40

Localized Algorithms for Sensor Networks

Jessica Feng
University of California at Los Angeles

Farinaz Koushanfar
University of California at Berkeley

Miodrag Potkonjak
University of California at Los Angeles

40.1 Introduction

40.1.1 Motivation

Recently, wireless multihop 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 the military environment as well as in 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 must operate in autonomous mode. In addition, in order to address low-energy, privacy, security, and scalability issues better, wireless sensor networks will require new types of algorithms that will use minimal amounts of communication. The goal of this chapter is to discuss the state of the art of algorithms commonly known as 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 has been developed for a wide range of combinatorial problems. These algorithms are developed under the following set of assumptions. The first is that constraints are on only two types of resources: storage and speed of computation. A number of models have been developed under this assumption, such as the Turing machine, Post's model, and the universal register machine. It has been demonstrated that these models are essentially equivalent. The inputs for the algorithm are

specified at the beginning of its execution; run time and storage requirements serve as measurements of the quality of the solutions and algorithms. It is customary to consider algorithms that have run-time as polynomial functions with respect to the length of the input expressed in bits as efficient and the ones that require exponential time as inefficient.

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, algorithms that guarantee optimal solutions and approaches guaranteeing that obtained solutions are within a certain vicinity of the optimal solution are widely studied (e.g., approximation algorithms), as well as algorithms that provide heuristic solutions when the problem is computationally intractable [3, 6, 7].

Although localized algorithms and even sensor networks have only been attracting research and development attention recently, already a wide literature and great variety of proposed approaches regarding the topic exist. It is already impossible to provide a comprehensive survey of all proposed algorithms for all wireless ad hoc sensor network tasks. The 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. The goal is to cover all the most important developments as well as provide insights on why these algorithms are effective. In addition to presentation of already published results, several new algorithms that are optimal or superior to the published ones in terms of performance are proposed.

40.1.2 Chapter Organization

Section 40.2 summarizes all the proposed models, abstractions, and foundations for designing and analyzing localized algorithms in wireless sensor networks. In the next section, centralized algorithms that provide a comparison metric to localized algorithms are discussed. Section 40.4 presents several case studies for canonical problems in wireless sensor networks, as well as the existing algorithms, approaches and general paradigms. A number of widely applied analysis metrics and standards are presented in Section 40.5. In order to enable distributed localized algorithms, the different protocols in Section 40.6 can be applied in developing them; proposed techniques and algorithms for distributed localized algorithms are also discussed. Finally, Section 40.7 states some of the future conceptual, technological, and theoretical challenges related to localized algorithms.

40.2 Models and Abstractions

This section summarizes information about relevant models and abstractions required to specify and analyze localized algorithms. Much diversity is present 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 of capturing real-life sensor networks and their suitability for analysis and development of a variety of optimization techniques. Attention is focused on two groups: (1) those mainly related to widely used models in the literature; and (2) models favored by current and expected technology trends.

Currently, only static networks are considered when one studies models related to network topology. However, in the near future, 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 bidirectional. The first option is used when all nodes are equipped with identical radio transmitters and receivers. The second indicates situations in which node A can hear node B, but not vice versa. In addition, sometimes one or more nodes have special positions as gateways to the Internet or as base stations. The most important assumption about the network is related to the question of how much each node knows about the locations and connectivity of all other nodes.

The current standard assumption is that each node is only aware of its own neighborhood, i.e., nodes to which it can directly communicate. Sometimes this definition is enhanced to k-hop neighbors. 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 plays a crucial rule in localized algorithms. It is also important to note that as storage technology rapidly emerges, assuming that each node has only information about its own neighborhood is unrealistic. However, although information in static networks can be easily stored in each node, it would be expensive for each node to inform too many nodes about its status when the network is mobile or when an 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 unit square areas. The assumption is justified in some scenarios, for example, when nodes are dropped from airplanes. However, it is obvious that new methodologies and approaches for WSNs with very different structural properties will emerge in order to address the needs of specific applications. In these networks, sensor placement will affect performance of localized algorithms in a very profound way.

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

Currently, the standard assumption is that all nodes are equipped with identical transceivers and identical omnidirectional antennas. This assumption has the 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 radios used. Four of the most intuitive options include:

- In the unit disk model, all nodes in the network have identical radio range.
- A generalization of the unit disk model is the arbitrary disk model, in which each node has an arbitrary radio range and is uniform along all directions. In this case, situations exist in which node A can hear node B, but node B cannot necessarily hear node A. Therefore, the arbitrary disk model requires directed graph for representation of the network connectivity.
- Another communication model relinquishes 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. Probability is different along different directions, but is a monotonically nonincreasing function along any given direction. Examples of the function that may be applied include the distance formula and the square of distance.
- Another 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 on the structure of transmitted data also play an important role in designing localized algorithms and evaluating their performance. The most widely adopted schemes are: (1) number of bits sent; (2) number of packets with no packet size restrictions; and (3) number of packets in which each packet has limited size.

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

A number of energy consumption models exist. A specific example of an energy consumption model for wireless radio is given by Digitan. Assume 2 Mb/s 802.11; transmission takes 1.9 W of energy; reception

takes roughly 1.5 W; idle/listening takes 0.75 W; and sleeping consumes only 0.025 W. The main observation is that unless the node is in the sleeping mode, no significant amount of energy can be conserved even if the node is in idle mode. The conclusion is simple and with strong ramifications: often it is more important to design localized algorithm that can be executed while a large percentage of nodes is 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. In indirect storage, data used by a node during execution of the localized algorithm are stored somewhere else — at some other node or possibly a separate gateway storage device. Therefore, this 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 for 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) because of their wireless communication and localized mode of operation.

Furthermore, use of such networks also enhances 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 existence of trees, plants, and other obstacles will lead to incorrect conclusions. Currently, fault tolerance is rarely addressed in sensor networks and the development of a fault-tolerant localized algorithm still must be addressed.

Sensing models capture sensitivity of a sensor as a function of parameters such as 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 in which the accuracy of sensing decreases according to a certain function of distance between the sensor and the target object. Linear and quadratic functions are often used.

40.3 Centralized Algorithm

This section discusses centralized algorithms for sensor networks. After the definition of centralized algorithms, their major advantages and disadvantages are briefly outlined. After that, several different scenarios in which centralized algorithms can be specified and analyzed are summarized. Special emphasis is placed on two phases: data collection and result dissemination. Several optimal centralized algorithms for common tasks in wireless sensor networks are presented.

Centralized algorithms in wireless ad hoc sensor network are procedures in which all information from all nodes in the network is first collected at a single, usually predefined, 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, three phases of centralized algorithms can be identified:

- Information collection in which readings of all sensors from all nodes are collected to a single computational point
- Optimization mechanism execution on that node
- Results of the optimization sent to all other nodes using multihop communication

One must study centralized algorithms for a given problem in which the primary goal is to develop the localized algorithm for several reasons. 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, this

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 a better solution than the centralized algorithm. In the case of communication cost, the centralized algorithm may get unlucky and some nodes are visited several times; therefore, energy consumption higher than the corresponding localized algorithm is the result.

There is a wide consensus that localized algorithms are the correct alternative for wireless ad hoc sensor networks. In a number of situations, centralized algorithms are obviously competitive if not better. For example, if the network is reasonably small and one must conduct several optimization problems at the same time, centralized algorithms are certainly attractive options to consider. Also, centralized algorithms are particularly well suited for a mapping problem in which each node must get a specific set of attributes.

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

Nevertheless, localized algorithms have significant advantages in many situations that often greatly outweigh their limitations. For example, if 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 mechanisms and communication models.

The advantages and disadvantages of centralized algorithms will be illustrated using several different abstractions and modeling scenarios. Three scenarios in which a centralized node is the Internet gateway that contains unlimited computation, storage, and energy supply resources will be considered. Note that, in this case, the centralized node has enough storage to contain all information regarding all nodes and their connectivity.

First consider a case in which the communication cost is measured in terms of transmitted data bits. This problem can be solved optimally. All that is required is that each node send 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, because each node is sending information using the most sufficient route, 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, the assumption that a packet has unlimited size is adopted; 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 must 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 that it has received in a decreasing order according to its distance from the centralized node.

The third scenario is the situation in which 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 because it must be solved only once per lifetime for the network. The following variables are introduced:

$$X_{ij} = \begin{cases} m & \textit{node i sends m bits to node j} \\ 0 & o/w \end{cases} \tag{40.1}$$

$$X_i = \begin{cases} I & \textit{node i sends l outgoing bits} \\ 0 & o/w \end{cases} \tag{40.2}$$

$$Y_{ij} = \begin{cases} k & \textit{node i sends k packets to node j} \\ 0 & o/w \end{cases} \tag{40.3}$$

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

$$\left(\sum_{i=1}^{n} x_{ij}\right) + \left(R_i\right) = \forall i \tag{40.4}$$

$$y_{ij} > \frac{x_{ij}}{P} \tag{40.5}$$

where

R_i = number of bits that 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 sent; therefore:

$$\min: \sum_{i=1}^{n} \sum_{j=1}^{n} y_{ij} \tag{40.6}$$

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 in which communication cost is measured in terms of bits, conduct the 1-to-n shortest path using Dijkstra's algorithm at each and every node, and select as the centralized node the one node with the smallest sum of shortest paths to all other nodes. In the case of the second scenario, in which 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, in which the packet size is limited, one arbitrary node solves the system using the same ILP formulation with the assumption of a 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 limited neighborhood information, the problem becomes more complicated. In this case, the "spiral" algorithm [10] is proposed. Starting from an arbitrarily selected node, the goal is to minimize the number of times each node is visited in order to

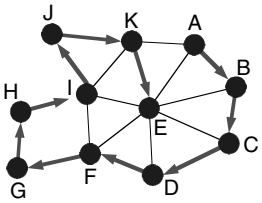

FIGURE 40.1 Example topology.

collect all the information in the network. The algorithm can be best understood in a geometric context. Consider the following illustrative example:

Figure 40.1 presents a network with 11 nodes; each is only aware of its own one-hop neighbors. Let node A be the arbitrary starting point; using the clockwise "sweeping" search technique, A finds the first occurrence of a nonvisited node, i.e., node B in this case. Therefore node A sends all its information to B. Now B applies the same technique to find the next first occurrence of a nonvisited node; this is node C. Node B forwards what node A has sent and node B's own data recorded to node C. This procedure continues until node E, which "sweeps" 360°. However, all the nodes encountered have been visited, so node E concludes that it has all the information in the network and announces that it is the centralized node.

Once all the information is present at the centralized node, it can apply various optimization techniques to obtain solutions. 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. Define the following variables:

$$X_i = \begin{cases} 1 & \textit{node i broadcasts} \\ 0 & o/w \end{cases} \tag{40.7}$$

$$X_{ij} = \begin{cases} 1 & \textit{node i sends message to j} \\ 0 & o/w \end{cases} \tag{40.8}$$

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

$$\sum_{\substack{i=1 \\ i \neq j}}^{n} x_{ij} \geq 1 \qquad j = 1, \ldots, n \tag{40.9}$$

where n = total number of nodes in the network.

$$x_{ij} \qquad \textit{if } E_{ij} \neq 1 \tag{40.10}$$

$$x_{ij} \leq x_i \qquad i = 1, \ldots n; \quad j = 1, \ldots n; \quad i \neq j \tag{40.11}$$

The objective is again to minimize the number of packets sent, i.e., minimize the number of nodes that broadcast:

$$\min: \sum_{i=1}^{n} x_i \tag{40.12}$$

40.4 Case Studies

40.4.1 Energy Management and Topology Maintenance

A number of alternative power minimization methods act above the MAC layer powering off redundant nodes' radios in order to expand the battery lifetimes. For example, AFECA [19] trades off energy consumption and the quality of the message delivery services based on the application requirements. GAF [20] 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. Because 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 nodes awake 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 compared to an unmodified ad hoc routing protocol, GAF also suggests that network lifetime increases proportionally to node density. On the other hand, a significant performance bottleneck can be easily created by grids that contain very limited number of nodes. Moreover, sometimes it is acceptable to let all nodes in some grids sleep (e.g., the boundary nodes) in order to reduce energy further. However, this situation cannot be recognized by GAF.

SPAN is a power-saving, distributed, randomized coordination approach [1] that preserves connectivity in wireless networks. The work presented in Koushanfar and colleagues [9] 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 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 Kravets and Krishnan [12] 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 [17] 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 multihop 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 up the node in case of an interesting event.

40.4.2 $(MI)^2$

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. This section introduces the 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, key assumptions are first summarized and typical sensor network optimization problems that will serve as illustrative examples briefly described. After that, an explanation is offered on how to apply the $(MI)^2$ strategy in a systematic way during each of the four phases of a localized algorithm: information gathering, system structuring, optimization mechanism, and result dissemination. Key insights and key

trade-offs in designing localized algorithms are described. The realization of such algorithms on a number of typical sensor network tasks, such as routing and minimum spanning tree, is illustrated.

Given a network, 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: (1) it is necessary to minimize storage requirements at each node; (2) nodes go to sleeping mode from time to time in order to minimize the energy consumption [2, 16]; and (3) 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 in the subset or has a neighbor in the subset. The importance of the selected problems for wireless ad hoc networks 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 [18].

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

40.4.2.1 Phase 1: Information Gathering

The information gathering (IG) phase is where the inputs to the procedure are prepared. If the information from multiple nodes is needed, routing between the nodes and the order in which nodes are visited and information 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 must select the next node to be visited or 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 a greedy local view in which one considers which nodes can be contacted in a few hops if a particular node is visited next.

When considering the objective function, one can 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 a node that is likely to have many unvisited neighbors as the next node. In this case, the amount of obtained information is maximized. Another alternative is to visit a node that has a large unexplored area within its communication range with a high likelihood of containing nodes relevant for optimization.

For example, in shortest path routing, one can always contact the node 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 others. 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 because 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.

40.4.2.2 Phase 2: System Structuring

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

According to the $(MI)^2$ paradigm, two principles for selection of computation centers are followed. The first is to assemble enough information initially to conduct at least part of the computation meaningfully as soon as possible. This point is particularly well illustrated on MRA [9] and exposure tasks [13]. The second principle is always to 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 in which 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, it is necessary to obtain information about a large neighborhood for each node before the optimization mechanism is started.

40.4.2.3 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, strong interdependence exists 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 algorithms require only one pass through all inputs, but 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 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 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 starting the execution of Kruskal's algorithm, it is necessary 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 consider nodes adjacent to already explored nodes. This must be done in such a way that communication requirements are reduced. Other optimization paradigms 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 an efficient localized algorithm, one must develop a new optimization mechanism and, therefore, a new centralized algorithm that operates locally and with the partial information.

40.4.2.4 Phase 4: Information Dissemination

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

40.4.3 Solving ILP Problems by $(Mi)^2$-Based Paradigm

To demonstrate the wide application range of a paradigm for designing localized algorithms, apply it to a set of specific problems that can be specified and solved using a particular optimization solving strategy.

This subsection presents an $(MI)^2$-based approach for solving an instance of a problem specified as an integer linear program. ILP is a widely used procedure for specifying and solving combinatorial optimization problems. ILP formulations are readily available for a large variety of combinatorial optimization problems, or they are easy to develop [14]. In particular, in a special case of ILP, called 0-1 ILP, all variables must be assigned to one of two binary values [14]. For example, all problems discussed in this chapter 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; the objective function and constraints must be linear. Note that, if the requirement that variable must be integers is removed, ILP reduces to a linear program (LP) that also has a wide range of applications [15]. An ILP defined over a set of variables x_i has the following standard form:

$$\text{Max } E^T.\mathbf{X} \tag{40.13}$$

$$\text{such that: } A^T.\mathbf{X} \quad B,\ C^T.\mathbf{X} = D \tag{40.14}$$

where A, B, C, D, and E are matrices composed of real constants, and $\mathbf{X}$ 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.

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 of each neighbor, but 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 a sensor network in order to reduce power consumption. Finally, each node must be informed about the value of all variables x_i important to it.

The $(MI)^2$-based approach for locally solving an ILP instance is based on the following observation and intuition: a particular value can be assigned to a particular variable x_i, only after information is obtained about all constraints that contain x_i. Furthermore, it is advantageous first to resolve variables that are components of the most difficult (strict) constraints. Also, it is important that at the time of assigning a particular value to a variable, as much information as possible is available about all constraints that contain the variables contained by the constraint under consideration. In order to maximize the objective function, it is important to assign high values to variables with high coefficients and to keep estimating the highest possible value of the OF in view of the already observed constraints.

One can envision two approaches with respect to the relationship between the OF and constraints: optimistic (in which it is preferable to maximize the objective function at potential danger that later some constraints will become unable to be satisfied), and pessimistic (in which constraints are favored at the expense of the OF). The $(MI)^2$-based localized ILP procedure is summarized using the pseudocode presented in Figure 40.2.

The IF is weighted sum of resolving power of the information available at the node and resolving power of its neighbors. The weighs of neighbors to a node are scaled by the average number of neighbors

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

from already visited nodes. Resolving power is proportional to reduction in information uncertainty, according to the classical information theoretical definition.

The optimization mechanism used is based on the maximally constrained, minimally constraining principle. Essentially, one tries to assign each variable in such a way that it resolves a maximal number of constraints or increases the 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.

40.4.4 GPSR

Routing is one of the fundamental tasks in wireless networks. Although one can envision a number of different types of routing, the focus here is on a case in which a single message must be sent from a node to another node. Only one localized routing algorithm will be considered so that it can be described and analyzed in sufficient detail.

Karp and Kung [8] have developed a stateless routing protocol for wireless networks: greedy perimeter stateless routing (GPSR). 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 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 only on the positions and 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 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 traditional routing protocols and is much more adaptive to mobility.

GPSR protocol has two phases: greedy forwarding and perimeter forwarding. Greedy forwarding refers to the phase in which 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 a situation in which 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 than it does. In this case, 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 an alternative route around and eventually converges 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 conceptually (and from implementation point of view) very simple and clean. 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 because the scope of the problem considered is limited with respect to available information. In addition, there is no guarantee that GPSR will eventually converge to the destination. Situations exist in which forwarding phase and perimeter phase oscillate within a set of nodes and never converge on the correct destination. It is also difficult, if not impossible, to see how to generalize the approach when additional information is available to a three-dimensional case, or in the presence of obstacles.

40.5 Analysis

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

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 for several reasons. First, it is not easy to identify which properties of the algorithms are interesting and important. Even when these properties are identified, it is often unclear how to define each of them exactly. In addition, it is often difficult to calculate or measure these properties. For example, some 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 priori* what should be the representative and realistic properties of instances. Finally, one can consider localized algorithms as generalizations of on-line algorithms in which the designer has an impact on information that 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 the amount of energy spent on communication. 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 nodes as possible 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 even more on the targeted applications, computation or some other components may have the dominant role.

The primary constraint is to achieve the user-requested level of optimality and/or accuracy. Because of 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 the protocols as the size of the network increases. Although many wireless sensor networks will be of limited size, scaling localized algorithms is already widely studied. A better way to evaluate localized algorithms for limited-sized networks is probably the development of benchmarks. Unfortunately, of the very few benchmarks available at present, all are synthetic.

In addition to these three metrics, amenability to provide fault tolerance, satisfy real-time constraints (such as throughput and latency), maintain privacy and security, and facilitate mobility will also be 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 a Pareto optimal curve. In the computer science literature, it is more common to take one metric as the optimization goal and others as constraints.

40.6 Protocols and Distributed Localized Algorithms

This section briefly discusses the distributed localized algorithms in which more than one thread of computation is executed at the same time. Distributed localized algorithms have a number of advantages in terms of their ability to respond faster to changes in the environment and the network, fault tolerance, and their resiliency against security attacks. First the desiderata for protocols that govern the execution of distributed localized algorithms are stated. After that, one generic approach is presented for development of protocols for distributed localized algorithms [9].

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 $v1$ 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 add an overhead as low as possible to the network, especially in terms of its power consumption and communication overhead.
- *Latency.* Higher latency in putting a node into sleeping mode implies more idle energy consumption. Also, 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 so that the faults in any number of nodes cannot corrupt its functionality.

Koushanfar et al [9] have developed an approach termed "distributed token mechanism" that attempts to fulfill the stated requirements. A token indicates that the node has control of the local flow of the sleeping procedure. At each point of time, more than one token can be present in the network to comply with the concurrency requirements. A token is generated by an awakened 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 acting 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 40.3; a node v_i that is not already locked by any other nodes considers running the sleeping procedure (steps 1 through 4) and therefore generates a token. A node that has slept before generates the token at a random time r_i within the interval ($0 < r_i < r\max$) (steps 5 through 9); a node that has already changed its state into sleep at least once generates a token as soon as it wakes up (steps 10 and 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).

The random initiation time (r_i) assigned to each node in the beginning of the procedure serves the purpose of avoiding simultaneous requests on the use of mutual resources. Because the sleep intervals are assigned independently to nodes depending on the power and topology of the neighborhood, the

```
Procedure Distributed Token mechanism
1.  at ∀ node v_i,
2.  {
3.      while (node v_i is not locked by another node)
4.      {
5.        if (never have slept before)
6.        {
7.           set a random initiation time r_i (0 < r_i < r_max);
8.           generate a token at the time r_i;
9.        }
10.       else {
11.            generate a token as soon as v_i wakes up;
12.            lock all the unlocked nodes in the v_i's scope;
13.           select the best node to sleep;
14.           select the sleep interval for the sleeping node;
15.           announce the decision in the neighborhood;
16.           unlock the locked nodes;
17.       }
18      }
```

FIGURE 40.3 Pseudocode for the distributed token mechanism procedure.

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 not to be concerned about the collisions in the network because they rely on the network's MAC layer to resolve any such conflicts.

40.7 Pending Challenges

This section outlines 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, one can identify some major trends. Future research directions are classified into two broad categories related to: (1) the conceptual novelties for developing localized algorithms; and (2) optimization and algorithmic techniques. 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 use for actuator-based system, are 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. When it is difficult to define a single widely applicable model, such as in parallel computing, progress is much slower. Currently, several models have been proposed for wireless ad hoc networks, including that of Zonoozi and Dassanayake [21]. However, it seems that the completely random nature of these models makes 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 must model at least four aspects of the systems: computation, communication, storage, and sensing.

Future algorithmic techniques can be naturally classified into two groups: one is related to design and the other is related to the analysis of localized algorithms. Design-related issues include the development of new paradigms that will facilitate systematic creation of localized algorithms, in particularly data collection and dissemination. An example of this is the maximally informing and maximally informed paradigm [11].

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

Obviously, a strong correlation exists between how nodes are deployed and performances of localized algorithms. It is easy to see that different localized algorithms are best suited for different wireless sensor network organizations. Interestingly, this topic has not been addressed. In particular, 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, 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 exposure coverage [13].

Another side of the coin for localized algorithm development is the analysis of localized algorithms. Soon many activities will be conducted to define and develop scalable algorithms that are scalable not only 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 in which one can decide which piece of information to obtain next. Competitive analysis of on-line algorithms has been a widely studied topic; it will be important for localized algorithms as well.

From a 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 analyze localized algorithms using statistical and perturbation analysis [4, 5].

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 in which software development is conducted and it will have a high impact in sensor networks.

Acknowledgment

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

References

1. Chen, B. et al. Span: an energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks, *Int. Conf. Mobile Computing Networking (MOBICOM)*, 85, 2001.
2. Estrin D. et al. Next century challenges: scalable coordination in sensor networks, *Int. Conf. Mobile Computing Networking (MOBICOM)*, 263, 1999.
3. Goemans, M.X. and Williamson, D.P. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming, *J. Assoc. Computing Machinery*, 42, 1115, 1995.
4. Grossglauser, M., and Tse, D.N.C. Mobility increases the capacity of ad hoc wireless networks, *IEEE/ACM Trans. Networking*, 10, 477, 2002.
5. Grossglauser, M., and Vetterli, M. Locating nodes with ease: mobility diffusion of last encounters in ad hoc networks, *Annu. Joint Conf. IEEE Computer Commun. Soc. (INFOCOM)*, 1954, 2003.
6. Hochbaum, D., Ed. *Approximation Algorithms for NP-hard Problems*, PWS Publishing Company, 1997.
7. Johnson, D. Approximation algorithms for combinatorial problems, *J. Computer Syst. Sci.*, 9, 256, 1974.
8. Karp, B. and Kung, H.T. GPSR: greedy perimeter stateless routing for wireless networks, *Int. Conf. Mobile Computing Networking (MOBICOM)*, 243, 2000.
9. Koushanfar, F. et al. Low power coordination in wireless ad hoc networks, *Int. Symp. Low Power Electron. Design*, 475, 2002.
10. Koushanfar, F. et al. Algorithms for resource discovery in wireless networks, unpublished manuscript, 2003.
11. Koushanfar, F. et al. Maximally-informing and maximally-informed algorithms for wireless networks, unpublished manuscript, 2003.
12. Kravets, P. and Krishnan, P. Application-driven power management for mobile communication, *Wireless Networks*, 6, 263, 2000.
13. Meguerdichian, S. et al. Exposure in wireless ad hoc sensor networks, *Int. Conf. Mobile Computing Networking (MOBICOM)*, 139, 2001.
14. Nemhauser, G.I. and Wolsey, L.A. *Integer and Combinatorial Optimization*, John Wiley & Sons, 1988.
15. Papadimitriou, C. and Steiglitz, K. *Combinatorial Optimization: Algorithms and Complexity*, Prentice Hall, Englewood Cliffs, NY, 1982.
16. Rabaey J.M. et al. PicoRadio supports ad hoc ultra-low power wireless networking, *Computer Mag.*, 42, 2000.

17. Schurgers, C., Tsiatsis, V., and Srivastava, M. STEM: topology management for energy-efficient sensor networks, *IEEE Aerospace Conf.*, 78, 2002.
18. Stojmenovic, I., Seddigh, M., and Zunic, J. Dominating sets and neighbor elimination based broadcasting algorithms in wireless networks, *IEEE Trans. Parallel Distributed Syst.*, 13, 14, 2002.
19. Xu, Y., Heidemann, J., and Estrin, D. Adaptive energy-conserving routing for multihop ad hoc networks, technical report *527*, USC/Information Science Institute, October, 2000.
20. Xu, Y., Heidemann, J., and Estrin, D. Geography-informed energy conservation for ad hoc routing, *Int. Conf. Mobile Computing Networking (MOBICOM)*, 70, 2001.
21. Zonoozi, M. and Dassanayake, P. User mobility modeling and characterization of mobility patterns, *IEEE J. Selected Areas Commun.*, 15, 1239, 1997.