Peer-to-Peer Netw Appl (2008) 1:75–90
DOI 10.1007/s12083-007-0007-x

# An inexpensive unstructured platform for wireless mobile peer-to-peer networks

**Mursalin Akon · Xuemin Shen · Sagar Naik · Ajit Singh · Qian Zhang**

Received: 2 August 2007 / Accepted: 28 December 2007 / Published online: 2 February 2008


**Abstract** In this paper, we propose an unstructured platform, namely ***I**nexpensive **P**eer-to-**P**eer **S**ubsystem (IPPS)*, for wireless mobile peer-to-peer networks. The platform addresses the constraints of expensive bandwidth of wireless medium, and limited memory and computing power of mobile devices. It uses a computationally-, memory requirement- and communication- wise inexpensive gossip protocol as the main maintenance operation, and exploits location information of the wireless nodes to minimize the number of link-level messages for communication between peers. As a result, the platform is not only lightweight by itself, but also provides a low cost framework for different peer-to-peer applications. In addition, further enhancements are introduced to enrich the platform with robustness and tolerance to failures without incurring any additional computational and memory complexity, and communication between peers. In specific, we propose schemes for a peer (1) to chose a partner for a gossip iteration, (2) to maintain the neighbors, and (3) to leave the peer-to-peer network. Simulation results are given to demonstrate the performance of the platform.

**Keywords** Peer-to-peer network · Wireless network · Mobile network · Communication platform

M. Akon · X. Shen (✉) · S. Naik · A. Singh
Department of ECE, University of Waterloo,
Ontario, Canada
e-mail: xshen@bbcr.uwaterloo.ca

M. Akon
e-mail: mmakon@ece.uwaterloo.ca

S. Naik
e-mail: naik@ece.uwaterloo.ca

A. Singh
e-mail: asingh@ece.uwaterloo.ca

Q. Zhang
Hong Kong University of Science and Technology,
Kowloon, Hong Kong, China
e-mail: qianzh@cs.ust.hk

## 1 Introduction

Peer-to-peer (P2P) networks have received considerable attention because of their broad applications such as file sharing [26], access of medical reports of patients among health service providers [15], faster access to *hot* on-line objects [35], and robust media streaming [25, 39], etc. P2P network is a distributed application layer network. Unlike traditional client-server based applications, peers[1] in P2P network collaborate to achieve a certain goal. Design of a good P2P network requires a well thought logical structure among the participating peers. Structure-wise, P2P networks can be divided into three categories: (1) centralized, (2) decentralized but structured and (3) decentralized and unstructured [22]. In a centralized network, peers rely on a central host for a few services. As a result, the central host is subject to a single point of failure. The decentralized but structured network has been well studied and is still a very active area of research. The topology of the members in such a network is ruled by several constraints. Contents are distributed among the members using either some hints [5] or the topology of the members [30, 31]. The

[1] In this paper, we use the terms peer, user, node or member of a P2P network interchangeably.

distributed and unstructured network has neither any central host to provide some crucial services nor any precise control over resource distribution. The topology may be constructed using some knowledge about the physical or logical properties of the underlying physical network but, unlike the structured network, it puts no constraint on content distribution. A well-known example of such a network is Gnutella [12].

With the deployment of high bandwidth 3G (and expected deployment of 3.5G and 4G) cellular networks and wireless LANs, there is an increasing interest in wireless P2P networks. However, results obtained for the wired networks can not be directly deployed in the wireless P2P networks due to the limitations of the wireless medium, expensive bandwidth, and the limitations of the mobile devices due to small memory and limited computation power. Therefore, like any application for wireless mobile networks, a P2P network should be computationally efficient, modest in memory requirement, and economical in bandwidth uses. Significant research efforts have been put recently to use P2P networks in wireless environment to tackle management problems such as, routing [10, 20], clustering [29], service discovery [4, 34], and applications such as, multimedia distribution [11, 43], game development [41, 42], file sharing [23], etc.

In this paper, we propose ***I**nexpensive **P**eer-to-**P**eer **S**ubsystem (IPPS)* [24], a location aware P2P network platform for dense wireless mobile networks. IPPS is unstructured in nature. It reduces the number of link-level message flows in the network, and consumes less energy. IPPS employs a gossip protocol to maintain the P2P network and uses information about the locations of the neighbors to minimize the number of hops between peers. At each gossip iteration, a peer builds up neighborhood relation with nearer peers and discards the peers at distant. Furthermore, the proposed platform is computationally and memory requirement-wise cheap. Given the limitations of wireless medium and wireless devices, IPPS provides a low cost and practical platform for real wireless mobile P2P applications. IPPS identifies impoverished gossip cycles and systematically utilizes them to make the platform robust. At the same time, IPPS parameters provide the platform with soft constraints which make it flexible for the users and tolerant to the dynamics of a distributed environment. A wide variety of applications, ranging from caching [1] to streaming multicast [17], can easily be developed on top of the proposed platform. In addition, to assist the development process, we describe a minimal interface exposed by the platform to the applications of interest. Extensive simulations have been performed to evaluate the proposed platform.

The remainder of the paper is organized as follows. Section 2 discusses some of the related works and the motivation behind this work. The basics of our proposed platform are presented in Section 3. Enhancements to the proposed platform are introduced in Section 4. Section 5 discusses some of the performance measurements from our simulations. Finally, the paper is closed with a follow-up discussion and conclusion in Sections 6 and 7, respectively.

## 2 Preliminaries

A wireless mobile network is a cooperative network where each node requires to collaborate with each other to forward packets from a source to a destination. In such a network, the entire available channel capacity may not be available to an wireless application, and the actual throughput is also determined by the forwarding load generated by other wireless nodes. Besides, mobile devices are battery operated. Unlike electronics, advances in battery technology still lag behind. Minimizing the number of link-level wireless hops helps in increasing the capacity available to the applications. Reduced number of link-level hops also means less number of transmission and less power consumption for a mobile node. Along with being thrifty about bandwidth consumption, a suitable application for mobile devices is required be computationally inexpensive to ensure prolonged battery life, and memory requirement-wise economical to confirm accommodation in the small system memory.

In spite of the limitations of wireless mobile networks, P2P over high capacity cellular networks and wireless LANs can provide a wide range of services such as sharing files [23]. In scenarios where accessing a commercial network is expensive, members of a P2P network can share downloaded objects with each other or even can collaborate to download a large popular object. This not only provides a cheaper way of sharing resources, but also enables low latency access to remote objects. Dissemination of rescue or strategic information in a disaster or war zone can be accomplished using mobile wireless P2P network. Short message broadcast, multimedia broadcast, text, audio and / or video based conference are some other examples.

Recently, large number of research articles on P2P networks have been appeared in the literature. At the same time, several implementations of different P2P networks became available for the users. Some of the highly structured P2P networks are CAN [30], Chord [36], Past [8, 31], SCRIBE [3, 32] and Tapestry [44]. Those networks employ specific resource placement

algorithms which are tightly coupled with the P2P topology. To retrieve or query for a resource, they use topology specific (in turn, resource distribution specific) routing mechanism. As a result, a search can be performed very efficiently in this type of networks. However, if the identification of a resource is partially available (i.e., not all properties of the meta data are available), a search fails. Moreover, due to this impractical assumption about the resource distribution policies, those networks have not been widely deployed. Freenet [5] and Tarzan [9] are examples of loosely coupled distributed P2P networks. Some of those networks use a centralized directory which is not robust. Others use hint-based resource distribution which can not support searching of objects whose information are partially available. Besides simple resource sharing, some of the loosely coupled P2P networks have considered issues like trust and security. Some researches propose to use existing or modified structured networks in wireless and ad-hoc networks. For example, XScribe [27] is modified from SCRIBE [3] to suite in ad-hoc networks. A comprehensive study of similar researches can be found in [2].

In general, structured P2P networks mandate that all the peers in the network fully conform with the system requirements. To satisfy that condition, all the peers must abide by the rules set by the administrative body. However, it is very difficult to achieve such a goal in a highly distributed environment. As a result, structured P2P networks are not able to gain popularity for resource sharing in an environment without any central administrative control such as, the Internet. However, success has been reported in developing large scale distributed storage system [8], scalable publish/subscribe system [32], and application level multicast or broadcast protocols [3]. On the other hand, a structured P2P network faces a high cost of maintenance of the network and the ability of this type networks to work in extremely unreliable environments has not yet been investigated. On the contrary, an unstructured P2P network is a low cost network which can sustain any extreme environment [35]. Although such a benefit is achieved at the expense of higher search cost, the network assumptions and the overall gain have made this kind of P2P networks so attractive that several unstructured P2P networks have been deployed and are being used by a huge user communities. For instance, an unstructured P2P network, named *PROOFS* [35], has been proposed to share hot Web content. The heart of PROOFS is a periodic gossip protocol, called *shuffle*, where two random neighboring peers rearrange their P2P neighbor sets through an exchange of randomly selected neighboring peers. Though the shuffle operation is simple and inexpensive, query success rate for popular objects is excellent (more than 95%). With a strong theoretical background, PROOFS is an excellent unstructured P2P network for wired systems where computing power and network bandwidth are ample, and changes to the membership of the P2P network are rare. With the limitations of wireless medium and mobile devices, and dynamic join and leave of the mobile peers in the P2P networks, the benefit of randomness in PROOFS diminishes. As we will compare IPPS with PROOFS in the later discussion, it becomes evident that deployment of PROOFS in wireless mobile networks does not yield in a well performing P2P network.

In a wired network, due to the abundance of resources, performance metrics of many applications are abstract. However, P2P networks in wireless mobile environment should be very economic about the resources of the wireless medium and devices. In this paper, we take first step towards a low cost P2P network over wireless mobile environment. Our goal is to propose an inexpensive and well performing P2P platform on which different P2P applications can be developed. To achieve the goal, an unstructured P2P network, exploiting location information, is examined. While designing the platform, careful choices are made to make it flexible, robust and fault tolerant. Note that those aspects are expected properties of an application, running on a highly dynamic environment, such as a wireless mobile network.

## 3 The basics of the proposed platform

In this section, we describe the basics of our proposed platform. In Sub-section 3.1, we describe the system model we consider in the reminder of the paper. In Subsection 3.2 to Sub-section 3.5, different components of IPPS are elaborated. In Sub-section 3.6, we evaluate the computational and required memory complexities of the platform. Some simplified analytical properties of the platform is showed in Sub-section 3.7. Finally, in Sub-section 3.8, we show some initial performance evaluation results.

### 3.1 System model

Our system model consists of a set of collaborative computing nodes, each equipped with a wireless interface. We assume that those nodes can form a network on-the-fly using an ad-hoc networking technology. In this research, we consider GeRaf [45, 46], an efficient location aware transmission (MAC) and forwarding (routing) scheme, to manage the network. Though

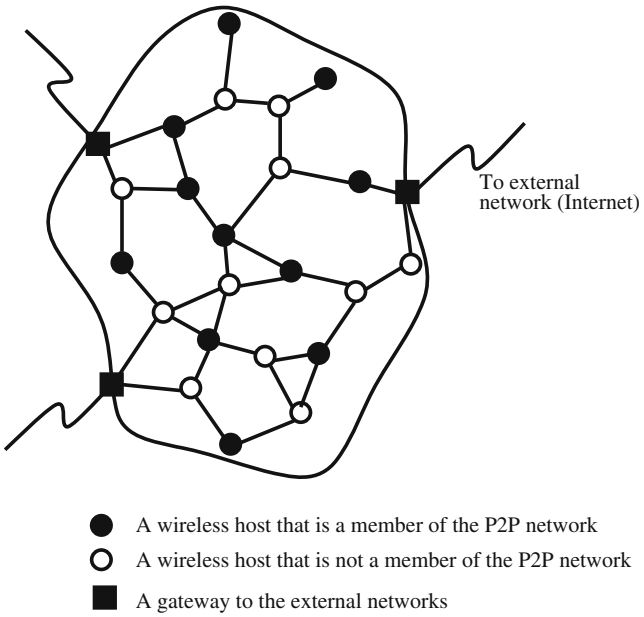



**Fig. 1** A P2P network

search for hardware technologies to engineer ad-hoc networks are still an active area of research, several of them have already been implemented in WLAN [14] and are intended to be implemented in future cellular networks [18, 19, 21].

In our model, for each node, participation in the P2P network is optional. However, irrespective of its membership in the P2P network, each node participates in routing messages from one node to another as a low level service. We assume that the network is equipped with low level (lower than application level) point-to-point unicast primitives, and each of the mobile devices has access to some form of location service [6, 28]. Through this location service, a node in the network can obtain the physical location of itself or other nodes. The information from the location service is used by the lower level network management (i.e., GeRaf) as well as by the P2P modules (i.e., IPPS library). As a result, either the network management modules expose interfaces to share the location information or be combined with the P2P modules as a cross-layer application. Figure 1 shows the considered network.

### 3.2 Maintaining the topology

Reformation is a gossip protocol where a number of neighbors are exchanged between peers. The concept of exchanging neighbors follows from the *shuffle* operation of *PROOFS* [35]. However, the goals of these two operations are exclusive. The purpose of shuffle is to provide randomness in the network, where as, reformation makes attempts to being neighboring peers closer to each other. As a result, the processes to exchange peers in IPPS and PROOFS are entirely different. We make the following claims about reformation.

*Claim:* It is expected that reformation reduces link level hop count between neighboring peers.

A peer $p$ of the network maintains a set of neighboring peers, denoted as $\mathcal{N}_p$. In this network, $q$ being the neighbor of $p$, i.e., $q \in \mathcal{N}_p$, does not necessarily mean that $p$ is also a neighbor of $q$. In other words, the neighborhood relation is unidirectional. Each peer performs reformation at a regular interval. During a reformation, $p$ exchanges $l$ number of neighbors with a participating peer, where $|\mathcal{N}^p| > l > 0$. The neighborhood relation of the participating peers is reversed after the reformation operation. Peer $p$ chooses the participating peer $q$ among its own neighbors with the intention of reducing the total distance between the peers. Distance between two peers convey the idea of physical distance between them. We expect that hop count between two neighboring peers is proportional to the distance between them. In fact, for our system model the following theorem holds.

**Theorem 1** *The expected number of hops between any two peers is an increasing function of the distance between them* (*See* [45] *for the proof*).

*Claim:* Reformation reduces the bandwidth requirement to forward P2P messages.

A peer usually forwards P2P messages, such as query messages, to its P2P neighbors only. As not all communication nodes participate in the P2P network, a P2P level hop may consist of several link level hops. Figure 2 shows the idea pictorially. There exists one non-P2P node between $s$ and $u$ (i.e., two hops), whereas there are two non-P2P nodes between $u$ and $v$ (i.e., three hops). In a random P2P network, on an average one P2P hop consists of average link level path length of the network. In the worst case, where two neighboring peers are located at the extreme ends, a single P2P hop has a link level hop count which is equivalent to the network diameter. Having a neighbor located

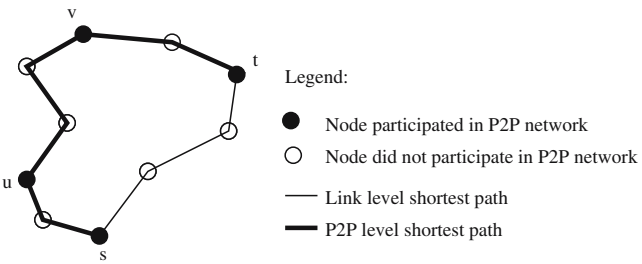



**Fig. 2** Shortest path in P2P and link level network

at a nearby location results in reduction in number of hops between the peers. This helps in reducing of number of link level messages which helps in reducing the total bandwidth consumption to forward P2P messages. Moreover, fewer hops mean reduced message latency. Note that both of these properties are very much desirable for wireless mobile applications, as reduced number of link level messages slows down energy consumption and boosts battery life of mobile devices.

To have peers located at a close geographic area, we introduce the concept of *distance gain*. During a reformation procedure between peers $p$ and $q$, if the initiating peer $p$ forwards another P2P neighbor $r$ to $q$, the distance gain is the reduction of the distances between the pairs $p$ and $r$ and the second pair $q$ and $r$. Figure 3 shows a reformation step where a directed edge from any peer $x$ to another peer $y$ means that $y$ is a neighbor of $x$. Now, the distance gain is formally given by:

$$d^p_{q,r} = |dist(p,r)| - |dist(q,r)| \tag{1}$$

where $dist(x, y)$ is the distance between $x$ and $y$. When a peer $p$ wants to engage in a reformation process, it finds the peer which results in the maximum distance gain. To compute such a metric, for each $q \in \mathcal{N}^p$, $p$ performs the following computations.

1. It computes a *preliminary reform-set* $\mathcal{NR}^p_q$ such that $|\mathcal{NR}^p_q| = l-1$ and $\mathcal{NR}^p_q \subset \mathcal{N}^p - \{q\}$. The preliminary reform-set must satisfy the following condition:

$$d^p_{q,u} \geqslant d^p_{q,v} \tag{2}$$

   where $u \in \mathcal{NR}^p_q$ and $v \in \mathcal{N}^p - \mathcal{NR}^p_q - \{q\}$. In other words, $\mathcal{NR}^p_q$ includes $l-1$ number of the most distance gain contributing neighbors of $p$, during a potential reformation with $q$;
2. it then computes the net gain for the preliminary reform-set as:

$$d^p_q = \sum_{r \in \mathcal{NR}^p_q} d^p_{q,r} \tag{3}$$


**a** Before reformation **b** After reformation

**Fig. 3** Reformation

Finally, $p$ chooses $t \in \mathcal{N}^p$ as the participator of the reformation process where $d^p_t = \max_{q \in \mathcal{N}^p}\{d^p_q\}$. During the reformation, $p$ sends over a *REFORM_REQUEST* message to $t$ accompanied with the reform-set $\mathcal{NR}^p_t \cup \{p\}$. When peer $t$ receives the reformation request from $p$, it computes the reform-set for $p$ and then sends the set back to $p$ as a *REFORM_RESPONSE* message. Unlike the reform-set from $p$, the set, computed by $t$, consists of a list of $l$ peers from $\mathcal{N}^t$ which maximizes the net distance gain for $p$. After a successful reformation operation, both $p$ and $t$ perform a merge operation as discussed in Sub-section 3.3. Detailed control flows of a reformation initiator and a participator are given in Algorithm 1 and 2.

### 3.3 The merge operation

In the above discussion, $p$ performs a merge operation after it gets back the reform-set from $t$. In contrary, $t$ performs the operation after it decides about the reform-set to send out. Without lose of generality, let $p$ be a peer performing a merge operation. $\mathcal{N}_{send}$ and $\mathcal{N}_{recv}$ are the reform-sets that are sent and received, respectively. During the merge operation peer $p$ computes $\mathcal{N}^{p'}$ as follows:

$$\mathcal{N}^{p'} = (\mathcal{N}_p - \mathcal{N}_{send}) \cup \mathcal{N}_{recv} \tag{4}$$

where $\mathcal{N}^{p'}$ is the new P2P neighbor set of $p$. Note that it is certainly possible that $(\mathcal{N}^p - \mathcal{N}_{send}) \cap \mathcal{N}_{recv} \neq \emptyset$. In such cases, $|\mathcal{N}^{p'}| < |\mathcal{N}^p|$. Measures should be taken to carefully handle such cases. This issue is further elaborated in Sub-section 3.5.

### 3.4 Join and leave

When a mobile device wants to participate in the P2P network, at first it acquires an initial neighbor set from one or more of the *known* P2P peers by sending out a *SHARE_REQUEST* message. After receiving a *SHARE_REQUEST* message from $p$, a known peer responds in the similar way for a *REFORM_REQUEST* message. The known peer at first computes the *share-set* $\mathcal{NS}$, where $|\mathcal{NS}| = l$ and $\mathcal{NS}$ maximizes the net distance gain for $p$. Then, it packs the share-set with *SHARE_RESPONSE* message and sends it to $p$. The difference between response to *REFORM_REQUEST* and *SHARE_REQUEST* is that the known peer does not perform the merge operation while responding to a *SHARE_REQUEST*.

Realization of the known peers is possible in several ways. The access points of a WLAN, the Mobile Services Switching Centers (MSC) or the Base Service Centers (BSC) of a cellular network and the fixed mesh

routers of a mesh network are some of the possible locations where known peers can be implemented. A returning peer, i.e., a peer that was a member of the P2P network in a near past and is going to join the P2P network again, may decide to use its previous neighbors as known peers. However, the time duration, the peer was absent from the P2P network, would have some impact on this decision. In a critical case where no known peer is available, a joining mobile device can initiate an expanded ring query to discover the nearby peers, treat the discovered peers as known peers and ask them to provide some elements for the initial neighbor set.

On the other hand, a peer may not be a neighbor of its own neighbor (refer to Sub-section 3.2). As there is no direct and simplified way to find out the peers who include the departing peer as their neighbors, we consider that a peer can leave the network at any time without providing any explicit notification. A peer eventually discovers unavailable peers when it initiates a P2P control message exchange procedure (i.e., a ping, reformation, search query, etc.) with them. In Section 4, we will modify this behavior for better performance.

### 3.5 Number of P2P neighbors

In our proposed platform, we put an upper and a lower bounds on the size of the P2P neighbor set that a peer can have. Those bounds are defined as $N_{max}$ and $N_{min}$, respectively and must satisfy the following condition.

$$N_{max} \geqslant N_{min} > l \tag{5}$$

There are some situations when the neighbor set size grows beyond the $N_{max}$ threshold (for example, when a joining peer gathers peers from several known peers for its initial neighbor set). In those cases, the peer will keep $N_{max}$ number of the nearest peers and discard the rest. Similarly, there are some scenarios where a neighbor list shrinks below the $N_{min}$ threshold (for example, when a neighboring peer fails to respond to a P2P control message). Therefore, the peer requests for a neighbor list either from one of the available neighbors or from some widely known repository, following the same procedure of a joining peer.

The upper bound $N_{max}$ puts a limit on the worst case computational and space complexity for a peer (Sub-section 3.6). The lower bound $N_{min}$ provides robustness to IPPS. By tuning those parameters, the connectivity of the network can be controlled. The gap between $N_{max}$ and $N_{min}$, i.e., $(N_{max} - N_{min})$, allows the platform different levels of fault tolerance. The larger the gap, the more a peer tolerates reduction of the size of the neighbor set, i.e., failure of neighbors. PROOFS can be mapped into a special scenario where $N_{max}$ and $N_{min}$ are equal. However, this makes PROOFS unfavorable for wireless mobile networks which suffer from temporal disconnections or for P2P networks which allow dynamic join and leave of participating peers. The reason is that to maintain a specific number of neighbors, PROOFS suffers from a huge number of initialization operation at detection of each unavailable neighbor.

**Algorithm 1**: Control flow of a reformation initiating peer

```
while true do
    Compute the participating peer
    Let, t be the participating peer
    Let, N_send be the reform-set
    Send a REFORM_REQUEST to t with the
    reform-set N_send
    if t responds before timeout then
        Let, N_send be the received reform-set
        in REFORM_RESPONSE
        N ← (N − N_send) ∪ N_recv
        if |N| < N_min then
            call AddNeighbor()
        else
            Shrink N to size min(|N|, N_max)
        end
        break
    else
        N ← N − {t}
        if |N| < N_min then
            call AddNeighbor()
        end
    end
end
```

**Algorithm 2**: Control flow of a reformation participating peer

```
Let, N_recv be the reform-set received from p
Compute the reform-set N_send to send to p
Send back a REFORM_RESPONSE to p
with reform-set N_send
N ← (N − N_send) ∪ N_recv
if |N| < N_min then
    call AddNeighbor()
end
```

**Procedure** `AddNeighbor`

```
repeat
    Send a SHARE_REQUEST to a known repository
    Let, N_recv be the set received in SHARE_RESPONSE
    N ← N ∪ N_recv
until |N| < N_min
Shrink N to size min(|N|, N_max)
```

### 3.6 Computational complexity

The computational complexity of reformation is the complexity faced by the reformation initiating peer. This is due to the fact that the initiating peer incurs

more computational complexity than the responding or participating peer. The following is an analysis of the complexity with simple data structures and straight forward algorithms:

1. The complexity to find the net distance gain for a specific neighbor is $\Theta(|\mathcal{N}|+(l-1))=\Theta(|\mathcal{N}|+l)=\Theta(|\mathcal{N}|)$;
2. For all neighbors, the complexity turns out to be $\Theta(|\mathcal{N}|^2)$;
3. By tracking properly during the previous computations, the neighbor with maximum net gain can be found in $\Theta(1)$ time.

Therefore, the total complexity becomes $\Theta(|\mathcal{N}|^2)$. The worst case scenario arises when $|\mathcal{N}| = N_{max}$ and then the computational complexity becomes $\Theta(N_{max}^2)$. A peer faces the worst case memory requirement when the neighbor list grows beyond $N_{max}$ and this requirement can be formally expressed as $\Theta(N_{max}+l)$. Note that, $N_{max}$ and $l$ are constants for a specific network and are small positive integers.

### 3.7 Simplified analytical bound

Denote the nodes in the network as $v_1, v_2, \ldots, v_{N_{total}}$. Distance between nodes $v_i$ and $v_j$ can be defined as $D_{ij} = |dist(v_i, v_j)|$. Let $E[n_{ij}]$ be the expected number of link level hops between $v_i$ and $v_j$. From [45], it can be shown that,

$$\frac{D_{ij}/R-1}{E[\zeta(D_{ij}/R)]}+1 \leqslant E[n_{ij}] \leqslant \frac{D_{ij}/R}{E[\zeta(1)]}+1 \tag{6}$$

where $R$ is the radio range of a node. $E[\zeta(D)]$ is the expected one hop advancement towards the destination where the distance between the current and the destination nodes is $D$ (expressed in unit radio range). The average link level hop count can then be defined as,

$$\frac{1}{N_{total}(N_{total}-1)}\sum_{i=1}^{N_{total}}\sum_{\substack{j=1\\ j\neq i}}^{N_{total}} E[n_{ij}] \tag{7}$$

### 3.8 Performance evaluation

We develop an event driven simulation tool to evaluate the performance of our proposed platform. In the simulation, we consider a rectangular area of size $175 \times 175$ *square units*, where 5000 mobile nodes are randomly distributed according to a Poisson process. Radio range of a mobile node is considered to be 5 *units*. For all the simulations, the random waypoint mobility model with zero pause (i.e., the stressed mobility model) is used as the mobility pattern of the mobile nodes. The mean speed of the nodes is 0.032 *unit distance*/*unit time* (if the given radio range is considered to be 250 *meters* and 1 *unit time* to be 1 *sec*, this speed is equivalent to 5.7 $km/hr$). Each of the simulation runs for 4000 *unit time*. The network is given a 2000 *unit time* of warm-up period to reach an equilibrium state. After that we collect different status from the network at an interval of 10 *unit time* and finally compute the average. Other parameters of the simulations are given in Table 1. All the results presented are the average of the ten readings (i.e. with minimum standard deviation) out of twenty simulation runs. Unless stated otherwise, we use the above measurements for all the simulations.

To the best of our knowledge, PROOFS [35] is the most relevant among the works related to this research. Therefore, wherever possible, we compare IPPS with PROOFS. Figures 4a and 4b show the average number of link level hops per one P2P hop using the PROOFS and IPPS, respectively. The figures also show the theoretical upper bound on the average number of link level hops, considering that each node has global view of the entire network and no existing node either leaves the P2P network nor a new node join in. In case of PROOFS, due to the randomness of the network, the theoretical upper bound is fairly followed. On the other hand, in case of IPPS, a node does not have the global view and it may not choose the optimal neighbors with the lowest distance between them. As can be seen in Fig. 4b, IPPS performs slightly poorer than the optimal upper bound. As the percentage of mobile nodes participated in the P2P network increases, the number of link level hops per one P2P hop decreases. In fact, as the participation level increases, the chance to find a P2P neighbor at a nearer location also increases. However, if a network uses the PROOFS system (which is random in nature), this metric remains approximately the same, irrespective of different levels of participation. In this case, as the neighbors of a peer are uniformly distributed all over the network, the average link level hop count is not affected at all by the participation level. Actually, the simulation results presented in [40] show that only in an ideal situation (which is a perfect random system with no network dynamics), PROOFS

**Table 1** Simulation parameters

| Parameter Name | Value |
|---|---|
| $N_{max}$ | 30 |
| $N_{min}$ | 20 |
| $l$ | 5 |
| Participation level | 30%∼ 80% |

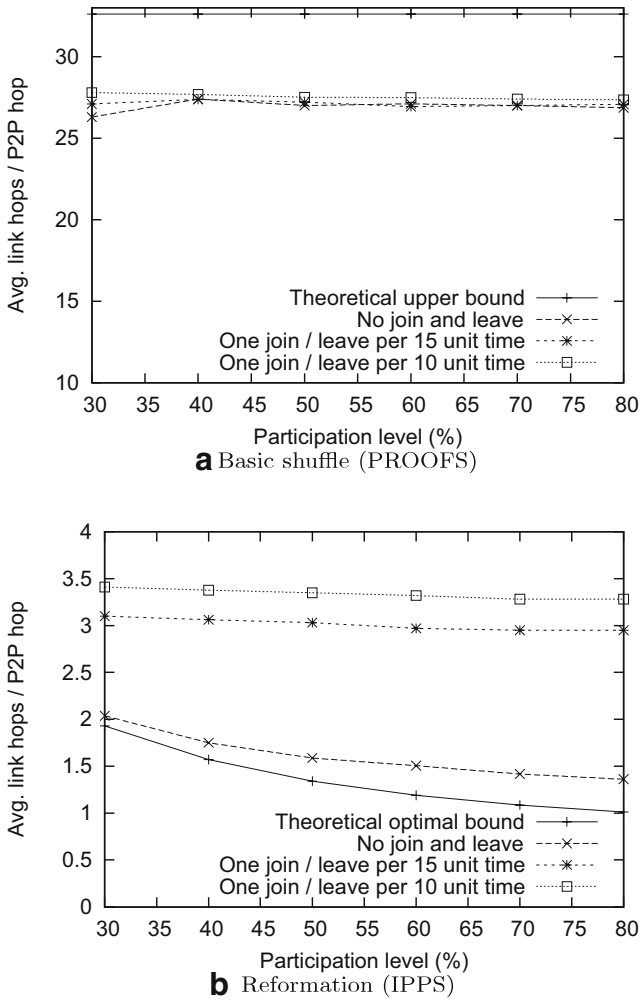



**Fig. 4** Number of link level hops per P2P hop

or similar systems can achieve the best performance where the average length of a single P2P hop is equivalent to the average path length of the whole network. Comparing Figs. 4a and 4b, link level hops per P2P level hop is significantly lower in our proposed platform. This indicates that IPPS reduces the bandwidth requirement and energy consumption to transmit P2P messages.

## 4 Performance enhancement

In this section, we further improve the performance of IPPS by modifying the way to choose the participating peer, then revising the merge process and reforming the way a peer leaves the network.

### 4.1 Modified reformation

Based on the discussion of Sub-section 3.2, we conclude that not all reformations result in a positive distance gain. The reason is as follows. With time, a peer eventually approaches to the optimal (i.e., the minimum) total distance from its neighbors. When a peer achieves this, the subsequent reformation always increases the total distance, i.e., the distance gain becomes negative. However, the reformation can not be stopped, because reformation is the process to keep the P2P network live. Therefore, we choose to use the reformation cycles, in which reformation does not help in reducing total distance, to perform some other useful tasks. We denote such cycles as *idle cycles* and have the following claim.

*Claim:* A neighbor for long time is a possible source of incorrect information.

As time pass by, properties of a peer may change. For instance, a peer may decide to leave the network or may move to a new location. To keep the platform simple, a peer communicates with its neighbor on demand basis only. As a result, an old neighbor without any communication for long time is a possible source of incorrect information. We improve the reformation process by refreshing the relation of a peer with such old neighbor during idle cycles.

We define a P2P *edge* from peer $p$ to $q$, if $q$ is a P2P neighbor of $p$. An edge in our platform is unidirectional, because a peer may not be the neighbor of its own neighbor. As a modification of the basic reformation, we introduce the concept of a clock associated with each P2P edge. Similar concept has been adopted in different areas of digital systems, including P2P networks. Operating systems use clocks to assist the page replacement module [37]. *CYCLON* [40], another gossip based P2P network, applies clocks to achieve load balancing. Our goal is to improve the reformation process.

Each P2P edge in our platform is equipped with a clock and is maintained by the peer from which the edge goes out. The clock counts zero at the beginning of the life time of an edge, i.e., when an edge is created. A new edge is created in two ways: (1) when peer $p$ engages in a reformation with participator peer $q$, an edge from $q$ to $p$ is created, (2) when $p$ acquires a set of peers with *SHARE_RESPONSE* messages, new edges are created from $p$ to a subset of the acquired peers, which are going to be the neighbors of $p$.

With each reformation operation at $p$, clocks of all the outgoing edges from $p$ count one more than the previous values. When $p$ forwards $r$ within the reform-set to $q$ (see Fig. 3), $p$ also forwards the clock associated with the $p$ to $r$ edge, and the new $q$ to $r$ edge continues to maintain the same clock. As a result, a clock of an edge indicates the number of reformation cycles the edge has encountered starting from its birth. Therefore, an edge with higher number of clock ticks (i.e., the

edge is around for longer time) indicates existence of possible incorrect information about the peer which it is pointing to. As a result, edges with higher tick counts are not desirable.

Figure 5 shows the scenario where a reformation introduces an edge with zero tick by replacing an edge with high tick count. In the figure, $T_q^p$ gives the tick count for the edge from $p$ to $q$. Thus, reformation operation can also be used to get rid of edges with higher tick count. To choose a participating peer, for each $q \in \mathcal{N}^p$, $p$ at first computes the net distance gain for the preliminary reform-set following (3). Then, the *benefit* of reformation with $q$ is computed as follows.

$$benefit_q^p = \exp\left(\frac{d_q^p}{RR} - \Delta\right) + \exp\left(\frac{T_q^p}{MAXTICK}\right) \quad (8)$$

where $RR$ is the radio range of a peer. The quantity $\frac{d_q^p}{RR}$ gives a measurement of expected gain in link level hops from a reformation between $p$ and $q$. The parameters $\Delta$ and $MAXTICK$ are system dependent parameters and provide a way to assign priority to distance gain or to clock ticks of an edge while choosing the participating peer. Values for these parameters are chosen based on network density, frequency of joining/leaving the P2P network and the way location service is provided. The term *benefit* combines the preference of distance gain and higher tick count into one metric. Finally, $p$ chooses $t \in \mathcal{N}^p$ as the participator of the reformation process where $benefit_t^p = \max_{q \in \mathcal{N}^p}\{benefit_q^p\}$. Except the way a peer chooses a participator, the rest of reformation process remains the same.

### 4.2 Revised merge

In the second step, we modify the semantics of the merge operation, presented in Sub-section 3.3. After receiving a reform-set at peer $q$, if $(\exists r)_S r \in \mathcal{N}_{recv} \cap \mathcal{N}^q$, the clock tick is set to $T_r^{q'} = \min(T_r^p, T_r^q)$. Here, $\mathcal{N}_{recv}$ is the received reform-set and $T_r^p$ and $T_r^q$ are the clock ticks of $p$ to $r$ and $q$ to $r$ edges, respectively. Figure 6 gives a visual clarification of the concept. The idea is to throw away an edge with higher clock tick, where there exists another parallel edge to the same peer with lower tick count.

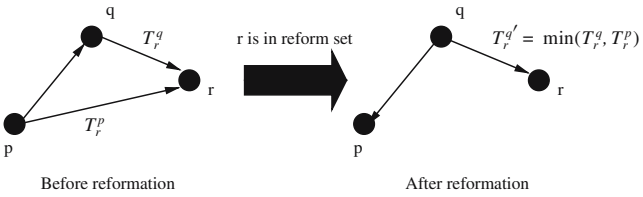



**Fig. 6** Improved merge after a reformation between $p$ and $q$

The gain of the above two modifications is threefold. Firstly, when the distance gain becomes negligible or negative, a peer tries to get rid of edges with high ticks rather than concentrating on distance gain alone, i.e., better use of reformation cycles. The second gain is the availability of updated information about the neighboring peers. Figure 7 shows the simulation results on number of *dangling edges* in the system. Dangling edges are edges which point to mobile nodes that are not members of the P2P system. If the rate of joining/leaving the P2P network increases, the number of dangling pointer increases faster with the basic IPPS. With the enhanced IPPS, the number of dangling edges reduces more than 50% as compared to the basic one. Note that this fault tolerant aspect is achieved without any extra communication cost. The third gain is better load balancing [40]. Since all P2P messages traverse using P2P neighbor information, on an average a peer needs to response to messages proportional to the number of peers from which it has incoming edges. Hence, to balance the load of the peers, in an ideal case, all the peers should have equal number of incoming edges. However, due to network dynamics and unavailability of instant global network status snapshot, it is difficult to achieve the

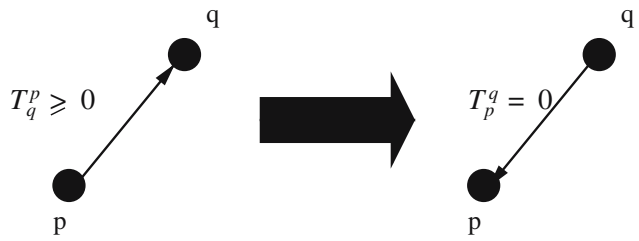



**Fig. 5** Towards improved reformation

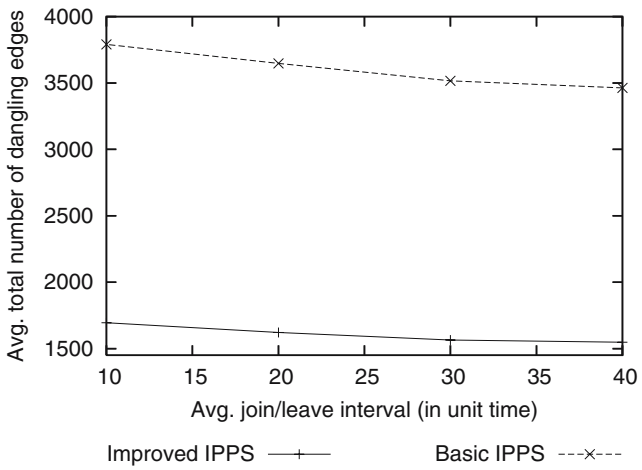



**Fig. 7** Effects of join/leave interval on dangling edges

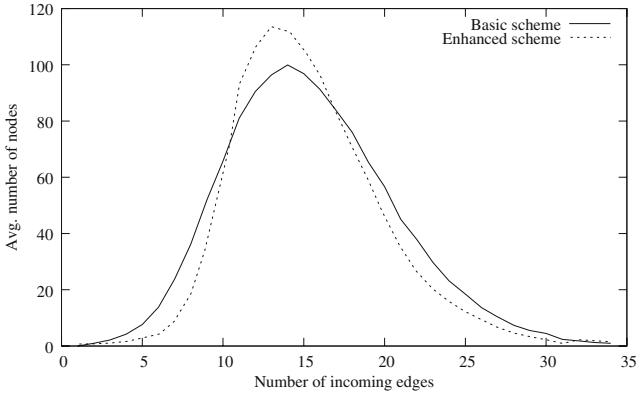



**Fig. 8** Incoming edge distribution

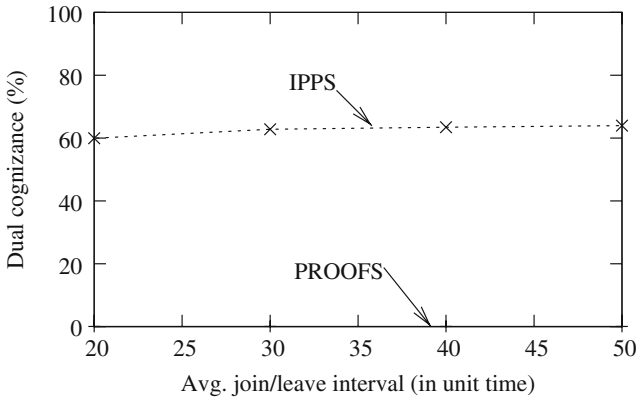



**Fig. 9** Effects of join/leave interval on dual cognizance

ideal situation. Figure 8 shows the distribution of nodes based on incoming edges for a network with 2,500 nodes. Here, 60% of the nodes participate to the P2P network and values of $N_{max}$ and $N_{min}$ are 20 and 15, respectively. In the figure, the basic IPPS produces distribution with lower variance than the enhanced one. These improvements are achieved without any increase in asymptotic time and space complexities.

### 4.3 Reformed leave process

*Claim:* In IPPS, the chance that a peer is a neighbor of its own peer is higher than that in the random networks.

The reformation establishes neighborhood relation among geographically close peers. At each reformation, a peer modifies the neighbor set with the peers that are closer than those of the previous set and the neighbors of that peer do the same. So, if $p$ finds $q$ to be at a closer location, it is likely that $q$ also finds $p$ the same and includes each other in their neighbor set. On the other hand, this situation is extremely unlikely in a random network, i.e., PROOFS. We define the property of a peer being the neighbor of its own peer as *dual cognizance*. Figure 9 shows the percentage of peers satisfying the dual cognizance property in PROOFS and IPPS schemes. In both the cases, we consider a network of 2500 nodes with a participation level of 60%. In case of PROOFS, each node maintains a neighbor set of size 25 with no join and leave. During the computation, we count a peer $n$-times if it has $n$ neighbors. Note that, in a perfect random PROOFS network, the concerned metric can be analytically defined as $(\frac{n}{N_{total}-1})^2$, where $N_{total}$ is the total number of peers. However, for an optimal IPPS (where each peer has a global view of the entire network), this metric is 1.

Based on this observation, we modify the way a peer leaves the platform. While leaving the network, a peer, $p$, may decide to notify each neighbor $q \in \mathcal{N}^p$ about its decision by sending an asynchronous *PURGE_REQUEST* message. On receiving a *PURGE_REQUEST* message from $p$, a peer $q$ searches for $p$ in its neighbor set and if $p$ is there, it is excluded. Formally, if $p \in \mathcal{N}^q$, the neighbor set of $q$ is updated with $\mathcal{N}^q - \{p\}$. As shown in Fig. 9, a good number of the neighbors include the leaving peer in their neighbor sets. So, the improved leave process keeps the network updated with information about the peers that are not member of the network. Note that, the reformed leave process is considered to be the preferred rather than a mandatory way of leaving the network.

## 5 Performance metrics

In this section, we discuss different performance metrics of our proposed platform. We consider the value of $\Delta$ and $MAXTICK$ to be 2 and $1.5 \times N_{max}$, respectively, and a departing peer informs its neighbors about its decision to leave the P2P networks with a probability of 0.5. Suppose that the underlying routing protocol can deliver a message between two P2P neighbors using the shortest path. This does not mean that the multi-hop P2P shortest path between peers $s$ and $t$ will also be the link level shortest path, as a P2P message is always propagated using the P2P neighbor information, not using the link level neighbor information (Fig. 2). In the best scenario, those two measurements can be the same and in the worst scenario a multi-hop P2P shortest path can be several times the network diameter. The measurements shown in Fig. 10 is the *stretch factor* of using P2P networks. Stretch factor is defined as the ratio of length of the shortest paths while using the P2P network and the link level network. A lower stretch factor is desired when control messages are flooded throughout or part of the P2P network. Searching the

network for an object is an example of such flooding. It should be noted that a peer hardly needs to communicate with non-neighbor peers directly. In case that such a communication is necessary (for example, communicating with a known peer while joining the P2P network or responding to a query request), the participating peers can always avoid the P2P network and communicate directly using the underlying network services. In Fig. 2, a search query may propagate following the path $s \rightarrow u \rightarrow v \rightarrow t$ and if the object is available at $t$, it may respond back directly to $s$, entirely bypassing the P2P network.

It has already been proved that given a connected network, no shuffle operation can make the network disconnected [35]. However, it is possible that the P2P network becomes disconnected as peers join and leave the P2P network. Mobility may further deteriorate the scenario, when the underlying network becomes physically disconnected as mobile nodes are unreachable from one another using radio links. During the simulation, the connectivity of the P2P network is computed. If $p$ is a neighbor of $q$, we consider that $q$ *knows* about $p$ and vice versa, and are connected in both way. Our simulation results fairly support the previous claims made in [35]. We have found that for almost all the cases more than 90% of the peers remain connected, given that they are also connected in their radio network. However, we are more interested about the worst case scenario, i.e., the minimum connectivity in the P2P network. Though we compute the network statistics every 10 *unit time* interval after the warm-up period, those results are reasonable approximation of the actual one, as they are computed from a very large number of samples. Finally, Fig. 11 shows the minimum connectivity of the network for different join/leave intervals. The numbers of peers in the largest connected peer graphs are computed and presented after normalizing in 1. As expected, the minimum connectivity decreases with decrement of participation level as well as with the frequency of joining/leaving the P2P network. It can seen that the worst case connectivity is higher than 70%. Hence, our proposed platform is robust.

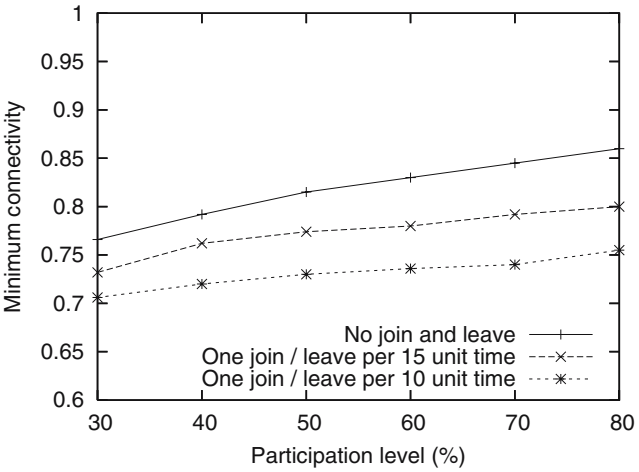



**Fig. 11** Minimum connectivity among peers for different join / leave intervals

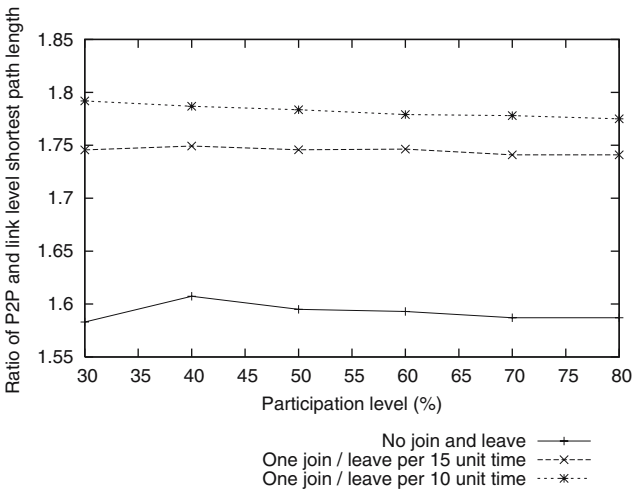
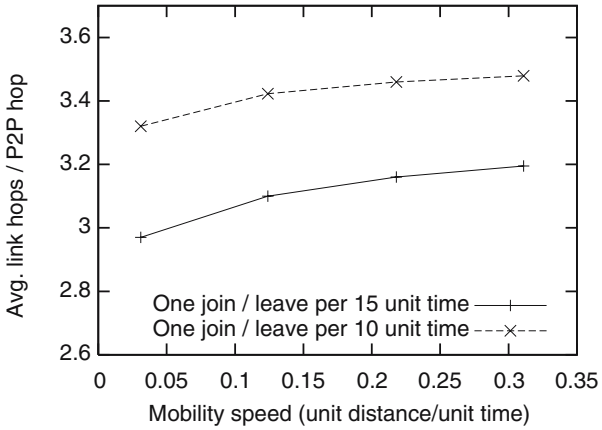


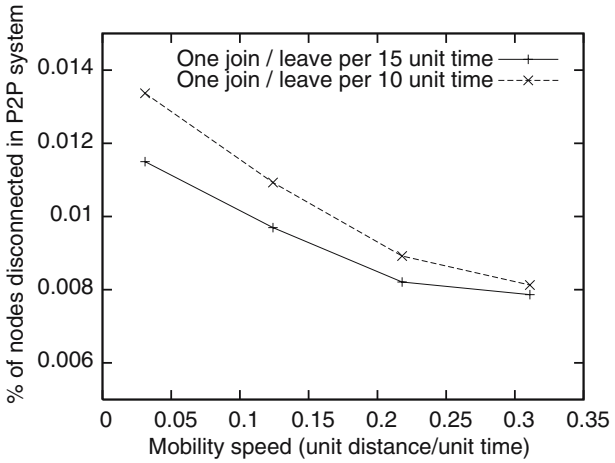

**Fig. 10** Ratio of hops in link level shortest path and P2P shortest path

Figure 12 shows the effect of mobility on the proposed platform. We simulate the network with four different mobility speeds: 0.032, 0.124, 0.218 and 0.32 *unit distance*/*unit time*, which are equivalent to the range from 5.7 to 57 $km/hr$. Figure 12a shows the effect on average number of link level hops per one P2P hop. For both the join/leave rates, the target measurement increases with the speed of the mobile nodes. However, in both cases, the increment becomes less steeper with the higher speed. As discussed before, while computing the average link level hops per P2P hop, we consider only those peers which are reachable from each other through the P2P network. Figure 12b shows that only a very small percentage of peers are disconnected in the P2P network where there exist link level paths in between them. However, with increased mobility, as the link level dis-connectivity increases and the dis-connectivity entirely at the P2P level (i.e., connected in the link level network, but disconnected in the P2P network) drops, they fairly equalize the effect of each other. In fact, in our simulation, the minimum (Fig. 11) and 90% connectivity suffer insignificantly due to the mobility.

The use of lower and upper bounds on the number of neighboring peers (i.e., $N_{min}$ and $N_{max}$, respectively) makes the platform more flexible. However, increasing those limits arbitrarily also increases the computational




**a** Average link level hops per P2P hop for different mobility




**b** % of disconnected nodes for different mobility

**Fig. 12** Effect of mobility on average length of P2P hops and number of disconnected peers

cost (Section 3). Increasing $N_{max}$ in an undetermined way is simply a waste of computing power. Figure 13a lists the minimum connectivity for different values of $N_{max}$. It can be seen that there exists no significant improvement from $N_{max} = 60$ to $N_{max} = 70$. Similarly, having a larger gap between $N_{max}$ and $N_{min}$ makes the platform robust and fault-tolerant to failed peers, but having a too small $N_{min}$ may also result in disconnected peers. Figure 13b shows the effect of $N_{min}$ on the minimum connectivity.

| $N_{\max}$ | 30 | 40 | 50 | 60 | 70 |
|---|---|---|---|---|---|
| Min. Connectivity | 0.736 | 0.752 | 0.761 | 0.764 | 0.765 |

**a** For different $N_{\max}$, where $N_{\min} = 25$

| $N_{\min}$ | 25 | 35 | 45 | 55 |
|---|---|---|---|---|
| Min. Connectivity | 0.764 | 0.769 | 0.772 | 0.774 |

**b** For different $N_{\min}$, where $N_{\max} = 60$

**Fig. 13** Effect of $N_{max}$ and $N_{min}$ on minimum connectivity (normalized in 1), where participation level is 40%

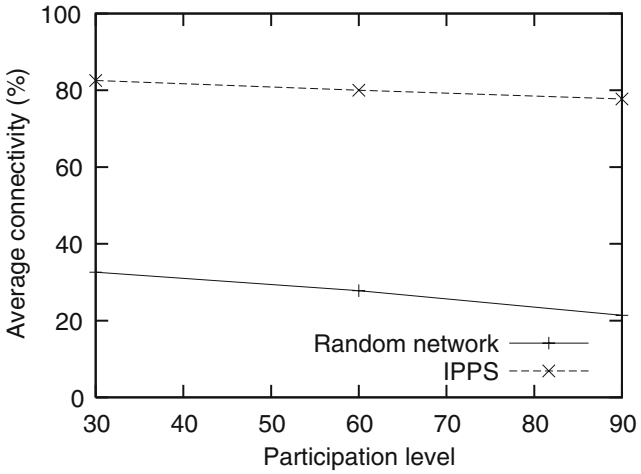



**Fig. 14** Average connectivity while considering the direction of edges

For the minimum connectivity metric, we considered each edge to be undirected. The force behind the choice is that at some point of time, each edge will change its direction due to a shuffle/reformation [35]. So, the minimum connectivity metric provides a notion of long term robustness. It does not convey the idea of instantaneous robustness. Consider a search protocol which can not wait for an indefinite time to forward the query (for an unpopular object) throughout the network. Consequently, the more interesting metric would be the average connectivity while considering the edges to be directed. Figure 14 compares PROOFS and IPPS in this regards. It can be seen that IPPS provides significantly better measurements than that of PROOFS.

## 6 Follow-up discussion

In this section, we first discuss the possible search and replication technique for the platform in Subsection 6.1. We then explain the available interface of the platform in Subsection 6.2. Finally, some example applications for this platform are given in Subsection 6.3.

### 6.1 Searching the network

Since, there is no logical difference between PROOFS and IPPS, the search and replication technique proposed in PROOFS [35] are equally implementable in IPPS. Other generic unstructured P2P network search techniques, such as expanding ring, random walk, $k$-way search, etc. can also be deployed on top of this platform [22]. Similar claim can be made for replication techniques in [22]. However, it should be noted that search in P2P network is vastly application dependent,

and therefore the decision to choose the proper search and replication strategy should be made by the application designer/developer.

### 6.2 Interface for the applications

IPPS has a simple interface for the application. The interface provides functionalities to join and leave the P2P network, and to communicate with the peers. The interface primitives are briefly described below.

- **Join** is used to join with the given P2P network. This primitive allocates and initializes the required data structures and follows the description of Subsection 3.4.
- **Leave** primitive disconnects the executing peer from the given P2P network. Opposite to *Join* primitive, *Leave* frees up all allocated data structures.
- A peer can synchronously or asynchronously send a message to a given set of peers using the **Send** primitive. A call to the asynchronous **send** returns a set of *future* [33] objects - one object for each of the destination peers.
- The counter part of the send primitive is **Receive** which can be used to receive a message from a peer with a given identification. Note that the identification may also mean any peer. Like asynchronous send, asynchronous *Receive* also gives a future object.
- The **Poll** primitive works on a set of future objects and checks the completion of the corresponding asynchronous send or receive operations.
- The last primitive, **GetNeighbors**, returns the set of neighbors for a given P2P network.

Though an implementation of IPPS may have a wider set of primitives, it can be shown that the above primitives are sufficient to develop a complex communication application [38].

### 6.3 Potential applications

IPPS is an inexpensive P2P platform. As a result, wireless applications, ranging from low to high cost, can be developed on top of it. For instance, a P2P application to handle flash crowd can easily be built on IPPS by following the application structure proposed in [35]. Similarly, a file sharing application [7] and a mobile application to collaborate Internet access [16] can be developed. IPPS is a suitable platform where a large number of packets are distributed among a group of participators. Such applications are streaming multicast [17], audio-video conference, etc. In [1], IPPS is considered as a possible P2P platform for collaborative caching. Currently, we are investigating the use of IPPS in information dissemination in intelligent vehicular transport system [13].

## 7 Conclusion

In this paper, we have proposed ***I**nexpensive **P**eer-to-**P**eer **S**ubsystem (IPPS)*, an unstructured P2P platform for wireless mobile networks. We have used an inexpensive gossip protocol to maintain the network. The platform considers distance between neighbors as a biasing factor and is economical in terms of communication, computation and memory requirement. The platform is also robust, fault tolerant and flexible. Our current work includes investigation for a integrated technique to search for both popular and unpopular objects in the IPPS network, and to develop several application domains where the proposed platform can be deployed.

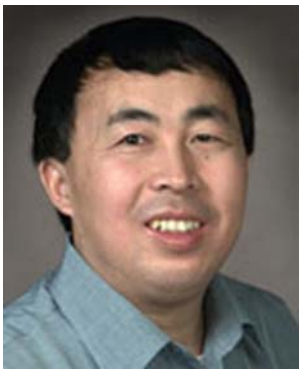
**Acknowledgements** We thank the reviewers for their interesting and helpful comments and tips. This research has been financially supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada.

## References

1. Akon MM, Singh A, Shen X (2007) Low cost peer-to-peer collaborative caching for clusters. In: IEEE consumer communications and networking conference, pp 550–555
2. Bellavista P, Corradi A (2006) The Handbook of Mobile Middleware. CRC Press, London, UK
3. Castro M, Druschel P, Kermarrec AM, Rowstron A (2002) SCRIBE: a large-scale and decentralised application-level multicast infrastructure. IEEE J Sel Areas in Commun (JSAC) 20(8):100–110
4. Chakraborty D, Joshi A, Yesha Y, Finin T (2006) Toward distributed service discovery in pervasive computing environments. IEEE Trans Mob Comput 5(2):97–112
5. Clarke I, Miller S, Hong T, Sandberg O, Wiley B (2002) Protecting free expression online with freenet. IEEE Internet Comput 6(1):40–49
6. Cleary DC, Parke DC (2005) "Finding Yourself" building location services in a peer-to-peer wireless world. In: EUROCON 2005, pp 60–63
7. Ding G, Bhargava B (2004) Peer-to-peer file-sharing over mobile ad hoc networks. In: IEEE conference on pervasive computing and communications workshops. FL, USA, pp 104–108
8. Druschel P, Rowstron A (2001) PAST: a large-scale, persistent peer-to-peer storage utility. In: HotOS VIII. Germany, pp 75–80
9. Freedman MJ, Sit E, Cates J, Morris R (2002) Introducing tarzan, a peer-to-peer anonymizing network layer. In: 1st intl. workshop on peer-to-peer systems, pp 121–129
10. Gatani L, Re GL, Gaglio S (2005) An adaptive routing protocol for ad hoc peer-to-peer networks. In: Sixth IEEE

international symposium on a world of wireless mobile and multimedia networks, pp 44–50
11. Ghandeharizadeh S, Krishnamachari B, Song S (2004) Placement of continuous media in wireless peer-to-peer networks. IEEE Trans Multimed 6(2):335–342
12. Gnutella (2005) Gnutella web-site. http://www.gnutella.com/
13. Goel S, Imielinski T, Ozbay K (2004) Ascertaining the viability of wifi based vehicle-to-vehicle network for traffic information dissemination. In: 7th annual IEEE intelligent transportation systems conference, pp 1086–1091
14. IEEE 802.11 (2005) IEEE 802.11 web-site. http://www.ieee802.org/11/
15. Irestig M, Hallberg N, Eriksson H, Timpka T (2005) Peer-to-peer computing in health-promoting voluntary organizations: A system design analysis. J Med Syst 29(5):425–440
16. Kang SS, Mutka MW (2004) Efficient mobile access to internet data via a wireless peer-to-peer network. In: Second IEEE international conference on pervasive computing and communications (PerCom'04), pp 197–206
17. Karlsson J, Eriksson J, Li H (2006) P2P video multicast for wireless mobile clients. In: Fourth international conference on mobile systems, applications, and services. Uppsala, Sweden, pp 19–22
18. Lao L, Cui JH (2006) Reducing multicast traffic load for cellular networks using ad hoc networks. IEEE Trans Veh Technol 55(3):882–830
19. Lin YD, Hsu YC (2000) Multihop cellular: A new architecture for wireless communications. In: IEEE Infocom, pp 1273–1282
20. Liu J (2005) Peer-Tree: a peer-to-peer message forwarding structure for relaying messages in mobile applications. In: Second international workshop on hot topics in peer-to-peer systems, pp 87–94
21. Luo H, Ramjee R, Sinha P, Li L, Lu S (2003) UCAN: a unified cellular and ad-hoc network architecture. In: Proceedings of ACM MobiCom, pp 353–367
22. Lv Q, Cao P, Cohen E, Li K, Shenker S (2002) Search and replication in unstructured peer-to-peer networks. In: ICS '02: Proceedings of the 16th international conference on Supercomputing. NY, USA, pp 84–95
23. Mavromoustakis CX, Karatza HD (2005) Segmented file sharing with recursive epidemic placement policy for reliability in mobile peer-to-peer devices. In: IEEE international symposium on modeling, analysis, and simulation of computer and telecommunication systems, pp 371–378
24. Akon MM, Naik S, Singh A, Shen XS (2006) A cross-layered peer-to-peer architecture for wireless mobile networks. In: IEEE international conference on multimedia & expo. Toronto, Canada, pp 813–816
25. Napster (2005) Napster web-site. http://www.napster.ca/
26. Networks E (2005) Ellacoya networks releases data on broadband subscriber and application usage in europe. http://www.ellacoya.com/news/pdf/2005/EllacoyaEuropeData.pdf
27. Passarella A, Delmastro F, Conti M (2006) XScribe: a stateless, cross-layer approach to P2P multicast in multi-hop ad hoc networks. In: International conference on mobile computing and networking. Los Angeles, California, pp 6–11
28. Pias M, Crowcroft J, Wilbur S, Harris T, Bhatti S (2003) Lighthouses for scalable distributed location. In: Second international workshop on peer-to-peer systems, pp 278–291
29. Ramaswamy L, Gedik B, Liu L (2005) A distributed approach to node clustering in decentralized peer-to-peer networks. IEEE Trans Parallel Distrib Syst 16(9):814–829
30. Ratnasamy S, Francis P, Handley M, Karp R, Shenker S (2001) A scalable content addressable network. In: ACM SIGCOMM
31. Rowstron A, Druschel P (2001) Pastry: scalable, distributed object location and routing for large-scale peer-to-peer systems. In: IFIP/ACM international conference on distributed systems platforms. Germany, pp 329–350
32. Rowstron A, Kermarrec AM, Castro M, Druschel P (2001) SCRIBE: the design of a large-scale event notification infrastructure. In: Networked group communication, pp 30–43
33. Schmidt DC (1996) A family of design patterns for applications-level gateways. Theory and Practice of Object Systems 2(1):15–30
34. Sethom K, Afifi H (2005) A new service discovery architecture for sensor networks. In: Wireless telecommunications symposium, pp 190–196
35. Stavrou A, Rubenstein D, Sahu S (2004) A lightweight, robust P2P system to handle flash crowds. IEEE J Sel Areas in Commun 22(1):6–17
36. Stoica I, Morris R, Karger D, Kaashoek MF, Balakrishnan H (2001) Chord: a scalable peer-to-peer lookup service for internet applications. In: ACM SIGCOMM 2001. CA, USA, pp 149–160
37. Tanenbaum AS, Woodhull AS (1997) Operating Systems: Design and Implementation, 2nd edn. Prentice Hall, NJ
38. Tokuda H, Manning, EG (1983) An interprocess communication model for a distributed software testbed. In: The symposium on communications architectures and protocols. Austin, TX, pp 205–212
39. Tran DA, Hua KA, Do TT (2004) A peer-to-peer architecture for media streaming. IEEE J Sel Areas in Commun 22(1):121–133
40. Voulgaris S, Gavidia D, van Steen M (2005) CYCLON: inexpensive membership management for unstructured P2P overlays. J Netw Syst Manag 13(2):197–217
41. Wang AI, Norum MS (2006) Issues related to development of wireless peer-to-peer games in J2ME. In: International conference on internet and web applications and services, pp 115–115
42. Wolfand H, Wang M (2005) A framework with a peer fostering mechanism for mobile P2P game development. In: International conference on mobile business, pp 109–115
43. Xiang Z, Zhang Q, Zhu W, Zhang Z, Zhang YQ (2004) Peer-to-peer based multimedia distribution service. IEEE Trans Multimed 6(2):343–355
44. Zhao BY, Kubiatowicz JD, Joseph AD (2001) Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Tech. Rep. UCB/CSD-01-1141, U. C. Berkeley
45. Zorzi M, Rao RR (2003) Geographic random forwarding (GeRaf) for ad hoc and sensor networks: multihop performance. IEEE Trans Mob Computg 2(4):337–348
46. Zorzi M, Rao RR (2003) Multihop performance of geographic random multihop performance of geographic random. IEEE Trans Mob Comput 2(4):349–365

**Mursalin Akon** received his B.Sc.Engg. degree in 2001 from the Bangladesh University of Engineering and Technology (BUET), Bangladesh, and his M.Comp.Sc. degree in 2004 from the Concordia University, Canada. He is currently working towards his Ph.D. degree at the University of Waterloo, Canada. His current research interests include peer-to-peer computing and applications, network computing, and parallel and distributed computing.



**Xuemin Shen** received the B.Sc.(1982) degree from Dalian Maritime University (China) and the M.Sc. (1987) and Ph.D. degrees (1990) from Rutgers University, New Jersey (USA), all in electrical engineering. He is a Professor and the Associate Chair for Graduate Studies, Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research focuses on mobility and resource management in wireless/wired networks, wireless security, ad hoc and sensor networks, and peer-to-peer networking and applications. He is a co-author of three books, and has published more than 300 papers and book chapters in different areas of communications and networks, control and filtering. Dr. Shen serves as the Technical Program Committee Chair for IEEE Globecom'07, General Co-Chair for Chinacom'07 and QShine'06, the Founding Chair for IEEE Communications Society Technical Committee on P2P Communications and Networking. He also serves as the Editor-in-Chief for Peer-to-Peer Networking and Application; founding Area Editor for IEEE Transactions on Wireless Communications; Associate Editor for IEEE Transactions on Vehicular Technology; KICS/IEEE Journal of Communications and Networks, Computer Networks; ACM/Wireless Networks; and Wireless Communications and Mobile Computing (Wiley), etc. He has also served as Guest Editor for IEEE JSAC, IEEE Wireless Communications, and IEEE Communications Magazine. Dr. Shen received the Excellent Graduate Supervision Award in 2006, and the Outstanding Performance Award in 2004 from the University of Waterloo, the Premier's Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada, and the Distinguished Performance Award in 2002 from the Faculty of Engineering, University of Waterloo. Dr. Shen is a registered Professional Engineer of Ontario, Canada.

**Sagar Naik** received his BS, M. Tech., M. Math., and Ph.D. degrees from Sambalpur University (India), Indian Institute of Technology, University of Waterloo, and Concordia University, respectively. From June 1993 to July 1999 he was on the Faculty of Computer Science and Engineering at the University of Aizu, Japan, as an Assistant and Associate Professor. At present he is an Associate Professor in the Department of Electrical and Computer Engineering, University of Waterloo. His research interests include mobile communication and computing, distributed and network computing, multimedia synchronization, power-aware computing and communication.

**Ajit Singh** received the B.Sc. degree in electronics and communication engineering from the Bihar Institute of Technology (BIT), Sindri, India, in 1979 and the M.Sc. and Ph.D. degrees from the University of Alberta, Edmonton, AB, Canada, in 1986 and 1991, respectively, both in computing science. From 1980 to 1983, he worked at the R & D Department of Operations Research Group (the representative company for Sperry Univac Computers in India). From 1990 to 1992, he was involved with the design of telecommunication systems at Bell-Northern Research, Ottawa, ON, Canada. He is currently an Associate Professor at Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His research interests include network computing, software engineering, database systems, and artificial intelligence.

**Qian Zhang** received the B.S., M.S., and Ph.D. degrees from Wuhan University, Wuhan, China, in 1994, 1996, and 1999, respectively, all in computer science. In July 1999, she was with Microsoft Research, Asia, Beijing, China, where she was the Research Manager of the Wireless and Networking Group. In September 2005, she joined Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong, as an Associate Professor. She has published about 150 refereed papers in international leading journals and key conferences in the areas of wireless/Internet multimedia networking, wireless communications and networking, and overlay networking. She is the inventor of about 30 pending patents. Her current research interests are in the areas of wireless communications, IP networking, multimedia, P2P overlay, and wireless security. She also participated in many activities in the IETF ROHC (Robust Header Compression) WG group for TCP/IP header compression. Dr. Zhang is an Associate Editor for the IEEE Transactions on Wireless Communications, IEEE Transactions on Multimedia, IEEE Transactions on Vehicular Technologies, and Computer Communications. She also served as the Guest Editor for a Special Issue on Wireless Video in the IEEE Wireless Communication Magazine and is serving as a Guest Editor for a Special Issue on Cross Layer Optimized Wireless Multimedia Communication in the IEEE Journal on Selected Areas in Communications. She received the TR 100 (MIT Technology Review) World's Top Young Innovator Award. She also received the Best Asia Pacific (AP) Young Researcher Award from the IEEE Communication Society in 2004. She received the Best Paper Award from the Multimedia Technical Committee (MMTC) of IEEE Communication Society. She is the Chair of QoSIG of the Multimedia Communication Technical Committee of the IEEE Communications Society. She is also a member of the Visual Signal Processing and Communication Technical Committee and the Multimedia System and Application Technical Committee of the IEEE Circuits and Systems Society.