A Learning Automata-based Version of SG-1 Protocol for Super-peer Selection in Peer-to-Peer Networks

Abstract. Super-peer topologies have been found efficient and effective in heterogeneous peer-to-peer networks. Due to dominant position of super-peers, super-peer selection requires a protocol that is aware of peer capacities. Lack of global information about other peers' capacity and dynamic nature of peer-to-peer networks are two major challenges that impose uncertainty in decision-making. SG-1, is a well-known super-peer selection protocol considering peer capacities, but lack of an appropriate decision-making mechanism makes this protocol slow at convergence and imposes overhead of client transfer between selected super-peers. In this paper, we propose an improved version of SG-1 that uses learning automata as an adaptive decision-making mechanism. For this purpose, each peer is equipped with a learning automaton which is used locally in the decisions taken by that peer. Simulations show effectiveness of proposed protocol in terms of convergence time, scalability, capacity utilization, behavior towards super-peer failure and communication cost, compared to SG-1.

Keywords: peer-to-peer network, super-peer, learning automata

1 Introduction

Peer-to-Peer (P2P) overlay networks are distributed systems formed on top of the underlying physical network. Main purpose of these networks is sharing information and resources (i.e., CPU cycles, memory, storage space and bandwidth). Recent years, research attention increasingly focused on various aspects of P2P networks. Wide variety in distributions of peer characteristics such as the uptime, bandwidth, or available storage space demonstrates heterogeneity in P2P networks. As a result, pure form of P2P architectures can lead in poor performance because all peers without respect to their capacities act equal in all operations. Specially, when size of network increases such situation may lead in overloading of low capacity peers. Super-peer topologies developed in order to defeat this scalability limitation [1]–[3].

In super-peer topologies, peers are taken one of super-peer or client role. Super-peers are connected to each other and form an overlay like pure P2P systems. On the

other hand, each client without taking part in search process just sends queries to its super-peer and receives results from it [4]. Efficient performance which is resulted by reducing the time needed for search, lead in increasing attention to super-peer systems such as Gnutella vs. 6 [5] and Skype [6]. However protocols used in these large-scale distributed applications have no dynamic mechanism to adapt them when it is needed to rebuild the relationships between clients and super-peers. Shortcomings of super-peer topologies are pointed out by [4]. Moreover they propose super-peer redundancy to improve reliability in super-peer topologies. However they do not address super-peer selection problem.

Super-peers have a dominant position in super-peer topologies, so selecting subset of peers as super-peer requires an approach that considers capability of peers for providing service. Moreover, lack of global information about other peers' capacity and dynamic nature of peer-to-peer networks make this problem more challenging. SG-1 is a super-peer selection protocol that considers some peer characteristics according to applications need, called capacity. SG-1 aims at forming a super-peer set with minimum cardinality to cover all other peers. For this purpose, SG-1 tries to select high capacity peers and utilizes capacity of selected super-peers in a greedy manner. As a result, in a network in which a larger number of peers have a relatively high capacity, SG-1 can lead in demoting some high capacity peers. Furthermore, SG-1 doesn't employ any appropriate decision-making mechanism. Therefore, inappropriate decisions taken by SG-1, makes this protocol slow at convergence and imposes overhead of client transfer between super-peers. In this paper, we propose an improved version of SG-1 that uses learning automata as an adaptive decision-making mechanism. We aim at minimizing number of super-peers to cover all other peers according to current network condition. Learning automata [7], [8] have been found [9], [10] to be useful in dynamic environments with uncertainty in decision-making. Therefore, it can be used to solve problems in P2P networks, where peers frequently join and leave and operate within a dynamic environment.

The rest of this paper is organized as follows. In Section 2, we review the related work. Learning automata will be discussed in Section 3. Section 4, represents our proposed protocol. Section 5, gives the performance evaluation of the proposed method, and finally Section 6 concludes the paper.

2 Related Work

Adaptive super-peer selection requires to consider a criterion that shows eligibility of peers to act as super-peer. SG-1 [11] is a well-known super-peer selection protocol considering peer capacities. Capacity is defined as a general concept that can be calculated by some peer properties, such as bandwidth and computational capabilities. In SG-1 every node maintains different views that each of them contains neighboring peers with some special characteristics. In their proposed protocol, information exchange with randomly selected neighbors lead in capacity comparison and rearranging the topology. The capacity parameter which is used in [12], [13] is similar to SG-1. In [12] the authors propose Myconet, a bio-inspired approach to model relation-

ships between peers in a super-peer topology with focusing on self-healing ability. But, there is no report on the overhead evaluation of Myconet. [13] presents SPS protocol that mainly focuses on quickly construction of a super-peer overlay. In SPS, each peer periodically rebuilds its set of super-peer candidates that imposes communication overhead. In protocols such as SG-1, Myconet and SPS each client is assigned to only one super-peer. [14] proposes another variation of this basic design that ordinary peers are allowed to be connected with each other and to be clients of more than one super-peer. They obtain efficiency in search operations but they do not provide any mechanism for super-peer election problem.

[15]–[19] considers lifetime of peer besides the concept of capacity. SPSI [15] introduce self-information theory to select super-peers. The mechanism which is used by ERASP [16] is similar to SG-1, but in ERASP client peers are allowed to have multiple super-peers. DLM [17] uses weighted metric for super-peer selection and presents a workload model to find the desirable number of super-peers adaptively. However, DLM is limited to file sharing applications. The same problem is defined in [18] as an optimization problem. In [18], the authors try to overcome shortcomings of DLM by particle swarm optimization. Some studies [20]–[22] focus on connections between super-peers and propose some special topologies. They do not address super-peer selection problem but [23] suggests gradient topology besides using a utility function that covers the concept of peer capacity.

In some applications, besides capacity the latency between super-peer and client is of great importance. In [24], authors define this problem as multiple colored domination and propose H2O, a super-peer selection protocol for unstructured P2P networks. But, H2O uses flooding mechanism that imposes considerable overhead. Natural evolution of SG-1 is SG-2 protocol [25] with bio-inspired approach for super-peer selection. SG-2 takes advantage of network proximity for building a super-peer overlay. The same problem is considered by [26], that proposes an algorithm inspired by a standard neural network learning algorithm. As another example, authors in [27], [28] aim at minimizing distance between super-peers and their clients, as well as between connected super-peers. In [27] they consider this problem as a hub-location problem and prove that it is NP-hard. After that in [28] they present a heuristic algorithm which is close to optimal, but requires a global view. Recently, [29] model super-peer selection problem in P2P streaming systems using game theory. Their proposed model improves system performance, by enabling upload capacity of entire network.

3 Overview of learning automata

learning automaton (LA) [7], [8] is an adaptive decision-making unit that has been shown to perform well in computer networks [9], [10]. LA can improve its performance by learning how to choose the optimal action from a finite set of allowed actions through repeated interactions with a random environment. At each iteration, the action is chosen at random based on a probability distribution kept over the action-set and at each instant the given action is served as the input to the random environment. Each LA belongs to one of fixed or variable structure category [8]. Later case are

represented by a triple $< \beta, \alpha, L >$, where β is the set of inputs, α is the set of actions, and L is learning algorithm that is a recurrence relation which is used to modify the action probability vector.

Let $\alpha_i(k) \in \alpha$ and $p(k)$ denote the action selected by LA and the probability vector defined over the action set at instant k, respectively. Let a and b denote the reward and penalty parameters and determine the amount of increases and decreases of the action probabilities, respectively. Let r be the number of actions that can be taken by LA. At each instant k, the action probability vector $p(k)$ is updated by the linear learning algorithm given in (1), if the selected action $\alpha_i(k)$ is rewarded by the random environment, and it is updated as given in (2), if the taken action is penalized. If $a = b$, the recurrence equations (1) and (2) are called linear reward-penalty (L_{RP}) algorithm. More information can be found in [7], [8].

$$p_j(n+1) = \begin{cases} p_j(n) + a\left(1 - p_j(n)\right), & j = i \\ (1-a)\, p_j(n), & \forall j,\ j \neq i \end{cases} \tag{1}$$

$$p_j(n+1) = \begin{cases} (1-b)\, p_j(n), & j = i \\ \frac{b}{r-1} + (1-b)\, p_j(n), & \forall j,\ j \neq i \end{cases} \tag{2}$$

4 Proposed method

This section contains overview of SG-1 protocol [11] and detailed description of our proposed protocol, SG-LA. At first, we introduce common concepts and notations used in these protocols: Given a network with n peers, each peer p is associated with a parameter $capacity(p)$, shows the maximum number of clients that p can handle if elected as super-peer. At any given time, $load(p)$ denotes the current number of client peers currently managed by super-peer p. If $load(p)$ is less than $capacity(p)$, p is set as under-loaded. In SG-LA, client c is associated to super-peer s, if and only if s is under-loaded and has more capacity. Being under-loaded is common constraint, but in SG-1it is allowed for a super-peer and its client to be equal in capacity. In both SG-1 and SG-LA, each client is assigned to only one super-peer.

4.1 SG-1 Protocol

In SG-1 every node has four neighbor sets: $\text{view}_{\text{connected}}$, $\text{view}_{\text{super-peer}}$, $\text{view}_{\text{under-loaded}}$ and $\text{view}_{\text{client}}$. Neighbor items are also called node descriptors, including the identifier, capacity, current role (super-peer or client) and other useful property of a peer. $\text{view}_{\text{connected}}$ contains the neighbors forming the underlying topology and it is maintained by an underlay peer sampling service (Newscast [30]). $\text{view}_{\text{super-peer}}$ and $\text{view}_{\text{under-loaded}}$ in each peer p respectively contains a random sample of super-peers and under-loaded super-peers in the system, that are known to p. Finally, $\text{view}_{\text{client}}$ for each super-peer, consists of clients currently associated with it. SG-1 assigns each node the initial role of super-peer. Each super-peer p periodically examines members in its under-loaded neighbor set and tries to find a suitable candidate for client trans-

fer. If such a candidate s exists, transferring process is done from p to s, according to Fig. 1 (here C denotes the number of clients that are allowed to be accepted by s). If p misses all of its clients, and s is not exhausted, peer p becomes client of s. Otherwise p remains in super-peer role and the comparison will be made between capacity of p and that of highest capacity client of s, denoted as r. If r's capacity is higher than capacity of p, role swapping is made between p and r. During this process, whenever each client loses its super-peer, it becomes a super-peer by itself.

4.2 SG-LA Protocol

In our proposed protocol, SG-LA, we apply learning automata in order to train nodes during gossip-based algorithms. Each peer p in the network is equipped with a learning automaton $Role\_LAp$ that uses L_{RP} algorithm to help the node for making decision about its suitable role according to network condition. Therefore the action set of each $Role\_LA$ consists of two actions; Super and Client. It should be noted that, two terms "action" and "role" are different from each other: each peer in the network introduces itself according to its current role and takes the corresponding responsibilities. But, the action set of each peer is related to its own $Role\_LA$. Each peer makes use of its selected action, locally, to be able to make decisions about its role.

When each peer p joins the system, SG-LA assigns client role to p. So, initially $view_{under\text{-}loaded}$ in each peer contains random sample of participating peers. This view is updated in each round by a gossiping protocol. As running SG-LA and arising super-peers in the network, $view_{under\text{-}loaded}$ rapidly converged to desirable neighbor set which is used in our algorithms.

After joining the system, each peer for the first time, performs its learning phase according to Fig. 2. In learning phase, each peer p updates its action probability vector according to "its selected action" and "the comparison which is made between its capacity and that of neighbor in its under-loaded view". Then peer p randomly chooses an action again. This process is repeated for each neighbor in under-loaded view of peer p. It is clear that our policies for getting reward or penalty are based on the goal of maximizing capacity of selected super-peers. The last selected action can be considered as the output of learning phase.

```
DoTransfer(super-peer p, super-peer s)
set C with min(capacity(s)-load(s), load(p))
Transfer C from p to s
If load(p) = 0 and load(s) < capacity(s)
    p becomes client of s
Else
    set r with the highest capacity client of s
    If capacity(r) > capacity(p)
        swap role of r and p
```

Fig. 1. DoTransfer method in SG-1

```
Learning phase for each peer p
For each s ∈ p.view underloaded do
    Peer p randomly chooses an action by p.Role_LA
    If p.Role_LA.action = client
      if capacity(s) > capacity(p)
        Update action probability vector by Equation (1)
      else if capacity(s) < capacity(p)
        Update action probability vector by Equation (2)
    Else if p.Role_LA.action = super-peer
      if capacity(s) > capacity(p)
        Update action probability vector by Equation (2)
      else if capacity(s) < capacity(p)
        Update action probability vector by Equation (1)
Peer p randomly chooses an action by p.Role_LA
```

Fig. 2. Learning phase for each peer p

Since under-loaded view of each peer is updated periodically, each peer performs learning phase in each round except for attached clients that perform this phase on demand. In other words learning phase is the starting step in our designed algorithms for unattached clients and super-peers but attached clients perform this phase if a request received. Since joining the system each peer periodically performs an algorithm, according to its current role. As shown in Fig. 3: unattached clients perform an algorithm shown in part (a) and super-peers act as part (b). Attached clients perform some trivial task that is discussed at the end of this section. At the rest, we give a detailed description of Fig. 3. In these algorithms some notations used as follow. **SP**: super-peer, **USP**: under-loaded super-peer, **AC**: attached client, **UC**: unattached client, **super(s)**: super-peer of s, **LPH**: output of current learning phase and p.**HigherSet**/p.**LowerSet** consists of neighboring nodes whose capacity is higher/lower than that of p. There are some key rules applied in our algorithms:

1. Each UC or AC can promote to super-peer role if and only if its LPH is super.
2. Each USP is allowed to accept more load if and only if its LPH is super.
3. Each UC can become AC if and only if its LPH is client.
4. SG-1.Dotransfer(p,s) method can be performed if and only if p and s are SPs, s has higher capacity and s.LPH is super.
5. Each SP, p is allowed to swap role of its highest capacity client with one of lower capacity SPs in $p.\text{view}_{\text{under-loaded}}$ if and only if p.LPH is super.

According to part (a) of Fig. 3, in first step, each UC such as p performs its learning phase. According to rule (1), if p.LPH is super, p itself promotes to super-peer role. Otherwise p considers its HigherSet: In each iteration, p picks its highest capacity neighbor, s, and sends it a request message. If s is an USP or it is an UC, this message made s to use its own LPH. If s is an USP and rule (2) is obeyed by s, p becomes its client. If s is an UC and follows rule (1) firstly s promotes as SP then according to rule (2), p becomes its client. If s is an AC, at first p considers $super(s)$, instead of s. If $super(s)$ is an USP and obeys rule (2), p becomes its client. Otherwise, s utilizes its own LPH, if rule (1) is followed by s, firstly s promotes as super-peer then according to rule (2), p becomes its client. Above process continues until p becomes AC or no more peer exists in HigherSet of p.

According to part (b) of Fig. 3, at first step, each SP such as p performs its learning phase. Then p utilizes its LPH to check its eligibility for acting as super-peer. If p.LPH is super, peer p is eligible to absorb more clients. So, if p is USP, it picks the lowest capacity neighbor x from p.LowerSet. If x is an UC, it uses its own LPH, if x follows rule (3), x is attached to p. Otherwise, if x is a SP, rule (4) is investigated to perform SG-1.DoTransfer(x,p) (as shown in Fig. 1). This loop continues until p becomes fully-loaded or no more peer exists in LowerSet of p. Then SP p, examines its view to check whether exists any super-peer m in its under-loaded view whose capacity is lower than that of p's highest capacity client, r, or not. If there is such super-peer, according to rule (5) swapping role is made between m and r.

Then as shown in part (b) of Fig. 3 (line 11), each SP such as p, regardless of its LPH, forms its HigherSet. In this way, SPs can compete with other higher capacity peers for absorbing clients. In each iteration, p picks its highest capacity neighbor, s.

If s is a SP, rule (4) is investigated to perform SG-1.DoTransfer(p,s). Otherwise If s is an UC, and follows rule (1) firstly s promotes as super-peer then rule (4) is investigated to perform SG-1.DoTransfer(p,s). Finally, If s is an AC, p considers $super(s)$ instead of s. Then rule (4) is investigated to perform SG-1.DoTransfer(p,$super(s)$). Selecting candidates in order to perform SG-1.DoTransfer method continues until SP p absorbed by other super-peer or when there is no more peer in its HigherSet.

	(a) **Local:** client peer *p*, **Remote:** peer *s*	(b) **Local:** super-peer *p*, **Remote:** peers *x*, *s*
01	*p* performs its learning phase according to Fig. 2	*p* performs its learning phase according to Fig. 2
02	**If** *p*. **LPH** = super	**If** *p*. **LPH** = super
03	*p* promotes to super-peer role	**While** *p*.LowerSet is not empty **and** *p* is **USP**
04	**Else**	pick *x*, lowest capacity peer in *p*.LowerSet
05	**While** *p*.HigherSet is not empty **and** Found=false	**if** *x* is **UC and** *x*. **LPH** = client
06	pick *s*, highest capacity peer in *p*.HigherSet	*x* becomes client of *p*
07	**If** *s* is **USP** and *s*. **LPH** = super	**else if** *x* is a **SP**
08	*p* becomes client of *s* //Found = true	**SG-1.DoTransfer**(*x*, *p*)
09	**Else if** *s* is **AC**	check eligibility of its highest capacity client to swap
10	**If super**(*s*) is **USP** and **super**(*s*).**LPH** = super	**While** *p*.HigherSet is not empty **and** *p* is **SP**
11	*p* becomes client of **super**(*s*)//Found = true	pick *s*, highest capacity peer in *p*.HigherSet
12	**Else**	**If** *s* is a **SP and** *s*. **LPH** = super
13	checkAttached = true	**SG-1.DoTransfer**(*p*, *s*)
14	**If** checkAttached = true **or** *s* is an **UC**	**else if** *s* is an **AC and super**(*s*). **LPH** = super
15	**If** *s*. **LPH** = super	**SG-1.DoTransfer**(*p*, **super**(*s*))
16	*s* promote to super-peer role	**else if** *s* is Unattached-Client **and** *s*. **LPH** = super
17	*p* becomes client of *s* //Found = true	*s* promotes to super-peer role
18	**End While**	**SG-1.DoTransfer**(*p*, *s*)

Fig. 3. Algorithm corresponding to (a): unattached clients, (b): super-peers.

Every round, each AC such as p forms a candidate list including its neighboring nodes whose capacity is higher than that of $super(p)$. Then, p sends this list by candidate message to its SP. As a result, more opportunities are given to SPs, without imposing significant responsibility to ACs.

5 Performance evaluation

In this section we have developed our protocol using PeerSim simulator [31] to evaluate efficiency of SG-LA in comparison with the well-known approach, SG-1. Two different distributions, power-law (with parameter $\alpha = 2$) and uniform are applied to initialize capacity of nodes. In general, power-law distribution, maximum capacity of 500, partial view of size 30 and a network size of 10^5 is used as a default configuration, unless noted explicitly. Results are averaged over 25 experiments with a=b=0.1 as reward and penalty parameters. At the rest of this section, we have focused on the following major aspects: 1) Convergence time, 2) Capacity utilization, 3) Robustness in the face of super-peers' failure, 4) Scalability, and 5) Communication cost.

Let convergence time be the number of rounds needed to converge in an overlay with a stable number of super-peers that no more clients join or leave them. Fig. 4 shows number of selected super-peers as simulation proceeds, (for two different capacity distributions). According to Network column in part (a) of Table 1, when using power-law distribution only a small number of peers have a comparably high capacity

but according to Network column in part (b), in uniform distribution capacity values assigned uniformly. Number of selected super-peers grouped by peer capacities and number of rounds needed to find each subset are extracted from experiments of Fig. 4 and shown in Table 1. According to part (a) and part (b) of Table 1, in both capacity distributions SG-LA and SG-1 only select their super-peers from highest capacity subsets. But the number of rounds needed to find super-peers included in each subset, is comparably lower in SG-LA. In other words, high capacity peers in SG-LA learn about their eligibility faster than SG-1. Because in SG-LA each peer is able to learn its eligibility compared to other participating peers in adaptive manner. However, in SG-1 each peer joins as super-peer and role changes are dependent on capacity comparison with a randomly selected neighbor. According to part (a) of Table 1, when capacity parameter follows power-law distribution both of SG-1 and SG-LA obtain the same number of super-peers which is close to minimum possible. When capacity parameter follows uniform distribution, number of participating peers included in two first subsets is close to each other. In such situation, majority of second highest subset are demoted as clients by SG-1. But, in SG-LA higher number of peers from second highest subset, find that they are eligible with respect to current network condition. The behavior of SG-1 can be explained as a result of greedy manner of this protocol in utilizing capacity of super-peers.

Fig. 5 shows capacity utilization of SG-1 and SG-LA as simulation rounds increase, (for both capacity distributions).

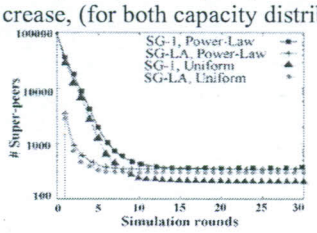

Fig. 4. Number of selected super-peers as simulation proceeds

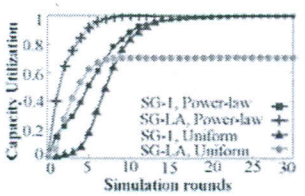

Fig. 5. Capacity utilization

Table 1. Number of selected super-peers grouped by peer capacities and number of rounds needed to find each subset, SG-LA compared with SG-1. (a): Power-law, (b): Uniform

(a)	Network	SG-LA		SG-1	
capacity	# peers	# supers	# rounds	# supers	# rounds
[401,500]	49	49	1	49	1
[301,400]	85	85	1	84	9
[201,300]	156	156	2	155	13
[101,200]	496	55	11	57	25
[1,100]	99214	0	-	0	-
(b)	**Network**	**SG-LA**		**SG-1**	
capacity	# peers	# supers	# rounds	# supers	# rounds
499	199	199	1	192	20
498	200	102	10	8	22
≤497	99601	0	-	0	-

At any given time capacity utilization (*CU*) can be defined by ratio of current number of attached clients to total capacity provided by super-peers, as given in (3) (let S denotes number of selected super-peers). According to Fig. 5, when power-law distribution is applied, SG-LA can reach 99% CU in only 9 rounds, but it takes SG-1 for 20 rounds.

$$CU = \#\ attached\ clients\ / \sum_{i=1}^{S} Capacity(i) \tag{3}$$

In SG-LA, each peer learns about its eligibility in its learning phase therefore high capacity peers are able to take super-peer role in first few rounds. But, SG-1 makes each decision by only one capacity comparison. Due to large number of role changes needed in SG-1, transferring clients between selected super-peers is more probable. As a result, it takes SG-1 a long time to utilize capacity of selected super-peers. In the case of uniform distribution, until round 9, SG-LA has higher CU, and finally it converges to 70% CU. Because according to part (b) of Table 1, in SG-LA higher number of peers find themselves eligible with respect to current network condition. So, as far as possible, SG-LA prevents high capacity peers from demotion. However, SG-1 utilizes capacity of super-peers in a greedy manner. So, until round 19, SG-1 tries to get 99% CU. In this way, some of high capacity peers are demoted by SG-1.

To evaluate robustness of our protocol in the face of super-peers' failure, Fig. 6 shows number of selected super-peers in a scenario in which 50% of super-peers removed at round 30, in the case of power-law distribution. The same scenario is evaluated with uniform distribution and *CU* of these experiments is shown in Fig. 7. Important results of these evaluations gathered in Table 2 and Table 3. According to Table 3 when using power-law distribution, in SG-LA after crash 50% of clients change into unattached statue. Then, all of them change into super-peer role, so in worst case SG-1 has experience of 16% *CU* (Fig. 7). Finally as shown in Table 2, SG-1 converges after 27 rounds. In the same situation, after crash, SG-LA starts with 17% unattached clients. Each of these clients performs its learning phase and makes a decision for its role according to the network situation after crash. As shown in Fig. 7, SG-LA reaches 99% utilization faster than SG-1 and in worst case never goes below 50% utilization. Finally, SG-LA converges in only 15 rounds (Table 2) and the minimum possible number of super-peers (546) is obtained.

According to Table 3, when uniform distribution is applied, on average SG-LA selects higher number of super-peers (281) compared with SG-1 (200). It should be noted that, according to Table 2, SG-LA selects these super-peers only from three highest capacity subsets (499,498,497) and selection from each subset is limited to scenarios in which other higher subsets have been selected completely. After crash in SG-LA, unattached clients start performing their learning phase. A high number of them in first 4 rounds find that they are eligible to be a super-peer, so SG-LA starts with low *CU* (Fig. 7). Then by repetition of learning phase and going into competition with other higher capacity peers for client transfer, in next 4 rounds, some of these super-peers demote and number of super-peers became stable. At the same time, *CU* increase and converges to 73% (Fig. 7). Due to larger number of high capacity peers in uniform distribution, this amount of *CU* is reasonable to prevent eligible super-peers from demotion. According to Table 3, this behavior of SG-LA resulted in lower client transfer compared with SG-1. According to Fig. 7, after crash, there is a sharp fall in *CU* of SG-1. Because all of peers promote as super-peer, except for small number of remaining attached clients. In next 18 rounds, SG-1 utilizes capacity of super-peers in a greedy manner which is resulted in higher number of client transfers compared with SG-LA (Table 3).

To evaluate scalability of our protocol, Table 4 shows number of rounds needed to converge, with both capacity distributions in different network sizes. SG-1 assigns

each node the initial role of super-peer. So, by growth in the size of overlay, the number of role changes needed to obtain the target topology increases. On the other hand SG-1 dose not employ any appropriate mechanism for making decision about demotions and promotions. But, in SG-LA each peer joins as unattached client and takes one of super-peer or client role with respect to the action chosen in its learning phase. Due to adaptive manner of SG-LA, it is less affected by growth in network size. According to Table 4, smaller deviation in number of rounds needed to converge as the network size increases, shows scalability of SG-LA compared with SG-1.

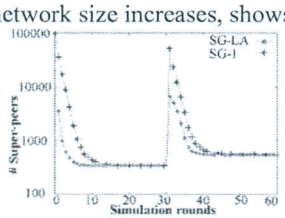

Fig. 6. Failure scenario, Power-law distribution

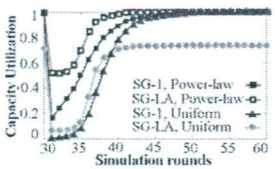

Fig. 7. Capacity utilization in failure scenario

Table 2. Percentage of selected super-peers grouped by peer capacities and number of rounds needed to find each subset (after crah), SG-LA compared with SG-1. (a): Power-law, (b): Uniform

(a)	SG-LA		SG-1	
capacity	% supers	# rounds	% supers	# rounds
[401,500]	100	1	100	1
[301,400]	100	1	100	1
[201,300]	100	1	100	2
[101,200]	85	15	86	27
[1,100]	0	-	0	-
(b)	**SG-LA**		**SG-1**	
capacity	% supers	# rounds	% supers	# rounds
499	100	1	99.95	9
498	98.8	7	47.9	20
497	16	12	0	-
≤496	0	-	0	-

Table 3. Failure scenarios, SG-1 compared with SG-LA

	Unattached clients		# supers		# rounds		# Client transfers	
	SG-1	SG-LA	SG-1	SG-LA	SG-1	SG-LA	SG-1	SG-LA
Power-law	٪52	٪17	552	546	27	15	16.33	9.07
Uniform	٪37	٪21	200	281	22	13	16.59	4.5

Overhead includes: 1. Number of messages sent to ask for information, and candidate messages sent by attached clients (Fig. 8), and 2. Number of client transfers (Fig. 9). According to Fig. 8, number of Request messages per peer in SG-LA and SG-1 is close to each other. Candidate message is dedicated to SG-LA but it is trivial.

Table 4. NS: Network Size, SD: Standard Deviation, (a): power-law, (b) uniform.

	SG-1		SG-LA	
	(a)	(b)	(a)	(b)
$NS=10^3$	10	8	5	4
$NS=10^4$	18	13	9	7
$NS=10^5$	25	24	11	10
SD	7.5	8.1	3.05	3

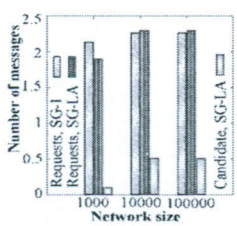

Fig. 8. Overhead of transmitted messages

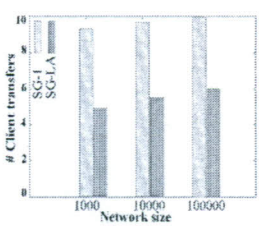

Fig. 9. Overhead of client transfer

Fig. 9 shows number of client transfers per peer in different network sizes. As shown in Fig. 9, even in a network with 10^5 peers, number of client transfers in SG-LA is less than SG-1. Because in SG-LA each peer makes its decisions based on the action which is selected in its learning phase but in SG-1 these decisions are made by comparing capacity with only one of randomly selected neighbors. On the other hand, in SG-LA only super-peers whose chosen action is super-peer, are allowed to accept more clients. As a result, client transfer in SG-LA is less probable. As shown in Fig. 8 and Fig. 9 total communication cost of SG-LA is less than that of SG-1.

6 Conclusion

This paper presents SG-LA, a learning automata based version of SG-1 protocol. We aim at minimizing number of super-peers according to network condition. For this purpose, each peer in the network makes use of its own learning automaton to be able to make decision about its eligibility. According to simulations, in both power-law and uniform capacity distributions, SG-LA can converge considerably faster compared with SG-1. Specifically in the case of power-law distribution, SG-LA can reach maximum capacity utilization faster than SG-1. And when capacity parameter follows uniform distribution, despite of SG-1, SG-LA prevents relatively high capacity peers from demotion, as far as possible. As shown in a catastrophic failure scenario, above results is not limited to normal situation. Furthermore, SG-LA is less affected by super-peer failure and it is able to recover faster. On the other hand, SG-LA reduces overhead of client transfers and shows better scalability compared with SG-1.

References

1. Androutsellis-Theotokis, S., Spinellis, D.: A survey of peer-to-peer content distribution technologies. In: ACM Computing Surveys (CSUR), vol. 36, no. 4, pp. 335–371 (2004)
2. Lua, E.K., Crowcroft, J., Pias, M., Sharma, R., Lim, S.: A survey and comparison of peer-to-peer overlay network schemes. In: IEEE Communications Surveys & Tutorials, vol. 7, no. 2, pp. 72–93 (2005)
3. Meshkova, E., Riihijärvi, J., Petrova, M., Mähönen, P.: A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks. In: Computer networks, vol. 52, no. 11, pp. 2097–2128 (2008)
4. Yang, B., Garcia-Molina, H.: Designing a super-peer network. In: Proceedings of 19th International Conference on Data Engineering pp. 49–60 (2003)
5. Gnutella, http://rfc-gnutella.sourceforge.net
6. Skype, http://www.skype.com/en/
7. Najim, K., Poznyak, A. S.: Learning automata: theory and applications. Pergamon Press, Inc (1994)
8. Narendra, K. S., Thathachar, M. A.: Learning automata: an introduction. Englewood Cliffs: Printice-Hall Inc (1989)
9. Akbari Torkestani, J., Meybodi, M. R.: An intelligent backbone formation algorithm for wireless ad hoc networks based on distributed learning automata. In: Computer Networks, vol. 54, no. 5, pp. 826–843 (2010)

10. Esnaashari, M., Meybodi, M. R.: Deployment of a mobile wireless sensor network with k-coverage constraint: a cellular learning automata approach. In: Wireless Networks, vo. 19, no. 5, pp. 945–968 (2013)
11. Montresor, A.: A robust protocol for building superpeer overlay topologies. In: Proceeding of Fourth International Conference on Peer-to-Peer Computing, pp. 202–209 (2004)
12. Snyder, P.L., Greenstadt, R., Valetto, G.: Myconet: A fungi-inspired model for superpeer-based peer-to-peer overlay topologies. In: 3rd International Conference on Self-Adaptive and Self-Organizing Systems, pp. 40–50 (2009)
13. Liu, M., Harjula, E., Ylianttila, M.: An efficient selection algorithm for building a super-peer overlay. In: Journal of Internet Services and Applications, vol. 4, no. 4, pp. 1–12 (2013)
14. Singh, A., Haahr, M.: Creating an adaptive network of hubs using Schelling's model. In: Communications of the ACM, vol. 49, no. 3, pp. 69–73 (2006).
15. Gao, Z., Gu, Z., Wang, W.: SPSI: A hybrid super-node election method based on information theory. In: 14th International Conference on Advanced Communication Technology, pp. 1076–1081 (2012)
16. Liu, W., Yu, J., Song, J., Lan, X., Cao, B.: ERASP: an efficient and robust adaptive superpeer overlay network. In: Progress in WWW Research and Development, pp. 468–474 (2008)
17. Xiao, L., Zhuang, Z., Liu, Y.: Dynamic layer management in superpeer architectures. In: IEEE Transactions on Parallel and Distributed Systems, vol. 16, no. 11, pp. 1078–1091 (2005)
18. Sachez-Artigas, M., Garcia-Lopez, P., Skarmeta, A.F.G.: On the feasibility of dynamic superpeer ratio maintenance. In: International Conference on Peer-to-Peer Computing, pp. 333–342 (2008)
19. Chen, N., Hu, R., Zhu, Y., Wang, Z.: Constructing fixed-ratio superpeer-based overlay. In: 3rd International Conference on Computer Science and Information Technology, vol. 7, pp. 242–245 (2010)
20. Li, J.S., Chao, C.H.: An Efficient Superpeer Overlay Construction and Broadcasting Scheme Based on Perfect Difference Graph. In: IEEE Trans. Parallel Distrib. Syst., vol. 21, no. 5, pp. 594–606 (2010)
21. Tan, Y., Lin, Y., Yu, J., Chen, Z.: k-PDG-Based Topology Construction and Maintenance Approaches for Querying in P2P Networks. J. Comput. Inf. Syst., vol. 7, no. 9, pp. 3209–3218 (2011)
22. Teng, Y.H., Lin, C.N., Hwang, R.H.: SSNG: A Self-Similar Super-Peer Overlay Construction Scheme for Super Large-Scale P2P Systems. In: 17th International Conference on Parallel and Distributed Systems, pp. 782–787 (2011)
23. Sacha, J., Dowling, J., Cunningham, R., Meier, R.: Using aggregation for adaptive super-peer discovery on the gradient topology. In: 2nd International Workshop on Self-Managed Networks, Systems & Services (2006)
24. Lo, V., Zhou, D., Liu, Y., GauthierDickey, C., Li, J.: Scalable supernode selection in peer-to-peer overlay networks. In: Hot Topics in Peer-to-Peer Systems, 2005. HOT-P2P 2005. Second International Workshop on, pp. 18– 25 (2005)
25. Jesi, G., Montresor, A., & Babaoglu, O.: Proximity-aware superpeer overlay topologies. In: IEEE Trans Network Serv. Manage, vol. 4, no. 2, pp. 78–83 (2007)
26. Dumitrescu, M., Andonie, R.: Clustering Superpeers in P2P Networks by Growing Neural Gas. In: 20th Euromicro International Conference on Parallel, Distributed and Network-Based Processing, pp. 311–318 (2012)

27. Wolf, S.: On the complexity of the uncapacitated single allocation p-hub median problem with equal weights. In: Internal Report 363/07, University of Kaiserslautern, Kaiserslautern, Germany (2007)
28. Wolf, S., Merz, P.: Evolutionary local search for the super-peer selection problem and the p-hub median problem. In: Hybrid Metaheuristics Lecture Notes in Computer Science, pp. 15–27 (2007)
29. Chen, J., Wang, R.M., Li, L., Zhang, Z.H., Dong, X.S.: A Distributed Dynamic Super Peer Selection Method Based on Evolutionary Game for Heterogeneous P2P Streaming Systems. In: Mathematical Problems in Engineering, (2013)
30. Jelasity, M., Kowalczyk, W., Van Steen, M.: Newscast computing. Technical Report IR-CS-006, Vrije Universiteit Amsterdam, Department of Computer Science, Amsterdam, The Netherlands (2003)
31. Jelasity, M., Montresor, A., Jesi, G. P., Voulgaris, S.: PeerSim P2P Simulator. http://peersim.sourceforge.net/ (2009)

The 10th International Conference on Computing and Information Technology (IC²IT2014)

May 8-9, 2014

Angsana Laguna, Phuket, Thailand

http://www.ic2it.org

Computing and Information Technology Drives Societies: Theory and Applications

General Chair

Phayung Meesad, KMUTNB, Thailand

Program Committee Chair

Herwig Unger, FernUni, Germany

Program Committee Members

M.Aiello, Groningen, the Netherlands
T.Bernard, Université de Reims, France
W.Bodhisuwan, KU, TH
S.Boonkrong, KMUTNB, TH
T.Böhme, TU Ilmenau, Germany
A. Bui, Université Paris 8, France
M.Caspar, Chemnitz, Germany
D.Delen, OSU, USA
T.Eggendorfer, Hamburg, Germany
G.Eichler, Telecom, Germany
P.Guenther, HS Esslingen, Germany
R.Gumzej, Uni Maribor, Slovenia
H.C.Ha, HNUE, Vietnam
M.Hagan, OSU, USA
W.Hardt, Chemnitz, Germany
C.Haruechaiyasak, NECTEC, TH
K.Hengpraphrom, NPRU, TH
J.Kacprzyk, Polish Acad. of Sci., Poland
P.Kropf, Neuchatel, Switzerland
K.Kyamakya, Klagenfurt, Austria
B.Lent, Lent AG, Switzerland
U.Lechner, UniBw, Germany
W.Limpakorn, NECTEC, TH
N.T.Loc, HNUE, Vietnam
J.Lu, UTS, Australia
A.Mikler, UNT, USA
A.Mingkhwan, KMUTNB, TH
S.Nitsuwat, KMUTNB, TH
S.Nuanmeesri, SSRU, TH
P.Prathombutr, NECTEC, TH
A.Preechayasomboon, TOT, TH
G.Quirchmayr, UNIVIE, Austria
C.Ramirez, USL, Mexico
T.Srikhacha, TOT, Thailand
W.Sriurai, UBU, TH
W.Tang, CityU, Hongkong
T.Tilli, Telecom, Germany
D.H.Tran, HNUE, Vietnam
H.Tsai, NTU, Taiwan
D.Tutsch, Wuppertal, Germany
N.Visitpongpun, KMUTNB, TH
K.Voraratpunya, KMITL, TH
W.Warakraisawad, BU, TH
M.Weiser, OSU, USA

Call for Papers

The huge amount of raw data generated by government, industry and business, commonly known as Big Data, require more and more efficient tools to turn them into useful information and knowledge. The 10th International Conference on Computing and Information Technology provides an exchange of the state of the art and future developments in the two key areas of this process: Computer Networking and Data Mining. Behind the background of the foundation of ASEAN, it becomes clear that efficient languages, business principles and communication methods need to be adapted, unified and especially optimized to gain a maximum benefit to the users and customers of future IT systems.

The organizing partners are King Mongkut's University of Technology North Bangkok (KMUTNB), Thailand; Fern University in Hagen (FernUni), Germany; Chemnitz University of Technology (CUT), Germany; Edith Cowan University (ECU), Australia; Oklahoma State University (OSU), USA; Hanoi National University of Education (HNUE), Vietnam; Mahasarakham University (MSU), Thailand; Ubon Ratchathani University (UBU), Thailand; Kanchanaburi Rajabhat University (KRU), Thailand; Nakhon Pathom Rajabhat University (NPRU), Thailand; Mahasarakham Rajabhat University (RMU), Thailand; Rajamangala University of Technology Lanna (RMUTL), Thailand; Rajamangala University of Technology Krungthep (RMUTK), Thailand; Rajamangala University of Technology Thanyaburi (RMUTT), Thailand; Prince of Songkla University, Phuket Campus, Thailand; National Institute of Development Administration, Thailand and Council of IT Deans of Thailand (CITT).

Scope

Conference topics include (but not limited to):

- **Data Mining and Machine Learning:** Artificial Neural Network, Fuzzy Systems, Hybrid Systems, Evolutionary Computation, Knowledge Discovery, Knowledge Transfer, Knowledge Management, Decision Support, Recommender Systems, Text Mining, Web Mining, Human-Computer Interface and Image Processing.
- **Data Network and Communication:** Computer Network, Security & Forensic, Wireless & Sensor Network, Telecommunication, Mobile Ad-Hoc Network, Cloud & Grid Computing, Decentralized computing, P2P networks, P2P protocols, and Semantic P2P networks.

Important Dates

Paper Submission Deadline for Review:	**December 15, 2013**
Decision Notification:	**December 23, 2013**
Camera Ready Version:	**January 13, 2014**
Advanced Registration:	**January 20, 2014**

Paper Submission

Springer

Papers must be written in English and should describe original work in details. Each paper must follow the format, according to the instructions on the conference web site at http://www.ic2it.org or Springer web site at http://www.springer.com/computer/lncs?SGWID=0-164-6-793341-0. Each paper must be accompanied by an abstract summarizing the contribution it makes to the field. Maximum paper length for regular papers is 10 pages (A4). The manuscript can be submitted at http://www.easychair.org/conferences/?conf=ic2it2014. Each paper will be reviewed by at least three reviewers. Submission of a paper constitutes a commitment that, if accepted, at least one author must attend and participate in the conference. Accepted papers will be published in the conference proceedings by ISI Proceedings, DBLP, Ulrich's, EI-Compendex, SCOPUS, Zentralblatt Math, MetaPress, Springerlink.

Contact Information

Assoc.Prof.Dr. Phayung Meesad, IC²IT 2014 General Chair
Prof.Dr. Herwig Unger, Fern University in Hagen, IC²IT2014 Technical Program Chair
Faculty of Information Technology
King Mongkut's University of Technology North Bangkok
1518 Pracharat 1 Rd., Wongsawang, Bangsue, Bangkok 10800, Thailand
Email: pym@kmutnb.ac.th; Tel: +662 555 2000 Ext.2711, 2719, 2726; Fax: +662 555 2734