# Combs, Needles, Haystacks: Balancing Push and Pull for Discovery in Large-Scale Sensor Networks

***Abstract*— In this paper we investigate efficient strategies for supporting on-demand information dissemination and gathering in large-scale wireless sensor networks. In particular, we propose a "comb-needle" discovery support model resembling an ancient method: use a comb to help find a needle in sands or a haystack. The model combines push and pull for information dissemination and gathering. The push component features data duplication in a linear neighborhood of each node. The pull component features a dynamic formation of an on-demand routing structure resembling a comb. It enables us to investigate the cost of a spectrum of push and pull combinations for supporting discovery and query in large scale sensor networks. Our results shows that the optimal routing structure depends on the frequency of query occurrence and the spatial-temporal frequency of related events in the network. The benefit of balancing push and pull for discovery in large scale geometric networks are demonstrated. We also raise the issue of query coverage in unreliable networks and investigate how redundancy can improve the coverage via both theoretical analysis and simulation.**

## I. Introduction

The goal of this work is to investigate efficient and reliable routing mechanisms for supporting queries on large scale ad hoc wireless sensor networks. Recently, wireless sensor network has attracted a lot of attention from the research community [1], [2], [3], [4], [5], [6], [7], [8]. Many emerging sensor network applications involve dissemination of observed information to interested clients and demand efficient dissemination mechanisms due to resource limitations. For instance, a sensor network might be deployed for enhancing soldiers' battle field situation awareness when visibility is low (either in the night or due to smoke), as shown in Figure 1. A solider might be interested in where the tanks are in the battlefield. The nodes detecting the tanks can periodically push (broadcast) the information throughout the network in anticipation of soldier's needs, as shown in Figure 2(a). This push-based information dissemination strategy is efficient when there are many solders in the network constantly in need of the information. But when the demand for the information is low, this push-based strategy incurs high communication overhead since lots of broadcast bandwidth are wasted as the demand for the data is low. Alternatively, the system may choose a pull-based information dissemination strategy, as shown in Figure 2(b). The soldier broadcasts a query for the information when it is needed, and when the nodes that have the information receive query, they send the information to the soldier. Even though this pull-based strategy has exact query broadcast process, it is relatively more efficient than the push-based strategy when the frequency of query is relatively low compared to the frequency of the interested events, because communication takes place only when it is needed. A natural question is: can we combine the advantages of both push and pull strategies and build an efficient query-support mechanism that adapts to the frequencies of query and events?

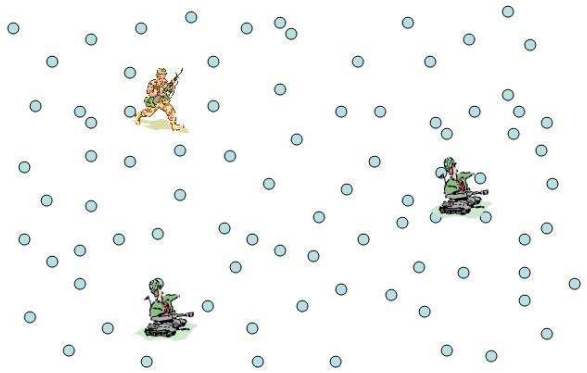


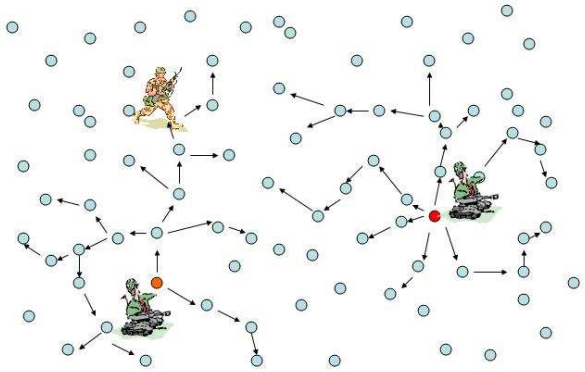
Fig. 1. A Query Example in Ad Hoc Networks

In this paper, we propose a novel "comb-needle" query support model that integrates both push and pull data dissemination, and analyze its the performance in large scale ad hoc wireless networks. Using this model, we are able to explore a whole spectrum of push and pull strategies shown in Fig.3. In the comb-needle model, each sensor node pushes its data to a certain neighborhood and the query is disseminated only to a subset of the network. More specifically, the query process builds a routing structure dynamically that resembles a comb and the sensor nodes push the data duplication structure like a needle, as illustrated in Figure 4. One can view this query process like combing for needles in the sands or haystack. The desirable properties of this mechanism include:

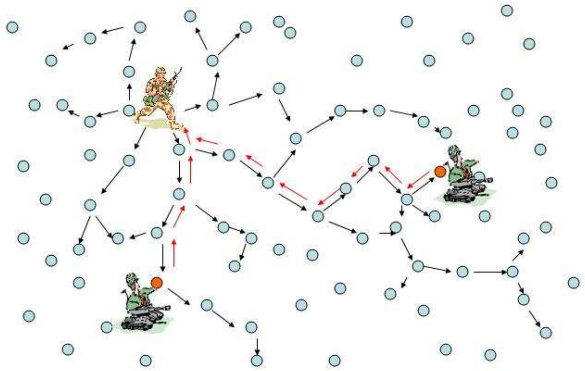
- In most cases, the comb-needle strategy is more efficient than both pure push and pure pull strategies. To some extend, pure push and pull strategies can be consider as the end points of the combing strategy.
- The combing granularity is dynamically adjustable based on demand. No infrastructure is built *a priori*.

We analyze the communication cost of the comb-needle strategy based on query frequency and event frequency, explore the issue of query coverage and reliability in the case of faulty networks, and study adaptive strategies for the case where the


Fig. 3. A Spectrum of Push-Pull Strategies

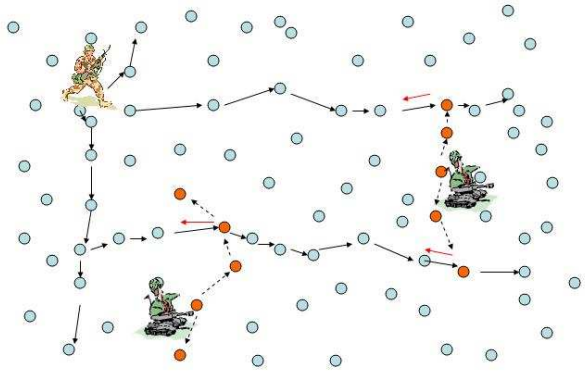


(a)

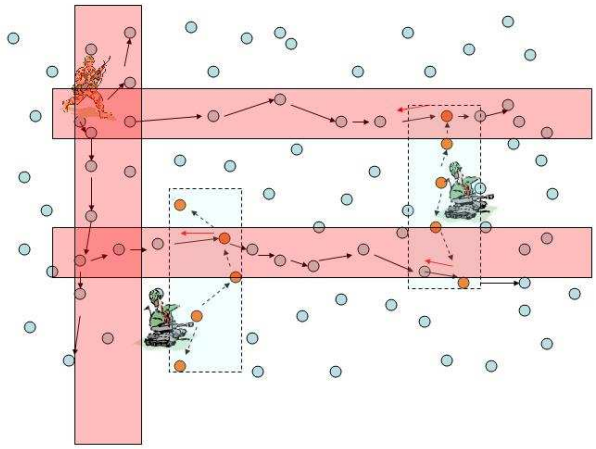


(b)

Fig. 2. (a) Pure push-based strategy (b) Pure pull-based strategy



(a)



(b)

Fig. 4. A Comb-Needle Example

frequencies of query and events are unknown *a priori* and time-varying.

This study adopts a few assumptions about the environment, the network and the application scenario:

- the event of interest can occur randomly at anywhere at any time.
- the query entry point can be anywhere in the network and occur at any time.
- the network is ad hoc and uniform, in the sense that all nodes are equivalent and the network does not have build-in hierarchy.
- the network is geometric and 2d. All nodes in the network have information of their locations in the 2d space.

The remainder of the paper is organized as follows: in Section II, we describe and analyze the comb-needle mechanism. In Section III, we present simulation results for supporting

queries on random grid networks. In Section IV, we explore the coverage issues in an unreliable network and present more robust strategies. We then propose the adaptive comb-needle strategy in Section V, which is followed by a related work and discussion section, and a conclusion section.

## II. Balancing Push and Pull

### A. Motivation

We consider the global discovery type query such as "where are all the tanks?", "what locations have a temperature exceeding 90 degrees?" etc. For such queries, the whole network need to be traversed by the underlying query resolution mechanism in order to get a complete response. Note that not all queries require a global discovery. Queries like "are there 5 tanks in the region?" could only traverse part of the network (up to the point where the answer is positive.)

For resolving a global query, a frequently used query dissemination mechanism is the flooding-based querying (FBQ)[9]. In FBQ, the underlying query support mechanism floods the request to the whole network. All nodes with relevant information respond. The flooding often represent the most significant message overhead for such queries. Note that we are only interested in one-shot query, rather than continuous query. In a continuous query, directed diffusion type of schemes work sufficiently well since the initial flooding cost can be amortized over the duration of the continuous information flow from the sensor nodes to the querier.

FBQ represents a pull-based query support mechanism. As noted in the introduction, there is a potential advantage in combining a push based strategy with the pull based strategy to increase query efficiency. For instance, let's consider a simple grid network like the one in Figure 5. Nodes $(4, 2)$ and $(7, 6)$ detected some abnormal event $E$, and push the information vertically in the networks, and have it temporarily stored on the nodes traversed. Node $(0, 8)$ is an entry point of a discovery query interested in such event. Note that the query only needs to be disseminated horizontally to be resolved, i.e., the information pull only need to occur horizontally rather than throughout the network. Assume that the events are only detected once at the two nodes, and the query only occurred once as well, the total message cost for supporting the query is about 30 by this push-pull model, comparing the approximately 100 messages needed for the FBQ method. In the present hardware state, communication cost often weights much more than the storage cost, so this push-pull strategy seems desirable. One also can see that the relative benefit of this push-pull strategy really depends on the relative rates of occurrence of the events and query in the network. If there's more queries, there is more saving in the message overhead for supporting the queries. In this section we introduce a more general pull-push model and analyze its performance.

### B. The Combing Strategy

The push-pull query support model we propose resembles an action of combing for needles in a haystack or in a pool of sand, and is thus dubbed as the "comb-needle" model. The

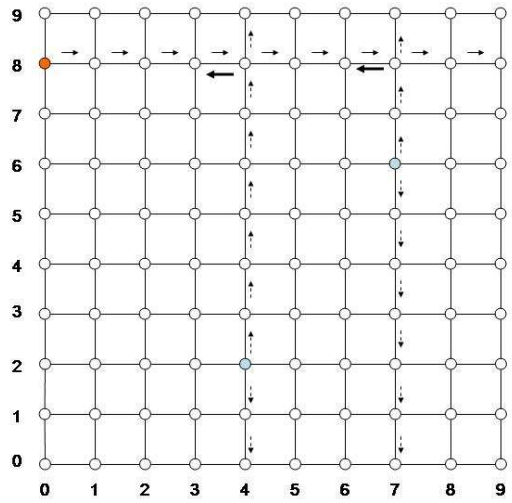



Fig. 5. A Special Push-Pull Strategy

comb-needle query support model combines both push and pull in the following way: each sensor node pushes its data to a certain neighborhood and the query node disseminates its request to a subset of the network. More specifically, the query process builds a routing structure dynamically that resembles a comb and the sensor nodes push the data duplication structure like a needle, as illustrated in Figure 4.

To analyze mathematically the cost and benefit of a query support mechanism, we need to have a model of the world that the sensor network is monitoring. In this paper, we assume the following random event and query model: events occur uniformly in space and time across the sensor network, and the discovery query entry point can be any node in the network with the same probability. Accordingly, we define two parameters:

$f_q$: the arrival frequency of discovery queries.
$f_e$: the arrival frequency of relevant events.

### C. Analysis of the Combing Strategy

For simplicity, like in the earlier example, first consider a grid network with $n^2$ nodes located at (i,j) where $0 \leq i, j < n$. Each node sends its state update to $2l$ of its vertical neighbors. For instance, node $(i, j)$ will send its state update to nodes $(i, j + 1)$, $(i, j + 2)$, ..., $(i, j + l)$ and $(i, j - 1)$, $(i, j - 2)$, ..., $(i, j - l)$. An example is shown in Fig. 6 with $l = 2$. Each data push incurs cost

$$C_l \sim 2l \tag{1}$$

in terms of the number of hops. We assume per-hop communication costs for the query propagation and data propagation are the same.

The query process first sends the query vertically then fans out horizontally. For instance, assume that node $(0, 0)$ in Fig. 7 is the query source. A query is sent vertically from $(0, 0)$ to $(n - 1, 0)$, and it is also fanned out horizontally from nodes $(0, 0)$, $(s, 0)$, $(2s, 0)$,..., $(\lfloor \frac{n-1}{s} \rfloor s, 0)$, and $(n - 1, 0)$.

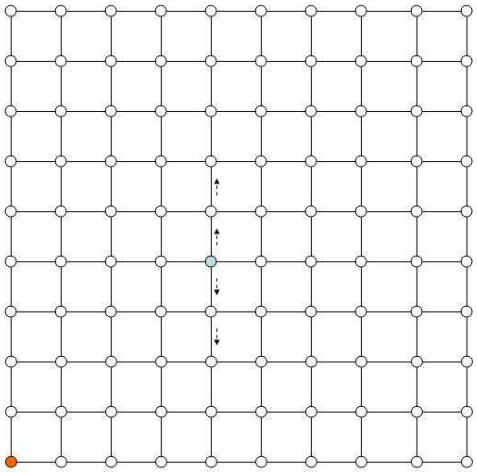


Fig. 6. Vertical Data Duplication

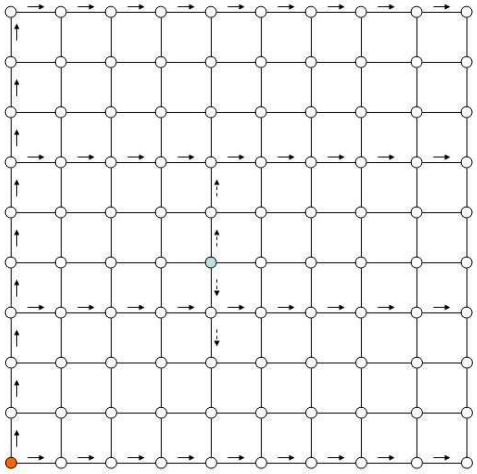


Fig. 7. Combing

The resulting routing structure looks like a comb. The query dissemination cost $C_{qd}$ is

$$C_{qd} = n - 1 + (n-1)(\lfloor \frac{n-1}{s} \rfloor + 1) \quad (2)$$

Note that if $s = 2l + 1$, the query dissemination reaches its finest level, in the sense that it can collect the sensing data from every node.

The cost of query response depends on the reply scheme used including data aggregation strategies used, if any. For simplicity, we first consider the simplest reply scheme without data aggregation. Each node on the comb replies immediately if it has the matching data, including those data pushed from the nodes not on the comb. The reply paths are the reverses of the comb growing paths for the simplicity of analysis. In reality, the reply path could be a shorter one found by any ad hoc geometric routing algorithm, since the location of the destination is known.

Assume there is a match in node $(i, j)$, where $i = 0, s, 2s, ....$ The reply cost is

$$c_{qr} = j + i \quad (3)$$

In the case when there is more than one matching data in a time window of $\tau$, and they are uniformly distributed in the network with density $f_e\tau/n^2$, the total reply cost is about:

$$C_{qr} \sim (n-1) * f_e\tau \quad (4)$$

Assume that the query frequency is $f_q$, the data event frequency is $f_e$, and $\tau = 1$ (in a unit time window). The total cost per query:

$$\begin{aligned} C &= C_{qd} + C_{qr} + f_e * C_l/f_q & (5)\\ &\simeq C_{qr} + n - 1 + \frac{(n-1)^2}{s} + s * \frac{f_e}{f_q} & (6)\\ &\geq C_{qr} + n - 1 + 2(n-1)\sqrt{\frac{f_e}{f_q}} & (7) \end{aligned}$$

This minimum of the cost is around

$$s_{optimal} \sim (n-1)\sqrt{\frac{f_q}{f_e}} \sim \sqrt{N}\sqrt{\frac{f_q}{f_e}} \quad (8)$$

where $N$ is the total number of nodes in the network. One can see that the result tells us that if the data frequency is relatively higher than the query frequency, the comb should be finer, i.e., with smaller inter-spike spacing $s$. In other words, when we have more queries and less related events/data, the "combing degree" $S = s$ should be larger and the "push degree" $L = 2l+1$ should be larger, i.e., more push and less pull is better in such scenarios. On the other hand, when there are more events and less queries, $L$ and $S$ should be smaller. Furthermore, note that the scenario shown in Figure 7 is a global-pull-local-push one. From formula 8 we can see that when $f_q > f_e$, $s_{optimal} \sim> \sqrt{N} \sim n$. What really means is that when $f_q > f_e$, the global-pull-local-push model is no longer an optimal one, but rather, local-pull-global-push scenarios become more economic. We will discuss this in more detail in Section VI. Henceforth, we focus on the $f_q \leq f_e$ scenario. The case where $f_q < f_e$ is essentially a symmetric one.

Note that the baseline cost (using FBQ) per query is $O(N)$. So, comparing to the naive flooding based approach, this comb-needle scheme dramatically reduces query cost to $O(\sqrt{N})$. Note that the possibility of achieving this cost reduction partly relies on some knowledge about the expected data and query frequency. Note also that the duplication of data in neighboring nodes incurs some storage cost as well. In this study, we focus on communication cost by assuming communication (energy) is a more restricted resource than storage in sensor networks. Last, throughout the paper, we assume that the per-hop communication costs for query propagation, data duplication, and data report are the same. Such an assumption can be easily generalized by including a weight in the communication cost.

## III. Simulation

In this section, we verify the theoretical results given in the previous sections via simulation. While in the previous section we make simplified assumption about the network model, in simulation we adopt a more realistic one.

### A. Radio Propagation Model

The radio propagation model determines the strength of a transmitted signal at a particular point of the space for all transmitters in the system. Based on this information the signal reception conditions for the receivers can be evaluated and collisions can be detected. The transmission model is given by:

$$P_{rec,ideal}(d) \leftarrow P_{transmit}\frac{1}{1+d^{\gamma}}, \quad where\ 2 \leq \gamma \leq 4 \qquad (9)$$

$$P_{rec}(i,j) \leftarrow P_{rec,ideal}(d_{i,j})(1+\alpha(i,j))(1+\beta(t)) \qquad (10)$$

where $P_{transmit}$ is the signal strength at the transmitter and $P_{rec,ideal}(d)$ is the *ideal* received signal strength at distance $d$, $\alpha$ and $\beta$ are random variables with normal distributions $N(0,\sigma_\alpha)$ and $N(0,\sigma_\beta)$, respectively. A network is asymmetric if $\sigma_\alpha > 0$ or $\sigma_\beta > 0$. A node $j$ can receive a packet from node $i$ if $P_{rec}(i,j) > \Delta$ where $\Delta$ is the threshold. There is a collision if two transmissions overlap in time and both could be received successfully. Furthermore, an additional parameter $p_{error}$ models the probability of a transmission error by any unmodeled effects.

### B. Topology Model

While our analysis in the previous section is based on a regular grid network, in simulation we use a random grid topology model which is more realistic in some deployment scenarios. An instance of the random grid network is generated as follows, first the nodes are put on a regular grid, then the $x$ and $y$ coordinates of each nodes is shifted independently via a random offset generated from a uniform distribution.

### C. Routing Protocol

Unlike in a regular grid network, building the comb-needle querying in a random grid network needs extra attention, since the "left-right-up-down" in the random network is no longer well-defined, and the "combs" and the "needles" can no longer be straight lines. To overcome this problem, we build approximations of the combs and needles using a Constrained Geographical Flooding (CFG) method, similar to the illustration in Figure 4(b). The "needles" and the spikes on the "comb" has a width of $W$, besides their lengths.

CGF is robust and also simple to implement. In CGF, whenever a new packet arrives, each node will decide if it should rebroadcast the packet according to the geographical constraints for the type of packet. More specifically, for a query packet, a node will broadcast only if it is at the comb branch, for an event packet, a node will broadcast only if it is vertically within the duplication distance to the source; for a report packet, a node will broadcast only if it is at the comb branch and closer to the query node. If the radio range is larger than the grid distance, duplicated number of packets are received at each node while the total number of packets are still bounded by the size of comb or needle. This increases the coverage and thus the robustness. The pseudo code of the protocol is shown in Figure 8, where $S$, $L$ and $W$ are the inter-spike distance, the needle push length and the CFG width, respectively.

**received** query $q$ at $w$ **do**
  **if** $new(q)$ and $atComb(q.x, q.y, w.x, w.y, S, W)$ **then**
    **broadcast** $q$;
    $events = getEvents(q)$;
    **if** $events \neq \emptyset$ **then**
      **broadcast** $r(events, q.x, q.y, w.x, w.y)$;
    **end**
  **end**
**end**

**received** event $e$ at $w$ **do**
  **if** $new(e)$ and $atNeedle(e.x, e.y, w.x, w.y, l, W)$ **then**
    **copy** $e$ to memory
    **broadcast** $e$;
  **end**
**end**

**received** report $r(events, q_x, q_y, s_x, s_y)$ at $w$ **do**
  **if** $new(r)$ and $q_x = w.x$ and $q_y = w.y$ **do**
    **return** events;
  **end**
  **if** $new(r)$ and $atPath(q_x, q_y, s_x, s_y, w.x, w.y, W)$ **then**
    **broadcast** $r$;
  **end**
**end**

**function** $out = atComb(q_x, q_y, w_x, w_y, S, W)$
$d = \mod(|q_y - w_y|, S)$;
$out = |q_x - w_x| < W$ or $\min(d, S-d) < W$;

**function** $out = atNeedle(e_x, e_y, w_x, w_y, L, W)$
$out = |e_x - w_x| < W$ and $|e_y - w_y| < L + W$;

**function** $out = atPath(q_x, q_y, s_x, s_y, w_x, w_y, W)$
$out = |s_y - w_y| < W$ and $(w_x - s_x)(q_x - w_x) > 0$ or
  $|q_x - w_x| < W$ and $(w_y - s_y)(q_y - w_y) > 0$

Fig. 8. Constrained geographical fboding for the comb-needle query model

### D. The Simulation Environment

Prowler [10] is used for the simulation of comb-needle query model. Prowler [10] is a probabilistic wireless network simulator based on the event-driven structure. It can be set to operate in either deterministic mode (to produce replicable results while testing the application) or in probabilistic mode that simulates the nondeterministic nature of the communication channel and the low-level communication protocol. In

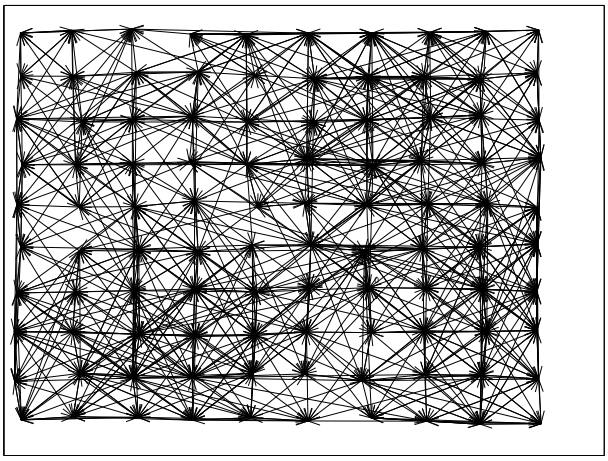


Fig. 9. An instance of network connectivity

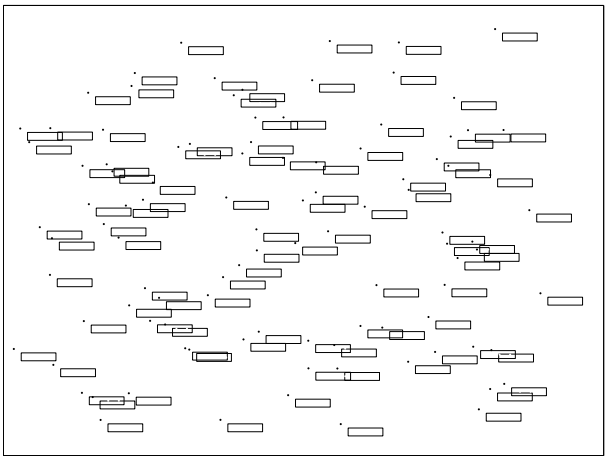


Fig. 10. A random network (grid with large offset)

particular, Prowler provides the radio model in (9,10) and a MAC layer that simulates the Berkeley motes' CSMA protocol, including random waiting and back offs.

A topology model, an application model and a performance model are developed on the top of Prowler. The topology model is used to generate networks with grid or random topologies, the application model is to specify properties and locations of sources and destinations, source/event rate and query rate, etc. The performance model provides performance metrics such as latency, success rate/loss rate, throughput, and energy consumption (for simplicity, we assume each transmission cost one unit of energy).

### *E. Performance Evaluations*

Our simulations are all performed on a 10x10 network using Prowler's default radio model, with $\sigma_\alpha = 0.45$, $\sigma_\beta = 0.02$, $p_{error} = 0.05$ and $\Delta = 0.1$. Each test is performed with 10 runs and then averaged (with query and event locations randomly selected). The transmit signal strength is set to 1, so that the maximum radio range is about $3d$, where $d$ is the distance between two neighbor nodes in the grid. Figure 9 shows an instance of the network connectivity of this radio model for a grid network with small random offset. Two separate tests are done in this environment. The first test is to discover what is the best spacing for the comb, given query rate and event rate, so as to verify the theoretical results discovered in the paper. The second test is to show the robustness of the protocol with a varying query width for a grid network with large random offset (Figure 10).

The first test was run on a network like Figure 9. We first let event rate be 1 packet per second (p/s), and query rate be 0.1 p/s ($f_q/f_e = 0.1$). Let the push length $l$ be 0, 1, 2, 3 and comb spacing $S$ be $1 + 2l$, i.e., 1, 3, 5, 7, respectively. The simulation runs 100 seconds and Figure 11 shows the result of energy consumption. In this case, $l = 1, S = 3$ has the minimum energy consumption. This results matches well with the predication using formula (8). We then reduce the event rate to 0.1 ($f_q/f_e = 1$). Figure 12 shows the result

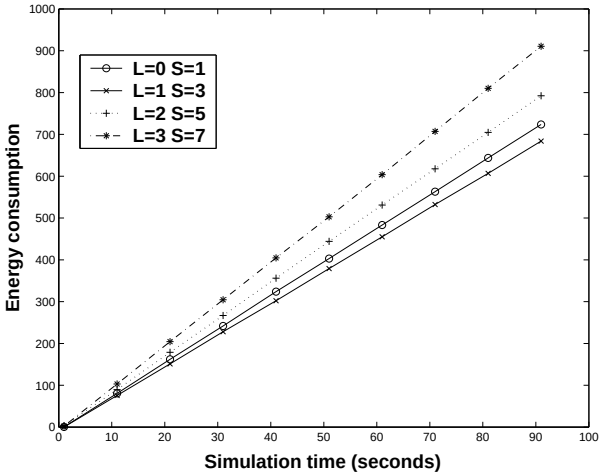



Fig. 11. Energy consumption: $f_q = 0.1$, $f_e = 1$

of energy consumption. In this case, $S = 5$ and $S = 7$ are close (with $S = 5$ slightly better) to be the minimum energy consumption. It is partly due to the boundary effect, since both $S = 5$ and $S = 7$ have the same number of nodes in the comb ($\lfloor \frac{9}{5} \rfloor = \lfloor \frac{9}{7} \rfloor$, but $S = 5, l = 2$ has shorter needles.

The second test was run on a network like Figure 10. In this case, we let $f_q = 0.1$, $f_e = 1$, $l = 1$ and $S = 3$, and set the query width $W$ to 0.5, 0.6, 0.7, 0.8, and 0.9. Figure 13 shows the success rate with respect to different CGF width. We can see that the wider the $W$ is, the higher the success rate. Figure 14 shows corresponding the energy consumption; the wider the cost is higher. Clearly there is a trade-off between the success rate and the communication cost. Note that when $W$ goes to 1.5, the combing is similar to flooding. This simulation results also raise the issue of reliability of the query support routing mechanism. Due to the nature of wireless communication in distributed networks, the link between the nodes are not totally reliable. For instance, there is certain probability that collisions may occur and data transmission is not successful. This led

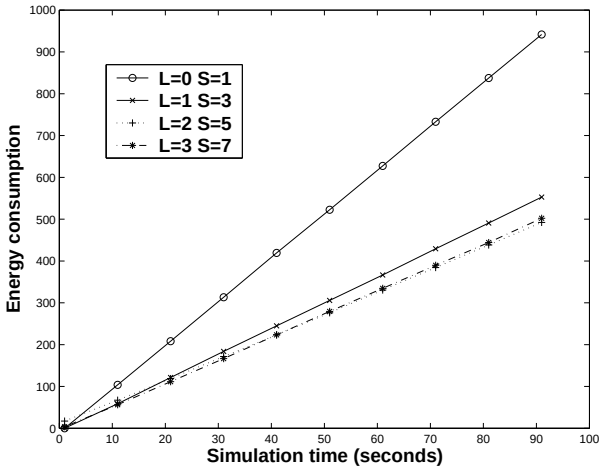



Fig. 12. Energy consumption: $f_q = 0.1$, $f_e = 0.1$

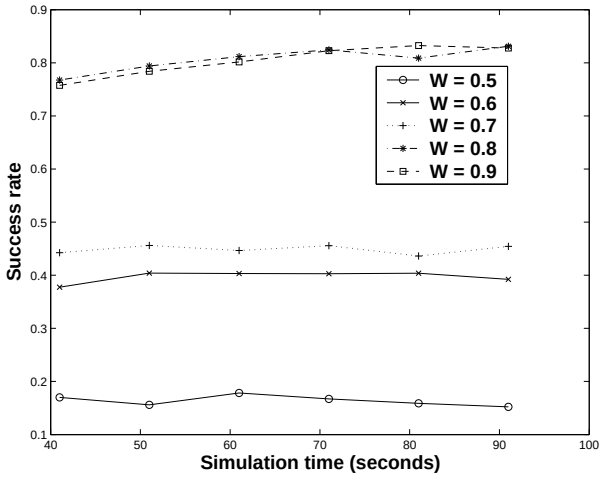



Fig. 13. Success rate for different query widths

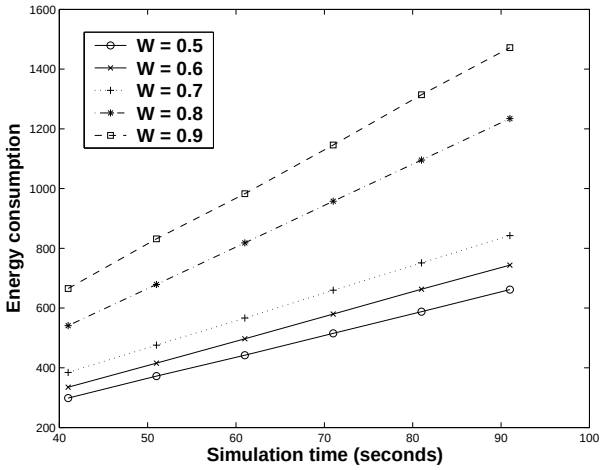



Fig. 14. Energy consumption for different query widths

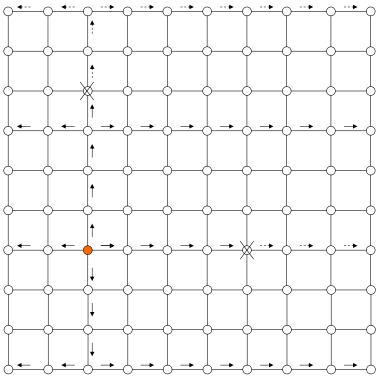


Fig. 15. Query coverage

us to investigate the impact of link failures on the routing structure and the issue of query support reliability.

## IV. Reliability

As discussed in the last section, wireless links are unreliable due to the nature of wireless communications. In addition, sensor nodes are deployed in harsh environment unattended, and thus prone to failures. In this section, we first analyze the effect of node failures on the query and report, and then study reliable query-and-report strategies, namely, local re-enhancement and spacial redundancy schemes.

### *A. Query Coverage*

First, we analyze the effect node failures on the query coverage. There are usually two types of failures: transient failure and permanent failure. Because wireless channels are unreliable and time-varying [12], [13], one possible cause of a transient failure is the fading environment, which is modeled by Eq. (10) in our simulation in Section III. When the channel between the transmitter and the receiver is in deep fading or collisions occur during the transmission, link-layer failures occur. Examples of permanent failure include the situations where nodes run out of energy or are physically damaged. We assume that each node has a failure probability $p$, which is independent of other nodes. (The independent assumption is discussed later.) Note that $p$ can be a function of time and model the time-evolving reliability of the network: as time elapses, more nodes run out of battery and become unreliable. We look at a snapshot of the network, and classify nodes as failure nodes and active nodes.

We consider the general case where any node can generate a query request. The question is how large an area can receive the query given the node failure model. The route of a query propagation in Figure 15. If a node fails, then the query propagation stops. In the figure, we indicate two node failures. The dashed lines with arrows indicate the designed query path that fails because of node failures. This is a simple query propagation scheme.

Now we calculate the average coverage area of queries by node $(i, j)$. Let $A_1(j)$ be the number of horizontal lines a

query message initiated by a node at row $j$ reaches and $A_2(i)$ be the number of nodes that a query reaches in a horizontal line initiated by a node in column $i$. Note that $A_1(j)$ and $A_2(i)$ are random variables and let $a_1(j)$ and $a_2(i)$ be their expectations respectively. Let $N(i,j)$ be the average number of nodes that a query from $(i,j)$ can reach.

$$\begin{aligned} N(i,j) &= E\left(\sum_{k=1}^{A_1(j)} A_2(i)\right) \\ &= E(A_1(j))E(A_2(i)) \\ &= a_1(j)a_2(i). \end{aligned}$$

Because $A_1(j)$ and $A_2(i)$ are independent, the second equation holds by the Wald's equation (p. 417, [14]). We have

$$\begin{aligned} & a_1(j) \\ = \; & 1 + \sum_{k=1}^{\lfloor \frac{i}{s} \rfloor} [(1-p)^s]^k + \sum_{k=1}^{\lfloor \frac{n-j-1}{s} \rfloor} [(1-p)^s]^k \\ = \; & \frac{1-(1-p)^{s(\lfloor \frac{i}{s} \rfloor + 1)}}{1-(1-p)^s} + \frac{1-(1-p)^{s(\lfloor \frac{n-j-1}{s} \rfloor + 1)}}{1-(1-p)^s} - 1, \end{aligned}$$

and

$$\begin{aligned} a_2(i) &= 1 + \sum_{k=1}^{i} (1-p)^k + \sum_{k=1}^{n-1-i} (1-p)^k \\ &= \frac{1-(1-p)^{i+1}}{p} + \frac{1-(1-p)^{n-i}}{p} - 1, \end{aligned}$$

where $\lfloor x \rfloor$ denotes the largest integer that does not exceed $x$. Note that $a_1((n-1)/2) \leq a_1(j) \leq a_1(0)$ and $a_2((n-1)/2) \leq a_2(i) \leq a_2(0)$ for all $j$ and $i$. In words, queries from nodes in the middle of the grid can reach more nodes on average than nodes near the boundary. For convenience, we assume that $n$ is an odd number. If $n$ is even, then indexes $n/2$ and $n/2-1$ have the same property.

Ignoring the boundary effect, the percentage of nodes covered by the query of node $(i,j)$ is given by

$$\eta(i,j) = a_1(j) * a_2(i) * s/n^2.$$

Here $a_1(j) * a_2(i)$ is the average number of nodes that receive the query in the comb. Each node reached on a horizontal line can cover $2l+1 = s$ nodes, including itself. In Figure 16, we show the coverage of nodes $(i,j)$ where $0 \leq i,j \leq n-1$.

### B. Local Enhancement

In the previous section, we study the coverage property of the query message in the existence of node failures. When there exists a single route between the query node and the data node, any failure in the route will result the failure of the query. As the number of hops increases, the failure probability increases dramatically. We now discuss more reliable query-and-report schemes. The basic idea of providing reliability is to use redundancy. There are usually two types of schemes to enhance the reliable transmission of a message.

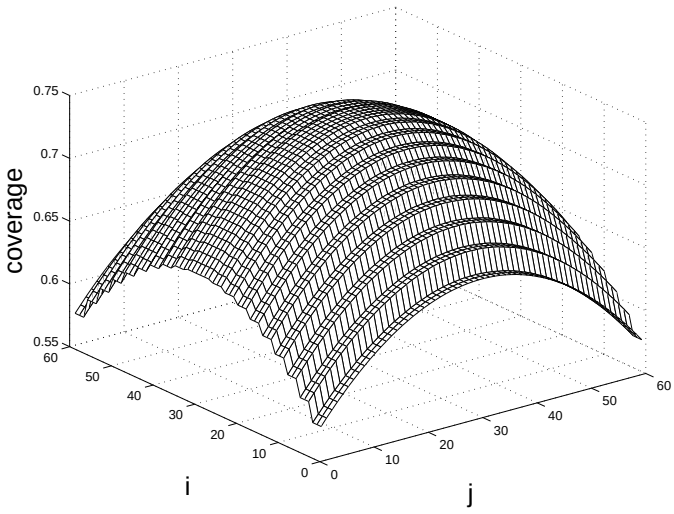



Fig. 16. Query Reachability in the Network

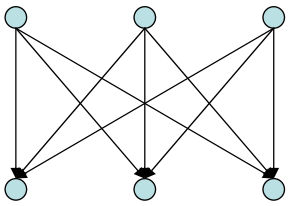


Fig. 17. Interleaved mesh

*a) Link Enhancement:* Reliable data delivery through vulnerable sensor networks has been a research challenge. In particular, the authors of [15] propose the usage of interleaving paths to increase robustness. Figure 17 shows an interleaving mesh, where paths in the mesh are implicit and interleaving. In this design, all nodes send data to all nodes at the next hop. The performance of such interleaving mesh is shown to be very robust against both independent link-layer packet loss and node failures. A necessary and sufficient condition for the route to exist is that there exists at least one active node in each hop. If each hop has $k$ nodes, and each node has a failure probability of $p$, then the hop fails with probability $p^k$, where $p^k << p$. The interleaves mesh is even more effective against independent link failures. For example, give a packet-loss probability of 0.3, 95% of packets can travel more than 200 hops if each hop has three nodes [15]. CFG used in the simulation in Section III exploits this type of strategy in a random network. We should note that the interleaved mesh does require additional power consumption for communication overhead and to keep more nodes awake, although the broadcast nature of wireless communication can be exploited,

*b) Reroute:* Another approach to provide more reliable query and report is to develop alternative routes. Consider the case where node $(i,j)$ is communicating with node $(i,j+1)$. In the previous studies, we assume that when a node $(i,j+1)$ fails, the query propagation ends there. However, after leaning

the failure of node $(i, j+1)$, node $(i, j)$ may detour the route as $(i+1, j) \rightarrow (i+1, j+1) \rightarrow (i+1, j+2) \rightarrow (i, j+2)$. Algorithms studied in the literature for developing alternative/backup routes can be applied here to enhance the transmission reliability.

Both the link enhancement and reroute schemes have their advantages and disadvantages. Link enhancement is simple and has low overhead. There is minimum requirement on the MAC/routing protocols. For example, no retransmission or acknowledgment is required. (Of course, retransmission and acknowledgment can improve the reliability at the cost of delay and energy consumption.) The functionality of each node can be simple listening and forwarding. On the other hand, the reroute scheme requires more routing functionality at each node and possibly a routing table be attached to the message. To develop alternative routes incurs additional overhead and delay in exchanging information. As discussed earlier, there are transient failures and permanent failures. For transient failures, especially in small time-scale, simple redundancy scheme may preform better while reroute is preferred for permanent failures. Another important factor is the query source. If the queries are generated by a small number of fixed nodes, it may worth the overhead to develop alternative routes, especially in the existence of permanent node failures. If the queries are generated by a large number of nodes randomly, it may be more cost-effective to use the link enhancement schemes.

### C. Spacial Diversity

We note that failures may not be independent and randomly in a network as usually modeled. Node failure can be highly correlated. For example, sensor nodes in one area may have a much higher probability of failure than other areas due to some unusual incidents, such as water flooding, overheat, etc. In addition, node failure can propagate. When a node fails, it neighboring nodes have to share the sensing and communication burden of the failed node. They consume more energy and die quickly. Thus, node failures propagate in the network. The previous schemes, such as interleaving mesh, are not designed specifically to address such correlated node failures in the network. Thus, we propose the use of spacial diversity to improve the reliability and coverage of query.

In Figure 18, we show a construction of query paths with spacial diversities. Basically, an additional vertical query propagation path is constructed. In each horizontal line, between the two vertical lines, the query message can be passed in either direction. If a node receives the same query message from both directions, it stops the message propagation. The basic idea here is to provide spatial diversity and redundancy while local enhancement can be used to for each path. If one part of the network is seriously damaged beyond local enhancement, say one of the vertical lines is gone, nodes in other part of the network can still receive query message propagated from the other vertical line.

Next, we calculate the probability of a node that receives the query message. We assume that a node is the center of

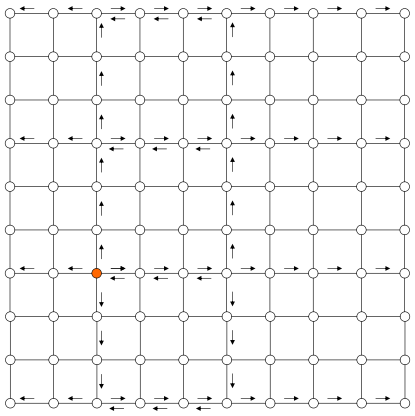


Fig. 18. level-2 redundancy

a (large-scale) failure with probability $p$. Assume that the query message is generated by node $(x, y)$. Let $p(i, j|(x, y), k)$ be the probability that a node at $(i, j)$ receives the query message when there is an additional vertical line at $(k, y)$, $i = 0, \cdots, n-1$ and $j = 0, s, 2s, \cdots$. Without loss of generality, we consider the case $0 \leq x < n/2$ and $k > x$. We consider three different cases, where $i < x$, $x \leq i < k$, and $k \leq i$. When $i < x$, the node is on the left side of both vertical lines. We have

$$\begin{aligned} c_1 &= (1-p)^{|y-j|} \\ c_2 &= (1-p)^{2(k-x)+|y-j|} \\ p(i, j|(x, y), k) &= (c_1 + c_2 - c_1 c_2)(1-p)^{x-i}. \end{aligned}$$

When $y \leq j < k$, i.e., the node is between two vertical lines, we have

$$\begin{aligned} c_1 &= (1-p)^{|y-j|+i-x} \\ c_2 &= (1-p)^{k-x+|y-j|+k-i} \\ p(i, j|(x, y), k) &= c_1 + c_2 - c_1 c_2. \end{aligned}$$

When $j \geq k$, the node is on the right side of both vertical lines. We have

$$\begin{aligned} c_1 &= (1-p)^{|y-j|+k-x} \\ p(i, j|(x, y), k) &= (2c_1 - c_1^2)(1-p)^{i-k}. \end{aligned}$$

The average query success probability is

$$p(x, y) = \max_k \sum_{i,j} p(i, j|(x, y), k).$$

For nodes on the right-hand side of the vertical lines, $p$ increases as $k$ increases. For all other nodes, $p$ increases as $k$ decreases. However, numerical results show that the average success probability is not sensitive to the value of $k$. In Figure 19, we show the query success probability of all nodes when the optimal value of $k$ is used. We have $n = 60$, $p = 0.01$, and $s = 3$, the same as in Figure 16. Compare with Figure 16, we observe that the query success probability has been improved significantly.

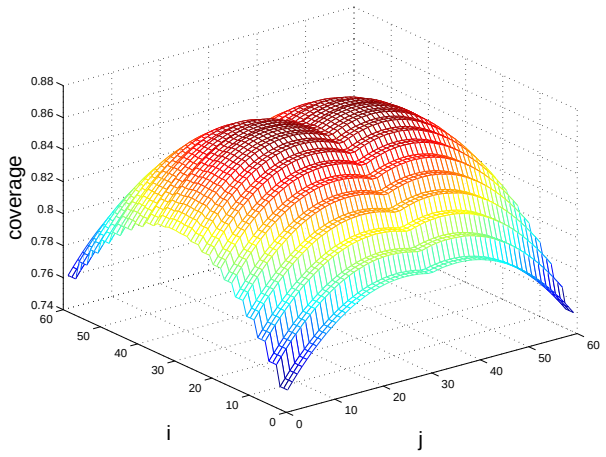



Fig. 19. Coverage probability when an additional vertical line is placed.

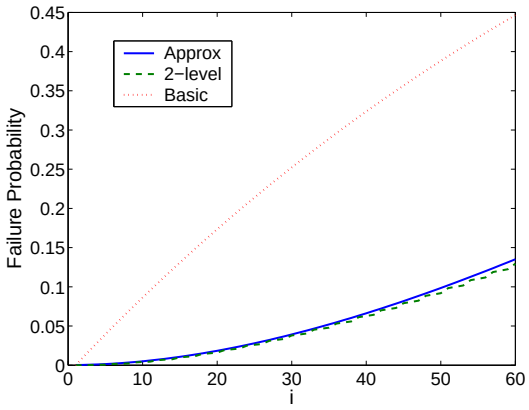



Fig. 20. Compare the difference between the calculation and approximation.

The query-report success probability can be further improved by extending the needle length. For instance, we can extend the vertical data duplication in both directions (approximately) equally and let the needle length be $2S$. Thus, each report can reach two nodes that are on the query propagation path. If one of them receives the query successful, the query-report will be successful. We call this scheme a 2-level redundancy scheme. In the basic comb scheme, we have one vertical query propagation path, one horizontal query propagation path, and one vertical data duplication path from the query node to a data note. To improve the reliability, redundancy is introduced to all three of them. To be more specific, we allow one of the vertical lines, one of the horizontal lines, and one of the data duplication lines to fail. The failure probability can be approximated by

$$\begin{aligned} P_f(i,j|(x,y),k) &\approx a_1+a_2+a_3-a_1a_2-a_2a_3-a_3a_1 \\ &\quad +a_1a_2a_3 \\ &\approx a_1+a_2+a_3, \end{aligned}$$

where $a_1$, $a_2$, and $a_3$ approximate the error probabilities of failures of both vertical lines, both horizontal lines, and both data duplication lines, respectively. We have

$$\begin{aligned} a_1 &= (1-(1-p)^{|y-j|})(1-(1-p)^{|y-j|+k}) \\ a_2 &= (1-(1-p)^{|x-i|})^2 \\ a_3 &= (1-(1-p)^q)^2. \end{aligned}$$

Furthermore, we can provide high-level redundancy protection by providing more protections for vertical query, horizontal query, and vertical data duplications.

In Figure 20, we show the error probabilities for query generated by node $(0,0)$ with $k=1$ for data nodes $(i,i)$, $1 \le i \le n-1$. We set $p=0.005$, $n=60$, and $s=3$. Node $(0,0)$ is chosen as the query node because it has the highest failure probability on average. It is not surprising that nodes closer to the query node has lower failure probability and nodes far away has high failure probability. It shows that the above approximation works well. In the figure, we also compare the redundancy scheme with the basic scheme. The spacial redundancy scheme significantly decreases the error probability especially for smaller values of $p$. Local enhancement (such as interleaving mesh) can be used to improve $p$.

In summary, in this section, we study the effect of node failures on the query-report process. Reliable transmissions studied in the literature can be used to enhance reliability locally and spacial redundancy can be implemented to further improve the query coverage of wireless sensor networks for correlated network failures.

## V. Adaptive Comb-needle Strategy

The basic comb-needle strategy assumes the knowledge of the query and event frequencies and analyzes the performance in a snap shot. In practice, the query and event frequencies may be time-varying, and thus a good query strategy should adapt to such changes. However, the values of the query and event frequencies may not be given. More importantly, other sensor nodes may not know the value of the inter-spike interval ($s$) used by a query node. This motivates the development of the adaptive comb scheme, as discussed in this section. Simulation results show that our adaptive scheme can track the changes well and keep a high success rate while maintaining a low communication cost.

We consider a time-slotted system where a query node is only interested in the event happened in the most recent time slot, i.e., $\tau=1$. Extension to the case where $\tau>1$ is not difficult. Let $f_q$ be the probability that a query is generated in a time slot. Let

$$f_d = \frac{f_e}{n^2}$$

be the probability that a sensor node detects an event in a time slot. In this case, we can rewrite Eq. (8) by

$$s_{optimal} \sim \sqrt{\frac{f_q}{f_d}}.$$

Thus, to calculate inter-spike interval and the needle length, a node only needs to estimate $f_d$ and $f_q$.

We assume that the distribution of $f_d$ and $f_q$ are known to both query nodes and data nodes. If such information is not available, a default distribution can be assumed. For the query node, it has an estimate of $f_d$ and $f_q$. (Initially, the best estimate is the mean values of $f_d$ and $f_q$.) The larger the number of queries the node performs, the better the estimate of $f_d$ and $f_q$ are. Based on the estimate, the query node calculate $s$.

Data nodes take a different approach. Let $L$ the be total needle length at a data node. In particular, if the data node does not duplicate its data to any other nodes, then $L = 1$. In addition, we allow asymmetric data duplications to fine tuned the needle length. For example, if the data is duplicated to one node south and to two nodes north, then $L = 4$. Analysis in Section II can be extended to such a case. Initially, each data node estimates its own data frequency $f_d$. Based on the estimation of $f_d$ and the knowledge of the distribution of $f_q$, each data node calculates $L$ such that

$$P(L > s) \geq P_m,$$

where $P_m$ a criterion for the importance of the data.

Even if the needle length of a node is smaller than the inter-spike interval, its data duplication may still reach one query line if its distance to the line is smaller than the needle length in that direction. To elaborate, the success probability is $L/s$ given $L < s$, and the over query success probability is

$$\begin{aligned} P_s &= P(L \geq s) + E\left(\frac{L}{s} | L < s\right) P(L < s) \\ &= E\left(\min\left(\frac{L}{s}, 1\right)\right). \end{aligned}$$

In general, the estimates of $f_d$ and $f_q$ at a data node may be poor and so does the value of $L$ calculated based on the estimates, specially when $f_d$ is small and the node misses some queries. Thus, the $s$ value obtained from previous query is important and should be used to estimate $L$. A node learns about the value of $s$ in two cases. First, a node may obtain information when its data is successfully queried. Second, we enhance the possibility that a node learns about the query by implementing a rotation of the query horizontal duplications. Suppose that the current horizontal queries occur on lines $0, s, 2s, \cdots, \lfloor (n-1)/s \rfloor * s$, then the next query occur on lines $1, 1+s, 1+2s, \cdots, 1+\lfloor (n-1)/s \rfloor * s$, and so on. A node may not generate an event that is successfully queried. However, when it is on a spike in the comb, it obtains the query information $s$. A node uses its most recent information on $s$ as an important factor to calculate its needle length.

Note that the knowledge of the distribution of $f_d$ and $f_q$ is not critical. A more important issue is that the needle length of data nodes should synchronize with the inter-spike interval. Since the query node can learn the value of $f_d$ and $f_q$ faster and better, it is important for data nodes to obtain the value of $s$ from the query node, which is enhanced by the query rotation strategy. Given the value of the most recently obtained $s$, the needle length at each data node is adjusted according to importance of the data and the desired query success rate. The rotation scheme does not trigger additional communication cost, but does require memory spaces. If there are a very large number of query nodes, a node may not be able to maintain the required information for all query noes.

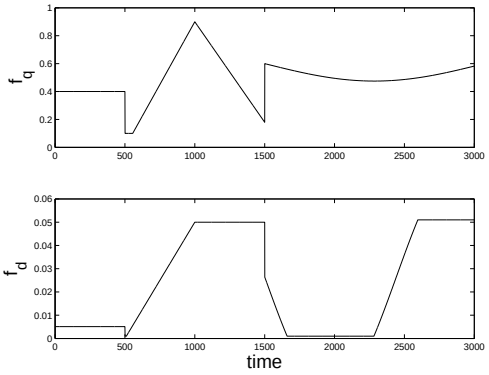



Fig. 21. The query frequency and the event frequency.

We simulate the adaptive scheme for 3000 time slots in a $20 \times 20$ regular grid. Figure 21 shows the query frequency and the data frequency. We assume node $(0, 0)$ is the query node, which generates query with probability $f_q$ in each time slot. Each other node generates events independently with probability $f_d$ in a time slot. In the adaptive scheme, both the query node and data nodes estimate the values of $f_q$ and $f_d$ using a moving average. The inter-spike interval and the needle length are calculated based on the estimations. In addition, the query rotation scheme is implemented in the simulation.

The adaptive scheme is compared with the ideal case, where the query node knows $f_q$ and $f_d$ precisely, and all data nodes know the value of current $s$. Such the ideal case consumes the least amount of energy to guarantee $100\%$ query successes.

In this simulation, the query-report success ratio of the adaptive scheme is $99.33\%$ and the total communication cost is $99.64\%$ of the ideal case. The query success ratio is the ratio of the successful reports in the adaptive scheme over that of the ideal one. The communication cost is defined as the total cost to report events to the query node, which includes query cost, data duplication cost, and data report cost. We count one hop of query/event as one unit of communication cost. The estimated case consumes less energy at the cost of lower success ratio. In particular, there are cases where needles are not long enough to reach the query line. Such events are not reported back to the query node. There exists a tradeoff between the success probability and the communication cost.

Figure 22 compares the ideal comb size and the adaptive scheme where $f_d$ and $f_q$ are estimated. We can see that the adaptive scheme can track the changes in the ideal case. The fluctuation is due to the estimation errors on $f_q$ and $f_d$, especially on $f_q$, at the query node.

Table I compares the comb scheme with that of push and pull in different time periods where the values of $f_d$ and $f_q$ differ. Each entry in the table represents a communication cost normalized over that of the ideal comb. We can observe

| Time | 1-500 | 501-1000 | 1001-1500 | 1501-2000 | 2001-2500 | 2501-3000 |
|---|---|---|---|---|---|---|
| Push | 179% | 131% | 118% | 169% | 160% | 114% |
| Pull | 156% | 138% | 157% | 131% | 139% | 153% |

TABLE I

COMPARE THE COMB STRATEGY (IDEAL CASE) WITH PURE-PULL AND PURE-PUSH STRATEGIES.

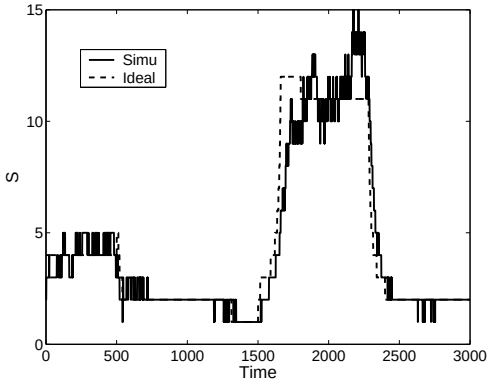



Fig. 22. The query frequency and the event frequency.

that the saving of the comb strategy varies as $f_d$ and $f_q$ vary. The communication costs of pure-push and pure-pull strategies are $132\%$ and $149\%$ of that of the ideal case for the total 3000 time-slot period on average. In all time periods, the adaptive scheme achieves over $99\%$ success rate with negligible additional communication cost compared to the ideal case.

In the simulation, we fix the query node to be $(0, 0)$. However, the exact strategy applies to the case where the query node is moving in the network, such as a soldier is moving within the deployed sensor network. The adaptive comb strategy also applies to the case where different nodes generate query requests. In such case, similar queries should be combined. In particular, a possible query node should estimate the value of $f_q$, which is the query frequency for all nodes, by observing the queries of its own and others. A query node can also use the current value of $s$ in the network to estimate the inter-spike interval of its own query.

## VI. RELATED WORK AND DISCUSSION

Sensor network applications often need support for periodic or sporadic monitoring of the environmental state. A simplest method for obtaining state from a large scale sensor network is to flood the query for the named data to the whole network and get responses from relevant sources. However, flooding in large-scale wireless networks is a very expensive operation and in general needs $O(N)$ radio transmissions, where $N$ is the number of nodes in the network.

Many strategies have recently been proposed to reduce the cost of discovery and query in large-scale wireless networks. Most relevant to this work includes three different types of approaches.

- The first approach is to improve flooding efficiency by reducing the number of potentially redundant forwarding in the flooding process using neighborhood topological information[16], [17] or via probabilistic retransmissions[18]. The goal of this approach can be viewed as to reduce the constant implicit in $O(N)$. In contrast, the comb-needle scheme achieves $O(\sqrt{N})$ in the best case by balancing push and pull.
- The second approach in the literature is to reduce discovery/query cost by taking into account application semantics[19]. For instance, discovery queries like "is there a tank in the field?" is potentially resolvable by only traversing part of the network in many cases. As a result, for this kind of queries, expanding-ring search or ACQUIRE[9] have been demonstrated to be more efficient than flooding. ACQUIRE considers the query as an active entity that is forwarded through the network either randomly or in some directed manner in search of a solution: nodes on the path that handle the active query use information from all nodes within d-hops in order to partially resolve the query. When the active query is fully resolved, the query forwarding process stops and a complete response is sent back to the querying node. This type of scheme limits the scope of necessary flooding by recognizing the specific nature of the query. Another outstanding example in this category is the Information-Driven Query Routing (IDSQ) scheme[20]. For instance, when tracking mobile entities, sensor nodes can keep a certain memory, so when a query like "where are the tanks?" comes in, the underlying system can quickly route the query to the nodes with most current information by following the traces of information exist in the network. This again eliminates the needs for flooding in many cases. In a recent work, [19] investigated the relative benefit of push and pull diffusion for queries of continuous type and arrived the conclusion of a need for a families of protocols catering to different application requirements. This paper investigates and demonstrates the benefit of taking into account both the application and the environmental characteristics in designing efficient data discovery and dissemination protocols.
- The third types of approach is to reduce the cost of search by using efficient distributed indexing schemes. For instance, [7] described a distributed indexing scheme called Semantic Routing Tree (SRT). SRT only supports query from a fixed node and constructs a route tree based on historical data, so only applies well to slow changing sensor readings. [21] proposed a distributed index for

multi-dimensional data (DIM) which allows query to be issued from any node. DIM achieves an average event insertion cost of $O(\sqrt{N})$ and average query cost of $O(\sqrt{N})$ in most cases. [21] also mentioned DIM outperforms flooding as long as the ratio of the number of insertions to the number of queries is less than $\sqrt{N}$. Earlier exploration in this direction includes GHT[22], DIMENSIONS[23] etc. The push component in our comb-needle scheme is similar in spirit with DIM in the sense that the data generated at a node are hashed/pushed to different locations. However, our scheme is much simpler and achieves similar cost savings. Furthermore, we investigate how the query support structure perform when links are unreliable, an important issue which is largely neglected in earlier works.

While in the paper we only discussed supporting discovery and query in a whole network, the use of comb-needle model is not limited to that. The idea is also applicable for supporting geographic queries like "how many tanks are there in region x?". For supporting such geographic queries, one traditional strategy is to geocast the query to the region, i.e., unicast the query to the region, and then broadcast the query in the region. Similarly, we can choose to unicast the query to the region, then "comb" the region with desired granularity. Note also that the push component not only helps to reduce the query cost in the comb-needle scheme, but also comes with the benefit of data redundancy which is desirable in an unreliable network.

Furthermore, as we shown in section II, given the values of $f_e$ and $f_q$, the optimal values of the comb width $s$ is calculated in Eq. 8. Note that $1 \leq s \leq n$. If $f_q \leq f_e/(n-1)^2$, then $s = 1$, which is equivalent to the pure pull strategy that flooding the query message in the network. On the other hand, if $f_q = f_e$, then the optimal strategy is the one shown in Figure 5. If $f_q > f_e$, then a reversed combing strategy is desired as shown in Figure 23. An extreme of the reverse combing is the pure pull-based strategy where each data node broadcasts its event to all other nodes. This highlights the principle of the comb-needle structure: adjust the communication strategy based on the frequency of the query and events. In particular, the lower the (relative) frequency of the query/event, the larger the number of nodes it propagates to. Combined with the reverse comb, the comb strategy covers the whole spectrum of the push and pull strategies.

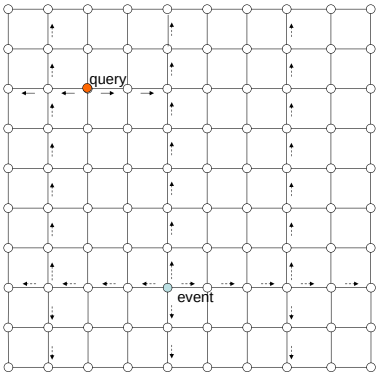



Fig. 23. Reverse comb

## VII. Conclusion

In this paper we proposed the comb-needle model, a simple yet efficient data discovery scheme for supporting queries in large-scale sensor networks. We also used the comb-needle model as a substrate for study the benefit of balancing push and pull in data gathering and dissemination in large-scale wireless networks. The comb-needle scheme (including the reverse one) covers a spectrum of the push and pull strategies, with the pure push-based and pure pull-based schemes in two extremes. We have demonstrated that in general, the comb-needle scheme performs better than both pure push-based and pure pull-based schemes. Furthermore, we raises and studied the issue of query reliability and query coverage when the underlying communication channel is not reliable. Finally, we proposed and studied an adaptive comb-needle strategy for cases where the utilization patterns and environmental activity frequency are time-varying. The contribution is highlighted not only by the theoretical analysis of the problems studied, but also by the confirmation of the theoretical observations via simulations. We note that the proposed scheme can be used in the hierarchical structure as well. Further, data aggregation and compression can be integrated into the comb-needle strategy to further reduce the communication cost. These are all interesting future work to explore. We also look forward to further validating this work in actual large-scale sensor network test-beds in the future and exploring new issues that might arise in the real world.

## References


[1] D. Estrin, R. Govindan, J. S. Heidemann, and S. Kumar, "Next century challenges: Scalable coordination in sensor networks," in *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, 1999, pp. 263–270. [Online]. Available: citeseer.ist.psu.edu/estrin99next.html

[2] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors," *Commun. ACM*, vol. 43, no. 5, pp. 51–58, 2000.

[3] 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.*, Atlanta, GA, September 2002.

[4] J. Liu, D. Petrovic, and F. Zhao, "Multi-step information-directed sensor querying in distributed sensor networks," in *Proceedings of the International Conference in Acoustics, Speech and Signal Processing (ICASSP)*, 2003.

[5] D. Li, K. Wong, Y. Hu, and A. Sayeed, "Detection, classifi cation and tracking of targets in distributed sensor networks," *IEEE Signal Processing Magazine*, vol. 19, no. 2, March 2002.

[6] B. Blum, P. Nagaraddi, A. Wood, T. Abdelzaher, S. Son, and J. Stankovic, "An entity maintenance and connection service for sensor networks," The First International Conference on Mobile Systems, Applications, and Services (MobiSys), San Francisco, CA, May 2003.

[7] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "The design of an acquisitional query processor for sensor networks," in *Proceedings of the 2003 ACM SIGMOD international conference on on Management of data*. ACM Press, 2003, pp. 491–502.

[8] Q. Huang, C. Lu, and G.-C. Roman, "Spatiotemporal multicast in sensor networks," in *Proceedings of the First ACM Conference on Embedded Networks Sensor Systems (SenSys)*, Los Angelos, California, Nov. 2003.

[9] N. Sadagopan, B. Krishnamachari, and A. Helmy, "Active query forwarding in sensor networks," *Elsevier Journal of Ad Hoc Networks*, 2003.
[10] G. Simon, "Probabilistic wireless network simulator," 2003, http://www.isis.vanderbilt.edu/projects/nest/prowler/.
[11] S. Shakkottai, R. Srikant, and N. Shroff, "Unreliable sensor grids: Coverage, connectivity and diameter," in *Proceedings of IEEE INFOCOM 2003*, 2003.
[12] J. Zhao and R. Govindan, "Understanding packet delivery performance in dense wireless sensor networks," in *ACM Sensys 2003*, 2003.
[13] A. Woo, T. Tong, and D. Culler, "Taming the underlying challenges of reliable multihop routing in sensor networks," in *ACM Sensys 2003*, 2003.
[14] S. M. Ross, *Introduction to probability models*, 7th ed. Harcourt Academic Press, 2000.
[15] F. Ye, G. Zhong, S. Lu, and L. Zhang, "Gradient broadcast: A robust data delivery protocol for large scale sensor networks," *ACM Wireless Networks (WINET)*, vol. 11, no. 2, March 2005.
[16] W. Peng and X. Lu, "On the reduction of broadcast redundancy in mobile ad hoc networks," in *Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, 2000.
[17] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint relaying: An efficient technique for flooding in mobile wireless networks," INRIA, Tech. Rep. Research Report RR-3898, Feb. 2000. [Online]. Available: citeseer.nj.nec.com/qayyum00multipoint.html
[18] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *Proceedings of the Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, August 1999, pp. 152–162.
[19] J. Heidemann, F. Silva, and D. Estrin, "Matching data dissemination algorithms to application requirements," in *Proceedings of the first international conference on Embedded networked sensor systems*. ACM Press, 2003, pp. 218–229.
[20] M. Chu, H. Haussecker, and F. Zhao, "Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks," *International Journal of High Performance Computing Applications*, vol. 16, no. 3, 2002.
[21] X. Li, Y. J. Kim, R. Govindan, and W. Hong, "Multi-dimensional range queries in sensor networks," in *Proceedings of the first international conference on Embedded networked sensor systems*. ACM Press, 2003, pp. 63–75.
[22] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu, "Data-centric storage in sensornets with ght, a geographic hash table," *Mob. Netw. Appl.*, vol. 8, no. 4, pp. 427–442, 2003.
[23] D. Ganesan, D. Estrin, and J. Heidemann, "Dimensions: Why do we need a new data handling architecture for sensor networks?" in *Proceedings of the First Workshop on Hot Topics In Networks (HotNets-I)*, Princeton, NJ, October 2002.