A Robust Algorithm for the Membership Management of Super-peer Overlay

Meirong Liu, Jiehan Zhou, Timo Koskela, Mika Ylianttila

Media Team Oulu research group, Information Processing Laboratory,
Department of Electrical and Information Engineering
University of Oulu, Finland
P.O.BOX 4500 FIN-90014
firstname.lastname@ee.oulu.fi

Abstract. Peer-to-Peer technologies have been widely applied for multimedia applications. The super-peer based approach provides an efficient way to run applications by exploring nodes' heterogeneity. In P2P live video streaming, even though the number of stable nodes is small, they have significant impact on the performance of the network. Thus, we present a super-peer-based overlay design, where stable nodes are assigned as super-peers that organize client nodes. A gossip-based super-peer selection algorithm (GSPS) is proposed to identify the stable nodes to be chosen as super-peers and to manage the client nodes (namely the membership management). The basic idea of the GSPS is: first, a set of super-peer candidates for a node is built based on the gossip, then the role of this node is identified and the corresponding operations are executed. Simulation results show that the GSPS is efficient in managing the super-peer overlay and robust to the failure of super-peers.

Keywords: Peer-to-Peer, super-peer, gossip, robustness.

1 Introduction

The peer-to-peer (P2P) paradigm provides an effective approach to construct large-scale systems with high robustness, mainly due to their inherent decentralization and redundant structures [1] [2] [3]. Moreover, the P2P systems are able to utilize the resources distributed in a large number of machines connected to the system through the Internet. Therefore, much work focuses on building the P2P overlay, such as [3] [4] [5] [6] [7] [8]. In these research studies, Random walk and Graph theory are utilized for the construction of the overlay. For the management of the overlay, the diameter, and the degree are two important qualities to consider.

The super-peer-based overlay (such as Gnutella [9] and Kazaa [10]) provides an effective way to run applications by utilizing the heterogeneity of the peer nodes. Specifically, the super-peer approach does not have the disadvantage of the traditional client-server model (the single point of failure), resulting in the increased robustness [11] [12] [13]. Moreover, the nodes with low capacity are shielded from the massive query traffic by the super-peers, which improves the scalability of the system.

Wang et al. [14] found that the number of stable nodes in P2P live video streaming is small, but these stable nodes make a significant contribution to the system performance. They presented a tiered overlay design for P2P live video streaming, where the stable nodes and the transient nodes are separated into two levels. Moreover, our previous work presented the P2P SCCM (Service-Oriented Community Coordinated Multimedia) paradigm [15], which leverages the roles of requestors, providers, and registry centre in the conventional SOA architecture as service peers for multimedia application. Those multimedia-intensive service peers are assumed to publish their services, access services provided by other peers through searching, and intermediately communicate with each other. The P2P SCCM uses super-peers to act as rendezvous peers that handle service indexing and identity mapping. Continuing the vision of the P2P SCCM, in this paper, we focus on how to manage the super-peer overlay, namely, how to select super-peers and rearrange the super-peer topology when some of the super-peers fail. To this end, we propose a gossip-based super-peer selection algorithm (GSPS) to identify stable nodes (the peers with high capacity) to act as super-peers and to manage client nodes. The capacity of a node is composed of the computational resource, the lifespan and the network connections [8] [14].

The basic idea of the GSPS is as follows. First, a set of super-peer candidates for a node is built based on the gossip communication, where this node exchanges information with its neighbors in order to select other nodes with higher capacity as the super-peers candidates. After that, if this node is in the list of the super-peer candidates, this node takes the role of a super-peer. Finally, the nodes execute the corresponding operations, joining a super-peer or recruiting a client node, according to their role. In the case of a disaster (i.e. the failure of a super-peer), the client nodes declare their state to be rebuilt and reconstruct their sets of super-peer candidates, selecting a new number of the required super-peers, and then join these super-peers. Simulations have been conducted and results show that the GSPS is efficient to manage the overlay, especially to select the super-peers. The GSPS is also robust to the failure of the super-peers. The terms node and peer are used interchangeably in the rest parts of the paper.

The remainder of this paper is organised as follows. In Section 2, the GSPS is presented to show how to select the powerful nodes to act as super-peers and how to manage the relation between the client nodes and a super node. How to handle the disaster of the super-peer failure is also considered. In Section 3, we show the evaluation results of the GSPS. In Section 4, we review the related work on the overlay management. The conclusion and future work are discussed in Section 5.

2 The proposed GSPS algorithm for membership management

In this section, the background of the communication method is first introduced, and then the GSPS algorithm is depicted. We assume that all the nodes are connected through an existing routed network. A node needs to store the identifiers of its neighbours in order to communicate with other nodes. Thus, each node can potentially communicate with other nodes directly or indirectly via other nodes. These

neighbourhood relations constitute the topology of the overlay. Both the neighbours of a node and the overlay topology can change dynamically.

2.1 Background of a gossip communication model

The gossip-based communication model in large-scale distributed systems has become a general paradigm for many important applications, which include information dissemination, aggregation, topology management and synchronization. We apply a gossip communication method called *Newscast* [16] to maintain a neighbor set of each peers in the network. The Newscast has been used for broadcast [17] and aggregation [17] [18] in several P2P protocols. In the Newscast, the state of a node is called a partial view, including a fixed-size set of descriptors of its neighbor peers. Two neighboring peers merge their partial views periodically to keep the freshest descriptors; thereby a new partial view is created. Peers always update their own, newly created descriptor into the partial view. By exchanging the partial view, old information is gradually replaced by new information. This approach enables the GSPS to rebuild neighbors of the nodes by excluding the crashed nodes.

2.2 Selecting super-peers and client peers

Building and maintaining a super-peer-based overlay topology are complex. On one side, the dynamic environment requires a robust and an efficient algorithm to self-organize and self-repair the super-peer overlay in the case of voluntary and unexpected events like joining, leaving and crashes [19] [20] [21]. On the other side, the nodes in the overlay are heterogeneous in their capacity [8]. We propose the GSPS to build and manage the super-peer overlay characterized by the minimum number of super-peers. Inspired by the method called VoRonoi Regions set up in the mobile network [22] and the clustering method [23], super-peers are first selected during the overlay construction.

We assume that the topology information such as the identifier, the capacity, the lifespan, the current role and the neighborhood of the participating nodes are disseminated through periodic gossip messaging. The notations used in the GSPS are given as follows for simplicity. Let n_i be a node in an N-node P2P network. Each super-peer maintains three sets: its neighbor sets, the client node set $S_c(n_i)$ and the set of the super-peer candidates. Each client node maintains the following information: its super-peer, its neighbors and the set of the super-peer candidates. Let SP(n_i) denote the super-peer of the node n_i. To distinguish the capacity of each node, let c_{ni} represent the abstracted capacity of a node n_i. The abstracted capacity c_{ni} denotes the number of the client nodes that the node n_i can manage according to its resource property, if n_i takes the role of a super-peer. We assume each node n_i knows its capacity parameter c_{ni}, which could be computed on-the-fly in the application implementation. Let $Ld(n_i)$ denote the current load of n_i. For each node, there are two states: idle and rebuilt, e.g. State(n_i)={idle, rebuilt}. The idle state denotes that a node n_i joins the overlay but does not suffer from the failure of its super-peer. The rebuilt state denotes a node n_i joins the overlay but suffers from the failure of its super-peer. Specifically, if a client node n_i suffers from its super-peer's failure, n_i declares its state

to be rebuilt. When the n_i with the rebuilt state is identified as a super-peer, the super-peer n_i possesses the state of rebuilt as well. The purpose of setting two states for the node n_i is to re-organize its super-peer candidates when its super-peer fails.

```
Input: A client node n_i
Operation:
if SP(n_i) is down and State(n_i) is idle then
    n_i changes its State(n_i) to be rebuilt;
    n_i removes its SP(n_i) from its set of super-peer candidates;
    n_i rebuilds its set of super-peer candidates.
end if
if State(n_i) is rebuilt then
      for each neighbor n_neig of n_i do
       n_neig deletes SP(n_i) from its set of super-peer candidates.
        if n_neig is a client node, SP(n_neig)≠null and is down then
          n_neig changes its state to be rebuilt and removes its super-peer from the set
          of super-peer candidates; n_neig rebuilds its set of super-peer candidates.
        end if
     end for
end if
if n_i has different super-peer candidates with its neighbors then
   n_i gets the super-peer candidates of each neighbor and compares their capabilities
   to get a new set of super-peer candidates with higher capacity CanSP ;
   n_i updates its super-peer candidates to be CanSP;
          for each neighbor n_neig of n_i do
                n_neig updates its set of super-peer candidates to be CanSP;
                 if n_neig is client and n_neig ∈ CanSP then
                    then n_neig   changes its role as a super-peer;
                 if n_neig is SP and n_neig ∉ CanSP then
                       n_neig changes its role as a client node;
                     if S_c(n_neig) ≠ null then
                       n_neig transfers its clients to the super-peers in CanSP.
                 end if
          end for
          if n_i ∈ CanSP then n_i changes its role as a super-peer.
else if n_i has the same super-peer candidates with its neighbors then
        n_i searches its neighbors to get one under-loaded super-peer;
          if n_i finds one super peer SP_j then n_i joins the peer group managed by SP_j.
          else
              n_i searches the super-peers of its neighbors that are under-loaded;
              if n_i finds one super-peer SP_j, then
                    n_i joins the peer group managed by SP_j.
              else if all the super-peers are full-loaded then
                    n_i declares itself as a super-peer.
          end else
end else
```

Fig. 1 The action of a client node in the GSPS

```
Input: A super node n_i and n_i is under-loaded
Operation:
if State(n_i) is rebuilt then
        for each neighbor n_neig of n_i
            if n_neig is a client node, SP(n_neig)≠null and is down then
                n_neig deletes SP(n_i) from list of super-peer candidates;
                n_neig changes its state to be rebuilt;
                n_neig rebuilds its set of super-peer candidates.
            end if
        end for
end if
if there is one node whose super-peer candidates are larger than n_i then
    n_i gets the set of super-peer candidates of each neighbor and compares their
  capacity to get a new set of super-peer candidates with higher capacity-CanSP;
    n_i updates its super-peer candidates to be CanSP.
      for each neighbor n_neig of n_i do
        n_neig updates its set of super-peer candidates to be CanSP;
        if n_neig is client and n_neig ∈ CanSP then
            n_neig changes its role as a super node.
        if n_neig is SP and n_neig ∉ CanSP then
            n_neig changes its role of n_neig as a client node;
            n_neig transfers its clients to the super-peers in the set CanSP.
        end if
      end for
      if n_i ∉ CanSP then
          n_i transfers its clients to the super-peers in the set CanSP;
          n_i changes its role to be a client node.
      end if
else  for each neighbor n_neig of n_i do
        if n_neig is client, SP(n_neig)=null and n_i is under-loaded then
            n_i adds n_neig as its client.
        if n_i is under-loaded then
            n_i searches its neighbours to get a client node n_neig whose
            super-peer is null and then adds n_neig as its client.
    end for
end else
```

Fig. 2 The action of a super-peer in the GSPS

Fig. 2 shows the action of a super-peer in the GSPS algorithm. It also includes two scenarios: the joining process and rebuild process. In the rebuild process, each neighbor of a super-peer removes the failed super-peer from its set of the super-peer candidates. To make the algorithm converge fast, a two-hop search method is utilized. That is, the super-peer first searches its neighbors and then searches the neighbors of its neighbors. The reason why we use the two-hop search is to prevent the worst case from happening that would increase the convergence time. Here, the worst case is as follows: all the neighbors of a peer have joined one service group, and these peer groups are full-loaded and cannot accept any more client nodes. In the worst case, the

peer cannot join any of the peer groups of its one-hop neighbors. Therefore, the two-hop search method is employed to help the peer to find a peer group of its neighbors for joining.

3 Evaluation of the GSPS algorithm

We have conducted experiments with PeerSim [24], which is a round driven P2P simulator, to validate the operation of the GSPS protocol. Here, a simulation round means all the nodes have finished performing the protocols deployed on the nodes one time. We focus on three main questions: (i) the impact of the GSPS's parameters on the performance of the GSPS protocol; (ii) the robustness of the protocol in the case of the super-peer failure; and (iii) the performance comparison with other related protocols. For the simulation, the size of the overlay is 10^5 if not separately specified. The initial simulation topology is a random graph and all the nodes take the role of a client node in the beginning. The frequency of information sharing among peers using the gossip protocol is set to take place every 10s (seconds).

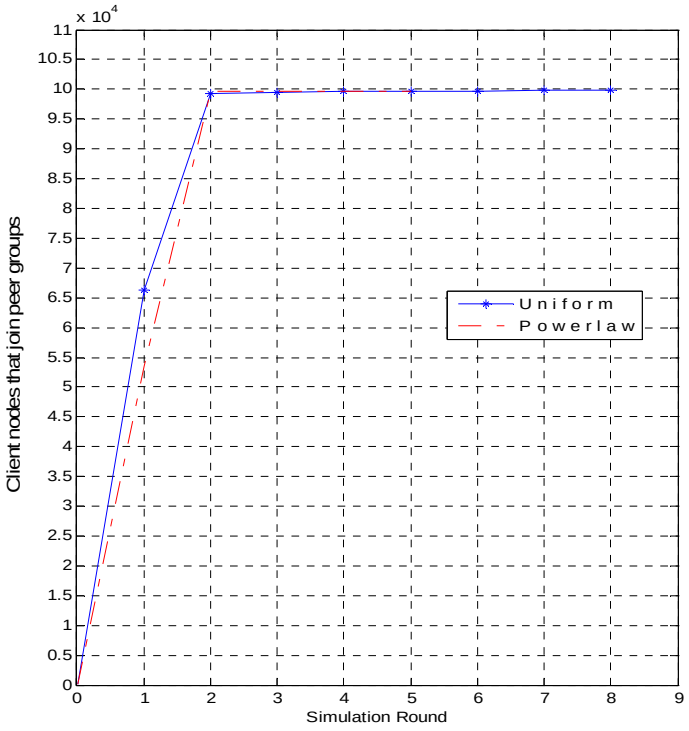

Fig. 3 The client nodes that join the super-peers during simulation rounds

Fig. 3 shows the efficiency of convergence of our GSPS algorithm over simulation time. We can find out using the uniform distribution that on average it takes about eight rounds for all client nodes to join one of the selected super-peers. However, for the power-law distribution, it takes approximately four rounds. During the

initialization of the simulation, we selected a random topology where all the peers take the role of a client-peer. The reason is twofold. On one side, every node joins the network and declares itself as a client node, after that, these nodes can execute the GSPS algorithm to select a super-peer. On the other side, the initial topology is the farthest from the target overlay and provides a better chance to verify the efficiency of our GSPS algorithm. The curves in Fig. 3 represent the number of client nodes that have joined the overlay after a number of simulation rounds. In the first round, the entire set of client nodes exchange information with their neighbors and select the required number of super-peers. After that, the client nodes join the peer groups managed by the selected super-peers. For the capacity of the nodes, two types of distributions are evaluated, the uniform distribution and the power-law distribution. Based on the above results, we can say that the GSPS algorithm performs fast when measured with the convergence time to the target overlay.

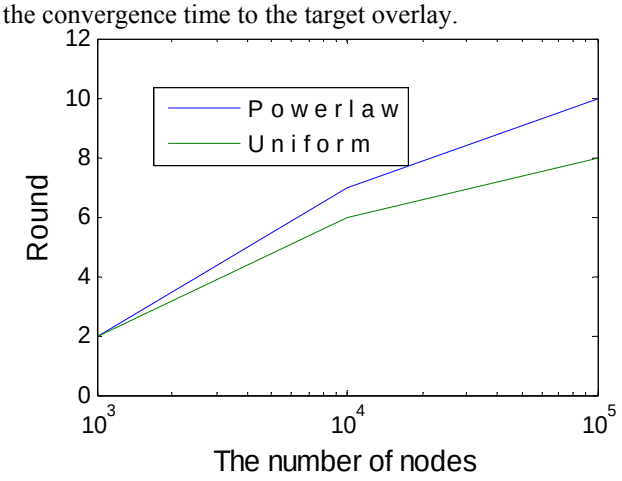

Fig. 4 The scalability in terms of different distribution of nodes' capacity

Fig. 4 demonstrates the scalability of the GSPS. Specifically, as the number of the peers in the network increases, what is the variation of the simulation rounds that are needed to make the overlay converge to the target overlay. In this experiment, two types of distribution of the peers' capacity are utilized: the power-law distribution and the uniform distribution. One can find out from the figure that the number of simulation rounds needed to achieve the target configuration grows a little bit with respect to the size of the network. The reason is that: even there is huge increase in the number of the nodes in the overlay, most of the super-peers can be selected in the first simulation round by all the nodes in the overlay; and only a few super-peers are not selected but they can be selected in the following second or third round. Based on these selected super-peers, the client nodes can join these super-peers based on their set of super-peers candidates very quickly to make the overlay converge to its target. Therefore, the increase in the number of the nodes in the overlay only has a little impact on the convergence to the target overlay. This result proves that the GSPS supports scalability very well.

Fig. 5 shows the comparison between the RASP, and the SG-1, which is implemented in [13], in terms of the influence of the super-peers' maximum capacity

on the convergence to the target overlay. In this experiment, only the number of simulation rounds needed to achieve the target topology is depicted. From this figure, we can see that, as the super-peers' maximum capacity increases, our GSPS takes a few more simulation rounds to converge the target overlay, peaking at 8.5 rounds on average when the capacity is equal to 600. However, the SG-1 takes a little longer simulation time to achieve the target overlay with the increase of the super-peers' maximum capacity, peaking at 14 rounds when the capacity is equal to 600 on average. This reason for this result is due to fact that different methods are utilized in the SG-1 and our GSPS. More specifically, when the super-peers' maximum capacity increases (that is, a super-peer can manage more client nodes), the number of required super-peers is reduced. However, in our GSPS, most of the super-peers can be selected in the first simulation round, no matter the number of the super-peer is reduces or not. Based on the selected super-peers, the client nodes can join one of the peer groups managed by one of the super-peers. Therefore, the increases of the super-peers' maximum capacity has a little impact on the GSPS algorithm. For the SG-1, when the maximum capacity of the super-peers increases, the number of the required super-peers that manage the client nodes in the overlay becomes less. That is to say, more information exchanges between the super-peers are required to change these super-peers to the client nodes, because all the peers in the overlay take the role of a super-peer in the beginning of the simulation. Thus, it takes longer time to build the target overlay with fewer super-peers. As conclusion, the increases of the super-peers' maximum capacity has a little influence on our GSPS algorithm in comparison to the SG-1.

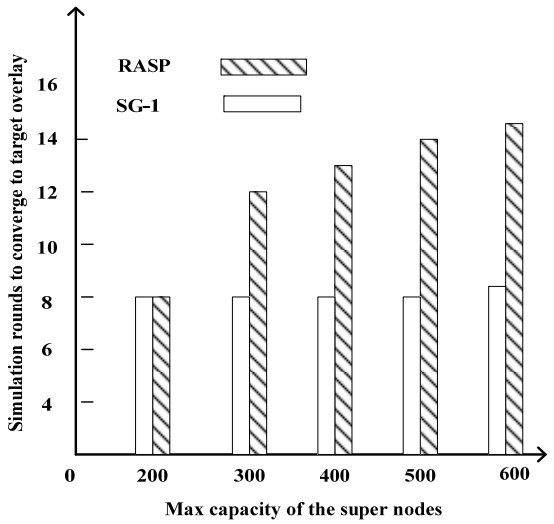

Fig. 5 The influence of the super- nodes' max capacity on the convergence to the target overlay

Fig. 6 demonstrates the robustness of the GSPS. We examined three catastrophic scenarios : 10% of the super-peers are removed at simulation round six; 20% of the super-peers are removed at round six and 30% of the super-peers are removed at round six. One can find out that even when the number of the failed super-peers increases, it almost takes the same number of simulation rounds for the GSPS

algorithm to reach the steady overlay state again. The rationale behind this result is as follows. When the crash of the super-peers happens, these failed super-peers are removed from the overlay; meanwhile, all the client peers, whose super-peers have crashed, remove their super-peer from the set of super-peer candidates and rebuild their set of super-peers as shown in Fig. 1. After that, these client nodes execute the GSPS algorithm to select new super-peers as usual and join the peer groups managed by these super-peers. Based on the simulation result, we can say that the GSPS is not only robust to the failure of super-peers but also efficient in re-organizing the overlay into a new topology.

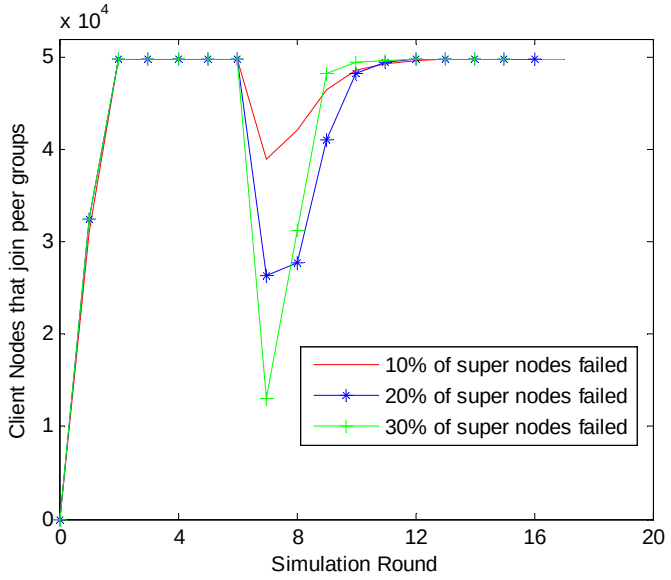

Fig. 6 The failure scenario: 50% of the super nodes are removed at the round 6

4 Related work

In this section, we review the related work on the building and maintaining the P2P overlay for different aims. Meanwhile, we also compare our GSPS algorithm with other research studies.

Some overlay management work focus on the diameter or degree of the overlay. For example, Angluin et al. [3] describe an asynchronous distributed algorithm to quickly convert the nodes in a weakly-connected pointer graph into the leaves of a Patricia tree by the length of node identifiers. Pandurangan et al. [4] present a protocol to build P2P networks with constant degree and logarithmic diameter in a distributed fashion. In [5], an algorithm is proposed that can construct low-diameter resilient topologies, where the degree of the nodes distribution follows a power-law. In [6], Swap Links approach utilizing random walks as a component is proposed to construct required graphs for the overlay, where the degree of each node is the only

parameter that needs setting. In [7], Mushtaq et al. present a quality adaptive mechanism for the multimedia streaming and video on demand services over hybrid P2P overlay networks. Their overlay is built by organization the sender peers in the networks based on the category of their offered video quality and the end-to-end probing among the sender and receiver peers. Different from these work, our work focus on building a super-peer overlay, where the role of a super-peer is identified.

Some researchers work on the construction of the unstructured overlay with random walk. For example, in [8], to build an unstructured overlay, a random walk is used to select neighbors for the new incoming peers in the joining process, in terms of capacity and connectivity to achieve load balancing. Li et al. [25] focus on the layer management in a super-peer architecture, specifically, employing workload model to maintain the optimal size ratio between the leaf-layer and the super-layer. Ganesh et al. [26] focus providing partial views for each node in the overlay without the knowledge of the system size. Thus, the memory cost and the synchronization overhead between nodes are reduced when build the overlay. Different from the above work, our work focuses on building the connection between the nodes, taking account of the nodes' heterogeneity.

Many works have discussed the performance trade-offs and the advantages of structuring peers into two layers: a super-layer, where the super-peers (i.e., the nodes with relatively long lifetime and large capacities) behave like proxies for the peers in the "normal" layer. Montresor [13] proposes a robust protocol SG-1 to construct and maintain overlay topology based on super-peers. In SG-1, a node exchanges information with randomly selected neighbor nodes to identify a super-peer and rearrange the topology according to the application requirements. Jesi et al. [27] propose a novel proximity-aware protocol SG-2 to manage super-peer topologies. The SG-2 employs a gossip-based protocol to spread messages to nearby nodes and a biology-inspired task allocation mechanism to promote the "best" nodes into a super-peer status. The comparisons between the SG-1, the SG-2 and our work are as follows. In the SG-1, the super-peers are selected through comparing the capacity between each two neighboring nodes recursively, where the node with the high capacity takes the role of a super-peer. The SG-2 utilizes a complex biology-inspired method to select the super-peer, considering the locality factor. In our GSPS, we first build a set of the super-peer candidates, and then a super-peer is selected in the way that if a client node belongs to the set of super-peer candidates, which is simple but does not take account of the locality.

5 Conclusion and Future work

In this paper, we have presented a gossip based super-peer selection algorithm (GSPS) that is robust in managing the super-peer overlay, e.g. selecting the super-peers and maintaining the relationship between the super-peers and the client peers. In the GSPS, the peers with higher capacity are selected to be the better super-peers candidates through exchanging information with the neighboring nodes. If a node belongs to the set of super-peer candidates, the node takes a role a super-peer and recruits client nodes. Otherwise, a node chooses to join a super-peer. Moreover, the simulation

results show that the GSPS is 1) efficient in building the target topology, 2) robust to the churn of nodes joining and leaving as well as to the failure of the super-peers.

In our future work, we will evaluate the communication overhead to build the target overlay. Specifically, we take account of two kinds of communication overheads. One is the total number of exchanged message sent to query the load of the neighboring super-peers on average and their sets of super-peer candidates. The other is the total number of communication overhead during the construction of the overlay on average. Another future work is to introduce the locality factor for the location-based optimization, which can improve the communication performance.

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable advice. This work was supported by the ITEA2 Expeshare project, funded by the Finnish Funding Agency for Technology and Innovation (Tekes), and the project of SOPSCC (Pervasive Service Computing: A Solution Based on Web Services), funded in the Ubiquitous Computing and Diversity of Communication (MOTIVE) program by the Academy of Finland.

References

1. A. Oram, “Peer-to-Peer: Harnessing the Benefits of a Disruptive Technology”, chapter 8. O’Reilly & Associates, Mar. 2001.
2. D. S. Milojicic et al. “Peer-to-Peer Computing”. Technical Report HPL-2002-57, HP Labs, Palo Alto, 2002.
3. D Angluin, J Aspnes, J Chen, Y Wu, Y Yin, “ Fast Construction of Overlay Networks”, Proceedings of the seventeenth annual ACM symposium on Parallelism in algorithms and architectures, 2005.
4. G Pandurangan, P Raghavan, E Upfal. “Building low-diameter peer-to-peer networks”, IEEE Journal on Selected Areas in Communications, 2003.
5. RH Wouhaybi, AT Campbell, “Phenix: Supporting Resilient Low-Diameter Peer-to-Peer Topologies”, In Proc. INFOCOM 2004.
6. V. Vishnumurthy, P. Francis, “On heterogeneous overlay construction and random node selection in unstructured P2P networks”, In Proc. IEEE INFOCOM, 2006.
7. M Mushtaq, T Ahmed, “Hybrid Overlay Networks Management for Real-Time Multimedia Streaming over P2P Networks”, In Proc. MMNS 2007, LNCS 4787, Springer.
8. KW Kwong, DHK Tsang, “Building heterogeneous peer-to-peer networks: protocol and analysis”, IEEE/ACM Transactions on Networking, 2008.
9. Lime Wire LLC, Rfc-Gnutella 0.6, http://rfcgnutella.sourceforce.net/development.
10. Kazaa, http://www.kazaa.com/us/help/glossary/p2p.htm (accessed 17-07-2009)
11. S. Saroiu, P. K. Gummadi, S. D. Gribble, “A measurement study of peer-to-peer file sharing systems”. In Proc. of Multimedia Computing and Networking (MMCN), San Jose, CA, USA, January 2002.
12. B. Yang, H. Garcia-Molina. “Designing a Super-peer Network”. In Proc. of the 19th Int. Conf. on Data Engineering (ICDE), Bangalore, India, Mar. 2003.

13. A. Montresor. “A robust protocol for building super-peer overlay topologies”, In proceedings of the 4th International Conference on Peer-to-Peer Computing, pages 202–209, 2004.
14. F Wang, J Liu, Y Xiong, “Stable peers: Existence, importance, and application in peer-to-peer live video streaming”, IEEE INFOCOM 2008.
15. J Zhou, Z Ou, M Rautiainen, M Ylianttila, “P2P SCCM: Service-oriented Community Coordinated Multimedia over P2P", In Proc. IEEE Congress on Services Part II, 2008.
16. M. Jelasity, A.Montresor., “Epidemic-Style Proactive Aggregationin Large Overlay Networks”. In Proc. of the 24th International Conf. Distributed Computing Systems, 2004.
17. M. Jelasity, W. Kowalczyk, M. van Steen. “Newscast computing.Technical Report IR-CS-006“, Vrije Universiteit Amsterdam, Dept. of Computer Science, Nov. 2003.
18. Y Chen, G Pandurangan, D Xu, “Robust Computation of Aggregates in Wireless Sensor Networks: Distributed Randomized Algorithms and Analysis”, IEEE Transactions on Parallel and Distributed Systems, 2006.
19. E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, “A Survey and Comparison of Peer-to-Peer Overlay Network Schemes”, IEEE Comm. Surveys and Tutorials, vol. 7, no. 2, pp. 72-93, 2005.
20. D Stutzbach, R Rejaie, “Characterizing the two-tier Gnutella topology”, In Proc, ACM SIGMETRICS, 2005.
21. S Voulgaris, D Gavidia, M Van Steen, “YCLON: Inexpensive Membership Management for Unstructured P2P Overlays”, Journal of Network and Systems Management, 2005.
22. Q Yuan, J Wu, “DRIP: A Dynamic VoRonoi RegIons-Based Publish/Subscribe Protocol in Mobile Networks”, In Proc. IEEE INFOCOM 2008.
23. S Basagni, “Distributed clustering for ad hoc networks”, Journal of Parallel Architectures, Algorithms, and Networks, 1999.
24. “PeerSim P2P Simulator,” http://peersim.sourceforge.net/
25. X Li, Z. Zhuang, Y. Liu. “Dynamic Layer Management in Superpeer Architectures”. IEEE Transactions on Parallel and Distributed Systems, 16, 2005.
26. AJ Ganesh, AM Kermarrec, L Massoulie, “Peer-to-peer membership management for gossip-based protocols”, IEEE transactions on computers, 2003.
27. GP Jesi, A Montresor, O Babaoglu, “Proximity-aware superpeer overlay topologies”, IEEE Transactions on Network and Service Management, 2007.