Fault-Tolerant Manycast to Mobile Destinations in Sensor Networks

Xianjin Zhu, Himanshu Gupta
Department of Computer Science, Stony Brook University, NY
Email: `xjzhu,hgupta@cs.sunysb.edu`

Abstract

Manycast is a group communication primitive wherein the source is required to send data packets to a certain number of a given set of destinations. In this article, we design fault-tolerant protocols for manycast operations in sensor networks with mobile destinations. To develop efficient protocols, we propose a location management scheme, which manages information about the locations of mobile destinations in a distributed manner. Based upon that, we develop rectangle-based and GridTree-based fault-tolerant manycast routing protocols. Simulation results show that GridTree approach achieves sufficiently high success ratio, while using minimal transmission cost.

1. Introduction

Sensor networks are ad hoc multihop wireless networks formed by a large number of resource constrained sensor nodes, equipped with short range radios, limited processing capacity and battery. In this article, we address the problem of fault-tolerant manycast operation in sensor networks with mobile destination nodes. The *manycast* operation is a group communication primitive where the source node is required to send a data packet to a certain number (or percentage) of a given set of destinations. When the sensor network is prone to link and/or node failures, providing robust manycast operation is a challenge. Our problem is motivated by applications that require alerts to be transmitted to a set of mobile destinations. For instance, in a battlefield sensor network, imminent threats need to be transmitted to a certain number of mobile soldiers/vehicles.

In our model of the problem, the network consists of source nodes and mobile destination nodes. For the most part (with generalizations discussed in Section 6), we assume that there is a single static source and multiple mobile destinations, with all the other nodes in the network being static. We assume that each node is aware of its location (either through GPS [6] or other localization techniques [2]). At the core of our developed techniques is a distributed location management scheme for the destinations. We divide the entire network region into grids, and store location information of the destinations at certain nodes in each grid. Based on the above scheme, the source node transmits the required packets to carefully chosen grids. In our rectangle-based approach, the source selects an optimal rectangular region to deliver packets, while in the more efficient GridTree approach, the source delivers packets using an appropriately constructed geometric tree over the grids. To the best of our knowledge, ours is the first article to address the problem of fault-tolerant manycast in sensor networks over mobile destinations.

Paper Organization. We start with the problem formulation, motivation, and a discussion on related work in Section 2. In Section 3, we describe the grid-based location management mechanism. Rectangle-based and GridTree approaches are described in Section 4 and Section 5 respectively. Generalization of our techniques to handle multiple and mobile source nodes is discussed in Section 6. We present performance results in Section 7, and end with concluding remarks in Section 8.

2. Problem Formulation and Related Work

In this section, we start with formally defining the addressed problem. Then, we present some motivating applications, and discuss the related works.

Problem Formulation. Consider a sensor network, where each node is aware of its own location. The network contains one static source node S, and a set D of mobile destination nodes. S is required to perform a manycast operation to D, i.e., S is required to send data packets to a certain number k of destinations, where $k < |D|$. The number k is not fixed and may be different for different manycast requests. The source will typically perform multiple (possibly, periodically) such manycast requests. Our goal is to design a fault-tolerant (in face of link/node failures) manycast protocol that will incur minimum communication cost.

Motivating Example. A concrete example that motivates such manycast operation can be a sensor network deployed in a battlefield to detect threats and send alerts to mobile soldiers and/or vehicles. The source may be any sensor node that detects a threat. As a more specific scenario, consider a sensor network with a special purpose source node S connected to a control command center. S is required to transmit certain alerts to the mobile soldiers/vehicles in the monitored region. In general, it is not necessary to reach all the soldiers, and the number of destinations required to reach depends on the type (criticality) of alerts. Moreover, due to the mobility of destinations, it is very inefficient to know the locations of (and hence, reach) all destinations at any given time.

Related Works. Although there has been a lot of work done on group communication in ad hoc networks, manycast problem has attracted researchers' attention only recently. *Multicast* and *anycast* operations are special cases of manycast. In multicast, the source is required to reach all destinations, while in anycast, the source is required to reach any one destination. Various tree-based [17] as well as mesh-based [10, 13] approaches have been proposed to implement multicast. Manycast operation has also been addressed [7] for mobile destinations (i.e., multiple mobile sinks). However, multicast can be a very inefficient way to implementing manycast, especially if the percentage of destinations required to reach in the given manycast operation is low. Also note that manycast cannot be implemented as multiple anycasts [14], since there is no way to guarantee that destinations reached in multiple anycasts are distinct. To the best of our knowledge, [3] is the only work in ad hoc networks that advocates providing support for manycast at the network layer. However, their suggested approach is built upon underlying routing protocols used in ad hoc networks, and hence, are not directly applicable to sensor networks which typically use localized location-based routing protocols.

In many applications, spatial proximity of destinations can be exploited to develop more efficient protocols with the aid of location information. In particular, in *Geocast*, the group of destinations is implicitly defined by a specific geographic region [8, 9, 12]. However, our manycast problem is very different from the geocast problem. In our setting, the destinations are specific mobile nodes not *defined* by their geographic location. The destinations in our manycast problem may even be distributed over the entire network.

Most works support fault-tolerant unicast routing either by interleaved mesh approach (e.g. GRAB [18]) or by multipath approach ([4, 19]). One advantage of GRAB is that nodes do not store routing information explicitly, and hence, the storage requirement at each node is minimal. However, such benefits do not exist when the technique is applied directly to our manycast problem, since every node will need to store cost for each destination, which requires storage proportional to number of destinations and is hard to maintain.

3. Location Information Management

To implement manycast in an efficient manner, it is necessary that the information about destination locations is stored somewhere in the network. One option could be to maintain or discover routes to destinations using proactive [16] or reactive [15] routing protocols. Other option could be to store all the location information of destinations at some centralized location, or require each destination to flood the entire network with updated location information. However, all the above approaches incur very high communication cost and intolerant to faults. Yet another option could be to use one of the existing location service techniques such as GLS [11] that stores precise location information of nodes in the network. But the GLS approach would be an overkill for our problem, since we do not need to look up location information of any particular node. Rather, we are only interested in a general distribution of the destinations over the network region, as we only wish to target a certain *number* of destinations.

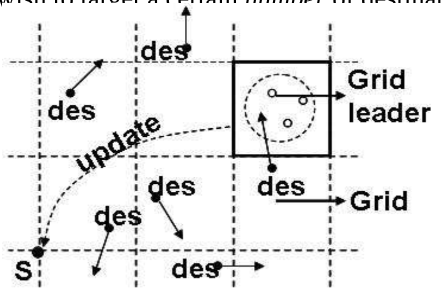


Figure 1. Destination location management scheme. On entering a grid, a destination informs grid leaders of its ID, location, and velocity (speed and direction). Leaders periodically update source about the number of destinations in their grids.

Overview. Our approach to manage the destination location information is to use a distributed virtual grid structure, wherein each grid has a certain number (depending on the desired fault-tolerance) of grid leader nodes which keep the location information of destinations in their grids (see Fig. 1). Specifically, we divide the entire sensor network region into *grids* each of size $d \times d$ for an appropriate chosen d. Since, each node is aware of its location, a node is also aware of the grid it belongs to. Within each grid, a certain number of nodes are elected to be *grid leaders*. Destinations report their locations to the grid leaders on entering a grid, and each grid leader reports to source the list of destinations in its grid.

Choosing Grid Size d. The grid size should be large enough so that the number of location updates to the source are small. On the other side, very large grids would result in inefficient routing (between grid leaders and destinations) in a grid. Moreover, large grids will also result in bottlenecks at the small number of chosen grid leaders. Note that we do not require the grid size to be smaller than the transmission radius of the nodes, and hence, routing between grid leaders of different grids may require relay through intermediate nodes.

Election of Grid Leaders. Since the destinations are randomly distributed in a grid, it is most efficient to pick the nodes that are close to the grid center as the grid leaders. Moreover, a localized grid-leader election algorithm is highly desirable. We use a simple election algorithm wherein each node I elects itself as a grid leader if at most $(l-1)$ of its neighbors are closer to the grid center than I. Here, l is appropriately chosen based on the network density and desired number of grid leaders (which determines fault tolerance and communication cost). The above election algorithm is robust to node failures, since failure of a grid leader would result in one or more of its neighbors electing themselves as new grid leaders if necessary. The above scheme results in the nodes close to the grid center being overloaded; however, this problem can be alleviated by periodically realigning (i.e., shifting the grid lines appropriately) the grids.

Notification from Destinations to Grid Leaders. On entering a new grid g, a destination notifies one or more grid leaders in g of its location and velocity. The notification is done by sending an appropriate data packet to g's center using location-based greedy routing. Note that a message routed using location-based greedy approach will always reach a node I that doesn't have any of its neighbor closer to the center than itself, and such a node I must have elected itself as a grid leader. Thus, any packet routed to the center of grid g is *guaranteed* to reach *at least one* grid leader (in absence of message losses) of g. If we assume that a destination continues to travel in a straight line for a short period of time after notifying the grid leaders, the grid leader can estimate the time when the destination would leave the grid and hence, a destination does not need to notify the grid leaders on leaving a grid.

Destination Information at Grid Leaders and Source. Based on the above notifications from the destinations, each grid leader maintains the list of destinations along with the associated velocity and notification timestamp. Since the notification message from any destination reached at least one grid leader, the union of destination lists at the grid leaders of a grid must yield the complete list of destinations in the grid. Each grid leader reports the list of destinations to the source either periodically or when the destination list changes sufficiently enough. Based on these reports, the source can estimate the number of destinations in each grid. Note that incorrect estimations by a grid leader of a destination leaving its grid can be corrected at the source using reports from leaders of neighboring grids.

Fault Tolerance of Destination Information. The above described scheme (comprised of destinations notifying at least one grid leader and the grid leaders periodically reporting the list of destinations to the source) is *guaranteed* to yield accurate location information of destinations at the source node if we assume robust message communication and straight-line movement of destinations. The fault-tolerance to message losses is provided by various aspects of our scheme, viz., notification from destinations reaching to multiple grid leaders, the source getting reports from multiple grid leaders of *each* grid, and the fact that the source tries to target slightly more number of destinations than required by the manycast operation. In addition, the reports from neighboring grids compensate for inaccuracies due to the assumption of straight-line movement of destinations.

4. Rectangle-based Manycast Protocol

In this section, we present two rectangle-based protocols for our manycast problem. The presented protocols select a rectangular region containing the source and required number of destinations, and then, use some routing scheme within the chosen rectangular region to reach destinations.

Selecting an Optimal Rectangular Region. To account for slight inaccuracy of location information, we choose a rectangular region that contains slightly more number of destinations than required by manycast operation. Since the location information is on a per-grid basis, we need to consider only those rectangles that are formed by a union of grids and there are only a polynomial (in number of grids) number of such rectangles. Thus, we can look at each feasible (containing desired number of destinations) rectangle, and pick the one that will incur minimum routing cost. Both our routing schemes (as described below) for rectangle-based approaches incur a cost proportional to the size of the rectangle (under uniform network density assumption). Thus, we choose the feasible rectangle of minimum size.

Rectangular Flooding. The simplest routing schemes to reach destinations is to use the chosen rectangular region as a forwarding zone. Each node in the forwarding zone that receives a data packet broadcasts it to all of its neighbors. Such a scheme is expected to have a very good success ratio, at the cost of high transmission cost.

GeoGrid-based Routing. To use the GeoGrid routing scheme [12] for our purposes, we lay an independent layer of grids called as *routing-grids*. The node closest to the center of each routing-grid is chosen as a "gateway" node. The size of each routing-grid is chosen to be small enough to guarantee that gateways in adjacent routing-grids can communicate directly. In the simple *flood-based GeoGrid* approach, all the gateway nodes that belong to the chosen rectangular region forward the data packet to their neighbors. In the *ticket-based GeoGrid* scheme, each gateway node g that receives the data packet also receives a ticket with a non-negative integer value T from the sender. Only if $T > 0$, g retransmits the packet and divides T into equal values for each of the gateway neighbors. Initially, the source node generates several tickets based on the size of chosen rectangle.

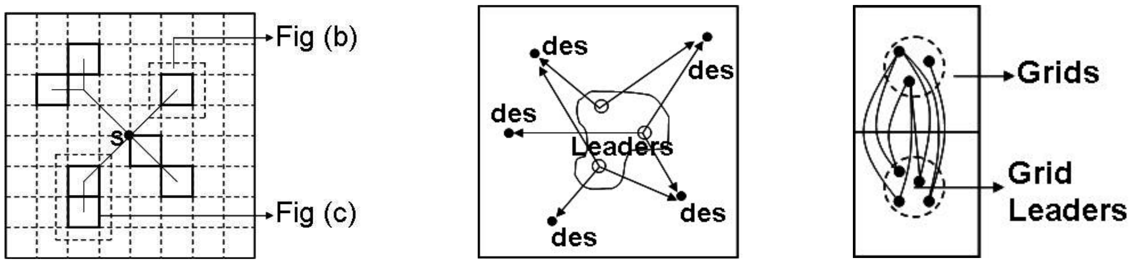
Bad Scenarios. In certain cases, the rectangle-based approach can incur a high routing cost. For instance, when all destinations are situated in the border or diagonal grids. In both the cases, the chosen rectangular region would necessarily include many grids with zero destinations. In the extreme case, the whole network field may need to be selected as the rectangular region.

5. GridTree Manycast Protocol

The disadvantage of rectangle-based approaches motivates us to develop more efficient protocols. One way to reduce routing cost is to select a set of grids called *target grids* such that the total number of destinations inside these grids is at least k, and then, use geocast routing to reach each of these target grids. However, reaching each of the target grids independently can again lead to overall inefficient transmission cost. In this section, we present *GridTree* approach (see Fig. 2) that optimizes overall transmission cost by using grid leaders as "path-gateways" and route data packets through a tree formed over grid-leaders. Simulation results show that GridTree approach achieves improved performance.

5.1. Selection of Target Grids

The selection of target grids depends on the routing scheme used, and is key to the efficiency of the overall



(a) Target Grids.

(b) Routing from grid leaders to destinations in a grid.

(c) Routing between grid leaders.

Figure 2. GridTree Approach

GridTree approach.

Definition 1 (Geometric Grid Graph) A *geometric grid graph* is a graph over grid centers as vertices. An edge exists between two vertices if their corresponding grids are adjacent to each other. The *cost* of each edge is the Euclidean distance between the grid centers in the plane, and the *weight* on each vertex is the number of destinations in the corresponding grid. □

Routing Cost to Reach Selected Target Grids. Let us assume that we have selected a set of target grids $\mathcal{S}$ that contain the required number of destinations (see Fig. 2(a)). To route to destinations in $\mathcal{S}$, the source node first constructs a minimal (actually, a 2-approximation) cost Steiner tree $\mathcal{T}(\mathcal{S})$ spanning over the grid centers of grids in $\mathcal{S}$. Each edge in $\mathcal{T}(\mathcal{S})$ connects grid centers of two adjacent grids. To reach destinations in $\mathcal{S}$, the source node routes the packet over $\mathcal{T}(\mathcal{S})$, wherein each node in $\mathcal{T}(\mathcal{S})$ is replaced by the corresponding set of grid leaders and each edge of $\mathcal{T}(\mathcal{S})$ is replaced by a set of paths connecting the set of grid leaders of adjacent grids. Once the data packet reaches the grid leaders of target grids, it is routed to the destinations in each target grid using greedy forwarding. Thus, given a set of target grids $\mathcal{S}$, the total routing cost incurred (using the above described scheme) in reaching the destinations in $\mathcal{S}$ is directly proportional to the total cost of the corresponding 2-approximation Steiner tree $\mathcal{T}(\mathcal{S})$ in the geometric grid graph. Based on the above cost model, we formulate the problem of selection of target grids as follows.

Selection of Target Grids. Given a geometric grid graph G and k (the number of destinations required to reach), select a set of vertices $\mathcal{S}$ with minimum cost $\mathcal{T}(\mathcal{S})$ such that the total weight of vertices $\mathcal{S}$ is k. The above problem is a generalization of the Steiner tree problem and NP-complete. Below, we give a 4-approximation algorithm for the target grids selection problem, based on the 2-approximation algorithm by Garg [5] for the k-MST problem [1].

<u>4-approximation Algorithm.</u> Construct a complete graph G^* over the set of vertices of the geometric grid graph G. The weight associated with an edge (s_1, s_2) in G^* is the length of the shortest weighted path between s_1 and s_2 in G. Using techniques similar to [5],[1] we can construct a tree T_k in G^* that spans over vertices with total weight at least k and total edge cost at most twice the minimum possible. Now, replace each edge (s_1, s_2) in T_k by the shortest weighted path connecting s_1 and s_2 in the geometric grid graph G to yield a subgraph G' of G. Let T' be a minimum spanning tree of G' spanning all the vertices of G'. Note that the total weight of the vertices of T' is at least k. Thus, the set of vertices of T' is a feasible solution of our original problem of selection of target grids. Let O be the optimal (minimum edge cost) tree in the geometric grid graph G such that the total weight of the vertices of O is at least k. Below, we show that the total cost of T' is at most 4 times that of O.

<u>Approximation Proof.</u> Let $\vec{O}$ be the directed graph that is obtained by replacing every undirected edge (u, v) of the optimal tree O by two directed edges (u, v) and (v, u) of the same weight as that of (u, v) in O. Consider an Euler tour $\mathcal{E}$ in the graph $\vec{O}$. Note that an Euler tour is guaranteed to exist in $\vec{O}$ since the indegree of each vertex is the same as its outdegree, and the total cost of $\mathcal{E}$ is equal to twice the cost of the O. It is easy to show that the cost of $\mathcal{E}$ is at least the optimal cost of a tree spanning vertices with at least weight k in G^*. Since, T_k is a 2-approximation of such a tree in G^*, we have $|O| = |\mathcal{E}|/2 > |T_k|/4 > |T'|/4$. Since the total routing cost of any set of vertices of total weight more than k is at least $|O|$, the set of vertices of T' is a 4-approximate solution of the target grids selection problem.

<u>Heuristic for k-Steiner Tree Problem.</u> Since the above approximation algorithm involves a rather complex approximation algorithm (involving relaxation of LP formulation and primal-dual conversion) for computing the approximate k-MST, we propose a simpler greedy heuristic keeping in mind the limited computing resources available at sensor nodes. The greedy algorithm works by growing the Steiner tree T_M over a set of already selected target grids M. Initially, M is the node in the geometric graph G corresponding to the grid containing the source, and T_M is the tree with the single node M. At each stage, we consider a new target grid s not in M based on the following benefit calculation. For each vertex s

[1]Garg looks at the unweighted k-MST problem. However, our weighted version can be reduced to the un-weighted version by making w copies of a node of weight w and connecting each pair of such copies by an edge of zero cost.

not in T_M, we compute the shortest path P_s connecting s to a node in T_M, for each node in T_M. For each such path P_s, we compute the total cost c_s of P_s and the total weight D_s of the vertices on the path. We select the target grid s with a path P_s that has the highest benefit D_s/c_s, and add the path P_s to T_M. The above algorithm continues until T_M contains vertices with total weight at least k.

5.2. Routing in GridTree Approach

In this subsection, we discuss in more detail the routing schemes used in our GridTree approach.

Routing Between Pair of Grid Leader Sets. In our approach, routing from one grid center to another in the geometric grid graph is done using fault-tolerant routing between pairs of corresponding sets of grid leaders (see Fig. 2(c)). In particular, the set of grid leaders L_1 that need to transmit a data packet to a set of grid leaders L_2 in grid g_2, can send the packet using any of the location-based routing schemes to the center of g_2. Such a strategy precludes the need to know the exact grid leaders in the network, and since the set of grid leaders is dynamic, it is not possible to build routes proactively between the grid leaders. In our simulations, we observe that location-based limited flooding scheme offers the best fault-tolerance with least amount of routing cost. For the limited-flood scheme, we use a limited region around the line segment connecting the corresponding grid centers as a forwarding region. The forwarding region is specified in an implicit manner by the grid leaders in L_1 in the data packet. In particular, a sensor node retransmits a data packet if and only if its distance from the destination grid center is at most the sum of the distance from the packet sending and a constant δ. The constant δ depends on the network density and transmission radius.

Routing from Grid Leaders to Destinations. After the grid leaders receive the data packet from the source node through the edges of geometric grid graph, they check to see if their grid is a target grid (the set of target grids is included in the packet header). If the grid is a target grid, the received leader node forwards the data packet to all the destinations within its grid based on the location information stored with the leader (see Fig. 2(b)). The current location of the destination can be predicted from the stored information of each destination. Then, the grid leader forwards the packet to the destination's location using the greedy forwarding approach, wherein each receiving node forwards the packet to the neighbor that is closest to the destination. The success ratio can be further improved by using GPSR to recover from stuck nodes.

6. Mobility and Multiple Sources

In this section, we generalize our techniques for multiple and/or mobile sensor nodes.

Multiple Static Sources. Certain applications may need multiple independent sources involved in manycast operations to the same set of mobile destinations. Our proposed techniques can be easily extended as follows to work for multiple static sources. Since the sources are static, the grid leaders can send information about number of destinations in their grids to each source independently. Based upon the collected information, each source can execute the rectangle-based or GridTree approach separately. The above approach can be made more efficient by replacing the multiple unicasts (from a grid leader to the various sources) by a multicast over an appropriately constructed tree. A more involved but efficient approach is to use a *virtual source* at the centroid of the source locations. Grid leaders can send periodic updates to the virtual source as before. Updates from grid leaders are aggregated at the virtual source, and multicast periodically to the sources using an appropriately constructed multicast tree from the virtual source to the sources.

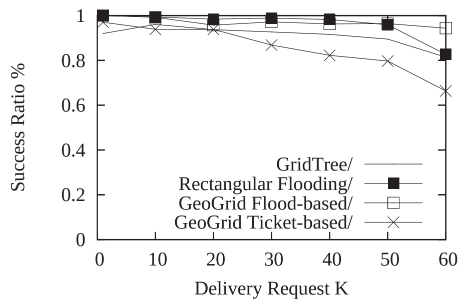
Multiple Mobile Sources. Certain applications may involve mobile sources such as an aircraft or helicopter that wishes to send alerts to sensor nodes on ground. To extend our techniques for mobile sources, we could have the sources gather updates from grid leaders on a demand basis, since it will be inefficient to have grid leaders keep tracking the locations of sources. The virtual source approach described in previous paragraph can be used here as well. In particular, the virtual source can store the destination location information sent by the grid leaders. Due to the mobility of the sources, the multicast tree from the virtual source to the sources is dynamic and hence, difficult to maintain. However, we can have the sources contact the fixed/static virtual source for destination location information whenever needed.

Mobile Sensor Nodes. Our manycast problem becomes much more challenging when the nodes (other than the sources and destinations) in the network are also mobile. In such a case, we can require each destination to periodically broadcast its ID, location, and velocity to all nodes within a certain number of hops. Thus, over a period of time each node in the network gathers some information about destination locations. A node that has gathered information about sufficient number of destinations transmits that information to the static virtual source at the center of the network. The virtual source assimilates the information received, and computes the distribution of destinations over grids. The information is distributed to the sources as in previous cases.

7. Performance Evaluation

In this section, we evaluate the performance of the various approaches developed, viz. rectangular flooding, GeoGrid (flood-based and ticket-based), and the GridTree approach. We investigate two scenarios in the following two subsections. In the first scenario, destinations are randomly distributed, while in the second scenario, they are constrained to peripheral grids (far away from source).

Simulator, Parameters, and Metrics. We constructed a simulator to evaluate the performance of our algorithms. All the messages in the simulator are transmitted with a constant probability of success, which is modeled as the *link failure*



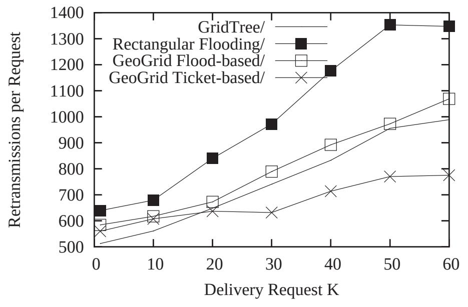
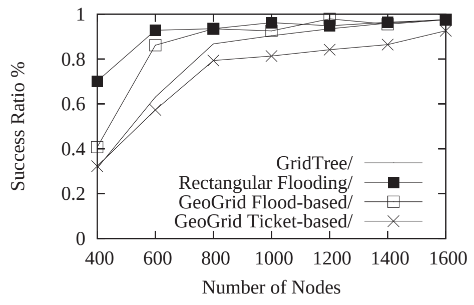


Figure 3. Effect of Delivery Request k. (a) Success Ratio. (b) Transmission Cost.



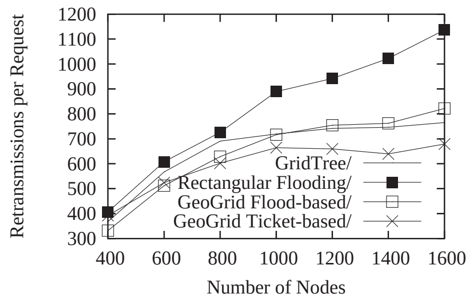
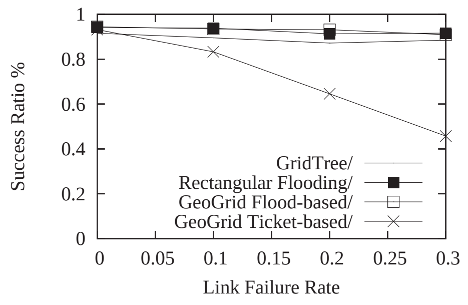


Figure 4. Effect of Network Density (a) Success Ratio. (b) Transmission Cost.



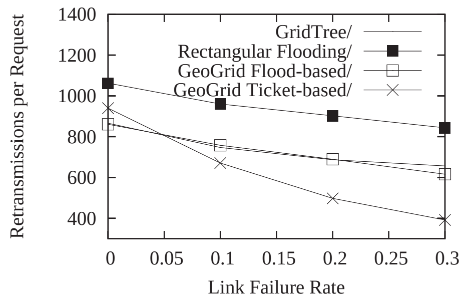
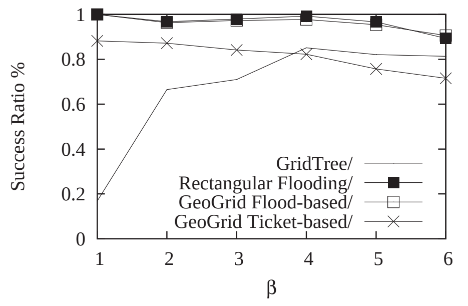


Figure 5. Effect of link failure rate (a) Success Ratio. (b) Transmission Cost.



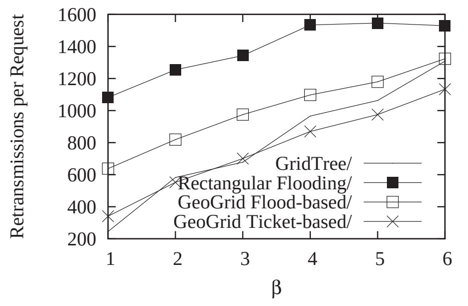


Figure 6. Effect of number ($\beta \times \beta$) of grids. (a) Success Ratio. (b) Transmission Cost.

rate parameter. While such a simulator models a uniform communication subsystem, it is sufficient for our purpose as we are only interested in counting message transmissions.

We generate a sensor network by randomly placing a certain number of nodes with transmission radius 10 units in a 150×150 units square region. The source node is fixed at the center. There are 60 destination nodes moving according to the Random Waypoint mobility model, wherein nodes randomly select a speed value V units/sec (from the range $[0, V_{max}]$) and a random point to travel to. On reaching the selected point, the node pauses for P seconds, and repeats the same process. Source initiates a manycast operation every 5 sec and each simulation is run for 100 secs per round. Simulation results are averaged over 10 rounds. A grid leader updates the location information at source if the information changes by at least 20%. In all approaches, we try to target $1.2k$ destinations for fault tolerance. For the GeoGrid approach, we choose the size of the routing-grids to be such that any node located at the center of the routing-grid is capable of talking to any gateway in its 8 neighboring routing-grids [12]. Thus, we choose routing-grids to be of size $d \times d$ where $d(= 4.7)$ is transmission radius (10) times $\sqrt{2}/3$. Moreover, as suggested in [12], we choose the number of tickets as $(w + l)/d$, where w and l are the width and length of the chosen rectangle.

We use two metrics to measure the performance of our approaches. The *success ratio* serves as the metric for robustness, and is defined as $|D'|/k$, where D' is the set of destinations that successfully receive the data packet and k is the number of destination requested by the manycast operation. The second metric is the total communication cost, which is computed as the total number of transmissions made per request including the overhead due to control messages such as location updates, grid-leader selection, etc.

7.1. Randomly Distributed Destinations

In this set of experiments, destinations move randomly over the entire network region. 1200 nodes are randomly placed and the entire network region is divided into 4×4 grids, where each grid has at least 2 grid leaders. When not being varied, the destinations move at a speed of $10m/sec$ with pause time of 20 seconds between paths, nodes and communication links fail with a probability of 10%, and the manycast operation request parameter k is set to 30.

Effect of Delivery Request k. Fig. 3 depicts the performance of various algorithms for different values of k, the number of destinations required by the manycast operation. We observe that all the approaches except the GeoGrid ticket-based scheme maintain high success ratio of around 90%. With larger k, the success ratio only falls slightly. As expected, the overall transmission cost of all approaches increases with increase in k. The GridTree approach always achieves higher success ratio than the GeoGrid ticket-based approach, and obtains comparable success ratio but much less transmission cost than the Rectangular flooding and GeoGrid flood-based approaches.

Effect of Network Density. To depict the impact of network density, we vary the number of nodes from 400 to 1600 in the fixed network area. In terms of the success ratio (Fig. 4(a)), the Rectangular flooding approach performs best for across network densities, followed by GeoGrid flood-based scheme, GridTree, and GeoGrid ticket-based scheme. Overall transmission cost of all approaches increases initially with the increase in network density, and then remains relatively constant (except for Rectangular flooding wherein as expected the routing overhead continues to increase) (Fig. 4(b)). In summary, the GeoGrid flood-based and GridTree approaches perform the best and similarly in terms of acceptable success ratio and minimal transmission cost.

Effect of Link Failure Rates. Here, we vary the communication link failure probability from 0 to 0.3, while keeping the node failure probability fixed at 0.1. The success ratio of the GridTree approach is above 90% (Figure 5(a)) with link failure rates of up to 0.3. The routing cost of various approaches (Figure 5(b)) decreases with the increase in failure rate, since less number of nodes receive and hence, transmit packets. In general, the trend of all approaches is similar, with the GridTree approach performing better than Rectangular flooding and GeoGrid approaches.

Effect of Number of Grids. To study the effect of the number of grids, we divide the network into $\beta \times \beta$ grids, and vary β from 1 to 6 and set $k = 50$. In Fig. 6(a), we can see that the success ratio of GridTree is very low for $\beta < 4$, due to coarseness of the location information of destination. Fig. 6(b) shows that the overall routing cost of all approaches increases with the increase in β, due to increase in the frequency of updates. We see that the $\beta = 4$ or 5 is most suitable for the GridTree approach for chosen parameters.

7.2. Destinations in Peripheral Grids

In this section, we consider the scenario wherein the movement of destinations is constrained to the peripheral grids. In a general sense, this scenario illustrates the situation of a large network field, wherein most of the destinations are far away from the source. We choose a network size of 1600 nodes, and divide the region into 6×6 grids.

Effect of Delivery Request k. In Fig. 7(a), we see that Rectangular flooding, GridTree, and GeoGrid flood-based approaches all achieve high success ratio of above 90% for various values of k. The success ratio of GeoGrid ticket-based approach decreases drastically with the increase in k. Fig. 7(b) shows that the GridTree approach significantly outperforms the Rectangular flooding and GeoGrid flood-based approaches, which become more prominent with the increase in k. Although GridTree incurs more transmission cost than GeoGrid ticket-based approach for large k, the success ratio of the GeoGrid ticket-based approach is unacceptably low.

Effect of Network Density. As in the case of the the first scenario, we show the impact of network density by keeping the network region fixed to 150×150 meters square, but vary the number of nodes from 1200 to 2000. Fig. 8(a) shows that

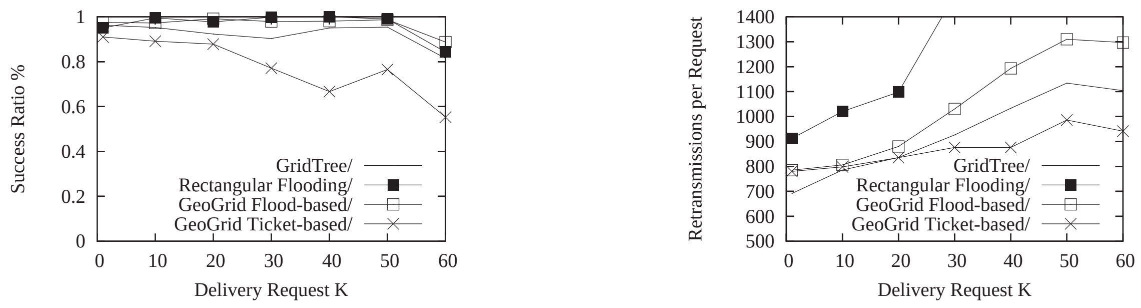


Figure 7. Effect of delivery request k when destinations are far away. (a) Success Ratio. (b) Transmission Cost.

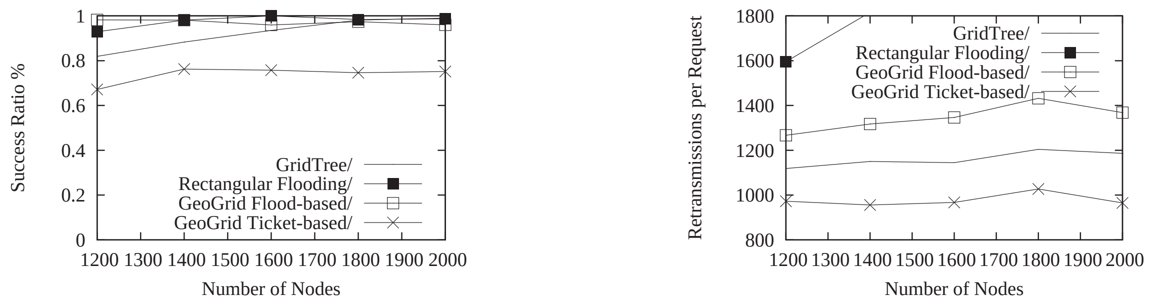


Figure 8. Effect of network density when destinations are far away. (a) Success Ratio. (b) Transmission Cost.

Rectangular flooding and GeoGrid flood-based approaches always achieve a success ratio of above 90%, while the GeoGrid ticket-based approach performs very bad. The success ratio of GridTree increases with the increase in number of nodes from 1200 to 1600, and then remains relatively unchanged. Fig. 8(b) shows that the overall routing cost of GridTree and GeoGrid approaches is not impacted much by network density, since there are already sufficient number of nodes. The overall transmission cost of Rectangular flooding is very high, and increases dramatically with the increase in number of nodes as expected.

Effect of Link Failure Rates. As in the first scenario , we vary the communication link failure probability from 0 to 0.3, while keeping the node failure probability fixed at 0.1. Figure 9(a) depicts the effect of failures rates on the success ratio of various algorithms. The success ratio of all approaches keep above 90% except GeoGrid ticket-based approach. The routing cost of GridTree is much less (Figure 9(b)) than GeoGrid flood-based and rectangular flooding approaches, while achieving similar success ratio.

8. Conclusions

Wide deployment of sensor networks encourages us to support the manycast operation, which represents a flexible group communication primitive. However, due to failure-prone nodes and links in the sensor networks, providing robust manycast is a research challenge, especially when the given destinations are mobile. In this article, we have presented several protocols in conjunction with a distributed location information management technique to support the manycast operation. The simulation results show that the GridTree approach performs the best, with the GeoGrid flood-based approach performing a close second.

References

[1] S. Arora and G. Karakostas. A 2+epsilon approximation for the k-mst problem. In *SODA*, 2000.

[2] P. Bahl and V. N. Padmanabhan. Radar: An in-building RF-based user-location and tracking system. In *INFOCOM*, 2000.

[3] C. Carter, S. Yi, and P. Ratanchandani. Manycast: Exploring the space between anycast and multicast in ad hoc networks. In *MobiCom*, 2003.

[4] B. Deb, S. Bhatnagar, and B. Nath. ReInForM: Reliable information forwarding using multiple paths in sensor networks. In *IEEE Conference on Local Computer Networks*, 2003.

[5] N. Garg. Saving an epsilon: A 2-approximation for the k-MST problem in graphs. Unpublished.

[6] B. H.-Wellenhof, H. Lichtenegger, and J. Collins. *Global Positioning System: Theory and Practice*. Springer-Verlag, 1997.

[7] H. S. Kim, T. F. Abdelzaher, and W. H. Kwon. Minimum-energy asynchronous dissemination to mobile sinks in wireless sensor networks. In *SENSYS*, 2003.

[8] Y. Ko and N. Vaidya. GeoTORA: A protocol for geocasting in mobile ad hoc networks. In *ICNP*, 2000.

[9] Y. B. Ko and N. H. Vaidya. Geocasting in mobile ad hoc networks: Location-based multicast algorithms. In *Workshop on Mobile Computing Systems and Applications (WMCSA)*, 1999.

[10] S. Lee, W. Su, and M. Gerla. On-demand multicast routing protocol in multihop wireless mobile networks. *Mobile Networks and Applications*, 7(6), 2002.

[11] J. Li, J. Jannotti, D. S. J. D. Couto, D. R. Karger, and R. Morris. A scalable location service for geographic ad hoc routing. In *MobiCom*, 2000.

[12] W. Liao, Y. Tseng, K. Lo, and J. Sheu. GeoGRID: A geocasting protocol for mobile ad hoc networks based on GRID. *J. of Internet Technology*, 1, 2000.

[13] E. L. Madruga and J. J. Garcia-Luna-Aceves. Scalable multicasting: The core-assisted mesh protocol. *ACM Mobile Networks and Applications*, 2001.

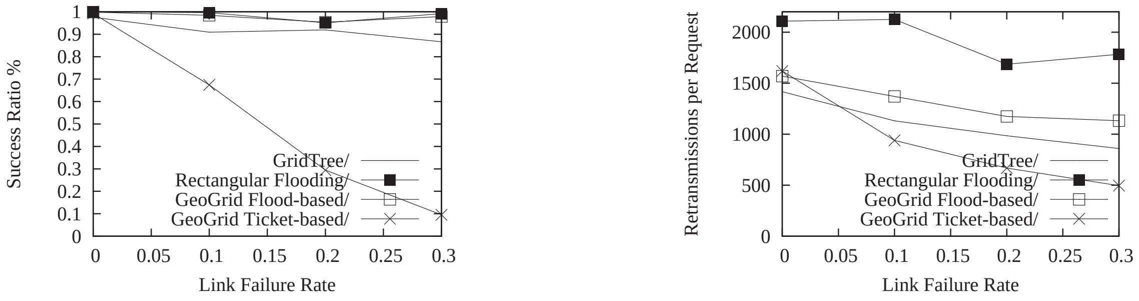


Figure 9. Effect of link failure rate when destinations are far away. (a) Success Ratio. (b) Transmission Cost.

[14] V. D. Park and J. P. Macker. Anycast routing for mobile services. In *Intl. Conf. on Information Sciences and Systems*, 1999.

[15] C. Perkins, E. Royer, and S. Das. Ad hoc on demand distance vector (AODV) routing. RFC 3561, 2003.

[16] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *SIGCOMM*, 1994.

[17] E. M. Royer and C. E. Perkins. Multicast operation of the ad-hoc on-demand distance vector routing protocol. In *MobiCom*, 1999.

[18] F. Ye, G. Zhong, S. Lu, and L. Zhang. GRAdient Broadcast: A robust data delivery protocol for large scale sensor networks. *ACM WINET*, 2005.

[19] Z. Ye, S. Krishnamurthy, and S. Tripathi. A framework for reliable routing in mobile ad hoc networks. In *INFOCOM*, 2003.