Hindawi Publishing Corporation
Journal of Computer Networks and Communications
Volume 2011, Article ID 680936, 11 pages
doi:10.1155/2011/680936

Research Article

Efficient and Reliable Dissemination in Mobile Ad Hoc Networks by Location Extrapolation

Adnan Agbaria,[1] Muhamad Hugerat,[1] and Roy Friedman[2]

[1] *Computer Science Department, The Academic Arab College, 31905 Haifa, Israel*
[2] *Computer Science Department, Technion, 32000 Haifa, Israel*

Correspondence should be addressed to Adnan Agbaria, adnan.agbaria@gmail.com

Received 22 March 2011; Accepted 6 July 2011

Academic Editor: Abdelhamid Mellouk

Data dissemination is an important service in mobile ad hoc networks (MANETs). The main objective of this paper is to present a dissemination protocol, called *locBcast*, which utilizes positioning information to obtain efficient dissemination trees with low-control overhead. This paper includes an extensive simulation study that compares locBast with selfP, dominantP, fooding, and a couple of probabilistic-/counter-based protocols. It is shown that locBcast behaves similar to or better than those protocols and is especially useful in the following challenging environments: the message sizes are large, the network is dense, and nodes are highly mobile.

1. Introduction

Disseminating data from any given node to all other nodes is a basic service in wireless mobile ad hoc networks. In ad hoc networks, data dissemination is more challenging because of the high mobility and instability of the network. For example, a fire-fighter in a search and rescue application (or a soldier in a military application) may wish to send a video stream from his helmet to all other fire-fighters (or soldiers). Alternatively, a child's mobile phone may wish to send a search request for a given video clip to all other devices in the school in an ad hoc file-sharing application. In addition, many wireless routing protocols need a dissemination service in order to search for the destination nodes. For example, many unicast routing protocols such as dynamic source routing (DSR), ad hoc on demand distance vector (AODV), and location aided routing (LAR) rely on disseminating to establish routes.

Two important, and sometimes conflicting, aspects of data dissemination are reliability and efficiency. Reliability measures the likelihood that a message will be received by all nodes in the network. Yet, in order to ensure high reliability, it may be necessary to send redundant messages. These messages increase the load on the network and therefore reduce its effective capacity, hence, the significance of the efficiency of the protocol.

The simplest way to implement data dissemination is by *flooding*. That is, the sender broadcasts its message to all its neighbors. Each of them, when it receives the message for the first time, rebroadcast the message. This repeats iteratively until all nodes have received and rebroadcasted the message at least once. Clearly, in flooding, a message is propagated along all possible paths in the networks, and therefore it can ensure high reliability. However, flooding is highly inefficient. Moreover, under high load, the large number of redundant messages generated by flooding results in many collisions, which ultimately also degrades its reliability.

Consequently, many works have been published on developing protocols that ensure high reliability in a much more efficient manner than flooding. Some of these protocols attempt to build an overlay and only disseminate messages along the overlay's edges, for example, [1–3], whereas others have resorted to probabilistic and counter based pruning techniques [4–6].

It is well known that the most efficient way to disseminate a message in a network is along the edges of a source-based spanning tree [7]. On the other hand, the cost of maintaining such a tree for each source can be prohibitively

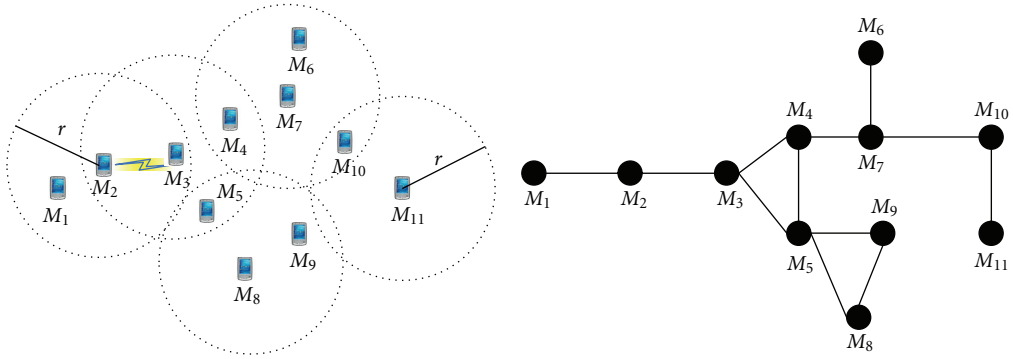

Figure 1: An example of the system.

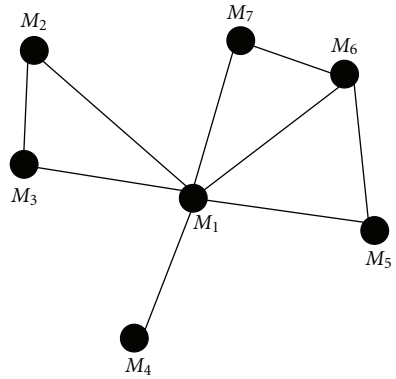

Figure 2: An example of a graph G_1.

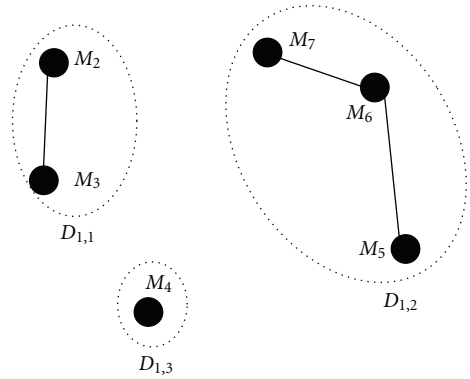

Figure 3: The obtained graph H_1.

expensive. This is especially true for opportunistic networks, in which the network topology is constantly changing, disconnections and reconnections may frequently occur, there is no central authority, and often network and power resources are limited.

In this paper, we refer to two techniques for implementing source-based dissemination trees: self pruning (*selfP*) and dominant pruning (*dominantP*) [2]. Specifically, selfP uses periodic "Hello" messages to learn their immediate (1-hop) neighborhood, and from that decide which nodes should forward a message. On the other hand, in dominantP, the "Hello" messages include information about the 2-hop neighborhood (i.e., each node can learn who are the neighbors of its neighbors). Hence, the forwarding decisions in dominantP generate a dissemination tree that is much closer to the optimal tree, meaning that dominantP obtains higher reliability and better efficiency compared to selfP. However, its "Hello" messages can become very large when the network is large and dense.

As many new smartphones are equipped with positioning capabilities, for example, GPS and GSM triangulation, we are motivated to explore whether these capabilities can be used to improve dissemination protocols. In particular, we seek a protocol that only relies on 1-hop neighborhood information and can still be at least as reliable and efficient as dominantP.

In this paper, we present a new approach in which locations of a node's 1-hop neighbors are extrapolated in order to disseminate a message efficiency and reliably. We demonstrate this idea by presenting a new broadcast protocol, called locBcast. Furthermore, we compare the performance of locBcast by simulations with selfP, dominantP, flooding, and a couple of probabilistic-/counter-based protocols. The results of the simulations show that locBcast is indeed more efficient and reliable than most other protocols we have checked and much faster than counter-based dissemination. locBcast particularly excels when messages are large, the network is dense, and nodes move fast.

2. Related Work

Several approaches for providing broadcast routing in mobile ad hoc networks have been proposed in the literature. In Section 5, we compare results from our new broadcast protocol with other five broadcast protocols, which are flooding, gossip, counterb, selfP, and dominantP. The comparison results show that our protocol achieves both high reliability and low latency, while the other protocols achieve either reliability or low latency.

Williams and Camp presented in [8] a classification of wireless broadcast protocols into four categories. The simple flooding protocol has its own category. The other protocols

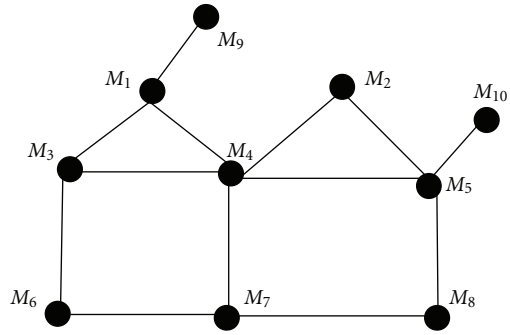

Figure 4: An example of ad hoc system with ten mobile nodes.

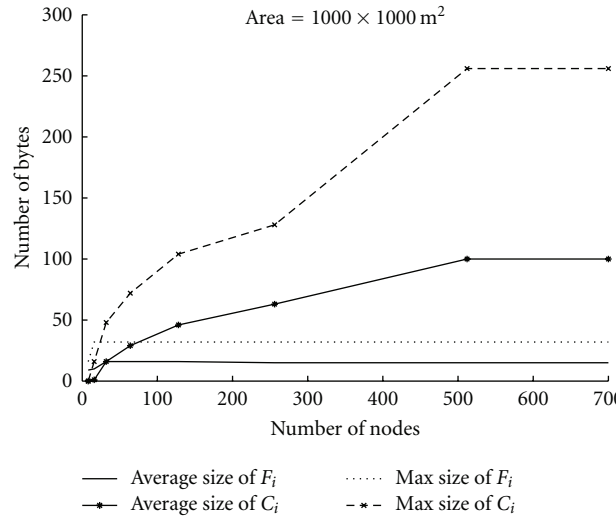

Figure 5: Size of F_i and C_i as a function of network density.

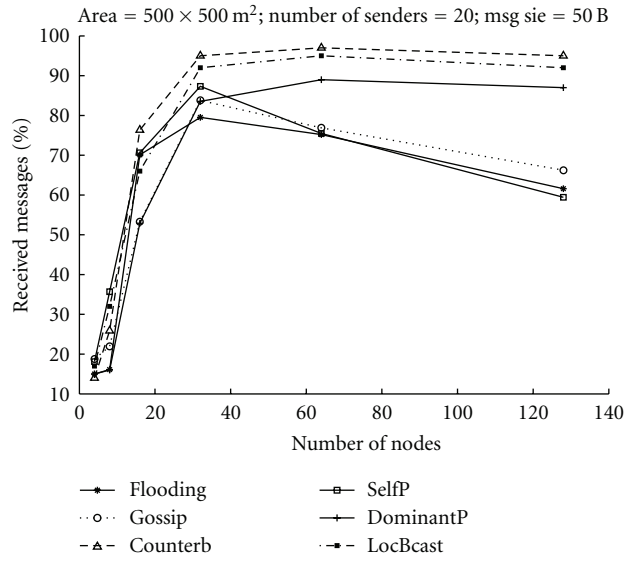

Figure 6: Reliability versus number of nodes with high load.

belong to the following classes: probability-based methods, location-based methods, and neighbor knowledge methods. Then, they showed a comparison between selected protocols from the four classes. The comparison showed that the protocols of neighbor knowledge achieve the best results in low latency, delivery ratio, and number of retransmitting nodes. Our protocol (locBcast) belongs to the neighbor knowledge class rather than the location-based class. This is because in locBcast we only extend the neighbor knowledge to include positioning information, where most of the location-based methods require global positioning information.

Tseng et al. [5] presented a location-based broadcast approach where every node attaches its location with the messages. As shown in [5], this approach can help a node whether to forward a message or not only if it has received the message from other nodes that are existing in specific locations. On other hand, our approach combines location-based and neighbor knowledge to let every node decides immediately if it needs to forward a message or not.

In Section 5, we considered the two neighbor knowledge protocols selfP and dominantP. Our results show how locBcast obtains higher reliability and lower latency than these protocols. The other neighbor knowledge protocols mentioned in [8] are similar to the dominantP protocol. The multipoint relaying protocol [9] and ad hoc broadcast protocol [10] are similar to dominantP in that rebroadcasting nodes are explicitly chosen by upstream senders and every node knows the network topology within a 2-hop radius. In addition, the Lightweight and Efficient Network-Wide Broadcast (LENWB) protocol [11] belongs to the neighbor knowledge class. This protocol relies on 2-hop neighbor knowledge obtained from "Hello" packets. However, instead of a node explicitly choosing nodes to rebroadcast, the decision is implicit. In LENWB, each node decides to rebroadcast based on the knowledge of which of its other one and two-hop neighbors are expected to rebroadcast.

We can summarize here that most of the neighbor knowledge protocols require 2-hop knowledge. To the best of our findings, the only two protocols that require 1-hop knowledge are selfP and locBcast. No doubt that the overhead of control messages is greater in the case of 2-hop knowledge, where a control message of node M_i needs to be transmitted $|N_i|$ times (N_i is the neighbor set of M_i). In our protocol, however, every control message is transmitted only once. This overhead of control messages affects the reliability and latency of the broadcast protocol. Moreover, since locBcast uses the location knowledge of the neighbors, it achieves a low number of transmitting nodes (see the results in Section 5).

From the probability-based class, we considered two protocols which are counterb that achieves good reliability results but incurs high latency and gossip protocols that disseminate messages fast, but this is not reliable enough. The advantage of these protocols is the low number of control messages they generate. RAPID [12] is another protocol in this class which improves the reliability more than counterb and gossip at the expense of periodic active gossip messages.

El Fawal et al. [13] presented a probability-based broadcast approach. Their work defines several functions to decide whether to forward a message or not. The most

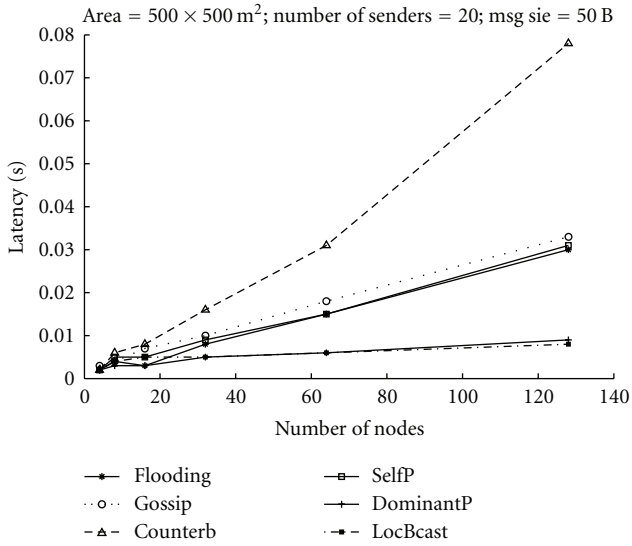

Figure 7: Latency versus number of nodes with high load.

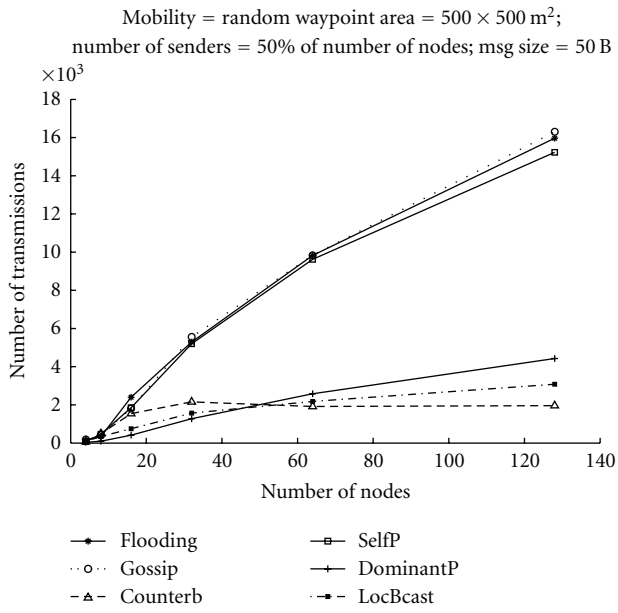

Figure 8: Total number of transmissions.

important functions are the inhibition and the spread-control functions. The inhibition function is identical to the gossip protocol. The spread control function depends on the TTL (time to live) value and some environment parameters that are provided by the user (or administrator). Therefore, this approach is considered as an improvement of the gossip protocol when some knowledge of the environment exists.

A podcasting system for delay tolerant networks has been presented in [14]. Podcasting requires dissemination capabilities. Yet, the model and goal are different compared to our case as they deal with delay-tolerant content and networking. Hence, in [14], nodes use opportunistic forwarding when they meet each other. Also, the content is assumed to be large media files, whereas in our case, messages are often relatively short and self-contained.

Most of the efficient routing protocols for MANETs were developed for point-to-point routing (unicast). Similar to our approach, Shah and Nahrstedt [15] present a predictive location-based QoS unicast routing protocol for MANETs. In addition, Hughes et al. describe event-based real-time middleware for VANET [16]. They use a proximity-based event-propagation technique to guarantee real-time constraints within the defined proximities only. The proposed RT-STEAM identifies and delivers events of interest based on location. Unlike our work that guarantees real-time communication between any two mobile nodes, RT-STEAM provides real-time delivery support only within predefined cells.

Sun et al. [17] propose a cross-layer QoS framework between MAC and IP for utilizing prioritization support in MAC. This paper assumes stability or low node mobility. In [18], Friedman and Kliot present a comprehensive survey of location services in MANETs. Cerdà et al. [19] present a mechanism for bandwidth management and distribution in MANETs to support the QoS guarantees.

3. System Model

We assume an opportunistic network, which is a wireless ad hoc network consisting of a set of mobile nodes that may communicate with each other using omnidirectional antennas. As presented in Figure 1(a), we denote the mobile nodes by $M_1, \ldots, M_i, \ldots$. We assume that all mobile nodes have the same *transmission range* of r. Two nodes M_i and M_j can communicate directly with each other if M_i is within the transmission range of M_j and vice versa. Let $d()$ be the *Euclidean distance* function, we say that M_i is a *neighbor* of another node M_j if $d(M_i, M_j) \leq r$. We denote by N_i all the neighbors of M_i (including M_i itself), that is, $N_i = \{M_j \mid d(M_i, M_j) \leq r\}$.

We impose no limit on the maximum number of mobile nodes in the system, but we assume that the mobile nodes are scattered in a given finite size area. A node can physically move within this area. It may move at any time in any direction and at any speed. New nodes may join and existing nodes may leave at any time. Therefore, the link connectivity and network topology change with nodes' movement. We model the dynamic changes in the network by a sequence of graphs indexed by time instances. We first decompose the time horizon $T = [0, \infty)$ into a set of time instances $T' = \{t_1, t_2, \ldots\}$ such that during the time $[t_i, t_{i+1})$ the network topology remains unchanged. So, at time t, we define the network connectivity graph as $G(t) = (V, E(t))$, where $E(t)$ represents the set of wireless links at time $t \in T'$. We denote the connectivity graph by G if the time t is irrelevant. Figure 1(b) shows the corresponding connectivity graph of the system presented in Figure 1(a).

We assume that every node has access to a location service such as global positioning system (GPS). Such a service provides position, velocity, and time (PVT) information about the node. In addition, we assume the standard

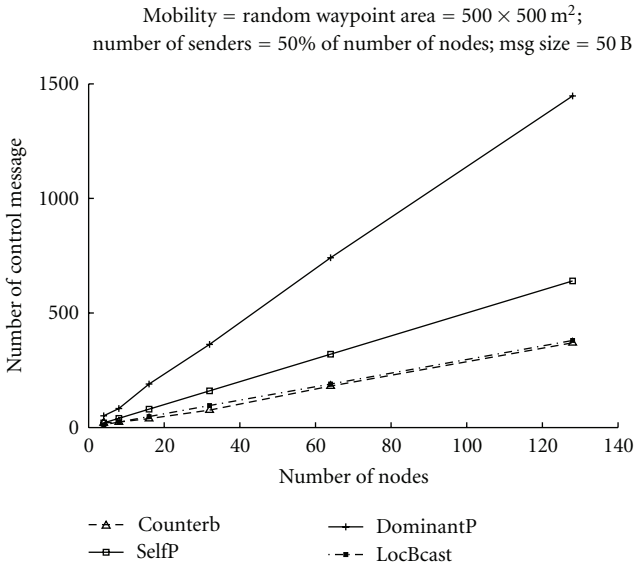

Figure 9: The number of generated control messages.

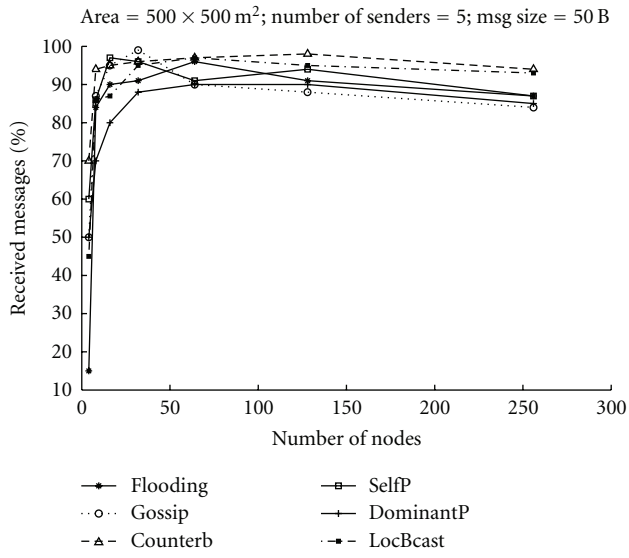

Figure 10: Reliability versus number of nodes with low load.

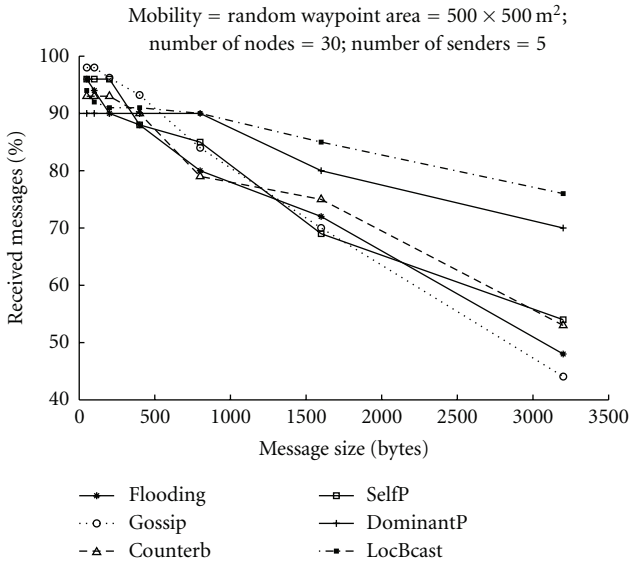

Figure 11: Reliability versus message size.

communication stack for the mobile nodes, where the MAC layer works with CSMA/CA transceiver. We assume that every mobile node has a unique address for routing.

4. Extrapolation-Based Dissemination

Given a mobile node M_i, to extrapolate the location of every neighbor node in N_i at any time, M_i needs to know the initial location and the updated velocity vector of N_i [15]. Therefore, in our approach, periodically, every mobile node M_i sends its location (denoted by L_i), velocity (denoted by V_i), and the number of 1-hop neighbors (denoted by $|N_i|$) to its 1-hop neighbors. Notice here that M_i can extrapolate when another node leaves N_i without extra information.

The nodes determine their location and velocity vectors using an external location service, for example, GPS. If the location service does not provide velocity vector information, it can be calculated from two consecutive location measurements.

In our approach, when a mobile node M_i decides to broadcast a packet P, it computes a *forwarding* set (denoted by F_i) and a *candidates* set (denoted by C_i) from N_i. It then attaches F_i (with the location of every node in F_i) and C_i to the transmitted packet. Furthermore, when a node $M_j \in N_i$ receives P for the first time, it checks F_i and C_i. If M_j is in F_i, then it transmits P after computing F_j and C_j, recursively. Otherwise, if M_j is in C_i, it might still decide to transmit P depending on whether the transmission of F_i and M_i reach all the nodes in N_j or not, as explained below.

We now turn to the details of the locBcast protocol, implementing our new approach. Suppose now that M_i wishes to broadcast a packet P. The handling of such a message is divided among the following three phases:

Phase 1 (determining disjoint sets in N_i). Consider the graph $G_i = (N_i, E_i)$. This graph is generated by the set of nodes in N_i; the vertices are the mobile nodes of N_i and an edge of E_i exists between two vertices in G_i only if they are neighbors. Notice that M_i has an edge in G_i to every other node. Consider now the subgraph H_i, which is obtained from G_i by extracting the node M_i. Since some neighbors of N_i may become disconnected without M_i, the subgraph H_i consists of disjoint connected components. Assume that we have m disjoint connected components in H_i. Let $D_i = \{D_{i,1}, \ldots, D_{i,m}\}$ be the collection of the disjoint connected sets in H_i. Namely, two nodes $M_l, M_h \in N_i$ belong to $D_{i,j}$ if there is a path between M_l and M_h in G_i that does not pass through M_i.

Figure 2 presents an example of a graph G_1. Notice that the node M_1 has a direct connection to every other node in the graph.

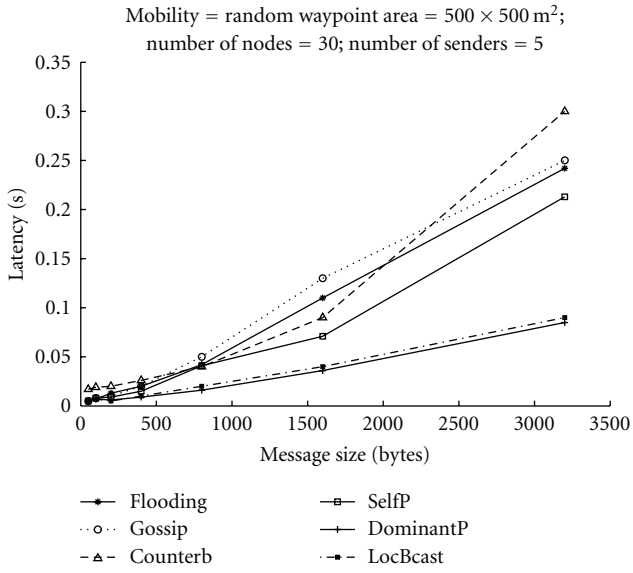

FIGURE 12: Latency versus message size.

Figure 3 presents the graph H_1, which is obtained from the graph of Figure 2 by extracting the node M_1. There are three disjoint connected sets in H_1.

Phase 2 (determining F_i and C_i). In this phase, M_i determines the forwarding and candidates sets. Consider the graph H_i of Phase 1. The forwarding set F_i contains the node from every $D_{i,j}$ that has the maximum number of neighbors that are not in N_i (breaking symmetry arbitrarily). Namely, the node from $D_{i,j}$ whose transmission can reach the largest number of new nodes. The candidates set C_i contains all the remaining nodes of $D_{i,j}$ that also have neighbors that are not in N_i. Namely, their transmission still contribute to the dissemination of P to reach additional new nodes.

Formally, for every $D_{i,j} \in D_i$ and for every $M_k \in D_{i,j}$, M_i computes $n_k = |N_k \setminus N_i| = |N_k| - |N_k \cap N_i|$. That is, n_k is the number of mobile nodes that are in N_k, but are not neighbors of M_i. Notice here that M_i has enough information to compute n_k even though it does not know the entire composition of N_k. This is because M_i knows the value of $|N_k|$ for every $M_k \in N_i$. Moreover, since M_i knows the location of all of its own neighbors and the transmission range, it can determine which of its neighbors is also a neighbor of M_k. In other words, it knows the set of joint neighbors $N_k \cap N_i$.

Let $M_h \in D_{i,j}$ be the node such that $n_h = \max\{n_k \mid \forall M_k \in D_{i,j}\}$. Then, update F_i and C_i as follows:

(1) if $n_h > 0$, add M_h with its extrapolated location to the forwarding set F_i;

(2) then, add every node $M_k \in D_{i,j}$ such that $n_k > 0$ to the candidates set C_i.

Algorithm 1 presents a pseudocode of the *determine* function for determining F_i and C_i at M_i.

In Line 1 of Algorithm 1, we initialize F_i and C_i to be empty. Line 2 starts with the main loop that ends at Line 16. In this loop, we examine every set $D_{i,j} \in D_i$. For every set $D_{i,j}$, n_h will have the maximum value of n_k. In Line 3, we initialize n_h to be -1 and its corresponding node to be unknown. We start an inner loop at Line 4 that computes n_k for every $M_k \in D_{i,j}$ (Line 5). Then, we set n_h to be n_k and remember its node if n_k has a greater value than n_h (Line 9). We add n_k (Line 11) or the previous value if n_h (Line 8) to C_i if they are greater than zero. Namely, if there are nodes in $N_k \setminus N_i$. After the inner loop, we examine if $n_h > 0$ (Line 13). If yes, we extrapolate the new location of the node with the value of n_h (Line 14). Then, we add this node with the extrapolated location to F_i (Line 15).

Last, after determining the forwarding and candidates sets, M_i sends the packet P attached with these sets. Moreover, for each node M_j in F_i, M_i also attaches the location information of M_j to the message.

Phase 3 (upon receiving the packet P). Let M_j be a mobile node in N_i. Upon receiving the packet P from M_i (for the first time), M_j behaves as follows:

(i) if $M_j \in F_i$, then it transmits the packet P after determining F_j and C_j, as described in Phase 2, with the exception that M_i cannot appear in either C_j or F_j;

(ii) otherwise, if $M_j \in C_i$, then it checks if there is a node in N_j that might not receive the packet P from M_i nor from any of the nodes in F_i. If there is a such node, then M_j transmits the packet P after determining F_j and C_j.

Notice that M_j can detect such nodes since it has the location information of all nodes in F_i. Hence, if $M_j \in C_i$, it transmits P only if the following condition is satisfied: $\exists M_h \in N_j$ s.t. $d(M_h, M_i) > r$, and $d(M_h, M) > r$, $\forall M \in F_i$.

4.1. A Running Example. Let us explain the locBcast protocol with an example. Consider the graph, which is generated from a network system, presented in Figure 4. Assume that M_4 wishes to disseminate a packet P to all the other nodes. We describe here the three phases of our protocol on this example.

Phase 1 (determining the disjoint sets in N_4). The mobile node M_4 has five neighbors, $N_4 = \{M_1, M_2, M_3, M_4, M_5, M_7\}$. So the graph G_4 has six vertices and the graph H_4 has three disjoint connected components. The collection D_4 has the following three sets: $\{D_{4,1} = \{M_1, M_3\}, D_{4,2} = \{M_2, M_5\}, D_{4,3} = \{M_7\}\}$.

Phase 2 (determining the forwarding set F_4 and the candidates set in C_4). For every node $M_k \in D_{4,j}$, $j = 1, 2, 3$, we compute n_k.

(1) $D_{4,1} = \{M_1, M_3\}$. For M_1, we have $n_1 = |N_1| - |N_1 \cap N_4| = |\{M_1, M_3, M_4, M_9\}| - |\{M_1, M_3, M_4\}| = 4 - 3 = 1$. For M_3, we have $n_3 = |N_3| - |N_3 \cap N_4| = |\{M_1, M_3, M_4, M_6\}| - |\{M_1, M_3, M_4\}| = 1$. Therefore,

determine()
(1) $F_i = \emptyset$, /* initialization*/
(2) $\forall D_{i,j} \in D_i$, **do**
(3) Let $n_h = -1$ and M $= \emptyset$
(4) $\forall M_k \in D_{i,j}$, **do**
(5) $n_k = |N_k| - |N_k \cap N_i|$
(6) **if** $(n_k > n_h)$, then
(7) **if** $(n_h > 0)$, then
(8) $C_i = C_i \cup$ M
(9) $n_h = n_k$ and M $= M_k$
(10) **else if** $(n_k > 0)$, then
(11) $C_i = C_i \cup \{M_k\}$
(12) **enddo** /* end of the inner loop*/
(13) **if** $(n_h > 0)$, then
(14) Loc = **predict Location**(M)
(15) $F_i = F_i \cup \langle$M,Loc$\rangle$
(16) **enddo** /* end of the external loop*/

Algorithm 1: The determination of F_i and C_i at M_i.

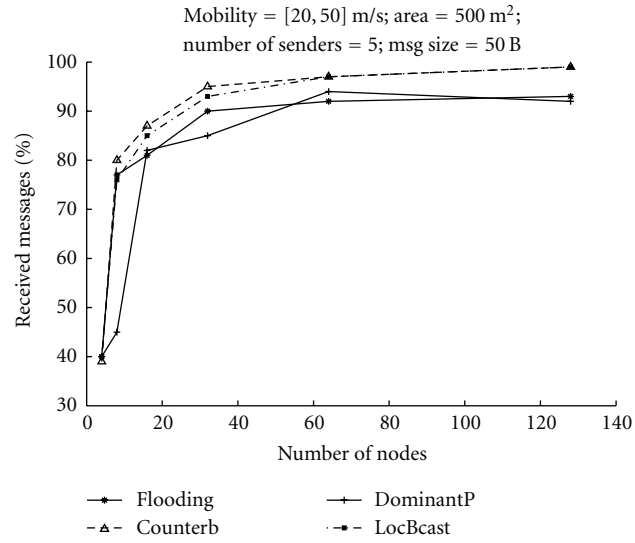

Figure 13: Reliability versus number of nodes with high mobility.

the maximum value is 1. So we choose either M_1 or M_3 to be in F_4 and the other in C_4. Let us choose M_1 to be in F_4 and M_3 to be in C_4.

(2) $D_{4,2} = \{M_2, M_5\}$. For M_2, we have

$$n_2 = |\{M_2, M_4, M_5\}| - |\{M_2, M_4, M_5\}| = 0. \quad (1)$$

For M_5, we have

$$n_5 = |\{M_2, M_4, M_5, M_8, M_{10}\}| - |\{M_2, M_4, M_5\}| = 2. \quad (2)$$

Therefore, n_5 is the maximum. So, we add M_5 to F_4, but we do not add M_2 to C_4 because n_2 is zero.

(3) $D_{4,3} = \{M_7\}$. For M_7 we have that

$$n_7 = |\{M_4, M_7, M_8, M_6\}| - |\{M_4, M_7\}| = 2. \quad (3)$$

Therefore, we add M_7 to F_4.

At the end of this phase, we have $F_4 = \{M_1, M_5, M_7\}$ and $C_4 = \{M3\}$. Then, M_4 transmits the message $\langle P, F, C_4\rangle$, where $F = \{\langle M_k, L_k\rangle \mid M_k \in F_4\}$. Notice that L_k is the extrapolated location of M_k.

Phase 3 (upon receiving the packet message of M_4). When a node $M_i \in N_4$ receives the message, it verifies if $M_i \in F_4$. If yes, M_i transmits P with the new F_i and C_i. Notice here that M_i, F_i, and C_i as described above with excluding M_4. However, if $M_i \in C_4$ but not in F_4, M_i needs to decide whether to transmit P or not. M_i decides to transmit P only if the transmission of F_4 does not cover all the neighbors of M_i. In our example, M_3 (the only node in the candidate set) can see that its only neighbor M_6 will receive the transmission of M_7, which is in F_4. Therefore, M_3 does transmit P. Notice here that since the location of M_7 is attached to P, M_3 can verify if M_6 belongs to N_7 or not.

4.2. Discussion of Protocol Properties. Since our protocol requires that every node sends its information to the immediate neighbors, by the classification of Williams and Camp [8], our protocol is considered a "neighbor knowledge" scheme. Therefore, the correctness of the protocol greatly depends on each node having accurate knowledge of its neighbors. In locBcast, every node sends a control message periodically or upon a significant velocity change, which ensures up-to-date neighborhood knowledge.

In terms of control messages overhead, locBcast only relies on periodic sending of a small amount of data: the number of node's neighbors, the ids of the nodes in F_i and their locations, and the ids of nodes in C_i (F_i and C_i are attached with data messages). In particular, the sizes of F_i and C_i are expected to be small. If the network is sparse, then each node only has a small number of neighbors in any case, and the combined size of F_i and C_i is bounded by the size of the 1-hop neighborhood. If the network is dense, then there is a great overlap between the areas covered by the transmission ranges of neighbors. Hence, most nodes do not participate neither in F_i nor in C_i, so their size is kept small. This is demonstrated in Figure 5, which summarizes the results of the following experiment. We have placed varying numbers of mobile nodes uniformly at random in a square area of $1000 \times 1000\,\text{m}^2$ and measured the maximum and average sizes of the C_i and F_i fields generated by locBcast. The number of nodes ranged from 1 to 700. As can be seen, both the maximum and average sizes of F_i remain constant. This can be expected, since the denser the network becomes, the more coverage we get from each single transmission. As for C_i, it appears that its size grows sublinearly with the density, and the latter roughly grows as $O(\sqrt{n})$.

Finally, as locBcast builds a source-based dissemination tree with little overlap, the dissemination of data messages themselves is also very efficient. As we report in Section 5, empirically, locBcast is indeed quite parsimonious compared to other "neighbor knowledge" approaches.

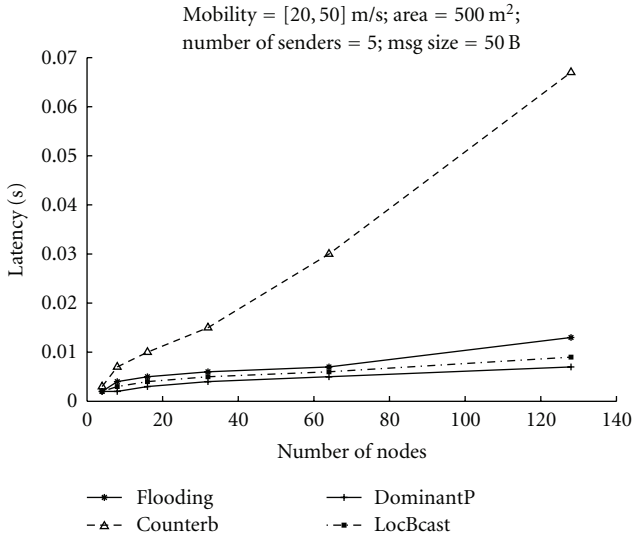

Figure 14: Latency versus number of nodes with high mobility.

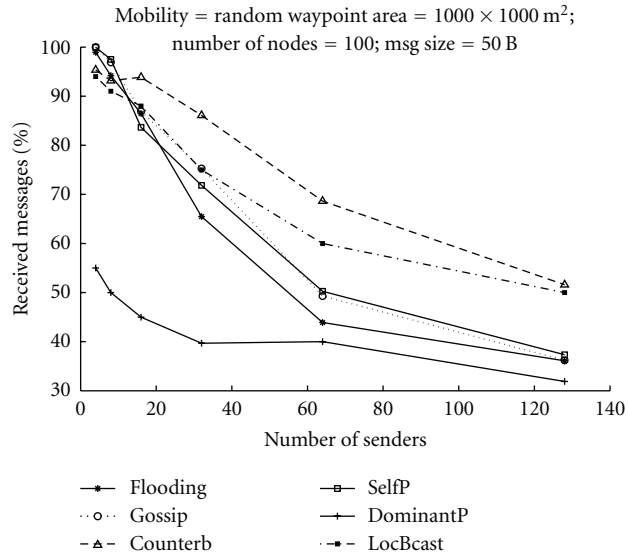

Figure 15: Reliability versus number of senders.

We now turn our attention to the reliability of locBcast. We start by showing that every node in the 2-hop neighborhood of the sender of a message receives the message, unless it moves away too quickly. By applying this claim recursively, we prove that locBcast disseminates messages reliably to the entire network, at least when the rate of change in the network is slower than the propagation speed of a message.

Claim 1. If a mobile node M_i transmits a packet P and there are no message losses, then every node M_j that is in P's 2-hop neighborhood at the time of transmission will either receive P or move out of M_i's 2-hop neighborhood.

Proof. If a node M_j moves out of the 2-hop neighborhood of M_i, then the claim is satisfied immediately. Hence, we concentrate on the situation in which M_j remains in the 2-hop neighborhood of M_i. Notice that if a node M_j is in the 1-hop neighborhood of M_i, then M_j will receive the message. As for a node M_j that is not in the 1-hop neighborhood, by the protocol's code, one of the joint neighbors of M_i and M_j will rebroadcast P, and M_j will receive it. □

Message losses may occur mainly due to collisions. However, by jittering broadcasts [20], it is possible to eliminate most of them. Additional remedy may include rebroadcasting each message more than once, or periodic gossip [12]. At any event, this problem is common in all neighbor knowledge protocols, and the same solutions can be applied to all of them. As reported in Section 5, our experimental results have shown that locBcast is at least as reliable as any other neighbor knowledge protocol we compared with. In some situations, counter-based protocols, such as [5], can be more reliable, but at the cost of much higher latency.

5. Performance Evaluation

In this section, we evaluate the performance of locBcast and compare it with the performance of five other broadcast protocols. In our simulations we have mainly measured the percentage of received messages (*reliability*) and the average latency to disseminate a message to all the nodes.

5.1. Simulation Setup. We have used the JiST/SWANS simulator [21] to evaluate the protocols. In JiST/SWANS, nodes use two-ray ground radio propagation model with IEEE 802.11 MAC protocol and 54 Mb/sec bandwidth. Two concurrent transmissions can collide, in which case, the messages will not be received by some of the nodes. The collision may occur without the transmitting node detecting the problem, a phenomenon known as the hidden terminal problem. The transmission range was set to roughly 200 meters. The nodes were placed at uniformly random locations in a square area of $500 \times 500\,\text{m}^2$ in some measurements and $1000 \times 1000\,\text{m}^2$ in others. Mobility was modelled by the Random-Waypoint model [22] with the speed of movement picked from the range 1–10 m/s. (Notice that by setting the minimal speed to a positive value, we avoid the known problem of Random-Waypoint of eventually ending up with a static network.) For high mobility, we set the speed from the range 20–50 m/s. In our simulations, we vary the number of nodes, the number of broadcasting nodes (senders), and size of data messages. In every run, each broadcasting node sends 10 messages and then after a cool-down period the simulation is being terminated. Namely, in every round there are $10 \times n$ messages, where n is the number of nodes. Each data point was generated as an average of 10 runs.

In addition to our locBcast protocol, we have simulated the following protocols:

Flooding. This is the basic flooding protocol where every node transmits a packet upon receiving it for the first time.

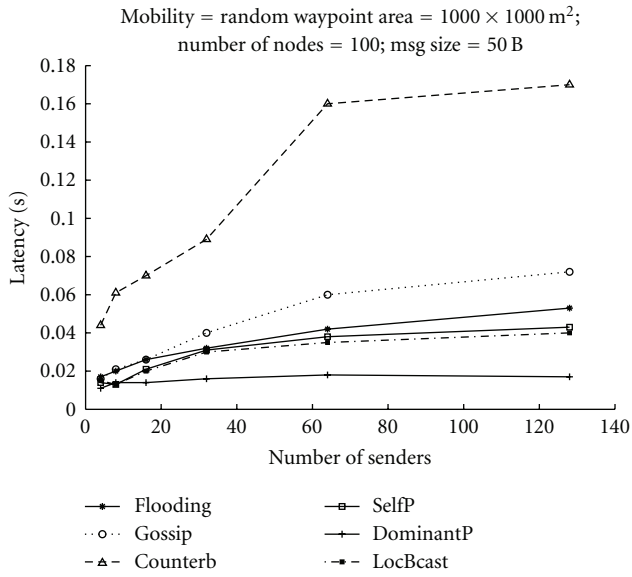

Figure 16: Latency versus number of senders.

Probabilistic Flooding (Gossip). In this protocol, when a node M_i receives a packet for the first time, it transmits it with probability p and with probability $1-p$ it discards the packet.

Counterb. This is the counter-based scheme presented by Tseng et al. [5]. In counterb, upon receiving a new packet by M_i, M_i initiates a counter with a value of one and sets a random assessment delay (RAD), which is randomly chosen between 0 and Tmax seconds. During the RAD, the counter is incremented by one for each redundant packet received. If the counter is less than a threshold value when the RAD expires, the packet is transmitted. Otherwise, it is simply dropped.

SelfP. This is the self-pruning protocol presented by Lim and Kim [2]. This protocol requires that each node have knowledge of its 1-hop neighbors, which is obtained via periodic "Hello" packets. A node includes its list of known neighbors in the header of each broadcast packet. A node receiving a broadcast packet compares its neighbor list to the senders neighbor list. If the receiving node would not reach any additional nodes, it refrains from transmitting; otherwise the node transmits the packet.

DominantP. This is the dominant pruning protocol presented by Lim and Kim [2]. dominatP requires 2-hop neighbor knowledge. Here, the sender determines the forwarding nodes from its 1-hop neighbors. Only those chosen nodes are allowed to transmit. When a node receives a packet, it checks the header to see if its address is part of the list.

5.2. Results. We conducted our simulation on different environments to demonstrate most of the different environments in MANETs. In the following section, we describe each environment and show the results.

5.2.1. Small Area with High Load. We start with the challenge environment of small area with highly message transmission. We consider here the behavior of our protocol comparing with the others.

Figure 6 shows the percentage of received messages (reliability) versus the number of nodes with high load of data messages, where the number of senders is 20. The area size is $500 \times 500\,m^2$ and the message size is 50 B. Notice that when the number of nodes is only 4, the network is disconnected, which is why all protocols achieve low delivery ratio. As the number of nodes increases and the network becomes connected, so the reliability also improves. Yet, as the network becomes too dense, collisions occur, thereby reducing the reliability of all protocols. Still, counterb degrades the most graceful, since it has built-in mechanism to adjust its overhead to density. Moreover, counterb includes a jitter mechanism, by which nodes delay their transmissions for a random period of time, which further helps to reduce collisions. Next are locBcast and dominantP, which build a fairly efficient dissemination tree. Their main overhead comes from their control message overhead (the "Hello" messages). locBcast is better here because it imposes less overhead than dominantP. Flooding, gossip, and selfP perform the worst, since they are too verbose, thereby generating too many collisions.

The latency of the various protocols, under the same settings as above, is depicted in Figure 7. As can be seen, couterb has the worst latency. This is due to the same jitter that enables counterb to be so reliable. On the other hand, dominantP and locBcast exhibit the shortest latency, as they both build highly efficient dissemination trees.

Figure 8 presents the total number of transmitted messages versus the number of nodes, using the same settings as above. As expected, counterb and locBcast generate the smallest number of messages transmissions, followed closely by dominantP. On the other hand, flooding, gossip, and selfP generate a large number of message transmissions. Notice that although dominantP has the full knowledge of the 2-hop neighborhood, locBcast generates fewer transmissions than dominantP. This is because in locBcast, the sender does not decide about all the forwarding nodes. Rather, it has a candidate set where a receiving node can make a better decision about whether to transmit or not.

Similarly, Figure 9 shows the total number of generated control messages versus the number of nodes. Flooding and gossip do not send any control messages. Since every control message needs to propagate within the 2-hop distance, dominantP has the highest number of control messages. Counterb and locBcast have few control messages to indicate the neighbors. Recall that in selfP control messages are generated periodically.

5.2.2. Small Area with Low Load. In this environment, we considered an area of $500 \times 500\,m^2$ and low message transmissions. We present here another experiment with the same settings described above. We vary the density of the network (number of nodes) while the data message load is low. Figure 10 presents the reliability versus number of nodes

with low load, where the number of senders is only five. We can see that every protocol achieves high reliability. This is because under low load, collisions between messages are very rare. Nevertheless, counterb and locBcast achieve the highest reliability among the others.

Now we vary the message size to examine its effect on the results.

Figures 11 and 12 present the reliability and latency versus message size, respectively. In this simulation, we have 30 mobile nodes in an area of $500 \times 500\,m^2$, and the number of senders is 5. We can see here that, especially for large messages, our protocol achieves the best reliability and latency results among all the protocols. As in the other simulations, counterb has the worst latency. Notice here that dominantP achieves good results since there is a small number of nodes, so its control messages rarely collide.

Additionally, in this environment, we would like to check the effect of high mobility on the results. Figures 13 and 14 present the reliability and latency versus the number of nodes with high mobility, where every node moves within a speed of 20–50 m/s. The message load is low and the message size is 50 B. We can see here that counterb and locBcast achieve the highest reliability. Regarding latency, locBcast achieves the lowest latency after dominantP. These results indicate the benefit of locBcast in the high-mobility environments, where every node can extrapolate the location of its neighbors without the need for control messages.

5.2.3. Large Area with Low Load. We now consider the environment of $1000 \times 1000\,m^2$ with low message transmission.

Figure 15 presents the reliability versus the number of senders with 100 mobile nodes in an area of $1000 \times 1000\,m^2$, where the message size is 50 B. Regardless of the number of senders, dominatP has the worst reliability. This is because when the network is dense, dominatP generates many large control messages that mostly collide. Therefore, every node has an incomplete view of its 2-hop neighbors. All other protocols achieve much better results, with counterb being the best, followed closely by locBcast.

With the same settings of Figures 15 and 16 presenting the latency versus the number of senders, as expected, counterb has the worst latency where every message is delayed before any further transmission. On the other hand, dominantP has the lowest latency with the bad reliability results. This is because the small number of transmissions in the incomplete view of its 2-hop neighbors. As in the reliability case, locBcast achieves the second best latency (after dominantP). Therefore, in many different simulations we notice that our protocol, locBcast, obtains a good tradeoff between reliability, latency, and message overhead compared to the other protocols.

6. Conclusions

In this paper, we have investigated the benefits of utilizing location information in dissemination protocols for MANETs. In particular, we have presented a new approach that utilizes positioning information in order to generate an efficient dissemination tree while only maintaining 1-hop neighborhood information.

Our simulation results have shown that this approach obtains a good tradeoff between reliability, latency, and message overhead. In particular, locBcast obtains good results in all these metrics. In contrast, for any other scheme we have compared with, if that scheme excels in one of these metrics, it performs poorly in another metric. Moreover, the performance of locBcast is especially in opportunistic networks, where mobility is high, network may be dense, and message size can be larger than 1 KB.

References

[1] K. M. Alzoubi, P.-J. Wan, and O. Frieder, "Message-optimal connected dominating sets in mobile ad hoc networks," in *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp. 157–164, Lausanne,Switzerland, 2002.

[2] H. Lim and C. Kim, "Multicast tree construction and flooding in wireless ad hoc networks," in *Proceedings of the the ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM '00)*, pp. 61–68, Boston, Mass, USA, 2000.

[3] J. Wu and H. Li, "On calculating connected dominating set for efficient routing in ad hoc wireless networks," in *Proceedings of the 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DialM '99)*, pp. 7–14, 1999.

[4] Z. Haas, J. Halpern, and L. Li, "Gossip-based ad hoc routing," *IEEE/ACM Transactions on Networking*, vol. 14, no. 3, pp. 479–491, 2006.

[5] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," *Wireless Networks*, vol. 8, no. 2-3, pp. 153–167, 2002.

[6] Y. Sasson, D. Cavin, and A. Schiper, "Probabilistic broadcast for flooding in wireless mobile ad hoc networks," in *Proceedings of the the IEEE Wireless Communicationand Networking Conference (WCNC '03)*, pp. 1124–1130, March 2003.

[7] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, MIT Press, 1986.

[8] B. Williams and T. Camp, "Comparison of broadcasting techniques for mobile ad hoc networks," in *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp. 194–205, Lausanne, Switzerland, June 2002.

[9] A. Qayyum, L. Viennot, and A. Laouiti, "An efficient technique for flooding in mobile wireless networks," Tech. Rep. RR-3898, INRIA, 2000.

[10] W. Peng and X. Lu, "AHBP: an efficient broadcast protocol for mobile ad hoc networks," *Journal of Computer Science and Technology*, vol. 16, no. 2, pp. 114–125, 2001.

[11] J. Sucec and I. Marsic, "An efficient distributed network-wide broadcast algorithm for mobile ad hoc networks," Tech. Rep. 248, Rutgers University, 2000.

[12] V. Drabkin, R. Friedman, G. Kliot, and M. Segal, "Reliable probabilistic dissemination in wireless ad hoc networks," in *Proceedings of the 26th IEEE International Symposium on Reliable Distributed Systems (SRDS '07)*, pp. 13–22, Beijing, China, 2007.

[13] A. El Fawal, J.-Y. Le Boudec, and K. Salamatian, "Multi-hop broadcast from theory to reality: practical design for ad hoc

networks," in *Proceedings of the 1st International Conference on Autonomic Computing and Communication Systems*, Rome, Italy, October 2007.

[14] V. Lenders, G. Karlsson, and M. May, "Wireless ad hoc podcasting," in *Proceedings of the IEEE Communications Society Conference on Sensor, Mesh, and Ad Hoc Communications and Networks (SECON '07)*, San Diego, Calif, USA, June 2007.

[15] S. H. Shah and K. Nahrstedt, "Predictive location-based QoS routing in mobile ad hoc networks," in *Proceedings of the IEEE International Conference on Communications (ICC '02)*, New York, NY, USA, April 2002.

[16] B. Hughes, R. Meier, R. Cunningham, and V. Cahill, "Towards real-time middleware for vehicular ad hoc networks," in *Proceedings of the 1st ACM Workshop on Vehicular Ad Hoc Networks*, Philadelphia, Pa, USA, October 2004.

[17] Y. Sun, E. M. Belding-Royer, X. Gao, and J. Kempf, "Real-time traffic support in large-scale mobile ad hoc network," in *Proceedings of BroadWIM*, San Jose, Calif, USA, October 2004.

[18] R. Friedman and G. Kliot, "Location services in wireless ad hoc and hybrid net-works: a survey," Tech. Rep. CS-2006-10, Department of Computer Science,The Technion, 2006.

[19] L. Cerdà et al., "A reservation scheme satisfying bandwidth QoS constrains for ad-hoc networks," in *Wireless Systems and Mobility in Next Generation Internet*, vol. 3427 of *Lecture Notes in Computer Science*, pp. 176–188, Springer, Berlin, Germany, 2005.

[20] IETF Mobile Ad-hoc Networks Working Group, "RFC 5148: Jitter considerationsin Mobile Ad Hoc Networks (MANETs)".

[21] JiST/SWANS Java in Simulation Time / Scalable Wireless Ad Hoc Network Simulator, http://jist.ece.cornell.edu./ .

[22] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*, T. Imielinski and H. Korth, Eds., vol. 353, Kluwer Academic Publishers, 1996.