ADAPTIVE MOBILITY-ASSISTED DATA DISSEMINATION IN MOBILE DISASTER/RECOVERY ENVIRONMENTS

Long Vu and Klara Nahrstedt
Department of Computer Science, University of Illinois at Urbana-Champaign, IL 61801
Email:{longvu2,klara}@cs.uiuc.edu

ABSTRACT

Efficient information dissemination over Mobile Ad hoc Networks (MANET) for urban Disaster/Recovery (D/R) missions is emerging as a very challenging and important research problem. In this paper, we present an adaptive mobility-assisted data dissemination framework as a solution for Disaster/Recovery missions. Our novel framework is based on “Importance Score” of D/R messages to: (1) optimize the number of disseminations due to the bandwidth limitations in MANET, and (2) discard invalid D/R messages due to memory space limitation on mobile devices. The corresponding “Importance Score” function, a linear combination of priority and deadline, ranks D/R messages according to metrics that obtain maximal area coverage and minimal delay in D/R dissemination. Once the D/R messages are ranked, our adaptive mobility-assisted data dissemination protocol broadcasts the top-k, tuning broadcast period according to network conditions. To ensure performance efficiency, we estimate times-to-send (TTS) to limit unnecessary transmissions. Our experimental results show that the presented framework with corresponding algorithms and protocols efficiently utilizes network bandwidth and node memory space (i.e. memory space at a mobile node), while achieving information coverage and delay objectives.

1. INTRODUCTION

The terrorist attack on the World Trade Center on September 11, 2001, has drawn increasing attention to improving rescue efforts following a disaster. Among proposed technologies, MANET is an emerging and promising technology for rescue forces due to the lack of communication infrastructure after an urban disaster. When the urban infrastructure collapses after a disaster (e.g. earthquakes, terror attacks, etc.), rescue teams (police, medical, fire, etc.) come and form an ad hoc wireless network. In this network, people broadcast various kinds of information such as emergency notifications, alerts, etc. On one hand, a message has its own delay and coverage constraints. On the other hand, network communication fully depends on available network bandwidth and the number of messages a mobile node can carry. Therefore, disseminating D/R messages efficiently to obtain delay and coverage objectives under limitations of network bandwidth and node memory space remains a challenging research problem.

Designing an efficient, reliable, and robust data dissemination protocol in MANET is challenging because of many reasons. First, the protocol must be efficient in terms of network bandwidth consumption and use of node memory space. Second, a data dissemination protocol must be reliable so that D/R messages can reach almost the entire network under packet loss, network partition, and transmission failure. Third, the protocol must be able to obtain delay and coverage objectives. In other words, delivered messages must cover almost the entire network by their deadlines. Last but not least, the protocol should be robust because MANET network is frequently or permanently partitioned and thus data dissemination protocols may fail or incur high overhead.

Previous work on data dissemination protocols in wireless network falls into two categories: (1) flooding-based protocols [3, 6], and (2) mobility-assisted protocols [1, 2, 9, 10]. In the first approach, different flooding protocols have been proposed, such as Selective/gossip [7], Hyper [8], and Self-pruning [4]. These methods work with dense networks and require large memory space at mobile nodes. Moreover, their performance significantly degrades with sparse or partitioned networks. Another flooding-based protocol is opportunistic dissemination, which prioritizes and broadcasts spatio-temporal messages based on their relevant scores [11]. Nevertheless, this protocol does not focus on delay constraints, which is crucial in D/R scenarios. In the second approach, mobility-assisted data dissemination protocols are designed for sparse or partitioned networks. A two-hop relay scheme and its variations are proposed in which the sender selects its nearest neighbors as relay nodes. These relay nodes forward the message to the receiver if they
are in the transmission range of each other [1, 5]. Despite its low overhead, this scheme assumes a uniformly distributed network and an unlimited memory space on mobile nodes. Another epidemic data dissemination scheme provides coverage/delay guarantees for one message [2]. However, this scheme does not address limitations of network bandwidth and node memory space. In summary, existing data dissemination schemes focus either on coverage/delay guarantees or network bandwidth and node memory space, but not at the same time.

In this paper, we present a novel efficient data dissemination framework over MANET for urban Disaster/Recovery missions. In particular, we present an adaptive mobility-assisted data dissemination framework which is based on “Importance Score” of D/R messages to: (1) optimize the number of disseminations due to the bandwidth limitation in MANET, and (2) discard invalid D/R messages due to memory space limitation on mobile devices. The corresponding “Importance Score” function ranks D/R messages according to metrics that obtain maximal area coverage and minimal delay in D/R dissemination. Once the D/R messages are ranked, our adaptive mobility-assisted data dissemination protocol broadcasts the top- \( k \) D/R messages, tuning broadcast period according to node density and node speed. We also estimate times-to-send TTS of a message to limit unnecessary transmissions. Our experimental results show that the presented framework achieves coverage and delay objectives under limitations of network bandwidth and node memory space.

The rest of this paper is organized as follows. Section 2 introduces our design objectives, models and overview of our presented framework. Next, details of the framework are presented in sections 3, 4, and 5. Section 6 evaluates our framework based on simulation results. Finally, we conclude the paper in section 7.

### 2. DESIGN OBJECTIVES, MODELS AND SYSTEM ARCHITECTURE

#### 2.1. Design Objectives

The first design objective is Delivery Delay. Essentially, D/R messages should be disseminated to almost the entire network before their deadlines. The second design objective is Network Coverage. D/R messages should be disseminated to almost all nodes under dynamic and partitioned network. The third design objective is Performance Efficiency. In other words, the protocol should be able to organize and disseminate a large number of D/R messages efficiently under limitations of network bandwidth and node memory space.

#### 2.2. Network Model

After a disaster, rescue teams come and form an ad hoc wireless network from their limited coverage wireless mobile devices, which are the main agents to store, carry, and broadcast D/R messages. In this paper, we assume that mobile wireless nodes follow Random Way Point mobility model. Although there is no perfect mobility model for all scenarios in wireless networks, we believe that Random Way Point is an acceptable assumption because in Disaster/Recovery scenarios distribution of nodes are heterogenous and their movements are relatively random. The random movement results in frequently or permanently partitioned network. However, the random movement itself helps disseminate messages more quickly to the entire network because nodes can buffer, carry and forward messages.

#### 2.3. Data Model

In this paper, a message is an information unit disseminated from node to node. Messages can be emergency notifications, survival alerts, environmental hazard notifications, etc. Table 1 shows the format of message \( m \), which is either stored at mobile nodes or disseminated in the network. Notice that all six attributes of \( m \) are stored at mobile nodes, but only the first four attributes are encapsulated and broadcasted. In particular, \( id \) uniquely identifies messages, \( dl \) -the relative deadline-specified valid period of \( m \) to the system. \( pri \) -priority-differentiates types of messages. For example, \( m \) has a high priority if it is an emergency message and \( m \) has a low priority if it is a normal message. \( dup \) is the number of duplicated messages and \( arTime \) is the time at which \( m \) arrives at node \( n \). Message \( m \) is considered invalid after its deadline expires. \( n \) updates \( m \)'s relative deadline (valid period) when it broadcasts \( m \) as follows:

\[
m.dl = m.dl - (cTime() - m.arTime)
\]

In equation 1, \( cTime() \) returns current time at node \( n \) and \( cTime() - m.arTime \) is the time period \( m \) stays at \( n \). Due to clock drift among nodes in the network, there exits some \( \epsilon \) error (perhaps in millisecond) results from \( cTime() \) function. However, because the unit of relative deadline is minute, clock drift can be negligible.

In our context, message \( m_1 \) is considered more important than \( m_2 \) to the network (hence \( m_1 \) has a higher Importance Score than \( m_2 \)), if broadcasting \( m_1 \) prior to \( m_2 \) improves our design objectives. Specifically, Importance Score of a message is a linear combination of its priority and deadline. We further present Importance Score in section 4.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>identifier</td>
<td>Unique Number</td>
</tr>
<tr>
<td>dl</td>
<td>relative deadline</td>
<td>Valid period</td>
</tr>
<tr>
<td>pri</td>
<td>priority</td>
<td>Low/high priority</td>
</tr>
<tr>
<td>cnt</td>
<td>content</td>
<td>Text of ( m )</td>
</tr>
<tr>
<td>dup</td>
<td># of duplications</td>
<td>Integer Value</td>
</tr>
<tr>
<td>arTime</td>
<td>arrival time</td>
<td>Time</td>
</tr>
</tbody>
</table>
In this paper, Time-to-Send (i.e., $TTS$) of a message $m$ is the number of broadcasts nodes in the network can perform on $m$. $TTS$ is analogous to TTL in TCP/IP and is a crucial parameter because overestimating $TTS$ results in redundant messages and waste of network bandwidth. In contrast, underestimating $TTS$ degrades delivery coverage of $m$. In section 5, we present an algorithm to estimate $TTS$.

2.4. System Architecture Overview

Figure 1 shows the system architecture of the adaptive mobility-assisted data dissemination framework with three main components: Mobility-assisted Data Dissem- nator, Adaptive Estimator, and Disaster/Recovery Message Manager. To shorten the notation, we henceforth use Dissemnator, Message Manager, and Estimator for corresponding components.

To begin with, Disseminator receives messages (by Receiver) and forwards them to Message Manager. Disseminator also updates neighbor list and periodically broadcasts the top-$k$ messages. When receiving messages from Disseminator, Message Manager uses its Operators to update, organize, and delete messages in its Message Collections. Message Manager also uses Importance Score Calculator to calculate the Importance Score of messages, rank them and send the top-$k$ messages. Estimator adaptively estimates broadcast period $T$ and $TTS$ according to network parameters. These estimations determine when Disseminator broadcasts messages and how many times a message is broadcasted so that we can utilize network bandwidth and node memory space. Next, we discuss in detail these three main components.

3. MOBILITY-ASSISTED DATA DISSEMINATOR

Disseminator is the interface of our architecture with lower layer. It has three sub-components: Receiver, Neighbor Manager, and Sender. The Receiver essentially receives message from lower layer, updates its arTime, and forwards the message to Message Manager. Neighbor Manager updates and refreshes the neighbor list, which is used by Estimator to estimate broadcast period $T$ (Section 5). Sender updates the top-$k$ message deadlines (Formula 1) when it broadcasts them.

4. D/R MESSAGE MANAGER

The Message Manager component updates, deletes, and ranks messages so that the top-$k$ messages are sent by the Sender. In particular, Message Manager consists of three sub-components: Message Collections, Importance Score Estimator, and Operators.

4.1. Message Collections

There are three collections: Remote, Local, and Deleted. Remote collection stores messages received from remote nodes, notice that Remote collection has a limited size due to the limitation of node memory space. Local collection keeps messages created by $n$ itself (i.e., local messages) and holds all these messages until their deadlines expire. Local collection preserves fairness between remote and local messages. If there exits only one message collection, all messages will be stored and ranked in this single collection. However, several local messages may be newly-created with lower Importance Score and can be deleted if there is not sufficient space. Therefore, these local messages are not disseminated, resulting in unfairness. Deleted collection is used to avoid message re-propagation. After $n$ deletes message $m$, $m$ should not be disseminated by $n$ after that. Deleted collection only keeps id of deleted messages. Notice that when a new message $m$ arrives, it can be added into Remote collection or deleted depending on available memory and its Importance Score. Incase memory is full, $m$ can only be added into Remote collection if its Importance Scores is higher than that of the least important message of this collection. Messages in Remote and Local collections are ranked and the top-$k$ are broadcasted.

4.2. Importance Score Calculator

Given a message $m$, its Importance Score - $Im(m)$-is computed as follows:

$$Im(m) = Pri(m) - Time2Dline(m) - Dup(m)$$ (2)

In equation 2, $Pri(m)$, $Time2Dline(m)$, and $Dup(m)$ are functions as shown in Table 2. In particular, MAXPRI - maximal priority, MAXDL - maximal deadline, MAXDUP - maximal duplicated messages, are constants representing the maximal values of corresponding attributes. For example, $MAXPRI = 2$ (low and high priority), $MAXDL = 60$ minutes, and
Table 2: Three functions used in Importance Score calculation for a message \( m \). They all return real values in \([0,1]\).

\[
\text{MAXDUP} = 50. \quad m.pri \text{ is priority of } m, \text{ so that } m.pri/\text{MAXPRI} \text{ is in the range } [0,1]. \text{ Likewise, } (m.d - \text{cTime()} - \text{pTime()})/\text{MAXDL} \text{ (Formula 1) and } m.dup/\text{MAXDUP} \text{ are also in the range } [0,1]. \text{ These normalized values equalize roles of factors in } Im(p). \text{ Function } \text{cTime()} \text{ returns current time at node } n \text{ and } \text{pTime()} \text{ returns approximate propagation delay to avoid late disseminated messages (i.e., messages are disseminated to nodes after they expire).}
\]

By using formula 2, we intuitively prefer messages with higher priority, tighter deadline, and fewer duplications. This Importance Score considerably improves message dissemination as shown in our evaluation (Section 6).

![Figure 2: Initial \( |T| = 2r/v \)]

\[
\alpha = 0.2. \quad \text{If the ratio } |A|/|B| \text{ is greater than } \alpha, \text{ then we shorten the broadcast period proportional to } (\alpha - |A|/|B|). \text{ In contrast, if } |A|/|B| \text{ is smaller than } \alpha, \text{ then period broadcast should be lengthened, again proportional to } (\alpha - |A|/|B|). \text{ By applying formula 3, we can smooth out short-term fluctuations of broadcast period and tune it adaptively. For example, if node } n \text{ moves into a dense network and its } L_{i-1} \text{ changes significantly, then we shorten } T_i \text{ so that } n \text{ can broadcast more to its dense vicinity. In contrast, if } n \text{ is in a sparse node density location or it moves very slowly, and hence its } L_{i-1} \text{ changes slightly, then we lengthen } T_i \text{ so that } n \text{ can save network bandwidth. Figure 2 shows a node } n \text{ moving from } A \text{ to } B \text{ through } C, \text{ with } AC = BC = r. \text{ Because we expect that any time we send a message, the covered area of the message is maximized. Therefore, initial value of } T \text{ can be } 2r/v. \text{ Algorithm 1 applies formula 3 to obtain } |T_i| \text{ from } |T_{i-1}|.
\]

**Algorithm 1** Estimate \( |T_i| \) for node \( n \)

\[
\begin{align*}
\text{INPUT:} & \quad \alpha, |T_{i-1}|, A, B \\
\text{BEGIN} & \\
& B = \text{List of unique neighbors of } n \text{ up to beginning of } T_i; \\
& A = \text{List of new-unique neighbors of } n \text{ during } T_{i-1}; \\
& |T_i| = (\alpha - |A|/|B|) \times |T_{i-1}| + |T_{i-1}|; \\
& \text{return } |T_i|; \\
\text{END}
\end{align*}
\]

5. ADAPTIVE ESTIMATOR

In the following sections, we present analysis and corresponding estimations of broadcast period \( T \) and time-to-send \( TTS \). Table 3 lists all parameters used by Estimator to estimate \( T \) and \( TTS \).

### 5.1. Broadcast Period Estimator

In reality, due to the heterogeneity of MANET, node density and node speed are not uniformly distributed. For example, a node \( n \) can move from a dense location to a sparse location or it can move with different speeds in different periods. Therefore, using a constant broadcast period is not suited for this realistic situation [2].

In this section, we present a formula to adaptively estimate the length of broadcast period \( T_i \) as follows:

\[
|T_i| = (\alpha - |A|/|B|) \times |T_{i-1}| + |T_{i-1}|
\]

In equation 3, \( \alpha \) is a constant in the range \((0,1)\), which specifies the threshold to estimate \( T_i \) from \( T_{i-1} \) (e.g.,

### 5.2. Times-to-Send Estimator

In this section, we present how Estimator estimates the \( TTS \) of messages. With a reasonable \( TTS \) value, nodes can terminate message dissemination on time to utilize network bandwidth and node memory space. Notice that \( TTS \) and deadline of a message \( m \) are orthogonal. The former is the number of times nodes can broadcast \( m \). The latter specifies how long \( m \) remains valid to the network. As long as its deadline does not expire, \( m \) is broadcasted. Correspondingly, when \( m \)'s \( TTS \) gets maximal, \( m \) should be deleted. To derive \( TTS \), we adapt analysis from [2] with suitable changes because analysis from [2] is for propagation of one message in the network without message deletion and no limitations of network bandwidth and node memory space.

We assume that \( N \) nodes uniformly distributed in circle area \( IA \) and periodically broadcasts a message \( m \) with a fixed broadcast period \( T \). Further, we assume that during each \( T \), \( p \) is the percentage of message \( m \) deleted by
we have CO joins the network. All subsequent messages created by node in each hexagon (or the average degree of node) is N/H. Let CO(t) be the expected number of nodes, which have received the message m at time t (or the coverage of m at time t). Because CO(t) is non-decreasing, we have CO(t + 1) ≥ CO(t). Our TTS value should be the minimum value δ at which CO(t0 + δT) ≈ N. Next, we derive a proposition to estimate this minimum TTS.

Proposition: The expected number of nodes which have received (or known) a given message at time t0 + (δ + 1)T satisfies CO(t0 + δ + 1)T ≥ (1 - p){N × (1 + e^CO(t0+δT))/H)}

Proof: Let t ⊆ {t0 + δT, t0 + (δ + 1)T}, there exists CO(t0 + δT) nodes knowing message m. The probability that a hexagon with radius r does not contain any nodes in CO(t0 + δT) nodes is (1 - 1/H)^CO(t0+δT). Therefore, at time t0 + (δ + 1)T, the expected number of hexagons which contains at least one node knowing m is H(1 - (1 - 1/H)^CO(t0+δT)). Notice that m gets deleted with probability p at each hexagon. Therefore, among H(1 - (1 - 1/H)^CO(t0+δT)) nodes, there are (1 - p){H(1 - (1 - 1/H)^CO(t0+δT))} nodes broadcasting m at time t. For shorter notation, let Δ be CO(t0 + δT), we have:

CO(t0 + (Δ + 1)T) = (N/H)(H[1 - (1 - 1/H)^Δ])(1 - p)
= (1 - p){N × (1 - 1/H)^Δ}
≥ (1 - p){N × (1 - e^CO(t0+δT))/H)}

Figure 3 shows the relationship between p and TTS under different average degrees and transmission ranges. In this figure, when p increases, TTS increases accordingly. In particular, when p = 0.5, TTS values of all configurations are less than 10. However, when p = 0.8, TTS values increase up to 20. This confirms that a good estimation of TTS allows nodes to terminate dissemination on time and thus utilize network bandwidth and node memory space.

We present algorithm 2 to estimate TTS for a message m. Estimator performs this algorithm only when node n joins the network. All subsequent messages created by n CO have this estimated TTS. When the network population changes significantly, n might need to rerun algorithm 2 to re-estimate a new TTS.

Algorithm 2 Estimate TTS

INPUT
N - number of nodes; p - probability message is deleted
IA = πR^2 - area of interest
H = πR^2/(2.6 × r^2) - number of hexagons covering IA
BEGIN
δ = 0; CO(t0) = 1; TTS=0;
while (CO(t0 + δT) < N) do
CO(t0 + (δ + 1)T) = (1 - p){N × (1 - 1/H)^CO(t0+δT)};
TTS ++; δ += ;
end while
return TTS;
END

6. EVALUATION

6.1. Simulation Settings

We use NS2 as our simulator and the Random Way Point mobility model to simulate node mobility. Random Way Point mobility model is suited for Disaster/Recovery scenarios because in this scenario nodes tend to move randomly. Table 4 shows our simulation settings. Our simulation time is 1000s. In the first 500s, 15 nodes randomly generate 200 messages with size 512 bytes and the interval between messages is 10s. The deadline of messages is generated randomly from 300s to 500s. The maximal sending buffer size SB (or memory size) of each node is 40 messages. TTS is estimated by Algorithm 2. We perform 10 runs of each simulation and plot the average.

In our context, a message is considered to be a Meet Deadline message if it is delivered to at least 90% of nodes in the network before its deadline expires. We also define weighted Importance Score as follows:

\[ w1m(m) = a \cdot Pri(m) - b \cdot Time2Dline(m) - Dup(m) \] (4)

In equation 4, a and b are weights denoting either priority or deadline is preferred. Depending on scenarios, priority can be set with higher weight than that of deadline,
Equal preference scheme has more “balanced results”. We have equal preference scheme if \( a = b \). Next, we present our results with Meet Deadline metric, weighted Importance Score, and the broadcast period \( T \).

6.2. Fixed broadcast period \( T \)

First, we evaluate our protocol with fixed broadcast period \( T \) (i.e. \( T = 2r/v \)). Figure 4 shows the relationship between Meet Deadline metric and transmission ranges with different \( k/\text{SB} \) ratios. From this figure, we can see that Meet Deadline increases if either transmission range or ratio \( k/\text{SB} \) increases. Obviously, with larger transmission range, node covers larger area for each broadcast, and thus, the message is disseminated to the network more quickly. Similarly, as \( k \) increases, node \( n \) can send more messages each broadcast, thus messages get higher chance to be delivered.

Figure 4 also presents the difference between high and low priority messages. Particularly, when the deadline is preferred, Figure 4(a) shows that the number of high priority messages meeting deadlines is from 79% to 89%. Meanwhile, the number of low priority messages (as shown in Figure 4(b)) varies from 12% to 38%. Correspondingly, in case of priority preferred scheme, Figures 4(c) and 4(d) indicate that the number of messages meeting deadlines is from 72% to 92%, and from 10% and 31%, for high and low priority messages, respectively.

The differences between priority preferred scheme and deadline preferred scheme are also shown in Figures 4(b) and 4(d). In particular, the number of high priority messages in Figure 4(b) that meet deadlines is higher than that of Figure 4(d). This is because when the deadline is preferred, the message with tighter deadline would be forwarded prior to other messages. This is especially true for low priority messages, which are usually ignored by the priority preferred scheme.

Equal preference scheme has more “balanced results”, as shown in Figures 4(e) and 4(f). Specifically, high priority messages have the Meet Deadline metric from 60% to 85%, and that of low priority messages is from 20% to 60%. This is because the equal preference scheme treats messages equally in terms of deadline and priority.
In conclusion, under limitations of network bandwidth, node memory space, and deadline constraint of messages, Importance Score differentiates, prioritizes high priority messages and disseminates them prior to low priority messages. This obviously improves performance of the system because we would prefer that high priority messages cover the network by their deadlines.

6.3. Tunable broadcast period $T$

Figure 5 shows that tunable $T$ scheme has better results than fixed $T$ scheme. Particularly, the Meet Deadline metric increases up to 98% for $k/SP = 0.5$ and $r = 8$. This is because tunable $T$ scheme allows nodes to adjust broadcast period according to node density and node speed. Therefore, nodes adapt better to network conditions and effectively utilize network bandwidth and their memory spaces. Figure 6 shows the Meet Deadline metric for high priority D/R messages, with different values of $\alpha$. Tunable $T$ scheme always has from 86% to 98% of the Meet Deadline metric. This result is an improvement over fixed $T$ scheme.

In conclusion, tunable $T$ scheme improves the performance of our protocol. In a large-scale network, we believe that this scheme can further impact data dissemination due the heterogeneity of node density, node speed, and node memory space. Obviously, the optimal value of $\alpha$ depends on various parameters such as the node density, node speed, etc. Finding the optimal values of $\alpha$ for a network, thus, is left as our future work.

7. CONCLUSION

To the best of our knowledge, this paper is the first work solving the problem of data dissemination in MANET to obtain delay and coverage objectives under limitations of network bandwidth and node memory space. Our adaptive mobility-assisted data dissemination framework is based on “Importance Score” of D/R messages to optimize the number of disseminations and discard invalid D/R messages. The corresponding “Importance Score” function ranks D/R messages according to metrics that obtain maximal area coverage and minimal delay in D/R dissemination. Once the D/R messages are ranked, our protocol broadcasts the top-$k$ messages, tuning broadcast period according to node density and node speed. We also estimate times-to-send TTS of a message to limit unnecessary transmissions.

Our simulation results show that Importance Score differentiates and prioritizes messages so that more important messages can cover the network. Tunable $T$ scheme further improves the performance of our framework. In the future, we plan to investigate this framework with other mobility models, different network configurations (node distribution, node speed variation, etc.), and find the optimal values of $\alpha$ for other networks.

References


