Chapter 1

DISSEMINATION PROTOCOLS FOR LARGE SENSOR NETWORKS

Fan Ye, Haiyun Luo, Songwu Lu, Lixia Zhang

UCLA Computer Science Department
Los Angeles, CA 90095-1596

yefan,hluo,slu,lixia@cs.ucla.edu

Abstract Dissemination protocols in a large sensor network typically take a data-centric paradigm in which the communication primitives are organized around the sensing data instead of the network nodes. A user expresses his interest in a given data type by specifying a query message, which then propagates through the network. When detecting a nearby stimulus that matches the query type, sensors generate data reports which traverse the network to reach the interested user. The large scale of a sensor network, the resource constraints of each node in terms of energy, computation and memory, and potentially frequent node failures all present formidable challenges to reliable data dissemination. The protocols must be energy efficient, robust against failures and channel dynamics, and be able to scale to large networks. In this chapter, we provide a brief overview and critique of the state-of-art dissemination protocols and discuss future research directions.

In this chapter, we focus on the design of data dissemination protocols for a large-scale sensor network. In such networks data flows from potentially multiple sources to potentially multiple, static or mobile sinks. We define a data source as a sensor node that detects a *stimulus*, which is a target or *an event of interest*, and generates data to report the event. A *sink* is defined as a user that collects data reports from the sensor network. In the example of battlefield surveillance, a group of soldiers, each of them serves as a sink, may be interested in collecting tank location information from a sensor network deployed in the battlefield. The sensor nodes surrounding a tank, which acts as a stimulus, detect it and generate data reports. These reports will be forwarded by intermediate sensors to each of the sinks.

The unique characteristics of wireless sensor networks make the problem of data dissemination challenging and different from conventional wired Internet or wireless ad-hoc networks. Three main design challenges are:

- **Network scale** A sensor network may have thousands or even hundreds of thousands of nodes to cover a vast geographical area. Each sensor node is a potential data source, and multiple sinks may exist. In contrast, a conventional ad-hoc network has no more than a few hundred nodes in general.

- **Constrained resources at each sensor node** To reduce cost and increase spatial sensing coverage, sensor nodes are typically simple and inexpensive devices manufactured according to the economy of scale. Compared to the typical notebook/desktop computers that serve a conventional ad-hoc network or wired Internet, sensors tend to have limited battery energy, relatively low-end CPUs, and constrained memory size. Once the battery is depleted, a node may be considered useless. Recharging nodes is difficult due to the ad hoc deployment and the large node population.

- **Unpredictable node failures and unreliable wireless channel** The limited energy supply and small size make it infeasible to equip nodes with powerful radios and large antennas for high communication channel quality. Potential environmental obstacles and interferences further exacerbate the situation, making the wireless communication channel used by the sensor nodes prone to errors. In a harsh environment, sensor nodes themselves may fail or be destroyed unexpectedly. These operating conditions can be much worse than those of conventional ad hoc wireless networks.

In the rest of this chapter, we first identify main design goals for dissemination protocols in large sensor networks. We next describe the state-of-the-art protocol designs by organizing them into two categories: approaches without hierarchy and the hierarchy approach. We compare their main differences, and provide critiques on their possible limitations. We then elaborate on related issues of exploiting data naming and location awareness and providing real-time delivery. We further identify possible future directions and finally conclude the chapter. It should be noted that there may exist other small-scale, human-planned sensor networks, which are not the focus in this chapter. We are mainly concerned with large sensor networks made of simple and inexpensive nodes.

1. Design Goals and Solution Roadmap

Generally speaking, there are three desired properties for data dissemination protocols in a wireless sensor network:

- **Scalable and distributed solution** The dissemination protocol should scale with a large number of nodes, multiple sources and sinks.

- **Energy efficiency** Since energy is the most critical resource, the protocol should minimize computation and especially communication operations.

- **Robustness** The protocol should be able to work with highly dynamic, error-prone wireless communication channels, as well as fallible nodes that may fail due to damage or energy depletion.

The design of data dissemination protocols often take into account other desired properties as well, possibly with lesser degree of importance. For example, mission-critical applications may require that the report delivery be timely and reliable.

Note that some of the above goals may be in partial conflict with each other. For example, a distributed design, which can scale well, may not achieve the optimal energy efficiency. The robustness of the protocol is often realized through inter-node communications which consumes additional energy. Given an application scenario and network setting, the protocol design seeks the best tradeoff between conflicting metrics.

Solution Roadmap

Dissemination protocols in wireless sensor networks typically take a *data-centric* design paradigm, where the communication primitives of sensor networks are organized around the sensing data instead of network nodes as in either wired or conventional wireless networks. The operations of the dissemination protocol typically take a two-phase, "query-reply" process. The process is usually initiated by a sink, which sends query messages towards a geographical region of interest or the entire network, rather than a specific node. The query carries the sink's interests such as stimulus types and possibly the frequency it intends the data to be delivered back. The query message informs sensor nodes in the target region to send data reports back to the sink when the interested stimuli are detected. Data reports are tagged with the location, type and detection time of the stimulus, not necessarily with the detecting node's identifier. During the process, the protocol may install dissemination states in intermediate sensor nodes in order to guide future data delivery.

Such application-driven design distinct sensor networks from conventional wired and and wireless network routing. The unique characteristics of a sensor network also offer several unique opportunities for its protocol design:

- **Application semantics** Because the data dissemination protocol is data centric, it can exploit application semantics to improve its performance. For example, to reduce communication cost it is possible to aggregate data reports based on their common semantics and to do in-network data processing . A data centric design further allows to collapse multiple layers of the protocol stack and reduce inter-layer communication overhead.

- **Location awareness** Sensing application typically requires that each node know its (possibly approximate) location through GPS or other location services. The protocol design can exploit this extra information to achieve better scalability and efficiency.

- **Stationary nodes** Sensor nodes typically remain stationary after deployment. Therefore a dissemination protocol does not need to incorporate sophisticated mobility management component and may greatly simplify its operation compared to that of ad-hoc network routing protocols.

- **Dense deployment** To enhance reliability and system life time the initial deployment of a sensor network may have high node density. The scale and dense deployment of a sensor network also offer new opportunities for protocol design. Though each individual sensor has stringent resource constraints, collectively multiple sensors help to achieve better system robustness by harnessing the node population. The protocol can prolong the system operation lifetime via maintaining a small set of working nodes and letting others doze off.

Data dissemination protocols in sensor networks can be broadly classified into two categories: those protocols that operate without building a hierarchy among the sensor nodes in the network, and those protocols that build and maintain a communication hierarchy to help scale data dissemination. In the first category of protocols, two popular approaches are reverse-path-based forwarding and cost-field-driven dissemination. In the second category, the protocol builds a communication hierarchy by applying clustering or exploiting location information. We will discuss all these approaches in the following sections.

Table 1.1 summarizes the different data dissemination protocols in sensor networks. It shows the basic approach adopted by each protocol and the functionalities achieved.

Table 1.1. **Summary of Data Dissemination Protocols for Sensor Networks**
(supported properties or functionalities: 1. multiple sinks 2. sink mobility 3. load balancing 4. robustness 6. real time 7. in-network processing 8. query dissemination to targeted regions)

Protocol	Properties	Basic Approach	Details
DRP [1]	7	Reverse path forwarding	per-sink tree; use semantic information
Directed Diffusion [8]	1,7	Reverse path forwarding	shared mesh; use semantic information.
GRAd [13]	4	cost-field	receiver-decided
GRAB [16]	4	cost-field	receiver-decided; use credit to control forwarding redundancy
ReInForm [2]	4	cost-field	receiver-decided; sender calculates forwarding probabilities for receivers
EAR [14]	3	cost-field	sender-decided
TTDD [12]	1,2	hierarchy	virtual geographic grid per source
LEACH [6]	3	hierarchy	two-hop data delivery, a cluster head sends to the sink directly
GPSR [9]	1	Geographical forwarding	uses planar graph to bypass "holes"
GEAR [17]	3,8	Geographical forwarding	combines location and energy to choose next hop
SPIN [7]	7	Flooding	use semantic data descriptor to reduce redundant transmissions
RAP [11]	6	Geographical forwarding, virtual "velocity"	schedule packets using priority queues at each node
SPEED [4]	6	Geographical forwarding, virtual "velocity"	explicitly control the delay at each hop

2. Reverse Path Forwarding

Reserve path forwarding is one general approach to data dissemination in sensor networks and representative protocols include Declarative Routing Protocol (DRP), and Directed Diffusion (DD) protocol. In this approach, data reports flow in the reverse direction of query propagation to reach a sink. The sink sends out a query message that expresses its interest, possibly using flooding, throughout the network. Whenever a node receives a query from a neighbor, it sets up a forwarding state in the form of a *vector*, pointing from itself

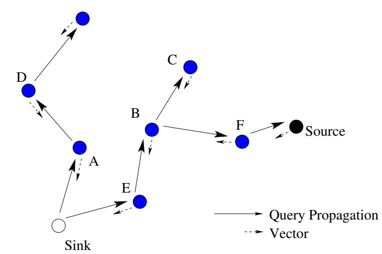

Figure 1.1. Query propagation sets up routing state at intermediate nodes in the form of vectors in reverse direction to guide data back to the sink.

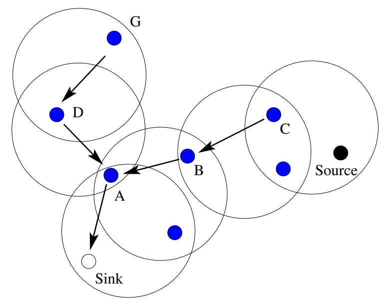

Figure 1.2. Routing tree building by query propagation. Big circles denote neighborhoods. Only one forwarding node is needed in each neighborhood.

to the neighbor, i.e., indicating the reverse data path. When a data report is generated by a source, it goes along the direction of the vector from one node to the next, until it reaches the sink. The protocols in this category differ in their techniques to achieve scalability, energy efficiency and robustness. In the following, we will discuss three such protocols: DRP, DD and its variant – resilient multipath directed diffusion.

Declarative Routing Protocol

Declarative Routing Protocol (DRP) [1] achieves energy efficiency through in-network aggregation. It uses reverse path forwarding to establish a routing tree for each sink to receive data reports for all other nodes. It also exploits the application-specific property of sensor networks to build a cross-layer declarative service for better efficiency.

DRP organizes nodes in the communication range of each other into *neighborhoods*. One node may belong to multiple neighborhoods at the same time. A sink sends out queries using a distributed algorithm, similar to an expanding-ring search, which ensures loop-free forwarding. As the query is disseminated from one neighborhood to the next, a weighted sum of several factors is used to select to which node of the next neighborhood the query should be sent. The factors include *reachability* (the number of nodes that can be reached), *diversity* (the uniqueness of the set of nodes that can be reached), *directionality* (the proximity to the path to the target region), *remaining energy* and *link quality* in terms of SNR (Signal-to-Noise Ratio).

After the query traverses all neighborhoods, the sink obtains a routing tree rooted at itself, with vectors pointing in the reverse direction of query propagation. Figure 1.2 shows an example of such a tree. The routing vectors are maintained at selected nodes A, B, C, D and G. A data source broadcasts its data report in its local neighborhood. Any neighbor that finds a match between the data and the user's interest should forward data back along the reverse path to the sink.

DRP maintains the consistency of routing state at different nodes using a sequence number. Each time a routing record is updated, the associated sequence number is incremented. Neighbors detects inconsistency by comparing the sequence numbers of their cached records and requesting the latest version through NACK. To improve efficiency, DRP proposes information sharing across multiple layers in the protocol stack. A new entity called *declaration service* spans the medium access, network and transport layers, and interacts with the sensor applications directly. Such a cross-layer design allows sharing of state and reduces cross-layer function calls, thus trading off system modularity for performance.

Directed Diffusion and Its Variants

Similar to DRP, Directed Diffusion [8] also uses reserve-path to guide data and follows the data-centric paradigm. However, Diffusion focuses more on scaling to multiple sinks and robustness under network dynamics.

Diffusion builds a shared mesh to deliver data from multiple sources to multiple sinks, rather than per tree each sink. To scale to multiple sinks, nodes do not keep per sink state. When receiving a query from a neighbor, a node does not differentiate whether the neighbor is a sink or one that merely forwards the query. It always sets up a vector (called "gradient") pointing from itself to the neighbor to direct future data flow, without associating the vector to any particular sink. This is different from the per sink state in DRP.

After a sink floods its query across the network, two vectors are established between each pair of neighbors, pointing to each other. The initial batch of data follow such vectors, effectively flood through the network to explore all possible paths. Each node has a cache to detect loops and drop redundant packets (See Figure 1.3 for an example).

After receiving the initial batch of data reports along multiple paths, a sink uses a mechanism called *reinforcement* to select a path with the best forwarding quality. It sends a positive reinforcement message to the neighbor from which it receives data with the minimum delay (or other metrics, depending on applications' needs). It may also send negative reinforcement messages to other neighbors to de-activate low-quality paths. Such positive and negative reinforcements are repeated upstream towards data sources. Gradually, data reports flow only along the best path (Figure 1.4 shows an example where the data forwarding path F$\rightarrow$B$\rightarrow$E is reinforced).

To handle network dynamics such as node failures, each source floods reports periodically at a rate much lower than normal data, to maintain alternative

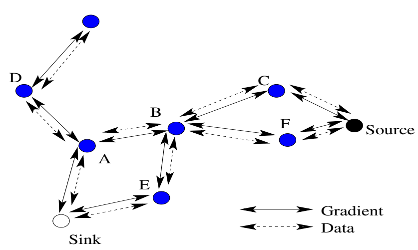

Figure 1.3. Interest flooding and multipath data forwarding

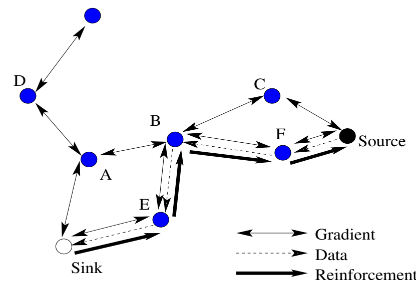

Figure 1.4. Reinforcement

paths. So nodes can reinforce other paths to route around failed nodes. This enhanced robustness is achieved at the cost of additional energy and bandwidth consumption due to flooding. In-network processing can be integrated into Diffusion. Each intermediate node actively aggregates queries carrying the same interest and reports containing correlated data. We will further elaborate in-network processing later.

Resilient multipath Directed Diffusion [3] enables quick path repairing in case of node failures by maintaining multiple backup paths together with the primary data forwarding path. These backup paths are used to quickly route around failed nodes. One method is to construct *braided* multipath. Every other node along the primary data forwarding path sends an *alternate path reinforcement* message to its next-preferred neighbor. Then the neighbor sends out a primary path reinforcement message to its most-preferred upstream neighbor. A node that is already on the primary forwarding path silently drops the alternate path reinforcement message. The final effect is that a node can immediately start pulling data from an alternative upstream neighbor if the current one fails, without waiting till the next data flooding.

Summary

Reverse path based forwarding approaches naturally integrate in-network processing. Whenever multiple reports from different reverse paths merge at a node, the node can process and aggregate data based on their semantics. This in-network processing saves energy and bandwidth resources. Both DRP and Diffusion adopt this feature to achieve better efficiency.

Since the vector state kept in nodes depends on the local connectivity, overhead has to be paid to maintain states in the presence of node failures. DRP uses periodic beacon exchanges among neighbors to maintain state, whereas Diffusion uses reinforcement and periodic low-rate data flooding to adapt.

These complexities and overhead are inevitable in reverse path approaches because the vector state has to be updated with the changes in connectivity.

Sink mobility is not well addressed in this approach. In DRP, the reverse path forms a tree rooted at a sink. Any change in the sink's location requires changes of the vector state in nodes. The sink needs to re-flood its queries to update the vector states in nodes to receive data reports when it roams. The same is true for Directed Diffusion.

3. Cost-field Based Forwarding

Cost-field based forwarding offers an alternative approach to data dissemination. In this approach, the forwarding states stored in an intermediate node take a form other than vector: a *scalar* denoting the node's "distance" to a sink. The scalar value denotes the total cost, which can be measured by the hop count, energy consumption or physical distance, to deliver a report from this node to the sink. The cost scalar of all the nodes forms a *cost-field*, where nodes closer to the sink having smaller costs and those farther away having higher costs. The cost value at each node is directionless, different from the vectors used in reverse-path forwarding. However, the cost field has implicit directions. A data report can flow from a node with a higher cost to a neighbor with a smaller cost, similar to water flowing from a hill to a valley, eventually reaching the sink (See Figure 1.5 for an illustration). A number of protocols, including Gradient Routing (GRAd) [13], GRAdient Broadcast (GRAB) [15, 16], ReInForM [2], Energy Aware Routing (EAR) [14] fall into this general approach.

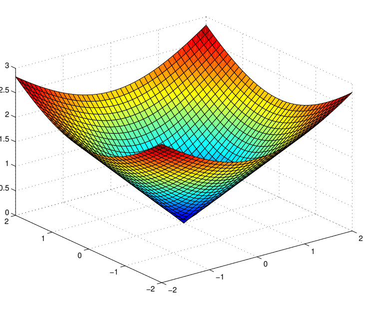

Figure 1.5. The "shape" of the cost field is like a funnel, with the sink sitting at the bottom. Data follow the decreasing cost direction to reach the bottom of the cost field, which is the sink.

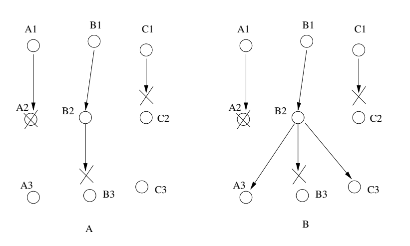

Figure 1.6. A: any single node failure or packet loss ruins a single path; B: interleaving paths can recover each other from failures or packet losses

Fundamentals of Cost-Field Approach

Cost-field approach has two main steps: setting up the cost field for a sink and how a node makes forwarding decisions. A sink sets up its cost field by flooding a query packet (ADV) similar to distance-vector routing. The packet carries a cost value, initially set to 0 by the sink. A node N_i receiving an ADV packet from node N_j calculates its cost as

$$C_i^{new} = C_j + C_{i,j}, \tag{1.1}$$

where C_j is node N_j's cost carried in its ADV packet, $C_{i,j}$ is the link cost to send a packet from N_i to N_j. If C_i^{new} is less than N_i's current cost C_i, it sets C_i to C_i^{new}; otherwise it does not change C_i. Then N_i broadcasts an ADV carrying its own cost C_i. N_i keeps the smallest one if it receives multiple ADVs carrying different costs from different neighbors during the flooding,

The cost at a node provides a sense of its "distance" to the sink. Nodes "closer" to the sink have smaller costs; the cost-decreasing direction points towards the sink. As long as a data report follows such directions, at each step it comes closer to the sink, and will not come back to the same node. Thus no loop can form. The vector in reserve path forwarding, however, does not provide any sense of distance. Vectors at different nodes may form loops, which have to be detected explicitly. The cost field is per-sink based. Each sink has to set up its own cost field to receive reports. A node keeps a cost for each sink. In the reverse-path-based approach, multiple sinks may share the same segments of data delivery paths. A node does not necessarily need to keep per-sink state.

In the cost-field-based approach, there are two ways a node makes forwarding decisions: *receiver-decided* and *sender-appointed*. In the former method, a sender includes its cost in a report and broadcasts the report, without addressing to any specific neighbor. Each receiver decides whether it wants to forward by comparing its own cost to that of the sender. Only receivers with costs less than that of the sender may forward the report. In the latter method, a sender decides which neighbor forwards the report and unicasts the report.

The receiver-decided method enables robust data forwarding with unreliable wireless channels and fallible nodes. Because typically several neighbors have smaller costs than the sender, at each hop multiple nodes can receive and forward different copies of a report in parallel. As long as there exists one node that receives the report successfully, packet losses or node failure of other nodes on this hop do not disrupt data delivery. Moreover, a single upstream forwarding node may spawn several copies to multiple downstream nodes successfully, thus recovering the parallelism in forwarding. The actual paths different copies take are not disjoint, but rather, they *interleave* and form a mesh (See Figure 1.6 for an example). This "self-recovery" mesh proves highly robust against packet losses and node failures, as demonstrated by GRAB.

The sender-appointed method may also form such a interleaving mesh, but the sender has to keep states regarding which neighbors are alive, and which links still function. In a harsh environment where both node topology and wireless connectivity change dynamically, such states have to be maintained with added complexity and may not be updated as quickly as changes happen.

GRAd, GRAB and ReInForm are receiver-decided. They achieve robustness in data delivery, but differ in whether and how they control the redundancy of the forwarding mesh. EAR is sender-appointed; it focuses on load balancing among different nodes.

Receiver-decided protocols

GRAd uses hop count as the cost. A sink builds its cost-field by flooding a REQUEST message. A source broadcasts a report; all neighbors with smaller costs forward the report. This process is repeated, and multiple copies of the report travel within a forwarding mesh of intermediate nodes towards the sink.

GRAB is designed for a harsh environment with high channel error rate and frequent node failures. The cost each node keeps is the total energy needed to send a packet to the sink. It assumes that each node knows the energy needed to transmit a report to a neighbor. GRAB further enables flexible trade-off between robustness and energy consumption. It controls the "width" of the forwarding mesh using a packet-carried state, *credit* . When a source sends a report, it assigns a credit to the report. The sum of the source's cost and the credit is the total budget that can be consumed to send the report downstream to the sink. The report can take any route that has a total cost less than or equal to the budget, but not those of costs exceeding the budget. A larger credit allows for a wider forwarding mesh; if the credit is "0", the report can only be forwarded over the minimum-cost route from the source. By controlling the amount of credit, the source effectively controls the degree of redundancy in the forwarding mesh.

ReInForm controls the redundancy differently. It calculates the number of parallel forwarding nodes needed at each hop to achieve a specified end-to-end delivery success probability r_s. It takes hop count as the cost. Each node keeps a measurement of local channel error rate e, and divides its neighbors into three groups H^-, H^0, H^+, i.e., neighbors with one less, the same, one more hop count. A source with a cost of h_s calculates P, the number of forwarding neighbors needed, as a function of e, r_s, h_s. If P exceeds the number of neighbors in H^-, some nodes in H^0 have to forward; if it exceeds the total number in H^- and H^0, some nodes in H^+ are needed. In such cases the source calculates three values P^-, P^0, P^+, representing the number of paths each node in the corresponding group needs to create. It then broadcasts the report, including the state P^-, P^0, P^+, e, h_s. A receiver knows its group and checks the corresponding P: it forwards the report if $P \geq 1$, or with probability P if $P < 1$. Each receiver that decides to forward repeats the same procedure as the source. It recalculates the P values for its three local neighbor groups using its own cost and channel error, then broadcasts the report.

Sender-appointed protocol

EAR achieves load balancing among multiple paths to extend the system lifetime. A node makes forwarding decision probabilistically among neighbors. A node N_a keeps one energy cost $C_{a,i}$ for each of the neighbors N_i from which an ADV is received (as in Equation 1.1). It assigns a probability P_i inverse proportional to $C_{a,i}$ of each of these neighbors. When sending a report, it selects one of these neighbors using the probability P_i's. The cost included in its ADV packets is $\sum_i P_i C_{a,i}$, a weighted average of the $C_{a,i}$'s.

Different from the other three cost-field protocols, EAR is single path forwarding. The sender chooses only one neighbor for each report. Its main goal is not to ensure robustness, but to statistically distribute the amount of consumed energy among different nodes. A path consuming more energy is used inversely proportionally less frequent. Thus forwarding nodes consume energy at about the same rate over time.

Summary

Compared with the reverse-path-based approach, the cost-field approach enables mesh forwarding and receiver-decided delivery. By comparing its cost to that of a sender, multiple receivers can each decide whether it continues forwarding a packet or not. Loops cannot form when packets follow the direction of decreasing cost. Mesh forwarding greatly enhances robustness in the presence of unpredictable node failures and harsh channel conditions. The receiver-decided forwarding eliminates the complexity at each sender to maintain explicit path states.

Similar to reverse path based approach, cost-field base approach does not address mobile sinks well. Since the cost depends on the "distance" to the sink, it has to be updated when a sink moves. Cost-field based approach does not scale well with the number of sinks, either. Each sink needs a separate cost at every node, thus the amount of state increases proportionally to the number of sinks. In the reverse-path-based approach, the vector does not need to be associated with a particular sink (as in Diffusion). Thus multiple sinks can share the same segments of paths, at the cost of maintaining a cache at each node to detect loops.

4. Routing with Virtual Hierarchy

So far we have presented forwarding protocols that take the "flat" network model, in which every sensor node has the same functionality and no network hierarchy is formed during data dissemination. In this section, we present two designs that builds virtual hierarchy to address sink mobility and achieve energy efficiency, respectively.

Two-tier Data Dissemination

Sink mobility brings new challenges to large-scale sensor networking. Both reverse-path and cost-field based approaches require mobile sinks to continuously propagate queries to update sensor state throughout the network, so that all sensor nodes get updated with the direction of sending future data reports. However, frequent query dissemination from mobile sinks can lead to both increased collisions in wireless transmissions and rapid power consumption of the sensor's limited battery supply.

Two-tier Data Dissemination (TTDD) [12] aims at efficient and scalable sink mobility support. It considers a sensor network of homogeneous sensor nodes, but enforces a virtual grid infrastructure for query and data forwarding. In TTDD, the data source *proactively* builds a grid structure throughout the sensor field. In a two-dimensional sensor field, a source divides the plane into a *grid* of cells. Each cell is an $\alpha \times \alpha$ square. A source itself is at one crossing point of the grid. It propagates data announcements to reach all other crossings, called *dissemination points*, on the grid. The sensor closest to a dissemination point serves as a dissemination node. A dissemination node stores a few pieces of information for the grid structure: the data announcement message, the dissemination point L_p it is serving and the upstream dissemination node's location. It then further propagates the message to its neighboring dissemination points on the grid except the upstream one from which it receives the announcement. The data announcement message is *recursively* propagated through the whole sensor field so that each dissemination point on the grid is served by a dissemination node. Figure 1.7 shows a grid for a source B and its virtual grid.

With this grid structure in place, a query from a sink traverses two tiers to reach a source. The sink firstly floods its query within a cell. When the nearest dissemination node for the requested data receives the query, it forwards the query to its upstream dissemination node toward the source, which in turns further forwards the query, until it reaches either the source or a dissemination node that is already receiving data from the source (e.g. upon requests from other sinks). This query forwarding process lays information of the path to the sink, to enable data from the source to traverse the same two tiers as the query but in the reverse order (see Figure 1.8 for an illustration).

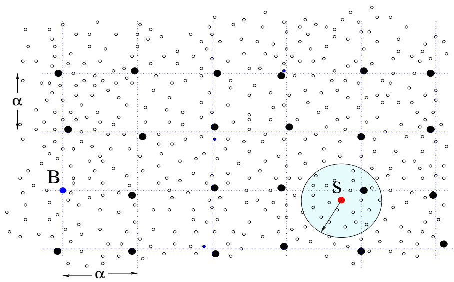

Figure 1.7. One source B and one sink S

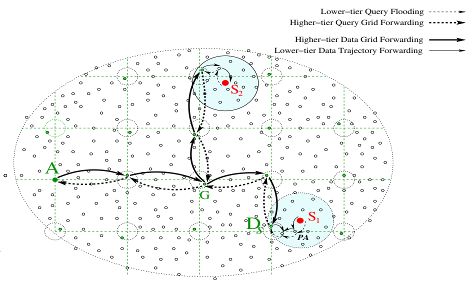

Figure 1.8. Two-tier query and data forwarding.

TTDD also makes explicit effort to maintain its grid. The grid maintenance is triggered on-demand by on-going queries or upstream updates. Compared with periodic grid refreshing, it trades computational complexity for less consumption of energy, which is a more critical resource in the context of large-scale sensor networks.

Low-Energy Adaptive Clustering Hierarchy

Low-energy adaptive clustering hierarchy (LEACH) [6] is designed towards energy-optimal network organization. It assumes that each sensor node be able to adjust its wireless transceiver's power consumption, thereby controlling the network topology. At a sensor node's maximum transceiver's power, it is able to communicate directly with the sink, which is fixed in certain location of the sensor field.

LEACH is motivated by the comparison on energy consumption between two forwarding algorithms. One is direct transmission between each sensor node and the sink. For distant sensor nodes, direct transmission to the sink requires a large amount of transmit power. However, direct transmission has the minimum energy consumption on reception, assuming the sink is not power-constrained. The other is minimum-transmission-energy (MTE). In MTE, rather than use one high-energy transmission, data messages go through n low-energy transmits and traverses n intermediate sensor nodes to reach the sink. MTE minimizes the energy consumption on transmission, however, the sensor network has to invest energy on reception. This amount of energy consumption is comparable to that on transmission given LEACH's radio model.

The analysis based on a certain network size and a first-order radio model results in LEACH's two-hop data dissemination, as a tradeoff between the transmission and receiving energy consumptions. Data messages from each sensor node are first transmitted to a local cluster head, and then forwarded to the base station using potentially long range radio. Periodically the system runs a randomized distributed algorithm to elect a number (determined a priori) of cluster heads. These cluster heads then advertise themselves to guide the clustering of the sensor networks. That is, a sensor node joins the cluster from which the strongest advertisement signal is received. Each sensor node reports itself to its cluster head. Based on cluster population, the cluster head creates a TDMA schedule telling each cluster member when it can transmit for intra-cluster transmission. LEACH also proposes to use CDMA spreading codes to decrease the inter-cluster interferences.

Summary

Compared with flat forwarding approaches based on reverse path and cost-field, nodes in hierarchy forwarding carry different functions. Some special nodes form a structure that helps to address new issues or achieve better performance. TTDD's design exploits the fact that sensor nodes are both stationary and location-aware, and use simple greedy geographical forwarding to construct and maintain the grid structure with low overhead. In TTDD, the virtual grid confines the impact of sink mobility. A sink does not need to flood a query across the entire network when it moves. Instead, it only needs to disseminate queries within a local cell to find a new data dissemination node. Only data forwarding within the sink's local cell is affected; the higher-tier grid forwarding changes only incrementally and remain the same most of the time. In LEACH, the presence of a cluster head node enables data aggregation within the cluster, in order to save energy consumed to transmit data to the base station.

Hierarchy forwarding addresses the mentioned issues at a cost: the complexities needed to maintain the structure. In TTDD, the grid has to be built and tore down; it needs to be repaired when a dissemination node fails. In LEACH, nodes need to agree on their cluster header and take turns to be the header. These complexities not only increase the overhead, but may lead to less robust operations.

5. Additional Approaches

In this section we describe a number of proposals that present other approaches to sensor network data forwarding. We first explain how two unique features of sensor networks, data-centric naming and location-awareness, can be exploited to improve forwarding efficiency. We then summarize protocols that seek to provide timely delivery.

Efficiency by in-network processing

The data-centric nature of a sensor network allows for semantics-based in-network processing . [5] advocates that low-level naming in sensor networks should be based on attributes related to the application, rather than network topology. The application-specific paradigm of sensor networks is in contrast to the Internet, where routers' processing capability is the main constraint and the network is shared by many applications. [5] further proposes a Diffusion-based architecture utilizing application-specific code (called "filter") that processes data in the network. Semantics-based matching rules are used to identify which sensing data should be processed by a given filter. Two examples of in-network data aggregation and nested query illustrate the benefit achieved by such an architecture.

SPIN [7] exploits data-naming to reduce the overhead of disseminating data from one node to the rest of the network. A node first sends data descriptors in the form of meta-data to its neighbors. A receiver that needs those data sends back a request, upon which the sender sends the requested data. SPIN reduces the redundancy in receiving data by this semantics-based negotiation.

Geographical forwarding

If each node knows its geographical location, this feature can be exploited to forward data to a given sink. The simplest method is greedy forwarding: each node always sends the packet to the neighbor that is closest to the destination. The problem with this simple approach is "void area," where the sender itself is closer to the destination than all its neighbors and the packet has to be forwarded backward temporarily to reach the destination. GPSR [9] solves this problem by forwarding a packet along the perimeter of a void area. It computes a planar topology where no two edges cross each other requiring only one-hop neighbor locations at each node. A packet follows the right-hand rule to circumvent the void area.

GEAR [17] exploits location information to address another issue of sending a query to all nodes within a target region. It first sends the query towards the centroid of the region; once the query reaches a node in the region, multiple copies are forwarded recursively to the sub-regions, until every node receives one copy. Different from pure geographical forwarding, GEAR uses a weighted sum of physical distance and residual energy when selecting its next-hop neighbor, thus achieving load balancing among nodes. It circumvents holes using a heuristic similar to the Learning Real Time A* algorithm in [10].

Real-time delivery

RAP [11] and SPEED [4] are two proposals providing real-time data delivery services in sensor networks. They are based on an idealized model that takes into account both the remaining lifetime before a data report expires and the remaining distance the report has to travel. Given the physical distance d to the destination and the lifetime t of a packet, a desired velocity $v = \frac{d}{t}$ can be calculated. If the packet travels along a straight line towards the destination at speed v, it will not miss the deadline. They both use greedy geographical forwarding, but differ in the exact mechanism used.

In RAP, the required velocity is updated at each hop to reflect the urgency of the report. A node uses multiple FIFO queues with different priorities to schedule reports of varying degrees of urgency. Each queue accepts reports of velocities within a certain range. RAP further adjusts the waiting and backoff times at MAC layer based on the priority of the report being transmitted, so that one with higher priority has greater probability of accessing the channel.

In SPEED, a node actively controls the data rate to avoid congestion and maintain a relatively stable relay-speed to each neighbor. The node measures the delay of sending a report to each neighbor using exponential weighted moving average. Given a report with velocity v, it computes the speed v_i of the report if neighbor N_i is chosen as the next hop. The sender chooses one neighbors with $v_i > v$ to forward the report. If no such a neighbor exists, the report is forwarded at a probability, which is the output of a proportional controller with the delivery miss ratios of neighbors as the input. This effectively reduces the traffic load on neighbors so that the desired relay speed can be achieved. Nodes in a congested region also feedback "back-pressure" messages upstream, so that data are detoured to bypass the region.

6. Future Directions

Service Differentiation. A sensor network application may require reports of different degrees of semantic importance. E.g., a forest fire sensing system needs both extreme temperature alarms and normal temperature readings. Reports notifying emergency events should be delivered with higher service assurance compared with normal "status" reports. This makes a case for service differentiation in query and data dissemination of large sensor networks.

There are several challenges to service differentiation in sensor networks. Each sensor has limited resources in terms of computational power, memory and energy supply. Therefore, many solutions proposed for the wired Internet, which require sophisticated state management, are not applicable. In a harsh environment, nodes are prone to failures. How to provide service differentiation that is robust to node failures poses another challenge. Furthermore, wireless bandwidth resource is location-dependent, not node-dependent, as in the

Internet. The solution has to address the unique issue of location-dependent resource management. Finally, many applications require that the solution scale to a large sensor population.

Security. There are many security aspects in data dissemination in large sensor networks, and current research on sensor network security serves a starting point. In many applications, each report message needs to be authenticated and encrypted, and such services have to be provided in a fully distributed manner. Moreover, even with security solutions deployed, malicious adversaries may subvert a limited number of sensors and operate behind the security defense. Compromised sensors may launch various attacks including injecting bogus report messages, and dropping or replaying valid reports. Such attacks may cause the sink to receive false events and miss real ones, drain the energy supplies of intermediate nodes, or even disrupt the functioning of the entire network system. How to build security solutions that are tolerant to compromised nodes poses a unique challenge. The resource constraints of sensors, such as computation, communication and energy, further aggravate the difficulty in designing such solutions.

Evaluation. Performance evaluation of dissemination protocols for sensor networks poses another research challenge. In most of the literature, the evaluation has been mainly based on simulations. Though simulations can provide useful results, they cannot characterize importance dimensions such as worst-case performance, optimality, and structural features such as stability and convergence. Some initial analytical techniques based on percolation theory, computational learning theory and stochastic processes have been reported in the literature, but much work needs to be done. In particular, most current analysis only provides asymptotic results for arbitrarily large networks or extremely dense topology, the middle ground of mid-sized network, which is in some sense the most interesting case from a practical perspective, remains wide open. Another aspect is to evaluate the protocols from a sensor network application perspective. In a sense, the application holds the ultimate key to effective data dissemination protocols in a sensor network.

7. Conclusion

Wireless sensor networks hold great promises in providing a new networking paradigm which allows for interacting with the physical world. Though each individual sensor may have severe resource constraints in terms of energy, memory, communication and computation capabilities, thousands or even hundreds of thousands of them may collectively monitor the physical world, disseminate information upon critical environmental events, and process the information on the fly. They enable potentially numerous, new applications such as forest fire monitoring, chemical and biological agent detection, soil pollution measurement and autonomous ocean ecology sampling. Toward this end, efficient data dissemination protocol serves as a critical component to realize a large-scale sensor network.

In this chapter, we have provided a survey on the state-of-the-art dissemination protocols for sensor networks and discussed some future directions. Our writeup intends to provide a brief overview of this active research area and serves as a tutorial to attract more researchers to further refine this critically important protocol of a sensor network.

References

[1] D. Coffin, D. V. Hook, S. McGarry, and S. Kolek. Declarative ad-hoc sensor networking. In *SPIE Integrated Command Environments*, 2000.

[2] B. Deb, S. Bhatnagar, and B. Nath. ReInForM: Reliable Information Forwarding Using Multiple Paths in Sensor Networks. In *28th Annual IEEE Conference on Local Computer Networks*, 2003.

[3] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly Resilient, Energy Efficient Multipath Routing in Wireless Sensor Networks. In *ACM MOBIHOC*, 2001.

[4] T. He, J. A. Stankovic, C. Lu, and T. Abdelzaher. SPEED: A Stateless Protocol for Real-time Communication in Sensor Networks. In *ICDCS*, 2003.

[5] J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin, and D. Ganesan. Building Efficient Wireless Sensor Networks with Low-level Naming. In *ACM Symposium on Operating System Principles (SOSP'01)*, 2001.

[6] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-Efficient Communication Protocols for Wireless Microsensor Networks. In *Hawaaian Int'l Conf. on Systems Science*, 2000.

[7] W. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive Protocols for Information Dissemination in Wireless Sensor Networks. In *ACM MOBICOM*, 1999.

[8] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. In *ACM MOBICOM*, 2000.

[9] B. Karp and H. T. Kung. Gpsr: Greedy perimeter stateless routing for wireless networks. In *ACM MOBICOM*, 2000.

[10] R. Korf. Real-time Heuristic Search. *Artificial Intelligence*, 42(2-3), March 1990.

[11] C. Lu, B. M. Blum, T. F. Abdelzaher, J. A. Stankovic, and T. He. RAP: A Real-Time Communication Architecture for Large-Scale Wireless Sensor Networks. In *Eighth IEEE Real-Time and Embedded Technology and Applications Symposium*, 2002.

[12] H. Luo, F. Ye, J. Cheng, S. Lu, and L. Zhang. TTDD: A Two-tier Data Dissemination Model for Large-scale Wireless Sensor Networks. *ACM/Kluwer Journal of Mobile Networks and Applications (MONET)*, 2003.

[13] R. Poor. Gradient Routing in Ad Hoc Networks. http://www.media.mit.edu/pia/Research/ESP/texts/poorieeepaper.pdf.

[14] R. C. Shah and J. Rabaey. Energy Aware Routing for Low Energy Ad Hoc Sensor Networks. In *WCNC*, 2002.

[15] F. Ye, S. Lu, and L. Zhang. GRAdient Broadcast: A Robust, Long-lived Large Sensor Network. http://irl.cs.ucla.edu/papers/grab-tech-report.ps, 2001.

[16] F. Ye, G. Zhong, S. Lu, and L. Zhang. GRAdient Broadcast: A Robust Data Delivery Protocol for Large Scale Sensor Networks. *ACM Wireless Networks (WINET)*, 11(2), March 2005.

[17] Y. Yu, R. Govindan, and D. Estrin. Geographical and Energy Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks. Technical Report UCLA/CSD-TR-01-0023, UCLA Computer Science Dept., May 2001.