40 Localized Algorithms for Sensor Networks

Article · May 2012

DOI: 10.1201/9780203489635.ch40

LOCALIZED ALGORITHMS FOR SENSOR NETWORKS

Jessica Feng, Farinaz Koushanfar* and Miodrag Potkonjak
Department of Computer Science
Univeristy of Califomia, Los Angeles
Los Angeles, CA, US 90095
**Department of EECS*
Univeristy of California, Berkeley
Berkeley, CA, US 94720

Abstract: Localized algorithms are optimization procedures that require limited amount of communication, computation, and storage because their execution targets are wireless sensor networks. There is a wide consensus that localized algorithms are crucial for energy efficient operations in sensor networks, security and privacy, reliability and fault tolerance. In addition, they serve as the enablers of scalable sensor networks. The main technical challenges are how to organize the collection of input data and the dissemination of results, how to optimize in presence of uncertainty, and how to analyze and evaluate localized algorithms.

We present sound computation and communication models for localized algorithms. The technical core of the paper is a system of paradigms for developing localized algorithms and several case studies including routing, topology management and sensor fusion. Furthermore, we present several approaches for developing protocols that support localized algorithms and analysis techniques. In addition, we summarize recent research of distributed localized algorithms. We conclude with a short survey of emerging research directions related to localized algorithms.

Key words: Wireless multi-hop networks, Wireless ad-hoc networks, Sensor networks, Data dissemination.

1. INTRODUCTION

1.1 Motivation

Recently, wireless multi-hop networks (WMNs) have emerged as a promising architecture for realization of a various embedded distributed networked systems. WMNs can be used for a variety of tasks including human communication and Internet-like data distribution. The most exciting application of wireless ad-hoc networks is probably serving as the building platform for wireless sensor networks. In wireless sensor networks, each node is equipped with a certain amount of communication, computing, storage, sensing, and, in some scenarios, actuating resources. Wireless ad-hoc sensor networks have the potential to bridge the gap between the Internet and the physical world. Numerous applications in military, personal and industrial tasks have been envisioned. At the same time, wireless ad-hoc sensor networks pose a number of new technological and optimization challenges. It is apparent that in order to address these challenges, sensor networks have to operate in an autonomous mode. In addition, in order to better address low energy, privacy, security, and scalability issues, wireless sensor networks will require new types of algorithms that will use minimal amount of communication. In this chapter, our goal is to state the state-of-the-art of these types of algorithms commonly known as the localized algorithms.

It is interesting to compare localized algorithms to other types of algorithms that have been excessively studied in computer science and related areas. In theoretical computer science and operational research, a great variety of algorithms have been developed for a wide range of combinatorial problems. These algorithms are developed under the following set of assumptions. The first assumption is that there are constraints on only two types of resources: storage and speed of computation. A number of models have been developed under this assumption such as Turing machines, Post's model, and universal register machines. It has been demonstrated that these models are essentially equivalent. The inputs for the algorithm are specified at the beginning of the execution of the algorithm; run time and storage requirements serve as the measurements of the quality of the solutions and algorithms. It is customary to consider algorithms that have run-times as polynomial functions with respect to the length of the input expressed in bits are efficient and the ones that require exponential time are not. On a more practical note, a number of paradigms that can be used to develop efficient algorithms have been identified, including divide and conquer, branch and bound, dynamic programming, and reduce and conquer. The key observation is that algorithms are designed and analyzed mainly based on how well they scale as the size of the input increases asymptotically. In addition, the algorithms that guarantee the optimal solutions and the approaches that guarantee the obtained solutions are within a certain vicinity of the optimal solution are widely studied (e.g. approximation algorithms), as well as the algorithms that provide heuristic solutions when the problem is computationally intractable [Goe95, Hoc97, Joh74].

Although localized algorithms and even sensor networks have only been attracting research and development attention recently, there already exists extensive literature and great variety of proposed approaches regarding the topic. It is already impossible to provide a comprehensive survey of all proposed algorithms for all wireless ad-hoc sensor network tasks. Our main objective in this chapter is to identify the most suitable abstractions and the most efficient techniques as foundations for developing localized algorithms. In addition, special emphasis is placed in summarizing how to analyze and evaluate localized algorithms. Our goal is to cover all of the most important developments as well as provide insights as to why these algorithms are effective. In addition to presenting the already published results, we also propose several new algorithms that are either optimal or are superior in terms of performance to the published ones.

1.2 Chapter Organization

The reminder of the Chapter is organized in the following way. In Section 2, we summarize all the proposed models, abstractions and foundations for designing and analyzing localized algorithms in wireless sensor networks. In the next section, we discuss centralize algorithms which provide a comparison metric to localized algorithms. After that in Section 4, we present several case studies for canonical problems in wireless sensor networks, and the existing algorithms, approaches and general paradigms. A number of widely applied analysis metrics and standards are presented in Section 5. In order to enable distributed localized algorithm, we present different protocols in Section 6 that can be applied in developing these algorithms and we discuss the proposed techniques and algorithms for distributed localized algorithms. Finally in Section 7, we state some of the future conceptual, technological and theoretical challenges related to localized algorithms.

2. MODELS AND ABSTRACTIONS

In this Section, we summarize information about relevant models and abstractions required to specify and to analyze localized algorithms. There is a great deal of diversity among potential combinations of properties of models that can be used for this task. Many of them are interesting because they provide favorable trade-offs between their capability for capturing real-life sensor networks, verses their suitability for analysis and development for variety of optimization techniques. We focus the attention on two groups. The first group is mainly related to the models that have been widely used in literature. The second one captures models that are favored by current and expected technology trends.

We start with the model related to network topology. Currently, only static networks are considered. However, in the near future, we expect that a variety of models for mobile networks will appear. In order to ensure connectivity of all nodes the standard assumption is that all nodes, when viewed at the graph level, form a single connected component. In addition, the edge between two nodes can be unidirectional or bi-directional. The first option is used when all nodes are equipped with identical radio transmitters and receivers. The second option indicates that there are situations where node A can hear node B, but vise versa is not true. In addition, sometimes one or more nodes have special positions as the gateway to the Internet or as a base station. The most important assumption about the network is related to the question of how much each node knows about the locations and the connectivity of all other nodes. The current standard assumption is that each node is only aware of it's own neighborhood, i.e. nodes to which it can directly communicate. Sometimes this definition is enhanced to k-hop neighbors. We believe that in the future, schemes that explicitly state what is stored at each node will emerge. Essentially, as data structures play a crucial rule in the development of standard computer algorithms, data placement play a crucial rule in localized algorithms. It is also important to note that as storage technology rapidly emerges, it is not realistic to assume each node has only information about its own neighborhood. However, while information in static networks can be easily stored in each node, it would be expensive for each node to inform many nodes about its status when the network is mobile or when energy saving procedure is conducted using sleeping mode.

Currently, it is most often assumed that nodes in the network are randomly deployed with uniform distribution in a unit square area. The assumption is justified in some scenarios, such as when nodes are dropped from airplanes. However, it is obvious that new methodologies and approaches for WSNs that have very different structural properties will emerge in order to address the needs of specific applications. In these networks, sensor placement will impact performance of localized algorithms in a very profound way.

Another aspect that is rarely discussed but is crucially important is related to space topology and obstacles. For example, in environmental monitoring, simply ignoring the trees and physical obstacles would inevitably result in incorrect conclusions. Finally note that tasks in 3-d are commonly significantly more difficult than 2-d.

Currently, the standard assumption is that all nodes are equipped with the identical transceivers and identical omni-directional antennas. This assumption has a direct ramification that all two communicating parties have the same transmission and reception strength. However, the communication range can be modeled in various ways depending on the radios used. We briefly state four of the most intuitive options. The first is the unit disk model where all nodes in the network have identical radio range. A generalization of this model is the arbitrary disk model, where each node has an arbitrary radio range and is uniform along all directions. In this case, there does exist situations where node A can hear node B, but node B can not necessarily hear node A. Therefore, arbitrary disk model requires directed graphs for representation of the network connectivity. The third communication model relinquish the assumption of the uniformity of signal propagation along all directions and captures the statistical behavior of propagation signal as a probabilistic function of distance between the communicating node pair. The probability is different along different directions, but is a monotonically non-increasing function along any given direction. Examples of the function that may be applied include: simply the distance formula and the square of distance. The fourth option aims to incorporate complete arbitrariness in communication patterns, it assumes that communication between any two nodes regardless of their positions is established with a certain user-defined probability.

In addition to communication range, assumptions about the structure of transmitted data also play an important role in designing localized algorithms and evaluating their performance. The most widely adopted schemes are: (i) number of bits sent, (ii) number of packets with no restriction of the size of each packet, and (iii) number of packets where each packet has limited size.

The first option does not involve the concept of packets. Information is measured in terms of the number of bits sent and received between two nodes that can communicate directly. The second scheme adopts the notion of packets, but packets are of a relatively large size compared to the information that has to be sent so that they can be considered as unlimited sized packets. The last option imposes an upper limit of information each packet can contain. Note that depending on the adopted communication models and the packet structure models, relative performances across different algorithms may significantly differ. Therefore, constructing algorithms that are most suited for particular set-ups so that they maximize the advantages of the assumptions is of great importance.

There exist a number of energy consumption models. Digitan gives a specific example of an energy consumption model for wireless radio. Assume that transmission on a 2Mb/s 802.11 link takes 1.9W of energy, reception takes roughly 1.5W, idle/listening takes 0.75W, and finally sleeping consumes only 0.025W. The main observation is that unless the node is in the sleeping mode, no significant amount of energy can conserved even if the node is in idle mode. The conclusion is simple and with strong ramification: often it is more important to design localized algorithm that can be executed while a large percentage of nodes are in the sleeping mode.

Storage models can be categorized in two classes: direct and indirect storage. Direct storage implies that all the information each node stores is kept physically within the node itself. Indirect storage is where data used by a node during execution of the localized algorithm is stored somewhere else, either at some other node or possibly a separate gateway storage device. Therefore, this second scheme requires an explicit step of referencing and communication in order to gain access to the information. Clearly, direct storage has advantages over indirect storage in terms of access time, flexibility and communication cost. On the other hand, indirect storage can enable significantly better sharing of data as well as significant storage capacity enhancement.

Fault models are a well-studied topic and have been discussed comprehensively in VLSI and computer architecture literature. However, fault tolerance and therefore fault models have never been one of the dominating concerns and objectives of VLSI designs. The reason is that the properties of VLSI technology and design styles facilitate strong resiliency against faults naturally. However, wireless ad-hoc sensor networks are vulnerable against faults (also equivalent attacks and data skewing) due to their wireless communication and localized mode of operation. Furthermore, usages of such networks also enhance the importance and the need for privacy and security. In addition, the observed physical world is full of obstacles that interfere with communication and sensing tasks. Sensor networks are often deployed in the physical world where the environment is complex or even hostile. For example, consider a habitat monitoring sensor network deployed in a forest. Simply ignoring the existences of trees, plants, and other obstacles will result to incorrect conclusions. Currently, fault tolerance is rarely addressed in sensor networks and the development of fault-tolerant localized algorithmstill has to be addressed.

Sensing models capture sensitivity of a sensor as a function of parameters such distances, properties of the environment, and position. For example, one can assume all sensors have only two sensitivity modes: detecting or not detecting an event. A widely used model for sensitivity is one where the accuracy of sensing decreases according to a certain function of distance between the sensor and the target object. Often used functions are linear and quadratic.

3. CENTRALIZED ALGORITHM

In this Section, we discuss centralized algorithms for sensor networks. After starting with the definition of the centralized algorithms, we briefly outline their major advantages and disadvantages. After that, we summarize several different scenarios in which centralized algorithm can be specified and analyzed. Special emphasis is placed on two phases: data collection and result dissemination. We present several optimal centralized algorithms for common tasks in wireless sensor networks.

Centralized algorithms in wireless ad-hoc sensor networks are the procedures where all information from all nodes in the network is first collected to a single, usually pre-defined, node. The problem is solved at this node and consequently the results of the optimization are disseminated to all nodes that requested this information. Therefore, we can identify the following three phases of centralized algorithms. The first is information collection where readings of all sensors from all nodes are collected to a single computational point. The second phase is optimization mechanism execution on that node. Finally, in the third phase, the results of the optimization are sent to all other nodes using multi-hop communication.

There are several reasons why one has to study centralized algorithms for a given problem where the primary goal is actually to develop the localized algorithm. The first reason is that the centralized algorithm provides an upper bound of what is achievable with respect to the quality of the solution. At the same time, centralized algorithm also provides an upper bound of expected communication cost with respect to the corresponding localized algorithm. Note that both of the previous bounds are not actually guaranteed. For example, in the case of upper bound of the quality of the solution, if the problem is computationally intractable, it may happen that the localized algorithm "gets lucky" and produces better solution than the centralized algorithm. Or in the case of communication cost, the centralized algorithm may "get unlucky", some nodes are visited several times and therefore it results in higher energy consumption than the corresponding localized algorithm. Although there is a wide census, localized algorithms are the right alternative for wireless ad-hoc sensor networks; there are a number of situations where centralized algorithms are obviously competitive if not better. For example, if the network is reasonably small and one has to conduct several optimization problems at the same time, then centralized algorithms are certainly an attractive options to consider. Also, centralized algorithms are particularly well suited for the mapping problem where each node has to get specific set of attributes.

It is important and interesting to consider relative advantages and disadvantages of centralized algorithms with respect to the corresponding localized algorithms. There are a number of aspects where centralized algorithms have significant advantages over localized algorithms. For example, the main logistic advantage is that optimization mechanisms do not have to be customized as in the case of localized algorithms. In addition, the same data collection and distribution algorithm and software can be applied to all problems. Furthermore, both synthesis and analysis of centralized algorithms are significant simpler conceptually and logistically than in the case of localized algorithms. In the case of mapping problems, centralized algorithms are often competitive in terms of the communication cost. Finally, note that performance and cost of centralized algorithms most often have significantly lower variance in terms of quality of solution and communication cost than localized algorithms.

Nevertheless, localized algorithms have significant advantages in many situations that often greatly outweigh their limitations. For example, as the size of the network increases, localized algorithms inevitably become the only realistic option. In particular, they show great advantages when search problems are addressed. Furthermore, localized algorithms provide strong advantages in terms of fault tolerance, security, and privacy. Finally, localized algorithms are much better suited for customization with respect to specific optimization mechanism and communication model.

We will illustrate both advantages and disadvantages of centralized algorithms using several different abstractions and modeling scenarios. We will consider three scenarios where a centralized node is the Internet gateway that contains unlimited computation, storage, and energy supply resources. Note that in this case, centralized node has enough storage to contain all information regarding all nodes and their connectivity. We first consider the case where the communication cost is measured in terms of transmitted data bits. In this case, the problem can be solved optimally. All that is required is that each node sends its information using the shortest path to the centralized node. Dijkstra's algorithm can provide the solution in linear time in terms of number of edges in the graph. Notice that since each node is sending information using the most sufficient route, consequently the optimality of the algorithm is guaranteed.

In the second scenario, the communication cost is measured in terms of the total number of packets transmitted. In this case, we adopt the assumption that packets have unlimited size. Note that this is reasonable when the network is relatively small and the packet size limit is relatively large. In this case, the problem can also be optimally solved. The solution is based on the observation that each node has to send its information at least once to some other node. Therefore, if the algorithm only requires each node to send its information once, the optimality is automatically achieved. The first step of the algorithm is to conduct breath first search (BFS) in order to find the distance in terms of hops of each node from the centralized node. After that, each node in the network is scheduled to transmit its data or the data it has received in a decreasing order according its distance from the centralized node.

The third scenario that we considered is the situation where the packet size is fixed to a certain amount and the goal is again to transmit the minimal number of packets. Unfortunately, the problem is now computationally intractable. Still, it can be solved optimally using integer linear programming (ILP)-based approaches. Note that in many situations, particularly when the network is relatively small and sparse, this is attractive since it has to be solved only once per lifetime of the network. We introduce the following variables:

$$x_{ij} = \begin{cases} m & node \quad i \quad sends \quad m \quad bits \quad to \quad node \quad j \\ 0 & o/w \end{cases} \tag{1}$$

$$x_i = \begin{cases} l & node \quad i \quad sends \quad l \quad outgoing \quad bits \\ 0 & o/w \end{cases} \tag{2}$$

$$y_{ij} = \begin{cases} k & node \quad i \quad sends \quad k \quad packets \quad to \quad node \quad j \\ 0 & o/w \end{cases} \tag{3}$$

There are two types of constraints. First, for each and every node, the outgoing number of bits it sends out must equal to the sum of the received bits plus the number of bits it has recorded. The second constraint ensures that the number of packets is sufficient to transfer the number of bits that need to be transmitted.

$$(\sum_{i=1}^{n} x_{ij}) + (R_i) = x_i \qquad \forall i \tag{4}$$

$$y_{ij} > \frac{x_{ij}}{P} \tag{5}$$

where R_i = number of bits node i has recorded
P = packet size limit in terms of bits
n = total number of nodes in the network

The objective function is to minimize the number of total packets send, therefore:

$$\min: \quad \sum_{i=1}^{n}\sum_{j=1}^{n} y_{ij} \tag{6}$$

Let's now consider the same three scenarios when there is no explicitly predefined centralized node. If the assumption is that each and every node is aware of the situation of the entire network, only minor modifications to the existing approaches would be sufficient. In the first scenario where communication cost is measured in terms of bits, we conduct the 1-to-n shortest path using Dijkstra's algorithm at each and every node, and select the one node that has the smallest sum of shortest paths to all other nodes to be the centralized node. In the case of second scenario where the packet size is large enough to be considered unlimited size, all nodes have the same quality to be the centralized node, therefore any arbitrary node can be served as the centralized node. In the case of the third scenario where the packet size is limited, one arbitrary node solves the system using the same ILP formulation with the assumption of different centralized node, selects the node that provides the best objective function value when it is assumed to be the centralized node, and notifies this node to continue the procedure.

If the assumption is that each node only knows its only limited neighborhood information, the problem becomes more complicated. In this case, we propose the "spiral" algorithm [Kou03+]. Starting from an arbitrarily selected node, the goal is to minimize the number of times each node is visited in order to collect all the information in the network. The algorithm can be best understood in the geometric context. Consider the following illustrative example:

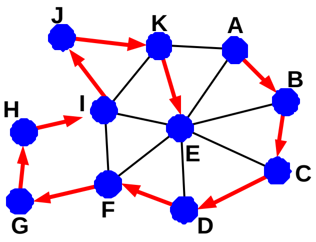

Figure 1. Example Topology

Figure 1 presents a network with 11 nodes, each only is aware of its own 1-hop neighbors. Let node A be the arbitrary starting point, using the clockwise "sweeping" search technique, A finds the first occurrence of a non-visited node, i.e. node B in this case, then node A sends all its information to B. Now B applies the same technique to find the next first occurrence of non-visited node, which is node C. Node B forwards what node A has sent as well as node B's own data to node C. This procedure continues till node E. Node E "sweeps" 360^o, but all the nodes encountered have been visited, therefore node E concludes that it has all the information in the network, and announces itself to be the centralized node.

Once all the information is present at the centralized node, it can apply various optimization techniques to obtain solutions. Let's focus on the last phase of the centralized algorithm, solution dissemination. The problem is equivalent to the broadcasting problem, which can be again addressed using ILP. We define the following variables:

$$x_i = \begin{cases} 1 & node \quad i \quad broadcasts \\ 0 & o/w \end{cases} \qquad (7)$$

$$x_{ij} = \begin{cases} 1 & node \quad i \quad sends \quad message \quad to \quad j \\ 0 & o/w \end{cases} \qquad (8)$$

Using these specified variables, we enforce the following three constraints. First, each node must receive the information from some other node in the network. The second type of constraint ensures that only nodes within the communication range of each other can communicate. The third type of constraint ensures the broadcasting node is only charged once no matter how many nodes have received messages from it.

$$\sum_{\substack{i=1 \\ i \neq j}}^{n} x_{ij} \geq 1 \qquad j = 1,\ldots,n \qquad (9)$$

where n = total number of nodes in the network

$$x_{ij} = 0 \qquad if \;\; E_{ij} \neq 1 \qquad (10)$$

$$x_{ij} \leq x_i \qquad i = 1,\ldots,n; \qquad j = 1,\ldots n; \qquad i \neq j \qquad (11)$$

The objective is, again, to minimize the number of packets sent. In other words, minimize the number of nodes that broadcast:

$$\min : \quad \sum_{i=1}^{n} x_i \qquad (12)$$

4. CASE STUDIES

4.1 Energy management and topology maintenance

There are a number of alternative power minimization methods that act above the MAC layer powering off redundant nodes' radios in order to expand the battery lifetimes. For example, AFECA [Xu00] trades-off energy consumption and the quality of the message delivery services based on the application requirements. GAF [Xu01] is another power saving scheme that saves energy by powering off the redundant nodes. GAF identifies the redundant nodes by using the geographic location and a conservative estimate of the radio ranges. It superimposes a virtual grid proportional to the communication radius of the nodes onto the network. Since all the nodes in one grid are equal from the routing perspective, the radios of the redundant nodes within a grid can be turned-off. The awake nodes within a grid rotate to balance their energy. One of the main advantages of GAF is that it is completely static and localized. All nodes are capable of estimating virtual grids and determining equivalent nodes. In addition to saving 40% to 60% of the energy compare to an unmodified ad hoc routing protocol, GAF also suggests that network lifetime increases proportionally to node density. On the other hand, the grids that contain a very limited number of nodes can easily create significant performance bottleneck. Moreover, sometimes it is acceptable to let all nodes in some grids to sleep (e.g. the boundary nodes) in order to further reduce energy. However, this situation cannot be recognized by GAF.

SPAN is a power saving, distributed and randomized coordination approach [Che01] that preserves connectivity in wireless networks. The work presented in [Kou03] has proved the necessary and sufficient conditions for putting the radios in the sleep mode, while still guaranteeing connectivity. A major advantage of this scheme is that all the decisions are made locally and individually. Therefore, it is much more robust, flexible and scalable than the centralized schemes. In addition, according to the condition of the network, coordinator nodes are adjusted and re-elected locally as well. However, SPAN shares some of similar limitations with

GAF, in particular with respect to energy savings. For example, in some situations not all coordinator nodes need to be awake.

There are also a number of research efforts that trade off between latency and energy consumption. The power management approach presented in [Kra00] selectively chooses short periods of time to suspend and shut down the communication unit. They queue the data before suspending the communication. STEM is a power saving strategy [Sch02] that does not try to preserve the capacity of the network. STEM works by putting an increasing number of nodes into sleep mode, and then encountering the latency to set up a multi-hop path. Nodes in STEM must have an extra low power radio (paging channel) that does not go into sleeping mode and constantly monitors the network to wake the node up in the cases of an interesting event.

4.2 $(MI)^2$

In traditional computer science, backbones for designing efficient algorithms are optimization paradigms such as branch-and-bound, dynamic programming, divide-and-conquer, and iterative improvement. In this Section, we introduce maximally-informed maximally-informing $(MI)^2$ paradigm, the first systematic approach for the design of localized algorithms. In order to make the presentation self-contained, we first summarize the key assumptions and briefly describe the typical sensor network optimization problems that will serve as illustrative examples. After that, we explain how one can apply the $(MI)^2$ strategy in a systematic way during each of the four phases of a localize algorithm: information gathering, system structuring, optimization mechanism, and result dissemination. We start by describing key insights and key trade-offs in designing localized algorithms. We also illustrate the realization of such algorithms on a number of typical sensor network tasks, such as routing and minimum spanning tree.

Given a network, we assume that each node has minimal state information about the network and is only aware of nodes within its communication range. This is so because: (i) there is a need to minimize storage requirements at each node, (ii) nodes go to sleeping mode from time to time in order to minimize the energy consumption [Est99, Rab00], and (iii) updating the routing tables might not be possible as a result of nodes' high mobility.

The goal of the shortest path problem is to find a path between S and D such that the path has the smallest cardinality (i.e. the smallest number of nodes on the path). The MST problem asks to find the minimum spanning tree for a subset of nodes in the network. The connected dominating set problem addresses selecting a subset of nodes of minimal cardinality in such a way that each node is either in the subset or has a neighbor that is in the subset. The importance of the selected problems for wireless ad-hoc network is self-evident. For example, the connected dominating set ensures that information can be efficiently collected or distributed from the nodes in the dominating set to all other nodes [Sto02].

While all the previous research in this area has implicitly specified the four phases in the design of localized algorithms, we explicitly identify and formalize the phases for the first time. More importantly, the novelty of our approach is that, in each step, we have developed insights and systematic generic methods to leverage the $(MI)^2$ paradigm. This results in efficient localized algorithms for a great variety of problems.

Phase 1: information gathering

Information gathering (IG) phase is where the input(s) to the procedure is prepared. If information from multiple nodes is needed, routing between the nodes, the order in which nodes are visited, and the information that is gathered will have a large effect on the amount of energy consumed. According to the maximally informing paradigm, each step of the IG phase has to select the next node to be visited/ contacted in such a way that the maximal amount of relevant information required for the application of the optimization mechanisms is acquired. The maximally informing principle can be realized in several ways depending on the considered scope and objective function of the optimization problem. When considering the scope, one can take either a greedy local view where one considers which nodes can be contacted in a few hops if a particular node is visited next.

When considering the objective function, one could either contact a node that will expose the largest number of constraints itself, or contact a node that has neighbor nodes that will reveal the largest number of constraints. For example, one alternative is to select as the next node one that is likely to have many unvisited neighbors. In this case, we maximize the amount of obtained information. Another alternative is to visit a node that has a large unexplored area within its communication range with a high likelihood of containing nodes that are relevant for optimization.

For example, in shortest path routing, one can always contact a node that is the closest to the destination node in a greedy way. An alternative is to contact the node with the largest area in its communication range, with a large percentage of points that are closer to the destination.

The final important observation related to the IG phase is that in certain situations, visiting some nodes is perhaps more important than visiting other nodes. One such situation is when the goal is to find the connected dominating set for all nodes in a geographic region. In this situation, it is crucial to visit all nodes on the outer perimeter of the network since their information could guarantee that all of the relevant nodes are considered. Therefore, in this situation, the $(MI)^2$ paradigm indicates that these nodes should be visited first.

Phase 2: system structuring

Every node in the system has some amount of processing capability. However, not all of the nodes in the system need to compute the optimization procedure all the time. In the system structuring phase, we decide when and where to conduct optimization mechanism computations.

According to the $(MI)^2$ paradigm, we follow two principles for selection of computation centers. The first is to initially assemble enough information to meaningfully conduct at least part of the computation as soon as possible. This point is particularly well illustrated on the MRA [Kou03] and exposure tasks [Meg01]. The second principle is to always conduct computations at the boundary of an already visited region in order to reduce the requirements for obtaining additional information. This point is clearly illustrated with the exposure task.

Finally, note that different optimization mechanisms dictate different system structuring phases. In some tasks, such as MRA, the local information is sufficient to guarantee the optimum solution. However, in computationally intractable optimization problems, where the interaction between all of the nodes in the system defines the output, the quality of the solution may be seriously hampered using only localized scopes. In such situations, we have to obtain information about a large neighborhood for each node before we start the optimization mechanism.

Phase 3: optimization mechanism

Once the needed input is at a computation center, the optimization procedure is executed. The separation of concerns principle suggests that the phases should be as independent as possible. However, in the majority of problems there is a strong interdependence between the information gathering phase and the optimization mechanism (OM) phase because based upon the specific needs for executing an optimization mechanism, the relevant information must be acquired.

The first observation is that constructive and deterministic algorithms are strongly preferred to iterative improvement and probabilistic algorithms. This is because the former type of algorithms require only one pass through all inputs, while the latter require multiple passes.

For example, for the MST problem, Prim's algorithm is much better suited for implementation as a localized algorithm than Kruskal's algorithms. This is the case because Prim's MST algorithm starts from an arbitrary node and at each step selects the shortest edge that is incident to one of the nodes already visited and does not form a cycle with the edges in the existing partial MST. This edge is then added to the partially built MST. Therefore, the algorithm uses only information about the nodes that are already visited and their neighbors. On the other hand, Kruskal's algorithm requires one to consider all edges in the graph at each step and select the globally shortest edge. Therefore, before one starts the execution of Kruskal's algorithm, they have to obtain information about the whole graph.

The $(MC)^2$ optimization paradigm is well-suited for use in conjunction with the $(MI)^2$ paradigm. In order to gain maximal benefit from the merged $(MC)^2(MI)^2$ paradigm, it is often advantageous to consider variants of $(MC)^2$ that only considers nodes adjacent to already explored nodes. This must be done in such a way the communication requirements are reduced. Other optimization paradigms that are naturally well-suited for design of localized algorithms and in particular with the $(MI)^2$ paradigm, are branch and bound and dynamic programming-based algorithms. Finally, note that in some cases, such as exposure calculations, one can directly use the available optimization mechanism. In others, such as the MRA problem, in order to design efficient localized algorithm, one has to develop a new optimization mechanism and therefore, a new centralized algorithm that operates locally and with the partial information.

Phase 4: information dissemination

The information dissemination phase is the step where the output of the optimization procedure is being sent to the nodes requiring that information. The maximally informed paradigm states that we should disseminate information about the output of the optimization node to a particular node while we are close to that node. In the ideal case of balanced optimization and information distribution phases, we should send all information that some node requires when we are visiting the last of its neighbors.

4.3 Solving ILP problems by $(MI)^2$-based paradigm

To demonstrate the wide application range of a paradigm for designing localized algorithms, we apply it to a set of specific problems that can be specified and solved using a particular optimization solving strategy. In this subsection, we present an $(MI)^2$-based approach for solving an instance of a problem specified as an integer linear program (ILP). ILP is a widely used procedure for specifying and solving combinatorial optimization problems. ILP formulations are either readily available for a large variety of combinatorial optimization problems, or they are easy to develop [Nem88]. In particular, a special case of ILP, called 0-1 ILP, where all variables have to be assigned to one binary value [Nem88]. For example, all problems discussed in this paper can be easily specified and solved using a 0-1 ILP formulation.

ILP formulation has three different components: variables, objective function, and constraints. Variables can take only integer values. Both the objective function and constraints have to be linear. Note that if we remove the requirement that variable have to be integers, ILP reduces to a linear program (LP) that also has a wide range of applications[Pap82]. An ILP that is defined over a set of variable x_i has the following standard form:

$$\textit{Max } E^T.X, \tag{13}$$

$$\textit{Such that}: \quad A^T.X = B,\ C^T.X=D \tag{14}$$

Where A, B, C, D, and E are matrices composed of real constants, and X is vector consisting of variables x_i. The first clause is the objective function (OF), while the equations on the second line are the constraints.

We assume that each node has information about one or more coefficients from matrices A, B, C, D, and E. Furthermore, the node has a list of its neighbors and a list of information about each neighbor, but it does not necessarily have all the information that each neighbor has. The reason for this assumption is that for many optimization parameters (such as energy level, sleep state, occupancy of buffers, collected sensor information) is transmitted only on demand in sensor network in order to reduce power consumption. Finally, each node has to be informed about value of all variables x_i that are of importance to it.

The $(MI)^2$-based approach for locally solving of an ILP instance is based on the following observation and intuition: We can assign a particular value to a particular variable x_i, only after we obtain information about all constraints that contain x_i. Furthermore, it is advantageous to first resolve variables that are components of the strictest constraints. Also, it is important that as much as possible information about the constraints that contain the variable is under consideration when assigning a particular value to that variable. In order to maximize the objective function, it is important to assign high values to variables that have high coefficients and to keep estimating the highest possible value of the objective function (OF) in view of the already observed constraints. One can envision two approaches with respect to the relationship between the objective function and constraints: optimistic, where we prefer to maximize the objective function at a potential danger that some constraints will become unclassifiable later, and pessimistic, where we favor constraints at expense of the objective function. The $(MI)^2$-based localized ILP procedure is summarized using the pseudocode presented in Figure 2.

```
Procedure (Localized (MI)^2-based ILP Procedure)
Initialization;
while (termination criteria ==No) {
    Contact a neighbor that has highest information function (IF);
    if (there are neighbors that do not have unvisited neighbors) {
        execute the optimization mechanism and communicate
        assigned values of the assigned variables to them } }
```

Figure 2. Pseudocode for $(MI)^2$-based localized ILP procedure

The IF is weighted sum of resolving power of the information available at the node and resolving power of its neighbors. The weights of neighbors of a node are scaled by the average number of neighbors from already visited nodes. The resolving power is proportional to reduction in information uncertainty according to the classical information theoretical definition.

The optimization mechanism that we use is based on the maximally-constrained, minimally-constraining principle. Essentially, we try to assign each variable in such a way that they resolve the maximal number of constraints or increase chance that they are later satisfied. The optimization function is treated as a constraint that is dynamically updated.

The initial value is provided by a simple probabilistic analysis and consequently the value is updated by extrapolating the values obtained from the already visited nodes.

4.4 GPSR

Routing is one of the fundamental tasks in wireless networks. While one can envision a number of different types of routing, we focus on the case where a single message has to be sent from a node to another node. We will consider only one localized routing algorithm in order to describe and analyze it in sufficient detail.

Karp and Kung [Kar00] have developed a stateless routing protocol for wireless networks, GPSR (Greedy Perimeter Stateless Routing). The development of GPSR is based on two main assumptions. First, it assumes that each node (router) in the network is aware of its geographic location and the geographic locations of all its direct (one-hop) neighbors. Second, it assumes that the geographic location of the destination is also known. GPSR abandons the traditional routing concepts that require continual distribution of the current map of the entire network's topology to all nodes. The packet forwarding decisions are made based on only the positions, knowledge of local nodes, and the final destination location. More specifically, each node considers the locations of all neighbors, and makes a greedy decision to forward the data packet to the node that is closest to the destination. Therefore GPSR is stateless in the sense that it does not keep additional information about the rest of the network beyond its neighborhood. As a consequence GPSR scales better than the traditional routing protocols and is much more adaptive to mobility.

GPSR protocol has two phases: greedy forwarding and perimeter forwarding. Greedy forwarding is referring to the phase where a series of nodes follow the same rule and each node makes a greedy decision of forwarding the data packet to the one neighbor that the current node believes is the closest to the destination. However, greedy forwarding would fail in the situation where a node is the local minimum in terms of its geographic distance to the destination, i.e. when all its neighbors have longer distances to the destination then itself. In this case, the control is switched to perimeter forwarding mode from greedy forwarding in order to escape the deadlock. Perimeter forwarding essentially follows the right-hand rule, which seeks to find alternative route around and eventually converge to the destination.

In addition to being stateless and having exceptional scalability, GPSR has a number of other noble properties. It is efficient in the sense that it often selects the optimal or near optimal path when the network is dense. It is also very simple and clean to implement and conceptually. However, it has a number of limitations. For example, if the network is not very dense, it is easy to show that the greedy approach is not the best choice since the scope of the problem considered is limited with respect to the available information. In addition, there is no guarantee that GPSR would eventually converge to the destination. There exist situations where forwarding phase and perimeter phase oscillate within a set of nodes, and never converge to the correct destination. It is also difficult if not impossible, to see how to generalize the approach when additional information is available to 3-d case, or in presence of obstacles.

5. ANALYSIS

Creation of algorithms has two interdependent phases: synthesis and analysis. While synthesis of localized algorithms is widely considered as a difficult and demanding task, analysis often does not receive the proper attention and treatment. In this Section, we discuss the most important issues related to analysis of localized algorithms.

The analysis of localized algorithms can be defined as a process of characterizing the effectiveness of a proposed localized algorithm for a given problem. It is a complex and often cumbersome task due to the following reasons. First of all, it is not easy to identify what properties of the algorithms are interesting and important. Even when these properties are identified, it is often unclear how to exactly define each of them. In addition, it is often difficult to calculate or measure these properties. For example, some of properties are associated with solving computationally intractable problems. The next layer of complexity comes from a need to consider more than one property simultaneously. Furthermore, it is not clear a priory that what should be the representative and realistic properties of instances. Finally, one can consider localized algorithms as generalization of on-line algorithms where designer has impact on what information will be obtained next. Therefore, unpredictability often results in randomness of characteristics of a particular algorithm.

The primary goal of localized algorithms is to minimize amount of energy spend on communication. Note that this does not necessarily mean minimization of the number of packets. The current technology indicates that the most effective way of saving energy is through placing the radios of as many as possible nodes into sleeping mode. Also note that in future applications, energy minimization will not be necessarily equivalent in the first approximation the minimization of energy devoted to communication. Depending on the technology and more on the targeted applications, computation or some other components may have dominant role.

The primary constraint is to achieve the user requested level of optimality and/or accuracy. Note that due to complex error propagation through the sensor fusion phase, it is sometimes difficult to select the most appropriate definition of accuracy.

Historically, the performance of algorithms has been evaluated as the size of their input asymptotically increases. Also, in traditional networking research, one of the key issues is scaling of the protocols as the size of the network increases. Although many of the wireless sensor networks will be of limited size, scaling of localized algorithms is already widely studied. A better way to evaluate localized algorithms for limited-sized networks is probably the development of benchmarks. Unfortunately, there are a very few benchmarks right now and all of them are synthetic.

In addition to these three metrics, amenability to provide fault-tolerance, satisfaction of real-time constraints (such as throughput and latency), maintenance of privacy and security, and facilitation of mobility will be also of prime importance for evaluation of localized algorithms.

One can envision many ways to combine two or more metrics. For example, in operation research literature, it is common to derive a set of solutions that form Pareto optimal curve. In the computer science literature, it is more common to take one metrics as the optimization goal and others as constraints.

6. PROTOCOLS AND DISTRIBUTED LOCALIZED ALGORITHMS

In this Section, we briefly discuss the distributed localized algorithms where more than one thread of computation is being executed at the same time. Distributed localized algorithms have a number of advantages in term of their ability to respond faster to the changes in the environment and the network, fault tolerance, and their resiliency against security attacks. We first state the desiderata for protocols that govern the execution of distributed localized algorithms. After that, we present one generic approach for the development of protocols for distributed localized algorithms [Kou03].

Proper computation and synchronization strategy should have the following characteristics:

Concurrency: The computation (decision making) should take place at as many places in the network as possible. In particular, nodes should be constantly updating their resources to cope with the dynamics in the network.

Synchronization (avoiding deadlocks): The computing nodes should not have a conflict on the resources they use. For example, assume that a node v1 finds a node v2 redundant in terms of a specific functionality. At the same time, v2 also finds v1 redundant. If both the v1 and v2 decide to go to sleep (using each other as a back up), a deadlock will occur. A good synchronization strategy must avoid deadlock situations like this.

Overhead: The computation and synchronization strategy should have as low as possible overhead added to the network, especially in terms of its power consumption and communication overhead.

Latency: Higher latency in putting a node to sleeping mode implies more idle energy consumption. Also, the nodes should be updated for changes in the network to adapt to the network dynamics.

Fault-tolerance: Fault is inevitable in sensor networks. The computation and synchronization strategy should be designed such that the faults in any number of nodes could not corrupt its functionality.

Koushanfar et al [Kou03] has developed an approach termed distributed token mechanism that attempts to fulfill the stated requirements. A token indicates that the node has the control of the local flow of the sleeping procedure. At each point of time, there can be more than one token present in the network to comply with the concurrency requirements. A token is generated by an awaken node that needs to check the eligibility of the nodes within its local scope to enter the sleep state. The token is eliminated as soon as it examines the functionality of its local scope of the network and selects the nodes for sleeping. The node with the token locks its local area of consideration so that no other nodes use the same resources and the nodes that act on the mutual resources are synchronized. To lock a node means to consider it only for one token at each point of time.

The localized and distributed nature of the token generation makes it very tolerant to faults at the individual nodes. The pseudocode for the distributed token mechanism procedure is shown in Figure 3. As shown in Figure 3, a node v_i that is not already locked by any other nodes considers running the sleeping procedure (step 1-4) and therefore, it generates a token. A node that was asleep before generates the token at a random time r_i that is within the interval ($0 < r_i <$ rmax) (step 5-9). A node that has already changed its state into sleep at least once generates a token as soon as it wakes up (step 10-11). A node with the token locks all of the unlocked nodes within its local scope of consideration (step 12). This node then runs the sleeping procedure, which decides which of the locked nodes can enter the sleep state (step 13) and for how long (step 14). The token node then announces the decision to its neighborhood (step 15) and unlocks the locked nodes (step 16).

```
Procedure *Distributed Token mechanism*

at $\forall$ node $v_i$,
{
    while (node $v_i$ is not locked by another node)
    {
        if (never have slept before)
        {
            set a random initiation time $r_i$ ($0 < r_i < r_{max}$);
            generate a token at the time $r_i$;
        }
        else {
            generate a token as soon as $v_i$ wakes up;
            lock all the unlocked nodes in the $v_i$'s scope;
            select the best node to sleep;
            select the sleep interval for the sleeping node;
            announce the decision in the neighborhood;
            unlock the locked nodes;
        }
    }
```

Figure 3. pseudocode for the distributed token mechanism procedure

The random initiation time (r_i) assigned to each node in the beginning of the procedure serves the purpose of avoiding simultaneous requests for the use of mutual resources. Since the sleep intervals are assigned independently to the nodes depending on the power and topology of the neighborhood, the wake up times are different. Therefore, after a node wakes up, it can immediately start another round of sleeping strategy without having too many locked nodes in its neighborhood. It is reasonable to not concern about the collisions in the network since they rely on the network's MAC layer to resolve any such conflicts.

7. PENDING CHALLENGES

In this Section, we outline some of the potential trends for developing localized algorithms. It is always dangerous to make predictions, in particular when the topic is broad and application dependent. Nevertheless, we believe one can identify some major trends. We classify future research directions in two broad categories. The first one is related to the conceptual novelties for the developing of localized algorithms. The second category is related to optimization and algorithmic techniques themselves. Due to space limitations, many important directions such as interaction of localized algorithms with privacy and security, mobility, fault-tolerance, applications within real-time systems, and the use for actuator-based system is omitted.

It is well known that mandatory prerequisites for developing high quality algorithms are sound theoretical foundations. In some cases, one can develop such foundations, for example, PRAM, URM and the Von Neumann models of computation. In the case where it is difficult to define a single widely applicable model, such as in the case of parallel computing, the progress is much slower. Currently, several models have been proposed for wireless ad-hoc networks, including [Zon97]. However, it seems that the complete random nature of these models make them of relatively limited practical relevance. Several other fields have also developed theoretical models. For example, in VLSI computations, the standard model is the one that assumes planarity and finite feature size of transistors and interconnects. The development of sound foundations for wireless sensor networks is a complex and difficult task because one has to model at least four aspects of the systems: computation, communication, storage and sensing.

Future algorithmic techniques can be naturally classified in two groups: one is related to design, the other is related to the analysis of localized algorithms. Design related issues include the development on new paradigms that will facilitate systematic creation of localized algorithms, in particular, data collection and dissemination. An example of this is the maximally-informing and maximally-informed paradigm [Kou03^{++}].

Currently, although a number of localized algorithms have been published, there is still relatively little knowledge about their optimality in terms of quality of solution and expected energy cost. Several approaches have been purposed for this purpose, including the development of low bounds and probabilistic analysis. We expect that this trend will continue and will include new hard bound techniques as well as statistical guarantees.

Obviously there is a strong correlation between how nodes are deployed and performances of localized algorithms. It is easy to see that for different wireless sensor network organizations, different localized algorithms are best suited. Interestingly, this topic has not been addressed. In particular, we believe sensor networks with regular structure such as grid can facilitate the development of fast and efficient localized algorithms. Another important issue with respect to localized algorithms for sensor networks is the development of optimization mechanisms that are resilient against unavoidable errors in sensor measurements. Finally, we expect that there will be a particular need to develop comprehensive approaches that combine continuous, discrete and statistical techniques in order to obtain efficient localized algorithms. An example of this is the exposure coverage [Meg01].

Another aspect of localized algorithm development is the analysis of localized algorithms. We expect that we will soon see a great deal of activities to define and to develop scalable algorithms that are not only scalable with respect to the size of the network, but also with respect to the intensity of errors and the quality of solutions. Localized algorithms are in a sense the generalization of the concept of on-line algorithms where one can decide which piece of information to obtain next. Competitive analysis of on-line algorithms have been a widely studied topic, we expect it will be important for localized algorithms as well.

From the practical point of view, the most urgent issue is to develop standard benchmark examples that can properly capture the properties of real life applications. Once the benchmarks are available, it would be important to statistically analyze localized algorithms using statistical and perturbation analysis [Gro02, Gro03].

We expect that a new network (distributed systems) architecture will appear, and it will be well suited for specific classes of tasks and applications. In addition, there is an urgent need for rapid prototyping and simulation platforms on which performances of localized algorithm can be accurately observed and quantified. Another important research direction is the development of design patterns for common localized algorithms. Design patterns changed the way how software development is conducted and we expect that it will have high impact in sensor networks.

ACKNOWLEDGEMENTS

This material is based upon work supported in part by the National Science Foundation under Grant No. ANI-0085773 and NSF CENS Grant.

REFERENCES

[Che01] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, “Span: an energy-efficient coordination algorithm for topology maintenance in ad-hoc wireless networks,” *International Conference on Mobile Computing and Networking (MOBICOM)*, pp. 85-96, 2001.

[Est99] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, “Next century challenges: scalable coordination in sensor networks,” *International Conference on Mobile Computing and Networking (MOBICOM)*, pp. 263-270, 1999.

[Goe95] M.X. Goemans, and D.P. Williamson, “Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming,” *Journal of Association for Computing Machinery*, Vol. 42, No. 6, pp. 1115-1145, 1995.

[Gro02] M. Grossglauser, and D.N.C. Tse, "Mobility increases the capacity of ad hoc wireless networks," *IEEE/ACM Transactions on Networking*, Vol. 10, No. 4, pp. 477-486, 2002.

[Gro03] M. Grossglauser, and M. Vetterli, "Locating nodes with ease: mobility diffusion of last encounters in ad hoc networks," *Annual Joint Conference of the IEEE Computer and Communications societies (INFOCOM)*, pp. 1954-1964, 2003.

[Hoc97] D.Hochbaum, editor, *Approximation Algorithms for NP-hard Problems*, PWS Publishing Company, 1997.

[Joh74] D. Johnson, "Approximation algorithms for combinatorial problems," *Journal of Computer and System Sciences*, Vol. 9, pp. 256-278, 1974.

[Kar00] B. Karp, and H.T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," *International Conference on Mobile Computing and Networking (MOBICOM)*, pp. 243-254, 2000.

[Kou03] F. Koushanfar, A. Davare, D.T. Nguyen, M. Potkonjak, and A. Sangiovanni-Vincentelli, "Low power coordination in wireless ad-hoc networks," *International Symposium on Low Power Electronics and Design*, pp. 475-480, 2002.

[Kou03$^+$] F. Koushanfar et al., "Algorithms for resource discovery in wireless networks," unpublished manuscript, 2003.

[Kou03^{++}] F. Koushanfar et al., "Maximally-informing and maximally-informed algorithms for wireless networks," unpublished manuscript, 2003.

[Kra00] R. Kravets, and P. Krishnan, "Application-driven power management for mobile communication," *Wireless Networks*, Vol. 6, No. 4, pp. 263-277, 2000.

[Meg01] S. Meguerdichian, F. Koushanfar, G. Qu, and M. Potkonjak, "Exposure in wireless ad-hoc sensor networks," *International Conference on Mobile Computing and Networking (MOBICOM)*, pp. 139-150, 2001.

[Nem88] G.L. Nemhauser, and L.A. Wolsey, *Integer and Combinatorial Optimization*, John Wiley, 1988.

[Pap82] C. Papadimitriou, and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Prentice Hall, Englewood Cliffs, NY, 1982.

[Rab00] J.M. Rabaey, M.J. Ammer, J.L. da Silva Jr, D. Patel, and S. Roudy, "PicoRadio supports ad hoc ultra-low power wireless networking," *Computer Magazine*, pp. 42-49, July 2000.

[Sch02] C. Schurgers, V. Tsiatsis, and M. Srivastava, "STEM: topology management for energy efficient sensor networks," *IEEE Aerospace Conference*, pp.78–89, 2002.

[Sto02] 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, pp. 14-25, January 2002.

[Xu00] Y.Xu, J. Heidemann, and D. Estrin, "Adaptive energy-conserving routing for multi-hop ad-hoc networks," *Technical Report 527*, USC/Information Science Institute, October, 2000.

[Xu01] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed energy conservation for ad hoc routing," *International Conference on Mobile Computing and Networking (MOBICOM)*, pp. 70-84, 2001.

[Zon97] M. Zonoozi, and P. Dassanayake, "User mobility modeling and characterization of mobility patterns," *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 7, pp.1239-1252, September 1997.