This is the accepted version of the following article: Wang, F. and Sun, Y. (2008), SELF-ORGANIZING PEER-TO-PEER SOCIAL NETWORKS. Computational Intelligence, 24: 213–233, which has been published in final form at http://onlinelibrary.wiley.com/doi/10.1111/j.1467-8640.2008.00328.x/abstract.

Self-organising Peer-to-Peer Social Networks

FANG WANG
Pervasive ICT Research Centre, British Telecom Group, Orion 1/12, Adastral Park, Ipswich IP5 3RE, UK
and
YAORU SUN*
Behavioural and Brain Sciences Centre, University of Birmingham, Birmingham B15 2TT, UK

Peer-to-Peer (P2P) systems provide a new solution to distributed information and resource sharing because of its outstanding properties in decentralisation, dynamics, flexibility, autonomy and cooperation, summarised as DDFAC in this paper. After a detailed analysis of the current P2P literature, this paper suggests to better exploit peer social relationships and peer autonomy in order to achieve efficient P2P structure design. Accordingly, this paper proposes Self-organising Peer-to-Peer Social Networks (SoPPSoNs) to self-organise distributed peers in a decentralised way, in which neuron-like agents following extended Hebbian rules found in the brain activity represent peers to discover useful peer connections. The self-organised networks capture social associations of peers in resource sharing, and hence are called P2P social networks. SoPPSoNs have improved search speed and success rate as peer social networks are correctly formed. This has been verified through tests on real data collected from the Gnutella system. Analysis on the Gnutella data has verified that social associations of peers in reality are directed, asymmetric and weighted, validating the design of SoPPSoN. The tests presented in this paper have also evaluated the scalability of SoPPSoN, its performance under varied initial network connectivity and the effects of different learning rules.

Categories and subject descriptors: C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design---network topology; C.2.4 [**Computer-Communication Networks**]: Distributed Systems---distributed applications; I.2.6 [**Artificial Intelligence**]: Learning---connectionism and neural nets; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence---intelligent agents
General Terms: Algorithms, Design
Additional Key Words and Phrases: Peer-to-peer, agents, hebbian learning, self-organising, social networks

1. INTRODUCTION

The rapid development in computer networks and telecommunications welcomes a new surge of Peer-to-Peer (P2P) networks, a kind of point-to-point communication between distributed computing nodes now widely used by millions of Internet users in applications such as Gnutella [7] and BitTorrent [4] for instant resource search and sharing. Strictly speaking, P2P is not new. The general concept can be traced back more than twenty years ago in Usenet [29] and the original Internet in which a symmetry of individual hosts were presented as equals. Due to the availability of inexpensive computing power, bandwidth and storage, modern P2P networks [26][27] have gained great popularity by taking advantage of resources at the edge of the Internet, including storage, cycle, content and human presence. They possess a number of outstanding properties that are summarised as Decentralisation, Dynamics, Flexibility, Autonomy and Cooperation (DDFAC) in this paper.

Properties of P2P networks

1. Decentralisation

A distinct feature of P2P is decentralisation. In P2P systems, peers can be both clients and servers by retrieving and providing resources at the same time. The major resource control and management tasks are decentralised from network centres, i.e., traditional servers, to the edge of the network, i.e., personal computing machines. Accompanied with this are decentralised running cost and administration work. The significant communications between peers bring together

* Corresponding author. Dr. Sun's current address is Nanjing University, Jiangsu, 210009, China. Email: yaorus@gmail.com

distributed computing nodes and resources over the world with few central control points or storages, for the purpose to avoid potential computation bottlenecks in a highly dynamic environment.

2. Dynamics

P2P networks may be one of the most dynamic systems in the world. The dynamics is omnipresent, covering nearly all aspects relevant to users, resources and computing and network conditions. For instance, users may join in or leave a P2P system at any time. Their networking facilities (e.g., computing devices) and corresponding IP addresses may change each time. Because users have full control of their resources on their own machines, they may add, modify or delete the resources with changing interests or attitudes. P2P systems are required to comply with this complex dynamics to provide proper services even in extremely unstable situations.

3. Flexibility

P2P networks offer great flexibility as their network structures are independent from the underlying organizational infrastructure, the Internet. The overlay composed of distributed peers is actually a virtual network of computers created on the fly. These computers may not be physically directly connected, but they hold references (e.g., IP addresses) of other computers to recognise their existence, and through these computers to perceive the remaining of the system. These references indicate a kind of logical link or connection between peers, directing information flow from one place to another. The topology of a P2P virtual network may develop from random peer connections or certain management protocols. An effective topology design, however, should obviously take full advantage of the flexibility of P2P systems to generate adaptable peer connections for dynamic changing resource sharing applications.

4. Autonomy

A direct benefit induced by the decentralization property of P2P systems is the autonomy granted to peers. As task management and control are shifted from network centres to personal computers, this not only reduces work load on servers, but also returns resources and choices to ordinary peers. Consequently peers have significant autonomy on what to provide and what to read. Peers can also actively choose communication channels and modify their (logic) connections to others. The autonomy supplies peers an opportunity to improve and customise their resource sharing. However, how to best exploit the autonomy to meet user targets is still a challenge, as we will discuss in the following sections.

5. Cooperation

A P2P system is a cooperative network. It will not work without peer collaboration. Frequent communications take place between cooperating peers for query sending, resource delivery, and most importantly, discovery of appropriate resources and peers for resource sharing. Although “free riding” presents a common problem in many P2P systems [2], it should be noted that those peers that do not offer any content are also valuable contributors, because they provide important

connections or routes for maintaining the distributed network as a whole. In P2P systems, the value lies in the contribution of multiple peers rather than the authority of one.

The DDFAC properties explained above distinguish modern P2P networks from other networks and make them particularly useful in large-scale distributed environments. The efficiency and effectiveness of a running P2P system, however, depend on how well those properties are exploited. In the next section, we will introduce recent work on P2P systems and how they are related to the DDFAC properties.

Review of P2P networks

Distributed peers in P2P systems need to preserve information of others peers so as to direct a search for resources. The basic information preserved usually includes a peer's IP address and port number so that messages can be sent to this peer on the Internet. More information such as the index of resources held by a peer and its network conditions can be preserved as well. A peer is called a *neighbour* of another peer if the later holds the information of the former. It is also said that the second peer holds a connection to the first one though this connection is only an abstract logical link. Obviously, the logic connections established differently will result in different P2P network topologies, which in turn establish different routes of information transportation and hence the final resource sharing results will be varied.

Current P2P systems can be classified into three types, *ad-hoc*, *fixed*, and *similarity based*, according to the way that peers are organised. In ad-hoc P2P systems, peer connections are established arbitrarily. Except a few choices such as bandwidth and client types, a peer may connect any other peers until a pre-defined number of connections are met. When a peer joins the network, it first obtains addresses of other peers to connect from permanent hosts or known peers [19]. Once the initial connections are established, more host addresses will be obtained through search for resources or by broadcasting a specific kind of message. This kind of network has high uncertainty in connectivity, but has been well applied in practice in an enormous scale, including Gnutella, Kazza [13] and Morpheus [18]. The number of hosts running on Gnutella was reported to be 1,200,000 in March 2005 [15]. These networks may be one of the few largest distributed computing systems ever, and more surprisingly, they have been running with great reliability and resilience in face of possibly the most ferocious dynamics. This may be due to the large number of users involved, the extremely flexible network structure and the power law node connectivity distribution as shown in the quantitative analysis [20]. The high flexibility and uncertainty in structure, on the other hand, result in unguaranteed resource availability, unpredicted search results and high latency especially for those with low-bandwidth connections [25]. Recent research by Yang and Garcia-Molina [33] investigated better search mechanisms in Gnutella by dynamically regulating search steps and utilising neighbour information for route selection.

In contrast to ad-hoc systems, peers in fixed P2P systems have determined connections by following well designed rules. Typical examples of this category include Chord [28], Pastry [24], Tapestry [35] and P-Grid [1]. Peers in these

systems are assigned with static identifiers. The organisation of peers is based on the distances of peer identities. Routing tables that are also built on the distances of peer identities are distributed onto some if not all of the peers. P2P systems designed in this way usually have well organised network topologies, e.g., a ring shape or a tree structure. Resource search in these networks is also relatively straightforward: a search message for a particular resource is sent across neighbouring links to the peers whose identities are progressively closer to the identity of the resource. Generally speaking, fixed P2P systems aim at high resource availability in a relatively persistent environment. They are more suitable to applications with unified resource distribution and user requirements, in which a good scalability can be obtained. This is, however, achieved at a cost of substantial maintenance, e.g., routing information and storage management [32]. The strict requirements (e.g., fixed resource/route table placement) sometimes make the fixed P2P systems difficult to deal with system dynamics and heterogeneity. Recent study showed that practical P2P systems own great heterogeneity with respect to a series of characteristics, including internet connection speeds, latencies, lifetimes, IP domains, and shared data [25]. The magnitude of these characteristics can vary between three and five orders of magnitude across the peers. When resources are unequally distributed or have varied popularity, fixed P2P systems may involve intensive traffic or information flows around certain peers while other peers remain relatively free.

The organisation of similarity based P2P networks is a method situated between ad-hoc and fixed networks. This method utilises similarity of peers or the resources they hold to route search messages while maintaining considerable network flexibility. A query message is always routed to the neighbouring peer that has the possibly greatest closeness to the requester or requested resources. Peer connections in similarity based networks are unfixed: peers are able to connect previously unknown neighbours in order to increase network connectivity. In Freenet [6], every peer involved in a search route created a new entry in its routing table to associate the resource holder after a successful search. Peers could also locally cache a copy of the resources they had transferred. A similar search strategy was adopted in Anthill [3], based on the distances of hashed resource keys. In addition, an ant-link agent was employed in Anthill to carry query messages and search in the network. In [14][21], peers connected other similar peers if they had the same defined attributes or objectives. Peers could also be linked together when they had similar access patterns to the same documents [12]. Under this situation, a kind of data-sharing graph was formed and the resulting file-sharing networks presented to be small-worlds. Wang proposed self-organising communities, to organise distributed peers into a series of communities [31]. Without sophisticated user similarity calculation, self-organising communities clustered users with similar interest or preferences when they had matching behaviour, e.g., one answered a query of another. The above works instantly formed P2P networks according to present search results. This resulted in a decentralised and self-organising P2P overlay, in which specialised nodes or routing tables in handling clusters of similar interests or resources were built.

Peers in ad-hoc, fixed and similarity based networks all work in a decentralised and cooperative manner to form P2P overlay networks. Because different strategies are applied to construct routing tables or peer neighbourhood, the resulting networks present different structures with varied dynamics, flexibility and peer autonomy. Table 1 summarises the DDFAC properties of those three types of P2P networks.

Table 1 DDFAC properties in ad-hoc, fixed and similarity based P2P networks

	Decentralisation	Dynamics	Flexibility	Autonomy	Cooperation
Ad-hoc	Yes	High	High	Low	Yes
Fixed	Yes	Low	Low	Low	Yes
Similarity based	Yes	Medium	Medium	High	Yes

Generally speaking, among ad-hoc, fixed and similarity based networks, ad-hoc P2Ps present the most flexible structure, while fixed P2Ps offer the best resource availability and possibly the best search efficiency, provided that peers and resources are relatively stable and homogeneous. Similarity based P2Ps seem to be a trade-off between ad-hoc and fixed networks. They offer adaptable connectivity in changing situations, while organising and searching resources in a relatively efficient way. This is achieved through a great exploitation of peer autonomy. That is, rather than forming network structures randomly or by following strict rules, peers self-organise themselves into a resilient structure based on peer or resource similarity.

This paper proposes a novel approach to form Self-organising P2P Social Networks (SoPPSoNs). SoPPSoN deploys distributed neuron-like agents to actively exploit peer social associations and self-organise synaptic type processes. This work draws inspirations from the biological brain in which numerous neurons cooperate in a decentralized manner to achieve specific tasks. In [11], we briefly introduced our initial work on the proposed model. This paper will give the proposed model a more comprehensive and theoretical analysis and detailed investigation by applying it to real Gnutella data. In the next section, the ideas of P2P social networks and cognitive peer agents will be addressed, which are the fundamentals of the SoPPSoN design. The proposed SoPPSoN mechanism and associated learning rules are described in Section 3. Section 4 examines SoPPSoN's performance in a series of simulated experiments by using real data collected from Gnutella. The last section concludes the paper.

2. SELF-ORGANISING P2P SOCIAL NETWORKS AND COGNITIVE PEER AGENTS – THE SoPPSoN DESIGN

Self-organisation is an important mechanism for the ease of configuration and administration of network systems, especially when the networks become large or highly dynamic [17]. The rhythm of local interaction between peers and the environment provides the fundamental information for the formation of conscious phenomena of the whole system [8]. In the self-organisation of P2P systems, there are two important issues directly related to the system design:

- What structures should peers self-organise into, or in other words, what are the suitable topologies for P2P?

- How to best develop peer autonomy so as to achieve efficient self-organisation and resource sharing in P2P networks?

This paper proposes the concepts of P2P social networks and cognitive peer agents to resolve the above questions.

P2P social networks

It has been known (see review in Section 1.2) that peers in ad-hoc and fixed P2P networks are allowed lower autonomy so they have either arbitrary or rigid connections with each other. This results in varied, unpredictable search efficiency and high maintenance cost respectively. In Freenet and Anthill, a pair of peers was linked together based on resource closeness. Interestingly, the resources might not be owned by the peers linked. Instead, the resources were originally possessed by other peers but that pair of linked peers was involved in the search and delivery of the resources. The preservation of others' resources may actually introduce harmful noises to the search of resources of a peer's own requests. When peers are purely tied according to their similarity, it is easier for them to share information with common interests. However, it becomes relatively difficult to introduce and seek resources of different kinds across peer clusters. Peers, whose owners are ordinary people, may not be connected via interests only. When people are associated by their social relationships, information propagation can be achieved within surprisingly short distances, such as the well known "six degrees of separation" phenomenon [16]. We believe P2P networks, similar to human social societies, should be constructed according to peer social relationships for a better resource sharing. That is, distributed peers may connect those that can satisfy their resource requirements and those that can provide useful (direct or indirect) links to resource providers. The formed networks capture social associations of peers in resource sharing, and hence are called ***P2P social networks***. Different from similarity based networks, peers in P2P social networks may not only be linked according to peer or resource similarity, but they may also be connected depending on a series of factors, such as their demographic information, common contacts, acquaintanceship, or the usefulness of each other (see the following explanation on peer utility). A peer association is attached with a direction and strength so peer relationships can be passed onto more than one connection. A P2P social network, therefore, owns a series of characteristics that differ itself from the most commonly studied P2P or complex networks:

- It is **directed**. This also suggests that links between peers are asymmetric. For example, while B is a useful contact of A, A may not be a contact of B at all. This is often the case in human societies and peer social networks as proved by the Gnutella data (see Section 4.1).
- It is **weighted**. This means peers do not have equal capital in resource sharing. A connection from peer A to peer B will be associated with a number, or *strength*.

- A **connection strength** indicates the **possible peer utility** of one to another, or how useful one peer is to another. This utility may reflect not only the familiarity of two peers but also a number of factors that affect the quality of services such as network bandwidth, data transmission speed, peer work load, and the value of resources delivered. A connection with a higher strength usually means that the end peer of the connection has contributed more to the resource provision for the start peer of the connection. This strength will be adjusted continuously in order to capture the actual and changing peer relationship.
- Peer connections have **adaptively decaying strengths** over time. Similar to human social networks and neural networks in the brain, nodes in P2P social networks can not only establish new contacts with unknown nodes and increase strengths for more familiar nodes, but also reduce connection strengths when connected nodes have less usefulness.

P2P social networks form peer connections by making use of peers' useful social associations. Peers that can provide resources or information about resource destinations will become a peer's neighbour or neighbours' neighbour. Accordingly resource sharing between distributed peers can be achieved easier and quicker. P2P social networks are self-organised in SoPPSoN via cognitive peer agents that autonomously and adaptively learn correct peer social relationships.

Cognitive peer agents

Better realization of peer autonomy can improve a P2P system's performance in many ways, such as less human design, more intelligent control and self-organisation in a large and distributed environment. While peers in P2P systems possess substantial potential autonomy, how to exploit peer autonomy is a challenge.

Nowadays a significant body of work implements peers as autonomous agents to deal with message processing and connection management [17] [21] or to solve different application problems [22] [34]. In these agent-based systems, peers tend to follow pre-designed and sometimes rigid rules to perform actions. These peers are more like a kind of reactive agent. While certain heuristic information was used in Anthill, Freenet and [33] to improve search efficiency, the internal states of those agents or peers did not change with experience, nor did they improve their capabilities to generate actions through learning.

In SoPPSoN, a special kind of neuron-like agent is designed to imitate the adaptive neurons in the brain. The mammalian brain, and in particular the highly developed human brain, may be the most adaptive and complex network in the world, composed of more than 10^{12} neurons and 10^{23} synapses. Surprisingly, the brain may not have a fixed functional or organisational structure, as new emerging studies have recently revealed [9]. Neural ensembles located in separate and even widely distant and unrelated regions appear to function in a decentralised and highly collaborative manner, by sprouting dendritic protrusions and establishing new synaptic terminal connections. The information flows or communications between neurons are highly diversified and modulated in nearly real time according to changes collected

from the external environment. (See [9] for more details.) Inspired by these findings, SoPPSoN deploys cognitive agents to work for individual peers. These agents apply a simple but effective synaptic-plasticity theory model, Hebbian learning [10] to implement decentralised symphonic collaboration with flexible connectivity.

In biological neural networks, learning is achieved mostly (but not exclusively) through changes in the strengths of the connections between neurons. One common way to calculate changes in neuron connection strengths is Hebbian learning rule, in which a change in the strength of a connection is a function of the pre- and postsynaptic neural activities. The proposer of Hebbian learning, Donald O. Hebb, accounted physiological evidence of behaviour into two main categories: the existence and properties of continuous cerebral activity and the nature of synaptic transmission in the central nervous system. Hebb combined these two principles to develop a theory of how learning occurs within an organism, that is, repeated stimulation of specific receptors leads slowly to the formation of "cell-assemblies", which can act as a closed system after stimulation has ceased. Generally, a Hebbian learning rule can be formulated as:

$$\Delta w_{ij} = \gamma \cdot x_i \cdot x_j$$

where x_i is the output of the presynaptic neuron, x_j the output of the postsynaptic neuron, w_{ij} the strength of the connection between them and γ the learning rate.

Within connectionism, Hebbian learning is a time-dependent cognitive process, and most importantly, an unsupervised training algorithm that autonomously learns associations between stimuli and responses. The property of unsupervised learning is vital to peer agents, as prepared data or information of the whole system is nearly impossible to obtain in an extremely dynamic and distributed P2P environment. In SoPPSoN, peer agents utilise Hebbian learning to discover useful social associations between peers and adjust association strengths based on instant search results. Accordingly a global P2P social network is formed based on local connections of pairs of peers. The next section describes the proposed SoPPSoN algorithm in detail.

3. SELF-ORGANISING PEER-TO-PEER SOCIAL NETWORK

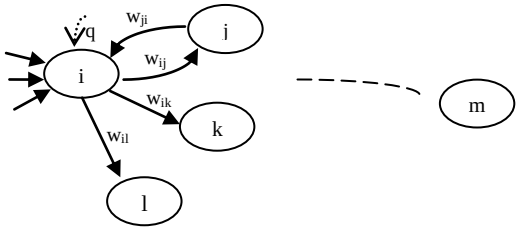


Figure 1: Illustration of the SoPPSoN network

3.1 SoPPSoN agents

Figure 1 gives an illustration of the SoPPSoN network. A peer, represented by a neuron-like agent, modifies its associations with others after receiving a stimulus (e.g., a search result). The general Hebbian learning is extended into a set of learning

rules: *frequency rule*, *feedback rule* and *symmetry rule*, to account for the different roles played by different peers during search. In addition, an *adaptive decay rule* is designed to simulate decaying connectivity over time.

Suppose that w_{ij} is the strength of an association from agent *i* to agent *j*, or from peer *i* to peer *j*, the above rules are explained as follows.

Frequency Rule

This rule is deployed when a neuron is fired in response to the firing of another neuron. In SoPPSoN, this happens when a peer *j* provides required resources to a requesting peer *i*. The strength of the connection from peer *i* to *j* is then updated according to equations (1) and (2):

$$w_{ij} = w_{ij} + \Delta w_{ij} \qquad (1)$$

$$\Delta w_{ij} = \gamma \cdot x_i \cdot x_j \qquad (2)$$

where γ is the learning rate. x_i is the output of peer *i* that sends a request for a particular resource. x_i has a value of 1 from time *t* when the request is sent to time *t+T*, and a value of 0 otherwise. x_j is the output of peer *j* replying to the request sent. If peer *j* provides search results within a period of time *T*, the output x_j is 1 from time *t'* when peer *j* sends the results to peer *i*, to time *t'+T*. Otherwise, x_j has a value of 0. The learning rate is defined as:

$$\gamma = \begin{cases} q \cdot (1 - \dfrac{\tau}{T+1}) \cdot \sigma & \tau \le T \\ 0 & \tau > T \end{cases} \qquad (3)$$

where τ is the time spent for searching a resource and *T* indicates the maximum time period allowed for the search. σ is a parameter to fine tune the learning rate γ: the larger σ is, the bigger the weight w_{ij} will change. $q \in [-1,1]$ indicates the value or quality of the resource provided. It is examined by the receiving peer *i*. The connection strength will be reduced if *q* is negative, that is, the resource provided is irrelevant to the search. A new connection will be constructed from peer *i* to *j* if they have no link before and the initial strength of the connection is:

$$w_{ij} = \Delta w_{ij} \qquad (4)$$

Feedback rule

When a resource search involves more than one peer, all the peers contributing to the search will update their connections involved, after the original requesting peer evaluates the resource provided. The evaluation information works as a feedback to reinforce the strengths of the connections involved. The feedback may be positive or negative depending on whether the resource provided is useful. Equation (5) describes the feedback rule:

$$w_{kl} = w_{kl} + \eta_f \cdot \Delta w_{ij} \qquad (5)$$

where η_f is a constant, *k* is any peer involved in the search except peer *i* which proposes the request (see Figure 2). *l* is the peer to which the search request was forwarded by peer *k*.

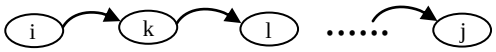


Figure 2 Example of a search route

Symmetry Rule

This rule is based on the assumption that if peer *i* is related to peer *j*, then peer *j* is possibly related to peer *i*. The symmetry rule is triggered to enforce the inverse link w_{ji} after w_{ij} is updated. This rule is described as follows:

$$w_{ji} = w_{ji} + \eta_s \cdot \Delta w_{ij} \qquad (6)$$

where η_s is a constant, whose value is no more than η_f in the feedback rule, because the link from peer *j* to peer *i* is only an inference from the actual connection from peer *i* to peer *j*.

If the feedback rule or symmetry rule implies adjustment of a weight for an association that does not currently exist, a new entry is added accordingly.

Adaptive decay rule with deviation detection

The strength of a connection from one peer to another will decay over time in the absence of any stimulus (i.e., lack of contribution) from the end peer. The decay rate can be simulated as the leakage current of a capacitance (Equation 7) or simply a linearly regressive function (Equation 8):

$$w_{ij}(s+1) = w_{ij}(s) \cdot \exp(-\frac{s}{\eta_d}) \qquad (7)$$

$$w_{ij}(s+1) = w_{ij}(s) - \eta_d \cdot s \qquad (8)$$

where η_d is the decay rate and *s=0* is the time measured from the point at which the last update in accordance with Equation (1), (5) or (6) occurred.

Figure 3 gives an illustration of the linear decay of a connection strength when $w_{ij}(s=0)$ is 0.01, η_d is 0.1 and *s* is sampled every million second. A connection is removed from the SoPPSoN network when its strength is decreased to zero or a very small value ε (e.g., ε=0.001).

From the above description, we can see that η_d is a very sensitive parameter that determines the decay degree of a peer connection. If η_d is set too high, it is possible that the network is disconnected before necessary associations are

established between peers. On the other hand, a too low value of η_d will take longer time to remove unnecessary peer associations. η_d therefore is one of the factors that determine the convergence rate of SoPPSoN. As the size of a peer network is usually very dynamic and unpredictable, which may range from tens to millions, it is difficult if not impossible to decide a unique value of η_d to suit all P2P networks.

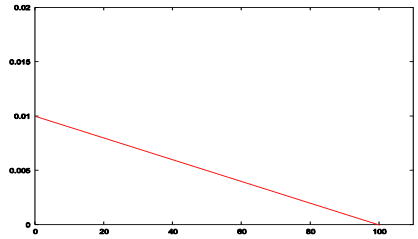


Figure 3: An illustration of the linear decay rule

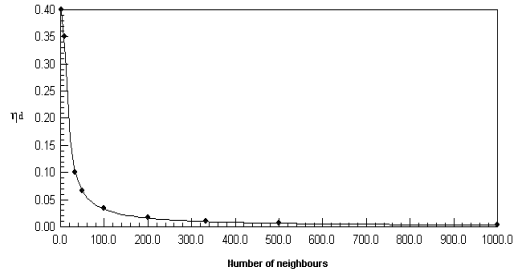


Figure 4: Adaptive η_d (a≈2.73 b≈0.54 c≈-0.31)

In order to implement autonomic and scalable P2P social networks, SoPPSoN agents adopt an adaptive decay rate η_d that adapts to varied network sizes. This adaptation is achieved in a decentralised way, only based on the local information of each agent, i.e., the number of neighbours the agent currently connects to:

$$\eta_d(i) = \frac{n_i}{a + bn_i - cn_i^2} \qquad (9)$$

where n_i is the number of present neighbours of agent *i*. So peers with fewer neighbours will have a smaller η_d and hence a faster decay rate, whereas peers with more associations will have a slower decay rate so that they will have more time to find useful connections. If an agent establishes more connections, the decay rate for its connections will become slower and vice versa. While all SoPPSoN agents adapt their connections collectively, the whole system will stabilise to a suitable structure within a reasonable time, despite the network size. Figure 4 illustrates the adaptive η_d.

In addition to adaptive η_d, a *deviation strategy* is designed in the adaptive decay rule in order to accelerate convergence. This strategy is especially helpful when a number of valuable associations have been discovered so that unuseful associations with comparatively low strengths can be removed from the neighbourhood of a peer. Classical methods to detect outliers in a set of data include Inter-Quartile-Range (IQR) computation, Grubb test and Z- or Q-test. In SoPPSoN, an association w_{ij} will be removed if there exists a neighbour *l* and the quotient w_{il} / w_{ij} is larger than a value κ, that is,

$$w_{ij} = 0 \quad if\ \exists l\ (l \neq j) \quad w_{il} / w_{ij} > \kappa \qquad (10)$$

The combination of the adaptive decay rate η_d and deviation detection helps SoPPSoN to work with varied network scales while maintaining a reasonable convergence rate, as verified in the following experiments.

3.2 Resource search on SoPPSoN

By following the learning rules introduced above, SoPPSoN agents at distributed locations will form a time-varying, weighted network whose connections indicate peer correlations or utility to each other. A wide range of search protocols can be applied on SoPPSoN to search for desired resources, such as those used in Gnutella, Anthill and [33]. In this paper, we introduce a simple but effective search technique as an example to exploit weighted SoPPSoN networks.

Suppose a peer *i* initiates a request for a resource *x*. Peer *i* will first check its neighbouring peers to see if any of them possesses the requested resource. This can be achieved by sending the request directly to the neighbours for an immediate answer or by inspecting the index of neighbours' resources if peer *i* maintains such information.

If the requested resource is owned by some of the neighbours, the resource will be sent from the neighbours to peer *i* so the search is completed successfully. Otherwise, peer *i* will select one or more neighbours to forward the request for a wider search. The selection of neighbouring peers is based on their varying strengths from peer *i* by following certain selection rules. The neighbouring peers can also compete for the request. For simplicity we use *roulette-wheel* selection [5] to choose neighbouring peers, so that those peers to which peer *i* has stronger weighted connections will have a higher probability of being selected. After receiving the request, a neighbouring peer *j* checks with its own peer associations to see if any has the required resource. If any neighbour has, the resource will be sent back by the neighbour to peer *i* and the search is terminated. If the neighbours of peer *j* do not have the required resource either, peer *j* will forward the request to some of its own contacts. Similarly, peers receiving the request will examine their neighbours first and if the required resource is unavailable, they will continue the search by forwarding the request to one or more of their neighbours. This process is repeated until the required resource is found (indicating a search success) or a 'stop' criterion (e.g., the maximum number of hops is reached) is met (indicating a search failure). Once the search is finished, the strengths of the peer connections involved during the search will be updated according to the learning rules introduced above.

3.3 Summary

From the above description, we can see that the network connectivity in SoPPSoN is dynamically adjusted according to information discovered during run time, based on the quality of services received. Peer connections playing different roles in a resource sharing task are updated respectively depending on their actual contributions. The network formed by SoPPSoN represents a self-organising autonomic P2P network in which peer agents autonomously recognise and manage their own social associations. The differing strengths of peer associations indicate the possible utility of the peers and they are continuously corrected in response to changing services provided. As SoPPSoN discovers and

exploits social relationships between peers in resource sharing, the search for resources can be achieved more efficiently and effectively without a global control. Figure 5 shows the work flow of SoPPSoN.

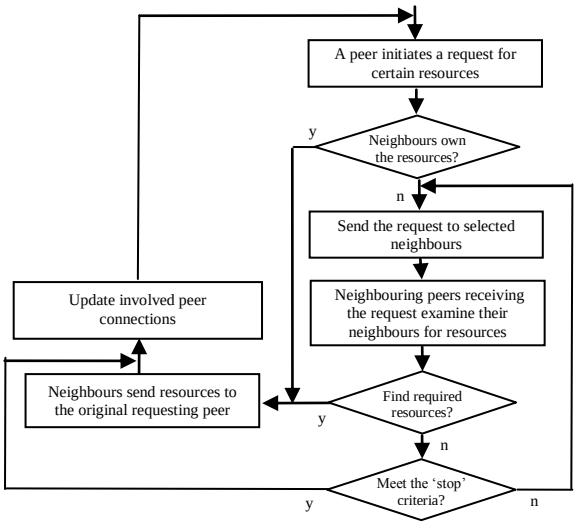


Figure 5: Work flow of SoPPSoN

4. EVALUAITON AND DISCUSSIONS

This section introduces the experiments carried out to verify the SoPPSoN mechanism. As a learning algorithm, a primary function of SoPPSoN is to learn correct associations between peers, or in other words, to converge to a proper P2P social network after learning. The peer associations discovered from the Gnutella system were applied for SoPPSoN to learn. In particular, this section examines the following major issues:

- Correctly formed peer social networks and improved search results;
- Learning scalability with respect to varied network scales;
- Convergence under different initial network connectivity;
- Effects of various learning rules in the SoPPSoN self-organisation.

Test data collected from Gnutella

We collected real user data from Gnutella to do the tests. An experimental machine running revised Gnucleus, a kind of Gnutella client, joined the Gnutella network as a super-node, so that it could be connected by more normal peers and many other super-nodes each of which was also connected by hundreds of normal peers. In order not to disturb the actual social links between peers, our experimental node did not provide any shared contents nor send queries for resources. It acted as a pure monitor to record the traffic passing through it. Particularly it recorded information relevant to the tests, such as which peer answered a query of which peer, indicating that the former may be a useful contact to the later.

The experimental Gnuclues node ran on the Gnutella network many times from 5 hours to 3 days and collected data involving 1,000 to 180,000 peers. If a peer answered a query of another peer, there was a social link from the later to the former. Such social links between peers were discovered from the raw data collected. Figure 6 shows a P2P social network with 3257 peers. The picture was created by Repast, a multi-agent simulation platform [23]. Red icons in this picture indicate individual peers and green links represent peer social connections. The little green box at one end of a link means that this link is directed towards the peer nearer the box. The topology of P2P social networks such as the one shown in Figure 6 presents a small world network, as analysed in our recent work [30]. Most of the peers in the data collected had only one inward edge (see Figure 6), suggesting that most peers were only answerers and they never initiated any request during the running time. Only a few peers acted as both requesters and answerers, playing a major role in P2P social networks. These peers are called *major peers*. Figure 7 shows a picture of a social network of major peers in one collection. In this picture, 94 major peers are discovered from more than 33,000 peers.

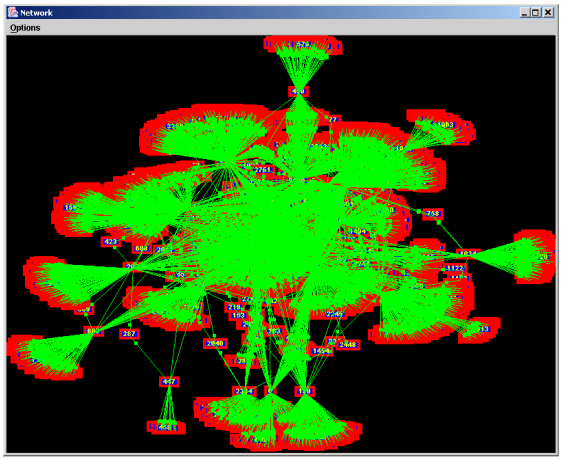


Figure 6: A social network of 3257 Gnutella peers

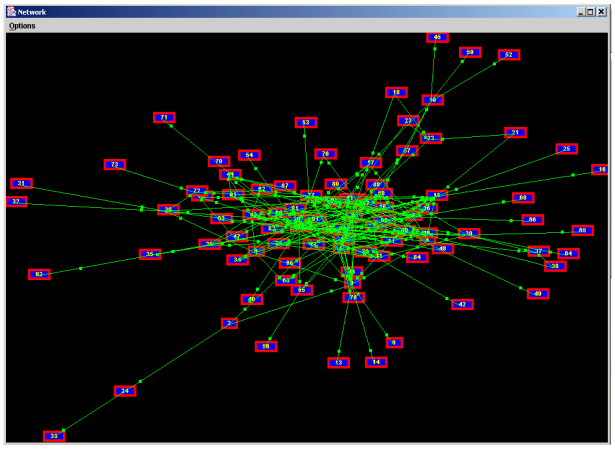


Figure 7: A social network of major peers that act as both requesters and answerers (94 major peers discovered from 33,000 peers)

The social relationships between major peers of Gnutella compose example P2P social networks for SoPPSoN to learn. Learning on large-scale, full sized P2P social networks will be presented in our future work. Table 2 summarises the social networks of major peers used in the tests. The sizes of those social networks ranged from 93 to 960 with 242 to 3282 connections. In these networks, there were only 4~32 symmetric connections, that is, only 4% ~ 12% peers had mutual connections between each other. Peer degrees ranged from 0 to 71 and the connection strengths varied from 1 to 157, suggesting that some peers answered up to 157 queries of another peer. These results suggest that *P2P social networks in reality are directed, asymmetric, and weighted*. (A detailed analysis of P2P social networks was presented in [30].)

Table 2: Test data sets of major peer social networks collected from Gnutella

	Peer numbers	Connection numbers	Pairs of symmetric links	Peer out-degree	Peer in-degree	Connection strengths
Data 1	93	242	4	0 ~ 39	0 ~ 14	1 ~ 111
Data 2	221	666	6	0 ~ 50	0 ~ 25	1 ~ 102
Data 3	256	1893	32	0 ~ 58	0 ~ 36	1 ~ 157
Data 4	287	1362	16	0 ~ 65	0 ~ 65	1 ~ 136
Data 5	459	1470	7	0 ~ 60	0 ~ 29	1 ~ 81
Data 6	960	3282	26	0 ~ 71	0 ~ 44	1 ~ 65

4.2 Experimental setup

SoPPSoN was applied to learn the various social networks of major peers summarised in Table 2. In the learning of each data set, a network with the same number of peers (or SoPPSoN agents) of the data set was bootstrapped with random connections. The initial network connectivity ranged from 0.1 to 1 and the initial strengths η_0 of connected links were set as 0.1. SoPPSoN agents were required to find correct peer associations as defined in the data set, which hold their desired resources (represented by a hash table of strings). At each step, every agent initiated a query for a resource and this request was sent to its current neighbouring peers. In this paper, we assume that resources requested in the experiment are all stable and available, held by one or more peers in the network, though in real world the types and instances of resources that people want to access can change quite frequently over time. The dynamics of both resources and peer participation (e.g., churn) will be our main research topics at the next stage.

After each resource search, simulated SoPPSoN agents in the experiments should modify their associations and association strengths based on the search results according to the learning algorithm introduced in Section 3. Every test was repeated 10 times and the results shown below were all averaged on 10 runs with 95% confidence intervals. Table 3 lists the parameter values used in the experiments. The value of T was set as 6, based on the "six degrees of separation" theory, while others were obtained through experiments.

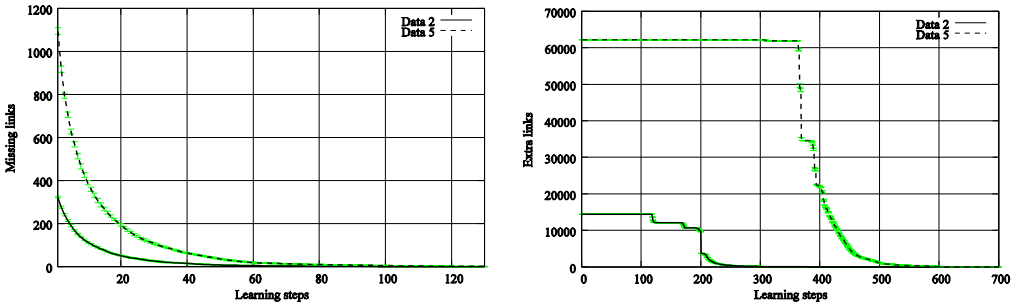
Table 3: Parameters used in experiments

Parameter	Value
T	6 hops
σ	0.25
η_f	0.25
η_s	0.05
ε	0.001
κ	10^{-8}

4.3 Correctly formed peer social networks and improved search results

The first test examined SoPPSoN's functionality to form correct peer social networks and to improve search results. In addition to the adaptive decay rule, only frequent rule and feedback rule were used. (Tests on learning with different rules are introduced in later sections.) Similar results were obtained when the initial network connectivity was set between 0.1 and 1, while SoPPSoN successfully learned all of the social networks contained in the data sets. SoPPSoN discovered correct peer associations through continuous search for resources and removed unuseful peer connections at the same time. Figure 8 shows the network formation results obtained on Data sets 2 and 5 when the initial network connectivity was set as 0.3. The reduced number of missing links (shown in Figure 8(a)) is due to the use of Hebbian-

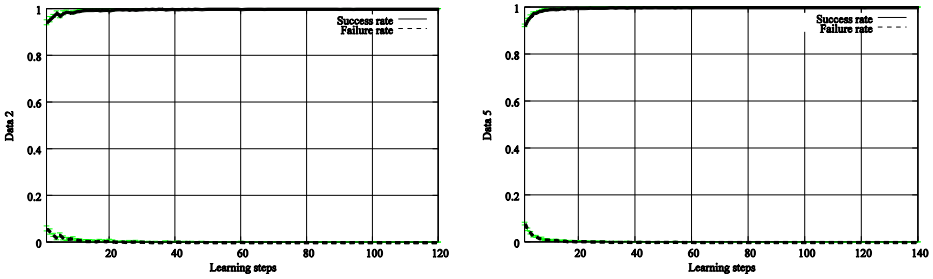
like learning rules, while the adaptive decay rule is mainly responsible for the reduced number of unnecessary links (shown in Figure 8(b)).



(a) (b)

Figure 8: Correctly formed peer social networks

As peers gradually identified their useful associations and directly connected them as neighbours, the search for resources became easier and more successful, though there was no central control or global information present in the system. Figure 9 shows that the success rate of resource search was improved from nearly 92% to 99% within 20 steps for learning on Data sets 2 and 5. The failure rate of search was accordingly dropped to nearly 0 at the same time. The success rate reached an optimal (100%) as all missing links were discovered within an average of 104 steps on Data set 2 (shown in Figure 9 (a)) and an average of 113 steps on Data set 5 (shown in Figure 9 (b)).



(a) (b)

Figure 9: Improved search success rate

In addition, because peers that hold resources of interest to other peers were eventually included in the contacts of those peers, the number of hops required for resource search was limited to 1 when peer social networks were correctly formed, as shown in Figure 10. The average length of search routes in Gnutella was around 2.31 on Data set 2 and 2.8 on Data set 5. This is at a similar level to the initial SoPPSoN networks that were constructed randomly, again suggesting that peer connections in Gnutella were random.

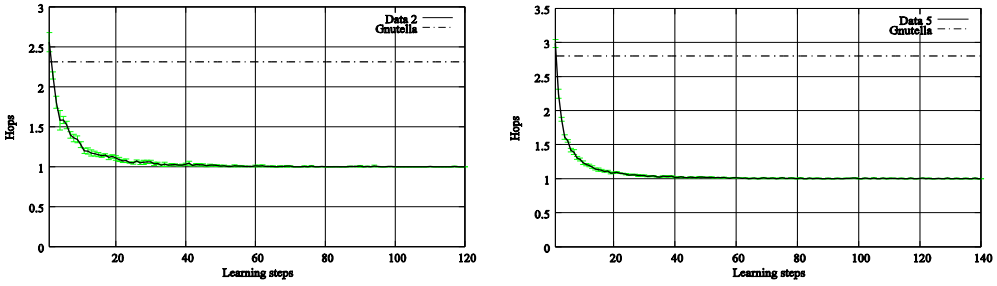


Figure 10: Reduced search hops in SoPPSoN

4.4 Learning scalability

We tested the learning scalability of SoPPSoN on various sizes of networks by using Data sets 1~6. The number of peers in these data ranged from 93 to 960. A SoPPSoN learning procedure is said converged if the correct social network structure is formed with no missing or extra peer connections, that is, no more or less connections. Figure 11 shows the results of SoPPSoN learning on those data sets when the initial network connectivity was 0.2 and 0.3 respectively. The relationship between learning steps and network sizes appeared to be approximately linear. The bump appeared on Data 4 when the network size was 287 was probably due to the relatively fewer connections between network nodes, which made learning take longer time to construct the whole network. In general, we can say that, the number of learning steps required for convergence gracefully increased as the number of peers grew, because the learning rules and particularly the adaptive decay rule are designed to adapt to various network scales. To investigate its scalability further, we are interested in examining SoPPSoN on much larger networks (e.g., with millions of peers) and a theoretical analysis of SoPPSoN will be presented in our future work.

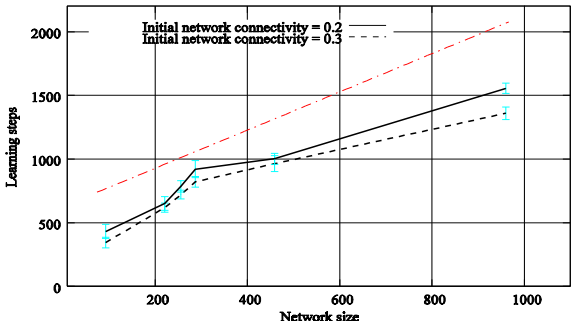


Figure 11: Scalability of SoPPSoN

4.5 Convergence under different initial network connectivity

The initial network setup obviously will affect the following formation of peer social networks evolving from original peer connections. An essential factor in the initial network setup is network connectivity that defines network density or connection frequency of a peer to others. Figure 12 shows the convergence rate of SoPPSoN learning on data set 2, as the initial network connectivity ranged from 0.1 to 1. When the initial network was very sparse, that is, peers had few connections to each other, some learning procedures could not converge. Usually, the networks became stable with a few missing links that could not be found in limited learning steps. Figure 12 shows the learning steps when SoPPSoN learning became stable, while Table 4 lists the results of learning that could not converge in 5000 steps.

From Table 4, it can be seen that unconverged learning happened when the initial connectivity was 0.1 or 0.2. In ten learning procedures, no learning starting from 0.1 network connectivity could obtain the correct peer social network. The missing links were up to 28, nearly 4% of the correct peer associations. Learning under connectivity 0.2 was better with half learning procedures converged, while the other half unconverged learning usually had only one missing link. The reason for unconverged learning is mainly due to the too sparse and sometimes separated initial

networks or a few relatively isolated peers that made the search of them difficult. When the initial network was well connected (i.e., connectivity ≥ 0.3), all SoPPSoN learning was performed successfully.

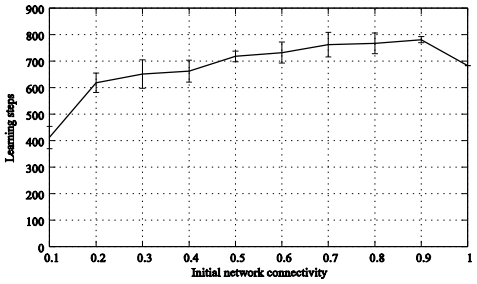


Figure 12: SoPPSoN learning under varied initial network connectivity

Table 4: Unconverged learning affected by initial network connectivity

Initial network connectivity	0.1	0.2	0.3 ~ 1.0
Unconverged learning procedures in 10 runs	10	5	0
Average missing links in unconverged learning	27.9	1	0

4.6 Effects of learning rules in SoPPSoN self-organisation

The experiments in the above sections only utilised the Frequency and Feedback learning rules. Use of different learning rules will result in different results. Figure 13 shows the time (learning steps) required when learning became stable (i.e., the network cannot be improved anymore via learning), when the Frequency rule, the combination of Frequency and Feedback rules, and all of the Frequency, Feedback and Symmetry rules were used during the learning of Data set 2. Table 5 summarises the results of those unconverged learning procedures.

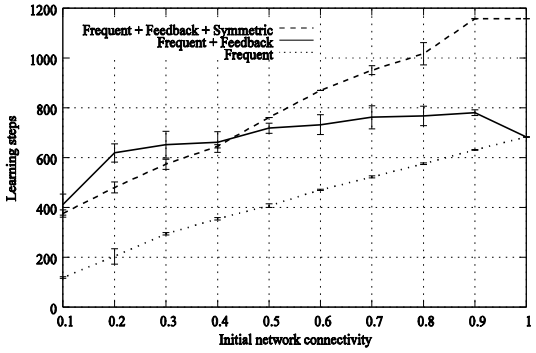


Figure 13: Effects of various learning rules

Table 5: Unconverged learning affected by various learning rules

Initial network connectivity		0.1	0.2	0.3 ~ 1.0
Frequent rule only	Learning procedures with missing links	10	10	0
	Unlearned links	-42.3	-1.3	0
Frequent + Feedback	Learning procedures with missing links	10	5	0
	Unlearned links	-27.9	-1	0
Frequent + Feedback + Symmetry	Learning procedures with missing links	8	0	0
	Unlearned links	-2.65 +507.4	+507.5	+507.7

In all learning procedures, learning with the Frequency rule required the least learning time when the network became stable. The integrated use of Frequency, Feedback and Symmetry rules together showed slightly better learning performance when the initial network connectivity was lower than 0.5 and worse performance when the connectivity was equal to and above 0.5, compared with the use of Frequency and Feedback rules. On the other hand, learning with Frequency rule only performed worst on sparse networks. It could not make any success in convergence in 10 runs when the initial network connectivity was below 0.2. Meanwhile, the missing links (indicated by negative numbers in Table 5)

were up to 42.3, nearly 6.4 percent of the correct peer associations. Learning with all rules (Frequency, Feedback and Symmetry) together was more successfully in discovering correct peer associations. Even when the initial network connectivity was 0.1, it only missed an average of 2.65 associations in the whole social network. However, this was at a cost of maintaining 507 extra peer links (indicated by the positive numbers in Table 5).

As the Frequency rule preserves correct peer associations after search, the function of Feedback and Symmetry rules is mainly to maintain extra links so as to help the following search and discovery of other useful connections. These two functions, however, operate in two different ways. The former enhances the strengths of existing links that have made contributions to previous search, while the latter establishes sometimes absolutely new links. Because peer associations often present asymmetric (see Section 4.1), the Symmetric rule will sustain redundant peer links in the network. These links, however, are useful for maintaining good network connectivity, especially in sparse networks.

5 CONCLUSIONS AND FUTURE WORK

Modern Peer-to-Peer systems have gained great popularity in distributed information and resource sharing. This is mainly due to their outstanding DDFAC properties, including decentralisation, dynamics, flexibility, autonomy and cooperation. The topology design of conventional P2P networks, however, usually does not take into consideration of peer proximity or relationships and hence cannot achieve efficient resource sharing in dynamic and distributed environments.

This paper proposes Self-organising Peer-to-Peer Social Networks (SoPPSoN) as a novel solution to form efficient P2P networks. Specifically, SoPPSoN has two important features:

- SoPPSoN constructs *P2P social networks*, in which peers identify and establish useful associations for their resource sharing. The strength of a connection reflects the possible utility of the end peer of the connection to the start peer of the connection, that is, how useful the former is perceived by the latter. A P2P social network is directed, asymmetric and weighted.
- SoPPSoN employs neuron-like agents to learn correct social associations of peers. These agents make use of extended Hebbian rules found in the brain activity to establish useful links to other agents and modify the connection strengths according to continuous search results. In addition, an adaptive decay mechanism is designed to simulate the decay of connection strengths over time. This reflects the less familiarity of two peers if no contact is presented from one to another for a long time.

SoPPSoN self-organises P2P social networks in an autonomous and decentralised way. Because it utilises social associations of peers to construct dynamic network connections, the resulting P2P systems have improved resource sharing results even under varied network conditions. The learning performance of SoPPSoN has been examined by a set of experiments by using real peer social networks collected from the Gnutella system. Further

investigation of SoPPSoN will include its learning on large-scale networks and the system's dynamic behaviour under resource change and churn, i.e., as peers join and leave the network constantly. This paper has tested the system with 93 to 960 peers. However, in order to verify the scalability and robustness of the proposed mechanism, tests should be undertaken with much more numbers of nodes, e.g., on the original peer networks collected from the Gnutella data, rather than the major peer networks only. In this paper, we assumed that the resources available in various network nodes were fixed but this is not always true in practice. In a real world, a node that did not have a resource at one time may have the resource at a later time and vice versa. While the learning of current SoPPSoN agents is more biased towards the past knowledge, which helps the search for a new query similar to the past ones, it should be extended to quickly catch the changes in resources and peers and accordingly modify the routes to available resources in a reasonable time. By doing so, the peer agents will be more adaptive and autonomous and hence have stronger capabilities to survive in dynamic and distributed P2P environments.

In the future, we are also interested to examine the proposed SoPPSoN in comparison with other related P2P systems such as Freenet, Anthill and [33] to see whether an improvement on network performance could be achieved, including reduced query latency and overhead. The P2P social networks formed by SoPPSoN are more concerned with the first layer of neighbourhood at the moment. The following study of SoPPSoN will investigate the social effects of indirect neighbours and the neuron-like agents will have enriched internal states and learning capability to make use of those social effects for more effective resource sharing.

This paper presents the first step towards the study on self-organising P2P social networks. It is clear that a better acknowledgement of peer social networks will provide more useful guidelines for the design of P2P systems. Future applications of P2P systems will not be limited to (music) file transfer or resource search only. P2P has potential applications in ad-hoc and sensor networks too as these networks are also highly dynamic and distributed. SoPPSoN should be able to contribute to these applications, by deploying autonomous peers to recognise and predict useful connections and to manage resilient peer associations in a mobile environment.

ACKNOWLEDGEMENTS

The authors are grateful to Di Liu for his work on Gnutella data collection. The author, Yaoru Sun, thanks the support of National Natural Science of Foundation of China (NSFC No. 60775019). The authors are grateful to the anonymous reviewers and the editors for their constructive comments and suggestions.

REFERENCES

[1] Aberer, K., Cudrě-Mauroux, P., Datta, A., Despotovic, Z., Hauswirth, M. and Punceva, M. 2003. P-Grid: a self-organizing structured P2P system. *ACM SIGMOD Record,* 32(3), 29-33.

[2] Adar E. and Huberman, B.A. August 2000. Free riding on Gnutella, Technical report, Xerox Parc.

[3] Babaoglu, O., Meling, H. and Montresor, A. 2003. Anthill: a framework for the development of agent-based peer-to-peer systems, *Proc. of the 22nd Int. Conf. on Distributed Computing Systems*, Vienna, Austria.

[4] http://www.bittorrent.com

[5] Blickle, T. and Thiele, L. 1996. A Comparison of Selection Schemes Used in Evolutionary Algorithms. *Evolutionary Computation*, 4(4):361-394.

[6] Clarke, I. and Sandberg, O. 2000. Freenet: a distributed anonymous information storage and retrieval system, *ICSI Workshop on Design Issues in Anonymity and Unobservability*, California, 25-26.

[7] http://www.gnutella.com

[8] Gonzalez, E., Broens, M. and Haselager, P. 2004. Consciousness and agency: the importance of self-organized action, *Network* 3-4, 103-113.

[9] Hanwehr, R.V. 2002. Information fusion in the human brain: the complexity of de-centralized symphonic collaboration, in Hyder, A.K. E. Shahbazian and E. Waltz, (eds.) *Multisensor Fusion*, Springer London, 1-36.

[10] Hebb, D.O. 1949. *The Organization of Behaviour*. John Wiley & Sons, New York.

[11] Ghanea-Hercock, R., Wang, F. and Sun, Y. 2006. Self-Organising Adaptive Peer-to-Peer Agent Systems. *IEEE Transactions on Systems, Man, Cybernetics (Part B)*, 36(6):1230-12360.

[12] Iamnitchi, A., and Ripeanu, M. and Foster, I. 2004. Small-world file-sharing communities. *The 23rd Conference of the IEEE Communications Society (InfoCom 2004)*, Hong Kong.

[13] http://www.kazza.com

[14] Khambatti, M., Ryu, K.D. and Dasgupta, P. 2004. Structuring peer-to-peer networks using Interest-based communities. *First International Workshop on Databases, Information Systems and Peer-to-Peer Computing, Berlin, Germany*, 48-63.

[15] http://www.limewire.com/english/content/netsize.shtml

[16] Milgram, S. 1967. The Small World Problem. *Psychology Today*, 2:60-67.

[17] Moore, M. and Suda, T. 2002. A decentralized and self-organizing discovery mechanism. *Proc. of the First Annual Symposium on Autonomous Intelligent Networks and Systems*, UCLA, US.

[18] http://www.morpheus.com

[19] Rea, S. and Pesch, D. 2004. Aperiodic cache provisioning for new network nodes in wireless mobile ad hoc networks, *IEEE Vehicular Technology Conference*, CA.

[20] Ripeanu, M. 2001. Peer-to-peer architecture case study: Gnutella network, *Proceedings of the First International Conference on Peer-to-Peer Computing*, Sweden, 99-100.

[21] Ogston, E., Overeinder, B., van Steen, M. and Brazier, F. 2003. Group formation among peer-to-peer agents: learning group characteristics. *Second International Workshop on Agents and Peer-to-Peer Computing*, Melbourne Australia, 59-70

[22] Purvis, M., Nowostawski, M., Cranefield, S. and Oliveira, M. 2003. Multi-agent interaction technology for peer-to-peer computing in electronic trading environments. *Second International Workshop on Agents and Peer-to-Peer Computing*, Melbourne Australia.

[23] http://repast.sourceforge.net/

[24] Rowstron, A. and Druschel, P. 2001. Pastry: scalable, decentralized object location and routing for large scale peer-to-peer systems. *Proceedings of 18th Conference on Distributed Systems Platforms.* Heidelberg.

[25] Saroiu, S., Gummadi, P.K. and Gribble, S.D. 2002. A measurement study of peer-to-peer file sharing systems. *Proceedings of Multimedia Computing and Networking*, San Jose, California.

[26] Schoder, D. and Fischbach, K. Core Concepts in Peer-to-Peer (P2P) Networking. In: Subramanian, R.; Goodman, B. (eds.): *P2P Computing: The Evolution of a Disruptive Technology*, Idea Group Inc, Hershey. 2005

[27] Steinmetz, R. and Wehrle K. (Eds). Peer-to-Peer Systems and Applications. *Lecture Notes in Computer Science*, Volume 3485, September 2005.

[28] Stoica, I., Morris, R., Karger, D., Kaashoek, M.F. and Balakrishnan, H. 2002. Chord: a scalable peer-to-peer lookup service for internet applications, *SIGCOMM*, 149-160.

[29] http://www.usenetnewsgroup.net/

[30] Wang, F., Moreno, Y. and Sun, Y. 2006. The structure of peer-to-peer social networks, *Physical Review E* 73, 036123 (2006).

[31] Wang, F. 2002. Self-organising communities formed by middle agents. *Proceedings of the First International Conference on Autonomous Agents and Multi-Agent Systems*, Bologna, Italy, 1333-1339.

[32] Xu, Z., Min, R. and Hu, Y. 2003, Reducing maintenance overhead in DHT based peer-to-peer algorithms. *Proceedings of the 3rd International Conference on Peer-to-Peer Computing*, Sweden, 218-219.

[33] Yang, B. and Garcia-Molina, H. 2002. Improving search in peer-to-peer networks. *Proceedings of 22nd International Conference on Distributed Computing Systems*, Vienna, Austria, 5-14.

[34] Yu, B. and Singh, M.P. 2003. Incentive mechanisms for agent-based peer-to-peer systems. *Second International Workshop on Agents and Peer-to-Peer Computing*, Melbourne Australia.

[35] Zhao, B., Huang, L., Stribling, J., Rhea, S.C., Joseph, A.D. and Kubiatowicz, J.D. 2003. Tapestry: A resilient global-scale overlay for service deployment. *IEEE Journal on Selected Areas in Communication* 22(1), 41-53.