A Two-Tier Data Dissemination Model for Large-scale Wireless Sensor Networks*

Fan Ye, Haiyun Luo, Jerry Cheng, Songwu Lu, Lixia Zhang
UCLA Computer Science Department
Los Angeles, CA 900095-1596
{yefan,hluo,chengje,slu,lixia}@cs.ucla.edu

ABSTRACT

Sink mobility brings new challenges to large-scale sensor networking. It suggests that information about each mobile sink's location be continuously propagated through the sensor field to keep all sensor nodes updated with the direction of forwarding future data reports. Unfortunately frequent location updates from multiple sinks can lead to both excessive drain of sensors' limited battery power supply and increased collisions in wireless transmissions. In this paper we describe *TTDD*, a *Two-Tier Data Dissemination* approach that provides scalable and efficient data delivery to multiple mobile sinks. Each data source in TTDD proactively builds a grid structure which enables mobile sinks to continuously receive data on the move by flooding queries within a local cell only. TTDD's design exploits the fact that sensor nodes are stationary and location-aware to construct and maintain the grid structures with low overhead. We have evaluated TTDD performance through both analysis and extensive simulation experiments. Our results show that TTDD handles multiple mobile sinks efficiently with performance comparable with that of stationary sinks.

Categories and Subject Descriptors

C.2.2 [**Communication Networks**]: Network Protocols

General Terms

Design, Performance

Keywords

Sensor networks, sink mobility, two-tier model

*This work is supported in part by the DARPA SensIT program under contract number DABT63-99-1-0010. Lu is also supported by an NSF CAREER award (ANI-0093484).

MOBICOM'02, September 23–26, 2002, Atlanta, Georgia, USA.

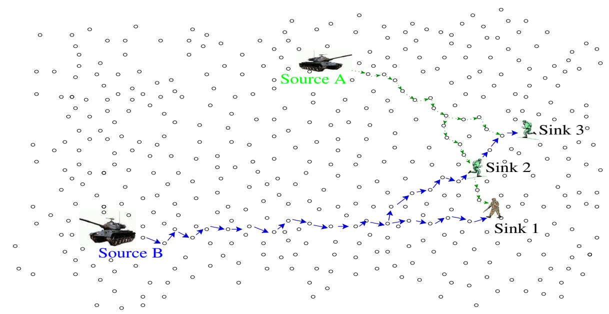

Figure 1: A sensor network example. Soldiers use the sensor network to detect tank locations.

1. INTRODUCTION

Recent advances in VLSI, microprocessor and wireless communication technologies have enabled the deployment of large-scale sensor networks where thousands, or even tens of thousands of small sensors are distributed over a vast field to obtain fine-grained, high-precision sensing data [9, 10, 15]. These sensor nodes are typically powered by batteries and communicate through wireless channels.

This paper studies the problem of scalable and efficient data dissemination in a large-scale sensor network from potentially multiple sources to potentially multiple, *mobile* sinks. In this work a source is defined as a sensor node that detects a *stimulus*, which is a target or an event of interest, and generates data to report the stimulus. A sink is defined as a user that collects these data reports from the sensor network. Both the number of stimuli and that of the sinks may vary over time. For example in Figure 1, a group of soldiers collect tank movement information from a sensor network deployed in a battlefield. The sensor nodes surrounding a tank detect it and collaborate among themselves to aggregate data, and one of them generates a data report. The soldiers collect these data reports. In this paper we consider a network made of stationary sensor nodes only, whereas sinks may change their locations dynamically. In the above example, the soldiers may move around, and must be able to receive data reports continuously.

Sink mobility brings new challenges to large-scale sensor networking. Although several data dissemination protocols have been developed for sensor networks recently, such as Directed Diffusion [10], Declarative Routing Protocol [5] and GRAB [20], they all suggest that each mobile sink need to continuously propagate its location information throughout the sensor field, so that all sensor nodes get updated with the direction of sending future data reports. However, frequent location updates from multiple sinks can lead to both increased collisions in wireless transmissions and rapid power consumption of the sensor's limited battery supply. None of the existing approaches provides a scalable and efficient solution to this problem.

In this paper, we describe *TTDD*, a *Two-Tier Data Dissemination* approach to address the multiple, mobile sink problem. Instead of propagating query messages from each sink to *all* the sensors to set up data forwarding information, TTDD design uses a grid structure so that only sensors located at grid points need to acquire the forwarding information. Upon detection of a stimulus, instead of passively waiting for data queries from sinks — the approach taken by most of the existing work, the data source *proactively* builds a grid structure throughout the sensor field and sets up the forwarding information at the sensors closest to grid points (henceforth called dissemination nodes). With this grid structure in place, a query from a sink traverses two tiers to reach a source. The lower tier is within the local grid square of the sink's current location (henceforth called cells), and the higher tier is made of the dissemination nodes at grid points. The sink floods its query within a cell. When the nearest dissemination node for the requested data receives the query, it forwards the query to its upstream dissemination node toward the source, which in turns further forwards the query, until it reaches either the source or a dissemination node that is already receiving data from the source (e.g. upon requests from other sinks). This query forwarding process lays information of the path to the sink, to enable data from the source to traverse the same two tiers as the query but in the reverse order.

TTDD's design exploits the fact that sensor nodes are both stationary and location-aware. Because sensors are assumed to know their locations in order to tag sensing data [1, 8, 18], and because sensors' locations are static, TTDD can use simple greedy geographical forwarding to construct and maintain the grid structure with low overhead. With a grid structure for each data source, queries from multiple mobile sinks are confined within their local cells only, thus avoiding excessive energy consumption and network overload from global flooding by multiple sinks. When a sink moves more than a cell size away from its previous location, it performs another local flooding of data query which will reach a new dissemination node. Along its way toward the source this query will stop at a dissemination node that is already receiving data from the source. This dissemination node then forwards data downstream and finally to the sink. In this way, even when sinks move continuously, higher-tier data forwarding changes incrementally and the sinks can receive data without interruption. Furthermore, because only those sensors on the grid points (serving as dissemination nodes) of a data source participate in its data dissemination, other sensors are relieved from maintaining states. Thus TTDD can effectively scale to a large number of sources and sinks.

The rest of this paper is organized as follows. Section 2 describes the main design including grid construction, the two-tier query and data forwarding, and grid maintenance. Section 3 analyzes the communication overhead and the state complexity of TTDD, and compares with other sink-oriented data dissemination designs. Simulation results are presented in Section 4 to evaluate the effectiveness of our design and analyze the impact of important parameters. We discuss several design issues in Section 5 and compare with the related work in Section 6. Section 7 concludes the paper.

2. TWO-TIER DATA DISSEMINATION

This section presents the basic design of TTDD which is based on the following assumptions:

- A vast field is covered by a large number of homogeneous sensor nodes which communicate with each other through short-range radios. Long distance data delivery is accomplished by forwarding data across multiple hops.
- Each sensor node is aware of its own location (for example through receiving GPS signals or through techniques such as [1]). However, mobile sinks may or may not know their own locations.
- Once a stimulus appears, the sensors surrounding it collectively process the signal and one of them becomes the source to generate data reports [20].
- Sinks (users) query the network to collect sensing data. There can be multiple sinks moving around in the sensor field and the number of sinks may vary over time.

The above assumptions are consistent with the models for real sensors being built, such as UCLA WINS NG nodes [15], SCADDS PC/104 [4], and Berkeley Motes [9].

In addition, TTDD design assumes that the sensor nodes are aware of their missions (e.g., in the form of the signatures of each potential type of stimulus to watch). Each mission represents a sensing task of the sensor network. In the example of tank detection of Figure 1, the mission of the sensor network is to collect and return the current locations of tanks. In scenarios where the sensor network mission changes, the new mission can be flooded through the field to reach all sensor nodes. In this paper we do not discuss how to manage the missions of sensor networks. However we do assume that the mission of a sensor network changes only infrequently, thus the overhead of mission dissemination is negligible compared to that of sensing data delivery.

As soon as a source generates data, it starts preparing for data dissemination by building a grid structure. The source starts with its own location as one crossing point of the grid, and sends a data announcement message to each of its four adjacent crossing points. Each data announcement message finally stops on a sensor node that is the *closest* to the crossing point specified in the message. The node stores the source information and further forwards the message to its adjacent crossing points except the one from which it received the message. This recursive propagation of data announcement messages notifies those sensors that are closest to the crossing locations to become the dissemination nodes of the given source.

Once a grid for the specified source is built, a sink can then flood its query within a local cell to receive data. The query will be received by the nearest dissemination node on the grid, which then propagates the query upstream through other dissemination nodes toward the source. Requested data will flow down in the reverse path to the sink.

The above seemingly simple TTDD operation poses several research challenges. For example, given that locations of sensors are random and not necessarily on the crossing points of a grid, how do nearby sensors of a grid point decide

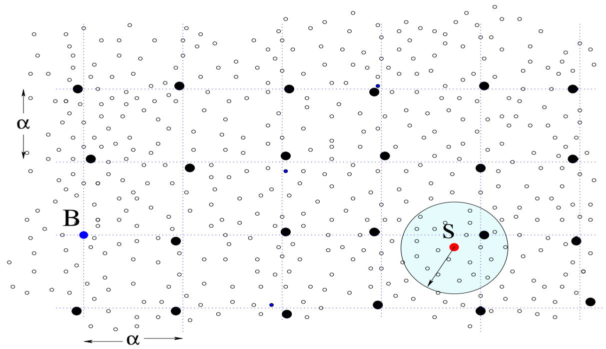

Figure 2: One source B and one sink S

which one should serve as the dissemination node? Once the data stream starts flowing, how can it be made to follow the movement of a sink to ensure continued delivery? Given individual sensors are subject to unexpected failures, how is the grid structure maintained, once it is built? The remaining of this section will address each of these questions in detail. We start with the grid construction in Section 2.1, and present the two-tier query and data forwarding in Section 2.2. Grid maintenance is described in Section 2.3.

2.1 Grid Construction

To simplify the presentation, we assume that a sensor field spans a two-dimensional plane. A source divides the plane into a *grid* of cells. Each cell is an $\alpha \times \alpha$ square. A source itself is at one crossing point of the grid. It propagates data announcements to reach all other crossings, called *dissemination points*, on the grid. For a particular source at location $L_s = (x, y)$, dissemination points are located at $L_p = (x_i, y_j)$ such that:

$$\{x_i = x + i \cdot \alpha, y_j = y + j \cdot \alpha; i, j = \pm 0, \pm 1, \pm 2, \cdots\}$$

A source calculates the locations of its four neighboring dissemination points given its location (x, y) and cell size α. For each of the four dissemination points L_p, the source sends a data-announcement message to L_p using simple greedy geographical forwarding, i.e., it forwards the message to the neighbor node that has the *smallest* distance to L_p. The neighbor node continues forwarding the data announcement message in a similar way till the message stops at a node that is closer to L_p than *all* its neighbors. If this node's distance to L_p is less than a threshold $\alpha/2$, it becomes a *dissemination node* serving dissemination point L_p for the source. In cases where a data announcement message stops at a node whose distance to the designated dissemination point is greater than $\alpha/2$, the node simply drops the message.

A dissemination node stores a few pieces of information for the grid structure, including the data announcement message, the dissemination point L_p it is serving and the upstream dissemination node's location. It then further propagates the message to its neighboring dissemination points on the grid except the upstream one from which it receives the announcement. The data announcement message is *recursively* propagated through the whole sensor field so that each dissemination point on the grid is served by a dissemination node. Duplicate announcement messages from different neighboring dissemination points are identified by the sequence number carried in the announcement and simply dropped.

Figure 2 shows a grid for a source B and its virtual grid. The black nodes around each crossing point of the grid are the dissemination nodes.

2.1.1 *Explanation of Grid Construction*

Because the above grid construction process does not assume any *a-priori* knowledge of potential positions of sinks, it builds a uniform grid in which all dissemination points are regularly spaced with distance α in order to distribute data announcements as evenly as possible. The knowledge of the global topology is not required at any node; each node acts based on information of its local neighborhood only.

In TTDD, the dissemination point serves as a reference location when selecting a dissemination node. The dissemination node is to be selected as close to the dissemination point as possible, so that the dissemination nodes are evenly distributed to form a uniform grid infrastructure. However, the dissemination node is not required to be globally closest to the dissemination point. Strictly speaking, TTDD ensures that a dissemination node is locally closest but not necessarily globally closest to the dissemination point, due to irregularities in topology. This will not affect the correct operation of TTDD. The reason is that each dissemination node includes its own location (not that of the dissemination point) in its further data announcement messages. This way, downstream dissemination nodes will still be able to forward future queries to this dissemination node, even though the dissemination node is not globally closest to the dissemination point in the ideal grid. This is to be further discussed in Section 2.2.1.

We set the $\alpha/2$ distance threshold for a node to become a dissemination node in order to stop the grid construction at the network border. For example, in Figure 3, sensor node B receives a data announcement destined to P which is out of the sensor field. Because sensor nodes are not aware of the global sensor field topology, they cannot tell if a location is out of the network. Comparing with $\alpha/2$ provides nodes a simple rule to decide if the propagation should be terminated.

When a dissemination point falls in a void area without any sensor nodes in it, the data announcement propagation might stop on the border of the void area. But propagation can continue along other paths of the grid and go around the void area, since each dissemination node forwards the data announcement to all three other dissemination points. As long as the grid is not partitioned, data announcements will bypass the void by taking alternative paths.

We choose to build the grid on a *per-source* basis, so that different sources recruit different sets of dissemination nodes. This design choice enhances scalability and provides load balancing and better robustness. When there are many sources, as long as their grids do not overlap, a dissemination node only has states about one or a few sources. This allows TTDD to scale to large numbers of sources. We will analyze the state complexity in section 3.3. In addition, the per-source grid effectively distributes data dissemination load among different sensor nodes to avoid bottlenecks. This is motivated by the fact that each sensor node is energy-constrained and the sensor nodes' radios usually have limited bandwidth. The per-source grid construction also leads to enhanced robustness in the presence of node failures.

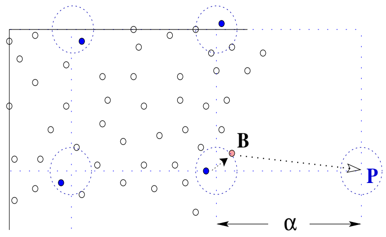

Figure 3: Termination on border

The grid cell size α is a critical parameter. As we can see in the next section, the general guideline to set the cell size is to *localize* the impact of sink mobility within a single cell, so that the higher-tier grid forwarding remains stable. The choice of α affects energy efficiency and state complexity. It will be further analyzed in Section 3 and evaluated in Section 4.

2.2 Two-Tier Query and Data Forwarding

2.2.1 Query Forwarding

Our two-tier query and data forwarding is based on the virtual grid infrastructure to ensure scalability and efficiency. When a sink needs data, it floods a query within a local area about a cell size large to discover nearby dissemination nodes. The sink specifies a maximum distance in the query thus the flooding stops at nodes that are more than the maximum distance away from the sink.

Once the query reaches a local dissemination node, which is called an *immediate dissemination node* for the sink, it is forwarded on the grid to the upstream dissemination node from which this immediate dissemination node receives data announcements. The upstream one in turn forwards the query further upstream toward the source, until finally the query reaches the source. During the above process, each dissemination node stores the location of the downstream dissemination node from which it receives the query. This state is used to direct data back to the sink later (see Figure 4 for an illustration).

With the grid infrastructure in place, the query flooding can be confined within the region of around a single cell-size. It saves significant amount of energy and bandwidth compared to flooding the query across the whole sensor field. Moreover, two levels of query aggregation[1] are employed during the two-tier forwarding to further reduce the overhead. Within a cell, an immediate dissemination node that receives queries for the same data from different sinks aggregates these queries. It only sends one copy to its upstream dissemination node, in the form of an *upstream update*. Similarly, if a dissemination node on the grid receives multiple upstream updates from different downstream neighbors, it forwards only one of them further. For example in figure 4, the dissemination node G receives queries from both the cell where sink S_1 is located and the cell where sink S_2 is located, and G sends only one upstream update message toward the source.

When an upstream update message traverses the grid, it installs soft-states in dissemination nodes to direct data

[1] For simplicity, we do not consider semantic aggregation [10] here, which can be used to further improve the aggregation gain for different data resolutions and types.

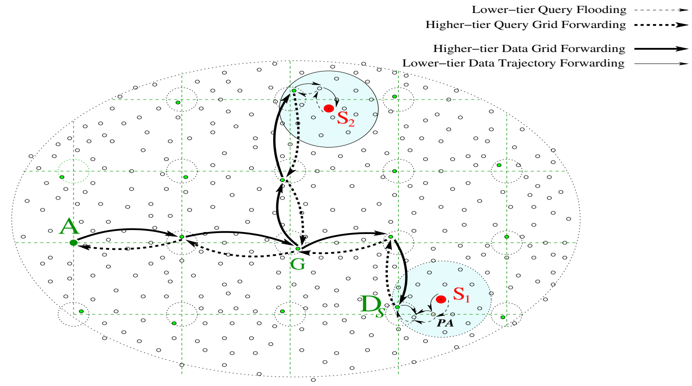

Figure 4: Two-tier query and data forwarding between Source *A* and Sink S_1, S_2. Sink S_1 starts with flooding its query with its primary agent PA's location, to its immediate dissemination node D_s. D_s records PA's location and forwards the query to its upstream dissemination node until the query reaches A. The data are returned to D_s along the way that the query traverses. D_s forwards the data to PA, and finally to Sink S_1. Similar process applies to Sink S_2, except that its query stops on the grid at dissemination node G.

streams back to the sinks. Unless being updated, these states are valid for a certain period only. A dissemination node sends such messages upstream periodically in order to receive data continuously; it stops sending such update messages when it no longer needs the data, such as when the sink stops sending queries or moves out of the local region. An upstream disseminate node automatically stops forwarding data after the soft-state expires. In our current design, the values of these soft-state timers are chosen an order-of-magnitude higher than the interval between data messages. This setting balances the overhead of generating periodic upstream update messages and that of sending data to places where it is no longer needed.

The two-level aggregation provides scalability with the number of sinks. A dissemination node on the query forwarding path maintains at most three states about which neighboring dissemination nodes need data. An immediate dissemination node maintains in addition only the states of sinks located within the local region of about a single cell-size. Sensor nodes that do not participate in query or data forwarding do not keep any state about sinks or sources. We analyze the state complexity in details in Section 3.3.

2.2.2 Data Forwarding

Once a source receives the queries (in the form of upstream updates) from anyone of its neighbor dissemination nodes, it sends out data to this dissemination node, which in turn forwards data to where it receives the queries, so on and so forth until the data reach each sink's immediate dissemination node. If a dissemination node has aggregated queries from different downstream dissemination nodes, it sends a data copy to each of them. For example in Figure 4 the dissemination node G will send data to both S_1 and S_2. Once the data arrive at a sink's immediate dissemination node, *trajectory forwarding* (see Section 2.2.3) is employed to further relay the data to the sink which might be in continuous motion.

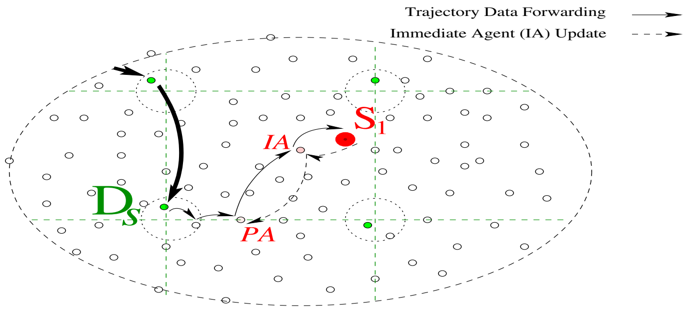

Figure 5: Trajectory Forwarding from immediate dissemination node D_s to mobile sink S_1 via primary agent PA and immediate agent IA. Immediate agent IA is one-hop away from S_1. It relays data directly to sink S_1. When S_1 moves out of the one-hop transmission range of its current IA, it picks a new IA from its neighboring nodes. S_1 then sends an update to its PA and old IA to relay data. PA is not changed as long as S_1 remains within some range to PA.

With the two-tier forwarding as described above, queries and data may take globally suboptimal paths, thus introducing additional cost compared to forwarding along shortest paths. For example in Figure 4, sink S_1 and S_2 may find straight-line paths to the source if they each flooded their queries across the whole sensor field. However, the path a message travels between a sink and a source by the two-tier forwarding is at most $\sqrt{2}$ times the length of that of a straight-line. We believe that the sub-optimality is well worth the gain in scalability. A detailed analysis is given in Section 3.

2.2.3 Trajectory Forwarding

Trajectory forwarding is employed to relay data to a mobile sink from its immediate dissemination node. In trajectory forwarding, each sink is associated with two sensor nodes: a *primary agent* and an *immediate agent*. A sink picks one neighboring sensor node as its primary agent and includes the location of the primary agent in its queries. Its immediate dissemination node sends data to the primary agent, which in turn relays data to the sink. Initially the primary agent and the immediate agent are the same sensor node.

When a sink is about to move out of the range of its current immediate agent, it picks another neighboring sensor node as its new immediate agent, and sends the location of the new immediate agent to its primary agent, so that future data are forwarded to the new immediate agent. To avoid losing data that have already been sent to the old immediate agent, the location is also sent to the old immediate agent (see Figure 5). The selection of a new immediate agent can be done by broadcasting a solicit message from the sink, which then chooses the node that replies with the strongest signal-to-noise ratio.

The primary agent represents the mobile sink at the sink's immediate dissemination node, so that the sink's mobility is made transparent to its immediate dissemination node. The immediate agent represents the sink at the sink's primary agent, so that the sink can receive data continuously while in constant movement. Thus a user that does not know his own location can still collect data from the network.

When the sink moves out of a certain distance, e.g., a cell size, from its primary agent, it picks a new primary agent and floods a query locally to discover new dissemination nodes that might be closer. To avoid receiving duplicate data from its old primary agent, TTDD lets each primary agent time out once its timer, which is set approximately to the duration a mobile sink remains in a cell, expires. The old immediate agent times out in a similar way, except that it has a shorter timer which is approximately the duration a sink remains within the one-hop distance. If a sink's immediate dissemination node does not have any other sinks or neighboring downstream dissemination nodes requesting data for a certain period of time (similar to the timeout value of the sink's primary agent), it stops sending update messages to its upstream dissemination node so that data are no longer forwarded to this cell.

An example is shown in figure 4, when the soft-state at the immediate dissemination node D_s expires, D_s stops sending upstream updates because it does not have any other sinks or neighboring downstream dissemination nodes requesting data. After a while, data forwarded at G only go to sink S_2, if S_2 still needs data. This way, all states built on the grid and in the old agents by a sink's old query are cleared.

With trajectory forwarding, sink mobility within a small range, e.g., a cell size, is made transparent to the higher-tier grid forwarding. Mobility beyond a cell-size distance that involves new dissemination node discoveries might affect certain upstream dissemination nodes on grids. Since the new dissemination nodes that a sink discovers are likely to be in adjacent cells, the adjustment to grid forwarding will typically affect a few nearby dissemination nodes only.

2.3 Grid Maintenance

To avoid keeping grid states at dissemination nodes indefinitely, a source includes a *Grid Lifetime* in the data announcement message when sending it out to build the grid. If the lifetime elapses and the dissemination nodes on the grid do not receive any further data announcements to update the lifetime, they clear their states and the grid no longer exists.

Proper grid lifetime values depend on the data availability period and the mission of the sensor network. In the example of Figure 1, if the mission is to return the "current" tank locations, a source can estimate the time period that the tank will stay around, and use this estimation to set the grid lifetime. If the tank stays longer than the original estimation, the source can send out new data announcements to extend the grid's lifetime.

For any structure, it is important to handle unexpected component failures for robustness. To conserve the scarce energy supply of sensor nodes, we do not periodically refresh the grid during its lifetime. Instead, we employ a mechanism called *upstream information duplication*, in which each dissemination node recruits from its neighborhood several sensor nodes and replicates in them the location of its upstream dissemination node. When this dissemination node fails, the upstream update messages from its downstream dissemination node that needs data will stop at one of these recruited nodes. The one then forwards the update message to the upstream dissemination node according to the stored information.[2] When data come from upstream later, a new dissemination node will emerge following the same rule as

[2]This neighbor can detect the failure of the dissemination

the source initially builds the grid.

Since this new dissemination node does not know which downstream dissemination node neighbors need data, it simply forwards data to all the other three dissemination points. A downstream dissemination node neighbor that needs data will continue to send upstream update messages to re-establish the forwarding state; whereas one that does not need data drops the data and does not send any upstream update, so that future data reports will not flow to it. Note that this mechanism also handles the scenario where multiple dissemination nodes fail simultaneously along the forwarding path.

The failure of the immediate dissemination node is detected by a timeout at a sink. When a sink stops receiving data for a certain time, it re-floods a query to locate a new dissemination node. The failures of primary agents or immediate agents are detected by similar timeouts and new ones will be picked.

Our grid maintenance is triggered on-demand by on-going queries or upstream updates. Compared with periodic grid refreshing, it trades computational complexity for less consumption of energy, which is a more critical resource. We show the performance of our grid maintenance through simulations in Section 4.4.

3. OVERHEAD ANALYSIS

In this section we analyze the *efficiency* and *scalability* of TTDD. We measure two metrics: the *communication overhead* for a number of sinks to retrieve a certain amount of data from a source, and the *complexity of the states* that are maintained in a sensor node for data dissemination. We study both the stationary and the mobile sink cases.

We compare TTDD with the sink-oriented data dissemination approach (henceforth called *SODD*), in which each sink first floods the whole network to install data forwarding state at all the sensor nodes, and then sources react to deliver data. Directed Diffusion [10], DRP [5] and GRAB [20] all take this approach, although each employs different optimization techniques, such as data aggregation and query aggregation, to reduce the number of messages to be delivered. Because both aggregation techniques are applicable to TTDD as well, we do not consider these aggregations when we compare the communication overhead. Instead, our analysis will focus on the *worst-case* communication overhead of each protocol. We aim at making the analysis simple and easy to follow while capturing the fundamental differences between TTDD and other approaches. We will add the consideration of the aggregations when we analyze the complexity in sensor state maintenance.

3.1 Model and Notations

We consider a square sensor field of area A in which N sensor nodes are uniformly distributed so that on each side there are approximately $\sqrt{N}$ sensor nodes. There are k sinks in the sensor field. They move at an average speed v, while receiving d data packets from a source in a time period of T. Each data packet has a unit size and both the query and data announcement messages have a comparable size l. The communication overhead to flood an area is proportional to the number of sensor nodes in it, and to send a message along a path by greedy geographical forwarding is proportional to the number of sensor nodes in the path. The average number of neighbors within a sensor node's wireless communication range is D.

In TTDD, the source divides the sensor field into cells; each has an area α^2. There are $n = \frac{N\alpha^2}{A}$ sensor nodes in each cell and $\sqrt{n}$ sensor nodes on each side of a cell. Each sink traverses m cells, and m is upper bounded by $1 + \frac{vT}{\alpha}$. For stationary sinks, $m = 1$.

3.2 Communication Overhead

We first analyze the worst-case communication overhead of TTDD and SODD. We assume in both TTDD and SODD a sink updates its location m times, and receives $\frac{d}{m}$ data packets between two consecutive location updates. In TTDD, a sink updates its location by flooding a query locally to reach an immediate dissemination node, from which the query is further forwarded to the source along the grid. The overhead for the query to reach the source, without considering query aggregation, is:

$$nl + \sqrt{2}(c\sqrt{N})l$$

where nl is the local flooding overhead, and $c\sqrt{N}$ is the average number of sensor nodes along the straight-line path from the source to the sink ($0 < c \leq \sqrt{2}$). Because a query in TTDD traverses a grid instead of straight-line path, the worst-case path length is increased by a factor of $\sqrt{2}$.

Similarly the overhead to deliver $\frac{d}{m}$ data packets from a source to a sink is $\sqrt{2}(c\sqrt{N})\frac{d}{m}$. For k mobile sinks, the overhead to receive d packets in m cells is:

$$\begin{aligned} & km \cdot \left(nl + \sqrt{2}(c\sqrt{N})l + \sqrt{2}(c\sqrt{N})\frac{d}{m}\right) \\ & = kmnl + kc\,(ml + d)\,\sqrt{2N} \end{aligned}$$

Plus the overhead Nl in updating the mission of the sensor network and $\frac{4N}{\sqrt{n}}l$ in constructing the grid, the total overhead of TTDD becomes:

$$CO_{TTDD} = Nl + \frac{4N}{\sqrt{n}}l + kmnl + kc\,(ml + d)\,\sqrt{2N} \qquad (1)$$

In SODD, every time a sink floods the whole network, it receives $\frac{d}{m}$ data packets. Data traverse straight-line path(s) to the sink. Again, without considering aggregation, the communication overhead is:

$$Nl + (c\sqrt{N})\frac{d}{m}$$

For k mobile sinks, the total worst-case overhead is:

$$\begin{aligned} CO_{SODD} &= k \cdot m \cdot \left(Nl + \left(c\sqrt{N}\right)\frac{d}{m}\right) \\ &= kmNl + kcd\sqrt{N} \end{aligned}$$

Note that here we do not count the overhead to update the sensor network mission because SODD can potentially update the mission when a sink floods its queries.

To compare TTDD and SODD, we have:

$$\frac{CO_{TTDD}}{CO_{SODD}} \approx \frac{1}{mk}\left(1 + \frac{4}{\sqrt{n}}\right) \qquad N \gg n, \left(\frac{d}{m}\right)^2$$

Thus, in a large-scale sensor network, TTDD has *asymptotically lower* worst-case communication overhead compared

node either through MAC layer mechanisms such as acknowledgments when available, or explicitly soliciting a reply if it does not overhear the dissemination node.

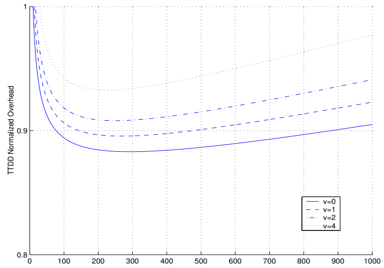

Figure 6: TTDD overhead v.s. cell size

with an SODD approach as the sensor network scale (N), the number of sinks (k), or the sink mobility (characterized by m) increases.

For example, a sensor network consists of $N = 10,000$ sensor nodes, there are $n = 100$ sensor nodes in a TTDD grid cell. Suppose $c = 1$ and $l = 1$, to deliver $d = 100$ data packets:

$$\frac{CO_{TTDD}}{CO_{SODD}} = \frac{0.024m + 1.4\frac{1}{k} + 1.414}{m+1}$$

For the stationary sink case, $m = 1$ and suppose we have four sinks $k = 4$, $\frac{CO_{TTDD}}{CO_{SODD}} = 0.89$. When the sink mobility increases, $\frac{CO_{TTDD}}{CO_{SODD}} \to 0.024$, as $m \to \infty$. In this network setup, TTDD has consistently lower overhead compared with SODD in both the stationary and mobile sink scenario.

Equation (1) shows the impact of the number of sensor nodes in a cell (n) on TTDD's communication overhead. For the example above, Figure 6 shows the TTDD communication overhead as a function of n under different sink moving speed. Because the overhead to build the grid decreases while the local query flooding overhead increases as the cell size increases, Figure 6 shows the total communication overhead as a tradeoff between these two competing components. We can also see from the Figure 6 that the overall overhead is lower with smaller cells when the sink mobility is significant. The reason is that high sink mobility leads to frequent in-cell flooding, and smaller cell size limits the flooding overhead.

3.3 State Complexity

In TTDD, only *dissemination nodes* and their neighbors which duplicate upstream information, sinks' *primary agents* and *immediate agents* maintain states for data dissemination. All other sensor nodes do not need to maintain any state. The state complexities at different sensor nodes are analyzed as follows:

Dissemination nodes There are totally $\left(\sqrt{N/n}+1\right)^2$ dissemination nodes in a grid, each maintains the location of its upstream dissemination node for query forwarding. For those on data forwarding paths, each maintains locations of at most all the other three neighboring dissemination nodes for data forwarding. The state complexity for a dissemination node is thus $O(1)$. A dissemination node's neighbor that duplicate upstream dissemination node's location also has $O(1)$ state complexity.

Immediate dissemination nodes A dissemination node maintains states about the primary agents for all the sinks within a local cell-size area. Assume there are k_{local} sinks within the area, the state complexity for an immediate dissemination node is thus $O(k_{local})$.

Primary and immediate agents A primary agent maintains its sink's immediate agent's location, and an immediate agent maintains its sink's information for trajectory forwarding. Their state complexities are both $O(1)$.

Sources A source maintains states of its grid size, and locations of its downstream dissemination nodes that request data. It has a state complexity of $O(1)$.

We consider data forwarding from s sources to k mobile sinks. Assume in SODD the total number of sensor nodes on data forwarding paths from a source to all sinks is P, then the number of sensor nodes in TTDD's grid forwarding paths is at most $\sqrt{2}P$. The total number of states maintained for trajectory forwarding in sinks' immediate dissemination nodes, primary agents, and immediate agents are $k(s+2)$. The total state complexity is:

$$s \cdot \left(b \left(\sqrt{\frac{N}{n}} + 1 \right)^2 + 3 \cdot \sqrt{2} \frac{P}{\sqrt{n}} \right) + k(s+2)$$

where b is the number of sensor nodes around a dissemination point that has the location of the upstream dissemination node, a small constant.

In SODD, each sensor node maintains a state to its upstream sensor node toward the source. In the scenario of multiple sources, assuming perfect data aggregation, a sensor node maintains at most per-neighbor states. For those sensor nodes on forwarding paths, due to the query aggregation, they maintain at most per-neighbor states to direct data in the presence of multiple sinks. The state complexity for the whole sensor network is:

$$(D-1) \cdot N + (D-1) \cdot P$$

The ratio of TTDD and SODD state complexity is:

$$\frac{S_{TTDD}}{S_{SODD}} \to \frac{sb}{n(D-1)} \qquad (\text{as } N \to \infty)$$

That is, for large-scale sensor networks, TTDD maintains around only $\frac{sb}{n(D-1)}$ of the states as an SODD approach. For the example of Figure 1 where we have 2 sources and 3 sinks, suppose $b = 5$ and there are 100 sensor nodes within a TTDD grid cell and each sensor node has 10 neighbors on average, TTDD maintains only 1.1% of the states of that of SODD. This is because in TTDD sensor nodes that are out of the grid forwarding infrastructure generally do not maintain any state for data dissemination.

3.4 Summary

In this section, we analyze the worst-case communication overhead, and the state complexity of TTDD. Compared with an SODD approach, TTDD has asymptotically lower worst-case communication overhead as the sensor network size, the number of sinks, or the moving speed of a sink increases. TTDD has a lower state complexity, since sensor nodes that are not in the grid infrastructure do not need to maintain states for data dissemination. For a sensor node that is part of the grid infrastructure, its state complexity is bounded and independent of the sensor network size or the number of sources and sinks.

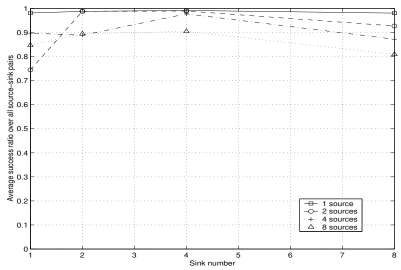

Figure 7: Success rate for different numbers of sinks and sources

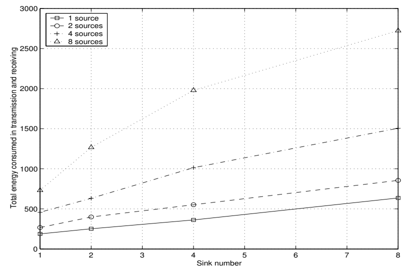

Figure 8: Energy for different numbers of sinks and sources

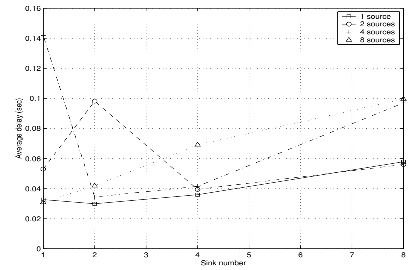

Figure 9: Delay for different numbers of sinks and sources

4. PERFORMANCE EVALUATION

In this section, we evaluate the performance of TTDD through simulations. We first describe our simulator implementation, simulation metrics and methodology in Section 4.1. Then we evaluate how environmental factors and control parameters affect the performance of TTDD in Sections 4.2 to 4.5. The results confirm the efficiency and scalability of TTDD to deliver data from multiple sources to multiple, mobile sinks. Section 4.6 shows that TTDD has comparable performance with Directed Diffusion [10] in stationary sink scenarios.

4.1 Metrics and Methodology

We implement the TTDD protocol in *ns-2*. We use the basic greedy geographical forwarding with local flooding to bypass dead ends [6]. To facilitate comparisons with Directed Diffusion, we use the same energy model as adopted in its implementation in *ns-2.1b8a*, and the underlying MAC is 802.11 DCF. A sensor node's transmitting, receiving and idling power consumption rates are 0.66W, 0.395W and 0.035W respectively.

We use three metrics to evaluate the performance of TTDD. The **energy consumption** is defined as the communication (transmitting and receiving) energy the network consumes; the idle energy is not counted since it depends largely on the data generation interval and does not indicate the efficiency of data delivery. The **success rate** is the ratio of the number of successfully received reports at a sink to the total number of reports generated by a source, averaged over all source-sink pairs. This metric shows how effective the data delivery is. The **delay** is defined as the average time between the moment a source transmits a packet and the moment a sink receives the packet, also averaged over all source-sink pairs. This metric indicates the freshness of data packets.

The default simulation setting has 4 sinks and 200 sensor nodes randomly distributed in a $2000 \times 2000\text{m}^2$ field, of which 4 nodes are sources. Each simulation run lasts for 200 seconds, and each result is averaged over three random network topologies. A source generates one data packet per second. Sinks' mobility follows the standard random Waypoint model. Each query packet has 36 bytes and each data packet has 64 bytes. Cell size α is set to 600 meters and a sink's local query flooding range is set to 1.3α; it is larger than α to handle irregular dissemination node distributions.

4.2 Impact of the numbers of sinks and sources

We first study the impact of the number of sinks and sources on TTDD's performance. The number of sinks and sources varies from 1, 2, 4 to 8. Sinks have a maximum speed of 10m/s, with a 5-second pause time. Figure 7 shows the success rates. Each curve is for a specific number of sources. For each curve, given the fixed number of sources, the success rate slightly decreases as the number of sinks increases. For example, in the 2-source case, success rate decreases slightly from 0.98 to 0.92 when the number of sinks reaches 8. For a specific number of sinks, the success rate decreases more visibly as the number of source increases. In the 8-sink case, the success rate decreases from close to 1.0 to about 0.8 as the sources increase to 8. This is because more sources generate more data packets, which lead to more contention-induced losses. Despite some fluctuations, the reasonably high success rates show that TTDD delivers most data packets successfully from multiple sources to multiple, mobile sinks, and the delivery quality does not degrade much as the number of sources or sinks increases.

Figure 8 shows the energy consumption. We make two observations. First, for each curve, the energy increases as the number of sinks increases, but the slope tends to decrease slightly. As the number of sinks doubles, the energy consumed for queries at the lower-tier flooding typically doubles. However, in the higher-tier grid forwarding, queries may be merged into one upstream update message toward the source and thus lead to energy savings. Therefore, when the number of sinks doubles, total energy consumption does not double. This is why the slope tends to decrease. Second, for a specific number of sinks (e.g., 4 sinks), energy consumption typically doubles as the number of sources doubles. This is because the total data packets generated by the sources increase proportionally and result in proportional increase in energy consumptions. An exception happens when the number of sources increases from one to two. This is because the lower-tier query flooding remains fixed as the number of sources increases, but it contributes a large portion of the total energy consumption in the 1-source case.

Figure 9 plots the delay metric, which tends to increase when there are more sinks or sources. More sources generate more data packets, and more sinks need more local query flooding. Both increase the traffic volume and lead to longer delivery time. Some exception points of the figure occur because data packets that have been cached in a sink's immediate dissemination node for some time are included in the delay calculation.

4.3 Impact of Sinks' Mobility

We next evaluate the impact of the sink's moving speed on TTDD. In the default simulation setting, we vary the maximum speed of sinks from 0, 5, 10, to 20m/s.

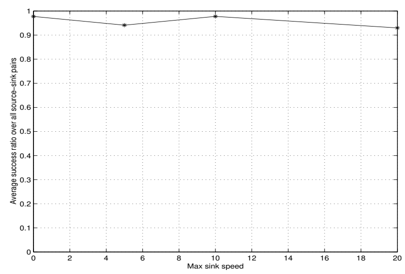

Figure 10: Success rate for sinks' mobility

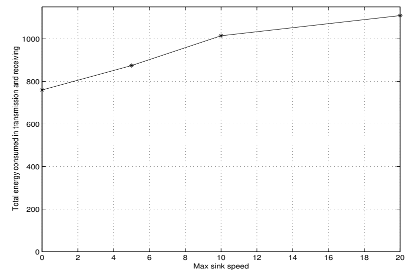

Figure 11: Energy for sinks' mobility

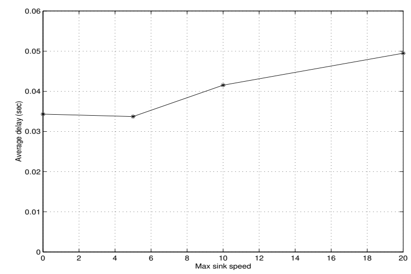

Figure 12: Delay for sinks' mobility

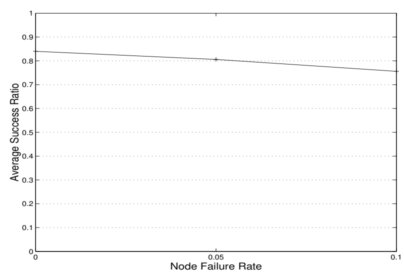

Figure 13: Success rate for sensor node failures

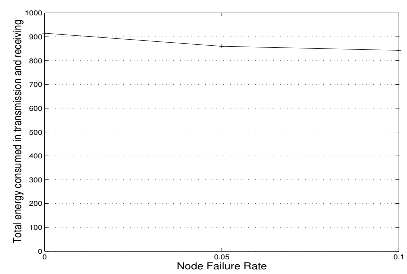

Figure 14: Energy for sensor node failures

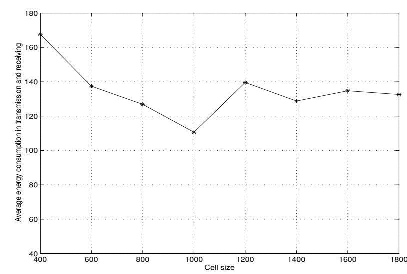

Figure 15: Energy consumption with different cell sizes

Figure 10 shows the success rate as the sinks' moving speed changes. The success rate remains around $0.9 - 1.0$ as sinks move faster. This shows that trajectory forwarding is able to deliver data to mobile sinks without much interruption even when sinks move at very high speed of 20m/s.

Figure 11 shows that the energy consumption increases as the sinks' moving speed increases. The faster a sink moves, the more frequently the sink needs a new immediate dissemination node, and the more frequently the sink floods its local queries to discover it. However, the slope of the curve decreases since the higher-tier grid forwarding is much less affected by the mobility speed. Figure 12 plots the delay for data delivery, which increases only slightly as the sink moves faster. This shows that high-tier grid forwarding effectively localizes the impact of sink mobility.

4.4 Resilience to Sensor Node Failures

We further study how node failures affect TTDD. In the default simulation setting of 200 nodes, we allow 5% or 10% randomly-chosen nodes to experience sudden, simultaneous failures at $t = 20s$. The detailed study of simulation traces shows that under such scenarios, some dissemination nodes on the grid fail. Without any repairing, failure of such dissemination nodes would have stopped data delivery to all downstream sinks and decreased the success ratio substantially. However, Figure 13 shows that the success rate drops mildly. This confirms that our grid maintenance mechanism of Section 2.3 works effectively to reduce the impact of node failures. As node failure becomes more severe, energy consumption also decreases due to reduced data packet delivery. This is shown in Figure 14. Overall, TTDD is quite resilient to node failures in all simulated scenarios.

4.5 Cell Size α

We have explored the impact of various environmental factors in previous sections. In this section we evaluate how cell size α affects TTDD. To extend the cell size to larger values while still having enough number of cells in the simulated sensor field, we would have to simulate over 2000 sensor nodes if the node density remains the same. Given the computing power available to us to run *ns-2*, we have to reduce the node density in order to reduce the total number of simulated sensor nodes. We use 960 sensor nodes in a $6200x6200m^2$ field, which are spaced at 200m distances regularly to make the simple, greedy geographical forwarding algorithm still work. There are one source and one sink. The cell size varies from 400m to 1800m with an incremental step of 200m. Because of the regular node placement, the success rate and the delay do not change much so we focus on studying the energy consumption trend.

Figure 15 shows that the energy consumption evolves the same as predicted in our analysis of Section 3. The energy first decreases as the cell size increases because it takes less energy to build a grid with larger cell size. Once the cell size increases to 1000m, however, the energy starts to increase. This is because the local query flooding consumes more energy in large cells. It degrades to global flooding if only one big cell exists in the entire sensor network.

4.6 Comparison with Directed Diffusion

In this section we compare the performance of TTDD and Directed Diffusion in the scenario of stationary sinks. We apply the same scenarios of Section 4.2 (except that sinks are stationary now) on both TTDD and Directed Diffusion to study the impact of different numbers of sinks and sources. All simulations have 200 sensor nodes in a $2000 \times 2000\text{m}^2$ field. The simulation results are shown in Figures 16–21.

We first look at success rates, shown in Figures 16 and 19. TTDD and Directed Diffusion have similar success rates, ranging between 0.8 and 1.0, except that Directed Diffusion

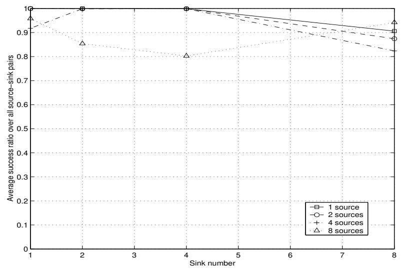

Figure 16: Success rate for TTDD of stationary sinks

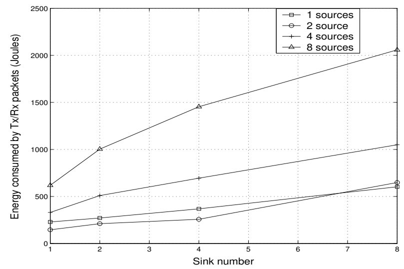

Figure 17: Energy for TTDD of stationary sinks

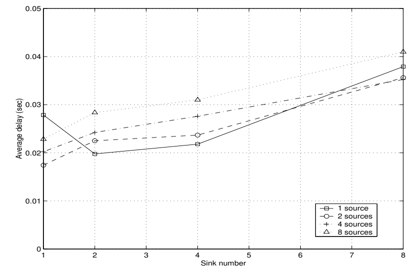

Figure 18: Delay for TTDD of stationary sinks

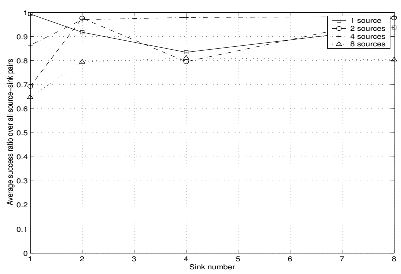

Figure 19: Success rate for Directed Diffusion

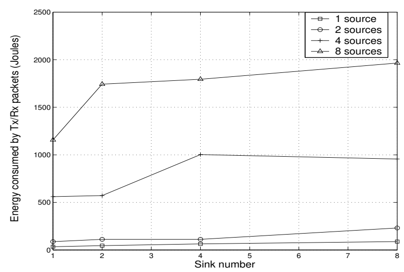

Figure 20: Energy for Directed Diffusion

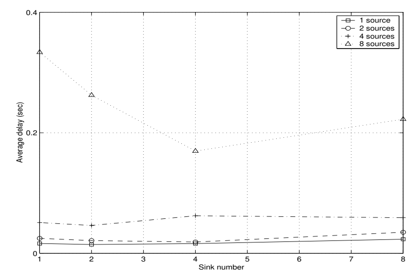

Figure 21: Delay for Directed Diffusion

has slightly larger fluctuations in some cases.

Figures 17 and 20 plot the energy consumption for TTDD and Directed Diffusion. For the same number of sinks, energy consumption nearly doubles as the number of sources doubles (except for the case of 1 to 2 sources). Given a specific number of sources, more energy is consumed as the number of sinks increases, but the slope of each curve tends to decrease. This shows that both TTDD and Directed Diffusion scale similarly to the number of sources and stationary sinks in terms of energy consumption. When the number of sinks is small (1 or 2 sinks), TTDD consumes less energy than Directed Diffusion. This is because query flooding in TTDD is confined to a local cell, while in Directed Diffusion a query propagates throughout the network field. For larger number of sinks (say, 8 sinks), Directed Diffusion aggregates queries from different sinks more aggressively; therefore, its energy consumption increases less rapidly.

Figures 18 and 21 plot the delay experienced by TTDD and Directed Diffusion, respectively. When the number of sources is 1, 2, or 4, they have comparable delay values. When the number of sources increases to 8, TTDD experiences much lower delay. This is because in Directed Diffusion data forwarding paths from different sources may cross or overlap with each other anywhere, thus there are more interferences when the number of sources is large. Whereas in TTDD each source has its own grid, thus data flows on different grids do not interfere each other that much.

5. DISCUSSIONS

In this section, we comment on several design issues and discuss future work.

Knowledge of the cell size Sensor nodes need to know the cell size α so as to build grids once they become sources. The knowledge of α can be specified through some external mechanism. One option is to include it in the mission statement message, which notifies each sensor the sensing task. The mission statement message is flooded to each sensor at the beginning of the network operation or during a mission update phase. The sink also needs α to specify the maximum distance a query should be flooded. It can obtain α from its neighbor. To deal with irregular local topology where dissemination nodes may fall beyond a fixed flooding scope, the sink may apply expanded ring search to reach the dissemination node.

Greedy geographical routing failures Greedy geographical forwarding may fail in scenarios where the greedy path does not exist, that is, a path requires temporarily forwarding the packet away from the destination. This problem has been solved by several approaches such as GPSR [11]. However, due to the the complexity of the complete solutions and the fact the greedy path almost always exists in a densely deployed sensor network [12], we only use a very basic version of greedy forwarding. In the rare cases where the greedy path does not exist, that is, the packet is forwarded to a sensor node without a neighbor that is closer to the destination, the node locally floods the packets to get around the dead end [6]. We find out that this simple technique works quite well in TTDD.

Mobile stimulus TTDD focuses on handling mobile sinks. In the scenario of a *mobile* stimulus, the sources along the stimulus' trail may each build a grid. To reduce the overhead of frequent grid construction, a source can reuse the grid already built by other sources. Specifically, before a source starts to build its grid, it locally floods a "Grid Discovery" message within the scope of about a cell size to probe any existing grid for the same stimulus. A dissemination node on the existing grid replies to the new source. The source can then use the existing grid for its data dissemination. We leave this as future work.

Non-uniform grid layout So far we assume no *a pri-*

ori knowledge on sink locations. For this case, a uniform grid is constructed to distribute the forwarding states as evenly as possible. However, this even distribution has a drawback of incurring certain amount of resource waste in regions where sinks never roam into. This problem can be partially addressed through learning or predicting the sinks' locations. If the sinks' locations are available, TTDD can be further optimized to build a globally non-uniform grid where the grid only exists in regions where sinks currently reside or are about to move into. The accuracy in estimation of the current locations or prediction of the future locations of sinks will affect the performance. We intend to further explore this aspect in the future.

Mobile sensor node This paper considers a sensor network that consists of stationary sensor nodes only. It is possible to extend this design to work with sensor nodes of low mobility. However, the grid states may be handed over between mobile dissemination nodes. Fully addressing data dissemination in highly mobile sensor network needs new mechanisms and is beyond the scope of this paper.

Sink mobility speed TTDD addresses sink mobility by localizing the mobility impact on data dissemination within a single cell and handling the intra-cell mobility through trajectory forwarding. However, there is also a limit for our approach to accommodate sink mobility. The sink cannot move faster than the local forwarding states are updated (within a cell size). The two-tier forwarding is best suited to deal with "localized" mobility patterns, in which a sink does not change its primary agent frequently.

Data aggregation We assume a group of nodes that detect an object or an event of interests can collaboratively process the sensing data and only one node generates a report as a source. Although TTDD benefits further from en-route semantic data aggregation [10], we do not evaluate this performance gain since it is highly dependent on the specific applications and their semantics.

6. RELATED WORK

Sensor networks have been a very active research field in recent years. Energy-efficient data dissemination is among the first set of research issues being addressed. SPIN [7] is one of the early work that focuses on efficient dissemination of an individual sensor's observations to *all* the sensors in a network. SPIN uses meta-data negotiation to eliminate the transmission of redundant data. More recent work includes Directed Diffusion [10], Declarative Routing Protocol (DRP) [5] and GRAB [20]. Directed Diffusion and DRP are similar in that they both take the *data-centric* naming approach to enable in-network data aggregation. Directed Diffusion employs the techniques of initial low-rate data flooding and gradual reinforcement of better paths to accommodate certain levels of network and sink dynamics. GRAB targets at robust data delivery in an extremely large sensor network made of highly unreliable nodes. It uses a forwarding *mesh* instead of a single path, where the mesh's width can be adjusted on the fly for each data packet.

While such previous work addresses the issue of delivering data to stationary or very low-mobility sinks, TTDD design targets at efficient data dissemination to multiple, both stationary and *mobile* sinks in large sensor networks. TTDD differs from the previous work in three fundamental ways. First of all, TTDD demonstrates the feasibility and benefits of building a virtual grid structure to support efficient data dissemination in large-scale sensor fields. A grid structure keeps forwarding states only in the nodes around dissemination points, and only the nodes between adjacent grid points forward queries and data. Depending on the chosen cell size, the number of nodes that keep states or forward messages can be a small fraction of the total number of sensors in the field. Second, this grid structure enables mobile sinks to continuously receive data on the move by flooding queries within a local cell only. Such local floodings minimize the overall network load and the amount of energy needed to maintain data-forwarding paths. Third, TTDD design incorporates efforts from both sources and sinks to accomplish efficient data delivery to mobile sinks; sources in TTDD proactively build the grid structure to enable mobile sinks learning and receiving sensed data quickly and efficiently.

Rumor routing [3] avoids flooding of either queries or data. A source sends out "agents" which randomly walk in the sensor network to set up event paths. Queries also randomly walk in the sensor field until they meet an event path. Although this approach shares a similar idea of making data sources play more active roles, rumor routing does not handle mobile sinks. GEAR [21] makes use of geographical location information to route queries to specific regions of a sensor field. If the regions of data sources are known, this scheme provides energy savings over network flooding approaches by limiting the flooding to a geographical region, however it does not handle the case where the destination location is not known in advance.

TTDD also bears certain similarity to the study on self-configuring ad hoc wireless networks. GAF [19] proposes to build a geographical grid to *turn off nodes* for energy conservation. The GAF grid is pre-defined and synchronized in the whole sensor field, with the cell size determined by the communication range of nodes' radios. The TTDD grid differs from that of GAF in that the former is constructed dynamically as needed by data sources, and we use it for a different purpose of limiting the impact of sink mobility.

There is a rich literature on mobile ad hoc network clustering algorithms [2, 13, 14, 16]. Although they seem to share similar approaches of building virtual infrastructures for scalable and efficient routing, TTDD targets at communication that is data-oriented, not that based on underlying network addressing schemes. Moreover, TTDD builds the grid structure over stationary sensor nodes using location information, which leads to very low overhead in the construction and maintenance of the infrastructure. In contrast, node mobility in a mobile ad hoc network leads to significantly higher cost in building and maintaining virtual infrastructures, thus offsetting the benefits.

Perhaps TTDD can be most clearly described by contrasting its design with that of DVMRP [17]. DVMRP supports data delivery from multiple sources to multiple receivers and faces the same challenge as TTDD, that is how to make all the sources and sinks meet without *a prior* knowledge about the locations of either. DVMRP solves the problem by letting each source flood data periodically over the entire network so that all the interested receivers can grasp on the multicast tree along the paths data packets come from. Such a source flooding approach handles sink mobility well but at a very high cost. TTDD inherits the source proactive approach with a substantially reduced cost. In TTDD a data source informs only a small set of sensors of its existence by propagating the information over a grid structure instead of notifying all the sensors. Instead of sending data over the grid TTDD simply stores the source information; data stream is delivered downward specific grid branch or branches only upon receiving queries from one or more sinks down that direction or directions.

7. CONCLUSION

In this paper we described TTDD, a two-tier data dissemination design, to enable efficient data dissemination in large-scale wireless sensor networks with sink mobility. Instead of passively waiting for queries from sinks, TTDD exploits the property of sensors being stationary and location-aware to let each data source build and maintain a grid structure in an efficient way. Sources proactively propagates the *existence* information of sensing data globally over the grid structure, so that each sink's query flooding is confined within a local gird cell only. Queries are forwarded upstream to data sources along specific grid branches, pulling sensing data downstream toward each sink. Our analysis and extensive simulations have confirmed the effectiveness and efficiency of the proposed design, demonstrating the feasibility and benefits of building an infrastructure in stationary sensor networks.

8. ACKNOWLEDGMENT

We thank our group members of Wireless Networking Group (WiNG) and Internet Research Lab (IRL) at UCLA for their help during the development of this project and their invaluable comments on many rounds of earlier drafts. Special thanks to Gary Zhong for his help on the *ns-2* simulator, and Jesse Cheng for his verification on the simulation results. We would also like to thank the anonymous reviewers for their constructive criticisms.

9. REFERENCES

[1] J. Albowitz, A. Chen, and L. Zhang. Recursive Position Estimation in Sensor Networks. *ICNP'01*, 2001.

[2] S. Basagni. Distributed Clustering for Ad Hoc Networks. *International Symposium on Parallel Architectures, Algorithms, and Networks (I-SPAN'99)*, 1999.

[3] D. Braginsky and D. Estrin. Rumor Routing Algorithm for Sensor Networks. *Submission to International Conference on Distributed Computing Systems (ICDCS-22)*, 2002.

[4] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao. Application Driver for Wireless Communications Technology. *ACM SIGCOMM Workshop on Data Communication in Latin America and Caribbean*, 2001.

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

[6] G. Finn. Routing and Addressing Problems in Large Metropolitan-scale Internetworks. Technical Report ISI/RR-87-180, Information Sciences Institute, March 1987.

[7] W. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive Protocols for Information Dissemination in Wireless Sensor Networks. *ACM International Conference on Mobile Computing and Networking (MOBICOM'99)*, 1999.

[8] J. Hightower and G. Borriello. Location Systems for Ubiquitous Computing. *IEEE Computer Magazine*, 34(8):57–66, 2001.

[9] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System Architecture Directions for Networked Sensors. *International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-IX)*, 2000.

[10] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. *ACM International Conference on Mobile Computing and Networking (MOBICOM'00)*, 2000.

[11] B. Karp and H. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. *ACM International Conference on Mobile Computing and Networking (MOBICOM'00)*, 2000.

[12] J. Li, J. Jannotti, D. D. Couto, D. R. Karger, and R. Morris. A Scalable Location Service for Geographic Ad Hoc Routing. *ACM International Conference on Mobile Computing and Networking (MOBICOM'00)*, 2000.

[13] C. Lin and M. Gerla. Adaptive Clustering for Mobile Wireless Networks. *IEEE Journal on Selected Areas in Communications*, 15(7):1265–1275, 1997.

[14] A. B. McDonald. A Mobility-Based Framework for Adaptive Clustering in Wireless Ad-Hoc Networks. *IEEE Journal on Selected Areas in Communications*, 17(8), 1999.

[15] G. Pottie and W. Kaiser. Wireless Integrated Network Sensors. *Communications of the ACM*, 43(5):51–8, May 2000.

[16] R. Sivakumar, P. Sinha, and V. Bharghavan. CEDAR: Core Extraction Distributed Ad hoc Routing. *IEEE Journal on Selected Areas in Communications, Special Issue on Ad hoc Networks*, 17(8), 1999.

[17] D. Waitzman, C. Partridge, and S. Deering. Distance Vector Multicast Routing Protocol. *RFC 1075*, 1988.

[18] A. Ward, A. Jones, and A. Hopper. A New Location Technique for the Active Office. *IEEE Personal Communications*, 4(5):42–47, 1997.

[19] Y. Xu, J. Heidemann, and D. Estrin. Geography Informed Energy Conservation for Ad Hoc Routing. *ACM International Conference on Mobile Computing and Networking (MOBICOM'01)*, 2001.

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

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