PAPER *Special Section on Parallel and Distributed Computing and Networking*

Social Network Based P2P Multicast Reducing Psychological Forwarding Cost in Mobile Networks

Hiroyuki KUBO[†a)], *Member*, **Ryoichi SHINKUMA**[†b)], *Senior Member*, *and* **Tatsuro TAKAHASHI**[†c)], *Fellow*

SUMMARY The demand for data/audio streaming/video streaming multicast services in large scale networks has been increasing. Moreover, the improved transmission speed and mobile-device capability in wireless access networks enable people to use such services via their personal mobile devices. Peer-to-peer (P2P) architecture ensures scalability and robustness more easily and more economically than server-client architecture; as the number of nodes in a P2P network increases, the amount of workload per node decreases and lessens the impact of node failure. However, mobile users feel much larger psychological cost due to strict limitations on bandwidth, processing power, memory capacity, and battery life, and they want to minimize their contributions to these services. Therefore, the issue of how we can reduce this psychological cost remains. In this paper, we consider how effective a social networking service is as a platform for mobile P2P multicast. We model users' cooperative behaviors in mobile P2P multicast streaming, and propose a social-network based P2P streaming architecture for mobile networks. We also measured the psychological forwarding cost of real users in mobile P2P multicast streaming through an emulation experiment, and verify that our social-network based mobile P2P multicast streaming improves service quality by reducing the psychological forwarding cost using multi-agent simulation.
key words: *mobile P2P multicast, psychological cost, cooperation problem, experimental measurement*

1. Introduction

The demand for data/audio streaming/video streaming multicast services in large scale networks has been increasing. Moreover, the improved transmission speed and mobile-device capability in wireless access networks enable people to use such services via their personal mobile devices. Peer-to-peer (P2P) architecture ensures scalability and robustness more easily and more economically than server-client architecture; as the number of nodes in a P2P network increases, the amount of workload per node decreases and lessens the impact of node failure [1]. In the next five to ten years, mobile users will require higher bit rates as wireless access networks continue to increase transmission speed to a few Mbps, which implies that P2P architectures must be introduced to mobile networks to handle large numbers of requests from mobile nodes (Fig. 1). However, in mobile P2P multicast, there remain several issues to be solved, including how to construct and maintain P2P topologies based on node capability. Particularly, in this paper, we consider how nodes in P2P multicast services will be required to forward data or streams to other nodes. In general, mobile users feel much larger psychological cost due to strict limitations on bandwidth, processing power, memory capacity, and battery life, and they want to minimize their contributions to these services [2]. Therefore, the issue of how we can reduce this psychological cost remains.

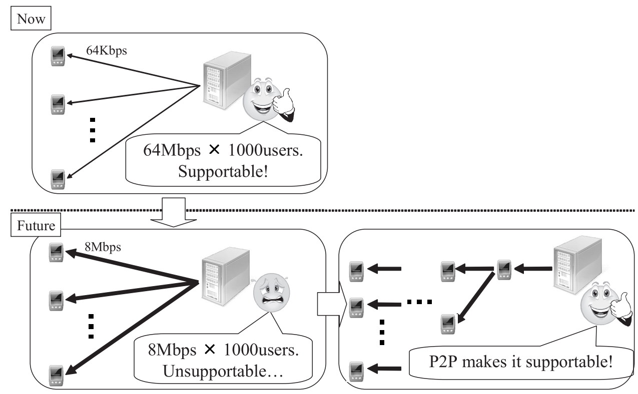


Fig. 1 Demand for mobile P2P multicast.

In our previous work, we proved that the psychological cost experienced by forwarding users is reduced when *child* users, who receive forwarded data or streams, are friends of the forwarders [3]. This observation motivated us to propose a social-network based P2P multicast architecture. More concretely, we consider here how effective a social networking service is as a platform for mobile P2P multicast. In SNS, *friends* are connected to each other via logical links. The network consisting of the logical links that represent friendships is called a social network (or social graph). In an SNS, users generate, upload, and share content with their friends, and vice versa. Moreover, in the most recent SNSs, users can view content uploaded by friends of their friends if they are permitted. This suggests that it is a natural occurrence for users to contribute their resources to other users in SNSs. Therefore, we can expect that users will feel a smaller psychological forwarding cost in mobile P2P multicast if the streams are forwarded along social links. However, if we tried to directly adopt social networks as a P2P multicast network, we would probably observe the following problems. 1) Unbalanced load problem: nodes who have many friends need to forward many streams. The number of friends in social networks follows a power law [4], [5], which implies that a limited number of

Manuscript received February 1, 2010.
Manuscript revised June 10, 2010.
[†]The authors are with the Graduate School of Informatics, Kyoto University, Kyoto-shi, 606–8501 Japan.
a) E-mail: kubo@cube.kuee.kyoto-u.ac.jp
b) E-mail: shinkuma@i.kyoto-u.ac.jp
c) E-mail: ttakahashi@i.kyoto-u.ac.jp
DOI: 10.1587/transinf.E93.D.3260

nodes have many friends, while most nodes have a limited number of friends. 2) Large hopcount problem: in mobile networks in particular, since communication quality is unstable and the buffer capacity is small, the maximum number of hopcounts needs to be limited to suppress the propagation of degraded communication quality, the range of jitter and the playback delay. 3) Isolated node problem: a node cannot receive data/streams if any of its friends are not in the P2P multicast.

The contributions of this paper are as follows: 1) We model users' cooperative behaviors in mobile P2P multicast streaming in Sect. 3. In general, battery capacity is the dominant psychological factor, which makes users reluctant to forward streams to others. 2) In Sect. 4, we propose a social-network based P2P streaming architecture for mobile networks. To solve the three problems listed above, our method defines the maximum number of forwarded streams per node to balance the load; our method chooses the highest centrality node as the first forwarder, which directly receives the stream from the source server; we examine the link between a node to friends of its friend, which can also be expected to reduce the psychological forwarding cost compared with the case of forwarding to strangers. 3) We measured the psychological forwarding cost of real users in mobile P2P multicast streaming through an emulation experiment. We explain the details in Sect. 5. 4) We verify that our social-network based mobile P2P multicast streaming improves service quality by reducing the psychological forwarding cost. We use a multi-agent simulation approach, in which actually measured psychological costs are input directly to the simulator as the simulation parameters.

2. Related Work

2.1 Social Networking Service

Social networking services (SNSs) have attracted millions of users [6]. Recently, Facebook [7] and MySpace [8], which are two of the most famous SNSs, reported that they currently have a few hundred million active users worldwide. Moreover, most SNSs support mobile use. Another SNS is specialized for mobile networks called Mobile Social Software (MoSoSo) [9]. These SNSs commonly have functionalities that allow users to share their profiles and manage a list of other users, referred to as *friend* with whom they want to share their produced content.

The content shared in SNSs has mainly been profiles, personal daily notes, and reviews. However, a couple of SNSs have introduced a new functionality for sharing multimedia content; for example, we can share video in MySpace within the system. Conversely, a few multimedia web services have started incorporating the SNS functionality to their services to enable users to build social communities in the services. Flickr (photo sharing) [10], Last. FM (music listening habits) [11], and YouTube (video sharing) [12] are examples.

2.2 Mobile P2P Multicast

P2P multicast is a method of delivering content over an application layer in which end nodes not only act as receivers but also as relays to forward the received data to others [1]. This is also called application-layer multicast, application-level multicast, or overlay multicast. This peer-to-peer solution enables us to quickly and easily deploy multicast applications without involving any routers. Recently, P2P multicast protocols for media streaming have been attracting a great deal of attention in both research and industrial fields [13]–[15]. Moreover, mobile P2P networks which mainly consist of mobile nodes have also been attracting a great deal of attention [16]–[19]. However, mobile nodes frequently leave the service because of limited battery capacity, instantaneous disconnection of wireless links, or user behavior [20]. Our previous work has found out that the relationship between forwarders and receivers should be considered in mobile P2P multicast through comparison with several context parameters of users [3]. Inspired by this, we have developed our social network based architecture like we will show in the following section.

3. Mobile P2P Streaming over Social Network

3.1 Scenario Assumption

At the beginning of this section, we describe the model of our assumed mobile P2P multicast. In our assumption, a multicast streaming service is provided by an SNS operator who manages the social network of their users. The service provider distributes video stream to a few tens of nodes, who become the first forwarding nodes; then, the stream is forwarded by nodes in a hop-by-hop manner. The flow of the service is described below:

1. The service provider lists the nodes who want to receive the video stream before the service is provided and constructs a distribution tree.
2. The service provider informs every node of the child nodes listed. At the beginning of the service, every node is required to forward the stream to their child nodes.
3. While nodes are receiving the stream, they are required to continue forwarding in a best-effort manner. They are allowed to stop forwarding if they become incapable of continuing.

With the above scenario, we conducted experiments to determine the lower bound of user contributions in mobile P2P multicast streaming.

3.2 Modeling of User Behaviors for Battery Capacity

While one user is forwarding a stream to other user, the user experiences several physical costs: processing cost, bandwidth cost, and battery cost. Additionally, there are psychological costs. In general, battery cost is the dominant

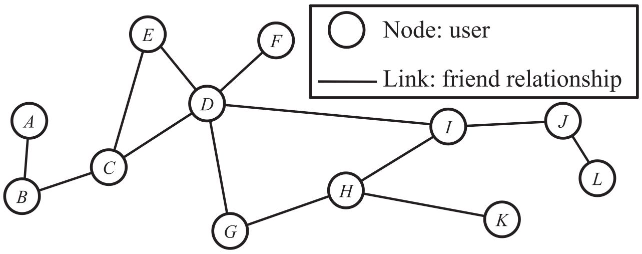


Fig. 2 An example of social graph.

psychological factor because users can do nothing once the battery runs out. It has also been reported that battery capacity limits the feasibility of users as peers in mobile P2P [21]. For these reasons, in this paper, we consider only battery capacity as the psychological cost factor.

Considering battery capacity, we can model user behaviors in mobile P2P multicast streaming as follows: users continue forwarding streams until the remaining battery capacity decreases to threshold θ_R; users may continue forwarding while the battery capacity decreases at a slower rate than threshold θ_S. In other words, if the remaining battery capacity is below θ_R, or the decreasing speed is faster than θ_S, users stop forwarding.

Since θ_R and θ_S are *psychologically* determined, they are different from user to user. Furthermore, these thresholds can change according to the relationship between a forwarder and the receivers if it is displayed for the forwarder who is receiving the forwarded stream. The details of this are described in the next section.

3.3 Psychological Cost Depending on Social Relationship

A social network represents social relationships between users using graphic elements like nodes and links. Figure 2 is an example of a social graph. As is well known, social graphs are scale-free and characterized as a small-world [4], [5]. For our discussion in this paper, we have to mention that the link degree distribution follows a power law, which means, in social networks, a limited number of nodes have many friends (social links), while most nodes have limited number of friends.

In general SNSs, the social graphs are undirected graphs. The hopcount k between two nodes represents the relationship between them; a one-hopcount means they are friends, while a two-hopcount means they have a common friend. In the example of Fig. 2, nodes A and B are friends, while node C is a friend of node A's friend, and vice versa. As we mentioned in the previous section, θ_R and θ_S can change depending on the relationship between a forwarder and the receiver. That is, we can introduce $\theta_R(k)$ and $\theta_S(k)$, where k is the hopcount between a forwarder and the receiver in the *social network* (not multicast network); $\theta_R(1)$ and $\theta_S(1)$ represent the thresholds of a forwarder for forwarding to friends; $\theta_R(2)$ and $\theta_S(2)$ represent those to friends of its friend; and $\theta_R(k)$ and $\theta_S(k) (k > 2)$ represent those to strangers. In general, the smaller the hopcount, the lower the psychological forwarding cost users feel, that is, $\theta_R(k) \leq \theta_R(k+1)$ and $\theta_S(k) \geq \theta_S(k+1)$. Figure 3 illustrates user behaviors that depend on the remaining battery capacity and the thresholds.

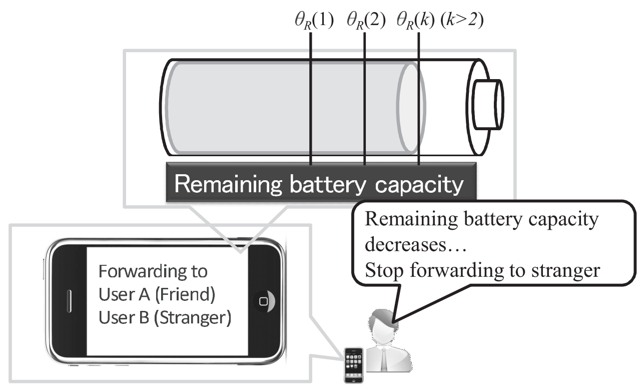


Fig. 3 User behaviors for remaining battery capacity.

4. Proposed Architecture: Distribution Tree Based on Social Graph

In this section, we present the basic concept of our proposed P2P multicast streaming architecture. As we modeled in the previous section, it is expected that users will continue forwarding to their friends when they have a lower battery threshold longer than they will to strangers. Therefore, distributing a multicast stream along the social network can make the service time longer. However, three problems remain when we try to directly use a social network as a distribution tree:

- In social networks, the link-degree distribution follows a power law. In other words, specific nodes have many social links, while others have a very small number of social links. This implies that the load will be unbalanced if we directly distribute the stream along the social network.
- How is the first forwarding node of a distribution tree chosen? For example, if we choose node A as the first forwarding node in Fig. 2, node K receives the stream after six hopcounts. This is undesirable because the jitter, the playback delay and the unreliability of forwarding are accumulated hop-by-hop.
- Even if the graph of the social network is a connected graph, it is not expected that all of the nodes will always join the same mobile P2P service. Therefore, our architecture is required to have a function that constructs a distribution tree even from a disconnected graph.

To solve these problems, we came up with three rules for our distribution tree construction:

Rule 1 We choose the highest central node, which has the minimum average hop distance from the other nodes that join the service, as the first forwarding node.
Rule 2 To balance the forwarding load, we introduce a constant number M as the number of forwarded nodes

(child nodes) per node initially assigned by the service provider. In other words, M represents the number of branches of the distribution tree. Even if a node has many friends that join the service, she is required to forward the stream only to M nodes. Note that nodes having a smaller number of friends than M are also required to forward to M nodes, which is described in detail shortly.

Rule 3 If for a forwarding node, the number of its friends that join the service is smaller than M, the service provider connects a node two hops from the forwarding node (a friend of its friend) to it. If this does not reach M, the service provider connects strangers to it. Therefore, a node will usually receive the forwarded stream from its friend. However, if no friend joins the service, the service provider first assigns a friend of its friend as the parent node. If not available, the node is supposed to receive the stream from a stranger.

```
Max. tree branches M;
int a = 1;
/*Hop count k_ij represents relationship between nodes i and j */
/*If  k_ij==1  node j is node i's friend */
/*If  k_ij==2  node j is friend of i's friend */
/*If  k_ij==3  nodes i and j do not know each other */
while(1)
  for(all  i)
    if(node i is in distribution tree)
      int  s_i = 0;
      while(1)
        while(s_i < M)
          for(all  j)
            if(k_ij==a && node  j  is not in distribution tree)
              locate node  j  as node  i's child;
              node  j joins distribution tree
              s_i ++;
        a++;
  if(all nodes are in distribution tree) terminate algorithm
```

Fig. 4 Construct distribution tree.

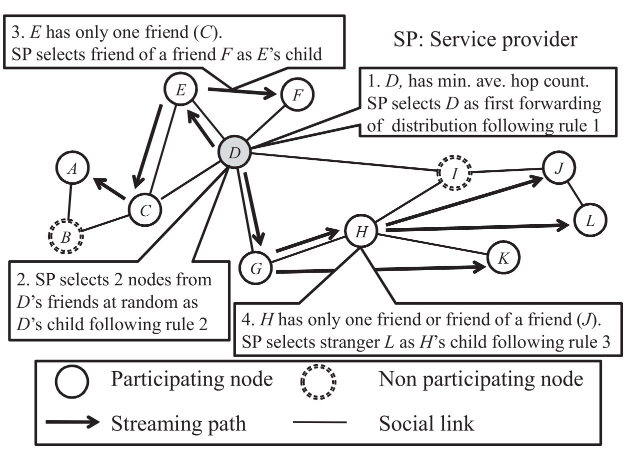


Fig. 5 Example of distribution tree construction.

Figure 4 shows the pseudocode to construct the distribution tree with our proposed rules. Figure 5 shows an example of our distribution tree construction with $M = 2$. We can see in this example nodes B and I do not request the stream. Node D is selected as the first forwarding node of the distribution tree following rule 1. Node D has four friends participating in the service: nodes C, E, F, and G. The service provider picks two nodes from them as node D's child nodes in the distribution tree (nodes E and G), following rule 2. The service provider first tries to select two nodes as the child nodes for E, however he has only one friend, node C. Therefore, the service provider selects node F, who is node D's friend as his child node following rule 3. In such a way, the service provider selects node A as node C's child node, and nodes H and K as node G's child nodes. For node H, who has no friends other than G and K, the service provider assigns nodes J and L as its child nodes to satisfy M following rule 3.

Note that we want to focus on the problem of user cooperation in mobile P2P in this paper, so the optimality of the topology is outside the scope of this paper.

5. Experimental Measurement of Psychological Cost Using Emulator

In this section, we report the results of experimental measurements of the thresholds modeled in Sect. 3. We built an emulator, as shown in Fig. 6, that displays a video together with the remaining battery capacity and the speed of the decreasing battery. We explained to subjects that, 1) the service is provided in an SNS you are in; 2) if you stop your forwarding, the service for a user who is receiving the stream from you will be terminated, and 3) as you forward to more people, the battery capacity will decrease faster.

In the emulation, we assumed the subjects were already forwarding the stream, and they were allowed to stop forwarding considering their remaining battery capacity and the speed of decreasing battery. The subjects were asked to indicate the threshold of remaining battery capacity θ_R and the threshold of battery decreasing speed θ_S. θ_S represents how many times it is as faster as the battery decreasing speed without forwarding. As we discussed in Sect. 3.3, the hopcount in social graph k between the forwarder and

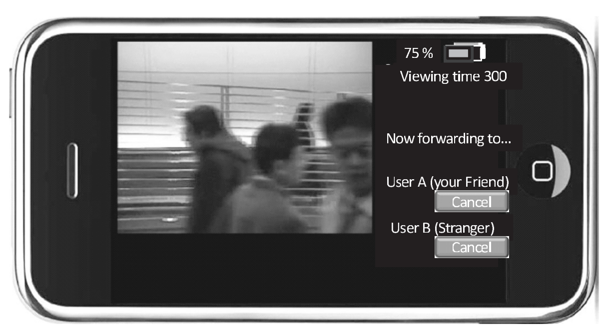


Fig. 6 Emulator for psychological measurement. Remaining battery capacity and playing time are displayed in the upper right.

Table 1 Average battery thresholds for subjects to stop forwarding.

	Ave. $\theta_R(k)$	Ave. $\theta_S(k)$
Friend ($k = 1$)	41.2%	1.86 times
Friends of a friend ($k = 2$)	55.4%	1.73 times
Stranger ($k > 2$)	60.8%	1.44 times

Table 2 Characteristics of CNN model network.

P	0.2	0.4	0.6	0.8
Cluster value	0.1631	0.3159	0.4331	0.5290
No. of links	247.2	326.1	472.9	912.7

the receiver influences these thresholds. We measured the thresholds of the subjects in three cases: the user receiving the stream from you is your friend ($k = 1$), a friend of your friend ($k = 2$), or a stranger ($k > 2$). In each case, we displayed the relationship of the receivers for the subjects in advance. We conducted this experiment with 58 subjects who were all students at Kyoto University. The emulator was built using Flash, which has a high affinity with video and easily enables interactive operation. We encoded an uncompressed video into Flash Video (FLV); the resolution was 400×300 pixels, the frame rate was 30 fps, and the bit rate was 700 kbps. We visualized our emulator like Fig. 6 so that subjects can easily perceive the remaining capacity and the decreasing speed of the battery, and were not reported any difficulty in the perception from the subjects.

Table 1 shows the average of $\theta_R(k)$, and $\theta_S(k)$ for $k = 1, 2$, and $k > 2$ that we measured. The results confirm our presumption given in Sect. 4.

6. Simulation

In this section, we describe how evaluated our proposed method through computer simulations. In our evaluation, to capture the effect of the psychological cost for battery capacity on the performance, we assumed that each node has large bandwidth enough to receive and forward the stream and no new node arrives at the service once it has started. In addition, even if a node terminates forwarding, we did not reassign its downstream nodes to another node because this requires a smart topology repairing algorithm, which is still an open issue in mobile P2P multicast and out of scope in this paper.

6.1 Simulation Model

6.1.1 Distribution System Model

To distribute the streaming data to more than ten thousand nodes, the service provider forwards the stream to a few tens of nodes from the source server. Each first forwarding node is positioned at the top of the distribution *stub*, which is a unit size of the distribution tree. As mentioned in Sect. 3, since we need to limit the maximum number of hopcounts due to the jitter constraint, we assume that one distribution stub includes 400 nodes at most.

6.1.2 Agent Model

We call nodes in our simulations agents because one agent precisely reflects the θ_R and the θ_S that are measured from one subject. We have conducted in this type of simulation, which is referred to as multi-agent simulation [3]. Agents in the simulation behave precisely as the subjects corresponding to them; they stop forwarding according to the individual θ_R and θ_S. When the number of agents in a simulation is larger than 58, which is the number of subjects, we randomly assign 58 values for θ_R and θ_S to each agent.

We also measured the user characteristics in each subject: regarding their real communities and their SNSs, how many friends they have in each of them. Through the correlation inspection, we did not observe any correlation between the number of friends of a subject and the thresholds of the subject.

6.1.3 Social Network Model

In our simulations, we also need to construct a social graph. Connecting Nearest Neighbors (CNN) is a network growth model that incorporates a process of connecting nearest neighbors [22]. The CNN model is known to be one of the network models that express social networks. In the CNN model, parameter P is the probability that a new agent will create a link with friends of its friend. Table 2 lists the number of links vs. CNN parameter P in a CNN network with 200 agents. The cluster value in this table represents the density of the friend relationships in the network. In Sect. 6.2.4, we will observe the performance of our method with varying P.

We defined α as the ratio of agents that request the stream to the total number of agents in the distribution stub n. In this simulation, we generated a social network with n agents using the CNN model, and randomly chose αn agents as the ones that actually request to receive the stream. In the case of $n = 200$ and $\alpha = 0.8$, the service provider constructs the distribution tree with 160 agents.

6.1.4 Energy Consumption Model

We introduce an energy consumption model, which is defined as:

$$E_i = m_i E^f + E^c [J/unittime] \tag{1}$$

Here, E_i, E^f, and E^c represent the total energy consumption of agent i per unit time, the single forwarding energy consumption, and other consumed energy, respectively, and m_i represents the number of forwarded agents (child agents) from agent i ($m_i < M$). Then, we normalize Eq. (1) by E^c,

$$\bar{E}_i = m_i E^f / E^c + 1 [/unittime] \tag{2}$$

which means agents consume 1 (normalized) energy per unit time even without forwarding ($m_i = 0$). The consumed energy linearly increases as m_i increases. For convenience,

$$E_f/E_c = \epsilon \tag{3}$$

The larger ϵ means the energy consumption for forwarding is more dominant in the total energy consumption. We should explicitly show how $\bar{E}_i^t$ is related to θ_R in our simulations. Suppose that agent i is forwarding to a receiver; the agent stops forwarding when,

$$B_i - \sum_{t=0}^{T} \bar{E}_i^t(m_i(t)) \leq \theta_R^i(k) \tag{4}$$

or

$$\bar{E}_i^t \geq \theta_S^i(k) \tag{5}$$

where B_i is the initial battery capacity of agent i, T is the current time, and $\theta_R^i(k)$ and $\theta_S^i(k)$ are the θ_R and θ_S when agent i forwards to an agent k hop far from it in the social graph. The value $\bar{E}_i^t$ is comparable to θ_S because this equivalently indicates the decreasing speed of the remaining battery capacity. In reality, users discretely detect the decrease of the battery capacity. Therefore, in our simulation, we assume the remaining battery capacity is displayed by ten levels and agents recognize the decreasing speed when the level falls down.

We cannot deterministically use ϵ in Eq. (3) because E_i includes energy for receiving, processing, decoding, encoding, and displaying; ϵ can be different from device to device. Therefore, we will evaluate our method with varying ϵ. Here, we set the initial battery capacity for each agent to a random value between 12000 and 15000. 15000 is the maximum capacity of the batteries, that is, 1500 corresponds to one level of the displayed battery mentioned above. Why we set the minimum to 12000 is because the experimental measurement described in Sect. 5 suggested that most of the users do not terminate their forwarding to strangers while their remaining battery capacity is more than 80% of the maximum battery capacity. In other words, if we set the initial battery capacity less than 80%, a certain number of users terminate their forwarding too soon after the service has started, which would make us hard to capture the effect of psychological cost for battery capacity.

6.1.5 Compared Methods

In this evaluation, we compare our method with four alternate methods.

- PFF: the proposed method.
- PF: the simplified proposed method; the distribution tree is constructed following rules 1, 2, and 3 described in Sect. 4 as PFF does, but friend of a friend relationship is not considered.
- RAND: RAND randomly locates agents in the distribution tree. It does not display the relationships to forwarding agents. RAND in which agents do not consider the relationship to others is the simplest conventional tree-based scheme.

Table 3 Simulation parameters.

No. of potential nodes in distribution stub n	200
Participate ratio α	0.8
CNN parameter P	0.5
Initial. battery capacity B_i	12000~15000
No. of branches of tree M	3
Ratio of battery consumption of single forwarding to others ϵ	0.5

- RR: RR constructs the distribution tree at random. However, unlike RAND, RR displays the relationships with the child agents to the forwarding agents. Therefore, in RR, the reduction of psychological cost can happen accidentally.
- SG: social networks are directly used as a distribution stub. This implies every agent always forwards to their friends.

6.2 Simulation Results

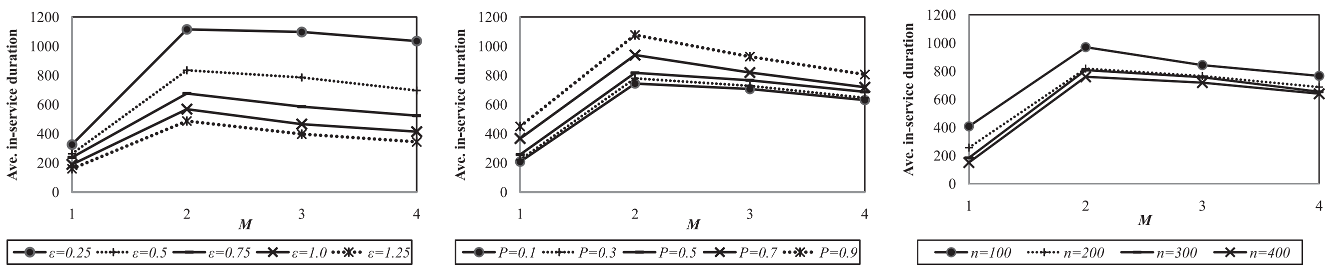
We evaluate and compare our method and the alternate methods introduced in Sect. 6.1.5. We used the parameters listed in Table 3 except when we observe the performance with varying ones of these parameters.

6.2.1 Distribution Tree Characteristics

Before the comparative evaluation with the other methods, we discuss the number of tree branches M in PFF. As an M increases, each node is required to forward a larger number of streams, while the maximum number of hopcounts becomes smaller. As described in Sect. 4, a larger number of hopcounts causes larger jitter and playback delay, which should be avoided, especially in mobile networks. Thus, a tradeoff exists between the load per node and the jitter/delay.

Figure 7 plots the average in-service duration of our method vs. ϵ, P, and n. The in-service duration is the duration from the initial time of the simulation to the time when the service for the agent is terminated because of the stop forwarding from its parent agent or because its battery has run out. This is used for just representing relative superiority of our methods to other methods. In Fig. 7 we can observe that $M = 2$ gives the best performance for every parameter, followed by $M = 3$. This is because an M that is too large decreases the remaining battery capacity of the agent very quickly, while the distribution tree is not robust against forwarding termination by an agent when $M = 1$.

Table 4 gives the average hopcounts and maximum hopcounts vs. M when the number of agents in the distribution stub is 160. As we have mentioned above, taking jitter and delay into account, we should choose large M to minimize the hopcounts, while too large M degrades the performance as in Fig. 7 because the number of forwarded streams per user becomes too large. In the following simulations, we set $M = 3$ as the basic parameter though the optimal M in a practical service depends on the acceptable delay and jitter and the number of acceptable forwarding streams per user.



[a] Parameter ϵ [b] Parameter P [c] Parameter n

Fig. 7 Performance of PFF vs. M with parameters ϵ, P, and n.

Table 4 Hopcounts vs. M.

M	1	2	3	4
Ave. hopcounts	80.0	4.35	3.10	2.66
Max. hopcounts	159	7	5	4

Table 5 No. of social relationship in each method.

	PFF	PF	RAND	RR	SG
Friend	45	49	0	4	199
Friend of a friend	48	0	0	8	0
Stranger	106	150	199	187	0

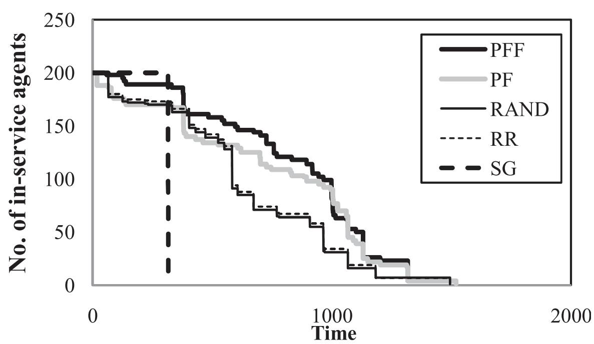


Fig. 8 Basic performance.

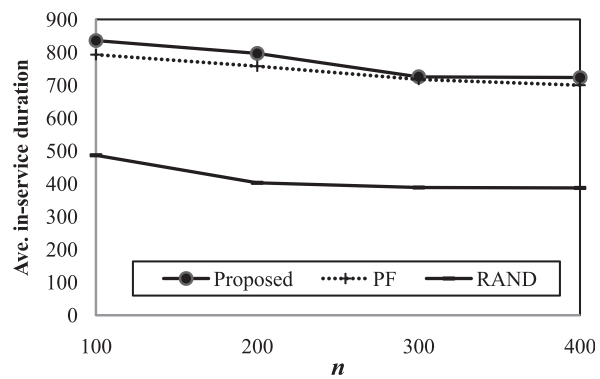


Fig. 9 Ave. in-service duration vs. n.

6.2.2 Basic Performance

Here, we compare the performance of the methods described in the previous section through a single simulation trial. Figure 8 plots the number of in-service agents, which are agents that successfully receive the stream as a function of the simulation time. We used the basic parameters in Table 3, and we set $\alpha = 1$. We can see that our method is obviously superior to the other methods. SG is very poor because, as we mentioned in Sect. 4, the forwarding load on agents with many friends is very high in SG. We observed when SG was used that the first forwarding agent decided to stop all of its forwarding according to its θ_R or θ_S, and then no stream was forwarded downstream. Table 5 indicates how many friend (and friend-of-a-friend) relationships are used in each method. Our methods PFF and PF obviously use many friend relationships, resulting in the large enhancement of the average in-service time. On the other hand, there is a very small difference between RR and RAND. As shown in Table 5, RR uses only four friend relationships and eight friend-of-a-friend relationships. From these results, we cannot expect that the reduction in psychological cost by a social relationship happens accidentally.

In other words, we need to construct the distribution tree itself based on the social relationships (or the social network). In the following sections, we omit RR and SG.

6.2.3 Performance vs. n

In this section, we evaluate PFF, PF, and RAND vs. n, which represents the total number of agents in the distribution stub. Note that, since α is fixed at 0.8, the number of agents that request to receive the stream is 80% of n. Figure 9 plots the average in-service duration vs. n. Our proposed PFF and PF methods are superior to RAND, while PFF is slightly but certainly superior to PF because the relationship of a friend of friends is taken into account in PFF but not in PF. As n increases, the performance of each method decreases because the number of hopcounts increases.

We also observed the performance vs. α, which represents the ratio of the number of agents that request to receive the stream. Although we did not plot the figure, we confirmed that the superiority of PFF was maintained regardless of α. These results with varied n and α have proved that our methods have high scalability for network size.

6.2.4 Performance vs. P

In this section, we observe the performance of PFF and PF

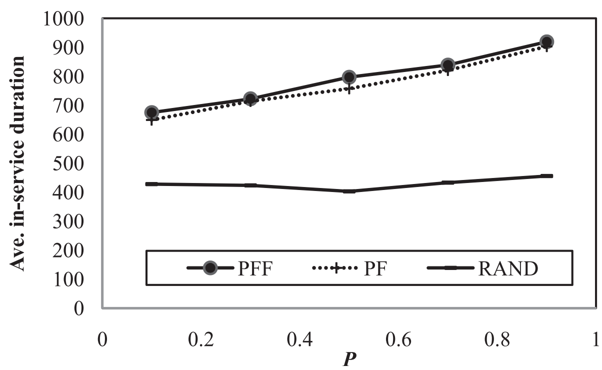


Fig. 10 Ave. in-service duration vs. P.

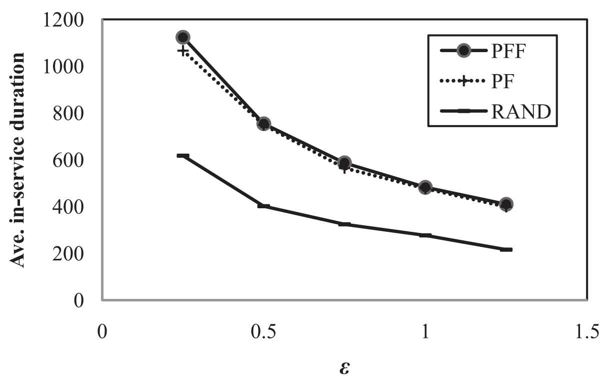


Fig. 11 Ave. in-service duration vs. ϵ.

with varying social network parameter P, which was explained in detail in Sect. 6.1.3. Figure 10 is the result. As P is increased, PFF and PF give better performance simply because the larger P brings the agents more friends in the social network. Note that, even in the case of $P = 0.9$, strangers are connected to each other in PFF and PF: the numbers are around 20 and 40 in PFF and PF, respectively.

6.2.5 Performance vs. ϵ

In this section, we evaluate the performance vs. ϵ. As we can easily predict, the performance of each method decreases as ϵ increases. We have to mention that PF and PFF still maintain superiority over RR even when ϵ is larger than 1.0. From Eqs. (2) and (3). $\epsilon = 1$ means that forwarding for one agent consumes an equal amount of energy as the total consumed energy for the other factors in a mobile device. Figure 11 plots the in-service duration vs. ϵ. This result verifies that our methods work better than the compared method even under such a strict energy constraint.

7. Discussion: Remaining Issues in Mobile P2P

We here discuss the remaining issues in mobile P2P multicast we did not deal with in this paper. Wolfson et al. [2] suggested there are three resource constraints in mobile P2P: bandwidth, energy, and storage. This means 1) a node with instable wireless bandwidth cannot maintain the quality of the forwarded stream high, while a node with narrow wireless bandwidth can forward the stream at the limited rate; 2) mobile nodes easily disconnect from the service because of running out the batteries and poor wireless channels. Those disconnections bring their downstream users interruption time even in the middle of the service [23], [24]; 3) in general, mobile nodes are equipped with the limited buffer capacity. Therefore, it is too optimistic to expect that the equipped buffers completely solve the delay/jitter problems.

Therefore, what we need to consider in mobile P2P multicast is how to suppress the negative effects listed above that propagate from the upstream to the downstream in the topology. The key techniques for this should be topology construction, maintenance, and repairing schemes that consider the capability of mobile nodes including bandwidth, energy, and storage. These are beyond the scope of this paper but should be included in future work.

8. Conclusion

In this paper, we commented on the demand for mobile P2P multicast streaming services and pointed out a remaining issue: the high psychological forwarding cost in mobile P2P multicast. To solve the problem, we first modeled user behaviors depending on the remaining battery capacity. Then we proposed a novel social-network based mobile P2P multicast streaming architecture that suppresses the psychological forwarding cost. We also measured the psychological forwarding cost from real users through an emulation experiment. We evaluated our architecture through a multi-agent simulation, in which the psychological costs we measured were directly incorporated. We confirmed our social network based architecture increased the in-service duration of users in mobile P2P multicast by reducing the psychological cost for forwarding stream to the other users.

In the future, we plan to develop our method so that it can be applied to more realistic conditions such as wireless channels, topology reconstruction, and complicated network model and to design an incentive mechanism that motivates users to contribute to other users and that ensure the fairness between users.

Acknowledgement

The authors wish to thank Hiroyuki Kasai (Graduate School of Information Systems, The University of Electro-Communications), and Makoto Yoshino (AVC Networks Company, Panasonic, Ltd.) for their great advices on this paper. This work is supported in part by the National Institute of Information and Communications Technology (NICT), Japan, under Early-concept Grants for Exploratory Research on New-generation Network and by the Mobile Radio Center Foundation, Japan, under MRC R&D Grant for Mobile Wireless.

References

[1] D. Schoder and K. Fischbach, "Peer-to-peer prospects," Commun. ACM, vol.46, no.2, pp.27–29, Feb. 2003.
[2] O. Wolfson, B. Xu, and R.M. Tanner, "Mobile peer-to-peer data dissemination with resource constraints," Proc. 8th International Conference on Mobile Data Management (MDM'07), pp.1–15, Mannheim, Germany, May 2007.
[3] M. Yoshino, H. Kubo, R. Shinkuma, and T. Takahashi, "Modeling user cooperation problem in mobile overlay multicast as a multi-agent system," IEEE Globecom 2009, Honolulu, HI, US, Nov. 2009.
[4] M.O. Jackson, Social and Economic Networks, Princeton University Press, NJ, 2007.
[5] A. Mislove, M. Marcon, K. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," Proc. 5th ACM/USENIX Internet Measurement Conference (IMC'07), San Diego, CA, USA, Oct. 2007.
[6] D.M. Boyd and N.B. Ellison, "Social network sites: Definition, history, and scholarship," J. Computer-Mediated Communication, vol.13, no.1, pp.210–230, Dec. 2007.
[7] Facebook, http://www.facebook.com/
[8] MySpace, http://www.myspace.com/
[9] N. Eagle and A. Pentland, "Social serendipity: Mobilizing social software," IEEE Pervasive Computing, vol.4, no.2, pp.28–34, 2005.
[10] Flickr, http://www.flickr.com/
[11] Last.fm, http://www.last.fm/
[12] Youtube, http://www.youtube.com/
[13] J. Liu, S.G. Rao, B. Li, and H. Zhang, "Opportunities and challenges of peer-to-peer internet video broadcast," Proc. IEEE, Special Issue on Recent Advances in Distributed Multimedia Communications, vol.96, no.1, pp.11–24, 2007.
[14] F. Wang, Y. Xiong, and J. Liu, "mTreebone: A hybrid tree/mesh overlay for application-layer live video multicast," Proc. IEEE ICDCS'07, June 2007.
[15] BitTorrent DNA, http://www.bittorrent.com/
[16] T. Noguchi and M. Yamamoto, "Robust application-level multicast tree construction for wireles/mobile hosts," Lect. Notes Comput. Sci., vol.3970, pp.108–119, 2006.
[17] T. Hobfeld, K. Tutschku, and F.U. Andersen, "Mapping of file-sharing onto mobile environments: Feasibility and performance of eDonkey with GPRS," Proc. IEEE Wireless Commun. & Networking Conf. (WCNC), no.1, pp.2453–2458, New Orleans, LA, USA, March 2005.
[18] J. Oberender, F.U. Andersen, H. Meer, I. Dedinski, T. Hosfeld, C. Kappler, A. Mader, and K. Tutschku, "Enabling mobile peer-to-peer networking," Wireless Systems and Mobility in Next Generation Internet (LNCS 3427), pp.219–234, Springer, 2005.
[19] W. Kellerer, Z. Despotovic, M. Michel, Q. Hofstätter, and S. Zöls, "Towards a mobile peer-to-peer service platform," Proc. IEEE SAINT-W, pp.15–19, Hiroshima, Japan, Jan. 2007.
[20] H. Kubo, R. Shinkuma, and T. Takahashi, "Mobile overlay multicast using information on physical stability for robust video streaming," Proc. IEEE PIMRC 2009, Tokyo, Japan, Sept. 2009.
[21] O. Kassinen, Z. Ou, and E. Harjula, "Effects of peer-to-peer overlay parameters on mobile battery duration and resource lookup efficiency," Proc. 8th International Conference on Mobile and Ubiquitous Multimedia (MUM'08), pp.177–180, Umeå, Sweden, Dec. 2008.
[22] A. Vazquez, "Growing network with local rules: Preferential attachment, clustering hierarchy, and degree correlations," Phys. Rev. E, vol.67, 056104, May 2003.
[23] X. Zhang, J. Liu, B. Li, and T.-S.P. Yum, "CoolStreaming/DoNet: A data-driven overlay network for efficient live media streaming," Proc. INFOCOM'05, vol.3, pp.2102–2111 March 2005.
[24] T. Kusumoto, J. Katto, and S. Okubo, "Tree-based application layer multicast using proactive route maintenance and its implementation," Proc. ACM P2PMMS 2005, Singapore, Nov. 2005.

Hiroyuki Kubo received the B.E. degree in Faculty of Integrated Human studies from Kyoto University in 2007, and the M.E. degree in Communications and Computer Engineering, Graduate School of Informatics from Kyoto University, Kyoto, Japan, in 2009. He is currently working toward the Ph.D. degree in Communications and Computer Engineering, Graduate School of Informatics, Kyoto University. He has been a Research Fellow of the JSPS since 2010. His current research interest is in mobile multicast, QoS and QoE in wireless networks, user cooperation in mobile networks, opportunistic communications. He received the TELECOM System Technology Award for Student from The Telecommunications Advancement Foundation in 2008, IEEE VTS Japan 2010 Young Researcher's Encouragement Award in 2010, respectively. He is a student member of IEEE.

Ryoichi Shinkuma received the B.E., M.E., and Ph.D. degrees in Communications Engineering from Osaka University, Japan, in 2000, 2001, and 2003, respectively. In 2003, he joined the faculty of Communications and Computer Engineering, Graduate School of Informatics, Kyoto University, Japan, where he is currently an Assistant Professor. He was a Visiting Scholar at Wireless Information Network Laboratory (WINLAB), Rutgers, the State University of New Jersey, USA, from 2008 Fall to 2009 Fall. His research interests include architecture, protocol, and model designs in wireless networks and overlay networks particularly inspired by social and economic aspects. He received the Young Researchers' Award from IEICE in 2006 and Ericsson Young Scientist Award in 2007, respectively. He is a member of IEEE.

Tatsuro Takahashi received the B.E. and M.E. in Electrical Engineering from Kyoto University, Kyoto, Japan, in 1973 and 1975 respectively, and Dr. of Engineering in Information Science from Kyoto University in 1997. He was with NTT Laboratories from 1975 to 2000, making R&D on high speed networks and switching systems for circuit switching, packet switching, frame relaying, and ATM. Since July 1, 2000, he is a Professor, Communications and Computer Engineering, Graduate School of Informatics, Kyoto University. His current research interests include high-speed networking, photonic networks and mobile networks. Prof. Takahashi received the Achievement Award from IEICE in 1996, and the Minister of Science and Technology Award in 1998. He was a Vice President of the ATM Forum from 1996 to 1997, and the Chairman of the Network Systems (NS) Technical Group in the Communications Society of IEICE from 2001 to 2002. Prof. Takahashi is an IEEE Fellow.