Glance: A Lightweight Querying Service for Wireless Sensor Networks

Murat Demirbas [*],

Computer Science and Engineering Department, University at Buffalo, SUNY, Buffalo, NY, 14260

Anish Arora, Vinodkrishnan Kulathumani

Computer Science and Engineering Department, The Ohio State University, Columbus, OH, 43210

Abstract

Distance-sensitivity guarantee in querying is a highly desirable property in wireless sensor networks as it limits the cost of executing a "query" operation to be within a constant factor of the distance to the nearest node that contains an answer. However, such a tight guarantee may require building an infrastructure for efficient resolution of queries, and the cost of maintaining this infrastructure may be prohibitive. Here we show that it is possible to implement distance-sensitive querying in an efficient way by exploiting the geometry of the network. Our querying service *Glance* ensures that a "query" operation invoked within d distance of an event intercepts the event's "advertise" operation within $d * s$ distance, where s is a "stretch-factor" tunable by the user.

Key words: wireless sensor networks, in-network querying, distance-sensitivity

1 Introduction

A major application area for wireless sensor networks (WSNs) is environmental monitoring [1,2,20,23,24]. The grand vision for these applications is to scatter thousands of wireless sensor nodes across an area of interest upon which the

[*] Corresponding author.
Email addresses: demirbas@cse.buffalo.edu (Murat Demirbas), anish@cse.ohio-state.edu (Anish Arora), vinodkri@cse.ohio-state.edu (Vinodkrishnan Kulathumani).

nodes self-organize into a network and enable monitoring and querying of events in the area. An example application is a disaster evacuation scenario where the rescue workers query the network to learn about fire or chemical threats in the area.

There are two main modes of operation in most WSN monitoring applications. The first mode is "centralized monitoring and logging". For monitoring and logging purposes it is important to gather information about events in the network [19, 26]. This can be easily satisfied by enforcing events to exfiltrate data to a basestation that could forward the data to a monitoring and control center. In our disaster evacuation scenario, the control and command center needs to get data about events for logistical purposes, such as deciding how many rescue workers to send to each region and coordinating the rescue efforts. These data are also valuable for keeping logs and statistics of events.

The second mode of operation is "in-network querying" or "location-dependent querying". In the context of the evacuation scenario, the rescue workers in each region would need to query the network for nearby events, such as fire/chemical threats, and vital statistics from victims. It is inefficient and unscalable, for most cases, to force the queriers to learn about events only from the basestation, since it would compel a querier that is very close to an event to communicate all the way back to a basestation to learn about that event. The inefficiency of the scenario is amplified if the querier needs to get a stream of data from the event. Using long routes for forwarding data not only increases the latency but also depletes the battery power of the relaying nodes in the network quickly. Using the basestation for every query also leads to a communication bottleneck for the network. Moreover, answering queries locally may also be important for preserving the correctness of applications deployed in large WSNs. As a specific example, for intruder-interceptor applications [5] it has been shown [4] that there exist Nash equilibrium conditions which imply that, for satisfying optimality constraints, the latency with which an interceptor requires information about the intruder it is tracking depends on the relative locations of the two: the closer the distance, the smaller the latency. Thus, it is important to discover short paths from queriers to nearby events.

The desirability of quick resolution of in-network queries via shortest possible paths is formalized by the distance-sensitivity property. Distance-sensitivity limits the cost of executing a query operation to be within a constant factor (we call this as the *stretch-factor*) of the distance to the nearest node that contains an answer. However, such a tight guarantee may require building an "in-network advertisement infrastructure" (such as a hierarchical partitioning of the network [9] or a network-wide advertisement tree [13, 18]) for efficient resolution of queries, and the cost of maintaining this infrastructure may be prohibitive. In fact many work on in-network querying [3, 21, 22] choose to avoid such a guarantee in favor of best-effort resolution of the queries.

Contributions of the paper. Here we show that it is possible to implement distance-sensitive querying in an efficient way–using minimal infrastructure–by exploiting the geometry of the WSN. Our main insight is to combine both modes of operation in WSN monitoring applications in a synergistic manner. As part of the data exfiltration mode, any interesting event detection is sent toward the basestation node, and so the basestation can act as a last resort for resolving an in-network query. As part of in-network querying mode, queries are also sent toward the direction of the basestation with the intention that the in-network advertisements of nearby events (if any) will intercept the query and answer it in a distance-sensitive manner, or else the query is answered at the basestation by default. It is at this point that using the geometry of the network comes handy. By using geometry, we determine the minimum area required for in-network advertisement for satisfying the distance-sensitivity requirement. More specifically, we observe that the local advertisements of events can safely ignore a majority of directions/regions while advertising and still satisfy a given distance-sensitivity requirement tightly.

As a result, we present a simple (using minimal infrastructure) and lightweight (cost efficient) distance-sensitive querying service, *Glance*. The distance-sensitivity of Glance, is easily tunable. Glance ensures that a query operation invoked within d distance of an event intercepts the event's advertisement information within $d * s$ distance, where s is a "stretch-factor" tunable by the user. By selecting appropriate values for s, the user can trade-off between query execution cost and advertisement cost, and adjust the pull-push balance to accommodate for different query-event rates in the WSN. Last, but not least, Glance achieves resilience of advertising and querying in the presence of node failures by employing greedy perimeter stateless routing (GPSR) [14] to route around the holes in the WSN.

Overview of Glance. Let C be a distinguished basestation node in the network. Let d_q be the Euclidean distance between a querying node q and C, d_e the distance between an event e and C, and finally d the distance between q and e. We observe that for the cost of query operations there are two possible cases with respect to the angle z formed by locations of q, C, and e. Figure 1 illustrates these two cases, with respect to querying nodes q' and q''.

(1) z is larger than a threshold: A large z implies that d is large relative to d_q and d_e. Thus, it is acceptable for the query to go to C to learn about the event, since the stretch-factor s can still be satisfied this way. For example, in Figure 1, z' is larger than the threshold angle and hence q' can still satisfy s by learning about e at C since $d_{q'} \leq d' * s$.
(2) z is smaller than the threshold: A small z implies that d is small relative to d_q and d_e. Thus, it is unacceptable for the query to go to C, as this violates the stretch-factor. In Figure 1, z'' is smaller than the threshold angle and hence q'' cannot satisfy s by going to C since $d_{q''} > d'' * s$.

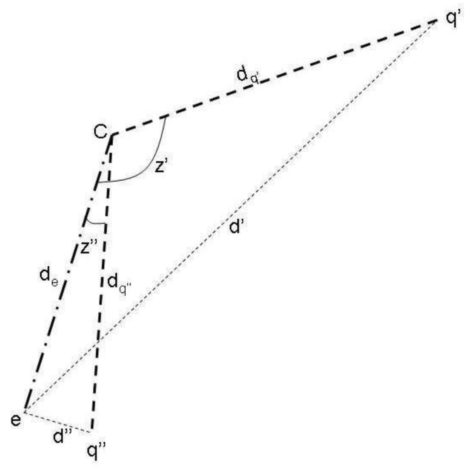


Fig. 1. Two cases with respect to z

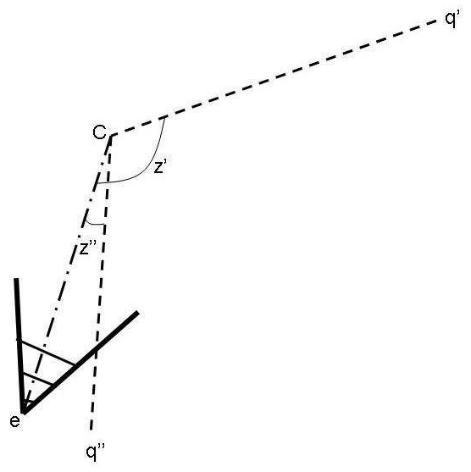


Fig. 2. Advertise operation for s=2

Our advertise operation in Glance (see Figure 2) seeks to optimize for the above two cases by combining both modes of operation in WSN monitoring applications:

- Data exfiltration to C proves useful in answering some in-network queries at C since that would still satisfy the stretch-factor for potential queriers with a large angle z' as in case 1 above.
- The advertise operation advertises the event in the network only along a cone boundary for some distance. The angle x for the advertisement cone is calculated based on the the stretch-factor s as $arcsin(1/s)$ (as described in Section 3.2). This cone-advertisement accounts for potential queriers q with a small angle z'', whose $d_q > d'' * s$.

The query operation is simply a *glance* to the direction of the basestation; it progresses as a straight path from the querying node toward C. Once an in-network advertisement for a matching event is found, the query operation stops forwarding the query any further and informs the querying node about the match by sending a reply. (In the worst case, the query reaches C and C sends a reply back to the querying node.) The querying node can then use the event location information included in the reply to learn more about the event.

By involving the basestation partially in answering of in-network queries and by exploiting geometry of the network, we observe that the advertise operation can constrain itself to a small region as in Figure 2 and still satisfy a given distance-sensitivity requirement tightly.

Outline of the paper. Next, we discuss related work. In Section 3 we present Glance. In Section 4, we analyze the performance of Glance and present simulation results. We conclude the paper with a discussion of future work in Section 5.

2 Related Work

Recently there has been much research on in-network querying. Early work includes adaptation of publish-subscribe tree structures from the Internet domain to the wireless ad hoc networks [12]. Although the basic ideas in publish-subscribe services may still be applicable for in-network querying problem in WSN, certain assumptions in the publish-subscribe model does not apply in WSN. For example, in contrast to the subscriptions that are long-lived, short-lived ad hoc queries is an important class of querying in WSN. These ad hoc queries may appear sporadically at any node in the network, as in our fire evacuation scenario. The event sources may be equally unpredictable in WSN, so it is unclear as to which nodes the subscription trees should be rooted at. Also typical network sizes considered in WSN are much larger than that of ad hoc network deployments and battery constraints are more severe in WSN, and hence scalability and inefficiency are a more critical concern for WSN querying services.

Rumor routing [3] provides a novel and tunable in-network querying mechanism without any need for localization information. In this approach, an event employs a set of advertising agents to do a random walk of the network creating paths that lead to the event. Querying node also sends out query agents which randomly traverse the network. Whenever a query agent encounters a path set up by an advertising agent with a matching interest, a route is established between the query and the event. The scheme is tunable in that for guaranteeing higher reliability it is possible to increase the number of agents sent from each event and query, however, rumor routing does not provide any distance-sensitivity guarantees or any deterministic guarantees for querying. Glance improves over rumor routing by providing a more structured approach to advertising and querying. Since both the advertise and query operations now target a common node, C, their meeting distance is shortened greatly compared to a random walk strategy. In addition, using the stretch-factor idea and the cone-advertisement, the meeting distance of the advertise and query are optimized. Glance also avoids wasting energy by not advertising the event in the regions where meeting at C is already an acceptable solution for the query and event.

Combs and needles algorithm [18] maintains an advertisement infrastructure over the network for efficient resolution of in-network queries. More specifically, the event advertisement builds a network-wide routing structure that resembles a comb, and the query operation searches for an event using a structure resembling a needle. The paper shows that due to the shapes of these structures the event and query are guaranteed to intersect. By arranging the distance between the teeth of the comb structure, Combs&Needles tunes the minimum length for the needle structure to guarantee that query operation intersects the advertise operation. Combs&Needles protocol forces the user to fix the cost of querying to be a constant cost in advance, and compels the advertise operation to do as much work as necessary to guarantee the fixed cost for querying. In contrast, in Glance, the cost of querying is designed to be within a constant factor of the distance to the nearest event, not within a fixed constant cost per se. By allowing the cost of querying to increase linearly when there is no event nearby (of course within the constraints of distance-sensitivity), Glance reduces the cost for advertise operation significantly. Also by using a common node C to focus the dissemination of information and forwarding of the queries, Glance is able to construct a very lightweight structure for advertising.

A simple and lightweight solution to in-network querying problem is to use Geographic Hash Tables (GHT) [21], which store and retrieve information by using a geographic hash function on the type of the information. However, the basic GHT protocol is not distance-sensitive since it can hash the event information far away from the nearby event-query pair and thus violates the stretch-factor. In contrast to GHT protocol, Glance provides distance-sensitivity guarantees and also tunability of stretch-factors. The distance-sensitivity problem of GHT can be alleviated by using hierarchies: either by a structured replication at different levels of a hierarchical partitioning [21], or by using geographically bounded hash functions at increasingly higher levels of a hierarchical partitioning as employed in DIFS protocol [11]. Hierarchical clustering of the network (via a quadtree) is also employed by another in-network querying protocol, Geographic Location System (GLS) [17]. Hierarchical GHT and GLS protocols still cannot achieve distance-sensitivity for all query-event pairs due to the multi-level partitioning problem: In a hierarchical partitioning it is possible that a query-event pair nearby in the network might be arbitrarily far away in the hierarchy due to multi-level partitioning between them.

Stalk [6], a WSN tracking protocol for mobile objects, also uses a hierarchical partitioning, but to account for the multi-level partitioning effects a querying node performs lateral searches to neighboring clusterheads (in addition to its own clusterhead) at increasingly higher levels of the hierarchy to reach the event information in a distance-sensitive manner. Recently, Distance Sensitive Information Brokerage (DSIB) protocol [9] achieved distance-sensitivity in a hierarchically partitioned network by using a similar technique for querying of static events. Instead of adapting a pull-based approach and using lateral searches to neighbors as in Stalk, DSIB adapts a push-based approach: an event advertises to neighboring clusterheads as well as its clusterhead at every level of the hierarchy. Accordingly, the responsibility of the query is decreased: querying node contacts immediate clusterheads at increasingly higher levels until it hits the event information.

3 Glance Algorithm

3.1 Model

We assume a dense, connected, and multihop WSN and the availability of localization information at the nodes. We use $dist(j, k)$ to denote the Euclidean distance between two nodes j and k. We assume an underlying geographic routing protocol, such as greedy perimeter stateless routing (GPSR) [14] or crossing link detection protocol (CLDP) [15], that achieves O(d) cost for communication over d distance.

We assume a distinguished basestation node C in the network. We denote the distance between a querying node q and C as d_q, and event e and C as d_e. We use d to refer to the distance between q and e. $z_{q,e}$ denotes the angle formed by location of q, C, and the location of e. We use a calculational proof notation [8] where a proof of $K \equiv M$ can be expressed as:

$$\begin{array}{ll} & K \\ \equiv & \{ \text{ reason why } K \equiv L \} \\ & L \\ \equiv & \{ \text{ reason why } L \equiv M \} \\ & M \end{array}$$

3.2 Details of the Glance algorithm

Here we explain the cone-advertisement needed for the advertise operation in detail, and discuss how the advertise operation ensures distance-sensitivity for a given stretch-factor, s.

Areas where stretch-factor is readily satisfied. We mentioned in the Introduction that there are two possible cases for the cost of a query operation invoked at a node q, for an event e, with respect to the angle $z_{q,e}$. To account for the case where $z_{q,e}$ is less than the threshold angle x, the advertise operation needs to advertise on a cone boundary. For $z_{q,e}$ greater than x no advertising is required as the stretch-factor is readily satisfied even when q contacts C directly, incurring a d_q cost. In order to be able to determine the boundaries of the advertisement cone precisely, we first need to calculate the threshold angle x for a given stretch-factor s. Here we show how we calculate x by determining the areas for which stretch-factor is readily satisfied.

As a simple example, let's take $s = 1$. We calculate the region where the stretch-factor is readily satisfied by taking successively larger circles centered

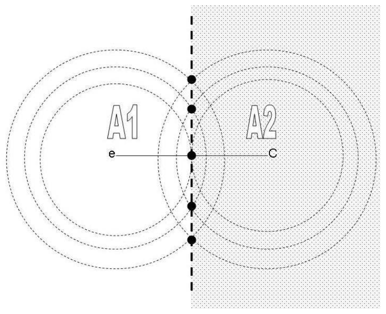


Fig. 3. Area where s=1 is readily satisfied

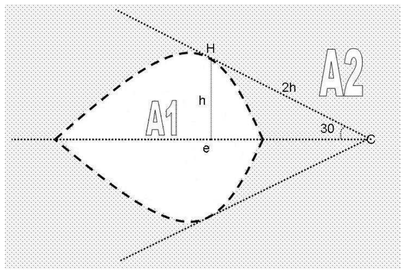


Fig. 4. Areas where s=2 is readily satisfied

at e and C and intersecting them. Figure 3 illustrates this method. There the dashed line consists of points obtained from intersecting circles with equal radii, r, $r \geq d_e/2$, centered at e and C. Thus, any point on the dashed line is equidistant to e and C. It follows that, any point in $A2$ is closer to C than it is to e, and hence, for any querying node in $A2$ stretch-factor $s = 1$ is readily satisfied by contacting C directly.

For $s > 1$, the same method is used for calculating the areas where stretch-factor is readily satisfied: we let a circle with radius r centered at e intersect with a circle with radius $s * r$ centered at C. Figure 4 shows an example for $s = 2$. Note that a circle centered at e with radius r intersects with the circle centered at C with radius $2r$ for $d_e/3 \leq r \leq d_e$. This is because for $r < d_e/3$ there is a gap of $d_e - 3r$ between the two circles, and for $r > d_e$ all the circles centered at e are subsumed by circles with radius $2 * r$ centered at C. Thus, the dashed line closes on itself and forms a bounded area $A1$. From our construction it follows that for $s = 2$ any point on the dashed line is twice as far away from C than it is from e. Also, for any point in $A2$ the distance to C is always less than twice as that to e. Hence, the stretch-factor $s = 2$ is readily satisfied for area $A2$. On the other hand, for area $A1$ stretch-factor may be violated, and cone-advertisement should account for the querying nodes in this region. From Figure 4 we observe that event e can safely ignore a majority of directions/regions advertising and still satisfy the given distance-sensitivity requirement tightly.

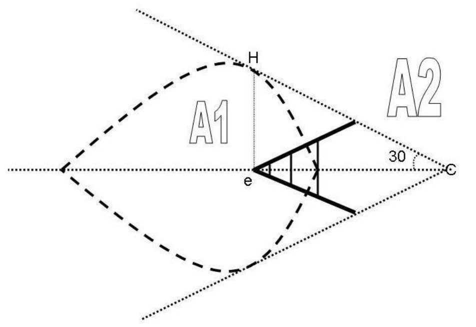
In Figure 4 consider a point H on the dashed line such that $\widehat{HeC}$ forms a right angle. Since any point on the dashed line is twice as far from C than it is from e, $|CH| = 2 * |eH|$, and hence $\widehat{eCH}$ is calculated as 30° from the HeC right triangle. Since $\widehat{HeC}$ is 90°, H determines the maximum angle between any intersection point and e with respect to C, so the threshold angle $x$for $s = 2$ is set as 30°. In general x is calculated as $arcsin(1/s)$, since $x = arcsin(|eH|/|CH|)$. For $s > 1$ (which we consider in this paper), we always have $90° > x > 0$. For $s = 1$ there is no feasible solution since $x = 90°$. For $s = 2$, $x = 30°$, and $s = 4$, $x = 14.5°$. So, as s increases the threshold angle decreases rapidly. The area $A1$ that the advertise operation has to account for is extremely small for $s = 4$.

The algorithm for query. Before we give the details of the advertise operation and prove its correctness, we present the query operation briefly. The query operation progresses as a straight path from the querying node toward C. For routing of the query toward C, GPSR is employed with the destination of the query packet set as C. During relaying of this query message hop-by-hop, if a node j with the advertisement information of a matching event is reached, j stops forwarding the query any further. (At worst the query will be answered at C, hence $j = C$ in that case.) To inform the querying node, whose address is included in the query message, about this matching event, j sends a reply to the querying node using the GPSR service. This reply contains advertised metadata about the event, such as its location, type, and time. By using the location information in the reply, the querying node can then contact the node that detected the event directly to learn more about the event.

For the cost of a query operation, we only include the communication cost of forwarding the query until it reaches a node j that has an answer to the query. We do not include the cost of j's reply to the query cost for the Glance protocol as well as for the other in-network querying protocols we consider in the analysis section (Section 4).

The algorithm for advertise. The advertise operation for $s = 2$ is depicted in Figure 5 with solid dark lines. Roughly speaking the advertisement is performed on the boundaries of a cone that is rooted at the event location and that widens toward C. The progress of the cone stops when the threshold angle is reached with respect to a straight line between the event and C, as there is no need to advertise outside the threshold angle. The idea behind this cone advertisement is to intercept any query that may be originating at $A1$ in a distance-sensitive manner. Note that without this cone advertisement the queries originating in $A1$ would go all the way to C violating the distance-sensitivity requirement. The angle for the cone advertisement for both the left-hand side cone boundary and the right-hand side cone boundary is selected to be equal to the threshold angle $x = arcsin(1/s)$. The reason behind this selection is to accommodate for varying threshold angles (those close to $90°$ as well as those close to $0°$) using a uniform strategy for the advertisement. The user may want to define different stretch-factor requirements (which lead to varying threshold angles) with respect to the type (i.e., severity) of events. As we prove in Lemma1, selecting the angle for cone advertisement to be equal to the threshold angle satisfies the distance-sensitivity requirement for any stretch factor greater than 1.

Beside the advertisement on the cone boundary, there is a need for some lateral advertisements within the boundaries of the cone. These lateral advertisements are needed for intercepting any query originating within the cone boundaries in



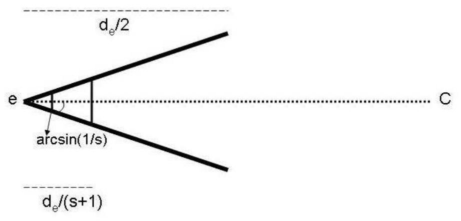


Fig. 5. Local advertisement for $s=2$

Fig. 6. Advertise operation

area $A1$. (Recall that those queries that are originating in $A1$ and outside the cone boundaries are intercepted by the advertisement on the cone boundary.) Consider a query within unit distance of e and that falls between e and C. By drawing the first lateral link at distance s from e, the stretch-factor is satisfied for this query. Moreover this first lateral link suffices for intercepting all queries within distance s of e inside the cone boundaries in a distance-sensitive manner. The second lateral link is drawn at distance s^2 and it handles queries within distance s–s^2 of e inside the cone boundaries. Proceeding in the same fashion other lateral links are drawn within area $A1$ inside the cone boundaries. The final lateral link is drawn at the boundary of $A1$, which is $d_e/(s+1)$ distance away from e. Figure 6 recaps the above discussion and shows the advertise operation for a given stretch-factor s.

For the implementation of the cone advertisement we exploit GPSR again. The event uses the angle of cone advertisement (defined as $\arcsin(1/s)$), its distance d_e from C, and its own coordinates to calculate the coordinates of the two endpoints of the cone and sends a "cone-boundary advertisement" message destined to each endpoint. While this message is being relayed hop-by-hop, each node it visits stores the metadata advertisement included in the message. Lateral link advertisement is performed similarly. The event calculates starting and ending points of lateral link advertisements, and sends a pair of "start lateral link" message to the calculated starting points on the eC line (i.e., 1, s, s^2, $\ldots$,$d_e/s+1$ away from e). The lateral link start points, when they receive these messages, repackages them into "lateral-link advertisement" messages, and send them to the endpoints precalculated by e. Each node relaying a lateral-link advertisement message stores the metadata about e included in the message.

Lemma 1. A query operation invoked in $A1$ within d distance of an event intercepts the event's advertise information within $d*s$ distance.

Proof: There are three cases. In the first case, the querying node q in $A1$ falls inside the boundaries of cone advertisement, whereas in the remaining two cases q is outside the cone advertisement boundaries. In case 2, the angle $\widehat{EQC}$ between the evader location, location of q, and that of C is less than

90° as depicted in Figure 7. And, in case 3 $\widehat{EQC}$ is greater than or equal to 90° as in Figure 8.

Case 1: In this case the querying node q falls within the cone boundaries. The lateral links advertisement within the cone satisfy the stretch-factor for these queries as discussed above.

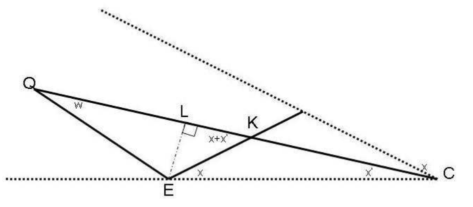


Fig. 7. Advertisement, case 2

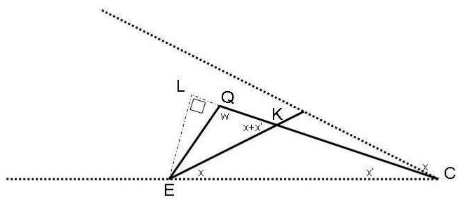


Fig. 8. Advertisement, case 3

Case 2 (see Figure 7): In order to prove that distance-sensitivity is satisfied for a query originating at Q, we need to show that the query is intercepted on its path from Q to C by the cone advertisement before the query travels more than $s * d$ distance. In other words, we need to prove that $|QK| < s|QE|$ in Figure 7.

$$|QK| < s|QE|$$
$$\equiv \{ |QK|{=}|QL|{+}|LK| \text{ and } |LK|{=}|LE|\cot(x+x')\}$$
$$|QL| + |LE|\cot(x+x') < s|QE|$$
$$\equiv \{ |QL| = |QE|\cos(w) \text{ and } |LE| = |QE|\sin(w)\}$$
$$|QE|\cos(w) + |QE|\sin(w)\cot(x+x') < s|QE|$$
$$\equiv \{ \sin(x) = 1/s \text{ (multiply both sides with } \sin(x)) \text{ also eliminate } |QE| \}$$
$$\sin(x)\cos(w) + \sin(x)\sin(w)\cot(x+x') < 1$$
$$\equiv \{ \text{Definition of } \cot(\alpha) \}$$
$$\sin(x)\cos(w) + \sin(x)\sin(w)\cos(x{+}x')/\sin(x{+}x') < 1$$
$$\equiv \{ \text{Definition of } \cos(\alpha+\beta) \text{ and } \sin(\alpha+\beta)\}$$
$$\sin(x)\cos(w) + \sin(x)\sin(w)(\cos(x)\cos(x') - \sin(x)\sin(x'))$$
$$/(\sin(x)\cos(x') + \cos(x)\sin(x')) < 1$$
$$\equiv \{ \text{Arithmetic} \}$$
$$\sin^2(x)\cos(w)\cos(x') + \sin(x)\cos(x)\cos(w)\sin(x') + \sin(x)\cos(x)\sin(w)\cos(x')$$
$$-\sin^2(x)\sin(w)\sin(x') < \sin(x+x')$$
$$\equiv \{ \text{Arithmetic} \}$$
$$\sin^2(x)(\cos(w)\cos(x') - \sin(w)\sin(x'))$$
$$+\sin(x)\cos(x)(\sin(w)\cos(x') + \cos(w)\sin(x')) < \sin(x+x')$$
$$\equiv \{ \text{Definition of } \cos(\alpha+\beta) \text{ and } \sin(\alpha+\beta)\}$$
$$\sin^2(x).\cos(w+x') + \sin(x)\cos(x)\sin(w+x') < \sin(x+x')$$
$$\equiv \{ \text{Arithmetic, definition of } \sin(\alpha+\beta)\}$$
$$\sin(x)\sin(x+w+x') < \sin(x+x')$$
$$\equiv \{ \text{Arithmetic} \}$$
$$\sin(x+w+x') < \sin(x+x')/\sin(x)$$

Note that $0 \leq x' \leq x \leq 90°$, also $x + x' + w < 180°$ as they are in a triangle. There are two cases.

Case A ($x + x' \leq 90°$): Then, $\sin(x + x')/\sin(x) > 1$ is satisfied due to property of *sine* for angles between $0° - 90°$. Since $\sin(\alpha) \leq 1$, for any α, we have $\sin(x + w + x') < \sin(x + x')/\sin(x)$.

Case B ($90° \leq x + x' < 180°$): Note that, $\sin(x + x')/\sin(x) > \sin(x + x')$, since $\sin(x) \leq 1$, for any x. Also, $\sin(x + x') > \sin(x + w + x')$, since $90° < x + w + x' < 180°$ and as angle increases sine decreases in that interval.

<u>Case 3</u> (see Figure 8): Similar to Case 2, in order to prove that distance-sensitivity is satisfied, we need to prove here that $|QK| < s|QE|$ in Figure 8.

$|QK| = |LK| - |QL|$. Note that $|QL| = |QE|\cos(180 - w) = -\cos(w)|QE|$.

$$
\begin{aligned}
&|QK| < s|QE| \\
\equiv\ & \{\ |QK| = |LK| - |QL|\} \\
&|LK| - |QL| < s|QE| \\
\equiv\ & \{\ |QL| = |QE|\cos(180 - w) = -\cos(w)|QE|\} \\
&|LK| + |QE|\cos(w) < s|QE| \\
\equiv\ & \{\ |LK| = |LE|\cot(x + x') \text{ and } |LE| = |QE|\sin(180 - w) = |QE|\sin(w)\} \\
&|QE|\sin(w)\cot(x + x') + |QE|\cos(w) < s|QE| \\
\equiv\ & \{\text{ Same inequality as in Case 1}\} \\
&\sin(x + w + x') < \sin(x + x')/\sin(x)
\end{aligned}
$$

Since $0 \leq x' \leq x \leq 90$ both subcases in Case 2 apply without modification. ◇

Theorem 1. A query operation invoked within d distance of an event intercepts the event's advertise information within $\mathbf{min}(d{*}s,\ d_q)$ distance, where d_q is the distance between the querying node and C.

Proof: There are two cases. If querying node is in $A1$, due to Lemma 1, the cost of querying is given as $d{*}s$. From the construction of $A1$, we have $d{*}s \leq d_q$ for any point q in $A1$, hence the querying cost $= \min(d{*}s,\ d_q)$ for this first case. If the querying node is in $A2$, then from construction $d_q < d*s$, and querying cost $= \min(d{*}s,\ d_q)$ is readily satisfied even in the worst case (when query goes d_q distance to C). ◇

3.3 Implementation and robustness issues

One simple example of a dense, connected, and multihop network is a grid WSN, where each node sits on a grid vertex and is in direct communication range of its grid neighbor nodes. The grid network model has been used in several real-world WSN deployments, such as "A Line In The Sand" [1] and ExScal [2]. Even for such simple and regular networks, due to deployment errors, localization errors, or the irregularities of wireless radio communication at the single-hop level, geometric properties such as triangle inequality may not be satisfied over small distances. However, for nodes that are sufficiently apart these errors at the single-hop level are immaterial, and geometric routing protocols such as GPSR and CLDP would satisfy $dist(j, k) + dist(k, l) \geq dist(j, l)$ for such nodes j, k, l.

Furthermore, due to energy depletion and faults, some nodes may fail leading to holes in the WSN topology (we assume that the network may not be partitioned). As we present above, Glance uses minimal infrastructure, and does not require costly constructions, such as hierarchical partitionings. As such Glance is less vulnerable against node failures in the network. In contrast to hierarchy based solutions [6,9] that are highly vulnerable to failure of the nodes at the higher levels of the hierarchy, Glance can sustain a gracefully degraded performance under node failures.

More specifically, Glance is able to tolerate holes in the network in a graceful manner by relying on the robustness of the underlying GPSR or CLDP routing protocol. The distance-sensitivity performance of Glance depends essentially on that of the underlying geometric routing protocol for the given WSN topology. For the delivery of query messages GPSR uses right-hand rule across the holes, thus, in case of failure of some nodes on the cone boundary, the query is still able to get an answer from the remaining nodes on the cone as GPSR routes the query around the hole to these nodes. This way, the loss in performance is proportional to the size of the holes, and in the worst case the query may reach C for resolution. Advertising is also robust in the presence of holes. For the delivery of a cone-boundary advertisement message to the endpoint on the left hand side of the event, left-hand rule is employed across a hole to ensure proper coverage for queries invoked from the left hand side of the event, and similarly for the delivery to the endpoint on the right hand side of the event, right-hand rule is employed upon encountering a hole. Finally, in case of failure of C, the data exfiltration will route messages around C, and so the nodes around the hole containing C obtain the event information. Later, when a query is routed along the hole containing C, it will be able to get answers from the area.

4 Performance Evaluation

We analyze the cost of advertise and query operations as well as tradeoffs involved in these costs in Section 4.1. Then, in Section 4.2, we compare the performance of Glance with other in-network querying protocols in the literature. Finally, in Section 4.3, we present our simulation results.

4.1 Cost of advertise and query

From Figure 6, we calculate the cost of advertise operation as follows. Since we choose the angle for cone advertisement to be equal to the threshold angle $x = \arcsin(1/s)$, the cone meets the threshold angle halfway through d_e, and the length of the cone boundary is calculated as $(d_e/2)/\cos(x)$. Thus, the two cone boundaries induce $2 * (d_e/2)/\cos(x) = d_e/\cos(\arcsin(1/s))$ cost. We also need to account for the cost of lateral explorations inside the cone boundaries. Recall from Section 3.2 that lateral explorations are performed with exponentially increasing intervals between subsequent explorations and for up to distance $d_e/(s+1)$ away from the event. The height of a lateral link is obtained by multiplying the distance between the lateral link and the event with $\tan(x)$, and doubling the result. Thus, the cost for the lateral explorations is calculated as $2 * \sum_{i=0}^{\log_s(d_e/(s+1))} s^i * \tan(x)$. Hence, the overall cost for advertisement comes up to $d_e/\cos(\arcsin(1/s)) + 2 * \sum_{i=0}^{\log_s(d_e/(s+1))} s^i * \tan(\arcsin(1/s))$. We can simplify this term by using the formula for sum of geometric series as: $\frac{d_e}{\cos(\arcsin(1/s))} + \frac{2*\tan(\arcsin(1/s))*(\frac{s*d_e}{s+1}-1)}{(s-1)}$.

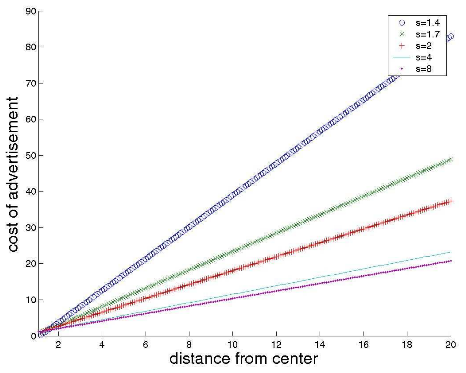


Fig. 9. Advertisement cost vs. d_e

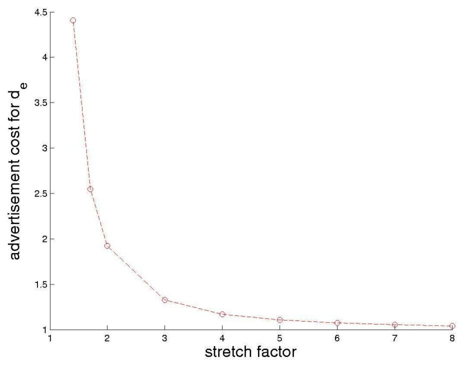


Fig. 10. Advertisement cost vs. s

As seen in Figure 9, the advertisement cost for an event e increases linearly with respect to the distance d_e of the event from the basestation C. The slope of this linear increase is determined by s. For s very close to 1, that is, for x close to 90°, the cost of advertisement can get high as $\cos(x)$ decreases, $(s-1)$ gets close to 0, and $\tan(x)$ increases. For example, for $s=1.15$ (i.e., $x=60°$), the cost of advertise is around $14 * d_e$. However, as s increases, the cost of advertise drops significantly fast. For example, $s=1.4$ (i.e., $x=45°$), the cost of advertise is less than $4.3*d_e$, and for $s=2$ (i.e., $x=30°$) the cost is less than $1.92 * d_e$, and for $s=4$ (i.e., $x=14.5°$) the cost is around $1.16 * d_e$. Figure 10 illustrates the relation between s and the cost of advertisement.

As proved in Theorem 1, the worst case cost of querying is $\min(d*s,\ d_q)$.

Analysis of tradeoffs in stretch-factor selection. In Glance, by tuning the stretch-factor s, the user specifies the level of distance sensitivity desired for answering queries. The cost of querying is directly proportional to s: by selecting small values for s, the cost is reduced. On the other hand, Figure 10 shows that the cost of advertise operation is inversely proportional to s: by selecting larger values for s, the cost of advertising is reduced. Thus, by tuning the value of s appropriately, the user can achieve tradeoffs between the cost of querying and advertising, and easily adjust the pull-push balance [18] to accommodate for different query-event rates in the WSN.

The user can define different stretch-factor requirements with respect to the type (i.e., importance) of events. One way to approach this tradeoff issue is to take a query-centric view. The user can first decide the highest tolerable stretch-factor in the application (e.g., based on real-time requirements of the query), and use this for the value of s. However, if there are no query-centric hard deadlines for the stretch-factor or the constraints for energy and communication efficiency dominates the design decisions, then it is possible to take an advertisement-centric approach. Depending on the query-event rates, the user can decide on the desired communication cost for advertising an event and then reverse engineer s using this cost. For example, when there are a lot of queries and few events in the WSN, higher advertisement costs for events become more affordable.

Analysis of scalability with respect to multiple events and queries. In the presence of multiple events and queries, Glance can be easily extended to use geographic hashing [21] and multiple basestations to improve load-balancing among basestations and achieve scalability with respect to the number of events and queries. The idea here is to partition events to multiple basestations based on the types of events so that network contention and bottlenecks are avoided at a basestation. Moreover, the user can define different stretch-factor requirements with respect to the type of events.

4.2 Performance comparison

In our comparisons we add to the cost of our advertise operation in Glance an extra d_e cost: the cost of data exfiltration to C which is in fact a part of the centralized monitoring mode operations. We do this so as not to put the other protocols at a disadvantage.

Comparison with GHT and hierarchical GHT. GHT hashes an event and a query for the event to a common broker. For comparing GHT and Glance, we assume that this broker is C located in the middle of the network.

The cost of storing an event at C corresponds to the cost of exfiltration of information to C in Glance. Hence, the cone advertisement in Glance remains as an extra cost over that of the advertise operation in GHT. For example, for $s = 2$, Glance pays an extra $1.92 * d_e$ cost for cone advertisement. The query operation in GHT, on the other hand, is more costly than that of Glance, since GHT does not satisfy distance-sensitivity. For a square network with diameter D, the average cost of querying (averaged over distance d_q of all querying nodes to C) in GHT is calculated as $D/3$. Note that this corresponds to the cost of going to C for the resolution of all queries. However, since Glance is distance-sensitive, queries are resolved in $\min(d * s, d_q)$ distance, where d is the distance to the nearest event, and a typical value for s is 2. Hence, the average cost of querying in Glance is lower than that of GHT. Especially, for a setup where the number of queriers are more than that of events, Glance would be more energy efficient than GHT, because the queries are answered locally. For the application setups where query frequencies increase with respect to the proximity to the event sources [4], almost all queries would be answered locally, leading to even more energy-savings for Glance. Also, Glance is preferable to GHT when there is a hard deadline (such as a real-time requirement) for the query operation.

Comparison with Stalk and DSIB. In Stalk, the advertisement cost of an event is calculated as $2 * d_e$. With this cost for the advertise operation, it is possible to achieve a stretch-factor of 4 for the querying cost in Glance. In contrast, the stretch factor in Stalk is given as $4 * w$, where w is the number of neighbors at any level of the hierarchy and ranges between 6 and 12. Thus, the cost of querying in Stalk is several times more than that calculated for Glance. However, we note that Stalk can achieve distance-sensitive tracking of mobile objects, whereas Glance does not address the mobility of events.

In DSIB, to achieve distance-sensitivity an event advertises to w, $6 \leq w \leq 12$ neighboring clusterheads as well as its clusterhead at every level of the hierarchy [9]. The cost of this advertisement is calculated as $2 * w * D$, where D is the diameter of the network. [1] In turn DSIB proves a stretch factor of 4 for the query operation. For $s = 4$ the advertisement cost in Glance corresponds to $2.16 * d_e$, including the cost of data exfiltration to C. Since d_e is the distance between the event and C, it is guaranteed to be less than D. Hence, Glance is able to achieve the same cost for querying as DSIB with around 1/9th of the cost required for advertisement in DSIB. On the other hand, an advantage of DSIB is that it can be implemented using the discrete centric hierarchy method [10] in the absence of localization information.

[1] This cost is equal to the sum of $2^0 w + 2^1 w + \ldots + 2^{\log(D)} w$, as the number of levels in the hierarchy is $\log D$.

Here, we evaluate the performance of *Glance* using simulations in *JProwler* [25], a discrete event simulator for WSNs with realistic radio and channel models. The goals of our simulation are the following: (1) to study the effects of discretization errors and routing overhead on the query and advertisement costs, (2) to study the effect of uniform node failures on the performance of *Glance*, and (3) to compare the average costs for querying and advertisement from the simulations with the upper bounds we derived earlier in our analysis.

Experimental Setup: Our simulations are performed on a 101-by-101 Mica2 mote network arranged in a grid topology. The center of the network C resides at coordinates $(50, 50)$. For routing messages, we implement geographic routing on the grid. In the presence of failures we employ the left and right hand rules in GPSR [14] to route around the holes in the WSN. We consider a grid separation of unit distance. Each node has upto 4 neighbors along the grid; there may be less number of neighbors in the presence of failures. We assume an underlying link maintenance layer using which the list of *up* neighbors is known at each node. We inject uniform node failures of upto 20% in the network and study the performance of *Glance* under these failures. We also simulate *Glance* under three different desired stretch factors: $s = 2$, $s = 3$ and $s = 4$ and compare the performance in terms of advertisement and query costs.

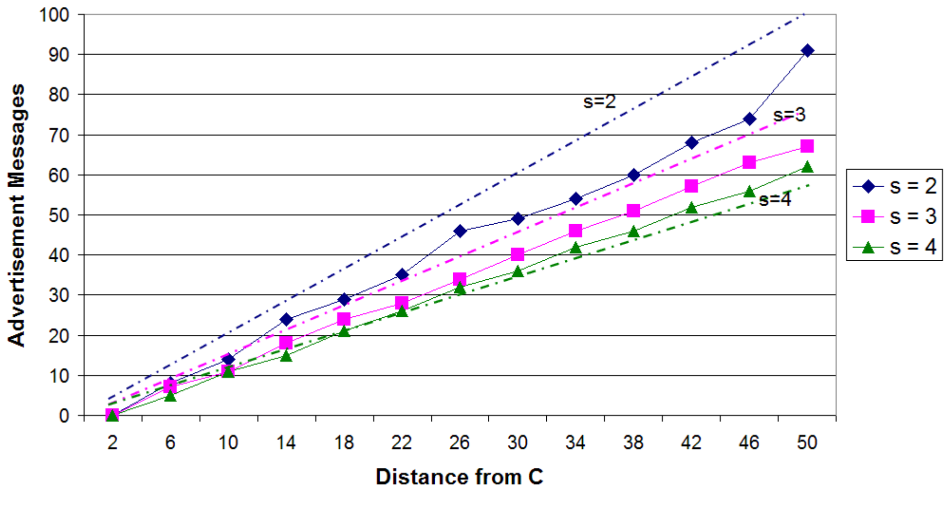


Fig. 11. Impact of s on advertisement cost

Advertisement Costs: For our first set of experiments, we initiate the advertising event at varying distances (2–50 units) from C and record the total number of messages exchanged for creating the *Glance* event advertisement structure. At each distance from C, the cost of advertisement is computed as the average over 1000 runs. Figure 11 presents the cost of advertisement messages under three different desired stretch factors. We notice that the cost of advertising decreases as s increases. This is because, for larger s, the advertisement cone becomes narrower. Despite the routing overhead due to routing on a grid, the cost of advertising in the simulations are lower than the analytical bounds (indicated by the dashed lines in the figure). In Figure 12, we study the impact of uniform failures on the advertisement costs by fixing $s = 2$. We notice that, as the failure rate increase (upto 20%), there is a gradual increase in the number of messages due to local rerouting in the presence of holes in the network.

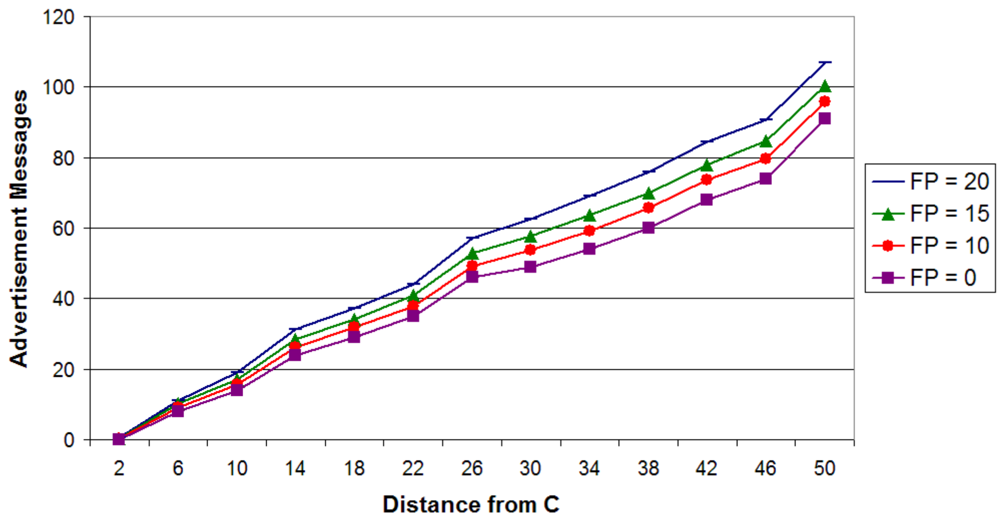


Fig. 12. Impact of failure rate on advertisement cost for $s = 2$

Query Costs: In order to calculate the querying costs, we fix the advertising event at varying distances (2–50 units) from C, and for each such location of the event, we uniformly vary the distance d between the querier and the event from 1 to 50 units. For each pair of locations for the querier and the event, the query cost is computed as an average over 20 such iterations. Figure 13 shows that the averages of the achieved stretch factors (calculated as the ratio of the number of find messages to d) are much lower than the desired stretch factor parameter s, since s imposes only an upper bound on the stretch factor. We investigate the variance in the achieved stretch factors in the next two graphs.

Figure 13 also illustrates the impact of failures on the querying cost. As the failure rates increase, the number of disconnections in the network increase, and we find that there is a small increase in the querying costs. The reason querying costs are not affected too much from failures is that, as failures increase, the advertisement structure by the event already compensates for it by advertising around the failures (as in Figure 12).

In Figure 14, we investigate the variance of the achieved stretch factor for $s = 2$ with respect to the distance d between the querying node and the event. We

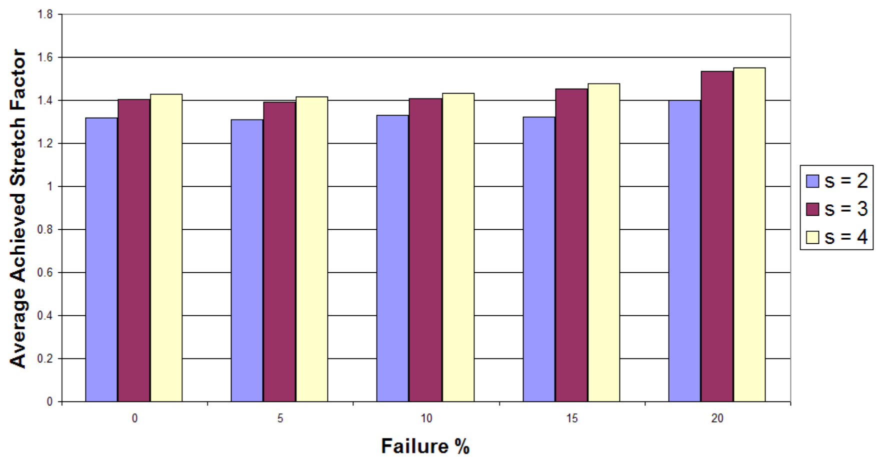


Fig. 13. Average Achieved Stretch Factor

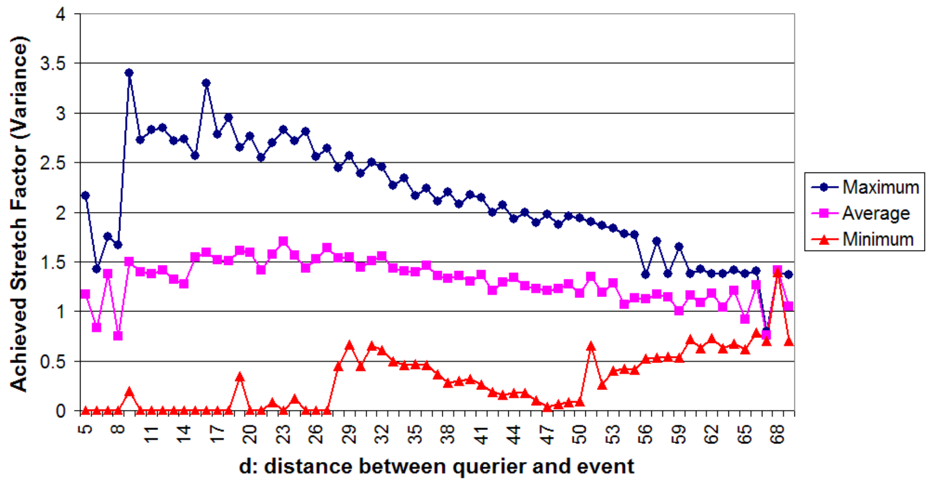


Fig. 14. Variation in achieved stretch factor for $s = 2$

notice that, as d increases, there is an increase followed by a decreasing trend in the *achieved maximum stretch factor*. The initial increase is due to the increase in the angles between the querying node and the event: since for this case the query should be intercepted by the event advertisements, the larger the angle the later the intercepting occurs. However, for larger d and larger angles between the querying node and the event, going to C becomes feasible, and a transition occurs in the achieved maximum stretch factors. We see that, as d increases further, the achieved maximum stretch factor starts to decrease and converge to 1. Even for large distances, the *achieved minimum stretch factor* remains small due to the cases where the event advertisement resides on the path from the querying node to C. However, for $d > 50$, this situation cannot arise and the achieved minimum stretch factor starts to converge towards 1.

In Figure 15, we compare the maximum achieved stretch factors for $s = 2$, $s = 3$ and $s = 4$. We observe that the maximum achieved stretch factors are close to

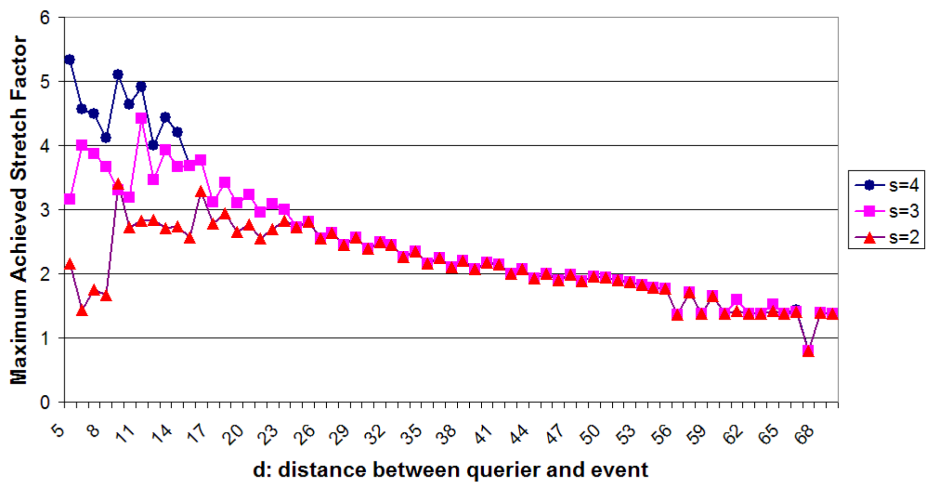


Fig. 15. Comparison of achieved maximum stretch factors

s for all three cases. The difference is due to the overhead of using geographic routing on a grid which is upperbounded by $\sqrt{2}$. The maximum achieved stretch factors observed in our simulations are thus close to the theoretical bounds we derived. As in Figure 14, we also notice that, as d increases, the maximum stretch factors start decreasing and converging together to 1. This is because, after a certain threshold, the query resolution strategy defaults to going to C and hence all the three cases start to behave similarly.

5 Concluding Remarks

In this paper we showed that it is possible to devise a simple and lightweight solution for distance-sensitive in-network querying in WSN by exploiting basic geometry concepts. Our main insight was to use the basestation node in an opportunistic manner for answering of some in-network queries. The knowledge that all queries target the basestation by default, combined with the geometry of the network, was useful in determining the minimum area required for in-network advertisements to satisfy a given distance-sensitivity requirement. We observed that in-network advertisements can safely ignore a majority of directions/regions and focus their advertisement to a small cone to be able to satisfy a given distance-sensitivity requirement.

As a result, we presented a simple and lightweight querying service *Glance*, that ensures that a query invoked within d distance of an event intercepts the event's advertisement within $d * s$ distance, where s is a "stretch-factor" tunable by the user. The user may define different stretch-factor requirements (which lead to varying angles for cone advertisement) with respect to the type (i.e., severity) of events. By selecting appropriate values for s it is possible to achieve trade-offs between query execution cost and advertisement cost. Glance is also resilient with respect to node failures and holes in the network.

It is possible to avoid the need for localization in the Glance protocol. The idea here is to use an approximation for the direction to the basestation node C. In this scheme, in the initialization phase C starts a one-time flood that annotates each node in the network with its hopcount from C and creates a spanning tree rooted at C. To send the query as a straight line to C, it is enough to route the message to the parent node along a branch in this tree. Since it is infeasible to draw cone borders as in Figure 5 in the absence of localization, this scheme may approximate that with occasional lateral exploration inside the cone by visiting the nodes with same hopcount at predefined distances from the event.

As a broader research direction, we will further investigate the adaptation of geometric ideas and techniques for devising distributed network algorithms. We note from our previous experience [6, 7, 16] that when the problem domain is constrained to geometric networks it is possible to devise simpler and more efficient algorithms than those designed for arbitrary graph topologies. With the recent advances in directional antenna technology and the availability of directional communication in WSN, we believe that the application of geometric ideas to the distributed WSN domain may yield new research opportunities.

References

[1] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y-R. Choi, T. Herman, S. S. Kulkarni, U. Arumugam, M. Nesterenko, A. Vora, and M. Miyashita. A line in the sand: A wireless sensor network for target detection, classification, and tracking. *Computer Networks (Elsevier)*, 46(5):605–634, 2004.

[2] A. Arora and et. al. Exscal: Elements of an extreme scale wireless sensor network. *Int. Conf. on Embedded and Real-Time Computing Systems and Applications*, 2005.

[3] D. Braginsky and D. Estrin. Rumor routing algorthim for sensor networks. In *WSNA*, pages 22–31, 2002.

[4] H. Cao, E. Ertin, V. Kulathumani, M. Sridharan, and A. Arora. Differential games in large scale sensor actuator networks. In *IPSN '06: Proceedings of the fifth international conference on Information processing in sensor networks*, pages 77–84, 2006.

[5] M. Demirbas, A. Arora, and M. Gouda. Pursuer-evader tracking in sensor networks. *Sensor Network Operations, IEEE Press*, 2006.

[6] M. Demirbas, A. Arora, T. Nolte, and N. Lynch. A hierarchy-based fault-local stabilizing algorithm for tracking in sensor networks. *8th International Conference on Principles of Distributed Systems (OPODIS)*, pages 299–315, 2004.

[7] M. Demirbas and H. Ferhatosmanoglu. Peer-to-peer spatial queries in sensor networks. *The Third IEEE Int. Conf. on Peer-to-Peer Computing*, pages 32–39, September 2003.

[8] E. W. Dijkstra. *A Discipline of Programming.* Prentice Hall, 1976.

[9] S. Funke, L. J. Guibas, A. Nguyen, and Y. Wang. Distance-sensitive routing and information brokerage in sensor networks. In *DCOSS*, 2006.

[10] J. Gao, L. J. Guibas, and A. Nguyen. Deformable spanners and applications. In *Symposium on Computational Geometry (SCG)*, pages 190–199, 2004.

[11] B. Greenstein, D. Estrin, R. Govindan, S. Ratnasamy, and S. Shenker. Difs: A distributed index for features in sensor networks. *Int. Workshop on Sensor Network Protocols and Applications*, May 2003.

[12] Y. Huang and H. Garcia-Molina. Publish/subscribe tree construction in wireless ad-hoc networks. In *Proceedings of the 4th International Conference on Mobile Data Management*, pages 122–140, 2003.

[13] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva. Directed diffusion for wireless sensor networking. *IEEE/ACM Trans. Netw.*, 11(1):2–16, 2003.

[14] B. Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *MobiCom*, pages 243–254, 2000.

[15] Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker. Geographic routing made practical. *Proceedings of the USENIX Symposium on Networked Systems Design and Implementation*, May 2005.

[16] V. Kulathumani, A. Arora, M. Demirbas, and M. Sridharan. Trail: A distance sensitive network protocol for distributed object tracking. Technical Report OSU-CISRC-7/06-TR67, The Ohio State University, 2006.

[17] J. Li, J. Jannotti, D. S. J. De Couto, D. R. Karger, and R. Morris. A scalable location service for geographic ad hoc routing. In *MobiCom*, pages 120–130, 2000.

[18] X. Liu, Q. Huang, and Y. Zhang. Combs, needles, haystacks: balancing push and pull for discovery in large-scale sensor networks. In *SenSys*, pages 122–133, 2004.

[19] S. Madden, M. Franklin, J. Hellerstein, and Wei Hong. Tinydb: an acquisitional query processing system for sensor networks. *ACM Trans. Database Syst.*, 30(1):122–173, 2005.

[20] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson. Wireless sensor networks for habitat monitoring. In *WSNA*, 2002.

[21] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker. Ght: a geographic hash table for data-centric storage. In *WSNA*, pages 78–87, 2002.

[22] S. Shenker, S. Ratnasamy, B. Karp, R. Govindan, and D. Estrin. Data-centric storage in sensornets. *SIGCOMM Comput. Commun. Rev.*, 33(1):137–142, 2003.

[23] R. Szewczyk, A. Mainwaring, J. Polastre, and D. Culler. An analysis of a large scale habitat monitoring application. In *Sensys*, 2004.

[24] G. Tolle, J. Polastre, R. Szewczyk, N. Turner, K. Tu, P. Buonadonna, S. Burgess, D. Gay, W. Hong, T. Dawson, and D. Culler. A macroscope in the redwoods. In *SenSys*, 2005.

[25] Vanderbilt University, `http://www.isis.vanderbilt.edu/Projects/nest/jprowler`. *JProwler*, 2004.

[26] Y. Yao and J. E. Gehrke. Query processing in sensor networks. *Conference on Innovative Data Systems Research (CIDR)*, 2003.