Journal of Information Processing Vol.22 No.2 106–117 (Apr. 2014)

[DOI: 10.2197/ipsjjip.22.106]

Regular Paper

Disaster Information Collection with Opportunistic Communication and Message Aggregation

Jovilyn Therese B. Fajardo[1,a)] Keiichi Yasumoto[1,b)] Naoki Shibata[1,c)] Weihua Sun[1,d)] Minoru Ito[1,e)]

Received: May 14, 2013, Accepted: October 9, 2013

Abstract: When a large-scale disaster strikes, the communications infrastructure is usually unavailable. However, accurate and timely information of the disaster area is important because first responders rely on this information to assess the situation in the affected area and to provide an effective and immediate assistance. In this paper, a data collection method from an area of interest (*AoI*) within the disaster zone is proposed that uses the mobile phones of the people to serve as sensing nodes. In order for maximum *AoI* coverage to be achieved while minimizing delay, we propose a disruption tolerant network (DTN)-based data aggregation method. In this method, mobile phone users create messages containing disaster-related information and merge them with their respective coverage areas resulting in a new message with the merged coverage. The merging or aggregation of multiple messages will reduce message size and minimize the overall message collection delay. However, simply merging the messages can result in duplicate counting thus, to prevent this, a Bloom filter is constructed for each message. Also, to reduce further the message delivery time, the expected reaching time of a node to its destination is introduced as a routing metric. Through computer simulation with a real geographical map, the proposed method achieved a 9.7% decrease in information collection delay confirming that the proposed method achieved a smaller delay with a smaller number of total exchanged messages in collecting disaster information covering the *AoI* than epidemic routing.

Keywords: delay tolerant network, disaster communication system, information collection

1. Introduction

Based on the report released by the United Nations International Strategy for Disaster Reduction, disasters had a large economic and human impact for the past 12 years, amounting to 1.3 trillion dollars (USD) in damages with 2.7 billion people affected and 1.1 million people killed. For the previous years, three major disasters occurred — the Haiti earthquake, the Pakistan flooding, and the Great East Japan earthquake and tsunami. During these disasters, the telecommunications infrastructure is damaged as in the case during the Great East Japan earthquake and tsunami. The unavailability of the communication network caused problems such as the transportation of food, gas, and rescue materials over some parts of Japan. Situational awareness over the disaster area is also at its lowest point but situational information is important for first responders to provide an effective and immediate response. Thus, even with the unavailability of the communication infrastructure, an efficient communication system is still needed to relay information between survivors and rescue teams in the shortest possible time and with maximum coverage of the area of interest (*AoI*). In an infrastructureless environment such as a disaster area, a mobile ad-hoc network serves a valuable role as a means for communication.

In a mobile ad-hoc network, the nodes are not constantly connected to each other, so information cannot be consistently delivered from one node to another, affecting network reliability. In networks where an end-to-end routing path between nodes is not guaranteed, a delay/disruption tolerant network (DTN) architecture can be utilized [10]. In this type of architecture, information is delivered via a store-carry-and-forward approach wherein information is temporarily stored at intermediate nodes for eventual delivery to the destination node. While this approach is expected to incur a certain delay in sending information, there is still a need to minimize this delay, especially in time-constrained networks such as a network in the area of a disaster. Current studies on the use of DTNs for the collection of information in a disaster area either deploy additional message ferries (e.g., unmanned aviation vehicles, robotic insects) to the disaster zone [20] or use the personal devices (e.g., smartphones) of the affected people [8]. However, setting up of additional utilities takes time and there is a surge in network traffic because of the large amounts of data generated by the people during the occurrence of a disaster. One particular study examined the role of *Twitter* in disseminating and sharing crisis information during the 2011 South East Queensland floods. Their report states that 1,100 tweets (Twitter messages) were created per hour during the onset of the floods [3]. Thus, there are a number of studies addressing the issue on network congestion in DTNs during a disaster. One of which introduces a

[1] Graduate School of Information Science, Nara Institute of Science and Technology, Ikoma, Nara 630–0192, Japan
[a)] jovilyn-f@is.naist.jp
[b)] yasumoto@is.naist.jp
[c)] n-sibata@is.naist.jp
[d)] sunweihua@is.naist.jp
[e)] ito@is.naist.jp

DTN architecture for an efficient disaster communication [7] and another study focuses on eliminating redundancies in disaster-related content [19] for fast message delivery to the disaster zone.

In this paper, we propose a data aggregation method using DTN to provide timely coverage of the *AoI* within the disaster area by collecting disaster-related information in the shortest possible time. The *AoI* is determined by the query sender, who is a user in the city hall or in an emergency operations center. People with mobile phones in the vicinity of the *AoI* serve as nodes for the DTN. They create disaster-related messages at various places in the *AoI*. They also collect messages by exchanging them among other nodes through short-range wireless communication such as Bluetooth or WiFi, in which the collected messages are delivered to the query sender. However, due to the limited data transfer capacity of a DTN and the surge in network traffic in the disaster area, it is not possible to quickly collect all the messages created in the *AoI* with DTN. Thus, in the proposed method, we apply data aggregation to reduce data size by merging individual messages into one message using aggregation functions like max, min, sum, count, and average. The reduction of data size allows the message to traverse the network faster minimizing the time for information delivery. In the proposed data aggregation, each message covers a specific area within the *AoI* and the aggregated message contains the merged coverage areas of the individual messages. However, during aggregation, message duplication may be present in the aggregated message especially that users may create and share the same information on a location resulting in duplicate information. To prevent the aggregation of duplicate information, a Bloom filter is constructed for each created message (atomic or raw message). The Bloom filter consists of an n-bit array that maps the location information of a message to one of the array positions and sets the corresponding array position to 1. Before aggregating messages, their corresponding Bloom filters are checked and messages with Bloom filters having the same set of array positions equal to 1 or partly overlapping array positions equal to 1 are not aggregated. The use of Bloom filter ensures that duplicate messages are not aggregated resulting in the accuracy of the aggregation results. Moreover, the aggregation granularity metric, which is the maximum area size covered by each aggregated message, is introduced to ensure the resolution of the collected information. Messages are only aggregated if the merged coverage area of the messages is less than the aggregation granularity. To reduce further the message delivery time, the expected reaching time of a node to its destination is also used in determining which node to forward the stored message. Information is opportunistically forwarded to the node with the lower expected reaching time.

We conducted computer simulations with a real geographical map and compared the proposed method to epidemic routing in terms of information collection delay for different number of mobile nodes. As a result, we confirmed that the proposed method achieved a lower latency than epidemic routing with limited flooding (called epidemic routing limited, hereafter) with varying number of mobile nodes present in the disaster area. Based on their average latencies, the proposed method achieved a 9.8%, 14.9%, 9.2%, 5.9%, and 9.7% decrease in information collection delay than epidemic routing limited using 200, 400, 600, 800, and 1,000 mobile nodes, respectively. Also, based on the cumulative distribution graphs of the proposed method and epidemic routing limited, 70% of all runs take less than t = 2,000 to attain 80% coverage of the *AoI* with 1,000 mobile nodes using the proposed method while only 50% of all runs take less than t = 2,000 with the same parameters using epidemic routing limited. From t = 0 to 2,000, there is a 40% increase in the number of runs that achieved 80% coverage of the *AoI* using the proposed method compared to epidemic routing limited further proving the effectiveness of the proposed method in collecting disaster information quickly.

The rest of this paper is organized as follows. Section 2 describes related literature to information collection in disaster areas, while Section 3 describes the data aggregation problem for disaster areas. Section 4 shows the proposed algorithm based on a greedy approach using the expected reaching time of a node to the sink. Section 5 shows the results achieved by simulation using the criteria for evaluating the proposed algorithm and finally, Section 6 concludes the paper.

2. Related Work

2.1 Existing Work

In disaster scenarios, it is of utmost importance that there is access to a wide range of information such as information collected by sensors already deployed in the area. However, in the aftermath of the disaster, these pre-deployed sensors are damaged posing a problem for information collection in the disaster area. In these situations, the mobile phones of people already present in the affected area can be used to gather information since current mobile phones have a rich set of embedded sensors such as accelerometers, global positioning systems (GPS), microphones, and cameras [13]. Mobile phones have been used in data gathering applications such as creating a noise map from sound samples collected using the phone's microphone [16]. They have also been used in achieving maximum coverage of a specific area [1] and in determining the optimum route for rescuers to save the most people in disaster-stricken areas [5].

However, networks in extreme environments, such as disaster areas, are unreliable because of the damage in its communication infrastructure. Network connectivity between users may not be available or there is none at all. In such cases, a disruption tolerant ad-hoc network can be deployed to provide end-to-end connectivity between users. Although a DTN is expected to incur a certain delay in sending the information, there is still a need to minimize the delivery latency, especially in time-constrained networks such as a network in the area of a disaster. Since DTN routing protocols utilize opportunistic contacts among its users, the presence of users in the affected area is important. Thus, a communication network relying on the use of mobile phones carried by people already within the disaster area can be implemented. A network of such kind was proven to be effective in disseminating emergency information to the population in a DTN. However, its effectiveness relied on the number of devices and the maximum allowed delay [4]. In general, DTN routing protocols are classified in two categories: flooding-based protocol and forwarding-

based protocol [14]. In flooding-based protocols, each node sends its messages to the nodes that it comes into contact. A modified Spray and Wait flooding-based protocol known as DTN message priority routing protocol was used in the delivery of messages in a disaster-stricken area [11]. A cluster mobility model designed for a disaster scenario was also evaluated using flooding-based routing protocols [17]. However, in flooding-based protocols, there is a possibility that a node has a copy of all the generated information exchanged in the network adding to the current network traffic. Perhaps, a forwarding-based protocol, such as a geographic routing protocol, is more appropriate to use in a disaster scenario. One such protocol is the Location-aware Message Delivery (LMD) protocol that forwards a single copy of a message to its neighbors based on their forwarding benefit, which is defined as the reduced amount of distance of the message from the destination caused by the forwarding [9]. However, most of the DTN studies focus on achieving a high message delivery ratio and only a few on minimizing delivery latency.

Aside from the delay caused by sparse contact opportunities in a DTN, its delay is also an effect of its limited data capacity and this is affected more greatly in emergency situations because of the increase in network traffic. There is a need then to reduce the data size for transmission and this can be achieved through data aggregation. Data aggregation has been used in wireless sensor networks to reduce a large amount of raw sensor data to a small number of messages [15]. Also, aggregating data at various nodes results in the reduction of the amount of bits transmitted over the network. Hence, delay can be shortened by reducing the demand for wireless resources and expediting data transmissions [12]. Thus, in this paper, we investigate the use of in-network data aggregation in a disruption tolerant mobile ad-hoc network using a participatory sensing framework for information collection in a disaster area. Specifically, instead of sending all the disaster information or raw data to another node, only aggregates or statistics (summaries) of the collected information need to be sent.

2.2 Contribution

This study aims to achieve the minimum delay of data collection in an *AoI* within the disaster area by using data aggregation and employing mobile nodes that are part of a participatory sensing framework. In Ref. [6], we implemented an aggregation-based delay tolerant network using the mobile phones of the people with minimal delivery latency for collecting information from an *AoI* in a disaster scenario. Simulations proved that aggregating messages has an effect in reducing message delivery latency. However, in this study, a more intensive evaluation of the proposed method was done to further prove the efficiency of the proposed algorithm for reducing the delay in collecting the disaster information.

3. Data Aggregation Problem for Disaster Areas

3.1 Target Environment

Consider a disaster scenario during the early period of the recovery phase. In this period, knowledge about the situation in the disaster area is at its lowest point. There is lack of information or perhaps none at all and even if information about the affected area is available, it still needs to be up-to-date. Also, most of the time, the communication infrastructure is destroyed resulting in the difficulty of information collection from the affected area.

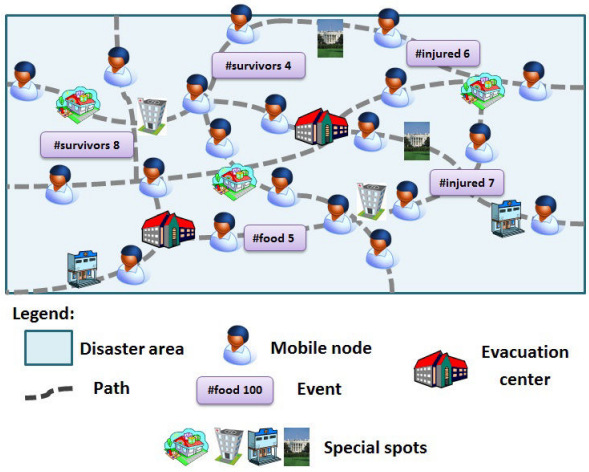


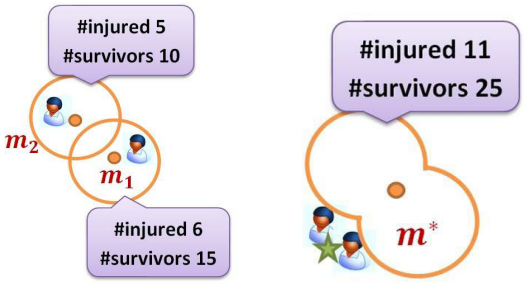
Fig. 1 Example of a disaster area.

As shown in **Fig. 1**, a disaster area denoted by A_d consists of links (roads) between *special spots* (e.g., evacuation centers, hospitals, city halls and so on) and mobile nodes. The set of mobile nodes[*1] existing in A_d is denoted by U. These mobile nodes move towards the nearest evacuation center within A_d, and after arrival at the center, the nodes move only in its vicinity. A stationary node at a special spot can send a snapshot query about a particular area of interest denoted by $AoI \subseteq A_d$. A query q is denoted by $\langle u_0, t_0, AoI, \phi \rangle$, where u_0 is the query node, t_0 is the time when the query is issued, and AoI is the area of interest from which information is to be collected with an aggregation granularity ϕ, the maximum area within which each message is effective. Information is then collected and aggregated from different users who pass through the AoI.

Each node receiving the query and existing in the AoI creates a message m when it finds a disaster-related event within the AoI and its current location is not covered by its retained messages. Each message m has an effective circular area denoted by $m.area$ with radius r_s. The messages are numerical information on survivors, shelter capacity, and available resources such as food, first responders, and utilities. Nodes exchange messages with other nodes upon contact so that the set of collected messages covers the AoI and is delivered to the sink. AoI coverage is determined from the union of the effective areas of the messages received by the sink.

Moreover, data aggregation in each node is based on the aggregation granularity metric ϕ, which specifies the maximum area (or radius) of the effective area of aggregated messages. We adopt our definition of aggregation granularity from the concept of information granularity, which by definition refers to the extent of detail within the information. At lower levels, information exhibits fine granularity and at higher levels, information becomes coarser because it is summarized or aggregate. Thus,

[*1] The terms user, mobile node, and node are used interchangeably.



(a) Atomic Messages (b) Aggregated Message

Fig. 2 Example of aggregating messages.

in our case, a lower aggregation granularity ϕ value means that the query sender needs a more detailed data from the area while a higher aggregation granularity ϕ value means that the sender only wants a summary data from the area. **Figure 2** shows an example of aggregating messages. In Fig. 2 (a), two users create *atomic messages* m_1 and m_2 with information on the number of injured persons and survivors from different areas. *Atomic messages* are nonaggregated messages (or raw messages) created by the user. As shown in Fig. 2 (b), upon contact, the messages of the two nodes are aggregated into one message m^* containing information on the total number of injured persons and survivors from the two atomic messages. The effective areas of the atomic messages are merged and the resulting merged area is the effective area of the aggregated message $m^*.area$. However, a node only aggregates the messages if the effective area of the aggregated messages is less than the aggregation granularity ϕ.

3.2 Assumptions

Each user in U is assumed to have a certain role in the disaster area, which affects its movement within the area. It is assumed that each user falls into a certain type of group in the disaster area: a first responder, an official, or an ordinary person. In disaster areas, there are spots that need urgent attention from the rescuers and these special spots are called *hotspots*. Examples of hotspots are buildings on fire, collapsed infrastructure, wounded persons, and the like. Naturally, first responders tend to move towards these hotspots while ordinary persons tend to move away from these hotspots. Also, officials tend to move towards the hotspots but only stay at the hotspot for about a minute or less while first responders stay at the hotspot for a longer time around 15 minutes, which is their average time at the scene during emergencies [2]. Using this special role-based disaster mobility model, any point in A_d can eventually be covered by some users in U, that is, the point will be visited by some user at some time in the future. Each user in U will also have direct or indirect contact with any other user in U eventually, wherein the direct contact represents the situation of two users existing in their common communication range and the indirect contact is defined as the transitive closure of the direct contact.

Each node $u \in U$ is able to *create*, *send*, *receive*, or *aggregate* messages in different locations. Its location at time t is denoted by $u.pos(t)$ and determined either through GPS or estimated based on some other means. The create action of node u includes sensing information and creating an atomic message m containing the information that covers a circular area $m.area$ with radius r_s. When creating an atomic message, its center location $m.center$ is the event place and $m.area$ does not contain any other events. Each node also has a limited storage for the collected information.

Each node is capable of short-range wireless communication, such as Bluetooth or WiFi, and a unit disc model is adopted with every node having the same communication range r_c. Within the range r_c, each node can send or receive data from other nodes with limited *contact duration*, defined as the available time when a contact (communication opportunity) occurs between two nodes. The contact duration between nodes u and u' starting from time t is denoted by $cd(u, u', t)$. In addition, the maximum transmission speed of the available short-range wireless communication is denoted by BW and the transferrable data amount of one contact is defined by $cd(u, u', t) \times BW$.

When a node creates or receives messages, it aggregates its stored messages depending on the aggregation granularity ϕ set by the query node. Let $\phi.a$ and $\phi.r$ denote the maximum area size and radius that a message can cover, respectively. Let m_1 and m_2 denote atomic messages with corresponding coverage areas $m_1.area$ and $m_2.area$. As shown in Fig. 2 (a), m_1 and m_2 will only be aggregated to m^* if the following conditions are met: (a) the coverage areas of the two messages are overlapping,

$$m_1.area \cap m_2.area \neq \emptyset \tag{1}$$

(b) the merged area is smaller than the maximum area size,

$$|m_1.area \cup m_2.area| \leq \phi.a \tag{2}$$

(c) the farthest distance from the center point of the merged area is not greater than $\phi.r$,

$$\textsc{Radius}(m_i.area \cup m_j.area) \leq \phi.r \tag{3}$$

and (d) the center locations of m_1 and m_2 are not within the radius r_s of each other,

$$\textsc{Dist}(m_i.center, m_j.center) > \textsc{Max}(m_i.r_s, m_j.r_s) \tag{4}$$

wherein this is the condition to avoid the merging of atomic messages with the same event.

Let t denote the time elapsed since a query is issued ($t = 0$ when the query is issued). As time t progresses, the node may move from one location to another or perform a certain action type $A_t = \{create, send, receive, aggregate\}$. Time is divided into time periods $T_0, T_1, \ldots$ with length P. Each period is also divided into two parts: *active interval* and *sleep interval*. In each time period, the first pP portion is assigned as the active interval and the remaining $(1 - p)P$ portion as the sleep interval, where p is a system parameter such that $0 < p < 1$. Each node is assumed to have a sufficiently accurate clock to synchronize with other nodes in active and sleep intervals wherein it sends a beacon message for finding other nodes only in the active interval. During the sleep interval, each node turns its wireless communication device to sleep mode if there is no contact with other nodes to save energy consumption.

3.3 Problem Definition

Given a query specifying the AoI, our problem is to derive the set of actions taken by each node in U that collects and delivers the set of messages M_0 covering the AoI to the sink u_0 in minimal time.

Each node $u_i \in U$ has the set of action tuples Act_i, where $a_{i_j} = \langle u_i, a_t, M_i, t, m\rangle \in Act_i$ refers to the jth action performed by u_i. The tuple $\langle u_i, a_t, M_i, t, m\rangle$ represents that node u_i performs action a_t on message set M_i at time t and the resulting (created or aggregated) message is m.

For every send action of node u_i, there is a corresponding receive action of node u_j. The send and receive actions must be performed during the contact duration, and the entire message must also be transferred within the duration. Thus, the following equation must hold.

$$\begin{aligned}&\forall u_i \in U, \quad \forall\langle u_i, send(u_j), M_i, t, m\rangle \in Act_i, \quad \exists u_j \in U,\\ &\exists\langle u_j, receive(u_i), M_i, t', m\rangle \in Act_j \wedge \exists t'' cd(u_i, u_j, t'') > 0\\ &\qquad \text{such that} \quad t'' \le t \le t' \le t'' + cd(u_i, u_j, t'')\\ &\qquad\qquad \wedge Size(M_i) \le cd(u_i, u_j, t'') \times BW \end{aligned} \tag{5}$$

where

$Size(M_i)$ is the total byte size of the messages in message set M_i.

The set of messages M_0 delivered to sink u_0 is defined as follows,

$$M_0 = \bigcup_{\langle u_0, receive, M, t, null\rangle \in Act_0} M \tag{6}$$

and must achieve full coverage of the AoI.

$$\bigcup_{m \in M_0} m.area \supseteq area(AoI), \tag{7}$$

Each message in M_0 must have been created or aggregated by some nodes according to the following condition:

$$\forall m \in M_0, \quad IsCreated(m) \vee IsAggregated(m) \tag{8}$$

where

$IsCreated(m) \stackrel{def}{=} \exists u_i \in U, \quad \exists\langle u_i, create, \emptyset, t, m\rangle \in Act_i$

$IsAggregated(m) \stackrel{def}{=} \exists u_i \in U,$
$\exists\langle u_i, aggregate, M_a, t, m\rangle \in Act_i \quad \wedge \quad \forall m' \in M_a,$
$IsCreated(m') \vee IsAggregated(m')$.

Data collection delay D at u_0 is defined as follows.

$$D = max\{t | \langle u_0, receive, M, t, m\rangle \in Act_0\} \tag{9}$$

Thus, given A_d, U, and a query q with u_0, t_0, AoI, and ϕ, the problem is defined as the minimum time data aggregation (MTDA) problem to decide the set of actions Act_i for each node u_i with the objective function defined as:

minimize D, **subject to** Eqs. (5), (7), **and** (8)

The MTDA problem is an NP-hard problem since it implies the minimum geometric disk cover problem as a special case, even when we do not apply aggregation to messages. Thus, we devise a heuristic solution to solve the MTDA problem in the next section.

4. Data Aggregation Algorithm

In this section, a greedy algorithm is presented to solve the MTDA problem described in Section 3.3. The NodeAction algorithm shown in Algorithm 1 is our main algorithm. This algorithm is executed at each node $u_i \in U$ independently of the other nodes and determines action $a_t \in \{create, send, receive, aggregate\}$ of u_i on its message set M_i over time. Each node runs the algorithm when it receives the query containing u_0, t_0, AoI, and ϕ. The query is initiated by a stationary node at a special spot (e.g., command post) known as the sink wherein the Time to Live (TTL) of the query is set and can be varied depending on the situation. These neighbor nodes, which may be mobile, forward the query to its subsequent neighbor nodes and so on. Query distribution is sent through flooding wherein most nodes in A_d receive the query at some point. Moreover, each node has an electric map of the target disaster area A_d and its time period P (e.g., 10 s) for the duty cycle is set, with an active interval (e.g., 1 s) and a sleep interval (e.g., 9 s). When a query is received, a node sends a beacon message to find its neighbor nodes during its active interval. However, during its sleep interval, it creates a message, exchanges messages with its neighbor nodes, aggregates messages, or just sleeps if there are no neighbor nodes or there are no messages that can be aggregated.

The variables used in the algorithms with their corresponding meanings are shown in **Table 1** for readability purposes. As shown in Algorithm 1, it is assumed that the location of node u_i is known and node u_i has received the query consisting of the identity of sink u_0 and its position $u_0.pos(0)$, time t_0 when the query is issued, area of interest AoI, and aggregation granularity ϕ. During this instance, time t is set to the time elapsed from the query sending time t_0 (Algorithm 1 line 1). The variable M_i represents the message set retained by u_i and is initialized to be empty (Algorithm 1 line 2). At this point, node u_i is not in contact with any node as represented by a null N_i (Algorithm 1 line 3). Node u_i then enters into a loop performing lines 4–18 (Algorithm 1) until reaching the predetermined deadline T, which may be equivalent to the time that the query is not needed anymore preventing node u_i from going into an infinite loop, or node u_i receives an acknowledgement, piggybacked in the messages, that node u_0 has

Table 1 Algorithm variables.

Variable	Meaning
$ert(u_i)$	Expected reaching time of u_i to u_0
m_i	Atomic or aggregated message
$m_i.area$	Coverage area of message m_i
$m_i.replica$	Replica number of message m_i
M_i	Current message set of mobile node u_i
M'	Aggregated message set
N_i	Neighbor node set of mobile node u_i
ϕ	Aggregation granularity
q	Query message
$replica$	Replica number of message
t	Elapsed time
t_0	Time when query was issued
T	Predetermined deadline of the query q
u_i, u_j	Mobile nodes
$u_i.pos(t)$	Location of mobile node u_i at time t
u_m	Mobile node with the lowest ert
u_0	Query sender or sink

Algorithm 1 NodeAction($u_i.pos(t), q$)

Input: $u_i.pos(t)$, $replica$, q=$\langle u_0, t_0, AoI, \phi \rangle$
Output: Node action schedule of u_i

```
1:  t ← current time - t0
2:  Mi ← ∅
3:  Ni ← ∅
4:  while t ≤ T or Mi covering AoI is received by u0 do
5:      Ni ← NeighborDiscovery(ui, Ni, t, t0)
6:      while t is in sleep interval do
7:          if ui.pos(t) is within AoI, within mi.area, and outside the covered
            area of Mi then
8:              ui creates mi
9:              mi.replica ← replica
10:             Mi ← Aggregate(Mi, {mi}, φ)
11:         end if
12:         if ui has a new message then
13:             Mi ← MessageExchange(ui, u0, Ni, Mi)
14:             Mi ← Aggregate(Mi, Mi, φ)
15:         end if
16:         t ← current time - t0
17:     end while
18: end while
```

received M_i.

The following subsections explain in detail the different parts of our main algorithm: neighbor discovery, message creation, message exchange, and message aggregation.

4.1 Neighbor Discovery

During the active interval in each time period, node u_i sends a beacon message for neighbor node discovery (Algorithm 2 line 5). The active interval period is divided into slots (e.g., 10 ms). Each node sends a beacon message during its active interval after the randomly decided backoff time (up to the active interval length). When node u_i receives the beacon message from another node u_j (Algorithm 2 line 8), nodes u_i and u_j are assumed to be in contact and node u_j is added to the neighbor node set N_i (Algorithm 2 line 9). All of the nodes within r_c of node u_i are added to its neighbor node set N_i. In addition, the clocks of all nodes are synchronized with the global clock using GPS or by some other means such that the active and sleep intervals of all nodes are synchronized with sufficient accuracy.

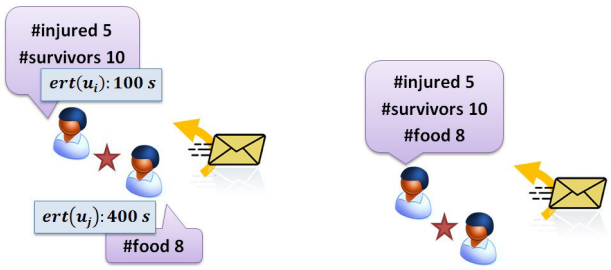
Algorithm 2 NeighborDiscovery(u_i, N_i, t, t_0)

Input: Mobile node u_i, current neighbor node set N_i
Output: Updated neighbor node set N_i

```
1:  t ← current time - t0
2:  backoff ← random number
3:  while t is in active interval do
4:      if backoff = 0 then
5:          ui sends a beacon message
6:          backoff ← random number
7:      end if
8:      if ui receives a beacon message from another node uj then
9:          Ni ← Ni ∪ {uj}
10:     end if
11:     backoff ← backoff − (current time−t)
12:     t ← current time - t0
13: end while
14: return Ni
```

4.2 Message Creation

When t is within the sleep interval, node u_i checks whether its current location is within the AoI, within the effective area $m_i.area$ of an event, and outside the covered area of its message set M_i (Algorithm 1 line 7). If it is true, u_i creates a message m_i with a center location and an effective area the same as the said event (Algorithm 1 line 8). A message replica count is also initialized to $replica$ (a constant number given in advance) to increase the probability of message delivery (Algorithm 1 line 9). The newly created message m_i is then aggregated with the other local messages M_i of node u_i, which is explained in detail in Section 4.4 (Algorithm 1 line 4).



(a) Before (b) After

Fig. 3 Message exchange.

4.3 Message Exchange

When node u_i has determined its neighbor set N_i, it decides whether to retain or to forward its stored messages (Algorithm 1 line 13). The expected reaching time of each contact node to sink u_0 denoted by $ert(u_i)$ is used to determine the node action. The expected reaching time $ert(u_i)$ can be determined from the node's speed $v(u_i)$, moving direction, and the distance of the node from u_0, $d(u_i, u_0)$, that can be computed from the positions of nodes u_i and u_0 as well as the links of disaster area A_d (Algorithm 3 line 4). Assume that node u_i is travelling at a speed $v(u_i) = 1$ m/s to a spot (intersection) 30 m far away from the current location and the spot is 70 m from node u_0, then $ert(u_i) = \frac{30+70}{1} = 100$ s. This means that it will take 100 s for u_i to come into the possible earliest contact with node u_0. Node u_i then determines the node with the lowest ert from its N_i (Algorithm 3 lines 5–13), say u_m. If node u_i is not the node with the lowest ert (Algorithm 3 line 14), node u_i sends its stored messages to node u_m (Algorithm 3 line 15) and node u_m receives the stored messages of node u_i (Algorithm 3 line 16). The replica number of each stored message is also decremented every time it is sent to another node (Algorithm 3 line 18). When node u_i sends its stored messages, it retains its messages until the replica number of a message reaches 0 wherein the message is deleted from the node's buffer (Algorithm 3 lines 19–22). Let us take the example shown in **Fig. 3** and assume that the replica number of both messages is 1. In Fig. 3 (a), $ert(u_i) < ert(u_j)$ thus, node u_j sends its stored messages to node u_i and removes the messages from its buffer as shown in Fig. 3 (b) since the replica number of the message after the exchange becomes 0. Moreover, if node u_i is the node with the lowest ert and it receives a message set M_j from a neighbor node (Algorithm 3 line 24), node u_i retains its stored messages and adds the received messages to its local messages (Algorithm 3 line 25).

4.4 Message Aggregation

4.4.1 Message Structure

Each message contains disaster-related information with a corresponding area coverage, a replica number, and a Bloom filter. The disaster-related information contained in the message follows a Tweet-based syntax with a certain limit (e.g., 140 char-

Algorithm 3 MessageExchange(u_i, u_0, N_i, M_i)

Input: Mobile node u_i, sink u_0, neighbor node set N_i, current message set M_i
Output: Updated message set M_i

```
1: M_j ← null
2: u_j ← null
3: u_m ← u_i
4: minert ← ert(u_i) calculated from the locations of u_i and u_0
5: while N_i ≠ ∅ do
6:     u_j ← select one node from N_i at random
7:     Determine ert(u_j) based on the locations of u_j and u_0
8:     if ert(u_j) < minert then
9:         minert ← ert(u_j)
10:        u_m ← u_j
11:    end if
12:    N_i ← N_i − {u_j}
13: end while
14: if u_m ≠ u_i then
15:    u_i sends M_i to u_m
16:    u_m receives M_i from u_i
17:    for all m_i ∈ M_i do
18:        m_i.replica ← m_i.replica − 1
19:        if m_i.replica = 0 then
20:            delete m_i
21:            M_i ← M_i − {m_i}
22:        end if
23:    end for
24: else if u_m = u_i and u_i receives a message set M_j then
25:    M_i ← M_i ∪ M_j
26: end if
27: return M_i
```

acters) and containing statistical information on survivors, shelter capacity, and available resources. The information consists of a keyword marked by a hashtag followed by a number that describes the keyword. Examples of Tweet-like messages are shown in Fig. 3. Let us take a message containing the following: `#survivors 10`. In this message, the keyword is survivors and the number is 10. This means that there are 10 survivors around the message's location. An atomic (raw) message will contain only one keyword with a corresponding number. However, aggregated messages will contain a number of keywords marked by hashtags with their corresponding numbers. To aggregate messages, the sums of the numbers corresponding to the same keywords are determined. Let us take the atomic messages containing the following: `#survivors 10` and `#survivors 5`. The resulting aggregate message of the two atomic messages is `#survivors 15`. However, messages are only aggregated if Eqs. (1) to (3) in Section 3 are satisfied. The equivalent coverages of the atomic messages are also merged by approximating a circle that will contain the coverages of the messages.

4.4.2 Aggregation Granularity

When a new message set M_j is received by u_i, it aggregates or concatenates message set M_j with the local messages M_i depending on ϕ. Let M' contain the aggregated messages, which is set to be empty initially (Algorithm 4 line 1). In the aggregation process, the entire message pairs of M_i and M_j are tried to be merged (Algorithm 4 lines 2–8). However, only the message pairs that satisfy Eqs. (1) to (3) in Section 3 are aggregated (Algorithm 4 line 3). If these conditions are met, messages m_i and m_j are aggregated depending on the aggregation function f_a resulting in the aggregated message m_a (Algorithm 4 line 4), which is then added to the set of aggregated messages M' (Algorithm 4 line 6). However, if m_i and m_j are atomic messages, the following condition must also hold before aggregation: the center locations of m_i and m_j are not within the radius r_s of each other.

Algorithm 4 Aggregate(M_i, M_j, ϕ)

Input: Message sets M_i and M_j, aggregation granularity ϕ
Output: Aggregated message set M'

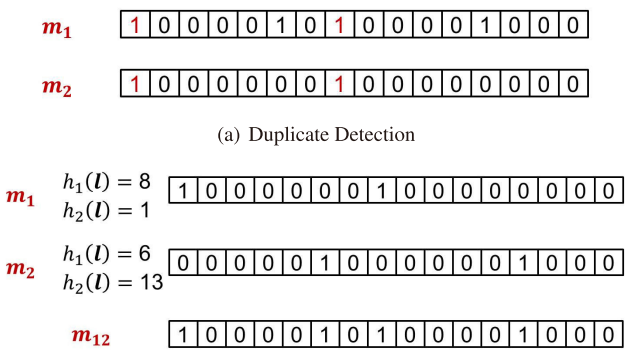
```
1: M' ← ∅
2: for each pair (m_i, m_j) ∈ M_i × M_j do
3:     if m_i.area ∩ m_j.area ≠ ∅ ∧ |m_i.area ∪ m_j.area| ≤ φ.a ∧
       Radius(m_i.area ∪ m_j.area) ≤ φ.r ∧
       AND(BF(m_i), BF(m_j)) ≠ BF(m_i) ∧
       AND(BF(m_i), BF(m_j)) ≠ BF(m_j) then
4:         m_a ← aggregate messages m_i and m_j
5:         BF(m_a) ← OR(BF(m_i), BF(m_j))
6:         M' ← M' ∪ {m_a}
7:     end if
8: end for
9: return M'
```

Consider node u_1 in the *AoI* as shown in Fig. 2 (a), in which node u_1 receives a query. During the sleep interval of u_1, it creates an atomic message m_1 with $r_s = 5$ m. Then, as node u_1 becomes active, it sends beacon messages to discover its neighbor nodes. Suppose that nodes u_1 and u_2 come into contact, node u_1 compares its expected reaching time $ert(u_1)$ with the expected reaching time of u_2, $ert(u_2)$. Let us suppose that the expected reaching time of u_1 $ert(u_1)$ is less than the expected reaching time of u_2 $ert(u_2)$, message m_2 of node u_2 is sent to node u_1. Aggregation of the messages occurs if Eqs. (1) to (3) of Section 3 hold. The aggregated message m^* will then contain the information on the total number of survivors from the two messages m_1 and m_2.

4.4.3 Bloom Filter

In data aggregation, there are instances that duplication occurs. To prevent this, a Bloom filter is constructed for each created message (atomic or raw message). It determines if a particular message is already part of the aggregated message. The Bloom filter is an array of n bits representing a set of messages. Initially, all the bits in the filter are set to zero and when an atomic message is added, k bits in the filter are set to 1 depending on the chosen number of hash codes. The Bloom filter uses a hash function to map the messages to random numbers within the index range of the filter. For the hash function, the MD5 hash algorithm is used since it is a popular choice for the hash function of Bloom filters [18]. Each atomic message is directly mapped to a bit sequence using its creation location. Let us take the example shown in **Fig. 4** (b). The creation location l of atomic message m_1 is hashed and the resulting index h_i is equivalent to its array position b_j in the Bloom filter. In the example, for atomic message m, there are two resulting hash indices $h_1(l)$ and $h_2(l)$. The resulting indices 8 and 1 correspond to the array positions of its Bloom filter thus, the positions b_8 and b_1 of its Bloom filter are set.

Before aggregating atomic messages, their corresponding Bloom filters are checked (Algorithm 4 line 3). If they contain the same set of array positions equal to 1, the atomic messages are not aggregated. If otherwise, the messages are aggregated and the Bloom filter of the aggregated message is the result of the OR operation between the corresponding Bloom filters of the atomic messages (Algorithm 4 line 5). Let us take for example Fig. 4 (b), wherein m_1 has the resulting hash indices of 8 and 1 while m_2 has 6 and 13. This means that the two messages do not contain any identical atomic message and can be aggregated. To determine the bit sequence of the aggregated message m_{12}, an OR



(a) Duplicate Detection

(b) Aggregation

Fig. 4 Bloom filter.

operation of the Bloom filters of the atomic messages m_1 and m_2 is executed that is, BloomFilter(m_{12}) = OR(BloomFilter(m_1), BloomFilter(m_2)). The resulting Bloom filter of the aggregated message m_{12} has positions b_1, b_6, b_8, and b_{13} set to 1. Moreover, in our proposed data aggregation, there are situations when atomic messages and/or aggregated messages are to be merged. To determine if the messages can be aggregated, their corresponding Bloom filters are checked. If their Bloom filters have the same set of array positions equal to 1 or partly overlapping array positions equal to 1, the messages are not aggregated just like in Fig. 4 (a), wherein the Bloom filters of m_1 and m_2 have an overlapping bit sequence of k bits ($k = 2$) that is, both positions b_8 and b_1 are set for the two Bloom filters. Therefore, there is an atomic message contained in m_1 and m_2 that are identical and the messages are not merged. If otherwise, the atomic message becomes part of the aggregated message and its corresponding Bloom filter is merged with the aggregated message Bloom filter using the OR operation. Thus, the resulting message set after aggregation consists of local atomic/aggregated messages and received atomic/aggregated messages.

In relation to the correctness or the accuracy of the message aggregation, let us take for example aggregated messages m_{12} and m_{13}, which contain atomic messages m_1 and m_2 and atomic messages m_1 and m_3, respectively. The Bloom filters of the aggregated messages will have an overlap because they contain a common atomic message m_1 thus, these messages would not be merged because it will result in the double counting of the information contained in atomic message m_1. These messages are then delivered to the sink as separate messages instead of one aggregate and when the sink receives these messages, it knows that they have an overlapping area but it cannot exactly extract the information in the overlapping area. Resolving this problem is part of the future work.

5. Evaluation

The performance of the proposed algorithm is evaluated using a custom simulator. In order to evaluate the proposed algorithm, the time for the aggregated messages covering 80% of the *AoI* to arrive at its destination is determined.

5.1 Simulation Configuration

In this study, a custom simulator is used since the contact times

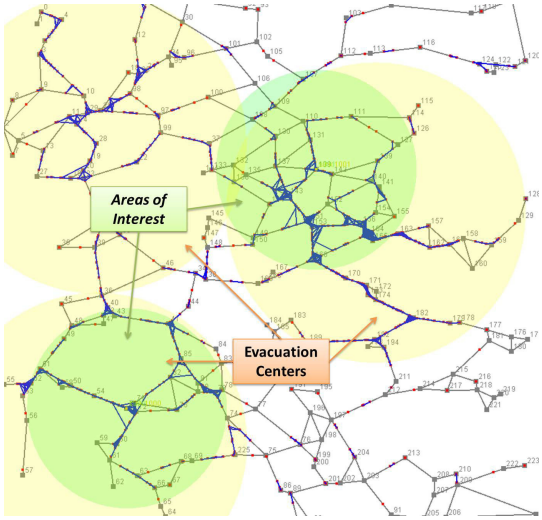


Fig. 5 Simulation environment.

of nodes are considered to be more important than its physical and link layer details. Therefore, we implemented the simulator so that a node can discover and connect with another node at transmission range instantly. Simulation consists of mobile nodes placed uniformly at random over a two-dimensional plane with a 3 km × 3 km area. The area is based on an actual map within the vicinity of Takayama Science Town as shown in **Fig. 5**. The skeleton map represents the road network, which is composed of streets between special spots (e.g., hospitals, universities, and the like). In addition, the locations of the evacuation centers are based on actual evacuation centers published by the Ikoma City government. A special role-based disaster mobility model (Section 3.2) is adopted wherein the nodes only travel along the road network. Each node moves at a random speed with destinations conforming to its role and with routes based on Djikstra's shortest path algorithm. Disaster-related events with varying radii were generated with locations selected randomly inside the *AoI* depending on the type of information. For example, information about an injured person has an effective radius of 1 m while information about a shelter has an effective radius of 4 m. Two queries were also issued at the start of the simulation with TTL = 1 hr and were distributed by flooding. The locations of the query senders (or sinks) were randomly selected among the locations of the special spots within the disaster area A_d. The flooding overhead was considered by specifying corresponding data sizes for query and beacon messages. A simulation warming time was set wherein during this time, the nodes move according to its mobility model without performing any action. All nodes also have the same buffer size and transmission range. A total of 5 runs were taken with two queries for each simulation setting. **Table 2** shows a summary of the default values used in the simulation. In order to facilitate a congested scenario in the network, the network bandwidth was set to 1 Mbps, the nonaggregated or raw message size to 5 KB, and the number of events within the *AoI* to 1,000 events. This is because there is usually a surge in network traffic within the disaster area as a result of the large amount of messages generated by the people.

Table 2 Simulation parameters.

Parameter	Default Value
Network	
Bandwidth	1 Mbps
Buffer size	100–500 KB
Transmission range	50 m
Map	
Disaster area size	3 km × 3 km
No. of map points	226
No. of hotspots	120
No. of evacuation centers	3
Evacuation center range	750 m
AoI radius	500 m
No. of events within AoI	1,000
No. of queries	2
No. of mobile nodes	200–1,000
Node ratio (citizens:officials:responders)	3:1:1
Node	
Speed	1.0–1.4 m/s
Duty cycle period	10 s
Active interval	1 s
Sleep interval	9 s
Warming time	1,000 s
Message	
Size	5 KB
Beacon message size	10 bytes
Query message size	300 bytes
Replica number	1–7
Aggregation granularity ϕ	0–100 m
Bloom filter size	256 bits

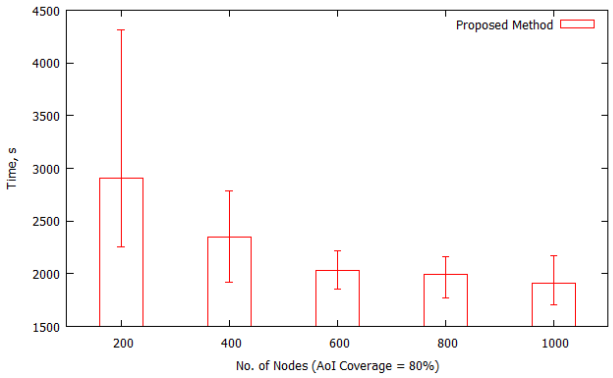


Fig. 6 Effect of node density on message delivery latency (buffer size = 500 KB, replica = 5).

5.2 Message Delivery Latency

We evaluated the time for the aggregated messages to arrive at the sink in response to a query. Different parameters were varied such as the number of mobile nodes present in the disaster area A_d wherein the average message delivery latencies and its corresponding variances were determined. **Figure 6** shows the effect on message delivery latency with the increase in the number of mobile nodes present in A_d. Based on the figure, there is a 34.1% decrease in message delivery latency when the number of mobile nodes present in A_d is increased from 200 nodes to 1,000 nodes. The significant decrease is due to the increase in contact opportunities between nodes. During these contact opportunities, messages are exchanged and aggregated to achieve the required coverage of the *AoI* and the increase in contact opportunities leads to a higher probability that more of the exchanged and aggregated messages reach the sink faster.

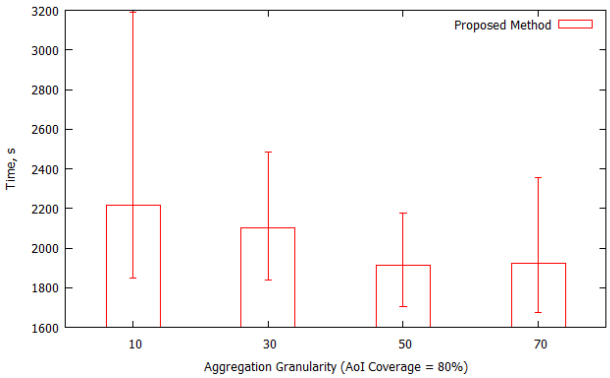


Fig. 7 Effect of aggregation granularity on message delivery latency (buffer size = 500 KB, No. of nodes = 1,000, replica = 5).

5.3 Aggregation Granularity

Another parameter that needs to be taken into consideration is the aggregation granularity ϕ. This sets the level of detail in the aggregated messages. The simulation result for the variation of the aggregation granularity metric is shown in **Fig. 7** with 1,000 mobile nodes present in A_d. Based on the results, aggregation granularity has an effect on message delivery latency. As shown in the figure, the average message delivery latency when ϕ = 10 m is 2216.6 s while the message delivery latency when ϕ = 50 m is 1913.2 s, which is a 13.7% decrease in latency. This decrease is due to the increase in aggregation granularity wherein more messages are merged into a single message thereby reducing message size and the aggregated message is transmitted over the network faster than unaggregated messages. Moreover, as shown also in the figure, there is a boundary on the decreasing effect of the aggregation granularity on message delivery latency. When ϕ is greater than 50 m, there seems to be no significant effect on the message delivery latency because of the absence of nonaggregated messages that satisfies the set conditions or other uncontrolled factors like the mobility of the nodes.

5.4 Comparison with a Conventional Method

The proposed algorithm was also compared to epidemic routing with limited flooding (called *epidemic routing limited*, hereafter). In epidemic routing limited, the node's stored messages are copied to its neighbor nodes based on the set replica number that is, the number of copied messages are limited to the replica number just like in the proposed method. We used epidemic routing limited instead of the general epidemic routing to prove the effectiveness of the proposed aggregation method in decreasing message delivery latency.

5.4.1 Node Density

The average message delivery latencies of varying node density were plotted including their corresponding variances. **Figure 8** compares the performance of the proposed method and epidemic routing limited in terms of message delivery latency by varying the number of mobile nodes in A_d. The figure shows that with varying mobile nodes present in A_d, the proposed method achieves a better performance in message delivery latency compared to epidemic routing limited. Using the proposed method, the information collection delay decreased by 9.8%, 14.9%, 9.2%, 5.9%, and 9.7% compared to epidemic rou-

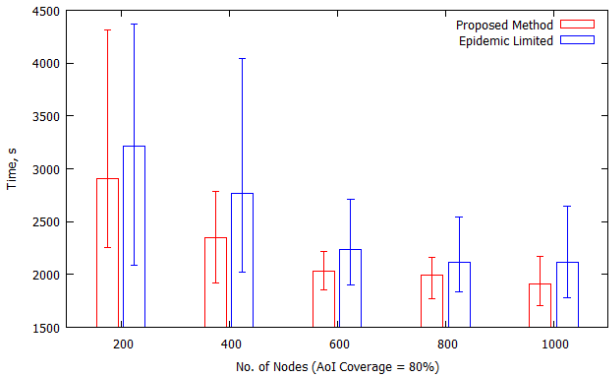


Fig. 8 Comparison of message delivery latency with epidemic routing vs. node density (buffer size = 500 KB, replica = 5).

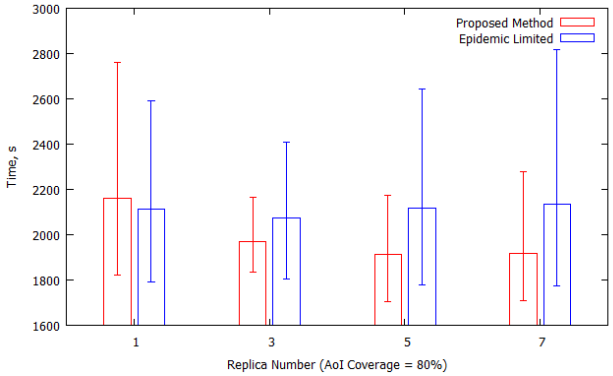


Fig. 9 Comparison of message delivery latency with epidemic routing vs. replica number (buffer size = 500 KB, No. of nodes = 1,000).

ting limited with 200, 400, 600, 800, and 1,000 mobile nodes, respectively.

5.4.2 Replica Number and Buffer Size

We also evaluated our proposed algorithm with varying replica numbers and buffer sizes to further show the efficiency of the proposed method over epidemic routing limited. **Figure 9** compares the performance of the proposed method and epidemic routing limited in terms of message delivery latency by varying the replica numbers of the message. The figure shows that with replica number = 1, the average message delivery latencies of the proposed method and epidemic routing limited were almost equal. However, as the replica number is increased, the proposed method outpasses epidemic routing limited in terms of achieving a lower latency. There was a 5.0%, 9.7%, and 10.1% decrease in message delivery latency using the proposed method as compared to epidemic routing limited with replica numbers 3, 5, and 7, respectively. **Figure 10** also compares the performance of the proposed method and epidemic routing limited in terms of message delivery latency by varying the buffer size allocated to each node. The figure clearly shows that even with varying buffer sizes, the proposed method achieves a better performance in message delivery latency compared to epidemic routing limited. There was a 6.0%, 8.5%, 9.7%, and 4.6% decrease in message delivery latency using the proposed method as compared to epidemic routing limited with buffer sizes 100 KB, 300 KB, 500 KB, and 700 KB, respectively. The decrease in message delivery latencies is due to the message buffer overflow that occurs when epidemic routing limited is used.

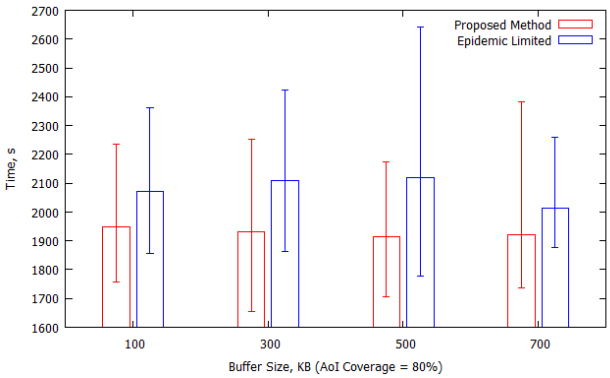
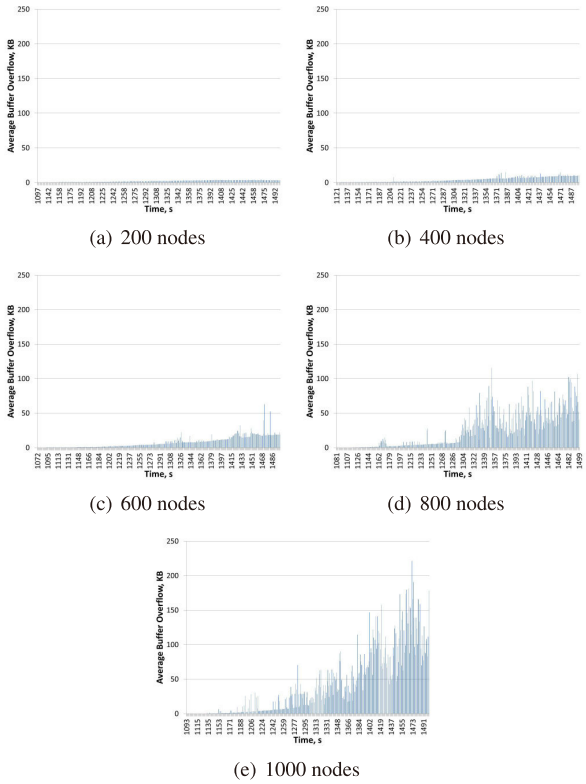


Fig. 10 Comparison of message delivery latency with epidemic routing vs. buffer size (No. of nodes = 1,000, replica = 5).



(a) 200 nodes (b) 400 nodes

(c) 600 nodes (d) 800 nodes

(e) 1000 nodes

Fig. 11 Average buffer flow using epidemic routing (buffer size = 500 KB, No. of nodes = 1,000, replica = 3).

5.4.3 Buffer Overflow

In our simulation, a 500 KB buffer is allocated to each node and when the buffer space is consumed, old messages are removed to make buffer space for new messages. As shown in **Fig. 11**, buffer overflow happens immediately during the first 500 s of simulation time after the warming period. However, this does not occur using the proposed method. The figure also shows that as the number of mobile nodes present in A_d increases, the amount of buffer overflow increases at a rapid pace. Even if this did not cause a big impact on the result in Fig. 8, it is expected that when the number of nodes or as the network gets more congested due to the increase in message size, there will be a rapid growth in message delivery latency using epidemic routing.

5.4.4 Cumulated Distribution Function

Moreover, cumulated distribution function (CDF) graphs of the time that it takes for the messages to reach its destination using

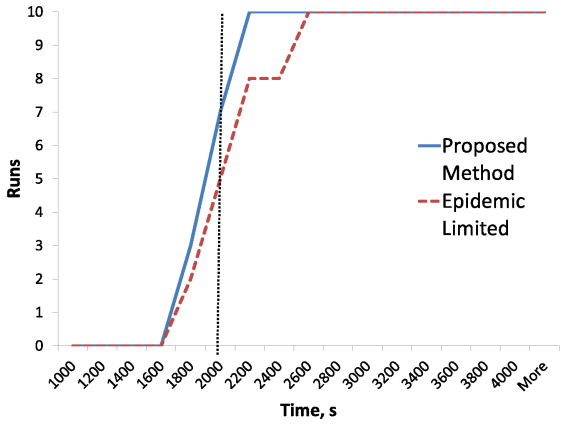


Fig. 12 Comparison of message delivery latency CDF with epidemic routing (buffer size = 500 KB, No. of nodes = 1,000, replica = 5).

the proposed method and epidemic routing are plotted. The CDF graph of message delivery latency with 1,000 mobile nodes is shown in **Fig. 12**. A CDF graph shows the percentage of data falling between two points. In the said figure, the CDF graph shows the number of runs that attain 80% coverage of the *AoI* within a specified period of time. Based on Fig. 12, from $t = 0$ to 2,000, 70.0% of all runs achieved 80% coverage of the *AoI* using the proposed method while 50.0% of all runs achieved 80% coverage of the *AoI* using epidemic routing from $t = 0$ to 2,000. This is a 28.6% decrease in the number of runs using the proposed method between $t = 0$ to 2,000 showing the effectivity of using the proposed method to decrease message delivery latency in collecting disaster information.

6. Conclusion

The absence of communication infrastructure is common in disaster areas. To provide sufficient information regarding the affected area, a DTN is implemented using the participatory sensing framework. Even though the network is disruption tolerant, delay must still be minimized. Thus, a DTN-based data aggregation algorithm is proposed to minimize this delay. Information is collected, aggregated, and sent through DTN by users present in the affected area. A query sender in a known location issues a query on an *AoI* and the time when the aggregated messages covering 80% of the *AoI* reaches the query sender is evaluated. Results show that the proposed method achieved a lower message delivery latency and a higher percentage of runs that attained 80% *AoI* coverage than epidemic routing limited proving that the proposed method is indeed effective in decreasing delay for message delivery in disaster areas. Moreover, using the proposed method, the maximum delay with the lowest number of mobile nodes was 4,316 s (approximately 1 hr and 12 mins), which is within the critical limit in the aftermath of the disaster (first 36 – 48 hours and the first 2 hours if people are wounded or trapped in buildings).

Future work includes an improvisation of the proposed algorithm by extracting information in overlapping areas of the message coverage areas as well as the implementation of the proposed algorithm on a real handheld device, in which people actually collect information from an area. This will further prove that the proposed algorithm can be implemented in a real environment.

Acknowledgments This work was supported in part by the JSPS KAKENHI Grant Numbers 25280031 and 25330104.

References

[1] Ahmed, A., Yasumoto, K., Yamauchi, Y. and Ito, M.: Probabilistic Coverage Methods in People-Centric Sensing, *Journal of Information Processing*, Vol.19, pp.473–490 (2011).
[2] Al-Ghamdi, A.S.: Emergency medical service rescue times in Riyadh, *Accident Analysis & Prevention*, Vol.34, No.4, pp.499–505 (online), DOI: 10.1016/S0001-4575(01)00047-1 (2002).
[3] Bruns, A., Burgess, J.E., Crawford, K. and Shaw, F.: # qldfloods and@ QPSMedia: Crisis communication on Twitter in the 2011 south east Queensland floods (2012).
[4] Cacciapuoti, A.S., Calabrese, F., Caleffi, M., Lorenzo, G.D. and Paura, L.: Human-mobility enabled wireless networks for emergency communications during special events, *Pervasive and Mobile Computing* (2012).
[5] Fajardo, J.T.B. and Oppus, C.M.: A mobile disaster management system using the android technology, *WTOC*, Vol.9, No.6, pp.343–353 (2010).
[6] Fajardo, J., Yasumoto, K., Shibata, N., Sun, W. and Ito, M.: DTN-based data aggregation for timely information collection in disaster areas, *2012 IEEE 8th International Conference on Wireless and Mobile Computing, Networking and Communications* (*WiMob*), pp.333–340 (online), DOI: 10.1109/WiMOB.2012.6379095 (2012).
[7] Fall, K., Iannaccone, G., Kannan, J., Silveira, F. and Taft, N.: *Information Systems for Crisis Response and Management* (*ISCRAM*) (2010).
[8] Fujihara, A. and Miwa, H.: Real-Time Disaster Evacuation Guidance Using Opportunistic Communications, *2012 IEEE/IPSJ 12th International Symposium on Applications and the Internet* (*SAINT*), pp.326–331 (online), DOI: 10.1109/SAINT.2012.59 (2012).
[9] Huda, M.N., Yasmeen, F., Yamada, S. and Sonehara, N.: An approach for short message resilience in disaster-stricken areas, *2012 International Conference on Information Networking* (*ICOIN*), pp.120–125 (2012).
[10] Ishimaru, Y., Sun, W., Yasumoto, K. and Ito, M.: DTN-based Delivery of Word-of-Mouth Information with Priority and Deadline, *5th International Conference on Mobile Computing and Ubiquitous Networking* (2010).
[11] Joe, I. and Kim, S.-B.: A Message Priority Routing Protocol for Delay Tolerant Networks (DTN) in Disaster Areas, *Future Generation Information Technology*, Kim, T.-h., Lee, Y.-h., Kang, B.-H. and Slezak, D. (Eds.), Lecture Notes in Computer Science, Vol.6485, pp.727–737 Springer Berlin/Heidelberg (2010).
[12] Joo, C., Choi, J.-G. and Shroff, N.B.: Delay Performance of Scheduling with Data Aggregation in Wireless Sensor Networks, *Proc. IEEE INFOCOM 2010*, pp.1–9 (2010).
[13] Lane, N., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T. and Campbell, A.: A survey of mobile phone sensing, *IEEE Communications Magazine*, Vol.48, No.9, pp.140–150 (2010).
[14] Liu, M., Yang, Y. and Qin, Z.: A Survey of Routing Protocols and Simulations in Delay-Tolerant Networks, *Wireless Algorithms, Systems, and Applications*, Cheng, Y., Eun, D., Qin, Z., Song, M. and Xing, K. (Eds.), Lecture Notes in Computer Science, Vol.6843, pp.243–253, Springer Berlin/Heidelberg (2011).
[15] Peng, A., Moen, D., Spinks, J., Meredith, L., He, T. and Lilja, D.: Reliable data aggregation and dissemination framework in tactical network architecture, *Military Communications Conference, 2010 - MILCOM 2010*, pp.569–574 (online), DOI: 10.1109/MILCOM.2010.5680436 (2010).
[16] Rana, R.K., Chou, C.T., Kanhere, S.S., Bulusu, N. and Hu, W.: Ear-phone: An end-to-end participatory urban noise mapping system, *Proc. 9th ACM/IEEE International Conference on Information Processing in Sensor Networks, IPSN '10*, pp.105–116, ACM (online), DOI: 10.1145/1791212.1791226 (2010).
[17] Saha, S., Sushovan, Sheldekar, A., Joseph C.R., Mukherjee, A. and Nandi, S.: Post Disaster Management Using Delay Tolerant Network, *Recent Trends in Wireless and Mobile Networks*, Özcan, A., Zizka, J. and Nagamalai, D. (Eds.), Communications in Computer and Information Science, Vol.162, pp.170–184, Springer Berlin Heidelberg (2011).
[18] Tarkoma, S., Rothenberg, C.E. and Lagerspetz, E.: Theory and Practice of Bloom Filters for Distributed Systems, *Communications Surveys Tutorials*, Vol.14, No.1, IEEE (2012).
[19] Weinsberg, U., Li, Q., Taft, N., Balachandran, A., Sekar, V., Iannaccone, G. and Seshan, S.: CARE: Content aware redundancy elimination for challenged networks, *Proc. 11th ACM Workshop on Hot Topics in Networks*, HotNets-XI, pp.127–132, ACM (online), DOI: 10.1145/2390231.2390253 (2012).

[20] Zhou, J., Li, J. and Burge, L.: Efficient scheduling of pigeons for a constrained delay tolerant application, *EURASIP Journal on Wireless Communications and Networking*, Vol.2010, pp.1:1–1:7 (2010).

Jovilyn Therese B. Fajardo received her B.S. degree in Chemistry and Computer Engineering from Ateneo de Manila University, Philippines in 2006 and 2007, respectively. She received her M.S. in Electronics Engineering from Ateneo de Manila University, Philippines in 2010. Currently, she is a Ph.D. student at the Graduate School of Information Science, Nara Institute of Science and Technology, Japan. Her research interests include mobile ad-hoc networks and participatory sensing.

Keiichi Yasumoto received his B.E., M.E., and Ph.D. degrees from Osaka University, Japan, in 1991, 1993, and 1996, respectively. He joined the faculty of Shiga University in 1995. Since 2011, he has been a professor of the Graduate School of Information Science at Nara Institute of Science and Technology. His current research interests include mobile computing, ubiquitous computing, and multimedia communication. He is a member of IEICE, ACM, and IEEE.

Naoki Shibata received his Ph.D. degree in Computer Science from Osaka University, Japan, in 2001. He was an assistant professor at Nara Institute of Science and Technology from 2001 to 2003 and an associate professor at Shiga University from 2004 to 2012. He is currently an associate professor at Nara Institute of Science and Technology. His research areas include distributed systems, inter-vehicle communication, mobile computing, multimedia communication, and parallel algorithms. He is a member of IPSJ, ACM and IEEE/CS.

Weihua Sun received his B.E., M.E., and Ph.D. degrees in Information and Computer Sciences from Osaka University in 2003, 2005, and 2008, respectively. He is an assistant professor at Nara Institute of Science and Technology. His research areas include vehicular communication and mobile computing. He is a member of IPSJ and IEEE/CS.

Minoru Ito received his B.E., M.E., and Ph.D. degrees in Information and Computer Sciences from Osaka University in 1977, 1979, and 1983, respectively. He joined the faculty of Osaka University in 1979. Since 1993, he has been a professor of the Graduate School of Information Science at Nara Institute of Science and Technology. He is a member of ACM and IEEE.