WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. 2009; **9**:303–311
Published online 19 March 2008 in Wiley InterScience
(www.interscience.wiley.com) DOI: 10.1002/wcm.613

Mobility handling in MAC for wireless ad hoc networks

Anjali Raja and Xiao Su*,†
Department of Computer Engineering, San Jose State University, San Jose, CA 95192, U.S.A.

Summary

Wireless Sensor Networks (WSNs) are becoming popular in a wide variety of applications. In a WSN, sensor nodes can be static or mobile, depending on applications. Handling mobility in a sensor network can pose interesting challenges in sensor protocol design, and special algorithms are needed to adapt to mobility in the network. In this paper, we survey the current state of art in handling mobility in sensor networks. We first study how mobility affects the design of different layers in the sensor network architecture, and then take a closer look at mobility handling in the MAC layer. We also describe the results of simulations that we conducted to compare and analyze existing mobility-aware MAC protocols. Finally, we discuss open issues in dealing with mobility in sensor networks. Copyright © 2008 John Wiley & Sons, Ltd.

KEY WORDS: mobility model; mobility pattern; wireless sensor network; medium access control

1. Introduction

A wireless sensor network (WSN) is an ad hoc network of autonomous low-powered sensors that are spatially distributed and communicate wirelessly to cooperatively achieve a task. The past few years have seen these wireless sensors being used for diverse applications like environmental monitoring, battlefield surveillance, and traffic control.

Figure 1 shows a typical deployment of sensor nodes in data gathering applications. A sensor node consists of the following components: a microcontroller, a sensor, a transceiver, and a power source. The data gathered by the sensor devices are processed by the microcontroller and sent to a gateway for further analysis using the wireless transceiver. The power source for a sensor is typically a battery. These sensors are usually deployed to monitor physical parameters in harsh environments. Once deployed, it should operate without human intervention. As recharging batteries is difficult under such deployments, energy is the scarcest resource in sensor networks [1].

Besides being energy constrained, nodes in sensor networks can be mobile due to several reasons. After deployment, nodes can fail due to hardware failure or running out of battery, causing the nodes to leave the network. New nodes need be added to the network to replace these failed nodes. In Reference [2], such movements are classified as weak mobility. In contrast, strong mobility is characterized by sensor nodes physically moving through the network, because of the movement of the medium (wind, water) or the nodes themselves.

With the advancement in sensor technologies, strong mobility plays an important part in an increasing number of applications. In patient monitoring

*Correspondence to: Xiao Su, Department of Computer Engineering, San Jose State University, San Jose, CA 95192, U.S.A.
†E-mail: xsu@email.sjsu.edu

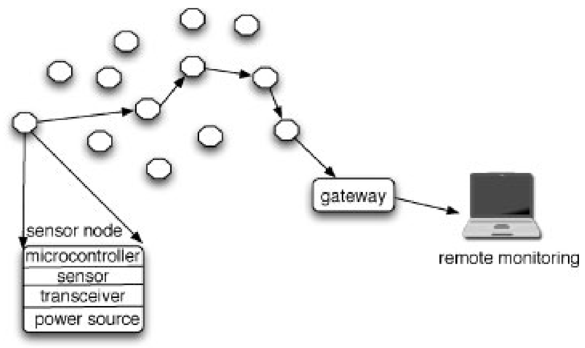


Fig. 1. Typical WSN deployment in data gathering applications.

applications, sensor nodes are attached on patients and follow their mobility pattern as they walk around. In disaster recovery applications, workers (or soldiers) are equipped with sensors, as they walk around the recovery scenes [3]. In both applications, sensor nodes are attached to moving people or other targets. Applications, such as battlefield surveillance, involve a different type of mobility, in which mobile sensor nodes are grouped in platforms that roam around the surveillance area to collect information. RoboMOTE [4] is an example of mobility in sensor platform. A third type of mobility called parasitic mobility is also gaining popularity [3]. Here nodes are designed to use local navigational intelligence to selectively attach and detach from external mobile hosts. Table I captures the mobility types and applications in a snapshot.

It is important to handle both types of mobility, weak and strong, in sensor network design in all layers of sensor network architecture for correct and efficient operation. The requirement to handle mobility adds another dimension to sensor network protocols, in addition to conservation of energy and computation resources.

In this paper, we survey the current state of art in handling mobility in sensor networks. Section 2 overviews various mechanisms to handle mobility in different layers of sensor network architecture. In Section 3, we discuss different mobility patterns and mobility models used in WSNs. Section 4 presents how to obtain mobility information through mobility estimation algorithms. We then move on to a comparative study of protocols in handling mobility at the MAC layer in Section 5. Finally, we discuss open issues in Section 7 and conclude the paper in Section 8.

Table I. Mobility in sensor applications.

Applications	Mobility
Patient monitoring disaster recovery	People wearing static sensors move around in an area
Platform mobility in battlefield surveillance	Mobile sensors to improve network coverage
Parasitic mobility environmental monitoring	Sensors intelligently attach or detach from mobile hosts

2. Handling Mobility in Sensor Networks

2.1. Common Issues Caused by Mobility

Mobility in sensor networks brings the following interesting issues in the design of protocols in different layers:

- In a mobile network, the position of sensor nodes changes over time. A mobility-aware protocol needs to estimate the position of each node in the network at any point in time.
- The movement of nodes can cause frame errors in the network. A mobile protocol needs to handle these frame errors and re-transmit frames where necessary.
- In the case of routing protocols, node mobility can result in route changes, which may lead to delivery failures if not propagated in time. To alleviate the problem, mobility information needs to be used in route calculation. To this end, neighbor table and route maintenance should adapt to the mobility in sensor networks [3].
- In the case of schedule-based MAC protocols, a neighborhood inconsistency of two-hop neighbor information can occur, when mobile nodes enter or leave the neighborhood. This leads to schedule inconsistencies. A mobility-aware MAC protocol needs to utilize the mobility information of nodes and determines which nodes enter or leave the neighborhood to eliminate these inconsistencies [3].

2.2. Current Solutions to Handle Mobility

The lack of overall sensor network architecture has plagued sensor network design from the beginning [5]. There have been a number of protocol proposals and subsystems developed for sensor networks, which have proven difficult to combine in a general way to build usable systems.

In general, the solutions proposed for sensor networks resemble a traditional networking stack with

Table II. Mobility handling in different network layers.

Network layer	Impact of mobility
Application	Leveraging mobility for easier data collection
Routing	Using mobility information to calculate the next hop
MAC	Adjusting frame time to reduce frame errors; allowing nodes to make faster connections on leaving a network

several layers including application, network, and MAC layers. However, the boundary between these layers is blurred in those solutions, where a vertically integrated design has components that work together. Some researchers have also produced monolithic solutions cutting across these boundaries as in Reference [3].

In some sensor network protocols, mobility is explicitly handled at a specific layer. For instance, in application layer, sink mobility is used for easier data collection. In this case, instead of all the nodes sending updates to the sink, the sink moves among the nodes to collect data as in Reference [6]. They have proposed the mobility patterns and data collection strategies that can be used to increase the energy efficiency in such an environment.

An example network layer protocol described in Reference [7] addresses node mobility and energy efficiency in a multi-path routing protocol. To determine the next hop relay node, the protocol takes all the following factors into account: the remaining battery capacity, the degree of mobility of a node, and the distance to the destination node from all the neighbors.

A MAC protocol, in particular, has a tight control over energy efficiency, delay, mobility, and other metrics of the network [1]. MMAC [2] and MS-MAC [8] are examples of MAC protocols that explicitly handle mobility in the network. MS-MAC uses the mobility information of the neighbors to help a mobile node form connections quickly while moving across virtual clusters. MMAC, on the other hand, adjusts the frame time and random access time according to the mobility of nodes. Table II captures the mobility handling at various layers of sensor network stack.

In general, the mobility information needed by the protocols includes the position of a node and the mobility information of its neighbors for better mobility detection and adaption. Therefore, the mobility information has to be periodically disseminated to neighboring nodes, incurring overhead in the form of control messages in each layer.

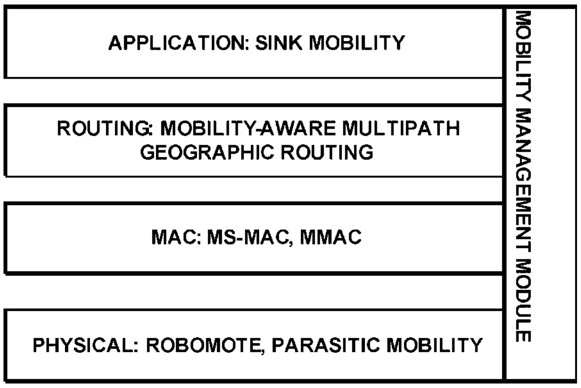

APPLICATION: SINK MOBILITY
ROUTING: MOBILITY-AWARE MULTIPATH GEOGRAPHIC ROUTING
MAC: MS-MAC, MMAC
PHYSICAL: ROBOMOTE, PARASITIC MOBILITY
MOBILITY MANAGEMENT MODULE

Fig. 2. Examples of mobility handling in various layers of sensor network architecture.

Instead of handling mobility individually at each layer, some researchers advocate an integrated mobility management solution common to all the layers. They propose to set up a common storage database of mobility information, accessible by all the layers, from applications to network layer, and to MAC layer. The mobility management database represents a cross-layer storage of mobility information [3].

Figure 2 summarizes example mechanisms of handling mobility in the network architecture.

3. Mobility Patterns and Mobility Models

In order to evaluate the sensor network algorithms that handle mobility, we need to understand different types of mobility in the network. We describe common analytical models that capture mobility patterns of sensor nodes for this purpose.

In the literature, mobility pattern and mobility model are sometimes used interchangeably. However, these are different concepts. A mobility pattern is the movement pattern of real-life objects such as people or vehicles. A mobility model is a formal mathematical model generalizing these mobility patterns. In this section, we describe representative mobility patterns and models that are relevant in sensor networks design.

3.1. Mobility Patterns

A mobility pattern can be characterized by properties such as dimension, limitation, group behavior, and predictability. The three main patterns that apply to various sensor applications are: pedestrian mobility, vehicular mobility, and dynamic medium mobility.

Pedestrian mobility deals with the motion patterns of people such as in disaster recovery or patient monitoring applications, where people walk around wearing static sensors. This pattern is manifested by its limited speed, obstacle avoidance, and its typical chaotic nature. The mobility is two-dimensional, and may or may not show a group behavior [9].

Vehicular mobility, in the case of sensor networks, deals with the motion of vehicles equipped with sensors. This type of mobility allows for higher speed. It may demonstrate group behavior, when vehicles communicate with each other about traffic conditions and other information. The movement is largely one-dimensional, because of the directional nature of vehicular motion to minimize chances of collision [9].

Dynamic medium mobility occurs when sensors move through a medium such as wind, water, and fluids. The mobility can be one-, two-, or three-dimensional depending on the type of the medium. The motion itself greatly varies depending upon the medium. For instance, with wind as the medium, the type and direction of the motion depend on the direction and speed of the wind itself.

3.2. Mobility Models

In general, mobility models fall into two categories: trace-based and synthetic models. In trace-based models, real-life mobility patterns are collected by observing systems with a large number of participants over a long duration. However, since large scale deployments of WSNs are still a work in progress, obtaining such traces becomes a challenge. Even if this data is gathered somehow, the sheer volume of the collected data renders such modeling unscalable. This is precisely where synthetic models come into play. Synthetic models attempt to realistically represent the behaviors of mobile nodes in a network. However, they may be less accurate than real-life traces and need to be well understood before being used in sensor designs.

There are three types of synthetic mobility models: random, deterministic, and hybrid. Random models assume an arbitrary movement without any constraints. Deterministic models assume a pre-defined movement path or a path with constraints. Hybrid models combine the above two approaches.

Most WSN simulations are based on random models [9]. There are three types of random models. In the Random Walk Mobility Model, nodes move from one location to another by choosing a random direction and a random speed within a specified range. The movement occurs either for a constant distance or for a constant amount of time. A new direction and speed are chosen, once the node reaches the destination. The Random Waypoint Mobility Model differs from the random walk model in that it chooses a pause time before any change of speed or direction. It is a widely used mobility model in designing sensor protocols. A related random model, Gauss–Markov Mobility Model, uses a tuning parameter to vary the degree of randomness in the mobility models.

Table III. Mobility patterns addressed in different mobility models.

Mobility pattern	Mobility models
Pedestrian	Random models Random walk, random waypoint, Gauss–Markov
Vehicular	Deterministic models City selection, Freeway and Manhattan
Medium	Random models Random walk, random waypoint, Gauss–Markov

Deterministic models are often used to model vehicular mobility, where a graph represents a road topology and nodes represent vehicles that follow casual paths over the graph [10]. For instance, the City Section Mobility Model [11] constrains the movement of nodes on a grid road topology, where the edges of the graph represents bi-directional, single-lane roads. Nodes can randomly choose one of the intersections as the destination and move toward it with a constant speed. Similarly, the Manhattan Model [11] also uses a grid road topology. It employs a probabilistic approach in the selection of node movements at each intersection: a node can decide to continue moving in the same direction with a probability of Ω, turn left with a probability of λ, turn right with a probability of β. The Freeway Model [11] uses a freeway of bi-directional multi-lane roads and restricts the vehicle movement to the lane that it is traveling on.

An example of a hybrid model can be found in Reference [12] that describes a proposal for combining Random Waypoint Model and Manhattan Model.

There are research efforts toward defining more realistic mobility models [13]. In these models, obstacles are introduced that limit node movements as well as wireless transmissions.

In Table III, we summarize the mobility models that can be used to characterize each mobility pattern.

4. Mobility Estimation

Mobility patterns and models are very useful in designing and evaluating sensor network algorithms under mobility conditions. In addition, mobility estimation algorithms are needed by sensor network protocols to estimate the level or direction of node mobility. Let us discuss a few mobility estimation algorithms commonly used in sensor network protocols.

The accuracy of mobility estimation depends on the accuracy of the underlying localization mechanism [14]. One example of a localization mechanism is an autoregressive model that predicts the mobility state of a node from its past mobility states. For instance, the AR-1 [15] model defines the mobility state of a node at time *t* in terms of the position, velocity, and acceleration of the node at that time. Future mobility states are predicted from the current state.

Other mechanisms, like the AR-3 model [16], give more accurate mobility state information, but they are computationally intensive, and not suitable for real-world sensor devices.

Some lightweight mobility estimation mechanisms use the received signal strength of messages from neighboring nodes. MS-MAC uses such a mechanism to determine the relative mobility in the neighborhood. The actual speed of the node itself is not important here, as it only needs to decide whether there is a change in the signal strength received at a different time. Similarly, In Reference [17], received signal strength is used to calculate the mobility level (more accurately), and each node is assigned a mobility index. The calculation of the mobility index of a node also depends on the mobility indices of the neighbor nodes.

For accurate mobility estimations, a node often needs to announce its mobility state at regular intervals and collect the mobility state of neighbors. Most of the algorithms use a portion of the control message to disseminate this information. A small overhead is thus incurred to gain accuracy in mobility estimations.

In general, mobility estimation algorithms are independent of any specific mobility model used in the design. The mobility of node is usually obtained through either measuring received signal strength, or predicting from its own historical values, or calculating from the values of the neighboring nodes.

5. Mobility at the MAC Layer

Mobility poses a great challenge to all protocol layers, especially for protocols in the MAC layer, which are mainly responsible for packet scheduling, transmission, collision avoidance, and resolution. The above tasks demand high energy consumption. Therefore, earlier MAC protocols have focused on energy efficiency as the main design objective. Handling mobility in this energy-restricted process involves careful trade-off in energy efficiency, throughput, and robustness under mobility. In this section, we discuss two protocols that handle mobility explicitly at the MAC layer, analyze, and compare their performance under mobile scenarios.

5.1. MS-MAC

MS-MAC [8] is an improved version of SMAC [18] to handle mobility in the network. SMAC or Sensor MAC is an energy-efficient MAC protocol for WSNs, which adopts periodic sleep schedules for nodes to conserve energy. The nodes synchronize their sleep schedules using periodic control messages called SYNC messages.

MS-MAC uses a simple mobility estimation algorithm to estimate the mobility in a neighborhood. In MS-MAC, the mobility of nodes is detected by the change in received signal level of the SYNC message from the neighbors. The nodes store the received signal strength values of the SYNC messages received from each neighbor. If the change in the received signal level exceeds a minimum threshold, the nodes assume that there is a movement in the neighboring node or themselves. From the change in received signal level, a node further detects whether its neighbor is moving and at what relative speed it is moving. The SYNC message is modified to include this mobility information, in addition to transmission schedules. If there are multiple mobile nodes in the neighborhood, the maximum speed is included in the SYNC notification. Nodes that gather this mobility information forms an active area around the mobile node, and run the synchronization algorithm more often than the default value. This lets the mobile node expedite the connection process in a new cluster without losing all the neighbors and thereby the connectivity.

The mobility handling algorithm in MS-MAC is only concerned about relative mobility in a network, so that an active zone can be formed around the mobile node. It does not concern itself with the actual speeds of each mobile node. The approximate values from the received signal level changes assist in detecting mobility and estimating the frequency with which to run the mobility handling algorithm.

One disadvantage of running the synchronization too often is the higher energy usage. The MS-MAC

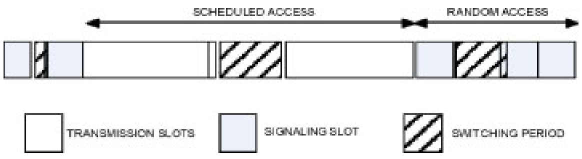


Fig. 3. TRAMA frame timing.

algorithm is designed to make the connections for mobile nodes faster, thus trading off higher energy consumption by a smaller set of nodes around the mobile node for the period of time it takes the mobile node to setup the connections in a new cluster.

5.2. MMAC

MMAC [2] is an improvement over a schedule-based protocol called TRAMA [19] by adding a mobility adaptive algorithm. As illustrated in Figure 3, TRAMA divides frame time into two parts: a random access period and a schedule access period. The random access period is used to collect neighbor information. Each node uses an adaptive election algorithm to determine the slot it can use to transmit packets. The schedule access period is then used to announce the schedule and perform the actual data transmission.

One of the major issues caused by mobility in such schedule-based protocol is that the two-hop topology information is inconsistent. In addition, with the fixed frame time, the mobile node has to wait longer to join the network. MMAC has a mobility adaptive algorithm, which addresses both problems by adjusting the frame time according to the mobility in the network (see Figure 4). The protocol handles both weak mobility (i.e., node joins, departures, and failures) and strong mobility where nodes physically move through the network.

MMAC uses the AR-1 estimation model to predict each node's mobility in the network. The basic premise is that depending on the number of neighbors expected to leave a neighborhood of a node, the frame time needs to be reduced and *vice versa*. The nodes that are expected to leave the neighborhood in the next frame time are not included in the schedule. A node needs to know its own mobility as well as its neighbor's mobility in order to adjust the frame time. The signal header and data header of MAC packets are suitably modified to include the predicted mobility state information of a node. At the start of each frame, a node calculates the next frame size as the average of all mobility states.

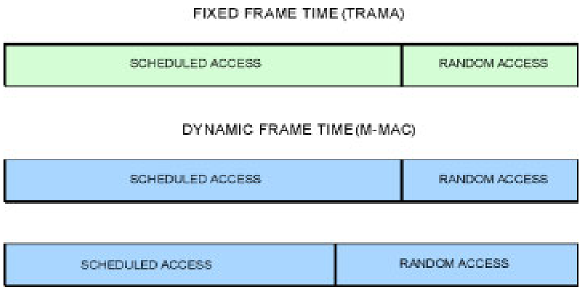


Fig. 4. TRAMA and MMAC frame times.

5.3. Summary of Mobility Handling in MS-MAC and MMAC

MS-MAC is a representative contention-based protocol with mobility handling, while MMAC is a schedule-based protocol enhanced with mobility support.

The MS-MAC protocol allows mobile nodes to make quicker connections, when crossing the boundary of a virtual cluster. It achieves this by having the neighbor nodes of a mobile node to stay awake for a longer time. The higher energy usage in that case is traded to avoid the loss of connectivity for a mobile node.

On the other hand, MMAC has a mobility adaptive algorithm that adjusts the frame time to handle mobility-related errors, in addition to allowing quicker connections for mobile nodes. The disadvantage of MMAC is the highly complex scheduling algorithm to calculate the transmitter of each slot in a frame time. The control overhead is higher because of the explicit transmission of scheduling packet. The duty cycle is also high because of the random access period and increased collisions due to mobility.

We summarize the comparisons of MS-MAC and MMAC in Table IV.

6. Comparison and Performance Analysis

We performed simulations to compare the performance of the following protocols: SMAC, MS-MAC, TRAMA, and MMAC.

6.1. Simulation Setup

We used the MAC simulator for our simulations. The MAC simulator is based on OMNeT++ discrete event simulator [20] and the energy model of the EYES prototype [21] of wireless sensor nodes using RFM TR1001 radio. We used a network topology consisting of 50 nodes uniformly placed over a $500 \times 500\,\text{m}^2$ network area. The transmission range of each node is

Table IV. Design highlights of MS-MAC and MMAC.

	MS-MAC	MMAC
Medium access method	Contention based	Schedule based
Mobility handling	Mobility handling algorithms allows a new mobile node to make quicker connections by making its neighbors stay awake for longer time	Mobility handling algorithm ensures that the nodes that will leave/enter the neighborhood in the near future are not included in the schedule. It also adjusts the frame time according to the level of mobility
Mobility estimation	Based on received signal strength of messages from neighbors	AR-1 model
Strength	Mobile nodes can join a virtual cluster fast	Neighborhood information is consistent; and its frame time is adjusted with mobility
Weakness	Energy consumption is high in the event of mobility	Scheduling is complex, and there is an overhead in sending mobility control packets

100 m. Each simulation lasts 500 s of simulated time, and the results are averaged over 10 runs with different random seeds. All the protocols are evaluated for the following performance metrics:

- average energy consumption,
- packet delivery ratio,
- latency.

Figure 5 illustrates the effects of mobility in the network. There are some mobile nodes in the network, and the speeds of the nodes vary. In Figure 5, we observe that, when mobility is increased, MMAC shows the most adaptivity among the four protocols: energy consumption values not increasing sharply (see Figure 5(a)), delivery ratios not decreasing fast (see Figure 5(b)), and the average latency values not building up quickly (see Figure 5(c)). This is because it adjusts the random access slots, schedule access slots, and the frame time, in response to mobility. By having frequent random access time, MMAC allows nodes to join the network quickly. On the other hand, we can see that TRAMA does not show this adaptivity: when the mobility increases, the energy consumption and latency values increase faster, and delivery ratio decreases more quickly.

In contention-based protocols like SMAC and MS-MAC, MS-MAC handles mobility explicitly. We see that as mobility increases, the energy consumption values go up for MS-MAC. This is because the nodes in the neighborhood of the mobile nodes stay awake for a longer time in order to let the mobile node join the network quickly. MS-MAC thus trades high energy for avoiding broken connections in the event of mobility. Among the four protocols, SMAC fares the worst when mobility is introduced.

Schedule-based protocols (MMAC and TRAMA) enjoy lower energy consumption and higher packet delivery ratios, at the expense of higher latency. The explicit calculation of transmission schedule allows nodes to sleep when they do not have anything to transmit, thus saving energy. The schedule also prevents collision errors, thereby increasing the delivery ratios. Contention-based protocols (SMAC and MS-MAC) have lower latency values because sensor nodes can make use of the next available slot for data transmissions rather than having to wait for a scheduled slot.

Figure 6 shows the effect of network dynamics on the performance of the protocols. We varied the mobility level by changing the number of mobile nodes in the network. The ratio of mobile nodes to total nodes

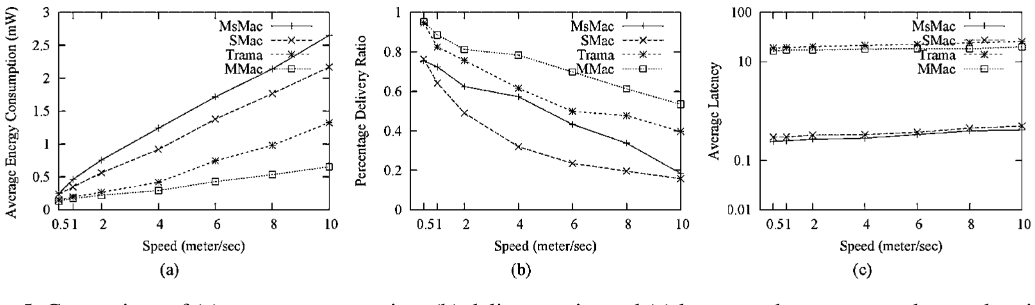


Fig. 5. Comparison of (a) energy consumption, (b) delivery ratio, and (c) latency, when sensor node speed varies.

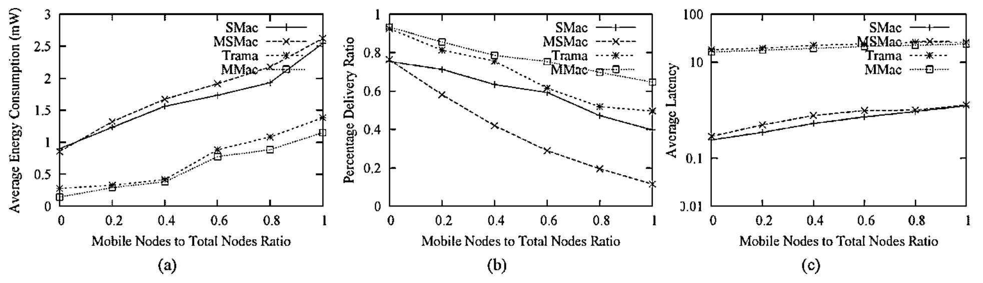


Fig. 6. Comparison of (a) energy consumption, (b) delivery ratio, and (c) latency, when sensor node dynamics change.

is plotted against average energy consumption, sleep ratio, delivery ratio, and latency, respectively.

In general, energy consumption is seen to increase with the number of mobile nodes in the network. The increase is prominent in contention-based protocols like SMAC and MS-MAC. As mobility increases, chances of contention, frame errors, and retransmissions increase. MS-MAC has the worst performance due to the increased frequency of resynchronization in the event of mobility. For schedule-based protocols like TRAMA and MMAC, the energy consumption values also rise with increase in mobility. MMAC performs better than TRAMA, because it adapts the frame time with mobility.

As expected, contention-based protocols have lower latency values than schedule-based protocols. In the event of mobility, the latency values increase, but without large variations. The latency increase of contention-based protocols is caused by the increasing number of retransmissions due to frame errors. The latency increase of TRAMA and MMAC comes from the increase in random access time.

7. Open Issues

Mobility has gained acceptance as an important design criteria for sensor network algorithms only recently. Many open issues and challenges are yet to be addressed in a number of areas:

- There have been proposed solutions at various levels of sensor network hierarchy, but most of them have not been actually verified in real-life settings. More prototyping is required to verify the feasibility of such algorithms.
- Storing mobility information in individual layers of the sensor network stack incurs repetitive overhead in control messages and memory resources. A common mobility management framework is helpful in avoiding duplication of information. Good steps toward defining a common mobility management scheme can be found in some recent work [3,5,22]. Instead of adding the mobility information in the control messages of different layers, it can be consolidated into common control messages per system. However, more research proposals are needed in that direction, as sensor network architecture standards are still shaping up.
- An important limitation in designing mobility adaptive algorithms for sensor networks is the choice of mobility models considered in the design. Even though there are different mobility models, it is important to choose the one that applies to real-life settings. More research is needed in developing realistic mobility models to reflect the mobility patterns that can apply to various sensor network applications.

8. Conclusions

In this paper, we conducted a survey of mobility handling in sensor networks. We discussed the issues posed by mobility in a sensor network and common solutions for handling them in various layers. To incorporate mobility awareness in the protocol design, it is important to understand how mobility can be characterized by mobility models. After discussing mobility patterns, models, and estimation algorithms, we then focused on two protocols that handle mobility at the MAC level by discussing their mobility handling approaches and comparing their performance through simulations. This paper also identifies some open issues in dealing with mobility in sensor networks. Before attempting widespread deployment of mobile sensor networks, such issues need to be addressed.

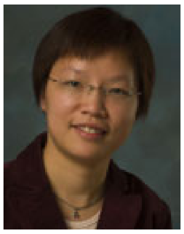
References

1. Demirkol I, Ersoy C, Alagoz F. MAC protocols for wireless sensor networks: a survey. *IEEE Communications Magazine* 2007; **44**(4): 115–121.

2. Ali M, Suleman T, Uzmi ZA. MMAC: a mobility-adaptive, collision-free mac protocol for wireless sensor networks. *Proceedings of the 24th IEEE IPCCC'05*, Phoenix, Arizona, USA, 2005.
3. Ali M, Voigt T, Uzmi ZA. Mobility management in sensor networks. *Proceedings of the 2nd IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*, San Francisco, USA, June 2006.
4. Dantu K, Rahimi M, Shah H, Babel S, Dhariwal A, Sukhatme GS. Robomote: enabling mobility in sensor networks. *Proceedings of IEEE/ACM Fourth International Conference on Information Processing in Sensor Networks (IPSN/SPOTS)*, 2005.
5. Culler D, Dutta P, Ee CT, *et al*. Towards a sensor network architecture: lowering the waistline. *Proceedings of the 10th Conference on Hot Topics in Operating Systems (HotOS)*, 2005.
6. Chatzigiannakis AK, Nikoletseas S. Sink mobility protocols for data collection in wireless sensor networks. *Proceedings of the ACM Workshop on Mobility Management and Wireless Access (MOBIWAC)*, 2006.
7. Liang Q, Ren Q. Energy and mobility aware geographical multipath routing for wireless sensor networks. *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC) 2005*, New Orleans, LA, 2005.
8. Pham H, Jha S. An adaptive mobility-aware MAC protocol for sensor networks (MS-MAC). *Proceedings of the IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, 2004.
9. Schindelhauer C. Mobility in wireless networks. *Proceedings of SOFSEM*, 2006.
10. Jerome H, Fethi F, Christian B. A framework for mobility models generation and its application to inter-vehicular networks. *Proceedings of the 3rd IEEE International Workshop on Mobility Management and Wireless Access*, 2005.
11. Fiore M. Mobility models in inter-vehicle communications literature. Technical Report, Rice University, November 2006.
12. Lu Y, Lin H, Gu Y, Helmy A. Toward mobility-rich performance analysis of routing protocols in adhoc networks: using contraction, expansion and hybrid models. *Proceedings of IEEE International Conference on Communications*, 2004.
13. Jardosh A, Royer E, Almeroth K, Suri S. Towards realistic mobility models for mobile ad hoc networks. *Proceedings of ACM MOBICOM*, 2003.
14. Hu L, Evans D. Localization for mobile sensor networks. *Proceedings of ACM MOBICOM*, 2004.
15. Zaidi ZR, Mark BL. Mobility estimation for wireless networks based on an autoregressive model. *Proceedings of IEEE Globecom*, Dallas, TX, December 2004.
16. Shibata R. Selection of the order of an autoregressive model by akaike's information criterion. *Biometrika* 1976; **63**(1): 117–126.
17. Wu H-T, Ke K-W, Chen C-H, Kuan C-W. Mobile awareness based cluster selection mechanisms in wireless ad hoc networks. *Proceedings of IEEE Vehicular Technology Conference*, 2005.
18. Ye W, Heidemann J, Estrin D. An energy-efficient MAC protocol for wireless sensor networks. *Proceedings of IEEE INFOCOM*, 2002.
19. Rajendran V, Obraczka K, Garcia-Luna-Aceves JJ. Energy-efficient collision-free medium access control for wireless sensor networks. *Proceedings of the 1st International Conference on Embedded Network Sensor Systems (SenSys)*, 2003.
20. Varga A. OMNeT++ Website. Available Online: http://www.omnetpp.org
21. EYES Project Website. Available Online: http://eyes.eu.org
22. Polastre J, Hui J, Levis P, *et al*. A unifying link abstraction for wireless sensor networks. *Proceedings of ACM SenSys*, November 2005.

Authors' Biographies

Anjali Raja received her MS in Computer Engineering from San Jose State University in 2007 and B.Tech. in Computer Science and Engineering from Regional Engineering College, Calicut, India in 2000. She is currently working as a Software Engineer in the Protocols Division of Force 10 Networks, San Jose. Previously, she has worked with Skipper Wireless Labs and NextHop Technologies in developing protocols and routing software for wired and wireless devices. Her current research interests are in ad-hoc and sensor networks, wireless routing and MAC protocols.



Xiao Su received her Ph.D. in Computer Science from the University of Illinois at Urbana-Champaign. She is currently an Associate Professor in the Computer Engineering Department, San Jose State University, San Jose, CA. Her research interests include network security, multimedia communications, media coding, and mobile computing. She has served as Co-Chair for the Int'l Symposium on Multimedia over Wireless and on technical committees on numerous IEEE sponsored conferences. She is a recipient of the National Science Foundation CAREER award.