages at an individual node is called a *node pair*. At the end of *floodmin*, each node determines its cluster head based on the entries in WINNER for the 2**D* rounds of *floodmax* and *floodmin* according to the following three rules:

- *Rule 1*: If a node receives its own ID in the second round, it declares itself a cluster head. Otherwise, *Rule 2* applies.
- *Rule 2*: Among all node pairs, a node selects the minimum node pair to be the cluster head. If a node pair does not exist for a node, then *Rule 3* applies.
- *Rule 3*: The maximum ID node in the first round is selected as the cluster head for this node.

After determining the cluster heads, each node broadcasts its cluster head to all of its neighbors. If all the neighbors of a node have the same cluster head as itself, then this node is a normal node. If any neighbor has a different cluster head, then this node becomes a gateway node.

To establish the backbone of the network, the gateway nodes begin by linking all the nodes in the cluster to its cluster head and also linking its cluster head to

other clusters. The time complexity and the storage complexity of the algorithm are both $O(D)$. As compared to LCA, the MMD clustering algorithm has fewer cluster heads, larger cluster size, and better cluster-head stability. Multiple paths may exist with the backbone between neighboring cluster heads, and therefore this algorithm provides fault tolerance in the network backbone. However, the MMD clustering algorithm does not consider node mobility and power efficiency.

6.2.2.3 Mobility-Based Clustering Algorithm. The mobility-based clustering (MBC) algorithm creates clusters adaptively based on mobility concepts (individual mobility, group mobility) [24] and position information via a reliable position locating system (i.e., GPS) [25]. The MBC algorithm considers the combination of both physical (geographic proximity) and logical (functional relation between nodes) partitions of the network, as well as the relative mobility of a node with respect to its peers. However, MBC does not depend on any underlying unicast routing protocol. This is different from the mobility-based adaptive algorithm proposed in Ref. [26], which uses a probabilistic model to characterize the future availability of network links maintained by some underlying unicast proactive routing algorithm.

The MBC algorithm is a mobility-based hierarchical clustering algorithm and may generate variable-size clusters depending on the mobility characteristics of the nodes. A group may consist of clusters that present similar mobility characteristics. Several groups can be hierarchically merged into one group depending on the mobility of each group. A node with higher relative mobility is more prone to be unstable, and therefore this node should not be elected as a cluster head. To construct and maintain various clusters, the MBC algorithm executes the following steps:

- *Step* 1: *Mobility Information Dissemination.* Each node periodically broadcasts its moving information (speed and direction) to its neighboring nodes.
- *Step* 2: *Calculation of Mobility Metrics.* Upon reception of a node's moving information, each neighboring node calculates its relative mobility between itself and that node. In this way, the relative mobility between any two adjacent nodes is updated periodically.
- *Step* 3: *Initial* (*Tentative*) *Cluster Construction.* Each node compares its relative mobility with a mobility threshold Th_{mob}. A tentative cluster head (TCH) is elected from those nodes whose relative mobility is $<Th_{\text{mob}}$. The mobility threshold Th_{mob} is a design parameter that can be used to control the stability of the generated clusters in different networks.
- *Step* 4: *Cluster Merging.* If one cluster with TCH_1 is going to merge into another cluster with TCH_2 according to Step 3, then the child cluster joins the parent cluster with its current cluster members. The TCH_2 of the parent cluster is selected as the cluster head of the merged cluster. Each node in

the child cluster still holds the information of TCH_1. TCH_1 holds the information of its parent cluster head, and so on. The newly generated cluster head holds routing information of all nodes within the new cluster.

- *Step* 5: *Cluster Maintenance or Reconstruction.* When a node moves from one cluster into another cluster, if the relative mobility between this node and the current cluster head is $<Th_{\text{mob}}$, no clustering is changed; otherwise, these steps are repeated.

The MBC algorithm has been shown to significantly reduce cluster-head changes as compared to both LID and HCN under random movement [27]. MBC provisions mobility management and geomulticast functions for mobile ad hoc networks.

Sensor nodes are typically less mobile, more limited in capabilities, and more densely deployed than traditional ad hoc networks. Innovation and novelty are required to tailor these popular solutions originally designed to address some of the conventional wireless networking problems for WSNs. Some of the major algorithms specially designed for WSNs are introduced in Section 6.3.

6.3 NODE CLUSTERING ALGORITHMS FOR WIRELESS SENSOR NETWORKS

In the previous section, some of the more popular node clustering algorithms for traditional wireless ad hoc networks have been discussed. However, these clustering algorithms are not well suited for the unique features and application requirements of WSNs.

6.3.1 Specialties for Clustering in Wireless Sensor Networks

The differences between WSNs and traditional ad hoc networks are outlined below:

- The number of sensor nodes in a WSN can be several orders of magnitude higher than that in an ad hoc network.
- Sensor nodes are densely deployed.
- Sensor nodes are prone to failures.
- The topology of a WSN may change rather frequently because a sensor node may alternate between the active and sleep states.
- Sensor nodes mainly use broadcast communications, whereas most ad hoc networks are based on point-to-point communications.
- Sensor nodes are limited in power, computational capacities, and memory.
- Sensor nodes may not have global identification (ID) because of the large amount of overhead and the large number of sensors.

Since a large number of sensor nodes are densely deployed, multihop communications are prone to occur in WSNs [28–31]. As compared to traditional ad hoc networks, the transmission power levels can be kept low, and the communications consume less power in WSNs. One approach is to cluster a WSN into clusters such that all members of the clusters are directly connected to the cluster heads. Sensor nodes in the same cluster can communicate directly with their cluster head without any intermediate sensor nodes. Cluster heads can transmit gathered information back to the base station through multihop communication among cluster heads. Clustering of sensor nodes not only allows aggregation of sensed information, but also minimizes the energy consumed within individual clusters and reduces both the traffic and contention for channel clustering.

A D-hop cluster is defined as a cluster with all the sensor nodes in the cluster reachable by a path with path length $\leq D$ hops [32]. It is very important to determine an optimal value of D that minimizes the overall energy consumption in a WSN. To design an optimized mechanism, various factors, for example, the data packet size (the amount of data to be transferred within each cluster and between clusters), frequency of transmissions, maximum allowable latency, local computation processes, and maintenance of partial database information, must be considered. Since hundreds or even thousands of sensor nodes are to be deployed in a WSN, it is unrealistic to assume that each sensor node has the information about the whole network connectivity.

Sensor nodes are typically less mobile, more limited in capabilities, and more densely deployed than MANETs. Solutions to some of the conventional wireless networking problems, for example, medium access control, routing, self-organization, bandwidth allocation, and security [33–36], must be novelized for WSNs. Exploiting the trade-offs among energy, accuracy, and latency, and using hierarchical (tiered) architectures are important techniques for prolonging the network lifetime.

6.3.2 Passive Clustering for Efficient Flooding

With the absence of pre-established infrastructure (no router, no access point, etc.) in WSNs, two nodes can communicate directly if they are within the transmission range of each other. Otherwise, two nodes can communicate via a multihop route with the cooperation of other nodes. Flooding is used to find a feasible route to a destination or to advertise routing information [37]. In flooding, a node transmits a message to all of its neighbors. The neighbors in turn relay the messages to their neighbors until the message has been propagated throughout the entire network. As the number of neighbors is large and nodes are densely populated in a network, the network performance of using this type of blind flooding is severely impaired because of redundant and superfluous packets, packet collisions, and congestion in the wireless medium. Efficient flooding focuses on developing efficient heuristics that select a suboptimal dominant set so as to lower forwarding overhead [38].

In fact, it may suffice to use only a subset of nodes to deliver the flood packets to every other node in the system. There are two basic approaches for selecting the dominant set: one adopts clustering and the other does not use clustering, which are referred to as the clustering approach and the nonclustering approach, respectively. The nonclustering approach can improve flooding efficiency by building a source tree with the maximal number of leaf nodes [39–41] to which the flood packets are not forwarded. To build such a source tree, knowledge of a complete neighbor list is required at the source nodes. Note that the neighbor-learning procedure is not trivial in WSNs and it involves substantial overhead with high node density and mobility.

The clustering approach is based on a 2-hop clustering structure; that is, any two nodes within one cluster are separated by at most 2-hops. The 2-hop clustering only requires direct neighbor information and can be easily constructed. The cluster structure shown in Fig. 6.5 is an example of a 2-hop cluster structure. The 2-hop clustering ends up with a structure similar to a cellular system. Cluster heads at the center of each cluster can communicate with any node in the cluster within a single-hop. Nodes belonging to more than one cluster are gateway nodes. No cluster heads are directly linked and the communications between cluster heads are through these gateway nodes. The rest of the nodes are ordinary nodes and only cluster heads and gateway nodes are required to forward the flood packets. For example, in Fig. 6.10, node 1 starts to forward flood packets; only the cluster heads of nodes 4, 6, 18 and the two gateway nodes of 10 and 16 are

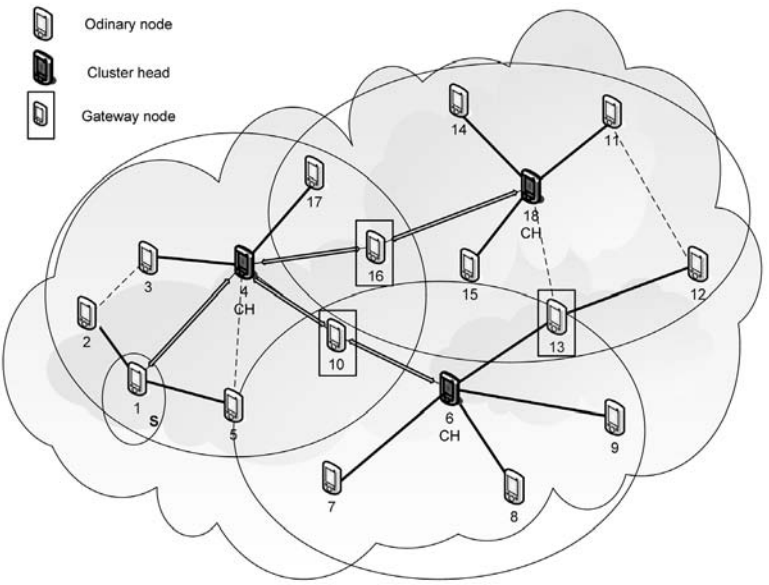


Fig. 6.10 Passive flooding structure.

involved in forwarding the flood packets. This dramatically reduces the flood packets within the network. Since the node clustering approach is based on the transmission range of the cluster heads, even if there are more sensor nodes in the network, the clustering structure and the broadcast will remain the same. Most clustering mechanisms rely on periodic broadcast of the neighbor list to ensure the correct collection of neighborhood information. These mechanisms use explicit control packets to elect a small set of nodes (cluster heads, gateway nodes, or flood-forwarding nodes), and restrict the flood forwarding function to such a set only. These proactive mechanisms will incur traffic overhead in the network.

In order to reduce flooding overhead, a flooding mechanism is proposed based on a passive on-demand node clustering algorithm [42]. Passive node clustering is an on-demand protocol that dynamically partitions the network into clusters interconnected by gateway nodes. Passive node clustering does not require periodic control messages to collect topological information. Instead, it exploits ongoing data packets to exchange cluster-related information. The cluster infrastructure can be constructed while monitoring user data packets that piggyback some predefined cluster information. In passive clustering, each node collects neighbor information from the MAC sender address carried by the incoming packets, and can construct clusters even without collecting the complete neighbor list.

The passive clustering and maintaining procedure is simple, easy to implement, and fully distributed. Clustering status information (2 bits for 4 states: INITIAL, CLUSTERHEAD, GATEWAY, and ORDINARY_NODE) is piggybacked in a reserved field in the MAC packet header. This is the only extra overhead required by passive clustering. All nodes maintain a cluster head list and a neighbor list with the sender's information on ID and reception time. Whenever sending or receiving MAC packets, a node examines and updates its cluster-head and neighbor lists. The following rules are applied to the passive clustering algorithm:

1. At the beginning, when a network is being setup, every sensor node is in the INITIAL state until it receives a MAC packet. If the sender's state is not CLUSTERHEAD, this sensor node can change its own status into CLUSTERHEAD. This sensor node will become a cluster head if it successfully transmits an outgoing packet before it receives any packets from another cluster head. If the sensor node receives a packet from another cluster head before it becomes a cluster head, it adds the sender to its cluster-head list, and changes itself to the ORDINARY_NODE state.
2. Any sensor node that hears from more than one cluster head becomes a GATEWAY. If it does not hear from more than one cluster head for a given period, its status is changed to the ORDINARY_NODE state.
3. If a sensor node receives packets from a cluster head, it updates or refreshes its cluster-head list.

4. If a cluster head node receives a packet from another cluster head, it goes into the ORDINARY_NODE state.
5. Every node collects the neighbor information as the clustering procedure proceeds. It stores its neighbors' ID, state, and idle time. If the idle time goes beyond the timeout threshold, the entry is removed. If the number of the cluster-head list becomes zero, the node will go to the INITIAL state.

This basic algorithm can produce a large number of gateway nodes and cause significant redundant flood packets. To select a minimal set of gateways, the cluster-head list for each gateway node has to be collected, and then one gateway node is chosen for each pair of cluster heads [43]. However, this method introduces extra communication and computation overhead because the cluster-head lists have to be exchanged among gateway nodes. A heuristic gateway node selection algorithm is proposed to enable a limited number of gateways and at the same time preserve adequate connectivity within the resulting cluster structure [44]. The nodes with more than two entries in their cluster-head list can be a candidate for a gateway node. Upon sending a packet, this gateway node candidate selects two cluster-heads from the cluster-heads list and announces itself as the gateway node between the selected two cluster-heads. If a gateway node receives a packet from another gateway node that announced the same pair of cluster-heads, this gateway node will compare its node ID with the sender's ID. If this node has a lower ID than the sender's, this node remains its role as the gateway node for that pair of cluster heads. Otherwise, this node can announce itself to be a gateway node for a new pair of cluster heads that has not been announced by other gateway nodes or this node changes its status from GATEWAY to ORDINARY_NODE state.

This heuristic algorithm also allows one distributed gateway node to interconnect two disconnected clusters [45]. If an ordinary node has only one gateway node in its neighbor list and it belongs to only one cluster, it can be a distributed gateway node as long as there is no other distributed gateway node in the same cluster, as shown in Fig. 6.10.

In summary, passive node clustering has several merits as compared to other efficient flooding mechanisms:

1. Passive clustering does not need any periodic messages; instead, it exploits existing traffic to piggyback its small control messages. Based on the passive clustering technique, it is very resource efficient regardless of the degree of neighbor nodes or the size of the network. Passive clustering provides scalability and practicality for choosing the minimum number of forwarding nodes in the presence of dynamic topology changes. Therefore, it can be easily applied to on-demand routing schemes to improve the performance and scalability.
2. Passive clustering does not have any setup latency and it saves energy when there is no traffic.

3. Passive clustering maintenance is well adaptive to dynamic topology and resource availability changes.

6.3.3 Energy-Efficient Adaptive Clustering

Cluster heads are responsible for coordination among the sensor nodes within their clusters and aggregation of their data (intracluster coordination), and communication with other cluster heads or external observers on behalf of their clusters (intercluster communication). As a cluster head needs to perform more load that other sensor nodes, it may consume energy at a much faster rate. A dynamically changing cluster-head algorithm for WSNs has been proposed to distribute energy consumption as evenly as possible [46]. In most scenarios, the following assumptions are made.

1. The base station is located far from the sensor nodes and is stationary.
2. All sensor nodes in the network are homogeneous and energy constrained.
3. All sensor nodes are able to reach the base station.
4. Sensor nodes have no location information.
5. The propagation channel is symmetric.
6. Cluster heads perform data compression.

The lifetime of a sensor network is defined by using three metrics: FND (first node dies), HNA (half of the nodes alive), and LND (last node dies).

Low-energy adaptive clustering hierarchy (LEACH) is a popular energy efficient adaptive clustering algorithm that forms node clusters based on the received signal strength and uses these local cluster heads as routers to the base station [47]. Since data transfer to the base station consumes more energy, all the sensor nodes within a cluster take turns with the transmission by rotating the cluster heads. This leads to balanced energy consumption of all nodes, and hence a longer lifetime of the network.

A predefined value, P (the desired percentage of cluster heads in the network), is set before starting this algorithm. LEACH works in several rounds where each round has two phases, the setup phase and the steady phase. During the setup phase, each node decides whether or not to become a cluster head. Each node chooses a random number p between 0 and 1, which is the probability to elect itself as a cluster head. If the probability p is less than a threshold $T(n)$ for node n, node n will become a cluster head for the current round r. This $T(n)$ is calculated as follows:

$$T(n) = \begin{cases} \dfrac{P}{1 - P[r \bmod (1/P)]} & \text{if node } n \in G, \\ 0 & \text{otherwise,} \end{cases} \tag{6.4}$$

where G is the set of sensor nodes that have not become cluster heads in the last $1/P$ rounds. Therefore, at the initial round 0, each node has the same probability p of becoming a cluster head. The parameter $T(n)$ is increased for the next round because there are fewer nodes left that are candidates for cluster heads. These cluster heads broadcast to all sensor nodes in the network, announcing they are the new cluster heads. The other sensor nodes determine the cluster to join based on the signal strength received from these cluster heads. Each sensor node informs the appropriate cluster head that it will be a member of the cluster, and the node clusters are organized.

During the steady phase, the sensor nodes can begin sensing and transmitting data to the cluster heads. The cluster heads also aggregate data from the sensor nodes in their cluster and send data to the base station. After a certain period of time spent on the steady phase, the network goes into another round of selecting the cluster heads. The duration of the steady phase is longer than the duration of the setup phase in order to minimize the overhead.

LEACH provides an optimized behavior for communication in WSNs based on self-organization methods. Mobility is also supported by LEACH, whereas new nodes have to be synchronized to the current round. Node failures may lead to less cluster heads to be elected than desired because the predefined P is a percentage of the total number of sensor nodes.

Considering a single round of LEACH, a stochastic cluster-head selection will not automatically lead to minimum energy consumption during the steady phase for data transfer of a given set of sensor nodes. For example, some of the cluster heads can be located near the edges of the network or some adjacent nodes can become cluster heads. In these cases, some sensor nodes are further away from a cluster head. However, considering two or more rounds, a selection of favorable cluster heads at the current round can result in an unfavorable cluster-heads selection in the later round.

Regarding energy consumption, a deterministic cluster-head selection algorithm can outperform a stochastic algorithm. One approach is to reduce the threshold $T(n)$ relative to the node's remaining energy [48]. This modification of the cluster-head threshold can increase the lifetime of LEACH network by 30% for FND and by >20% for HNA.

The modification of the threshold equation by the remaining energy may bring up another problem. Since the remaining nodes have a low energy level after a number of rounds, the cluster-head threshold will become too low. Some cluster heads will not have enough energy to transmit data to the base station. The network cannot work well although there are still nodes available with enough energy to perform this task.

The threshold equation can be modified further by including a factor that increases the threshold for any node that has not been a cluster head for a certain number of rounds. The chance of this node becoming a cluster head increases because of the higher threshold.

6.3.4 Energy-Efficient Distributed Clustering

Another popular energy-efficient node clustering algorithm is the hybrid, energy-efficient, and distributed (HEED) clustering approach for ad hoc sensor networks [49]. This was proposed with four primary goals: (1) prolonging network lifetime by distributing energy consumption, (2) terminating the clustering process within a constant number of iterations, (3) minimizing control overhead (to be linear in the number of nodes), and (4) producing well-distributed cluster heads and compact clusters. HEED periodically selects cluster heads based on a hybrid of two clustering parameters: The primary parameter is the residual energy of each sensor node and the secondary parameter is the intracluster communication cost as a function of neighbor proximity or cluster density. The primary parameter is used to probabilistically select an initial set of cluster heads while the secondary parameter is used for breaking ties.

The clustering process at each sensor node requires several rounds. Every round is long enough to receive messages from any neighbor within the cluster range. As in LEACH, an initial percentage of cluster heads in the network, C_{prob}, is predefined. The parameter C_{prob} is only used to limit the initial cluster-head announcements and has no direct impact on the final cluster structure. In HEED, each sensor node sets the probability CH_{prob} of becoming a cluster head as follows

$$CH_{\text{prob}} = C_{\text{prob}} \cdot \frac{E_{\text{residual}}}{E_{\max}}, \tag{6.5}$$

where E_{residual} is the estimated current residual energy in this sensor node and $E_{\max}$ is the maximum energy (corresponding to a fully charged battery), which is typically identical for homogeneous sensor nodes. The CH_{prob} value must be greater than a minimum threshold $p_{\min}$. A cluster head is either a *tentative cluster-head*, if its CH_{prob} is <1, or a *final cluster-head*, if its CH_{prob} has reached 1.

During each round of HEED, every sensor node that never heard from a cluster head elects itself to become a cluster head with probability CH_{prob}. The newly selected cluster heads are added to the current set of cluster heads. If a sensor node is selected to become a cluster head, it broadcasts an announcement message as a *tentative cluster-head* or a *final cluster-head*. A sensor node hearing the cluster-head list selects the cluster head with the lowest cost from this set of cluster heads. Every node then doubles its CH_{prob} and goes to the next step.

If a node completes the HEED execution without electing itself to become a cluster head or joining a cluster, it announces itself as a *final cluster-head*. A *tentative cluster-head* node can become a regular node at a later iteration if it hears from a lower cost cluster head. Note that a node can be selected as a cluster head at consecutive clustering intervals if it has higher residual energy with lower cost.

Since a WSN is assumed to be a stationary network, where nodes do not die unexpectedly, the neighbor set of every node does not change very frequently.

Here HEED does not need to do neighbor discovery very often. In addition, distribution of energy consumption of HEED extends the lifetime of all the nodes in the network, thus sustaining stability of the neighbor set. Nodes also automatically update their neighbor sets in multihop networks by periodically sending and receiving messages.

The HEED clustering improves network lifetime over LEACH clustering because LEACH randomly selects cluster heads (and hence cluster sizes), which may result in faster death of some nodes. The final cluster heads selected in HEED are well distributed across the network and the communication cost is minimized.

6.3.5 Energy-Efficient Hierarchical Clustering

This section introduces three energy efficient hierarchical clustering algorithms proposed in the literature.

6.3.5.1 Multitier Hierarchical Clustering. Multitier hierarchical clustering is a hierarchical clustering algorithm proposed in Ref. [50], which takes into consideration several cluster's properties, for example, cluster size and the degree of overlap while grouping nodes. The goals of this algorithm include the following aspects:

- There is no isolated cluster in the network.
- All clusters should have minimum and maximum size constraints.
- A node in one layer of the hierarchy belongs to a constant number of clusters in that layer.
- The degree of overlap between any two clusters within one layer should be low.
- The clusters formation should be stable across node mobility.

In this algorithm, any node in the network can initiate the cluster formation process. If multiple nodes initiate the cluster formation process at the same time, a predetermined policy will be adopted to break the tie so that only one instance is allowed to proceed. For example, the node with the lowest ID is allowed to proceed. This algorithm is based on a graph theoretic framework and is carried out in two phases: tree discovery and cluster formation.

The purpose of the tree discovery phase is to generate a breadth-first search (BFS) tree rooted at the initiator node r. Each node u broadcasts a beacon carrying the information about its shortest hop-distance to the root node r. If any neighbor v of u finds out the route to r through u is shorter, it will choose u as its parent and will accordingly update its hop-distance to the root r. The beacon contains the information, for example, source ID, parent ID, root ID, and subtree size. Every node updates its subtree size when its children subtree size changes. A simple example is shown in Fig. 6.11, where node *5* is originally at 3-hops

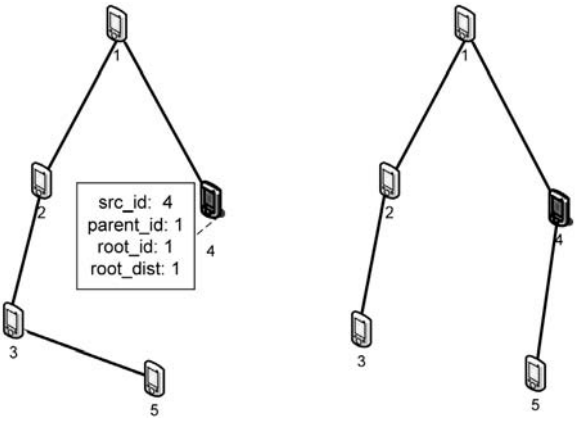


Fig. 6.11 The change of part of the BFS tree.

away from the root node *1*. It receives a beacon from node *4*, which is at 1-hop away from the root node *1* and consequently change its parent node from node *3* to node *4*. The distance from node *5* to the root node *1* decreases to 2-hops.

The cluster formation phase is started when the size of any subtree on a node exceeds the size constraint k. This node will initiate cluster formation on its subtree. A single cluster for the entire subtree is created if the entire subtree size is $<2k$. Otherwise, multiple clusters will be formed. After the cluster creation phase, keeping cluster information is crucial for clusters while maintaining the BFS tree is unimportant. This algorithm is suitable for handling dynamic environments, for example, in the presence of mobile nodes.

6.3.5.2 Energy-Efficient Hierarchical Clustering. Energy-efficient hierarchical clustering (EEHC) is a distributed randomized clustering algorithm that maximizes the lifetime of a network with a large number of sensor nodes [51]. The EEHC algorithm organizes the sensors in a network into clusters with a hierarchy of cluster heads, as shown in Fig. 6.12. The cluster heads collect the information from the sensor nodes within their clusters and send an aggregated report through the hierarchy of cluster heads to the base station. The EEHC algorithm assumes that the communication environment is contention and error free. The energy consumed in the network will depend on (1) the probabilities of each sensor node becoming a cluster head at each level in the hierarchy and (2) the maximum number of hops allowed between one cluster node and its cluster head. Optimal clustering parameters are obtained through hierarchical clustering to minimize the total energy consumption in the network. The EEHC algorithm is based on two-stage clustering: single-level clustering and multilevel clustering.

At the single-level clustering stage, each sensor node becomes a cluster head at a predefined probability p and announces itself as a volunteer cluster head to its neighbor nodes that are within k-hops communication range. Any node that

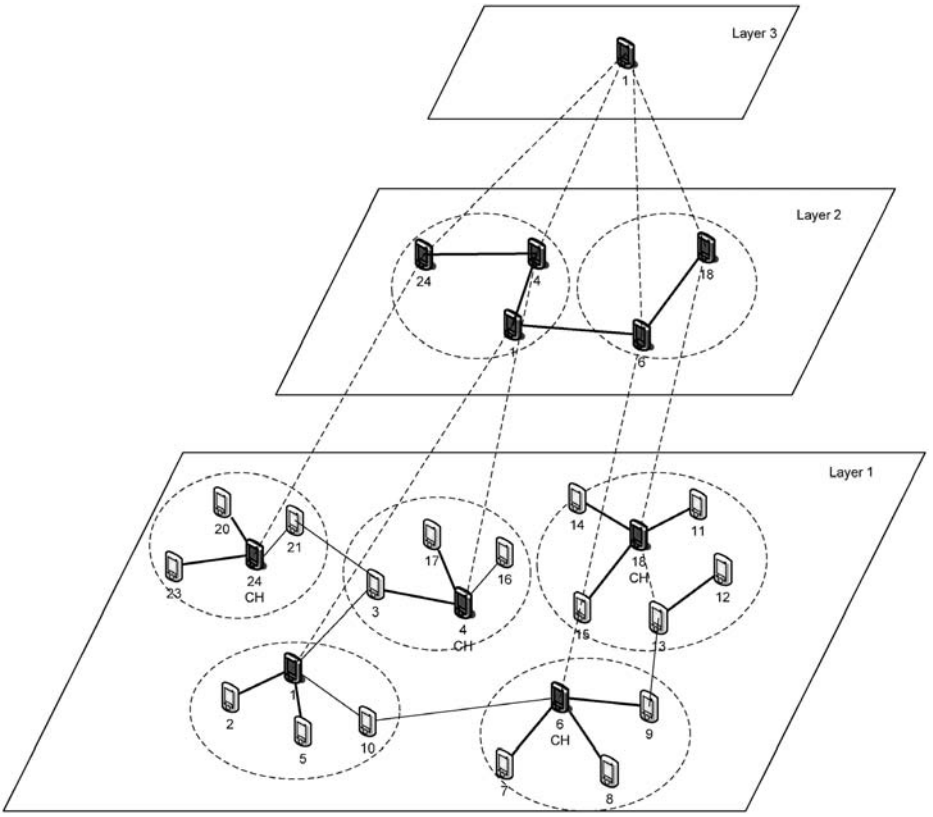


Fig. 6.12 An example of a three-layer hierarchy.

receives such an announcement will become a member of the closest cluster if it is not a cluster head. Those nodes that are neither cluster heads nor belong to a cluster will become forced cluster heads. If a node does not receive any announcement within a preset time interval t, this node will assume that it is not within k-hops of all volunteer cluster heads and thus become a forced cluster head. The time interval t is calculated based on the duration for a packet to reach a node that is k-hops away.

The energy consumed by the network for sending the information gathered by the sensor nodes to the processing center depends on the parameters p and k. To determine the optimal parameters, EEHC makes the following assumptions [51]:

1. The distribution of the sensor nodes is based on a homogeneous spatial Poisson process of intensity λ in two-dimensional (2D) space.
2. All sensor nodes transmit at the same power level, and hence have the same transmission range r.

3. Data exchanged between two sensor nodes beyond range r is forwarded by intermediate nodes.
4. The transmission hops between any sensor and its cluster head is equivalent to $\lceil d/r \rceil$ hops.
5. The energy consumed to transmit or receive one unit of data is one unit at each node.
6. When a sensor node communicates data to another sensor node, only the sensor nodes on the routing path forward the data.
7. There is no data retransmission because the communication environment is contention and error free.

At the second stage, the same mechanism is extended from bottom-up to multilevel clustering. Assume that there are h levels in the clustering hierarchy with level-1 being the lowest level and level-h being the highest. The information collected at all sensor nodes is first sent to level-1 cluster heads. Then the level-1 cluster heads aggregate the information and forward the aggregated report to the level-2 cluster heads, and so on. Finally, the level-h cluster heads send the aggregated report to the base station. The cost of transmitting the information from the sensor nodes to the base station is the energy consumed by the sensor nodes to send the information to the level-1 cluster heads plus the energy consumed by the level-1 cluster heads to the base station via h-hop cluster heads at different levels. EEHC has a time complexity of $O(k_1 + k_2 + \ldots + k_h)$, where k_i is the k-hops limitation at level i, which is a significant improvement over many clustering algorithms (e.g., LCA) with a complexity of $O(n)$, thus making it suitable for use in WSNs with a large number of nodes. Energy consumption for network operations (e.g., sensor data collection, aggregated information transmission to the base station) will depend on the optimization of the parameters p and k.

Those sensor nodes that become cluster heads in the hierarchical architecture consume relatively more energy than other sensor nodes because cluster heads have more loads to handle. Hence, cluster heads may run out of their energy faster than other sensors. The EEHC algorithm can be run periodically for load balancing or triggered as the energy levels of the cluster heads fall below a certain threshold.

6.3.5.3 Distributed Weight-Based Hierarchical Clustering. Distributed weight-based energy-efficient hierarchical clustering (DWEHC) is another clustering algorithm proposed to achieve balanced cluster sizes and optimize intracluster topologies for WSNs [52]. The DWEHC algorithm makes no assumptions on the size and the density of a network, but assumes that sensor nodes are location aware and transmit at the same fixed power levels. The cluster radius; that is, the farthest transmission distance from one cluster member node to its cluster head is fixed for the whole network. After individually running seven iterations on each node, DWEHC generates a multihop intracluster structure in which a

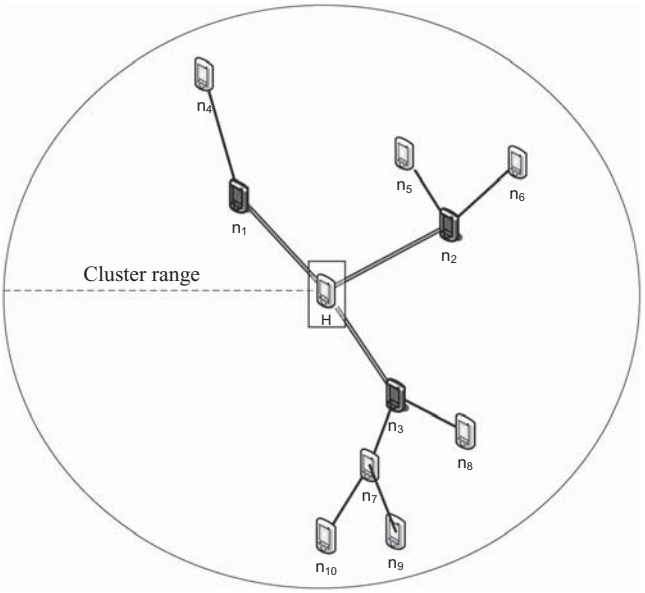


Fig. 6.13 An example of one cluster in multihop intracluster topology.

cluster head is at the root and member nodes are in a breadth-first order. Figure 6.13 illustrates an intracluster example, in which each cluster has a hierarchical structure with one cluster head and the child nodes of the first level, second level, and so on. Each cluster has multilevels of child nodes. Since there are no assumptions about the size and topology of the network, the number of levels within one cluster is determined by the cluster radius and the minimum energy path from one member node to its cluster head is established by DWEHC. Each node only responds to its nearest parent's request, and that parent then responds to its own parent until the data reaches the cluster head. Time division multiple access (TDMA) is used for intracluster communication and the cluster heads contend for the channel using the 802.11 protocol to send data to the base station.

During initialization, each sensor node first broadcasts its (x, y) coordinates to find its neighbors [53]. After locating the neighboring nodes in its area, it calculates its weight as follows:

$$w_{\text{weight}}(s) = \left(\sum_{u \in N_{a,c}(S)} \frac{(R-d)}{6R} \right) \times \frac{E_{\text{residual}}(s)}{E_{\text{initial}}(s)}, \tag{6.6}$$

where R is the cluster range that is fixed for the entire network; d is the distance from node s to neighbor node u; $N_{\alpha,c}(s)$ is the set of the neighbors of node s, where α is the transmitter power factor equal to 2 or 4, and c is a constant; $E_{\text{residual}}(s)$ is the residual energy of node s, and $E_{\text{initial}}(s)$ is the initial energy of node s, which is the same for all nodes.

A node that has the largest weight among all its neighbors will become a temporary cluster head. A temporary cluster head can become a real cluster head only if a given percentage of its neighbors elect it as their cluster head. This percentage is 100% in the first iteration ($i = 0$) and in subsequent six iterations ($i < 6$), it is decreased to $(6\text{-}i)/6$.

At this stage, the neighboring nodes are considered as the first-level child members with respect to the cluster head. A node progressively adjusts such membership in order to reach a cluster head using the least amount of energy. Basically, a node checks with its noncluster-head neighbors to find out their minimal cost for reaching a cluster head. Given the node's knowledge of the distance to its neighbors, the node can assess whether it is better to stay as a first-level member or become a second-level one; that is, reaching the cluster head over a 2-hop path. It is possible that the node may switch to a new cluster head other than its original one. The process continues until all nodes settle on the most energy-efficient intracluster topology. The cluster generating process runs at most seven times (including finalization) because each node has at most six neighbors [53]. Figure 6.13 illustrates the structure of the intracluster topology, where H is the cluster head, n_1, n_2, and n_3 are the first-level children, n_4, n_5, n_6, n_7, and n_8 are the second-level children, and n_9 and n_{10} are the third-level children. A parent node and its child nodes are neighbors. For example, n_1, n_2, and n_3 are neighbors with H.

The DWEHC algorithm is completely distributed over the whole network. Each node is either a cluster head or a child member in a cluster. Each cluster contains the minimum-power topology, which is locally optimal. Each parent node has a limited number of child nodes, which is important in terms of scalability. This algorithm achieves good load balance per node, thus prolonging the lifetime of a cluster head.

Both DWEHC and HEED consider energy reservation in cluster-head selection and do not make any assumptions about the network size. However, clusters generated by DWEHC are more load balanced than those by HEED. The DWEHC algorithm also achieves significantly lower energy consumption in intracluster and intercluster communication.

6.3.6 Algorithm for Cluster Establishment

Different from other distributed clustering algorithms, the algorithm for cluster establishment (ACE) in Ref. [54] employs an emergent cluster formation algorithm that uses just three rounds of feedback to induce the formation of a highly efficient cover of uniform clusters over a WSN.

In a WSN, an emergent algorithm is a localized algorithm that emerges as a result of repeated local interaction and feedback between the nodes to achieve the desired global property without a central control or global visibility. Emergent algorithms commonly require several rounds of feedback between a node and its neighbors before the whole network converges on the desired global property. Complex global properties can be more easily expressed with emergent

algorithms. The explicit coordination and calculation required for such tasks as efficient cluster formation is sidestepped with iterated feedback. Since all interactions are repeated several times, a small number of missing or incorrect interactions are unlikely to have a large effect on the whole process. This iterative nature of emergent algorithms helps to improve robustness against transient node failures and may allow these algorithms to tolerate some errors in consistency and synchronization between nodes.

Spawning new clusters and migration of existing ones are the two logical parts of ACE. Each node assesses its potential as a cluster head before becoming one. When a sensor node decides to become a new cluster head, it broadcasts an invitation message to start the process to form new clusters. Upon getting the invitation, a neighboring sensor node may join the new cluster. A node could receive invitations from more than one cluster; however, it can be a member of only one single cluster.

Migration of an existing cluster is controlled by the cluster heads. Each cluster head periodically evaluates the ability of its neighbors for becoming a new cluster head. A node will be considered as the best candidate for the new cluster head if the newly formed cluster could have the largest number of member nodes with minimal overlaps with existing clusters.

A node can have three possible states: unclustered (not belongs to any cluster), clustered (a member of one cluster) or a cluster head, as illustrated in Fig. 6.14. All nodes are unclustered at the beginning of ACE. Each node waits for a random iteration interval before deciding on what action to take for that iteration. In ACE, the iterations do not require individual nodes to be synchronized. When a node's iteration arrives, its available choice of actions will depend on its current state.

In ACE, feedback occurs when node A affects node B, which then directly or indirectly affects node A again. Considering the trade-off between communication overhead and cluster size, ACE generates the clusters covering the entire network in three rounds. A modified ACE was proposed in Ref. [55] to improve

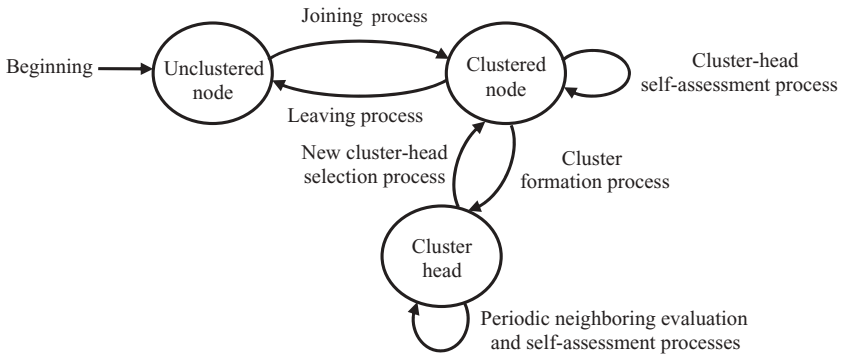


Fig. 6.14 Three possible states of a node in ACE.

the regularity of the separation between cluster heads. The number of iterations used in the modified ACE is increased to five in order to increase the regularity of cluster layout; however the communication cost is increased as well.

ACE increases the spatial coverage in the network because it increases the separation among clusters in areas where the degree of nodes is high, while allowing the clusters to be overlapped where the degree of nodes is low. Such an approach allows spreading of the clusters according to the node density throughout the area of interest. Experimental validation of ACE has indicated that it achieves low variance and high average of cluster sizes when compared to node-ID-based algorithms like LCA. In addition, ACE can easily repair structure damage in the network caused by node failures and can also integrate new nodes.

ACE is an emergent algorithm that uses just three rounds of feedback to induce the formation of a highly efficient cover of uniform clusters over the network. This efficiency of coverage approaches that of hexagonal close packing. The ACE is fast, robust against packet loss and node failures, and efficient in terms of communications. It completes in constant time regardless of the size of the network and uses only local communications between nodes. The algorithm does not require geographic location information or any kind of distance or directional estimation between nodes.

6.3.7 Secure Clustering

WSNs are usually deployed in open and unattended environments where all nodes do not have physical protection, and thus are vulnerable to various potential malicious attacks, in particular, in a hostile environment such as battlefield [56,57]. Conventional security mechanisms are not directly applicable to WSNs because they do not consider the unique constraints of WSNs. To protect a sensor network from malicious attacks, it is crucial to design effective security mechanisms or protocols with both energy efficiency and security strength.

Access control preventing illegitimate nodes from participating in a network can preserve much of a network's operation under most scenarios. Cryptographic mechanisms with key distribution are the most common solutions to securing the access control in a network. However, most key distribution schemes in the literature cannot be applied to WSNs because of [58]:

1. The processing requirement of public-key-based distributions.
2. The security vulnerabilities with global keying schemes.
3. The memory requirements of complete pairwise keying distributions.
4. The inefficiency and energy consumption of center-based key distributions.

Securing a WSN with random key predistribution has been intensively studied in the context of flat networks [59–61]. Most of the key distribution

schemes proposed in the literature assume that each node only interact with a static set of neighbors, which are predefined at deployment and are not well suited to clustered WSNs. In a cluster-based network, cluster heads are more prominent targets for attacks because cluster-based protocols rely on the cluster heads for data aggregation and routing. A compromised cluster head can be used to attack the network with selective forwarding and sinkhole or relay bogus information into the network. Those clustering algorithms with rotating cluster heads, like LEACH, make it harder for an adversary to identify the cluster heads and compromise them [56].

Given the communication patterns in LEACH, there are two kinds of authentication for legitimate nodes: authenticated broadcast (from the cluster heads to the rest of the network) and pairwise authentication (node-to-cluster-head and cluster-head-to-base-station). F-LEACH is a security-enhanced LEACH mechanism, where each node has two symmetric keys: an authenticated broadcast key from a key chain held by the base station and a pairwise key shared with the base station [62]. The base station is trusted and has more resources. Each cluster head sends a slightly modified adv (advising) message consisting of (1) the ID of the cluster head in plain text used by the ordinary nodes; and (2) a message authentication code produced using the key that the cluster head shares with the base station. The base station listens and authenticates adv messages from all cluster heads, and then compiles a list of legitimate cluster heads and sends the list to the network by using the broadcast authentication scheme [63]. Once receiving the list, ordinary nodes know which of the adv messages they received are from legitimate nodes, and thus can proceed with the rest of the LEACH protocol, choosing a cluster head from the list broadcasted by the base station. Figure 6.15 illustrates the key exchange schemes at the setup phase and the steady-state phase.

Since there are only two keys per node, F-LEACH does not provide a complete and efficient solution to node-to-cluster-head authentication. In particular, join-request messages in the setup phase are not authenticated, and ordinary nodes only share keys with the base station. This means that the cluster heads cannot verify the message authentication codes from ordinary nodes and, in turn, have to forward them.

SecLEACH is another LEACH based random key predistribution scheme that provides efficient security to pairwise node-to-cluster-head communication [64].

In a random key predistribution scheme, each node is assigned a set of keys drawn from a much larger key pool. Regardless of the key assignment algorithm adopted for different key distribution schemes, the probabilistic key is eventually shared among all nodes in the network after three phases:

1. Key predistribution—which takes place prior to network deployment. A large pool of S keys and their IDs are generated. Each node is then assigned a ring of m keys drawn from the pool at random without replacement.

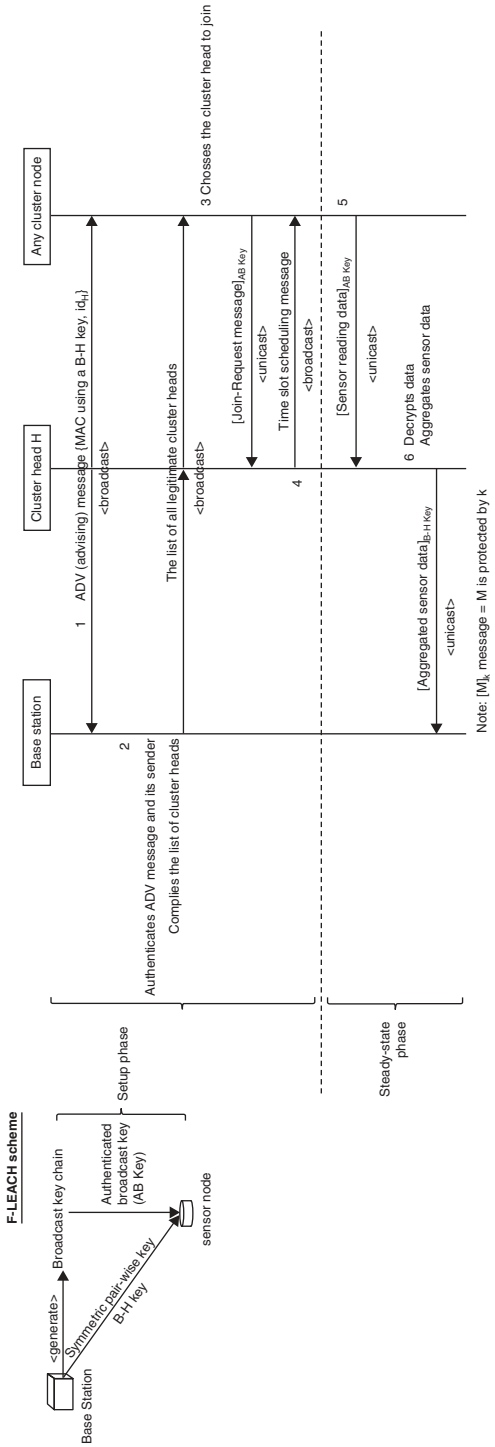


Fig. 6.15 The F-LEACH scheme.

2. Shared-key discovery—which takes place during network setup. Each node broadcasts the IDs of the keys on its key ring. Through these broadcasts, a node finds out the keys shared with its neighbors within its communication range. These keys can be used for establishing secure links between two neighboring nodes.
3. Path-key establishment—in which each pair of neighboring nodes that do not share a key can establish their own keys as long as they are connected by two or more secure links at the end of the shared key discovery phase.

A key can be found in more than two nodes and can be used in multiple communication links. Once a node is compromised, all its keys are compromised and all the links secured by these keys are also compromised.

SecLEACH generates a large pool of S keys with their IDs prior to network deployment. Each node is then assigned a ring of m keys drawn from the pool pseudorandomly [65] without replacement. Each node X generates a unique ID id_X with one pseudorandom function (PRF). id_X is then used to seed a pseudorandom number generator (PRNG) to produce a sequence of m numbers. The R_X parameter is the set of key IDs assigned to X and can be obtained by mapping each number in the sequence to its corresponding value modulus s. Each node is assigned with a pairwise key shared with the base station prior to network deployment.

The LEACH algorithm can then be run with the following modifications: when a self-elected cluster head broadcasts its adv message, it includes the IDs of the keys in its key ring; the remaining nodes then cluster around the closest cluster head with whom they share a key. Figure 6.16 illustrates the procedures in one cycle of the SecLEACH protocol, which are outlined as follows:

1. Each self-elected cluster head H broadcasts its ID id_H and a nonce that is used to prevent replay attacks.
2. Ordinary nodes A_i compute the set of H's key IDs using the pseudorandom scheme described above, choose the closest cluster head with whom they share a key $k_{[r]}$, and send it a join-request message, which is protected by a message authentication code. The message authentication code is generated by using $k_{[r]}$, and includes the nonce from H's broadcast in Step 1 as well as the ID r of the key chosen to protect this link.
3. The cluster heads send the timeslot schedule to their member nodes that are chosen to join their clusters. This completes the setup phase.
4. In the steady-state phase, node-to-cluster-head communications are protected by using the same key that was used to protect the join-request message in Step 2. A value *nonce*, computed from the nonce and the reporting cycle (j), is also included to prevent replay attacks.

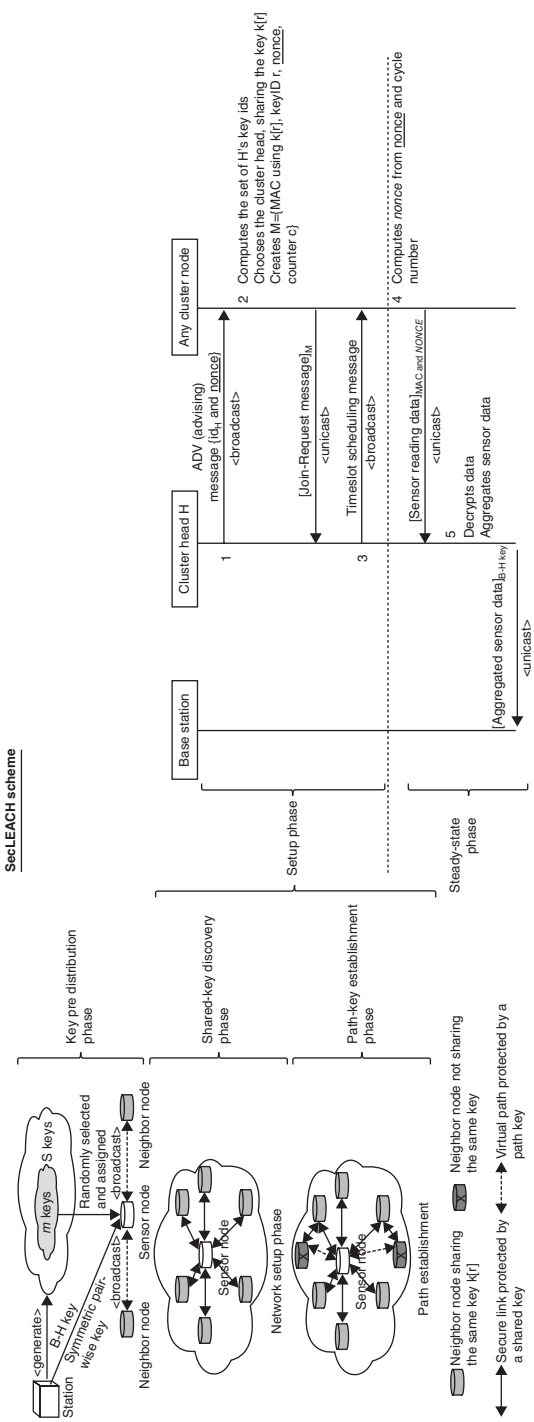


Fig. 6.16 Sec-LEACH scheme.

5. The cluster heads can decrypt the data reports they received, perform data aggregation, and send the aggregated result to the base station. The aggregate result is protected with the symmetric key shared between the cluster head and the base station. To ensure the freshness of the result, a counter c shared between the cluster head and the base station is included in the message authentication code as well.

In practice, there will be multiple cycles in each round j. For each cycle, the counter value c in Step 5 is incremented by 1, and the value of the "freshness token" *nonce* in Step 4 is updated. Since the random key predistribution does not authenticate the broadcasts in Steps 1 and 3, the SecLEACH scheme runs in conjunction with the authenticated broadcast proposed in F-LEACH.

At the end of the clustering process, some nodes will not be clustered with a cluster head because of the key sharing constraints. The number of these nodes depends on the values of S and m, which determine the probability that two nodes will share a key. To achieve maximum security and energy efficiency in the context of SecLEACH, the appropriate values of S and m, and the number of cluster heads have to be optimized based on the application requirements.

6.4 SUMMARY AND FUTURE DIRECTIONS

Node clustering is very important in WSNs because it provides a topology control approach to reduce transmission overheads and exploit data aggregation among a large number of sensor nodes. One critical step in node clustering is to select a set of cluster heads and group the remaining sensor nodes into clusters with these cluster heads. This chapter introduced four major cluster-head selection algorithms. The research work on clustering techniques in ad hoc networks have been focused on routing and resource allocation. Node clustering algorithms for ad hoc networks can be classified with respect to their objectives, as discussed in Section 6.2.

Most existing node clustering algorithms for WSNs are tailored from the algorithms originally designed for traditional ad hoc networks. They are primarily focused on scalability and energy conservation in WSNs. This chapter introduced and discussed several state-of-the-art clustering algorithms along with their challenges. One of the most important constraints on sensor nodes is the low power consumption requirement. The power sources for sensor nodes are limited and generally irreplaceable. Therefore, while the objectives of traditional ad hoc networks focus on high QoS provisioning, WSNs must focus primarily on power conservation. It is important to exploit the trade-off between longer network lifetime and lower throughput or higher transmission delay.

Clustering in WSNs has several unique challenges in its deployment, for example, ensuring connectivity among sensor nodes, determining optimal cluster sizes, and dynamically optimizing clustering structures based on the status of cluster members. To achieve optimal performance, it is important to consider the

interactions between different layers of the protocol stack. Several key issues in node clustering for WSNs have yet to be addressed satisfactorily, including self-organization of sensor nodes into clusters according to the application requirements (e.g., task triggered clustering), application-specific functionalities of sensor nodes, and dynamic clustering of mobile targets.

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey", *Computer Networks*, vol. 38, no. 4, Mar. 2002, pp. 393–422.

[2] G. Pottie and W. Kaiser, "Wireless integrated sensor networks (WINS)", *Communications of the ACM*, vol. 43, no. 5, May 2000, pp. 51–58.

[3] H. Nakayama, N. Ansari, A. Jamalipour, Y. Nemoto, and N. Kato, "Fault-resilient sensing in wireless sensor networks", *Computer Communications*, vol. 30, no. 11, Sept. 2007, pp. 2375–2384.

[4] R. Rajagopalan and P. Varshney, "Data-aggregation techniques in sensor networks: A survey", *IEEE Communications and Surveys and Tutorials*, vol. 8, no. 4, 4th Quarter 2006, pp. 48–63.

[5] J. Al-Karaki and A. Kamal, "Routing techniques in wireless sensor networks: A survey", *IEEE Wireless Communications Magazine*, vol. 11, no. 6, Dec. 2004, pp. 6–28.

[6] R. Shah, S. Roy, S. Jain, and W. Brunette, "Data mules: Modeling and analysis of a three-tier architecture for sparse sensor networks", *Ad Hoc Networks*, vol. 1, nos. 2–3, Sept. 2003, pp. 215–233.

[7] I. Chatzigiannakis, A. Kinalis, and S. Nikoletseas, "Sink mobility protocols for data collection in wireless sensor networks", in *Proceeding of the 4th ACM Workshop on Mobility Management and Wireless Access (MobiWac'06)*, Los Angeles, CA, Oct. 2006, pp. 52–59.

[8] E. Royer and C. Toh, "A review of current routing protocols for ad hoc mobile wireless networks", *IEEE Personal Communications*, vol. 6, no. 2, Apr. 1999, pp. 46–55.

[9] I. Stojmenovic, *Handbook of Sensor Networks Algorithms and Architectures*. John Wiley & Sons, Inc., Hoboken, NJ, 2005.

[10] V. Mhatre and C. Rosenberg, "Design guidelines for wireless sensor networks: Communication, clustering and aggregation", *Ad Hoc Networks*, vol. 2, no. 1, Jan. 2004, pp. 45–63.

[11] V. Mhatre and C. Rosenberg, "Homogeneous vs heterogeneous clustered sensor networks: A comparative study", in *Proceeding of IEEE ICC'04*, vol. 6, Paris, France, June 2004, pp. 3646–3651.

[12] C. Bettstetter, "On the minimum node degree and connectivity of a wireless multihop network", in *Proceeding of the 3rd ACM International Symposium on Mobile Ad Hoc Networking & Computing, Lausanne*, Switzerland, June 2002, pp. 80–91.

[13] S. Megerian, F. Koushanfar, M. Potkonjak, and M. B. Srivastava, "Worst and best-case coverage in sensor networks", *IEEE Transactions on Mobile Computing*, vol. 4, no. 1, Jan.–Feb. 2005, pp. 84–92.

[14] P. Agrwal and C. Procopiuc, "Exact and approximation algorithms for clustering", *Proceeding of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, San Francisco, CA, Jan. 1998, pp. 658–667.

[15] R. Rajaraman, "Topology control and routing in ad hoc networks: A survey", *ACM SIGACT News*, vol. 33, no. 2, June 2002, pp. 60–73.

[16] C. R. Lin and M. Gerla, "Adaptive clustering for mobile wireless networks", *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 7, Sept. 1997, pp. 1265–1275.

[17] M. Gerla and J.T.C Tsai,. "Multicluster, mobile, multimedia radio network", *ACM/ Baltzer Journal of Wireless Networks*, vol. 1, no. 3, Sept. 1995, pp. 255–265.

[18] F. Nocetti, J. Gonzalez, and I. Stojmenovic, "Connectivity based *k*-Hop clustering in wireless networks", *Telecommunication Systems*, vol. 32, nos. 1–4, Apr. 2003, pp. 205–220.

[19] C. Chiang, H. Wu, W. Liu, and M. Gerla. "Routing in clustered multihop, mobile wireless networks with fading channel", in *Proceeding of IEEE Singapore International Conference on Networks (SICON'07)*, Singapore, Apr. 1997, pp. 197–211.

[20] M. Chatterjee, S. Das, and D. Turgut, "WCA: A weighted clustering algorithm for mobile ad hoc networks", *Journal of Cluster Computing*, vol. 5, no. 2, Apr. 2002, pp. 193–204.

[21] D. Baker and A. Ephremides, "The architectural organization of a mobile radio network via a distributed algorithm", *IEEE Transactions on Communications*, vol. 29, no. 11, Nov. 1981, pp. 1694–1701.

[22] A. Ephremides, J. Wieselthier, and D. Baker, "A design concept for reliable mobile radio networks with frequency hopping signaling", in *Proceedings of the IEEE*, vol. 75, no. 1, Jan. 1987, pp. 56–73.

[23] A. Amis, R. Prakash, T. Vuong, and D. Huynh, "Max-min d-cluster formation in wireless ad hoc networks", in *Proceeding of the of IEEE INFOCOM'00*, Tel Aviv, Israel, Mar. 2000, pp. 32–41.

[24] X. Hong, M. Gerla, G. Pei, and C. C. Chiang, "A group mobility model for ad hoc wireless networks", in *Proceeding of the ACM/IEEE MSWiM'99*, Seattle, WA, Aug. 1999, pp. 53–60.

[25] B. An and S. Papavassiliou. "Geomulticast: Architectures and protocols for mobile ad hoc wireless networks", *Journal of Parallel and Distributed Computing*, vol. 63, no. 2, Feb. 2003, pp.182–195.

[26] A. McDonald and T. Znati, "A mobility-based framework for adaptive clustering in wireless ad hoc networks", *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, Aug. 1999, pp. 1466–1487.

[27] B. An and S. Papavassiliou, "A mobility-based clustering approach to support mobility management and multicast routing in mobile ad hoc wireless networks", *International Journal of Network Management*, vol. 11, no. 6, June 2001, pp. 387–395.

[28] H. Gharavi and K. Ban, "Multihop sensor network design for wide-band communications", in *Proceedings of the IEEE*, vol. 91, no. 8, Aug. 2003, pp. 1221–1234.

[29] T. Hou and T. Tsai, "An access-based clustering protocol for multihop wireless ad hoc networks", *IEEE Journal on Selected Areas in Communications*, vol. 19, no.7, July 2001, pp. 1201–1210.

[30] A. Chakrabarti, A. Sabharwal, and B. Aazhang, "Multi-hop communication is order-optimal for homogeneous sensor networks", in *Proceeding of the 3rd International*

Symposium on Information Processing in Sensor Networks, Berkeley, CA, Apr. 2004, pp. 178–185.

[31] J. Pan, Y. Hou, L. Cai, Y. Shi, and S. Shen, "Topology control for wireless sensor networks", in *Proceeding of the 9th annual international conference on Mobile computing and networking*, San Diego, CA, Sept. 2003, pp. 286–299.

[32] S. Bandyopadhyay and E. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks", in *Proceeding of INFOCOM 2003*, vol. 3, San Francisco, CA, Apr. 2003, pp. 1713–1723.

[33] S. Lindsey and C. Raghavendra, "PEGASIS: Power-efficient gathering in sensor information systems", in *Proceeding of 2002 IEEE Aerospace Conference*, vol. 3, Big Sky, MT, Mar. 2002, pp. 1125–1130.

[34] A. Manjeshwar and D. P. Agarwal, "TEEN: A routing protocol for enhanced efficiency in wireless sensor networks", in *Proceeding of the 1st International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing*, San Francisco, CA, Apr. 2001, pp. 2009–2015.

[35] P. Krishna, N. H. Vaidya, M. Chatterjee, and D. K. Pradhan, "A cluster-based approach for routing in dynamic networks", *Computer Communication Review*, vol. 27, no.2, Apr. 1997, pp. 49–64.

[36] A. Perrig, J. Stankovic, and D. Wagner, "Security in wireless sensor networks", *Communications of the ACM*, vol. 47, no. 6, June 2004, pp. 53–57.

[37] H. Lim and C. Kim, "Flooding in wireless ad hoc networks", *IEEE Computer Communications*, vol. 24, no. 3, Feb. 2001, pp. 353–363.

[38] Y. Yi, M. Gerla, and T. Kwon, "Efficient flooding in ad hoc networks: A comparative performance study", in *Proceeding of 2003 IEEE International Conference on Communications (ICC'03)*, vol. 2, Seattle, WA, May 2003, pp. 1059–1063.

[39] Y. Tseng, S. Ni, Y. Chen, and J. Sheu, "The broadcast storm problem in a mobile ad hoc network", *Wireless Networks*, vol.8, nos. 2–3, Mar.–May 2002, pp. 153–167.

[40] R. Ogier, F. Templin, B. Bellur, and M. Lewis, "Topology broadcast based on reverse-path forwarding (TBRPF)", *Request for Comments 3684*, Feb. 2004.

[41] B. Chen, K. H. Jamieson, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks", *ACM Wireless Networks Journal*, vol. 8, no. 5, Sept. 2002, pp. 481–494.

[42] T. Kwon and M. Gerla, "Efficient flooding with passive clustering (PC) in ad hoc networks", *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 1, Jan. 2002, pp. 44–56.

[43] Y. Tseng, S. Ni, and E. Shih, "Adaptive approaches to relieving broadcast storms in a wireless multihop mobile ad hoc network", in *Proceeding of 2001 IEEE International Conference on Distributed Computing Systems*, Phoenix, AZ, Apr. 2001, pp. 481–488.

[44] Y. Yi, M. Gerla, and T. Kwon, "Efficient flooding in ad hoc networks using on-demand (Passive) cluster formation", in *Proceeding of the 2nd Annual Mediterranean Ad Hoc Networking Workshop (Med-hoc-Net 2003)*, Annapolis, MA, June 2003, pp. 44–56.

[45] A. Ephremides, L. E. Wieselthier, and D.Y. Baker, "A design concept for reliable mobile radio networks with frequency hopping signaling", *Proceedings of the IEEE*, vol.75, no. 1, Jan. 1987, pp. 56–73.

[46] S. Basagni, "Distributed and mobility-adaptive clustering for multimedia support in multi-hop wireless networks", in *Proceeding of the 50th IEEE VTS Vehicular Technology Conference*, vol. 2, Piscataway, NJ, Sept. 1999, pp. 889–893.

[47] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks", *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, Oct. 2002, pp. 660–670.

[48] M. Handy, M. Haase, and D. Timmermann, "Low energy adaptive clustering hierarchy with deterministic cluster-head selection", in *Proceeding of IEEE International Conference on Mobile and Wireless Communications Networks*, Stockholm, Sweden, Sept. 2002, pp. 368–372.

[49] O. Younis and S. Fahmy, " Heed: A hybrid, energy-efficient, distributed clustering approach for ad-hoc sensor networks", *IEEE Transactions on Mobile Computing*, vol. 3, no. 4, Oct.–Dec. 2004, pp. 366–379.

[50] S. Banerjee and S. Khuller, "A clustering scheme for hierarchical control in multi-hop wireless networks", in *Proceeding of IEEE INFOCOM'01*, Anchorage, AK, Apr. 2001, pp. 1028–1037.

[51] S. Bandyopadhyay and E. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks", in *Proceeding of IEEE INFOCOM'03*, San Francisco, CA, Apr. 2003, pp. 1713–1723.

[52] P. Ding, J. Holliday, and A. Celik, "Distributed energy efficient hierarchical clustering for wireless sensor networks", in *Proceeding of IEEE International Conference on Distributed Computing in Sensor Systems(DCOSS'05)*, Marina Del Rey, CA, June 2005, pp. 322–339.

[53] X. Li and P. Wan, "Constructing minimum energy mobile wireless networks", in *Proceeding of 2001 ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM'01)*, Rome, Italy, July 2001, pp. 55–67.

[54] H. Chan, and, A. Perrig, "ACE: An emergent algorithm for highly uniform cluster formation", in *Proceeding of the 1st European Workshop on Sensor Networks (EWSN'04)*, Berlin, Germany, Jan. 2004, pp. 154–171.

[55] H. Chan, M. Luk, and A. Perrig, "Using clustering information for sensor network localization", in *Proceeding of 2005 IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS'05)*, Marina Del Rey, CA, June 2005, pp. 109–125.

[56] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures", *Ad Hoc Networks*, vol. 1, nos. 2–3, Sept. 2003, pp. 293–315.

[57] A. D. Wood and J. A. Stankovic, "Denial of service in sensor networks", *IEEE Computer*, vol. 35, no. 10, Oct. 2002, pp. 54–62,

[58] S. Zhu, S. Setia, and S. Jajodia. "LEAP: Efficient security mechanisms for large-scale distributed sensor networks", in *Proceeding of the 10th ACM conference on Computer and Communication Security*, Washington, DC, Oct. 2003, pp. 62–72.

[59] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks", in *Proceeding of IEEE Symposium on Security and Privacy (S&P'03)*, May 2003, pp. 197–213.

[60] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz, and A. Khalili, "A pairwise key predistribution scheme for wireless sensor networks", *ACM Transactions on Information and System Security*, vol. 8, no. 2, May 2005, pp. 228–258.

[61] J. Hwang and Y. Kim, "Revisiting random key predistribution schemes for wireless sensor networks", in *Proceeding of the 2nd ACM workshop on Security of ad hoc and sensor networks*, Washington, DC, Oct. 2004. pp. 43–52.

[62] A. C. Ferreira, M. A. Vilaça, L. B. Oliveira, E. Habib, H. C. Wong, and A. A. F. Loureiro, "On the security of cluster-based communication protocols for wireless sensor networks", in *Proceeding of the 4th IEEE International Conference on Networking (ICN'05), vol. 3420 of Lecture Notes in Computer Science*, Reunion, Island, Apr. 2005, pp. 449–458.

[63] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and J. D. Tygar, "SPINS: Security protocols for sensor networks", *Wireless Networks*, vol. 8, no. 5, Sept. 2002, pp. 521–534.

[64] L. Oliveria, H. Wong, M. Bern, R. Dahab, and A.A.F. Lourerio, "SecLEACH-ARandom Key Distribution Solution for Securing Clustered Sensor Networks", in *Proceeding of the 50th IEEE International Symposium on Network Computing and Applications (NCA'06)*, Cambridge, MA, July 2006, pp. 145–154.

[65] S. Zhu, S. Xu, S. Setia, and S. Jajodia, "Establishing pairwise keys for secure communication in ad hoc networks: A probabilistic approach", in *Proceeding of the 11th IEEE International Conference on Network Protocols (ICNP'03)*, Atlanta, GA, Nov. 2003, pp. 326–335.

7

QUERY PROCESSING AND DATA AGGREGATION

Torsha Banerjee and Dharma P. Agrawal

University of Cincinnati, USA

7.1 INTRODUCTION

In wireless sensor networks (WSNs), a sensor is a device that can sense specific physical parameters of a system or a region of interest, convert the sensed data into electrical signals, and transmit the signals to a base station (BS) or sink using wireless radio. Unlike conventional sensors, wireless sensors are limited in energy because they are run by small batteries that are difficult or even impossible to be recharged in remote or hostile environments. Moreover, they have much smaller memory buffers and the embedded processors have relatively slower processing speeds. For these reasons, most of the research efforts made in this area have been directed to reducing the energy consumption of a network. The most energy-consuming components in a sensor device include the sensor transducer, embedded processor, and transceiver.

- *Sensor Transducer.* This component is responsible for capturing environmental parameters. It samples measured physical data and converts them into electrical signals. The main energy consumption of this component depends on its specific hardware and the application it is used for.

Wireless Sensor Networks: A Networking Perspective, Edited by Jun Zheng and Abbas Jamalipour

- *Embedded Processor.* This component is equipped with memory and is responsible for controlling the sensing, computation, and communication units [1]. There are two types of energy consumed in this component: switching energy and leakage energy. The switching energy is consumed when computation is performed while the leakage energy is consumed when no computation is being performed.
- *Transceiver.* This component consists of a transmitter for sending data and a receiver for receiving the data from its 1-hop neighbors. Energy is consumed for both transmitting and receiving data. To transmit a l-bit message over a distance d, following a multipath fading channel model, the overall transmission energy $E_{Tx} = l \times E_{Tx\text{–elec}} + l \times \in_{\text{amp}} \times d^4$, where $E_{Tx\text{–elec}}$ is the energy consumed for transmitting 1-bit of data and $\in_{\text{amp}} \times d^4$ is the amplifier energy. Correspondingly, the overall reception energy for receiving a l-bit message $E_{Rx} = l \times E_{Rx\text{–elec}}$, where $E_{Rx\text{–elec}}$ is the energy consumed for receiving 1-bit of data.

Due to the spatial correlation property [1], the physical parameters sensed by neighboring sensor nodes are usually similar in nature. On the other hand, a sensor node may sense similar data values during consecutive time periods because of the temporal correlation property [1]. To exploit the inherent redundancy (introduced due to this spatio-temporal correlation of data) in sensor data, it is desirable to remove or reduce the redundancy and transmit the relevant data to the BS or sink. The process or technique of reducing data redundancy by combining the data from neighboring sensor nodes is called data aggregation [2]. In a WSN, a user requests the BS for relevant information in the form of queries, which are also termed as spatio-temporal queries. Regardless of the network topology and the type of queries, a query packet from the BS eventually reaches the source nodes through a path computed by a routing protocol [3]. These nodes are the main sources of data being queried by the BS. After the query is received by the sensor nodes, which of them is responsible for routing the data back to the BS depends on the type of routing employed. In this context, there are two types of routing: address-centric (AC) routing and data-centric (DC) routing [4].

- *Address-Centric Routing.* In AC routing, a query is routed to a specific address or sensor node based on the address specified in the query. The sensed data is then sent from this specific location to the BS.
- *Data-Centric Routing.* In DC routing, a query is broadcast to all nodes within a range of interest specified in the query. The source node with an address specified in the query sends the sensed data directly to the BS.

After a query is sent by the BS to a sensor node, what to do next is to process the query, which is followed by data collection from sources and aggregation of the collected data. This chapter discusses the major issues and challenges for

query processing and data aggregation. It presents an overview of the major query processing and data aggregation techniques proposed thus far for WSNs. Section 7.2 introduces the characteristics of queries, discusses the challenges in query processing, and presents an overview of the major query processing techniques. Section 7.3 discusses the challenges in data aggregation and gives an overview of the major data aggregation techniques. Section 7.4 concludes with a brief summary of the chapter, as well as a brief discussion of future research directions.

7.2 QUERY PROCESSING IN WIRELESS SENSOR NETWORKS

To request the physical attributes in a region of interest, a user can pose a set of queries to the BS, which is then disseminated by the BS to the underlying sensor network. As in conventional database systems, a query describes a logical set of data that the user is interested in, but does not describe the actual protocol and software modules that the system uses to collect the answer set. A query processing technique is then applied based on the nature of the query. In this regard, several techniques can be designed for processing complicated queries, controlled flooding for retrieving a common set of data from all the nearby sensor nodes or snapshot querying for retrieving a summary knowledge of the entire region, and querying few nodes for that. The system can choose from these techniques and operator orderings for any given logical query. For example, to find the average temperature of a specific subregion of the network, the system may collect readings from every sensor node in the network and then filter the list of collected readings for the particular subregion the user is interested in.

7.2.1 Query Characteristics

There are two main types of messages involved in any query processing process: *query request* and *query (or answer) response*, as shown in Fig. 7.1.

- *Query Request.* A query request conveys a query from the BS to the sensor nodes in a region of interest.
- *Query Response.* A query response carries a query answer back to the BS for further processing by the user).

A typical monitoring scenario involves aggregate queries or correlation queries that give a bird's eye view of the environment being monitored, as well as more detailed queries zooming on a particular region of interest. The structure of a query can be viewed as being similar to an SQL (Structured Query Language) query with SELECT-FROM-WHERE-GROUPBYHAVING blocks to support various operations like join, projection, aggregation, and grouping [5]. From the point of view of a query, sensor data is considered as a single virtual

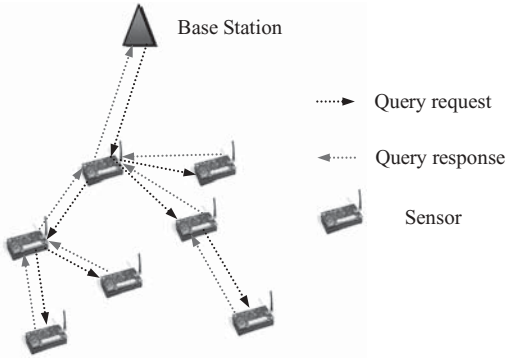


Fig. 7.1 Query processing components.

table with one column per sensor type. The underlying database system appends tuples to this table at well-defined intervals specified as the parameters of the posed query. The time between sample intervals is defined as an epoch and provides a suitable means for structuring computation to minimize power consumption [5].

Sensor queries are usually long running as a sensor network is mostly employed for continuous monitoring of an environment. Each sensor query is associated with a time interval of the form $[S, S + T]$ [6], where S is the time at which the query is submitted to the underlying sensor database and T is the number of time units for which it is running.

7.2.1.1 Query Operators. Sensor queries involve two types of data: stored data and sensor queries, which can be termed as relations (a predefined table format for storing data) and sequences, respectively, in database terminology. A sensor query can be defined as an acyclic graph of relational and sequence operators [6]. The inputs of a relational operator are base relations or the output of another relational operator, whereas the inputs of a sequence operator are base sequences or the output of another sequence operator. In other words, relations are handled using relational operators and sequences are manipulated using sequence operators [6]. In the operation process of a long-running query, relations and sensor sequences may be updated. An update to a relation R can involve various kinds of data-handling operations like an insert, a delete, or modifications of a record in R. An update to a sensor sequence S is the insertion of a new record. Since sensors sense external parameters and convert them into electrical signals for processing, the above mentioned operations mean that each sensor inserts incrementally the set of records produced by a signal processing function at the position corresponding to the time it was produced.

7.2.1.2 Query Classification. Before moving ahead, it is helpful to give a classification of queries. According to different criteria, sensor queries can be classified into different types, which are described as follows:

Criterion 1: Frequency of Query Response

Historical Query. An historical query is mainly used for analysis of historical data stored at a remote BS or any specific node in the network in the absence of a BS.

One-Time Query. A one-time query provides an instantaneous (or snapshot) view of the network. If a user wants to know data at a specific instance, the query is posed just once and data is returned only for that particular instance. It has the same start and end time.

Persistent Query. A persistent query is used to monitor a network for a continuous period of time. It is used when a user wants to know data periodically starting from a particular moment to a logically infinite time. In this case, the start time can be "now" and the end time is "infinite" if the user wants to know the result starting at the time the query is generated.

Criterion 2: Nature of Search Space

Spatial Query. A spatial query looks for a particular attribute value occurring in a given space of interest. For example, for a SQL-type query, this implies SELECT *attr_val* FROM *sensor* WHERE *loc* = (20, 30).

Temporal Query. A temporal query looks for a particular attribute value occurring during a specified period of time. For example, for a SQL-type query, this implies SELECT *attr_val* FROM *sensor* WHERE *time* = 6:50 am–2:30 am.

Spatio-Temporal Query. An example of spatio-temporal queries is SELECT *attr_val* FROM *sensor* WHERE *loc* = (40, 25) and *time* = 1:30 pm–4:30 pm.

Criterion 3: Semantic Nature of Query Response

Exploratory Query. An exploratory query retrieves a single record from the database. An example of exploratory queries is *Return the record on 01-18-07 at 5:00 pm*.

Monitoring Query. A monitoring query is defined as a more complex query that searches in the database and returns more than one record within the same RDF file (a file format that makes the semantic network information machine readable). An example of monitoring queries is *Return all records for 01-18-07*.

Range Query. A range query requires importing several RDF files into a particular database model [7] and returns the data satisfying the query in each file. An example of range queries is *Return all days in the month of March when only fan number 1 was on*.

Criterion 4: Range of Data in Query Response

Filtering Query. A filtering query returns sensor data only if it is within a specified range or has a condition to be satisfied. In case of a filtering query

with a range specified, data that are out of range are filtered out and results only satisfying the range condition are returned back to the user.

Non-filtering Query. A non-filtering query does not have any condition and it returns all raw sensor data.

Criterion 5: Number of Attributes in Query Request

One-Dimensional Query. A one-dimensional (1D) query requests results with only one type of attributes or sensed data. An example of 1D queries is *Return all the data sensed in the temperature range 30–45 °C*.

Multidimensional Query. A multidimensional query requests two or more types of attributes or sensed data. An example of multidimensional queries is *Return all the data sensed in the temperature range of 30–45 °C and humidity of 80–85%*.

7.2.2 Challenges in Query Processing

There are several challenges in query processing. Upon a user request, the BS broadcasts the query request throughout the network. The interface for sending and receiving a query should be simple to handle the query. It is also crucial to define a robust database for temporary storage and retrieval of query attributes, that is, responses of a query. The major challenges in query processing are described as follows:

1. *Query Broadcast.* Query processing involves broadcasting of a query from the query originator, which in most cases is the BS or sink node, as shown in Fig. 7.1. A complex query can be broken into multiple simple queries, which are broadcast by the BS at a high power level to all sensor nodes in a single powerful transmission or sent to nearby sensors, which can broadcast the query further to the rest of the network. Due to the broadcast nature of communication in WSNs, each node may receive the same query more than once and the major challenge is to select a particular query for answering. For each query, a node should process only the first query request received, discarding subsequent copies of the same message. When a query is received, for commonly used DC type routing, the node first broadcasts the query, and then selects the locally stored data relevant to the query (if any), waits for its neighbors' data packets and merges them with its own, and returns the merged packet to the neighbor that it received the query from. Once the query originator node receives the relevant data from all nodes (the query response as shown in Fig. 7.1), it can get a complete answer to the specific query.
2. *Querying Interface.* The querying interface should be substantially simpler than the underlying operating system's (TinyOS [8] is the currently used operating system especially suited to mote capabilities) embedded programming model. The interface should also allow users to collect information and process the information in an efficient way.

3. *Efficient Database for Querying.* A language needs to be developed for querying and task assignments, as well as a database that can be readily queried. A key component of the database system of each sensor is called query proxy [5]. Query proxy provides higher level services through queries. Architecturally, it should lie between the network layer and the application layer of each sensor node. TinyDB [9] is the currently developed database for query storage and retrieval in WSNs.
4. *Uncertainty and Transience in Sensor Readings.* Since most of the attributes sensed by sensor nodes are environmental parameters, they are associated with some inherent uncertainty caused by noise in the environment. Moreover, malfunctioning of a sensor can also generate inaccurate data, and adverse sensor placement (e.g., placing a temperature sensor directly beside an air conditioner) may bias individual sensor readings. In addition, a quiet sensing environment may suddenly have an occurrence of a major catastrophe, in which case the sensor readings will change dramatically and rapidly.
5. *Probabilistic Queries and Answers.* Since a WSN cannot obtain all possible data, any reading from a sensor is approximate or probabilistic, that is, it only represents the true state of the world at the distinct instants and locations where sampling was performed. In another word, the query answers are obtained at specific sensors based on the discrete nature of the query requests, thus both having a probabilistic nature [10]. A solution to handling this type of query requests and answers is to incorporate a statistical model in the query processing architecture, which models the real-world process. This model helps in making the system more robust against failures and sensors running out of energy by accounting for spatial biases and extrapolating data in regions where sensors have run out of energy.

7.2.3 Sensor Selection for Query Processing

Each sensor node consists of one or more sensors attached to it and connected to the physical world. The types of sensors can be classified into temperature sensors, light sensors, or pressure sensors that can determine the occurrence of events (e.g., the sudden change in their readings) in their surrounding area. Therefore, each sensor can be considered as a separate data source that generates records with different fields, for example, the distinct sensor ID, location of the sensor that generates the reading, a time stamp denoting the time at which the data are sensed, and the value of the reading. Based on the nature of a query, a subset of sensors is selected for returning their data values within the range of the query. For example, a query with GROUP BY and HAVING clauses computes the average value for each set of sensors and rejects data values with an average smaller than some prespecified threshold. It is also possible to use JOIN operators as in databases to either reduce or augment the resulting data size [5].

In addition, it is possible to handle location-based SQL-type queries like SELECT *temperature* FROM *sensors* WHERE *location* = (x, y) or Give maximum acoustic data in the target area ($10 £ x £ 40, 10 £ y £ 40$) every 3 seconds. By using different types of query operators as discussed, it is possible to filter the data set to generate useful results and thus reduce the overall size of data exchanged between sensors.

7.2.4 Query Processing Techniques

There are broadly two techniques for processing sensor queries: the warehousing technique and the distributed technique. In the warehousing technique, processing of sensor queries is kept completely separate from the access to the underlying sensor network, where the actual data is sensed and stored. It involves two processing steps. First, data is extracted from the sensor network in a predefined fashion and is stored in a centralized database located on a unique front-end server. Thus, this type of query processing is suitable for processing predefined queries over historical data [5]. In the distributed technique, the query workload determines the data that should be extracted from sensors. The distributed technique is more flexible because it enables different queries to extract different data from the sensor network. Meanwhile, it proves to be more efficient because only relevant data are extracted from the network. In addition, the distributed technique allows the sensor database system to efficiently utilize the computing resources on the sensor nodes. The sensor query in this case can be evaluated at the front-end server, in the network itself, at the sensors, or at some mixture of the three. However, the goal of this chapter is not to discuss query processing under these headings, but instead to dig into the methods for query processing based on the region of the sensors queried.

Once a query is sent to the BS by a user, it is then the job of the BS to handle and process the query in an energy-efficient manner. The BS floods the query request to all sensors in the network. Based on the content of the request, the specific sensors, called sources of data, respond to the query and send the response back directly or through intermediate nodes to the BS. Sometimes the user wants to query a portion of sensors in the network for specific answers instead of flooding the entire network. In this case, a snapshot query is sent to the sources. Again, sometimes it is crucial to optimize the query in terms of cost, time, and power so that the lifetime of the network is given the highest priority. Acquisitional query processing does this job.

7.2.4.1 Query Flooding. A query can be flooded among all nodes in a WSN; that is, broadcast to all the nodes reachable from the BS.

FullFlood

FullFlood is a flooding technique described in Ref. [11]. With *FullFlood*, the answer to a query is guaranteed to reach the query originator. Each node floods the query to all the nodes within its radio range, as illustrated by

the bold arrows in Fig. 7.2d. However, this operation causes unnecessary communication overhead. A variation of this technique is to contact a fixed set of nodes that are relevant to the query within a predefined spatial window (to limit the number of nodes to be contacted) [11]. This variation reduces congestion drastically because less query traffic is generated in the network, thereby reducing the time for the answer to the query (query response) to reach the BS.

Spatio-Temporal Window

Spatio-Temporal Window (*STWin*) is a technique for query processing [5], which involves two processing phases, which are described as follows:

- In the first phase or *GREEDYDF* phase, a path needs to be found to a node inside *STWin* for given a query originator (Q_O), which can assume the role of the query coordinator (Q_C) to broadcast the query to all potential source sensor (SS) nodes.
- The second phase has two variations: *WinFlood* and*WinDepth*. In *GREEDYDF*, Q_O routes the packet to its neighbor that is closest to Q_C. The optimal location of Q_C is the center of *STWin* as represented by the rectangular area in Fig. 7.2. However, *GREEDYDF* does not assure the establishment of a routing path within *STWin*. In *WinFlood*, when a node receives a query, it broadcasts the query to all its neighbors (irrespective of whether the neighbors are inside or outside *STWin*) if it itself is within *STWin*. This flooding stops when the query reaches a node outside the window. To reduce the number of nodes involved in the routing process, an alternative variation called *WinDepth* is used, which employs depth first search traversal to individually send the query to the neighbors within *STWin* only if they have not yet received the query. Fig. 7.2b and c illustrate the request phases of *WinFlood* and *WinDepth,* respectively, whereas Fig. 7.2e and f depict the corresponding response phases.

However, *WinFlood* may prove to be more cost effective assuming the fact that the cost of a single broadcast message is lower than that of multiple point-to-point messages. With the increase in node density, *FullFlood* involves more overhead as the number of routing messages increases. However, increasing the window size in [11] induces flooding over an increased number of nodes, eventually degenerating into the *FullFlood* method. For a larger query window, *FullFlood* uses less energy per node compared to Ref. [11] although it spans the entire region.

Directed Diffusion

Directed diffusion [13] is another technique for disseminating queries across the entire sensor network by flooding. In directed diffusion, the data

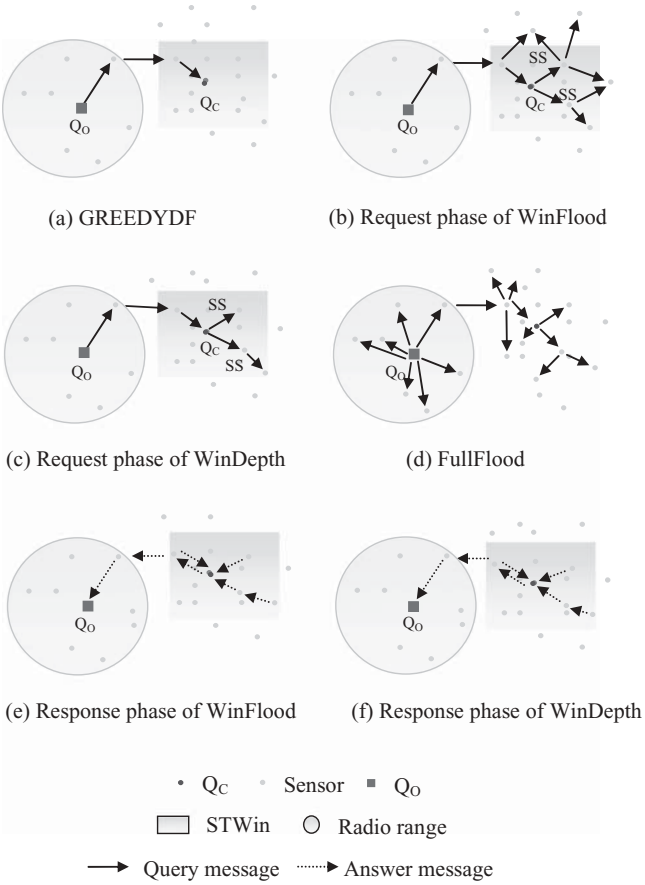


Fig. 7.2 Illustration of basic methodology for query processing.

generated by a sensor is identified by an attribute value pair. The query answer generated in response to a named interest is data centric. The dissemination sets up a gradient, which draws the event data toward the disseminators of the queries. This gradient is specific to the value of the data wanted and the direction in which the data needs to be sent. To receive the temperature data in a specific rectangular region over a specific period of time, the diffusion query looks like the following: type = temperature, interval = 10 ms, duration = 40 s, and Rect = [–200,200,400,600].

Directed diffusion frequently exploits the spatio-temporal correlation in the data streams and is more applicable for dense, static sensor networks. Therefore, they are not highly suitable to the low-correlation situations of other kinds of networks, for example, mobile sensor networks.

7.2.5 Snapshot Querying

WSNs are prone to node failures due to nodes' running out of their battery. Therefore, it is important to use a data-centric network in such scenarios, which allows access to the collected measurements in an integrated manner. In case of a node failure, the network should be able to self-heal by using surplus stand-by nodes to keep the network running. Thus, nodes should be able to generate a model of their surrounding environment so that they can represent the sensed data of the entire network. If, instead of allowing all sensor nodes receiving a query to respond, only a few preselected nodes are allowed to respond to a query, energy efficiency of sensor nodes can be further improved. This type of querying is called snapshot querying. The locations and values of the representative nodes present a picture of the value distribution in the entire network.

In Ref. [12], an example of snapshot querying is presented. In this example, after receiving a query, neighboring nodes synchronize on a localized basis to select a set of representative nodes for answering the queries, denoted by R_N. A node N_i can obtain the knowledge of the value, x_j', sensed by neighboring node N_j (i.e., the rectangular node) either through N_j broadcasting its readings to all the nodes within its radio range, as shown in Fig. 7.2d, or through N_j sending explicit periodic announcements to its neighbors. Among all the neighboring sensor nodes that can be considered as representatives of N_j, the one that has the largest number of nodes in its neighborhood is selected as the representative of N_j. In case of a tie, the node with the largest ID, N_i, is selected as the representative. In case N_j fails or is out of range from other sensor nodes, N_i can serve as the representative of N_j for sending its readings. If x_j is the actual reading at N_j, N_i can represent N_j if $d(x_j, x_j')$ is $\leq T$, where d can be a relative, absolute, or sum-squared error (defined as the deviation of the approximate reading of N_j, x_j' from the actual reading sensed by N_j, x_j) and T is a prespecified threshold. To ensure that a representative node always exists, N_j transmits a periodic beacon signal to N_i, which includes its current reading. In case N_i does not respond or if $d(x_j, x_j') > T$, N_j invites new representative nodes from its local neighborhood.

To make the readings coordinated with each other, each node N_i maintains a cache_line consisting of a list of pairs $x_i(t)$ and $x_j(t)$ collected at the same time, that is,

$$\text{cache\_line}(N_i) = \{(x_i(t_1), x_j(t_1)), (x_i(t_2), x_j(t_2)) \ldots (x_i(t_n), x_j(t_n))\}.$$

The cache_line is always kept fresh by proper cache admission and replacement strategies [12]. If the cache is full and a new entry comes in, the gain obtained by giving an entry to this new value at the cost of removing an existing value is measured and a replacement occurs if a positive gain is realized [12].

Time Series Snapshot

It is often necessary to report a time series snapshot of data to the BS for scientific analysis [14]. For example, to support a distinctive historical

query *Find the average humidity in a certain region for each year from 1989 to 1994*, the BS needs to maintain a time series snapshot of humidity within an acceptable error scope for the monitored region. The given monitored region is partitioned into a set of subregions and the value distribution in each subregion is bounded by a predefined range of the attribute value. To find a snapshot of the attribute at a given time in the region, the readings from sensors located close to the borders of all subregions are obtained. In this way, sensors that are not located at the surrounding area of the boundaries do not have to transmit their data to the BS, thus controlling the overall energy consumption.

7.2.5.1 Acquisitional Query Processing. Acquisitional query processing is a novel and energy-efficient technique for querying a network, which answers significant questions like when and where data should be sampled and how often the sampling should be done. Smart sensors, which are defined as … a sensor that provides functions beyond those necessary for generating a correct representation of a sensed quantity … [15], are employed to answer these types of queries in order to save a substantial amount of energy as compared to those passive systems, where data is considered to be static. Attributes in a smart sensor network can be high cost as well as low cost. By establishing a correlation between these types of attributes, the cost of answering a query can be significantly reduced by selecting the low-cost attributes in place of the high-cost ones [16].

Query Optimization

For optimizing the query in terms of cost, time, and power involved in its retrieval, a multidimensional probability distribution is associated with attributes and an exponential-time algorithm in Ref. [16]. Query optimization is essential because the cost of acquiring a result per second is directly proportional to the cost of executing the query. On the other hand, the cost of retrieving a reading from a table of query plans (which is defined as an ordered set of steps used to access information from a huge collection of sensor data and is expressed in terms of one or more attribute values like temperature >20 °C and light <100 Lux), is inversely proportional to the selectivity of a predicate (which is defined as the fraction of rows selected from the table by the predicate if the condition is satisfied). For example, in a query containing two predicates, the predicate on temperature has a high probability of being false during night and the predicate on light is likely to be false during day based on simple knowledge of physics. Hence, depending on the time of the day when the query is being evaluated, the selectivity; that is, the number of rows selected from the table varies and therefore the expected cost of this plan could alter automatically.

In the experiment, two data sets are considered. The first data set is a lab data set of 400,000 light, temperature, and humidity readings collected every 2 min for quite a few months from ~45 motes. These readings also comprise the IDs of the sensors generating the data, time of day, and

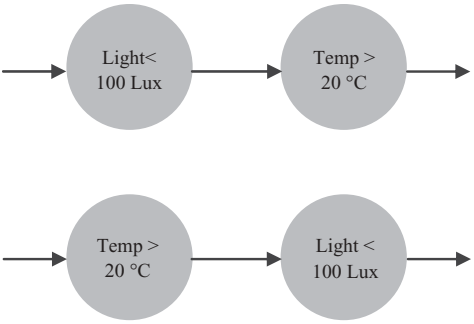
battery voltage. The second data set, called garden, is collected from a set of 20 Berkeley motes deployed in a forest. Each mote is equipped with two temperature and light sensors. There is also a single sensor for humidity and air pressure. To reduce the complexity of the algorithm, the domain size of the data set is reduced by subsampling. The plans generated by the algorithm are built using a set of training data, and evaluated on a disjoint set of test data. Readings from these two sets are derived from nonoverlapping time windows. The latter simulates the situation where historical data is used to create a future model that is run later within the sensor network. It is observed from the experiment that queries with a feature of very low selectivity do not show a major performance gain. The overall advantage of this algorithm is that more expensive attributes (e.g., light) are predicted by observing cheaper attributes (e.g., temperature and sensor voltage) and exploiting the spatial correlations of the sensors using a statistical model instead of measuring the expensive attributes directly. However, this overall process of understanding and learning the correlation among attributes takes a substantial amount of time, energy, and node samples. Again, this algorithm involves learning a conditional plan based on an initial training sample of the data, and then utilizing this plan throughout the query execution process. This initial training is done offline and requires a huge amount of collected training data. Also, the learning of the conditional plan involves complex decision tree building algorithms.

Binary Decision Tree Approach

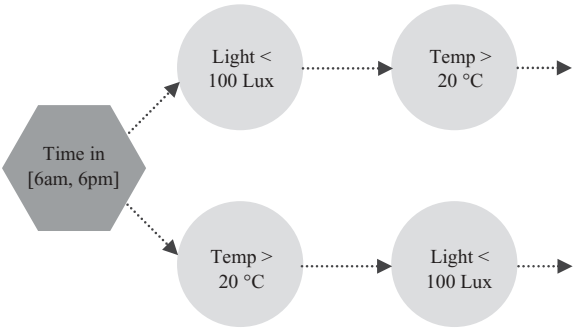
To execute queries efficiently, a binary decision tree is built on a conditional plan P, which splits the plan at an internal node n_j [denoting the conditioning predicate, $T_j(x)$] into two alternate conditional plans $P_{T_j(x)=T}$ and $P_{T_j(x)=F}$ (as shown in Fig. 7.3), where each plan is an independent subproblem, covering a disjoint subspace of attribute-domain space. At the node n_j, the query processor evaluates $T_j(x)$ and according to the value of the evaluated predicate, one of the plans is executed. The expected cost of plan P is the sum of the cost starting from the root node, *Root* (P), and the expected cost of the value of each new predicate, T_j. A *GreedySplit* algorithm is proposed in Ref. [16], which starts with a single root node and performs a single split on that node. The algorithm maintains a priority queue (ordered on the expected cost of splitting a node) of nodes and splits the node n_i into two children n_i' and n_i'', which are then appended to the queue. In this way, the cost of the plan gets updated recursively with each new split.

TinyDB Approach

In Ref. [9], an acquisitional query processing (ACQP) system is presented, which uses a distributed query processor called TinyDB in each of the nodes in a sensor network. TinyDB incorporates acquisitional techniques



(a) Traditional plan



(b) Conditional plan (selectivity based on time of day)

Fig. 7.3 Example of a query involving two predicates.

and runs on the Berkeley mote platform, on top of the TinyOS [8] operating system. TinyOS comprises a set of components for running and accessing the mote hardware, and a "C-like" programming language called nesC. The OS has been ported to a range of hardware platforms that include UC Berkeley's Rene, Dot, Mica, Mica2, and Mica2Dot motes. In the design of the ACQP system, the questions addressed include when samples for a specific query should be taken, what sensor nodes have data pertinent to a specific query, in what order samples for this query should be taken, and how sampling should be interleaved with other operations. The data structure in TinyDB is a table called *sensors* and there are sensor tuples associated with the table. Each tuple denotes one row per node per instant in time, with one column per attribute (light, temperature, etc.) that the device can create. Records in this table are acquired only when needed to satisfy the query and are usually stored for a small period of time or delivered directly out of the network. The table is partitioned across all of the devices in the network with each device producing and storing its own readings. To compare readings from different sensors, those readings must

be collected at some common node, which could be a root for a hierarchical network. Queries in TinyDB consist of a SELECT-FROM-WHERE-GROUPBY clause that supports selection, join, projection, and aggregation of data. The semantics of SELECT, FROM, WHERE, and GROUP BY clauses are 'SQL-like'. For the query: SELECT node-id, pressure, humidity, FROM sensors, SAMPLE PERIOD 5 s FOR 40 s, it specifies that each sensor should report its own node-id, the attributes it senses; that is, pressure and humidity (contained in the virtual table sensors) once per 5 s for 40 s. The results of this query are sent to the root of the network via a multihop topology. The output to the user from the root consists of a stream of tuples, clustered into 5-s time intervals. Each tuple includes a time stamp consequent to the time it was created. The ACQP system helps in designing query processors, which increases the longevity of the battery and thus reduces the maintenance cost of the overall sensor system. A key issue with TinyDB, as with other generic query processors, is that the same code is installed on each sensor irrespective of whether that node will need or be able to support all the functionalities of the installed query system. Consequently, a large query processing system has to be installed on all nodes of the network regardless of their capabilities.

7.3 DATA AGGREGATION IN WIRELESS SENSOR NETWORKS

Sensor nodes are usually deployed in large or even huge number for continuous monitoring of a system or an area of interest. Because of the dense pattern of sensor deployment, neighboring sensor nodes may sense similar data on a specific phenomenon, which is referred to as spatial correlation. Since sensor nodes are run by battery power, it is critical to perform every operation in an energy-efficient manner. For this purpose, it is desirable for a sensor node to remove the redundancy in the data received from its neighboring nodes before transmitting the final data to the BS. Data aggregation is an effective technique for removing data redundancy and improving energy efficiency in WSNs. The basic idea is to combine the data received from different sources so that the redundancy in the data is minimized and the energy consumption for transmitting the data is reduced in the aggregation process. The data-centric nature of WSNs makes data aggregation a crucial task [4]. Sometimes, however, users may be interested in knowing the attribute values at some specific locations. In this case, the locations of reporting sensor nodes, which can be geographical coordinates, should not be left out in performing data aggregation [17].

7.3.1 Challenges in Data Aggregation

The challenges in data aggregation include four major aspects: energy efficiency, timing control, application orientation, and QoS support. Their impacts on a data aggregation scheme are described as follows, respectively:

- *Energy Efficiency.* Since sensor nodes have limited battery power, efficient sleep scheduling protocols should be employed for saving the energy of idle sensor nodes. Since most of the sensor nodes in a WSN are in a close proximity, the attribute data sensed by neighboring nodes are usually similar, especially in an environmental monitoring system. For example, in the case of an event, some physical attributes exhibit a gradual and continuous variation over the two-dimensional (2D) Euclidean space by following a diffusion phenomenon [18]. Therefore, it is desirable to aggregate the data through collaboration among neighboring sensors to remove the redundancy in the data so that only the useful data are sent to the BS. This can save energy consumed by the sensor nodes for sending redundant data all the way to the BS.
- *Timing Control.* In addition to saving energy, it is also important to perform data aggregation within reasonable time bounds.
- *Application Orientation.* A WSN is usually designed with a specific application in mind. For those applications requiring only a summary of the attribute data in a large region instead of minute details, data compression becomes necessary for reducing the cost of communication in terms of bandwidth and energy.
- *QoS Support.* In addition to energy efficiency and timing control, sometimes it is also necessary to meet specific application requirements in terms of quality of severe (QoS) in data aggregation. In this context, there is not much work already done in WSNs. Existing related work is briefly described in Section 7.3.2.

7.3.2 Data Aggregation Techniques

There are a variety of techniques for data aggregation in WSNs. Since conserving energy is one of the most important challenges in data aggregation, an efficient data aggregation technique should be able to balance the amount of energy consumed by each sensor node in each round of data gathering. Moreover, a WSN is mostly designed with a specific application in mind. According to the requirements of the targeted application, there may be some QoS metrics that need to be guaranteed, for example, packet loss and delay. In the following sections, we will introduce the major data aggregation techniques proposed for WSNs, including energy-efficient data aggregation, application-oriented data aggregation, and QoS constrained data aggregation.

7.3.2.1 Energy-Efficient Data Aggregation.

1. *Use of Simple Mathematical Operators.* Since energy is one of the most crucial parameters for a longer network lifetime, most of the research efforts have been directed to this issue. A novel idea for population estimation (i.e., tracking the total number of animate or inanimate objects) is

proposed in Ref. [19] for sensor-based resource inventory applications. In this population estimation algorithm, aggregate operators like COUNT and SUM are sensitive to duplicate values. Any population estimation essentially uses the SUM operators for calculating the total count of population. Hence, eliminating duplicate readings is very important in overlapping sensing regions. The proposed algorithm gives an estimated count of the population of a specific object type. The problem of the estimated count to be biased toward the methodology used has been overcome in Ref. [19]. Individual objects are distinguished by the sensors by employing RFID (Radio-Frequency Identification) Technology or by specifically sampling the physical locations of the objects. A conventional approach is to do gradual aggregation of sensed data packets as they go up along the data aggregation tree.

2. *Aggregating Object Reports at Source* Another approach is to aggregate the object reports right at the source node and then send the object report to the BS for processing [20]. As illustrated in Fig. 7.4, we see that each sensor (in a rectangular shape) can sense a target in its sensing region. Each sensor reports the target count and the sensing region to the BS. The required population of sensors is approximated so as to fully cover the region of interest by calculating lower and upper bounds of the estimation as follows. Using the maximal independent set method in graph theory, the disjoint regions are identified. In Fig. 7.4, they are A and C; B and C. The reported counts from each of these sets are summed and the lower bound is set to the maximum sum. The upper bound considers the regions where sensing spots are covered by only a single sensor node to count the objects, at least once in these irreplaceable regions.

According to the simulation results in Ref. [20], the lower bound gives readings consistently lower than the exact population by 30%, whereas the upper bound is found to be consistently higher than the exact population by 70–80%. The overall energy savings occurs on the order of 65% and is scalable with the increase of sensed object population. This constant estimate range gives a good

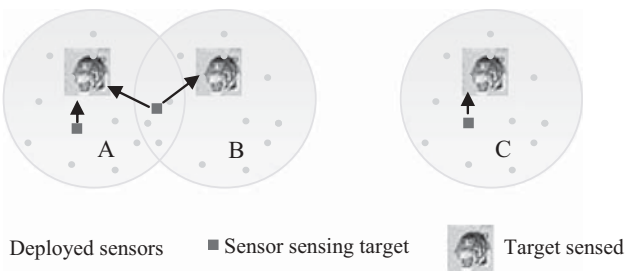


Fig. 7.4 Sensing regions performing object tracking and location estimation.

estimation of the actual population. However, this technique produces an energy-accuracy trade-off and has an increased computational complexity if a higher accuracy needs to be achieved. To achieve better results at the same time not compromising on energy savings, improved statistical techniques need to be employed.

7.3.2.2 Neural-Network-Based Data Aggregation. Data aggregation can be achieved in an innovative manner by combined knowledge of artificial intelligence and networking concepts. A possible technique for compressing redundant data is to perform dimensionality reduction [21]. This can be done in some scenarios by employing a neural-network-based algorithm, which runs based on an iterative pattern of adjusting weights to learn irregular inputs. A neural-network-based algorithm can be applied on Smart-It units (i.e., sensor motes consisting of a sensory module and a communication module [21]) to report cluster numbers, where a specific sensory input pattern is classified instead of reporting raw data.

As shown in Fig. 7.5a, a number of Smart-It units form a set of clusters whose readings (inputs are analog signals) are collected by the cluster heads (CHs), where a FuzzyART is implemented [22]. Each CH is also a Smart-It unit itself and is an ART (Adaptive Resonance Theory [23]) network, which consists of three layers: F_0, F_1, and F_2, as shown in Fig. 7.5b. The top-most CH implements ART1 (input signals are binary) and classifies the classification readings sent to it by the lower clusters. The input layer of the CH, F_0, stores the input bits that are transmitted up to the category layer, F_2 (for classification), via the comparison layer, F_1. Each input pattern is compared to the reading stored at each category node n_i and an activation, T_i, is calculated [22] for each category node. The node with maximum T_i declares as the winner and its weight is compared to the current input at F_1. If the matching condition satisfies a prespecified threshold (a system parameter) called ρ or vigilance, then n_i captures the current input and updates its weight.

The comparison cycle described above continues until a stored category is found matching the input pattern. If not, the network learns by assigning a new category node in F_2. In this way, each input pattern is finally classified to some cluster unit and this cluster unit is sent to the corresponding CH. Let n be the number of sensors in each cluster unit and k be the number of units reporting to a CH. Let the size of raw sensor data be r bytes and the number of different categories, which is an integer, in each unit be c bytes. The energy savings in communications, E_s, can be calculated as the ratio of the input size to the output size as follows:

$$E_s = \frac{n.k.r}{k.c} = \frac{n.r}{c},$$

where E_s is of the order of k and this huge proportion of energy savings is obtained due to $r >> c$.

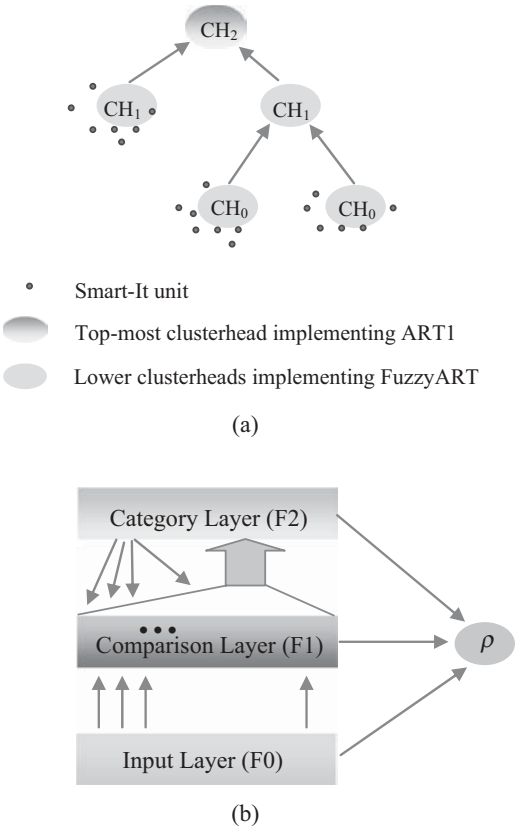


Fig. 7.5 (a) Data collection by CHs from lower clusters. (b) Three internal layers of each CH.

7.3.2.3 Delay-Constrained Data Aggregation. In data aggregation, a sensor node needs to combine its own data and the data received from its 1-hop neighbors, and then send the combined data to the BS. Hence, the node needs to wait for its neighbors' data before transmitting to the BS. This introduces a delay in the data transmission, which presents a timing control problem in data aggregation. To address this problem, a data aggregation tree (DAT) based approach can be used, in which different tree nodes respond to a query with different delays depending on the tree levels at which they are located. The delay of data aggregation can be reduced by employing a finite state machine (FSM) based feedback control scheme [24].

FSM Based Feedback Control Scheme

The waiting time at each level of the DAT tree to respond to a query is adjusted according to the output of the FSM, while maintaining a substantial level of accuracy. Figure 7.6 shows a data aggregation tree. For ease

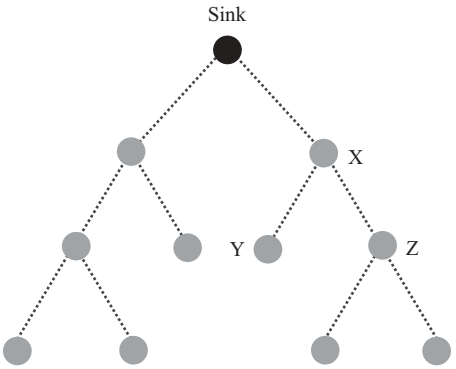


Fig. 7.6 Data aggregation tree.

of exposition, only leaf nodes of the tree are assumed to generate data. In this tree, the timing delay for data aggregation at node X is affected by its two children nodes Y and Z. If X waits for Z's packet, it faces a longer delay, as Z needs to first aggregate its children's data with its own data before sending the data up to X, unlike node Y, which does not have children and therefore does not have to spend time for aggregation. The initial part of the FSM algorithm keeps the waiting time to be high so that more query responses can be collected for better accuracy. Once the accuracy level reaches a stable value, the waiting time is adjusted accordingly to also control the time delay. A setup phase is followed to distribute the necessary parameters like depth among the tree nodes, which are useful for tree construction. Based on the depth of the tree, the maximum latency, and the optimal number of responses, the root or the sink calculates an approximated value for the data aggregation period, T_{n+1}. After the setup phase, the sink broadcasts this maximum aggregation period, T_{n+1} along with data requests. The hop count information from leaves to all nodes in the tree is disseminated because it is needed for calculating T_{n+1} later on. According to the level of the tree a node belongs to, the node adjusts its T_{n+1} and waits for this period to receive data from the lower levels. Then it sends the data to the nodes at the upper levels, and so on.

Variants of FSM Based Scheme

Based on the received responses from its nodes, the sink calculates a more appropriate value of T_{n+1} for the next period using the FSM and the optimal number of responses (N_{opt}), which is again dependent on the system topology, and broadcasts the data request along with the current period of data aggregation (T_n) downward. In the first version of the FSM based scheme, at each step of calculation in the aggregation period, the optimal number of messages received is compared with the actual number received. If fewer messages are received, the aggregation period is

increased by one atomic unit. If more messages are received than the optimal number, the aggregation period is decreased accordingly.

In the second version, the scheme is made to adapt to the performance and is based on the difference between the actual number of received responses and the desired optimal number of responses. This decision is made based on whether the aggregation period needs to be changed at a faster speed, instead of the addition/subtraction operations at the FSM. Data responses are sent bottom-up starting from the leaf nodes to the sink. Each data response contains some extra information, indicating the total number of responses that have been aggregated into a single message. Based on this information, the next aggregation period T_{n+1} is calculated as follows:

$$T_{n+1} = T_n - 1 \quad if\ N_{\text{rec}} > N_{\text{opt}},$$
$$T_{n+1} = T_n + 1 \quad if\ N_{\text{rec}} < N_{\text{opt}},$$

where N_{rec} is the number of received responses. This scheme provides energy-efficient data aggregation while not compromising on the overall delay of the data collection process.

7.3.2.4 QoS Constrained Data Aggregation. In addition to optimizing energy efficiency and timing control in data aggregation, it is also essential to meet specific task requirements and achieve a desired QoS level in performing data aggregation. In this context, not much work has been done to address specific QoS constraints in terms of delay, packet loss, and so on.

One QoS strategy for data aggregation has been proposed in Ref. [25], in which sending reports to the BS is deferred based on several factors. This strategy introduces a distributed mechanism for data aggregation to meet various QoS requirements, for example, end-to-end latency and measurement accuracy. The intermediate nodes between a source and a destination solely perform data aggregation. They independently determine whether or not to perform data aggregation randomly with some specific precalculated probability based on the resource conditions and the specific task requirements.

Unlike Ref. [24], no clustering or tree formation is needed in Ref. [25] and a decision on when and where to perform the task is based on the availability of local information. The end-to-end QoS constraint considered is the end-to-end latency requirement D of a transmitted packet (also called report). If the report is delivered to the sink (BS) within D after its initial generation, it is considered a successful delivery. At an intermediate node, if the delay between the origination of a packet and the receipt of the packet is larger than D, the packet is discarded. Three cases may arise based on the delay constraint.

- If the delay constraint can be satisfied, the intermediate node (aggregator) defers the report for a prespecified time interval with a prespecified

probability, during which it processes and aggregates any packets that arrive and generates a new packet before transmitting it to the next hop.

- If the delay constraint can be satisfied only if the report is not deferred, the aggregator tries to forward this report to the next hop.
- If the delay constraint cannot be satisfied in any case, the aggregator discards the report to avoid further wasting of any additional resources.

The lower bound of P_{succ} (the probability that a packet is delivered to the sink within the delay constraint D) is derived to be equal to the probability that a report experiences at most C times of data aggregation and reporting along its path where C is the upper bound on the reporting time. The results reveal that a smaller D will result in a lower P_{succ}.

Balancing Trade-Offs

Apart from the above results, a data aggregation scheme is also proposed to balance the trade-offs among energy-efficiency, delay requirement, accuracy, and buffer overflow probability in Ref. [25]. A periodic data reporting is done at every t units. If at some point the change of the sensed data is beyond a predefined threshold u, a sample is also collected independent of the time. Each intermediate node also collects and saves N samples before it generates a packet for transmission to its next hop node. The parameters t and u determine the measurement quality or accuracy. As N and t increase, the energy efficiency of the data collection and transmission increases, while the corresponding delay and buffer size requirements at each aggregator increase as well.

On the other hand, the accuracy of the collected data increases as u and t decrease. According to the simulation results in Ref. [25], the probability of buffer overflow will increase when the rate at which data is consumed from the buffer decreases, the network load increases, or the channel conditions deteriorate. Also, when the network load increases, the waiting time at each aggregator increases, implying that performing data aggregation can reduce the network load and therefore result in the reduction of the end-to-end delay in the network.

Packet Delivery Probability

It is also observed that the successful packet delivery probability of the system with a delay constraint under the QoS strategy is better than the estimated probability under the strategy that does not discard any packets due to the delay constraint. Energy consumption decreases as the deferred period increases. This happens as the average number of packets that can be used for data aggregation increases with an increase in the deferred aggregation period. Similarly, network lifetime increases with an increase in the deferred period because its increase implies reduced communication traffic due to increased aggregation. It is also observed that as the

probability of deferring a report increases, the total energy consumption of the system decreases during the same operation period, while at the same time the average delay increases. In addition, the data loss decreases when the value of N increases. This result is due to the fact that with aggregation the total traffic load in the network, as well as the corresponding communication overhead, decreases.

7.3.2.5 Data Aggregation for Range Query. A novel protocol, which combines the idea of data aggregation and dissemination to gather data from regular- and irregular-shape ranges in a grid-based sensor network, is proposed in Ref. [26]. The network consists of mobile sinks (equipped with PDAs), which request nearby static sensors to sense event data, and a specific sensor in a grid, which has knowledge about the event, is designated as the head of the grid. For a regular-shape range query, a mobile sink designates the range (e.g., a range of rectangle) for data aggregation, and requests the source (i.e., a static sensor sensing data) to collect the data in the designated range. The head of the grid receives the aggregated data from the source and sends the data to the mobile sink. Thus, it is also called an agent. The aggregated data from a source, which are a collection of packets reported to all the children of the source, are reported to the nearby head of the grid. To account for its mobility and to ensure that the mobility does not affect the data transmission, a mobile sink receives data from the source constantly. It checks its location every second and if it detects that it moves out of the original grid, it chooses the head of the new grid as a new agent. An irregular-shape query is used to monitor and collect the data of the diffusing event, that is, the event data from another neighboring grid that may diffuse into the grid in question. A cache mechanism is used to resolve the identical queries issued from mobile sinks. The protocol can reduce the overall energy consumption of the network.

7.3.2.6 Structure-Free Data Aggregation. Data aggregation can be performed based on an aggregation structure or aggregation tree in a hierarchical manner [27]. However, maintaining a hierarchical structure introduces additional cost, especially in a dynamic environment with mobile sensors. This is because in a dynamic environment, mobile sensors may frequently go out of the transmission range and a new structure needs to be built more often, leading to high maintenance overhead. This problem may also arise if a static sensor in the hierarchical structure runs out of energy and disconnects the topology. At the other extreme is opportunistic (OP) aggregation where data packets are aggregated only if they happen to meet at a sensor at the same time, which leads to inefficiency in the aggregation process. In the OP approach, there is no prespecified set of aggregation nodes for combining the data, which leads to last minute routing decisions to be taken and thus becomes time consuming. To alleviate both problems, a structure-free data aggregation approach is proposed in [28]. To increase aggregation efficiency, a MAC protocol called data-aware anycast (DAA) is also proposed because packets need to be aggregated early on their

way to the BS. It is observed that if some nodes wait for other nodes to send data it can lead to efficient aggregation [27]. Therefore, the impact of randomized waiting (RW) on improving data aggregation is also studied.

Spatial convergence and temporal convergence during transmission are two essential conditions for data aggregation. For aggregation, data packets need to be transmitted to a node at the same time. The DAA protocol is proposed to improve spatial convergence while the RW technique is proposed to improve temporal convergence. Without global knowledge of the network topology, it is not possible to construct a structure for data aggregation. Therefore, in the DAA protocol, an independent set among the sources (i.e., nodes sensing data) is created. Nodes are assumed to be time synchronized, and therefore aggregate data at the same time. Nodes in the independent set act as aggregation points and are created on the fly when data is being transmitted to the BS. The data path may follow separate routes that are determined by a specific routing algorithm. This reduces the extra overhead for the creation of a structure. The DAA protocol is based on anycasting (if the source address of a packet is an anycast address, it implies that it does not have a specific destination) at the MAC layer to resolve the next hop for each transmission. Anycasting uses RTS packets to obtain CTS responses from the neighbors before data transmission begins. There is an aggregation ID (AID) associated with each packet, which is used as the metric for aggregation. The RTS contains the AID of the transmitted packet and any neighbor that has a packet with the same AID (it is the timestamp in Ref. [28]) can respond with a CTS. Therefore, two packets that are generated at the same time can be aggregated. Since there could be multiple receivers capable of aggregating packets, the receivers delay the CTS transmissions at random to prevent CTS collisions.

Temporal convergence requires data packets to be sent to the same node at the same time. The order of transmissions can be determined by a number of factors, for example, interference from other data flows and interference from the same flow. Randomized waiting introduces artificial delays at the sources of each packet and thereby increases temporal convergence. Each source delays its transmission by an interval between 0 and τ, where τ is the maximum delay. The optimum value of τ depends on the size of the event and the time to transmit a packet. However, sensors are unable to know the value of τ beforehand as they are unaware of the event size initially. Since DAA is performed at every hop of the aggregation process, early aggregation without delay is possible to be achieved. Therefore, DAA is not very sensitive to the length of the delay [28]. In [29], the aggregation tree (AT) approach is compared with the DAA + RW approach [28]. It is observed from the comparison results that by combining RW and DAA the performance gain for delay is longer than 1.6 s, which is the transmission time for ~40 packets. This result is insignificant for events with a diameter of 400 m. The AT approach, a structure-based approach with diverse maximum delay, does not perform better than the DAA + RW approach when the maximum delay is <2.4 s, which means that a structure-based approach is sensitive to the waiting time. This is due to the fact that the parent nodes in an aggregation tree need to wait for

all their children before they can send the final aggregated packet to the root. In Ref. [28], experiments are also conducted using a Kansei testbed [29] with Mica motes. The results show that the DAA + RW approach outperforms the OP approach in all experiments. This means that DAA + RW can efficiently aggregate packets and decrease the number of data packets transmitted in the network without need for any preconstructed structure.

7.4 SUMMARY AND FUTURE DIRECTIONS

The real challenge in query processing is to support multiquery optimization. At a given time, several long-running queries from multiple users may run over a sensor network. In this case, an important concern is how to support resource sharing among multiple queries to balance and minimize the overall resource usage in the network. Another concern is how to deploy the sensors in the network in order to minimize the overall resource usage. In addition, it is also important to incorporate some fault tolerance mechanisms to prevent queries from losing data if a particular node fails. A data model that represents various kinds of attributes sensed by a sensor needs to be developed.

With the data sensed in a region increasing, data aggregation becomes important because of the limitation of battery capacity in a sensor. Meanwhile, it is necessary to allow controlled data aggregation so that an optimal balance is maintained between the time taken for data collection and the information actually sent to the BS. In any sensor network application, users first specify to the BS the data they want to collect through simple and declarative queries. These queries are the high-level statements of logical interests over the entire network, which are delivered top-down from the BS to source sensors, while data are aggregated bottom-up from the source sensors to the BS.

So far, most of the research on query processing and data aggregation has been focused on homogeneous sensor networks, in which all sensor nodes are equally powerful. Future networks should be designed to support several tiers of nodes with different performance characteristics. Because of the dynamic nature of sensor data, query optimization should also be performed in a decentralized way. In other words, it is necessary to develop a distributed system of heterogonous sensors with one or more sensing capabilities of physical parameters.

A WSN may consist of thousands of sensors, which presents a challenge for reducing the redundancy in the data sensed by the sensors in data aggregation. Despite many attempts by the industry and the research community, there are still some open issues in designing efficient query processing and data aggregation schemes.

1. *User Mobility.* In a mobile environment, a user can be mobile. In that case, the query processing scheduling at the BS requires some particular considerations [30]. If a mobile user leaves its current position when the answer to a query is ready to be submitted to it, the query answer needs

to be rerouted to the new location. In this case, the user will either reject the answer from the BS or ask the BS to process the query again, which will eventually affect the network performance. Thus, there is a need for optimal scheduling algorithms for mobile queries taking into account the constraints caused by the user mobility.

2. *Data Security.* In many applications, sensors are deployed in open environments and are vulnerable to physical attacks, which may compromise the sensors' cryptographic keys used for secured data exchange. On the other hand, allowing malicious sensors to participate in aggregation incorporates faulty readings and thus affects overall aggregation accuracy. Therefore, ensuring security, that is, authentication and confidentiality of data and sensors, is a key task for ensuring efficient and secure aggregation of information. The two main practical issues involved in implementing data encryption at the sensors are the size of the encrypted messages and the execution time for encryption at the sensors. It is well-known that encryption algorithms like RSA are computation intensive and take an exponential amount of time on large messages. The trade-off between security and computation complexity should be considered when implementing security in data aggregation. Another key aspect of security in WSNs is the establishment of secret keys between the sensors and the BS, with cautious key exchange so that no intruder can know the secret key when it is being shared.
3. *Node Heterogeneity.* Energy saving is one of the crucial objectives of a WSN. One way to achieve this objective is to introduce some degree of node heterogeneity in the network. A small number of high-end nodes can be leveraged to benefit a large number of inexpensive nodes. Heterogeneity can be achieved by equipping some nodes with more powerful battery, introducing controlled mobility, increasing link bandwidth in high-end nodes, and so on.

REFERENCES

[1] C. M. Cordeiro and D. P. Agrawal, *Ad Hoc and Sensor Networks—Theory and Applications*, World Scientific Publishing, ISBN 981-256-681-3, Singapore, 2006.

[2] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks", *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, Oct. 2002, pp. 660–670.

[3] D. P. Agrawal and Q-A. Zeng, "*Introduction to Wireless and Mobile Systems*" (2nd ed), Brooks/Cole Publishing, ISBN 0–534–49303–3, Apr. 2005.

[4] B. Krishnamachari, D. Estrin, and S. Wicker, "The impact of data aggregation in wireless sensor networks", in *Proceedings of 2002 International Workshop on Distributed Event-Based Systems (DEBS'02)*, Vienna, Austria, July *2002*, pp. 575–578.

[5] Y. J. Gehrke and S. Madden, "Query processing in sensor networks", *IEEE Pervasive Computing*, vol. 3, no. 1, Jan.–Mar. 2004, pp. 46–55.

[6] P. Bonnet, J. E. Gehrke, and P. Seshadri, "Towards sensor database systems", in *Proceedings of the 2nd International Conference on Mobile Data Management*. Hong Kong, Jan. 2001, pp. 3–14.

[7] M. Lewis, D. Cameron, S. Xie, and B. Arpinar, "ES3N: A semantic approach to data management in sensor networks," in *Proceedings of Semantic Sensor Networks Workshop*, Athens, GA, US, Nov. 2006.

[8] TinyOS, available at www.tinyos.net

[9] TinyDB, available at telegraph.cs.berkeley.edu/tinydb

[10] A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein, and W. Hong, "Model-driven data acquisition in sensor networks", in *Proceedings of Very Large Data Bases (VLDB'04)*, Toronto, Canada, Aug. 2004, pp. 588–599.

[11] A. Coman, M. A. Nascimento, and J. Sander, "A framework for spatio-temporal query processing over wireless sensor networks", in *Proceedings of the 1st international workshop on Data management for Sensor Networks(DMSN'04)*, Toronto, Canada, Aug. 30, 2004, pp. 104–110.

[12] Y. Kotidis, "Snapshot queries: Towards data-centric sensor networks", in *Proceedings of the 21st International Conference on Data Engineering (ICDE'05)*, Tokyo, Japan, Apr. 2005, pp. 131–142.

[13] C. Intanagonwiwat, R. Govindan, D. Estrin, and J. Heidemann, "Directed diffusion for sensor networking", *IEEE/ACM Transactions on Networking (TON)*, vol. 11, no. 1, Feb. 2003, pp. 2–16.

[14] J. Lian, L. Chen, K. Naik, Y. Liu, and G. B. Agnew, "Gradient boundary detection for time series snapshot construction in sensor networks", *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, vol. 18, no. 10, Oct. 2007, pp. 1462–1475.

[15] Smart Sensor, available at ieee1451.nist.gov/1451-2%20TERMS.pdf

[16] A. Deshpande, C. Guestrin, W. Hong, and S. Madden, "Exploiting correlated attributes in acquisitional query processing", in *Proceedings of the 21st International Conference on Data Engineering (ICDE'05)*, Tokyo, Japan, Apr. 2005, pp. 143–154.

[17] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks", *IEEE Communications Magazine*, vol. 40, no. 8, Aug. 2002, pp. 102–114.

[18] A. V. Mamishev. J. H. Bau, B.C. Lesieutre, and M. Zahn, "Evaluation of diffusion-driven material property profiles using three-wavelength interdigital sensor", *IEEE Transactions on Dielectrics and Electrical Insulation*, vol. 8, no. 5, Oct. 2001, pp. 785–798.

[19] T.-H. Lin and P. Huang, "Sensor data aggregation for resource inventory applications", in *Proceedings of 2005 IEEE Wireless Communications and Networking Conference (WCNC'05)*, vol. 4, New Orleans, LA, Mar. 2005, pp 2369–2374.

[20] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TAG: A tiny aggregation service for ad-hoc sensor networks", in *Proceedings of the 5th Symposium on Operating Systems Design and Implementation*, Boston, MA, Dec. 2002, pp. 131–146.

[21] E. Catterall, K. Van Laerhoven, and M. Strohbach, "Self-organization in ad hoc sensor networks: An empirical study", in *Proceedings of the 8th International Conference on the Simulation and Synthesis of Living Systems*, Sydney, Australia, Dec. 2002, pp. 260–263.

[22] A. Kulakov, D. Davcev, and G. Trajkovski, "Application of wavelet neural-networks in wireless sensor networks", in *Proceedings of the 6th International Conference on*

Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and the 1st ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05), Towson, MD, May 2005, pp. 262–267.

[23] G. A. Carpenter and S. Grossberg, "Integrating symbolic and neural processing in a self-organizing architecture for pattern recognition and prediction", in V. Honavar and L. Uhr (Eds.), *Artificial intelligence and neural networks: Steps towards principled prediction*. Academic Press, San Diego, CA, 1994, pp. 387–421.

[24] H. Fei, C. May, and C. Xiaojun, "Data aggregation in distributed sensor networks: Towards an adaptive timing control", in *Proceedings of the 3rd International Conference on Information Technology: New Generations (ITNG'06)*, Las Vegas, NV, Apr. 2006, pp. 256–261.

[25] J. Zhu, S. Papavassiliou, and J. Yang, "Adaptive localized QoS-constrained data aggregation and processing in distributed sensor networks", *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 9, Sept. 2006, pp. 923–933.

[26] T-S Chen, Y.-S. Chang, H.-W. Tsai, and C.-P. Chu, "Data aggregation for range query in wireless sensor networks", in *Proceedings of 2007 IEEE Wireless Communications and Networking Conference (WCNC'07)*, Hong Kong, China, Mar. 2007, pp. 4127–4132.

[27] J. Wong, R. Jafari, and P. Potniak, "Gateway placement for latency and energy efficient data aggregation", in *Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks*, Nov. 2004, pp. 490–497.

[28] K.-W. Fan, S. Liu, and P. Sinha, "Structure-free data aggregation in sensor networks", *IEEE Transaction on Mobile Computing*, vol. 6, no. 8, Aug. 2007, pp. 929–942.

[29] E. Ertin, A. Arora, R. Ramnath, M. Nesterenko, V. Nik, S. Bapat, V. Kulathumani, M. Sridharan, H. Zhang, and H. Cao, "Kansei: A testbed for sensing at scale", in *Proceedings of the 4th Symposium on Information Processing in Sensor Networks (IPSN/SPOTS Track)*, Nashville, TN, Apr. 2006, pp. 399–406.

[30] N. Marsit, A. Hameurlain, Z. Mammeri, and F. Morvan, "Query processing in mobile environments: A survey and open problems", in *Proceedings of the 1st International Conference on Distributed Frameworks for Multimedia Applications (DFMA'05)*, Besancon, France, Feb. 2005, pp. 150–157.

8

NODE LOCALIZATION

Nayef A. Alsindi and Kaveh Pahlavan

Worcester Polytechnic Institute, USA

8.1 INTRODUCTION

The development of micro-electro-mechanical systems (MEMS) technology, as well as the advancement in digital electronics and wireless communications, has made it possible to design small size, low-cost energy-efficient sensor nodes that could be deployed in different environments for a variety of applications [1]. Node localization is an enabling technology for wireless sensor networks (WSNs) because sensor nodes deployed in an area of interest usually need position information for routing and application-specific tasks, for example, temperature and pressure monitoring [2]. In many applications, a WSN is deployed to help improve localization accuracy in environments where the channel condition poses a challenge on range estimation [3]. In such environments, cooperative localization provides a potential for many applications in the commercial, public safety, and military sectors [3,4]. In commercial applications, there is a need for localizing and tracking inventory items in warehouses, materials and equipment in manufacturing floors, elderly in nursing homes, medical equipment in hospitals, and objects in residential homes. In public safety and military applications, however, indoor localization systems are needed to track inmates in prisons and navigate

Wireless Sensor Networks: A Networking Perspective, Edited by Jun Zheng and Abbas Jamalipour

policemen, fire fighters, and soldiers to complete their missions inside buildings [4]. Node localization plays an important role in all these WSN applications.

This chapter focuses on node localization in WSNs. We will introduce the fundamental concepts related to node localization, discuss the major challenges for node localization in WSNs, and present the major node localization techniques proposed for WSNs. Section 8.2 introduces related background and challenges of node localization technologies. Section 8.3 provides an overview of the most popular ranging techniques and discusses their performance. Section 8.4 introduces basic wireless localization algorithms using the ranging techniques and describes their importance to sensor networks. Section 8.5 introduces the most popular cooperative localization algorithms in WSNs. Specifically, we will provide an overview of centralized and distributed algorithms and highlight their strengths and weaknesses. In Section 8.6, we will conclude with a summary of the chapter and a brief discussion of future research directions.

8.2 CONCEPTS AND CHALLENGES OF NODE LOCALIZATION TECHNOLOGIES

In this section, we first introduce the evolution of localization technologies and the basics of localization in traditional network settings, and then introduce the main approaches to cooperative localization in WSNs and discuss the major factors affecting their performance.

8.2.1 Evolution of Localization Technologies

The problem of locating mobile radios originated with military operations during World War II, where it was critical to locate soldiers in emergency situations. About 20 years later, during the Vietnam conflict, the US Department of Defense launched a series of global positioning system (GPS) satellites to support military operations in combat areas. In 1990, the signals from GPS satellites were made accessible to the private sector for commercial applications, for example, fleet management, navigation, and emergency assistance. Today, GPS technology is widely available in the civilian market for personal navigation applications. Despite its success, however, the accuracy of GPS positioning is significantly impaired in urban and indoor areas, where received signals can suffer from blockage and multipath effects.

In 1996, the Federal Communications Commission (FCC) introduced regulations requiring wireless service providers to be able to locate mobile callers in emergency situations with specified accuracy—100-m accuracy 67% of the time. Such emergency service is called E-911 in the US and E-112 in many other countries. In a manner similar to the release of the ISM bands and subsequent emergence of the wireless local area network (WLAN) industry, the FCC mandate for E-911 services quickly gave rise to the development of the wireless geolocation industry. In time, technologies have been developed to implement the E-911

mandate [5,6] including GPS assisted techniques, and a variety of time of arrival (TOA), angle of arrival (AOA), and received signal strength (RSS) techniques. A variety of TOA, time differential (TDOA) or extension of time differential (EOTD) techniques require special location-measurement hardware integrated in the base stations and in some cases accurate synchronization between the mobile terminals and base stations (for cellular applications). In contrast to those approaches, RSS systems provide a lower-cost solution that can avoid additional hardware installation but does require incorporating training functions into the system.

In the late 1990s, at about the same time that E-911 technologies were emerging, another initiative for accurate indoor geolocation began independently. It was motivated by a variety of envisioned applications for indoor location sensing in commercial, public safety, and military settings [4,8,9]. In commercial applications for residences and nursing homes, there is an increasing need for indoor location-sensing systems to track people with special needs, for example, the elderly, and children who are away from visual supervision. In public safety and military applications, indoor location sensing systems are needed to track inmates in prisons and to guide policemen, fire-fighters, and soldiers in accomplishing their missions inside buildings. More recently, location sensing has found its applications in location-based handoffs in wireless networks [10], location-based ad hoc network routing [11,12], and location-based authentication and security [13]. These and other applications have stimulated interests in modeling the propagation environment to assess the accuracy of different sensing techniques [14,15], as well as in developing novel technologies to implement the systems [16–18]. The implementation of the first generation of indoor positioning products using a variety of technologies has been reported in Refs. [19–21].

Finally, the natural evolution of these ranging and localization technologies makes their integration into WSN applications possible. Understanding the fundamental concepts and challenges of these technologies in traditional localization is a necessary bridge to WSN localization.

8.2.2 Localization Systems

In general, a localization system incorporates range measurements to determine the location estimate. Figure 8.1 illustrates a block diagram of the main components in a traditional localization system.

Essentially, the process for obtaining a location estimate involves different levels of complexities. At the physical layer, the mobile terminal (MT) or the sensor node transmits and receives a waveform. From this radio frequency (RF) waveform, it is possible to extract the relevant range measurements. In RSS systems, for example, the total signal energy that a node/MT receives from an anchor/reference point (RP) can be used to estimate the distance. For a given received power, it is possible to estimate the corresponding distance with some certainty. The RSS technique is usually simple to implement, but suffers from inaccuracies, especially in multipath-rich environments. On the other hand, for

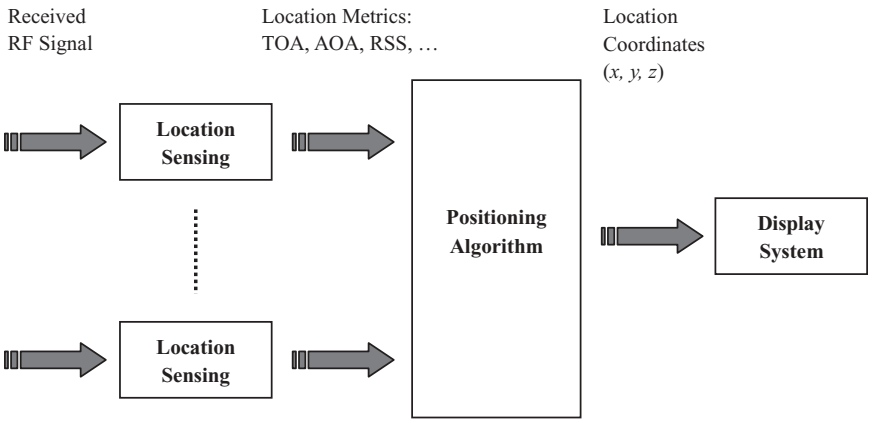


Fig. 8.1 Localization block diagram.

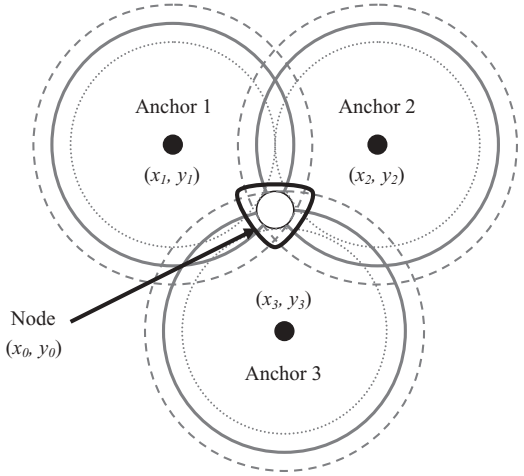


Fig. 8.2 Localization with three anchors.

TOA based systems, the distance is estimated by sending an RF signal and recording the time it takes to receive it. This approach is more accurate because the arrival time corresponds to the direct path distance.

Once 3(4) range measurements are obtained from different anchors/RPs, the node/MT passes this information to a positioning algorithm, where the two- or three-dimensional (2D or 3D) position is then estimated. The range measurements essentially constrain the possible location of the MT. The area of uncertainty of a location estimate decreases as the accuracy of range measurements improves. Figure 8.2 shows an example of 2D localization, where a node/MT has

three range measurements to different anchors/RPs. The positioning error, as will be described later in more detail, is affected by the accuracy of the range measurements, the number of anchors/RPs, and their relative geometry to the sensor node/MT. Finally, the estimate of the location is displayed to the user with information regarding its quality or accuracy.

The WSN localization is a general case of the traditional localization, but it is fundamentally dependant on the building blocks in Fig. 8.1. As a result, we will dedicate the first part of this chapter to ranging and localization techniques in traditional network settings and the second part to localization in WSNs. Understanding of ranging techniques and localization algorithms is essential in building a fundamental basis for WSN cooperative localization. First, we describe the two most popular ranging techniques that are used traditionally in wireless networks, which have a great potential for WSNs. Specifically, we show that the ranging accuracy and localization performance are directly related to the complexity of the wireless channel. Then, we discuss popular localization algorithms commonly implemented in systems, for example, GPS and cellular geolocation. Finally, we relate these concepts to cooperative localization in WSNs, where we describe some of the emerging centralized and distributed solutions to the problem.

8.2.3 Challenges of Node Localization in Wireless Sensor Networks

In general, there are two main approaches to node localization in a WSN. The first is centralized, where localization information of each node in the network is determined centrally through a computer usually at one edge of the network. The range estimates between all node pairs in the network are forwarded to the processing unit, where a complex centralized algorithm estimates the location of each node in the network. The advantage of this approach is that all ranging information between node pairs is available to the central processor. As a result, the processor has a top-level view of the connectivity of the network. The amount of information allows the centralized algorithm to generate more accurate localization results. The drawbacks, on the other hand, include traffic congestion and computational complexities, especially for larger sensor networks. In the former, the possibility of congestion that occurs close to the central processing unit due to information going back and forth can reduce the effectiveness of this approach. Similarly, the latter drawback imposes constraints on the computation time needed to handle estimating the node positions in a large WSN. The second approach used in WSN localization is distributed in nature. The process is usually iterative, where sensor nodes attempt to localize themselves first and then aid the remaining nodes in the localization process. Distributed positioning algorithms provide the best alternatives so far in their approach. The algorithms are self-organizing and energy efficient.

The major challenges facing WSN localization can be categorized into network and channel parameters. When considering network parameters, localization is usually constrained by the size (i.e., the number of nodes and *anchors*),

the topology, and the connectivity of the network. Anchors or *beacons* are sensor nodes that are aware of their locations (usually through GPS or preprogrammed during setup) and they are necessary for WSN applications that require localization with respect to an absolute global frame of reference, for example, GPS. Network connectivity is determined by node density, which is usually defined as the number of nodes in a meter square (nodes/m^2). A network with a high node density exhibits improved localization performance compared to a sparse network. Further, in sparse WSNs, there is a high probability of ill-connected or isolated nodes and in such cases localization accuracy can be degraded substantially. Therefore, it is always favorable to increase the node density (higher connectivity information means a lower probability of ill-connected networks) to improve the accuracy of localization. However, with increased sensor nodes, the error propagates and accumulates from one hop to the next, which can be a serious problem in WSN localization algorithms. Error propagation in WSN localization is the accumulation of errors in estimated sensor node positions in each iterative step. When a node *transforms* into an anchor, the error in the range estimates used in the localization process impacts its position estimate. When other nodes in the network use this newly transformed anchor, the position error will *propagate* to the new node. Therefore, in several iterative steps, error propagation can substantially degrade the localization performance. Finally, the topology and geometric relation between nodes will further add limitations to the localization performance.

The second and most limiting factor affecting WSN localization is the wireless RF channel. Effective cooperative localization hinges on the RF ranging technology and its behavior in the deployed environment. The most popular ranging techniques are TOA, RSS, and AOA. The TOA techniques have been widely accepted as the most accurate, while RSS the most practical, but with lower accuracy and precision. The behavior of these techniques varies significantly in different deployment environments. For example, deploying hundreds of nodes in outdoor environments faces different challenges as opposed to trying to locate sensors inside a building. The WSNs in indoor areas, particularly, face severe multipath fading and harsh radio propagation environments, which causes large ranging estimation errors that impact localization performance directly. To develop practical and accurate cooperative localization algorithms, the behavior of the wireless channel must be investigated and incorporated. Specifically, the localization algorithms used to determine the position must be able to assess the quality of the ranging estimates and integrate that information into the localization process.

8.3 RANGING TECHNIQUES FOR WIRELESS SENSOR NETWORKS

The RF location sensors operating in different environments can measure the RSS, AOA, phase of arrival (POA), TOA, and signature of the delay-power profile as location metrics to estimate the ranging distance [4,7]. The deployment

environment (i.e., wireless RF channel) will constrain the accuracy and the performance of each technique. In outdoor open areas, these ranging techniques perform very well. However, as the wireless medium becomes more complex, for example, dense urban or indoor environments, the channel suffers from severe multipath propagation and heavy shadow fading conditions. This finding in turn impacts the accuracy and performance in estimating the range between a pair of nodes. For this reason, this chapter will focus its ranging and localization discussion on indoor environments. This is important because many of the WSN applications are envisioned for deployment in rough terrain and cluttered environments and understanding of the impact of the channel on the performance of ranging and localization is important. In addition, range measurements using POA and AOA in indoor and urban areas are unreliable. Therefore, we will focus our discussion on two practical techniques, TOA and RSS. These two ranging techniques, which have been used traditionally in wireless networks, have a great potential for use in WSN localization.

The TOA based ranging is suitable for accurate indoor localization because it only needs a few references and no prior training. By using this technique, however, the hardware is complex and the accuracy is sensitive to the multipath condition and the system bandwidth. This technique has been implemented in GPS, PinPoint, WearNet, IEEE 802.15.3, and IEEE 802.15.4 systems. The RSS based ranging, on the other hand, is simple to implement and is insensitive to the multipath condition and the bandwidth of the system. In addition, it does not need any synchronization and can work with any existing wireless system that can measure the RSS. For accurate ranging, however, a high density of anchors or reference points is needed and extensive training and computationally expensive algorithms are required. The RSS ranging has been used for WiFi positioning in systems, for example, Ekahau, Newbury Networks, PanGo, and Skyhook.

This section first introduces TOA based ranging and the limitations imposed by the wireless channel. Then it will be compared with the RSS counterpart focusing on the performance as a function of the channel behavior. What is introduced here is important to the understanding of the underlying issues in distance estimation, which is an important fundamental building block in WSN localization.

8.3.1 TOA Based Ranging

In TOA based ranging, a sensor node measures the distance to another node by estimating the signal propagation delay in free space, where radio signals travel at the constant speed of light. Figure 8.3 shows an example of TOA based ranging between two sensors. The performance of TOA based ranging depends on the availability of the direct path (DP) signal [4,14]. In its presence, for example, short distance line-of-sight (LOS) conditions, accurate estimates are feasible [14]. The challenge, however, is ranging in non-LOS (NLOS) conditions, which can be characterized as site-specific and dense multipath environments [14,22]. These environments introduce several challenges. The first corrupts the TOA estimates

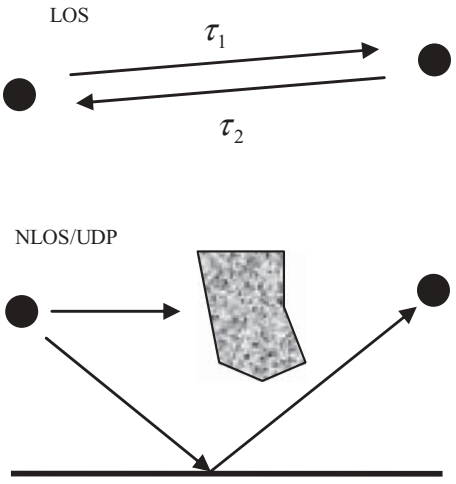


Fig. 8.3 The TOA ranging between sensors.

due to the multipath components (MPCs), which are delayed and attenuated replicas of the original signal, arriving and combining at the receiver shifting the estimate. The second is the propagation delay caused by the signal traveling through obstacles, which adds a positive bias to the TOA estimates. The third is the absence of the DP due to blockage, also known as undetected direct path (UDP) [14]. The bias imposed by this type of error is usually much larger than the first two and has a significant probability of occurrence due to cabinets, elevator shafts, or doors that are usually cluttering the indoor environment.

In order to analyze the behavior of the TOA based ranging, it is best to resort to a popular model used to describe the wireless channel. In a typical indoor environment, the transmitted signal will be scattered and the receiver node will receive replicas of the original signal with different amplitudes, phases, and delays. At the receiver, the signals from all these paths combine and this phenomenon is known as multipath. In order to understand the impact of the channel on the TOA accuracy, we resort to a model typically used to characterize multipath arrivals. For multipath channels, the impulse response $h(\tau)$ characterizes the arrival paths, their respective amplitudes, and delays. Mathematically, it can be represented as a summation of all the arriving multipath components or

$$h(\tau) = \sum_{k=1}^{L_p} \beta_k e^{j\phi_k}\, \delta(\tau - \tau_k), \tag{8.1}$$

where L_p is the number of MPCs, and β_k, ϕ_k, and τ_k are amplitude, phase, and propagation delay of the kth path, respectively [7,23]. Let β_1^{DP} and τ_1^{DP} denote the DP amplitude and propagation delay, respectively. The distance between the sensor node and the RP or anchor is $d^{\mathrm{DP}} = v \times \tau_1^{\mathrm{DP}}$, where v is the speed of signal

propagation. In the absence of the DP, ranging can be achieved using the amplitude and propagation delay of the non-direct path (NDP) component given by β_1^{NDP} and τ_1^{NDP}, respectively; resulting in a longer distance $d^{\text{NDP}} = v \times \tau_1^{\text{NDP}}$, where $d^{\text{NDP}} > d^{\text{DP}}$. For the receiver to identify the DP, the ratio of the strongest MPC to that of the DP given by

$$\kappa_1 = \frac{\max\left(|\beta_i|_{i=1}^{L_p}\right)}{|\beta_{\text{DP}}|} \tag{8.2}$$

must be less than the receiver dynamic range κ and the power of the DP must be greater than the receiver sensitivity φ. These constraints are given by

$$\kappa_1 \leq \kappa, \tag{8.3a}$$

$$P_{\text{DP}} > \varphi, \tag{8.3b}$$

where $P_{\text{DP}} = 20\log_{10}\left(|\beta_1^{\text{DP}}|\right)$.

In general, ranging and localization accuracy is constrained by the ranging error, which is defined as the difference between the estimated and the actual distance; that is,

$$\varepsilon = \hat{d} - d. \tag{8.4}$$

In an indoor environment, the node/MT will experience a varying error behavior depending on the availability of the DP and in the case of its absence on the characteristics of the DP blockage. It is possible to categorize the error based on the following ranging states [24]. In the presence of the DP, both (8.3a) and (8.3b) are met and the distance estimate is very accurate, yielding

$$\hat{d}_{\text{DP}} = d_{\text{DP}} + \varepsilon_{\text{DP}} + z, \tag{8.5a}$$

$$\varepsilon_{\text{DP}} = \begin{cases} b_m & \text{LOS}, \\ b_{\text{pd}} + b_m & \text{NLOS}, \end{cases} \tag{8.5b}$$

where b_m is the random bias induced by the multipath, b_{pd} is the bias corresponding to the propagation delay caused by NLOS conditions, and z is a zero-mean additive measurement noise. It has been shown that b_m is indeed a function of the bandwidth and the signal to noise ratio (SNR) [14], while b_{pd} is dependant on the medium of the obstacles. When the node experiences sudden blockage of the DP, Eq. (8.3a) is not met and the DP is shadowed by some obstacle, burying its power under the dynamic range of the receiver. In this situation, the ranging estimate experiences a larger error compared to Eq. (8.5a). Emphasizing that ranging is achieved through the NDP component, the estimate is then given by

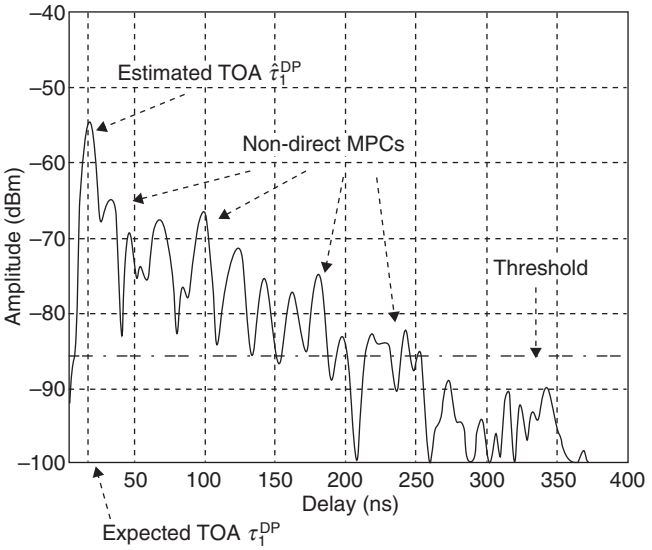


Fig. 8.4 The TOA estimation in the presence of DP at 200-MHz bandwidth.

$$\hat{d}_{\mathrm{NDP}} = d_{\mathrm{DP}} + \varepsilon_{\mathrm{NDP}} + z, \tag{8.6a}$$

$$\varepsilon_{\mathrm{NDP}} = b_m + b_{\mathrm{pd}} + b_B, \tag{8.6b}$$

where b_B is a deterministic additive bias representing the nature of the blockage. Unlike the multipath biases, but similar to the biases induced by the propagation delay, the dependence of b_B on the system bandwidth and SNR has its own limitations as reported in Ref. [14]. Formally, these ranging states can be defined as

$$\zeta_1 = \left\{\hat{d} = \hat{d}_{\mathrm{DP}}\right\}, \tag{8.7a}$$

$$\zeta_2 = \left\{\hat{d} = \hat{d}_{\mathrm{NDP}}\right\}. \tag{8.7b}$$

Figures 8.4 and 8.5 provide sample channel profiles of these two ranging situations [24].

The performance of TOA based ranging can be determined by the Cramer-Rao lower bound (CRLB), which has been studied extensively for existing systems. The variance of TOA estimation σ^2_{TOA} is bounded by the CRLB [25]

$$\sigma^2_{\mathrm{TOA}} \geq \frac{1}{8\pi^2 \gamma T \omega f_0^2 \left(1 + \dfrac{\omega^2}{12 f_0^2}\right)}, \tag{8.8}$$

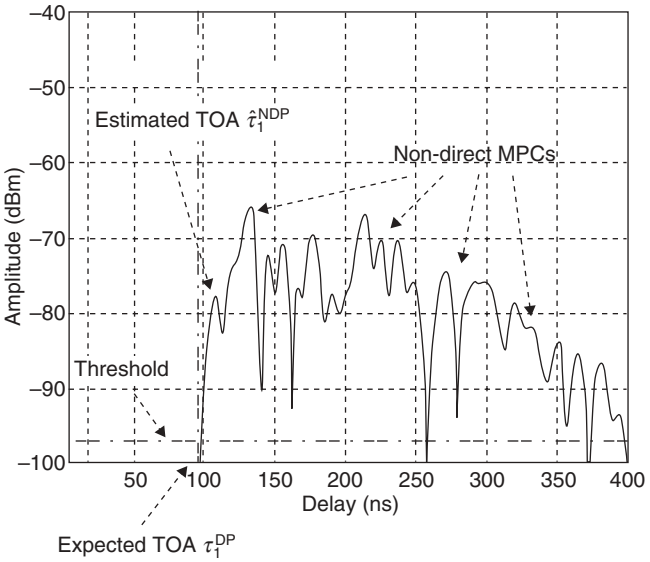


Fig. 8.5 The TOA estimation in NDP at 200-MHz bandwidth.

where T is the signal observation time, γ is the SNR, f_0 is the frequency of operation, and ω is the system bandwidth.

In practice, TOA can be obtained by measuring the arrival time of a wideband narrow pulse, which can be obtained either by using spread spectrum technology or directly.

8.3.1.1 Direct Spread Spectrum. One TOA estimation technique based on the direct spread spectrum (DSS) wideband signal has been used in GPS and other ranging systems for many years. In such a system, a signal coded by a known pseudorandom (PN) sequence is transmitted and a receiver cross-correlates the received signal with a locally generated PN sequence using a sliding correlator or a matched filter. The distance between the transmitter and the receiver is determined from the arrival time of the first correlation peak. Because of the processing gain of the correlation at the receiver, DSS ranging systems perform much better than competing systems in suppressing interference from other radio systems operating in the same frequency band. In these band-limited systems, super-resolution techniques for TOA estimation have been applied successfully. Results have shown that these high-resolution algorithms can provide improved accuracy [25].

8.3.1.2 Ultra-Wideband Ranging. A promising alternative to DSS systems is ultra-wideband (UWB) ranging [26]. According to Eq. (8.8), it is clear that in multipath propagation environments, the performance of TOA estimation is inversely related to the system bandwidth. Increasing the system bandwidth

(i.e., narrower time-domain pulse) results in higher time resolution and thus better ranging accuracy. As a result, these systems have attracted considerable attention in recent years [16,22,26]. For UWB applications, the FCC regulation allocated an unlicensed flat frequency band 3.1–10.6 GHz for which there are two proposals: direct sequence (DS)–UWB and multiband orthogonal frequency division multiplexing (MB–OFDM). The former is pulse based, which utilizes large bandwidths, for example, 3 GHz, while the latter occupies a bandwidth of 528 MHz. The accuracy of these systems can be evaluated by examining their behaviors in the multipath channel. Sample measurements in indoor office environments are provided in Fig. 8.6a for 500-MHz systems, resembling the MB–OFDM channels and Fig. 8.6b for 3-GHz bandwidth, resembling the wider channel of the DS–UWB. The expected TOA between the transmitter and the receiver is 40.5 ns and the estimated arrival with 500-MHz and 3-GHz bands are 45.5 and 40.7 ns, respectively. The 5- and 0.2-ns errors in TOA estimation results in 1.67-m and 7-cm errors, respectively, clearly illustrating the impact of a higher system bandwidth on accuracy.

One important observation from these measurement results is that higher bandwidths improve time-domain resolution, which resolves the pulse into respective components, resulting in improved accuracy. The trade-off, however, is that higher resolution implies lower energy per MPC, which means a higher probability of DP blockage. This means that the ranging coverage of 500-MHz systems is larger than that of the 3-GHz counterpart. Although UWB can reduce multipath significantly, combating the excess propagation delay and UDP becomes challenging because the amount of delay and the type of blocking material are not known in advance and cannot be mitigated through large bandwidths alone. Understanding of the error behavior in light of these major error contributors is necessary to enable effective UWB ranging. Specifically, WSN localization algorithms must analyze the channel statistics and attempt to identify and mitigate DP blockage [27,28].

8.3.2 RSS Based Ranging

Ranging through RSS is accomplished by sensing the received signal and measuring the total received power, which can provide a distance estimate between the target object and the location sensor. The average RSS at a certain distance is given by

$$\mathrm{RSS}_d = \sum_{i=1}^{L} \overline{\left|\beta_i^d(t)\right|^2}, \tag{8.9}$$

where β is the amplitude of the arriving paths defined in Eq. (8.1). Figure 8.7 shows a ranging example using the RSS based technique. The measurement of the average RSS is independent of the bandwidth of the measurement device.

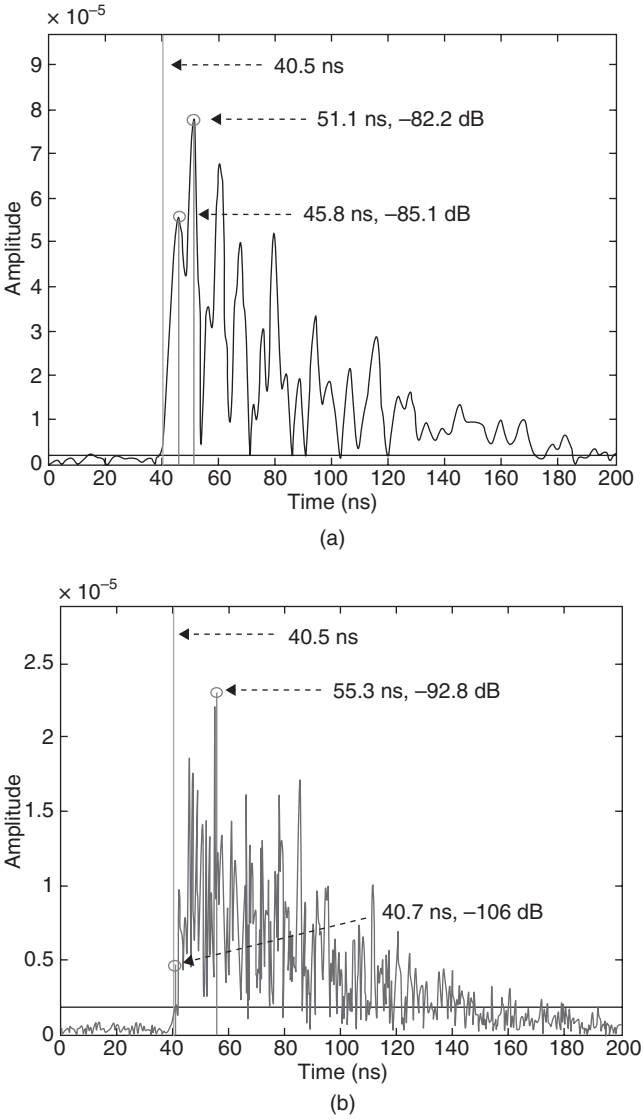


Fig. 8.6 The UWB channel profile for bandwidth (a) 500 MHz and (b) 3 GHz.

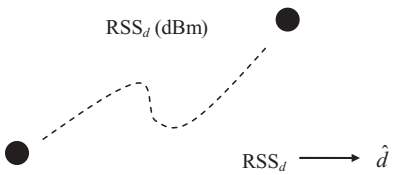


Fig. 8.7 The RSS ranging between sensors.

In wideband measurements, the effect of multipath fading is averaged over the spectrum of the signal. This is done through measuring the strength of each arriving path and using Eq. (8.9) to compute the RSS. According to the multipath fading characteristics, only one arriving pulse with a fluctuating amplitude is received. As a result, averaging the signal over a longer period can effectively eliminate multipath. In addition to the independence of the ranging error in RSS to the system bandwidth, this technique is relatively simple and reliable. Nonetheless, the relation between the measured RSS and the distance is complex and diversified. Therefore, the performance of these techniques depends on the accuracy of the model used for the estimation of the RSS.

A number of statistical models describing the behavior of the RSS to the distance between a transmitter and a receiver in indoor areas have been developed for wireless communications [7]. These models can be used for estimating the ranging distance between two nodes. The common principal behind all statistical models for calculation of the RSS in a distance d is given by

$$\mathrm{RSS}_d = 10\log_{10} P_r = 10\log_{10} P_t - 10\alpha \log_{10} d + X, \tag{8.10}$$

where P_t is the transmitted power, d is the distance between the transmitter and the receiver, α is the so-called distance-power gradient of the environment, and X is a log normal random variable representing the shadow fading component. Since the location sensor using RSS does not know the exact value of α and X, the distance calculated from these models is not as reliable as compared to the TOA counterpart. Figure 8.8 shows the relationship between RSS and distance.

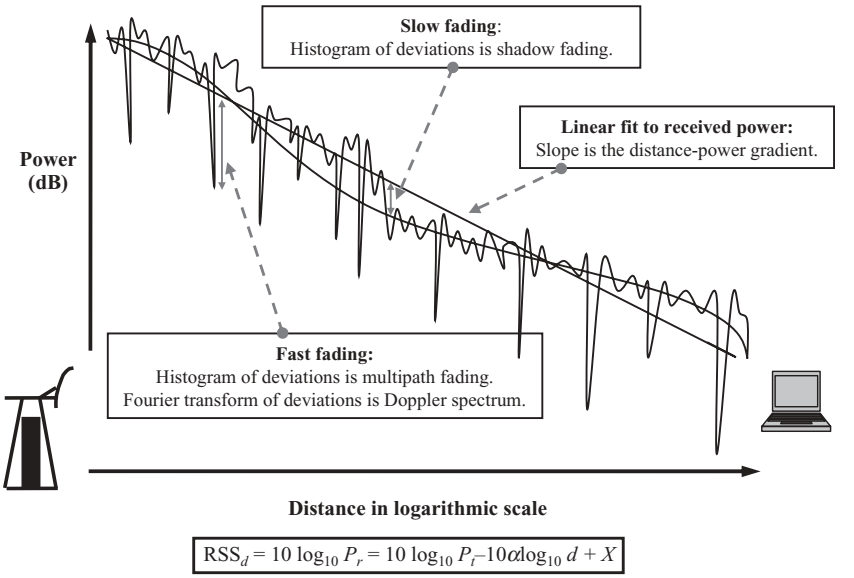


Fig. 8.8 The RSS and distance relationship in multipath environments.

TABLE 8.1 Overview of Ranging Techniques

Signal Metric	Advantages	Disadvantages
TOA	Accurate	Accuracy depending on bandwidth
	No need for exhaustive training	Complex hardware
	Good scalability	DP Blockage problems
		Complex timing requirements
RSS	Simple hardware	Less accurate
	No need for complex timing	Long training procedure
	Resilient to DP Blockage	Complex algorithms
	Less bandwidth sensitive	Does not scale easily to large areas

The performance of RSS based ranging can be evaluated similar to the TOA counterpart using well-established CRLB analysis. The analysis shows that the CRLB degrades significantly with distance and the relationship is given by [29]

$$\sigma_{\text{RSS}}^2 \geq \frac{(\ln 10)^2}{100} \frac{\sigma_{\text{sh}}^2}{\alpha_p} d, \tag{8.11}$$

where σ_{sh} is the standard deviation of the zero-mean log-normally distributed random variable X in Eq. (8.10), α_p is the pathloss exponent, and d is the distance between the two sensor nodes.

As a result of these channel limitations, RSS sensors are either used in applications where accuracy is not a prime concern or used with pattern recognition algorithms that need substantial calibration measurements. On the positive side, measuring the RSS is much simpler than the TOA counterpart. In cellular and WLAN networks, the RSS is readily available and can be calculated for power control and hand-off applications. As a result, such systems have attracted considerable attention for urban and indoor geolocation systems. In order to improve RSS reliability, one of the following methods is needed. First, certain intelligence must be incorporated into the system to identify the sensor location through previous calibration measurements. Second, complex building-specific models, for example, ray tracing, must be used in order to assess and implement ranging with higher accuracy. Finally, complex pattern recognition algorithms for location finding [7] can be incorporated with this technique to further improve the performance.

Table 8.1 provides an overview of the advantages and disadvantages of TOA versus RSS ranging techniques.

8.4 WIRELESS LOCALIZATION ALGORITHMS

This section first provides a brief background on the concept of wireless localization algorithms, and then introduces two popular geometrical triangulation

techniques that use the ranging estimates from multiple anchors to estimate the location of a node/MT. Specifically, we will introduce the least-squares (LS) and weighted least-squares (WLS) algorithms, whose importance is obvious when used for WSN localization. Finally, pattern recognition techniques, which are an alternative approach to wireless localization, will be briefly discussed.

8.4.1 Background

By using range estimates from multiple anchors, it is possible to employ simple geometrical triangulation techniques to estimate the location of a sensor. Due to estimation errors in the acquired TOA ranges, for example, the geometrical triangulation technique can only provide a region of uncertainty, instead of a single position fix for a sensor node. To obtain an estimate of the location coordinates, a variety of direct and iterative statistical positioning algorithms have been developed to solve the problem by formulating it into a set of nonlinear iterative equations. In some wireless geolocation applications, the purpose of the positioning systems is to provide a visualization of the possible mobile locations instead of an estimate of the location coordinates. In either case, the position accuracy is not constant across the area of coverage and poor geometry of relative position of the mobile terminal and RPs can lead to high geometric dilution of precision (GDOP). Further, geometric and statistical triangulation algorithms are used when both the region of uncertainty and the estimate of the location are required [7].

Localization algorithms with well-defined properties, for example, the LS and maximum-likelihood algorithm, are available for satellite-based GPS systems. In addition, there are various types of sequential filters, including formulations, which adaptively estimate some unknown parameters of the noise processes [30,31]. The GPS, in particular, has focused a great deal of attention on positioning algorithms based on TOA with considerable success. A global positioning system can provide positioning accuracy ranging from tens of meters to centimeters in real time depending on a user's resources [30]. In essence, these techniques are readily applicable to indoor location sensing systems. However, indoor location sensing involves quasi-stationary applications and a number of unreliable reference points for which the existing GPS algorithms, designed for mobile systems with a few reliable reference points, do not provide the optimum solution.

8.4.2 Geometrical Triangulation Techniques

Geometrical techniques are based on iterative algorithms that estimate the node position by formulating and solving a set of nonlinear equations. When the statistics of the ranging error, be it TOA or RSS, are not available *a priori*, the LS algorithms can provide the best solution. However, if the statistics of the ranging error are available, a WLS algorithm can be implemented, which *weighs* the range measurements with the variance of the respective error distributions.

Thus the availability of the range error information can substantially improve the accuracy of the localization process. Again, it is important to realize that the distribution of the ranging error is directly related to the RF wireless propagation channel.

8.4.2.1 Least-Squares Algorithm. Estimating a node's position in 2D(3D)s requires range information to at least 3D(4D) anchors/RPs. For simplicity, we will provide an analysis for 2D localization and an extension to higher dimensions can be easily obtained. Let $\boldsymbol{\theta} = [x,y]$ be the sensor node's x- and y-coordinates and let $\boldsymbol{\varphi}_i = [x_i^a, y_i^a]$ denote the coordinates of the ith anchor, where $i \in \{1, \ldots, M\}$. The range estimate between the ith anchor and the sensor node is then given by

$$\hat{d}_i = \|\boldsymbol{\theta} - \boldsymbol{\varphi}_i^a\| + \varepsilon_i + \tilde{z}_i = \sqrt{(x - x_i^a)^2 + (y - y_i^a)^2} + \varepsilon_i + \tilde{z}_i, \tag{8.12}$$

where ε_i is the ranging error that can be either one of the ranging conditions given in Eqs. (8.5) or (8.6), in the case the TOA technique is used, and $\tilde{z}_i$ is additive measurement noise. Note that the statistics of ε_i are not necessarily identically distributed. In indoor environments, the ranging error will experience different means and variances depending on the distances between the nodes and the blockage condition. Also, for the sake of simplicity and noting that the errors induced by the channel are substantially more significant than synchronization errors, we assume that the nodes involved in localization are synchronized. Given m noisy measurements to respective anchors, it is possible to obtain an estimate of the sensor node location $\hat{\boldsymbol{\theta}}$. Figure 8.2 shows an example of 2D localization with three noisy measurements from the respective anchors.

The problem of LS localization is essentially to obtain a solution from a set of nonlinear equations given by

$$\mathbf{F}(\boldsymbol{\theta}) = \begin{bmatrix} \sqrt{(x - x_1^a)^2 + (y - y_1^a)^2} \\ \vdots \\ \sqrt{(x - x_M^a)^2 + (y - y_M^a)^2} \end{bmatrix}, \tag{8.13}$$

where the nonlinear problem in Eq. (8.13) requires minimizing the cost function given by

$$E[\hat{\boldsymbol{\theta}}] = [\mathbf{d} - \mathbf{F}(\hat{\boldsymbol{\theta}})]^H [\mathbf{d} - \mathbf{F}(\hat{\boldsymbol{\theta}})], \tag{8.14}$$

where H is the Hermitian or the transpose conjugate of a matrix. In order to obtain an LS solution, first we linearize the set of nonlinear equations around $\boldsymbol{\theta}_0$. Linearizing $\mathbf{F}(\boldsymbol{\theta})$ can be achieved by using first-order Taylor series expansion around $\boldsymbol{\theta}_0$ and retaining the first two terms; that is,

$$\mathbf{F}(\boldsymbol{\theta}) \approx \mathbf{F}(\boldsymbol{\theta}_0) + \mathbf{J}(\boldsymbol{\theta} - \boldsymbol{\theta}_0), \tag{8.15}$$

where $\mathbf{J}$ is the Jacobian of $\mathbf{F}$ given by

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial \theta_1} & \cdots & \frac{\partial f_1}{\partial \theta_M} \\ \cdots & \cdots & \cdots \\ \frac{\partial f_N}{\partial \theta_1} & \cdots & \frac{\partial f_N}{\partial \theta_M} \end{bmatrix}_{\boldsymbol{\theta}=\boldsymbol{\theta}_0}. \tag{8.16}$$

For the three-anchor example in Fig. 8.2, the Jacobian is evaluated by computing the partial derivatives in Eq. (8.16); that is,

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \\ \frac{\partial f_3}{\partial x} & \frac{\partial f_3}{\partial y} \end{bmatrix} = \begin{bmatrix} \frac{x - x_1^a}{\sqrt{\left(x - x_1^a\right)^2 + \left(y - y_1^a\right)^2}} & \frac{y - y_1^a}{\sqrt{\left(x - x_1^a\right)^2 + \left(y - y_1^a\right)^2}} \\ \frac{x - x_2^a}{\sqrt{\left(x - x_2^a\right)^2 + \left(y - y_2^a\right)^2}} & \frac{y - y_2^a}{\sqrt{\left(x - x_2^a\right)^2 + \left(y - y_2^a\right)^2}} \\ \frac{x - x_3^a}{\sqrt{\left(x - x_3^a\right)^2 + \left(y - y_3^a\right)^2}} & \frac{y - y_3^a}{\sqrt{\left(x - x_3^a\right)^2 + \left(y - y_3^a\right)^2}} \end{bmatrix}. \tag{8.17}$$

The linearized LS solution is then given by [32]

$$\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}_0 + \left(\mathbf{J}^H \mathbf{J}\right)^{-1} \mathbf{J}^H \left[\mathbf{d} - \mathbf{F}(\boldsymbol{\theta}_0)\right]. \tag{8.18}$$

This algorithm introduces errors when the linearized function does not accurately approximate the original nonlinear function. Also, it requires an initial estimate of the unknown parameters, that is, the initial estimate of the node location coordinate. With a random initial estimate of the unknown parameters, this algorithm may converge to a local optimum, instead of a global optimum. This problem can be somewhat alleviated by performing this algorithm iteratively with each successive estimate being closer to the optimum estimate; that is,

$$\hat{\boldsymbol{\theta}}_{i+1} = \hat{\boldsymbol{\theta}}_i + \left(\mathbf{J}^H \mathbf{J}\right)^{-1} \mathbf{J}^H \left[\mathbf{d} - \mathbf{F}\left(\hat{\boldsymbol{\theta}}_i\right)\right]. \tag{8.19}$$

The iteration can be stopped when some criteria is met. For example, for a given small tolerance σ, the iterative algorithm must stop if $\left|E\left(\hat{\boldsymbol{\theta}}_{i+1}\right) - E\left(\hat{\boldsymbol{\theta}}_i\right)\right| < \sigma$. Alternatively, the algorithm can terminate after a maximum number of iterations has been performed.

8.4.2.2 Weighted Least-Squares Algorithm. In the case that the statistics of the ranging error are available, localization performance can be improved by applying a weighed least-squares (WLS) technique. The WLS algorithm solution is formed as the vector $\hat{\boldsymbol{\theta}}$ that minimizes the cost function

$$E_w\left(\hat{\boldsymbol{\theta}}\right) = \left[\mathbf{d} - \mathbf{F}\left(\hat{\boldsymbol{\theta}}\right)\right]^H \mathbf{W}\left[\mathbf{d} - \mathbf{F}\left(\hat{\boldsymbol{\theta}}\right)\right], \tag{8.20}$$

where $\mathbf{W} = \text{diag}\{w_1 \ldots w_N\}$ is a diagonal weighting matrix with positive elements. Usually we choose small weights, where errors are expected to be large, and vice versa. Minimization of E_w yields the WLS estimator given by

$$\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}_0 + (\mathbf{J}^H \mathbf{W} \mathbf{J})^{-1} \mathbf{J}^H \mathbf{W}[\mathbf{d} - \mathbf{F}(\boldsymbol{\theta}_0)], \tag{8.21}$$

where it is assumed that the inverse of the matrix $\mathbf{J}^H \mathbf{WJ}$ exists. If the distance estimation error vector has a zero mean, that is, $E\{\mathbf{e}\} = \mathbf{0}$, we can obtain the minimum variance (MV) or Markov estimator, which is the best linear unbiased estimator (BLUE) by choosing $\mathbf{W} = \mathbf{R}_e^{-1}$, where $\mathbf{R}_e$ is the correlation matrix of the distance estimation error vector [32].

8.4.2.3 Practical Performance Considerations. If the range measurements are corrupted by zero-mean normally distributed random noise, the unbiased CRLB can be achieved through the use of WLS algorithms for identically and nonidentically distributed errors, respectively. However, in the case that those measurements are biased, for example, in indoor TOA estimation, simply applying WLS techniques will not provide an optimal solution. In order to implement these algorithms in the indoor environments, the statistics of the bias must be incorporated. Obtaining the statistics of the bias in indoor environments requires extensive measurements and modeling campaigns [24]. In addition, identification of NLOS on specific range measurements must be integrated with mitigation techniques that adjust the weights in WLS to improve the localization accuracy [33].

Another factor affecting the quality of location estimation is the relative geometry of the anchors to the sensor node. Geometric dilution of precision (GDOP) is commonly used in localization applications to quantify the geometrical impact on precision. The GDOP expression has many different forms [31], but a simple expression in terms of the angles between the anchors and the sensor node is given by [34]

$$\text{GDOP}(M, \phi) = \sqrt{\frac{M}{\sum_i \sum_{jj>1} |\sin(\phi_{ij})|^2}}, \tag{8.22}$$

where M is the number of anchors involved in the localization process and ϕ is the angle between each pair of anchors. An example illustrating the impact of geometry on the precision of localization is given in Fig. 8.9. In the simulation example, the statistics of the ranging error between the node and the anchors are identical.

In Fig. 8.9a, the anchors are 120° relative to each other. While in Fig. 8.9b, they are 20° apart. The figure highlights the impact of geometry on the LS algorithm precision, where the effect of sensor node and anchor geometry can be clearly seen. The spreading of the ranging error in the 20° case results in higher uncertainty.

Fig. 8.9 Effect of geometry on sensor node position estimation: (a) good geometry and (b) bad geometry.

8.4.3 Pattern Recognition Techniques

Alternative approaches to LS and WLS techniques have mainly focused on pattern recognition techniques. For indoor and urban geolocation applications, the building floor plans and road maps are easily accessible as electronic documents. The availability of these electronic maps is a key feature that can be exploited in positioning algorithms. The basic operation of pattern recognition positioning algorithms is simple. Buildings and urban areas are unique in their signal propagation characteristics, with each location having a unique signature in terms of RSS or TOA. A pattern recognition algorithm determines the unique features or location signatures of an area of interest in a training process. The information is then used to develop a set of rules for the recognition procedure. The metrics associated with the features could be extracted from actual measurements or using models that accurately predict their empirical values. The challenge for such algorithms is to distinguish between locations with similar signatures. In addition, the computational complexity for storage and processing of the database increases substantially for large areas. To build the signature database, a terminal is carried through the service area to collect the desired location sensing metrics from all sensing elements (RPs). The service area is then divided into nonoverlapping zones or grids and the algorithm analyzes the received signal patterns and compiles a unique signature for each zone. For quasistationary applications, the simplest way for pattern recognition is using a nearest-neighbor method [35]. In this method, the Euclidean distance is calculated between the measured metrics, RSS or TOA, and all entities in the signature database. The entry in the database that has the minimum Euclidean distance to the current location is the location estimate [17,18].

8.5 WIRELESS SENSOR NODE LOCALIZATION

Section 8.4 provided an understanding of the different traditional approaches to the localization problem. It is evident that the localization accuracy depends on

the ranging metric, deployment environment (which affects the ranging error statistics), and the relative geometry of the sensor nodes to the anchors. The major difference between traditional localization and WSN localization is cooperative localization. Cooperative localization refers to the collaboration between sensor nodes to estimate their location information. In traditional wireless networks, range information is transferred between RPs to a MT. The RPs are terminals with some *a priori* knowledge of their own coordinates, usually preprogrammed or obtained with some minor uncertainty through GPS. In WSNs, RPs are often referred to as anchors and MTs are referred to as either nodes or *blind* nodes. In the remainder of this chapter, RP and anchor will be used interchangeably and will refer to the same thing.

8.5.1 Cooperative Localization

In WSNs, only a fraction of nodes are anchors that have a prior knowledge of their locations. The main difference from traditional wireless networks is that cooperative localization allows for the transfer of range information between *blind* nodes. In a typical WSN, there are N sensor nodes and M anchors, where $N >> M$. In a 2D plane, the sensor node coordinates are given by

$$\boldsymbol{\theta} = [\mathbf{x}, \mathbf{y}]^T, \tag{8.23}$$

where $\mathbf{x} = [x_1,\ldots,x_N]$ and $\mathbf{y} = [y_1,\ldots,y_N]$ are the x- and y-coordinates of the $\{1,\ldots,N\}$ sensor nodes. Similarly, the coordinates of the M anchor nodes are given by

$$\boldsymbol{\varphi} = [x_{N+1}, \cdots, x_{N+M}, y_{N+1}, \cdots, y_{N+M}]^T. \tag{8.24}$$

For node pairs, i and j, which are within communication coverage, a ranging measurement $\hat{d}_{ij}$ can be obtained using one of the ranging techniques, TOA or RSS. For example, in TOA ranging the range estimate between the ith and jth nodes is given by

$$\hat{\tau}_{ij} = \hat{d}_{ij} \times v, \tag{8.25}$$

where $\hat{d}_{ij}$ is the estimated distance between the nodes and v is the speed of signal propagation. Regardless of the ranging technique, the distance estimate will be corrupted by noise

$$\hat{d}_{ij} = d_{ij} + \varepsilon_{ij}, \tag{8.26}$$

where $d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ is the actual distance between the pair of nodes and ε_{ij} is some random variable describing the statistics of the error that is specific to the ranging technique and the environment. Note that the statistics between different node pairs need not be the same. For example, in indoor multipath

environments, it is well known that for TOA based systems the error is directly related to the availability of the DP signal. Thus ε_{ij} condition between some node pairs i and j may be characterized as zero mean Gaussian, due to the operation in LOS conditions (the availability of the DP signal). On the other hand, the statistic may change drastically when two nodes are separated by some indoor obstacle (e.g., cabinets, elevators or walls), causing ε_{ij} to be nonzero mean Gaussian with a higher variance that reflects the degraded condition. These examples are based on the assumption that the normality component of the error (or the variance) is attributed to the error caused by temporal variations, while the nonzero mean component of the normal distribution is due to a deterministic but unknown bias. See Eqs. (8.5) and (8.6).

For a given sensor network configuration, a ranging technique, and coverage characteristics, the nodes can collaborate in a cooperative fashion to reach a final estimate given by

$$\hat{\boldsymbol{\theta}} = [\hat{x}_1, \cdots, \hat{x}_N, , \hat{y}_1, \cdots, \hat{y}_N]^T. \tag{8.27}$$

Figure 8.10 illustrates the idea of localization through cooperation. In traditional wireless networks, nodes can only range to specific anchors, as shown in Fig. 8.10a. As a result, nodes that are beyond the coverage of sufficient anchors fail to obtain a location estimate. In a cooperate WSN, however, nodes do not need to have a single-hop connection to anchors in order to localize. Cooperative localization makes propagating range information throughout the network possible. Note that due to random deployment in a WSN some parts of the network may still be isolated or ill-connected, which further introduces limitations in position estimation, for example, node (x_1,y_1) in Fig. 8.10b. Obviously, increasing the sensor node density can reduce the probability of isolated subnetworks, but this approach has its own limitations.

The WSN cooperative localization is usually achieved through two major approaches: centralized and distributed. The difference is the reliance of the former on dedicated hardware to solve an optimization problem. The latter, however, allows individual sensor nodes to share range information to reach some global location estimates. In centralized localization, extensive analytical computations are carried out to solve an optimization problem of the entire network. Naturally, this complex computation approach requires a central processing unit external to the sensor network that performs the localization procedure and informs the network of the solution. Figure 8.11 clarifies the difference between the centralized approach and the distributed approach. In either approach, global location estimates can only be achieved if anchors are used. In the absence of anchors, only relative location estimates are possible. More details about relative localization can be found in Ref. [36].

The performance of WSN localization algorithms can be determined through very well established CRLB analysis that has recently attracted attention from different scholars and researchers. The definitions of the bound, and thus the analytical derivation involved, are similar, but they usually differ in their

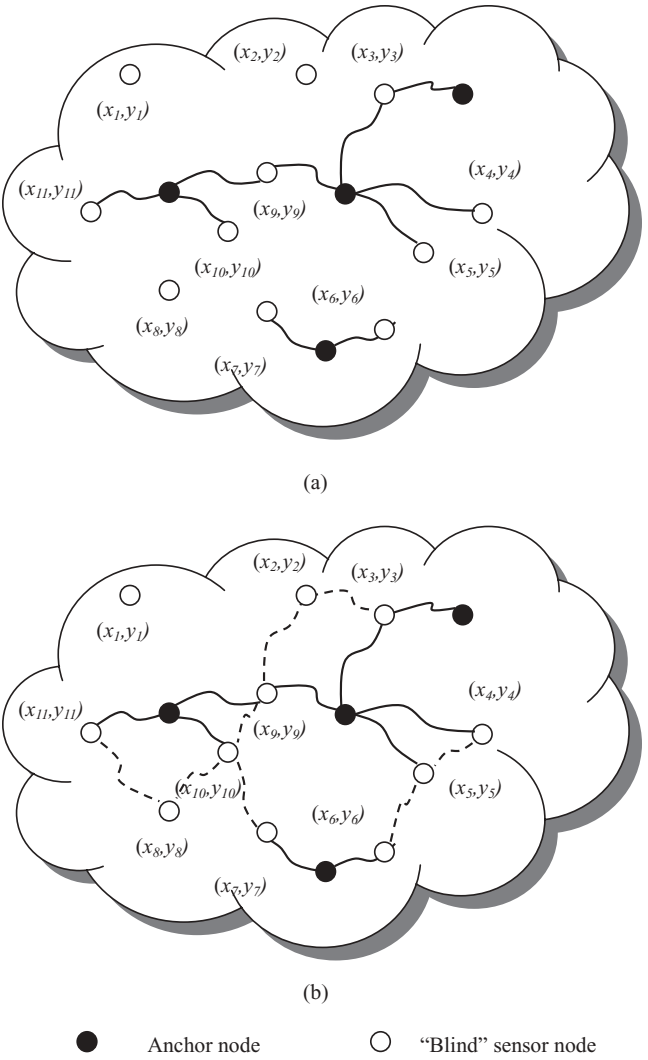


Fig. 8.10 Cooperative localization: (a) Traditional wireless networks. (b) Wireless sensor networks.

assumptions about the characteristics of the corrupting noise. The details of these different approaches for computing the CRLB are beyond the scope of this chapter and the interested reader can find more details in [37–39]. Due to its simplicity and applicability to sensor networks, we now provide an overview of the CRLB analysis provided by Ref. [38] for unbiased estimate of the sensor positions. Although this is not the case in certain environments, for example, indoors, it provides, nonetheless, a very important analytical foundation for analyzing the localization performance in WSNs.

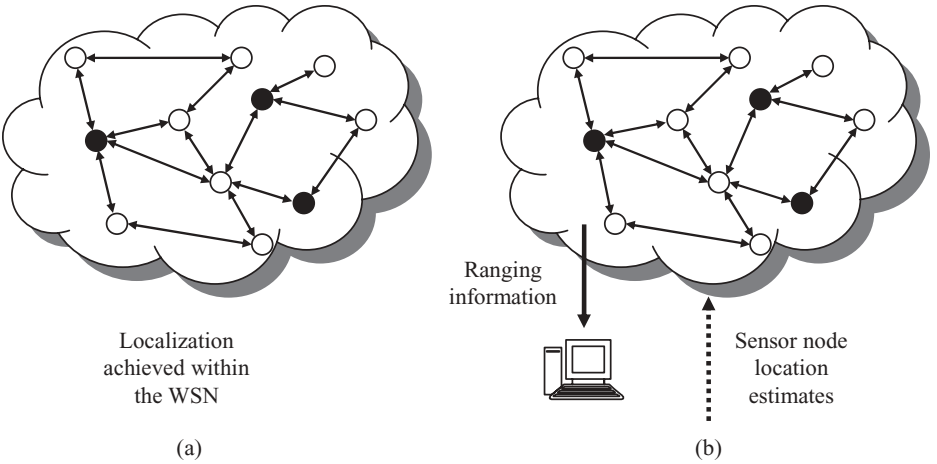


Fig. 8.11 The WSN localization: (a) distributed and (b) centralized.

For known anchor locations, $\boldsymbol{\varphi} = [x_{N+1},\cdots,x_{N+M},y_{N+1},\cdots,y_{N+M}]^T$, we wish to estimate the unknown locations of sensor nodes, $\boldsymbol{\theta} = [x_1,\cdots,x_N,y_1,\cdots,y_N]^T$. The CRLB provides a lower bound on the error covariance matrix for an unbiased estimate of $\boldsymbol{\theta}$ [38]. For a given estimate of the sensor locations $\hat{\boldsymbol{\theta}}$ and Gaussian range measurement noise z, the Fisher information matrix (FIM) can be represented by [38]

$$J(\boldsymbol{\theta}) = E\left\{[\nabla_\theta \ln f_z(Z;\boldsymbol{\theta})][\nabla_\theta \ln f_z(Z;\boldsymbol{\theta})]^T\right\}, \tag{8.28}$$

where $f_z(Z;\boldsymbol{\theta})$ is the joint Gaussian probability density function (PDF) given by

$$f_z(Z;\boldsymbol{\theta}) = \frac{1}{(2\pi)^K |\Sigma|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}[z-\mu(\boldsymbol{\theta})]^T \Sigma^{-1}[z-\mu(\boldsymbol{\theta})]\right\}, \tag{8.29}$$

where $\mu(\boldsymbol{\theta})$ is the vector of the actual distances between the sensor nodes corresponding to available K measurements. The FIM for the specific PDF in Eq. (8.29) can be written as

$$J(\boldsymbol{\theta}) = [G(\boldsymbol{\theta})]^T \Sigma^{-1}[G(\boldsymbol{\theta})], \tag{8.30}$$

where $G(\boldsymbol{\theta})$ contains the partial derivatives of $\mu(\boldsymbol{\theta})$. The CRLB is then given by

$$CRLB = [J(\boldsymbol{\theta})]^{-1}. \tag{8.31}$$

More detailed implementation of the CRLB expression can be found in [38].

The WSN algorithms should then compare the localization performance to the widely available CRLB analysis in the literature. One important note here is that both the bound and the algorithm performance rely mainly on the statistics of the ranging error. Although sensor density and geometry have an impact on the performance, the statistics of the ranging error specifically provides the main challenge for accurate localization. If the ranging error assumptions taken into the algorithm and the CRLB analysis do not reflect the actual behavior of the propagation channel, both the algorithm performance and the bound will be nonrealistic. Let us return to the indoor ranging example where range estimates are not just corrupted by zero mean Gaussian noise. As the discussion revealed earlier, a positive bias can corrupt the TOA measurements requiring analyzing CRLB for biased estimates. In the presence of such biases, the Generalized-CRLB (G-CRLB) can be obtained instead and it has been derived for traditional wireless networks in Ref. [40] and for indoor WSNs in [41].

In the subsequent sections, several examples of emerging localization algorithms based on centralized and distributed approaches will be introduced.

8.5.2 Centralized Localization Algorithms

This section introduces two examples of popular centralized algorithms. One is a convex position estimation, which is an algorithm for estimating unknown node positions in a sensor network based exclusively on connectivity-induced constraints [42]. With this algorithm, the peer-to-peer communication in the network is modeled as a set of geometric constraints on the node position. Thus for nodes operating with a specific type of RF ranging (RSS or TOA), the constraints for the estimates of the location will be provided in an area bounded by the set of constraints. For example, ranging with RSS or TOA causes the constraints to be based on the radial communication coverage. Figure 8.12 shows an example of three different constraints and how they impact the bounding region, where the shaded area represents the possible set of locations constrained by the anchor

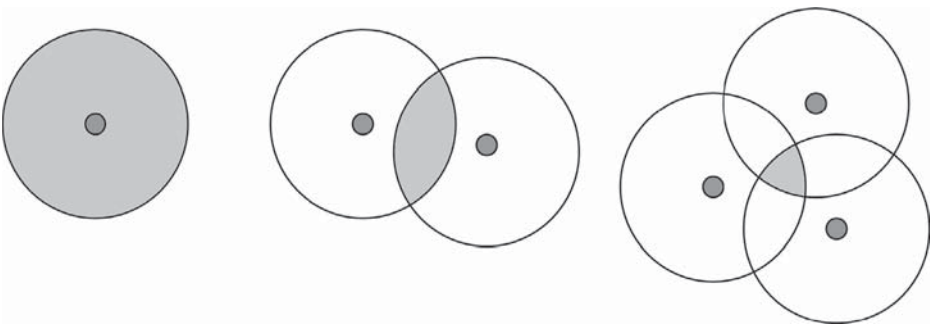


Fig. 8.12 Example of RF radial constraints.

nodes (dark circles). As the number of anchors increase from (a) to (c), the constraints yield smaller feasible sets

As the number of constraints increases, the accuracy of estimating the node position increases. In convex position estimation, the problem is viewed as a graph with the nodes located at the vertices and the bidirectional communication constraints as the edges. Using the proximity constraints and the N sensor nodes and M anchor nodes, it is possible to estimate the position of the sensor nodes [42].

The complexity of the procedure requires centralized computation. Thus all nodes must communicate their connectivity information to an external computer in order to solve the optimization problem. The major advantage of this algorithm is that all connectivity information in a single network is used to obtain the solution. The disadvantage, however, is that the communication load between the nodes and the computer may create a bottleneck that translates into a limitation on the size of the deployed sensor network. The algorithm also provides a rectangular upper bound on each feasible set obtained through the solution. Figure 8.13 shows how the rectangular upper bound is obtained, where the area of the rectangle is related to the number of connections the node has.

Naturally, the accuracy of this algorithm depends on the sensor node density. As the radius of connectivity increases, the number of connections increases, which is equivalent to an increase in node density. The enhanced connectivity (or higher node density) improves the mean error performance. Similarly, as the number of anchors increases, the error substantially decreases. These network parameters highlight the importance of maintaining sufficient node densities in order to achieve an acceptable localization performance. For further details regarding the performance of this algorithm, please refer to Ref. [42].

The other popular algorithm is Multi-Dimensional Scaling (MDS). This algorithm has been applied in the fields of machine learning and computational chemistry, where it consists of a set of data analysis techniques that display the

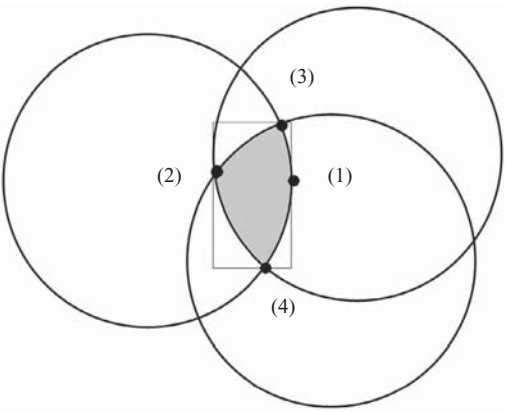


Fig. 8.13 Rectangular upper bound on the constraint.

structure of distance-like data as a geometrical picture [43]. The MDS algorithm starts with one or more distance matrices derived from points in a multidimensional space. It is usually used to find a placement of the points in a low-dimensional space. It is often used as part of exploratory data analysis or information visualization [44]. In classical MDS, the data is quantitative and the proximities of objects are treated as distances in a Euclidean space [45]. The goal of MDS is to find a configuration of points in a multidimensional space such that the interpoint distances are related to the provided proximities by some transformation. If the measured metric did not have any error, the classical MDS would recreate the configuration of points exactly. One advantage of this algorithm is that even with the error present, the solution provided can be reliable due to the over determined nature of the solution.

The basic classical MDS consists of three steps [44]. The first is to compute the shortest paths between all pairs of nodes in the regions under consideration. These shortest-path distances are used to construct the distance matrix. The second is to apply MDS to the distance matrix, where the 2(3) largest eigenvalues and eigenvectors help to construct a 2D(3D) relative map. Finally, with sufficient anchors, it is possible to transform the relative map to an absolute map based on the absolute coordinates of the anchors.

The MDS algorithm is complex and the interested reader can find more details in Ref. [44]. Distributed versions of the MDS localization have been recently proposed and several results have been published in Ref. [46].

8.5.3 Distributed Localization Algorithms

Distributed localization algorithms iteratively achieve an estimate through the sharing of the range and location information. They can be further subdivided into two branches: direct ranging (DR) based and extended ranging (ER) based. The DR based algorithms are usually referred to as recursive position estimation (RPE), while the ER based algorithms are usually referred to as multihop network localization (MNL). Most of the proposed algorithms in the literature are the derivatives of these two algorithms and the distinction is based on the method by which a distance between a node and a given anchor is obtained. In DR, a node only obtains range estimates to anchors. Once the node has range measurements to 3(4) anchors, it is possible to obtain the 2D(3D)-position estimate. The node then joins the existing anchors and helps the remaining nodes in the localization process. Figure 8.14 illustrates the DR-RPE distributed algorithm.

In this example, node A is the only node in the network that has DR measurements to three other anchors. As a result, it obtains a position estimate through the LS or WLS algorithm described earlier. In step 2, node B, with the help of newly transformed A, obtains a position estimate. Node B upgrades to an anchor in step 3. The process repeats and node C becomes an anchor in step 4. Note that one drawback of this algorithm is that it is possible that some nodes on the *edge* of the network lack sufficient direct connectivity anchors and thus are unable to localize themselves.

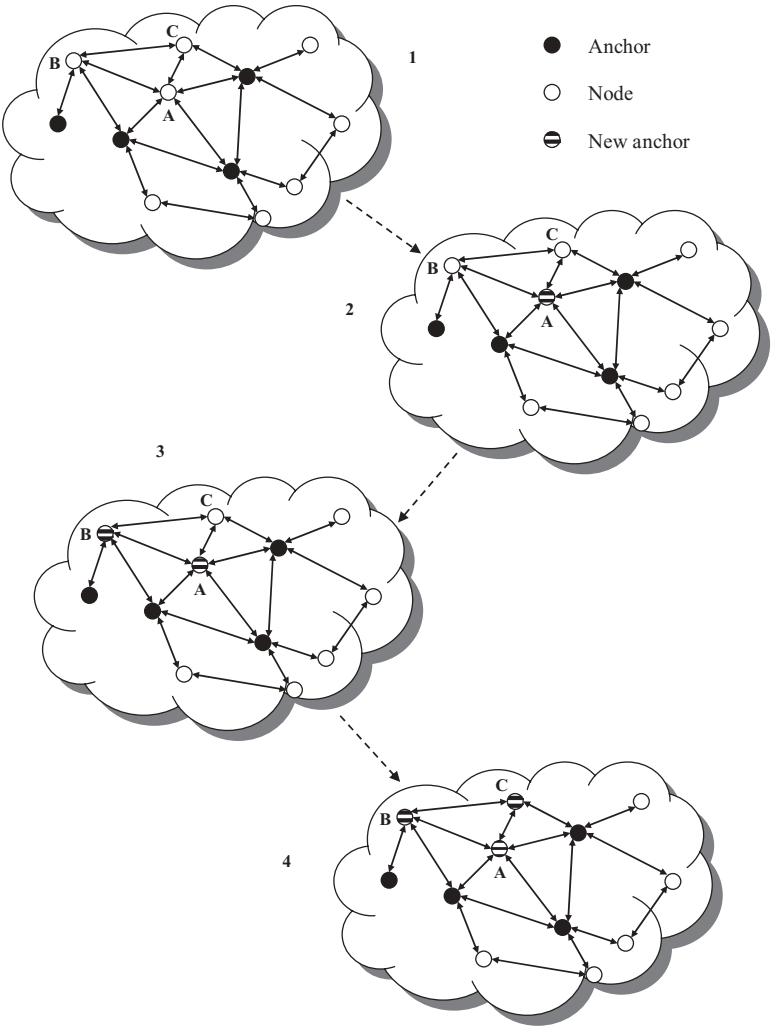


Fig. 8.14 Direct ranging: recursive position estimation distributed localization.

In the case of ER, however, nodes attempt to estimate the distances between themselves and a fixed pool of anchors. The nodes will obtain the distance estimate through a variety of methods, including counting the number of hops to an anchor, measuring the distance to the anchor (adding all distances in the path), or more accurately trying to obtain a geometric estimate of the distance by relying on the relative location of nodes surrounding it. In other words, nodes *extend their range* to anchors by measuring and cooperating to provide an estimate of their distances to an anchor, which is beyond their coverage. Figure 8.15 provides an example of ER distributed localization.

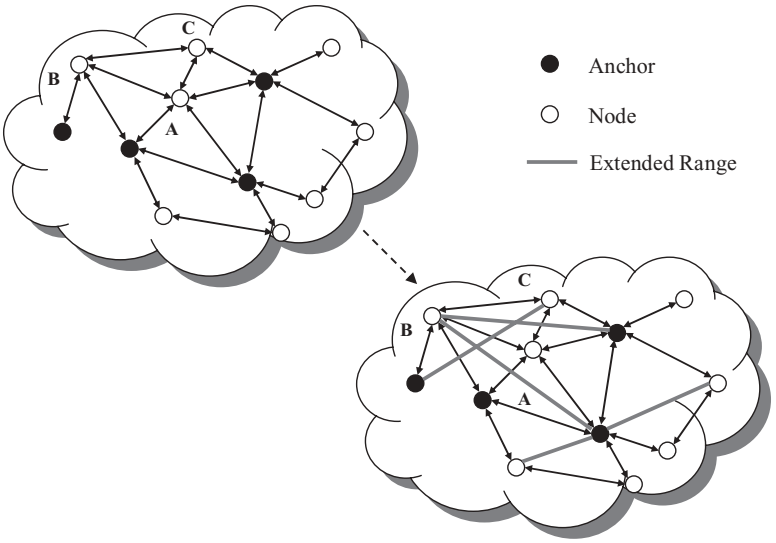


Fig. 8.15 Extended ranging: multihop distributed localization.

In this example, nodes A, B, and C attempt to estimate their distances to the fixed anchors. Once they have that information, they localize themselves. In this fashion, the nodes that are not in the direct range of the anchors get a best-effort estimate of the range. Intuitively, the DR based algorithms are more accurate because there is no error accumulation in the range information. The major drawback of the DR based algorithms, however, is the requirement for a certain node and anchor densities. The advantages include very accurate localization and substantially less error propagation. The DR based algorithms have been reported in Refs. [47,48]. The ER based algorithms, on the other hand, have less reliable error characteristics because the distance to an anchor is not measured. Instead, it is estimated by either the number of hops or geometric estimation. Although the ER based algorithms have less stringent requirements on the densities of anchors and nodes, they exhibit substantial error propagation characteristics, which explain the divergence problems that some of the algorithms in the literature have reported. The ER based algorithms have been used in the N-hop multilateration [49], robust positioning algorithm [49,50], and ad hoc positioning algorithm (APS) [51].

The MNL algorithm is easier to implement than the RPE algorithm because the multihop positioning algorithm requires a minimum of three reference nodes within the whole operational field, assuming mobile nodes can communicate with all reference nodes through multihop communications, while the RPE algorithm has a stricter requirement on the deployment density of reference nodes and mobile nodes. For example, when the deployment density of reference nodes or mobile nodes is low, in some situations, the iterative process may not be able to continue due to the lack of nodes in the close neighborhood.

It is reasonable to expect improved overall performance by integrating the two algorithms. For example, the MNL algorithm can be implemented as a complement of the RPE algorithm; that is, the iterative multilateration algorithm will be used whenever it is possible and when it is not possible, the multihop positioning algorithm will be used to obtain a rough estimate of the location coordinates of the remaining mobile nodes. A coordination procedure needs to be designed to smoothly integrate these two algorithms.

In the remaing sections, we will first introduce ER multihop distributive algorithms and then describe some emerging DR-RPE algorithms.

8.5.3.1 Multihop Network Localization. The basic idea behind MNL is estimating the distance between an unknown node and a certain anchor node. Once the distance to at least 3(4) anchors are obtained, the LS or WLS algorithms can be used to obtain a position estimate $\hat{\boldsymbol{\theta}}_i = [\hat{x}_i, \hat{y}_i]$. Three major methods can be used to obtain an extended/multihop range. The first method is using hop number, which counts the number of node hops to the anchor. Then the node uses some distance/hop metric to obtain an estimate of the distance to the anchor, that is,

$$\hat{d}_{ij} = H \times \pi, \tag{8.32}$$

where H is the number of hops to the anchor, π is a certain distance/hop metric, and $\hat{d}_{ij}$ is the estimated distance between the ith node and the jth anchor. This method is also known as *DV-Hop* and *Hop-TERRAIN* in [51] and [49], respectively.

The second method is using hop distance, which differs from the previous one in that the measured distance between a node and an anchor is propagated instead of the number of hops. This method is extremely sensitive to the measurement errors and it is also known as *DV-distance* in Ref. [51]. In this method, the extended range estimate to the anchor can be given by

$$\hat{d}_{ij} = \sum_{k=1}^{L} \hat{d}_k, \tag{8.33}$$

where $\hat{d}_k$ are the L distances between the nodes along the path to the anchor. Essentially, the distance estimate in Eq. (8.33) becomes less reliable when the node density decreases because the interdistance spacing between the nodes becomes larger.

The last method is based on computing the geometric distance from the node to the far anchor. This method has been implemented as *Euclidean* and *TERRAIN* in [51] and [49], respectively. As examples, we will give a brief description of *TERRAIN* and *Hop-TERRAIN* in the next couple of subsections.

TERRAIN

TERRAIN stands for *triangulation via extended range and redundant association of intermediate nodes*. It is an extension of the *assumption-based coordinates* (ABC) algorithm [48] that offers enhanced position estimation. The ABC algorithm determines the locations of unknown nodes one at a time in the order that they establish communication, making assumptions where necessary, and compensating for errors through corrections and redundant calculations as more information becomes available [49]. These assumptions are necessary at first in order to deal with the underdetermined set of equations presented by the first few nodes. The description of the general algorithm assumes the perspective of node n_0.

The algorithm begins with the assumption that n_0 is located at $(0, 0, 0)$. The first node to establish communication with n_0, n_1, is assumed to be located at $(r_{01}, 0, 0)$, where r_{01} is the determined distance between n_0 and n_1. The location of the next node n_2, which is given by (x_2,y_2,z_2), can then be explicitly solved for two assumptions: the square root involved in finding y_2 is assumed to yield a positive result, and z_2 is assumed to be 0 [49]; that is,

$$x_2 = \frac{r_{01}^2 + r_{02}^2 + r_{12}^2}{2r_{01}} \qquad y_2 = \sqrt{r_{02}^2 - x_2^2}\,. \tag{8.34}$$

From this point forward, the system of equations used to solve for further nodes is no longer underdetermined, and thus the standard algorithm can be employed for each new node. Under ideal conditions, this algorithm will produce a topologically correct map with a random orientation relative to a global coordinate system [49].

The ABC algorithm allows a network of nodes with unknown positions to cooperatively locate themselves with respect to each other, but not with respect to any global reference system. In other words, ABC can be used to create a topologically correct map that is referenced to as a coordinate system known only to those nodes in the network. This coordinate system will have a completely random orientation with respect to any global coordinate system. Although rotations could be performed to fix this orientation problem, TERRAIN is layered on top of ABC to achieve this end with less error in the final position estimates [49].

TERRAIN makes use of the ABC algorithm to incorporate each node into several independent maps, one map for each anchor node. Once a node is included in a sufficient number of maps, it is then able to locate itself with respect to the global coordinate system. Each anchor node in a network initiates the ABC algorithm, creating M independent sets of coordinate systems (if M is the number of anchors), each anchored at its respective anchor node. From the perspective of just one of these anchor

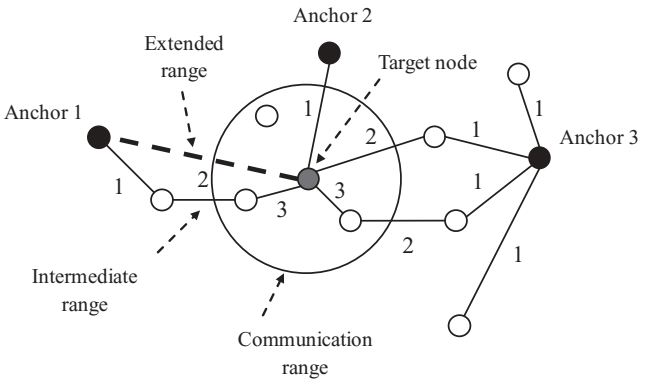


Fig. 8.16 Illustration of the TERRAIN algorithm.

nodes, the ABC algorithm will propagate through the network, causing each node to be located within the coordinate system for that independent ABC algorithm. A node in the center of the network will eventually acquire M sets of coordinates for itself, each relative to one of the maps generated at one of the anchor nodes.

Although each of these positions is referenced to a seemingly random coordinate space, they can be used to derive an estimated distance (or range) to the anchor node associated with each coordinate system [49]. These distances are the *extended ranges* of TERRAIN, and they are used to artificially extend the visible range of each node so that it can associate itself with each anchor in the network. Once a node has estimated its range to at least one more anchor than the dimension of the space (i.e., three anchors in 2D, four anchors in 3D), it is able to perform a standard triangulation computation to discover its position in the global space. Figure 8.16 illustrates how the TERRAIN algorithm works.

Hop-TERRAIN

The Hop-TERRAIN algorithm is based on the TERRAIN concept of extending ranges in order to enable standard triangulation computations. Unlike TERRAIN, the Hop-TERRAIN algorithm finds the number of routing hops from a node to each of the anchor nodes in a network and then multiplies this hop count by a shared metric (e.g., average hop distance) to estimate the ranges between the node and each anchor. These computed ranges are then used together with the anchor nodes' known positions to perform a triangulation and get the node's estimated position. Hop-TERRAIN realizes advantages over TERRAIN in two ways. First, Hop-TERRAIN does not use the magnitude of the measured range, but rather only checks to see if communication was established. Thus, fluctuations in range accuracy are ignored. Second, Hop-TERRAIN does not

iteratively compound errors like TERRAIN does. These advantages help make Hop-TERRAIN much more robust against range errors, on average yielding more consistent and accurate position estimates.

Figure 8.17 shows the basic idea of the Hop-TERRAIN algorithm. In this figure, the average hop metric is 2 m and the distance from anchor 1 to the targeted node is 3-hops, which translates to an average of 6 m. Similarly, the distance between the targeted node and the other two anchors will be found and then the targeted node can be triangulated with these three estimated extended ranges.

The performance of the multihop positioning algorithm is also closely related to the density of reference nodes and mobile nodes. For example, if mobile nodes are densely deployed in the field, the hop-by-hop route from a mobile node to a remote anchor node will closely be a straight line. For sparse densities, however, the hop-by-hop distance from a node to an anchor is far from straight and it is possible to see why this algorithm suffers in these situations. The multihop positioning algorithm also experiences error accumulation and the divergence problem similar to the situation with the iterative multilateration algorithm. The performance and convergence property of the algorithm needs to be carefully studied using computer simulations.

8.5.3.2 Recursive Position Estimation. The basic idea of recursive position estimation (RPE) is that sensors iteratively transform into anchors when possible and aid the remainder of the nodes in the network to localize. This type of algorithms is based on the concept reported in Ref. [47]. In the earlier description of ranging, it was highlighted that the major error contribution in localization stems from corrupted range estimates. Therefore, an RPE algorithm must integrate the channel condition into the localization process. In this section, we focus on a specific RPE algorithm that is based on a single-hop ranging method, that is, Cooperative LOcalization with Quality of Estimate (CLOQ). The CLOQ

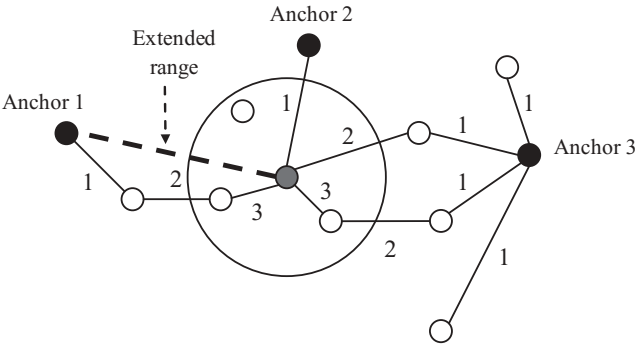


Fig. 8.17 Illustration of the Hop-TERRAIN Algorithm.

algorithm integrates the quality of the channel information into the localization procedure to produce reliable estimates. The algorithm takes advantage of UWB TOA based indoor measurement and modeling.

CLOQ

Looking at the problem from a channel-modeling perspective, it becomes evident that the ranging error is a major obstacle facing the cooperative positioning algorithm. As a result, the methodology followed in this algorithm utilizes channel models that characterize the relationship between the first path power and the statistics of the ranging error. The sensor node measures the TOA of the first path arrival and its power. The results in [48] have shown a strong correlation between the power and the ranging error. As a result, the CLOQ algorithm effectively ranks the range measurements to the anchors according to the received power. Higher signal power implies statistically lower ranging error. Figure 8.18 illustrates the relationship between the first path power and the distance errors.

When the sensor node is close to the reference point, the power of the DP is very strong. As a result, the first path can be easily detected, which means very accurate ranging. This region of operation is usually referred to as detected direct path (DDP). In the second region, a mixture of DDP and UDP conditions provide acceptable ranging, but occasionally large error. When the DP can no longer be detected, the third region of operation is known as UDP, where it suffers the most errors. As such, it exhibits the worse ranging accuracy. It is possible then to characterize the statistics; that is, mean and variance, of each region of operation. The relationship between the power and the statistics of the ranging error is provided in Table 8.2.

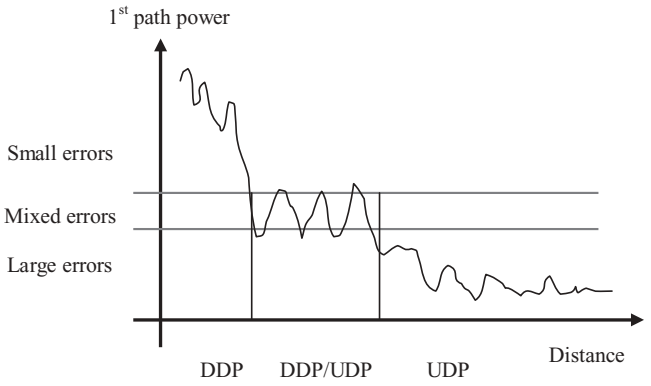


Fig. 8.18 Relationship among first-path power, distance, and distance measurement errors.

TABLE 8.2 Multiregion Ranging Model

DME Region	First Path Power (dB)	μ(m)	σ^2 (m^2)
1	Pr > –90	0.1	0.01
2	–105 < Pr < –90	0.4	0.09
3	–115 < Pr < –105	1	1.96
No Coverage	Pr < –115	NA	NA

Pr = received power.

TABLE 8.3 Quality of Position Estimates[a]

QoL Combinations	QoE	Average Position Error (m)	Maximum Position Error (m)
$\sigma_1^2,\sigma_1^2,\sigma_1^2$	0.03	0.10	0.32
$\sigma_1^2,\sigma_1^2,\sigma_2^2$	0.05	0.19	0.76
$\sigma_1^2,\sigma_2^2,\sigma_2^2$	0.07	0.26	0.90
$\sigma_2^2,\sigma_2^2,\sigma_2^2$	0.09	0.30	0.91
$\sigma_1^2,\sigma_1^2,\sigma_3^2$	1.98	0.74	3.25
$\sigma_1^2,\sigma_2^2,\sigma_3^2$	2.00	0.83	3.37
$\sigma_2^2,\sigma_2^2,\sigma_3^2$	2.02	0.83	3.02
$\sigma_1^2,\sigma_3^2,\sigma_3^2$	3.93	1.27	7.40
$\sigma_2^2,\sigma_3^2,\sigma_3^2$	3.95	1.28	7.31
$\sigma_3^2,\sigma_3^2,\sigma_3^2$	5.88	1.59	8.00

[a]Quality of position estimates = QoE.

The CLOQ algorithm is composed of four stages: anchor selection, position estimation, anchor nomination, and new anchor incorporation.

In the first stage, a node is listening to the channel for candidate anchors. Once the node receives range measurements to different anchors, it ranks them according to the quality of link (QoL), which is an index that corresponds to one of the distance measurement error (DME) regions in Table 8.2. The node then selects 3 anchors with the best QoL for triangulation. The transformation to an anchor then involves estimating the impact of the channel on the position accuracy and thus forming another metric called quality of estimate (QoE). The QoE index serves the main purpose of ranking the position integrity of the anchors, while QoL provides information regarding the integrity of the RF channel. The relationship between the QoL and QoE indices can be obtained from Table 8.3. The procedure is best illustrated in an example that is provided in Fig. 8.19.

The example starts with four original anchors or RPs and three newly transformed anchors, nodes A, B, and C. At that moment, node D listens to the channel and hears from five different anchors. The original anchors or RPs have a QoE index of 0 because it is assumed that they know their locations exactly.

The process starts with node D ranking the anchors according to the QoL, and if there are similar QoLs, it ranks them according to the QoE of the

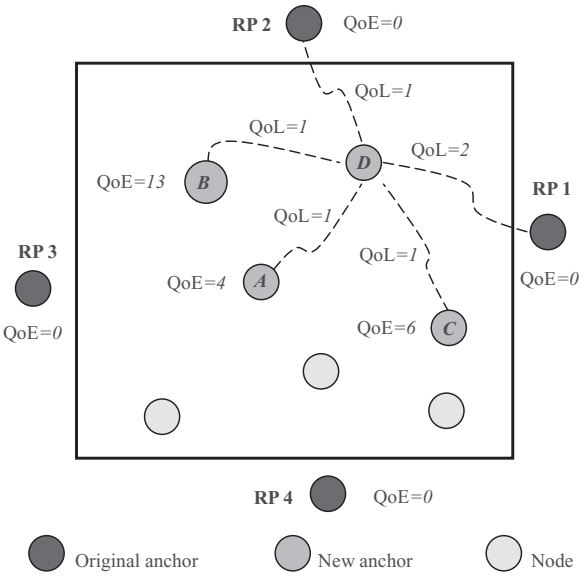


Fig. 8.19 The CLOQ algorithm and the anchor selection criteria.

respective anchor. In this case, node D will choose anchors RP2, A, and C for triangulation and QoE computation. The QoE index computation is carried out by essentially weighting the previous anchors' QoEs and the QoLs of the respective range.

After triangulating their own positions, the nodes compute their QoE and enter into a transition state called *anchor nominee*, where they have to compete with other neighboring nominees, as illustrated in Fig. 8.20. If an anchor nominee has the best QoE (lowest value), then it establishes itself as an anchor. If it receives better QoE, then it returns to the node mode and attempts this procedure yet another time. In the cases that there are several nominees with the same best QoE (i.e., nodes A and D), they all become anchors. In actual implementation, a spanning tree technique should be used to avoid unnecessary flooding to the network. Only new information triggers a broadcast, that is, when a nominee receives better QoE information. In Fig. 8.20, node B does not broadcast the information it received from node C regarding the QoE of six to node A. The reduction in unnecessary message exchange can greatly reduce the overhead of the CLOQ algorithm.

Finally, all the nodes that had the chance to upgrade to anchors join the pool of other anchors in helping the remaining nodes to estimate their positions. The newly elected anchors start broadcasting their positions and their own QoE indices to the entire network. Eventually, all the nodes end up in this stage where they have estimated their positions with great accuracy. Further details of the CLOQ algorithm are provided in [48].

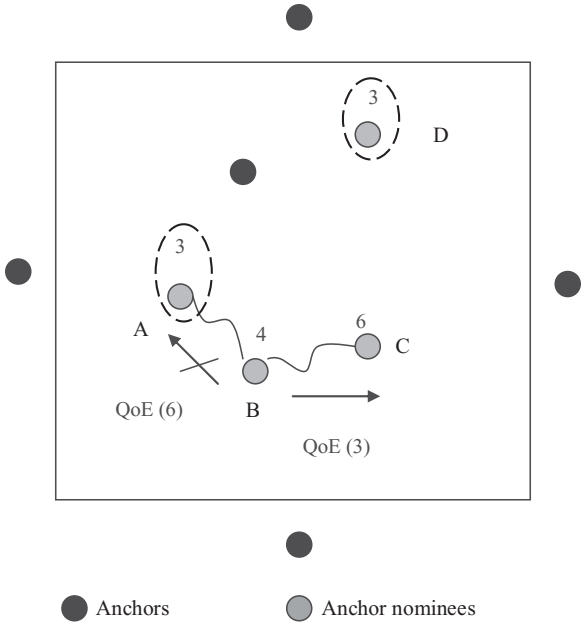


Fig. 8.20 The CLOQ algorithm and the anchor nomination process.

8.6 SUMMARY AND FUTURE DIRECTIONS

The theory of localization dates all the way back to the early days of GPS. Its gradual evolution to wireless systems made possible enhancing of existing applications with location information. The major ranging techniques, RSS and TOA, which were popular in traditional wireless systems, have gained considerable attention for WSNs. The RSS techniques are popular due to their simplicity in implementation, but have problems in accuracy due to the complexity of the radio channel propagation characteristics. The TOA techniques, especially UWB based, promise accuracy within centimeters and indeed deliver in LOS channel conditions or low obstruction environments. In NLOS, however, the TOA techniques face challenging multipath and DP blockage that can corrupt the range estimate. In addition, the enhanced accuracy comes at a price of increased system complexity.

Traditional localization techniques, for example, LS and WLS, can be incorporated into WSNs. Two main approaches to cooperative localization are centralized and distributed. Centralized algorithms focus on convex positioning and MDS techniques, where the problem is to estimate all nodes given the available range information. Distributed algorithms, on the other hand, focus on disseminating position and range information to other nodes and from the anchors in order to establish global position coordinates. Centralized algorithms are usually

more complex in implementation. Distributed algorithms, although simpler in implementation, nonetheless have their own limitations in terms of coverage problems and error propagation throughout the sensor network.

In either approaches, two major challenges face WSN localization. The first is the error inherent in the ranging technique used. As described in this chapter, the statistics of the ranging error are significantly different for different techniques and deployment environments. As a result, algorithms must have a mechanism to incorporate environment-specific channel statistics and quantify the quality of the range and position estimates. The second challenge is the problem of error propagation where the location error from sensor nodes close to anchors propagates through the network, causing divergence problems for nodes further away. Thus algorithms must contain methodologies to identify and bound error propagation. Addressing these two major issues will enhance localization accuracy and improve the reliability of location information for WSN applications.

REFERENCES

[1] I. Akyildz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks", *IEEE Communications Magazine*, vol. 40, no. 8, Aug. 2002, pp. 102–114.

[2] N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero, R. L. Moses, and N. S. Correal, "Locating the nodes: Cooperative localization in wireless sensor networks", *IEEE Signal Processing Magazine*, vol. 22, no. 4, July 2005, pp. 54–69.

[3] K. Pahlavan, F. O. Akgul, M. Heidari, A. Hatami, J. M. Elwell, and R. D. Tingley, "Indoor geolocation in the absence of the direct path", *IEEE Wireless Communications Magazine*, vol. 13, no. 6, Dec. 2006, pp. 50–58.

[4] K. Pahlavan, X. Li, and J.-P. Makela, "Indoor geolocation science and technology", *IEEE Communications Magazine*, vol. 40, no. 2, Feb. 2002, pp. 112–118.

[5] J. Caffery and G. Stuber, "Subscriber location in CDMA cellular networks", *IEEE Transactions on Vehicular Technology*, VT-47(2), May, 1998, pp. 406–416.

[6] J. McGeough, "Wireless location positioning, based on signal propagation data", White Paper, http://www.wirelessdevnet.com/software/, 2002.

[7] K. Pahlavan and P. Krishnamurthy, *Principles of Wireless Networks: A Unified Approach*, Prentice Hall, Upper Saddle River, NJ, 2002.

[8] H. Koshima and J. Hoshen, "Personal locator services emerge", *IEEE Spectrum*, vol. 37, no. 2, Feb. 2000, pp. 41–48.

[9] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors", *Communications of the Association for Computing Machinery (ACM)*, vol. 43, no. 5, May 2000, pp. 51–58.

[10] K. Pahlavan, P. Krishnamurthy, A. Hatami, M. Ylianttila, J. Makela, R. Pichna, and J. Vallstrom, "Handoff in hybrid mobile data networks (invited paper)", *IEEE Personal Communications Magazine*, vol. 7, no. 2, Apr. 2000, pp. 34–47.

[11] Y. Ko and N. H. Vaidya, "Location-aided routing (LAR) in mobile ad hoc networks", in *Proceedings of 1998 ACM/IEEE Mobile Computing and Networking* (MOBICOM'98), Dallas, TX, Oct. 1998, pp. 66–75.

[12] R. Jain, A. Puri, and R. Sengupta, "Geographical routing using partial information for wireless ad hoc networks", *IEEE Personal Communications Magazine*, vol. 8, no. 1, Feb. 2001, pp. 48–57.

[13] A. Smailagic and D. Kogan, "Location sensing and privacy in a context-aware computing environment", *IEEE Wireless Communications*, vol. 9, no. 5, Oct. 2000, pp. 10–17.

[14] K. Pahlavan, P. Krishnamurthy, and J. Beneat, "Wideband radio channel modeling for indoor geolocation applications", *IEEE Communications Magazine*, vol. 36, no. 4, Apr. 1998, pp. 60–65.

[15] P. Krishnamurthy and K. Pahlavan, "Radio propagation modeling for indoor geolocation applications", in *Proceedings of IEEE Personal, Indoor and Mobile Radio Communications* (PIMRC'99), Kyoto, Japan, Sept. 1999, pp. 446–450.

[16] R. Fontana and S. Gunderson, "Ultra-wideband precision asset location system", in *Proceedings of 2002 IEEE Conference on UWB Systems and Technologies* (UWBST'02), Baltimore, MD, May 2002, pp. 147–150.

[17] P. Bahl and V. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system", in *Proceedings of IEEE INFOCOM'02*, vol. 2, Tel Aviv, Israel, Mar. 2000, pp. 775–784.

[18] P. Bahl, V. N. Padmanabhan, and A. Balachandran, "*Enhancements to the RADAR User Location and Tracking System*", Tech. Rep. MSR-TR-00-12, Microsoft Research, Feb. 2000.

[19] J. Werb and C. Lanzl, "Designing a positioning system for finding things and people indoors", *IEEE Spectrum*, vol. 35, no. 9, Sept. 1998, pp. 71–78.

[20] T. Roos, P. Myllymaki, H. Tirri, P. Miskangas, and J. Sievanen, "A probabilistic approach to WLAN user location estimation", *International Journal of Wireless Information Networks*, vol. 9, no. 3, July 2002, pp. 155–164.

[21] T. Roos, P. Myllymaki, and H. Tirri, "A statistical modeling approach to location estimation", *IEEE Transactions on Mobile Computing*, MC-1(1), Jan. 2002, pp. 59–69.

[22] J. Y. Lee and R. A. Scholtz, "Ranging in a dense multipath environment using an UWB radio link", *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 9, Dec. 2002, pp. 1677–1683.

[23] K. Pahlavan and A. Levesque, *Wireless Information Networks*, John Wiley & Sons, Inc., New York, 2nd ed., 2005.

[24] N. Alsindi, B. Alavi, and K. Pahlavan, "Spatial characteristics of UWB ranging in indoor multipath environments", in *Proceedings of 2007 IEEE International Symposium on Personal Indoor and Mobile Radio Communications* (PIMRC'07), Athens, Greece, Sept. 2007, pp. 1–6.

[25] X. Li and K. Pahlavan, "Super-resolution TOA estimation with diversity for indoor geolocation", *IEEE Transactions on Wireless Communications*, vol. 3, no. 1, Jan. 2004, pp. 224–234.

[26] S. Gezici, Z. Tian, G. B. Giannakis, H. Kobayashi, A. F. Molisch, H. V. Poor and Z. Sahinoglu, "Localization via ultra-wideband radios", *IEEE Signal Processing Magazine*, vol. 22, no. 4, July 2005, pp. 70–84.

[27] M. Heidari, F. O. Akgul, and K. Pahlavan, "Identification of the absence of direct path in indoor localization systems", in *Proceedings of IEEE International Symposium on Personal Indoor and Mobile Radio Communications* (PIMRC'07), Athens, Greece, Sept. 2007, pp. 1–6.

[28] I. Guvenc, C.-C. Chong, and F. Watanabe, "NLOS identification and mitigation for UWB localization systems", in *Proceedings of IEEE Wireless Communications and Networking Conference* (WCNC'07), Hong Kong, China, Mar. 2007, pp. 1571–1576.

[29] Y. Qi and H. Kobayashi, "On relation among time delay and signal strength based geolocation methods", in *Proceedings of 2003 IEEE Global Communications Conference* (GLOBECOM'03), vol. 7, San Francisco, CA, Dec. 2003, pp. 4079–4083.

[30] P. Misra and P. Enge, *Global Positioning System: Signals, Measurements and Performance*, Ganga-Jamuna Press, Lincoln, MA, 2002.

[31] E. D. Kaplan, *Understanding GPS: Principles and Applications*, Artech House, Norwood, MA, 1996.

[32] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*, Prentice Hall, Upper Saddle River, NJ, 1993.

[33] P.-C. Chen, "A non-line-of-sight error mitigation algorithm in location estimation", in *Proceedings of 1999 IEEE Wireless Communications and Networking Conference* (WCNC'99), New Orleans, LA, Sept. 1999, pp. 316–320.

[34] M. Spirito, "On the accuracy of cellular mobile station location estimation", *IEEE Transactions on Vehicular Technology*, vol. 50, no. 3, May 2001, pp. 674–684.

[35] M. Kanaan and K. Pahlavan, "CN-TOA: A new algorithm for indoor geolocation", in *Proceedings of 2004 IEEE Wireless Communications and Networking Conference* (WCNC'04), Atlanta, GA, Sept. 2004, pp. 1906–1910.

[36] Y. Shang, J. Meng, and H. Shi, "A new algorithm for relative localization in wireless sensor networks", in *Proceedings of the 18th International Parallel and Distributed Processing Symposium*, Santa Fe, NM, Apr. 2004, pp. 24–34.

[37] E. G. Larsson, "Cramer-Rao bound analysis of distributed positioning in sensor networks", *IEEE Signal Processing Letters*, vol. 11, no. 3, Mar. 2004, pp. 334–337.

[38] A. Savvides, W. L. Garber, R. L. Moses, and M. B. Srivastava, "An analysis of error inducing parameters in multihop sensor node localization", *IEEE Transactions on Mobile Computing*, vol. 4, no. 6, Nov. 2005, pp. 567–577.

[39] C. Chang and A. Sahai, "Cramer-Rao type bounds for localization", *EURASIP Journal on Applied Signal Processing*, vol. 2006, article id 94287, Jan. 2006, pp. 1–13.

[40] Y. Qi, H. Kobayashi, and H. Suda, "Analysis of wireless geolocation in a non-line-of-sight environment", *IEEE Transactions on Wireless Communications*, vol. 5, no. 3, Mar. 2006, pp. 672–681.

[41] N. Alsindi and K. Pahlavan, "Cooperative localization bounds for indoor ultra wideband wireless sensor networks", *EURASIP Journal on Applied Signal Processing (ASP)*, vol. 2008, article id 852809, 2008, pp. 1–13.

[42] L. Doherty, K. S. J. Pister, and L. El Ghaoui, "Convex position estimation in wireless sensor networks", in *Proceedings of IEEE INFOCOM'01*, vol. 3, Anchorage, AK, Apr. 2001, pp. 1655–1663.

[43] I. Borg and P. Groenen, *Modern Multidimensional Scaling, Theory and Applications*, Springer-Verlag, New York, 1997.

[44] Y. Shang, W. Ruml, Y Zhang, and M. Fromherz, "Localization from connectivity in sensor networks", *IEEE Transactions on Parallel and Distributed Systems*, no. 11, vol. 15, Nov. 2004, pp. 961–974.

[45] W. S. Torgerson, "Multidimensional scaling of similarity", *Psychometrika*, vol. 30, Dec. 1965, pp. 379–393.

[46] J. A. Costa, N. Patwari, and A. O. Hero III, "Distributed weighted-multidimensional scaling for node localization in sensor networks", *ACM Transactions on Sensor Networks (TOSN)*, vol. 2 no. 1, Feb. 2006, pp. 39–64.

[47] J. Albowicz, A. Chen, and L. Zhang, "Recursive position estimation in sensor networks", in *Proceedings of 2001 IEEE International Conference on Network Protocols* (ICNP'01), Riverside, CA, Nov. 2001, pp. 35–41.

[48] N. Alsindi, K. Pahlavan, B. Alavi, and X. Li, "A novel cooperative localization algorithm for indoor sensor networks", in *Proceedings of 2006 IEEE International Symposium on Personal Indoor and Mobile Radio Communications* (PIMRC'06), Helsinki, Finland, Sept. 2006, pp. 1–6.

[49] C. Savarese, J. M. Rabaey, and J. Beutel, "Locationing in distributed ad-hoc wireless sensor networks", in *Proceedings of 2001 International Conference on Acoustics, Speech, and Signal Processing* (ICASSP'01), Salt Lake City, UT, May 2001, pp. 2037–2040.

[50] C. Savarese, K. Langendoen, and J. Rabaey, "Robust positioning algorithms for distributed ad-hoc wireless sensor networks", in *Proceedings of USENIX Technical Annual Conference*, Monterrey, CA, June 2002, pp. 317–328.

[51] D. Niculescu and B. Nath, "Ad-hoc positioning system", in *Proceedings of 2001 IEEE Global Communications Conference* (Globecom'01), San Antonio, TX, Nov. 2001, pp. 2926–2931.

9

TIME SYNCHRONIZATION

Fikret Sivrikaya and Bülent Yener

Rensselaer Polytechnic Institute, USA

9.1 INTRODUCTION

Time synchronization, as in all distributed systems, is an important component of a wireless sensor network (WSN), which aims to provide a common timescale for local clocks of nodes in the network. Since all hardware clocks are imperfect, those at different nodes may drift away from each other in time. For this reason, the observed time or durations of time intervals may differ for each node in the network. However, for many applications or networking protocols, it is required that a common view of time exist and be available to all or some of the nodes in the network at any particular instant.

This chapter focuses on the time synchronization problem, and reviews existing synchronization methods and protocols for WSNs. Section 9.1 introduces the synchronization problem and the common challenges for synchronization methods. Sections 9.2 and 9.3 discuss the need for synchronization and the requirements of synchronization in WSNs, respectively. Section 9.4 reviews existing synchronization methods and protocols for WSNs. Section 9.5 concludes with a summary of this chapter and a brief discussion of future research directions.

Wireless Sensor Networks: A Networking Perspective, Edited by Jun Zheng and Abbas Jamalipour

9.1.1 Computer Clocks and the Synchronization Problem

Computing devices are mostly equipped with a hardware oscillator-assisted computer clock, which implements an approximation $C(t)$ of real time t as

$$C(t) = k\int_0^t \omega(\tau)\, d\tau + C(t_0),$$

where $\omega(t)$ is the angular frequency of the hardware oscillator, k is a proportionality coefficient, and t_0 is the initial value of the clock [1]. For a perfect clock, dC/dt would equal 1. However, all clocks are subject to a clock drift; the oscillator frequency will vary unpredictably due to various physical effects. Even though the frequency of a clock changes over time, it can be approximated with good accuracy by an oscillator with a fixed frequency [2]. Then for some node i in the network, we can approximate its local clock as

$$C_i(t) = a_i t + b_i,$$

where a_i is the clock drift, and b_i is the offset of node i's clock. *Drift* denotes the rate (frequency) of the clock, and *offset* is the difference in value from real time, t. Using the equation above, we can compare the local clocks of two nodes in a network, say, nodes 1 and 2 as

$$C_1(t) = a_{12} \cdot C_2(t) + b_{12}. \tag{9.1}$$

We call a_{12} the *relative drift*, and b_{12} the *relative offset* between the clocks of nodes 1 and 2. If two clocks are perfectly synchronized, their relative drift is 1, meaning that the clocks have the same rate, and their relative offset is zero, meaning that they have the same value at that instant. Some studies in the literature use "skew" instead of drift, defining it as the *difference* (as opposed to *ratio*) between clock rates [3,4]. Also, offset may equivalently be mentioned as "phase offset".

The synchronization problem on a network of n devices corresponds to the problem of equalizing the computer clocks of different devices. The synchronization can be either *global*, trying to equalize $C_i(t)$ for all $i = 1, 2, \ldots, n$, or *local*, trying to equalize $C_i(t)$ for some set of the nodes—mostly those that are spatially close or on the same path between communicating nodes. Equalizing just the instantaneous values (correcting the offsets) of clocks is not enough for synchronization because the clocks will drift away afterward. Therefore, a synchronization scheme should either equalize the clock rates as well as offsets, or repeatedly correct the offsets to keep the clocks synchronized over a time period.

The above definition of synchronization actually outlines the strictest form of synchronization, where one seeks perfect matching of time on different clocks, but this definition can be relaxed to different degrees, according to the need of an application. In general, the synchronization problem can be classified into three basic types [5]. The first and simplest type of synchronization deals only

with the ordering of events or messages. The aim of such synchronization is to tell whether an event E_1 has occurred before or after another event E_2 (i.e., just compare the local clocks for order rather than have them synchronized). The synchronization protocol proposed in Ref. [6] is an example of this type. The second type of synchronization targets maintaining relative clocks. In such synchronization, a node runs its local clock independently, but keeps information about the relative drift and offset of its clock to other clocks in the network so that at any instant the local time of the node can be converted to some other node's local time and vice versa. Most of the synchronization protocols proposed for sensor networks use this model [2,3,7]. The third and most complex type of synchronization is the "always on" model, where all nodes maintain a clock that is synchronized to a reference clock in the network. The goal of this type of synchronization is to preserve a global timescale throughout the network. The synchronization protocol proposed in Ref. [5] conforms to this model, but the use of the "always on" model is not mandatory in the protocol.

9.1.2 Common Challenges for Synchronization Methods

All network time synchronization methods rely on some sort of message exchange between nodes. Nondeterminism in the network dynamics, for example, propagation time or physical channel access time, makes the synchronization task challenging in many systems. When a node in the network generates a timestamp to send to another node for synchronization, the packet carrying the timestamp will face a variable delay until it reaches and is decoded at its intended receiver. This delay prevents the receiver from exactly comparing the local clocks of the two nodes and accurately synchronizing to the sender. We can basically decompose the sources of errors in network time synchronization methods into four basic components:

1. *Send Time.* This is the time spent to construct a message at the sender. It includes the overhead of the operating system (e.g., context switching) and the time to transfer the message to the network interface for transmission.
2. *Access Time.* Each packet faces some delay at the medium access control (MAC) layer before actual transmission. The sources of this delay depend on the MAC scheme used, but some typical reasons for delay are waiting for the channel to be idle or waiting for the time-division multiple access (TDMA) slot for transmission.
3. *Propagation Time.* This is the time spent in propagation of the message between the network interfaces of the sender and the receiver.
4. *Receive Time.* This is the time needed for the network interface of the receiver to receive the message and transfer it to the host.

In large networks, the propagation time may become quite large and important because it includes the queuing and switching delays at the routers on a path

between two nodes. However, for two nodes in a sensor network within the transmission range of each other, this delay is just the propagation time of a packet in the air, which is typically very small. In the Mica hardware platform, the sensor node architecture of Berkeley [8], a system-level optimization for wireless sensor architecture is proposed, which can be used for removing the effect of delay caused by *send time* and *access time* on the synchronization accuracy. If there is a tight coupling between the application and its communication protocol, the MAC layer can inform the application what delay a packet experiences before it is transmitted. This information can even be used to modify the packet once the transmission begins so that the timestamp in the packet reflects the exact time when it was sent [8]. Similarly, if the arrival time can be timestamped at a low enough level at the receiver, the error due to *receive time* can be decreased because it would not include operating system overheads, or the time to transfer the message from the network interface to the host [3].

9.2 NEED FOR SYNCHRONIZATION IN WIRELESS SENSOR NETWORKS

There are several reasons for addressing the synchronization problem in WSNs. First, sensor nodes need to coordinate their operations and collaborate to achieve a complex sensing task. Data fusion is an example of such coordination in which data collected at different nodes are aggregated into a meaningful result. For example, in a vehicle tracking application, sensor nodes report the location and time at which they sense the vehicle to a sink node, which in turn combines this information to estimate the location and velocity of the vehicle. Clearly, if the sensor nodes lack a common timescale (i.e., are not synchronized), the estimate will be inaccurate.

Second, synchronization can be used by power-saving schemes to increase network lifetime. For example, sensors may *sleep* (go into a power-saving mode by turning off their sensors and/or transceivers) at appropriate times and wake up when necessary. When using the power-saving mode, the nodes should sleep and wake up at coordinated times, such that the radio receiver of a node is not turned off when there is some data directed to it. This requires precise timing between sensor nodes.

Scheduling algorithms, for example, TDMA, can be used to share the transmission medium in the time domain to eliminate transmission collisions and conserve energy. Thus, synchronization is an essential part of transmission scheduling.

Traditional synchronization schemes, for example, the network time protocol (NTP) [9] or global positioning system (GPS) [10] are not suitable for use in sensor networks because of complexity and energy issues, cost and size factors. The NTP works well for synchronizing the computers on the Internet, but is not designed with the energy and computation limitations of sensor nodes in mind.

A GPS device may be too expensive to be attached on cheap sensor devices, and GPS service may not be available everywhere (e.g., inside buildings or under water).

9.3 REQUIREMENTS OF SYNCHRONIZATION IN WIRELESS SENSOR NETWORKS

This section presents a broad set of requirements for the synchronization problem. These requirements can also be regarded as the metrics for evaluating synchronization schemes for WSNs. However, there are trade-offs between the requirements on an efficient synchronization solution (e.g., *precision* vs. *energy efficiency*). A single scheme may not satisfy them altogether.

- *Energy Efficiency.* As with all protocols designed for sensor networks, synchronization schemes should take into account the limited energy resources in sensor nodes.
- *Scalability.* Most sensor network applications need deployment of a large number of sensor nodes. A synchronization scheme should scale well with increasing number of nodes and/or high density in the network.
- *Precision.* The need for precision, or accuracy, may vary significantly depending on a specific application and the purpose of synchronization. For some applications, even a simple ordering of events and messages may suffice, whereas for some others the requirement for synchronization accuracy may be on the order of a few microseconds.
- *Robustness.* A sensor network is typically left unattended for a long time of operation in possibly hostile environments. In the case of failure of a few sensor nodes, the synchronization scheme should remain valid and functional for the rest of the network.
- *Lifetime.* The synchronized time among sensor nodes provided by a synchronization algorithm may be instantaneous, or may last as long as the operation time of the network. If the synchronization scheme synchronizes the drifts and removes the offsets, the lifetime for the synchronized time is typically much higher.
- *Scope.* The synchronization scheme may provide a global time base for all nodes in the network or local synchronization only among spatially close nodes. Because of scalability issues, global synchronization is difficult to achieve or too costly (considering energy and bandwidth usage) in large sensor networks. On the other hand, a common time base for a large number of nodes may be needed to aggregate data collected from distant nodes, dictating a global synchronization.
- *Cost and Size.* Wireless sensor nodes are very small and inexpensive devices. Therefore, as noted earlier, attaching relatively large or expensive

hardware (e.g., a GPS receiver) on a small cheap device is not a logical option for synchronizing sensor nodes. A synchronization method for sensor networks should be developed with limited cost and size in mind.

- *Immediacy.* Some sensor network applications, for example, emergency detection (e.g., gas leak detection, intruder detection) require the occurring event to be communicated immediately to the sink node. In such applications, the network cannot tolerate any kind of delay when an emergency is detected. This is called the *immediacy* requirement, and may prevent a protocol designer from relying on excessive processing after such an event of interest occurs.

9.4 SYNCHRONIZATION PROTOCOLS FOR WIRELESS SENSOR NETWORKS

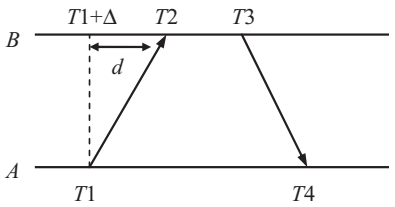
There has been such a significant amount of research on the time synchronization problem in WSNs that it would be impractical to present the details of all here. Instead, we identify three primary tracks for studying synchronization methods, and present some existing protocols as representatives for each track. The first one is the *synchronization primitives*, covering the methods for establishing instantaneous synchronization between neighbor nodes. In a long-lived multihop sensor network, the synchronization primitives are typically used as the building blocks for achieving networkwide and/or long-term synchronization. Hence, the second and third tracks consider the *multihop synchronization* and *long-term synchronization*, respectively. At the end of the section, we summarize and comment on some other protocols and relevant work.

9.4.1 Synchronization Primitives

This section focuses on the methods for providing instantaneous synchronization between the local clocks of neighbor nodes in a sensor network. We classify these methods as synchronization primitives because they are mostly used as the basic building blocks for synchronizing nodes distributed throughout the network.

9.4.1.1 Two-Way Message Exchange. Two-way message exchange between a pair of nodes is the conventional method of synchronizing local clocks in a network, which is employed by NTP for traditional wired networks. This method is also the basic building block of many networkwide synchronization protocols for sensor networks, for example, the timing-sync protocol for sensor networks (TPSN), which we explain in more detail in Section 9.4.2.2. Though many subtleties may exist in the implementation of this method, we present the scheme used by TPSN here.

In order to obtain a definitive relation between the two clocks with a single message exchange, two basic assumptions need to be made.



$T2$,$T3$ are measured by node B clock

$T1$,$T4$ are measured by node A clock

Fig. 9.1 Two-way message exchange between a pair of nodes.

1. The offset between the clocks is constant in the small time period during the message exchange;
2. The propagation delay is the same in both directions.

Consider a two-way message exchange between nodes A and B as shown in Fig. 9.1. Node A initiates the synchronization by sending a packet at $T1$ (according to its local clock), which includes A's current local time $T1$. Node B receives this packet (according to its local clock) at $T2 = T1 + \Delta + d$, where Δ is the relative clock offset between the two nodes, and d is the propagation delay of the pulse. Node B responds at time $T3$ with an acknowledgment packet, which includes the level number of B and the values $T1$, $T2$, and $T3$. Then, node A can calculate the clock offset and propagation delay as below and synchronize itself to B:

$$\Delta = \frac{(T2 - T1) - (T4 - T3)}{2},$$

$$d = \frac{(T2 - T1) + (T4 - T3)}{2}.$$

9.4.1.2 Reference Broadcast Synchronization. Reference broadcast synchronization (RBS), proposed by Elson et al. [3], uses a *third party* for synchronization. Instead of synchronizing the sender with a receiver, this scheme synchronizes a set of receivers with one another. Although its application in sensor networks is novel, the idea of *receiver–receiver synchronization* was previously proposed for synchronization in broadcast environments [11]. In the RBS scheme, nodes send reference beacons to their neighbors. A reference beacon does not include a timestamp. Instead, its time of arrival is used by receiving nodes as a reference point for comparing clocks.

By removing the sender's nondeterminism from the critical path (see Fig. 9.2), RBS may achieve better precision compared to traditional synchronization methods that use two-way message exchanges between synchronized nodes. As the sender's nondeterminism has no effect on RBS precision, the only sources of

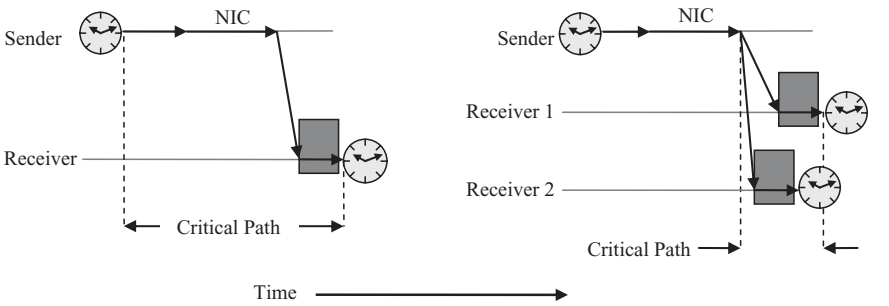


Fig. 9.2 Comparison of traditional synchronization systems to RBS [3].

errors are the nondeterminism in propagation time and receiving time. The authors claim that a single broadcast will propagate to all receivers at essentially the same time; hence the propagation error is negligible. This finding is especially true when the radio ranges are relatively small (compared to the speed of light timed by the required synchronization precision), as is the case for sensor networks [4]. Accordingly, they only account for the receiving time errors when analyzing the accuracy of their model.

In the simplest form of RBS, a node broadcasts a single pulse to two receivers. The receivers, upon receiving the pulse, exchange their receiving times of the pulse and try to estimate their relative offsets. This basic RBS scheme can be extended in two ways:

1. Allowing synchronization between n receivers by a single pulse, where n can be larger than two.
2. Increasing the number of reference pulses to achieve higher precision.

It is shown by simulations that 30 reference broadcasts (for a single synchronization in time) can improve the precision from 11 to 1.6 μs when synchronizing a pair of nodes. This redundancy can also be used for estimating clock skews. Instead of averaging the phase offsets from multiple observations (e.g., each of 30 reference pulses), one can perform a least-squares linear regression to this data. Then the frequency and phase of the local node's clock with respect to the remote node can be recovered from the slope and intercept of the line, which is explained next for the Tiny-Sync protocol.

9.4.1.3 Tiny-Sync and Mini-Sync. Tiny-Sync and Mini-Sync are two lightweight synchronization algorithms proposed by Sichitiu and Veerarittiphan [2]. The authors assume that each clock can be approximated by an oscillator with a fixed frequency. As argued in Section 9.1.1, two clocks, $C_1(t)$ and $C_2(t)$, can be linearly related under this assumption as

$$C_1(t) = a_{12} \cdot C_2(t) + b_{12}, \tag{9.2}$$

where a_{12} is the relative drift, and b_{12} is the relative offset between the two clocks.

These algorithms use a method similar to the conventional two-way messaging scheme, but obtain a relation between clocks in a rather different way. Node 1 sends a probe message to node 2, timestamped with t_o, the local time just before the message is sent. Node 2 generates a timestamp when it gets the message at t_b, and immediately sends back a reply message. The immediate reply assumption can be relaxed without loss of generality, but we skip that case here for brevity. Finally, node 1 generates a timestamp t_r as the time when it gets this reply message. Using the absolute order between these timestamps and Eq. (9.2), the following inequalities can be obtained.

$$t_0 < a_{12} \cdot t_b + b_{12}, \tag{9.3}$$

$$t_r > a_{12} \cdot t_b + b_{12}. \tag{9.4}$$

The 3-tuple of the timestamps (t_o, t_b, t_r) is called a *data point*. Tiny-Sync and Mini-Sync work with some set of data points, each collected by a two-way message exchange as explained. As the number of data points increases, so does the precision of the algorithms. Each data point corresponds to two constraints on the relative drift and the relative offset [Eqs. (9.3) and (9.4)]. The constraints imposed by the data points are depicted in Fig. 9.3. Note that the line corresponding to Eq. (9.2) must lie between the vertical intervals created by each data point. One of the dashed lines in Fig. 9.3 represents the steepest possible such line satisfying Eq. (9.2). This line gives the upper bound for relative drift, $\overline{a_{12}}$ (slope

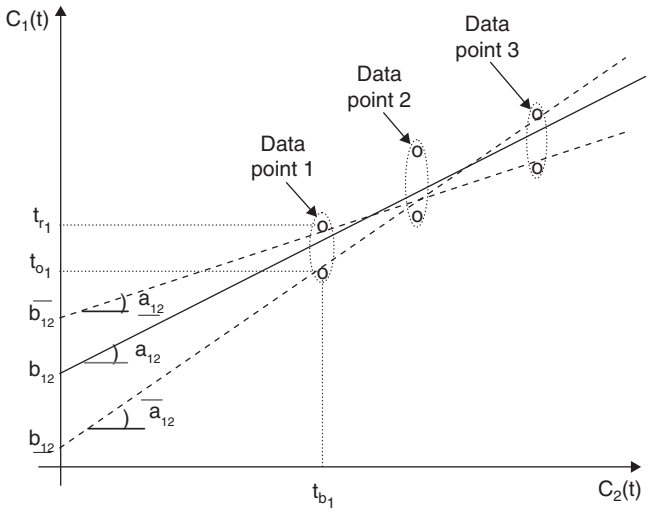


Fig. 9.3 The constraints imposed on a_{12} and b_{12} by data points [2].

of the line) and the lower bound for relative offset, $\underline{b_{12}}$ (y-intercept of the line) between the two clocks. Similarly, the other dashed line gives the lower bound for relative drift $(\underline{a_{12}})$ and the upper bound for relative offset $(\overline{b_{12}})$. Then, the relative drift a_{12} and the relative offset b_{12} can be bounded as

$$\underline{a_{12}} \le a_{12} \le \overline{a_{12}},$$

$$\underline{b_{12}} \le b_{12} \le \overline{b_{12}}.$$

Exact drift and offset values cannot be determined by this method (or any other method as long as message delays are unknown), but can be well estimated by

$$a_{12} = \frac{\overline{a_{12}} + \underline{a_{12}}}{2} \pm \frac{\overline{a_{12}} - \underline{a_{12}}}{2},$$

$$b_{12} = \frac{\overline{b_{12}} + \underline{b_{12}}}{2} \pm \frac{\overline{b_{12}} - \underline{b_{12}}}{2}.$$

The tighter the bounds get, the higher the chance that the estimates will be good (i.e., the precision of synchronization gets higher as the above bounds get tighter). In order to tighten the bounds, one can solve the linear programming problem consisting of the constraints dictated by all data points, in order to get the optimal bounds resulting from the data points. However, the linear programming problem gets larger with the increasing number of data points and this approach is quite complex for sensor networks because it requires high computation and storage for keeping all data points in memory.

The basic intuition behind Tiny-Sync and Mini-Sync algorithms is the observation that not all data points are useful. Consider, for example, the three data points in Fig. 9.3; the intervals $\left[a_{12}, \overline{a_{12}}\right]$ and $\left[b_{12}, \overline{b_{12}}\right]$ are only bounded by data points 1 and 3. Therefore, data point 2 is useless in this example. Following this intuition, Tiny-Sync keeps only the four constraints—the ones that yield the best bounds on the estimates—among all data points. The resulting algorithm is much simpler than solving a linear programming problem. However, this scheme does not always give the optimal solution for the bounds. The algorithm may eliminate some data point, considering the data point useless, although it would actually give a better bound together with another data point that is yet to occur.

Mini-Sync is an extension of Tiny-Sync that finds the optimal solution with an increase in complexity. The idea is to prevent the Tiny-Sync algorithm from eliminating the constraints that may be used by some future data points to give tighter bounds. The authors argue by using experimental results that although suboptimal, the performance of Tiny-Sync is comparable to that of the optimal Mini-Sync.

9.4.2 Multihop Synchronization

A WSN typically spans a much larger area compared to the transmission range of the radio transmitters used; hence, data collected at a sink node may have originated at different nodes that are several hops apart. A common view of the time between such nodes can only be established through multihop synchronization protocols, which we review next.

9.4.2.1 Multihop RBS. Section 9.4.1.2 presented how RBS synchronizes a set of receivers in a single neighborhood. In many cases, the nodes that need synchronized time may not be in the coverage area of some common node. Then, some other nodes should act as gateways for time translation between neighborhoods to *route* the time information from one node to another.

Figure 9.4 depicts a case where multihop synchronization is required. For example, nodes 1 and 7 are not in the same neighborhood; that is, they do not share a common sender from which they can both receive a synchronization pulse. In this case, node 4 acts as a gateway node between the two neighborhoods. When senders A and B broadcast synchronization pulses to their neighborhood as usual, node 4 gets both of these pulses and can thus relate the local clocks of A and B; that is, the two neighborhoods. When a beacon sender broadcasts a synchronization pulse, it essentially creates a set of nodes (a neighborhood) in which nodes can relate their local clocks among each other. Now consider a graph whose vertices correspond to sensor nodes in the network. An edge between two vertices in this graph exists if the corresponding nodes in the network are within the same neighborhood formed by RBS; that is, if the two nodes can receive synchronization pulses from the same beacon sender. Then multihop synchronization can be performed along the edges of this graph. To this end, the concept of "time routing in multihop networks" is introduced. Finding the shortest path between two nodes would yield a minimal error multihop synchronization path for this pair of nodes. Moreover, the authors proposed assigning weights to edges to represent the quality of pairwise synchronizations (e.g., using the residual error of the linear fit).

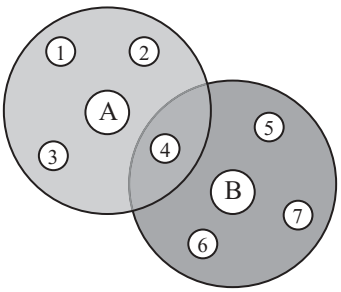


Fig. 9.4 A topology where multihop time synchronization is required [3].

In the analysis of the multihop RBS algorithm, the authors argue that there is just a slow decay in precision by multihop synchronization; the average synchronization error is proportional to $\sqrt{n}$ for an n-hop network. By using the implementation of RBS on IPAQ and 802.11-based testbed, and including the kernel level timestamping, an error of $3.68 \pm 2.57\,\mu$s was measured after four hops.

9.4.2.2 Timing-Sync Protocol. Ganeriwal et al. [5] proposed a networkwide time synchronization protocol for sensor networks, called the timing-sync protocol for sensor networks (TPSN). The protocol works in two phases: *level discovery* and *synchronization*. The aim of the first phase is to create a hierarchical topology in the network, where each node is assigned a level. Only one node is assigned level 0, the *root node*. In the second phase, a node of level *i* synchronizes to a node of level *i-1*. At the end of the synchronization phase, all nodes are synchronized to the root node, and the networkwide synchronization is achieved.

This protocol is like a practical adaptation of NTP [9], where each computer can simultaneously be a server for the computers lower in the hierarchy or a client of the computers higher in the hierarchy. The basic structural difference is that NTP makes use of the existing infrastructure in the Internet, while there is no infrastructure in a sensor network, and such a protocol needs to create a virtual hierarchy before applying the synchronization scheme.

- *Level Discovery Phase.* This phase is run once at the network deployment. First, a node should be determined as the root node. This could be a sink node in the sensor network, and the sink may have a GPS receiver, in which case the algorithm will synchronize all nodes to an external time (time in the physical world). If such a sink is not available, sensor nodes can periodically take over the functionality of the root node. An existing leader election algorithm may be used for this periodic root node election step.

  The root node is assigned level 0, and initiates the level discovery phase by broadcasting a *level_discovery* packet. This packet contains the identity and level of the sender node. Upon receiving this packet, the neighbors of the root node assign themselves level 1. Then each level 1 node broadcasts a *level_discovery* packet with its level and identity in the packet. Once a node is assigned a level, it discards further incoming *level_discovery* packets. This broadcast chain goes on through the network, and the phase is completed when all nodes are assigned a level.
- *Synchronization Phase.* This phase is initiated by the root node's *time_sync* packet. On receiving this packet, level-1 nodes initiate a two-way message exchange with the root, as explained in Section 9.4.1.1. Before initiating the message exchange, each node waits for some random time in order to minimize collisions in the medium access. Once they get back a reply from the root node, they adjust their clocks to the root node. Level-2 nodes, overhearing some level-1 node's communication with the root, initiate a

two-way message exchange with a level-1 node, again after waiting for some random time to ensure that level-1 nodes have completed their synchronization. This procedure eventually gets all nodes synchronized to the root node.

TPSN is implemented on Berkeley's Mica architecture [8] and makes use of timestamping packets at the MAC layer in order to reduce the uncertainty at the sender, as mentioned in Section 9.1.2. Ganeriwal et al. [5] claim that TPSN achieves two times better precision than RBS and that the precision reported for RBS is due to using a superior operating system (Linux) and much more stable crystals available in IPAQs. Thus RBS is implemented on Mica sensor architecture, as well as TPSN, in order to compare their performances. They report an average of 29.13-μs precision for their implementation of RBS on Mica motes, while that of TPSN is 16.9 μs on the same hardware platform. Essentially, it is claimed that uncertainty at the sender contributes very little to the total synchronization error as it is minimized by the use of low-level timestamps at the sender, and therefore the classical sender–receiver synchronization is more effective than receiver–receiver synchronization in sensor networks.

9.4.2.3 Lightweight Tree-Based Synchronization. Lightweight tree-based synchronization (LTS), proposed by Greunen and Rabaey [7], is distinguished from other work in the sense that the aim is not to maximize accuracy, but to minimize the complexity of the synchronization. Thus, the required synchronization accuracy is assumed to be given as a constraint and the target is to devise a synchronization algorithm with minimal complexity to achieve the given precision. This approach is supported by the claim of the authors that the maximum time accuracy required in sensor networks is relatively low (within fractions of a second). Therefore, it is sufficient to use a relaxed or lightweight synchronization scheme in sensor networks. Clearly, this assumption may not hold for some applications or services of sensor networks, for example, measuring the time-of-flight of sound [12], forming a TDMA schedule [13], and distributing an acoustic beamforming array [14], which require synchronized time with high precision. However, a loose synchronization may be acceptable in most other cases within the wide range of applications projected for WSNs.

Two LTS algorithms are proposed for multihop synchronization of the network based on the pairwise synchronization scheme described in Section 9.4.1.1. Both algorithms require nodes to synchronize to some *reference node(s)*, for example, a sink node in the sensor network. The first algorithm is centralized and needs a spanning tree to be constructed first. Then pairwise synchronization is done along the $n - 1$ edges of the spanning tree. In the centralized algorithm, the reference node is the root of the spanning tree and has the responsibility of initiating a *resynchronization* as needed. Using the assumption that the clock drifts are bounded and given the required precision, the reference node calculates the time period which a single synchronization step will be valid for. Since the depth of the spanning tree affects the time to synchronize the whole network, as

well as the precision error at the leaf nodes, the depth of the tree is communicated back to the root node so that it can use this information in its resynchronization time decision.

The second multihop LTS algorithm performs networkwide synchronization in a distributed fashion. Each node decides the time for its own synchronization and a spanning tree structure is not used in this algorithm. When node i decides that it needs to synchronize (using the desired accuracy, its distance from the reference node, and the clock drift), it sends a synchronization request to the closest reference node (by any routing mechanism available). Then all nodes along the path from that reference node to node i must be synchronized before node i can be synchronized. The advantage of this scheme is that some nodes may have less frequent events to deliver, and therefore may not need frequent synchronization. Since nodes have the opportunity to decide on their own synchronization, this saves unnecessary synchronization effort for such nodes. On the other hand, letting each node decide on resynchronization may boost the number of pairwise synchronizations because for each synchronization request all nodes along the path from the reference node to the resynchronization initiator need to be synchronized. As the number of synchronization requests increase, the overall effect of synchronizations along these paths may be a significant waste of resources. Hence, the idea of aggregating synchronization requests is proposed; when any node wishes to request synchronization, it queries adjacent nodes to discover the existence of any pending request. If any exists, the synchronization request of this node could be aggregated to a pending request, decreasing the inefficiency that would be caused by two separate synchronizations along the same path.

The performance of the LTS algorithms are tested by the simulations of a connected ad hoc network consisting of 500 nodes, placed uniformly at random in a 120 m × 120 m rectangular area. The transmission range is set to 10 m. It is assumed that there is a single reference node at the center of the area, which has access to an accurate time. All nodes should synchronize to this reference node. The required accuracy is determined as 0.5 s, and the simulation is executed for 10 h. As a metric to evaluate the performance of the synchronization algorithms, the number of pairwise synchronizations required to keep the network synchronized is analyzed. The average number of synchronizations required for each node is 36 for the centralized LTS over 10 h of simulation time. If the number of participating nodes (the nodes that need synchronized time and thus participate in the algorithm) is low, the distributed algorithm performs much better; for 65% participation, the number of synchronizations per node drops to around four to five synchronizations when the distributed LTS is used. Another metric is the average depth of the spanning tree and an average of five to seven is reported for both algorithms.

9.4.2.4 Flooding Time Synchronization Protocol. The flooding time synchronization protocol (FTSP) [15] utilizes and enhances some key ideas from TPSN and RBS, and combines them with periodic flooding of synchronization messages to achieve networkwide synchronization, which is robust against node

and link failures. FTSP implements MAC layer timestamping as in TPSN and drift compensation with linear regression as in RBS.

A critical performance enhancement of FTSP over prior work is due to its focus on the detailed analysis of the transceiver pipeline in the wireless channel. FTSP uses a single message per synchronization and introduces the use of multiple timestamps for a single synchronization message. Timestamps are made at each byte boundary as they are transmitted or received and the final timestamp on the message is computed by an average of the normalized timestamps. This effectively reduces the jitter of the interrupt handling and encoding/decoding times through the CPU (central processing unit), radio, and antenna of the sender and the receiver. Though the achievable error correction using this technique is bounded by the number of bytes, an experiment on the Mica2 platform reports roughly a 10-fold improvement in the precision with only six timestamps.

In FTSP, all nodes in the network synchronize to a dynamically (re)elected *root* node by the use of controlled flooding. Similar to the *data points* in Tiny-Sync described in Section 9.4.1.3, nodes use *reference points* for synchronization, each of which is a pair of local and "global" timestamps. A reference point is collected by receiving a synchronization message from the *root* or another node that is previously synchronized to the root. When a node gathers enough reference points, it performs synchronization by estimating its clock drift and offset using linear regression, after which it can also start broadcasting synchronization messages.

A synchronization message contains three fields: *timeStamp, rootID*, and *seqNum*. The *timeStamp* is the synchronized "global" time as estimated by the sender of this synchronization message. The *rootID* field contains the unique ID of the root node as currently known to the sender of this message. The *seqNum* is used to control the flooding of messages and is incremented at every synchronization round initiated by the root node. A node uses only the first message arrived for each *rootID* and *seqNo* pair. When a node does not receive synchronization messages for a certain duration of time, it declares itself as the root. In order to eliminate the problem of having multiple roots, a node that receives a message with smaller *rootID* gives up its root status; hence, only the node with the smallest ID remains as the single root.

The experiments with an FTSP implementation on Mica2 motes report an average synchronization error of 3 μs in a 6-hop network, resulting in a 0.5-μs per hop accuracy, which is evidently better than that of RBS and TPSN. FTSP is also reported to use less network resources than the other two protocols; if the resynchronization period is T seconds, then each node sends 1 message per T seconds in FTSP, 2 messages per T seconds in TPSN (1 message to parent and 1 response) and 1.5 message per T seconds in RBS (0.5 for a reference broadcast and 1 for a timestamp exchange message).

9.4.3 Long-Term Synchronization

The synchronization protocols presented so far mainly aim to provide a common timescale between clocks at a given instant. However, as argued earlier, the

achieved harmony of the clocks may be quickly disrupted by varying clock drifts. The most straightforward method for achieving time synchronization over long durations (e.g., the lifetime of a sensor network) is the periodic application of one of those schemes that provide instantaneous synchronization. As a better alternative, adaptive schemes carefully designed for long-term synchronization have been proposed for better use of limited resources and/or higher precision in WSNs.

9.4.3.1 Post-facto Synchronization. Postfacto synchronization was a pioneering work by Elson and Estrin [16,17], which has led afterward to their RBS scheme. They proposed that unlike in traditional synchronization schemes, for example, NTP [9], local clocks of the sensor nodes should normally run unsynchronized in their own pace and should synchronize only when necessary. This way, local timestamps of two nodes at the occurrence time of an event are synchronized later by extrapolating backwards to estimate the offset between clocks at a previous time (at the time of the event). Postfacto synchronization can also be termed as *reactive synchronization*, while the traditional schemes are *proactive*, requiring the clocks of sensor nodes to be synchronized before an event of interest occurs.

9.4.3.2 Time-Diffusion Synchronization Protocol. The time-diffusion protocol (TDP) [18] is a networkwide synchronization protocol that maintains an equilibrium time throughout the network, allowing only a small deviation from the equilibrium. The deviation tolerance can be adjusted based on the specific sensor network application.

TDP achieves long-term synchronization by defining *active* and *inactive* periods (see Fig. 9.5). At every δ seconds during the active period, some nodes are elected as *master* nodes that broadcast timing information to their neighbors

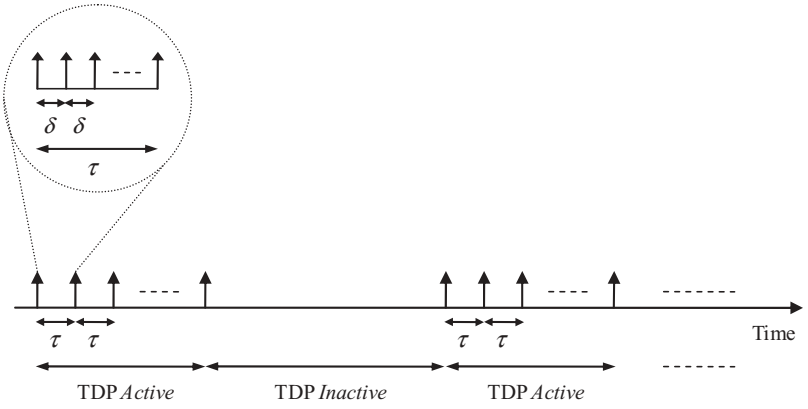


Fig. 9.5 Duty cycle of TDP with its active–inactive schedule [18].

at every μ seconds. Nodes receiving timing information from the master nodes self-determine to become *diffused leader* nodes that further broadcast the timing information to their neighbors. Neighbors of diffused leaders may also become diffused leader nodes, spreading the timing information further away from the current master nodes. Therefore, in effect tree-like structures are temporarily created for diffusing the time information from the master nodes to the rest of the network.

The distributed autonomous diffusion process combined with the periodic reelection of master nodes provides networkwide synchronization with tunable parameters τ and δ, which together determine the length of the TDP active period. The appropriate duration of the active period depends on the desired synchronization accuracy throughout the network, whereas the duration of the inactive period is dictated by how much the clocks are allowed to drift from each other in the worst case.

9.4.3.3 Rate Adaptive Time Synchronization. Rate adaptive time synchronization (RATS) by Ganeriwal et al. [19] is an energy-efficient long-term synchronization protocol that can adapt to variable clock drifts while achieving the precision specified by the application. The design of RATS is based on an in-depth analysis of empirical measurements to investigate the interplay between three key parameters affecting the long-term synchronization, including the synchronization rate, the history of synchronization data, and the estimation scheme.

The objective of the RATS protocol is to dynamically implement the control loop illustrated in Fig. 9.6. The sample repository in the figure represents the synchronization data points (observations) collected by a node in relation to another node's local clock. A window of these samples is input to an estimator that uses the observations to estimate the relative model between the two clocks. A new sampling period is then calculated based on the comparison of the prediction error of this model to the application-specific error bound. A detailed analysis of the long-term empirical measurements guides the choice or learning of RATS parameters such as the optimal window size.

The experiments with a RATS implementation on Mica2 motes show that for an error bound of 225 microseconds the average sampling period is roughly 30 min. The main use proposed for the RATS protocol is more efficient duty cycling in sensor networks. Hence, it is integrated with a MAC layer protocol, B-MAC [20], which does not assume any time synchronization but instead uses

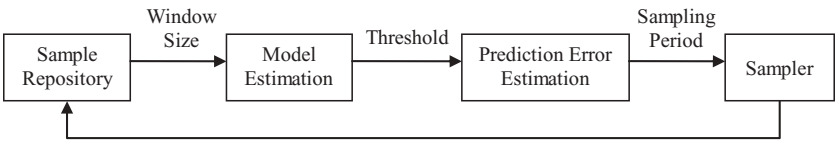


Fig. 9.6 Time synchronization control loop for RATS.

a very long preamble to guarantee that the receiver wakes up before the actual data transmission. The integration of RATS in B-MAC enables the use of shorter preambles and longer duty cycles, which is reported to provide an order of magnitude reduction in energy consumption of a node with a negligible impact on the packet loss rate.

9.4.4 Other Protocols and Relevant Work

Younis and Fahmy [21] proposed a pair of distributed protocols, SYNC-IN and SYNC-NET, for synchronization of clustered sensor networks. The goal is to provide an end-to-end synchronization between communicating nodes, rather than a global timescale throughout the network. It is assumed that the network is clustered using any clustering approach and that nodes can tune their transmission power. The SYNC-IN protocol is used for intracluster synchronization using the smaller in-cluster transmission power, where the nodes are synchronized to the cluster heads. The cluster heads are synchronized with the SYNC-NET protocol using a higher transmission power.

The asynchronous diffusion protocol proposed by Li and Rus [22] provides a simple approach for synchronization. Though diffusion based as in TDP, it is completely asynchronous; nodes average time readings obtained from their neighbors and then broadcast the computed value as their updated clock reading. One drawback of this method is that a clock may run backwards after an update, causing the same time reading to occur more than once.

In Ref. [6], a message ordering scheme for sensor networks is proposed. The intention is not to synchronize clocks, but to reason about the relative order between messages or events. The scheme described in this work complies with the most relaxed version of synchronization and is not applicable to most synchronization needs in WSNs.

In a study that is more of theoretical interest [23], the authors considered an infinitely large sensor network and proposed an approach in which nodes collaborate to generate a waveform that carries enough synchronization information to all nodes in the network. They argued that as the number of nodes goes to infinity, optimal synchronization is possible at a reasonable complexity.

A CENS (Center for Embedded Networked Sensing—University of California, Los Angeles) technical report also presents a study on optimal and global time synchronization in sensor networks [4]. They considered the problem of finding the best path (chain) of pairwise synchronizations that would yield the optimum synchronization between any pair of nodes in the network. They claimed that an appropriately weighted combination of alternating paths for synchronization should yield better precision than any single path could provide. By the use of such weighted combination of paths, the optimal global synchronization problem can be abstracted as a network flow formulation. In this work, the authors did not aim at giving a practical synchronization method, but presented the theoretical results for optimal global synchronization, which can be used as a reference to compare the performance of global synchronization methods.

The reachback firefly algorithm (RFA) [24] is a distributed synchronicity algorithm inspired by the Mirollo–Strogatz (MS) mathematical model [25], which was previously proposed for explaining how neurons and fireflies spontaneously synchronize. The main goal of RFA is not time synchronization but synchronicity (defined as the ability for all nodes in the network to agree on a common period and phase for firing pulses). However, synchronicity can be used as a primitive to obtain time synchronization. Though it presents a new approach for time synchronization, RFA has significant overhead and its performance is not yet evaluated as a time synchronization protocol.

Another biologically inspired algorithm DESYNC [26] unconventionally uses the MS model for *desynchronization*, a primitive introduced by the authors as the logical opposite of synchronization. Instead of performing periodic tasks at the same time, the nodes try to schedule tasks so that they are as far away from each other as possible. In other words, the firing events are evenly distributed in a given time frame, rather than having them coincide at the same instant. This primitive can be used for many sensor networking tasks, such as periodic resource sharing, distributed collaborative sensing, and channel access scheduling.

The adaptive-rate synchronization protocol (ARSP) [27] addresses not only adjusting the local clock values at nodes, but also the synchronization intervals to ensure that the synchronization errors remain within a given tolerance with high probability. Motivated by the energy limitations and various precision needs of sensor network applications, ARSP aims to provide a tunable tool for different scenarios. It utilizes both two-way message exchange and receiver–receiver synchronization primitives, while the synchronization intervals are adjusted at run time as a function of the intolerance probability of the various nodes.

Reference [28] gives an overview of the time synchronization problem in WSNs, and defines the requirements and various issues for designing a synchronization algorithm for WSNs. The authors argue that such an algorithm should be multimodal, tiered and tunable so that it can satisfy the diverse needs of various sensor network applications. Moreover, they suggest that the local clock of each node be free-running; that is, one should not adjust the local clock. Instead, the synchronization scheme should build up a table of parameters that enables each node to convert its local clock to that of another, and vice versa.

9.5 SUMMARY AND FUTURE DIRECTIONS

This chapter introduced the synchronization problem and common challenges for synchronization, discussed the need for synchronization and requirements of synchronization methods in WSNs, and reviewed the major synchronization methods and protocols for WSNs. The two synchronization protocols, RBS and TPSN, both report very high precisions on the orders of a few microseconds although they use completely different approaches. The receiver–receiver synchronization of RBS completely eliminates the uncertainty at the sender by using a third party node, while TPSN minimizes this uncertainty by low-level

timestamping at the sender. On the other hand, the receiver–receiver synchronization requires four messages sent and three messages received for synchronizing two nodes, while the sender–receiver synchronization requires only two sent and two received messages. As radio communication is known to be the most energy consuming component of sensor node operations, this is almost a two times increase in energy complexity. This increase in the complexity of receiver–receiver synchronization can be reduced to some degree by synchronizing many receivers by a single synchronization pulse broadcast by the sender. Although TPSN does not suffer from energy complexity in this respect, it needs a hierarchical structure of nodes to be formed, which may increase the synchronization cost. FTSP combines and enhances the key ideas of TPSN and RBS and has superior performance compared to both, with smaller communication overhead. The LTS algorithms offer very low-cost synchronization, however, with very limited accuracy and thus limited applicability.

Studying the long-term behavior of synchronization schemes for WSNs is a rather less explored area. Most protocols suggest periodic reapplication of instantaneous synchronization methods, which can be costly for resource-constrained sensor networks. We have presented three different approaches for efficiently maintaining synchronized time in the network for longer time periods. Before concluding this chapter, we review some open issues and possible research directions in this field.

Most of the time synchronization work in the literature analyzes and presents their results based on experiments or simulations. For single-hop synchronization, there are also *analytical models* to define the accuracy characteristics of a proposed synchronization scheme. However, there is a lack of analytical models for multihop synchronization. When two nodes apart are synchronized using multiple pairwise synchronization steps, errors are usually expected to grow. However, since the pairwise errors may have different signs and magnitudes, the overall effect of multihop synchronization is usually much smaller than the sum of magnitudes of single-hop errors. An analytical model for this artifact may be developed, accounting for the probabilistic variations in the sign and magnitude of single-hop synchronization errors.

Identification or discovery of nodes to act as beacon senders in RBS is an important issue. If there is more than one beacon sender in a single neighborhood, the resulting redundancy may be used to improve precision, but also increase the consumption of limited resources in the network. The correlation between this redundancy and precision may be investigated, and methods for identifying beacon senders to achieve some desired point in this trade-off curve proposed.

Extensive research on sensor networks boosts the evolution of these systems. Although sensor networks are mostly considered as having fixed topologies (with stationary sensor nodes), and sensor network protocols so far usually assume that the nodes are stationary, next generation sensor networks may be expected to include mobile sensor nodes. Indeed, the Networked Infomechanical Systems (NIMS) project is a recent initiative toward this, and has already announced the

development and deployment of initial prototypes that operated in a forest field biology station (More information is available online at http://research.cens.ucla.edu/research/). As such systems evolve, synchronization methods that take mobility into account will be needed. Global synchronization protocols may even benefit from the mobility because mobile nodes will "carry" time information from one part of the network to other parts, potentially increasing global synchronization accuracy.

REFERENCES

[1] K. Römer, "Time synchronization in ad hoc networks", in *Proceedings of 2001 ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'01)*, Long Beach, CA, Oct. 2001, pp. 173–182.

[2] M. L. Sichitiu and C. Veerarittiphan, "Simple, accurate time synchronization for wireless sensor networks", in *Proceedings of 2003 IEEE Wireless Communications and Networking Conference (WCNC'03)*, New Orleans, LA, Mar. 2003, pp. 1266–1273.

[3] J. Elson, L. Girod, and D. Estrin, "Fine-grained time synchronization using reference broadcasts", in *Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI'02)*, Boston, MA, Dec. 2002, pp. 147–163.

[4] R. Karp, J. Elson, D. Estrin, and S. Shenker, "Optimal and global time synchronization in sensornets", Technical Report 0009, Center for Embedded Networked Sensing, University of California, Los Angeles, CA, Apr. 2003.

[5] S. Ganeriwal, R. Kumar, and M. Srivastava, "Timing sync protocol for sensor networks", in *Proceedings of ACM SenSys'09*, Los Angeles, , Nov. 2003, pp. 138–149.

[6] K. Römer, "Temporal message ordering in wireless sensor networks", in *Proceedings of IFIP MedHocNet'03*, Mahdia, Tunisia, Jun. 2003, pp. 131–142.

[7] J. V. Greunen and J. Rabaey, "Lightweight time synchronization for sensor networks", in *Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications (WSNA'03)*, San Diego, CA, Sept. 2003, pp. 11–19.

[8] J. Hill and D. Culler, "A wireless embedded sensor architecture for system-level optimization", Technical Report, U.C. Berkeley, May 2001.

[9] D. Mills, "Network time protocol (version 3) specification, implementation and analysis", RFC 1305, available at http://www.faqs.org/ftp/rfc/rfc1305.pdf, Mar. 1992.

[10] E. D. Kaplan, *Understanding GPS: Principles and Applications*, Artech House, Norwood, MA, Feb. 1996.

[11] P. Verissimo and L. Rodrigues, "A posteriori agreement for fault-tolerant clock synchronization on broadcast networks", in *Proceedings of the 22th International Symposium on Fault-Tolerant Computing*, Boston, July 1992, pp. 527–536.

[12] L. Girod and D. Estrin, "Robust range estimation using acoustic and multimodal sensing", in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'01)*, Maui, HI, Mar. 2001, pp. 1312–1320.

[13] G. Asada, M. Dong, T. S. Lin, F. Newberg, G. Pottie, W. J. Kaiser, and H. O. Marcy, "Wireless integrated network sensors: low power systems on a chip", in *Proceedings of the European Solid State Circuits Conference*, Den Hague, The Netherlands, Sept. 1998, pp. 9–16.

[14] H. Wang, L. Yip, D. Maniezzo, J. C. Chen, R. E. Hudson, J. Elson, and K. Yao, "A wireless time-synchronized COTS sensor platform part II–applications to beamforming", in *Proceedings of IEEE CAS Workshop on Wireless Communications and Networking*, Pasadena, CA, Sept. 2002.

[15] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, "The flooding time synchronization protocol", in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys'04)*, Baltimore, , Nov. 2004, pp. 39–49.

[16] J. Elson and D. Estrin, "Time synchronization for wireless sensor networks", in *Proceedings of IEEE IPDPS Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing*, San Francisco, CA, Apr. 2001, pp. 1965–1970.

[17] J. Elson, *Time Synchronization in Sensor Networks*, PhD Thesis, University of California Los Angeles, 2003.

[18] W. Su and I. F. Akyildiz, "Time-diffusion synchronization protocol for wireless sensor networks", *IEEE/ACM Transactions on Networking*, vol. 13, no. 2, Apr. 2005, pp. 384–398.

[19] S. Ganeriwal, D. Ganesan, H. Shim, V. Tsiatsis, and M. B. Srivastava, "Estimating clock uncertainty for efficient duty-cycling in sensor networks", in *Proceedings of the 3rd ACM SenSys Conference*, San Diego, CA, Nov. 2005, pp. 130–141.

[20] J. Polastre, J. Hill, and D. Culler, "Versatile low power medium access for wireless sensor networks", in *Proceedings of the ACM Conference on Embedded Networked Sensor Systems (SenSys'04)*, Baltimore, MD, Nov. 2004, pp. 95–107.

[21] O. Younis and S. Fahmy, "A scalable framework for distributed time synchronization in multi-hop sensor networks", in *Proceedings of 2005 IEEE Sensor and Ad Hoc Communications and Networks (SECON'05)*, Santa Clara, CA, Sept. 2005, pp. 13–23.

[22] Q. Li and D. Rus, "Global clock synchronization in sensor networks", *IEEE Transactions on Computers*, vol. 55, no. 2, Feb. 2006, pp. 214–226.

[23] A. Hu and S. D. Servetto, "Asymptotically optimal time synchronization in dense sensor networks", in *Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications (WSNA'03)*, San Diego, CA, Sept. 2003, pp. 1–10.

[24] R. Nagpal, A. Patel, G. Tewari, M. Welsh, and G. Werner-Allen, "Firefly-inspired sensor network synchronicity with realistic radio effects", in *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, San Diego, CA, Nov. 2005, pp. 142–153.

[25] R. Mirollo and S. Strogatz, "Synchronization of pulse-coupled biological oscillators", *SIAM*, vol. 50, no. 6, Dec. 1990, pp. 1645–1662.

[26] J. Degesys, I. Rose, A. Patel, and R. Nagpal, "DESYNC: Self-organizing desynchronization and TDMA on wireless sensor networks", in *Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN'07)*, Cambridge, MA, Apr. 2007, pp. 11–20.

[27] D. Macii and D. Petri, "An adaptive-rate time synchronization protocol for wireless sensor networks", in *Proceedings of the IEEE Instrumentation and Measurement Technology Conference (IMTC'07)*, Warsaw, Poland, May 2007, pp. 1–6.

[28] J. Elson, and K. Römer, "Wireless sensor networks: A new regime for time synchronization", in *Proceedings of the 1st Workshop on Hot Topics in Networks (HotNets-I)*, Princeton, NJ, Oct. 2002, pp. 149–154.

10

ENERGY EFFICIENCY AND POWER CONTROL

Nikolaos A. Pantazis

Technological Educational Institute of Athens, Greece

Dimitrios D. Vergados

University of Piraeus, Greece

10.1 INTRODUCTION

Wireless sensor networks (WSNs) have received tremendous attention in recent years because of the development of sensor devices, as well as wireless communication technologies. WSNs make it easier to monitor and control physical environments from remote locations and present many significant advantages over wired sensor networks for a variety of civilian and military applications [1,2]. A WSN is usually randomly deployed in inaccessible terrains, disaster areas, or polluted environments, where battery replacement or recharge is difficult or even impossible to be performed. For this reason, network lifetime is of crucial importance to a WSN. To prolong network lifetime, there is a need for efficient power control mechanisms to reduce power consumption in sensor nodes and energy-efficient techniques should be employed at all layers of the network [3–5], which should take into account the following unique characteristics and application requirements of WSNs [1]:

Wireless Sensor Networks: A Networking Perspective, Edited by Jun Zheng and Abbas Jamalipour

- The topology of a WSN changes frequently.
- Sensor nodes are densely deployed in a sensed field.
- Sensor nodes mainly use broadcast communication, whereas most wireless ad hoc networks are based on point-to-point communications.
- Sensor nodes may not have global identification due to the large amount of overhead introduced and their large number.
- Sensor nodes are limited in power, computational capacity, and memory.

For this purpose, a lot of research has been conducted and a variety of power conservation mechanisms have been proposed in the literature.

This chapter is dedicated to the energy efficiency and power control issues in WSNs. Section 10.2 discusses the need for energy efficiency and power control, introduces the major issues and challenges in designing efficient power conservation mechanisms, and presents a classification of power conservation mechanisms for WSNs. Sections 10.3 and 10.4 give an overview of major passive and active power conservation mechanisms for WSNs, respectively. Section 10.5 summarizes the chapter with a brief discussion of future directions.

10.2 NEED FOR ENERGY EFFICIENCY AND POWER CONTROL IN WIRELESS SENSOR NETWORKS

A question that naturally arises is *Why are energy efficiency and power control needed and important in WSNs*? The answer is simple. In a WSN, sensor nodes are typically operated by batteries, which are limited in energy capacity, and difficult or even impossible to be replaced or recharged. For this reason, power control is needed to efficiently make use of the limited energy resources in order to minimize the energy consumed by the sensor nodes and thus prolong network lifetime. For this purpose, energy efficiency must be considered in every aspect of network design and operation, not only for individual sensor nodes, but also for the communication of the entire network. Energy efficiency and power control are the basic guarantee of the network performance, for example, throughput and delay.

10.2.1 Power Consumption in Sensor Nodes

A sensor node typically consists of four basic components: a sensing unit, a processing unit, a transceiver unit, and a power supply unit [1,6], as shown in Fig. 10.1. The *sensing unit* consists of several sensing devices (i.e., sensors) and/or actuators that ensure the link between the sensor node and its outside world (e.g., the other sensor nodes, gateways, and base station in the network). Energy saving in a sensor node can be accomplished through the use of low-power consuming electronic components at the cost of under-performance, which is necessary.

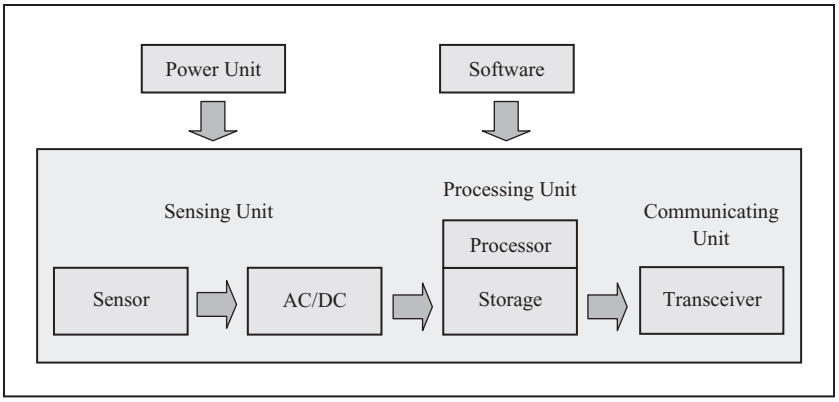


Fig. 10.1 Architecture of a sensor node.

The *processing unit* incorporates a microprocessor responsible for the overall control of the sensor node, including processing the received information and ensuring the communication with the other nodes in the network. Instead of sending original data to the sink or base station for processing, a sensor node makes use of its processing ability to locally perform simple computation and transmit only the required and partially processed data to the sink for further processing [5,7–9]. A microprocessor is usually constructed in such a way that it can operate under different modes (e.g., a sensor node must reduce its activity when its battery runs low) to save power and thus improve energy efficiency and prolong network lifetime. However, the changeover between different operation modes would result in considerable consumption of power. Therefore, energy consumption in different operation modes should be carefully taken into account while examining the battery lifetime of each node in the network.

The energy that is consumed by a microprocessor depends on two important factors: operating voltage and operating frequency. Dynamic voltage scheduling (DVS) is one of the common energy-saving mechanisms, which takes both factors into account and will be described in Section 10.4. Recent research has demonstrated that the energy consumed during the data processing process is much less than that consumed during the communication process [1,5]. Therefore, it has been a focus to optimize data processing such that less and shorter packets are transmitted during the communication process and the duration of the communication between the sensor nodes is thus minimized.

The *transceiver unit* comprises a short-range transceiver (radio), whose functionality is to maintain uninterrupted communication with the other sensor nodes in the network. The limited battery energy imposes a limit on the transmission range of a transceiver, thus requiring low-power multihop transmission mechanisms. This finding also means that sensor nodes should be deployed very closely to each other in order to communicate with the least possible power. There are four modes of transceiver operation: transmit, receive, idle, and sleep.

A number of experiments have demonstrated that it saves a large amount of energy if the transceiver is turned off rather than staying in the idle mode when it is not transmitting or receiving data. Although this power-saving operation mode, at a first glance, seems to be the most energy-efficient one, the fact that sensor nodes usually use short data packets in their communication should not be overlooked. In sensor nodes, energy is consumed during the transition from the switch-on (wake up) state to the switch-off (sleep) state, and vice versa. If the radio is turned off during each idle slot for a certain period of time and then turned on, it would result in more energy consumption than if the radio is left on all the time because the state transition causes energy consumption. The shorter the packets are, the more frequent the on–off switching is, and the more start-up energy is consumed. Therefore, the power-saving operation mode is energy efficient only if the time a sensor stays in that mode is greater than a certain threshold [1]. On the other hand, the selection of the modulation/demodulation technique, filtering technique, and frequency band is equally important for minimizing the energy consumed by the sensor node. The usual modulation techniques used in WSNs include OOK (on–off keying), ASK (amplitude shift keying), and FSK (frequency shift keying). The selection of the frequency band for the wireless links concerns not only the size of the employed antenna, but the energy saving as well. For optimized signal transmission, an antenna of $\lambda/4$ is recommended, where λ is the carrier wavelength. Note that the use of a higher frequency implies the consumption of more energy [1]. The industrial, scientific, and medicine (ISM) frequency band is widely used in WSNs because it is free of charge, and disposes an extended frequency spectrum and global availability. In addition, the use of ultra-wideband 3.1–10.6 GHz is also remarkable, and is considered as one of the best solutions to minimizing power consumption and maximizing energy savings. Since power consumption for radio communication dominates the power consumption of the entire sensor node, power control is of critical importance and has a great impact on the battery lifetime of the sensor node.

The *power unit* comprises a battery indispensable for normal power supply for the basic components of the sensor node. It is important to persistently monitor the power consumption of a battery because if a high current lasts for a long time, the battery will be depleted very quickly. Sometimes the minimum energy required for achieving the performance of the sensor node may be less than the operating power of the battery being used, which would lead to a shorter battery lifetime. There are two possible ways to increase the battery lifetime: (1) by drastically reducing the battery current through elaborate data processing techniques; (2) by occasionally turning off the sensor node when there is no processing or communicating demand.

Sensor nodes are usually deployed in a field of interest in a distributed manner, as shown in Fig. 10.2. Each sensor node should be in a position of sensing its environment, collecting and transmitting data to the destination (sink or base station). Moreover, data can be sent back to the sink through a more-than-one-hop path with no infrastructure [1].

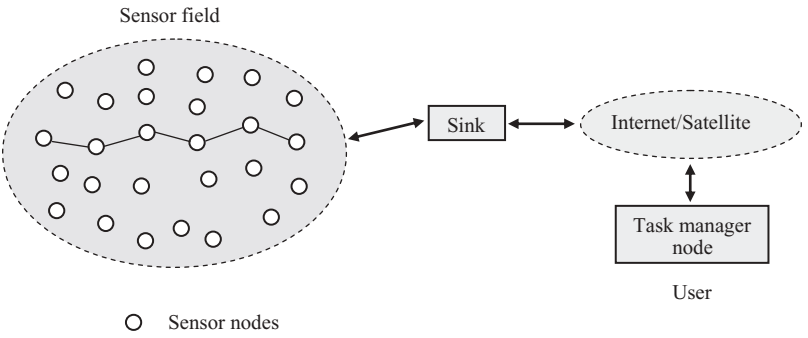


Fig. 10.2 Sensor nodes deployed in a sensor field.

10.2.2 Power Control at Different Protocol Layers

Power control involves several aspects that should be considered in the design and operation of a WSN. The lifetime of a network can be prolonged by improving energy efficiency at different protocol layers [e.g., physical, data link, medicine access control (MAC), network, and transport] and in the operating system as well. At the MAC layer, for example, the power-saving operation mode in sensor nodes should be considered no matter which type of MAC mechanisms is employed. At the network layer, the most energy-efficient routing algorithm should be used, which takes into account either the residual energy in a sensor node or the least energy-consuming path from a sensor node to the base station. At the transport layer, probing can alter TCP's (transmission control protocol's) retransmission behavior by minimizing unnecessary retransmissions, thus achieving lower power consumption and higher throughput [10].

Another aspect that should be considered is that in a multihop network each sensor node plays a dual role of data origination and relay. When a node acts as a relay node, it means that most of the control and data packets received by the node are not destined for it and thus should be forwarded. There exists advanced hardware that is able to identify and forward the packets destined for other sensor nodes. This can avoid unnecessary computing that would otherwise take place in an intermediate node. Moreover, the power depletion of even a very few sensor nodes may cause a major change in the network topology. As a result, it is unavoidable to reroute the packets and reconfigure the network [1]. Therefore, power control is of additional significance in WSNs.

In power control, the determination of the power level for transmitting a data packet is very important. It is a composite problem that is not as simple as it looks like at first glance because it affects many aspects of the network operation. The transmission power level affects the following elements:

- The quality of the signal received at the sink (or a receiver), thus affecting the physical layer because the transmission affects the physical layer components.

- The range of the transmission, thus affecting the network layer because the transmission range affects routing and path selection.
- The magnitude of the interference it produces, thus affecting the transport layer because interference causes congestion of the channel and further of the network.

In general, power control exerts a composite effect on the whole system performance, including the following aspects:

- The MAC layer performance of the network because the contention for the common transmission medium depends on the number of nodes within the transmission range, as well as the number of hops and thus the end-to-end delay;
- The throughput capacity of the network because a larger transmission power results in a larger transmission range, causing a larger topology that may either have a larger throughput due to the increased number of alternative paths or a smaller throughput due to the higher level of contention.
- The connectivity of the network and thus the ability to deliver a packet to its destination because a larger transmission power makes a dense topology more fault-tolerant and less sensitive to sensor failures than a sparse one.

Transmission power is an important metric for measuring energy consumption. A change of the power level may result in a change of route selection, and accordingly change the paths and network connectivity. It can create a unidirectional link if the power level of a node is high enough for another node to hear it, but not vice versa. In many routing protocols, bidirectionality is a fundamental assumption on the links. The MAC protocols, for example, IEEE 802.11, are also based upon the bidirectionality assumption. Many protocols, for example, Ad Hoc On-Demand Distance Vector (AODV) and Dynamic Source Routing (DSR), reverse a route followed by the *Route Request* packets [11]. Therefore, power control is a cross-layer design problem, which affects all layers of the protocol stack, from the physical layer to the transport layer, and thus has a great impact on several key performance metrics, including throughput, delay, and energy consumption.

In the design of a WSN, there are several principles for power control that should be followed:

- The transmission power level is determined in such a way that the network connectivity is guaranteed and at the same time the probability of collision is minimized, taking into account the network topology.
- The reduction of the transmission power level implies the reduction of the average contention at the MAC layer.
- The impact of power control on total energy consumption depends on the hardware energy consumption pattern, including the power consumed in

the processing unit for data processing, the power consumed by the power amplifier for data transmission, the power consumed when the radio is on, but no signal is being received (P_{idle}), and the power consumed when the radio is turned off (P_{sleep}). Since P_{sleep} is $\leq P_{idle}$, it is a significant energy saving strategy to turn off the radio whenever possible. Moreover, under a high traffic load, a lower power level results in a lower end-to-end delay. Under a low traffic load, a higher power level also results in a lower end-to-end delay [1].

- Power control can basically be considered as both a network layer and a MAC layer problem.

To maximize network lifetime, another important issue is how to distribute the traffic load in a network in the most efficient way. If the traffic always goes through the same path to the sink, the energy of those nodes on the path will be depleted quickly, which would disrupt the network connectivity and thus the normal network operation. For this reason, a more uniform distribution of the traffic load is highly desirable.

10.2.3 Classification of Power Conservation Mechanisms for Wireless Sensor Networks

There are many power conservation mechanisms (PCMs) proposed for WSNs, which can be classified into two main categories [10]: *active* mechanisms and *passive* mechanisms, as shown in Fig. 10.3. *Active* mechanisms refer to those that achieve energy conservation by utilizing energy-efficient network protocols, rather than turning-off the radio (or transceiver) interface of a sensor node, while *passive* mechanisms refer to those that conserve power by turning-off the radio (or transceiver) interface. Passive power conservation mechanisms can further be classified into three basic categories based on the possible control levels for turning-off the radio interface module:

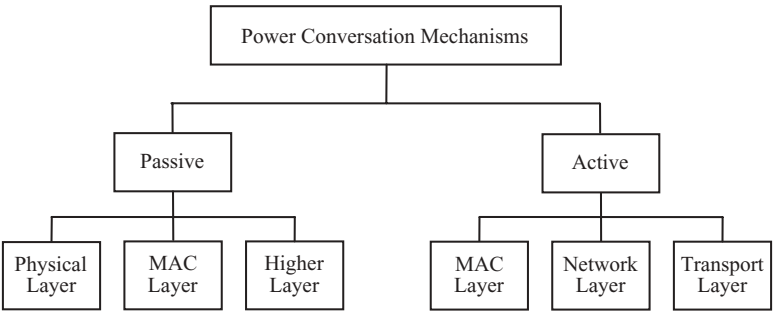


Fig. 10.3 Classification of power conservation mechanisms.

- Physical-layer power conservation mechanisms.
- MAC layer power conservation mechanisms.
- Higher layer power conservation mechanisms.

Active power conservation mechanisms can also be further classified into three basic categories based on different protocol layers:

- MAC layer mechanisms.
- Network layer mechanisms.
- Transport layer mechanisms.

In the subsequent sections, we will give a more detailed introduction of these power conservation mechanisms.

10.3 PASSIVE POWER CONSERVATION MECHANISMS

Passive power conservation mechanisms reduce the energy consumption of a sensor node by turning-off its transceiver interface module when there is no communication activity [10]; that is, the transceiver is turned-off during the periods when the sensor node is neither transmitting nor receiving data [12].

The concept of turning-off the transceiver was first introduced in IEEE 802.11, Part 11 (ISO/IEC 8802-11). According to this standard, a sensor node may switch to a sleep mode by turning-off its transceiver in accordance with the network allocation vector (NAV). Also, every mobile sensor node in the network must wake-up during an announcement traffic indication message (ATIM) period, when the transmitter of a sending mobile sensor node informs its destination not to turn to a power-saving mode. If no notification is received, the mobile sensor node can turn to a power-saving mode and wake-up right in the next ATIM period [13–15]. A sending mobile sensor node can also defer its transmission (or at least decrease the transmission rate) in a noisy channel. It is possible to try to compensate any loss when the channel gets better.

This section presents a comprehensive overview of the most efficient passive power control mechanisms for WSNs based on the classification of power control mechanisms shown in Fig. 10.3.

10.3.1 Physical-Layer Power Conservation Mechanisms

The use of turn-off techniques at the physical layer can achieve substantial energy savings by minimizing the energy consumption of the sensor node processor (or CPU) in an idle state. However, additional energy savings may also be achieved by optimizing the performance of the processor in an active state. If peak performance is not always required, significant energy savings can be achieved without affecting the peak performance of the processor. The processor must be

scheduled in such a way that it is active only when there is data to be processed. This implies dynamically adapting the processor's operating voltage and frequency to instantaneous processing requirements. Dynamic Voltage Scheduling [16] and Dynamic Power Management (DPM) [17] are typical examples of passive power conservation mechanisms at the physical layer.

10.3.1.1 Dynamic Voltage Scheduling. One technique to reduce power consumption in a processing unit is to use a *variable speed processor* (VSP), which can change its speed by varying the clock frequency along with the supply voltage while keeping (not degrading) the required performance [16]. The power consumption of a VSP is reduced by exploiting the idle intervals of the processor.

Dynamic Voltage Scheduling is a voltage scheduling mechanism based on a VSP, which assigns a supply voltage for each task of a processor to minimize the energy consumption of the processor [16]. According to DVS, reducing the operating frequency and accordingly the voltage during the periods of reduced activities can result in a linear reduction of power consumption without affecting the total energy consumed per processing task. The reduction of the operational voltage implies larger path delays, resulting in a continuous effort to compromise the peak performance. The use of a VSP is one of the most promising techniques to reduce power consumption. Like the VSP, the main idea behind DVS is to allow a processor to dynamically change its speed (while in operation and under software control) by varying the operating clock frequency along with the power supply (or processor voltage) to match the dynamic workload without degrading the required performance. The power consumption of a VSP can be reduced by exploiting the idle intervals of the processor. This allows the processor to provide the minimum required clock frequency with the maximum possible energy efficiency. To implement this, DVS requires an algorithm, termed *voltage scheduler* (VS), to determine the operating frequency of the processor at run-time. The implementation of DVS, for a general-purpose microprocessor, requires substantial software support and new metrics to fully realize and understand the advantages of this capability.

10.3.1.2 Dynamic Power Management. Dynamic Power Management is another physical-layer operating-system-directed power management mechanism proposed in Ref. [17], which can achieve additional power savings and thus increase the lifetime of a sensor node. DPM is an efficient mechanism for reducing system power consumption without significantly affecting its performance. This mechanism is actually based on the DVS introduced above [16] and deals with the transition of a node state in an energy-efficient manner. The basic idea behind it is to turn-off the components of a sensor node (i.e., A/D converter, processor, memory, and transceiver) when no event occurs and get them back or wake them up when needed. Such event-driven power management is critical to achieving a maximum battery lifetime.

Figure 10.4 illustrates the delay and power consumption during the transition of a node from the active sate to the sleep state, and inversely. It is seen that

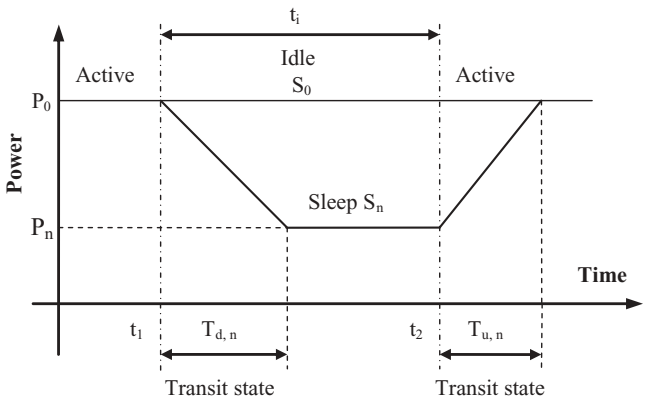


Fig. 10.4 State transition delay and power consumption.

when a sensor node n detects an event in its area at a given time, it completes the processing of the event at time t_1, while the next event occurs at time $t_2 = t_1 + t_i$. At time t_1, node n goes over the transit state $T_{d,n}$ to the sleep state s_n from the active state s_0. Each state s_n has a power consumption P_n.

This power-saving mechanism, at first look, provides considerable energy gains. However, one should not overlook the fact that sensor nodes communicate with each other using short data packets. The shorter the data packets are, the more the consumption of start-up energy is. This is because the transition of a node from one state to another takes a certain period of time and causes some overhead. The sleep-state transition requires storing the processor state and turning off power. The awakening process takes a certain period of time. Therefore, if a node keeps turning the transceiver off during each idling slot, over a certain period of time, it may end up with more energy consumed than if the transceiver is left on. Therefore, the operation in a power-saving mode is energy efficient only if the time a node stays in that mode is longer than a certain threshold. It is obvious that a correct policy for the sleep-*state* transition is critical for the success of DPM [17]. There can be many different operation modes for a sensor node. The number of operation modes depends on the number of states of the node components. Each operation mode can be characterized by its power consumption and latency overhead, which is the transition power to and from that mode.

In the DPM mechanism, a sensor node has five sleep states or power-saving operation modes, as shown in Table 10.1. Each of the sleep states corresponds to a particular combination of component power states (or modes) of the sensor node. The energy consumed in each state a sensor node can be in, for example, ON (idle or active) and OFF, with respect to data transmission, data reception, and data processing, is given in Table 10.1. Each sleep state is characterized by latency and power consumption. The deeper a sleep state, the less the power consumption, and the more the latency. It can be seen in Table 10.1 that not all combinations of component states are useful. For example, if the processor is in

TABLE 10.1 Sensor Node's Sleep State

Sleep States	Sensor with A/D Converter	Processor	Memory	Transceiver (Radio)	Sleep State Power (mW)
S_0	On	Active	Active	T_X/R_X	1,040
S_1	On	Idle	Sleep	T_X	400
S_2	On	Sleep	Sleep	R_X	270
S_3	On	Sleep	Sleep	Off	200
S_4	Off	Sleep	Sleep	Off	10

an idle state, the memory should be in a sleep state. This removes some combinations from the node states. According to Table 10.1, the least energy is always consumed in sleep state S_4, where there is no sensing and communicating activity (i.e., the sensing and transceiver devices are turned off), while at the same time the processing and storing devices are in the sleep state. The largest energy consumption happens in sleep state S_0 when the sensing device is ON, the processing and storing devices are in the active state, and the transceiver is transmitting and receiving data.

10.3.1.3 Embedded Power Supply for Low-Power Digital Signal Processors. In many digital signal processor (DSP) systems, the number of operations performed per sample can be minimized dynamically by exploiting time-varying signal characteristics. Embedded power supply is a power control mechanism, which uses dynamically adjustable power supplies to minimize power dissipation in DSPs [18]. In this mechanism, power-down techniques can be used to make power dissipation directly proportional to the computational load per sample. The basic idea is to lower the power supply voltage and slow the clock during the periods of reduced load instead of running at a fixed speed and idling. It has been shown in Ref. [18] that this mechanism can yield a typical power savings of up to 30–50%. If latency is tolerable, it can yield power savings of an order of magnitude in some applications by buffering data and averaging the processing rate.

10.3.1.4 Energy-Efficient System Partitioning. Local computation of sensor data at the sensor node level can be highly energy efficient because it can reduce redundant data transmission in a network [19]. One of the most efficient techniques for power conservation at the chip level is to exploit distributed parallel computation at multiple sensor nodes. Partitioning overall computation among multiple sensor nodes and performing the partitioned computations in parallel locally at different nodes can provide a better control on latency, which can result in energy consumption through voltage and frequency scaling. Parallel computation leads to lowered voltage supply level and slowed clock frequency of sensor nodes, which in turn reduces energy consumption. Therefore, it is of great importance to develop energy-efficient signal processing algorithms running at the

sensor node level. The results in Ref. [19] show that an energy reduction of up to 60% can be achieved in a source localization application through distributed parallel computation.

10.3.1.5 Energy-Efficient Link Layer. Shih et al. [20] investigated the impact of energy-efficient techniques that adapt link- and physical-layer parameters, for example, output transmitting power and error control code, on system energy dissipation. Since the communication cost over a long distance can be very high, minimizing the energy for communication is of great importance. In general, the minimum output power required to transmit a signal over a distance d is proportional to d^n, where n is a variable ($2 \le n \le 4$) derived from the experimentation. Reliable data transfer can be achieved by either increasing the transmitting power (P_{out}) of a transceiver (radio) or adding a forward error correction (FEC) code to the data to be transmitted. The bit error probability (P_b) for any fixed value of transmitting power can be decreased with the use of FEC. However, FEC, at any rate, requires additional processing and thus additional energy at the transceiver, which is not desirable. Wang et al. [19] attempt to minimize the system energy required to send data from one node to another by partitioning the energy between transmitting data and processing error correction. Shih et al. [20] attempt to minimize the system energy consumption through a compromise on the quality of the established link layer. This can be achieved by maintaining the bit error rate (BER) just below the user requirements. More specifically, the selected processor of a sensor node is adapted to provide dynamic voltage scaling, while the micro-operating system (μ-OS) is customized to allow software to scale the energy consumption of the processor. The on-board phase-locked loop (PLL), transmitter, and receiver can be turned off via software (or hardware) control for energy dissipation reduction. The encoding and decoding of error-correcting codes can be performed on different platforms. The energy consumed for encoding and decoding data is directly measured instead of being modeled.

Basically, using the error-correcting technique can lower the decoding energy per information bit by up to five orders of magnitude. Thus, sensor data can be encoded using a convolution code to allow for lower output transmitting power.

10.3.2 MAC Layer Power Conservation Mechanisms

The IEEE MAC WLAN Specifications (ISO/IEC 8802-11) employs a low-energy consumption mechanism to prolong the battery lifetime of sensor nodes and adapted CSMA/CA to reduce the impact of hidden nodes. In this protocol, request-to-send/clear-to-send (RTS/CTS) handshake packets are used to reserve a transmission floor (a threshold) for subsequent data packets. The handshake signaling packets are used only for relatively long data [21]. Sensor nodes transmit their control and data packets at a common maximum power level, preventing all potentially interfering sensor nodes from starting their own transmissions. Only one transmission is allowed at a time because all sensor nodes are within

the carrier-sense range of each other. Therefore, interfering nodes are not allowed to transmit concurrently. The protocol addresses the transmission issue from a single-layer perspective, which is inefficient. The IEEE 802.11 mechanism uses the RTS/CTS packets to silence the neighboring nodes. The maximum power (P_{max}) is used to determine node connectivity. The P_{max} parameter is a fixed power level at which sensor nodes send their control (RTS/CTS) packets. This mechanism allows for communication with any sensor node that is within the maximum range and hence produces a higher level of access toward its destination per hop. The transmitter–receiver separation distance is not continuous. Therefore, sensor nodes using the IEEE 802.11 mechanism cannot achieve the maximum range by using P_{max} and energy is wasted. To improve the deficiencies of the IEEE 802.11 mechanism, a variety of low-energy consumption mechanisms for conventional WSNs have been proposed in the literature.

The MAC layer power conservation mechanisms allows the MAC layer to decide whether there is a frame transmission that is destined to it, and then turn off the radio interface module for just one transmission frame. As a result, a sensor node can save power from every frame transmission. Moreover, there will be no delay to all incoming traffic to a sensor node because a sensor node is never turned off for more than one transmission frame [10].

PAMAS is a MAC protocol for wireless ad hoc networks [22] and a typical example of MAC layer power conservation mechanisms. It brought an improvement on energy savings (compared with the standardized IEEE 802.11 distributed coordination function—DCF) (ISO/IEC 8802-11) by trying to avoid overhearing among neighboring sensor nodes. It is built on the multiple access with collision avoidance wireless (MACAW) protocol [23]. Actually, it is a combination of the original multiple access with collision avoidance (MACA) protocol [24] and the idea of using a separate signaling channel [13,25–27].

The main characteristic of PAMAS is that it requires two independent radio channels. In most cases, this means two independent radio systems on each sensor node. PAMAS does not attempt to reduce idle listening, which is a disadvantage compared to sensor-MAC (S-MAC) [28], another well-known MAC protocol, which will be described in Section 10.3.3. It saves the battery power of a sensor node by intelligently turning off the sensor node when it is not transmitting data. In the PAMAS protocol, a receiving mobile sensor node transmits a busy tone (in a separate control channel) when it starts receiving data frames so that other mobile sensor nodes know when to turn off. When a mobile sensor node does not have data to transmit, it should power itself off if a neighbor starts transmitting to some other node. A sensor node should be turned off even if it has data to transmit if at least one of its neighbor pairs is communicating. A mobile sensor node, which has been turned off when one or more of its neighbor pairs started communication, can determine the length of time that it should be turned off by using a probe protocol. In this probe protocol, the sensor node performs a binary search to determine the time when the current transmission will end. However, the loss of probe frames may cause significant power wastage [22].

Singh et al. [22] showed that by using PAMAS power savings in the range from 10% (for sparsely connected sensor networks) to almost 70% (for fully connected sensor networks) could be achieved without affecting the delay or throughput performance.

10.3.3 Higher Layer Power Conservation Mechanisms

A higher layer power conservation mechanism operates at a layer higher than the MAC layer and controls the radio interface module operation so that the radio interface module can be turned off longer than the transmission time of a single MAC frame [10]. Compared to a MAC layer mechanism, it makes use of higher layer information to decide the time period when the radio interface module of a sensor node is turned off.

Higher layer power conservation mechanisms can be classified into two basic categories: *nonstructure based* and *structure based*. In nonstructure-based mechanisms, all sensor nodes in a network are equal in terms of their functionalities [10]. Each sensor node independently schedules its sleeping intervals based on both internal information and/or neighbor information. The coordination of sleeping schedules between sensor nodes is done implicitly through exchanged beacon or hello messages. This section introduces several nonstructure-based mechanisms, including S-MAC [28], Energy Efficiency Using Sleep Mode TDMA Scheduling [14,15], the TDMA scheduling algorithm [29], and Self-Stabilizing Deterministic TDMA [30].

In structure-based mechanisms, all sensor nodes in a network are organized into a structure, for example, a group of clusters. A power conservation mechanism is performed only in each cluster-head node, which can have a better view of its local cluster. In each cluster, it may perform a coordination task for the cluster member nodes, such as synchronization of their sleeping schedules to ensure enough bandwidth and to function as a proxy for sleeping sensor nodes [10]. Section 10.3.3.9, introduces a structure-based power conservation mechanism called SPAN.

10.3.3.1 Sensor-MAC. Sensor-MAC (S-MAC) is a sensor MAC layer protocol where sensor nodes are allowed to discover their neighbors and organize a network for communication without requiring the existence of master nodes in the network. Thus, there are no clusters or cluster heads in the network and the network topology is flat. Sensor-MAC focuses mainly on energy conservation in major energy wastage sources, while achieving good scalability and collision avoidance capability. The major energy wastage sources are classified into overhearing, idle listening, collision, and control overhead. All sensor nodes try to achieve a single common task and do not require an equal opportunity to transmit.

Sensor-MAC introduces three techniques to reduce energy consumption. First, neighboring nodes are synchronized to go to sleep periodically (see Fig. 10.5) so that they do not waste energy when a neighboring node is transmitting

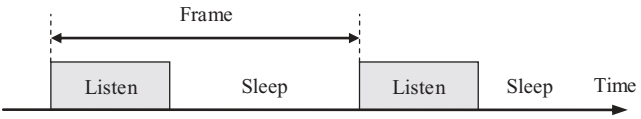


Fig. 10.5 Periodic listen and sleep.

to another node or by listening to an empty channel. This addresses the overhearing problem. Second, control packet overhead is kept low because synchronized neighboring nodes form virtual clusters to synchronize their wake-up and sleep periods. There is no real clustering and no intercluster communication problem. Third, message passing is used to reduce contention latency and control overhead.

Comparison between S-MAC and IEEE 802.11

S-MAC has good energy conserving performance compared with IEEE 802.11, as shown in Fig. 10.6. The MAC layer can have a big impact on power consumption. S-MAC saves energy by periodically switching the operational state of a sensor node between a SLEEP mode and a WAKE-UP mode. The sensor node does not have to stay in an active state continually but only when it has control or data packets to send. In contrast, a sensor node using IEEE 802.11 is always in an active state because it has to continually exchange synchronization packets with its neighboring nodes, which consumes its energy quickly [28]. Another interesting property of S-MAC is that it has the ability to make trade-offs between energy and latency according to traffic conditions.

According to the experimental results obtained in [28], IEEE 802.11 MAC consumes more than twice the energy consumed by S-MAC when the traffic is heavy, as shown in Fig. 10.6. Moreover, energy savings from periodic sleeping is very limited because idle listening rarely occurs. S-MAC achieves energy savings mainly by avoiding overhearing and efficiently transmitting a long message. The complete (with periodic sleep) S-MAC protocol has the best energy savings and far outperforms IEEE 802.11 MAC when the message inter-arrival period is larger than 4 s or the traffic load becomes light.

Comparison between S-MAC and PAMAS

Compared with PAMAS, S-MAC does not use any out-of-channel signaling, while PAMAS requires two independent radio channels and thus two independent radio (transmitter and receiver) systems on each sensor node. Moreover, PAMAS does not consider saving energy by reducing idle listening.

10.3.3.2 Energy Efficiency Using Sleep Mode TDMA Scheduling. The Energy Efficiency Using Sleep Mode TDMA Scheduling is an energy-efficient

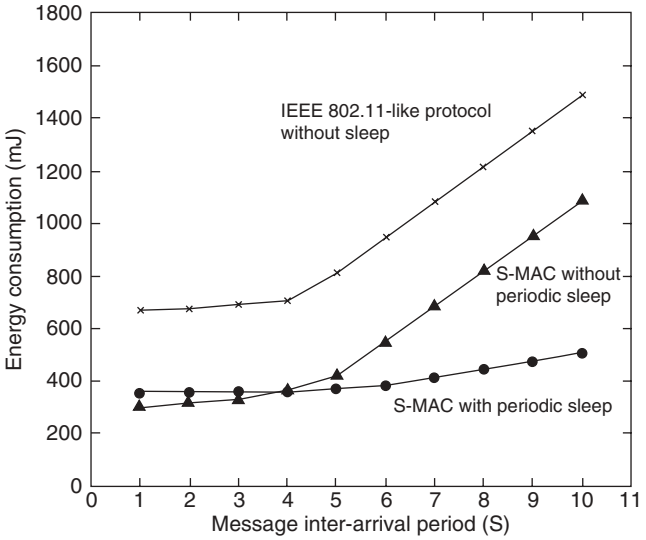


Fig. 10.6 Comparison between S-MAC and IEEE 802.11 protocols.

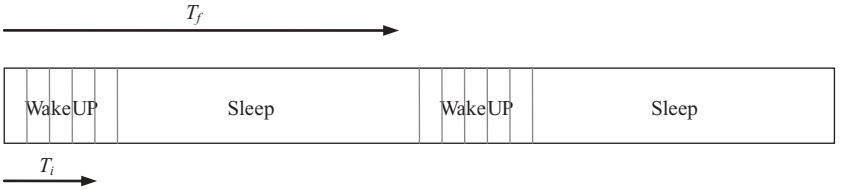


Fig. 10.7 Sleep and wake-up packets of the time division multiple access (TDMA) scheduling.

scheduling algorithm for WSNs [14,15]. This scheduling algorithm takes the advantage of the power conservation mechanism in the S-MAC protocol [28] and extends it in order to minimize end-to-end delay. The main disadvantage of S-MAC is that each sensor node has to wait until the next WakeUP (WU) time for the next hop before forwarding a message. This makes end-to-end delay proportional to the number of intermediate forwarding nodes times the sleep time of each node. In contrast, in the TDMA scheduling algorithm, all nodes in the network are synchronized to sleep at the same time, and wake up during the WU period, as shown in Fig. 10.7. Instead of transmitting the entire message during the WU period, a node transmits a short WU packet, which is forwarded until it reaches the gateway. Nodes that receive a WU packet remain in the idle mode, anticipating the subsequent packet reception, whereas nodes that do not receive a WU packet go to the sleep mode. Also, multiple WU packets can be aggregated when two paths merge in order to minimize the WU duration and to

avoid unnecessary transmissions. A further improvement to the algorithm is not to allow the transmitter of each node on for the entire WU period, but only for the specific timeslots the node anticipates a reception.

The questions that arise are *Which slot should each node use to transmit its WU messages (originated, the ones it produces, or forwarded)?* and *At which slots should each node listen to the channel for incoming data?* Path WU is a technique that is used to wake up all the nodes in the path between a sending sensor node and its destination before the node starts to send data, and requires that the first nodes in the path be assigned timeslots earlier than the nodes that follow. On the other hand, collisions can be avoided if nodes that receive simultaneously are not 1-hop neighbors. The TDMA scheduling algorithm does not suffer from the exposed terminal situation. Also, possible transmissions to the same destination should be assigned in different timeslots. However, a scheduling algorithm should maximize the concurrent receptions made by nodes that are not 1-hop neighbors in order to minimize the total frame length. Therefore, timeslot scheduling should take into account routing paths and neighboring information. These limitations make distributed TDMA scheduling algorithms inefficient because they do not take into account the desired order of transmissions. The TDMA scheduling algorithm proposed by Vergados et al. [14,15] can create a TDMA schedule appropriate for WU transmissions in sensor networks.

10.3.3.3 SS-TDMA: A Self-Stabilizing MAC. The TDMA scheduling algorithm enables a sensor node to minimize idle listening and can thus save energy. It is also convenient in adapting an existing scheduling algorithm into one suitable for WSNs. On the other hand, the synchronization clock in TDMA may drift, which would lead to the corruption of assigned slots. In that case, a scheduling algorithm should be able to be self-stabilizing in order to recover the normal states of the assigned slots.

SS-TDMA is a self-stabilizing deterministic scheduling algorithm for WSNs [31]. This algorithm is based on systematically reusing a graph traversal algorithm and requires a sensor node only aware of its neighbors. The optimization of bandwidth utilization and the recovery of corrupted slots are also considered. Kulkarni et al. [31] focused on the problem of designing a TDMA service for a grid-based sensor network. This kind of networks can be found in many sensor applications, for example, environmental monitoring and hazard detection. Three communication patterns, *broadcast*, *convergecast*, and *local gossip*, occur frequently in such networks. They developed a TDMA service that can be customized based on the application requirements and also provided guidance about using this service when the communication pattern is unknown or varies with time. With these customizations, whenever a sensor receives a message, it can forward it to its successors with a small delay. They showed that this TDMA service is collision free, whereas existing CSMA based approaches suffer significant collisions. They also showed how this service can be extended to deal with other deployments in a two-dimensional (2D) field, sensor failures, and sensors that are sleeping as part of a power management scheme. Furthermore, they

showed that this service can be used in a mobile sensor network that provides localization service.

10.3.3.4 Link Scheduling. Gandham et al. [32] considered the problem of link scheduling in a WSN and proposed a link scheduling algorithm that employs a TDMA MAC protocol. The link scheduling algorithm consists of two phases. In the first phase, a color is assigned to each edge in the network such that no two edges incident on the same sensor node are assigned the same color. For this purpose, they proposed a distributed edge coloring algorithm that requires at most $(\delta + 1)$ colors, where δ is the maximum degree of an arbitrary planar graph with no self-loops and no multiple edges, into which the network can be decomposed. In the second phase, each color is first mapped to a unique timeslot, attempting to identify a direction of transmission along each edge in order to avoid the exposed terminal problem. Then a direction of transmission for each edge is obtained by using additional timeslots. Finally, another feasible direction of transmission is obtained by reversing the direction of transmission along every edge. A TDMA MAC schedule is obtained by using both transmission assignments, which enables two-way communication between each pair of neighbors. The simulation results have shown that for sparse graphs with cycles the number of timeslots assigned is very close to $2(\delta + 1)$.

10.3.3.5 Energy-Latency Trade-Offs for Data Gathering. One of the most useful approaches for saving energy in communications is to explore the trade-offs between energy and latency. For this purpose, different techniques, for example, modulation scaling, have been proposed in the literature. Yu et al. [33] explored the trade-offs in the context of data gathering, subject to application-level performance constraints. They considered a real-time scenario where the crude data gathered from the source nodes need to be aggregated and transmitted to the sink within a certain latency constraint. Their technique can be applicable to any given aggregation function. They proposed algorithms to reduce the overall energy consumption of the sensor nodes in the aggregation tree, subject to a latency limitation. They used an accurate energy model to abstract the energy characteristics for packet transmission. More specifically, the transmission energy may increase when the transmission time exceeds some threshold value. For the off-line version of the problem, which cannot be controlled by the program, they proposed a numerical algorithm for the optimal solution, and a pseudo-polynomial time approximation algorithm based on dynamic programming. They also discussed the techniques for handling the interference among sensor nodes. Their simulation results show that under different settings of several key parameters, energy savings between 20 and 90% could be achieved compared to the classic shutdown techniques. Moreover, energy conservation between 15 and 90% could be achieved by using an on-line distributed algorithm that needs only local information of the aggregation tree. It is also demonstrated through several run-time scenarios that the algorithm is adaptable to different packet sizes and latency constraints.

10.3.3.6 TDMA Scheduling. A problem in developing TDMA scheduling algorithms in a multihop network is how to determine the smallest length conflict-free assignment of timeslots for each link or node. This is based on the assumption that there are many independent point-to-point flows in the network. In WSNs, however, data are often transferred from originating sensor nodes to a few central data collectors (e.g., gateways or base stations). To address the above problem, Ergen and Varaja [29] proposed a TDMA scheduling algorithm, which determines the smallest length conflict-free assignment of timeslots for sending the packets of each node to their destination node. They showed that the minimum-delay scheduling can always be found by using a simple algorithm for TDMA based routing algorithms when the network is loop free and has only one sink node. More specifically, they first proposed two centralized heuristic algorithms for solving the problem: node-based scheduling and level-based scheduling. The node-based scheduling algorithm directly schedules the nodes in a network whereas the level-based scheduling algorithm schedules the levels in a routing tree before scheduling the nodes. The performance of these algorithms depends on the distribution of the nodes across the levels. The experimental results in Ref. [29] show that a substantial reduction of energy and delay is possible. Actually, they proposed a load distribution algorithm based on the distributed coloring of the nodes, which increases the delay by a factor of 10–70 over centralized algorithms for 1000 nodes. They also obtained an upper bound for these schedules as a function of the total number of packets generated in the network. Note that the term "distributed" in the proposed algorithms is used to characterize the load distribution in the network and not the network architecture.

10.3.3.7 Wave Scheduling. Since radio communication is the largest power consumption source in sensor nodes, it is important to develop energy-efficient data dissemination techniques to extend their lifetime. A very common communication pattern in WSNs is to send sensor readings from multiple source nodes to a designated sensor node, called *view node*. In order to achieve energy efficiency in such data communication, Trigoni et al. [34] addressed several challenges intrinsic to ad hoc network communication in, for example, collision minimization at the MAC layer, energy-efficient radio management, and selection of energy-efficient routes. More specifically, given a set of sensor nodes, arranged in a rectangular grid, Trigoni et al. [34] proposed a class of periodic activation schedules that conserve energy by first avoiding interference at the MAC layer and secondly allowing sensor nodes to turn off their radios whenever there is no communication activity. These schedules are called *wave schedules*, where every edge of the rectilinear grid is activated periodically at well-defined communication intervals, called *send–receive* intervals. Actually, they considered data dissemination strategies that avoid collisions, and therefore message retransmissions at the cost of higher message latency. They proposed a new methodology for trading latency for energy in WSNs. More specifically, they proposed a new scheduling algorithm, which carefully schedules message transmissions in such a

way that collisions are avoided at the MAC layer. In this algorithm, all nodes in the network adhere to the same schedule. Thus, sensor node radios can be turned off most of the time and wake up only during well-specified time intervals. Since current generation radios consume nearly as much power when listening or receiving as when transmitting, turning them off when not needed yields significant energy savings. In addition, they also showed how routing protocols can be optimized to interact with the scheduling decisions in a symbiotic way, resulting in significant energy savings but at the cost of higher latency.

10.3.3.8 Joint Optimization with Energy Constraints. Cui et al. [35] considered energy-constrained WSNs where energy consumption must be minimized while satisfying given throughput requirements. Moreover, energy consumption must take into account both the transmission energy and the data processing energy for short-range communications. They emphasized that energy efficiency must be supported across all layers of the protocol stack through a cross-layer design. They also discussed energy-efficient joint routing, scheduling, and link adaptation strategies that maximize the network lifetime. First, they proposed a variable-length TDMA scheduling algorithm, in which the slot length is optimally assigned according to the routing requirement while minimizing the energy consumption across the network. More specifically, they proposed a simple link scheduling algorithm to find the minimum-delay schedule that provides slot lengths for all links. Afterward, they combined the obtained results with their previous work on an energy-optimal cross-layer design in order to minimize the delay for transferring a fixed number of bits from the source nodes to the sink in an energy-limited manner. Moreover, they studied the trade-off between the total energy consumption and delay. Their experimental results show that multihop transmissions are more energy efficient when only the transmission energy is considered, while single-hop transmissions may be more efficient when the processing energy is considered.

10.3.3.9 Energy-Efficient Coordination for Topology Maintenance. SPAN is a distributed power-saving coordination algorithm for multihop ad hoc wireless networks, which can reduce energy consumption without significantly affecting the connectivity or capacity of a WSN [36]. In SPAN, all nodes make local decisions on whether to sleep or to join a forwarding backbone as a coordinator. The decision made by each node is based on an estimation of the number of its neighbors benefiting from the node's being awake and the amount of energy available to the node. Actually, SPAN is a randomized algorithm where coordinators rotate with time, demonstrating how localized node decisions lead to a connected capacity-preserving global topology. It is a very common power conservation mechanism utilizing a backbone to facilitate routing. It modifies the IEEE 802.11 ad hoc power-saving mode and uses it to lengthen the sleeping interval (longer than one MAC transmission frame), and to reduce packet loss and delay. It is mainly based on the following observation: When a region of a shared-channel wireless network has a sufficient density of nodes, only a small

number of them need to be ON at any time to forward traffic for active connections. By using SPAN, the network lifetime increases as the ratio of idle-to-sleep energy consumption increases and as the density of the network increases. Chen et al. [36] showed that SPAN improves communication, latency, capacity, and network lifetime when run in conjunction with the IEEE 802.11 power-saving mode.

Comparison between SPAN and PAMAS

PAMAS [22] turns off a node's transceiver when the node is overhearing a packet not addressed to it. It is suitable for transceivers where the processing of a received packet is expensive compared to listening to an idle radio medium. In contrast, SPAN assumes the presence of an ad hoc polling mechanism, for example, that provided by IEEE 802.11, and can potentially work harmonically with application hints. Such hints apply only to sleeping nodes, not to coordinators.

10.4 ACTIVE POWER CONSERVATION MECHANISMS

Active power conservation mechanisms reduce the energy consumption of a sensor node by improving the node operation instead of turning off its radio interface module into a power-saving mode [10]. This section presents an overview of major active power control mechanisms for WSNs based on the classification of the PCMs in Section 10.2 and focuses on power-aware routing protocols.

10.4.1 MAC Layer Mechanisms

One approach to power conservation is to reduce collision probability and thus control the power consumption of a sensor node at the MAC layer by adaptively adjusting the transmission power to an appropriate level for generating signal strength just enough to reach the next hop destination. There are many MAC layer mechanisms proposed in the literature. This section introduces several typical examples.

10.4.1.1 Multiple Access with Collision Avoidance. Multiple Access with Collision Avoidance [24] was one of the first channel-access protocols proposed for addressing the hidden node and the exposed node problems in wireless networks. This protocol also provides the ability to perform transmission power control per frame without using carrier sensing. In MACA, a three-layer handshake RTS (ready to send)/CTS (clear to send)/DATA is adopted, which is based on the RTS–CTS exchange. A source station (or sender) transmits an RTS frame to the destination station (or receiver) to request transmission. If the destination receives the RTS frame correctly, it will receive the transmission by sending back a CTS frame. When a mobile node overhears some RTS/CTS frames corresponding to the transmissions of other nodes, it is not necessary to remain silent

completely. Instead, it can communicate with other neighboring nodes with lower transmission power [12]. Moreover, other stations that hear the RTS or CTS frames need to build a new scheduling of their transmissions at a later time in order to avoid frame collisions. The recovery of collisions is then left up to the transport layer, which greatly decreases throughput and therefore power consumption.

10.4.1.2 Multiple Access with Collision Avoidance Wireless. Multiple Access with Collision Avoidance Wireless (MACAW) [23] is another derivative of the CSMA/CA protocol, which makes use of the RTS/CTS/DS/DATA/ACK handshake signaling (or message exchange) and incorporates a significantly different back-off algorithm. It is a modified version of the MACA protocol, where link layer acknowledgments (ACKs) are added. In the MACAW protocol, a sender has the possibility of retransmitting a packet that was not successfully received by the receiver. The use of the acknowledgment improves the reliability of a wireless link, and avoids the long recovery cost at the transport layer, thus consuming less energy in transmitting a packet.

10.4.1.3 Floor Acquisition Multiple Access. The Floor Acquisition Multiple Access (FAMA) protocol [27] was introduced for ad hoc networks and wireless LANs, which are based on a single channel and asynchronous transmissions. It is actually a MAC protocol that employs RTS/CTS/DATA handshake signaling like MACA [24]. Its main proposition is that a station must acquire a channel before transmitting its data. One of the ways to acquire the channel is to use RTS/CTS exchange. More specifically, FAMA is based on a three-way handshake between a sender and a receiver. The sender uses non-persistent carrier sensing to transmit an RTS, while the receiver sends a CTS that lasts much longer than the RTS serving as a "busy tone" that forces all hidden nodes to back off long enough in order to allow a collision-free data packet to arrive at the receiver.

Comparison between MACA, MACAW, FAMA, and PAMAS

The IEEE 802.11 standard for wireless LANs includes the collision avoidance of MACA and MACAW. Moreover, all directed traffic uses positive ACKs (as in MACAW). Actually, PAMAS was based on MACA, MACAW, and FAMA protocols in order to improve power consumption.

The performance comparison among MACA, MACAW, FAMA, and PAMAS protocols reveals that PAMAS is more energy efficient than the rest of the protocols, but it comes behind as far as the throughput and delay concerns, as shown in Table 10.2, where "+" or "–" denote "improvement/falloff (degradation)" in the corresponding performance metric, while 0 denotes "no effect".

10.4.1.4 Intelligent Medium Access with Busy Tone and Power Control. Intelligent Medium Access with Busy Tone and Power Control is another MAC protocol for saving power in mobile ad hoc networks [37]. It

TABLE 10.2 Performance Comparison

Protocols	Throughput/delay	Fairness	Energy efficiency
MACA	+ + +	0	+
MACAW	+ + +	0	+
FAMA	+	+	0
PAMAS	–	0	+ +

combines the concept of power control with the RTS/CTS based and busy-tone-based MAC protocols in order to further improve channel utilization [13,25].

The main idea of this protocol is to use the exchange of RTS and CTS packets between two intending communicators to determine their relative distance. This information is then utilized to limit the power level on which a mobile host transmits its data packets. The use of lower power can increase channel reuse and thus channel utilization. It can also save the limited battery energy of mobile (or static) sensor nodes and reduces cochannel interference with their neighbors. With Dual Busy Tone Multiple Access (DBTMA) [13,25], it is possible to use busy tones to save power. According to the DBTMA protocol, the single common channel is split into two subchannels: a data channel and a control channel. The control channel is used to transmit RTS/CTS messages. The use of lower transmission power can increase channel reuse in a physical area. The main idea here is to tune properly the power level of each transmitter so that all communication pairs can coexist without any interference. Power control is incorporated into the original protocol, which follows the following rules: (a) The data packets and transmission busy tone (Btt) are transmitted with power control that is based on the power level of the received CTS. (b) The CTS and receiver busy tone (Btr) are transmitted at the largest power level, and RTS is transmitted at a power level determined based on how strong the Btr tones are around the requesting host. For the performance evaluation, a comparison was made with the DBTMA protocol. The analysis was made only for two communication pairs. Extending to more communication pairs would be difficult, if not possible. In practice, the power levels provided by the physical layer may not be tunable without limitation. A more realistic assumption is that only a certain number of discrete power levels are possible.

10.4.1.5 Power Controlled Multiple Access. Power Controlled Multiple Access (PCMA) [38] is a MAC protocol that can achieve power controlled transmission and thus collision avoidance. This protocol is applicable to WSNs though it was originally proposed for wireless ad hoc networks, where all nodes share a single channel and there is no centralized access control and power control for increasing channel efficiency rather than battery lifetime. The goal of the PCMA protocol is to achieve power controlled multiple access within the framework of CSMA/CA base multiple access protocols. The protocol has 1:1 analogies with the key components in standard CSMA/CA protocols [24]. At the sender side,

monitoring the busy tone in PCMA is equivalent to sensing the carrier in CSMA/CA. At the receiver side, pulsing periodically the busy tone in PCMA is equivalent to sending CTS for collision avoidance in CSMA/CA.

It has been shown in [38] that PCMA significantly outperforms IEEE 802.11 and the performance of PCMA increases as the number of busy tone pulses increases (ISO/IEC 8802-11), which can approach the performance of an ideal power controlled (IPC) protocol. Moreover, if a network becomes more clustered, its throughput will increase because a greater number of concurrent transmissions are possible and less sensor nodes compete within each cluster. The PCMA protocol can improve aggregate channel utilization by more than a factor of 2 compared to the IEEE 802.11 protocol. The control and data packets must be transmitted with a fixed power (ISO/IEC 8802-11). However, from the view of channel reuse, the adjusting transmission for data has no consequence in terms of increasing channel reuse, and is equivalent to a "fixed-power" MAC protocol.

Comparison between PCMA and IEEE 802.11

The comparison between PCMA and IEEE 802.11 protocols [38] shows the following results: IEEE 802.11 protocol has an equal probability of sending packets to destinations at any distance since the transmission power is not taken into account while contending. However, because of all the transmissions sent at a fixed power level, there is less noise protection for destinations further from their sources resulting in a great number of lost packets at greater network loads. On the other hand, PCMA has the same amount of protection for destination in all ranges. The main idea here is to increase the range distributions, while still limiting at the same time the transmission ranges to the same distance. This is the way to improve the fairness for power controlled multiple access protocols. For dense networks, with a spatial reuse to be exploited, PCMA performs significantly better than the IEEE 802.11 protocol.

10.4.1.6 Power Adaptation for Starvation Avoidance. Power Adaptation for Starvation Avoidance (PASA) [39] is another MAC protocol for reducing the energy consumption and prolonging the battery lifetime of a sensor node. It is a simple, effective, and autonomous mechanism without control message overhead. Despite the collision avoidance mechanisms developed for wireless ad hoc networks in recent years, IEEE 802.11 cannot eliminate collisions completely, which may lead to the channel capture phenomenon where a common channel is monopolized by a single or a few nodes [40,41]. Capture leads to starvation in some nodes, thus degrading the fairness and throughput of a network.

The PASA protocol dynamically adjusts the transmission power in each sensor node to break capture and achieve higher spatial reuse, thus providing to all sensor nodes fair access to the transmission channel. Specifically, PASA adjusts the transmission power in each sensor node according to its current condition, so that all mobile nodes in the network can share the medium channel more efficiently. In contrast, most of previous protocols focus on modifying a MAC protocol [42].

The PASA protocol has the following properties:

- It is a control mechanism without control message overhead. It dos not require any change in MAC or protocols at other layers.
- Every node in the network adapts its power level independently.
- It is able to resolve starvation in many channel capture scenarios.
- Its implementation does not depend on the underlying MAC protocols.
- It is also applicable in nonline-of-sight (NLOS) wireless systems with fixed antennas or base stations.
- It is fair because it can achieve better short-term fairness in channel sharing among the nodes.

Chen et al. [39] showed that PASA can efficiently break starvation, and hence achieve substantially better fairness without compromising throughput. It is self-adjustable in nature and a sensor node adjusts its power continuously and dynamically to find the minimum power. The simulation results confirm that the power of a sensor node would not reach a level far below the minimum. By using PASA, two sources can share the channel much more fairly. Such fairness is achieved by the power adjustment according to the status of each node. It is also shown that the start-up delay with PASA is much lower than that with other MAC protocols.

10.4.2 Network Layer Mechanisms

At the network layer, a routing protocol should take into account the following objectives in order to reduce energy consumption and prolong network lifetime [43,44].

- **Minimization of the Energy Consumed per Packet.** The energy consumed per packet is the energy that is used for the transmission of the packet from the source to the base station (or gateway). The main drawback of this objective is that some sensor nodes will exhibit enormously different energy consumption profiles, that is, some of the sensor nodes will lose more energy than other sensor nodes because they may be in the more frequently used paths, resulting in the early death of some sensor nodes in the network.
- **Maximization of the Network Partition Time.** The network partition time is the time elapsing before the network loses its connectivity. This objective is very useful for critical applications, for example, a battlefield sensor network. Its main drawback is that it cannot provide simultaneously low delay and high throughput.
- **Minimization of the Variance in Node Power Levels.** The variance in sensor node power levels can be measured by the different number of packets these nodes possess. This metric captures the scale or the degree that the

power level is spread out. This objective means that all sensor nodes in the network are of the same importance and no sensor node should be penalized more than any other. It ensures that all the sensor nodes in the network remain alive and run together for as long as possible.

- **Minimization of the Cost per Packet.** The cost per packet is defined as the energy consumed for sending a packet from the source to the destination over the shortest path. The objective is to minimize the cost per packet, which can be achieved by avoiding those sensor nodes with the least residual energy on selected paths.
- **Minimization of the Cost per Node.** The cost per node is defined as the amount of energy consumed by each node for transmitting a packet to the destination. This objective is to minimize the cost per node so that the node lifetime is maximized.

In the subsequent subsections, we will introduce several typical routing protocols for improving energy efficiency and prolonging network lifetime in WSNs.

10.4.2.1 Minimum Cost Forwarding. Ye et al. [45] proposed a routing algorithm called Minimum Cost Forwarding Algorithm (MCFA), which exploits the fact that a routing direction to the fixed external base station is always known in a large-scale WSN. In MCFA, a sensor node is not required to have a unique ID or maintain a routing table. Instead, it only needs to maintain the least-cost estimate from is position to the base station. More specifically, whenever a sensor node has a message to forward, it transmits the message to its neighbors. When a neighbor sensor node receives the message, it will first checks if it is on the least-cost path between the originating sensor node (or source) and the base station. If this really happens, it will retransmit the message to its neighbors. This procedure continues until the message reaches the base station. Note that it is important that each node should know the least-cost path estimate from itself to the base station. Otherwise, the above-described procedure may lead to a situation in which some sensor nodes will have multiple updates and the nodes far away from the base station will get more updates from those closer to the base station, which is not desirable. To avoid the difference in the updates received by different nodes, MCFA was modified in such a way that a back-off algorithm is run at the setup phase. The back-off algorithm decides that a node will not send the updated message until $a \cdot l_c$ time units have elapsed from the time the message was updated, where a is a constant and l_c is the cost of the link from which the message was received.

10.4.2.2 Energy Aware Routing. Energy Aware Routing (EAR) [46] is another routing protocol for prolonging the lifetime of a WSN. The idea is to occasionally use a set of suboptimal paths to increase the network lifetime. The choice of the suboptimal paths is based on a probability function, which depends on the energy consumption of each path. Moreover, network survivability is the

main issue that the protocol is concerned with. It is argued that using the minimum energy path all the time will deplete the energy of nodes on that path. Therefore, EAR uses one of multiple paths between a source node to the data sink with a certain probability so that the lifetime of the entire network is increased. The protocol assumes that each sensor node is addressable through some addressing scheme, which includes the locations and types of the nodes.

Energy aware routing is similar to directed diffusion in the way that potential paths from data sources to the sink are discovered. In directed diffusion, data is sent through multiple paths, one of them being reinforced to have a higher rate. In contrast, EAR selects a single path randomly from multiple alternatives in order to save energy. Compared to directed diffusion, it provides an overall improvement of 21.5% energy saving and a 44% increase in network lifetime. However, such single-path usage hinders the ability of recovering from a sensor node or path failure as opposed to directed diffusion. In addition, the protocol requires gathering the location information and setting up the addressing mechanism for the sensor nodes, which complicates route setup compared to directed diffusion.

10.4.2.3 Minimum Power Configuration. A WSN should reduce the energy consumed in each of the radio's power states (i.e., transmission, reception, and idle) in order to minimize its overall energy consumption. This requires that the network effectively reduces the energy consumed for transmitting or receiving a packet from a sensor node to the base station. However, Xing et al. [47] indicated that the relationship between the different power conservation approaches [5,15,21,28,34,43,44] depend on the network traffic load and hence cannot be combined in a straightforward fashion. For example, when the traffic load is low, the power consumption of the network is dominated by the idle state. In this case, by scheduling nodes to sleep, it saves significant amounts of power. Therefore, it is more power efficient for active sensor nodes to use long communication ranges because that requires fewer nodes to remain awake to relay packets. Conversely, short radio ranges may be preferable when the traffic load is high because the radio spends more time for transmission and reception. Xing et al. [47] proposed a novel approach called *minimum power configuration (MPC)*, which integrates topology control, power-aware routing, and sleep management into a joint optimization problem, and aims at reconfiguring a network in a dynamic way based on current data rates in order to minimize the energy consumption. The simulation results based on realistic models of the Mica2 motes showed that MPC outperforms significantly other existing minimum power routing and topology control protocols in terms of energy conservation.

10.4.2.4 Cost-Effective Maximum Lifetime Routing. Some of the recently proposed routing protocols aim at improving energy efficiency and prolonging network lifetime by balancing the residual energy of the sensor nodes in a network. Although these energy-efficient routing protocols can maintain the stability and connectivity of the network, they are not as cost effective as traditional

routing protocols. Hossain et al. [48] proposed a reactive routing protocol that ensures a satisfactory compromise between two conflicting factors: routing cost and network lifetime, for the best possible route selection. This protocol has two objectives: one is to minimize the cost of routing and the other is to maximize the network lifetime, where the cost of routing can be defined as a function of several parameters, for example, hot count and transmission power. To achieve these objectives, they found a compromise between the cost and the lifetime for each of the possible paths. The proposed protocol can result in a more stable network than that by existing energy-efficient routing protocols and offer a much lower routing cost than that by existing lifetime predictive routing protocols.

10.4.2.5 Power-Aware Sensor Selection. In WSNs, energy is consumed mainly for computation and communication between sensor nodes. The typical ratio of energy consumption for communication and computation is on the scale of 1000. Therefore, minimizing the communication between sensor nodes is crucial for extending the lifetime of WSNs. Another significant metric of WSNs is the accuracy of the sensed data because some sensors within the same area can provide redundant data. Data coming from different sensors may have different qualities because of various physical conditions, for example, distance and noise. Thus, the accuracy of the sensed data depends on the selection of the appropriate sensor nodes by a leader node. A leader node is one that sends querying requests to the sensor nodes within its range in order to accomplish a specific task. Let us suppose that a leader node, after having compared the information and communication cost among different sensor nodes, decides to invoke one of them at a certain time. It is very possible that the same node is chosen more than once. This means that the selected node could die earlier than the rest of the nodes in the local area, which would reduce the network lifetime and thus cause the network partition. Therefore, another approach is to compromise between the quality of the sensed data and the power stored in candidate nodes that will decide for the next sensor node, which may be considered as the most appropriate one for the accuracy of the sensed data. The best thing for the leader nodes to do is to obtain informative sensing data from their neighboring nodes. By balancing the energy of all sensor nodes, the network could be maintained on the same order of power as long as possible so as to maximize the lifetime of the entire network. Therefore, it is critical to introduce collaborative information processing and data aggregation to prolong the lifetime of a network. For this purpose, it is important to carefully select sensor nodes to participate in the collaborative information processing and data aggregation. In this context, Kang et al. [49] proposed an energy-efficient information processing and data aggregation approach. This approach combines the idea of information utility measurement with power awareness and synergistically considers three key factors: sensing quality, communication cost, and power level. Moreover, by using the *autonomic computing* technologies, it enables self-optimization to improve the network performance and at the same time to maintain a good energy level of the local areas in a systematic way.

10.4.2.6 Self-Organizing Routing. Subramanian et al. [50] introduced a self-organizing protocol to build a routing architecture for supporting non-homogeneous mobile or immobile sensor nodes. In this protocol, sensor nodes sense the environment and send sensed data to a prescribed set of nodes which act as relays. The relay nodes are immobile nodes, which form a backbone for communication. Through the relay nodes, the data received from the sensing nodes are forwarded to more powerful base station nodes. Each sensing node should be capable of reaching a relay node in order to be part of the network. The routing architecture, which requires addressing of each sensor node, is hierarchical, where groups of nodes are formed and merged when needed. A local Markov loop algorithm, which is a classical approach to deal with complex combinatorial computations related to mathematical sequences, is used to support fault tolerance through broadcasting. According to this protocol, sensor nodes can be addressed individually in the routing architecture. Therefore, it can be used for applications where communication to a particular node is required. However, this protocol introduces a small additional cost for maintaining routing tables and keeping a balanced routing hierarchy. Through a lot of experiments, it was found that by using this protocol the energy consumed for transmitting a packet is less than that consumed using the SPIN protocol.

10.4.3 Transport Layer Mechanisms

The Transmission Control Protocol (TCP) [51] was not originally designed for energy efficiency but for reducing data retransmissions in a network. The ATCP (TCP for mobile ad hoc networks) and TCP-Probing [52] can alter TCP's retransmission behavior by minimizing unnecessary retransmissions, thus achieving lower power consumption and higher throughput [10]. In the context of WSNs, not much work has been done to address the energy consumption issue at the transport layer. To achieve fair and reliable sensor-to-sink data delivery, it is important to design efficient transport layer protocols with reasonable cost in terms of power and resource consumption, and less complexity and modification to lower layers (e.g., MAC). This section introduces a couple of transport protocols, as well as an experimental study on TCP's energy consumption for WSNs.

10.4.3.1 Experimental Study on TCP's Energy Consumption. An experimental study on TCP's energy consumption is presented in [53], which investigated how TCP's energy consumption can be reduced while remaining within a certain limit imposed by TCP standards. In this study, a sensor node's energy was utilized to improve TCP performance. Different alterations and fine tunings to TCP code (i.e., a code suitable for TCP) were investigated for conserving battery power at sensor nodes by saving software overhead and reducing protocol processing.

A WSN has a relatively small bandwidth (11 Mbits/s for the new WaveLAN cards) and the propagation delay is low because the distance between a source

and a destination is usually not long. For this reason, the delay-bandwidth product for a WSN is much smaller than that for a Long Fat Network (LFN) and many of the options supporting LFNs are thus not suitable to WSNs. The various options included in current TCP implementations are suitable only for LFNs and they need to be modified for slow WSNs. This study describes each of these options and explains why they may need to be modified for WSNs. The experimental results show that as much as a 25% improvement in TCP's efficiency for the same amount of energy consumed can be achieved if certain modifications are made in the implementation of TCP code.

10.4.3.2 Reliable and Energy-Efficient Transport Protocol. Most traditional transport protocols cannot be applied in WSNs because they mostly rely on end-to-end and hop-by-hop retransmission for reliable data delivery. End-to-end retransmission makes a WSN not scalable because the number of sensor nodes in the network is very large and may be impossible for the receiving node (sink) to track thousands of connections from the reporting sensors. On the other hand, hop-by-hop retransmission is energy inefficient because wireless transmission is the largest energy consumption source in sensors. To address this problem, Djukic and Valaee [54] introduced Diversity Coded Directed Diffusion (DCDD), a reliable energy-efficient transport layer protocol on top of directed diffusion, for a WSN. In DCDD, the sink makes use of a number of receiving nodes named "prongs", which are connected to DCDD with reliable links. Multiple network paths and forward erasure codes (FEC) are used to increase reliability and improve energy efficiency of the network. Sensor nodes split their observations into many fragments with an erasure code and generate parity fragments using an FEC algorithm. Afterward, these fragments are distributed over the communication paths and sent to the sink. The sink is capable of reconstructing the packets (observations) if it receives part of the fragments, which (the part) is of the same size as their original observations. By using an ns-2 simulator, they examined the ability of the DCDD protocol to increase end-to-end reliability and reduce energy consumption in the network. Their simulations results indicated that the network using DCDD outperforms the network in which sensor nodes use only MAC retransmissions to increase reliability. Therefore, DCDD increases not only the energy efficiency but also the end-to-end reliability, and decreases the delay in the network.

10.4.3.3 Sensor Transmission Control Protocol. Most of existing transport layer protocols for WSNs assume that sensor nodes employ a particular network layer or MAC layer protocol. Thus, these protocols may not be applicable in many sensor applications. An ideal transport layer protocol should be able to support multiple applications in the same network, provide reliability, address congestion, reduce latency, and maximize throughput. For this purpose, Iyer et al. [55] proposes a sensor transmission control protocol (STCP), which is a reliable, scalable, and energy-efficient transport layer protocol that meets all the requirements. In this protocol, most of the control functionalities are

implemented at the base station. Each sensor node can be the source of multiple data flows with different characteristics (e.g., flow type, transmission rate, and reliability level). STCP offers congestion detection and avoidance, controlled reliability, and supports multiple applications in the same network. In Ref. [55], the impact of incorrect timers on the network is studied and it is verified that the latency induced is within a tolerable limit. In addition, it is also shown that STCP outperforms other relevant transport layer protocols, and can increase network lifetime through controlled reliability in different network scenarios.

10.5 SUMMARY

Power control is crucial for prolonging network lifetime and ensuring normal network operation of WSNs. This chapter introduced the fundamental concepts related to power control, discussed its major issues and challenges, and presented an overview of various power conservation mechanisms for WSNs. We classified power conservation mechanisms into two broad categories: passive and active, where the passive mechanisms are further classified into physical layer, MAC layer, and higher layer mechanisms, while the active mechanisms are further classified into MAC layer, network layer, and transport layer mechanisms. For each category, we gave an overview of typical mechanisms in terms of their objectives, characteristics, and performance. We hope that this chapter can help the readers to have a good understanding of the power control issues in WSNs and motivate further research on this critical issue.

REFERENCES

[1] I. Akyildiz et al., "Wireless sensor networks: a survey", *Computer Networks*, vol. 38, no. 4, Mar. 2002, pp. 393–422.

[2] K. Akkaya et al., "A survey on routing protocols for wireless sensor networks", *Ad-hoc Networks, Elsevier*, vol. 3, no. 3, May 2005, pp. 325–349.

[3] N. Pantazis, D. D. Vergados, and D. J. Vergados, "Power control schemes in tactical wireless sensor networks", in *Proceedings of the 12th European Wireless (EW'06) Conference*, Athens, Greece, Apr. 2006.

[4] N. Pantazis, D. J. Vergados, and D. D. Vergados, "Increasing intelligent wireless sensor networks survivability by applying energy-efficient schemes", *IFIP International Federation for Information Processing*, vol. 204, Springer, New York, 2006, pp. 657–664.

[5] N. Pantazis and D. D. Vergados, "A survey on power control issues in wireless sensor networks", *IEEE Communications Surveys & Tutorials*, vol. 9, no. 4, 4th Quarter 2007, pp 86–107.

[6] V. Raghunatan, C. Schurgers, S. Park, and M. B. Srivastava, "Energy-aware wireless Microsensor networks", *IEEE Signal Processing Magazine*, vol. 19, no. 2, Mar. 2002, pp. 40–50.

[7] R. H. Katz et al., "Mobile networking for smart dust", in *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'99)*, Seattle, WA, Aug. 1999, pp. 188–196.

[8] J. M. Rabaey et al., "Pico radio supports ad-hoc ultra low power wireless networking", *IEEE Computer*, vol. 33, no. 7, July 2000, pp. 42–48.

[9] K. Sohrabi et al., "Protocols for self-organization of a wireless sensor network", *IEEE Communications*, vol. 7, no. 5, Oct. 2000, pp. 16–17.

[10] C. Srisathapornphat and C.-C. Shen, "Coordinated power conservation for ad-hoc networks", in *Proceedings of 2002 IEEE International Conference on Communications (ICC'02)*, vol.5, New York, Apr.–May 2002, pp. 3330–3335.

[11] V. Kawadia and P. R. Kumar, "Principles and protocols for power control in ad hoc networks", *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 1, Jan. 2005, pp. 76–88.

[12] T. Issariyakul, E. Hossain, and D. I. Kim, "Medium access control protocols for wireless mobile ad-hoc networks: Issues and approaches", *Wireless Communications and Mobile Computing*, vol. 3, no. 8, Nov. 2003, pp. 935–958.

[13] F. A. Tobagi and L. Kleinrock, "Packet switching in radio channels: Part II- The hidden terminal problem in carrier sense multiple access modes and the busy tone solution", *IEEE Transactions on Communications*, vol. 24, no. 8, Aug. 1976, pp. 832–845.

[14] N. A. Pantazis, D. J. Vergados, D. D. Vergados, and C. Douligeris, "Energy efficiency in wireless sensor networks using sleep mode TDMA scheduling", *Elsevier Ad Hoc Networks*, vol. 7, no. 2, Mar. 2009, pp. 322–343.

[15] D. D. Vergados et al., "A new approach for TDMA scheduling in ad-hoc networks", in *Proceedings of the 10th IFIP International Conference on Personal Wireless Communications (PWC'05)*, Colmar, France, Aug. 2005, pp. 107–114.

[16] C. Imrich, H. Kim, and S. Ha, "Dynamic voltage scheduling technique for low-power multimedia applications using buffers", in *Proceedings of the International Symposium on Low Power Electronics and Design*, Huntington Beach, CA, June–July 2001, pp. 34–39.

[17] A. Sinha and A. P. Chandrakasan, "Dynamic power management in wireless sensor networks", *IEEE Design and Test of Computers*, vol. 18, no. 2, Mar.–Apr. 2001, pp. 62–74.

[18] V. Gutnik and A. P. Chandrakasan, "Embedded power supply for low-power DSP", *IEEE Transaction on VLSI Systems*, vol. 5, no. 4, Dec. 1997, pp. 425–435.

[19] A. Wang and A. P. Chandrakasan, "Energy efficient system partitioning for distributed wireless sensor networks", in *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing*, vol. 2, Salt Lake City, UT, July 2001, pp. 905–908.

[20] E. Shih, B. H. Calhoun, S. H. Cho, and A. P. Chandrakasan, "Energy-efficient link layer for wireless micro-sensor networks", in *Proceedings of IEEE Computer Society Workshop on VLSI*, Washington DC, May 2001, pp. 16–22.

[21] S. Doshi, S. Bhandare, and T. X. Brown, "An on-demand minimum energy routing protocol for a wireless ad-hoc network", *ACM SIGMOBILE Mobile Computing and Communication Review*, vol. 6, no. 3, Sept. 2003, pp. 50–66.

[22] S. Singh and C. Raghavendra, "PAMAS: Power aware multi-access protocol with signaling for ad-hoc networks", *ACM SIGCOMM Computer Communication Review*, vol. 28, no. 3, July 1998, pp. 5–26.

[23] V. Bharghavan, A. Demers, S. Shenkar, and L. Zhang, "MACAW: A media access protocol for wireless LANs", in *Proceedings of ACM SIGCOMM'94*, London, UK, Sept. 1994, pp. 212–225.

[24] P. Karn, "MACA—A new channel access method for packet radio", in *Proceedings of the ARRL CRRL Amateur Radio 9th Computer Networking Conference*, Redondo Beach, CA, Apr. 1990, pp. 134–140.

[25] J. Deng and Z. J. Hass, "Dual busy tone multiple access (DBTMA): A new medium access control for packet radio networks", in *Proceedings of International Conference on Universal Personal Communication*, Florence, Italy, Oct. 1998, pp. 973–977.

[26] C. L. Fullmer and G. L. Aceves, "Solutions to hidden terminal problems in wireless networks", in *Proceedings of ACM SIGCOMM'77*, Cannes, France Sept. 1997, pp. 39–49.

[27] C. L. Fullmer and G. L. Aceves, "Floor acquisition multiple access (FAMA) for packet radio networks", in *Proceedings of ACM SIGCOMM'95*, Cambridge, MA, Aug. 1995, pp. 262–273.

[28] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks", in *Proceedings of IEEE INFOCOM'02*, vol. 3, New York, NY, June 2002, pp. 1567–1576.

[29] S. C. Ergen and P. Varaja, "TDMA scheduling algorithms for sensor networks", available at http://paleale.eecs.berkeley.edu/~varaiya/papers_ps.dir/tdmaschedule.pdf, July 2005.

[30] M. Arumugam and S. Kulkarni, "Self-Stabilizing deterministic TDMA for sensor networks", in *Proceedings of the 2nd International Conference on Distributed Computing and Internet Technology (ICDCIT)*, LNCS3816, Bhubaneswar, India, Dec. 2005, pp. 69–81.

[31] S. S. Kulkarni and M. Arumugam, "SS-TDMA: A self-stabilizing MAC for sensor networks", *Sensor Network Operations*, Piscataway, NJ, IEEE Press, Apr. 2006.

[32] S. Gandham, M. Dawande, and R. Prakash, "Link scheduling in sensor networks: Distributed edge coloring revisited", in *Proceedings of IEEE INFOCOM'05*, Miami, FL, Mar. 2005, vol. 4, pp. 2492–2501.

[33] Y. Yu, B. Krishnamachari, and V. Prasana, "Energy-latency tradeoffs for data gathering in wireless sensor networks", in *Proceedings of IEEE INFOCOM'04*, Hong Kong, China, Mar. 2004, vol. 1, pp. 244–255.

[34] N. Trigoni, Y. Yao, A. Demers, J. Gehrke, and R. Rajaraman, "WaveScheduling: Energy-efficient data dissemination for sensor networks", in *Proceedings of the 1st International Workshop On Data Management For Sensor Networks*, Toronto, Canada, Aug. 2004, pp. 48–57.

[35] S. Cui, R. Madan, A. Goldsmith, and S. Lall, "Joint routing, MAC, and link layer optimization in sensor networks with energy constraints", in *Proceedings of 2005 IEEE International Conference on Communications (ICC'05)*, vol. 2, Seoul, South Korea, May 2005, pp. 725–729.

[36] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad-hoc wireless networks", in *Proceedings of ACM MobiCom'01*, Rome, Italy, July 2001, 481–494.

[37] S.-L. Wu, Y.-C. Tseng, and J.-P. Sheu, "Intelligent medium access for mobile ad-hoc networks with busy tones and power control", in *Proceedings of IEEE Journal on Selected Areas in Communications*, vol. 18, no. 9, Sept. 2000, pp. 1647–1657.

[38] J. Monks, V. Bharghavan, and W.-M. Hwu, "A power controlled multiple access protocol for wireless packet networks", in *Proceedings of IEEE INFOCOM'01*, vol. 1, Anchorage, AK, Apr. 2001, pp. 219–228.

[39] J. Chen, S.-H. G. Chan, Q. Zhang, W.-W. Zhu, and G. Chen, "PASA: Power adaptation for starvation avoidance to deliver wireless multimedia", *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 10, Dec. 2003, pp. 1663–1673.

[40] E. S. Jung and V. Nitin, "A power control MAC protocol for ad-hoc networks", in *Proceedings of ACM International Conference On Mobile Computing and Networking (MobiCom'02)*, Atlanta, GA, Feb. 2002, pp. 55–66.

[41] T. Nandagopal, T. F. K. Gao, and V. Bharghavan, "Achieving MAC layer fairness in wireless packet networks", in *Proceedings of ACM International Conference on Mobile Computing and Networking (MobiCom'02)*, Boston, MA, Aug. 2000, pp. 87–98.

[42] M. Cagalj, J. P. Hubaux, and C. Enz, "Minimum-energy broadcast in all-wireless networks: NP-completeness and distribution issues", in *Proceedings of ACM (MobiCom'02)*, Atlanta, GA, Sept. 2002, 172–182.

[43] S. Singh, M. Woo, and C. S. Raghavendra, "Power-aware routing in mobile ad-hoc networks", in *Proceedings of ACM MobiCom'98*, Dallas, TX, Oct. 1998, pp. 181–190.

[44] J. Gomez, A. T. Campbell, M. Naghshineh, and C. Bisdikian, "Power aware routing in wireless packet networks", in *Proceedings of the 6th IEEE International Workshop on Mobile Multimedia Communications*, San Diego, CA, *N*ov. 1999.

[45] F. Ye, A. Chen, S. Liu, and L. Zhang, "A scalable solution to minimum cost forwarding in large sensor networks", in *Proceedings of the 10th International Conference on Computer Communications and Networks (ICCCN'01)*, Scottsdale, AZ, Oct. 2001, pp. 304–309.

[46] R. Shah et al., "Energy aware routing for low energy ad-hoc sensor networks", in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC'02)*, Orlando, FL, Mar. 2002, pp. 350–355.

[47] G. Xing, C. Lu, Y. Zhang, Q. Huang, and R. Pless, "Minimum Power Configuration in wireless sensor networks", in *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc'05)*, Urbana-Champaign, IL, Apr. 2005, pp. 390–401.

[48] M. J. Hossain, O. Chae, M. Mamun-Or-Rashid, and C. S. Hong, "Cost-effective maximum lifetime routing protocol for wireless sensor networks", in *Proceedings of AICT/SAPIR/ELETE*, Lisbon, Portugal, July 2005, pp. 314–319.

[49] H. Kang and X. Li, "Power-aware sensor selection in wireless sensor networks", in *Proceedings of the 5th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN '06)*, Nashville, TN, Apr. 2006.

[50] L. Subramanian and R. H. Katz, "An Architecture for Building Self Configurable Systems", in *Proceedings of IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing*, Boston, MA, Aug. 2000, pp. 63–73.

[51] V. Tsaoussidis and H. Badr, "TCP-probing: Towards an error control schema with energy and throughput performance gains", in *Proceedings of International Conference on Network Protocols*, Osaka, Japan, Nov. 2000, pp. 12–21.

[52] J. Liu and S. Singh, "ATCP: TCP for mobile ad-hoc networks", *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 7, July 2001, pp. 1300–1315.

[53] S. Agrawal and S. Singh, "An experimental study of TCP's energy consumption over a wireless Link", in *Proceedings of the 4th European personal Mobile Communications Conference*, Vienna, Austria, Feb. 2001.

[54] P. Djukic and S. Valaee, "Reliable and energy efficient transport layer for sensor networks", in *Proceedings of the IEEE Global Telecommunications Conference (Globecom'06)*, San Francisco, CA, Nov.–Dec., 2006, pp. 1–5.

[55] Y. G. Iyer, S. Gandham, and S. Venkatesan, "STCP: A generic transport. layer protocol for wireless sensor networks", in *Proceedings of the 14th International Conference Computer Communications and Networks (ICCCN'05)*, San Diego, CA, Oct. 2005, pp. 449–454.

11

TRANSPORT PROTOCOLS AND QUALITY OF SERVICE

Chonggang Wang

University of Arkansas at Fayetteville, USA

Bo Li

Hong Kong University of Science and Technology, China

Kazem Sohraby

University of Arkansas at Fayetteville, USA

11.1 INTRODUCTION

Rapid advances in wireless communications, as well as microelectromechanical systems (MEMS) technologies have enabled the availability of cheap sensors and the deployment of large-scale wireless sensor networks (WSNs). A WSN is generally composed of one or more sinks (or base stations) and tens or thousands of sensor nodes scattered in a physical space. With the integration of information sensing, computation, and wireless communication capabilities, the sensor nodes can sense the physical information, process the sensed information, and report the processed information to the sink. The sink can query information and even control the behavior of the sensor nodes. A WSN can be used to monitor events

Wireless Sensor Networks: A Networking Perspective, Edited by Jun Zheng and Abbas Jamalipour

and collect data in some special environments where other networks are difficult or costly to be deployed. Therefore, WSNs are usually event-oriented and data centric, and have several distinctive features:

- *Unique Network Topology.* The sensor nodes are usually organized into a multihop star-tree topology, which could be either flat or hierarchical. The sink sitting at the root of the tree plays the role of information collection and relaying to exterior networks. Since the sensor nodes may be mobile, the wireless links may not always be good enough for establishing normal connection between neighboring nodes, and the sensor nodes may die due to depletion of energy, the topology of a WSN can be highly variable and dynamic.
- *Diverse Applications.* A WSN can be used to support diverse applications ranging from habitat monitoring and target tracking to security surveillance, and so on [1]. These applications may be interested in different sensory data, and therefore have different requirements on quality of service (QoS) and reliability. For example, the applications to monitor climate, for example, temperature and humidity, can tolerate certain delivery latency and packet loss; however, for the application to detect fire inside a building, the sensory data should be correctly reported as soon as possible.
- *Traffic Characteristics.* In a WSN, traffic usually flows upstream from the sensor nodes to the sink although the sink may occasionally generate certain downstream traffic for the purpose of control and management. The upstream traffic is a kind of many-to-one communications compared to that of the peer-to-peer (P2P) communications in ad hoc networks. According to the mechanism to trigger data transmission in sensor nodes, the traffic models of different applications can be classified into four types: *event-driven delivery*, *continuous delivery*, *query-driven delivery*, and *hybrid delivery*. For example, in event-driven delivery, the sensor nodes start event reporting only if the target events (e.g., the temperature is below zero) have been sensed. The sensory data for the events is usually very small in size, for example, a couple of 10 bytes or even one binary bit sometimes.
- *Resource Constraints.* Despite diverse commercially available sensor products [2], sensor nodes usually have limited resources, including low computation capability, small memory size, low communication bandwidth, and finite and unrechargeable battery. The limitation of resources narrows the designing space of various protocols and algorithms for WSNs.
- *Small Message Size.* Messages in WSNs usually have a small size compared with those in other networks. For this reason, there is usually no concept of segmentation in most applications of WSNs.

These distinctive features pose a big challenge on the design of a WSN that can meet application requirements and operate as long as possible. Usually, the QoS of an application and the lifetime of a network are contradictory to each

other. To address this problem, we need to carefully cope with the energy-conservation, security, QoS, and reliability issues by means of different network protocols and algorithms or combination of them. For example, a medium access control (MAC) protocol determines how packets are transmitted within 1-hop and therefore has an impact on energy efficiency and QoS. A routing protocol can improve reliability and QoS through optimal path selection. Lots of existing work focuses on MAC [3] and routing protocols [4]; however neither routing nor MAC protocols can independently address any of these problems satisfactorily.

Recently, transport protocols have received much attention for improving network performance in WSNs, especially for congestion control and reliability guarantee [5]. There are several reasons that transport protocols are necessary and important, which are described as follows:

1. Congestion is unavoidable in sensor networks, especially in wireless multimedia sensor networks [6], which can generate high-speed sensory data. Congestion leads to packet loss that impairs QoS and wastes energy, and thus may result in low QoS and low energy efficiency. As in traditional wired and wireless networks, MAC and routing protocols are not enough to deal with congestion. Transport protocols can be used to avoid or mitigate congestion and thus reduce packet loss. For example, if a transport protocol can detect congestion and remove it in time, packet loss due to congestion can be highly reduced. The reduced packet loss leads to a smaller number of retransmissions and further indirectly improves not only energy efficiency, but also the successful packet delivery ratio.
2. Transport protocols are necessary for guaranteeing reliability. Usually, packet loss occurs due to bit errors and/or buffer overflow. To guarantee reliability, it is important to recover lost packets to ensure that the successfully transmitted packets meet the requirements. Although retransmission at the link layer can recover packets due to bit errors, it can neither detect nor recover lost packets due to congestion. For this reason, transport protocols are needed. However, traditional transport protocols cannot be applied directly to WSNs because they do not take energy efficiency into consideration. In addition, transport protocols are helpful for guaranteeing fairness, which is in the sense that each senor node can report its sensory data with the same chance.

Therefore, transport protocols are very important for enhancing energy efficiency, QoS, and reliability because they can be used to avoid or mitigate congestion and thus reduce packet loss, to provide fairness in bandwidth allocation, and to guarantee end-to-end transmission reliability. However, traditional transport protocols for wired networks, for example, TCP (Transport Control Protocol) [7] and UDP (User Datagram Protocol) [8], cannot be directly used in WSNs, which will be explained in Section 11.2. Because transport protocols are very important and so far there is no full-fledged transport protocol for WSNs, the focus of this chapter is put on the design of transport protocols for WSNs.

This chapter also introduces the principles of traditional transport protocols, discuss several open issues, and present major existing transport protocols for WSNs. The remainder of the chapter is organized as follows. Section 11.2 briefly introduces traditional transport protocols, including their principles and disadvantages for WSNs. Section 11.3 discusses basic principles and design criteria in transport protocols for WSNs. Section 11.4 presents existing transport protocols for WSNs. Section 11.5 concludes the chapter and outlines a list of directions for further studies.

11.2 TRADITIONAL TRANSPORT PROTOCOLS

This section introduces the principles of traditional transport protocols and discuss the disadvantages of TCP and UDP.

11.2.1 Principles of Traditional Transport Protocols

The architecture of a computer communication network is often stratified into several different layers, including the physical, the data-link, the network, the transport, and the upper layers, for example, application layer. Each lower layer serves as a service provider transparently providing some services to its upper layer, the service customer, through so-called service access points (SAPs). For example, the data-link layer aims to provide a reliable link to the network layer. The network layer provides addressing service and routing service to the transport layer, which in turn provides end-to-end message transportation service to the upper layers. In this layered network model, the lower three layers sit in each intermediate network node or communication entity; however, the transport and the upper layers generally exist only in end points or hosts, and belong to end-to-end protocols.

The transport layer sits on the network layer. It utilizes the services provided by the network layer to enable end-to-end message transportation, where messages are fragmented to chains of segments at senders and reassembled into original messages at receivers, and does not concern with the underlying path carrying segments. A protocol at the transport layer is called a transport protocol, for example, TCP [7], UDP [8], and the Stream Control Transmission Protocol (SCTP) [9]. Both TCP and UDP are standard protocols that have been widely deployed in the Internet for many years. SCTP is a new transport protocol with the aim to support reliable signaling transmission.

Transport protocols can be generally classified into two types: connection oriented and connectionless. A connectionless protocol does not need to establish a connection before data is transferred and has only one phase: *data transmission*. In contrast, a connection-oriented protocol has three phases for each transmission process [10]: *connection establishment*, *data transmission*, and *disconnection*. Moreover, a transport protocol can be *responsive* (e.g., TCP) or *unresponsive* (e.g., UDP). A *responsive* protocol can adjust the source sending rate (e.g.,

increase or decrease), while an *unresponsive* protocol does not change. A transport protocol, especially a connection-oriented protocol, usually provides the following functions to the upper layers:

- *Orderly Transmission.* Since multiple paths may exist between a source and a destination, a packet that is sent earlier may arrive at the destination later. This phenomenon is called *disordered transmission*. A transport protocol may need to provide sequential transmission service for real-time applications. The common approach is to piggyback a *sequence number* in the header of each transmitted segment. In this way, the destination can sort the received segments based on the sequence number carried in each segment.
- *Flow Control and Congestion Control.* The hosts often have different characteristics, for example, the capacity of communication and computation. If the source transmits segments with a higher rate than the destination can receive, or in other words the source sending rate exceeds the bottleneck link bandwidth on the path between the source and the destination, congestion will occur and thus incur segment loss. Therefore, the transport layer still needs to provide flow control and congestion control service to coordinate the sending rate from the source to the destination.
- *Loss Recovery.* Congestion in a network leads to packet loss because a node has finite memory. Although the data-link layer can recover data loss caused by bit errors, it is unable to recover data loss caused by buffer overflow. For this reason, a transport protocol should support loss recovery so as to provide a loss-free virtual tunnel to the upper-layer applications, especially those loss-sensitive applications, for example, File Transfer Protocol (FTP).
- *QoS.* For real-time applications, for example, voice over IP (VoIP) that are delay critical and need to sustain a certain bandwidth, a transport protocol should provide low delay and high throughput under the constraints. For this purpose, the transport protocol can incorporate QoS design into flow control and congestion control.

Generally, a connectionless transport protocol, for example, UDP neither supports flow control, congestion control, or loss recovery, nor guarantees reliable end-to-end transmission. A connection-oriented protocol often provides richer services than a connectionless protocol. For example, TCP supports all the above listed functions to some extent.

11.2.2 Disadvantages of TCP and UDP

As two popular transport protocols, TCP and UDP have been broadly deployed in the Internet and wired networks. However, neither of them is a good choice

for WSNs. In what follows, we respectively discuss their disadvantages when used in WSNs:

- TCP is a connection-oriented protocol. Before data transmission, there is a three-way handshake interactive process. If and only if after a TCP connection has been established between a TCP sender and a TCP receiver, the TCP sender can begin to transmit data. In WSNs, the sensory data for event-based applications is only several bytes or so (a value of an interest). The three-way handshake process will be a big overhead compared to the small volume of data. Also, since a wireless link is error prone, the time to set up a TCP connection can be much longer than that in the Internet. Therefore, the data may become outdated after the TCP connection has been established.
- TCP assumes that segment loss is resulted from congestion and triggers window-based flow control and congestion control once it detects segment loss. This can incur that TCP unwisely reduces the sending rate under WSNs even if there is no congestion, and further leads to low throughput, especially under a multihop wireless environment. Therefore, it is hard for sensor nodes, especially those far away from the sink, to obtain enough bandwidth to support those applications that require continual data transmissions.
- The control congestion in TCP is end-to-end. This approach usually has a tardy response when congestion occurs, and will thus result in lots of segment dropping. The segment dropping wastes valuable energy and implies low energy efficiency.
- TCP uses end-to-end acknowledgment (ACK) and retransmission to guarantee reliability. The end-to-end approach not only takes longer time to recover lost segments, but also consumes much energy and reduces network lifetime.
- In WSNs, the sensor nodes may have different hops and different round-trip time (RTT) from the data sink. TCP in such an environment may cause unfairness and make the sensor nodes near the sink get more opportunities to transmit data, and therefore deplete their energy first. In this case, the whole network may become disjointed if no other mechanism is introduced.
- UDP is a connectionless transport protocol. But it is also unsuitable for WSNs considering that: (a) there is no flow and congestion control mechanism in UDP. If UDP is used for WSNs, it will cause lots of datagram dropping when congestion occurs. At this point, UDP is at least not energy efficient for WSNs; (b) UDP contains neither ACK mechanism, nor any reliability mechanism. The datagram loss can only be recovered by lower MAC protocols or upper layers, including the application layer.

Although many TCP based new protocols have been proposed, they are either only suitable for wireless environments with one wireless hop [11] or do not consider energy conservation [12]. Therefore, neither TCP nor UDP is suitable for WSNs.

11.3 TRANSPORT PROTOCOL DESIGN FOR WIRELESS SENSOR NETWORKS

A transport protocol runs over the network layer. It enables end-to-end message transmission, where a message may be fragmented into several segments at the transmitter and reassembled at the receiver. This protocol provides the following functions: orderly transmission, flow and congestion control, loss recovery, and possibly QoS (e.g., timing and fairness) guarantee. In WSNs, several new factors, for example, the convergent nature of upstream traffic and limited wireless bandwidth, can result in congestion, which may affect normal data exchange and lead to packet loss. Moreover, the impairments of wireless channels, for example, path loss and fading, introduce packet loss due to bit errors, which not only affects reliability, but also wastes energy. For these reasons, congestion and packet loss are two major problems that a transport protocol needs to cope with in a WSN. Next, we discuss the performance metrics and required functions of transport protocols for WSNs, as well as their design options.

11.3.1 Performance Metrics

In a WSN, a transport protocol should provide end-to-end reliability and end-to-end QoS in an energy-efficient manner. The performance of a transport protocol can be evaluated using different metrics, for example, energy efficiency, reliability, QoS (e.g., packet loss ratio or packet delivery latency), and fairness.

Energy Efficiency. A sensor node usually has limited energy. For this reason, it is most important for a transport protocol to keep high energy efficiency in order to prolong the network lifetime. Due to bit errors and/or congestion, packet loss is common in a WSN. For loss-sensitive applications, packet loss leads to retransmission and inevitably consumes additional energy. Therefore, packet loss is a primary factor that affects energy efficiency at the transport layer. If we define *retransmission distance* as the hop number from the node that requests retransmission to the node that retransmits lost packets, different retransmission mechanisms may have different retransmission distances. A mechanism with a longer retransmission distance consumes more energy. Therefore, retransmission distance is the second factor that affects energy efficiency. The third factor that affects energy efficiency is the use of control messages. A transport layer may use control messages to perform congestion control and loss recovery.

The transmission of control messages leads to transmission overheads and thus reduces energy efficiency more or less. Therefore, a transport protocol that has a smaller packet loss ratio with a shorter retransmission distance and uses fewer control messages can successfully transmit more packets with certain energy and is thus more energy efficient. It is not the purpose of this chapter to give out a standard definition of energy efficiency, but it is obvious that a network is energy efficient if it is able to run for a long period of time while meeting the requirements of its applications.

Reliability. For different applications, different levels of reliability may be required. There are several definitions of reliability in the literature [13,14]. According to the sensitivity of an application to packet loss, two types of reliability can be classified: *packet reliability* and *event reliability*. *Packet reliability* means that the application is very loss sensitive and requires the successful transmission of each packet. One example of such applications is downstream code distribution or queries. *Event reliability* stands for the requirement of loss-tolerant applications that allow certain packet loss. For example, the sensor nodes with digital camera can be used to send images to the sink. Since images are somewhat loss tolerant, the sink does not need to correctly receive every packet, but only a certain percentage of packets.

According to the scope of the destinations of downstream traffic from the sink to the sensor nodes, three kinds of packet reliability or message delivery schemes are defined in Ref. [14]: (1) *Message delivery to all sensor nodes*: This requires the messages to be successfully delivered to all sensor nodes. (2) *Message delivery to the sensor nodes in a subarea*: In this case, the messages only need to be successfully delivered to the sensor nodes in a specific subarea. (3) *Message delivery to cover the entire sensor area or a subarea*: It means that the messages need to be successfully received by the sensor nodes that can cover the entire sensor area or a subarea. In reality, it is possible that sensor nodes are installed with different types of sensors, for example, the humidity sensor, the temperature sensor, and the wind speed sensor. Sometimes the sink needs to configure or control only a specific kind of sensors, and therefore only the sensor nodes with such specific sensors need to receive the messages. Therefore if sensor nodes have different kinds of sensors, we still need to define the fourth kind of reliability: (4) *Message delivery to the sensor nodes with a particular kind of sensors*.

QoS. Sensor nodes can be used to transmit continuous images for target tracking. In this scenario, the sensor nodes generate high-speed data flows and need larger bandwidth than event-based applications. For some delay-sensitive applications, for example, commands to task sensor nodes, a network must guarantee real-time data transmission. As mentioned before, some applications are loss sensitive and cannot stand packet loss or require a very small packet-loss ratio. Therefore, a transport protocol may need

to support traditional QoS in terms of throughput, packet delivery latency, and packet-loss ratio [6,15]. The only difference is that in a WSN these QoS metrics should be guaranteed in a highly energy-efficient way.

Fairness. Sensor nodes are usually scattered in a geographical area to collect information. Due to the many-to-one convergent nature of upstream traffic, it is much more difficult for the sensor nodes far away from the sink to successfully transmit their sensory data to the sink. In order to let the sink have full information on the entire sensed area, a transport protocol should provide fair bandwidth allocation among all sensor nodes so that the sink can get the same number of packets from each sensor node during a period of time [16].

11.3.2 Congestion Control

There are mainly two reasons that result in congestion in a WSN. The first is the packet arrival rate exceeding the packet service rate. This is more likely to occur at the sensor nodes closer to the sink because they usually carry more combined upstream traffic. The second reason is contention, interference, and the bit error rate on a link, which can result in congestion on the link.

In a WSN, congestion has a direct impact on energy efficiency and application QoS. For example, congestion can cause buffer overflow that may lead to larger queuing delay and higher packet loss. Not only can packet loss degrade reliability and application QoS, but also waste the limited energy of a node. Congestion can also degrade link utilization. Moreover, if a contention-based link protocol, for example, Carrier Sense Multiple Access (CSMA), is used to share the radio resources, it can result in transmission collision and link-level congestion, which will in turn increase packet service time and waste energy. Therefore, congestion must be efficiently controlled, either avoided or mitigated. Typically, there are three mechanisms that can deal with this problem: *congestion detection*, *congestion notification*, and *congestion mitigation and avoidance*.

11.3.2.1 Congestion Detection. In TCP, congestion is observed or inferred at the end nodes based on a timeout or redundant acknowledgment. In a WSN, however, it is preferred to use proactive mechanisms. A common mechanism is to use queue length [17,18], packet service time [16], or the ratio of packet service time over packet interarrival time at the intermediate nodes [19]. If a network uses a CSMA like MAC protocol, channel loading can be measured and used as an indication of congestion. Therefore, measurement can be used as a means for detecting congestion as in Ref. [18].

11.3.2.2 Congestion Notification. After detecting congestion, a transport protocol needs to propagate the congestion information from the congested node to its upstream nodes or the source nodes that contribute to congestion. The information can be transmitted using, for example, a single binary bit, called congestion notification (CN) bit in Refs. [13,17,18], or more information, for

example, allowable data rate as in Ref. [16], or the congestion degree as in Ref. [19].

The approaches to disseminate congestion information can be categorized into *explicit congestion notification* and *implicit congestion notification*. The *explicit congestion notification* uses special control messages to notify the involved sensor nodes of congestion information. For example, in Ref. [18], an intermediate sensor node will broadcast a suppression message upstream toward the source when it perceives congestion. The node receiving the suppression message will continue to relay the message toward the source unless the suppression message has traversed for a certain hops, called *depth of congestion* [18]. In contrast, the *implicit congestion notification* does not need any additional control message to propagate congestion information. This approach usually piggybacks congestion information on normal data packets. For example, in Ref. [13], a sensor node sets a CN bit in the header of its data packets. After receiving these packets with the CN bit, the sink can know if congestion will occur in the next reporting interval. In Refs. [17,18], the *implicit congestion notification* is performed hop-by-hop. The intermediate sensor node that detects congestion piggybacks congestion information on the data packets to be forwarded. Taking the advantage of the broadcast nature of wireless channels, the child nodes of this intermediate node can overhear the congestion information whenever it is forwarding data packets. In the *implicit congestion notification*, transmission of an additional control message is avoided and energy efficiency can therefore be improved.

11.3.2.3 Congestion Mitigation and Avoidance. There are two general approaches to mitigate and avoid congestion: *network resource management* and *traffic control*. The first approach tries to increase network resources (e.g., bandwidth) to mitigate the congestion when congestion occurs. In a wireless network, power control and multiple radio interfaces can be used to increase bandwidth and mitigate congestion. For example, the virtual sinks in Siphon [20] have two radio interfaces: one primary low-power mote radio with smaller bandwidth and another long-range radio with larger bandwidth. When congestion occurs, the long-range radio is used as a shortcut or "siphon" to mitigate the congestion. With this approach, it is necessary to guarantee precise and exact network resource adjustment in order to avoid overprovided resources or underprovided resources. However, this is a hard task in wireless environments. Unlike the approaches based on network resource management, traffic control implies controlling congestion through adjusting the traffic rate at source nodes or intermediates nodes. This approach is helpful to saving network resources, and is more feasible and efficient when exact adjustment of network resources becomes difficult. Most existing congestion control protocols belong to this type. According to the control behavior, there are two general approaches for traffic control in WSNs: end-to-end and hop-by-hop. The end-to-end control can impose exact rate adjustment at each source node and simplify protocol design at intermediate nodes; however, it results in slow response and relies highly on RTT. In contrast, the hop-by-hop congestion control has faster response. However, it is usually difficult to adjust

the packet forwarding rate at intermediate nodes mainly because the packet forwarding rate is dependent on the MAC protocol and can be variable. For traffic control approaches, a sensor node can usually adjust its transmission rate upon receiving a congestion indication. If a single CN bit is used, the Additive Increase Multiplicative Decrease (AIMD) scheme or its variants are usually applied as in Refs. [13] and [18]. On the other hand, if additional congestion information is available, accurate rate adjustment can be applied as in Refs. [16] and [19].

11.3.3 Loss Recovery

In wireless environments, both congestion and bit errors can cause packet loss, which would degrade end-to-end reliability and QoS, and further decrease energy efficiency. Other factors that result in packet loss include node malfunction, incorrect or outdated routing information, and energy depletion. In order to address this problem, one can increase the source sending rate or introduce retransmission-based loss recovery. The former approach, which is also used in Event-to-Sink Reliable Transport (ESRT) [13], is effective for guaranteeing event reliability for event-driven applications that require no packet reliability; however, this approach is not energy efficient compared to loss recovery. Loss recovery is more active and energy efficient, and can be performed at both the link layer and the transport layer. At the link layer, loss recovery is performed on a hop-by-hop basis, while at the transport it is usually done on an end-to-end basis. In what follows, we introduce a loss recovery approach that consists of two phases: loss detection and notification, and retransmission recovery.

11.3.3.1 Loss Detection and Notification. Since packet loss can be far more common in WSNs than in wired networks, a loss detection mechanism has to be carefully designed. A common mechanism is to include a *sequence number* in each packet header. The continuity of *sequence numbers* can be used to detect packet loss. Loss detection and notification can be performed either *end-to-end* or *hop-by-hop*. In the end-to-end approach, the end points (destination or source) are responsible for loss detection and notification as in TCP. In the hop-by-hop approach, intermediate nodes detect and notify packet loss.

For several reasons, the end-to-end approach is not very effective for WSNs. First, the control messages that are used for end-to-end loss detection requires a return path consisting of several hops, which is not energy efficient; Second, control messages travel through multiple hops and could be lost with a high probability due to either link errors or congestion; Third, the end-to-end loss detection approach inevitably leads to end-to-end retransmissions for loss recovery. However, the end-to-end retransmission consumes more energy than hop-by-hop retransmission.

In the hop-by-hop approach, which can be performed at the transport layer, link layer, or both layers, a pair of neighboring nodes is responsible for loss detection, and only requires local retransmission, which is more energy efficient

compared to the end-to-end approach. The hop-by-hop loss detection approach can further be categorized into *receiver based* or *sender based* depending on where packet loss is detected. In the *sender-based* loss detection, the sender detects packet loss either on a timer-based or overhearing mechanism. In the timer-based detection, a sender starts a timer each time it transmits a packet. If it does not receive an acknowledgment from the targeted receiver before the timer expires, it infers the packet has been lost. Taking advantage of the broadcast nature of wireless channels, the sender can listen to the targeted receiver to see if the packet has been successfully forwarded, and passively detect packet loss in an indirect manner.

In the *receiver-based* loss detection, a receiver infers packet loss when it observes out-of-sequence packet arrivals. There are three ways to notify the sender: ACK (acknowledgment), NACK (negative ACK), and IACK (implicit ACK). Both ACK and NACK rely on special control messages, while IACK [21] piggybacks ACK in the header of a packet. In IACK, if a packet is overheard being forwarded again, it implies that the packet has been successfully received, and therefore acknowledged simultaneously. Therefore, IACK avoids control message overhead and is more energy efficient. However, the application of IACK depends on whether sensor nodes have the capability to overhear the physical channel. In case that the transmission is corrupt or the channel is not bidirectional or sensor nodes access the physical channel using TDMA based protocols, IACK may not be feasible.

Loss detection and notification can also pinpoint the reason for packet loss, which can be further used to improve system performance. For example, if the packet loss is caused by buffer overflow, the source nodes need to reduce the sending rate. However, if the packet loss is resulted from channel errors, it is unnecessary to reduce the sending rate in order to maintain high link utilization and throughput.

11.3.3.2 Retransmission Recovery. Retransmission of lost or damaged packets can also be performed either end-to-end or hop-by-hop. In the end-to-end retransmission, the source performs the retransmission. In the hop-by-hop retransmission, an intermediate node that intercepts loss notification searches its local buffer. If it finds a copy of the lost packet in the buffer, it retransmits it. Otherwise, it relays the loss information upstream to other intermediate nodes.

If we define the node with a cached packet as *cache point* and the node where the lost packets are detected as *loss point*, the hop number between them can be referred to as *retransmission distance*. The retransmission distance is an indication of retransmission efficiency in terms of energy consumed in the process of retransmission. In the end-to-end retransmission, for example, in TCP, the cache point is the source node. However, in the hop-by-hop retransmission, the cache point could be the predecessor node of the loss point. The end-to-end retransmission has a longer retransmission distance, while the hop-by-hop retransmission is more energy efficient. However, the hop-by-hop retransmission requires intermediate nodes to cache packets. The end-to-end retransmission allows for

application-dependent variable reliability levels like that realized by ESRT. In contrast, the hop-by-hop retransmission is preferred if 100% packet reliability is required, although some applications in WSNs, for example, event-driven applications, may not require 100% reliability from a sensor node. However, hop-by-hop loss recovery cannot assure message delivery in the presence of a node failure.

Since both end-to-end and hop-by-hop retransmissions require caching transmitted packets at cache points for a possible future request for retransmission, a question would be how long a cache point should buffer a transmitted packet. This is especially important if the cache point does not receive an acknowledgment. For end-to-end retransmission, the cache duration should be close to the round-trip-time (RTT). In a wireless system that uses NACK based acknowledgments, NACK messages could be lost or corrupted on the reverse channel and the destination would be required to send NACK more than once. In this case, the source nodes need to buffer a packet for a time duration that is longer than RTT. For hop-by-hop retransmission, the cache duration is only affected by the total local packet service time and 1-hop packet transmission time.

Next, we discuss several problems related to hop-by-hop retransmission in WSNs. The first problem is when to trigger retransmission. The retransmission can be triggered immediately upon the detection of a packet loss. This results in a shorter delay, which is desirable by time-sensitive applications. However, if the packet loss is caused by congestion, immediate retransmission could aggravate the congestion situation and cause more packet losses. The second problem is where to cache the transmitted packets. In the hop-by-hop retransmission, each packet could be cached at each intermediate node. Given the limited memory in sensor nodes, packets may only need to be cached at selected nodes. The central problem is how to distribute cached packets among a set of nodes. The solutions, for example, in distributed TCP cache (DTC) [22], balance the buffer constraints and retransmission efficiency by using probability-based selection for cache points. In order to optimize retransmission efficiency, another possible solution is to cache packets at the intermediate node that is closer to the potential congested node, where packet loss is more likely to occur.

11.3.4 Design Guidelines

In order to design an efficient transport protocol, several factors must be taken into consideration, including the network topology, diversity of applications, traffic characteristics, and resource constraints. The two most significant constraints in WSNs are energy and fairness among different geographically deployed sensor nodes. A transport protocol should provide high energy efficiency and flexible reliability, as well as QoS in terms of throughout, packet-loss rate and end-to-end delay if necessary.

Therefore, transport protocols for WSNs should have components including congestion control and loss recovery because these have a direct impact on energy efficiency, reliability, and application QoS. There are generally two

approaches to perform this task. The first approach is to design separate protocols or algorithms, respectively, for congestion control and loss recovery. Most existing protocols use this approach and address congestion control or reliable transport separately. With this separate and usually modular design, applications that need reliability can invoke only a loss recovery algorithm, or invoke a congestion control algorithm if they need to control congestion. For example, COngestion Detection and Avoidance (CODA) [18] is a congestion control protocol while Pump slowly Fetch Quickly (PSFQ) [23] provides reliable transport. The joint use of these two protocols may provide full functionality required by the transport protocols for WSNs. Second, the design considerations should take into account a fully fledged transport protocol that provides congestion and loss control in an integrated way. For example, the Sensor Transmission Control Protocol (STCP) [24] implements both congestion control and flexible reliability in a single protocol. For different applications, STCP offers different control functionalities to both guarantee application requirements and improve energy efficiency.

The first approach divides a problem into several subproblems and is more flexible to implement. The second approach may need to optimize congestion control and loss recovery because loss recovery and congestion control in WSNs are often correlated. For example, congestion on contention-based wireless links can lead to packet loss. The combination of CODA and PSFQ may achieve both congestion control and reliability. However, it is not well documented in the literature that such control protocols can be seamlessly integrated in an energy-efficient way. We believe there is a trade-off between the architectural or modular design (the first approach) and the integrated design with performance optimization (the second approach). The same trade-off could also be observed between the traditional protocol stack and the cross-layer optimization.

11.4 TRANSPORT PROTOCOLS FOR WIRELESS SENSOR NETWORKS

There are several transport protocols already proposed for WSNs. Some of them address either congestion or reliability, while the others address both of them. We categorize these transport protocols into three types: (1) protocols for congestion control; (2) protocols for reliability; (3) protocols for both congestion control and reliability, as shown in Fig. 11.1. Due to the limitation of space, we only give a brief overview of these protocols in this section. Readers are referred to related references for more details.

11.4.1 Protocols for Congestion Control

Several congestion control protocols have been proposed for upstream convergent traffic in WSNs. They differ in congestion detection, congestion notification, or rate-adjustment mechanisms (see Table 11.1).

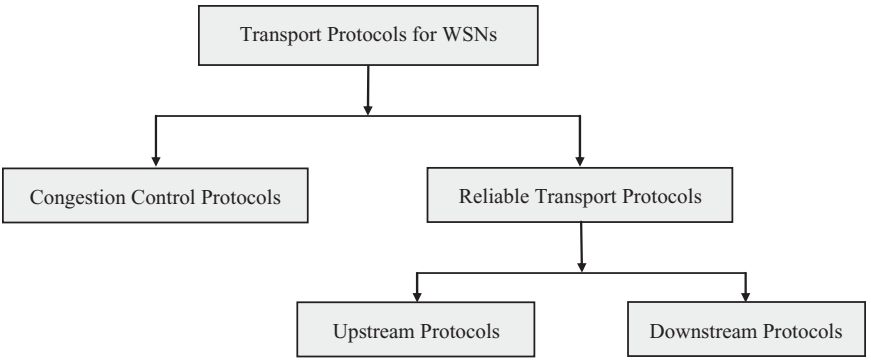


Fig. 11.1 Classification of existing transport protocols for WSNs.

TABLE 11.1 Congestion Control Protocols for WSNs

Features / Protocols	Reference	Congestion Detection	Congestion Notification	Congestion Mitigation
STCP	[24]	Queue length	Implicit	AIMD-like end-to-end rate adjustment
Fusion	[17]	Queue length	Implicit	Stop-and-start hop-by-hop rate adjustment
CODA	[18]	Queue length and channel status	Explicit	AIMD-like end-to-end rate adjustment
CCF	[16]	Packet service time	Implicit	Exact hop-by-hop rate adjustment
PCCP	[19]	Packet interarrival time and packet service time	Implicit	Exact hop-by-hop rate adjustment
ARC	[21]	Packet forwarding status	Implicit	AIMD-like hop-by-hop rate adjustment
Siphon	[20]	Queue length and application fidelity	N/A[a]	Traffic redirection
Trickle	[25]	Number of received packets	N/A[a]	Polite gossip

[a]Not available = N/A.

11.4.1.1 Fusion. Fusion is a hop-by-hop congestion control protocol for upstream traffic in WSNs [17]. In fusion, congestion detection is based on queue length and uses implicit congestion notification by means of setting a CN bit in the header of each outgoing packet. Due to the broadcast nature of the wireless channel, the neighboring nodes of a congested node can overhear such CN bits. Once getting the CN bit, a neighboring node stops forwarding packets to the congested node so as to eliminate congestion quickly. However, this kind of rate adjustment is non-smooth, which may affect link utilization and fairness more or less.

11.4.1.2 Congestion Detection and Avoidance. CODA is a congestion control protocol proposed in Ref. [18]. It detects congestion based on current buffer occupancy and wireless channel load. It uses a particular suppression message to explicitly notify whether there is congestion or not to the upstream nodes. After receiving such a suppression message, the upstream nodes will multiplicatively reduce their sending rates. On the other hand, the upstream neighboring nodes will linearly increase their sending rate if they do not receive any suppression message over a period of time, which is actually an AIMD like rate adjustment. CODA employs a closed-loop end-to-end approach, which is also AIMD like. It may result in decreased reliability, especially under scenarios with sparse sources, and/or high data rates [18]. The suppression message and ACK control message used in CODA consumes additional energy and bandwidth.

11.4.1.3 Congestion Control and Fairness. We refer Congestion Control and Fairness (CCF) to a hop-by-hop transport protocol proposed in Ref. [16], which indirectly detects congestion using packet service time and uses implicit congestion notification like Fusion [17]. Specifically, each intermediate node i first deduces its local service rate r based on the measured packet service time, which means the total time from the instant when receiving a packet at a node from the upper layer to the instant when this packet is successfully transmitted to the next node. Then, it calculates $r/[N(i) + 1)]$ as the sending rate of each of its child nodes and piggybacks such information in each of its outgoing packets, where $N(i)$ is the child node number of node i. In order to further improve fairness, node i in CCF still maintains $N(i) + 1$ separate subqueues, respectively, for its child nodes and itself, and employs two scheduling algorithms to fairly schedule them. The two scheduling algorithms, Probabilistic Selection (PS) and Epoch-Based Proportional Selection (EPS), may increase packet delivery latency although they improve fairness [16]. In addition, the rate adjustment in CCF will lead to low link utilization if some nodes have smaller traffic than $r/[N(i) + 1)$, while others have enough traffic.

11.4.1.4 Priority-Based Congestion Control Protocol. Priority-Based Congestion Control Protocol (PCCP) is a transport protocol recently proposed for WSNs [19]. The node priority in PCCP is assumed to be application dependent and could be configured and updated by the sink. First, as a hop-by-hop

congestion control protocol, it jointly uses packet interarrival time and packet service time to estimate current local congestion degree in each intermediate sensor node. The combined use of the packet interarrival and the packet service time not only precisely calculates congestion degree, but effectively helps differentiate the reason of packet loss occurrence in wireless environments because the packet interarrival time (or service time) may become small (or large) if congestion is going to occur. Second, PCCP uses implicit congestion notification and avoids the overhead caused by control messages. Third, PCCP introduces a flexible priority-based rate control based on the measured congestion degree. This flexible rate control allows the nodes with more traffic to increase their sending rate when some nodes have smaller traffic than allowed so as to keep high throughput, and/or allocates more bandwidth to the nodes with a higher priority so as to guarantee priority-related fairness. In contrast, in order to guarantee that the sink gets the same number of packets from each node, CCF employs a work-conserving fair packet scheduling algorithm, which, however, could cause low throughput when some nodes have small traffic even if others have more. More comparisons between CCF and PCCP can be found in Ref. [19].

11.4.1.5 Adaptive Rate Control. Adaptive Rate Control (ARC) is an LIMD like algorithm [21]. In ARC, whenever an intermediate node overhears the successful packet forwarding by its parent node, it will increase its sending rate by a constant α. Otherwise, it will multiply the sending rate by a factor β, where $0 < \beta < 1$. In order to guarantee fairness between source traffic and transit traffic (or called route-thru traffic), each node runs independent rate adjusting for source traffic and transit traffic, respectively. The transit traffic is configured with bigger values of α and β. In ARC, there is neither explicit congestion detection nor explicit congestion notification. It avoids the use of control messages. However, the LIMD like rate adjustment cannot exactly adjust the sending rate and inevitably may introduce either low link utilization or high packet loss more or less to a certain extent.

11.4.1.6 Siphon. Siphon [20] is a traffic redirecting protocol, which manages upstream traffic overload with the novel introduction of multiradio virtual sinks (VS). The virtual sinks are supposed to be installed with at least two radio interfaces: One is a low-power mote radio and the other is a long-rage radio, for example, IEEE 802.11. The primary mote radio is used to connect sensor nodes, while the secondary powerful radio can be used to connect other virtual sinks or even the physical sink that provides a gateway to the Internet. In addition to using the same way in CODA for detecting congestion, Siphon also employs a "post-facto" mechanism, through which the physical sink measures the perceived application fidelity and infers congestion accordingly. When congestion occurs, Siphon triggers traffic redirection from sensor nodes to the virtual sinks, which in turn forward the traffic using the long-range radio to other virtual sinks or even the physical sink. As a result, congestion can be mitigated quickly. The virtual sinks actually provide effective shortcuts for data delivery under traffic congestion.

11.4.1.7 Trickle. Trickle [25] is a controlled broadcasting protocol designed for propagating and maintaining code updates in WSNs and therefore operates in the direction of downstream. Trickle uses a local "Polite Gossip" to exchange code data among neighboring nodes. With Polite Gossip, each node tries to broadcast a summary of its data periodically so that all sensor nodes can get the same updates. In each period, each node can "politely" suppress its own broadcasting if the number of the same metadata, which this node receives from neighboring nodes, exceeds a threshold. On the other hand, if the nodes receive a new code or metadata, they can shorten the broadcast period, which thus leads to earlier broadcast of the new code. Trickle independently runs in each node and introduces no additional control overhead.

Among all these protocols, Fusion and CODA detect congestion based on queue length at intermediate nodes, while CCF infers congestion based on packet service time. PCCP calculates a congestion degree as the ratio of packet interarrival time and packet service time. Siphon [20] uses the same approach as in CODA to infer congestion. In addition, it can detect congestion based on the perceived application fidelity at the sink. CODA uses explicit congestion notification, while the others [16,17,19] use implicit congestion notification. In ARC, there is no congestion detection or notification. ARC maintains two independent sets of α and β, respectively, for source traffic and transit traffic to guarantee fairness. In contrast, Fusion controls congestion in a stop-and-start non-smooth manner. In Fusion, neighboring nodes stop forwarding packets to the congested node immediately when congestion is detected and notified. CODA adjusts the sending rate similar to AIMD, while CCF and PCCP use an exact rate adjustment algorithm. Compared to CCF, PCCP provides priority-based fairness and overcomes the drawbacks of using work-conserving scheduling. However, there is no rate adjustment in Siphon. When congestion occurs, Siphon redirects traffic to virtual sinks, which, in addition to the primary low-power mote radio, has another long-rage radio used as a shortcut or "siphon" to mitigate congestion. Trickle [25] uses Polite Gossip to control traffic. In Trickle, each node tries to broadcast a summary of its data periodically. In each period, a node can "politely" suppress its own broadcasting if the number of the same metadata this node receives from its neighboring nodes exceeds a threshold. On the other hand, if the nodes receive new code or metadata, they can shorten the broadcast period, and therefore broadcast the new code sooner. In Trickle, metadata is used to describe the code that sensor nodes use, which is usually smaller in size than the code itself.

11.4.2 Protocols for Reliability

There are several transport protocols already proposed for reliability, which are shown in Table 11.2. Some of them examine upstream reliability [13,24,26,27], while the others investigate downstream reliability [14,23]. In the upstream direction, ESRT [13] discusses the fidelity of an event stream and only guarantees *event reliability* through end-to-end source rate adjustment. In contrast, Reliable

TABLE 11.2 Reliable Transport Protocols for WSNs

Protocols / Features	End-to-End		Hop-by-Hop			
	STCP[a]	ESRT[b]	RMST[c]	RBC[d]	GARUDA[e]	PSFQ[f]
Direction	Upstream	Upstream	Upstream	Upstream	Downstream	Downstream
Loss Detection and Notification	ACK and NACK	No	NACK	IACK	NACK	NACK
Loss Recovery	End-to-end	No	Hop-by-hop	Hop-by-hop	Hierarchical recovery	Hop-by-hop
Reliability	Event and packet reliability	Event reliability	Packet reliability	Packet reliability	Packet reliability	Packet reliability

[a]Reference [24].
[b]Reference [13].
[c]Reference [26].
[d]Reference [27].
[e]Reference [14].
[f]Reference [23].

Multi-Segment Transport (RMST) [26] and Reliable Bursty Convergecast (RBC) [27] provide *packet reliability* through hop-by-hop loss recovery. In the downstream direction, traffic is multicast or one-to-many. The explicit loss detection and notification has the same problem of control message implosion as that in conventional reliable IP multicast. However, existing protocols for reliable IP multicast, for example, GARUDA [14] and PSFQ [23], do not consider several distinctive features of WSNs, especially the resource constraints and application diversity. Therefore, these protocols are not feasible for WSNs. Both GARUDA and PSFQ use NACK based loss detection and notification, and local retransmission for loss recovery, but they employ different mechanisms to provide scalability.

11.4.2.1 Reliable Multi-Segment Transport. Reliable Multi-Segment Transport [26] is a transport protocol that guarantees the successful transmission of each packet in the upstream direction. It jointly uses a selective NACK and timer-driven mechanism for loss detection and notification. In RMST, intermediate nodes can operate in a cache or non-cache mode for retransmission. For the cache mode, the missing packets can be recovered hop-by-hop at intermediate nodes. If an intermediate node fails to find the missing packets or it works in the non-cache mode, it will forward the received NACK upstream toward the source node. RMST introduces control message overhead.

11.4.2.2 Reliable Bursty Convergecast. Reliable Bursty Convergecast [27] is a hop-by-hop implicit ACK based reliable transport protocol with caching at each hop. It aims to guarantee reliable and real-time upstream convergecast. First, RBC uses implicit ACK and employs a windowless block-acknowledge mechanism, which detects and notifies packet loss more reliably and energy efficiently. A receiver piggybacks information of all successfully received packets in the header of each data packet to be further forwarded, and a sender can listen to get such information and start retransmission if required. If the sender does not get the expected information for a period of time, the timer set for each transmitted packet will expire and the sender will start hop-by-hop retransmission. In RBC, the value of a retransmission timer is dependent on the queue length at the next-hop node. Second, RBC manages retransmission in two ways to avoid retransmission-caused congestion: intra-node packet scheduling and inter-node packet scheduling. With the intra-node scheduling, each node maintains virtual queues to give a higher priority to the packets with the smaller number of retransmissions. The packets in the same virtual queue have the same number of retransmissions. With the inter-node scheduling, RBC employs a differentiated contention control to allow a node with more packets in buffer to have a higher priority so that it can seize the channel with more chances.

11.4.2.3 Pump Slowly Fetch Quickly. Pump Slowly Fetch Quickly [23] is a hop-by-hop and downstream transport protocol proposed for WSNs, which

aims to reliably distribute data from the sink-to-sensor nodes by pacing data at a relatively slow speed, but allowing nodes that experience data loss to fetch (or recover) any missing packet directly from immediate neighbors quickly. Therefore, it is a kind of hop-by-hop retransmission-based local recovery. The motivation of PSFQ is to minimize loss recovery cost through localized loss recovery with loose delay bounds. It contains three components: pump operation, fetch operation, and report operation. First, the sink slowly and periodically broadcasts packets with such fields as file ID, file length, sequence number, TTL, and report bit to its neighbors until all the data fragments have been sent out. Second, a sensor node goes into the fetch mode once a sequence number gap in a file fragment is detected and further sends a NACK control message on the reverse path so as to recover the missing fragment. The NACK message will not be relayed unless the number of times the same NACK is heard exceeds a predefined threshold while the missing segments requested by the NACK message are no longer retained in a sensor node. Third, the sink can make sensor nodes feedback information on data delivery status to it through a simple and scalable hop-by-hop report mechanism. PSFQ is unable to detect the loss of a single packet because it uses NACK based loss detection. The slow pump or data distribution may cause big latency. Moreover, the overhead of NACK control messages reduces energy efficiency more or less. But PSFQ can be configurable to use all the bandwidth and thus to overcome the delay caused by the slow pump.

11.4.2.4 GARUDA. GARUDA [14] is a reliable transport protocol that guarantees downstream reliability in WSNs. It constructs a two-tier architecture for packet retransmission. The sensor nodes with $3i$-hops from the sink are selected as core sensor nodes if none of their neighboring nodes is a core sensor node, where i in a non-negative integer. The core sensor nodes constitute the first tier, and the others form the second tier. Each non-core sensor node chooses a nearby core node as its core node, where it can recover lost packets. GARUDA uses NACK control messages for loss detection and notification. The loss recovery is realized through two phases: (1) loss recovery among core sensor nodes and (2) loss recovery between non-core sensor nodes and their core nodes. Therefore, the retransmission for loss recovery in GARUDA looks like a hybrid scheme between hop-by-hop and end-to-end. GARUDA still employs a repeated WFP (wait for first packet) pulse transmission to guarantee the success delivery of a single or first packet, and the pulse transmission is additionally used to compute hop number and select core sensor nodes in order to establish the two-tier architecture. GARUDA can provide diverse destination-based reliability, but it requires additional NACK control messages as RMST.

11.4.3 Protocols for Congestion Control and Reliability

In Table 11.2, STCP [24] and ESRT [13] are transport protocols that are able to provide both congestion control and reliability.

11.4.3.1 Sensor Transmission Control Protocol. Sensor Transmission Control Protocol [24] is a generic end-to-end upstream transport protocol for WSNs that is able to provide both congestion control and reliability. In STCP, an intermediate sensor node employs a RED (random early detection) like algorithm to detect congestion based on queue length. If local congestion occurs, an intermediate sensor node will set a bit in normal data packets to be forwarded by it. The sink can capture such bits, and notify the source nodes that send the packets through such intermediate nodes by using ACK or NACK messages. The source nodes can either choose another route or reduce their data rates to mitigate the congestion in the network. It is a kind of network-assisted but end-to-end congestion control, sharing certain similarity with the combined use of the traditional TCP, RED with marking enabled, and the explicit congestion notification (ECN) protocols used in the Internet.

One of the novelties in STCP is that it provides controlled variable reliability for applications through end-to-end retransmissions. In other words, STCP intelligently incorporates different transport functions and employs different mechanisms in a single transport protocol for different applications. In SCTP, there is a session initiation process, through which source nodes inform the sink of such application features as the type of data flow, the transmission rate, and the required reliability. For applications producing continuous data flows, the sink measures the current reliability. If it is below the required reliability and the expected packets do not arrive within the anticipated time, the sink will send a NACK for a retransmission. For applications with event-driven flows, the sink utilizes ACK to inform the source node whether packets are lost or successfully received. Then the source node can retransmit the lost packets if the required reliability is not currently satisfied. For data-centric applications where a number of source nodes may be involved and it is not practical and energy efficient to acknowledge all source nodes, STCP can provide neither ACK nor NACK, and behave like the traditional UDP. In summary, STCP implements various reliability and congestion control to meet the requirements of diverse applications while improving energy efficiency and other performance metrics.

11.4.3.2 Event-to-Sink Reliable Transport. Event-to-Sink Reliable Transport [13] is an end-to-end upstream transport protocol, which aims at providing event reliability in upstream through end-to-end rate adjustment. In ESRT, the source sending rate at each sensor node is dependent on the current perceived reliability at the sink, as well as the network status (congestion or not). ESRT is actually the first protocol that addresses both congestion control and reliability simultaneously. In ESRT, each intermediate node detects congestion based on buffer increment and then set a CN bit in the header of packets that will be forwarded to the sink. Based on the received packets from the sensor nodes, the sink can know the network status and current event reliability, which is defined as the number of successfully received event packets during a period of time. Using an AIMD like algorithm, ESRT calculates appropriate reporting

frequency based on the current network status, the current event reliability, and the desired event reliability. Then it notifies sensor nodes of the calculated frequency through a supposed high-power 1-hop channel, and the sensor nodes adjust the source sending rate according to the received new reporting frequency. ESRT only relies on rate adjustment at source sensor nodes to guarantee the desired event reliability, which has two disadvantages: (1) it is energy inefficient compared to retransmission-based loss recovery and (2) it cannot improve (or guarantee) the desired event reliability when congestion occurs. However, loss recovery can still improve reliability. ESRT regulates the reporting frequency of all sensor nodes using the same value. But if the sensor nodes produce data with different importance, for example, due to their position, they will need different reporting frequency. ESRT still needs a dedicated channel with high power to transmit notifications to sensor nodes, which will affect the on-going data transmission.

11.4.4 Open Problems

The transport protocols discussed above consider either congestion control, reliability guarantee, or both. Some of them use end-to-end control while the others use hop-by-hop control. Some of them guarantee event reliability while the others provide packet reliability. However, the existing protocols for WSNs have two primary limitations.

First, sensor nodes may have different importance in specific applications. For example, they can be equipped with different kinds of sensors and deployed in different geographical locations. Therefore, sensor nodes can generate sensory data with different characteristics and have different importance with respect to reliability and bandwidth requirements. However, most existing transport protocols do not consider nodes' different importance although the recent approach in PCCP [19] provides a priority-based congestion control. For example, most congestion control protocols guarantee simple fairness, which means that the sink needs to get the same throughput from all nodes. In addition, most reliability protocols use a single and identical loss-recovery algorithm for all nodes and applications except for STCP. However, sensor nodes and applications may consist of diversified features and priorities, which require flexible loss recovery in order to optimize energy efficiency.

Second, the existing transport protocols for WSNs assume that single-path routing is used at the network layer. Scenarios with multipath routing are not considered except PCCP. It is not clear whether these transport protocols can be directly applied to WSNs employing multipath routing. For example, when multipath routing is utilized, a problem with congestion control protocols is how a sensor node adjusts its own sending rate and the sending rate of its child nodes in a fair and scalable manner. This is because with multipath routing a node may have multiple parents and multiple paths to the sink. The problem could be even more complicated if some nodes have multiple paths, while the others do not.

11.5 SUMMARY AND FUTURE DIRECTIONS

This chapter introduced the principles of traditional transport protocols, discussed the major issues in transport protocol design, and presented an overview of existing transport protocols for WSNs. A transport protocol for WSNs is expected to have high energy efficiency, flexible reliability, and guaranteed application-dependent QoS. Despite the existing transport protocols already proposed, further research is still needed to address those open problems and can be carried out in the following directions.

First, we are interested in designing WSN transport protocols that support node priority with different importance. The existing transport protocols, except STCP, consider only a single type of sensing devices. In the real world, however, it is not uncommon that a single node is equipped with multiple types of sensors for sensing different phenomena, for example, temperature and humidity. For this reason, sensor nodes may generate sensory data with different characteristics and have different priorities in terms of packet loss, bandwidth, delay, and reliability requirements. Therefore, such node priority diversity should be taken into account in the design of transport protocols.

Second, the existing transport protocols only consider single-path routing. When multipath routing is used at the network layer, some issues, for example, fairness, will arise and thus need to be addressed.

Third, almost all the existing protocols either address congestion control or loss recovery. None of them except STCP considers both simultaneously. In fact, effective congestion control can reduce packet loss and provide better throughput, and loss recovery can enhance reliability. Therefore, a transport protocol should consider both congestion control and loss recovery together with performance optimization, energy efficiency, and other performance metrics.

Finally, the existing transport protocols rarely consider cross-layer interactions. In a WSN, link level performance, for example, the bit error rate may have a significant impact on the performance of a transport layer protocol. Similarly, routing can also affect hop-by-hop retransmissions. Therefore, cross-layer optimization is highly desirable in the design of transport protocols for WSNs.

REFERENCES

[1] N. Xu, "A survey of sensor network publications", available at: http://enl.usc.edu/~ningxu/papers/survey.pdf

[2] J. Hill, M. Horton, R. Kling, and L. Krishnamurthy, "The platforms enabling wireless sensor networks", *Communications of the ACM*, vol. 47, no. 6, June 2004, pp. 41–46.

[3] I. Demirkol, C. Ersoy, and F. Alagoz, "MAC protocols for wireless sensor networks: A survey", *IEEE Communications Magazine*, vol. 44, no. 4, Apr. 2006, pp. 115–121.

[4] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: A survey", *IEEE Wireless Communications*, vol. 11, no. 6, Dec. 2004, pp. 6–28.

[5] C. Wang, B. Li, K. Sohraby, M. Daneshmand, and Y. Hu, "A survey of transport protocols for wireless sensor networks", *IEEE Network Magazine*, vol. 20, no. 3, May/June 2006, pp. 34–40.

[6] I. F. Akyildiz, T. Melodia, and K. R. Chowdury, "A survey on wireless multimedia sensor networks", *Computer Networks (Elsevier)*, vol. 51, no. 4, Mar. 2007, pp. 921–960.

[7] M. Allman et al., "TCP congestion control", *IETF RFC2581*, Apr. 1999.

[8] J. B. Postel, "User datagram protocol", *IETF RFC 768*, Aug. 1980.

[9] R. Stewart and Q. Xie, "Stream control transport protocol", IETF RFC2960, Oct. 2001.

[10] W. Stallings, *Data and Computer Communications* (7th ed.), Pearson Prentice Hall, New York, 2004.

[11] K. Pentikousis, "TCP in wired-cum-wireless environments", *IEEE Communications Surveys*, vol. 3, no. 4, Oct. 2000, pp. 2–14.

[12] K. Sundaresan, V. Anantharaman, H.-Y. Hseeh, and R. Sivakumar, "ATP: A reliable transport protocol for ad-hoc networks", in *Proceedings of 2003 ACM International Symposium on Mobile Ad Hoc Networking and Computing* (Mobihoc'03), Annapolis, MA, June 2003, pp. 64–75.

[13] Y. Sankarasubramaniam, O. B. Akan, and I. F. Akyildiz, "ESRT: Event-to-sink reliable transport in wireless sensor networks", in *Proceedings of 2003 ACM International Symposium on Mobile Ad Hoc Networking and Computing* (Mobihoc'03), Annapolis, MD, June 2003, pp. 177–188.

[14] S.-J. Park, R. Vedantham, R. Sivakumar, and I. F. Akyildiz, "A scalable approach for reliable downstream data delivery in wireless sensor networks", in *Proceedings of 2004 ACM International Symposium on Mobile Ad Hoc Networking and Computing* (MobiHoc'04), Roppongi, Japan, May 2004, pp. 78–89.

[15] D. Chen and P. K. Varshney, "QoS support in wireless sensor networks: A survey", *in Proceedings of 2004 International Conference on Wireless Networks* (ICWN'04), Las Vegas, NV, June 2004.

[16] C.-T. Ee and R. Bajcsy, "Congestion control and fairness for many-to-one routing in sensor networks", in *Proceedings of 2004 ACM Conference on Embedded Networked Sensor Systems* (Sensys'04), Baltimore, MD, Nov. 2004, pp. 148–161.

[17] B. Hull, K. Jamieson, and H. Balakrishnan, "Mitigating congestion in wireless sensor networks", in *Proceedings of 2004 ACM Conference on Embedded Networked Sensor Systems* (Sensys'04), Baltimore, MD, Nov. 2004, pp. 134–147.

[18] C.-Y. Wan, S. B. Eisenman, and A. T. Campbell, "CODA: Congestion detection and avoidance in sensor networks", in *Proceedings of 2003 ACM Conference on Embedded Networked Sensor Systems* (Sensys'03), Los Angeles, CA, Nov. 2003, pp. 266–279.

[19] C. Wang, B. Li, K. Sohraby, M. Daneshmand, and Y. Hu, "Upstream congestion control in wireless sensor networks through cross-layer optimization", *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 5, May 2007, pp. 786–795.

[20] C.-Y. Wan, S. B. Eisenman, A. T. Campbell, and J. Crowcroft, "Siphon: Overload traffic management using multi-radio virtual sinks in sensor networks", in *Proceedings of 2005 ACM Conference on Embedded Networked Sensor Systems* (SenSys'05), San Diego, CA, Nov. 2005, pp. 116–129.

[21] A. Woo and D. C. Culler, "A transmission control scheme for media acess in sensor networks", in *Proceedings of 2001 ACM Annual International Conference on Mobile Computing and Networking* (Mobicom'01), Rome, Italy, July 2004, pp. 221–235.

[22] A. Dunkels, J. Alonso, T. Voigt, and H. Ritter, "Distributed TCP caching for wireless sensor networks", in *Proceedings of Third Annual Mediterranean Ad Hoc Networking Workshop*, Bodrum, Turkey, June 2004.

[23] C.-Y. Wan, A. T. Campbell, "PSFQ: A reliable transport protocol for wireless sensor networks", in *Proceedings of the 1st ACM International Workshop on Wireless Sesnor Networks and Applications* (WSNA'02), Atlanta, GA, Sept. 2002, pp. 1–11.

[24] Y. G. Iyer, S. Gandham, and S. Venkatesan, "STCP: A generic transport layer protocol for wireless sensor networks", in *Proceedings of 2005 14th IEEE International Conference on Computer Communications and Networks* (ICCCN'05), San Diego, CA, Oct. 2005, pp. 449–454.

[25] P. Levis, N. Patel, D. Culler, and S. Shenker, "Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks", in *Proceedings of First Symposium on Networked Systems Design and Implementation* (NSDI'04), San Francisco, CA, Mar. 2004, 2–2.

[26] F. Stann and J. Heidemann, "RMST: Reliable data transport in senor networks", in *Proceedings of IEEE International Workshop on Sensor Network Protocols and Applications* (SNPA'03), Anchorage, AK, May 2003.

[27] H. Zhang, A. Arora, Y. Choi, and M. G. Gouda, "Reliable bursty convergecast in wireless sensor networks", in *Proceedings of The ACM International Symposium on Mobile Ad Hoc Networking and Computing* (Mobihoc'05), Urbana-Champaign, IL, May 2005, pp. 266–276.

12

NETWORK SECURITY AND ATTACK DEFENSE

Yun Zhou and Yuguang Fang

University of Florida, USA

12.1 INTRODUCTION

Wireless sensor networks (WSNs) have many potential civilian and military applications, for example, environmental monitoring, battlefield surveillance, and homeland security. In many important military and commercial applications, it is critical to protect a sensor network from malicious attacks, which presents a demand for providing security mechanisms in the network. However, designing security protocols is a challenging task for a WSN because of the following unique characteristics:

1. Wireless channels are open to everyone. With a radio interface configured at the same frequency band, anyone can monitor or participate in the communication in a wireless channel. This provides a convenient way for attackers to break into a network.
2. As in the case of the Internet, most protocols for WSNs do not consider necessary security mechanisms at their design stage. On the other hand, most protocols are publicly known due to the needs for standardization. For these reasons, attackers can easily launch attacks by exploiting security holes in those protocols.

Wireless Sensor Networks: A Networking Perspective, Edited by Jun Zheng and Abbas Jamalipour

3. The constrained resources in sensor nodes make it very difficult to implement strong security algorithms on a sensor platform due to their complexity. In most cases, symmetric key cryptography is the first choice for designing a security protocol for WSNs, though public key cryptography is possible under careful optimization in design and implementation. In addition, a WSN may scale up to thousands of sensor nodes. These pose the demand for simple, flexible, and scalable security protocols. A stronger security protocol costs more resources in sensor nodes, which can lead to the performance degradation of applications. In most cases, a trade-off has to be made between security and performance. However, weak security protocols may be easily broken by attackers.
4. A WSN is usually deployed in hostile areas without any fixed infrastructure. It is difficult to perform continuous surveillance after network deployment. Therefore, it may face various potential attacks.

Network security is a research area that involves many technical issues. To protect a network, there are usually several security requirements, which should be considered in the design of a security protocol, including confidentiality, integrity, and authenticity. An effective security protocol should provide services to meet these requirements. This chapter introduces the most common security services for WSNs, including confidentiality, integrity, authenticity, nonrepudiation, freshness, availability, intrusion detection, and key management. For each service, we first introduce basic concepts, describe typical problems and attacks in WSNs, and then present major effective solutions available in the literature.

Section 12.2 focuses on the confidentiality problem and discuss how to use encryption to protect data information and anonymous routing to ensure the privacy of identity information. Section 12.3 introduces the integrity problem due to packet errors and presents some typical error-control mechanisms. Section 12.4 presents two cryptographic techniques, message authentication code and signature, to ensure authenticity, and discuss how to use these techniques to provide unicast and broadcast–multicast authentication. Section 12.5 discusses the nonrepudiation problem and some typical solutions using signatures. Section 12.6 presents how to use timestamp or sequence number to ensure packet freshness. Section 12.7 introduces several typical attacks compromising the availability of network services and their specific countermeasures. Section 12.8 discusses the intrusion detection problem and presents several solutions based on node monitoring. Section 12.9 discusses the key management problem by presenting both symmetric key and asymmetric key solutions. Finally, we conclude with a brief summary of the chapter.

12.2 CONFIDENTIALITY

Confidentiality is an assurance of authorized access to information. In the context of networking, confidentiality means that the information about

communications should be kept in secret from anyone without authorized access permission.

12.2.1 Eavesdropping

Providing confidentiality is a challenging task in wireless networks. The major problem is that radio spectrum is an open resource and can be used by anyone equipped with proper radio transceivers. An attacker can eavesdrop on the packets transmitted in the air as long as he is able to keep track of the radio channels used in the communication.

In general, a packet contains a data portion, called *protocol data unit* (PDU), and a header. A packet header contains the network addresses of the source and destination nodes, which are used for intermediate nodes to route the packet from the source to the destination, and other control information, which describes the property of the PDU and/or tells how to process the PDU. At each hop along the route from the source to the destination, the packet is encapsulated into one or multiple link-layer frames. Similar to a packet, a frame also contains a link-layer PDU and a frame header. The frame header contains the *medium access control* (MAC) addresses of two neighboring nodes along the route from the source to the destination, as well as other frame control information. Without proper protection, the PDU of a packet could be disclosed to the public. If the PDU contains important information, for example, business or military secrets, this information disclosure will cost invaluable loss. Moreover, the packet header and the frame header are also of interest to attackers. Since the headers contain information about the communicating nodes and PDUs, the disclosure of headers will lead to the privacy problem. For example, an attacker can identify a traffic flow based on the source and the destination addresses contained in the packet headers. By tracking the flow, the attacker can determine the location of the source node or the destination node. In the applications of monitoring wild lives, the location of the source node can imply the appearance of the wild animals, which is attractive to illegal hunters. In military applications of a WSN, the destination of most traffic flows can be a higher rank commander, whose location can be found through location tracking. In both the examples, the attacker can determine the locations of important nodes and then launch attacks.

In addition to packet headers, an attacker can also eavesdrop on control or management packets exchanged among nodes, such as routing maintenance packets, and then derive the network topology from them. This renders the attacker the ability to determine the identities and locations of all the nodes of interest. By attacking those important nodes, the attacker can destroy the network very easily.

12.2.2 Node Compromise

An active attack failing confidentiality is called node compromise. In a direct way, an attacker can capture a node, dig into it with special tools, and find useful data.

In an indirect way, the attacker can derive the secrets in a node without capturing it, which can be done by analyzing the secret data collected from other compromised nodes and/or packet PDUs. Under the attacker's control, the new compromised node can be used to launch more malicious attacks.

Node compromise is one of the most detrimental attacks to WSNs. A WSN is usually deployed in a hostile environment, for example, military battle fields. Continuously monitoring the network in such an environment is not guaranteed. This renders attackers the opportunity to compromise sensor nodes. Because sensor nodes are manufactured as low-cost devices without strong protection, node compromise can be rather easy in WSNs.

12.2.3 Encryption

Encryption is an effective method to counteract eavesdropping. The idea is to transform useful data into a special form, which is understandable to intended receivers, but unintelligible to unauthorized attackers before transmission.

There are two techniques to realize encryption. The first one is *symmetric key* cryptography. In a symmetric key system, the sender and the receiver share a key K, which is kept in secret from others. The sender encrypts a plaintext M with K by an encryption algorithm E to get a ciphertext $C = E(M, K)$. Then it transmits the ciphertext C instead of the plaintext M to the receiver. After receiving the ciphertext C, the receiver inputs C and the shared key K into a decryption algorithm D to recover the original plaintext $M = D(C, K)$. The point here is that because the secretly shared key K is unknown to anyone else, no one else can decrypt the ciphertext C into the plaintext M, and thus confidentiality is preserved. Some well-known symmetric key algorithms include *data encryption standard* (DES) [1], *advanced encryption standard* (AES) [2], and *rivest cipher 5* (RC5) [3], and so on.

The other widely used technique is *asymmetric key* cryptography. In an asymmetric key system, each user has a pair of keys $\{K_s, K_p\}$. The user keeps her private key K_s in secret while publishing her public key K_p. When a sender wants to send a plaintext M to a receiver, the sender uses the receiver's public key K_p to encrypt M into a ciphertext $C = E(M, K_p)$, and then transmits the ciphertext C to the receiver. Because K_p is publicly well known, anyone can send messages to the receiver. Because K_s is secret, only the receiver can decrypt the ciphertext C into the plaintext $M = D(C, K_s)$. Since a public key is used here, asymmetric key systems are also called *public key* systems. Two most popular asymmetric key algorithms are Diffie–Hellman [4] and RSA [5].

In general, symmetric key algorithms require simple hashing, bit rotation, or scrambling operations, which can be efficiently implemented in hardware or software. Asymmetric key algorithms require exponential operations over a field modulo a large prime number, which are more complex than symmetric key operations. Therefore, symmetric key algorithms are more viable on resource-constrained low-end devices. Alternatively, asymmetric key algorithms are more secure and support more security services, for example, authentication and/or

nonrepudiation, while symmetric key algorithms are mostly used for encryption. Both symmetric key and asymmetric key algorithms can provide confidentiality for unicast communications as long as the sender and the receiver are equipped with corresponding keys. For broadcast/multicast communications, however, the situation is different. Asymmetric key algorithms cannot be used to encrypt broadcast/multicast traffic. The reason is that in asymmetric key algorithms the key used for encryption belongs to the receiver and in broadcast/multicast communications there are multiple receivers, each of which has their own asymmetric key pair. In this scenario, symmetric key algorithms can be used. In particular, all the member nodes in a broadcast/multicast group share a common symmetric key and this key is used to encrypt/decrypt communication traffic inside the group.

Usually, encryption is performed over PDUs so that attackers cannot eavesdrop on them to learn important information. Not encrypting packet headers does not mean that they are not important. The reason is that packet headers contain routing information and some volatile fields, which are modified in every forwarding node. Encrypting packet headers can cause problems to routing protocols.

If node identities do need to be protected in some privacy-critical applications, the link-layer encryption can be performed over each hop. In this scenario, link-layer frame PDUs are encrypted. This can effectively prevent external attackers from eavesdropping on network-layer packets. However, internal forwarding nodes can still learn the content of each packet because they can decrypt each link-layer frame. If any intermediate forwarding node is compromised, the identity information in forwarded packets is still exposed. In addition, the MAC addresses in a link-layer frame header can also leak some identity information on the nodes in 1-hop communications.

TinySec [6] is a link-layer security architecture fully implemented for WSNs. It defines a frame format called *authenticated encryption* (AE) for encryption at the link layer. In an AE frame, the PDU is encrypted according to a symmetric key algorithm called Skipjack [7], which is a light-weight block cipher and can be implemented in resource-constrained sensor nodes.

12.2.4 Privacy

Privacy is becoming a very important security issue as the concerns on the disclosure of personal information, for example, identity to unauthorized attackers, are getting much stronger. As discussed before, an attacker can track a traffic flow to find the identities and then the locations of the source and destination nodes. In WSNs, this location tracking can be easier because in most scenarios sensor nodes are static. From the network administrator's point of view, it is desirable if the location information of the source and destination nodes can be hidden from attackers.

As discussed in the previous section, encryption is a method to protect the privacy of node identities and thus realize anonymous communications to some extent at the cost of increased routing inefficiency. In Ref. [8], a link-layer

symmetric key is used to encrypt frame PDUs and the corresponding MAC addresses in frame headers. In order to maintain the frame forwarding function, node anonyms are used as new MAC addresses. A similar approach is taken in MASK [9], anonymous on-demand routing protocol, but MASK uses link anonyms to identify links and forward frames.

There are other methods using non-cryptographic designs to protect privacy. For example, the source location privacy is very important in WSNs because the location of the source is usually an area where a target of high value is located. In order to enhance the source location privacy, a new routing protocol, called phantom routing, is introduced in Ref. [10]. In the phantom routing, the delivery of each packet experiences two phases:

1. A random walk phase, which may be a pure random walk or a directed walk, for directing the packet to a phantom source, that is, a fake source.
2. A subsequent flooding/single-path routing phase for delivering the packet to the destination.

The phantom routing significantly increases the source location privacy by introducing phantom sources while marginally increasing the communication overhead compared with the flooding and the single-path routing.

The destination location privacy is a dual problem to the source location privacy. In particular, the location of the base station in a WSN is also very important. The base station is a gateway of the network to the external wired world. Therefore, the base station can become a failure point of interest to attackers because all traffic flows will go through it. In order to disguise the location of the base station, fake traffic is introduced in Ref. [11], where each forwarding node creates fake traffic flows toward the directions opposite to the destination in such a way that the traffic pattern in the network is almost evenly distributed.

12.3 INTEGRITY

Integrity is an assurance that packets are not modified in transmission. This is a basic requirement for communications because the receiver needs to know exactly what the sender wants her to know. However, this is not an easy task in wireless communications.

12.3.1 Transmission Errors

Transmission errors are inherent in wireless communications because of the instability of wireless channels, which is due to many reasons, for example, channel fading, time-frequency coherence, and inter-band interference. A packet

bearing errors is useless and causes extra processing at the sender and the receiver.

12.3.2 Processing Errors

Errors can also happen in each forwarding node because no electronic devices are perfect. When the operation conditions, for example, temperature or humility, are out of the normal range, electronic devices can run into malfunction, which can cause errors in packets. Those errors may not be noticed by the forwarding node and thus those error packets may still be sent out, causing troubles at downstream nodes.

12.3.3 Packet Modifications

In a hostile environment, an attacker can modify a packet before it reaches the receiver. This can cause many problems. The attacker can simply introduce radio interference to some bits in transmitted packets to change their polarities. The unintelligible packets will be dropped at the receiver, leading to a simple *Denial of Service* (DoS) attack [12].

More serious damages can be caused if the attacker understands the packet format and the semantic meaning of the communication protocol. In that case, the attacker can modify a packet to change its content so that the receiver obtains wrong information. In a WSN, for example, a packet containing the location of an important event can be modified so that a wrong location is reported to the base station. Control and management packets can be changed so that nodes have inconsistent knowledge on the network topology, which causes many routing problems.

12.3.4 Error Control

There are some error control mechanisms at the link layer dealing with the transmission errors. One is the error detection code. The idea is to attach each link-layer frame with some redundancy bits, which is calculated according to an error detection algorithm and is usually called checksum. Each receiving node can find out whether there is an error in a received frame by inspecting its checksum. If an error happens, the receiver can send a notice frame to the sender to request a retransmission of the original frame. This feedback mechanism is called *automatic repeat request* (ARQ). If more redundancy bits are attached to each frame, the receiving node may even correct errors and thus avoiding ARQ. This mechanism is called *forward error correction* (FEC) and the checksum in each frame is computed according to an error correction code algorithm.

Both ARQ and FEC can also be used at the transport layer to deal with the processing errors incurred in intermediate forwarding nodes. The source node computes a checksum for each transport-layer PDU and the destination node inspects the checksum to detect or correct errors.

12.3.5 Message Integrity Code

Neither ARQ nor FEC can be used to deal with malicious modification by attackers. The reason is that all the error detection/correction code algorithms are publicly well known. An attacker can modify a frame and recompute its checksum. Therefore, a receiving node cannot detect the modification because the modified frame matches its checksum.

In order to guarantee integrity, two cryptographic methods can be used. One is *message integrity code* (MIC) and the other is using signature. These concepts will be discussed in Section 12.4 because they can also be used for authentication.

12.4 AUTHENTICITY

Authenticity is an assurance of the identities of communicating nodes. Every node needs to know that a received packet comes from a real sender. Otherwise, the receiving node can be cheated into performing some wrong actions.

12.4.1 Packet Injection

In addition to modifying existing packets, an attacker can directly inject packets if he knows the packet format defined in the network protocol stack. The injected packets can carry false information, which may be accepted by receiving nodes. Applications deployed in a WSN, for example, environmental monitoring or object tracking, can be disrupted by the false information. Routing protocols can fail due to the false routing information.

The Sybil attack [13] is a typical example of packet injection. In a Sybil attack, an attacker illegitimately takes on multiple identities by injecting false packets containing spoofed source IP or MAC addresses, which can pose a serious threat to distributed storage, routing protocols, data aggregation, voting, fair resource allocation, intrusion detection, and so on.

12.4.2 Message Authentication Code

In order to deal with false packets, authentication is indispensable to ensure the origin of received packets. *Message authentication code* (MAC) is a tool to solve the problem. It can also be called MIC because it ensures packet integrity as well.

To compute a MAC, a symmetric key shared between the sender and the receiver is required. For a packet payload M, the sender concatenates it with the shared key K and then computes a MAC as $C = H(M\|K)$, where $H(\cdot)$ is a collision-resistant hash function [14] and "||" is the concatenating operator. The packet including payload M and the MAC C is sent to the receiver. The receiver recomputes a MAC C' with the payload M and the shared key K and then checks whether $C' = C$ holds. If the equation holds, the payload M is authenticated and

not modified because only the sender knows the shared key. Otherwise, the packet is either modified or injected.

In TinySec [6], another type of frame Auth is defined in addition to AE. Both Auth and AE frames carry a MAC so that the link-layer authentication is supported between neighboring nodes.

12.4.3 Challenge Response

The rationale behind the MAC design is that the key is tight to the identities of the communicating nodes. This is usually used to test the identity of a node. The method is called *challenge response*. In particular, one node selects a random number, encrypts it with the shared key, and sends the ciphertext, which is called a challenge, to the other node. If the challenged node can decrypt the challenge and return the original random number, the identity of the challenged node is assured because it has the correct key. This challenge-response protocol can be performed mutually between two nodes so that each of them authenticates the other.

The challenge-response protocol has been used to detect the Sybil attack in WSNs. One method in Ref. [13] requires that each sensor node be preloaded with a set of keys selected from a global key pool according to its node identity. By choosing proper sizes of the key set and the global key pool, the network administrator ensures that each sensor node shares keys with multiple other nodes. Therefore, the identity of a suspected Sybil node can be verified through the challenge-response protocol by a set of validating nodes that share keys with the suspected node. A similar method is used in Ref. [15]. The difference is that every pair of neighboring nodes can compute a shared symmetric key based on their identities and locations during the initialization phase. Therefore, multiple identities will need multiple keys, which is impossible for a Sybil node to achieve.

12.4.4 Signature

Signature is an asymmetric key technique, which is widely used in authentication. A sender node keeps its private key K_s in secret while publishing its public key K_p. In order to authenticate a plaintext M to the receiver, the sender uses its private key K_s to sign M into a signature $S = S(M, K_s)$ and then transmits the signature S, as well as the plaintext M to the receiver. Because K_s is secret, only the sender can generate the signature. Because K_p is publicly well known, any receiver can verify the signature S by inputting the signature S, the plaintext M, and the public key K_p into a verification algorithm M to compute $V(S, M, K_p)$. If the output is TRUE, the plaintext M is authenticated, and otherwise not.

12.4.5 Man-in-the-Middle

There are two requirements for an authentication procedure based on signature. One is that the private key K_s of the sender should be secret. The other is that

the public key K_p should be known by all intended receivers. The first requirement is easy to achieve, but the second is difficult to achieve in reality because how to know whether a public key belongs to a key owner is a difficult problem. Failing to addressing this problem can lead to severe attacks to an authentication system.

A typical example of the impacts of failing to authenticating public keys is the man-in-the-middle (MiM) attack. In this attack, a malicious node T intercepts all the communications between two nodes A and B who do not know the public key of each other. Node T selects a pair of asymmetric keys $\{K_s', K_p'\}$, keeps K_s' in secret, and claims to A that K_p' is B's public key. Then node T selects another pair of keys $\{K_s'', K_p''\}$, keeps K_s'' in secret and claims to B that K_p'' is A's public key. In this way, A will see T as B and as well B will see T as A. Thus T successfully intercepts all the traffic between A and B.

12.4.6 Authenticating Public Key

The reason that the MiM attack is possible is that the authenticity of the public key cannot be assured. Therefore, authenticating public keys in asymmetric key systems is a very critical problem.

The conventional solution to the public key authentication is to rely on a *public key infrastructure* (PKI). In the PKI, there is a *certificate authority* (CA), which is trusted by all the members using the PKI. The public key of the CA is accepted by all the member nodes as an authenticated one in default. The CA signs the public key of each member node and issues a certificate including the public key and the corresponding signature to the member node. When two nodes need to communicate, one of them sends its public key certificate to the other node that can verify the authenticity of the public key in the certificate with the well-known public key of the CA. In this way, the two nodes can authenticate each other.

The PKI discussed above can be illustrated as a two-level tree with the CA as the root and all the member nodes as leafs. In reality, a PKI can be described as a multilevel tree. Each non-leaf node acts as a local CA and manages its children CAs at lower levels.

Public key certificates have been widely used in the Internet and other wireless networks, for example, *wireless local area networks* (WLANs). However, there is one obstacle to using public key certificates in WSNs. Most asymmetric key algorithms, for example, Diffie–Hellman [4] and RSA [5] are very expensive on resource-constrained sensor platforms. Therefore, how to efficiently perform asymmetric key algorithms becomes a very important and popular issue in WSNs.

One approach is to use specific parameters that can speed asymmetric key algorithms without compromising security too much. TinyPK [16] is an example of using RSA based public key certificates in WSNs. In particular, an external user needs to acquire a certificate from the network administrator, who acts as the CA. The certificate is verified by sensor nodes so that the user may be authorized to access the network to collect data or issue commands. In order to simplify

the certificate verification in sensor nodes, the CA's RSA public key is chosen as 3, while an ordinary value of a RSA public key for a satisfying security is usually hundreds of bits long. Meanwhile, researchers are looking for new algorithms that are more efficient than traditional asymmetric key algorithms. A more promising technique is the *elliptic curve cryptography* (ECC) [17,18]. The fundamental operation underlying RSA is the modular exponentiation in integer rings. Its security stems from the difficulty of factorizing large integers. Currently, there only exist subexponential algorithms to solve the integer factorization problem. ECC operates on groups of points over elliptic curves and derives its security from the hardness of the *elliptic curve discrete logarithm problem* (ECDLP). The best algorithms known for solving the ECDLP are exponential. Therefore, attacking ECC is more difficult than attacking RSA. As a result, ECC can achieve the same level of security as RSA with smaller key sizes. For example, the 163-bit ECC can provide comparable security to the conventional 1024-bit RSA. Under the same security level, smaller key sizes of ECC offer merits of faster computational efficiency, as well as memory, energy, and bandwidth savings. Therefore, ECC is better suited for the resource constrained devices. In an access control protocol for WSNs based on ECC [19], the *elliptic curve digital signature algorithm* (ECDSA) [20] is used to authenticate new sensor nodes when they join the network and the ECC based Diffie–Hellman algorithm is used to establish shared keys between sensor nodes.

Another approach of authenticating public keys attempts to avoid public key certificates by using the symmetric key technique or identity-based cryptography. A public key authentication scheme [21] uses a symmetric key technique, called a Merkle tree [22]. In a Merkle tree, each parent is a hash of the concatenation of its children, and each leaf is corresponding to a node and is calculated as a hash of the concatenation of the node identity and its public key. For each leaf, there is a witness, which is the set of the siblings of the nodes along the path from the leaf to the root. The root can be recovered based on each leaf node and its witness. An example is illustrated in Fig. 12.1. For the leaf h_2, its witness consists of $\{h_3, h_{01}, h_{47}\}$ and the root can be covered as $h_{07} = H(H(h_{01}||H(h_2||h_3))||h_{47})$.

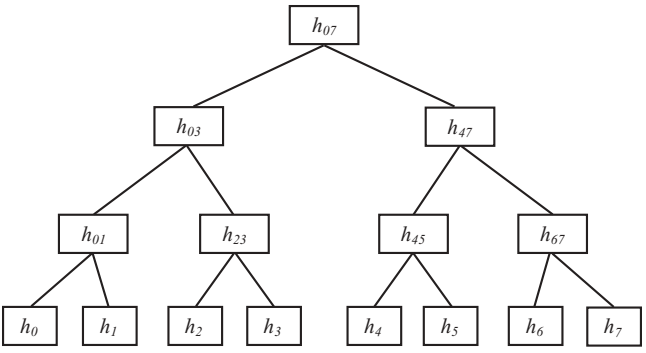


Fig. 12.1 An example of a Merkle tree.

When a sensor node needs to authenticate its public key to other nodes, it attaches its public key with its witness. Other sensor nodes verify whether they can recover the root whereby to authenticate the public key.

The reason for authenticating public keys is that each public key should be bound to a particular node. Otherwise, the MiM attack is possible. In the *identity-based cryptography* (IBC) [23], however, this key-node binding relationship is removed because the publicly known identity information of a node is directly used as its public key. Therefore, IBC removes the necessity of public key certificates and thus saves cost on certificate verification. One application of IBC is proposed in Ref. [15], in which the public key of each node is chosen as a combination of its identity and location because the sensor nodes in the network are assumed to be static.

12.4.7 Broadcast and Multicast Authentication

Broadcast/multicast is a common mechanism to disseminate information from a source node to a group of destination nodes. As in unicast, each packet in broadcast/multicast should also be authenticated. This can be done by using the symmetric key technique or the asymmetric key technique.

A simple symmetric key technique is to configure all the nodes in a broadcast/multicast group with a shared group key, which is an extension of the shared pairwise key in unicast authentication. A source node simply attaches each outgoing packets with a MAC computed with the group key and all the destination nodes verify the MAC with the group key. This method assumes that all the group members are trustful. This assumption may fail if an attacker is able to compromise any member node. A compromised member node can impersonate the source node and spread false information because he also knows the group key.

In order to solve the problem of internal attackers, a special symmetric key technique called *one-way hash chain* (OHC) can be used. An OHC is a sequence of numbers $\{K_0, K_1, \ldots, K_n\}$, such that $K_{j-1} = H(K_j)$, $\forall j \in \{1, 2, \ldots, n\}$, where the hash function $H(\cdot)$ satisfies two properties:

1. Given x, it is easy to computer $y = H(x)$.
2. Given y, it is computationally infeasible to compute x such that $y = H(x)$.

The first number of the OHC, K_0, is securely sent to all the receiving nodes as a commitment. When the source node needs to broadcast/multicast a packet in the kth round, it includes K_k in the packet. Then every member node can authenticate K_k by verifying whether

$$H^k(K_k) = H^{k-1}(K_{k-1}) = \ldots = H(K_1) = K_0$$

holds. However, K_k cannot be directly used to authenticate packets in the kth round. The reason is that if an attacker has the ability to capture a packet, he can

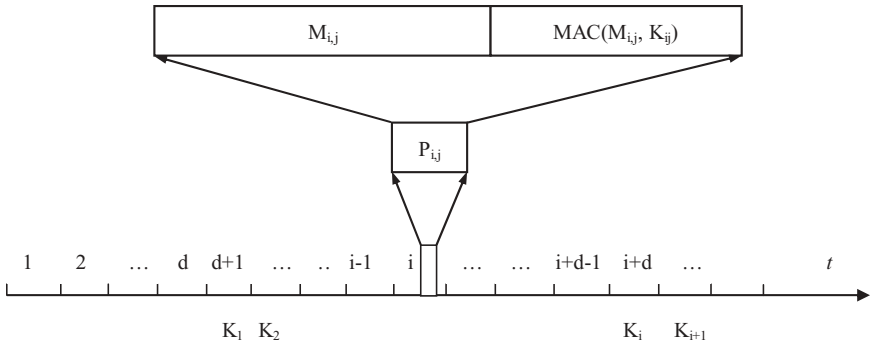


Fig. 12.2 The μTESLA.

extract the key attached to it, generate a false packet, and send it to a node in the same round before the true packet arrives at that node.

By using the OHC technique, μTESLA in SPINS [24] simulates the asymmetry of the asymmetric key technique and provides authenticated broadcast services from the base station. In μTESLA, the first key of the OHC is sent to all the receiving nodes as a commitment in advance, as shown in Fig. 12.2. A broadcast packet in the kth time slot carries a MAC generated by using the kth key K_k of the OHC. Every node does not know the kth key when it receives the packet. In the $(k + d)$th time slot, the base station discloses the kth key K_k, which is used by every node to authenticate the cached packet. This introduced asymmetry by delayed key release can efficiently prevent malicious nodes from impersonating the source node.

However, μTESLA requires the distribution of the key chain commitment K_0 to all the nodes, which is not communication efficient because the commitment has to be unicasted to each node. Moreover, a key chain may not support broadcast for a long time because the length of the key chain is fixed. The requirements of time synchronization and OHC management are too strict for low-end devices. Therefore, μTESLA targets the global broadcast [24] from the base station. In some circumstances, each individual node needs the local broadcasts to fulfill some specific functions in its neighborhood, for example, exchanging routing information or cluster-head election. Obviously, the local broadcast also needs to be authenticated.

Zhou and Fang proposed a batch-based broadcast authentication scheme called BABRA in Ref. [25]. In BABRA, broadcasted packets are sent in batches and each batch is a burst sequence of packets. There is a key associated with each batch. All the packets in one batch carry a MAC calculated based on the associated key. The key will be disclosed only after a certain delay from the end of the batch. Therefore, the asymmetry introduced by the delayed key disclosure prevents attackers from injecting bogus packets because the sender never sends any packet of one batch after the key disclosure period. In order to authenticate each batch key, the hash of the key is embedded in the packets of previous batches.

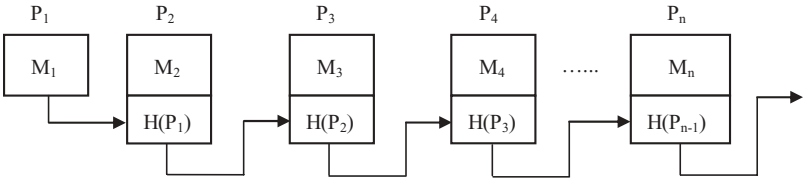


Fig. 12.3 A hash chain.

Thus when one packet of the current batch is authenticated, the key hash in it can be used to authenticate the key of the next batch. In BABRA, each key is independent of the others. The elimination of key chains makes BABRA suitable for both the network broadcast by a base station and the local broadcast by sensor nodes.

As in unicast authentication, signatures can also be used in broadcast/multicast authentication. In an ideal case, each packet includes a signature generated with the sender's private key and each receiver verifies the signature with the sender's public key. As it is well known that existing digital signature algorithms are computationally expensive, the ideal approach raises a serious challenge to sender's and receivers' computational capability.

A block-based approach is proposed in the literature to reduce the number of signature operations [26]. In particular, the sender divides a broadcast/multicast stream into blocks, associates each block with a signature, and spreads the effect of the signature across all the packets in the block through some efficient data structures.

One data structure is a hash chain, which is shown in Fig. 12.3. In each block, the hash of each packet is embedded into several other packets in a deterministic or probabilistic way. The hashes form chains linking each packet to the block signature. Each receiver verifies the block signature and authenticates all the packets through hash chains. In order to tolerate a certain level of packet loss, each packet can be linked to multiple other packets and further to the block signature.

Erasure coding is another method to disperse the effect of signature. A block consists of n packets. For each packet payload, a hash is computed. A block signature is generated for the concatenation of all the hashes. Then the signature and all the hashes are input into an (m, n) erasure coding algorithm, which outputs n pieces. Each packet in the block is attached with one piece before being sent out. Each receiver can recover the original n hashes and the block signature as long as it receives at least m pieces. An example of using erasure coding is illustrated in Fig. 12.4.

The block-based approach can achieve computational efficiency because the computation requirement is reduced to one signature operation plus some hash or decoding operations for a block of packets. However, they suffer from some drawbacks. First, hash chains and erasure codes establish relationship among all the packets in one block. The relationship, however, makes the block-based

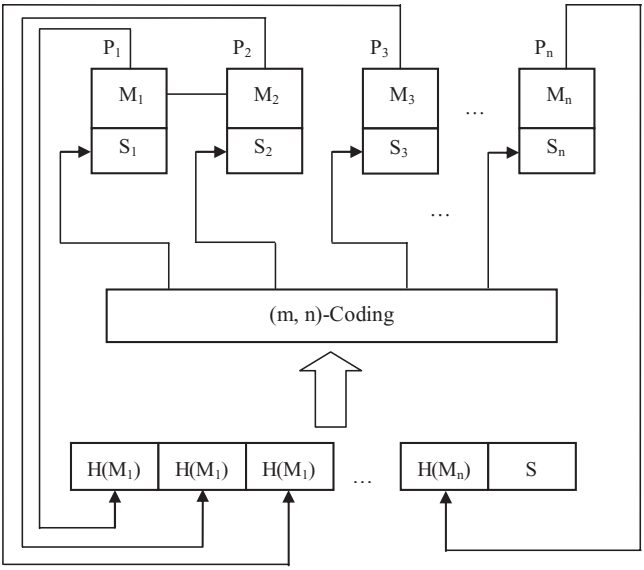


Fig. 12.4 Erasure coding.

approach vulnerable to packet loss, which is very common in current Internet and wireless networks. The loss of a certain number of packets can result in the failure of authentication of other received packets. In an extreme case, the loss of the signature of one block makes the whole block of packets unable to be authenticated. Second, the block design requires the sender and/or receivers buffer a certain number of packets before processing them. A larger block size can achieve higher computational efficiency, but incurs longer buffering delay. This authentication latency at the sender and/or receivers can compromise the real-time requirements in many multimedia application scenarios, for example, live video show or stock quotes delivery.

In Refs. [27,28], a novel broadcast/multicast authentication protocol called *multicast authentication based on batch signature* (MABS) uses an efficient asymmetric cryptographic primitive called *batch signature*, which supports the authentication of any number of packets simultaneously. In particular, a sender generates a signature for each outgoing packet with its private key. When a receiver collects n packets $p_i = \{m_i, s_i\}$, $i = 1, \ldots, n$, where m_i is the data payload, s_i is the corresponding signature, and n can be any positive integer, the receiver can input them into an algorithm, $BatchVerify(p_1, p_2, \ldots, p_n) \in \{True, False\}$. If the output is *True*, then the n packets are authentic. Otherwise, they are not authentic.

In order to support authenticity and efficiency, the algorithm *BatchVerify*() should satisfy the following properties: (1) given a batch of packets that have been signed by the sender, *BatchVerify*() outputs *True*, (2) given a batch of packets including some unauthentic packets, the probability that *BatchVerify*() outputs *True* is very low, and (3) the computational complexity of *BatchVerify*()

is comparable to that of verifying one signature and is increased gradually when the batch size n increases.

By using batch signature, each receiver can achieve the computational efficiency comparable to conventional block-based schemes in the sense that a batch of packets can be authenticated simultaneously through one batch signature verification operation. The MABS protocol also eliminates the correlation among packets, and thus is perfectly resilient to packet loss in the sense that no matter how many packets are lost, the rest can also be verified by receivers. In addition, using per packet signature instead of per block signature eliminates the authentication latency at the sender and/or receivers. Each receiver can verify the authenticity of all the received packets in its buffer anytime.

12.5 NONREPUDIATION

Nonrepudiation is an assurance of the responsibility to an action. Particularly, in a successful transmission between a source node and a destination node, the source node should not be able to deny having sent a message and the destination should not be able to deny having received the message. This service consists of two parts, that is, nonrepudiation of origin and nonrepudiation of receipt.

Nonrepudiation defends against cheating parties. When a dispute between the source and the destination happens, they can turn to an adjudicator who decides which of them is responsible for the dispute. In this procedure, both the source and the destination should present to the adjudicator their own evidences of the actions taken in their communications.

Nonrepudiation evidences are computed based on signatures so that they can prove the identities of the source and the destination. How to construct a nonrepudiation protocol using the evidences is a challenging problem. Consider a simple protocol using signatures. A source node generates a signature for a message, and then sends the message and the signature to a destination. Now the destination knows the message and uses the signature as an evidence for nonrepudiation of origin. In order to complete the nonrepudiation protocol, the destination should reply to the source with a signed acknowledgment as the evidence for nonrepudiation of receipt. However, the destination can deny the receipt of the message by not replying any signed acknowledgment.

We can modify the simple protocol in the following way. The source node sends the destination a commitment to the message. Then the destination replies a receipt. Last, the source sends the message to the destination. In this case, the source can obtain evidence for nonrepudiation of receipt, but he can refuse to send the message to the destination and thus compromise the protocol.

From the two examples discussed above, we can see that a nonrepudiation protocol cannot be finished in a one-round handshake between the source and the destination. In Ref. [29], the simple protocol of one-round handshake is extended into multiple rounds. In each round, the source node sends evidence for nonrepudiation of origin and the destination replies an evidence for

nonrepudiation of receipt. Only after finishing all the rounds is the destination able to recover the message. The total number of rounds is randomly and secretly chosen by the source, and thus the destination does not know when the protocol finishes. In order to obtain the message, the destination has to reply an evidence for nonrepudiation of receipt to the source in each round until the source stops.

Another method in designing a nonrepudiation protocol is to use a *trusted third party* (TTP). The source and the destination have to interact with the TTP in an online or offline way so that the TTP collects enough evidences for both nonrepudiation of origin and nonrepudiation of receipt [30]. In any case of dispute, the TTP will act as a witness and help the adjudicator make a decision.

No matter which method is used, a nonrepudiation protocol is far more complex than most other network security protocols, for example, encryption or authentication. A nonrepudiation protocol without a TTP incurs too much communication and computation overhead. In a nonrepudiation protocol with a TTP, the TTP becomes a bottleneck of performance and a failure point of attackers' interest. In reality, nonrepudiation is presented only when its necessity is proved in crucial applications, for example, electronic contract signing, electronic voting, and so on. In WSNs, however, nonrepudiation is usually not considered because it is impossible for resource-constrained sensor nodes to support such complex protocols and also the most popular WSN applications, for example, environment monitoring or event tracking, do not require the nonrepudiation service.

12.6 FRESHNESS

All information describes a temporary status of an object and thus is valid in only a limited time interval. Therefore, when a node receives a packet, it needs to be assured that the packet is fresh. Otherwise, the packet is useless because the information conveyed in it is invalid.

12.6.1 Packet Replaying

Packet replaying is a major threat to the freshness requirement in network communications. An attacker can intercept a packet from a network, hold it for any amount of time, and then reply it into the network. The out-dated information contained in the packet can cause many problems to the applications deployed in the network. In a WSN, for example, a packet indicating the emergence of an event will conflict with an old packet containing no indication of the event. If some old routing control packets are replayed, sensor nodes will be put into a chaos about the network topology and thus the routing protocol will fail.

In addition to the replay in time dimension, packets can also be replayed in space dimension. An example is the Wormhole attack in WSNs [31]. In a

Wormhole attack, an attacker establishes a secret low-latency broadband channel between two distant locations in a WSN. Then he records packets at one location, tunnel them through the secret channel, and replay them at the other location. The replayed packets may still be fresh because the channel has a low latency. However, this abrupt spatial change can distort the network topology by making two distant nodes believe they are neighbors, thus becoming a serious attack to the routing protocols. In fact, these relayed packets should not be considered fresh because they arrive at the time earlier than the expected time.

12.6.2 Timestamp

In order to ensure the freshness of a packet, a timestamp can be attached to the packet. A receiving node can compare the timestamp in the packet with its own time clock and determine whether the packet is valid or not. In Ref. [31], for example, a timestamp is used to alleviate the Wormhole attack. In particular, each packet is securely attached with the location and time information on its source node. Each receiving node compares the location and timestamp in each received packet with its own location and time. If a packet arrives earlier than the expected time, a Wormhole attack may be present.

Timestamps can be used when all nodes in a network have synchronized clocks. This is a strict requirement in many scenarios, especially in WSNs. A method to avoid using timestamps is to use sequence numbers. In particular, each node chooses an initial number (usually zero) and an augmentation step (usually one). In each outgoing packet, a sequence number is embedded as the summation of the augmentation step and the sequence number in the previous outgoing packet. When the sequence number reaches its maximum, it is rounded back to its initial value and starts increasing again. A receiving node simply compares the sequence number in a received packet with the value recorded from the previous one and then determines whether the packet is fresh or not.

Using sequence numbers is simple, but ensures freshness only to some extent. The problem is that a sequence number can reach its maximum and return to its initial value. In order to alleviate this problem, the sequence number in a communication session usually has a very large range that can cover the whole lifetime of the session.

12.7 AVAILABILITY

Availability is an assurance of the ability to provide expected services as they are designed in advance. It is a very comprehensive concept in the sense that it is related to almost every aspect of a network. Any problem in a network can result in the degradation of the network functionality and thus compromise the network availability, leading to the DoS [12]. This section discusses several typical attacks against the availability.

12.7.1 Selective Forwarding

Forwarding packets is a major responsibility of a routing node. A malicious routing node, however, can intentionally drop any packet in passing traffic flows. In an extreme case, the malicious node can drop all the packets to act like a black hole. In a moderate case, the malicious node can drop the packets from some selected nodes and forward those from the other nodes. An even more subtle way is to drop packets intermittently so that the routing node is just like an unstable channel, which is difficult to detect.

If a routing node acts normally, the number of packets it needs to forward must be equal to the number of packets it receives. This observation is utilized in Ref. [32] to detect malicious routing nodes in WSNs. In particular, each sensor node can work under a promiscuous mode so that it can overhear the transmission of its neighboring nodes. If a neighbor of a suspected node finds that the number of packets that the suspected node fails to forward is larger than a certain threshold, the neighbor can collaborate with other neighbors of the suspected node into forming a decision on whether the suspected node is malicious or not, and what punishment is to be taken.

12.7.2 Radio Jamming

Jamming is a direct way to compromise network communications. A powerful attacker can constantly jam an intended spectrum band so that all the communications in the band are interrupted. For the sake of cost consideration, the attacker can also introduce intermittent radio interferences to degrade the condition of a channel of interest. The intermittent jamming can be very effective because only a change of one bit in a packet can make it dropped by the receiver.

It is very challenging to counteract radio jamming. Since jamming is done at the physical layer, all the security defense measures at the higher layers can do nothing about it. Currently, the only viable solution is to use specific physical-layer communication technologies, for example, directly spread spectrum, frequency hopping, or *ultra-wideband* (UWB). The idea is to spread the power of signals over a wide channel band so that jamming at a particular frequency point has a less impact on the communications. If the attacker can launch radio jamming in a broad spectrum range, those physical-layer technologies are still vulnerable to jamming. In this extreme case, what we can do is to identify the spatial region under the jamming attack and avoid the region by reestablishing new routes.

12.7.3 Multipath Routing

Multipath routing is an active approach to providing robust and secure data transmission services over insecure networks. It can alleviate the impact of packet dropping or radio jamming at the cost of increased routing overhead. Based on the secret sharing technique [33], a source node can decompose a message into many shares and spread those shares into multiple routes toward a destination

node. As long as the number of shares collected by the destination is larger than a threshold, the message can be recovered [34]. How many shares each route can be assigned depends on the security or the reliability of that route.

Multipath routing has many merits. A direct one is that it provides a certain level of resistance to packet loss due to channel variance, packet dropping, radio jamming, and so on. Even though some routes fail, a message also may be recovered at the destination. Moreover, the confidentiality of a message can be strengthened because attackers have to compromise a certain number of the routes to capture a message.

12.7.4 False Reports

A major application of WSNs is to detect and monitor some specific events of interest in a terrain. When an event comes out, the surrounding sensor nodes will have similar observations of the event. If all the sensor nodes send their notifications to the base station, they will result in too much energy waste. Therefore, the in-network processing is indispensible to fuse data in WSNs, where the surrounding nodes of an event cooperate with each other such that a final report can be generated by a data fusion node and sent by the fusion node to the base station. However, if some malicious nodes are involved in the data fusion procedure, they can send out false reports to disrupt the data fusion procedure. This injection of false reports by internal malicious nodes cannot be solved solely by authentication because those internal malicious nodes may have correct authentication keys. In order to detect and prevent false reports, multiple methods have been proposed.

In order to deal with false reports, witness nodes can be used [35] to provide data assurance. In particular, a data report should be checked and signed by t witness nodes, which have similar observations, before being forwarded by the fusion node, which also signs the report, to the base station. The base station will perform verification based on the received $t + 1$ signatures to decide whether the report is valid or not.

If false reports are always forwarded to the base station, a malicious node can keep sending false reports to deplete the resources of other forwarding nodes and the base station. In order to counteract this problem, en-route filtering is proposed in [36], where the signatures carried by a report are verified by the nodes along the route from the data fusion node to the base station, so that false reports may be filtered out before they arrive at the base station. This method requires that the nodes generating signatures have shared keys with those nodes on the route.

Most data fusion protocols organize sensor nodes into a tree hierarchy rooted at the base station, thus eliminating the data redundancy in the sensor data of a network. Hence, this reduces the communication cost and energy expenditure in data collection. The non-leaf nodes act as aggregators, fusing the data collected from their child nodes before forwarding the results toward the base station. In this way, data are processed and fused at each hop on the way to the base station, and communication overhead can be largely reduced.

During a normal hop-by-hop aggregation process in a tree topology, the high-level nodes (i.e., the nodes closer to the root) are of higher interest to attackers than the low-level nodes. In other words, if a compromised node is closer to the root, the bogus aggregated data from it will have a larger impact on the final result computed by the root. In order to alleviate this impact, the *secure data aggregation protocol* (SDAP) [37] takes the approach of reducing the trust on the high-level nodes, which is realized by using the principle of divide and conquer. More specifically, by using a probabilistic grouping method, SDAP dynamically partitions the topology tree into multiple logical groups (subtrees) of similar sizes. SDAP performs hop-by-hop aggregation in each logical group and generates one aggregate with a commitment from each group. Once a group commits its aggregate, this group cannot deny it later. After the base station has collected all the group aggregates, it then identifies the suspicious groups. Finally, each group under suspicion participates in an attestation process to prove the correctness of its group aggregate. The base station will discard the individual group aggregate if a group under attestation fails to support its earlier commitment made in the collection phase. The final aggregate is calculated over all the group aggregates that either are normal or have passed the attestation procedure. Since fewer nodes will be under a high-level node in a logical subtree, the potential security threat by a compromised high-level node is reduced.

Another proactive approach to deal with false reports is to design a resilient fusion function so that the impact of attacks is reduced as much as possible at the fusion node. In Ref. [38], statistical estimations are proposed to model the data fusion procedure so that the well-established estimation theory can be used to evaluate the resilience of the fusion function. A fusion function can be modeled as a random function $\hat{\Theta} = f(X_1, X_2, \ldots, X_n)$ and the fusion error can be computed as $rms(f) = \mathrm{E}\left[\left(\hat{\Theta} - \theta\right)^2\right]^{1/2}$, where X_1, X_2, … , X_n are the random variables and θ is the real value that should be reported. If k out of n samples are modified by an attacker, then the fusion error becomes $rms*(f, A) = \mathrm{E}\left[\left(\hat{\Theta} - \theta\right)^2\right]^{1/2}$, where A is a k-node attack and $\hat{\Theta}*$ is the fusion result based on the false samples. Then a fusion function is said to be (k, α)-resilient if $\max\{rms*(f, A)\} \le \alpha \cdot rms(f)$. The intuition is that the (k, α)-resilient functions, for a small α, are the ones that can be computed meaningfully and securely in the presence of up to k compromised or malicious nodes, a feature we desire to fight against false reports. This approach provides a sound theoretical basis for designing fusion functions satisfying specific security requirements.

12.7.5 Node Replication

In the node replication attack [39], an attacker intentionally puts many replicas of a compromised node at many places in a network to incur inconsistency. Like the Sybil attack, the node replication attack can also render attackers the abilities

to subvert data aggregation, misbehavior detection, and voting protocols by injecting false data or suppressing legitimate data. Conventional methods to detect the node replication attack usually include centralized computing based on node locations or the number of simultaneous connections, which is vulnerable to the single-point failure. Distributed detection of the node replication attack was proposed in Ref. [39], where each node is assumed to know its location and is required to send its location to a set of witness nodes. If a witness node finds a contradiction in the location claims of a suspected node identity, this suspected node identity must be replicated many times. The asymmetric key technique is used here to guarantee the authenticity of location claims. A similar approach is discussed in Ref. [15], where each node has a private key corresponding to its location and the location-based key can be used to detect the node replicas.

12.8 INTRUSION DETECTION

In many cases, no matter how carefully we design a security infrastructure for a network, attackers may still find a way to break into it and launch attacks from the inside of the network. If they just keep quiet to eavesdrop on traffic flows, they can stay safe without being detected. If they behave more actively to disrupt the network communications, there will be some anomalies, indicating the existence of malicious attacks. Intrusion detection mechanisms can detect malicious intruders based on those anomalies.

Usually, the neighbors of a malicious node are the first entities learning those abnormal behaviors. Therefore, it is convenient to let each node monitor its neighbors such that intrusion detection mechanisms can be triggered as soon as possible. The security protocol proposed in Ref. [40] uses local monitoring, in which a neighbor of both a sender and a receiver can oversee the communication behaviors of the receiver. If the receiver has any abnormal behavior on the received packets, it may be detected. If the number of abnormal behaviors is larger than a threshold, the neighbors of the detected malicious node refuse to receive packets from and send packets to it so that the malicious node is isolated from the network.

Cumulative observations of anomalies can be used to evaluate the honesty of sensor nodes. In Ref. [41], a reputation-based framework is established, in which each node holds reputations for other nodes. Based on the observations of whether other nodes are cooperative or not, those reputations are updated through an iterative procedure and are used as criteria to decide whether a node is malicious or not.

Continuous monitoring may be energy consuming, which is not desirable in WSNs. Therefore, a cluster-based detection approach is developed for WSNs in Ref. [42]. In this approach, a network is divided into clusters. Each cluster head monitors its cluster members. All the members in a cluster are further divided into groups and the groups take turns to monitor the cluster head.

Not all the sensor nodes keep monitoring, thus reducing the overall network energy cost.

12.9 KEY MANAGEMENT

Most security protocols are based on the cryptographic operations using keys. The security of a cryptographic system mainly relies on the secrecy of the key it uses. If an attacker can find the key, the entire system is broken because the attacker can use the key to decrypt the intercepted ciphertexts to find the original plaintexts by cryptanalyzing the eavesdropped packets transmitted over the wireless medium. Due to the existence of the redundancy of message sources in the real world, the attacker may know more or less information about the key used. Therefore, the sender and the receiver may need to update the key used between them from time to time. In a WSN, some sensor nodes may be captured by an attacker. As a result, the key information thus is accessible to the attacker and the attacker can use the information to launch other serious attacks. Therefore, how to securely manage the keys between the sender and the receiver is a very important issue.

In general, establishing a key between two nodes includes two steps. First, the two nodes are configured with some key materials as well as necessary cryptographic algorithms. Second, the two nodes perform several rounds of communications to agree on a pairwise shared key computed based on their key materials. According to the algorithms used to establish a pairwise key, current key management solutions can be classified into symmetric key schemes and asymmetric key schemes.

12.9.1 Symmetric Key Management

A simple symmetric key scheme is to configure all the nodes with a globally shared key [43]. This scheme is secure to external attackers that do not know the key, but not to internal attackers because the key can be exposed if any node is compromised.

In a WSN, the base station can act as a centralized *key distribution center* (KDC) [24]. In particular, each sensor node shares a unique key with the base station. If two nodes need to communicate securely, they can acquire a shared key from the base station who unicasts the key to each of them. This centralized approach may incur a large amount of communication overhead because two close nodes may have to do handshakes through the KDC at a distant place. In addition, the KDC may become a potential point of failure in that the entire network is broken down if the server is corrupted by an attacker.

Most recent solutions to key establishment in WSNs follow a distributed approach, where every sensor node is configured with some key material, whereby it establishes shared keys with other nodes. There are two components in this approach: one is how to distribute key materials, and the other is how to establish

a shared key with key materials. Next, we discuss basic key agreement models and then present various methods to distribute key materials.

12.9.1.1 Key Agreement Models. To let a pair of sensor nodes share a key, the simplest way is to configure them with a shared key before they are deployed into a WSN. The pair of nodes can use the shared key directly after deployment. To guarantee that every pair of nodes in a network of N nodes has a unique shared key, each node needs to store $N - 1$ keys. As the size of the network increases, this number becomes unacceptably large. Therefore, some scalable methods are desirable.

Blom [44] proposed a key agreement method based on $(N, t + 1)$ linear codes. In particular, a KDC first constructs a $(t + 1) \times N$ public matrix P over a finite field, and then selects a random $(t + 1) \times (t + 1)$ symmetric matrix S over the same finite field, where S is secret and only known to the KDC. An $N \times (t + 1)$ matrix $A = (S \cdot P)^T$ is computed, where $(\cdot)^T$ denotes the transpose operator. Because S is symmetric, it is easy to see that

$$K = A \cdot P = (S \cdot P)^T \cdot P = P^T \cdot S^T \cdot P = P^T \cdot S \cdot P = (A \cdot P)^T = K^T.$$

Node pair (i, j) will use K_{ij}, the element in row i and column j of K, as a shared key. Because K_{ij} is calculated as the product of the ith row of A and the jth column of P, the KDC assigns the ith row of A and the jth column of P to node i. Therefore, when nodes i and j need to establish a shared key, they first exchange their columns of P and then compute K_{ij} and K_{ji}, respectively, using their private rows of A.

Blom's scheme has a t-secure property in the sense that in a network of N nodes the collusion of $< t + 1$ nodes cannot reveal any key shared by other pairs of nodes. The memory cost per node in Blom's scheme is $t + 1$. Therefore, Blom's scheme trades the security for the memory cost. To guarantee perfect security in a network of N nodes, the $(N - 2)$-secure Blom's scheme should be used, which means that the memory cost per node is $N - 1$.

Blundo et al. [45] proposed to use a t-degree bivariate symmetric polynomial to achieve key agreement. It is a special case of Blom's scheme in that a Vandermonde matrix is used as the generator matrix of linear code. A t-degree bivariate polynomial is defined as $f(x, y) = \sum_{i=0}^{t} \sum_{j=0}^{t} a_{ij} x^i y^j$ over a finite field. When $a_{ij} = a_{ji}$ holds, we can know that $f(x, y) = f(y, x)$ holds. Assume that each node has a unique, integer-valued, and non-zero identity. For a pair of nodes u and v, the KDC assigns a polynomial share $f(u, y)$ to u and another share $f(v, y)$ to v. In order to establish a shared key, both nodes broadcast their IDs. Subsequently, node u computes $f(u, v)$ and node v computes $f(v, u)$. Due to the polynomial symmetry, the shared key has been established as $K_{uv} = f(u, v) = f(v, u)$. Like Blom's scheme, a t-degree bivariate polynomial is also t-secure, meaning that attackers have to compromise $\geq t + 1$ nodes holding shares of the same polynomial to reconstruct it.

12.9.1.2 Random Key Material Distribution. The key agreement models described above can guarantee that every pair of nodes in a network of N nodes has a unique shared key, but the cost is that each node needs to store $N - 1$ keys. This is impractical for a WSN due to the memory constraints of sensor nodes and the possible large scale of the network. Instead, most recent research papers in this field relax the security requirement in the sense that key materials are predistributed such that some sensor nodes can establish shared keys directly and then help to establish indirect shared keys between other sensor nodes.

A typical scheme is the random key predistribution (called RKP hereafter) [46], in which each node is configured with a subset of keys, called a key ring, randomly selected from a global pool of keys, such that any pair of neighboring nodes can share at least one key with a certain probability. After deployment, two neighboring nodes can have a shared key directly or negotiate an indirect key through a secure path, along which every pair of neighboring nodes has a direct shared key.

The theoretical foundation of RKP is the random graph theory. A random graph $G(n, p)$ is a graph of n nodes in which the probability that a link exists between two nodes is p. The graph does not have any edge if $p = 0$ or is fully connected if $p = 1$. There is a transition from the non-connected graph to the fully connected graph, when p increases. RKP exploits this property by setting p larger than a certain value, such that the network is almost connected. Here the size of the global key pool and the size of the key subset for an individual node can be tuned to achieve such a property.

A major concern of RKP is node compromise. The random selection of a key ring for each node means the reuse of each key by multiple nodes. An attacker may compromise a node and expose its key ring, out of which some keys may be used by other non-compromised nodes. This leads to the failures of the links among those non-compromised nodes.

To mitigate the impact of node compromise, several schemes have been proposed. The q-composite RKP [47] follows RKP except that any pair of neighboring nodes is required to share at least q keys with a certain probability. It can improve the resilience to node compromise when the number of compromised nodes is small. Unfortunately, it is not effective when the number is large.

Another problem of RKP is the lack of authentication because of the reuse of the same key by multiple nodes. To solve the problem, node identity information is used to derive key rings for sensor nodes [48]. A similar approach is taken in the *Random-Pairwise Key* (RPK) [47] scheme, where each node keeps a set of keys, each of which is uniquely shared with another node. In Refs. [49,50], a global pool of Blom's matrices or polynomials is used to replace the global key pool in Ref. [46]. In those schemes, each key is tied to the identities of the nodes sharing it. In this way, the identity of a node can be verified through the normal challenge-response protocol.

RKP requires the storage of a key ring by each node to make the network almost connected. In some cases, where sensor nodes do not have enough memory

resource, this becomes a problem. In Ref. [51], RKP and its variations are revisited into new ones, in which the amount of key materials that each node keeps is reduced. Therefore, the probability p of direct key sharing is smaller than that required in RKP. The smaller p cannot guarantee that the network is almost connected, but only assures that the network consists of multiple isolated clusters, of which there is one largest cluster connecting most nodes. In this way, less memory cost still achieves a certain network connectivity, but the trade-off is that some small clusters of nodes are isolated because they do not share keys with the largest one.

12.9.1.3 Deterministic Key Material Distribution. According to the underlying random-graph theory, the probabilistic nature of the random distribution of key materials cannot guarantee that two neighboring nodes establish a shared key. For this reason, some sensor nodes may not be able to establish shared keys with their neighbors and thus are isolated, which is not desirable. In order to solve the problem, two deterministic approaches have been developed.

One approach is to use a strongly regular graph or a complete graph to replace the random graph to do key configuration [52,53]. In a strongly regular graph (n, r, λ, μ), there are n nodes, each of which has a degree of r and any pair of which has λ common neighbors when they are adjacent and μ common neighbors when they are nonadjacent. In the strongly regular graph, every pair of nodes is connected through a path. Each link (or edge) can be assigned with a unique key that is preloaded into the two end vertices (nodes). Besides the regular graph, the block design in the set theory can be used in key predistribution, in which all the nodes form a complete graph at the network layer. The tool is the *balanced incomplete block design* (BIBD). A (v, r, λ)-BIBD is an arrangement of v objects into many blocks such that each block contains r distinct objects and every pair of objects occurs in exactly λ blocks. For example, when an $(n^2 + n + 1, n + 1, 1)$-BIBD is applied in a WSN, each sensor node is preloaded with $n + 1$ keys, which form a block out of a pool of $n^2 + n + 1$ keys, and every pair of nodes has one common key.

The other approach is to use a multidimensional grid to replace the random graph [54,55]. In particular, each node is assigned an identity $(n_1, n_2, \ldots, n_k)$ such that all the nodes form a k-dimensional grid. A t-degree $(k + 1)$-variate symmetric polynomial

$$f(x_1, x_2, \ldots, x_k, x_{k+1}) = f(x_{\sigma(1)}, x_{\sigma(2)}, \ldots, x_{\sigma(k)}, x_{\sigma(k+1)})$$

for any permutation σ: $\{1, 2, \ldots, k + 1\} \rightarrow \{1, 2, \ldots, k + 1\}$ is used to compute the share

$$f_1(x_{k+1}) = f(n_1, n_2, \cdots, n_k, x_{k+1}) = \sum_{i_{k+1}=0}^{t} b_{i_{k+1}} x_{k+1}^{i_{k+1}}$$

for node $(n_1, n_2, \ldots, n_{k+1})$. If node u with identity $(u_1, u_2, \ldots, u_k)$ and node v with identity $(v_1, v_2, \ldots, v_k)$ have only one component mismatch in their identities (similar to the case that the Hamming distance between the two bit patterns is one), say $u_i \neq v_i$ for some i, but $u_j = v_j = c_j$ for other $j \neq i$, then nodes u and v can compute a shared key as

$$K_{uv} = f(c_1, \ldots, u_i, \ldots, c_k, v_i) = f(c_1, \ldots, v_i, \ldots, c_k, u_i).$$

For two nodes with more than one component mismatches in their IDs, they can find a path to negotiate an indirect key because all the nodes are organized in a grid.

In those deterministic schemes, a node can find whether it has a direct shared key with another node based on the identity of that node. This can provide an authentication service in that the identity of a node can be challenged based on its keys that are related to its identity.

12.9.1.4 Location-Based Key Material Distribution. In the aforementioned random and deterministic key material distribution schemes, key materials are uniformly distributed in the entire terrain of a network. The uniform distribution makes the probability that two neighboring nodes share a direct key at one hop, that is, local secure connectivity, rather small. Therefore, a lot of communication overhead is inevitable for the establishment of indirect keys over multihop paths. To improve the local secure connectivity, many researchers proposed to involve location information into key establishment.

In Ref. [56], the entire sensor network is divided into square cells. Each cell is associated with a unique t-degree bivariate polynomial. Each sensor node is preloaded with shares of the polynomials of its home cell and four other cells horizontally and vertically adjoining to its home cell. After deployment, any two neighboring nodes can establish a pairwise key according to the polynomial scheme [45] if they have shares of the same polynomial. There are other schemes simply replacing the polynomial model with other RKP schemes, for example, RKP [46] in Ref. [57] and RPK[47] and MSKP [49] in Ref. [58].

Besides the square cells used in previous schemes, hexagon [59] and triangle [60] grid models are also investigated to improve spatial diversity. Unlike those cell-based key material distribution, the cell-pair-based method is investigated, in which each pair of neighboring cells is associated with a unique t-degree bivariate polynomial or a matrix. It has been shown in Refs. [59,60] that distributing to each pair of neighboring cells instead of each individual cell can reduce the memory cost while increasing the resilience to node compromise. In addition, Zhou and Fang [61,62] combined the location information with the scalable key agreement model [54,55] and developed more secure and efficient link-layer and transport-layer key establishment schemes. Their schemes can significantly improve the security and reduce the memory cost while still maintaining high secure connectivity.

Another approach is not cell based. On the contrary, it estimates the nodes that are supposed to be deployed close to each other and then preloads them with related key materials. In this way, these close nodes may have shared keys between them after deployment. In Ref. [63], a group-based key predistribution framework is established, which may incorporate previous schemes. It divides the entire network into many deployment groups. In each group, a specific keying material distribution scheme can be applied to provide the in-group connectivity. A node is picked from each group and all those picked nodes form a cross-group. There is also a specific keying material distribution scheme for each cross-group. Therefore, two nodes from different deployment groups can establish a shared key through a path in a cross-group. A similar approach is taken in Refs. [64,65]. The difference is that all the nodes in one group are preloaded with pairwise keys with each other and each node may be preloaded with a pairwise key shared with a node in another group. Those preloaded keys can build up a secure path between any two nodes. In Ref. [66], a key-position map is used to map a location with a key. If a node is expected to reside in an area according to some probabilistic distribution, it is preloaded with some keys corresponding to some randomly selected locations around the expected resident point. Therefore, if two nodes are expected close to each other, they more likely share a common key.

12.9.1.5 Comparison of Symmetric Key Schemes. Here we carry out comparisons between the random [46–51], deterministic [52–55], and location-based [56–66] schemes. The comparisons mainly focus on the following three aspects:

1. Memory cost. The memory resource of sensor nodes is scarce. We cannot distribute too much key material into each node. The memory cost should be as small as possible.
2. Resilience to node compromise. Usually it is unavoidable to prevent an attacker from compromising some nodes. We can do nothing to rescue those compromised nodes. However, a good scheme should reduce the impact of the node compromise attack on other normal nodes as much as possible. In particular, the attacker can learn the keys that the compromised node uses to communicate with other nodes, but he should not be able to learn other keys that the compromised node does not know so that the communications between other normal nodes are still safe.
3. Local secure connectivity. Since each node cannot store too much key material, it is usually able to establish shared keys with a subset of its neighboring nodes. Local secure connectivity is the probability that two neighboring nodes establish a shared key directly, that is, the portion of neighbors with whom a node can establish shared keys in 1-hop. It is directly related to the communication overhead of key establishment. In WSNs, high local secure connectivity is desirable because it means that each node does not need to spend too much energy on the establishment of indirect keys with neighbors through multihop routing, thus saving a lot of communication overhead.

TABLE 12.1 Memory Cost

Key Material Distribution	References	Memory Cost
Random	46–51	$O(N)$
Deterministic	52,53	$O(N)$ or $O(\sqrt{N})$
	54,55	$O(\sqrt[k]{N})$
Location Based	56–60, 63–66	$O(N)$ or $O(\sqrt{N})$
	61,62	$O(\sqrt[k]{N})$

TABLE 12.2 Link Compromise Probability

Key Agreement	References	Link Compromise Probability
Predistributed Keys	46–48, 51–53, 57, 63–66	Approximately linear or quickly increasing
Matrics or Polynomials	49,50, 54–56, 58–62	Threshold Based

Table 12.1 shows the memory costs of different schemes. The random distribution schemes [46–51] require that each node store a key ring. In order to maintain a certain level of connectivity, the size of a key ring cannot be small and is usually on the order of $O(N)$. The graph-based deterministic distribution schemes [52,53] also require that each node store a key ring. The memory cost of these schemes is either $O(N)$ for regular graph or $O(\sqrt{N})$ for BIBD design. The grid-based deterministic schemes [54,55] have the memory cost only on the order of $O(\sqrt[k]{N})$, where $k > 1$, because they use k-dimension grid to organize the network. Most location-based schemes [56–60, 63–66] combine the location information and random distribution schemes, and have less memory cost than the random distribution schemes. Their memory cost is on the order of $O(N)$ or $O(\sqrt{N})$. One exception is in Refs. [61,62], where the location information and the deterministic scheme [54,55] are combined, and thus the memory cost of [61,62] is on the order of $O(\sqrt[k]{N})$.

Table 12.2 shows the resilience to node compromise of different schemes. The probability of link compromise is used to evaluate the resilience to node compromise because the key information in compromised nodes can be used to derive the keys used by the links between non-compromised nodes. For the schemes [46–48, 51–53, 57, 63–66] in which keys are directly predistributed, the link compromise probability is approximately linear or quickly increasing with respect to the number of compromised nodes, because every time one more node is compromised, more keys from the global key pool are disclosed. However, the matrices or polynomials-based schemes [49, 50, 54–56, 58–62] have a nice property of threshold-based resilience, which means that the network can tolerate up

TABLE 12.3 Local Secure Connectivity

Key Material Deployment	References	Local Secure Connectivity
Uniform	46–55	Low (<0.5)
Location Based	56–66	High (>0.5)

to a certain number of compromised nodes while still keeping the links between non-compromised nodes safe.

Table 12.3 shows the local secure connectivity of different schemes. The local secure connectivity of the uniform distribution schemes [46–55] is lower than that of the location-based schemes [56–66]. Therefore, by combining location information, each node can establish direct keys with more neighboring nodes whereby to save energy on the establishment of indirect keys through multihop paths with other neighbors.

12.9.2 Asymmetric Key Management

Though it is much more computationally expensive, the asymmetric key technique is easier to manage and more resilient to node compromise than the symmetric key technique. Each node can keep its private key secret and only publish its public key. Therefore compromised nodes cannot provide any clue to the private keys of non-compromised nodes.

Since asymmetric key algorithms are expensive, they cannot be used in encryption all the time. Usually, the asymmetric key technique is used to establish a shared key between two nodes. The two nodes then use the shared key in encryption so that the computational advantage of the symmetric key technique can be taken. In order to establish a shared key, one node can use the other node's public key to encrypt a randomly selected secret pairwise key and send it to the other node. Only the receiving node can learn the secret pairwise key because no one else knows its private key.

Diffie–Hellman is the most popular asymmetric key algorithm in key exchange [4]. Suppose that two nodes a and b agree on a large prime number p and a number g less than p. Node a selects a secret number x as its private key and computes its public key as $g^x \bmod p$. Node b as well selects its private key as y and computes its public key as $g^y \bmod p$. After exchanging their public keys, nodes a and b can compute a shared key as $K_{ab} = g^{xy} \bmod p$. If an attacker attempts to compromise the shared key, he needs to compute x from $g^x \bmod p$ or y from $g^y \bmod p$. This is a *discrete logarithm problem* (DLP), which is very difficult to solve.

Diffie–Hellman is widely used in the Internet security protocols, such as IPSec [67] and TLS/SSL [68]. In WSNs, however, it cannot be directly used because of the limited computation capability of sensor nodes. Some simplifications are necessary. In TinyPK [16], the Diffie–Hellman algorithm is used to

exchange keys between sensor nodes, where the base of exponentiation is chosen as $g = 2$ such that the exponential operation is simplified. In Ref. [19], the ECC based Diffie–Hellman algorithm is used to establish shared keys between sensor nodes, in which the computation efficiency comes from the elliptic curve cryptography.

The public keys used in Diffie–Hellman need to be authenticated. Otherwise, the MiM attack is possible. This issue can be addressed by public key certificates or identity-based cryptography, as discussed in Section 12.9.1.

12.9.3 Group Key Management

According to different communication scales, there are two types of group communication. One is global broadcast/multicast. Usually, it is done by a base station because the network broadcast/multicast requires huge communication overhead, which cannot be supported by sensor nodes. The other is local broadcast, where each node collaborates with its neighbors to fulfill various purposes, for example, routing information exchange or cluster head selection. Both types require a group key to encrypt communications.

LEAP is a group key management protocol for WSNs, which identifies a group key for global broadcast and cluster keys for local broadcast [69]. A group key is a key shared by all the nodes in the network. To tolerate node compromise, the group key is updated occasionally based on μTESLA [24]. A cluster key is a key shared by a node and all its neighbors, and it is mainly used for securing locally broadcast messages. A node encrypts a cluster key with the pairwise key shared with each neighbor and unicasts it to the neighbor. Because the number of neighbors is usually small, this unicast does not incur too much communication overhead.

A problem with LEAP is that each node learns a group key from the base station individually. If an attacker compromises a node, the group key is exposed. To mitigate this threat, local collaboration is introduced in the group key distribution [70]. In particular, each node has to get extra secret information from its neighbors, as well as the broadcast information from the base station. Only by combining that secret information and its own secrets preloaded before deployment, can each node recover the group key. If a node is detected by its neighbors as a malicious one, its neighbors will not collaborate with it. In this way, a malicious node can encounter difficulty in calculating the global key.

In most time, a base station takes charge of group key management for the entire network. This may introduce too much management overhead at the base station. In order to reduce the overhead, a level key infrastructure for group communications is proposed in Ref. [71]. In particular, all the nodes involved in broadcast/multicast are organized in a tree rooted at a base station, and each parent node takes charge of key update for its immediate children nodes. In this way, the overhead of key management is localized, which is different from centralized group key-management schemes, for example, LEAP [69].

12.10 SUMMARY

Security is becoming a major concern for WSN protocol designers because of the broad security-critical applications of WSNs. This chapter discussed general security problems in WSNs and introduced corresponding solutions. On one hand, WSNs are still under development, and many protocols designed so far for WSNs have not taken security into consideration. On the other hand, the salient features of WSNs make it very challenging to design strong security protocols while still maintaining low overheads. Therefore, network security for WSNs is still a very fruitful research direction to be further explored.

ACKNOWLEDGMENTS

This work was supported in part by the US National Science Foundation under grants CNS-0716450 and CNS-0626881.

REFERENCES

[1] FIPS PUB 46-2, "Data encryption standard (DES)", Dec. 1993.

[2] FIPS PUB197, "Advanced encryption standard (AES)", Nov. 2001.

[3] IETF RFC 2040, "The rc5, rc5-cbc, rc5-cbc-pad, and rc5-cts algorithms", Oct. 1996.

[4] W. Diffie and M. E. Hellman, "New directions in cryptography", *IEEE Transactions on Information Theory*, vol. IT-22, no. 6, Nov. 1976, pp. 644–654.

[5] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems", *Communications of the ACM*, vol. 21, no. 2, Feb. 1978, pp. 120–126.

[6] C. Karlof, N. Sastry, and D. Wagner, "TinySec: A link layer security architecture for wireless sensor networks", in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys'04)*, Baltimore, MD, Nov. 2004, pp. 162–175.

[7] National Security Agency, "Skipjack and KEA algorithm specifications", May 1998.

[8] J. Deng, R. Han and S. Mishra, "Enhancing base station security in wireless sensor networks", Technical Report CU-CS-951–03, Department of Computer Science, University of Colorado, Apr. 2003.

[9] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "MASK: Anonymous on-demand routing in mobile ad hoc networks", *IEEE Transactions on Wireless Communications*, vol. 5, no. 9, Sept. 2006, pp. 2376–2385.

[10] P. Kamat, Y. Zhang, W. Trappe, and C. Ozturk, "Enhancing source-location privacy in sensor network routing", in *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS'05)*, Columbus, OH, June 2005, pp. 599–608.

[11] Y. Jian, S. Chen, Z. Zhang, and L. Zhang, "Protecting receiver-location privacy in wireless sensor networks", in *Proceedings of IEEE INFOCOM'07*, Anchorage, AK, May 2007, pp. 1955–1963.

[12] A. Wood and J. Stankovic, "Denial of service in sensor networks", *IEEE Computer Magazine*, vol. 35, no. 10, Oct. 2002, pp. 54–62.

[13] J. Newsome, E. Shi, D. Song, and A. Perrig, "The Sybil attack in sensor networks: Analysis & defenses", in *Proceedings of the 3rd IEEE International Symposium on Information Processing in Sensor Networks (IPSN'04)*, Berkeley, CA, Apr. 2004, pp. 259–268.

[14] M. Bellare, R. Canetti and H. Krawczyk, "Keying hash functions for message authentication", in *Proceedings of Advances in Cryptology (Crypto'96), Lecture Notes in Computer Science*, Springer-Verlag, vol. 1109, June 1996, pp. 1–15.

[15] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "Location-based compromise-tolerant security mechanisms for wireless sensor networks", *IEEE Journal on Selected Areas in Communications, Special Issue on Security in Wireless Ad-Hoc Networks*, vol. 24, no. 2, Feb. 2006, pp. 247–260.

[16] R. Watro, D. Kong, S. Cuti, C. Gardiner, C. Lynn, and P. Kruus, "TinyPK: Securing sensor networks with public key technology", in *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN'04)*, Washington, DC, Oct. 2004, pp. 59–64.

[17] N. Koblitz, "Elliptic curve cryptosystems", *Mathematics of Computation*, vol. 48, 1987, pp. 203–209.

[18] V. Miller, "Uses of elliptic curves in cryptography", in *Proceedings of Advances in Cryptology (Crypto'85)*, Lecture Notes in Computer Science, Springer-Verlag, vol. 218, 1986, pp. 417–426.

[19] Y. Zhou, Y. Zhang and Y. Fang, "Access control in wireless sensor networks", *Elsevier Ad Hoc Networks Journal, Special Issue on Security in Ad Hoc and Sensor Networks*, vol. 5, no. 1, Jan. 2007, pp. 3–13.

[20] S. Vanstone, "Responses to NIST's proposal", *Communications of the ACM*, vol. 35, no. 7, July 1992, pp. 50–52.

[21] W. Du, R. Wang, and P. Ning, "An efficient scheme for authenticating public keys in sensor networks", in *Proceedings of the 6th ACM International Symposium on Mobile Ad hoc Networking and Computing (MobiHoc'05)*, Urbana-Champaign, IL, May 2005, pp. 58–67.

[22] R. Merkle, "Protocols for public key cryptosystem", in *Proceedings of 1980 IEEE Symposium on Research in Security and Privacy (SP'80)*, Los Alamitos, CA, Apr. 1980.

[23] D. Boneh and M. Franklin, "Identify-based encryption from the weil pairing", in *Proceedings of Advances in Cryptology (Crypto'01), Lecture Notes in Computer Science*, Springer-Verlag, vol. 2139, 2001, pp. 213–229.

[24] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "SPINS: Security protocols for sensor networks", *ACM Wireless Networks*, vol. 8, no. 5, Sept. 2002, pp. 521–534.

[25] Y. Zhou and Y. Fang, "BABRA: Batch-based broadcast authentication in wireless sensor networks", in *Proceedings of the 49th Annual IEEE Global Telecommunications Conference (GLOBECOM'06)*, San Francisco, CA, Nov. 2006, pp. 1–5.

[26] Y. Challal, H. Bettahar, and A. Bouabdallah, "A taxonomy of multicast data origin authentication: issues and solutions", *IEEE Communication Surveys & Tutorials*, vol. 6, no. 3, Oct. 2004, pp. 34–57.

[27] Y. Zhou and Y. Fang, "Multicast authentication over lossy channels", in *Proceedings of 2007 IEEE Military Communications Conference (MILCOM'07)*, Orlando, FL, Oct. 2007, pp. 1–7.

[28] Y. Zhou and Y. Fang, "Multimedia broadcast authentication based on batch signature", *IEEE Communications Magazine*, vol. 45, no. 8, Aug. 2007, pp. 72–77.

[29] O. Markowitch and Y. Roggeman, "Probabilistic non-repudiation without trusted third party", in *Proceedings of the 2nd Conference on Security in Communicaiton Networks (SCN'99)*, Amlfi, Italy, Sept. 1999, pp. 27–38.

[30] S. Kremer, O. Markowitch, and J. Zhou, "An intensive survey of fair non-repudiation protocols", *Elsevier Computer Communications*, vol. 25, no. 17, Nov. 2002, pp. 1606–1621.

[31] Y. Hu, A. Perrig and D. B. Johnson, "Packet leashes: A defense against wormhole attacks in wireless networks", in *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'03)*, vol. 3, San Francisco, CA, Mar. 2003, pp. 1976–1986.

[32] G. Wang, W. Zhang, G. Cao, and T. La Porta, "On supporting distributed collaboration in sensor networks", in *Proceedings of 2003 IEEE Military Communications Conference (MILCOM'03)*, vol. 2, Boston, MA, Oct. 2003, pp. 752–757.

[33] A. Shamir, "How to share a secret", *Communications of the ACM*, vol. 22, no. 11, Nov. 1979, pp. 612–613

[34] W. Lou, W. Liu, and Y. Fang, "SPREAD: Enhancing data confidentiality in mobile ad hoc networks", in *Proceedings of the 23rd Annual IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'04)*, vol. 4, Hong Kong, China, Mar. 2004, pp. 2404–2413.

[35] W. Du, J. Deng, Y. S. Han, and P. Varshney, "A witness-based approach for data fusion assurance in wireless sensor networks", in *Proceedings of 2003 IEEE Global Communications Conference (GLOBECOM'03)*, vol. 3, San Francisco, CA, Dec. 2003, pp. 1435–1439.

[36] F. Ye, H. Luo, S. Lu, and L. Zhang, "Statistical en-route filtering of injected false data in sensor networks", in *Proceedings of the 23rd Annual IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'04)*, vol. 23, no. 4, Hong Kong, China, Mar. 2004, pp. 839–850.

[37] Y. Yang, X. Wang and S. Zhu, "SDAP: A secure hop-by-hop data aggregation protocol for sensor networks", in *Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing (Mobihoc'06)*, Florence, Italy, May 2006, pp. 356–367.

[38] D. Wagner, "Resilient aggregation in sensor networks", in *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN'04)*, Washington, DC, Oct. 2004, pp. 78–87.

[39] B. Parno, A. Perrig, and V. Gligor, "Distributed detection of node replication attacks in sensor networks", in *Proceedings of 2005 IEEE Symposium on Security and Privacy (SP'05)*, Oakland, CA, May 2005, pp. 49–63.

[40] I. Khalil, S. Bagchi, and C. Nita-Rotaru, "DICAS: Detection, diagnosis and isolation of control attacks in sensor networks", in *Proceedings of the 1st IEEE International Conference on Security and Privacy for Emerging Areas in Communications Networks (SecureComm'05)*, Athens, Greece, Sept. 2005, pp. 89–100.

[41] S. Ganeriwal and M. Srivastava, "Reputation-based framework for high integrity sensor networks", in *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN'04)*, Washington, DC, Oct. 2004, pp. 66–77.

[42] C-C. Su, K-M. Chang, Y-H. Kuo, and M-F. Horng, "The new intrusion prevention and detection approaches for clustering-based sensor networks", in *Proceedings of 2005 IEEE Wireless Communications and Networking Conference (WCNC'05)*, vol. 4, New Orleans, LA, Mar. 2005, pp. 1927–1932.

[43] S. Basagni, K. Herrin, D. Bruschi, and E. Rosti, "Secure pebblenets", in *Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking and Computing (Mobihoc'01)*, Long Beach, CA, Oct. 2001, pp. 156–163.

[44] R. Blom, "An optimal class of symmetric key generation systems", in *Proceedings of Advances in Cryptology: EUROCRYPT'84, Lecture Notes in Computer Science*, Springer-Verlag, vol. 209, 1985, pp. 335–338.

[45] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly-secure key distribution for dynamic conferences", in *Proceedings of Advances in Cryptology (CRYPTO'92), Lecture Notes in Computer Science*, Springer-Verlag, vol. 740, 1992, pp. 471–486.

[46] L. Eschenauer and V. Gligor, "A key management scheme for distributed sensor networks", in *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS'02)*, Washington DC, Nov. 2002, pp. 41–47.

[47] H. Chan, A. Perrig, and D, Song, "Random key predistribution schemes for sensor networks", in *Proceedings of 2003 IEEE Symposium on Security and Privacy*, May 2003, pp. 197–213.

[48] R. D. Pietro, L. V. Mancini, and A. Mei, "Random key-assignment for secure Wireless Sensor Networks", in *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS'03)*, Washington DC, Oct. 2003, pp. 62–71.

[49] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, "A pairwise key pre-distribution scheme for wireless sensor networks", in *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS'03)*, Washington DC, Oct. 2003, pp. 42–51.

[50] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks", in *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS'03)*, Washington DC, Oct. 2003, pp. 52–61.

[51] J. Hwang and Y. Kim, "Revisiting random key pre-distribution schemes for wireless sensor networks", in *Proceedings of the 2nd ACM Workshop on Security of Ad hoc and Sensor Networks (SASN'04)*, Washington, DC, Oct. 2004, pp. 43–52.

[52] J. Lee and D. R. Stinson, "A combinatorial approach to key pre-distribution mechanisms for wireless sensor networks", in *Proceedings of 2005 IEEE Wireless Communications and Networking Conference (WCNC'05)*, New Orleans, LA, Mar. 2005, pp. 1200–1205.

[53] S. A. Camtepe and B. Yener, "Combinatorial design of key distribution mechanisms for wireless sensor networks", in *Proceedings of the 9th European Symposium on Research in Computer Security (ESORICS'04)*, Sophia Antipolis, France, Sept. 2004, pp. 293–308.

[54] Y. Zhou and Y. Fang, "A scalable key agreement scheme for large scale networks", in *Proceedings of 2006 IEEE International Conference on Networking, Sensing and Control (ICNSC'06)*, Ft. Lauderdale, FL, Apr. 2006, pp. 631–636.

[55] Y. Zhou and Y. Fang, "Scalable and deterministic key agreement for large scale networks", IEEE Transactions on Wireless Communications, vol. 6, no. 12, Dec. 2007, pp. 4366–4373.

[56] D. Liu and P. Ning, "Location-based pairwise key establishments for relatively static sensor networks", in *Proceedings of 2003 ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN'03)*, Fairfax, VA, Oct. 2003, pp. 72–82.

[57] W. Du, J. Deng, Y. S. Han, S. Chen, and P. K. Varshney, "A key management scheme for wireless sensor networks using deployment knowledge", in *Proceedings of the 23th Annual IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'04)*, Hong Kong, Mar. 2004, pp. 586–597.

[58] D. Huang, M. Mehta, D. Medhi, and L. Harn, "Location-aware key management scheme for wireless sensor networks", in *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN'04)*, Washington, DC, Oct. 2004, pp. 29–42.

[59] Y. Zhou, Y. Zhang, and Y. Fang, "LLK: A link-layer key establishment scheme in wireless sensor networks", in *Proceedings of 2005 IEEE Wireless Communications and Networking Conference (WCNC'05)*, New Orleans, LA, Mar. 2005, pp. 1921–1926.

[60] Y. Zhou, Y. Zhang, and Y. Fang, "Key establishment in sensor networks based on triangle grid deployment model", in *Proceedings of 2005 IEEE Military Communications Conference (MILCOM'05)*, Atlantic City, NJ, Oct. 2005, pp. 1450–1455.

[61] Y. Zhou and Y. Fang, "Scalable link-layer key agreement in sensor networks", in *Proceedings of 2006 IEEE Military Communications Conference (MILCOM'06)*, Washington, DC, Oct. 2006, pp. 1–6.

[62] Y. Zhou and Y. Fang, "A two-layer key establishment scheme for wireless sensor networks", *IEEE Transactions on Mobile Computing*, vol. 6, no. 9, Sept. 2007, pp. 1009–1020.

[63] D. Liu, P. Ning, and W. Du, "Group-based key pre-distribution in wireless sensor networks", in *Proceedings of the 4th ACM Workshop on Wireless Security (WiSe'05)*, Cologne, Germany, Sept. 2005, pp. 11–20.

[64] L. Zhou, J. Ni, and C. V. Ravishankar, "Efficient key establishment for group-based wireless sensor deployments", in *Proceedings of the 4th ACM Workshop on Wireless Security (WiSe'05)*, Cologne, Germany, Sept. 2005, pp. 1–10.

[65] L. Zhou, J. Ni and C. V. Ravishankar, "Supporting secure communication and data collection in mobile sensor networks", in *Proceedings of the 25th Annual IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'06)*, Barcelona, Catalunya, Spain, Apr. 2006, pp. 1–12.

[66] T. Ito, H. Ohta, N. Matsuda, and T. Yoneda, "A key pre-distribution scheme for secure sensor networks using probability density function of node deployment", in *Proceedings of the 3rd ACM Workshop on Security of Ad hoc and Sensor Networks (SASN'05)*, Alexandria, VA, Nov. 2005, pp. 69–75.

[67] IETF RFC 2401, Security architecture for the Internet protocol, Nov. 1998.

[68] IETF RFC 4346, The transport layer security (TLS) protocol, version 1.1, Apr. 2006.

[69] S. Zhu, S. Setia, and S. Jajodia, "LEAP: Efficient security mechanism for large-scale distributed sensor networks", in *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS'03)*, Washington DC, Oct. 2003, pp. 62–72.

[70] A. Chadha, Y. Liu and S.K. Das, "Group key distribution via local collaboration in wireless sensor networks", in *Proceedings of the 2nd Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON'05)*, Santa Clara, CA, Sept. 2005, pp. 46–54.

[71] J. Huang, J. Buchingham, and R. Han, "A level key infrastructure for secure and efficient group communication in wireless sensor network", in *Proceedings of the 1st IEEE International Conference on Security and Privacy for Emerging Areas in Communications Network (SecureComm'05)*, Athens, Greece, Sept. 2005, pp. 249–260.

13

SENSOR NETWORK STANDARDS

Stefano Chessa

University of Pisa, Italy

13.1 INTRODUCTION

The standardization of wireless sensor networks proceeds along two main directives: the IEEE 802.15.4 standard [1] and ZigBee [2]. These two standards specify different subsets of layers: IEEE 802.15.4 defines the physical and medium access control (MAC) layers, and ZigBee defines the network and application layers, as shown in Fig. 13.1. The two protocol stacks can be combined to support low data rate and long-lasting applications on battery-powered wireless devices. Application fields of these standards include sensors, interactive toys, smart badges, remote controls, and home automation.

The first release of IEEE 802.15.4 was delivered in 2003 and it is freely distributed in Ref. [3]. This standard was revised in 2006, but the new release is not yet freely distributed. The ZigBee protocol stack was proposed at the end of 2004 by the ZigBee alliance, an association of companies working together to develop standards (and products) for reliable, cost-effective, low-power wireless networking. The first release of ZigBee has been revised at the end of 2006 (both releases can be freely downloaded from [4]). The 2006 version introduces extensions relating the standardization of application profiles and some minor improvements to the network and application layers. This chapter will focus on the main

Wireless Sensor Networks: A Networking Perspective, Edited by Jun Zheng and Abbas Jamalipour

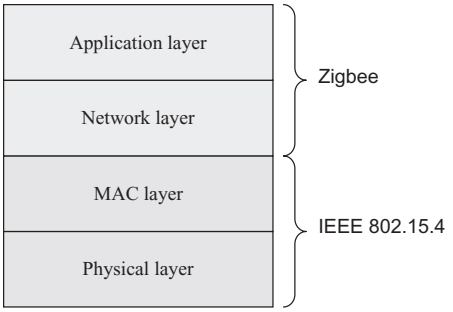


Fig. 13.1 The protocol stack of the IEEE 802.15.4 and ZigBee standards.

functionalities shared by the two releases of ZigBee. A survey of the IEEE 802.15.4 and ZigBee standards against the research state of the art can be found in Ref. [5].

This chapter presents the main features of the two standards. It is organized in two parts: the first part, given in Section 13.2, introduces the physical and MAC layers of IEEE 802.15.4 and the second part presents the network and application layers of ZigBee, which is given in Section 13.3.

13.2 IEEE 802.15.4 STANDARD

The IEEE 802.15.4 standard [1] specifies the physical and MAC layers for low-rate wireless personal area networks (PAN). Its protocol stack is simple and flexible, and does not require any infrastructure, which is suitable for short-range communications (typically within a range of 100 m). For these reasons, it features ease of installation, low cost, and a reasonable battery life of the devices.

The physical layer of the IEEE 802.15.4 standard has been designed to coexist with other IEEE standards for wireless networks, for example, IEEE 802.11 and IEEE 802.15.1 (Bluetooth). It features activation and deactivation of the radio transceiver and transmission of packets on the physical medium. It operates in one of the following three license-free bands:

- 868–868.6 MHz (e.g., Europe) with a data rate of 20 kbps.
- 902–928 MHz (e.g., North America) with a data rate of 40 kbps.
- 2400–2483.5 MHz (worldwide) with a data rate of 250 kbps.

The MAC layer provides data and management services to the upper layers. The data service enables transmission and reception of MAC packets across the physical layer. The management services include synchronization of communications, management of guaranteed time slots, and association and disassociation of devices to the network. In addition, the MAC layer implements basic security mechanisms.

TABLE 13.1 Acronyms Used in the IEEE 802.15.4 MAC Layer

Acronym	Definition
ACL	Access Control List
CAP	Contention Access Period
CFP	Contention Free Period
CSMA-CA	Carrier Sense Multiple Access with Collision Avoidance
FFD	Full-Function Devices
GTS	Guaranteed Time Slots
MAC	Medium Access Control layer
PAN	Personal Area Network
RFD	Reduced Function Devices

Since the description of the physical layer for wireless sensor networks is out of the scope of this book, we limit this chapter to the MAC layer features of this standard. The acronyms used in this section are listed in Table 13.1.

13.2.1 Overview of the MAC Layer

The MAC layer defines two types of nodes: reduced function devices (RFDs) and full function devices (FFDs). The RFDs are meant to implement end devices with reduced processing, memory, and communication capabilities, which implement a subset of the MAC layer functions. In particular, the RFDs can only be associated with an existing network and they depend on FFDs for communication. One RFD can be associated to only one FFD at a time. Examples of RFDs include simple sensors or actuators like light switches, lamps, and similar devices.

The FFDs implement the full MAC layer and they can act either as a PAN coordinator or as a generic coordinator of a set of RFDs. The PAN coordinator sets up and manages the network. In particular, it selects the PAN identifier and manages association or disassociation of devices. In the association phase, the PAN coordinator assigns to the new device a 16-bit address. This address can be used alternatively to the standard 64-bit extended IEEE address, which is statically assigned to each device.

The FFDs cooperate to implement the network topology. The actual network formation is performed at the network layer, but the MAC layer provides support to two types of network topologies: star and peer-to-peer.

In the star topology, one FFD is the PAN coordinator and is located in the star center. All the other FFDs and RFDs behave as generic devices and can only communicate with the coordinator, which synchronizes all the communications in the network. Different stars operating in the same area have different PAN identifiers and operate independently of each other. An example of the star topology is shown in Fig. 13.2a.

In peer-to-peer topology, each FFD is capable of communicating with any other device within its radio range. One FFD (normally the FFD that initiated

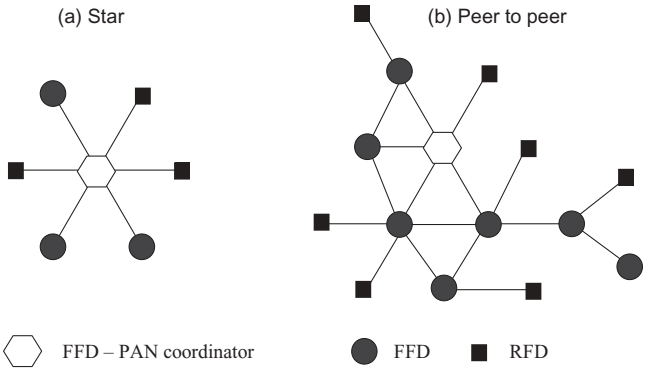


Fig. 13.2 Network topologies supported by the IEEE 802.15.4 MAC layer.

the network) acts as a PAN coordinator, and the other FFDs act as routers or end devices to form a multihop network, as shown in Fig. 13.2b. The RFDs act as end devices and each RFD is connected only with one FFD.

13.2.2 Channel Access

The MAC protocol has two types of channel access: with a superframe structure and without a superframe structure. The channel access with a superframe structure is used in star topologies (it can also be used in peer-to-peer topologies organized in trees) and provides synchronization between nodes to enable energy savings of the devices. The channel access without a superframe structure is more general and can be used to support communications in arbitrary peer-to-peer topologies.

13.2.2.1 Communications with a Superframe Structure. A superframe is composed of an active portion and an inactive portion. All the communications happen during the active portion. Hence, the PAN coordinator (and the connected devices) may enter a low power (sleep) mode during the inactive portion. The active portion comprises up to 16 equally sized timeslots. The first timeslot is the beacon frame and is sent by the PAN coordinator to begin the superframe. The beacon frames are used to synchronize the attached devices, to identify the PAN, and to describe the structure of the superframes. The actual communications between the end devices and the coordinator take place in the remaining timeslots. The timeslots in the active portion are divided into a contention access period (CAP) and a (optional) contention free period (CFP).

In the CAP period, the devices compete for channel access using a standard slotted CSMA-CA protocol (carrier sense multiple access with collision avoidance). This means that a device wishing to transmit data frames first waits for the beacon frame and then randomly selects a timeslot for its transmission. If the selected timeslot is busy because another communication is already ongoing (this

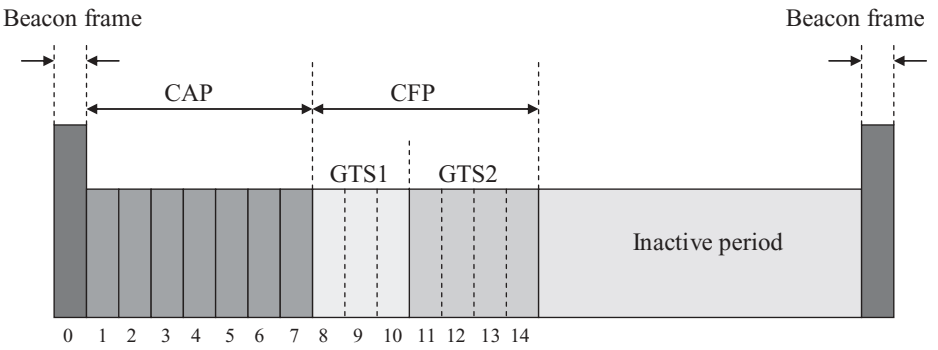


Fig. 13.3 The superframe structure.

is detected using carrier sense), the device randomly selects another timeslot. If the channel is idle, the device can begin transmitting on the next slot.

The CFP period is optional and is used for low-latency applications or applications requiring specific data bandwidth. For this purpose, the PAN coordinator may assign portions of the active superframe (called guaranteed timeslots or GTS) to specific applications. The GTSs form the CFP, which always begins at the end of the active superframe starting at a slot boundary immediately following the CAP. Each GTS may comprise more than one timeslot and is assigned to an individual application that accesses it without contention.

In any case, the PAN coordinator always leaves a sufficient number of frames for the CAP period for the other devices and to manage the association/disassociation protocols. Note also that all contention-based transactions are completed before the beginning of the CFP, and each device transmitting in a GTS completes its transmission within its GTS. The superframe structure is shown in Fig. 13.3.

13.2.2.2 Communications without a Superframe Structure. The PAN coordinator may optionally avoid the use of a superframe structure (thus the PAN is called *nonbeacon enabled*). In this case, the PAN coordinator never sends beacons and communication happens on the basis of the unslotted CSMA-CA protocol. The coordinator is always on and ready to receive data from an end device while data transfer in the opposite direction is poll based: the end device periodically wakes up and polls the coordinator for pending messages. The coordinator responds to this request by sending the pending messages or by signaling that no messages are available.

13.2.3 Data-Transfer Models

The standard supports three models of data transfers: end device to the coordinator, coordinator to an end device, and peer to peer. The star topology uses only the first two models because the data transfers can happen only between the PAN

coordinator and the other devices. In the peer-to-peer topology, all three models are possible because data can be exchanged between any pair of devices. The actual implementation of the three data-transfer models depends on whether the network supports the transmission of beacons.

13.2.3.1 Data Transfers in Beacon-Enabled Networks.

Data Transfer from an End Device to a Coordinator. The end device first waits for a network beacon to synchronize with the superframe. When the beacon is received, if it owns a GTS, it will directly use the GTS. Otherwise, it will transmit the data frame to the coordinator using the slotted CSMA-CA protocol in one of the frames in the CAP period. The coordinator may optionally acknowledge the successful reception of the data by transmitting an acknowledgment frame in a successive timeslot. This protocol is shown in Fig. 13.4a.

Data Transfer from a Coordinator to an End Device. The coordinator stores the message (a data frame) and indicates in the network beacon that the data message is pending. The end device usually sleeps most of the time and periodically listens to the network beacon to check for pending messages. When it notices that a message is pending, it will explicitly request the message to the coordinator using the slotted CSMA-CA in the CAP period. In turn, the coordinator will send the pending message in the CAP period using the slotted CSMA-CA. The device will thus acknowledge the reception of the data by transmitting an acknowledgment frame in a successive timeslot so that the coordinator can remove the pending message from its list. This protocol is shown in Fig. 13.4b.

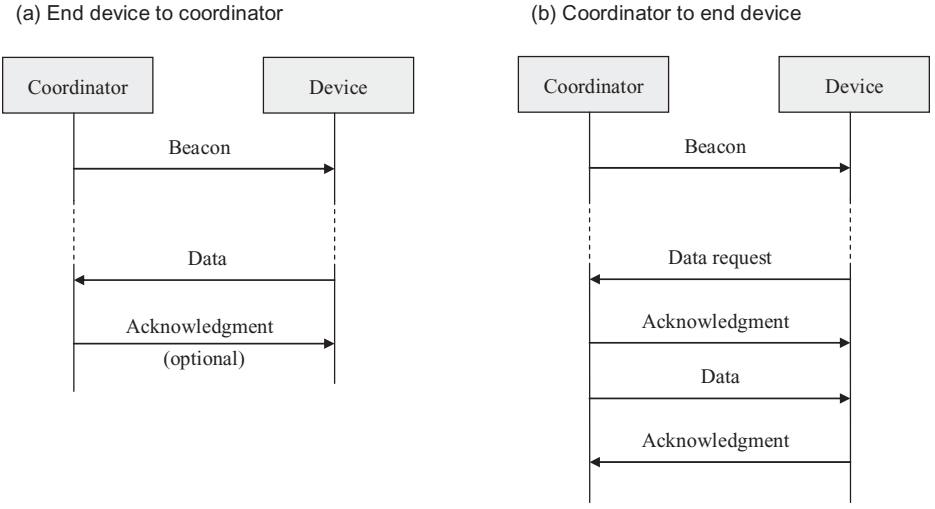


Fig. 13.4 Data-transfer modes in beacon-enabled networks.

Peer-to-Peer Data Transfer. If the sender or the receiver is an end device, one of the above data-transfer models is used. Otherwise, both the source and the destination are coordinators and they send their own beacons. In this case, the sender must first synchronize with the beacon of the destination and act as an end device. The measures to be taken in order to synchronize coordinators are beyond the scope of the IEEE 802.15.4 standard and are thus left to the upper layers.

13.2.3.2 Data Transfers in Nonbeacon-Enabled Networks.

Data Transfer from an End Device to a Coordinator. The end device directly transmits its data frame to the coordinator using the unslotted CSMA-CA protocol. The coordinator acknowledges the successful reception of the data by transmitting an optional acknowledgment frame. This protocol is shown in Fig. 13.5a.

Data Transfer from a Coordinator to an End Device. The coordinator stores the message (a data frame) and waits for a device to request for the data. A device can request to the coordinator the pending messages by transmitting a request using the unslotted CSMA-CA protocol (this request happens at an application-defined rate). The coordinator acknowledges the successful reception of the request by transmitting an acknowledgment frame. If there are pending messages, the coordinator transmits the messages to the device using the unslotted CSMA-CA protocol. Otherwise, if no messages are pending, the coordinator transmits a message with a zero-length payload (which indicates that no messages are pending). The device acknowledges the successful reception of the messages by transmitting an acknowledgment frame so that the coordinator can discard the pending messages. This protocol is shown in Fig. 13.5b.

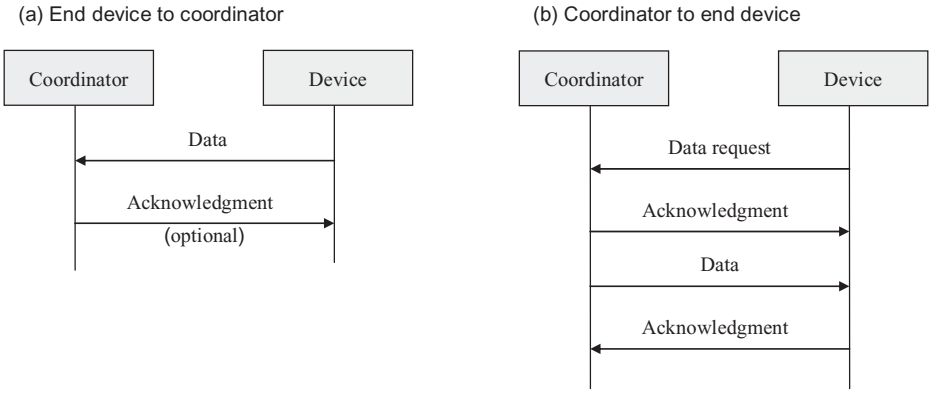


Fig. 13.5 Data-transfer modes in nonbeacon-enabled networks.

Peer-to-Peer Data Transfer. In peer-to-peer PANs, each device can communicate with each other device within its radio range. In order to perform this effectively, the devices wishing to communicate need either (1) to keep the radio constantly active in order to be ready to receive incoming messages or (2) to synchronize with each other. In the former case, the device can directly transmit the data using unslotted CSMA-CA while in the second case the device has to wait until the destination device is ready for receiving the data. Note, however, that the device synchronization is beyond the scope of the IEEE 802.15.4 standard and is left to the upper layers.

13.2.4 MAC Layer Services

The MAC layer provides data and management services to the upper layer (normally the ZigBee network layer). Each service is specified by a set of primitives, which can be classified into four generic types, as illustrated in Fig. 13.6, and can use all or part of the four primitives depending on its needs.

- Request: It is invoked by the upper layer to request for a specific service.
- Indication: It is generated by the MAC layer and is directed to the upper layer to notify the occurrence of an event related to a specific service.
- Response: It is invoked by the upper layer to complete a procedure previously initiated by an indication primitive.
- Confirm: It is generated by the MAC layer and is directed to the upper layer to convey the results of one or more service requests previously issued.

13.2.4.1 Data Service. The data service comprises one main service that exploits only the request, confirm and indication primitives. The DATA.request primitive is invoked by the upper layer to send a message to another device. The result of a transmission requested with a previous DATA.request primitive is reported by the MAC layer to the upper layer by the DATA.confirm primitive, which returns the status of transmission (either success or an error code). The

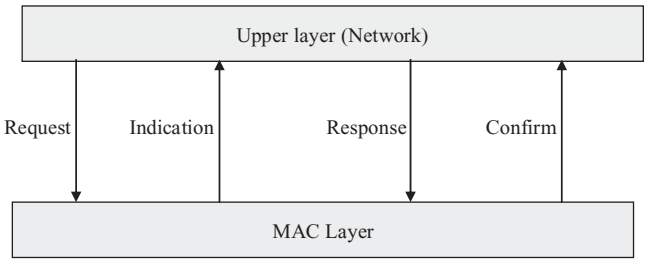


Fig. 13.6 The four types of primitives used to implement the MAC services.

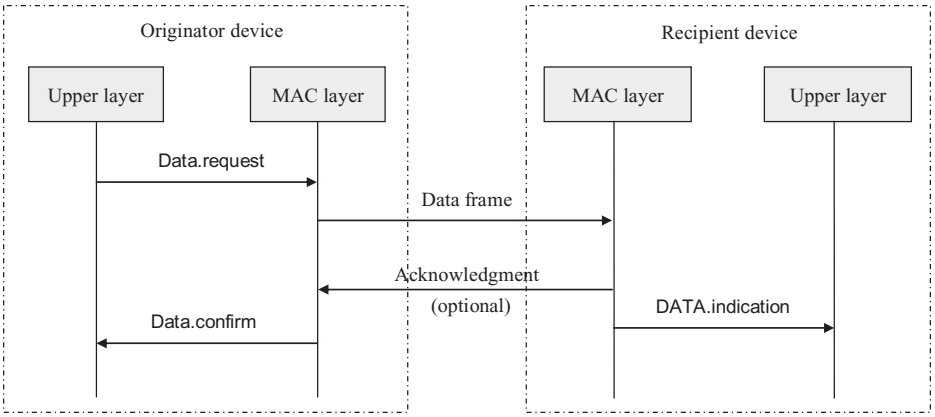


Fig. 13.7 Implementation of the DATA service.

DATA.indication primitive corresponds to a "receive" primitive: it is generated by the MAC layer on receipt of a message from the physical layer to pass the received message to the upper layer.

Figure 13.7 illustrates the sequence of messages and primitives occurring during a data exchange between two nodes.

13.2.4.2 Management Service. The management services of the MAC layer include functionalities for PAN initialization, device association/disassociation, detection of existing PANs, and other services exploiting some of the features of the MAC layer. The main management services are summarized in Table 13.2. In this table, symbol X in a cell corresponding to service S and primitive P denotes that S uses the primitive P, while symbol O means that primitive P is optional for the RFDs.

As an example, we describe the protocol and functionalities of the ASSOCIATE service here. This service is invoked by a device wishing to be associated with a PAN that it has already identified by preliminarily invoking the SCAN service. The ASSOCIATE.request primitive takes as parameters (among others) the PAN identifier, the coordinator address, and the 64-bit extended IEEE address of the device. The primitive sends an association request message to a coordinator (either the PAN coordinator or a router). Since the association procedure is meant for beacon-enabled networks, the association request message is sent during the CAP using the slotted CSMA-CA protocol.

The coordinator acknowledges the reception of the association messages immediately. However, this acknowledgment does not mean that the request has been accepted. On the coordinator side, the association request message is passed to the upper layers of the coordinator protocol stack (using the ASSOCIATE.indication primitive), where the decision on the association request is actually made. If the request is accepted, the coordinator selects a short 16-bit address that the device may use later in place of the 64-bit extended IEEE address. The

TABLE 13.2 Main Management Services of the MAC Layer

Name	Request	Indication	Response	Confirm	Functionality
ASSOCIATE	X	O	O	X	Request for association of a new device to an existing PAN.
DISASSOCIATE	X	X		X	Leave a PAN.
BEACON-NOTIFY		X			Provide to the upper layer the received beacon.
GET	X			X	Read the parameters of the MAC.
GTS	O	O		O	Request of GTS to the coordinator.
SCAN	X			X	Look for for active PANs.
COMM-STATUS		X			Notify the upper layer about the status of a transaction begun with a response primitive.
SET	X			X	Set the parameters of the MAC layer.
START	O			O	Start a PAN and begins sending beacons. Can also be used for device discovery.
POLL	X			X	Request for pending messages to the coordinator.

upper layers of the coordinator thus invoke the ASSOCIATE.response primitive of the coordinator MAC layer. This primitive takes as parameters the 64-bit address of the device, the new 16-bit short address, and the status of the request (which can be association successful or an error code). The primitive thus generates an association response command message, which is sent to the device requesting association using indirect transmission, that is, the command message is added to the list of pending messages stored in the coordinator. The MAC layer of the device automatically issues a data request message to the coordinator after a predefined period following the acknowledgment of the association request command. Note that there are two ways for a device to request a pending data message to the coordinator: by the POLL service or automatically after a

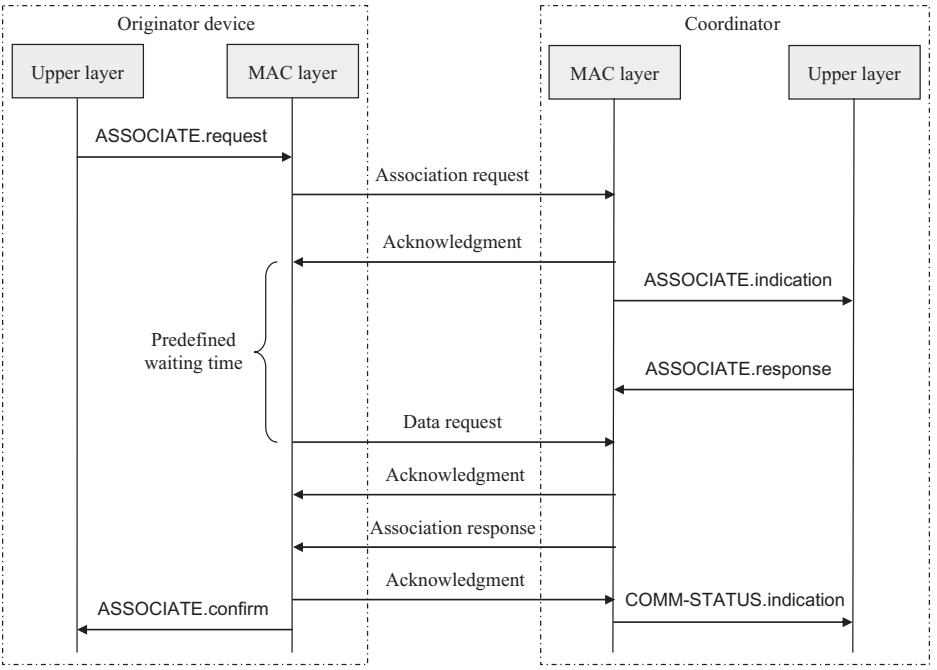


Fig. 13.8 Implementation of the ASSOCIATE service.

predefined period following the acknowledgment of a previous request command (as in the ASSOCIATE service). The coordinator then sends the association response command message to the device.

Upon receiving the command message, the MAC layer of the device issues an ASSOCIATE.confirm primitive, while the MAC layer of the coordinator issues a COMM-STATUS.Indication primitive to inform the upper layer that the association protocol is concluded either with success or with an error code. The association protocol is shown in Fig. 13.8.

13.2.5 Security

The IEEE 802.15.4 MAC layer provides a basic support for security, and leaves advanced security features (e.g., key management and device authentication) to the upper layers. All the security services are based on symmetric keys and use keys provided by the higher layers. The MAC layer security services also assume that the keys are generated, transmitted, and stored by the upper layers in a secure manner. Note also that the security features of the MAC layer are optional and the applications can decide when and which functionality they use.

The security services provided by the MAC layer include access control, data encryption, frame integrity, and sequential freshness, which are described as follows:

- *Access Control.* Access control allows a device to keep a list of devices (called the access control list, or ACL) with which it is enabled to communicate. If this service is activated, each device in the PAN maintains its own ACL and discards all the incoming packets received from the devices not included in the ACL.
- *Data Encryption.* Data encryption uses symmetric cryptography to protect data from being read by parties without the cryptographic key. The key can be shared by a group of devices (typically stored as the default key) or it can be shared between two peers (stored in an individual ACL entry). Data encryption may be provided on data, command, and beacon payloads.
- *Frame Integrity.* Frame integrity uses an integrity code to protect data from being modified by parties without the cryptographic key and to assure that the data comes from a device with the cryptographic key. As in the data encryption service, the key can be shared by a group or by pairs of devices. Integrity may be provided on data, beacon, and command frames.
- *Sequential Freshness.* Sequential freshness orders the sequence of input frames to ensure that an input frame is more recent than the last received frame.

13.3 ZIGBEE STANDARD

ZigBee builds upon the IEEE 802.15.4 standard. It specifies the network and the application layers. The network layer provides support to star, tree, and peer-to-peer multihop network topologies, and the application layer provides a framework for distributed application development and communication. The application layer comprises the application framework, the ZigBee device objects (ZDO), and the application support sublayer (APS). The application framework contains up to 240 application objects (APOs), that is, user-defined application modules that implement a ZigBee application. The ZDO provides services that allow the APOs to organize themselves into a distributed application. The APS provides data and management services to the APOs and ZDO. An overview of the ZigBee protocol stack is shown in Fig. 13.9 and Table 13.3 summarizes the acronyms used in this section.

13.3.1 Network Layer

The network layer defines three types of devices: the end device that corresponds to a RFD or a FFD acting as a simple device, the router that is a FFD with routing capabilities, and the network coordinator that is a FFD managing the whole network. Besides the star topology (that naturally maps to the star topology of IEEE 802.15.4), the network layer also supports tree and mesh topologies (the ZigBee network topologies are shown in Fig. 13.10). The network layer provides

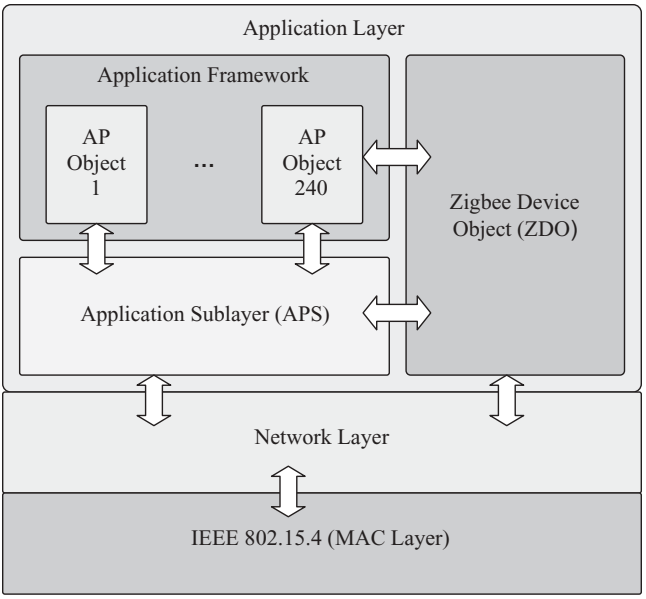


Fig. 13.9 ZigBee functional layer architecture and protocol stack.

TABLE 13.3 Acronyms Used in the ZigBee Network and Application Layers

Acronym	Definition
APO	Application Objects
APS	Application Sublayer
RDT	Route Discovery Table of the network layer
RREQ	Route Request message (network layer)
RREP	Route Reply message (network layer)
RT	Routing Table of the network layer
ZDO	ZigBee Device Object

services for the initialization of the network, device addressing, route management, routing, and management of connections and disconnections of devices. Table 13.4 lists the set of services of the network layer. Unlike the IEEE 802.15.4 MAC layer, the network layer services are defined only in terms of request, indication, and confirm primitives. In the next few sections, we describe in more details the main network protocols that implement the services for network creation, join, and routing.

13.3.1.1 Network Formation. The procedure to establish a new network is initiated by the NETWORK-FORMATION.request primitive. This primitive can

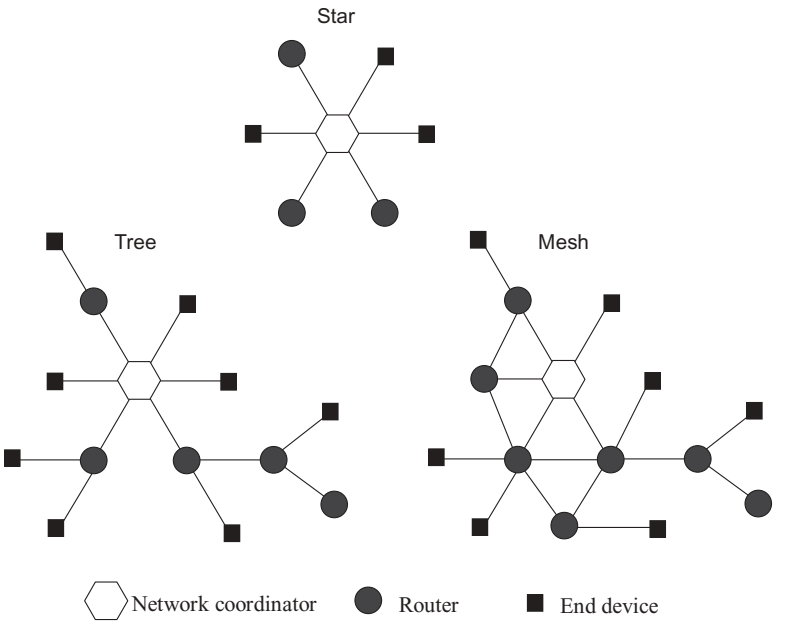


Fig. 13.10 ZigBee network topologies.

be invoked only by FFD devices that can behave as a coordinator and have not joined another network. The primitive first uses the MAC layer services to look for a channel which does not conflict with other existing networks.

If a suitable channel is found, the primitive selects a PAN identifier that is not already in use by other PANs, and assigns to the device (which is also the coordinator of the new PAN) the 16-bit network address 0x0000. The primitive then invokes the SET.request primitive of the MAC layer to set the PAN identifier and the device address; then it invokes the START.request primitive of the MAC layer to start the PAN. In response to this primitive, the MAC layer starts generating the beacons.

13.3.1.2 Joining a Network. The join procedure can be requested by a device wishing to join an existing network (*join through association*), by a router, or by the coordinator to force a device to join its PAN (*direct join*). The join through association procedure is described below.

When the application layer running on a device *D* wishes to join an existing network, it first invokes the NETWORK-DISCOVERY service to look for existing PANs. This procedure exploits the MAC layer SCAN service to learn about neighboring routers that announce their networks. Once this procedure is completed, the application layer is notified of the existing networks. In turn, the application layer selects one network (several ZigBee networks may spatially overlap, using different channels) and invokes the JOIN.request primitive with

TABLE 13.4 Services Provided by the Network Layer

Name	Request	Indication	Confirm	Description
DATA	X	X	X	Data transmission service.
NETWORK-DISCOVERY	X		X	Look for existing PANs.
NETWORK-FORMATION	X		X	Create a new PAN (invoked by a router or by a coordinator).
PERMIT-JOINING	X		X	Allow associations of new devices to a PAN (invoked by a router or by a coordinator).
START-ROUTER	X		X	(Re-)initialize the superframe of the PAN coordinator or a router.
JOIN	X	X	X	Request to join a PAN (invoked by any device).
DIRECT-JOIN	X		X	Request to other devices to join a PAN (used by a router or by the coordinator).
LEAVE	X	X	X	Leave a PAN.
RESET	X		X	Reset the network layer.
SYNC	X	X	X	Allow the application layer to synchronize with the coordinator or a router and/or to extract pending data from it.
GET	X		X	Read the parameters of the network layer.
SET	X		X	Set the parameters of the network layer.

two parameters: the PAN identifier of the selected network and a flag indicating whether it joins as a router or as an end device.

The JOIN.request primitive in the network layer selects a "parent" node *P* (in the desired network) from his neighborhood. The parent should be a device in the PAN allowing joins. For example, in the case of a star topology, the parent is the coordinator and the devices join as an end device. The network layer then performs the MAC layer association procedure to node *P*. Upon receiving an indication of the association request from the MAC layer, the network layer of node *P* assigns node *D* a 16-bit short address and lets the MAC layer successfully reply to the association request. Node *D* will use the short address for any

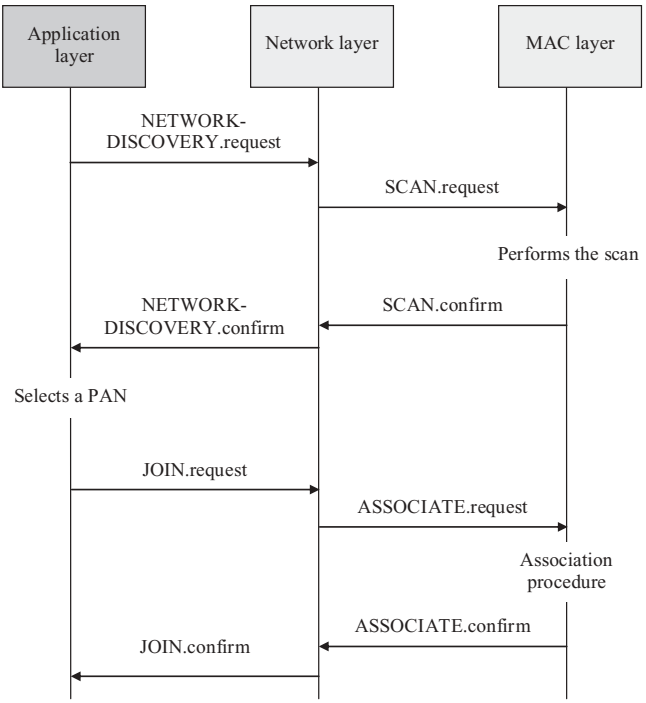


Fig. 13.11 The JOIN protocol at the child's side.

further network communication. Fig. 13.11 shows the join procedure at the device side.

The parent–child relationships established as a result of joins shape the whole network in the form of a tree with the ZigBee coordinator as the root, the ZigBee routers as internal nodes, and ZigBee end devices as leaves. This tree structure is also the basis of the distributed algorithm for network address assignment. The ZigBee coordinator fixes the maximum number of routers (R_m) and end devices (D_m) that each router may have as children and also fixes the maximum depth of the tree (L_m). On the basis of its depth in the tree, a newly joined router is assigned a range of consecutive addresses (16-bit integers). This range is such that the router will have enough addresses for all of its children and descendants, and it is computed based on R_m, D_m, and L_m. Figure 13.12 shows an example of address assignment in a network with $R_m = 2$, $D_m = 2$ and $L_m = 3$, where all addresses have been assigned to routers (white nodes) and end devices (gray nodes). The address of a node is shown inside the circle representing the node, while the assigned address ranges are shown in brackets next to each router.

Although the addresses are always assigned based on a tree topology, the network layer can be configured by the application layer to implement a mesh or a tree topology. If the configuration is the mesh, all the nodes (coordinator, routers, and end devices) can communicate without a superframe structure.

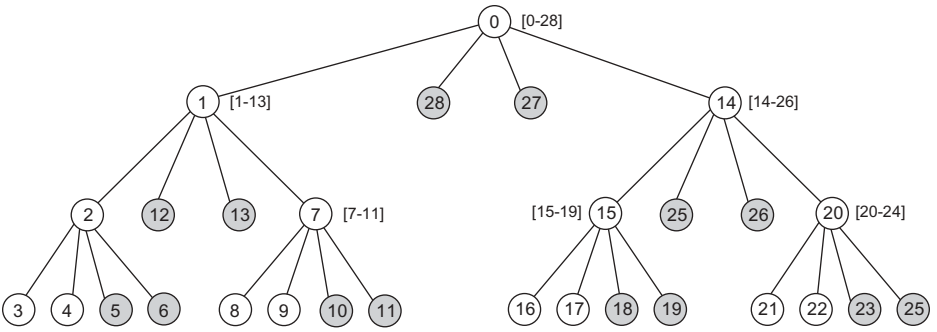


Fig. 13.12 A tree topology and address allocations for $R_m = 2$, $D_m = 2$ and $L_m = 3$.

TABLE 13.5 The Fields of an Entry of the Routing Table (RT) in a ZigBee Router

Field Name	Size	Description
Destination Address	16-bits	Network address of the destination.
Next-hop Address	16-bits	Network address of the next hop towards destination.
Entry Status	3-bits	Route status: Active, Discovery_underway, Discovery_failed, or Inactive.

Otherwise, if the topology is a tree, the network can communicate with a superframe structure. In this case, all newly joined routers invoke the primitive START-ROUTER.request to begin transmitting their beacons. To avoid overlaps of the activity periods, the routers should have (relatively) long inactive periods and have neighboring routers start their superframe in the inactive period of the other routers to avoid overlapping. Communication from a child to a parent happens in the CAP of the parent while communication from a parent to a child is indirect. In any case, a node has to synchronize with the parent's beacon to exchange data with it, while it drives communication with its children according to its superframe.

13.3.1.3 Routing. On receipt of a data frame (a message), the network layer routes the message depending on the capability of the device. If the sender is an end device, it forwards the message to its parent, which has routing capability. Otherwise, if the sender is a router (or the coordinator), it maintains a routing table RT (the fields of an RT entry are shown in Table 13.5) and routes the message according to the following procedure.

If the destination is a child, the message is forwarded directly using the DATA service of the MAC layer. Otherwise, the actual routing protocol depends on the topology used in the network (tree or mesh).

If the topology is a mesh, the network layer looks for an entry corresponding to the destination in RT. If the entry corresponding to the destination in RT is not

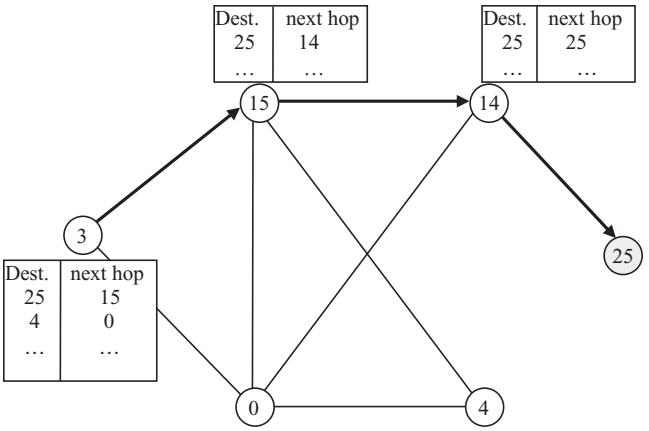


Fig. 13.13 Routing of a message from node 3 to 25 in a mesh.

active or such an entry does not exist, the network layer initiates a route discovery procedure (see Section 13.3.1.4) and the message is buffered until the discovery is complete. Otherwise, if the table entry for the destination is active, the table contains the address of the next hop toward the destination, and the message is forwarded to the destination through the next hop. Figure 13.13 shows the route followed by a message in a mesh topology.

If the topology is a tree, the network routes the packets along the tree. In the tree topology, routers maintain the addresses of their children and their parent. Given the way the addresses are assigned, a router that needs to forward a message can easily determine whether the destination is one of its end-device children or if it belongs to the subtree rooted in one of its children. If the destination is one of its end-device children, it routes the packet to the appropriate child; otherwise, it routes the packet to its parent. Figure 13.14 shows the route followed by a message in a tree topology.

Note that tree and mesh topologies may live together, that is, the routers can maintain both information for mesh and tree routing. In this case, a router forwarding the message can switch from one routing algorithm to the other. For example, if a route to a destination in the mesh routing is not yet available, the message can be forwarded through the tree.

Also note that while mesh routing is more complex to handle and does not allow beaconing (it works in the networks without a superframe structure), tree routing allows the routers to operate in beacon-enabled networks.

13.3.1.4 Route Discovery. Route discovery is a protocol initiated by the network layer of a source device *S* when it needs to send a message to destination device *D*, but its RT does not contain information suitable to route the message. A route discovery table (RDT) (the fields of an RDT entry are shown in Table 13.6) is maintained by routers and the coordinator to implement route discovery.

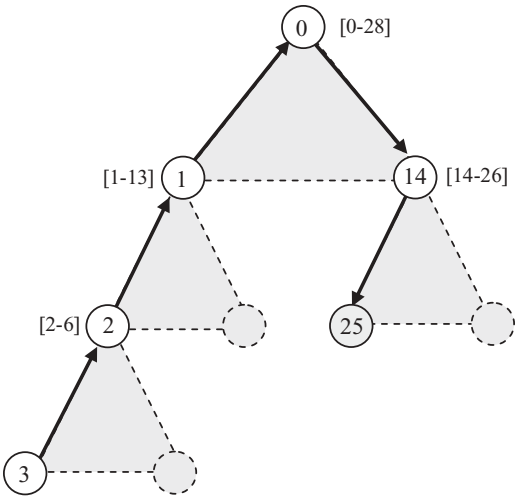


Fig. 13.14 Routing of a message from node 3 to node 25 in a tree.

TABLE 13.6 The Fields of an Entry of the Route Discovery Table

Field Name	Size	Description
RREQ ID	8-bits	Unique ID (sequence number) given to every RREQ message being broadcasted.
Source Address	16-bits	Network address of the initiator of the route request.
Sender Address	16-bits	Network address of the device that sent the most recent lowest cost RREQ.
Forward Cost	8-bits	The accumulated path cost from the RREQ originator to the current device.
Residual Cost	8-bits	The accumulated path cost from the RREQ originator to the current device.
Expiration time	16-bits	A timer indicating the number of milliseconds until this entry expires.

To initiate the route discovery, *S* broadcasts a route request (RREQ) message that contains the RREQ ID, the destination address, and the path cost that is initially set to be 0. The RREQ ID is an integer that is incremented every time the device *S* sends a new RREQ message. Thus the RREQ ID and the address of *S* can be used as a unique reference for a route discovery process.

As the RREQ propagates in the network, an intermediate device *I* receiving the RREQ performs the following actions:

1. It updates the path cost field by adding the cost of the last traversed link. The cost of a link can be a constant or a function of the link quality estimation provided by the IEEE 802.15.4 interface.

2. It searches within its RDT for an entry corresponding to the RREQ. If no match is found, a new RDT entry is created for the discovery process and a route request timer is started (upon timer expiration the RDT entry will be removed). Conversely, if an entry is found in the RDT, the node compares the path cost for the RREQ message and the corresponding value in the RDT entry. If the former is higher, the node drops the RREQ message. Otherwise, it updates the RDT entry.
3. If *I* is not the route discovery destination, it allocates an RT entry for the destination with status DISCOVERY_UNDERWAY and rebroadcasts the RREQ after updating its path cost field.
4. If the node is the final destination, it replies to the originator with a route reply (RREP) message that travels back along the path.

The RREP message is sent to the route discovery originator and carries a residual cost value field that each node increments when it forwards the message.

Upon receipt of a route reply (RREP) message, a node performs the following actions:

1. If the node is the RREQ originator and this is the first RREP it has received, it sets the corresponding RT entry to ACTIVE and records the residual cost and the next hop in the RDT entry.
2. Otherwise (the node is not the RREQ originator):
   a. If the residual cost of the RREP is higher than the residual cost of the corresponding RDT entry, the node discards the RREP message.
   b. Otherwise, it updates the RDT entry (residual cost) and the RT entry (next hop).
   c. It forwards the RREP toward the originator. Note that intermediate nodes never change the RT entry status to ACTIVE as a result of receiving an RREP message. They will only change the entry status upon receipt of a data message for the given destination.

13.3.2 Application Layer

The application layer defines the application framework under which the programmers develop applications in terms of APOs. The APOs exploit the services offered by the ZDO and the APS sublayer, which include data, binding, and discovery services.

13.3.2.1 Application Framework. The application framework contains up to 240 APOs, each of which is interfaced with an application endpoint numbered from 1 to 240. The endpoint 0 is reserved for the ZDO. Each APO in the network is uniquely identified by combining its endpoint address and the network

address of the hosting device. The APOs define the behavior of Zigbee applications. They can have complex states and communicate exploiting the data services of the APS.

Figure 13.15 shows an example of a simple ZigBee application. Device *A* contains two APOs (attached to endpoints 10 and 25, respectively), each of which controls a switch. Device *B* contains three APOs (attached to endpoints 5, 6, and 8, respectively), each controlling a lamp. One switch (10*A*) controls two lamps (5*B* and 6*B*) and the other switch (25*A*) controls lamp 8*B*. In this simple example, the APOs 5*B*, 6*B*, and 8*B* could have a single attribute containing the status of the lamp (on/off), which can be set remotely from the APOs 10*A* and 25*A*.

For the purpose of specification of services and applications, the ZigBee standard introduces the concept of clusters and profiles. A cluster is the specification in a standard format of the messages managed by an APO. Clusters are numbered within a given application profile with an 8-bit identifier.

An application profile is the specification in a standard format of the behavior of an application possibly operating on several ZigBee devices. An application profile describes a set of devices and clusters. The application profiles are assigned with a unique identification number, which is assigned by the ZigBee alliance.

13.3.2.2 Binding and Discovery Services. Binding and discovery of services and devices are the main services provided to the APOs, which are described as follows:

Device Discovery. Device discovery allows a device to obtain the (network or MAC) address of other devices in the network. A router (or the coordinator) responds to a device discovery query by returning its address and the addresses of all its associated end devices.

Service Discovery. Service discovery exploits cluster descriptors and cluster identifiers to determine the services offered by a given APO. It can be accomplished by issuing a query for each endpoint on a given device or

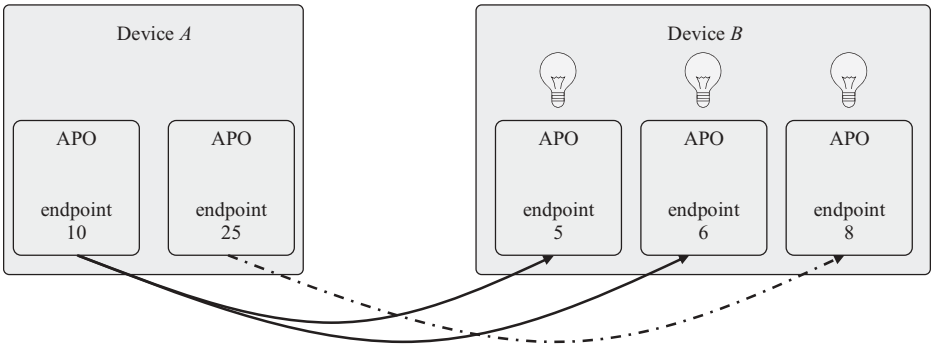


Fig. 13.15 A simple ZigBee application.

by using a match service feature. In the example of Fig. 13.15, device and service discovery can be used by device *A* to determine the address of *B* and the services offered by its APOs. Once device *A* has discovered the address and the services offered by *B*, it can issue request messages to *B* according to the cluster descriptions of the APOs of *B*.

Binding. A message is normally routed from the source to the destination APO based on the destination address pair <destination endpoint, destination network address>. However, this kind of addressing (called *direct* addressing) may be unsuitable for extremely simple devices that may be unable to store information about the address of the destination device. For this reason, ZigBee also offers the *indirect* addressing that exploits binding tables to translate the source address (in terms of network and endpoint address) and the cluster identifier of the message into the pair <destination endpoint, destination network address>. The binding table is stored in the ZigBee coordinator and/or in the routers and is updated on an explicit request of the ZDO in the routers or in the coordinator. A binding table for the example of Fig. 13.15 is shown in Table 13.7.

13.3.2.3 Application Support Sublayer. The APS offers the binding service to the ZDO and the data service to both the APOs and the ZDO.

The data service enables the exchange of messages between two or more devices within the network using either direct or indirect addressing. The data service is defined in terms of the *request, confirm*, and *indication* primitives. The primitive *request* implements a send and the *indication* implements a receive. The primitive *confirm* returns to the sender the status of the transmission (either success or an error code).

The binding services comprises the BIND and UNBIND services, both defined in terms of the *request* and *confirm* primitives. These services can be invoked only by the ZDO of the coordinator or of a router. The BIND.request primitive takes as input parameters the tuple <source address, source endpoint, cluster identifier, destination address, destination endpoint>, and creates in the binding table of the device in which it is invoked by an entry corresponding to the tuple in input. The UNBIND.request primitive deletes the entry corresponding to its input parameters from the binding table. The BIND.confirm and UNBIND.confirm primitives return the result of the corresponding request primitive (either success or an error code).

TABLE 13.7 The Binding Table for the ZigBee Application of Fig. 13.15

<source address, endpoint address, cluster identifier>	<destination address, endpoint number>
<*A*,10,15>	<*B*,5>,<*B*,6>
<*A*,25,15>	<*B*,8>

13.3.2.4 ZigBee Device Object. The ZDO behaves as a special application that uses the network and APS primitives to implement ZigBee end devices, ZigBee routers, and ZigBee coordinators. The ZDO is attached to the APS through endpoint 0 and is specified by a special profile, the ZigBee Device Profile, which describes the clusters that must be supported by any ZigBee device. In particular, the ZigBee device profile defines how the ZDO should implement the discovery and binding services, and how it should manage the network and the security.

Device and Service Discovery. The ZDO implements these services depending on the capability of the hosting device:

- The discovery of end devices and their services is the responsibility of the ZDO of the coordinator. The reason is that the end devices may sleep most of the time and their ZDO may be unable to respond to discovery requests. However, the ZDO of an end device should respond to discovery requests when the device is active.
- The ZDO in the coordinator and the routers should respond to discovery requests on behalf of their associated sleeping end devices.
- In any case, the ZDO of any device should offer the discovery services to the local APOs.

The device and service discovery requests can be conducted based on different input parameters. Typically, device discovery takes in input a 64-bit extended IEEE address of a device and returns its network address and/or the list of the network addresses of its associated devices. Service discovery is more complex and takes in input a network address and optionally a endpoint number, a cluster identifier, a profile identifier, or a device descriptor. The queried device returns a set of endpoints matching the query (for example, the endpoints which implement a given cluster).

Binding Management. The ZDO processes the binding requests received from a local or remote endpoint, adding or deleting entries from the APS binding table. The ZDO of the coordinator supports the binding of end devices that are requested on the basis of button presses or other manual means.

Network Management. This function implements the coordinator, a router, or an end device according to the configuration settings established either at run time by an application or during installation. If the device is a router or an end device, the network management function offers services for the selection of an existing PAN to join. If the device is a coordinator or a router, this function provides the ability to create a new PAN. Note, however, that it is also possible to deploy a network without a device predesignated as a coordinator if the first activated FFD assumes automatically the role of the coordinator.

Node Management. The ZDO serves incoming requests aimed at performing network discovery, retrieving the routing and binding tables of the device, and manages joins/disconnections of nodes to the network.

Security Management. The ZDO determines whether security is enabled or disabled and, if enabled, manages the keys used for the encryption of the messages.

13.3.3 Security in ZigBee

The security model in ZigBee ensures protection of individual devices, but not individual applications in the same device. This allows the reuse of the same keying material among the different layers on the same device, thus reducing the storage costs. The security requirements are message integrity, device authentication, message encryption, and message freshness (to avoid message duplicates). Authentication and encryption are possible either at the network level or the device level. Network-level authentication and encryption are achieved by using a common network key. This prevents outsider attacks while adding little in memory cost. Device-level authentication and encryption are achieved by using unique link keys between pairs of devices. This prevents insider and outsider attacks, but has a higher memory cost.

The ZigBee architecture includes security mechanisms at both the network and application layers. The network layer is responsible for securely transmitting outgoing frames and securely receiving incoming frames. It enforces security using symmetric encryption of outgoing messages and decryption of incoming messages using keys provided by the application layer. Observe that even if security is enabled some command messages cannot be encrypted. This is the case, for example, of the messages used to associate new devices to the network.

The application layer provides the services for the management of the security policies and the keys. For this purpose, ZigBee defines the *Trust Center* (assumed to be located in the ZigBee coordinator) that is responsible for providing the keys to the other devices in the network. The trust center generates the keys for network- and device-level authentication and encryption, and maintains a list of associated devices and keys in use. The devices in the network establish a secure communication link with the trust center using the master key, which could be either preassigned or provided to the devices using special procedures (it could be manually inserted by the user), and use the secure link to request to the trust center the keys for their needs.

13.4 SUMMARY

This chapter presents the most significant standards for wireless sensor networks. The standardization process of wireless sensor networks is, however, far to be complete, and the evolutions of the standards presented in this chapter are expected in the near future. Improvements to these standards are expected in

the energy efficiency strategies, in particular with respect to mesh networking and synchronization to allow longer inactive period of the devices.

REFERENCES

[1] Institute of Electrical and Electronics Engineers, Inc., "IEEE Std. 802.15.4-2003: Wireless medium access control (MAC) and physical layer (PHY) specifications for low rate wireless personal area networks (LR-WPANs)", New York, IEEE Press. Oct. 1, 2003.

[2] ZigBee Alliance, "ZigBee specifications", Dec. 2006.

[3] http://www.ieee802.org/15/pub/TG4.html

[4] http://www.ZigBee.org/en/index.asp

[5] P. Baronti, P. Pillai, V. Chook, S. Chessa, A. Gotta, and Y. F. Hu, "Wireless sensor networks: A survey on the state of the art and the 802.15.4 and ZigBee standards", Computer Communications, vol. 30, no. 7, May 2007, pp. 1655–1695.

14

FUTURE TRENDS IN WIRELESS SENSOR NETWORKS

Mehmet Can Vuran

University of Nebraska-Lincoln, USA

Dario Pompili

Rutgers, The State University of New Jersey, USA

Tommaso Melodia

State University of New York at Buffalo, USA

14.1 INTRODUCTION

Wireless sensor networks (WSNs) [1] have attracted tremendous attention of the research community in recent years. A vast amount of research work has been conducted to solve the practical and theoretical issues that are still open, which has resulted in a surge of civil and military applications over the last few years. As of today, most deployed WSNs measure scalar physical phenomena like temperature, pressure, humidity, or location of objects. In general, most sensor networks are designed for delay-tolerant and low-bandwidth applications. For this reason, most research on sensor networks has concentrated on this low-power and delay-tolerant network paradigm, which we refer to as terrestrial sensor networks. While the research challenges of traditional sensor networks can by no

Wireless Sensor Networks: A Networking Perspective, Edited by Jun Zheng and Abbas Jamalipour

means be addressed sooner or later, the foreseeable future will see a growing interest in alternative paradigms, where sensor networks are deployed (1) to perform specialized tasks, for example, acting on environments, thus requiring ad hoc advanced communication protocols, algorithms, and architectures; (2) in challenging environments, for example, underwater and underground, thus requiring communication protocols that are able to cope with the characteristics and impairments of the propagation medium and the environmental characteristics of such environments.

This chapter will introduce the recent evolution of the sensor network paradigm by presenting some of our recent efforts and visions along these two directions. The first category of efforts includes so-called *wireless multimedia sensor networks* (WMSNs), that is, networks of wirelessly interconnected devices that allow retrieving video and audio streams, still images, and scalar sensor data. The design challenges and main applications of WMSNs will be discussed in Section 14.2. This category also includes what is usually referred to as *wireless sensor and actor networks* (WSANs), that is, heterogeneous networks of embedded devices, where a subset of the devices (sensors) gather environmental information and another set of more powerful and less resource-constrained devices (actors) physically manipulate the environment and interact with it. The main characteristics of WSANs will be introduced in Section 14.3 and their major research challenges will also be discussed.

The second category of efforts will be introduced in Section 14.4, where we will discuss sensor network applications in challenging environments. In particular, we will describe ongoing and open research issues for *underwater acoustic sensor networks* (UW-ASNs) and *wireless underground sensor networks* (WUSNs), that is, networks of sensors deployed underground. In such challenging environments, the medium propagation characteristics require new communication protocols explicitly designed for such environments. Challenges and recent efforts in the development of such protocols will also be described and discussed.

Section 14.5 will present an overview of recent efforts aimed at improving the performance of WSNs by means of cross-layer design methodologies. In cross-layer protocols, the functionalities of different layers in the communication protocol stack are not handled separately, but considered in a joint fashion with the objective to optimize some performance metric of interest, for example, minimize the overall energy consumption of the communication protocol. We will present and compare different ongoing research efforts in cross-layer protocol design for WSNs. This chapter concludes with a brief summary in Section 14.6.

14.2 WIRELESS MULTIMEDIA SENSOR NETWORKS

The recent availability of inexpensive hardware, for example, CMOS cameras and microphones, which can ubiquitously capture multimedia content from an environment is fostering the development of WMSNs [2], which consist of sensor

devices that are interconnected by a wireless communication channel and allow retrieving video and audio streams, still images, and scalar sensor data. With rapid improvement and miniaturization in hardware, a single sensor device can be equipped with audio and visual information collection modules. In addition to the ability to retrieve multimedia data, a WMSN will also be able to store, process in real time, correlate, and fuse multimedia data originated from heterogeneous sources.

WMSNs will enable several new applications, most of which will be described in Section 14.2.1. Many of these applications require the sensor network paradigm to be rethought to deliver multimedia content with a certain level of quality of service (QoS). In contrast to this, most previous and current research in sensor networks has focused on reducing energy consumption exclusively. Hence, effective mechanisms to efficiently deliver application-level QoS and to map these requirements to network-layer metrics, for example, latency and jitter, have not been the object of investigation.

The QoS delivery of multimedia content in sensor networks is a very challenging and largely unexplored task. First, embedded sensors are highly constrained in terms of battery, memory, processing capability, and achievable data rate. Second, while the capacity of each link in wired networks is assumed to be fixed and predetermined, the attainable capacity of each wireless link in multihop wireless networks depends on the interference level perceived at a receiver. The interference level, in turn, depends on the interaction of functionalities that are handled by all network devices in a distributed manner, for example, power control, routing, and rate policies. Therefore, capacity and delay attainable on each link are location dependent, vary continuously, and may be bursty in nature, thus making QoS provisioning a challenging task. Furthermore, in multihop wireless networks, there is a strict interdependence among the functionalities of different layers in the communication protocol stack. These functionalities are inherently and strictly coupled due to the shared nature of the wireless communication channel. Hence, the functionalities aimed at QoS provisioning should not be treated separately when efficient solutions are sought. Finally, with a few exceptions, processing of multimedia content has usually been approached as a problem isolated from the network-design problem. For this reason, existing solutions that address the content delivery aspects have typically not considered the characteristics of the source content and have primarily studied cross-layer interactions among lower layers of the protocol stack. However, processing and delivery of multimedia content are not independent operations and their interaction has a major impact on the QoS levels that can be delivered. In WMSNs, the QoS required at the application level will be delivered by means of a combination of cross-layer optimization of the communication process and in-network processing of raw data streams that describe the phenomenon of interest from multiple views with different media and on multiple resolutions. Therefore, it is necessary to develop application-independent and self-organizing architectures to flexibly perform in-network processing of multimedia content.

14.2.1 Applications of Wireless Multimedia Sensor Networks

Wireless multimedia sensor networks will enable several new applications, which are described as follows:

- *Multimedia Surveillance Sensor Networks.* Wireless video sensor networks are composed of interconnected battery-powered miniature video cameras, each packaged with a low-power wireless transceiver capable of processing, sending, and receiving data. Video and audio sensors are used to enhance and complement existing surveillance systems against crime and terrorist attacks. Large scale networks of video sensors can extend the ability of law enforcement agencies to monitor areas, public events, private properties and borders, help to infer and record potentially relevant activities (thefts, car accidents, traffic violations), and make video/audio streams or reports available for future queries.
- *Traffic Avoidance, Enforcement, and Control Systems.* It will be possible to monitor car traffic in big cities or on highways and deploy services that offer traffic routing advices to avoid congestion. In addition, smart parking advice systems based on WMSNs [1] will allow monitoring available parking spaces and provide drivers with automated parking advices, thus improving mobility in urban areas.
- *Advanced Health Care Delivery.* Telemedicine sensor networks [3] can be integrated with 3 G multimedia networks to provide ubiquitous health care services. Patients will carry medical sensors to monitor parameters, for example, body temperature, blood pressure, pulse oximetry, electrocardiogram, and breathing activity. Furthermore, remote medical centers will perform advanced remote monitoring of their patients via video and audio sensors, location sensors, motion, or activity sensors, which can also be embedded in wrist devices [3]. Networks of wearable or video and audio sensors can infer emergency situations and immediately connect elderly patients with remote assistance services or their relatives.
- *Environmental Monitoring.* Several projects on habitat monitoring that use acoustic and video feeds are being envisaged, in which information has to be conveyed in a time-critical fashion. For example, arrays of video sensors have already been used by oceanographers to determine the evolution of sandbars via image processing techniques.
- *Industrial Process Control.* Multimedia content, such as images, temperature, or pressure, may be used for time-critical industrial process control. For example, in quality control of manufacturing processes, details or final products are automatically inspected to find defects. The integration of machine vision systems with WMSNs can simplify and add flexibility to the systems for visual inspections and automated actions that require high-speed, high-magnification, and continuous operation.

14.2.2 Design of Wireless Multimedia Sensor Networks

There are several factors that mainly influence the design of a WMSN, which are outlined as follows:

- *Application-Specific QoS Requirements.* The wide variety of applications envisaged on WMSNs will have different QoS requirements. In addition to data delivery modes typical of scalar sensor networks, multimedia data include *snapshot* and *streaming multimedia* contents. Snapshot-type multimedia content contains event-triggered observations obtained in a short period of time. Streaming multimedia content is generated over a longer period of time and requires sustained information delivery. Hence, a strong foundation is needed in terms of hardware and supporting high-level algorithms to deliver QoS and consider application-specific requirements. These requirements may pertain to multiple domains and can be expressed, among others, in terms of a combination of bounds on energy consumption, delay, reliability, distortion, or network lifetime.
- *High Bandwidth Demand.* Multimedia content, especially video streams, requires transmission bandwidth that is orders of magnitude higher than that supported by currently available sensors. For example, the nominal transmission rate of the state-of-the-art IEEE 802.15.4 compliant components, such as Crossbow's MICAz or TelosB motes, is 250 Kbit/s. Data rates at least one order of magnitude higher may be required for high-end multimedia sensors with comparable power consumption. Hence, high data rate and low power consumption transmission techniques need to be leveraged. In this respect, the ultra-wideband (UWB) transmission technique seems particularly promising for WMSNs, which will be discussed later in this chapter.
- *Multimedia Source Coding Techniques.* Uncompressed raw video streams require excessive bandwidth for a multihop wireless environment. Hence, it is apparent that efficient processing techniques for lossy compression are necessary for multimedia sensor networks. Traditional video coding techniques used for wired and wireless communications are based on the idea of reducing the bit rate generated by a source encoder by exploiting source statistics. To this aim, encoders rely on *intraframe* compression techniques to reduce redundancy within one frame, while they leverage *interframe* compression (also known as *predictive encoding* or *motion estimation*) to exploit redundancy among subsequent frames to reduce the amount of data to be transmitted and stored, thus achieving good rate-distortion performance. Since predictive encoding requires complex encoders and powerful processing algorithms, and entails high energy consumption, it may not be suited for low-cost multimedia sensors. However, it has recently been shown [4] that the traditional balance of complex encoders and simple decoders can be reversed within the framework of so-called *distributed*

source coding, which exploits the source statistics at the decoders, and by shifting the complexity at this end, allows the use of simple encoders. Clearly, such algorithms are very promising for WMSNs and especially for video sensor networks, where it may not be feasible to use existing video encoders at the source node due to processing and energy constraints.

- *Multimedia In-Network Processing.* A WMSN allows performing multimedia in-network processing algorithms on the raw data extracted from the environment. This requires new architectures for collaborative, distributed, and resource-constrained processing that allow for filtering and extraction of semantically relevant information at the edge of the sensor network. This may increase the system scalability by reducing the transmission of redundant information and merging data originated from multiple views on different media and with multiple resolutions. For example, in video security applications, information from uninteresting scenes can be compressed to a simple scalar value or not be transmitted altogether, while in environmental applications, distributed filtering techniques can create a time-elapsed image. Hence, it is necessary to develop application-independent architectures to flexibly perform in-network processing of the multimedia content gathered from the environment.
- *Power Consumption.* Power consumption is a fundamental concern in WMSNs, even more than in traditional wireless sensor networks. In fact, sensors are battery-constrained devices, while multimedia applications produce high volumes of data, which require high transmission rates and extensive processing. While the energy consumption of traditional sensor nodes is known to be dominated by the communication functionalities, this may not necessarily be true in WMSNs. Therefore, protocols, algorithms and architectures to maximize the network lifetime while providing the QoS required by the applications are a critical issue.
- *Flexible Architecture to Support Heterogeneous Applications.* The WMSN architectures will support several heterogeneous and independent applications with different requirements. It is necessary to develop flexible, hierarchical architectures that can accommodate the requirements of all these applications in the same infrastructure.
- *Multimedia Coverage.* Some multimedia sensors, in particular video sensors, have larger sensing radii and are sensitive to the direction of acquisition (directivity). Furthermore, video sensors can capture images only when there is an unobstructed line of sight between the event and the sensors. Hence, coverage models developed for traditional WSNs are not sufficient for predeployment planning of a multimedia sensor network.
- *Integration with the Internet (IP) Architecture.* It is of fundamental importance for the commercial development of a sensor network to provide services that allow querying the network to retrieve useful information from anywhere and at any time. For this reason, future WMSNs will be

remotely accessible from the Internet and will therefore need to be integrated with the IP architecture. The characteristics of WSNs rule out the possibility of all-IP sensor networks and recommend the use of application-level gateways or overlay IP networks as the best approach for integration between WSNs and the Internet.

- *Integration with Other Wireless Technologies.* Large-scale sensor networks may be created by interconnecting local "islands" of sensors through other wireless technologies. This needs to be achieved without sacrificing on the efficiency of the operation within each individual technology.

14.2.3 Ultra-Wideband Technology

The UWB technology has the potential to enable low-power consumption and high data rate communications within tens of meters, which are the characteristics that make it an ideal choice for WMSNs. The U.S. Federal Communications Commission (FCC) defines UWB as a signal with either a fractional bandwidth of 20% of the center frequency or 500 MHz (when the center frequency is >6 GHz). The FCC calculates the fractional bandwidth as $2(f_H - f_L)/(f_H + f_L)$, where f_H represents the upper frequency of the –10 dB emission limit and f_L represents the lower frequency limit of the –10 dB emission limit [5].

Recently, the FCC Notice of Inquiry in 1998 and the First Report and Order in 2002 [6] inspired a renewed flourish of research and development efforts in both academia and industry due to the characteristics of UWB that make it a viable candidate for wireless communications in dense multipath environments. Although UWB signals, as per the specifications of the FCC, use the spectrum from 3.1 to 10.6 GHz with appropriate interference limitation, UWB devices can operate using spectrum occupied by existing radio services without causing interference, thereby permitting scarce spectrum resources to be used more efficiently. Instead of dividing the spectrum into distinct bands that are then allocated to specific services, UWB devices are allowed to operate overlaid and thus interfere with existing services at a low enough power level that existing services would not experience performance degradation. The first report and order by the FCC includes standards designed to ensure that existing and planned radio services, particularly safety services, are adequately protected.

There exist two main variants of UWB. The first, known as time-hopping impulse radio UWB (TH-IR-UWB) [5] and mainly developed by Win and Scholtz [7], is based on sending very short duration pulses (in the order of hundreds of picoseconds) to convey information. Time is divided into frames, each of which is composed of several chips of very short duration. Each sender transmits one pulse in a chip per frame only, and multiuser access is provided by pseudo-random time hopping sequences (THS) that determine in which chip each user should transmit. A different approach, known as MultiCarrier UWB (MC-UWB), uses multiple simultaneous carriers and is usually based on orthogonal frequency division multiplexing (OFDM) [8]. MC-UWB is particularly well suited for avoiding interference because its carrier frequencies can be precisely chosen to

avoid narrowband interference to or from narrowband systems. However, implementing a MC-UWB front-end power amplifier can be challenging due to the continuous variations in power over a very wide bandwidth. Moreover, when OFDM is used, high-speed FFT processing is necessary, which requires significant processing power and leads to complex transceivers.

TH-IR-UWB signals require fast switching time for the transmitter and receiver, as well as highly precise synchronization. Transient properties become important in the design of the radio and antenna. The high instantaneous power during the brief interval of the pulse helps to overcome interference to UWB systems, but increases the possibility of interference from UWB to narrowband systems. The RF front end of a TH-IR-UWB system may resemble a digital circuit, thus circumventing many of the problems associated with mixed-signal integrated circuits. Simple TH-IR-UWB systems can be very inexpensive to construct.

Although no sound analytical or experimental comparison between the two technologies is available to our knowledge, we believe that TH-IR-UWB is particularly appealing for WMSNs. Indeed, it enables high data rate and very low power wireless communications on simple-design low-cost radios (carrierless baseband communications) [7]. Its fine delay resolution properties are appropriate for wireless communications in dense multipath environments by exploiting more resolvable paths [7] and thus providing large processing gain in the presence of interference. Furthermore, it provides flexibility because the data rate can be traded for power spectral density and multipath performance. Finding suitable codes for THS is trivial (as opposed to CDMA codes) and no assignment protocol is necessary. TH-IR-UWB naturally allows for integrated MAC/PHY solutions [9]. Moreover, interference mitigation techniques [9] allow realizing MAC protocols that do not require mutual temporal exclusion between different transmitters. Hence, simultaneous communications of neighboring devices are feasible without complex receivers as required by CDMA. Last, but not least, the large instantaneous bandwidth enables fine time resolution for accurate position estimation [10] and for network time distribution (synchronization), while UWB signals have extremely low-power spectral density with a low probability of intercept/detection (LPI/D), which is particularly appealing for military covert operations.

Particularly appealing for WMSNs are the UWB high data rate with low-power consumption and its positioning capabilities. Positioning capabilities are needed in sensor networks to associate the physical meaning with the information gathered by sensors. Moreover, knowledge of the position of each network device allows for scalable routing solutions [11]. While angle-of-arrival techniques and signal strength based techniques do not provide advantages with respect to other transmission techniques, time-based approaches in UWB allow ranging accuracy on the order of centimeters [12]. Excellent comprehensive surveys of the UWB transmission techniques and the localization techniques for UWB systems are provided in [12] and [13], respectively.

14.2.4 Cross-Layer Design

Recent work has been directed toward designing efficient communication protocols for WMSNs. Existing sensor networks are mostly based on variants of the *carrier sense multiple access with collision avoidance* (CSMA/CA) medium access control (MAC) protocol. The CSMA/CA mechanism has demonstrated to be effective in sharing a common wireless channel among uncoordinated devices in a distributed manner. However, it requires mutually exclusive transmissions, that is, when a device is receiving data, transmissions from all the devices in its transmission range are impeded. Mutual exclusion is achieved by coordinating the transmissions of different sensors in a distributed manner mainly by means of two mechanisms, that is, *carrier sense* and *random timers* to defer transmissions. While random timers lead to variable and uncontrollable access delays, carrier sense causes consistent energy consumption for idle listening; still, frequent collisions occur due, for example, to the well-studied *hidden node* problem, which in turn leads to increased energy consumption and delays. The transmitted power of currently off-the-shelf motes, for example, Crossbow's MicaZ based on the Chipcon 2420 chipset, is still high on the order of 1 mW. Exact Tx : Rx : Idle power ratios depend on hardware, but idle power is in general not negligible and accounts for a considerable portion of the overall energy consumption. Introducing sleep periods to reduce idle listening reduces the energy consumption at the expense of latency and coordination complexity.

For the above reasons, although recent proposals have modified existing protocols based on CSMA/CA and geographical routing to provide delay-sensitive and error-resilient services in sensor networks, we believe that the application requirements of WMSNs call for a new design perspective and next-generation wireless technologies. Hence, a new cross-layer communication architecture to reliably and flexibly deliver QoS to heterogeneous applications in WMSNs is proposed in Ref. [9]. According to the application requirements, this solution leverages and controls the interactions among the functionalities of different layers. The architecture is based on the TH-IR-UWB transmission technique and its main design principles are outlined as follows:

Network Layer QoS Support Enforced by a Cross-Layer Controller

The proposed cross-layer communication architecture provides QoS support at the network layer, that is, it provides packet-level service differentiation in terms of throughput, end-to-end packet error rate, and delay. Figure 14.1 shows the cross-layer control unit (XLCU) used in the architecture. XLCU configures and controls the networking functionalities at the physical, MAC, and network layers by using a unified logic that makes decisions based on (1) application requirements specified by the application layer; (2) the status of the functional blocks implementing the networking functionalities. In this way, cross-layer interactions can be leveraged without sacrificing on upgradeability, modularity, and ease of system design.

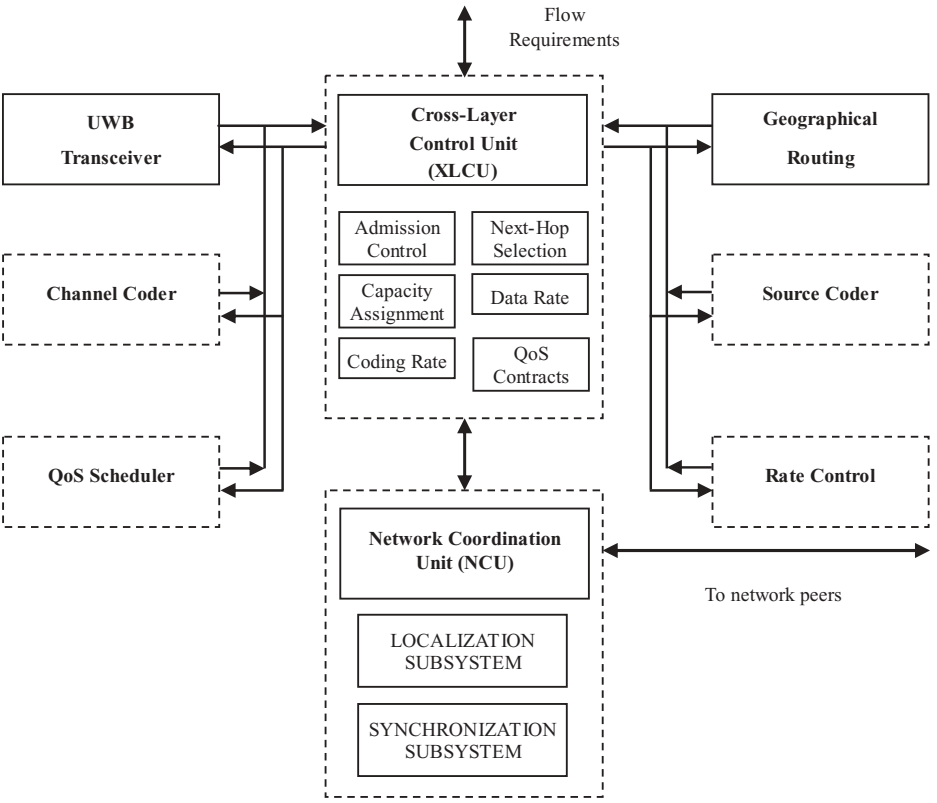


Fig. 14.1 Cross-layer control unit.

Geographical Routing

Time-based approaches in UWB allow ranging accuracy on the order of centimeters [12]. Hence, the geographical routing module leverages geographical information to provide QoS. Positioning capabilities are needed in sensor networks to associate the physical meaning with the information gathered by sensors. Moreover, knowledge of the position of each network device allows for scalable routing solutions [9].

Hop-by-Hop QoS Contracts

End-to-end QoS requirements are enforced through local interactions. Each device is responsible for locally guaranteeing given performance objectives. The global end-to-end requirement is thus guaranteed by the joint local decisions made by the participating devices.

Receiver-Centric Scheduling for QoS Traffic

In multihop wireless environments, interference is location dependent. For this reason, we provide QoS through receiver-centric scheduling. A receiver can be responsive to the dynamics of the channel based on local measurements and consequently control loss recovery and rate adaptation, thus avoiding feedback overheads and latency.

UWB Physical/MAC Layer

We rely on an integrated MAC and physical layer based on UWB. Like CDMA, TH-IR-UWB allows multiple transmissions in parallel. This allows devising MAC protocols with minimal coordination. While CDMA is usually associated with complex transceivers and cumbersome code assignment protocols, TH-IR-UWB transceivers are simple to implement.

Dynamic Channel Coding

As previously discussed, power control is not beneficial in TH-IR-UWB. Hence, adaptation to interference at a receiver is achieved through dynamic channel coding, which can be considered as an alternative form of power control because it modulates the energy per bit based on the interference perceived at the receiver [9].

14.3 WIRELESS SENSOR AND ACTOR NETWORKS

Another innovative research area is what is usually referred to as wireless sensor and actor networks (WSANs) [14]. A WSAN is a heterogeneous network composed of embedded devices, where a subset of the devices (sensors) gather environmental information and another set of more powerful and less resource-constrained devices (actors) physically manipulate the environment and interact with it.

Recent technological advances have led to the emergence of distributed WSANs, which are capable of observing the physical world, processing the data, making decisions based on the observations, and performing appropriate actions/tasks. In WSANs, the sensing and acting tasks are performed by sensor and actor nodes, respectively. Sensors are low-cost and low-power devices with limited sensing, computation, and wireless communication capabilities. Actors are resource-rich devices equipped with better processing capabilities, higher transmission powers, and longer battery lifetime. Moreover, the number of sensor nodes deployed in a target area may be on the order of hundreds or thousands, whereas such a dense deployment is usually not necessary for actors because actors have higher capabilities and can act on large areas.

It may be worth specifying the meaning that we attribute to the term *actor* and how this is different from the more conventional notion of *actuator*. An actuator is a device to convert an electrical control signal to a physical action and may be used for flow-control valves, pumps, positioning drives, motors, switches,

relays, and meters. It constitutes the mechanism by which an agent acts upon the physical environment. An actor, besides being able to act on the environment by means of one or several actuators, is also a *single network entity* that performs networking-related functionalities, that is, receive, transmit, and relay data. For example, a robot may interact with the physical environment by means of several motors and servomechanisms (actuators). However, from a networking perspective, it constitutes a single entity, which we refer to as actor.

The collaborative operation of the sensors enables the distributed sensing of a physical phenomenon. After the sensors detect an event that is occurring in the environment, the event data is processed and transmitted to the actors in a distributed manner, which gather, process, and eventually reconstruct the event data. The process of establishing data paths between sensors and actors is referred to as sensor–actor coordination [14]. Once the event is detected, the actors coordinate to reconstruct its data, estimate its characteristics, and make a collaborative decision on how to perform the action. This process is referred to as actor–actor coordination [14]. As a result, the operation of a WSAN can be thought of as an event-sensing, communication, decision, and acting loop, whose objective is to timely react to sensor information with an effective action. For this reason, real-time coordination and communication is an important concern in WSANs to guarantee the timely execution of correct actions. The energy efficiency of network communications is also crucial because sensors are resource-constrained nodes with limited battery lifetime. Furthermore, sensor network protocols and algorithms should be scalable and localized because the number of nodes can be arbitrarily high. Given the above requirements, we propose to base the sensor–actor coordination on an event-driven partitioning paradigm in the framework of *geographical routing*.

14.3.1 Applications of Wireless Sensor and Actor Networks

Several applications for a WSAN are concerned with enhancing and complementing existing sensor network applications. In these applications, the performed actions serve the purpose of enhancing the operation of the sensor network by enabling or extending its monitoring capability. Conversely, we are concerned with new applications where actors are part of the network and perform actions based on the information gathered by sensors. We envision that WSANs will be an integral part of systems, for example, battlefield surveillance, nuclear, biological or chemical attack detection, home automation, and environmental monitoring [1]. In fire detection applications, for example, sensors can relay the exact origin and intensity of the fire to water sprinkler actors that will extinguish the fire before it spreads. Moreover, sensors can detect plumes; that is, visible or measurable discharges of contaminants in water or in the air, and actors can reactively take countermeasures. Similarly, motion, acoustic, or light sensors in a building can detect the presence of intruders and command cameras or other instrumentations to track them. Alternatively, mobile actors can be moved to the area where the intruder has been detected to get high resolution images and prompt or block the intruder.

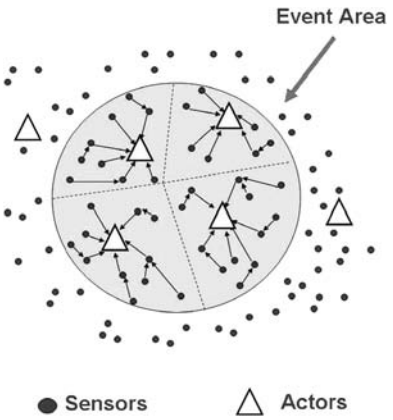


Fig. 14.2 Event-driven clustering with multiple actors.

14.3.2 Sensor and Actor Coordination

Given the specific requirements of WSANs, we propose to base the sensor–actor coordination on an event-driven clustering paradigm, where cluster formation is triggered by an event and clusters are created on the fly to optimally react to the event in Ref. [15]. In this approach, sensors detecting an event coordinate to optimally associate each sensor with an actor. In this way, only the event area is clustered and each cluster consists of those sensor nodes that send their data to the same actor. The resulting architecture is shown in Fig. 14.2. Next, we will describe the sensor–actor and actor–actor coordination problems as well as the proposed solutions. The performance results will also be discussed briefly.

14.3.2.1 Sensor–Actor Coordination. As discussed previously, sensor–actor communications may have real-time requirements. Hence, we introduce a novel notion of reliability that accounts for the percentage of packets generated by the sensors in the event area and received within a predefined latency bound, which we refer to as *reliable packets*. The *latency bound B* is the maximum allowed time between the instant when the physical features of the event are sampled by the sensors and the instant when the actor receives a data packet describing these event features. A data packet that does not meet the latency bound *B* when it is received by an actor is said to be *expired* and thus *unreliable*. Similarly, a data packet received within the latency bound *B* is said to be *unexpired* and thus *reliable*. The *event reliability r* is the ratio of *reliable* data packets over all the packets generated in a decision interval. Whenever one or more packets are dropped by an intermediate sensor, the actor is notified about the lost packet(s) in the header of the next data packet so that the packet loss can be taken into account in the computation of the reliability. The *event reliability threshold* r_{th} is the minimum *event reliability* required by the application. Given

these formal definitions, the sensor–actor coordination problem consists of establishing data paths from each sensor residing in the event area to the actors by (1) ensuring that the observed reliability r is above the event reliability threshold r_{th} (i.e., $r > r_{th}$); (2) minimizing the energy consumption associated with data delivery paths. We refer to our solution for the sensor–actor coordination problem as *event-driven clustering with multiple actors*. The objective of the optimization problem is to find *data aggregation trees* (da-trees) from all the sensors that reside in the event area (referred to as sources) to the appropriate actors. A da-tree is composed by aggregating individual *flows*, where a flow is defined as a connection between a sensor and an actor. The optimal strategy for event-driven clustering is formulated as an *integer linear program* (ILP) [16].

In addition, we introduce a scalable and distributed protocol to address the sensor–actor coordination problem in WSANs. The objective of the protocol is to build energy-efficient da-trees between the sources that reside in the event area and the actors to provide the required reliability r_{th} with minimum energy consumption. We refer to the protocol as *distributed event-Driven clustering and routing* (DECR) protocol. Based on the observed reliability that each actor advertises, the proposed protocol favors a local behavior for each individual sensor node that results in a global network behavior compliant with the application requirements; that is, provide event reliability r above the required threshold r_{th} and minimize the energy consumption. The reliability is controlled based on the idea of adjusting the delays by modifying the average end-to-end path length.

The optimization problem was implemented in AMPL [17] and solved using CPLEX [18]. The start-up, speed-up, and aggregation states were implemented in a C++ simulator, which we used to evaluate the energy consumption, and in the J-Sim Simulator [19], which implemented the whole protocol stack of a sensor node from the physical layer to the application layer, including CSMA/CA MAC. We considered different simulation scenarios. In Scenario 1, the deployment area is circular with a radius equal to 20 m. For each deployed sensor, the distance from the center of the area and the angle are uniformly distributed random variables. In Scenario 2, sensor nodes are randomly deployed in a square area of 25 m × 25 m. The event area is circular, with a varying radius ranging in [2,12] m in different simulations. The epicenter of the event area is randomly selected such that the event area completely falls into the terrain. Scenario 3 is similar to Scenario 2, but the side of the square area is 100 m. Four actors are randomly deployed in each scenario. The transmission range of sensors is set to 10 m. Since the global network behavior depends on several application-dependent parameters, here we only present the results related to particular network configurations that constitute upper and lower bounds on the achievable performance. Hence, we refer to start-up configuration, speed-up configuration, and aggregation configuration as those configurations where all nodes are in the start-up state, speed-up state, and aggregation state, respectively. This allows us to discuss the benefits of the proposed solution without depending on the choice of parameters that govern the transitions among states.

Let us first consider the difference between the optimal solution to the event-driven clustering problem, and the energy consumption in the start-up, speed-up, and aggregation configurations with varying event ranges. Noticeably, the optimal solution is almost independent of the event range. This is due to two contrasting phenomena. The number of sources increases when the event range increases, leading to a potentially higher energy consumption; conversely, since more nodes are involved, aggregation can be increasingly leveraged. These two trends compensate each other. Conversely, the energy consumption in the start-up and speed-up configurations highly increases with the event range.

Now, let us consider the average energy consumption versus the number of sensors with different event ranges for the start-up and aggregation configurations in Scenario 2. The energy consumption of the aggregation configuration is two orders of magnitude lower than in the start-up configuration. The energy consumption increases sublinearly with the number of sensors. Interestingly, not only is the energy consumption of the speed-up configuration around one order of magnitude higher than in the start-up configuration; also, when the aggregation configuration is reached from a speed-up configuration, the network converges to a less energy-efficient configuration, compared to when the aggregation configuration is reached directly from the start-up configuration.

14.3.2.2 Actor–Actor Coordination. The objective of the actor–actor coordination is to select the best actor(s) to perform appropriate action on the event area. Actor–actor coordination presents several analogies with the so-called *multi-robot task allocation* (MRTA) problem encountered in robotics. In fact, a fundamental question faced when designing cooperative multi-robot systems is *Which robot should execute which task in order to cooperatively achieve the global goal*? [20]. At the end of the sensor–actor coordination phase, one or multiple actors, which are called *collectors*, receive sensor readings from the sources that sense the event. These sources define the *event area*. The event area corresponds to the *action area*; that is, the area where the actors should act. In particular, each collector receives data from a subset of the sources (*cluster*). Each cluster area identifies a portion of the action/event area and is under the responsibility of the corresponding collector. However, the collector may not be able to act on its entire responsible area; that is, this area may not be entirely within the collector's *action range*. The action range defines the circular area where an actor is able to act. Moreover, the collector may not be the "best" actor for that task in terms of *action completion time* and/or *energy consumption*, where the former is the time to perform the action and the latter is the energy consumed for the action. For these reasons, actor–actor coordination is required before initiating the action. The coordination objective of multiple collector actors is to find the optimal actors to timely act on the portion of the event area under their own responsibility.

In addition, we present a distributed solution to the actor–actor coordination problem, which is based on a real-time auction protocol that describes the behavior of the actors participating in transactions as buyers/sellers. The objective of

the auction is to select the best set of actors to perform the action on each *overlapping area*, defined as an area where multiple actors can act upon.

The optimization problem was implemented in AMPL and solved using the MINLP (mixed integer non-linear programming) solver available through the NEOS optimization server [21]. We compare the average residual energy with three different solution approaches, namely, the *optimal*, *1-actor*, and *localized auction*. In the optimal solution, the best set of actors is chosen so that the average residual energy of the involved actors is maximized, while guaranteeing that the action is completed before the action completion time. In the 1-actor heuristic, the action is performed by one actor only for each overlapping area; that is, the actor with the highest residual energy after the completion of the action. In the localized auction each overlapping area is taken care of by an auctioneer that divides it among the actors based on their bids. In the experiments performed, we concentrate on two scenarios with three overlapping areas, one with homogeneous actors and one with heterogeneous actors (half of which are low-efficiency actors, while the other one-half are high-efficiency actors). Interestingly, the localized auction mechanism leads to near-optimal residual energy, as each auctioneer calculates the optimal solution separately for its overlapping area. However, this greatly simplifies the problem and can be achieved with local communications among actors. Moreover, in the heterogeneous scenario, the proposed localized solution effectively exploits the high-efficiency actors, thus reducing the dissipated energy to complete the action.

14.4 SENSOR NETWORK APPLICATIONS IN CHALLENGING ENVIRONMENTS

In this section, we introduce sensor network applications in challenging environments. In particular, we describe ongoing and open research issues for UW-ASNs and WUSNs, which consist of a number of sensors deployed underground. In such challenging environments, the medium propagation characteristics require new communication protocols explicitly designed for such environments. Challenges and recent efforts in the development of such applications are also described and discussed.

14.4.1 Underwater Acoustic Sensor Networks

Underwater sensor networks have the potential to enable unexplored applications and to enhance human's ability to observe and predict the ocean. Unmanned or autonomous underwater vehicles (UUVs or AUVs) equipped with underwater sensors are also envisioned to find applications in exploration of natural undersea resources and gathering of scientific data in collaborative monitoring missions. These potential applications will be made viable by enabling wireless communication among underwater devices. UW-ASNs [22] will consist of sensors

and vehicles deployed underwater and networked via wireless acoustic links to perform collaborative monitoring tasks.

UW-ASNs enable a broad range of applications, including ocean sampling, environmental monitoring, undersea explorations, disaster prevention, seismic monitoring, equipment monitoring, assisted navigation, distributed tactical surveillance, and mine reconnaissance. Acoustic communications are the typical physical-layer technology used in underwater networks. In fact, radio waves propagate at long distances through conductive salty water only at extra low frequencies (30–300 Hz), which require large antennae and high transmission power. Optical waves do not suffer from such high attenuation, but are affected by scattering. Furthermore, transmitting optical signals requires high precision in pointing the narrow laser beams. Thus, links in underwater networks are typically based on *acoustic wireless communications* [22]. The traditional approach for *ocean-bottom* or *ocean-column* monitoring is to deploy underwater sensors that record data during the monitoring mission and then recover the instruments [23]. This approach has several disadvantages:

1. Recorded data cannot be accessed until the instruments are recovered, which may happen several months after the beginning of the monitoring mission.
2. Interaction between onshore control systems and the monitoring instruments is not possible, which impedes any adaptive tuning or reconfiguration of the system.
3. If *failures* or *misconfigurations* occur, it may not be possible to detect them before the instruments are recovered.
4. The amount of data that can be recorded by every sensor during the monitoring mission is limited to the capacity of the onboard storage devices.

These disadvantages can be overcome by connecting untethered underwater instruments by means of wireless links that rely on acoustic communications. Although there exist many recently developed network protocols for wireless sensor networks, the unique characteristics of the underwater acoustic communication channel, for example, limited capacity and high and variable propagation delays [23], require very efficient and reliable data communication protocols.

The major challenges in the design of underwater acoustic networks include the following aspects:

- The available bandwidth of the underwater channel is severely limited.
- The underwater channel is impaired because of multipath and fading.
- Propagation delay is five orders of magnitude higher than in radio frequency (RF) terrestrial channels.
- Variable and high bit error rates as well as temporary losses of connectivity (shadow zones) can be experienced.

- Batteries of underwater sensors have limited power and usually cannot be recharged.
- Underwater sensors are prone to failures because of fouling and corrosion.

14.4.1.1 Differences from Terrestrial Sensor Networks. The main differences between terrestrial sensor networks and underwater sensor networks include the following aspects:

- *Cost.* While terrestrial sensor nodes are expected to become increasingly inexpensive, underwater sensors are expensive devices. This is mainly due to the low economy of scale caused by a small relative number of suppliers.
- *Deployment.* While terrestrial sensor networks are densely deployed, the deployment of underwater sensor networks is generally sparser.
- *Power.* The power needed for acoustic underwater communications is higher than in terrestrial radio communications because of the different physical-layer technology (acoustic vs. RF waves), the longer distances, and more complex signal processing techniques implemented at the receivers.
- *Memory.* While terrestrial sensor nodes have very limited storage capacity, underwater sensors need to be able to do some data caching as the underwater channel may be intermittent, and therefore have larger storage capacity.
- *Spatial Correlation.* While the readings from terrestrial sensors are often correlated, this is more unlikely in underwater networks due to the higher distance among sensors.

14.4.1.2 Factors Influencing the Design of Underwater Protocols. Acoustic communications in the underwater environment are mainly influenced by *transmission loss*, *noise*, *multipath*, *Doppler spread*, and *high and variable propagation delay*. All these factors determine the *temporal and spatial variability* of the acoustic channel, and make the available bandwidth of the underwater acoustic channel limited and dramatically dependent on both range and frequency. Long-range systems that operate over several tens of kilometers may have a bandwidth of only a few kilohertz, while a short-range system operating over several tens of meters may have >100 kHz of bandwidth. In both cases, these factors lead to low bit rates [24] on the order of tens of kbps for existing devices.

Underwater acoustic communication links can be classified according to their range as *very long*, *long*, *medium*, *short*, and *very short* links [22]. Acoustic links are also roughly classified as *vertical* and *horizontal*, according to the direction of the sound ray with respect to the ocean bottom. Their propagation characteristics differ considerably, especially with respect to time dispersion, multipath spreads, and delay variance. In the following, as usually done in oceanic literature, *shallow water* refers to water with depth lower than 100 m, while *deep water* is

used for deeper oceans. The main factors that influence acoustic communications include the following aspects:

- *Transmission Loss.* It consists of *attenuation* and *geometric spreading*. The attenuation is mainly provoked by absorption due to conversion of acoustic energy into heat, and increases with distance and frequency.
- *Noise.* It can be classified as *man-made noise* and *ambient noise*. The former is mainly caused by machinery noise and shipping activity, while the latter is related to hydrodynamics, and to seismic and biological phenomena.
- *Multipath.* Multipath propagation may be responsible for severe degradation of the acoustic communication signals because it generates inter symbol interference (ISI). The multipath geometry depends on the link configuration. Vertical channels are characterized by little time dispersion, whereas horizontal channels may have long multipath spreads.
- *High Delay and Delay Variance.* The propagation speed in the UW-A channel is five orders of magnitude lower than in the radio channel. This large propagation delay (0.67 s/km) and its variance can reduce the system throughput.
- *Doppler Spread.* The Doppler frequency spread can be significant in UW-A channels [22], causing degradation in the performance of digital communications.

14.4.1.3 Communication Architectures. This section presents some reference communication architectures for UW-ASNs, which constitute a basis for discussion of the challenges associated with the underwater environment.

A reference architecture for two-dimensional (2D) underwater networks is shown in Fig. 14.3a. A group of sensor nodes are anchored to the bottom of the ocean. Underwater sensor nodes are interconnected to one or more *underwater gateways* (uw-gateways) by means of wireless acoustic links. Uw-gateways are network devices in charge of relaying data from the ocean bottom network to a surface station. To achieve this objective, they are equipped with two acoustic transceivers, namely, a *vertical transceiver* and a *horizontal* transceiver. The horizontal transceiver is used by the uw-gateways to communicate with the sensor nodes in order to (1) send commands and configuration data to the sensors (uw-gateway to sensors); (2) collect monitored data (sensors to uw-gateway). The vertical transceiver is used by the uw-gateways to relay data to a *surface station*. In deep water applications, vertical transceivers must be long-range transceivers. The surface station is equipped with an acoustic transceiver that is able to handle multiple parallel communications with the deployed uw-gateways. It is also endowed with a long range RF and/or satellite transmitter to communicate with the *onshore sink* (os-sink) and/or to a *surface sink* (s-sink). In shallow water, bottom-deployed sensors/modems may directly communicate with the surface buoy, with no specialized bottom node (uw-gateway).

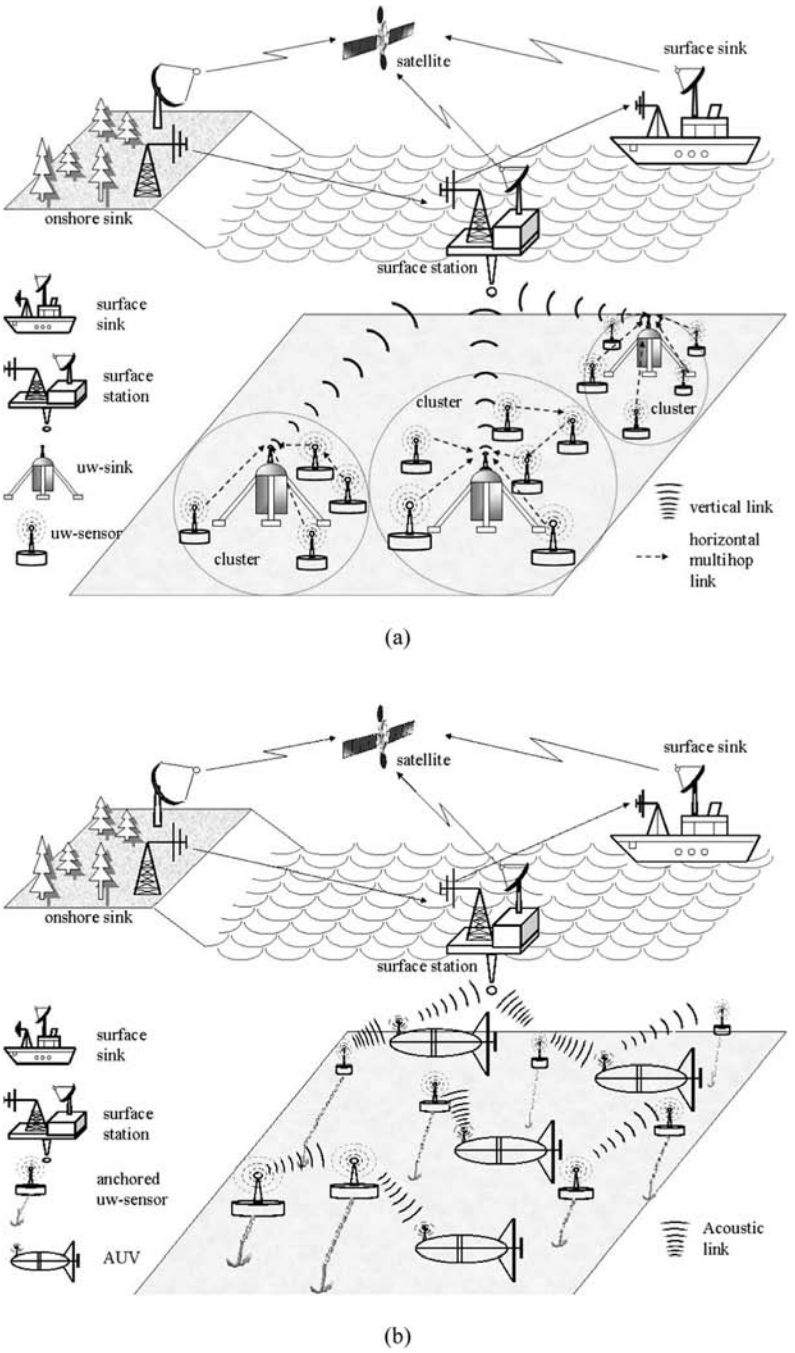


Fig. 14.3 (a) Two-dimensional underwater sensor network. (b) Three-dimensional sensor network with autonomous underwater vehicles.

Three-dimensional (3D) underwater networks are used to detect and observe the phenomena that cannot be adequately observed by means of ocean bottom sensor nodes; that is, to perform cooperative sampling of the 3D ocean environment. In 3D underwater networks, sensor nodes float at different depths to observe a phenomenon. In this architecture, each sensor is anchored to the ocean bottom and equipped with a floating buoy that can be inflated by a pump, as shown in Fig. 14.3b. The buoy pushes the sensor toward the ocean surface. The depth of the sensor can then be regulated by adjusting the length of the wire that connects the sensor to the anchor by means of an electronically controlled engine that resides on the sensor. Sensing and communication coverage in a 3D environment are rigorously investigated in Ref. [25]. In Ref. [26], we present a statistical analysis for different deployment strategies for 2D and 3D communication architectures for UW-ASNs.

AUVs can function without tethers, cables, or remote control, and therefore they have a multitude of applications in oceanography, environmental monitoring, and underwater resource studies. Previous experimental work has shown the feasibility of relatively inexpensive AUV submarines equipped with multiple underwater sensors that can reach any depth in the ocean (see Fig. 14.3b). The integration of UW-ASNs with AUVs requires new network coordination algorithms for such adaptive sampling and self-configuration.

14.4.2 Wireless Underground Sensor Networks

The usefulness of WSNs as a remote monitoring technology is not limited to traditional terrestrial applications introduced in Section 14.2 or to wireless sensors utilizing acoustic communications for underwater monitoring, as described in the previous section. We believe that another domain with useful applications of WSN technology lies in the underground; that is, Wireless Underground Sensor Networks (WUSNs) [27], which are useful in the following few areas:

- *Agriculture.* WUSNs can be used to monitor soil conditions so that parameters, for example, water content, mineral content, salinity, and temperature, can be maintained at optimal levels. Real-time knowledge of soil conditions is also beneficial to landscaping, where WUSNs can be combined with automatic sprinkler systems such that grass, trees, and flowers are watered only when needed.
- *Security.* Sensors buried at a shallow depth could be used to detect movement via pressure, vibration, or sound. This may be useful for business and home security, as well as for military applications.
- *Infrastructure Monitoring.* A significant amount of infrastructure, including plumbing, as well as electrical and communications wiring, exists underground. With many miles of pipes to monitor, wireless sensors will allow for quick and cost-efficient deployment of a leakage detection system. The WUSNs can also be used to monitor the soil around underground storage tanks such as those at a fuel station.

While underground sensors are already in use for many of these applications [28], most existing solutions are wired. Those that are wireless require aboveground antennas and are only capable of direct communication with a centralized base station. Based on this fact, we define a WUSNs as a group of nodes whose means of data transmission and reception (e.g., an antenna when electromagnetic waves are used for communication) is completely subterranean. This includes situations in which a node is underground, yet in an open space, for example, a cave or mine, as well as when a node is completely embedded within dense soil or rock. Given the usefulness of monitoring conditions in the underground, we set out to determine whether current WSN solutions are applicable to the underground sensing environment. Since soil and rock are lossy dielectric materials, the propagation of electromagnetic waves in the underground environment can be severely impeded [29]. To determine whether existing WSN solutions can be used in the underground, we performed tests of the communication capabilities of the popular MicaZ WSN motes from Crossbow [30] when buried in soil at various depths.

14.4.2.1 Experimental Setup. The experiments described below are designed to test the packet error rate and the received signal strength of correctly received packets for a communication link between two underground sensors, as well as between an underground sensor and one on or above the surface. The latter scenario is useful for communication between the WUSNs and an aboveground sink. An illustration of the setup is given in Fig. 14.4. These parameters were evaluated for both the forward and reverse channels between the sensors while varying the depth (d) of the underground sensor, the height (h) of the aboveground sensor, and the horizontal distance (l) between the two. Here we define the forward channel as transmissions from Sensor A to Sensor B, and the reverse channel as transmissions from Sensor B to Sensor A. The remainder of this section discusses the physical test environment, relevant hardware characteristics of the MicaZ, and the software we implemented on the motes to gather the statistics of interest [31].

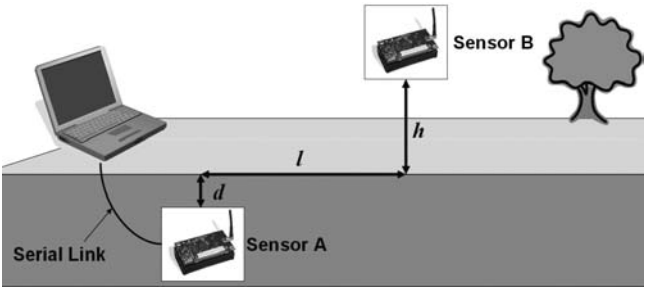


Fig. 14.4 Experimental set up.

14.4.2.2 Physical Environment. Soil properties, for example, density, water content, and mineral content, play an important role in determining losses for a propagating electromagnetic wave in the underground [29]. We report that the experiments were carried out in an open field in midtown Atlanta, where the soil had a dense and clay-like consistency with moderate water content. Wet clay soils produce the most attenuation of an electromagnetic wave, while dry sandy soils produce the least [29].

14.4.2.3 MicaZ Wireless Sensor Motes. The wireless sensor mote we used for these experiments was the MicaZ from Crossbow. MicaZ motes operate in the 2.4-GHz band and use a Zigbee-compliant Chipcon CC2420 radio. Although the MicaZ is Zigbee compliant, as of this writing TinyOS, the MicaZ's operating system, does not implement the Zigbee standard. Instead, the B-MAC media access protocol described in Ref. [32] is used. The MicaZ's radio supports variable output power, which can be set anywhere between –24 and 0 dBm [30]. For these experiments, the radio was always set to its maximum transmission power of 0 dBm. The stated reception sensitivity of the radio is a minimum of –90 dBm, with a typical value of –94 dBm [30]. Crossbow advertises the indoor communication range of the motes as 20–30 m, and an outdoor range of 75–100 m. The motes were used with the supplied quarter-wavelength whip antenna.

14.4.2.4 Software Design. The tests were designed to collect packet error rates at the application layer, as well as the received signal strength indicator of correctly received packets. To accomplish this, we created an application using TinyOS designed to send packets of a specified size at a specified rate between a source mote and a destination mote. One of the motes was connected to the Crossbow MIB510 programming board, allowing for two-way communication with a laptop via a wired serial connection, illustrated in Fig. 14.4. To begin a test, the parameters (e.g., the number of packets, packet size, interval between packets, source and destination node IDs) are passed from the laptop through the serial connection to the underground mote. This mote then forwards the parameters to the remote mote and waits for either an acknowledgment of the setup or one of the test packets. The remote mote, upon receiving the setup packet, sends an acknowledgment to the sink and begins transmitting using packets of the specified size and with the specified interval between packets.

14.4.2.5 Experimental Results. Although tests were attempted between two underground motes at the same depth, we found communication impossible regardless of the horizontal separation. Therefore, we focus on communication between one underground mote and one aboveground mote. Data was also gathered for the underground sensor at a depth of 0 cm (on the surface) as a baseline. Experiments showed a direct correlation between the depth of the underground sensor and the amount of signal attenuation experienced—the greater the amount of soil through which the signal must propagate, the greater the experienced attenuation. For any given horizontal separation, when

comparing the attenuation to that experienced by both sensors at a depth of 0 cm, an additional attenuation of 15–20 dB is seen for one sensor at a depth of 6 cm. At a depth of 13 cm, the difference in attenuation increases to ~25 dB. Given that the CC2420 radio used on the MicaZ is capable of a maximum transmission power of 0 dBm and typically has a reception sensitivity of –90 dBm, a 25-dB loss overtop losses due to geometric spreading, interference, and other traditional factors, represents a significant reduction of the communication range of these motes. There is a clear advantage for transmissions in the forward channel (Sensor A to Sensor B). In all tests, in fact, the strength of the signal received in this direction was 2–3 dB stronger than a transmission in the reverse direction. The experiments were performed without moving the motes—the direction of the transmission was modified in software. Therefore, issues such as differences in antenna orientation and density of the soil atop the underground mote, which could occur if the placement of the motes needed to be exchanged, have not affected the results. Although this result is interesting, it may not be a concern because most communications will be directed from the underground sensors to an aboveground sink. The WUSNs sink nodes will likely be located at the surface, where they can more easily be interfaced with a data collection system or long-haul radio to serve as a backhaul for sensor readings.

We also performed tests when the aboveground sensor was elevated a distance of 1 m off the ground. This case may occur with a mobile sink that moves around above the WUSNs deployment area to collect sensor readings directly from the motes. In this case, an interesting phenomenon occurs. The received power of the signal increases slightly for the underground mote at a depth of 13 cm as the horizontal separation between the sender and the receiver increases from 50 to 200 cm. The signal then falls off with distance as expected. This slight increase is likely due the radiation pattern of the antennas used on the MicaZ. The aboveground mote had its antenna oriented in the vertical direction, where it has a null in its pattern at either end. Thus, as the aboveground mote moved further away, the null of its radiation pattern was no longer pointed directly at the underground mote. Interestingly, the elevation of the aboveground mote improved the achievable communication range from 4 to 7 m. As the received signal strength approaches the receiver sensitivity of –90 dBm, the packet error rate approaches 100%. The packet error rate typically remains <20% as long as the received signal strength was >~–85 dBm.

14.5 CROSS-LAYER DESIGN FOR WIRELESS SENSOR NETWORKS

This section provides an overview of the future trends in communication protocol design for WSNs to accommodate the various new application areas explained in the previous sections. There exists an exhaustive amount of research on enabling efficient communication in WSNs [1]. Most of the proposed communication protocols improve the energy efficiency to a certain extent by exploiting the collaborative nature of WSNs and its correlation characteristics. However, the

main commonality of these protocols is that they follow the traditional layered protocol architecture. Recent work on WSNs [33,34] reveal that cross-layer integration and design techniques result in significant improvement in terms of energy conservation due to three main reasons. First, the stringent energy, storage, and processing capabilities of wireless sensor nodes necessitate such an approach. The significant overhead of layered protocols results in high inefficiency. Moreover, recent empirical studies necessitate that the properties of low power radio transceivers and the wireless channel conditions be considered in protocol design [33,35]. Finally, the event-centric paradigm of WSNs requires application-aware communication protocols. Considering the scarce energy and processing resources of WSNs, joint optimization and design of networking layers are needed; that is, cross-layer design stands as the most promising alternative to inefficient traditional layered protocol architectures.

This section describes the performance improvement and the consequent risks of cross-layer design methodologies. More specifically, we overview significant findings and representative communication protocols that are relevant to the cross-layering philosophy. So far, the term *cross-layer* has carried at least two meanings. On the one hand, the cross-layer interaction is considered, where the traditional layered architecture is preserved, while each layer is informed about the state of other layers. However, the internal architecture of each layer still stays intact. On the other hand, there is still much to be gained by rethinking the mechanisms of network layers in a unified way so as to provide a single communication module for efficient communication in WSNs. We present some concerns and precautionary considerations regarding cross-layer design architectures and overview the research directions still open in this area.

14.5.1 Cross-Layer Resource Allocation

Resource allocation in the context of multihop wireless networks has been extensively studied in the last few years. However, in most cases, resource allocation problems are treated either heuristically, or without considering cross-layer interdependencies, or by considering pairwise interactions between isolated pairs of layers.

A typical example of the tight coupling between the functionalities of different layers is the interaction between the congestion control and power control mechanisms [36]. In multihop wireless networks, the attainable capacity of each wireless link depends on the interference levels, which in turn depend on the power control policy. Hence, congestion control and power control are inherently coupled and should not be treated separately. Furthermore, the physical, MAC, and routing layers together impact the contention for network resources. The physical layer has a direct impact on a MAC mechanism because it affects the interference at the receivers. The MAC layer determines the bandwidth allocated to each transmitter, which naturally affects the performance of the physical layer in terms of successfully detecting the desired signals. Consequently, designing models that focus on the joint power control

and MAC problem and/or power control and routing problem is a major challenge.

14.5.1.1 Pairwise Resource Allocation. The problem of jointly optimizing resource allocation has generally been considered at two layers of the protocol stack. Typical examples of pairwise resource allocation are joint scheduling and power control, joint routing and power control, and joint routing and scheduling. For the joint scheduling and power control problem, the main challenge is to schedule the maximum number of links in the same timeslot, where the objective is to develop a power-control-based multiple access algorithm [37]. In Ref. [38], the joint power control and scheduling problem is addressed under the assumption that the session paths are already given. This work aims at satisfying the rate requirements of the sessions not only in the long term, as considered in Ref. [39], but also in the short term, in order to prevent the sessions with low jitter or bounded delay requirement from suffering from the ambiguity of the long-term guarantees. The main conclusion in Ref. [38] is that independent decisions at different layers for achieving a local objective would deteriorate the performance of other layers. An example of joint resource allocation at the physical and routing layers is discussed in Ref. [9], where the proposed analytical framework allows analyzing the relationship between the energy efficiency of the geographical routing functionality and the extension of the topology knowledge range for each node. Wider topology knowledge may improve the energy efficiency of the routing tasks but increases the cost of topology information due to signaling packets needed to acquire this information. Finally, joint routing and scheduling for multihop wireless networks has been considered in Ref. [40], where the necessary and sufficient conditions for the achievability of a given rate vector between sources and sink are determined.

14.5.1.2 Joint Routing, Scheduling, and Power Control. In addition to pairwise cross-layer resource allocation studies, joint routing, scheduling, and power control is a difficult problem in WSNs. In particular, the minimization of the total average transmission power, subject to given constraints on the minimum the average data rate per link, as well as the peak transmission power constraints per node, has been investigated in Ref. [39]. Interestingly, even though the focus is on minimizing transmission power, the optimal joint allocation does not necessarily route traffic over minimum-energy paths. The problem of maximizing the network lifetime of a WSN also has been tackled by investigating joint resource allocation at the physical, MAC, and routing layers [41]. This problem is hard to solve and requires extensive communication among the nodes.

The joint optimal power control, scheduling, and routing problem is also formulated for time-hopping ultra-wideband networks with the objective of maximizing log utility of flow rates subject to power constraint nodes [42]. The problem can be solved by using a centralized algorithm. The results obtained in Ref. [42] reveal that given the optimization objective, power control is not needed: The design of the optimal MAC protocol is independent of the choice of the

routing protocol; and transmitting over minimum-energy routes is always optimal even though the objective is to maximize the transmission rate.

14.5.1.3 Joint Resource Allocation Based on Dual Decomposition. In addition to the cross-layer techniques that focus on certain problems in communication, a generalized framework that governs all the aspects of communication is another open research issue. An important step in this direction of defining a unified methodology for cross-layer design is constituted by the pioneering work by Low [43] and Chiang [36], where the need for integrating various protocol layers into a coherent framework is demonstrated, to help provide a unified foundation for the analysis of the resource allocation problems and to develop systematic techniques for cross-layer design of multihop wireless networks. This can be accomplished by interpreting the parameters describing congestion as primal and dual optimization variables, while the algorithm is interpreted as a distributed primal-dual algorithm solving an implicitly defined distributed network utility maximization problem.

The problem of joint optimization of transmitted power levels and congestion window sizes has been formulated with the objective of maximizing a utility function of the source rates, and hence optimizing the network throughput [36]. The amount of bandwidth supplied to the upper layers is nonlinearly coupled to the bandwidth demanded by the congestion control through a dual variable. A quantitative framework for joint design of transport and physical layer protocols is provided and a suboptimal version of the algorithm is proposed for scalable architectures. In Ref. [44], the cross-layer design of congestion control, routing, and scheduling is jointly tackled through network utility maximization. By dual decomposition, the resource allocation problem is decomposed into the three sub-problems (i.e., congestion control, routing, and scheduling) that interact through congestion prices. Based on this decomposition, a distributed subgradient algorithm is derived, which is shown to both converge to the optimal solution and be practical with low communication overhead.

A joint source coding, routing, and channel coding problem for WSNs is formulated in Ref. [45]. The proposed resource allocation framework jointly considers the physical, network, and application layers. The joint optimization problem in the dual domain leads to separation of the different subproblems at the different layers. A primal-dual method is also proposed for a distributed solution. Also in Ref. [46] a primal-dual method is used to decompose the joint optimization of multicast routing with network coding and power allocation into two subproblems: data routing and power allocation. The dual variables play the role of coordinating the network-layer demand for bandwidth and the physical-layer supply. The research in the resource allocation problems reveal that cross-layer design may lead to a new perspective in the definition of networking functionalities. However, most of these solutions do not provide algorithms or protocols for distributed operation in WSNs. In the following sections, we summarize the solutions in cross-layer protocol design and provide guidelines for future research directions.

14.5.2 Pairwise Cross-Layer Protocols

The experience gained through both analytical studies and experimental work in WSNs reveal important interactions between different layers of the network protocol stack. As an example, the broadcast nature of the wireless channel results in significantly different flooding trees than predicted by the unit disk graph model [33]. Moreover, the interdependency between local contention and end-to-end congestion is shown to be significant for protocol operation and guidelines for protocol design are provided in Ref. [34]. In addition to the wireless channel impact and cross-layer interactions, the content of the information sent by sensor nodes is also shown to impact cross-layer design in Refs. [47] and [48]. Dense deployment of sensor nodes results in the sensor observations being highly correlated in the space domain (*spatial correlation*). Similarly, the nature of the energy-radiating physical phenomenon yields *temporal correlation* between each consecutive observation of a sensor node. The correlation between the observations of nodes can be modeled by a correlation function based on two different source models, that is, point and field sources. Based on this theory, the estimation error resulting in exploiting the correlation in the network can be calculated. This error is defined as *distortion*. The results obtained in Ref. [47] show that by using a small subset of nodes for reporting an event (e.g., 15 out of 50), the same distortion level can be achieved. Similarly, the same distortion level can be achieved by reducing the sampling rate of sensor nodes. These results reveal that significant energy savings are possible when the correlation in the content of information is exploited.

The recent findings on the cross-layer interactions have motivated the design of different cross-layer principles. We classify these principles in terms of interactions or modularity among the physical (PHY), MAC, routing, transport, and application layers.

14.5.2.1 Transport and PHY Interactions. Transport-layer functionalities, for example, congestion control and reliability management, depend on the underlying physical properties of both the sensor transceiver and the physical phenomenon. In Ref. [36], the analysis of interactions between power control and congestion control is performed and the trade-off between the layered approach and the cross-layer approach is presented. Consequently, the received signal of a node is modeled as a global and nonlinear function of all the transmission powers of the neighbor nodes. Based on this framework, a cross-layer communication protocol is proposed, where the transmission power and the transmission rate are jointly controlled. The nodes control their transmission power based on the interference of other nodes and determine the transmission rate accordingly. In a different perspective, the spatial correlation between sensor nodes is exploited in Ref. [49] with the definition of a new reliability notion: event-to-sink reliability. This notion relies on the fact that the readings of a group of sensors in an event area are spatially correlated and the reporting rate of these sensors can be collectively controlled to ensure both reliability and congestion prevention. As a

result, in the event-to-sink reliable transport (ESRT) protocol [49], the transmission rate of sensor nodes are controlled by the sink iteratively through calculations during a decision interval.

14.5.2.2 Routing and PHY Interactions. Routing protocols are also affected by the transmission power at the PHY layer as explained before. Accordingly, a cross-layer optimization of network throughput for multihop wireless networks is presented in Ref. [46]. The throughput optimization problem is split into two subproblems: multihop flow routing at the network layer and power allocation at the physical layer. Based on this solution, a CDMA/OFDM based solution is provided such that the power control and the routing are performed in a distributed manner. In Ref. [50], new forwarding strategies for geographic routing are proposed based on the results in Ref. [35]. The forwarding algorithms require the packet reception rate (PRR) of each neighbor for the determination of the next hop and construct routes accordingly. More specifically, a $PRR \times DIST$ scheme that selects the node with the maximum PRR and advancement (DIST) product is proposed. It is shown that this scheme is optimal for networks where ARQ is implemented.

14.5.2.3 MAC and PHY Interactions. The non-uniform properties of signal propagation in low-power wireless channels need to be considered in MAC protocol design. Hence, an accurate wireless channel model is required for both evaluation and design of MAC protocols. In this respect, the trade-off between energy consumption due to processing and energy consumption due to communication is important to be considered in the design of joint MAC and PHY protocols. In Ref. [51], it is concluded that single-hop communication can be more efficient when real radio models are used in a linear network topology. This result necessitates new techniques for MAC protocols because the number of potential interferers increases significantly when single-hop communication is considered. Although this is an interesting result, it is necessary to generalize these conclusions for networks with arbitrary topologies.

In addition to the characteristics of the wireless channel and the radio circuitry, the content of the information that is sent by sensor nodes is also important in MAC protocol design. The content of such information is closely related to the properties of the physical phenomenon. The spatial correlation between the information gathered by each sensor node can be exploited for energy-efficient operation. In Ref. [52], a cross-layer solution among the MAC layer, the physical phenomenon, and the application layer is proposed for WSNs, where it is shown that a sensor node can act as a representative node for several other sensor nodes. Accordingly, a distributed and spatial correlation-based collaborative medium access control (CC-MAC) protocol is developed, which constructs correlation clusters, as shown in Fig. 14.5. Note that the representative node transmits its record on behalf of the entire correlation region, while all correlation neighbors suppress their transmissions. As a result, a smaller number of communication attempts are performed, which leads to lower contention, energy consumption,

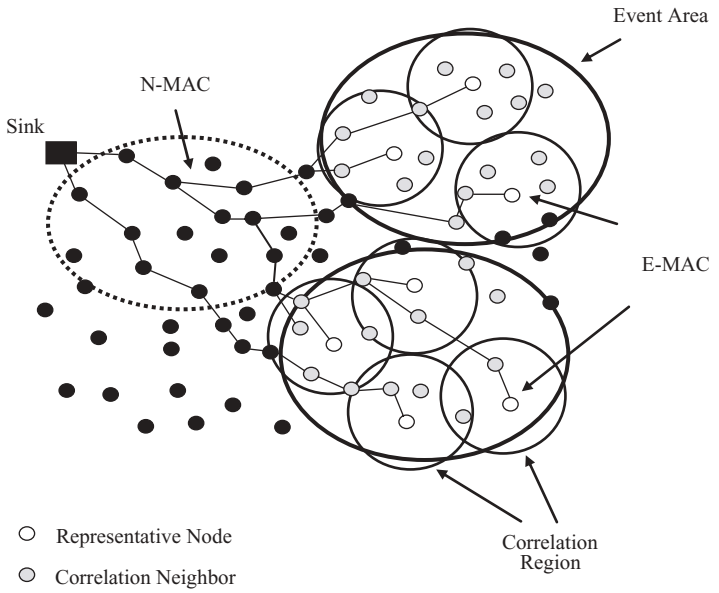


Fig. 14.5 The CC-MAC protocol and its components Event-MAC (E-MAC) and Network-MAC (N-MAC).

and latency while achieving acceptable distortion for reconstruction of the event information at the sink.

14.5.2.4 MAC and Routing Interactions. Recently, exploiting cross-layer interaction has gained much interest among the MAC and routing layers. In this context, three main approaches are emerging. On one hand, the functions related to determining the next hop in a network are closely coupled with medium access. This concept is referred to as receiver-based routing, where the next hop is chosen as a result of the contention in the neighborhood [53]. Figure 14.6 illustrates the typical receiver-based routing, where the recent findings in geographical routing and channel-aware routing techniques are coupled with medium-access procedures. When a node i has a packet to send, the nodes that are closer to the sink than the node i contend for routing. In order to provide the minimum number of hops in routing, the nodes closer to the sink are given higher priority in this contention.

Another aspect of cross-layer interaction of the MAC and routing layers is through duty-cycle operation, where a node is switched to sleep as long as it is not required to relay packets in the network. However, to prevent network disconnection, routing algorithms need to consider the duty-cycle operation at the MAC layer. As an example, for periodic traffic, where nodes form distributed on–off schedules for each flow in the network, the routes can be established such

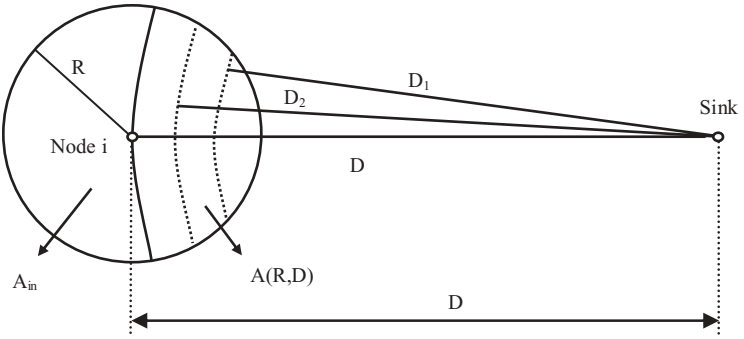


Fig. 14.6 Receiver-based routing.

that the nodes are only awake when necessary [54]. A comprehensive analysis of the trade-off between on–off schedules and the connectivity of the network show that there is a strong interdependency between these two functionalities [54].

Another approach in cross-layering MAC and routing layers is based on interference avoidance. Since WSNs are characterized by multiple flows from closely located nodes to a single sink, potential interfering routes can be established. In Ref. [55], the effect of the broadcast nature of MAC on routing is investigated to minimize the MAC interference between the routes by constructing interference-aware routes.

14.5.3 Cross-Layer Module Design

The communication protocols that focus on pairwise cross-layer interaction provide valuable insights for the development of more general cross-layer approaches that span more than two protocol layers. The ultimate goal in the design of communication protocols for WSNs is to develop a *cross-layer module* that provides the necessary functionalities of a WSN without the inefficiencies of the layered protocol stack. Although the existing solutions incorporate cross-layer interactions into protocol design, the layering concept still remains intact in these protocols. However, there is still much to be gained by rethinking the functionalities of each protocol layer and melting them into a single cross-layer module. Next, we outline our solution to cross-layer design in WSNs, which incorporates transport, routing, MAC, and physical-layer functionalities into a single cross-layer module (XLM) [56].

XLM replaces the entire traditional layered protocol architecture that has been used so far in WSNs through a novel *initiative* concept. The initiative concept constitutes the core of XLM and implicitly incorporates the intrinsic functionalities required for successful communication. A node initiates transmission by broadcasting a request-to-send (RTS) packet to advertise its neighbors that it has a packet to send. The neighbors, upon receiving the RTS packet, decide to

participate in the communication through *initiative determination*. Denoting the initiative as I, it is determined as follows:

$$I = \begin{cases} 1 & if \begin{cases} \xi_{\text{RTS}} \geq \xi_{Th} \\ \lambda_{\text{relay}} \leq \lambda_{\text{relay}}^{Th}, \\ \beta \leq \beta^{\max} \\ E_{\text{rem}} \geq E_{\text{rem}}^{\min} \end{cases} \\ 0 & \text{otherwise,} \end{cases}$$

where ξ_{RTS} is the received SNR value of the RTS packet, λ_{replay} is the rate of the packets that are relayed by a node, β is the buffer occupancy of the node, and E_{rem} is the residual energy of the node, while the terms on the right side of the inequalities indicate the associated threshold values for these parameters, respectively. The first condition ensures that reliable links be constructed for communication. The second and third conditions are used for local congestion control in XLM. The second condition prevents congestion by limiting the traffic a node can relay. The third condition ensures that the node does not experience any buffer overflow. The last condition ensures that the remaining energy of a node E_{rem} stays above a minimum value $E_{\text{rem}}^{\min}$. The cross-layer functionalities of XLM lie in these constraints that define the initiative of a node to participate in communication. Using the initiative concept, XLM performs local congestion control, hop-by-hop reliability, and distributed operation. Analytical performance evaluation and simulation experiment results show that XLM significantly improves the communication performance and outperforms the traditional layered protocol architectures in terms of both network performance and implementation complexity.

14.5.4 Precautionary Guidelines and Open Research Problems

As explained in Sections 14.5.1–14.5.3, there have been remarkable efforts on cross-layer design to develop new communication protocols. However, there is still much to be gained by rethinking the protocol functionalities of the network layers in a unified way. In fact, cross-layer design is interdisciplinary in nature and involves several research areas, for example, adaptive coding and modulation, channel modeling, traffic modeling, queuing theory, network protocol design, and optimization techniques.

Before detailing the still-open research issues, we first provide an overview of precautionary guidelines. Cross-layer design, in fact, makes the interfaces among different functionalities open and yields much more optimized solutions for resource-constrained devices. However, these results are often obtained by decreasing the architecture modularity and by making the logical separation between designers and developers looser. For these reasons, when a cross-layer solution is proposed, the system performance gain needs to be weighed against the possible longer term downfalls raised by a diminished degree of modularity.

Here, we present concerns and precautionary considerations for cross-layer design and suggest possible research directions.

- *Modularity.* In the classical layered design approach, complex problems can be easily broken into easier subproblems, which can then be solved in isolation. Conversely, a cross-layer design approach may lose the decoupling between the design and development processes, which may impair both of these concepts and slow down the innovation. Instead, it is important to keep some degree of modularity in the design of cross-layer solutions. This could be achieved by relying on functional entities, as opposed to layers in the classical design philosophy, which implement particular functions.
- *Stability.* The effect of any single design choice in cross-layer design may affect the whole system, leading to various negative consequences, for example, instability. Moreover, the fact that some interactions are not easily foreseen makes cross-layer design choices even trickier. Hence, there is a need to integrate and further develop control theory techniques to study the stability properties of system design following a cross-layer approach.
- *Robustness.* In order to preserve the robustness of a cross-layer system, techniques, for example, timescale separation and performance tracking and verification, may need to be employed in a design phase to separate interactions and verify the system performance on the fly, as suggested in Ref. [57]. Moreover, an accompanying theoretical framework may be needed to fully support cross-layer design and study its robustness properties beforehand.

Considering these precautionary guidelines, it is necessary to develop sound models to include an accurate description of the end-to-end delay that results from the interaction of the different layers. In particular, there is a need to develop mathematical models to accurately describe contention at the MAC layer. This is particularly important for the design of sensor network protocols for monitoring applications that require real-time delivery of event data [14].

Moreover, the characteristics of the physical-layer communication, for example, modulation and error control, which impact the overall resource allocation problem, should be incorporated in the cross-layer design. It is important to assess the interactions between error control, MAC, and routing to determine the best scheme for a particular application [58]. These cross-layer techniques are also exploited to design a packet size optimization framework for WSNs [59]. Furthermore, joint consideration of adaptive modulation, adaptive FEC, and scheduling would provide each user with the ability to adjust the transmission rate and achieve the desired error protection level, thus facilitating the adaptation to various channel conditions.

Last, but not least, new cross-layer network simulators need to be developed. Current discrete-event network simulators, for example, OPNET, ns-2, J-Sim, and GloMoSim, may be unsuitable for implementing a cross-layer solution because their inner structure is based on a layered architecture, and each implemented

functionality run by the simulator engine is tightly tied to this architecture. Hence, implementing a cross-layer solution in one of these simulators may turn into a nontrivial task. For this reason, there is a need for developing new software simulators that are based on this new paradigm so as to ease the development and test of cross-layer algorithmic and protocol solutions.

14.6 SUMMARY

This chapter highlighted future research trends in WSNs. First, we described the emerging area of WMSNs along with their characteristics and requirements, as well as an initial attempt for communication in such networks. Second, WSANs, which combine sensing with action, were discussed. In such networks, the coordination among different components is a major challenge and our solution in this respect was introduced. Moreover, WSNs have recently found their applications in challenging environments. Our discussion on future trends on WSNs continued with two examples of these challenging environments: underwater and underground. The intrinsic properties of these environments were summarized and open research issues were highlighted for the provision of efficient communication. Finally, we reviewed and classified literature on cross-layer design methodologies and protocols for WSNs. We overviewed the communication protocols proposed for WSNs that focus on cross-layer design techniques. We classified these techniques based on the network layers they aim to replace in the classical Open Systems Interconnection (OSI) network stack. Furthermore, we discussed systematic methodologies for the design of cross-layer solutions for sensor networks as resource allocation problems in the framework of nonlinear optimization. We outlined open research issues in the development of cross-layer methodologies for sensor networks and discussed possible research directions. The interesting new application areas of WSNs and the many challenges in these areas are envisioned to drive the research in WSNs for the future.

ACKNOWLEDGMENTS

The authors are in debt to their previous advisor, Dr. Ian F. Akyildiz, for his continuous guidance during the pursuing of their Ph.D. degrees at Georgia Institute of Technology, and also to their previous lab members, Kaushik R. Chowdhury and Eric Stuntebeck, for some of the work done together and included in this chapter.

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, Y., and E. Cayirci, "Wireless sensor networks: A survey", *Computer Networks (Elsevier) Journal*, vol. 38, no. 4, Mar. 2002, pp. 393–422.

[2] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, "A survey on wireless multimedia sensor networks", *Computer Networks (Elsevier)*, vol. 51, no. 4, Mar. 2007, pp. 921–960.

[3] F. Hu and S. Kumar, "QoS considerations for wireless sensor networks in telemedicine", in *Proceedings of 2003 Intl. Conf. on Internet Multimedia Management Systems*, Orlando, Florida, Sept. 2003, pp. 323–334.

[4] B. Girod, A. Aaron, S. Rane, and D. Rebollo-Monedero, "Distributed video coding", *Proceedings of the IEEE*, vol. 93, no.1, Jan. 2005, pp. 71–83.

[5] J. Reed, *Introduction to Ultra Wideband Communication Systems*, Prentice Hall, Englewood Cliffs, NJ, June 2005.

[6] Revision of Part 15 of the Commission's Rules Regarding Ultra-Wideband Transmission Systems. First note and Order, Federal Communications Commission, ET-Docket 98-153, Adopted Feb. 14, 2002, released Apr. 22, 2002.

[7] M. Z. Win and R. A. Scholtz, "Ultra-wide bandwidth time-hopping spread-spectrum impulse radio for wireless multiple-access communication", *IEEE Transactions on Communications*, vol. 48, no.4, Apr. 2000, pp. 679–689.

[8] Anuj Batra et al., "Multi-band OFDM physical layer proposal for IEEE 802.15 task group 3a. IEEE P802.15", Working Group for Wireless Personal Area Networks (WPANs), Mar. 2004.

[9] T. Melodia and I. F. Akyildiz, "Cross layer QoS support for UWB-based wireless multimedia sensor networks", in *Proceedings of 2008 IEEE INFOCOM'08, Mini-Conference*, Phoenix, AZ, Apr. 2008.

[10] R. Merz, J. Widmer, J.-Y. L. Boudec, and B. Radunovic, "A joint PHY/MAC architecture for low-radiated power THUWB wireless ad-hoc networks", *Wireless Communications and Mobile Computing Journal*, vol. 5, no. 5, July 2005, pp. 567–580.

[11] T. Melodia, D. Pompili, and I. F. Akyildiz, "On the interdependence of distributed topology control and geographical routing in ad hoc and sensor networks", *Journal of Selected Areas in Communications*, vol. 23, no. 2, Mar. 2005, pp. 520–532.

[12] L. Yang and G. B. Giannakis, "Ultra-wideband communications: An idea whose time has come", *IEEE Signal Processing Magazine*, vol. 3, no. 6, Nov. 2004, pp. 26–54.

[13] S. Gezici, Z. Tian, G. B. Giannakis, H. Kobayashi, A. F. Molisch, H. V. Poor, and Z. Sahinoglu, "Localization via ultra-wideband radios", *IEEE Signal Processing Magazine*, vol. 22, no. 4, July 2005, pp. 70–84.

[14] I. F. Akyildiz and I. H. Kasimoglu, "Wireless sensor and actor networks: Research challenges", *Ad Hoc Networks (Elsevier)*, vol. 2, no. 4, Oct. 2004, pp. 351–367.

[15] T. Melodia, D. Pompili, V. C. Gungor, and I. F. Akyildiz, "Communication and coordination in wireless sensor and actor networks", *IEEE Transactions on Mobile Computing*, vol. 6, no. 10, Oct. 2007, pp. 1116–1129.

[16] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Englewood Cliffs, NJ: Prentice Hall, Feb. 1993.

[17] R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming*. Duxbury Press, Pacific Grove, CA, 2002.

[18] CPLEX solver, available at http://www.cplex.com.

[19] The J-Sim Simulator, available at http://www.j-sim.org.

[20] B. P. Gerkey and M. J. Mataric, "A formal analysis and taxonomy of task allocation in multi-robot systems", *The International Journal of Robotics Research*, vol. 23, no. 9, Sept. 2004, pp. 939–954.

[21] J. Czyzyk, M. Mesnier, and J. Mor, "The NEOS server", *IEEE Journal on Computational Science and Engineering*, vol. 5, no. 3, July–Sept. 1998, pp. 68–75.

[22] I. F. Akyildiz, D. Pompili, and T. Melodia, "Underwater acoustic sensor networks: research challenges", *Ad Hoc Networks (Elsevier)*, vol. 3, no. 3, May 2005, pp. 257–273.

[23] J. G. Proakis, J. A. Rice, E. M. Sozer, M. Stojanovic, "Shallow water acoustic networks", in: J. G. Proakis (Ed.), *Encyclopedia of Telecommunications*, John Wiley & Sons, Hoboken, NJ, 2003.

[24] J. Catipovic, "Performance limitations in underwater acoustic telemetry", *IEEE Journal of Oceanic Engineering*, vol. 15, no. 3, July 1990, pp. 205–216.

[25] V. Ravelomanana, "Extremal properties of three-dimensional sensor networks with applications", *IEEE Transactions on Mobile Computing*, vol. 3, no. 3, July–Aug 2004, pp. 246–257.

[26] D. Pompili, T. Melodia, and I. F. Akyildiz, "Deployment analysis in underwater acoustic wireless sensor networks", in *Proceedings of ACM International Workshop on UnderWater Networks (WUWNet'06)*, Los Angeles, CA, Sept. 2006, pp. 48–55.

[27] I. F. Akyildiz and E. Stuntebeck, "Wireless underground sensor networks: Research challenges", *Ad Hoc Networks Journal (Elsevier)*, vol. 4, no. 6, Nov. 2006, pp. 669–686.

[28] "Campbell scientific soil science brochure". Available at http://www.campbellsci.com/documents/apsheet/soil.pdf.

[29] D. Daniels, *Surface-Penetrating Radar*. The Institution of Electrical Engineers, London, UK, Aug. 1996.

[30] "Micaz datasheet," Crossbow Technology Inc., available at http://www.xbow.com.

[31] E. Stuntebeck, D. Pompili, and T. Melodia, "Underground wireless sensor networks using commodity terrestrial motes", in *Proceedings of IEEE SECON'06*, Reston, VA, Sept. 2006, pp. 112–114.

[32] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks", in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (Sensys)*, New York, NY, Nov. 2004, pp. 95–107.

[33] D. Ganesan, et al., "An empirical study of epidemic algorithms in large scale multihop wireless networks", *Technical Report, Intel Research*, Berkeley, CA, 2002.

[34] M. C. Vuran, V. B. Gungor, and O. B. Akan, "On the interdependency of congestion and contention in wireless sensor networks", in *Proceedings of SENMETRICS'05*, San Diego, CA, July 2005, pp. 136–147.

[35] M. Zuniga and B. Krishnamachari, "Analyzing the transitional region in low power wireless links", in *Proceedings of IEEE SECON'04*, Santa Clara, CA, Oct. 2004, pp. 517–526.

[36] M. Chiang, "Balancing transport and physical layers in wireless multihop networks: jointly optimal congestion control and power control", *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 1, Jan. 2005, pp. 104–116.

[37] T. ElBatt and A. Ephremides, "Joint scheduling and power control for wireless ad-hoc networks", in *Proceedings of IEEE INFOCOM'02*, New York, NY, June 2002, pp. 976–984.

[38] U. C. Kozat, I. Koutsopoulos, and L. Tassiulas, "A framework for cross-layer design of energy-efficient communication with QoS provisioning in multi-hop wireless networks", in *Proceedings of INFOCOM'04*, Hong Kong, China, Mar. 2004, pp. 1446–1456.

[39] R. Cruz and A. Santhanam, "Optimal routing, link scheduling and power control in multi-hop wireless networks", in *Proceedings of IEEE INFOCOM'03*, San Francisco, CA, Mar. 2003, pp. 702–711.

[40] M. Kodialam and T. Nanagopal, "Characterizing achievable rates in multi-hop wireless networks: the joint routing and scheduling problem", in *Proceedings of ACM Mobicom'03*, San Diego, CA, Aug. 2003, pp. 868–880.

[41] R. Madan, S. Cui, S. Lall, and A. Goldsmith, "Cross-layer design for lifetime maximization in interference-limited wireless sensor networks", in *Proceedings of IEEE INFOCOM'05*, Miami, FL, Mar. 2005, pp. 1964–1975.

[42] B. Radunovic and J. Y. Le Boudec, "Optimal power control, scheduling, and routing in UWB networks", *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 7, Oct. 2004, pp. 1252–1270.

[43] S. H. Low, "A duality model of TCP and queue management algorithms", *IEEE/ACM Transactions on Networking*, vol. 11, no. 4, Aug. 2003, pp. 525–526.

[44] L. Chen, S. H. Low, M. Chiang, and J. C. Doyle, "Cross-layer congestion control, routing, and scheduling for wireless ad hoc networks", in *Proceedings of IEEE INFOCOM'06*, Barcelona, Spain, Apr. 2006, pp. 1–13.

[45] W. Yu and Jun Yuan, "Joint source coding, routing, and resource allocation for wireless sensor networks", in *Proceedings of IEEE ICC'05*, Seoul, Korea, May 2005, pp. 737–741.

[46] J. Yuan, Z. Li, W. Yu, and B. Li, "A Cross-Layer optimization framework for multicast in multi-hop wireless networks wireless internet", in *Proceedings of WICON'05*, Visegrad-Budapest, Hungary, July 2005, pp. 47–54.

[47] M. C. Vuran, O. B. Akan, and I. F. Akyildiz, "Spatio-temporal correlation: Theory and applications for wireless sensor networks", *Computer Networks Journal (Elsevier)*, vol. 45, no. 3, June 2004, pp. 245–261.

[48] M. C. Vuran and O. B. Akan, "Spatio-temporal characteristics of point and field sources in wireless sensor networks", in *Proceedings of IEEE ICC'06*, Istanbul, Turkey, June 2006, pp. 234–239.

[49] O. B. Akan and I. F. Akyildiz, "Event-to-sink reliable transport in wireless sensor networks", *IEEE/ACM Transactions on Networking*, vol. 5, no. 13, *Oct.* 2005, pp. 1003–1017.

[50] K. Seada, M. Zuniga, A. Helmy, and B. Krishnamachari, "Energy-efficient forwarding strategies for geographic routing in lossy wireless sensor networks", in *Proceedings of ACM Sensys'04*, Baltimore, MD, Nov. 2004, pp. 108–121.

[51] J. Haapola, Z. Shelby, C. Pomalaza-Raez, and P. Mahonen, "Cross-layer energy analysis of multi-hop wireless sensor networks", in *Proceedings of EWSN'05*, Istanbul, Turkey, Feb. 2005, pp. 33–44.

[52] M. C. Vuran and I. F. Akyildiz, "Spatial correlation-based collaborative medium access control in wireless sensor networks", *IEEE/ACM Transactions on Networking*, vol. 14, no. 2, Apr. 2006, pp. 316–329.

[53] M. Zorzi and R. Rao, "Geographic random forwarding (GeRaF) for ad hoc and sensor networks: multihop performance", *IEEE Transactions on Mobile Computing*, vol. 2, no. 4, Oct.–Dec. 2003, pp. 337–348.

[54] M. L. Sichitiu, "Cross-layer scheduling for power efficiency in wireless sensor networks", in *Proceedings of IEEE INFOCOM'04*, Hong Kong, China, Mar. 2004, pp.1740–1750.

[55] Y. Fang and B. McDonald, "Dynamic codeword routing (DCR): A cross-layer approach for performance enhancement of general multi-hop wireless routing", in *Proceedings of IEEE SECON'04*, Santa Clara, CA, Oct. 2004, pp. 255–263.

[56] I. F. Akyildiz, M. C. Vuran, and O. B. Akan, "A Cross-layer protocol for wireless sensor networks", in *Proceedings of CISS'06*, Princeton, NJ, Mar. 2006, pp.1102–1107.

[57] V. Kawadia and P. R. Kumar, "A cautionary perspective on cross-layer design", *IEEE Wireless Communications Magazine*, vol. 12, no. 1, Feb. 2005, pp. 3–11.

[58] M. C. Vuran and I. F. Akyildiz, "Cross-layer analysis of error control in wireless sensor networks", in *Proceedings of IEEE SECON'06*, Reston, VA, Sept. 2006, pp. 585–594.

[59] M. C. Vuran and I. F. Akyildiz, "Cross-layer packet size optimization for wireless terrestrial, underwater, and underground sensor networks", in *Proceedings of IEEE INFOCOM'08*, Phoenix, AZ, April 2008.

INDEX

Wireless Sensor Networks: A Networking Perspective, Edited by Jun Zheng and Abbas Jamalipour