Mobile Message Board: Location-based Message Dissemination in Wireless Ad-Hoc Networks

Ying Mao Jiayin Wang Bo Sheng
University of Massachusetts Boston
{yingmao, jane, shengbo}@cs.umb.edu

***Abstract*—Smartphones play an important role in mobile social networks. This paper presents a Mobile Message Board (MMB) system for smartphone users to post and share messages in a certain area. Our system is built upon ad-hoc communication model, and allows the users to browse the nearby information without pre-registration with any servers. Our algorithm design focuses on the message management on each phone considering its own schedule of turning the wireless device on and off. We present algorithms for two different cases to maximize the availability of the messages. Furthermore, we have implemented our solutions on commercial smartphones, and conducted experiments and simulation for evaluation. The results are supportive and shows that the MMB system is efficient and effective for location-based message dissemination.**

I. Introduction

Smartphones have become more and more popular in the past few years, and play an important role in in mobile social networks. This paper develops a new message dissemination system, called Mobile Message Board (MMB), for nearby smartphones to share message with each other based on ad-hoc communication model.

The message dissemination in the current social networks follows the client-server architecture. For example, when a Facebook/Twitter user or a forum user posts a message, the message is uploaded to a central server and hosted there. The server will check the user's friend list or followers and forward the message to them, or a user will browser the forum and view the post. This framework, however, does not exploit the strong but hidden link in a social network, the location. Generally, people in the proximity may share common interests. For example, the recommendation of a restaurant may not be interesting to the friends in another city, but could be very useful for a tourist nearby. In addition, a user may want to discover the events around him by browsing/searching the messages posted by other people nearby (probably strangers) without registering every interest with a server.

In addition, the client-server architecture does not work for some applications. For example, one of the major motivating applications in this paper is the Rogue Access Point detection. We consider in a public area, there might be some maliciously installed WiFi Access Points (AP) that lure users to connect to them. There have been solutions in the literature that can detect different types of rogue APs. The problem we consider is that if a user detects the existence of some rogue APs, how can he effectively notify other users in the area. With the traditional client-server model, the user can report the rogue APs to a server, and then other users, after recognizing their locations, can fetch this information from the server. However, this solution requires other users to connect to the Internet first in order to check with the server, and the connection to a rogue AP may have been established before contacting the server. Once connected with a rogue AP, the rest of the Internet traffic cannot be trusted. The rogue AP may redirect, alter, and manipulate the packets. Therefore, rogue AP detection may fail if it relies on the connection to a server.

We believe that phone-to-phone communication in the ad-hoc network model is an appropriate alternative to help local message dissemination. The ad-hoc model can effectively link nearby users even if they are all strangers to each other. In our prior work [1], we have developed a connectionless communication model built upon Bluetooth or WiFi-Direct that is suitable for short message dissemination. In this paper, we use this model to build a mobile message board for host messages in a particular area. We consider that each smartphone may have its own schedule to turn the Bluetooth/WiFi-Direct device on or off because of its energy strategy. The main challenge is how to select smartphones to host the messages so that the messages can be available in the system as long as possible. Ideally, we hope the messages are always accessible, i.e., when a new user joins the system at any time he can certainly read all the messages. However, when no smartphone keeps its wireless device always on, there is no guarantee of each message's availability. Our goal is to design appropriate strategies for each phone to manage the messages it hosts to maximizing the chance a user can read all the messages.

The rest of the paper is organized as follows. Section II introduces the background and related work. In Section III, we present the system model and problem formulation. Our algorithm design is described in Section IV. Section V includes the evaluation results based on both experiments and simulation. Finally, we conclude in Section VI.

II. Related work

In the research of intermittently connected networks, for example, Delay Tolerant Networks and Opportunistic Networks, a number of approaches have been proposed in different areas including link construction, data forwarding and content distribution and security [2]–[9].

To provide a secure network, Musubi [7] proposed a decentralized trusted social services on personal mobile devices. All the communication, in Musubi, is supported using

public key encryption thus leaking no user information to a third party. Focusing on enhancing the P2P connection, Point&Connect [9] implements pointing gestures of moving one device towards another in order to enable spontaneous device pairing. In addition, BubbleRap [5] utilizes group membership information to improve standard unicast routing. However, all these works are based on a network connected with Internet or P2P techniques, like WiFi-Direct and Bluetooth, that requires network construction and management.

In PASA [1], the authors introduce a new communication model named passive broadcast that utilizes the peer discovery process in P2P techniques to initialize and manage the ad-hoc network. This approach bypasses the real connection between the users by using the device name as an information carrier. The proposed Mobile Message Board (MMB) system takes advantage of the passive broadcast. However, relative to PASA that focuses on spreading the information, the MMB system mainly concerns about extending the activation period of the messages on the board.

One application of the MMB system is rogue access detection. Rogue Access Point is a wireless access point that has either been installed at a public area, like shopping malls and airports, without explicit authorization from a local network administrator, or has been created by a naive user for convenience. Multiple solutions have been proposed to prevent user from Rogue Access Point [10]–[14]. These solutions attempt to address the Rogue Access Points based on a centralized server that analyzing wireless traffic and reporting suspicious devices. However, those approaches require users to query their servers to check the database or connect to the access point first. In our scenarios, users may not or very costly have Internet access, for example, in airports while roaming abroad. If users connect to the Rogue Access Point first, they have already exposed to the attackers.

III. Problem Formulation

In this paper, we aim to develop a mobile message board (MMB) system without the support of the Internet infrastructure. We define a target area that hosts the message board, such as an airport terminal or a train station, a dining area with several restaurants, and a building with public hotspots. We assume that the users in the target area are within each other's communication range and they collaborate to host a virtualized message board. A user can leave messages on the board that will be shared with other nearby users, but are effective only in this area.

A. Ad-hoc communication between phones

Our problem setting and solution are based on the new connectionless communication model, *passive broadcast*, which has been introduced in [1]. Basically, every node conducts a periodical *scan* to fetch data from nearby nodes if available. When a node intends to send a message, it puts the message in a local buffer which will be sent to the nearby nodes later during their scanning processes. This communication model requires no connection to be established between any two nodes, thus is extremely efficient for message dissemination.

In this paper, the connectionless model is implemented based on the mandatary 'peer discovery' function in Bluetooth and WiFi-Direct. Specifically, we apply the field of 'device name' to carry the messages. When sending a message, the user will replace the Bluetooth or WiFi-Direct device name with the target messages. The messages will be delivered to other nearby phones when they conduct peer discovery.

We have prototyped this new model on commercial phones. According to our experiments, it does not affect regular Bluetooth/WiFi-Direct functions. For example, a phone using this message delivery model can still pair with other Bluetooth devices such as a headset or a keyboard.

B. States and Operations of each phone

According to the working procedure of Bluetooth module on the smartphone, in general, there are two states of each phone, **Bluetooth ON** and **Bluetooth OFF**. During the **ON** state, the users are willing to contribute to the local Mobile Message Board system through the ad-hoc communication with the others devices nearby. However, in the **OFF** state, the Bluetooth module is turned off to save energy, suggesting it cannot send or receive any messages in the system. In our setting, we assume each user already configures his own states schedule according to the energy budget and other user-specific factors.

The operations of a smartphone in the system includes: initial scan, message update, assignment and delegation.

Initial scan: Initial scan is the phase when all the users in the system are performing the peer discovery process to collect the parameters from nearby devices such as the state schedules and the limit of message each user can host. Based on the experiment in [1], the initial scan usually takes approximately 12.8 seconds.

Message update: The connectionless communication model utilizes the device name. However, the device name usually has a limited length, e.g., a Bluetooth's device name in Android can be up to 248 bytes. Multiple short messages can be merged into one "device name". But a large message will have to be fragmented to fit in and a phone can periodically update the device name to rotate multiple messages or fragments. In our solution, we use a special character string as the prefix to distinguish the device names that adopt our scheme from the regular ones.

Message assignment: Since each phone has its own states schedule, it is possible that the contributor who wants to share the message in the Mobile Message Board has very limited **ON** time. In order to yield a longer coverage or availability of a message so that others are more likely to receive it, the message owner will assign it to the other phones nearby that can help disseminate the message. The owner has to analyze each phones' state schedule and compare it with its own schedule to choose the best relay phones.

Message delegation: This operation is conducted when a phone is leaving the target area of the MMB. It has to choose a

set of phones in the system to delegate the active messages that are currently held by itself. This delegation keeps the messages remain active even when the owner and relay phones leave the system. To choose the right delegated phones, it compares the schedule with the current relay phones that hold the same message and others who are available to hold new messages. Message delegation is also used to improve the active length of the messages.

C. Objective and Constraints

In this paper, our goal is to develop efficient algorithms for *message assignment* and *message delegation* to maximize the availability of the active messages on the MMB. Specifically, we assume there are n users, $U = \{u_1, u_2, \ldots, u_n\}$, and k active messages, $M = \{m_1, m_2, \ldots, m_k\}$. Assume each user u_i follows a pre-configured schedule to periodically turn its device on and off. Let T_i be the duration for which u_i keeps its device on and I_i be the interval of two consecutive active periods. We use $L_i = T_i + I_i$ to represent the *cycle length* of each user u_i. Let $x_{ij} \in \{0, 1\}$ be the indicator of whether the user u_i hosts the message m_j. According to the schedule of each user u_i, we use $y_i(t)$ to denote the status of the user at the time point t, i.e., if u_i's device is turned on, then $y_i(t) = 1$; otherwise $y_i(t) = 0$.

Let L_{LMC} be the least multiple common of all the users' cycle lengths of their schedule, i.e.,

$$L_{LMC} = LMC(L_1, L_2, \ldots, L_n).$$

For each message m_j, let $c_j(t)$ indicate if the message is active at time t, and w_j be its weight indicating its importance or urgency. In addition, we include two threshold parameters: each phone can host at most τ distinct messages, and each message can have up to θ replicas in the system.

Therefore, our problem is formulated as follows:

$$\text{maximize} \sum_{j \in [1,k]} w_j \cdot \frac{\sum_t c_j(t)}{L_{LMC}} \tag{1}$$

$$s.t. \quad \forall i, j, \sum x_{ij} \cdot y_i(t) \geq c_j(t) \tag{2}$$

$$\forall i, \sum_j x_{ij} \leq \tau \tag{3}$$

$$\forall j, \sum_i x_{ij} \leq \theta \tag{4}$$

$$x_{ij} \in \{0, 1\} \tag{5}$$

The problem is NP-hard (due to the page limit, we omit the proof here.), and in the next section, we'll present our algorithms based on greedy strategy.

IV. Our Solution

In our solution, each user uses its phone's "device name" to carry the messages on the mobile message board (MMB) and enable the communication protocols with other users. Particularly, three categories of data are hosted on the device name:

- Payload messages: These messages are posted on the MMB and supposed to be accessible to nearby users.
- Control messages: These messages are part of particular communication protocols developed in our solution.
- Header: The header contains general profile information about the user such as the scanning schedule.

Since the length of the device name is limited, our solution allocates a fixed segment for each category of data. We assume that each user device can host at most θ payload messages with the knowledge of the average length of a message. In addition, each user reserves a certain space to hold one control message (the size of the control message will be introduced later when we present the protocols). The rest of the space on the device name will be allocated to the header information.

The following Fig. 1 illustrates an example of the data hosted on the device name. All the devices participating our protocols will use a special prefix in the beginning of the device name in order to distinguish themselves. The payload messages are listed with their owner and weight information separated by "##". In our solution, each device is identified by two bytes which is the hashed value of its MAC address. The control message starts with a list of recipients which is followed by the message content. Different from the payload messages, the control messages in our protocols usually are not broadcast messages, but with certain target recipients. Finally, the header contains its ID and the system parameters which include the state schedule and the maximum number of messages it can host.

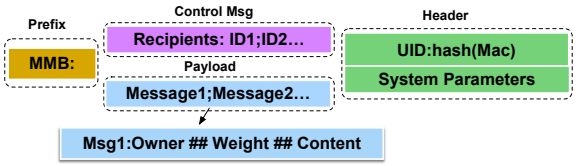

Fig. 1. Message Format carried by device name

The following Table I lists some notations we will use in the rest of this paper. In the following subsections, we present our algorithms in two different cases depending on if a coordinator node exists in the system.

u_i / m_j / w_j	the i-th user / the j-th message / the weight of m_j
T_i / I_i	the length of "on" period / "off" period of u_i
x_{ij}	the indicator of whether u_i hosts m_j
$y_i(t)$	the indicator of whether u_i is on at time t
$c_j(t)$	the indicator of whether m_j is available at time t
τ / θ	the limit of # of msgs per user / # of replicas per msg
AN_i	the set of *accessible neighbors* of u_i
HU_n	the set of messages that u_n currently hosting

TABLE I
Notations

A. With a coordinator

A *coordinator* in our system is defined as a node that is aware all other nodes' schedule. In another word, it has

Algorithm 1 Centralized Message Assignment

1: $L_{LMC} = LMC(L_1, L_2, \ldots, L_n)$
2: msgs=$\{m_1, m_2, \ldots m_k\}$
3: **for** message m_j **do**
4: $\mathcal{S}_j = \{1, 2, \ldots, n\}$, $\mathcal{A}_j = \{\}$, $\mathcal{F}_j[1..LMC] = 0$
5: **while** msgs $\neq \phi$ and $\cup \mathcal{S}_j \neq \phi$ **do**
6: $OPT = i* = j* = 0$
7: **for** message m_j in msgs **do**
8: **for** $i \in \mathcal{S}_j$ **do**
9: $c = 0$
10: **for** $t = 1$ to L_{LMC} **do**
11: **if** $\mathcal{F}_j[t] = 0$ and $y_i(t) = 1$ **then**
12: $c = c + 1$
13: **if** $c \cdot w_j > OPT$ **then**
14: $OPT = c \cdot w_j$, $i* = i$, $j* = j$
15: $\mathcal{A}_{j*} = \mathcal{A}_{j*} + i*$
16: $CM_{j*} = CM_{j*} + 1$, $CU_{i*} = CU_{i*} + 1$
17: **if** $CM_{j*} = \theta$ **then**
18: $msgs = msgs - m_{j*}$
19: **if** $CU_{i*} = \tau$ **then**
20: **for** message m_j **do**
21: $\mathcal{S}_j = \mathcal{S}_j - u_{i*}$
22: **for** $t = 1$ to L_{LMC} **do**
23: $\mathcal{F}_{j*}[t] = \mathcal{F}_{j*}[t] \mid y_{i*}(t)$

successfully scanned all other nodes' information during the initial scan. Specifically, the coordinator recognizes the $y_i(t)$ for all $u_i \in U$. When such a coordinator exists in the system, it will be responsible for assigning the messages to each smartphone and maximize the total weights as shown in objective 1.

We develop an algorithm 1 for the coordinator to complete the assignment. The basic intuition is to apply greedy strategy to select a subset of nodes to host each message. First, it calculates the least multiple common of all the users' cycle lengths of their schedule and combines all the active messages into a set named msgs (lines 1–2). For every message $m_j \in msgs$, it initializes the candidate smartphone set, $\mathcal{S}_j$, which contains all the available users, the selected user set, $\mathcal{A}_j$, which is empty initially and activation function, $\mathcal{F}_j$, that indicates whether the message m_j is discoverable in the MMB system (lines 3–4). The main part of the algorithm is a while loop which will enumerates every message. The loop terminates when all the messages are assigned ($msgs = \phi$), or all the phones are fully loaded ($\mathcal{S}_j = \phi$). The parameters OPT, $i*$ and $j*$ are temporary variables where OPT stores the current optimum for this message m_j; $i*$ is the current selected user id and $j*$ currently message id (lines 5–6). For every candidate in the $\mathcal{S}_j$, the coordinator calculates the activation period indicator, c, and the total weight, $c \times w_j$. If the $c \times w_j$ is larger than the current optimum OPT, then it updates the temporary parameters (lines 8–14). When m_j is assigned to u_i, it updates the m_j's selected user set, $\mathcal{A}_j$, the counter of currently hosting message, CM_j^* and the counter of currently hosting users, CU_i^* (lines 15–16). The algorithm monitors these two counters CM_j^* and CU_i^* to check if they reach the preset thresholds and updates the $msgs$ and $\mathcal{S}_j$ sets when needed (lines17–21). Finally, the coordinator refreshes activation function for m_j (line 22).

B. Without a coordinator

In some scenarios, a coordinator may not exist in the system because of the misalignment of the state schedules among all the nodes. In this subsection, we present a solution for the MMB without a coordinator. Our solution includes two stages: initial assignment and message delegation, with the objective of maximizing the message's active period.

The initial assignment is performed by every message owner. Since the owner may have very limited ON period (according to its state schedule), whenever a user generates a message, it runs through the initial assignment to choose the best holders for it.

Algorithm 2 Initial Assignment (by the message owner u_o)

1: users = $\{u_i \mid CU_i < \tau$ and $u_i \in AN_o\}$, $\mathcal{A} = \{\}$, $CM = 0$
2: **while** users $\neq \phi$ and $CM < \theta$ **do**
3: $OPT = i* = 0$
4: **for** user u_i in users **do**
5: c=0
6: **for** $t = 1$ to L_{LMC} **do**
7: **if** $\mathcal{F}_j[t] = 0$ and $y_i(t) = 1$ **then**
8: $c = c + 1$
9: **if** $c > OPT$ **then**
10: $OPT = c$, $i* = i$
11: $\mathcal{A} = \mathcal{A} + u_{i*}, CU_{i*} = CU_{i*} + 1$, $CM = CM + 1$
12: **if** $CU_{i*} = \tau$ **then**
13: users = users - u_{i*}

Algorithm 2 describes the details of the initial assignment stage. Firstly, the owner, u_o, initializes the selected user set, $\mathcal{A}$ and counter of replicas, CM. In addition, it needs to combine its accessible neighbors into a set named $users$. This set excludes the users that have already reached their maximum number of messages threshold (line 1). When the set $users$ is non-empty and CM is still under the threshold, for every $u_i \in users$, it computes the activation period if u_i is chosen to be a message holder by using the message's activation function in a L_{LMC}. The algorithm continues until it finds the largest c among all the $u_i \in users$ (lines 2–10). After choosing the candidate u_i from AN_{u_0}, we update selected user set and the counter of replicas for this message. Moreover, we add u_i^* to the selected user set and refresh the number of hosted messages of u_o. Finally, it keeps monitoring whether u_i has meet its limit. If it is, we remove it from the candidate user set. (lines 11–13).

Without the coordinator's assistance, the owner may fail to choose the best users to host the message due to the limit of accessible neighbors. In this case, after the initial assignment, the selected users keep tracking to see if there is any better

Algorithm 3 Message Delegation (by the selected u_o)

1: users = $\{u_i \mid CU_i < \tau$ and $u_i \in AN_o\}, \mathcal{A} = \{\}, CM = 0$
2: $holders_{m_j}$ = $\{u_n \mid m_j \in HU_n\}, OPT$
3: **while** $holders_{m_j} \neq \phi$ and excludes u_o **do**
4: **for** Every $u_n \in holder_{m_j}$ **do**
5: $n = n^*$
6: **for** $t = 1$ to L_{LMC} **do**
7: **if** $y_i(t) = 1$ **then**
8: $\mathcal{F}_j(t) = 1$
9: $holders_{m_j} = holders_{m_j} - u_{n^*}$
10: **while** users $\neq \phi$ and $CM < \theta + 1$ **do**
11: **for** Every u_i in users **do**
12: **for** $t = 1$ to L_{LMC} **do**
13: **if** $y_i(t) = 1$ **then**
14: $\mathcal{F}_j(t) = 1$
15: $OPT^* = \sum_1^{LMC} \mathcal{F}_j(t)$
16: **if** $OPT^* > OPT$ **then**
17: $OPT = OPT^*, DU = u_i$

user that can hold the message and lengthen the total active time.

Algorithm 3 illustrates how u_o delegate the message m_j to a better user. First, it initializes the users set, the selected user set and message m_j's current holder set which is the set $\mathcal{A}$ from algorithm 2 (lines 1–2). Then, we calculate the activation function , $\mathcal{F}_j(t)$, without u_o's hosting (lines 3–10). Next, the algorithm computes whether the active time becomes longer if one of the users in u_o's available neighbors is chosen to host m_j. After enumerating all the available neighbors, we select the user that can return the maximum OPT (lines 11–13).

V. Implementation and Evaluation

In this section, we will first introduce the system implementation of our MMB system and then present the performance evaluation results from our experiments and simulation.

A. System Implementation

In the implementation, we take advantage of the MMB system in the application of warning the Rogue Access Point (RAP) in an airport when users are not able to or very costly to connect to the Internet. In this scenario, users report the RAP by changing their Bluetooth name and publish the information to the MMB. When a user finds a possible RAP, it changes its Bluetooth name to "MMB:Mac:X", where MMB is a prefix that is used to filter out unrelated Bluetooth devices, Mac is the RAP's MAC address and X stands for 1 or 2 which indicates rogue access point or slow speed access point.

We implement the prototype as an Android application for smartphone. However, for off-the-shelf phones, as long as they have Bluetooth modules and can change their namespaces, they can contribute to the system by reporting the suspected access points. Fig. 2 is a set of screen shots of the Android application. Upon opening the application, the users need to initiate scanning to discover the nearby MMB warning messages (the left figure of Fig. 2). Then, they can start discovering the active nearby Wi-Fi hotspots and the reported access points are clearly marked (the middle figure of Fig. 2). When a user attempts to click "connect" button, the warning message will pop up (the right figure of Fig. 2).

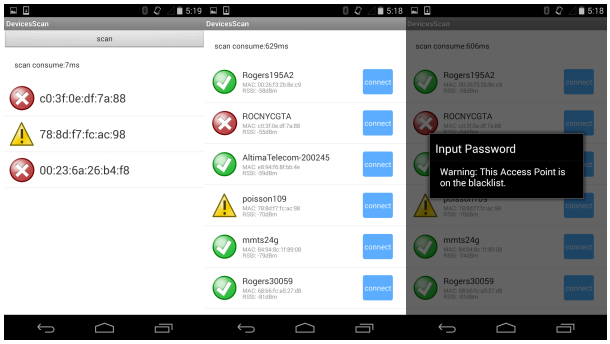

Fig. 2. Rogue Access Point warning application of MMB system

B. Environment Settings

1) Experiments: In the experiments, we use 6 Android smartphones that include LG Nexus 4, Nexus 5, Google Nexus 7 and Samsung Galaxy S4. During the experiment, we distribute the smartphones on the desks in our lab which is around $20m^2$ to form a fully connected network. The state schedule, T_i and I_i, is preloaded in each phone. In addition, other user settings, like messages and thresholds, are pre-configured.

2) Simulation: We use the Crawdad dataset from Dartmouth [15] for the simulation. The particular dataset we use is Sigcomm 2009 trace that contains Bluetooth encounters, opportunistic messaging, and social profiles of 76 users of MobiClique [16] application. Based on the length of encounter time and activity, we filter out the users who only appear in a short period and lack of activities. Then, there are 52 users left whom we consider as the valid clients. We drive the encounter length and number of social messages from the dataset as the parameters. In the simulation, we randomly pick up the users from the valid clients pool. Due to lack of the Bluetooth OFF state data, we assign each user a OFF length randomly so that each user has a state schedule. Finally, the parameters are fed to our simulator to examine the Mobile Message Board system.

C. Performance Evaluation

In this subsection, we present the evaluation results from both Android smartphones experiments (small scale) and simulation (large scale). We consider the following four different cases for our evaluation.

- **Case 1**: Only one message generated by one user in the MMB system.
- **Case 2**: Multiple messages generated by the same user.
- **Case 3**: One message generated by multiple users.
- **Case 4**: Multiple messages generated by different users.

The major performance metric we exam is the Activation Rate (AR) for each message. For each $m_j \in M$, its activation

rate is defined as,

$$AR_j = \frac{\sum_{t\in[0,L_{LMC}]} c_j(t)}{L_{LMC}} \quad (6)$$

where, according to Table I, $c(t)$ is the indicator of whether m_j is active at time t and L_{LMC} is the function that returns the least multiple common of all the users' state schedules in the MMB system.

1) Experiments: We manually set the required state schedule and randomly choose the limit number of messages per user in the range from 1 to 3 for each phone as shown in Table II. In the experiments, we mainly focus on the parameters', τ and θ, impact on activation rate. (the parameters' definitions are in Table I). As for the message owners, we randomly pick from the six candidates for different cases.

u_{id}	T	I	τ	u_{id}	T	I	τ
1	10	10	3	2	15	10	1
3	15	25	1	4	10	20	1
5	15	20	2	6	10	15	2

TABLE II
PARAMETERS

Fig. 3 plots the average activation rate per message under different θ. As the figure displays, if the $\theta = 1$, it suggests each message only has one host. Except case 3, the activation rate of the other three cases is very low (less than 0.6). The reason lies in the fact that for the case 1, 2, and 4, each message can only has one host controlled by θ. However, in case 3, since different users generate the same message, it can bypass the limit of θ. The activation rate grows along with the replicas increasing since the more hosting users get involved, the less OFF state is in each period.

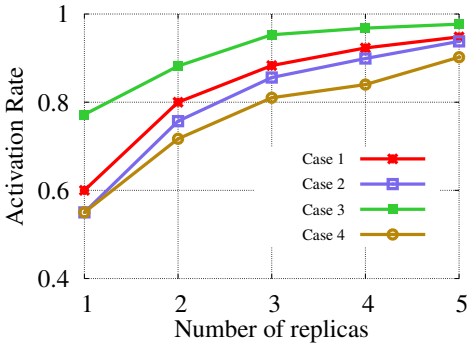

Fig. 3. Activation Rate versus number of replicas (θ)

Then we focus on the impact of τ which controls the maximum number of messages each user can host. As shown in Table II, τ is randomly selected from 1 to 3. Fig. 4 compares the case 4 under the random setting above and the maximum setting that assigns every user the limit of 3 messges. The figure indicates that the maximum τ setting can improve the activation rate. The difference of state schedules results in the improvement since the user with largest ON state portion, user 2, can hold more messages (1 v.s 3).

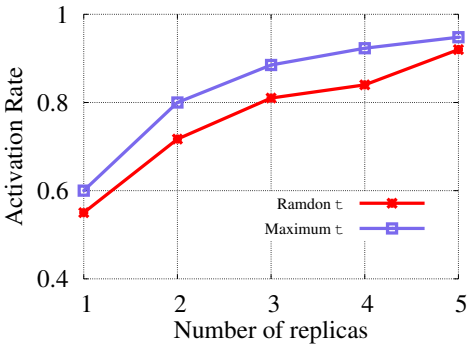

Fig. 4. Comparison of random and maximum τ values (case 4)

2) Simulation: In the simulation, we use the parameters that derived from the trace. However, it lacks the records of Bluetooth OFF state and messages replicas. Therefore, we randomly pick the OFF state from 5 to 50 and the number of replicas from 1 to 5.

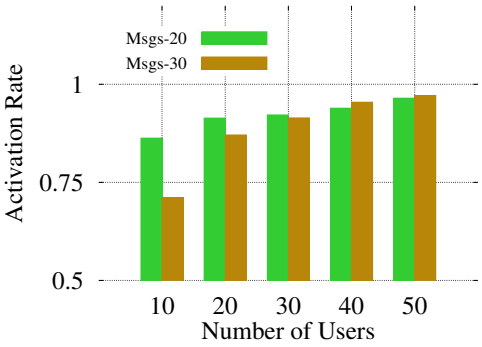

Fig. 5. Activation Rate (case 4) with random number of replicas

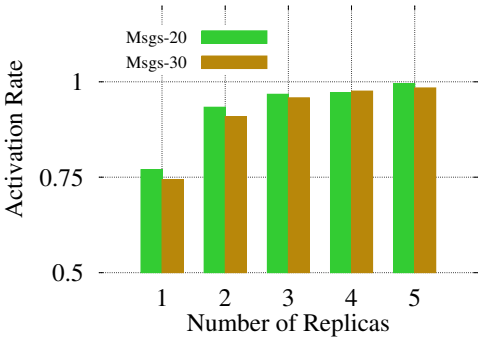

Fig. 6. Activation Rate (case 4) with fixed number of replicas and 30 users

Fig. 5 illustrates activation rates (with coordinator) under different number of users. In addition, the number of messages is set to 20, 30. Consequently, the activation rate grows with the user size increase. The more users are in the system, the more choices are for each message. The figure also implies a fact that the activation rate is higher for 20 messages than 30.

It can be attributed that, with fixed user size, less messages in the system results in more choices for each message.

Fig. 6 plots the activation rate (with coordinator) under different fixed replicas (all messages has the same θ) with 30 users. Since the user size is fixed, the activation rate goes up along with the replicas. Moreover, the Msg-20 experiment has a higher activation rate than Msg-30. It reflects the fact that with the same user group, the smaller message set size results in more choices for each message.

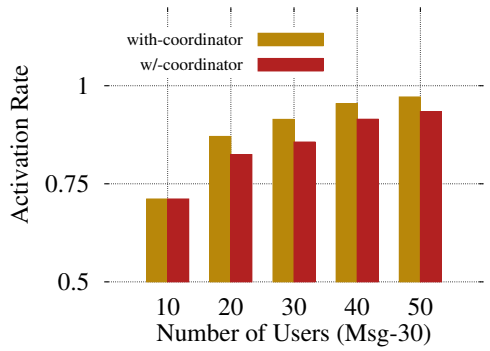

Fig. 7. Activation Rate (case 4) with fixed number of replicas and 30 Msgs

Next, we compare the two approaches, with and without the coordinator. To compensate the absent of coordinator, in the second approach, the message, after initial assignment, will be delegated to other users if it can improve the activation period. It takes some time for the delegation process. Therefore, the simulation is configured to keep running for two periods (two L_{LCM}). Fig. 7 compares the two approaches with Msg-30 under different size of users. As shown on the figure, two approaches achieve the same performance with 10 users. This is because the central message assignment by the coordinator and initial message assignment by the owners choose the same holders for the messages. Without the coordinator's assistance, the second approach fails to choose the optimal users for some messages in the other settings that results in the decrease of average activation rate in the system. However, message delegation can compensate some of the decrease. For example, in user 20 setting, the average activation rate is 0.77 after message initial assignment. The overall activation rate improves to 0.82 in two periods after the delegation.

VI. Conclusion

This paper presents a Mobile Message Board (MMB) system for smartphone users to share messages in a target area. Our solution is built on ad-hoc communication model without the support from the Internet. We present algorithms that appropriately manage the messages on each participating phone to maximize the message availability in the system. In addition, we have implemented our solution on off-the-shelf phones. Our evaluation based on experiments and simulation shows that our system is efficient and effective for disseminating messages in the proximity.

References

[1] Ying Mao, Jiayin Wang, Joseph Paul Cohen, and Bo Sheng. Pasa: Passive broadcast for smartphone ad-hoc networks. In *Computer Communication and Networks (ICCCN), 2014 23rd International Conference on*, pages 1–8. IEEE, 2014.

[2] Te-Yuan Huang, Kok-Kiong Yap, Ben Dodson, Monica S. Lam, and Nick McKeown. PhoneNet: a phone-to-phone network for group communication within an administrative domain. In *Proceedings of the second ACM SIGCOMM workshop on Networking, systems, and applications on mobile handhelds*, MobiHeld '10, 2010.

[3] Zygmunt J. Haas, Joseph Y. Halpern, and Li Li. Gossip-based ad hoc routing. *IEEE/ACM Trans. Netw.*, 14(3):479–491, 2006.

[4] Elizabeth M. Daly and Mads Haahr. Social network analysis for routing in disconnected delay-tolerant manets. In *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '07, 2007.

[5] Pan Hui, Jon Crowcroft, and Eiko Yoneki. Bubble rap: Social-based forwarding in delay-tolerant networks. *IEEE Transactions on Mobile Computing*, 10(11), November 2011.

[6] Adam C. Champion, Zhimin Yang, Boying Zhang, Jiangpeng Dai, Dong Xuan, and Du Li. E-smalltalker: A distributed mobile system for social networking in physical proximity. *IEEE Trans. Parallel Distrib. Syst.*, 24(8), August 2013.

[7] Ben Dodson, Ian Vo, T.J. Purtell, Aemon Cannon, and Monica Lam. Musubi: disintermediated interactive social feeds for mobile devices. In *Proceedings of the 21st international conference on World Wide Web*, WWW '12, pages 211–220, 2012.

[8] Jakob Eriksson, Hari Balakrishnan, and Samuel Madden. Cabernet: Vehicular Content Delivery Using WiFi. In *14th ACM MOBICOM*, San Francisco, CA, September 2008.

[9] Chunyi Peng, Guobin Shen, Yongguang Zhang, and Songwu Lu. Point&Connect: intention-based device pairing for mobile phone users. In *Proceedings of the 7th international conference on Mobile systems, applications, and services*, MobiSys '09, 2009.

[10] AirDefense. http://www.airdefense.net/.

[11] AirWave Management. http://www.airwave.com/.

[12] CiscoWorks Wireless LAN Solution Engine. http://www.cisco.com/c/en/us/products/cloud-systems-management/ciscoworks-wireless-lan-solution-engine-wlse/index.html.

[13] AirMagnet. http://www.airmagnet.com/.

[14] NetStumbler. http://www.NetStumbler.com/.

[15] Crawdad Dartmouth. http://crawdad.org/index.html.

[16] Anna-Kaisa Pietiläinen, Earl Oliver, Jason LeBrun, George Varghese, and Christophe Diot. Mobiclique: Middleware for mobile social networking. In *Proceedings of the 2Nd ACM Workshop on Online Social Networks*, WOSN '09, pages 49–54, New York, NY, USA, 2009. ACM.