Wireless Sensor Network Protocols

Mark A. Perillo and Wendi B. Heinzelman
Department of Electrical and Computer Engineering
University of Rochester
Rochester, NY, USA

1 Introduction to Wireless Sensor Networks

Efficient design and implementation of wireless sensor networks has become a hot area of research in recent years, due to the vast potential of sensor networks to enable applications that connect the physical world to the virtual world. By networking large numbers of tiny sensor nodes, it is possible to obtain data about physical phenomena that was difficult or impossible to obtain in more conventional ways. In the coming years, as advances in micro-fabrication technology allow the cost of manufacturing sensor nodes to continue to drop, increasing deployments of wireless sensor networks are expected, with the networks eventually growing to large numbers of nodes (e.g., thousands). Potential applications for such large-scale wireless sensor networks exist in a variety of fields, including medical monitoring [1, 2, 3], environmental monitoring [4, 5], surveillance, home security, military operations, and industrial machine monitoring. To understand the variety of applications that can be supported by wireless sensor networks, consider the following two examples.

Surveillance. Suppose multiple networked sensors (e.g., acoustic, seismic, video) are distributed throughout an area such as a battlefield. A surveillance application can be designed on top of this sensor network to provide information to an end-user about the environment. In such a sensor network, traffic patterns are many-to-one, where the traffic can range from raw sensor data to a high level description of what is occurring in the environment, if data processing is done locally. The application will have some quality of service (QoS) requirements from the sensor network, such as requiring a minimum percentage sensor coverage in an area where a phenomenon is expected to occur, or requiring a maximum probability of missed detection of an event. At the same time, the network is expected to provide this quality of service for a long time (months or even years) using the limited resources of the network (e.g., sensor energy and channel bandwidth) while requiring little to no outside intervention. Meeting these goals requires careful design of both the sensor hardware and the network protocols.

Medical Monitoring. A different application domain that can make use of wireless sensor network technology can be found in the area of medical monitoring. This field ranges from monitoring patients in the hospital using wireless sensors to remove the constraints of tethering patients to big, bulky, wired monitoring devices, to monitoring patients in mass casualty situations [6], to monitoring people in their everyday lives to provide early detection and intervention for various types of disease [7]. In these scenarios, the sensors vary from miniature, body-worn sensors to external sensors such as video cameras or positioning devices. This is a challenging environment in which dependable, flexible, applications must be designed using sensor data as input. Consider a personal health monitor application running on a PDA that receives and analyzes data from a number of sensors (e.g., ECG, EMG, blood pressure, blood flow, pulse oxymeter). The monitor reacts to potential health risks and records health information in a local database. Considering that most sensors used by the personal health monitor will be battery-operated and use wireless communication, it is clear that this application requires networking protocols that are efficient, reliable, scalable and secure.

To better understand why traditional network protocols are not suitable for these types of sensor network applications, in the remainder of this section we will categorize the unique features of sensor networks and the performance metrics with which protocols for sensor networks should be evaluated.

1.1 Taxonomy of Sensor Networks

As research in sensor networks has grown, so too has the range of applications proposed to make use of this rich source of data. Such diversity of sensor network applications translates to differing requirements from the underlying sensor network. To address these varying needs, many different network models have been proposed, around which protocols for different layers of the network stack have been designed. While there are many ways to classify different sensor network architectures, the following list highlights some fundamental differences in sensor networks that affect protocol design.

- *Data sink(s).* One of the most important aspects of a sensor network is the nature of the data sink(s). In some situations, the end user(s) may be embedded within the sensor network (e.g., actuator(s) that correct abnormalities in environmental conditions, access points that network with the outside world) or may be less accessible mobile access points that collect data once in a while (e.g., data collectors in the DATA Mules project [8] and in a sensor reachback scenario [9]). This distinction may be important, as efficient distributed data storage techniques may be effective in the latter scenario.

- *Sensor mobility.* Another classification of sensor networks may be made based on the nature of the sensors being deployed. Typically, it can be assumed that sensors are immobile; however, some recent sensor networks projects such as the ZebraNet project [10] have used mobile sensor nodes. Also, in military operations, additional sensors may be mounted on soldiers or UAVs to interact with a deployed sensor network. The mobility of sensors can influence protocols at the networking layer as well as those for localization services.

- *Sensor resources.* Sensor nodes may vary greatly in the computing resources available. It is obvious that memory and processing constraints should influence protocol design at nearly every level.

- *Traffic patterns.* Another important aspect to consider is the traffic generated on the network. In many event-driven applications, sensors may operate in a sentry state for the majority of time, only generating data traffic when an event of interest is detected. In other applications such as environmental monitoring, data should be continuously generated.

As can be seen by the above discussion, there are many features of the sensors, the network and the application that should influence protocol design. Accordingly, much research has gone into designing protocols for these different scenarios.

1.2 Unique Features of Sensor Networks

It should be noted that sensor networks do share some commonalities with general ad hoc networks. Thus, protocol design for sensor networks must account for the properties of ad hoc networks, including the following.

- Lifetime constraints imposed by the limited energy supplies of the nodes in the network.
- Unreliable communication due to the wireless medium.
- Need for self-configuration, requiring little or no human intervention.

However, several unique features exist in wireless sensor networks that do not exist in general ad hoc networks. These features present new challenges and require modification of designs for traditional ad hoc networks.

- While traditional ad hoc networks consist of network sizes on the order of 10s, sensor networks are expected to scale to sizes of 1000s.
- Sensor nodes are typically immobile, meaning that the mechanisms used in traditional ad hoc network protocols to deal with mobility may be unnecessary and overweight.
- Since nodes may be deployed in harsh environmental conditions, unexpected node failure may be common.
- Sensor nodes may be much smaller than nodes in traditional ad hoc networks (e.g., PDAs, laptop computers), with smaller batteries leading to shorter lifetimes, less computational power, and less memory.
- Additional services, such as location information, may be required in wireless sensor networks.
- While nodes in traditional ad hoc networks compete for resources such as bandwidth, nodes in a sensor network can be expected to behave more cooperatively, since they are trying to accomplish a similar universal goal, typically related to maintaining an application-level quality of service (QoS), or fidelity.
- Communication is typically data-centric rather than address-centric, meaning that routed data may be aggregated/compressed/prioritized/dropped depending on the description of the data.
- Communication in sensor networks typically takes place in the form of very short packets, meaning that the relative overhead imposed at the different network layers becomes much more important.
- Sensor networks often have a many-to-one traffic pattern, which leads to a “hot spot” problem.

Incorporating these unique features of sensor networks into protocol design is important in order to efficiently utilize the limited resources of the network. At the same time, to keep the protocols as light-weight as possible, many designs focus on particular subsets of these criteria for different types of applications. This has led to quite a number of different protocols from the data-link layer up to the transport layer, each with the goal of allowing the network to operate autonomously for as long as possible while maintaining data channels and network processing to provide the application’s required quality of service.

1.3 Performance Metrics

Because sensor networks posses these unique properties, some existing performance metrics for wireless network protocols are not suitable for evaluating sensor network protocols. For example, since sensor networks are much more cooperative in nature than traditional ad hoc networks, fairness becomes much less important. Also, since data sinks are interested in a general description of the environment rather than in receiving all raw data collected by individual nodes, throughput is less meaningful. Depending on the application, delay may be either much more or much less important in sensor networks.

Much more important to sensor network operation is energy-efficiency, which dictates network lifetime, and the high level QoS, or fidelity, that is met over the course of the network lifetime. This QoS is application-specific and can be measured a number of different ways. For example, in a typical surveillance application, it may be required that one sensor remains active within every subregion of the network, so that any intruder may be detected with high probability. In this case, QoS may be defined by the percentage of the environment that is actually covered by active sensors. In a typical tracking application, this QoS may be the expected accuracy of the target location estimation provided by the network.

1.4 Chapter Organization

The rest of this chapter will describe protocols and algorithms that are used to provide a variety of services in wireless sensor networks. Sections 2 and 3 provide examples of MAC and network protocols, respectively, for use in sensor networks. Section 4 presents some high-level protocols for energy-efficient management of sensor networks at the transport layer. Section 5 presents time synchronization and localization protocols that are often essential in sensor network applications. Section 6 presents a discussion of open research issues in the design of sensor networks.

2 Medium Access Control Protocols

Medium Access Control (MAC) protocols that have been designed for typical ad hoc networks have primarily focused on optimizing fairness and throughput efficiency, with less emphasis on energy conservation. However, the energy constraint is typically considered paramount for wireless sensor networks, and so many MAC protocols have recently been designed that tailor themselves specifically to the characteristics of sensor networks. Protocols such as MACAW [11] and IEEE 802.11 [12] eliminate the energy waste caused by colliding packets in wireless networks. Further, enhancements have been made to these protocols (e.g., PAMAS [13]) to avoid unnecessary reception of packets by nodes that are not the intended destination. However, it has been shown that idle power consumption can be of the same order as the transmit and receive power consumption, and if so, can greatly affect overall power consumption, especially in networks with relatively low traffic rates. Thus, the focus of most MAC protocols for sensor networks is to reduce this idle power consumption by setting the sensor radios into a sleep state as often as possible.

2.1 Sensor-MAC (S-MAC)

S-MAC was one of the first MAC protocols to be designed for sensor networks [14]. The basic idea behind SMAC is very simple — nodes create a sleep schedule for themselves that determines at what times to activate their receivers (typically $1 - 10\%$ of a frame) and when to set themselves into a sleep mode. Neighboring nodes are not necessarily required to synchronize sleep schedules, although this will help to reduce overhead (see Figure 1(b)). However, they must at least share their sleep schedule information with others through the transmission of periodic SYNC packets. When a source node wishes to send a packet to a destination node, it waits until the destination's wakeup period and sends the packet using CSMA with collision avoidance. S-MAC also incorporates a message passing mechanism, in which long packets are broken into fragments, which are sent and acknowledged successively following the initial RTS-CTS exchange. In addition to avoiding lengthy retransmissions, fragmentation helps address the hidden node problem, as fragmented data packets and ACKs can serve the purposes of the RTS and CTS packets for nodes that wake up in the middle of a transmission, having missed the original RTS-CTS exchange.

2.2 Timeout-MAC (T-MAC)

Several protocols have been developed based on S-MAC that offer solutions for various deficiencies and limitations of the original S-MAC protocol. T-MAC seeks to eliminate idle energy further by adaptively setting the length of the active portion of the frames [15]. Rather than allowing messages to be sent throughout a predetermined active period, as in S-MAC, messages are transmitted in bursts at the beginning of the frame. If no "activation events" have occurred after a certain length of time, the nodes set their radios into sleep mode until the next scheduled active frame. "Activation events" include the firing of the frame timer or any radio activity, including received or transmitted data, the sensing of radio communication, or the knowledge of neighboring sensors' data exchanges, implied through overheard RTS and CTS packets. An example of how T-MAC works is shown for a transmission from node A to node E in Figure 1(c). Since nodes D and E cannot hear node A's transmissions, they timeout after a delay of T_A. The end-to-end transmission from A to E is resumed during the next active period. The gains achieved by T-MAC are due to the fact that S-MAC may require its active period to be longer than necessary to accommodate traffic on the network with a given latency bound. While the duty cycle can always be tuned down, this will not account for bursts of data that can often occur in sensor networks (e.g., following the detection of an event by many surrounding neighboring sensors).

2.3 DMAC

As many wireless sensor networks consist of data gathering trees rooted at a single data sink, the direction of packets arriving at a node, if not the arrival times, are fairly stable and predictable. DMAC takes advantage of this by staggering the wakeup times for nodes based on their distance from the data sink [16]. By staggering the wakeup times in such a way, DMAC reduces the large delays that can be observed in packets that are forwarded for more than a few hops when synchronizing schedules as in S-MAC and T-MAC. The wakeup scheme consists of a receiving period and send period, each of length μ (set to accommodate a single transmission), followed by a long sleep period. Nodes on the data gathering tree begin their receiving period after an offset of $d * \mu$, where d represents the node's depth on the tree. In this way, a node's receiving period lines up with its upstream neighbor's send period and a node simply sends during downstream neighbors' receive periods, as shown in Figure 1(d). Contention within a sending period is accomplished through a simple random backoff scheme, after which a node sends its packet without a preceding RTS-CTS exchange.

2.4 TRaffic-Adaptive Medium Access (TRAMA)

While the aforementioned protocols attempt to minimize power consumption by reducing the time that the radio remains in the idle state, TRAMA attempts to reduce wasted energy consumption caused by packet collisions [17]. Nodes initially exchange neighborhood information with each

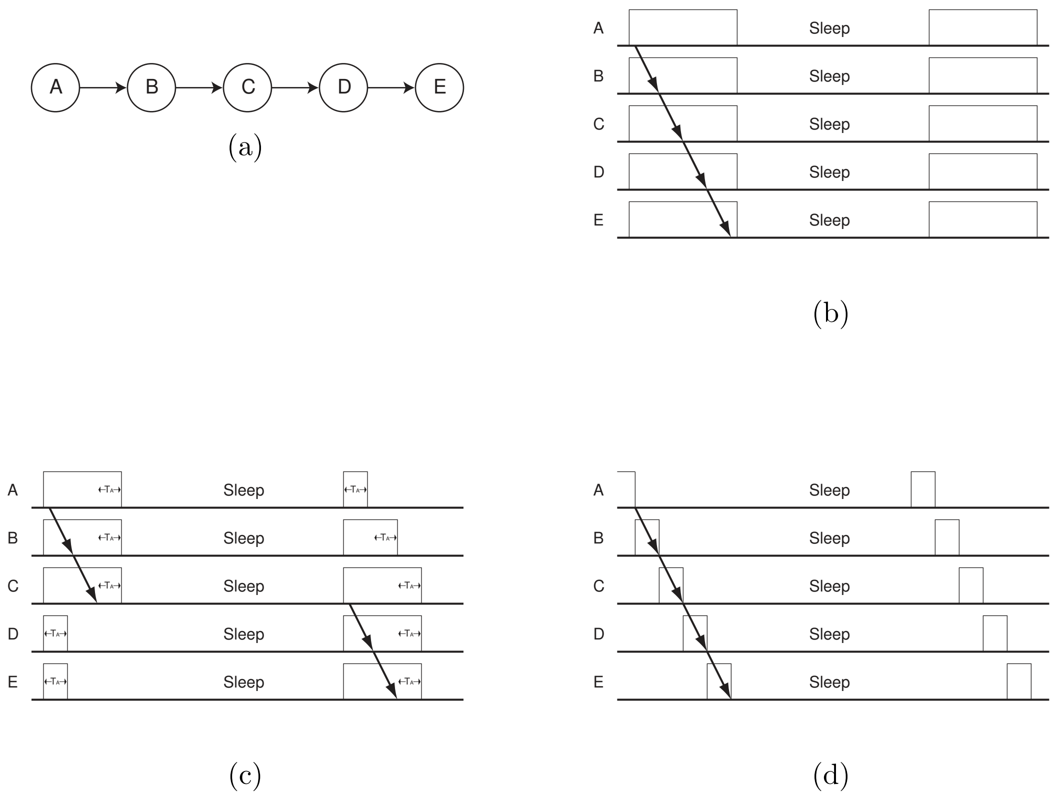


Figure 1: Chain network scenario (a). Sleep schedule for S-MAC [14] (b), T-MAC [15] (c), and DMAC [16] (d). In S-MAC, nodes synchronize their sleep schedules and remain awake for a predetermined period of time, even if no traffic is being sent, leading to wasted energy due to overhearing. In T-MAC, nodes D and E timeout after not hearing any channel activity for a duration of T_A. This leads to improved energy efficiency but can cause lengthy delays, as the nodes must wait until the next awake phase to complete the last two hops. In DMAC, the sleep schedules are staggered in a way so that it offers good energy efficiency and low delay for networks consisting of aggregating trees.

other during a contention period via a Neighbor Protocol (NP) so that each node has knowledge of all two-hop neighbors. These random access periods are followed by scheduled access periods, where nodes transmit schedule information via the Schedule Exchange Protocol (SEP) as well as actual data packets. Using the neighbor information acquired using NP and the traffic schedule information acquired using SEP, nodes determine their radio state using the Adaptive Election Algorithm (AEA). In AEA, each node calculates a priority for itself and all two-hop neighbors for the current slot using a hashing function. If a node has the highest priority for that slot and has data to send, it wins that slot and sends its data. If one of its neighbors has the highest priority and the node determines that it should be the intended receiver through information acquired during SEP, it sets itself to the receive mode. Otherwise, it is able to sleep and conserve energy. Since two nodes within the two-hop neighborhood of a node may consider themselves slot winners if they are hidden from each other, nodes must keep track of an Alternate Winner, as well as the Absolute Winner for a given time slot, so that messages are not lost. For example, consider a node N who determines that the Absolute Winner for a time slot is one of its two hop neighbors N_{2-hop}. If a one-hop neighbor N_{1-hop} who does not know of N_{2-hop} believes that it has won the slot, and wishes to send to N, N must stay awake even though it does not consider N_{1-hop} to have won the slot. Since a node may win more slots than necessary to empty its transmission buffer, some slots may remain unused that could have been used by nodes who won too few slots. To accommodate for this, the Adaptive Election Algorithm assigns priorities for the unused slots to the nodes needing extra slots.

2.5 Sparse Topology and Energy Management (STEM)

In the case of many sensor network applications, it is expected that nodes will continuously sense the environment, but transmit data to a base station very infrequently or only when an event of interest has occurred. In STEM, all sensors are left in a sleep state while monitoring the environment but not sending data and are only activated when traffic is generated [18]. In other words, transceivers are activated reactively rather than proactively, as with the other MAC protocols described in this section. When data packets are generated, the sensor generating the traffic uses a paging channel (separate from the data channel) to awaken its downstream neighbors. Two versions of STEM have been proposed—STEM-T, which uses a tone on a separate channel to wake neighboring nodes, and STEM-B, in which the traffic generating node sends beacons on a paging channel and sleeping nodes turn on their radios with a low duty cycle to receive the messages (the paging channel simply consists of synchronized time slots within the main communication channel). While STEM-T guarantees that minimal delay will be met (since receivers are turned on nearly instantaneously after data is generated), it requires more overhead than STEM-B since the receivers on the channel where the tones are sent must be idle listening all of the time. Also, STEM-T may require extra hardware as a separate radio is needed for this channel.

3 Network Protocols

When designing network protocols for wireless sensor networks, several factors should be considered. First and foremost, because of the scarce energy resources, routing decisions should be guided by some awareness of the energy resources in the network. Furthermore, sensor networks are unique from general ad hoc networks in that communication channels often exist between events and sinks, rather than between individual source nodes and sinks. The sink node(s) are typically more interested in an overall description of the environment, rather than explicit readings from the individual sensor devices. Thus, communication in sensor networks is typically referred to as data-centric, rather than address-centric, and data may be aggregated locally rather than having all raw data sent to the sink(s) [19]. These unique features of sensor networks have implications in the network layer and thus require a re-thinking of protocols for data routing. In addition, sensors often have knowledge of their own location in order to meaningfully assess their data. This location information can be utilized in the network layer for routing purposes. Finally, if a sensor network is well connected (i.e., better than is required to provide communication paths), topology control services should be used in conjunction with the normal routing protocols. This section describes some of the work that has been done to address these sensor network-specific issues in the routing layer.

3.1 Resource-Aware Routing

As resources are extremely limited in wireless sensor networks, it is important to consider how to most efficiently use them at all levels of the protocol stack. Many different approaches have been developed that consider the sensors' resources when making routing decisions. Initially, protocols were developed that considered only the sensors' energy resources. Later work considered not only individual sensors' energy but also the sensors' sensing resources.

3.1.1 Energy-aware Routing

Because of the scarce energy supplies available in sensor networks, a great deal of effort has been put forth in creating energy aware routing protocols that consider the energy resources available at each sensor and that try to balance the power consumption such that certain nodes do not die prematurely. Singh et al. were among the first to develop energy aware routing metrics [20]. They proposed that the lifetime of the network could be extended by minimizing the cumulative cost c_j of a packet j being sent from node n_1 to node n_k through intermediate nodes n_2, n_3, etc., where

$$c_j = \sum_{i=1}^{k-1} f_i(z_i) \tag{1}$$

$$f_i(z_i) = \frac{1}{1 - g(z_i)} \tag{2}$$

and $g(z_i)$ represents the normalized remaining lifetime corresponding to node n_i's battery level z_i. Further work by Chang et al. solved the problem of maximizing network lifetime by finding an optimal energy aware routing cost [21]. In their work, the routing cost of sending a packet was the sum of the routing costs of the individual links. The cost c_{ij} of a link between node i and node j was set to

$$c_{ij} = e_{ij}^{x_1} \underline{E}_i^{-x_2} E_i^{x_3} \tag{3}$$

where e_{ij} represents the energy necessary to transmit from node i to node j, $\underline{E}_i$ represents the residual energy of node i, and E_i represents the initial energy of node i. Brute force simulation methods were used to find the optimal values of x_1, x_2, and x_3.

From the intuition that can be taken from this initial work, several energy-aware routing protocols have been developed for sensor networks, including the one proposed by Shah et al. [22]. In this protocol, query interests are sent from a querying agent by way of controlled flooding toward the source node(s). Each node N_i has a cost $Cost(N_i)$ associated with it that indicates its reluctance to forward messages. Each upstream neighbor N_j of node N_i calculates a link cost C_{N_j,N_i} associated with N_i that depends on $Cost(N_i)$ as well as the energy e_{ij} required to transmit over this link and the normalized residual energy R_i at node N_i.

$$C_{N_j,N_i} = Cost(N_i) + e_{ij}^{\alpha} R_i^{\beta} \tag{4}$$

α and β are tunable parameters. Each node N_j builds a forwarding table FT_j consisting of its lowest cost downstream neighbors and the link cost C_{N_j,N_i} associated with those neighbors. Node N_j assigns a probability P_{N_j,N_i} to each neighbor as

$$P_{N_j,N_i} = \frac{1/C_{N_j,N_i}}{\sum_{k \in FT_j} 1/C_{N_j,N_k}} \tag{5}$$

such that received messages will be forwarded over each link with this probability. Before forwarding its message, N_j must determine its own value of $Cost(N_j)$, which is simply the weighted average of the costs in its forwarding table FT_j

$$Cost(N_j) = \sum_{i \in FT_j} P_{N_j,N_i} C_{N_j,N_i} \tag{6}$$

3.1.2 Fidelity-aware Routing

DAPR (Distributed Activation based on Predetermined Routes) is similar to these energy-aware routing protocols but was designed specifically for maintaining high-level QoS requirements (e.g., coverage) over long periods of time [23]. Rather than assigning cost according to individual nodes based on the residual energy at those nodes, DAPR considers the importance of a node to the sensing application. Since sensors in a coverage application typically cover redundant areas and redundancy can vary throughout the network, some nodes might be considered more important than others. In DAPR, a node first finds the subregion within its region of coverage that is the most poorly covered. The cost assigned to that node is related to the combined energy of all nodes capable of redundantly covering this poorly covered region. Large gains in network lifetime can be seen when considering the importance of a node to the overall sensing task when making routing decisions if the sensor deployment is such that there is a high variation in the density in different subregions of the environment. However, there is an added overhead associated with this approach, as it requires nodes to acquire additional information from neighboring nodes.

3.2 Data-Centric Routing Protocols

Sensor networks are fundamentally different from ad hoc networks in the data they carry. While in ad hoc networks individual data items are important, in sensor networks it is the aggregate data or the information carried in the data rather than the actual data itself that is important. This has led to a new paradigm for networking these types of devices – data-centric routing. In data-centric routing, the end nodes, the sensors themselves, are less important than the data itself. Thus, queries are posed for specific data rather than for data from a particular sensor, and routing is performed using knowledge that it is the aggregate data rather than any individual data item that is important.

3.2.1 Sensor Protocol for Information via Negotiation (SPIN)

SPIN is a protocol that was designed to enable data-centric information dissemination in sensor networks [24]. Rather than blindly broadcasting sensor data throughout the network, nodes receiving or generating data first advertise this data through short ADV messages. The ADV messages simply consist of an application-specific meta-data description of the data itself. This meta-data

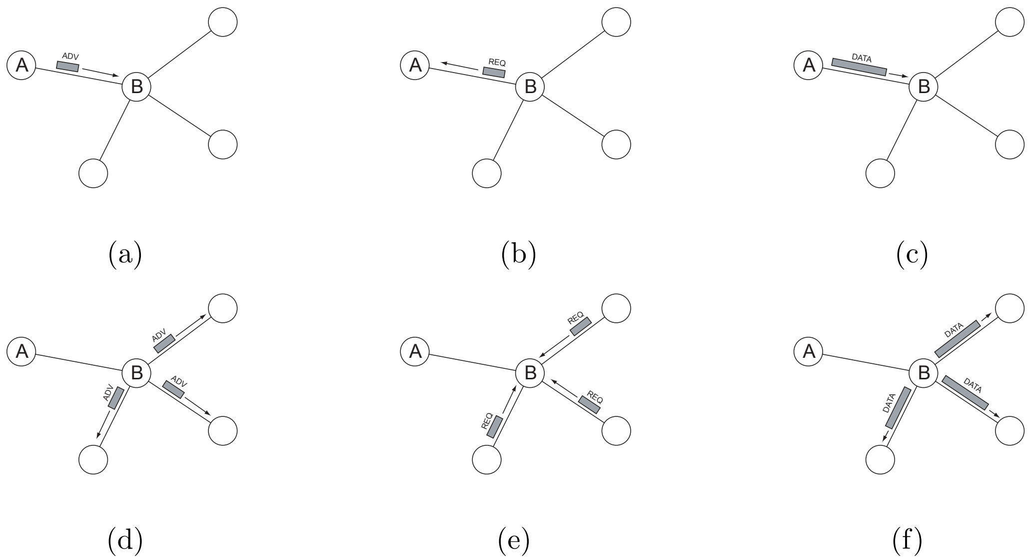


Figure 2: Illustration of message exchange in the SPIN protocol [24]. Nodes advertise their data with ADV messages (a). Any node interested in receiving the data replies with a REQ message (b), to which the source node replies with the transmission of the actual data (c). The receiving node then advertises this new data (d) and the processes continues (e,f).

can describe such aspects as the type of data and the location of its origin. Nodes that are interested in this data request the data from the ADV sender through REQ messages. Finally, the data is disseminated to the interested nodes through DATA messages that contain the data. This procedure is illustrated in Figure 2.

The advantage of SPIN over blind flooding or gossiping data dissemination methods is that it avoids three costly problems: implosion, overlap and resource blindness. Implosion occurs in highly connected networks that employ flooding and thus each sensor receives many redundant copies of the data (see Figure 3a). For large data messages, this wastes considerable energy. In SPIN, on the other hand, short ADV messages will suffer from the implosion problem, but the costly transfer of data messages is greatly reduced. Overlap occurs due to the redundant nature of sensor data. Thus two sensors with some common data will both send their data, causing redundancy in data transmission and thus energy waste (see Figure 3b). SPIN is able to solve this problem by naming data so that sensors only request the data or parts of data they are interested in receiving. Finally, in SPIN, there are mechanisms whereby a sensor that is running low on energy will not advertise its data in order to save its dwindling energy resources. Thus SPIN solves the resource blindness problem by having sensors make decisions based on the current level of available resources.

3.2.2 Directed Diffusion

Directed Diffusion is a communication paradigm that has been designed to enable data-centric communication in wireless sensor networks [25]. To perform a sensing task, a querying node creates an interest, which is named according to the attributes of the data or events to be sensed. When an interest is created, it is injected into the network by the sink node by broadcasting an interest message containing the interest type, duration, and an initial reporting rate to all neighbors. For example, one interest might be to count the number people in a given area every second for the next

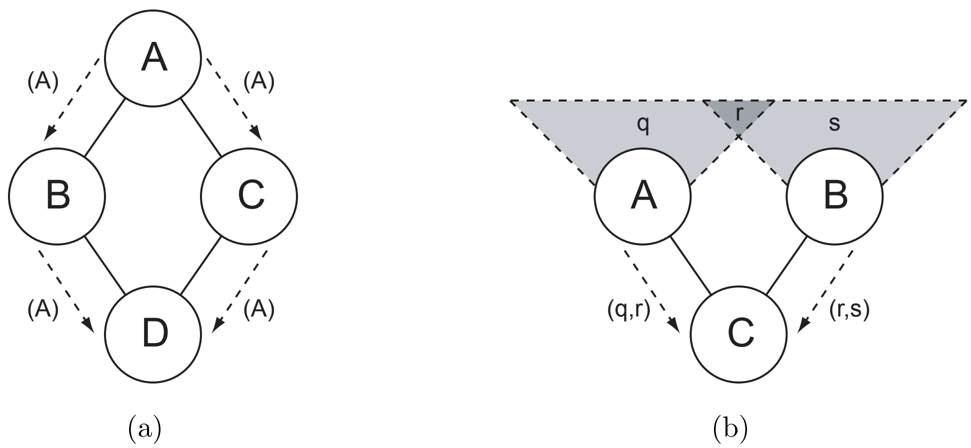


Figure 3: Problems with blind flooding of sensor data. (a) Implosion occurs in a highly connected network where nodes receive duplicate copies of data, wasting energy and bandwidth resources. As seen in this figure, node D receives two copies of node A's data. (b) Overlap occurs due to the redundant nature of sensor data. This figure shows that C receives data about region r from nodes A and B, again wasting valuable sensor resources.

10 minutes. Local interest caches at each node contain entries for each interest of which the node is aware that has been created on the network. An entry in the caches contains information about the interest's type, duration, and gradient (a combination of the event rate and direction toward the data sink). Nodes receiving the interest messages find (or create) the relevant interest entry in their caches and update the gradient field toward the node from which the message was received to the rate defined in the interest message. Each gradient also has expiration time information, which must be updated upon the reception of the interest messages.

Interests are diffused throughout the network toward the sink node using one of a number of forwarding techniques. For example, Figure 4 shows a network in which the interest was sent to the region of interest via controlled flooding. Once the interest reaches the desired region, sensor nodes within the region process the query and begin producing data at the specified rate (if more than one entry for the same interest type exist, data is produced at the maximum rate of these entries). Data pertaining to these interests are then forwarded to each node for which a gradient exists at the rate specified for each individual gradient. After receiving low rate events from the source (recall that the initial reporting rate is set low), the data sink may reinforce higher quality paths, which might be chosen, for example, as those that experience low latency or those in which the confidence in the received data is deemed to be high by some application-specific measure (Figure 4). Reinforcement messages simply consist of the original interest messages set to higher reporting rates. These reinforced routes are established more conservatively than the original low rate interest messages so that only a single or few paths from the event to the sink are used.

3.2.3 Rumor Routing

While long-lived queries/data flows justify the overhead involved in establishing cost fields in a network, it may not be worth this effort when executing short-lived and one-shot queries. Rumor routing was designed for these types of queries [26]. When an event is detected by a sensor, it probabilistically creates an agent in the form of a data packet, and forwards it throughout the network in a random manner (solid line in Figure 5). Nodes through whom the agent is forwarded

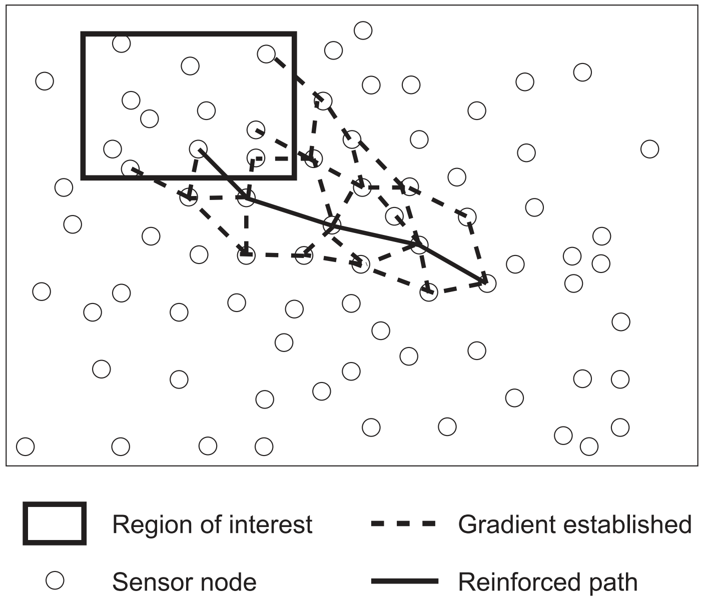


Figure 4: Establishing gradients in Directed Diffusion [25]. As the query is routed toward the region of interest, gradients for that interest are established in the reverse direction of the query dissemination. After data begins to arrive at the querying node, the path of highest quality is reinforced.

maintain local state information about the direction and distance to the event. Should an agent traverse a node with knowledge of a path to other events, it adds this information so that subsequent nodes that the agent flows through will maintain state information regarding these events as well. When a node wishes to perform a query related to a given event, it simply forwards a query packet in a random direction so that the query traverses a random walk throughout the network (dashed line in Figure 5). Because of the fact that two lines drawn through a given area are likely to cross, there is a high likelihood that the query will eventually reach a node with a path to the specified event, especially if multiple agents carrying that event are sent through the network. If multiple queries happen not to reach the event, the querying node may resort to flooding queries over the entire network.

3.3 Geographic Routing

Often times, wireless sensor networks require a query packet to be forwarded to a particular region of interest in the network. A natural approach to perform this forwarding is to utilize geographic forwarding. Geographic forwarding reduces the amount of routing overhead, which is largely due to route discovery, and requires little memory utilization for route caching compared to typical address-centric ad hoc routing protocols. Furthermore, geographic routing protocols can enable geographically distributed data storage techniques such as Geographic Hash Tables (GHT) [27].

3.3.1 Greedy Perimeter Stateless Routing (GPSR)

GPSR is a geographic routing protocol in which nodes make local packet forwarding decisions according to a greedy algorithm [28]. Under normal circumstances, a packet that is destined for

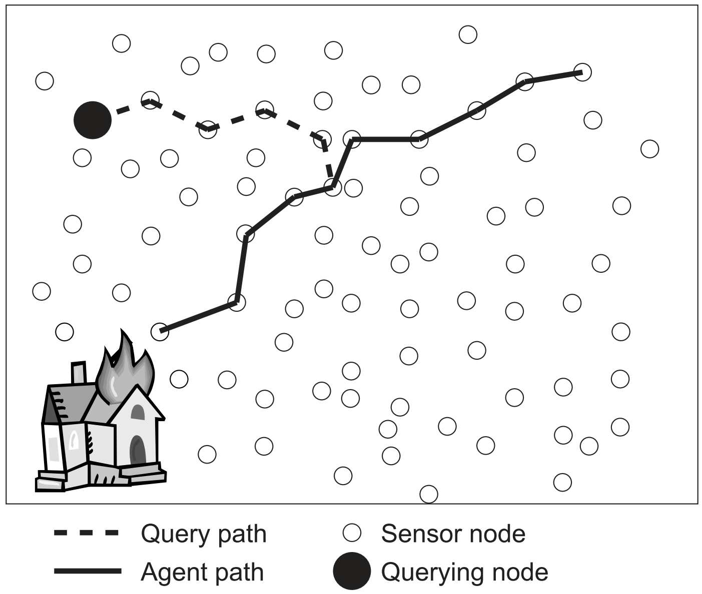
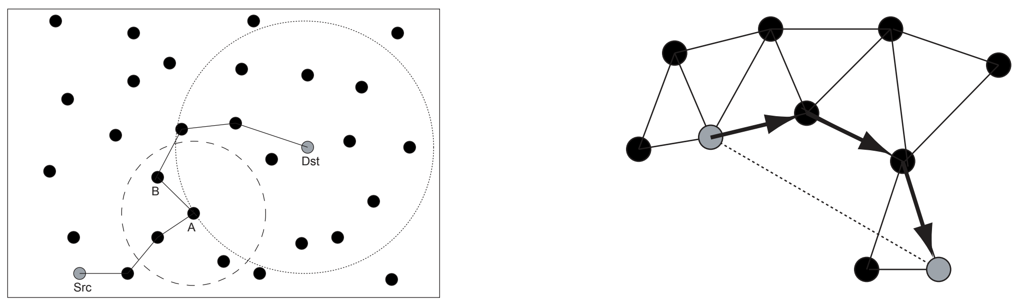


Figure 5: Query handling in Rumor Routing [26]. After an event is detected, and agent is initiated and sent on a random path through the network, establishing state at each node on the path. A query packet is similarly sent in a random direction and hopefully crosses paths with the agent, allowing the query to be answered and returned to the querying node.

some node D is forwarded to the node's neighbor that enables the maximum progress toward D (such a greedy forwarding scheme was originally proposed in the work of Takagi and Kleinrock [29]). However, obstacles or a lack of adequate sensor density can cause voids in the network topology so that packets reach a hole, from which the packet cannot be progressed any further without first being sent backward. GPSR accounts for this by incorporating a perimeter routing mechanism. These voids can be detected by the nodes surrounding them, and routes which circumnavigate the voids can be established heuristically. When a packet reaches these voids, these routes can be used (routing by the right hand rule) until normal greedy routing can be used again This process is illustrated in Figure 6(a). While this approach works well, another more robust perimeter routing algorithm is also proposed. In this algorithm, the graph that can be drawn from the complete network topology is first reduced to a planar graph in which no edges cross. Once a packet reaches a void, the forwarding node N finds the face of the planar graph which is intersected by the line connecting N and the destination (see Figure 6(b)). N then forwards the packet to the node along the edge that borders this face. This procedure continues with each forwarding node finding the face that the line connecting N and the destination intersects and routing along an edge bordering the face until the void has been cleared.

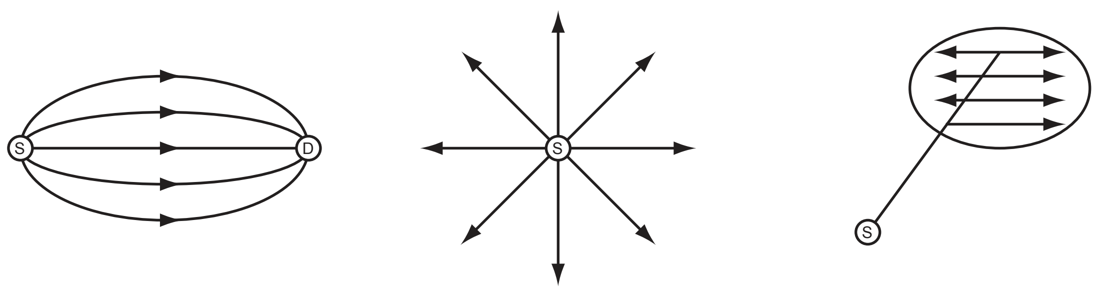
3.3.2 Trajectory Based Forwarding (TBF)

Trajectory Based Forwarding is a useful paradigm for geographic routing in wireless sensor networks [30]. Rather than sending a packet along a straight path toward its destination (as methods such as GPSR would do under ideal scenarios with dense deployment and no obstructions), TBF allows packets to follow a source-specified trajectory, increasing the flexibility of an overall forward-



(a) (b)

Figure 6: GPSR [28] greedy forwarding policy (a) and perimeter routing algorithm (b)



(a) (b) (c)

Figure 7: Possible trajectories to use in TBF [30] for robust multipath routing (a), spoke broadcasting (b) and broadcast within a remote region (c)

ing strategy. For example, multipath routing can be achieved by sending multiple copies of a single packet along separate geographic trajectories, increasing resilience to localized failures or congestion in certain parts of the network. Also, TBF can increase the efficiency of many different forwarding techniques, including multipath forwarding (Figure 7(a)), spoke broadcasting (Figure 7(b)), and broadcast to a remote subregion (Figure 7(c)).

3.4 Clustering for Data Aggregation

As sensor networks are expected to scale to large numbers of nodes, protocol scalability is an important design criteria. If the sensors are managed directly by the base station, communication overhead, management delay, and management complexity become limiting factors in network performance. Clustering has been proposed by researchers to group a number of sensors, usually within a geographic neighborhood, to form a cluster that is managed by a cluster head. A fixed or adaptive approach may be used for cluster maintenance. In a fixed maintenance scheme, cluster membership does not change over time, whereas in adaptive clustering scheme, sensors may change their associations with different clusters over time (see, for example, Figure 8).

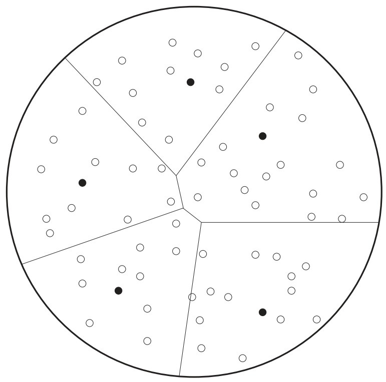


Figure 8: Adaptive clustering of the network.

Clustering provides a framework for resource management. It can support many important network features within a cluster, such as channel access for cluster members and power control, as well as between clusters, such as routing and code separation to avoid inter-cluster interference. Moreover, clustering distributes the management responsibility from the base station to the cluster heads, and provides a convenient framework for data fusion, local decision making and local control, and energy savings [31, 32, 33].

3.4.1 Low Energy Adaptive Clustering Hierarchy (LEACH)

In-network processing can greatly reduce the overall power consumption of a sensor network when large amounts of redundancy exist between nearby nodes. Rather than requiring all sensors' data to be forwarded to a base station that is monitoring the environment, nodes within a region can collaborate and send only a single summarization packet for the region. This use of clustering was first introduced in the Low Energy Adaptive Clustering Hierarchy (LEACH) protocol [32]. In LEACH, nodes are divided into clusters, each containing a cluster head whose role is considerably more energy intensive than the rest of the nodes; for this reason, nodes rotate roles between cluster head and ordinary sensor throughout the lifetime of the network.

At the beginning of each round, each sensor node makes an independent decision through a randomized algorithm about whether or not to assume a cluster head role. Nodes that choose to be cluster heads announce their status to the rest of the network. Based on the received signal strength of these announcements, sensors join the cluster that requires the least power to communicate with the cluster head (assuming transmission power control is available). During the round, the ordinary sensors in each cluster send data to their respective cluster heads according to a time-division multiple access (TDMA) schedule. Inter-cluster interference is reduced using different spreading codes in neighboring clusters. The cluster head aggregates data from all the cluster members and sends the aggregate data to the base station. The length of each round is chosen such that each node is expected to be able to perform a cluster head role once during its lifetime.

Because there is no interaction between nodes when deciding roles, the cluster heads may be chosen such that there is no uniformity throughout the network and certain sensors are forced to join clusters located at large distances from them. To mitigate this problem, a centralized version of LEACH called LEACH-C has been developed. LEACH-C uses simulating annealing to choose the cluster heads for a given round so that the average transmission power between sensors and their cluster heads is minimized.

3.4.2 Hybrid Energy-Efficient Distributed Clustering (HEED)

Nodes in LEACH independently decide to become cluster heads. While this approach requires no communication overhead, it has the drawback of not guaranteeing that the cluster head nodes are well distributed throughout the network. While the LEACH-C protocol solves this problem, it is a centralized approach that cannot scale to very large numbers of sensors.

Many papers have proposed clustering algorithms that create more uniform clusters at the expense of overhead in cluster formation. One approach that uses a distributed algorithm that can converge quickly and has been shown to have low overhead is called HEED [34]. HEED uses an iterative cluster formation algorithm, where sensors assign themselves a "cluster head probability" that is a function of their residual energy and a "communication cost" that is a function of neighbor proximity. Using the cluster head probability, sensors decide whether or not to advertise that they are a candidate cluster head for this iteration. Based on these advertisement messages, each sensor selects the candidate cluster head with the lowest "communication cost" (which could be the sensor itself) as its tentative cluster head. This procedure iterates, with each sensor increasing its cluster head probability at each iteration until the cluster head probability is one and the sensor declares itself a "final cluster head" for this round. The advantages of HEED are that nodes only require local (neighborhood) information to form the clusters, the algorithm terminates in O(1) iterations, the algorithm guarantees that every sensors is part of just one cluster, and the cluster heads are well-distributed.

3.5 Querying a Distributed Database

Since sensor networks can be thought of as a distributed database system, several architectures (e.g., Cougar [35], SINA [36], TinyDB [37]) propose to interface the application to the sensor network through an SQL-like querying language. However, since sensor networks are so massively distributed, careful consideration should be put into the efficient organization of data and the execution of queries.

3.5.1 Tiny AGgregation (TAG) Service

TAG is a generic aggregation service for wireless sensor networks that minimizes the amount of messages transmitted during the execution of a query [37]. In contrast to standard database query execution techniques, in which all data is gathered by a central processor where the query is executed, TAG allows the query to be executed in a distributed fashion, greatly reducing the overall amount of traffic transmitted on the network. The standard SQL query types (`COUNT`, `AVERAGE`, `SUM`, `MIN`, `MAX`), as well as more sophisticated query types, are included in the service, although certain query types allow more energy savings than others. Time is divided into epochs for queries requiring values to be returned at multiple times. When a query is sent by some node (initially the root), the receiving nodes set their parents to be the sending node and establish an interval within the epoch (intervals may be set to a length of `EPOCH_DURATION`$/d$, where d represents the maximum depth of the aggregating tree) during which their eventual children should send their aggregates (this interval should be immediately prior to their sending interval).

3.5.2 TinyDB/ACQP

TinyDB is a processing engine that runs Acquisitional Query Processing (ACQP) [38], providing an easy-to-use generic interface to the network through an enhanced SQL-like interface and enabling the execution of queries to be optimized at several levels. ACQP allows storage points containing windows of sensor data to be created so that queries over the data streams can be executed more easily. Such storage points may be beneficial, for example, in sliding window type queries (e.g., find the average temperature in a room over the previous hour once per minute). ACQP also supports queries that should be performed upon the occurrence of specific events as well as queries that allow sensor settings such as the sensing rate to be adapted to meet a certain required lifetime.

Perhaps most importantly, ACQP provides optimization of the scheduling of sensing tasks as well as at the network layer. Since the energy consumption involved in the sensing of certain types of data is not negligible compared to the transmission costs of sending such packets, the scheduling of complex queries should be optimized in order to avoid unnecessary sensing tasks. ACQP optimizes this scheduling based on sensing costs and the expected selectivity of the query so as to minimize the expected power consumption during a query. Significant power savings can also be achieved by the ACQP's batching of event-based queries in some cases.

The topology of an aggregating tree can also be optimized by considering the query in its formation. TinyDB uses Semantic Routing Trees (SRTs). Rather than requiring children to choose a parent node solely based on link quality, the choice of a parent nodes during the construction of an SRT also depends on the predicates of the query for which the tree is being built (i.e., the conditions that should be met for inclusion in the query). Specifically, children nodes choose a parent either to minimize the difference between their attributes of the predicate in the query or to minimize the spread of the attributes of the children of all potential parents. When a query is processed, a parent knows the attributes of all children and can choose not to forward the message if it determines that none of its children can contribute to the query (based on the query predicate and the attributes of its children).

3.5.3 Geographic Hash Table (GHT)

Geographic Hash Tables (GHT) provide a convenient, data-centric means to store event-based data in wireless sensor networks [27]. Storing data in a distributed manner provides an energy-efficient alternative in large-scale sensor networks, where the number of messages involved in the querying of the network becomes very large, and in networks where many more events are detected than are queried, where the hot spot around the querying node seen in external storage techniques can be avoided. When an event is sensed, the location at which the data related to the event is should be stored is found by hashing its key to a location within the network. This location has no node associated with it when it is hashed, but the data will eventually find a home node closest to the hashed location. Once the location is determined, a data packet is sent using GPSR [28], although with no destination node explicitly included in the routing packet. Eventually the packet will arrive at the closest node to the intended storage location, and GPSR will enter into perimeter mode, routing the packet in a loop around the intended location and eventually sending it back to the node originally initiating the perimeter routing. The node beginning and ending this loop and those on the perimeter path are called the home node and the home perimeter, respectively. To account for dynamic network topologies, a Perimeter Refresh Protocol (PRP) is used, in which the home node periodically sends the packet in a loop on the home perimeter and the home perimeter nodes assume the role of home node if they do not hear these refresh packets after a certain timeout interval.

3.6 Topology Control

Research groups have shown that because of the low duty cycles of sensor nodes' radios, the dominant aspect of power consumption is often idle listening. Unless communication is tightly synchronized, even intelligent MAC protocols such as those described in Section 2 of this chapter cannot completely eliminate this wasted power consumption. However, since sensor networks are

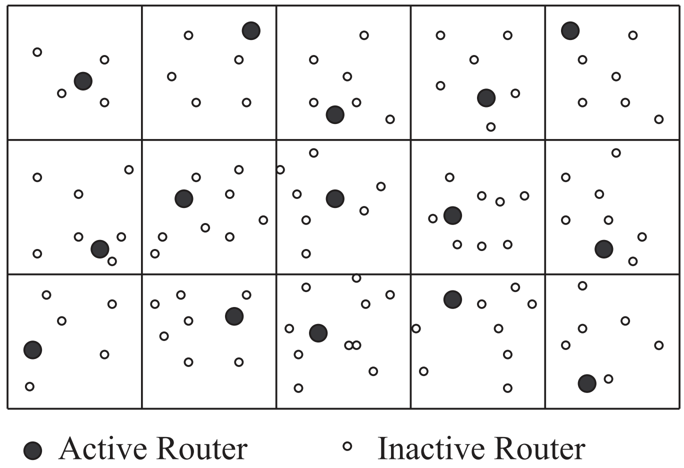


Figure 9: Example of a GAF virtual grid [42]. Only one node per cell is activated as a router.

expected to be characterized by dense sensor deployment, it is not necessary for all sensors' radios to remain on at all times in order for the network to remain fully connected. While traditional topology control protocols attempt to maintain a predetermined number of neighbor nodes through transmission power control (e.g., [39, 40, 41]) so that congestion is reduced, several topology control protocols designed for ad hoc and sensor networks achieve energy efficiency by assigning the role of router to only enough nodes to keep the network well-connected. In other words, the goal of these protocols is to maintain a fully connected dominating set. While some of these protocols were originally designed for use in general ad hoc networks, most are suitable for sensor networks as well.

3.6.1 Geographic Adaptive Fidelity (GAF)

GAF is a topology control protocol that was originally designed for use in general ad hoc networks [42]. GAF divides the network into a virtual grid and selects only a single node from each virtual grid cell to remain active as a designated router at a given time, as illustrated in Figure 9. As long as the cell dimensions are chosen small enough ($\frac{transmission\_range}{\sqrt{5}}$), most nodes in the network, except those near the border of the network, will retain neighbors in all four directions and the network will remain fully connected. Nodes initially enter the discovery state and listen for messages from other nodes within their cell. If another node within the cell is determined to be the designated router for the cell, the node will enter a sleep state and conserve energy. From the sleep state, a node will periodically enters the discovery state. If a node determines that it should be the designated active router for its cell, it will enter the active state and participate in data routing, eventually falling back into the discovery state. As the density of a network implementing GAF increases, the number of activated nodes per grid cell remains constant while the number of nodes per cell increases proportionally. Thus, GAF can allow a network to live for an amount of time approximately proportional to a network's density.

3.6.2 Span

Span is a topology control protocol that allows nodes that are not involved in a routing backbone to sleep for extended periods of time [43]. In Span, certain nodes assign themselves the position of "coordinator." These coordinator nodes are chosen to form a backbone of the network, so that the capacity of the backbone approaches the potential capacity of the complete network. Periodically, nodes that have not assigned themselves the coordinator role initiate a procedure to decide if they should become a coordinator. The criteria for this transition is if the minimum distance between any two of the node's neighbors exceeds three hops. To avoid the situation where many nodes simultaneously decide to become coordinator, backoff delays are added to nodes' coordinator announcement messages. The backoff delays are chosen such that nodes with higher remaining energy and those potentially providing more connectivity in their neighborhood are more likely to become a coordinator. To ensure a balance in energy consumption among the nodes in the network, coordinator nodes may fall back from their coordinator role if neighboring nodes can make up for the lost connectivity in the region.

3.6.3 Adaptive Self-Configuring sEnsor Networks Topologies (ASCENT)

ASCENT is similar to Span in that certain nodes are chosen to remain active as routers while others are allowed to conserve energy in a sleep state [44]. In ASCENT, the decision to become an active router is based not only on neighborhood connectivity, but also on observed data loss rates, providing the network with the ability to trade energy consumption for communication reliability. Nodes running the ASCENT protocol initially enter a test state where they actively participate in data routing, probe the channel to discover neighboring sensors and learn about data loss rates, and send their own "Neighborhood Announcement" messages. If, based on the current number of neighbors and current data loss rates, the sensor decides that its activation would be beneficial to the network, it becomes active and remains so permanently. If the sensor decides not to become active, it falls into a passive state, where it gathers the same information as it does in the test state (as well as any "Help" messages from neighboring sensors experiencing poor communication links), but it does not actively participate in data routing. From this state, the node may reenter the test state if the information gathered indicates poor neighborhood communication quality, or enter the sleep state, turning its radio off and saving energy. The node periodically leaves the sleep state to listen to the channel from the passive state.

3.6.4 Energy-Aware Data Centric Routing (EAD)

EAD is an algorithm for constructing a minimum connected dominating set among the sensors in the network, prioritizing nodes so that those with the highest residual energy are most likely to be chosen as non-leaf nodes [45]. To establish a broadcast tree, control messages containing transmitting nodes' type (undefined, leaf node, or non-leaf node), level (in the broadcast tree), parent, and residual energy are flooded throughout the network, starting with the data sink. During the establishment of the tree, undefined nodes listen for control messages. If an undefined node receives a message from a non-leaf node, it becomes a leaf node and prepares to send a message announcing its leaf status after sensing the channel to be idle for some backoff time T_2^v. Alternatively, if an undefined node receives a message from a leaf node, it becomes a non-leaf node after sensing the channel idle for some backoff time T_1^v and sending a control message indicating its non-leaf status. However, if a message is received from a non-leaf node during its backoff interval, the node as it would when receiving such a message during its original undecided state. To ensure that nodes with more residual energy are more likely to assume the more energy intensive non-leaf roles, T_1^v

and T_2^v should be monotonically decreasing functions of the residual energy. Also, the minimum possible value of T_1^v should be larger than the maximum possible value T_2^v so that the resulting set of non-leaf nodes is of minimal size. If at any point, a leaf node received a message from a neighboring non-leaf node indicating that it is the neighbor's parent, it immediately becomes a non-leaf node and broadcasts a message indicating so. Eventually, all connected nodes in the network will assume the role of a leaf or a non-leaf and the resulting non-leaf nodes will comprise an approximation of a minimum connected dominating set with a high priority attached to nodes with the highest remaining energy supplies.

4 Protocols for QoS Management

Perhaps one of the most differentiating features of wireless sensor networks is the way in which Quality of Service (QoS) is redefined. Whereas delay and throughput are typically considered the most important aspects of QoS in general ad hoc wireless and wired networks, new application-specific measures as well as network lifetime are more suitable performance metrics for wireless sensor networks. Because of the redundancy and the application-level importance associated with the data generated by the network, QoS should be determined by the content as well as the amount of data being delivered. In other words, it may be true that the application will be more satisfied with a few pieces of important, unique data than with a large volume of less important, redundant data. Thus, while it is important to use congestion control in some cases so that the reliability of the sensor network is not reduced due to dropped packets [46], this congestion control can be enhanced by intelligently selecting which nodes should throttle their rates down or stop sending data. Furthermore, the congestion aspect aside, it is important to reduce the amount of traffic generated on the network whenever possible to extend the lifetime of network because of the tight energy constraints imposed on sensor nodes. This general strategy is often referred to as sensor management, or fidelity control, and is summarized in [47].

4.1 Transport layer

Transport layer protocols are used in many wired and wireless networks as a means to provided services such as loss recovery, congestion control, and packet fragmentation and ordering. While popular transport layer protocols such as TCP may be overweight and many typical transport layer services may not be necessary for most wireless sensor network applications, some level of transport services can be beneficial. This section describes some transport level protocols that are suitable for message delivery in wireless sensor networks.

4.1.1 Pump Slowly Fetch Quickly (PSFQ)

The PSFQ protocol was designed to enable reliable distribution of retasking/repogramming code from a sensor network base station to the sensors in the network [48]. PSFQ provides reliability on a hop-by-hop basis, unlike many end-to-end transport protocols. When new code needs to be distributed, it is fragmented and sent via a pump mechanism that slowly injects packets into the network (with inter-packet spacing of at least T_{min}). At a relaying node, a TTL field is decremented and the message is rebroadcast after a delay chosen on the interval $[T_{min}, T_{max}]$ as long as the local data cache does not indicate a packet loop. To avoid excessive broadcast overlap in dense networks, the packet is removed from the transmit buffer if 4 copies of the packet have been received by the node. The significant delays are introduced so that normal operation (sensor-to-sink traffic) is not interfered with and to allow the quick recovery of packet losses without requiring large amounts

Operating region	Updated Reporting Rate
(NC,LR)	$f_{i+1} = \frac{f_i}{\eta_i}$
(OOR)	$f_{i+1} = f_i$
(NC,HR)	$f_{i+1} = \frac{f_i}{2}(1 + \frac{1}{\eta_i})$
(C,HR)	$f_{i+1} = \frac{f_i}{\eta_i}$
(C,LR)	$f_{i+1} = f_i^{\eta_i/k}$

Table 1: Rate adaption in ESRT [49]. After each round, the data sink requires that the new reporting rate is set to f_{i+1}, based on the current operating region, the current reporting rate f_i, the current normalized reliability η_i, and an arbitrary constant k.

of buffer space at the forwarding nodes. Once a node detects that a packet is received out of sequence, it begins the fetch operation, aggressively trying to quickly recover the lost fragments. PSFQ assumes that most packet loss in sensor networks is caused by poor link quality rather than congestion. Thus, the aggressive recovery approach is not expected to further compound any congestion problem. When packet losses are detected, a node sends a NACK packet indicating the lost packets after a very short delay. If a reply is not received after T_r ($T_r << T_{min}$), the NACK is resent up to a threshold number of retires, after which the node gives up. NACKs are withheld if similar NACKs are overheard from neighboring sensors. PSFQ also contains a proactive fetch operation that can be used to detect losses at the end of a sequence (since no subsequent packets will allow the receiving node to know that packets have been lost).

4.1.2 Event-to-Sink Reliable Transport (ESRT)

The ESRT protocol [49] was designed as a solution to the problem posed in Tilak's work [46]. The protocol achieves energy efficiency by requiring the sensors to send only enough traffic to meet the application's reliability requirements, and it contains mechanisms for detecting and alleviating congestion. From an observed plot of reliability as a function of the sensors' reporting rate (see Figure 10), it can be seen that the network can operate in one of five regions:

- No congestion, low reliability (NC,LR)
- The optimal operating region (OOR)
- No congestion, high reliability (NC,HR)
- Congestion, high reliability (C,HR)
- Congestion, low reliability (C,LR)

In ESRT, the data sink in the sensor network periodically broadcasts a revised reporting rate to the sensors, attempting to choose the frequency that will move the network into the OOR. If η_i represents the reliability observed at the sink during interval i, the frequency f_{i+1} for interval $i+1$ is set to the value as indicated in Table 1 and broadcast to the sensors in the network.

While reliability can easily be observed at the sink by counting the number of received packets, congestion is detected by requiring routers to explicitly notify the sink in the event of a buffer overflow.

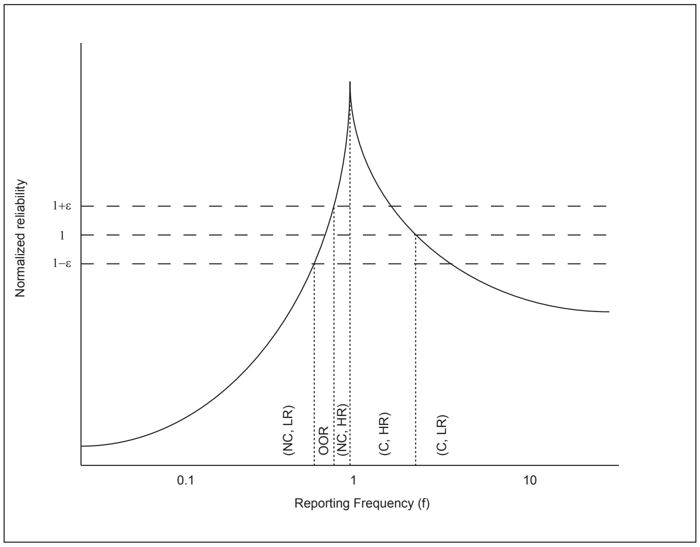


Figure 10: Example plot of reliability vs. sensor reporting rate. ESRT [49] adapts traffic rates according to the region of this plot that the network is operating in.

4.2 Providing Coverage of an Environment

Traditional rate control protocols such as ESRT define a network's reliability as the number of total received packets at a base station during a given time interval. However, in many applications, such a definition is only a very coarse approximation of the fidelity of the data that has been aggregated. To get a true measure of fidelity, it is often required to look at the origin and contents of the received packets.

A common application for sensor networks is for the sensors within some region to sense the environment or a subregion in the environment so that it is completely covered. In general, these applications require K-coverage, meaning that each location in the region to be monitored should have K active sensors located within their sensing ranges, with all other sensors turning off in order to save energy (many applications simply require 1-coverage).

4.2.1 Probing Environment and Adaptive Sleeping (PEAS)

PEAS is a protocol that was developed to provide consistent environmental coverage and robustness to unexpected node failures [50]. Nodes begin in a sleeping state, from which they periodically enter a probing state. In the probing state, a sensor transmits a probe packet, to which its neighbors will reply after a random backoff time if they are within the desired probing range. If no replies are received by the probing node, the probing sensor will become active; otherwise, it will return to the sleep state. The probing range is chosen to meet the more stringent of the density requirements imposed by the sensing radius and the transmission radius. The probing rate of PEAS is adaptive and is adjusted to meet a balance between energy savings and robustness. Specifically, a low probing rate may incur long delays before the network recovers following an unexpected node failure. On the other hand, a high probing rate may lead to expensive energy waste. Basically, the probing rate of individual nodes should increase as more node failures arise, so that a consistent expected

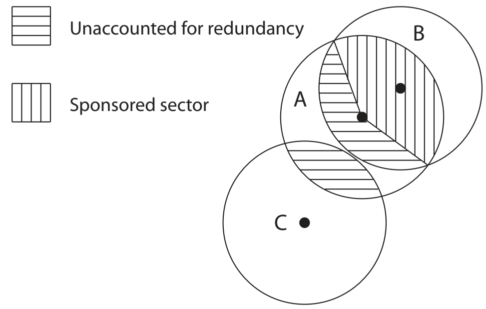


Figure 11: A sponsored sector, as defined by [51]. Sensor A admits the redundant coverage of sensor B in the vertically shaded regions. The additional redundancy of sensors B and C shown in the horizontally shaded regions is not accounted for.

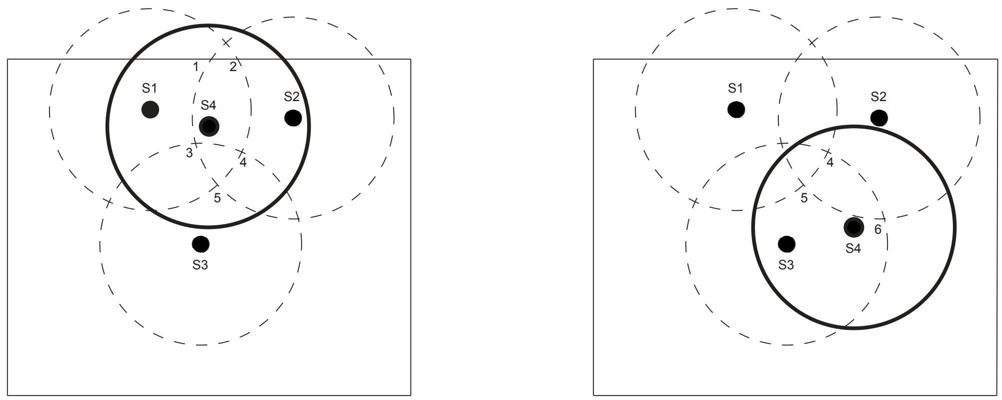
recovery time is maintained.

4.2.2 Node Self Scheduling Scheme

A node self scheduling scheme for sensor networks is presented in [51]. In this scheme, a node measures its neighborhood redundancy as the union of the sectors/central angles covered by neighboring sensors within the node's sensing range. At decision time, if the union of a node's "sponsored" sectors covers the full 360° (see Figure 11), the node will decide to power off. It should be noted that additional redundancy may exist between sensors and that the redundancy model is simplified at a cost of not being able to exploit this redundancy. At the beginning of each round, there is a short self-scheduling phase where nodes first exchange location information and then decide whether or not to turn off after some backoff time. Scenarios of unattended areas due to the simultaneous deactivation of nodes are avoided by requiring nodes to double check their eligibility to turn off after making the decision.

4.2.3 Coverage Configuration Protocol (CCP)

In CCP, an eligibility rule is proposed to maintain K-coverage [52]. First, each node finds all intersection points between the borders of its neighbors' sensing radii and any edges in the desired coverage area. The CCP rule assigns a node as eligible for deactivation if each of these intersection points is K-covered, where K is the desired sensing degree. The CCP scheme assumes a Span-like protocol and state machine that can use the Span rule for network connectivity or the proposed CCP rule for K-coverage, depending on the application requirements and the relative values of the communication radius and sensing radius. An example of how the CCP rule is applied is given in Figure 12. In Figure 12(a), node S4, whose sensing range is represented by the bold circle, must decide whether it should become active in order to meet a coverage constraint of $K = 1$. It is assumed that D knows that S1, S2, and S3, whose sensing ranges are represented by the dashed circles, are currently active. The intersection points within D's sensing range are found and enumerated 1-5 in the figure. Since S2 covers points 1 and 3, S3 covers points 2 and 4, and S1 covers point 5, S4 determines that the coverage requirements have already been met and remains inactive. In the case illustrated in Figure 12(b), there is an intersection point (labeled 6 in the figure) that is not covered by any of S4's neighbors. Thus, S4 must become active and sense the environment.



(a) (b)

Figure 12: Illustration of the CCP [52] activation rule for K-coverage, $K = 1$. Node S4 decides whether or not to activate in situations (a) and (b) knowing that neighbors S1, S2, and S3, are already active. In (a), Node S4 may remain inactive since all of its intersection points are K-covered. However, in situation (b), S4 must become active since intersection point 6 is not covered by any of its neighbors.

4.2.4 Connected Sensor Cover

The Connected Sensor Cover algorithm provides a joint topology control and sensing mode selection solution [53]. The problem addressed in this work is to find a minimum set of sensors and additional routing nodes necessary in order to efficiently process a query over a given geographical region. In the centralized version of the algorithm, an initial sensor within the query region is randomly chosen, following which additional sensors are added by means of a greedy algorithm. At each step in this algorithm, all sensors that redundantly cover some area that is already covered by the current active subset are considered candidate sensors and calculate the shortest path to one of the sensors already included in the current active subset. For each of these candidate sensors, a heuristic is calculated based on the number of unique sections in the query region that the sensor and its routers would potentially add and the number of sensors on its calculated path. The sensor with the most desirable heuristic value and those along its path are selected for inclusion in the sensor set. This process continues until the query region is entirely covered. The algorithm has been extended to account for node weighting, so that low energy nodes can be avoided, and to be implemented through distributed means, with little loss in solution optimality compared with the centralized version.

5 Time Synchronization and Localization Protocols

One of the main benefits of wireless microsensor networks is the spatial diversity that they provide, enabling applications such as target tracking in which a target's location and speed can be measured as it moves throughout the field where the sensors are deployed. However, such applications require two critical services — localization and time synchronization. These services could potentially be provided by installing GPS radios on the devices; however, in order to deploy microsensors on a mass scale, they should be very inexpensive devices. Furthermore, absolute position and time information is not necessary for many sensor network applications, as relative information can often suffice. If absolute information is necessary, a single or a few high resource nodes can be deployed in the network as references. Thus, there is a need for low-energy distributed algorithms that allows sensors to resolve relative location and time information. There has been a modest amount of research in these areas as wireless sensor networks have grown in popularity over the last several years.

5.1 Time Synchronization

To enable applications such as target tracking, sensor networks require time synchronization on a much finer scale than classic synchronization methods such as the Network Time Protocol (NTP) [54]. However, the energy constraints on sensor nodes require that the necessary improvement in synchronization be achieved while at the same time limiting message overhead. Several time synchronization algorithms are provided here that try to meet these goals simultaneously.

5.1.1 Römer's Algorithm

Römer was among the first to address the time synchronization issue for wireless ad hoc and sensor networks [55]. In the proposed algorithm, nodes do not regularly synchronize clocks; rather, when an event is sensed and a packet needs to be sent to the sink(s) within the network, the elapsed time since the event was originally sensed is updated within the packet along the path as the packet is routed toward the destination. The forwarding of messages is made somewhat complicated by the uncertainty in time estimation due to clock drift and non-deterministic delays involved in message transfer. Specifically, when transforming some computer clock time delay ΔC from node 1 to node 2, the delay must be estimated by node 2 as an interval $[\Delta C\frac{1-\rho_2}{1+\rho_1}, \Delta C\frac{1+\rho_2}{1-\rho_1}]$, where ρ_i represents the maximum clock drift of node i. When estimating the elapsed time since the event occurred, the receiving node must make an estimation of the transmission delay between when the packet was sent and when the acknowledgment was received by the previous node. While this estimation is simple to perform at the sending node, it is a bit less obvious at the receiving node. Referring to Figure 13, however, it can be seen that this estimation can actually be accomplished without requiring the sending node to send an extra packet explicitly indicating this delay. The round trip time between a sender and receiver can be estimated at the receiver by the interval $[0, (t_5 - t_4) - (t_2 - t_1)]$ (or $[0, (t_5 - t_4) - (t_2 - t_1)\frac{1-\rho_r}{1+\rho_s}]$ when accounting for clock skew). The time difference $(t_5 - t_4)$ is referred to as rtt_s and may be measured directly by the receiving node while the time difference $(t_2 - t_1)$ is referred to as $idle_s$ and may be piggybacked onto the message packet. It should be noted that this method of delay estimation makes use of two consecutive packet transmissions and the uncertainty in the delay increases with the inter-packet delay. Thus, if this delay is too large, it may be necessary to send dummy packets once in a while in order to make these estimations.

If s_i and r_i represent the local time at which node i sends and receives a message, respectively, time stamp estimation can be describe as follows. The time at which the event occurs can be estimated by node 2 (first hop) as the time that the packet was received by node 2 (r_2) minus the time that the packet was waiting at node 1 ($s_1 - r_1$). However, there is uncertainty in the transmission delays, which are lower bounded by 0 and upper bounded by $(rtt_1 - idle_1)$. Accounting for potential clock skews, the event's time stamp may be estimated at the second node as

$$[r_2 - (s_1 - r_1)\frac{1+\rho_2}{1-\rho_1} - (rtt_1 - idle_1\frac{1-\rho_2}{1+\rho_1}), r_2 - (s_1 - r_1)\frac{1-\rho_2}{1+\rho_1}] \quad (7)$$

This estimation process is repeated iteratively so that at the N^{th} node, the local estimate of the time of the event is

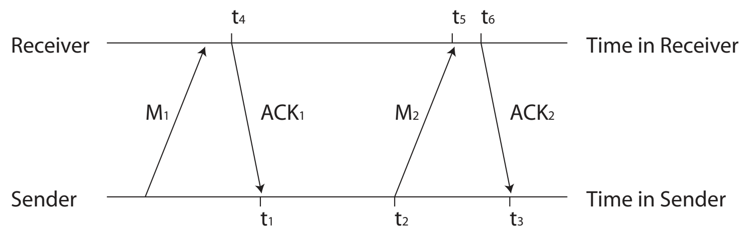


Figure 13: Timing diagram for message delay estimation needed in the algorithm proposed by Römer [55]. The sender may estimate message delay as $(t_3 - t_2) - (t_6 - t_5)$. The receiver may estimate the message delay as $(t_5 - t_4) - (t_2 - t_1)$.

$$[r_N - (1+\rho_N)\sum_{i=1}^{N-1}\frac{s_i - r_i + rtt_{i-1}}{1-\rho_i} - rtt_{N-1} + (1-\rho_N)\sum_{i=1}^{N-1}\frac{idle_i}{1+\rho_i}, r_n - (1-\rho_N)\sum_{i=1}^{N-1}\frac{s_i - r_i}{1+\rho_i}] \quad (8)$$

To implement this algorithm, the three summations in this interval are tracked and updated within the message packet from the source to the destination.

5.1.2 Reference-Broadcast Synchronization (RBS)

While Römer's time synchronization method enables fairly lightweight on-demand event synchronization for sensor networks, finer grained synchronization may be required in certain applications, such as target trajectory estimation. RBS allows nodes to synchronize their clocks to the resolution necessary for such sensor network applications [56]. Rather than broadcasting a time stamp in a synchronization packet as in protocols such as NTP [54], RBS allows the nodes receiving the synchronization packets to use the packet's time of arrival as a reference point for clock synchronization. Because most of the non-deterministic propagation time involved in transmitting a packet over a wireless channel lies between construction of the packet and the sender's transceiver (e.g., sender's queue delay, MAC contention delay, etc.), RBS removes most delay uncertainty involved in typical time synchronization protocols. For single-hop networks, the RBS algorithm is very simple. First, a transmitter broadcasts some number m reference broadcasts. Each receiver that receives these broadcasts exchanges the time that each reference broadcast was received locally with its neighbors. Nodes then calculate phase shifts relative to each other as the average of the difference of the time stamps of the nodes' local clocks for the m reference broadcasts. In multihop networks, time synchronization can be performed hop by hop between two nodes as long as the nodes on each link along the path have a common node whose reference broadcasts they can synchronize to.

5.2 Sensor Localization

Many localization algorithms for sensor networks require nodes to discover relative positioning information (e.g., distance estimations or directional estimations) of neighboring nodes. The ability to attain these estimates is provided in the radio module and is the basis of localization algorithms. Once local distance information is known, simple geometric relations can be used to calculate the local topology, which can then be subsequently disseminated throughout the network, providing globally coordinated localization [57]. This trilateration method is also used in the the Global Positioning System (GPS) [58].

A Received Signal Strength Indicator (RSSI) is one method that can be used to infer distances between nodes. However, this method is highly susceptible to errors, especially in environments prone to multipath propagation and shadowing effects. Time of Arrival (ToA), which is used in GPS, is another method that can be used; however, clocks on sensor devices may not be able to resolve propagation delays well enough to be able to acquire distance estimation with the required resolution. Time Difference of Arrival (TDoA), proposed for use in the Cricket platform [59], is a more practical method for estimating distances. In a TDoA system, an RF signal is transmitted simultaneously with an ultrasonic signal. The difference of times at which the signals are received can be easily translated to distance by multiplying by the difference in speeds. Finally, on sensors in which antenna arrays are used, Angle of Arrival (AoA) may be used to estimate directional information. In such networks, triangulation is used to find location estimations.

Locations can be estimated by forwarding the local constraints of all nodes in the network to a central server, which can then solve a large program to find location estimates [60, 61]. However, there has also been an effort to develop distributed localization protocols for sensor network, as they scale better and may be more practical for large-scale networks.

5.2.1 Reference Point Centroid Scheme

Among the first distributed localization schemes for wireless sensor networks was a simple scheme proposed by Bulusu et al. in which sensors listen for beacons that are broadcast from a few reference points in the network [62]. Sensors hearings these beacons compute their locations to be the centroid of the the locations of the reference points whose reference beacons they can hear.

5.2.2 Ad-Hoc Localization System (AHLoS)

In networks where beacon deployment is sparse enough that location estimations cannot be made directly by each sensor in the network, AHLoS allows nodes to iteratively resolve their locations through indirect means [63]. In this system, nodes that can acquire position information from and approximate distance to three or more neighboring beacons use a simple atomic multilateration technique. In this technique, nodes estimate their location so as to minimize the mean square error between the estimated location's distances to the beacons and the measured distances to the beacons. Nodes with location estimates in turn become beacon nodes and can be used by their neighbors for atomic multilateration. If a point in this iterative process is reached where no sensor with an unresolved location is within range of three or more beacons, a cooperative multilateration technique may be used in some cases. In the cooperative multilateration technique, nodes collaboratively solve an over-constrained problem with constraints based on approximated distances to each other and to beacon nodes. For example, in Figure 14(b), nodes 5 and 6 can receive messages from beacons 1 and 2, and beacons 3 and 4, respectively, as well as each other. It can be seen that based on the constraints imposed by the measured distances between each pair of communicating nodes, the positions of nodes 5 and 6 can be uniquely determined.

5.2.3 DV-Hop

In networks with extremely sparse beacon deployment, the DV-Hop algorithm may be used for node positioning [64]. DV-Hop relies on several landmark nodes located within the network that know their position information through GPS or manual programming. These landmarks first broadcast messages to each other, with the forwarding nodes keeping track of the number of hops these messages are forwarded. Each landmark then calculates a correction factor, which is the average distance to other landmarks (which it can calculate from its own known position and the

Figure 14: Scenarios when the AHLoS system [63] can use atomic multilateration (a) and collaborative multilateration (b).

positions advertised in the broadcast packets) that it is aware of divided by the average number of hops to these landmarks. The correction factors are then sent to nodes surrounding the landmarks by means of controlled flooding. Once the nodes in the network know this correction factor and the distance (in hops) to at least a few landmark nodes in the network, they can use triangulation techniques in order to calculate a position estimation for themselves.

An example of this procedure is shown in Figure 15. Here, landmark nodes L1, L2, and L3 broadcast beacons to each other to calculate the correction factor. If $L2$ and $L3$ are the only landmarks that $L1$ is aware of, it will set the correction factor to

$$C = \frac{d(L1, L2) + d(L1, L3)}{num\_hops(L1, L2) + num\_hops(L1, L3)} = \frac{1090m + 1170m}{6 + 7} = 174m/hop$$

The node $S1$ trying to find its position can now perform triangulation using the positions of the $L1$, $L2$, and $L3$ and the estimated distance to these landmarks, which are $d(S1, L1) = 174m \times 3 = 522m$, $d(S1, L2) = 174m \times 4 = 696m$, and $d(S1, L3) = 174m \times 4 = 696m$. While DV-Hop provides reasonably accurate location estimation in networks with sparsely deployed landmark nodes, it has been shown that location estimations acquired through methods similar to DV-Hop can be improved further through a refinement stage [65].

6 Open Issues

As can be seen by the numerous protocols discussed in this chapter, sensor networks provide many challenges not faced in conventional wireless networks and thus require a rethinking of all layers of the protocol stack. While the current body of work on sensor networks has enabled these networks to produce high quality results for longer periods of time, many open research issues still remain.

- **Appropriate QoS Model**. Due to the data-centric nature of sensor networks, describing QoS remains a challenge. In traditional networks, parameters like delay, packet delivery ratio and jitter can be used to specify application QoS requirements. In sensor networks, on the other hand, these parameters are replaced with ones like probability of missed detection of an event, signal-to-noise ratio and network sensing coverage. It is much more difficult to translate these data-specific QoS parameters into meaningful protocol parameters.

- **Cross-layer Architectures**. To make best use of the limited resources of the sensors, the entire protocol stack should be tailored to the specific needs of the sensor network application. Furthermore, the protocols should be integrated with the hardware, such that any hardware parameters are set to meet the sensor network goals and any protocols are adapted to the

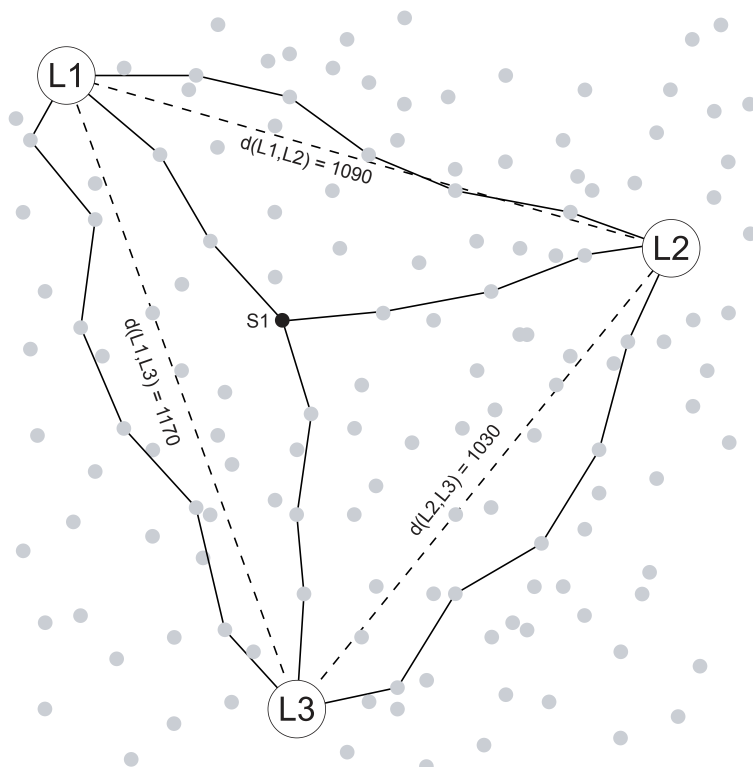


Figure 15: Position estimation in DV-Hop [64]. The landmark nodes $L1$, $L2$, and $L3$ first forward beacons to each other to find a correction factor in terms of distance per hop. Then node $S1$ can use this correction factor along with its known distance (in hops) to $L1$, $L2$, and $L3$ to find its own location.

specific features of the hardware. While this integrated approach can provide long network lifetime, it trades-off generality and ease of network design to achieve these lifetime increases.

- **Reliability**. In sensor networks, links and sensors themselves may fail, either temporarily or permanently. Designing protocols that provide reliable service in the presence of such failures is an important yet challenging problem.

- **Heterogeneous Applications**. The sensor nodes may be shared by multiple applications with differing goals. Sensor network protocols that can efficiently serve multiple applications simultaneously will be very important as the use of sensor networks increases.

- **Heterogeneous Sensors**. Much existing work assumes the network is composed of homogeneous nodes. Making best use of the resources in heterogeneous sensor networks remains a challenging problem.

- **Security**. Some initial work has focused on different aspects of security such as ensuring privacy and preventing denial-of-service attacks, but many open questions remain. How much and what type of security is really needed? How can data be authenticated? How can misbehaving nodes be prevented from providing false data? Can energy and security be traded-off such that the level of network security can be easily adapted? These and many other security-related topics must be researched to find low energy approaches to securing sensor networks.

- **Actuation**. Eventually sensor networks will "close the loop" by providing not only sensing capabilities but also the ability to automatically control the environment based on sensing results. In this case, data do not need to reach any sort of base station or sink points, and thus current models for sensor networks may not be valid. Research is needed to find good protocols for this new sensor network model.

- **Distributed and Collaborative Data Processing**. While much work has been done on architectures to support distributed and collaborative data processing, this is by no means a solved problem. One open question is how to best process heterogeneous data? Furthermore, how much data and what type of data should be processed to meet application QoS goals while minimizing energy drain? These and other questions remain to be solved.

- **Integration with Other Networks**. Sensor networks may indeed interface with other networks, such as a WiFi network, a cellular network, or the Internet. What is the best way to interface these networks? Should the sensor network protocols support (or at least not compete with) the protocols of the other networks? Or should the sensors have dual network interface capabilities? For some sensor network applications, these questions will be crucial and research is needed to find good solutions.

- **Sensor Deployment**. Given that sensor networks suffer from the "hot spot" problem due to the many-to-one traffic patterns, if it is possible to place the sensors at particular locations (or at least certain areas), how should the sensors be deployed so that both sensing and communication goals can be satisfied?

This list highlights just a few of the open research questions. Given the numerous current and envisioned applications for sensor networks, it is likely that research on ways to make these networks better will continue for years to come.

References

[1] C. Kidd et al. The aware home: A living laboratory for ubiquitous computing research. In *Proceedings of the Second International Workshop on Cooperative Buildings (CoBuild)*, 1999.

[2] S. Intille. Designing a home of the future. *IEEE Pervasive Computing*, 1(2):76–82, April 2002.

[3] L. Schwiebert, S. Gupta, and J. Weinmann. Research challenges in wireless networks of biomedical sensors. In *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2001.

[4] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson. Wireless sensor networks for habitat monitoring. In *Proceedings of the ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

[5] D. Steere, A. Baptista, D. McNamee, C. Pu, and J. Walpole. Research challenges in environmental observation and forecasting systems. In *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2000.

[6] D. Malan, T. Fulford-Jones, M. Welsh, and S. Moulton. Codeblue: An ad hoc sensor network infrastructure for emergency medical care. In *Proceedings of the International Workshop on Wearable and Implantable Body Sensor Networks*, 2004.

[7] University of rochester center for future health.

[8] R. Shah, S. Roy, S. Jain, and W. Brunette. Data MULEs: Modeling a three-tier architecture for sparse sensor networks. In *Proceedings of the First IEEE Workshop on Sensor Network Protocols And Applications (SNPA)*, 2003.

[9] J. Barros and S. Servetto. On the capacity of the reachback channel in wireless sensor networks. In *Proceedings of the IEEE Workshop on Multimedia Signal Processing*, 2002.

[10] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet. In *Proceedings of the Tenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASLOS)*, 2002.

[11] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang. MACAW: A media access protocol for wireless LANs. In *Proceedings of ACM SIGCOMM*, 1994.

[12] IEEE Computer Society LAN MAN Standards Committee. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. In *IEEE Std. 802.11*, 1999.

[13] S. Singh and C. Raghavendra. PAMAS: Power aware multi-access protocol with signalling for ad hoc networks. *ACM SIGCOMM Computer Communication Review*, 28(3):5–26, July 1998.

[14] Y. Wei, J. Heidemann, and D. Estrin. An energy-efficient mac protocol for wireless sensor networks. In *Proceedings of the Twenty-First International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2002.

[15] T. van Dam and K. Langendoen. An adaptive energy-efficient mac protocol for wireless sensor networks. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2003.

[16] G. Lu, B. Krishnamachari, and C. Raghavendra. An adaptive energy-efficient and low-latency MAC for data gathering in sensor networks. In *Proceedings of the Fourth International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks (WMAN)*, 2004.

[17] V. Rajendran, K. Obraczka, and J. Garcia-Luna-Aceves. Energy-efficient, collision-free medium access control for wireless sensor networks. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2003.

[18] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava. Optimizing sensor networks in the energy-latency-density design space. *IEEE Transactions on Mobile Computing*, 1(1):70–80, January 2002.

[19] S. Pattem, B. Krishnamachari, and R. Govindan. The impact of spatial correlation on routing with compression in wireless sensor networks. In *Proceedings of the Third International Symposium on Information Processing in Sensor Networks (IPSN)*, 2004.

[20] S. Singh, M. Woo, and C. Raghavendra. Power-aware routing in mobile ad hoc networks. In *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, 1998.

[21] J. Chang and L. Tassiulas. Energy conserving routing in wireless ad hoc networks. In *Proceedings of the Nineteenth International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2000.

[22] R. Shah and J. Rabaey. Energy aware routing for low energy ad hoc sensor networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2002.

[23] M. Perillo and W. Heinzelman. DAPR: A protocol for wireless sensor networks utilizing an application-based routing cost. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2004.

[24] W. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. In *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, 1999.

[25] C. Intanagonwiwat, R. Govindan, and Estrin D. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networks (MobiCom)*, 2000.

[26] D. Braginsky and D. Estrin. Rumor routing algorithm for sensor networks. In *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

[27] S. Ratnasamy and B. Karp. GHT: A geographic hash table for data-centric storage. In *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

[28] B. Karp and H. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2000.

[29] H. Takagi and L. Kleinrock. Optimal transmission ranges for randomly distributed packet radio terminals. *IEEE Transactions on Communications*, 32(3):246–257, March 1984.

[30] D. Niculescu and B. Nath. Trajectory based forwarding and its applications. In *Proceedings of the Ninth Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2003.

[31] P. Varshney. *Distributed Detection and Data Fusion*. Springer, New York, 1997.

[32] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on Wireless Communications*, 1(4):660–670, October 2002.

[33] J. Deng, Y. Han, W. Heinzelmaan, and P. Varshney. Balanced-energy sleep scheduling scheme for high density cluster-based sensor networks. In *Proceedings of the 4th Workshop on Applications and Services in Wireless Networks (ASWN)*, 2004.

[34] O. Younis and S. Fahmy. Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach. In *Proceedings of the Twenty-Third Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2004.

[35] P. Bonnet, J. Gehrke, and P. Seshadri. Querying the physical world. *IEEE Personal Communications*, 7(5):10–15, October 2000.

[36] C. Shen, C. Srisathapornphat, and C. Jaikaeo. Sensor information networking architecture and applications. *IEEE Personal Communications*, 8(4):52–59, August 2001.

[37] S. Madden, M. Franklin, J. Hellerstein, and W. Hong. TAG: a tiny aggregation service for ad-hoc sensor networks. In *Proceedings of the ACM Symposium on Operating System Design and Implementation (OSDI)*, 2002.

[38] S. Madden, M. Franklin, J. Hellerstein, and W. Hong. The design of an acquisitional query processor for sensor networks. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2003.

[39] Y. Tseng, Y. Chang, and P. Tseng. Energy-efficient topology control for wireless ad hoc sensor networks. In *Proceedings of the International Computer Symposium*, 2002.

[40] R. Ramanathan and R. Hain. Topology control of multihop wireless networks using transmit power adjustment. In *Proceedings of the Nineteenth International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2000.

[41] V. Rodoplu and T. Meng. Minimum energy mobile wireless networks. In *Proceedings of the IEEE International Conference on Communications (ICC)*, 1998.

[42] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for ad hoc routing. In *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2001.

[43] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *ACM Wireless Networks*, 8(5):481–494, September 2002.

[44] A. Cerpa and D. Estrin. ASCENT: Adaptive self-configuring sensor network topologies. In *Proceedings of the Twenty-First International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2002.

[45] X. Cheng A. Boukerche and J. Linus. Energy-aware data-centric routing in microsensor networks. In *Proceedings of the Sixth ACM/IEEE International Symposium on Modeling, Analysis, and Simulation of Wireless and Mobile Systems (MSWiM)*, 2003.

[46] S. Tilak, N. Abu-Ghazaleh, and W. Heinzelman. Infrastructure tradeoffs for sensor networks. In *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

[47] M. Perillo and W. Heinzelman. Sensor management. In C. Raghavendra, K. Sivalingam, and T. Znati, editors, *Wireless Sensor Networks*, pages 351–372. Kluwer Academic Publishers, 2004.

[48] C. Wan, A. Campbell, and L. Krishnamurthy. PSFQ: A reliable transport protocol for wireless sensor networks. In *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002.

[49] Y. Sankarasubramaniam, O. Akan, and I. Akyildiz. ESRT: Event-to-sink reliable transport in wireless sensor networks. In *Proceedings of the Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2003.

[50] F. Ye, G. Zhong, J. Cheng, S. Lu, and L. Zhang. PEAS: A robust energy conserving protocol for long-lived sensor networks. In *Proceedings of the Twenty-Third International Conference on Distributed Computing Systems (ICDCS)*, 2003.

[51] D. Tian and N. Georganas. A node scheduling scheme for energy conservation in large wireless sensor networks. *Wireless Communications and Mobile Computing Journal*, 3(2):271–290, March 2003.

[52] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, , and C. Gill. Integrated coverage and connectivity configuration in wireless sensor networks. In *Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2003.

[53] H. Gupta, S. Das, and Q. Gu. Connected sensor cover: Self-organization of sensor networks for efficient query execution. In *Proceedings of the Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2003.

[54] D. Mills. Internet time synchronization: the network time protocol. *IEEE Transactions on Communications*, 39(10):1482–1493, October 1991.

[55] K. Romer. Time synchronization in ad hoc networks. In *Proceedings of the Second ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2001.

[56] J. Elson, L. Girod, and D. Estrin. Fine-grained network time synchronization using reference broadcasts. In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (OSDI)*, 2002.

[57] S. Capkun, M. Hamdi, and J. Hubaux. GPS-free positioning in mobile ad-hoc networks. In *Proceedings Thirty-Fourth Annual Hawaii International Conference on System Sciences (HICSS-34)*, 2001.

[58] I. Getting. The global positioning system. *IEEE Spectrum*, 30(12):36–47, December 1993.

[59] N. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networks (MobiCom)*, 2000.

[60] L. Doherty, K. Pister, and L. Ghaoui. Convex position estimation in wireless sensor networks. In *Proceedings of the Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, 2001.

[61] Y. Shang, W. Ruml, Y. Zhang, and M. Fromherz. Localization from mere connectivity. In *Proceedings of the Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2003.

[62] N. Bulusu, J. Heidemannm, and D. Estrin. GPS-less low-cost outdoor localization for very small devices. *IEEE Personal Communications*, 7(5):28–34, October 2000.

[63] A. Savvides, C. Han, and M. Srivastava. Dynamic fine-grained localization in ad-hoc sensor networks. In *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2001.

[64] D. Niculescu and B. Nath. Ad hoc positioning system (APS). In *Proceedings of the Global Telecommunications Conference (GLOBECOM)*, 2001.

[65] C. Savarese, J. Rabaey, and J. Beutel. Robust positioning algorithms for distributed ad-hoc wireless sensor networks. In *Proceedings of the USENIX Annual Technical Conference*, 2002.