Wireless Netw
DOI 10.1007/s11276-015-0984-2

A cooperative forwarding scheme for social preference-based selfishness in mobile social networks

Sun-Kyum Kim[1] · **Ji-Hyeun Yoon**[1] · **JunYeop Lee**[1] · **Gi-Young Jang**[1] · **Sung-Bong Yang**[1]

Abstract Most schemes in mobile social networks (MSNs) assume that nodes simply forward messages without considering selfishness. We therefore first devise social preference-based selfishness for MSNs by which nodes decide to drop or keep (forward) and replace messages to save buffer space according to the message preference and the communities of nodes. We then propose a novel cooperative forwarding scheme for social preference-based selfishness in MSNs, *the social preference-aware forwarding scheme* (SPF) incorporates the proposed message forwarding scheme and a buffer replacement policy for the message preference. It takes advantage of social information with the home-cell community-based mobility model. Considering the contact probability and buffer replacement policy for the message preferences, SPF, therefore, efficiently delivers messages to the destination by reflecting the degree of selfishness to which nodes cooperatively manage their buffer spaces and how frequently and how recently they meet. Consequently, all nodes can cooperatively drop or keep (forward) and replace the messages in the buffer spaces for the message preferences in SPF. SPF outperforms Epidemic, PRoPHET, and SimBet in terms of delivery ratio, network traffic, buffer space, hop count, and replacement frequency.

Keywords Social selfishness · Forwarding · Routing · Mobile social network (MSN) · Cooperative

1 Introduction

For Internet of Things [1, 2], various research has been studied in sensors and wireless communications such as wireless sensor networks [3–6], wireless mesh networks [7, 8], soft-defined wireless networks [9], cognitive radio networks [10, 11], green communications [12, 13], composite radio environments [14], online social networks [15], mobile ad-hoc social networks [16] and (opportunistic) mobile social networks [17]. Among them, mobile social networks (MSNs)—also known as opportunistic networks [17] or pocket switched networks [18] —have quickly emerged and have attracted significant attention for use in wireless communication paradigms. MSNs are networks developed from mobile ad hoc networks (MANETs) [19] and delay-tolerant networks (DTNs) [20] with social properties of human behavior [21, 22]. Like DTNs, MSNs likewise suffer from intermittent connectivity and frequent disconnections in sparse environments due to their short transmission range and node mobility [23].

Just as people can be selfish, nodes may selfishly behave; they may refuse to forward some messages and may use their limited resources for their own benefit because of resource constraints, such as battery and storage limitations. The nodes would like to obtain benefits from other nodes; however, their own resources may not be available to help others [24]. Therefore, we address the

✉ Sung-Bong Yang
sbyang@cs.yonsei.ac.kr

Sun-Kyum Kim
skyum@yonsei.ac.kr

Ji-Hyeun Yoon
y_chick@yonsei.ac.kr

JunYeop Lee
kotzyi@yonsei.ac.kr

Gi-Young Jang
centertong@yonsei.ac.kr

1 Department of Computer Science, Yonsei University, Seoul, Korea

Published online: 09 June 2015

problem of selfishness in the perspective of forwarding in MSNs. So far minimal research on selfishness for contact nodes has been introduced and no research has considered selfishness for how much the nodes favor the messages for MSNs to the best of our knowledge. Hence, we adopt and modify the "degree of selfishness" (i.e., the message preference) from handling selfishness in replica allocation [24]. In this environment, a node may decide to drop or keep (for forwarding) and replace some messages according to both the message preferences and the communities of the nodes. We concentrate on when and how a node drops, forwards a message, and replaces a message.

In addition, it is very essential for nodes to cooperate with each other in such a selfish environment under which nodes have limited resources. Thus, we focus on which nodes cooperate as relay nodes to forward messages and how efficiently they manage their buffer spaces. To this end, we propose a novel cooperative forwarding scheme for social preference-based selfishness, the social preference-aware forwarding scheme (SPF) that incorporates the proposed message forwarding scheme and buffer replacement policy.

For message forwarding, SPF leverages the contact probability with respect to the destination node for the message preference; it is computed with the refined degree centrality [25], inter-contact time [17], and available buffer space. For the buffer replacement policy in SPF, when the buffer space of a node is full, it replaces the oldest message among the lowest preference messages with a new one.

We conduct extensive simulations for experiments with the NS-2 network simulator and compare Epidemic [26], PRoPHET [39], and SimBet [34] in terms of delivery ratio, traffic, transmission delay, hop count, and replacement frequency. The experimental results show that SPF outperforms these schemes for most cases.

The main contributions of this paper are outlined below.

1. We devise novel social preference-based selfishness in MSNs. A node may decide to drop or keep messages and replace some messages according to its message preferences to maintain its buffer space more efficiently.
2. We propose a novel cooperative forwarding scheme that incorporates the proposed message forwarding scheme and the buffer replacement policy. In particular, it takes advantage of the contact probability by considering the refined degree centrality, inter-contact time, and available buffer space.
3. We enables SPF to take advantage of snooping for selfishness detection. Each node can estimate the relay nodes' message preference by snooping messages.

The remainder of this paper is organized as follows. In Sect. 2, we discuss related work. After introducing the system model and assumptions in Sect. 3, we describe the social preference-based selfishness in Sect. 4 and the proposed scheme in Sect. 5. The performance evaluation is presented in Sect. 6. Finally, our conclusions and future work are presented in Sect. 7.

2 Related work

Existing routing schemes for MANETs [19] include dynamic source routing, ad hoc on-demand distance vector, split multipath routing, shortest multipath source, and AntHocNet. However, none of these schemes can be applicable to MSNs regardless of their improved performances because no stable paths may exist between the source and destination nodes in MSNs.

The opportunistic routing schemes for MSNs can be classified into two categories: zero-knowledge schemes and non-zero-knowledge schemes. Zero-knowledge schemes do not exploit social information, whereas non-zero-knowledge schemes, which are used in MSNs, take advantage of information about nodes' behaviors or social relations to make decisions for forwarding messages. Zero-knowledge schemes include Epidemic [26], Spray and Wait [27], Homing Spread [28], backpressure with adaptive redundancy (BWAR) [29], backpressure-based routing [30], CodePipe [31], spatial reusability-aware single-path routing (SASR) and anypath routing (SAAR) [32], and the hotspot-based forwarding scheme (HFS) [33].

Various non-zero-knowledge schemes have been proposed for MSNs [34–43]. These approaches can be further classified into three schemes: centrality/similarity-based, social context-based, and probability-based. Centrality/similarity-based schemes include SimBet [34], Bubble Rap [35], and social-aware networking (SANE) [36]. Social context-based schemes include Label [37] and HiBop [38]. Finally, Probability-based schemes include PRoPHET [39], PeopleRank [40], MobySpace [41], and delegation forwarding [42].

Several issues on resource constraints for forwarding have been studied for wireless communications; quality of service (QoS) [43–47], scheduling [48], trust management [1], and so on. Among them, several approaches to selfishness have been proposed [49–54] for MSNs. The socially selfish-aware routing (SSAR) scheme [49] provides two types of selfishness: individual and social. Nodes are willing to forward messages to other nodes with strong ties. Meanwhile, other nodes do not forward messages to others with weak ties. The former and latter ones are social selfishness and individual selfishness, respectively. Give2Get [50] considers a modified Epidemic scheme and employs a forwarding scheme using Nash equilibrium to control the number of replicas. Manam et al. [51] set selfishness according to two types of node communication ranges (i.e., large and small ones). Li et al. [52] evaluated Epidemic in social selfishness environments [49]. For sparse sensor nets (i.e., wireless sensor networks), Yang et al. [53] proposed rational selfishness, which means that a phone owner is willing to relay sensor data as long as he or she can benefit from it. This conceptually differs from social selfishness. Finally, Hui et al. [54] conducted research on the impact of altruistic behavior on communication throughput in MSNs. The concept of "altruism" is similar to that of "willingness" [49], whereby a node forwards messages for other nodes. However, unlike "willingness," "altruism" is used in an absolute sense and denotes the probability that a node will forward a received message. In addition, their focus is on the impact of altruism on opportunistic communication.

In MANETs, and even in DTNs and MSNs, individual selfishness has been widely studied [49]. The solutions have been classified into three categories: credit-based, reputation-based, and gaming-based approaches. The main idea is to stimulate users to forward messages for others. However, because these schemes do not consider "social information," they cannot be directly applied to MSN forwarding for selfishness.

3 System model and assumptions

We model MSNs as social contact graph $G = <V, E>$, where vertex set V and edge set E consist of all nodes and all links between the nodes, respectively. Whenever the nodes meet, they update information, including their social contact graphs. Weight w of an edge indicates the number of contact counts; that is, how many times two encountered nodes are connected. In order to calculate the contact probability for forwarding, we refine the degree centrality [25]. Figure 1 shows an example of a social contact graph that consists of N_1, N_2, N_3, N_4, and N_5. Basically, the degree centralities of N_1, N_2, N_3, N_4, N_5, N_6, and N_7 are 4, 2, 2, 1, 2, 2, and 1, respectively. Here, however, these are 16, 9, 13, 2, 4, 7, and 5 because each node considers the contact counts (i.e., edge weights) of the encountered nodes. Afterwards, we use the refined degree centrality and inter-contact time to calculate the contact probability for forwarding. We explain this in more detail in Sect. 5.

Each node in our MSN has a unique identifier. The nodes are denoted by $N = \{N_1, N_2, N_3, \ldots, N_m\}$, where m is the number of nodes in the network. The set of encountered nodes of node N_i is denoted by S_i. For simplicity, because nodes have limited bandwidth, power, and computational capability, we do not consider these resources. Instead, we assume that each node has limited buffer space for messages from other nodes. Again for simplicity, all messages are assumed to be the same size. In addition, a specified message lifetime (i.e., time to live, or TTL) is limited to four because it is sufficient for delivering the message to the destination. After a lifetime expires, a message should be dropped and cannot be available.

We exploit the Home-cell Community-based Mobility Model (HCMM) [55] for the devised MSN environments as the node mobility model in this paper because it is a generally used model for the spatial and temporal properties of human mobility in social information. Each node belongs to a community. The community in which a node is initially located is called the node's home community. Each node frequently visits its home community and infrequently visits other communities. We assume that each node knows whether other nodes belong to its home community. Nodes know minimal global information except for node identifiers. In addition, each node N_i has a community ID H_i [37], which denotes its home community index in which the number of communities is four. Because we assume that each node has positioning system equipment to determine its speed and location, it is aware of its own speed and current location. Each node can periodically measure its location.

4 Social preference-based selfishness

In the real world, people may receive messages that they like or dislike. They decide to behave according to whom the messages are from and how much they favor the ones.

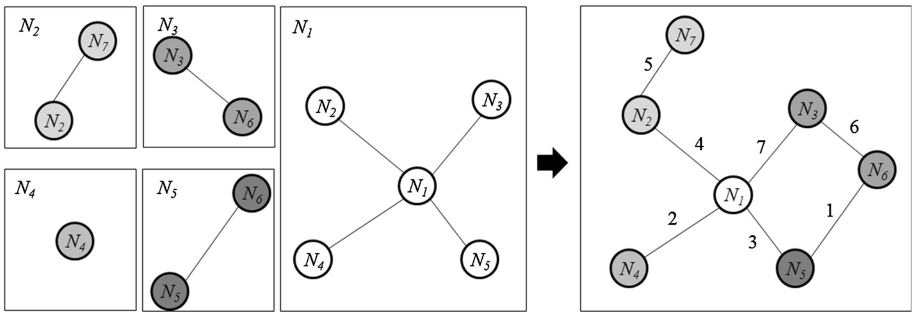


Fig. 1 Social contact graph

Similarly, nodes may be willing to forward or receive messages based on who generates them (i.e., whether the source node belongs to the same community as the node) and if they "like" messages, or they may not forward or receive the messages they "dislike." In order to appropriately handle the forwarding process and buffer policy, the modified degree of selfishness with nodes communities is classified into six degrees: "same community/like," "same community/partially like," "same community/dislike," "different community/like," "different community/partially like," and "different community/dislike." Note that "same community" and "different community" indicate the relationship of the source node and the current node receiving the message. We call this "social preference-based selfishness" and assume nodes behave according to the above social preference-based selfishness. As a result, the nodes appropriately decide to drop or keep (forward) and replace messages to save their buffer space according to the message preferences and the communities of nodes in the devised environments. Table 1 shows a node's forwarding behavior for social preference-based selfishness.

When the node receives the "like" or "partially like" message, and the source node generating the message belongs to the same community, the received node keeps it. Otherwise, the node drops the "dislike" message as soon as it is received. On the other hand, the node retains only the "like message" when the source node belongs to a different community. Moreover, when the buffer is full, the node replaces the "partially like" message first; it then replaces the "like" message when the node no longer has the "partially like" message in its buffer.

For simplicity, we assume that all messages are classified into five subjects to take advantage of the message preferences. Each message must be forwarded to the destination node. Because each message has its own destination node ID, each node generates the message belonging to the subject with its "first preference". In other words, the destination node ID in each message belongs to the subject with its first preference, but the source and destination nodes do not belong to the same subject. Additionally, each node is assumed to know which messages belong to the subject with its first preference. All messages are selected uniformly at random as the hierarchical preferences among the following subjects with the uniform distributions.

$$\text{Subject} = \{\text{Computer, Biology, Physics, Chemistry, Architecture}\}$$

For example, when there are 40 nodes in the network, there are 8 messages of the same subject. When a node selects a subject as its first preference, it means the node "likes" messages belonging to the selected subject, and the node generates such messages. When the buffer is full, the node does not replace the message with its first preference except for when only messages with its first preference are in the buffer. When the node selects the next two subjects as its second and third preferences, meaning that the node "partially likes" messages belonging to subjects with its second and third preferences, it forwards the messages. However, when the buffer is full, the node replaces the message with its third preference first. Finally, when the node chooses the last two subjects as its fourth and fifth preferences, it means that the node "dislikes" messages belonging to the subjects with its fourth and fifth preferences; when the node receives the messages, it immediately replaces them. We assume that each node knows which message belongs to its first preference. We make an example of node N_i's selfish behavior in Fig. 2 when we assume N_i belongs to the same community as the source node.

N_i selects Computer, Biology, Physics, Chemistry, and Architecture as first, second, third, fourth, and fifth preferences, respectively. Because Computer is N_i's first preference, N_i generates and retains a message belonging to the Computer subject. When N_i's buffer is full, it does not replace the message except when it receives a message with the same preference of one. Additionally, Biology and Physics are selected as second and third preferences, respectively. When N_i receives the message from the source node belonging to the same community as N_i, N_i keeps the message. N_i replaces the message belonging to the "Physics" subject with the new one first when N_i's buffer is full because the preference of Biology is higher than that of Physics. On the other hand, when N_i receives the message from the source

Table 1 Node forwarding behavior for social preference-based selfishness

Source node's community	Preferences	Retaining
Same	Like	O
Same	Partially like	O
Same	Dislike	X
Different	Like	O
Different	Partially like	X
Different	Dislike	X

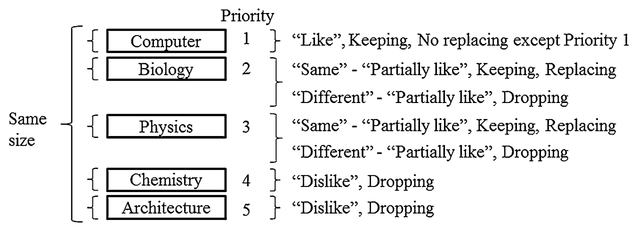

Priority
Same size
Computer
1
"Like", Keeping, No replacing except Priority 1
Biology
2
"Same" - "Partially like", Keeping, Replacing
"Different" - "Partially like", Dropping
Physics
3
"Same" - "Partially like", Keeping, Replacing
"Different" - "Partially like", Dropping
Chemistry
4
"Dislike", Dropping
Architecture
5
"Dislike", Dropping

Fig. 2 Example of N_i's selfish behavior for social preference-based selfishness

node belonging to the same community that N_i does not generate, N_i drops the message. Finally, N_i selects Chemistry and Architecture as tis fourth and fifth preference, respectively, and it immediately receives and drops messages relating to the Chemistry and the Architecture subjects.

5 Proposed scheme

In this section, we detail the proposed scheme, social selfishness detection, the forwarding value consisting of the contact probability and buffer weight value, the buffer policy, and the information exchange protocol of the proposed scheme, SPF.

5.1 Overview

SPF obeys the rules of typical message forwarding; each node forwards a message with respect to the destination node via relay nodes. In MSNs, nodes do not maintain a routing table for forwarding messages because there may be no stable path between the source and destination nodes. Thus, it is important to find proper relay nodes for successful delivery of the message within a reasonable delay. Especially, because nodes first receive then drop messages, they "dislike" messages in our devised selfish environments, which causes low traffic efficiency for forwarding. Thus, the key challenge is to appropriately detect nodes' selfishness and select relay nodes to forward the messages. To solve this problem, the proposed scheme detects nodes' selfishness by snooping and it uses the forwarding value consisting of contact probability by considering the refined degree centrality, inter-contact time, and buffer weight value to select relay nodes. Because nodes initially lack global information, they obtain their forwarding information about those metrics by exchanging an information vector during the forwarding process as much as possible. This means that the performance is weak at the beginning but improves as time passes because the nodes continuously obtain the global information by updating the information.

The message forwarding ends when the message is delivered to the destination node. In the following sections, we define the forwarding values, including the contact probability and buffer weight value, as well as social selfishness detection and the buffer replacement policy.

Table 2 outlines notations for understanding SPF. We detail the notations in the next sections.

5.2 Contact probability

Contact probability is a significant factor in forwarding a message to the destination. A node with a high contact probability with respect to the destination is likely to forward a message to the destination. Basically, because nodes drop or keep and replace the message according to the defined social preference-based selfishness, which affects performance, SPF uses the new contact probability for forwarding the messages. The contact probability considers the refined degree centrality and inter-contact time; these reflect how frequently and recently the nodes meet one another. We incorporate these metrics using the following equation.

$$P(i,j) = \frac{1}{I(i,j)} \times \frac{c(i,j)}{z_i}, \qquad (1)$$

where $P(i, j)$ indicates the contact probability of N_i and N_j, $I(i, j)$ is the inter-contact time between N_i and N_j, Z_i shows the refined degree of N_i, and $C(i, j)$ represents the contact counts of N_{i-} and N_j. Note that $Z_i = \sum_{j=1}^{k} C(i, j)$, where j and k are the number of encountered nodes and the number of edges of N_i, respectively. In other words, the refined degree Z_i is the total number of contact counts, and $C(i, j)$ is the weight value of encountered node N_j. However, because a specific node can initially obtain a higher value, the contact probability considers inter-contact time $I(i, j)$ to compensate for how recently they met. As mentioned, during the process, each node individually builds the social graph and updates the contact probability.

In Eq. (1), the slower inter-contact time and higher contact count of N_i and N_j have the highest contact probability that two nodes can have. Even though Eq. (1) compensates for how recently they met, if a long time has passed since N_i and N_j encountered each other, their contact probability must be modified with aging factor γ in terms of time t as

Table 2 Notation table

Notation	Value
N_i	Node ID i
H_i	Community ID of N_i
IV_i	Information vector of N_i including F_i, P_i, B_i, I_i, C_i
F_i	Forwarding value list of N_i with encountered nodes N_j
P_i	Contact probability list of N_i with encountered nodes N_j
B_i	Buffer weight value list of N_i with encountered nodes N_j
I_i	Inter-contact time list of N_i with encountered nodes N_j
C_i	Contact count list of N_i with encountered nodes N_j
R_i	Subject of N_i with first preference
A_i	Table of message preference matrix
M_i	Message generated by N_i including S_i, $PrevID_i$, $PrevH_i$, $Data_i$
$PrevID_i$	Previous relay node ID list in M_i
$PrevH_i$	Previous relay node community ID list in M_i
$Data_i$	Data in M_i

$$P(i,j)_{new} = P(i,j)_{old} \times \gamma^t, \tag{2}$$

where t is the time that has elapsed since their last encounter. In addition, the contact probability must incorporate the transitivity property; that is, when N_i meets N_j after N_j encountered N_k, N_i updates $P(i, k)$ according to $P(i, j)$ and $P(j, k)$ as

$$P(i,k) = (1 - P(i,j)) \times (1 - P(j,k)), \tag{3}$$

Note that N_i does not update $P(i, k)$ if N_i had previously encountered N_k because $P(i, k)$ was already updated at that time. We can now use contact probability $P(i, j)$ for forwarding messages.

5.3 Buffer weight value

Because SPF replaces messages in selfish environments to save constraint resources, it considers the buffer weight value to efficiently manage the buffer space for forwarding. The buffer weight value indicates the ratio of how many empty spaces each buffer has. As a node's buffer weight value increases, the node more efficiently receives and keeps the "partially like" message. Whenever N_i meets encountered node N_j to forward, the buffer weight value $B(i)$ of N_i is calculated as

$$B(i) = \frac{E_i}{T_i}, \tag{4}$$

where T_i and E_i are the total number of buffer space and the total number of empty buffer space, respectively. Equation (4) captures the ratio of empty space in the total number of buffer spaces. It is used for integration with contact probability to calculate the forwarding value.

5.4 Integration of contact probability and buffer weight value

We now integrate contact probability $P(i, j)$ and buffer weight value $B(i)$ (if N_i considers N_j, it is $B(j)$) to determine forwarding value $F(i, j)$, which is calculated as

$$F(i,j) = P(i,j) \times B(i), \tag{5}$$

$$B(i) = \begin{cases} 1, & \text{if the message is like or no information about } N_i \\ \frac{E_i}{T_i}, & \text{if the message is patially like} \end{cases}$$

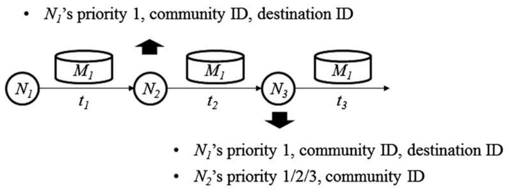
In Eq. (5), the contact probability is multiplied by the buffer weight value because the contact probability and buffer weight value vary from 0 to 1.0. Additionally, N_i and N_j calculate both $F(i, j)$ and F(j, i). Consequently, the forwarding value considers the contact probability and buffer spaces for forwarding according to the message preference. When the message is "like" for encountered node N_j, the forwarding value $F(j, i)$ equals the contact probability $P(j, i)$. Otherwise, if the message is the "partially like" for N_j, N_i calculates $F(j, i)$ by considering N_j's buffer space. Suppose that the destination of N_i is assumed to be N_d. Whenever N_i meets N_j, N_i calculates $F(i, d)$ and sends the message to N_j when $F(j, d)$ is greater than $F(i, d)$. To calculate N_i has to know the "like" and "partially like" messages for N_j. Hence, we describe it as selfishness detection in Sect. 5.5. Consequently, all nodes cooperatively forward the messages, with consideration of the message preference and buffer space by the forwarding value.

5.5 Selfishness detection

In order to forward messages to the encountered relay nodes, SPF must detect the nodes' selfishness to calculate the relay nodes' preference for the forwarding value. In MSNs, because there may be no stable path between the source and destination nodes, it is difficult for nodes to obtain acknowledgement packets, including those for whether a certain node behaves selfishly. Thus, SPF takes advantage of snooping for selfishness detection. SPF needs both message preference and nodes communities. Since each node can know the communities of nodes, we focus on detecting who generates and how much the nodes like the messages.

Note that message M_i, which N_i generates, consists of the previous relay node ID list, $PrevID_i$, the previous relay node community ID list, $PrevH_i$, and data, $Data_i$. During the forwarding process, after node N_i exchanges its information vectors with encountered node N_j when they meet, they can know each other's information, such as their respective community IDs and subjects with first preference. After N_j receives message M_i, it can also know which node generated the message, $PrevID_i$, and $PrevH_i$ by snooping information from M_i. In addition, N_j can estimate the previous nodes' message preference because the previous nodes do not forward and receive the message unless they have the "like" or "partially like" message. Each node repeats this process to gather the others' information; it can then efficiently forward the other message by considering the encountered node's information. Each node keeps the table of message preference A, including the other nodes' priorities. Whenever each node meets, it updates the table of message preference A by estimating, snooping, and gathering the other nodes' information. Consequently, each node can know the other nodes' subjects, community IDs, and destination IDs for forwarding via snooping. Figure 3 shows an example of selfishness detection.

Let us assume that N_i is the source node, and N_j and N_k are the relay nodes. N_i must forward message M_i (i.e., N_i's subject with its first preference) to the destination. At time t_1, N_i and N_j exchange their information vectors, including their node IDs, community IDs, and so on for discovery, and


• N_1's priority 1, community ID, destination ID
N_1
M_1
t_1
N_2
M_1
t_2
N_3
M_1
t_3
• N_1's priority 1, community ID, destination ID
• N_2's priority 1/2/3, community ID

A_1

Node ID / Priority	Priority 1	Priority 2	Priority 3
N_1	Computer	Biology	Physics

A_2

Node ID / Priority	Priority 1	Priority 2	Priority 3
N_1	Computer		
N_2	Biology	Computer	Chemistry

A_3

Node ID / Priority	Priority 1	Priority 2	Priority 3
N_1	Computer	.	.
N_2		Computer	
N_3	Chemistry	Physics	Biology

Fig. 3 Example of selfishness detection

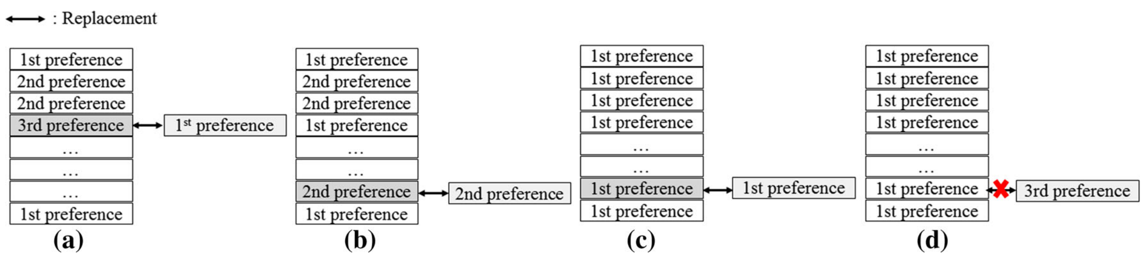


Fig. 4 Example of preference-based buffer policy

N_i forwards M_i, including the source node ID, source community ID, destination node ID, and $data_i$ to N_j. N_j receives M_i, and reviews and learns N_i's first preference and community ID. In the same way as for N_j, when N_k receives M_i at time t_2, N_k also reviews and learns N_i's first preference and community ID. N_k additionally knows at least that N_j's first, second or third preferences, while also knowing its community ID, because each node receives and forwards the "like" or "partially like" messages. The next relay nodes can continuously snoop and know information about the previous relay node and source node until delivering the message to the destination. If the information has been gathered, each node can forward messages by considering the buffer weight value to efficiently manage buffer spaces. Because each node forwards one message, each node can build the table of message preferences with other nodes. Consequently, each node initially lacks information about the others' priorities; nevertheless, it can learn the encountered nodes' priorities as times passes because many generated messages go through relay nodes by snooping.

5.6 Buffer replacement policy

The buffer replacement policy is very important for efficiently managing buffer space. Nodes basically behaves based on the above social preference-based selfishness in which each node continuously replaces any messages for the performance. Thus, SPF exploits a proper replacement policy, the preference-based buffer policy for the environment.

The proposed preference-based buffer policy works in a way similar to the queue with the message preference. However, the preference-based buffer policy replaces the oldest message among ones with the lowest preference first rather than using FIFO. For this reason, all nodes can efficiently protect messages with higher preferences, including ones with their first preferences. SPF performs the preference-based buffer policy when each node's buffer is full. Figure 4 depicts how the preference-based buffer policy works.

The preference-based buffer policy is classified into four cases. When a node's buffer is full, buffer replacement occurs. As shown in Fig. 4(a), when the node receives the message with its first preference, it replaces the message with the lowest preference (third) message with a new one. As depicted in Fig. 4(b), when the node receives the message with its second preference, and it has only messages with its first or second preference, the node replaces the old message with the new one. In Fig. 4(c), when it has only messages with its first preference and receives also a new one with its first preference, the node replaces the oldest one with the new one, except for the generated one in the bottom of its buffer Finally, as shown in Fig. 4(d), when the node has only messages with its first preference, as described in Fig. 4(c), the node does not replace them because the messages with its first preference that the node keeps is higher than the new message.

5.7 Information vector exchange and forwarding protocol

We now detail the information vector exchange and forwarding protocol. Each node N_i maintains the message information vector $IV_i = < ID_i, H_i, F_i, P_i, B_i, I_i, C_i, R_i, A_i >$ to update the information vector and forward the message, where ID_i is the ID of N_i, H_i is the community number of N_i, $F_i = \{F(i, 1), F(i, 2), \ldots, F(i, m), F(2, i), F(3, i), \ldots F(m, i)\}$, $P_i = \{P(i, 1), P(i, 2), \ldots, P(i, m), P(2, i), P(3, i), \ldots P(m, i)\}$, $B_i = \{B(i), B(2), \ldots, B(m)\}$, $I_i = \{I(i, 1), I(i, 2), I(i, 3), \ldots I(i, m)\}$, and $C_i = \{C(i, 1), C(i, 2), C(i, 3), \ldots C(i, m)\}$. In addition, R_i indicates the selected subject for the destination of N_i with its first preference, and A_i represents the table of message preferences of N_i. $F(i, j)$ is commuted with $P(i, j)$ and $B(i)$. N_i also simultaneously calculates $F(j, i)$ to reduce extra communication between them. Moreover, N_i generates message $M_i = < PrevID_i, PrevH_i, Data_i >$, where $PrevID_i$, $PrevH_i$, and $Data_i$ are the previous relay nodes' ID lists, including the source node ID and previous relay nodes' community ID list. These include the source node community ID and data of N_i to distinguish it from other messages that N_i generates, respectively.

The SPF forwarding algorithm is outlined in Algorithm 1. Assume that N_i has a message destined for D_i. When N_i encounters N_j, N_i updates the contact probability $P(i, j)$ with $C(i, j)$ and $I(i, j)$, with each $P(i, k)$ for $k \neq i \neq j$ in P_i using Eq. (3) with P_j. Then, N_i exchanges IV_i when N_j updates the buffer weights of value $B(i)$ and $B(j)$ in B_i, as in Eq. (4). Additionally, N_i computes $F(i, d)$ and $F(j, d)$ using Eq. (5). Finally, when H_j equals H_d, and H_i does not equal H_j, N_i forwards M_i to N_j. Otherwise, when H_i equals H_j and H_j equals H_d, or when H_i does not equal H_j and H_j does not equal to H_d, and if $F(j, d)$ is greater than $F(i, d)$, N_i forwards the message M_i to N_j. Finally, N_j updates $PrevID_i$, $PrevH_i$ in M_i.

Figure 5 presents an example of SPF forwarding. We assume that N_1 attempts to forward M_i for destination N_d to other nodes N_2, N_3, N_4, N_5, and N_6 when they meet within the communication range after exchanging their information vectors. Moreover, N_1 has information about other nodes' message preferences. N_3 likes M_i, whereas N_1, N_2, N_4, and N_5 partially like it, and N_6 dislikes it. Note that Fig. 5 indicates "like," "partially like," and "dislike" as L, P, and D, respectively. N_1 forwards M_i to N_2 because both nodes' community IDs are different from that of N_d and the forwarding value $F(1, d)$, 0.25, of N_1 is higher than the one, $F(2, d)$, 0.49, of N_2. N_1 also forwards to N_3 because $B(3)$, 0.3 is lower than $B(1)$, 0.5. However, N_3 likes M_i; therefore, $F(3, d)$ equals $P(3, d)$ (i.e., $B(3)$ is 1). Finally, $F(3, d)$, 0.5 is higher than $F(1, d)$, 0.25. N_1 does not forward M_i to N_4 because $P(4, d)$, 0.9 is lower than $P(1, d)$, 0.5; however, $F(4, d)$, 0.09 is lower than $F(1, d)$, 0.25 because of $B(4)$, 0.1. Even though $F(5, d)$, 0.16 is lower than $F(1, d)$, 0.25, N_1 forwards M_i to N_5 because N_5's community ID is the same as that of N_d. On the other hand, $F(6, d)$, 0.81 is higher than $F(1, d)$, 0.25, and N_6's community ID is the same as that of N_d. However, N_1 does not forward to N_6 because N_6 dislikes M_i.

Algorithm 1. Pseudo code for SPF forwarding

```
01: // After discovery of encountered node $N_j$
02: SPF_Forwarding {
03:    Update_ContactProbability($P_i$);                  // Update the contact probabilities
04:    Exchange_information_vector($N_i$, $N_j$);          // $IV_i$ with $IV_j$
05:    If message preference information of $N_j \in A_i$
06:       Update_BufferWeight($B_i$);                     // Update the buffer weight value of $N_i$
07:    Calculation_FValue($B_i$, $F(i, d)$, $F(j, d)$);    // Calculate $F(i,d)$ and $F(j, d)$
08:    If $H_j = H_d$ & $H_i$ != $H_j$
09:       Forward $M_i$ to $N_j$;
10:    Else If ($H_j = H_j$ & $H_j = H_d$) || ($H_i$ != $H_j$ & $H_j$ != $H_d$)
11:       If $F(i,d) < F(j, d)$
12:          Forward $M_i$ to $N_j$;
13: }
```

6 Performance evaluations

6.1 Simulation setup

We employed the network simulator NS-2 v2.35 [56] for our simulations. We ran each scheme 20 times to determine the average results. The movement of a node followed HCMM [55], which is a frequently employed moving pattern in MSN simulations. The network area was

	$P(x, d)$	$B(x)$	H_x
N_1	0.5	0.5	2
N_2	0.7	0.7	3
N_3	0.5	0.3	2
N_4	0.9	0.1	4
N_5	0.2	0.8	1
N_6	0.9	0.9	1
N_d			1

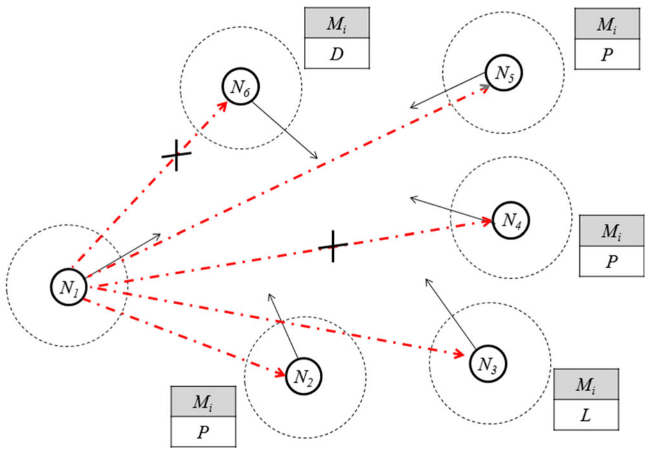


Fig. 5 Example of SPF forwarding

Table 3 Parameters for the simulation

Parameter	Value (default)
Network area	450×450 m^2
Community size	150×150 m^2
Number of grids	9
Number of communities	4
Buffer size	5, 10, 15, 20, 25, 30, 35, 40, (20)
Number of nodes	40, 50, 60, 70, (40)
Communication range	10, 20, 30, 40, 50, (10) m
TTL (limited hop count)	4
Velocity of nodes	1–9 m/sec
Simulation time	9000 s

set to 450×450 m; the community size was 150×150 m. The number of grids was nine. The number of communities was four among the grids, and each community had ten or more nodes. Each node had 5 $\sim$ 40 buffer spaces with five spaces. The number of nodes was set to 40, 50, 60, and 70. The communication range varied from 10 to 50 m. The TTL was set to four using the hop count. The velocity of a node ranged from 1 to 9 m/sec, which is appropriate for either people or vehicles. After a source node transmitted its message to other nodes, it did not delete the message. The total simulation time was 9000 seconds because Epidemic, PRoPHET, and SimBet in MSNs, these are very suitable for a simulation comparison. For these schemes, each node can be aware of which messages is the "like", "partially like", or "dislike" ones when each node meets the encountered node. In our simulator, each node generated and issued messages to a random destination with exponential distribution up to a third of the total simulation time. Table 3 summarizes the parameters used in our simulation. All the simulation environments followed as in [26].

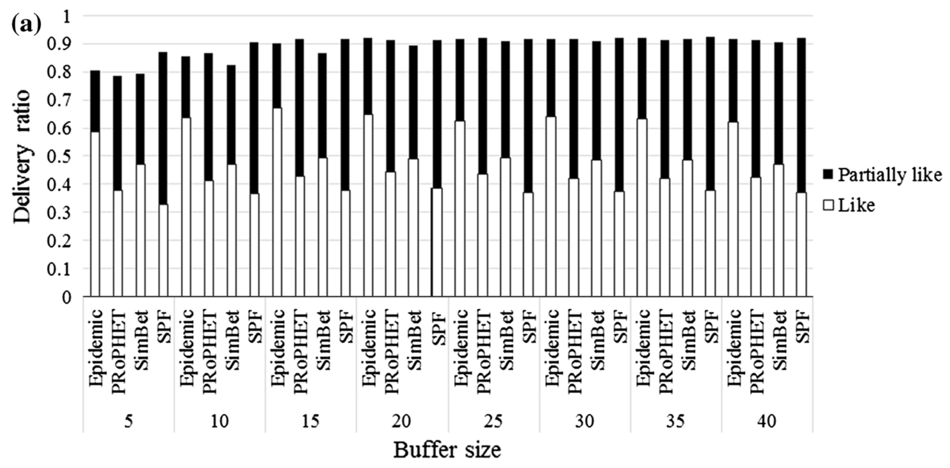
We evaluated the proposed scheme with the following performance metrics:

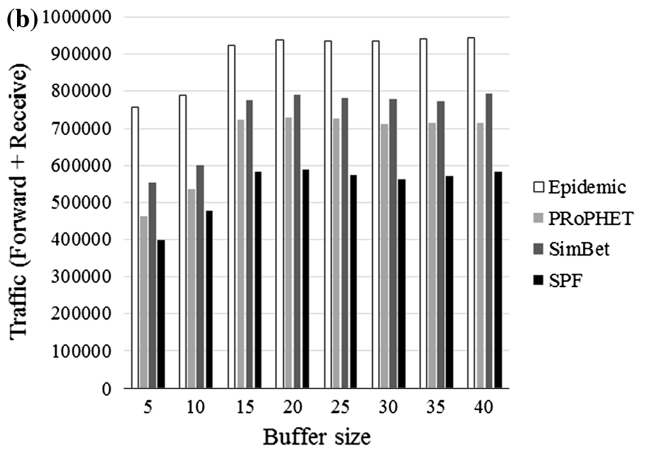
1. Delivery ratio: The ratio of the number of delivered messages to the total number of messages issued.
2. Network traffic: The total number of messages sent and received.
3. Transmission delay: The time needed for a message to travel from the source to destination node.
4. Hop count: The average number of hops required for a message to travel from the source to destination.
5. Replacement frequency: The number of replaced messages according to each of buffer replacement policy when the buffer is full.

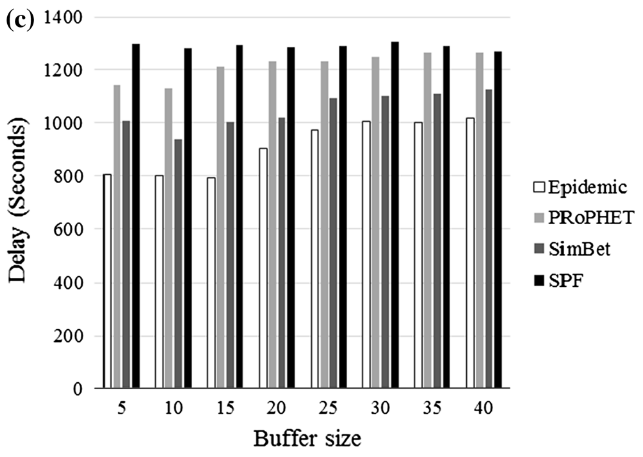
6.2 Simulation results

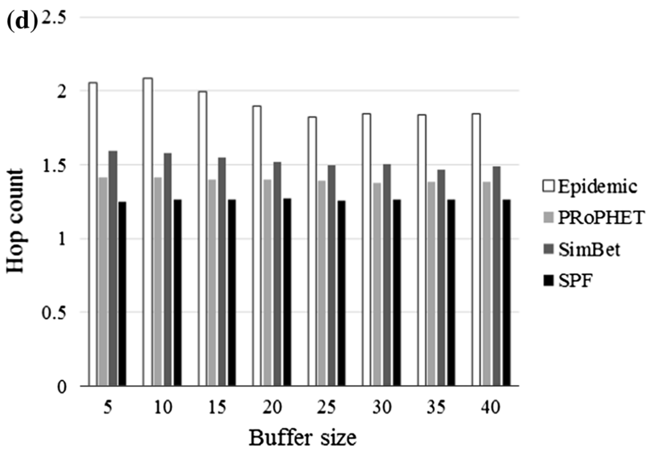
6.2.1 Effect of the buffer size

We examined the performance of the schemes as the buffer size number increased. The number of nodes and the communication range were set to 40 and 10 m, respectively. The buffer size varied from 5 to 40 with 5 spaces. As shown in Fig. 6(a), as the buffer size increased, the delivery ratio of these schemes reached close to 1.0. However, it was difficult to achieve a 1.0 delivery ratio within the 9000-sec simulation time because nodes selfishly behave. In particular, SPF initially had a higher delivery ratio than other schemes because SPF kept both the "like" and "partially like" messages based on the forwarding value using the contact probability and buffer weight value. Additionally, as shown in Fig. 6(b), as the buffer size increased, the network traffic of SPF was lower than other schemes because SPF properly selected the relay nodes to forward messages by the forwarding value. From the 15 buffer spaces, all sets of network traffic remained stable because the delivery ratio was continuously steady.









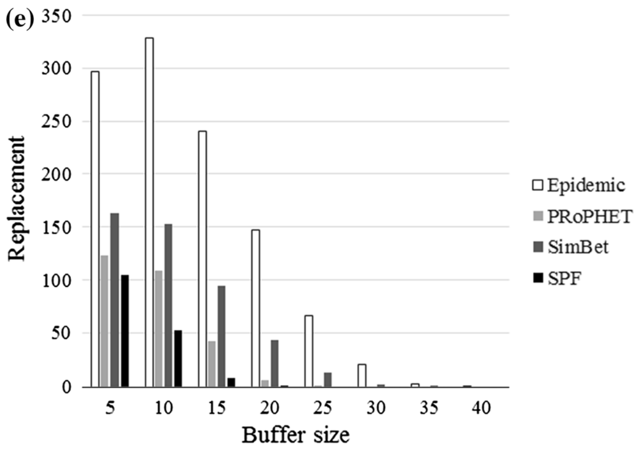


Fig. 6 Effect of buffer size (a) Delivery ratio (b) Network traffic (c) Transmission delay (d) Hop count (e) Replacement frequency

On the other hand, SPF showed a longer transmission delay than other schemes because SPF did not forward messages to any nodes for a while; it required more time to find the proper nodes, as shown in Fig. 6(c). As illustrated in Fig. 6(d), because SPF identified proper relay nodes to forward the messages, they could be delivered in fewer hops. However, other schemes could quickly deliver messages to the destination; however, the hop count was high

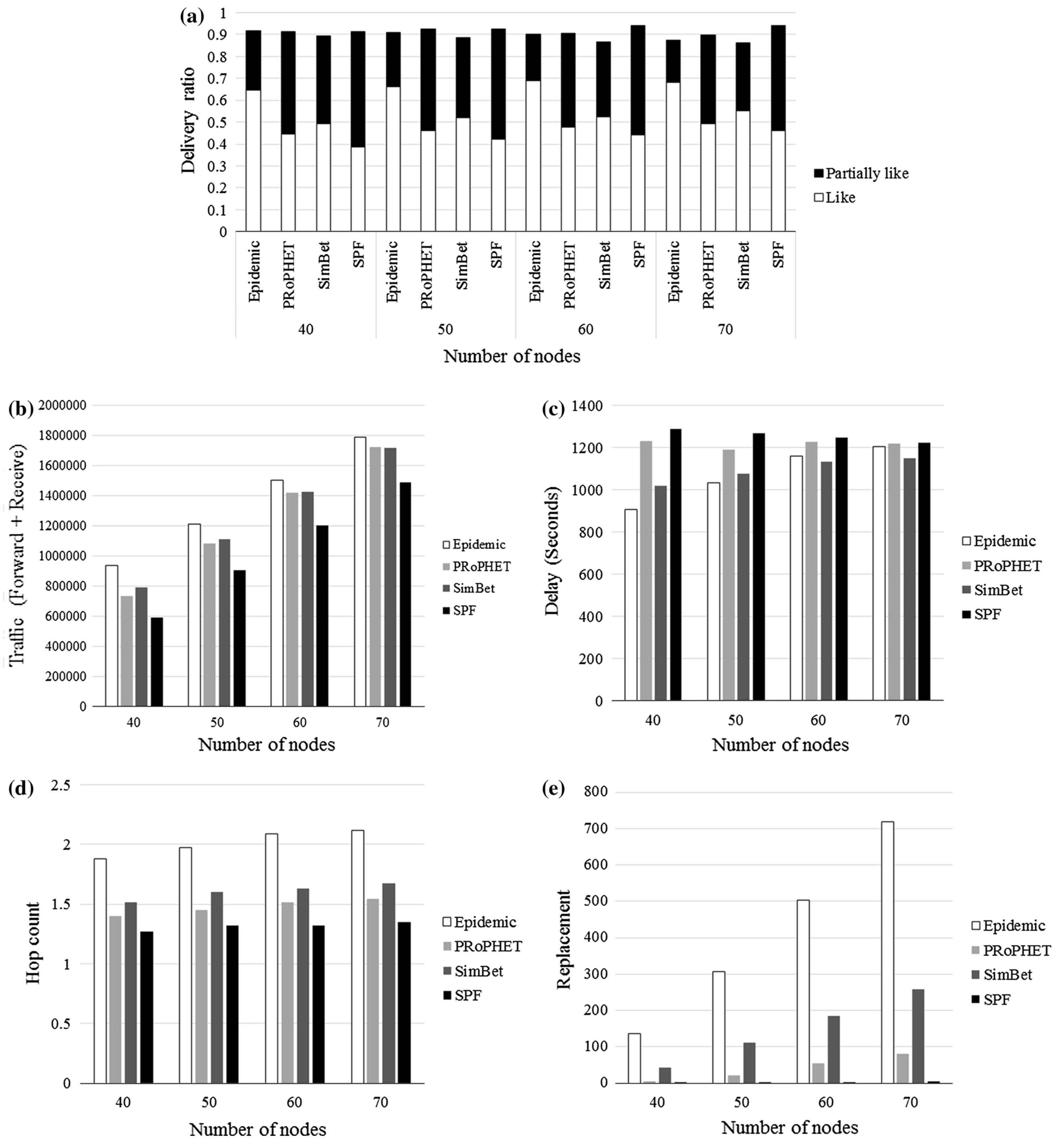


Fig. 7 Effect of the number of nodes (a) Delivery ratio (b) Network traffic (c) Transmission delay (d) Hop count (e) Replacement frequency

because other schemes did not other nodes' message preferences and then had a high hop to find the destination. Finally, SPF had a lower message loss than other schemes, as shown in Fig. 6(e). When the buffer size increased, the replacement frequency of schemes decreased. When the buffer size was ten, the replacement frequency of Epidemic was the highest because other schemes unconditionally forwarded messages; moreover, the buffers of each of the 40 nodes were unable to retain generated messages in their limited buffer sizes. The replacement frequency of other schemes except Epidemic were higher since they did not consider other nodes' buffer space. The replacement frequency of SPF decreased and reached 0 from the 25 buffer spaces because the nodes efficiently selected relay nodes to

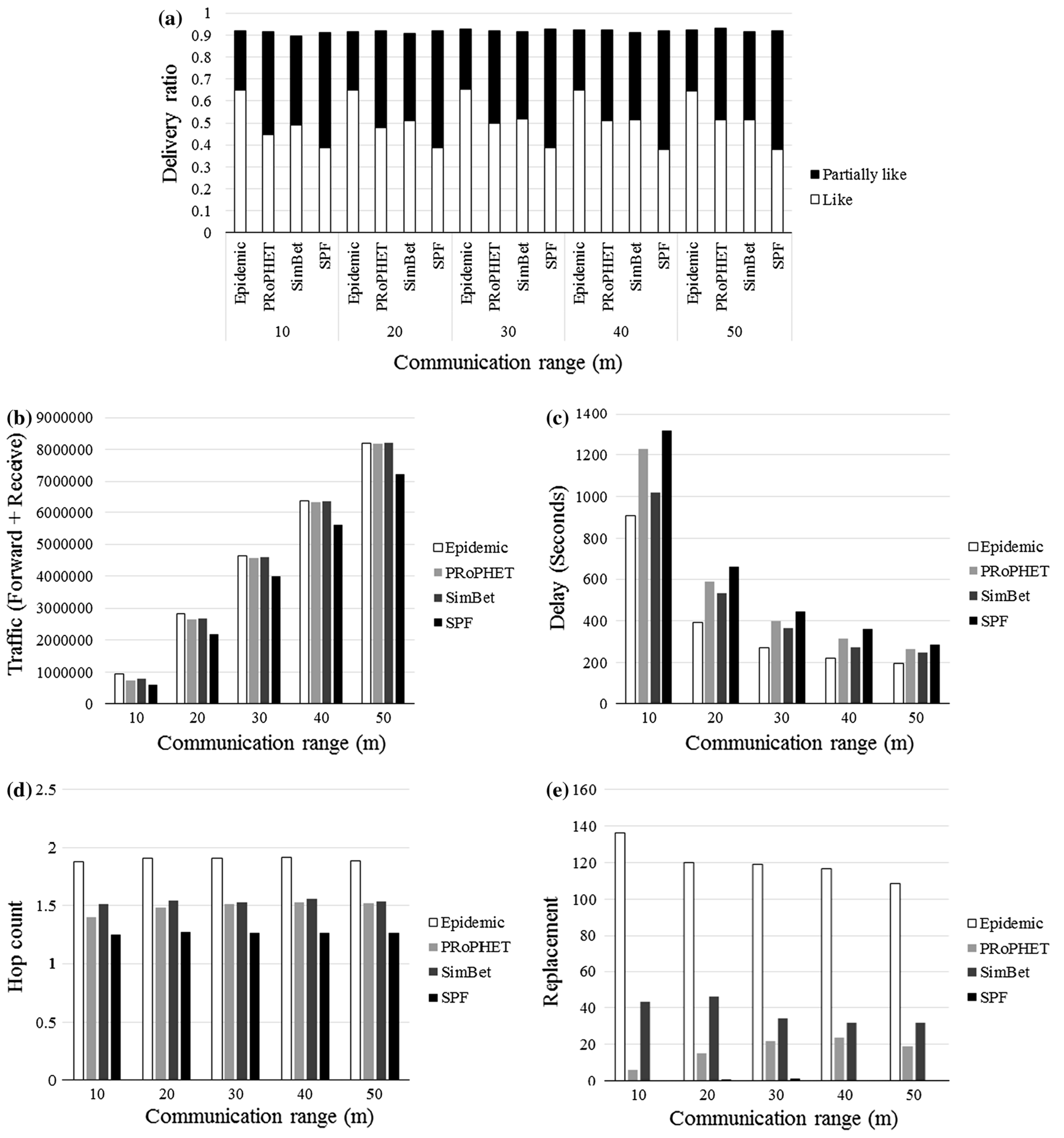


Fig. 8 Effect of the communication range (a) Delivery ratio (b) Network traffic (c) Transmission delay (d) Hop count (e) Replacement frequency

forward the messages by the forwarding values and preference-based buffer policy. Consequently, SPF could be implemented in the environment with a lower buffer size.

6.2.2 Effect of the number of nodes

We evaluated the performance of the schemes as the number of nodes increased. The buffer size and communication range were set to 20 and 10 m, respectively. Because the number of nodes was 8 in each subject for 40 nodes, it was 10, 12, and14 for 50, 60, and 70 nodes. In Fig. 7(a), the delivery ratio of the schemes was maintained at more than 0.9; however, the delivery ratio of other schemes were lower than SPF because they less consider the “partially like” messages. Moreover, each node continuously replaced many generated messages in the buffer space before delivering them to the destination. On the other hand, SPF maintained a higher delivery ratio using both the "like" and "partially like" messages. As shown in Fig. 7(b), as expected, the schemes traffic amounts increased as the number of nodes increased. Other schemes had higher network traffic. Because they had difficulty finding the destination and was a waste of network traffic. On the other hand, SPF maintained minimal amounts of traffic because it retained both the "like" and "partially like" messages based on the forwarding value according to the message preference even though the number of nodes increased. Figure 7(c) shows the transmission delay. The delays of the schemes remained steady as the number of nodes increased. However, the delays of other schemes were lower than SPF because many replacements occurred and it required more time to find the destination. In SPF, because each node cooperatively forwarded the messages until delivering them to the destination node, the delay decreased slightly. In Fig. 7(d), the hop counts of other schemes increased up to 50 nodes because there were numerous nodes to which to forward messages. On the other hand, SPF maintained a stable hop count because it maintained a more than 0.9 delivery ratio. As depicted in Fig. 7(e), other schemes replaced many messages as the number of nodes increased because other schemes retained only the "like" messages; lots of replacements occurred relatively, which degraded the performance. On the other hand, SPF replaced a small number of messages and maintained a higher performance because it forwarded the messages to nodes that were likely to forward them to the destination based on the forwarding values.

6.2.3 Effect of the communication range

Finally, we compared the effect of the communication range of the nodes for each scheme. The number of nodes and the buffer size were set to 40 and 20, respectively. Basically, all schemes maintained stable performance with 40 nodes. As shown in Fig. 8(a), as the communication range increased, the delivery ratio of all schemes was maintained at 0.9. SPF retained more "partially like" messages than "like" ones even though the communication range increased. As depicted in Fig. 8(b), as the communication range increased, the scheme network traffic likewise increased. However, SPF had lower traffic than other schemes because of the former's proper selection of relay nodes to which to forward messages based on the forwarding values. Such a result confirms that SPF can be well implemented in a sparse network. As shown in Fig. 8(c), most schemes incurred shorter delays as the communication range increased when each node forwarded messages to more nodes with a wider communication range. Other schemes showed the smallest delays because they spread messages to not only inappropriate nodes but also appropriate ones. However, because the number of nodes increased, each node could properly select relay nodes to which to forward messages; the gap between other schemes and SPF decreased as the communication range increased. The hop counts of other schemes were higher than that of SPF because other schemes did not consider other nodes' message preferences and then had a high hop to find the destination. as the communication range increased, as shown in Fig. 8(d). Interestingly, because the number of nodes did not increase, the hop counts of all schemes did not noticeably increase or decrease. As shown in Fig. 8(e), the number of nodes replaced many messages in other schemes because there were many nodes to which to forward messages as the communication range increased. On the other hand, SPF had a low message loss because of its cooperative forwarding of messages to proper relay nodes.

7 Conclusion

In MSNs, most schemes assume that nodes simply forward messages. Some of these schemes have been introduced for selfishness; however, minimal research exists about selfishness for contact nodes, and no studies have considered selfishness for message preference for MSNs. We therefore proposed social preference-based selfishness in which nodes decide to drop or keep (forward) and replace messages to save their buffer space according to the message preference and the nodes' communities. We additionally proposed a novel cooperative forwarding scheme for social preference-based selfishness in MSNs, SPF. SPF incorporates the proposed message forwarding scheme that considers contact probability and buffer replacement policy for the message preference. As a result, SPF outperformed Epidemic, PRoPHET, and SimBet in terms of delivery ratio, network traffic, hop count, and replacement frequency because SPF properly selected relay nodes by considering how frequently the nodes met and how much buffer space the nodes had, while maintaining reasonable transmission delay. In future work, we plan to study trust management in this MSN environments.

Acknowledgments This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2013R1A1A2011114).

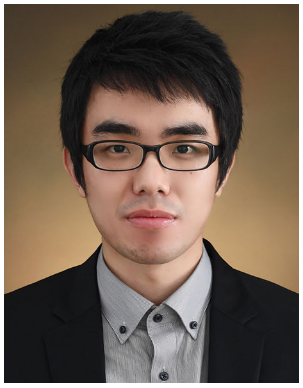
Conflict of interest The authors declare that they have no conflict of interest.

Ethical standards This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Yan, Z., Peng, Z., & Vasilakos, A. V. (2014). A survey on trust management for internet of things. *Journal of Network and Computer Applications, 42*, 120–134.
2. Sheng, Z., Yang, S., Yu, Y., Vasilakos, A. V., Mccann, J., & Leung, K. (2013). A survey on the ietf protocol suite for the internet of things: Standards, challenges, and opportunities. *Wireless Communications, 20*(6), 91–98.
3. Li, M., Li, Z., & Vasilakos, A. V. (2013). A survey on topology control in wireless sensor networks: Taxonomy, comparative study, and open issues. *Proceedings of the IEEE, 101*(12), 2538–2557.
4. Yao,Y., Cao, Q., & Vasilakos, AV. (2013). EDAL: An energy-efficient, delay-aware, and lifetime-balancing data collection protocol for wireless sensor networks. In *Proceedings of Mobile ad-hoc and sensor systems,* 182–190.
5. Xiang, L., Luo, J., & Vasilakos, AV. (2011). Compressed data aggregation for energy efficient wireless sensor networks. In *Proceedings of Sensor, mesh and ad hoc communications and networks (SECON)*, 46–54.
6. Song, Y., Liu, L., Ma, H., & Vasilakos, A. V. (2014). A biology-based algorithm to minimal exposure problem of wireless sensor networks. *IEEE Transactions on Network and Service Management, 11*(3), 417–430.
7. Cheng, H., Xiong, N., Vasilakos, A. V., Yang, L. T., Chen, G., & Zhuang, X. (2012). Nodes organization for channel assignment with topology preservation in multi-radio wireless mesh networks. *Ad Hoc Networks, 10*(5), 760–773.
8. Duarte, P. B., Fadlullah, Z. M., Vasilakos, A. V., & Kato, N. (2012). On the partially overlapped channel assignment on wireless mesh network backbone: A game theoretic approach. *IEEE Journal on Selected Areas in Communications, 30*(1), 119–127.
9. Yang, M., Li, Y., Jin, D., Zeng, L., Wu, X., & Vasilakos, A. V. (2015). Software-defined and virtualized future mobile and wireless networks: A survey. *Mobile Networks and Applications, 20*(1), 4–18. **Software-defined networks**.
10. Youssef, M., Ibrahim, M., Abdelatif, M., Chen, L., & Vasilakos, A. V. (2014). Routing metrics of cognitive radio networks: A survey. *Communications Surveys & Tutorials, 16*(1), 92–109.
11. Attar, A. H., Tang, H., Vasilakos, A. V., Yu, F. R., & Leung, V. C. (2012). A survey of security challenges in cognitive radio networks: Solutions and future research directions. *Proceedings of the IEEE, 100*(12), 3172–3186.
12. Zeng, Y., Xiang, K., Li, D., & Vasilakos, A. V. (2013). Directional routing and scheduling for green vehicular delay tolerant networks. *Wireless Networks, 19*(2), 161–173.
13. Wang, X., Vasilakos, A. V., Chen, M., Liu, Y., & Kwon, T. T. (2012). A survey of green mobile networks: Opportunities and challenges. *Mobile Networks and Applications, 17*(1), 4–20.
14. Demestichas, P. P., Stavroulaki, V. A. G., Papadopoulou, L. M., Vasilakos, A. V., & Theologou, M. E. (2004). Service configuration and traffic distribution in composite radio environments. *IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews, 34*(1), 69–81.
15. Li, Y., Qian, M., Jin, D., Hui, P., & Vasilakos, A. V. (2015). Revealing the efficiency of information diffusion in online social networks of microblog. *Information Sciences, 293*, 383–389.
16. Zhang, D., Zhang, D., Xiong, H., Hsu, C. H., & Vasilakos, A. V. (2014). BASA: Building mobile Ad-hoc social networks on top of android. *IEEE Network, 28*(1), 4–9.
17. Conti, M., Giordano, S., May, M., & Passarella, A. (2010). From opportunistic networks to opportunistic computing networks. *IEEE Communication Magazine, 48*(9), 126–139.
18. Hui, P., Chaintreau, A., Scott, J., Gass, R., Crowcroft, J & Diot, C. (2005). Pocket switched networks and human mobility in conference environments. In *Proceedings of ACM SIGCOMM Workshop on Delay-Tolerant Networking*, 244–251.
19. Zafar, H., Alhamahmy, N., Harle, D., & Andonovic, I. (2011). Survey of reactive and hybrid routing protocols for mobile ad hoc networks. *Communication Networks and Information Security, 3*(3), 193–202.
20. Vasilakos, A. V., Zhang, Y., & Spyropoulos, T. (2011). *Delay tolerant networks: Protocols and applications*. Boca Raton: CRC Press.
21. Jin, L., Chen, Y., Wang, T., Hui, P., & Vasilakos, A. V. (2013). Understanding user behavior in online social networks: A survey. *Communications Magazine, 51*(9), 144–150.
22. Wang, Y., Vasilakos, A. V., Ma, J., & Xiong, N. (2015). On studying the impact of uncertainty on behavior diffusion in social networks. *IEEE Transactions on Systems, Man, and Cybernetics: Systems, 45*(2), 185–197.
23. Conti, M., & Kumar, M. (2010). Opportunities in opportunistic computing. *IEEE Computer, 43*(1), 42–50.
24. Choi, J., Shim, K., Lee, S., & Wu, K. (2012). Handling selfishness in replica allocation over a mobile ad hoc network. *IEEE Transaction on Mobile Computing, 11*(2), 278–291.
25. Freeman, L. C. (1997). A set of measures of centrality based on betweenness. *Sociometry, 40*(1), 5–41.
26. Vahdat, A., & Becker D. (2000). Epidemic routing for partially-connected ad hoc networks. Technical Report CS-2000-06, Duke University.
27. Spyropoulos, T., Psounis, K., & Raghavendra C. (2005). Spray and wait: An efficient routing scheme for intermittently connected mobile networks. In *Proceedings of ACM SIGCOMM*, 252–259.
28. Wu, J., Xiao, M., & Huang L. (2013). Homing spread: Community home-based multi-copy routing in mobile social networks. In *Proceedings of IEEE INFOCOM*, 138–146.
29. Dvir, A., & Vasilakous, AV. (2011). Backpressure-based routing protocol for DTNs. In *Proceedings of ACM SIGCOMM*, 405–406.
30. Alresaini, M., Sathiamoorthy, M., Krishnamachari, B., & Neely MJ. (2010). Backpressure with adaptive redundancy. In *Proceedings of IEEE INFOCOM*, 2300–2308.
31. Li, P., Guo, S., Yu, S., & Vasilakos AV. (2012). CodePipe: An opportunistic feeding and routing protocol for reliable multicast with pipelined network coding. In *Proceedings of IEEE INFOCOM*, 100–108.
32. Meng, T., Wu, F., Yang, Z., Chen, G., & Vasilakos, A. V. (2015). Spatial reusability-aware routing in multi-hop wireless networks. *IEEE Transactions on Computer, 99*, 1–13.
33. Kim, S., Choi, J., & Yang, S. Hotspot: Location-based forwarding scheme in an opportunistic network. Ad hoc & Sensor Wireless Network, In press.
34. Daly, E., & Haahr, M. (2007). Social network analysis for routing in disconnected delay-tolerant MANETs. In *Proceedings of ACM MobiHoc*, 32–40.
35. Hui, P., Crowcroft, J., & Yoneki, E. (2008). Bubble rap: Social-based forwarding in delay tolerant networks. *IEEE Transaction on Mobile Computing, 10*(11), 1576–1589.
36. Mei, A., Morabitto, G., Santi, P., & Stefa, J. (2011). Social-aware stateless forwarding in pocket switched networks, In *Proceedings of IEEE INFOCOM*, 251–255.
37. P. Hui and J. Crowcroft (2007). How small labels create big improvements. In *Proceedings of Pervasive Computing and Communications Workshops*, 244–251.
38. Boldrini, C., Conti, M., & Passarella, A. (2008). Exploiting users' social relations to forward data in opportunistic networks: The HiBop solution. *Pervasive and Mobile Computing, 4*(5), 633–657.

39. Lindgren, A., Doria, A., & Schelen, O. (2004). Probabilistic routing in intermittently connected networks. *Service Assurance with Partial and Intermittent Resources (LNCS), 3126*, 239–254.
40. Mtibaa, A., May, M., Diot, C., & Ammar M. (2010). PeopleRank: Social opportunistic forwarding. In *Proceedings of IEEE INFOCOM*, 1–5.
41. Leguay, J., Friedman, T., & Conan, V. (2007). Evaluating mobyspace based routing strategies in delay tolerant networks. *Wireless Communications and Mobile Computing, 7*(10), 1171–1182.
42. Erramilli, V., Crovella, M., Chaintreau, A., & Diot, C. (2008). Delegation forwarding. In *Proceedings of ACM MobiHoc*, 251–259.
43. Yen, Y. S., Chao, H. C., Chang, R. S., & Vasilakos, A. V. (2011). Flooding-limited and multi-constrained QoS multicast routing based on the genetic algorithm for MANETs. *Mathematical and Computer Modelling, 53*(11), 2238–2250.
44. Busch, C., Kannan, R., & Vasilakos, AV. (2012). Approximating Congestion+Dilation in Networks via Quality of Routing&#x201D. IEEE Transactions on Games. Computers, 61(9), 1270–1283.
45. Cianfrani, A., Eramo, V., Listanti, M., Polverini, M., & Vasilakos, A. V. (2012). An OSPF-integrated routing strategy for QoS-aware energy saving in IP backbone networks. *IEEE Transactions on Network and Service Management, 9*(3), 254–267.
46. Vasilakos, AV., Ricudis, C., Anagnostakis, K., Pedryca, W., & Pitsillides, A. (1998). Evolutionary-fuzzy prediction for strategic QoS routing in broadband networks. In *Proceedings of Fuzzy Systems*, 2, 1488–1493.
47. Zhang, X., Zhang, Y., Yan, F., & Vasilakos, A. V. (2014). Interference-based topology control algorithm for delay-constrained mobile ad hoc networks. *IEEE Transactions on Mobile Computing, 14*(4), 742–754.
48. Zhou, L., Chao, H. C., & Vasilakos, A. V. (2011). Joint forensics-scheduling strategy for delay-sensitive multimedia applications over heterogeneous networks. *IEEE Journal on Selected Areas in Communications, 29*(7), 1358–1367.
49. Li, Q., Gao, W., Zhu, S., & Cao, G. (2012). A routing protocol for socially selfish delay tolerant networks. *Ad Hoc Networks, 10*(8), 1619–1632.
50. Mei, A., & Stefa, J. (2012). Give2Get: Forwarding in social mobile wireless networks of selfish individuals. *IEEE Transaction on Dependable Secure Computing, 9*(4), 569–582.
51. Manam, V. K. C., Mahedran, V., & Murthy, C. S. R. (2014). Performance modeling of DTN routing with heterogeneous and selfish nodes. *Wireless Networks, 20*(1), 25–40.
52. Li, Y., Su, G., Wu, D. O., Jin, D., Su, L., & Zeng, L. (2011). The impact of node selfishness on multicasting in delay tolerant networks. *IEEE Transaction on Vehicular Technology, 60*(5), 2224–2237.
53. Yang, S., Adeel, U., & McCann, J. A. (2013). Selfish mules: Social profit maximization in sparse sensornets using rationally-selfish human relays. *IEEE Journal on Selected Areas in Communication, 31*(6), 1124–1134.
54. Hui, P., Xu, K., Li, V., Crowcroft, J., Latora, V & Lio, P. (2009). Selfishness, altruism and message spreading in mobile social networks. In *Proceedings of IEEE INFOCOM Workshop*, 1–6.
55. Boldrini, C., & Passarella, A. (2010). HCMM: Modelling spatial and temporal properties of human mobility driven by users' social relationships. *Computer Communications, 33*(9), 1056–1074.
56. Network simulator-2, http://www.isi.edu/nsnam/ns/



Sun-Kyum Kim received his M.S. in computer science from Yonsei University in Korea in 2012. He is currently a Ph.D. candidate at Yonsei University. His research interests include mobile social networks, delay tolerant networks and social network analysis.

Ji-Hyeun Yoon is currently an Ph.D. candidate in computer science at Yonsei University in Korea. His research interests include mobile social networks, delay tolerant networks and social network analysis.

JunYeop Lee is currently an Ph.D. candidate in computer science at Yonsei University in Korea. His research interests include mobile social networks, delay tolerant networks and social network analysis.

Gi-Young Jang is currently an M.S. candidate in computer science at Yonsei University in Korea. His research interests include mobile social networks, delay tolerant networks and social network analysis.

Sung-Bong Yang received his M.S. and Ph.D. from the Dept. of Computer Science at the University of Oklahoma in 1986 and 1992, respectively. He has been a professor at Yonsei University since 1994. His research interests include graph algorithms, mobile computing, and social network analysis.