Neighbor Selection in Peer-to-Peer Overlay Networks: A Swarm Intelligence Approach

Hongbo Liu[1], Ajith Abraham[1,2,3] and Youakim Badr[3]

[1] School of Computer Science and Engineering, Dalian Maritime University, 116026 Dalian, China. lhb@dlut.edu.cn, http://hongboliu.torrent.googlepages.com/

[2] Norwegian Center of Excellence, Center of Excellence for Quantifiable Quality of Service, Norwegian University of Science and Technology, Trondheim, Norway ajith.abraham@ieee.org, http://www.softcomputing.net/

[3] National Institute of Applied Sciences
INSA de Lyon, F-69621, France
youakim.badr@insa-lyon.fr

Summary. Peer-to-peer (P2P) topology has a significant influence on the performance, search efficiency and functionality, and scalability of the application. In this chapter, we investigate a multi-swarm approach to the problem of Neighbor Selection (NS) in P2P networks. Particle swarm share some common characteristics with P2P in the dynamic socially environment. Each particle encodes the upper half of the peer-connection matrix through the undirected graph, which reduces the search space dimension. The portion of the adjustment to the velocity influenced by the individual's cognition, the group cognition from multi-swarms, and the social cognition from the whole swarm, makes an important influence on the particles' ergodic and synergetic performance. We also attempt to theoretically prove that the multi-swarm optimization algorithm converges with a probability of 1 towards the global optima. The performance of our approach is evaluated and compared with other two different algorithms. The results indicate that it usually required shorter time to obtain better results than the other considered methods, specially for large scale problems.

Key words: P2P Swarming Networks, Neighbor Selection, Particle Swarm, Genetic Algorithm, Undirected graph.

1 Introduction

Peer-to-peer computing has recently attracted great interest and attention of the computing industry and gained popularity among computer users and their networked virtual communities [1, 2], since it allows the implementation of large distributed repositories of digital information. Many peer-to-peer systems also have emerged as platforms for users to search and share information over the Internet [3]. In essence, a peer-to-peer system can be characterized as a distributed network system in which all participant computers/nodes have symmetric capabilities and responsibilities. In the system, numerous nodes of equal roles are connected through an arbitrary network and exchange data or services directly with each other. All participants in a peer-to-peer system act as both clients and servers to one another, thereby surpassing the conventional client/server model and bringing all participant computers together with the purpose of sharing resources such as content, bandwidth, CPU cycles [4]. Peer-to-peer networks are applied to many fields, which includes communication and collaboration, distributed computing, Internet service support, database system, and content/data distribution, even service platform for public welfare (e.g. providing processing power to fight cancer) [5, 6, 7, 8, 9, 10]. More specifically, P2P file sharing systems set up a network or pool of peers on Internet and provide facilities for searching and transferring files between them. Since these systems provide a economical platform for data-sharing that is highly scalable and robust, a great number of commercial and academic projects have been developed using this technology. However, it is reported in a recent survey that Peer-to-Peer applications generate one-fifth of the total Internet traffic, and it is believed that it will continue to grow [11, 12].

In pure P2P systems, individual computers communicate directly with each other and share information and resources without using dedicated servers. A node cannot realistically keep the addresses of all other peers, so an overlay network need be constructed where each node keeps addresses of a few other peers (called its neighbors) at the application level. These connections may be directed, may have different weights and are comparable to a graph with nodes and vertices connecting these nodes. Defining how these nodes are connected affects many properties of an architecture that is based on a P2P topology, which has a significant impact on application properties such as the performance, search efficiency, reliability and scalability of a system. The virtual topology also determines the communication costs and efficiency associated with running the P2P application, both at individual hosts and in the aggregate. A common difficulty in the current P2P systems is caused by the dynamic membership of peer hosts. The neighbor selection mechanism and topology control become very important topics in P2P networks [13].

On the other hand, the performance and availability of these systems relies on the voluntary participation of their users, and so they may be highly variable and unpredictable, which results in a large proportion of the participants (20 to 40% of Napster and almost 70% of Gnutella peers) share few or no files [14]. This phenomenon is known as free-loading: peers that consume more resources than they contribute. One of the reasons for this problem is that those users, called free-riders, benefit largely from contributions of other users but reduce the system performance for contributing users. And self-interested behavior of the peers had no taken into account at the design stage. In fact, the P2P system's users act rationally trying to maximize the benefits obtained from using the system's shared resources. Therefore, it will be necessary to find mechanisms that provide incentives and encourage cooperative behavior between the peers.

Particle Swarm Optimization (PSO) algorithm is inspired by social behavior patterns of organisms that live and interact within large groups. In particular, PSO incorporates swarming behaviors observed in flocks of birds, schools of fish, or swarms of bees, and even human social behavior, from which the Swarm Intelligence (SI) paradigm has emerged [15]. It could be implemented and applied to solve various function optimization problems, or the problems that can be transformed to function optimization problems. As an algorithm, the main strength of PSO is its fast convergence, which compares favorably with many global optimization algorithms [16]. In this chapter, we explore the neighbor-selection problem based PSO for P2P Networks. We introduce the crossover neighborhood organization mechanism from the social networks to improve the swarm algorithm, which results in more mutual trust, mutual benefit, equality and cooperation among the participants.

This chapter is organized as follows. We introduce the problem and formulate the objective in Section 3. Our approach based on particle swarm algorithm is presented in Section 4. In this section, the issues about the algorithm design, dynamic chaotic characteristics, and convergence theoretical analysis are also discussed. In Section 5, experiment results and discussions are provided in detail, followed by some conclusions in Section 6.

2 Related Research Work

P2P comprises peers and the connections between these peers. A common difficulty in the current P2P systems is caused by the dynamic membership of peer hosts. This results in a constant reorganization of the overlay topology [17, 18, 19, 20]. As the size of distributed systems keeps growing, no entity has a global knowledge of the system. As much as this property is essential to ensure the scalability, monitoring the system under such circumstances is a complex task [21]. Meo and Milan [22] investigated the design of content management at the nodes. They proposed criteria for the QoS design of content management policies. And they evaluated its performance by an analytical model based on a Markovian approach. In the application system, finding the desired resource and constructing the efficient topology are the critical issues in peer-to-peer networks. Risson and Moors [23] surveyed the search methods about finding the resource in the recent research towards robust peer-to-peer networks. In this chapter, we pay more attentions on peer selection, since it offers a unique opportunity for P2P networks to tackle both the free-riding and the quality-of-service (QoS) challenges [24].

Lo *et al.* [25] defined the supernode selection problem which has emerged across a variety of peer-to-peer applications. Supernode selection involves selection of a subset of the peers to serve a special role. The supernodes must be well-dispersed throughout the peer-to- peer overlay network, and must fulfill additional requirements such as load balance, resource needs, adaptability to churn, and heterogeneity. The supernode selection problem must meet the additional challenge of operating within a huge, unknown and dynamically changing network. They describe three generic supernode selection protocols. They developed for peer-to-peer environments: a label-based scheme for structured overlay networks, a distributed protocol for coordinate-based overlay networks, and a negotiation protocol for unstructured overlays. Kothapalli and Scheideler presented a general methodology for designing supervised peer-to-peer systems [26]. It can be seen as being between server-based systems and pure peer-to-peer systems. The supervisor has to store a constant amount of information about the system at any time and needs to send a small constant number of messages to integrate or remove a peer in a constant amount of time. Koulouris, *et al.* [27] presented a framework and an implementation technique for a flexible management of peer-to-peer overlays. The framework provides means for self-organization to yield an enhanced flexibility in instantiating control architectures in dynamic environments, which is regarded as being essential for P2P services to access, routing, topology forming, and application layer resource management. In these P2P applications, a central tracker decides about which peer becomes a neighbor to which other peers.

A peer randomly choosing logical neighbors without any knowledge about the underlying physical topology causes topology mismatch between the P2P logical overlay network and physical underlying network. In unstructured peer-to-peer (P2P) systems, there exists a serious topology mismatch problem between physical and logical network. Liu, *et al.* [28] analyzed the relationship between the property of the overlay and the corresponding message duplications incurred by queries in a given overlay, and prove that computing an optimal overlay with global knowledge is an NP-hard problem. Leung and Kwok [29] proposed a greedy server-peer selection algorithm to decide from which peer should a client download files so that the level of fairness of the whole network is increased and expected service life of the whole file sharing network is extended. Mastronarde, *et al.* [30] proposed a distributed and efficient framework for resource exchanges that enables peers to collaboratively distribute available wireless resources among themselves based on their quality of service requirements, the underlying channel conditions, and network topology. The resource exchanges are enabled by the scalable coding of the video content and the design of cross-layer optimization strategies, which allow efficient adaptation to varying channel conditions and available resources. They compare the designed low complexity distributed resource exchange algorithms against an optimal centralized resource management scheme and show how their performance varies with the level of collaboration among the peers. They measure system utility in terms of the multimedia quality and show that collaborative approaches achieve 50% improvement over non-collaborative approaches. Additionally, their distributed algorithms perform within 10% system utility of a centralized optimal resource management scheme.

Fenner, *et al.* [31] presented a stochastic model for a social network, where new actors may join the network, existing actors may become inactive and, at a later stage, reactivate themselves. The model captures the evolution of the network, assuming that actors attain new relations or become active according to the preferential attachment rule. They derived the mean-field equations for this stochastic model and shown that, asymptotically, the distribution of actors obeys a power-law distribution. The result illustrated that the distribution of user accesses was asymptotically a power-law distribution. Sacha, *et al.* [32] proposed and evaluated a search algorithm. The results indicated that it achieved significantly better performance than random walking. The approach can be used by certain classes of applications to improve the availability and performance of system services by placing them on the most stable peers, as well as to reduce the amount of network traffic required to discover and use these services. They demonstrated the design of a naming service on the gradient topology.

Bisnik, *et al.*[33] developed a model for random walk-based search mechanisms in unstructured P2P networks. The model is used to obtain analytical expressions for the performance metrics of random walk search in terms of the popularity of the resource being searched for and the random walk parameters. Simulation results illustrated that the performance of the equation-based adaptive search was significantly better than the non-adaptive random walk and other straightforward adaptive mechanisms. Kersch, *et al.*[34] defined a loose and stochastic long-range connection maintenance mechanism, which can significantly reduce maintenance overhead in large networks with high churn rates without affecting routing performance. They used Kleinberg's small worlds model to describe and (re)construct long-range connections. The maintenance method scale logarithmically with the system's size, which is the theoretical lower bound for maintenance traffic to ensure connectivity of the network.

Researchers have also considered clustering close peers based on their IP addresses (e.g., [35, 36]) or probed distances [37]. Xu [38] presented a decentralized and fault tolerant protocol called Alpha-Beta Cluster-based protocol, for ABC. In ABC, a cluster of nodes work together to offer efficient greedy routing and the size of each cluster can vary between an upper bound (Alpha) and a lower bound (Beta). The flexible cluster scheme helps to maintain the stability of the system. Ramaswamy, *et al.* [39] described a Connectivity-based Distributed Node Clustering scheme (CDC). The scheme presented a scalable and efficient solution for discovering connectivity-based clusters in peer networks. To cope with the typical dynamics of P2P networks, they provided mechanisms to allow new nodes to be incorporated into appropriate existing clusters and to gracefully handle the departure of nodes in the clusters. These mechanisms enable the CDC scheme to be extensible and adaptable in the sense that the clustering structure of the network adjusts automatically as nodes join or leave the system. Their experiments shown that utilizing message-based connectivity structure can considerably reduce the messaging cost and provide better utilization of resources, which in turn improved the quality of service of the applications executing over decentralized peer-to-peer networks. Huang, *et al.* [3] proposed a cluster-based peer-to-peer system, called PeerCluster, for sharing data over the Internet. In PeerCluster, all participant computers are grouped into various interest clusters, each of which contains computers that have the same interests. The intuition behind the system design was that by logically grouping users interested in similar topics together, it can improve query efficiency. To efficiently route and broadcast messages across/within interest clusters, a hypercube topology was employed. In addition, to ensure that the structure of the interest clusters is not altered by arbitrary node insertions/deletions, they have devised corresponding JOIN and LEAVE protocols. The experimental results shown that PeerCluster outperformed previous approaches in terms of query efficiency, while still providing the desired functionality of keyword-based search. Tewari and Kleinrock [40] provided mechanisms for modeling clustering in file popularity distributions and the consequent non-uniform distribution of file replicas. They derived relations shown the effect of the number of replicas of a file on the search time and on the search cost for a search for that file for the clustered demands case in such networks for both random walk and flooding search mechanisms. The derived relations were used to obtain the optimal search performance for the case of flooding search mechanisms. The potential performance benefited that clustering in demand patterns affords was captured by our results. Interestingly, the performance gains ware shown to be independent of whether the search network topology reflects the clustering in file popularity (the optimal file replica distribution to obtain these performance gains, however, does depend on the search network topology).

Empirical studies have shown free-riding (consuming resources without contributing) to be prevalent in P2P file-sharing networks. Contributors to the system are rewarded with flexibility and choice in peer selection, resulting in high quality streaming sessions. Free-riders are given limited options in peer selection, if any, and hence receive low quality service. Idris and Altmann [5] proposed an incentive scheme for P2P networks that motivates users to collaborate within the system. The solution has an impact on the topology formation of a P2P network. Using the market-managed topology formation algorithm (IUTopForm) for P2P networks, contributing users would be clustered within clubs that are different to clubs of free-riders. The differentiation was possible because of a reputation system, which considers users' past contributions. The effect of this approach was that service requests of free-riders will take longer to be answered (if at all) than service requests of resource-contributing users. The results shown that their approach improved the overall utility of the system. Habib and Chuang [24] proposed an incentive mechanism that provides service differentiation in peer selection for P2P streaming based on relative contribution of the peers. The incentive mechanism follows the characteristics of rank-order tournaments theory that considers only the relative performance of the players, and the top prizes are awarded to the winners of the tournament. The simulation and wide-area measurement results illustrate that the approach can provide near optimal streaming quality to the cooperative users until the bottleneck shifts from the streaming sources to the network. To solve the neighbor discovery problem and network organization problem in practical wireless P2P networks, Leung and Kwok [29] proposed a topology control protocol, which consists of two components, namely, Adjacency Set Construction (ASC) and Community-Based Asynchronous Wakeup (CAW). The protocol is able to enhance the fairness and provide an incentive mechanism in wireless P2P file sharing applications. It is also capable of increasing the energy efficiency.

Kurmanowytsch, *et al.* [41] developed the P2P middleware systems to provide an abstraction between the P2P topology and the applications that are built on top of it . These middleware systems offer higher-level services such as distributed P2P searches and support for direct communication among peers. The systems often provide a pre-defined topology that is suitable for a certain task (e.g., for exchanging files). Gupta *et al.* [42] discussed the system architecture, functionality, and applications of the CompuP2P architecture. They had implemented a Java-based prototype, and the results shown that the system was light-weight and can provide almost a perfect speedup for applications that contain several independent compute-intensive tasks. Zeinalipour-Yazti, *et al.* [43] presented the Peer Fusion (pFusion) architecture that aims to efficiently integrate heterogeneous information that was geographically scattered on peers of different networks. The approach built on work in unstructured P2P systems and uses only local knowledge. Our empirical results, using the pFusion middleware architecture and data sets from Akamai's Internet mapping infrastructure (AKAMAI), the Active Measurement Project (NLANR), and the Text REtrieval Conference (TREC) show that the architecture we propose is both efficient and practical.

Ghanea-Hercock, *et al.* [44] presented an algorithm based on P2P agent application in which each agent has a goal to maintain a preferred number of connections to a number of service providing agents. The agents updated a weight value associated with each connection, based on the perceived utility of the connection to the corresponding agent. This utility function can be a combination of several node or edge parameters, or frequency of the message response from the node. The weight is updated using a set of Hebbian-style learning rules, such that the network as a whole exhibits adaptive self-organizing behavior. The result was the finding that by limiting the connection neighborhood within the overlay topology, the resulting P2P network can be made highly resilient to targeted attacks on high-degree nodes, while maintaining search efficiency.

To get some insights into the performance of different peer organization strategies, Biersack, *et al.* [45] analytically three different distribution models: linear chain architecture, tree architecture, and forest architecture. The results indicated that the service capacity of these systems grew exponentially with the number of chunks a file consists of. Therefore, several heuristics and meta-heuristics have been proposed to solve the problems within the feasible runtime. Koo *et al.* [19] investigated the neighbor-selection process in the P2P networks, and proposed an efficient neighbor-selection strategy based on Genetic Algorithm (GA).

Networks seem to be the natural way chosen by nature to organize individuals, resources and interactions in an effective and robust structure. Studies about natural networks focused on the central role of emerging structures in distributed environments, and pointed out some properties such as small-world effect and communities which are of the most importance to guarantee a fast and efficient communication among nodes. Carchiolo *et al.* [46] proposed a model for P2P networks which mimics behaviours of peers in social and biological networks and naturally evolves to a robust graph of peers with some interesting properties, including small-world effect and community decomposition. Zhuge and Li [47] proposed three improved gossip mechanisms by mapping links into metric space and dynamically adapting the number of selected neighbors to disseminate messages. Experiments and comparisons shown that these mechanisms can improve the performance of gossip in peer-to-peer networks. It was the effect of mapping a network into a metric space that differentiates nodes and links according to linking characteristics and controlling local information flow with knowing such differences. An intrinsic rule is found by experimental comparisons and analysis: The performance of a P2P network can be improved by designing an appropriate mapping from the network into metric space or semantic space. These research works indicated the neighbor selection in P2P would be improved further by matching social characteristics of P2P system.

3 Neighbor-Selection Problem

In a P2P system, all participating peers form a P2P network on top of an underlying physical network. A P2P network is an abstract, logical network called an overlay network. Based on existing research [11, 28, 4, 44, 48], we formulate the neighbor-selection problem for P2P overlay networks in this Section. As given by Liu, *et al.* [28], a P2P network can be modeled based on the following assumptions:

- An overlay connection between a pair of peering nodes consists of a number of physical links which form a shortest path between the pair of end nodes in the physical topology, and Internet paths are relatively stable.
- The same size packets traversing the same physical link in a short period of time will have similar delay, as assumed by many other measurement applications.

3.1 Modeling P2P Networks

The P2P overlay networks can be modeled by an undirected graph $G = (V, E)$ where the vertex set V represents units such as hosts and routers, and the edge set E represents physical links connecting pairs of communicating unit. And $f : V \to \{1, \cdots, n\}$ be a labeling of its nodes, where $n = |V|$. For instance, G could model the whole or part of the Internet. Given an undirected graph $G = (V, E)$ modeling an interconnection network, and a subset $X \subseteq V(G)$ of communicating units (peers), we can construct a corresponding weighted graph $D = (V, E)$, where $V(D) = X$, and the weight of each $uv \in E(D)$ is equal to the length of a shortest path between peer u and peer v in G. D includes the connected edges, and is referred to as the distance graph of G. Usually we start with a physical network G (perhaps representing the Internet), and then choose a set of communicating peers X. The resulting distance graph D is the basis for constructing a P2P overlay graph $H = (V, E)$, which is done as follows. The vertex set $V(H)$ will be the same as $V(D)$, and edge set $E(H) \subseteq D(G)$. The key issue here is how to select $E(H)$. If $E = [e_{ij}]_{n \times n}$ is such that $e_{ij} = 1$ if $(i, j) \in E$, and 0 otherwise, i.e., E is the incidence matrix of G, then the neighbor-selection problem is to find a permutation of rows and columns which brings all non-zero elements of E into the optimal possible interconnection around the diagonal.

3.2 Metrics

In P2P file sharing, an interested file is divided into many fragments. The size of each fragment ranges from several hundred kilobytes to several megabytes. When a new peer joins the network, it begins to download fragments from other peers. As long as it obtains one fragment of the file, the new peer can start to serve other peers by uploading fragments. Since peers are downloading and uploading at the same time, when the network becomes large, although the demands increase, the service provided by the network also increases [49]. Given N peers, a graph $G = (V, E)$ can be used to denote an overlay network, where the set of vertices $V = \{v_1, \cdots, v_N\}$ represents the N peers and the set of edges $E = \{e_{ij} \in \{0, 1\}, i, j = 1, \cdots, N\}$ represents their connectivities : $e_{ij} = 1$ if peers i and j are connected, and $e_{ij} = 0$ otherwise. For an undirected graph, it is required that $e_{ij} = e_{ji}$ for all $i \neq j$, and $e_{ij} = 0$ when $i = j$. Let C be the entire collection of content fragments, and $\{c_i \subseteq C, i = 1, \cdots, N\}$ denotes the collection of the content fragments each peer i has. The disjointness of contents from peer i to peer j is denoted by $c_i \setminus c_j$, which can be calculated as:

$$c_i \setminus c_j = c_i - (c_i \cap c_j). \tag{1}$$

where $\setminus$ denotes the intersection operation on sets. This disjointness can be interpreted as the collection of content fragments that peer i has but peer j does not. In other words, it denotes the fragments that peer i can upload to peer j. Moreover, the disjointness operation is not commutative, *i.e.*, $c_i \setminus c_j \neq$

$c_j \setminus c_i$. Let $|c_i \setminus c_j|$ denote the cardinality of $c_i \setminus c_j$, which is the number of content fragments peer i can contribute to peer j. In order to maximize the disjointness of content, we maximize the number of content fragments each peer can contribute to its neighbors by determining the connections e_{ij}'s. Define ϵ_{ij}'s to be sets such that $\epsilon_{ij} = C$ if $e_{ij} = 1$, and $\epsilon_{ij} = \emptyset$ (null set) otherwise.

In an overlay network, every node is a potential neighbor of each other node since the network's topology is a logical one. So the full connection is an ideal solution for the peer's connectivity. For the networks, we have to consider some constraints [20, 48]:

- based on the underlying network characteristics, i.e., delay or capacity of actual links;
- based on location of data and services;
- based on the nodes's capabilities of managing peers, e.g., the number of direct neighbors a node can maintain. some peers are tied down since they possess relative more content fragments. This resource constraint can be independent of the underlying network.

In the environment, the maximum number of each peer need to be considered, i.e., each peer i will be connected to a maximum of d_i neighbors, where $d_i < N$. Therefore we have the following optimization problem:

$$\max_{E} \sum_{j=1}^{N} \left| \bigcup_{i=1}^{N} (c_i \setminus c_j) \cap \epsilon_{ij} \right| \tag{2}$$

Subject to

$$\begin{aligned} \sum_{j=1}^{N} e_{ij} &\leq d_i \text{ for all } i \\ \sum_{i=1}^{N} e_{ij} &\leq d_j \text{ for all } j \end{aligned} \tag{3}$$

4 Particle Swarm Heuristic for Neighbor-Selection

For applying the particle swarm algorithm successfully for any problem, one of the key issues is how to map the problem solution to the particle space, which affects its feasibility and performance [50]. The constraint conditions have to be satisfied, and the particle would search the solutions in as efficient a search space as possible. In this section, a new approach to the problem space mapping is depicted for particle swarm optimization with reference to the neighbor-selection problem. For solving the problem, the upper half of the peer-connection matrix through the undirected graph is encoded to the particle's position, which reduces the search space dimension significantly. Since particle swarm shares some common characteristics with P2P in the dynamic socially environment, a multi-swarm interactive pattern is introduce to match the corresponding mechanism. We analyze the dynamic characteristic of the single particle in the swarm, and then illustrate theoretically the convergence of our algorithm.

4.1 Algorithm Design

Given a P2P state $S = (N, C, M, f)$, in which N is the number of peers, C is the entire collection of content fragments, M is the maximum number of the peers which each peer can connect steadily in the session, f is to goal the number of swap fragments, *i.e.* to maximize Eq.(2). It is to be noted that the routing and connection between peers must satisfy the constraint in Eq.(3) because of bandwidth, etc. To apply the particle swarm algorithm successfully for the NS problem, one of the key issues is the mapping of the problem solution into the particle space, which directly affects its feasibility and performance. Usually, the particle's position is encoded to map each dimension to one directed connection between peers, *i.e.* the dimension is $N * N$. But the neighbor topology in P2P networks is an undirected graph, *i.e.* $e_{ij} = e_{ji}$ for all $i \neq j$, and $e_{ij} \equiv 0$ for all $i = j$. To reduce the space complexity, we set up a search space of D dimension as $N * (N-1)/2$. Accordingly, each particle's position is represented as a binary bit string of length D. Each dimension of the particle's position maps one undirected connection. The domain for each dimension is limited to 0 or 1.

The particle swarm model consists of a swarm of particles, which are initialized with a population of random candidate solutions. They move iteratively through the D-dimension problem space to search the new solutions, where the fitness f can be measured by calculating the number of swap fragments in the potential solution. Each particle has a position represented by a position-vector $\mathbf{p}_i$ (i is the index of the particle), and a velocity represented by a velocity-vector $\mathbf{v}_i$. Each particle remembers its own best position so far in a vector $\mathbf{p}_i^{\#}$, and its j-th dimensional value is $p_{ij}^{\#}$. The best position-vector among the swarm so far is then stored in a vector $\mathbf{p}^*$, and its j-th dimensional value is p_j^*. When the particle moves in a state space restricted to zero and one on each dimension, the change of probability with time steps is defined as follows:

$$\begin{aligned} P(p_{ij}(t) = 1) = f(&p_{ij}(t-1), v_{ij}(t-1), \\ &p_{ij}^{\#}(t-1), p_j^*(t-1)). \end{aligned} \tag{4}$$

where the probability function is

$$sig(v_{ij}(t)) = \frac{1}{1 + e^{-v_{ij}(t)}}. \tag{5}$$

At each time step, each particle updates its velocity and moves to a new position according to Eqs.(6) and (7):

$$\begin{aligned} v_{ij}(t) = {} & wv_{ij}(t-1) + c_1 r_1 (p^{\#}_{ij}(t-1) - p_{ij}(t-1)) \\ & + c_2 r_2 (p^{*}_{j}(t-1) - p_{ij}(t-1)) \end{aligned} \tag{6}$$

$$p_{ij}(t) = \begin{cases} 1 & \text{if } \rho < sig(v_{ij}(t)); \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

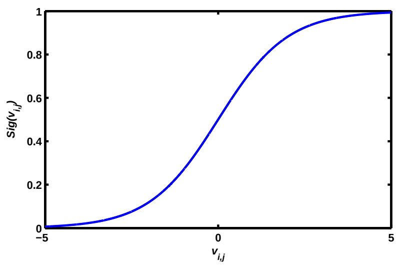

Fig. 1. Sigmoid function for PSO

Where c_1 is a positive constant, called as coefficient of the self-recognition component, c_2 is a positive constant, called as coefficient of the social component. r_1 and r_2 are the random numbers in the interval [0,1]. The variable w is called as the inertia factor, which value is typically setup to vary linearly from 1 to near 0 during the iterated processing. ρ is random number in the closed interval [0,1]. From Eq.(6), a particle decides where to move next, considering its current state, its own experience, which is the memory of its best past position, and the experience of its most successful particle in the swarm. The particle has a priority levels according to the order of peers. The sequence of the peers will be not changed during the iteration. Each particle's position indicates the potential connection state.

The particle swarm algorithm can be described generally as a population of vectors whose trajectories oscillate around a region which is defined by each individual's previous best success and the success of some other particle. Some previous studies have discussed the trajectory of particles and its convergence [51, 52, 53, 54]. It has been shown that the trajectories of the particles oscillate as different sinusoidal waves and converge quickly, sometimes prematurely. Various methods have been used to identify some other particle to influence the individual. Eberhart and Kennedy called the two basic methods as "gbest model" and "lbest model" [55]. In the gbest model, the trajectory for each particle's search is influenced by the best point found by any member of the entire population. The best particle acts as an attractor, pulling all the particles towards it. Eventually all particles will converge to this position.

In the lbest model, particles have information only of their own and their nearest array neighbors' best (lbest), rather than that of the whole swarm. Namely, in Eq.(6), gbest is replaced by lbest in the model. The lbest model allows each individual to be influenced by some smaller number of adjacent members of the population array. The particles selected to be in one subset of the swarm have no direct relationship to the other particles in the other neighborhood. Typically lbest neighborhoods comprise exactly two neighbors. When the number of neighbors increases to all but itself in the lbest model, the case is equivalent to the gbest model. Some experiment results testified that gbest model converges quickly on problem solutions but has a weakness for becoming trapped in local optima, while lbest model converges slowly on problem solutions but is able to "flow around" local optima, as the individuals explore different regions [56]. Some related research and development during the recent years are reported in [57, 58, 59, 60].

As mentioned above, one of the most important applications is to share files, distribute content in corporate networks by the dynamic membership of peer hosts. Those users usually share some common interests in some virtual spaces. They are apt to cluster into different groups. Sometime they are also the members of several groups at the same time [61]. To match the social characteristics, we introduce a multi-swarm search algorithm for neighbor-selection problem in P2P networks. In the algorithm, all particles are clustered spontaneously into different sub-swarms of the whole swarm. Every particle can connect more than one sub-swarm, and a crossover neighborhood topology is constructed between different sub-swarms. The particles in the same sub-swarm would carry some similar functions as possible and search their optimal. Each sub-swarm would approach to its appropriate position (solution), which would be helpful for the whole swarm to keep in a good balance state. Figure 2 illustrates a multi-swarm topology. In the swarm system, a swarm with 30 particles is organized into 10 sub-swarms, which one consists of 5 particles. Particles 3 and 13 have the maximum membership level, 3. During the iterated process, the particle updates its velocity following by the location of the best fitness achieved so far by the particle itself and by the location of the best fitness achieved so far across all its neighbors in all sub-swarms it belongs to. The process makes an important influence on the particles' ergodic and synergetic performance.

Since the positions of all the particles indicate the potential assigned solutions, the binary bit strings of length D can be "decoded" to the feasible solution. "1" denotes the two corresponding peers are selected in the neighborhood. On the contrary, "0" denotes the two corresponding peers are disconnected. The position may violate the constraint (3) after some iterations. We scan each column and row before the decoding procedure. The latest binary bits are set to "0" if $\sum_{j=1}^{N} e_{ij} > d_i$ or $\sum_{i=1}^{N} e_{ij} > d_j$. The scan direction are reversed after each scan. The pseudo-code for the multi-swarm search algorithm is illustrated as follows:

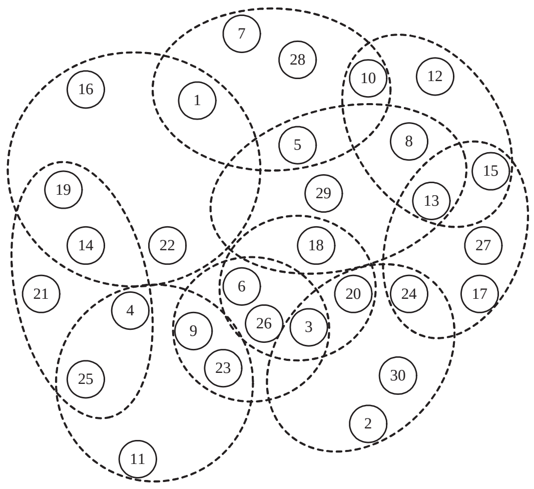

Fig. 2. A multi-swarm topology.

Step 1. Initialize the size of the particle swarm n, and other parameters. Initialize the positions and the velocities for all the particles randomly.
Step 2. Multiple sub-swarms n are organized into a crossover neighborhood topology. A particle can join more than one sub-swarm. Each particle has the maximum membership level l, and each sub-swarm accommodates default number of particles m.
Step 3. Decode the positions and evaluate the fitness for each particles.

```
3.01 For $s = 1$ to $n$
3.02   If ( $reverse$ )
3.03     For $i = 0$ to $N - 1$
3.04       $e = 0$
3.05       For $j = 0$ to $N - 1$
3.06         If $(j == i)$ $e_{ij} = 0$;
3.07         If $(j < i)$ $a = j; b = i$;
3.08         If $(j > i)$ $a = i; b = j$;
3.09         If $(e > d_j)$ $p_{[a*N+b-(a+1)*(a+2)/2]} = 0$
3.10         else
3.11           If $(p_{[a*N+b-(a+1)*(a+2)/2]})$,
3.12             Calculate $c_i \setminus c_j$; $e++$;}
3.13         End if
3.14       Next $j$
3.15     Next $i$
3.16   else
3.17     For $i = N - 1$ to 0
3.18       $e = 0$
3.19     For $j = N - 1$ to 0
```

```
3.20          If $(j == i)$ $e_{ij} = 0$;
3.21          If $(j < i)$ $a = j; b = i$;
3.22          If $(j > i)$ $a = i; b = j$;
3.23          If $(e > d_j)$ $p_{[a*N+b-(a+1)*(a+2)/2]} = 0$
3.24          else
3.25            If $(p_{[a*N+b-(a+1)*(a+2)/2]})$,
3.26              Calculate $c_i \setminus c_j$; $e++$;}
3.27          End if
3.28        Next $j$
3.29      Next $i$
3.30    End if
3.31    Calculate $f = f + \left| \bigcup_{i=1}^{N} (c_i \setminus c_j) \cap \epsilon_{ij} \right|$;
3.32    If ( $rand(0,1) < 0.5$ ) $reverse = 0$
3.33    else $reverse = 1$;
3.34 Next $s$
```

Step 4. Find the best particle in the swarm, and find the best one in each sub-swarms. If the "global best" of the swarm is improved, $noimprove = 0$, otherwise, $noimprove = 1$. Update velocity and position for each particle at the iteration t.

```
4.01 For $m = 1$ to $subs$
4.02    $\mathbf{p}^* = argmin_{i=1}^{subs_m}(f(\mathbf{p}^*(t-1)), f(\mathbf{p}_1(t))$,
4.02    $f(\mathbf{p}_2(t)), \cdots, f(\mathbf{p}_i(t)), \cdots, f(\mathbf{p}_{subs_m}(t)))$;
4.03    For $ss = 1$ to $subs_m$
4.04      $\mathbf{p}_i^{\#}(t) = argmin(f(\mathbf{p}_i^{\#}(t-1)), f(\mathbf{p}_i(t))$;
4.05      For $d = 1$ to $D$
4.06        Update the $d$-th dimension value of $\mathbf{p}_i$ and $\mathbf{v}_i$
4.06          according to Eqs.(6) and (7);
4.07      Next $d$
4.08    Next $ss$
4.09 Next $m$
```

Step 5. If $noimprove = 1$, goto Step 2, the topology is re-organized. If the end criterion is not met, goto Step 3. Otherwise, output the best solution, the fitness.

4.2 Dynamic Ergodic Characteristics

Clerc and Kennedy have stripped the particle swarm model down to a most simple form [51, 54]. If the self-recognition component c_1 and the coefficient of the social-recognition component c_2 in the particle swarm model are combined into a single term c, *i.e.* $c = c_1 + c_2$, the best position $\mathbf{p}_i$ can be redefined as follows:

$$\mathbf{p}_i \leftarrow \frac{(c_1\mathbf{p}_i + c_2\mathbf{p}_g)}{(c_1 + c_2)} \tag{8}$$

Then, the update of the particle's velocity is defined by:

$$\mathbf{v}_i(t) = \mathbf{v}_i(t-1) + c(\mathbf{p}_i - \mathbf{x}_i(t-1)) \tag{9}$$

The system can be simplified even further by using $\mathbf{y}_i(t-1)$ instead of $\mathbf{p}_i - \mathbf{x}_i(t-1)$. Thus, the reduced system is then:

$$\begin{cases} \mathbf{v}(t) = \mathbf{v}(t-1) + c\mathbf{y}(t-1) \\ \mathbf{y}(t) = -\mathbf{v}(t-1) + (1-c)\mathbf{y}(t-1) \end{cases}$$

This recurrence relation can be written as a matrix-vector product, so that

$$\begin{bmatrix} \mathbf{v}(t) \\ \mathbf{y}(t) \end{bmatrix} = \begin{bmatrix} 1 & c \\ -1 & 1-c \end{bmatrix} \cdot \begin{bmatrix} \mathbf{v}(t-1) \\ \mathbf{y}(t-1) \end{bmatrix}$$

Let

$$\mathbf{P}_t = \begin{bmatrix} \mathbf{v}_t \\ \mathbf{y}_t \end{bmatrix}$$

and

$$A = \begin{bmatrix} 1 & c \\ -1 & 1-c \end{bmatrix}$$

we have an iterated function system for the particle swarm model:

$$\mathbf{P}_t = A \cdot \mathbf{P}_{t-1} \tag{10}$$

Thus, the system is completely defined by A. Its norm $\|A\|_2$ (also written $\|A\|$) is determined by c. The relationship of A and its dependence on c is illustrated in Figure 3.

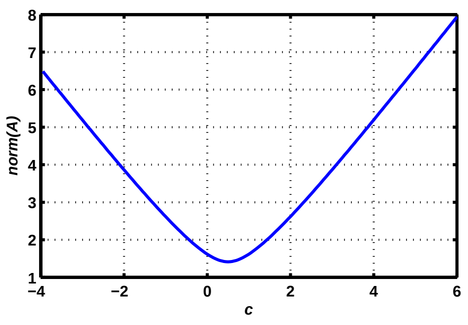

Fig. 3. Norm of A

IFS is sensitive to the values of c. It is possible to find different trajectories of the particle for various values of c. Figure 4(a) illustrates the system for a torus when c=2.9; Figure 4(b), a hexagon with spindle sides when c=2.99; Figure 4(c), a triangle with spindle sides when c=2.999; Figure 4(d), a simple triangle when c=2.9999. As depicted in Figure 4, the iteration time step used is 100 for all the cases. Another system sensitivity instance is illustrated in Figure 5. It is to be noted that Figures 4 and 5 illustrate only some 2-dimensional representations of the iterated process. In multi-dimensional search space, the particle displays the characteristics of ergodicity, which will be analyzed theoretically in Subsection 4.3.

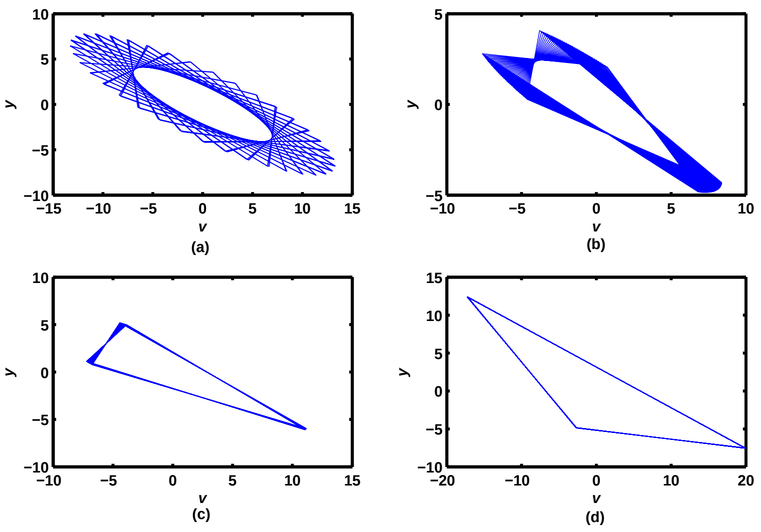

Fig. 4. Trajectory of the particle (a) $c = 2.9$, (b) $c = 2.999$, (c) $c = 2.999$, (d) $c = 2.9999$.

4.3 Convergence Analysis of Multi-Swarm Algorithm

For analyzing the convergence of the multi-swarm algorithm, we first introduce the definitions and lemmas [62, 63, 64], and then theoretically prove that the algorithm converges with a probability 1 or strongly towards the global optimal.

Xu, *et al* [65] analyzed the search capability of an algebraic crossover through classifying the individual space of genetic algorithms, which is helpful to comprehend the search of genetic algorithms such that premature convergence and deceptive problems [66] could be avoided. In this subsection, we also attempt to theoretically analyze the performance of the multi-swarm algorithm with crossover neighborhood topology. For the sake of convenience, let crossover operator $|_c$ denote the wheeling-round-the-best-particles process.

Consider the problem (P) as

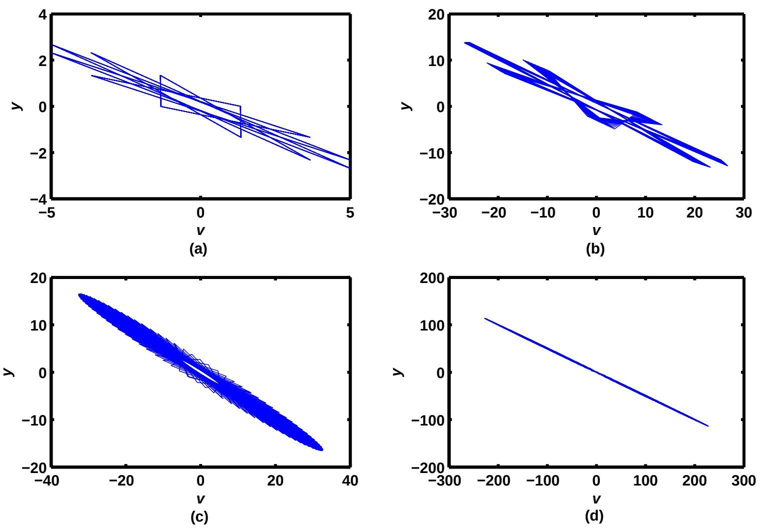

Fig. 5. Trajectory of the particle (a) $c = 3.7321$, (b) $c = 3.8$, (c) $c = 3.9$, (d) $c = 3.999$.

$$(P) = min\{f(\mathbf{x}) : \mathbf{x} \in D\} \tag{11}$$

where $\mathbf{x} = (x_1, x_2, \cdots, x_n)^T$, $f(\mathbf{x}) : D \to R$ is the objective function and D is a compact Hausdorff space. Applying our algorithm the problem (P), it can be transformed to P' as

$$(P') = \begin{cases} minf(\mathbf{x}) \\ \mathbf{x} \in \Omega = [-s, s]^n \end{cases} \tag{12}$$

where Ω is the set of feasible solutions of the problem. A swarm is a set, which consists of some feasible solutions of the problem. Assume S as the encoding space of D. A neighborhood function is a mapping $\mathcal{N} : \Omega \to 2^{\Omega}$, which defines for each solution $S \in \Omega$ a subset $\mathcal{N}(S)$ of Ω, called a neighborhood. Each solution in $\mathcal{N}(S)$ is a neighbor of S. A local search algorithm starts off with an initial solution and then continually tries to find better solutions by searching neighborhoods [67]. Most generally said, in swarm algorithms the encoding types S of particles in the search space D are often represented as strings of a fixed-length L over an alphabet. Without loss of generality, S can be described as

$$S = \underbrace{z_m \times \cdots \times z_m}_{L} \tag{13}$$

where z_m is a finite field about integer number $\mod m$. Most often, it is the binary alphabet, *i.e.* $m = 2$.

Proposition 1. *If k alleles are '0's in the nontrivial ideal Ω,* i.e. *$L-k$ alleles are uncertain, then θ_Ω partitions Ω into 2^k disjoint subsets as equivalence classes corresponding to Holland's schema theorem [68, 69],* i.e., *each equivalence class consists of some '1's which k alleles in Ω with '0' are replaced by '1's. Let $A \in S/\theta_\Omega$, then there is an minimal element m of A under partial order $(S, \vee, \wedge, \neg)$, such that $A = \{m \vee x \mid x \in \Omega\}$.*

Theorem 1. *Let A, B, C are three equivalence classes on θ_Ω, where θ_Ω is the congruence relation about Ω. $\exists\ x \in A,\ y \in B$, and $x \mid_c y \in C$, then $C = \{x \mid_c y \mid x \in A, y \in B\}$.*

Proof. Firstly, we verify that for any $d_1, d_2 \in \Omega$, if $x \mid_c y \in C$, then $(x \vee d_1) \mid_c (y \vee d_2) \in C$. In fact,

$$\begin{aligned}(x \vee d_1) \mid_c (y \vee d_2) =& (x \vee d_1)c \vee (y \vee d_2)\bar{c} \\ & (xc \vee y\bar{c}) \vee (d_1 c \vee d_2 \bar{c}) \\ & (x \mid_c y) \vee (d_1 c \vee d_2 \bar{c})\end{aligned} \tag{14}$$

Obviously, $(d_1 c \vee d_2 \bar{c}) \in \Omega$, so $(x \vee d_1) \mid_c (y \vee d_2) \equiv (x \mid_c y)(\mod \theta_\Omega)$, *i.e.* $(x \vee d_1) \mid_c (y \vee d_2) \in \Omega$.

Secondly, from Proposition 1, $\exists m, n, d_3, d_4 \in \Omega$ of A, B, such that $x = m \vee d_3$, $y = n \vee d_4$. As a result of analysis in Eq.(14), $x \mid_c y \equiv (m \mid_c n)(\mod \theta_\Omega)$, *i.e.*, $m \mid_c n \in C$.

Finally, we verify that $m \mid_c n$ is a minimal element of C and $(m \mid_c n) \vee d = (m \vee d) \mid_c (n \vee d)$. As a result of analysis in Eq.(14), if $d_1 = d_2 = d$, then $m \mid_c n \vee d = (m \vee d) \mid_c (n \vee d)$. Therefore $m \mid_c n$ is a minimal element of C.

To conclude, $C = \{(m \mid_c n) \vee d \mid d \in \Omega\} = \{x \mid_c y \mid x \in A, y \in B\}$. The theorem is proven.

Proposition 2. *Let A, B are two equivalence classes on θ_Ω, and there exist $x \in A$, $y \in B$, such that $x \mid_c y \in C$, then, $x \mid_c y$ makes ergodic search C while x and y make ergodic search A and B, respectively.*

Definition 1 (Convergence in terms of probability). *Let ξ_n a sequence of random variables, and ξ a random variable, and all of them are defined on the same probability space. The sequence ξ_n converges with a probability of ξ if*

$$\lim_{n\to\infty} P(|\xi_n - \xi| < \varepsilon) = 1 \tag{15}$$

for every $\varepsilon > 0$.

Definition 2 (Convergence with a probability of 1). *Let ξ_n a sequence of random variables, and ξ a random variable, and all of them are defined on the same probability space. The sequence ξ_n converges almost surely or almost everywhere or with probability of 1 or strongly towards ξ if*

$$P\left(\lim_{n\to\infty} \xi_n = \xi\right) = 1; \tag{16}$$

or

$$P\left(\bigcap_{n=1}^{\infty} \bigcup_{k \geq n} [|\xi_n - \xi| \geq \varepsilon]\right) = 0 \tag{17}$$

for every $\varepsilon > 0$.

Theorem 2. *Let* $\mathbf{x}^*$ *is the global optimal solution to the problem (*P'*), and* $f^* = f(\mathbf{x}^*)$. *Assume that the clubs-based multi-swarm algorithm provides position series* $\mathbf{x}_i(t)$ $(i = 1, 2, \cdots, n)$ *at time* t *by the iterated procedure.* $\mathbf{p}^*$ *is the best position among all the swarms explored so far,* i.e.

$$\mathbf{p}^*(t) = arg \min_{1 \leq i \leq n} (f(\mathbf{p}^*(t-1)), f(\mathbf{p}_i(t))) \tag{18}$$

Then,

$$P\left(\lim_{t \to \infty} f(\mathbf{p}^*(t)) = f^*\right) = 1 \tag{19}$$

Proof. Let

$$\begin{aligned} D_0 &= \{\mathbf{x} \in \Omega | f(\mathbf{x}) - f^* < \varepsilon\} \\ D_1 &= \Omega \setminus D_0 \end{aligned} \tag{20}$$

for every $\varepsilon > 0$.

While the different swarm searches their feasible solutions by themselves, assume Δp is the difference of the particle's position among different club swarms at the iteration time t. Therefore $-s \leq \Delta p \leq s$. $Rand(-1, 1)$ is a normal distributed random number within the interval [-1,1]. According to the update of the velocity and position by Eqs.(6)$\sim$(7), Δp belongs to the normal distribution, *i.e.* $\Delta p \sim [-s, s]$. During the iterated procedure from the time t to $t+1$, let q_{ij} denote that $\mathbf{x}(t) \in D_i$ and $\mathbf{x}(t+1) \in D_j$. Accordingly the particles' positions in the swarm could be classified into four states: q_{00}, q_{01}, q_{10} and q_{01}. Obviously $q_{00} + q_{01} = 1$, $q_{10} + q_{11} = 1$. According to Borel-Cantelli Lemma and Particle State Transference [59], proving by the same methods, $q_{01} = 0$; $q_{00} = 1$; $q_{11} \leq c \in (0, 1)$ and $q_{10} \geq 1 - c \in (0, 1)$.

For $\forall \varepsilon > 0$, let $p_k = P\{|f(\mathbf{p}^*(k)) - f^*| \geq \varepsilon\}$, then

$$p_k = \begin{cases} 0 & \text{if } \exists T \in \{1, 2, \cdots, k\},\ \mathbf{p}^*(T) \in D_0 \\ \bar{p}_k & \text{if } \mathbf{p}^*(t) \notin D_0,\ t = 1, 2, \cdots, k \end{cases} \tag{21}$$

According to Particle State Transference Lemma,

$$\bar{p}_k = P\{\mathbf{p}^*(t) \notin D_0, t = 1, 2, \cdots, k\} = q_{11}^k \leq c^k. \tag{22}$$

Hence,

$$\sum_{k=1}^{\infty} p_k \leq \sum_{k=1}^{\infty} c^k = \frac{c}{1-c} < \infty. \tag{23}$$

According to Borel-Cantelli Lemma,

$$P\left(\bigcap_{t=1}^{\infty}\bigcup_{k\geq t}|f(\mathbf{p}^*(k))-f^*|\geq\varepsilon\right)=0 \tag{24}$$

As defined in Definition 2, the sequence $f(\mathbf{p}^*(t))$ converges almost surely or almost everywhere or with probability 1 or strongly towards f^*. The theorem is proven.

5 Algorithm Performance Demonstration

To illustrate the effectiveness and performance of the particle swarm optimization algorithm, we illustrate an execution trace of the algorithm for the NS problem. A file of size 7 MB is divided into 14 fragments (512 KB each) to distribute, 6 peers download from the P2P networks, and the connecting maximum number of each peer is 3, which is represented as $(6, 14, 3)$ problem. In some session, the state of distributed file fragments is as follows:

$$\begin{bmatrix} 1&0&0&4&0&6&7&8&0&10&0&12&0&14\\ 0&0&0&4&5&0&7&0&9&0&11&0&13&0\\ 0&2&0&0&0&6&0&0&0&0&11&12&0&14\\ 0&2&3&4&0&6&0&0&0&0&11&0&0&0\\ 0&2&0&0&0&0&7&8&0&10&0&12&0&14\\ 1&2&0&0&5&0&0&0&9&10&11&0&13&14 \end{bmatrix}$$

The optimal result search by the multi-swarm algorithm is 31, and the neighbor selection solution is illustrated below:

$$\begin{array}{c} \\ 1\\2\\3\\4\\5\\6 \end{array}\begin{array}{c} \begin{array}{cccccc} 1&2&3&4&5&6 \end{array}\\ \begin{pmatrix} 0&0&0&1&1&1\\ 0&0&0&0&1&1\\ 0&0&0&1&1&1\\ 1&0&1&0&0&0\\ 1&1&1&0&0&0\\ 1&1&1&0&0&0 \end{pmatrix} \end{array}$$

We also tested other three representative instances (problem (25,1400,12), problem (30,1400,15), problem (35,1400,17) and problem (100,1400,20)) further. In our experiments, the algorithms used for comparison were mainly SPSO (standard PSO) ([55]) and GA (Genetic Algorithm) ([70]). These algorithms share many similarities. GA is powerful stochastic global search and optimization methods, which are also inspired from the nature like the PSO. Genetic algorithms mimic an evolutionary natural selection process. Generations of solutions are evaluated according to a fitness value and only those candidates with high fitness values are used to create further solutions via crossover and mutation procedures. Both methods are valid and efficient methods in numeric programming and have been employed in various fields due to their strong convergence properties. The considered algorithms were repeated 4 times with different random seeds. Each trial had a fixed number of 50 or 80 iterations. Other specific parameter settings of the algorithms are described in Table 1, where D is the dimension of the position. The average fitness values of the best solutions throughout the optimization run were recorded. The average and the standard deviation were calculated from the 4 different trials.

Table 1. Parameter settings for the algorithms.

Algorithm	Parameter name	Value
	Size of the population	$(even)(int)(10 + 2 * sqrt(D))$
GA	Probability of crossover	0.8
	Probability of mutation	0.01
	Swarm size	$(even)(int)(10 + 2 * sqrt(D))$
	Self coefficient c_1	$0.5 + log(2)$
PSO(s)	Social coefficient c_2	$0.5 + log(2)$
	Inertia weight w	0.91
	Clamping Coefficient ρ	0.5

Figures 6, 7, 8 and 9 illustrate the performances during the search processes using the considered algorithms to solve the NS problems. The best values, mean values, the standard deviations for 4 trials are shown in Table 2. As evident, the multi-swarm algorithm obtained better results much faster than other algorithms, especially for large scale problems. The multi-swarm algorithm offered the advantages of steady performance, since it has the least standard deviations.

Table 2. Performance comparison of the three algorithms.

Instance	Item	GA	SPSO	MPSO
	Best	8716.00	8717.00	8721.00
$(25, 1400, 12)$	Mean	8.714.30	8716.00	87192.00
	Std. dev.	1.7078	1.1547	1.3292
	Best	10513.00	10514.00	10515.00
$(30, 1400, 15)$	Mean	10504.00	10512.00	10514.00
	Std. dev.	6.3443	1.2990	1.2910
	Best	12321.00	12332.00	12332.00
$(35, 1400, 17)$	Mean	12319.00	12329.00	12330.00
	Std. dev.	1.7078	2.5166	1.1690
	Best	35047.00	35057.00	35061.00
$(100, 1400, 20)$	Mean	35042.25	35055.00	35059.25
	Std. dev.	3.6996	1.2247	1.0897

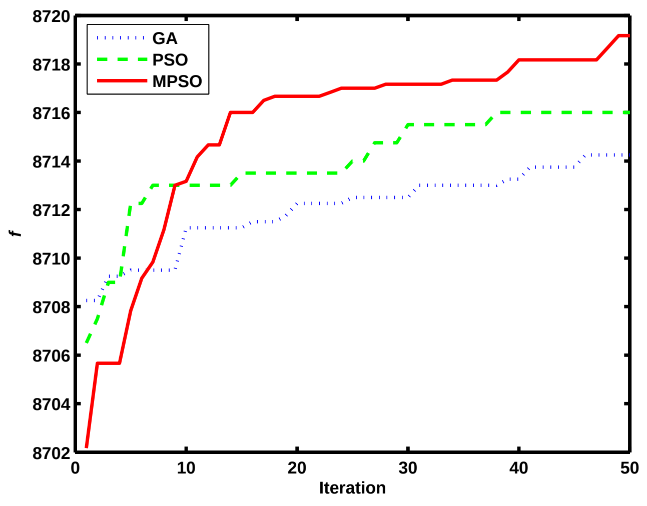

Fig. 6. Performance for the NS $(25, 1400, 12)$

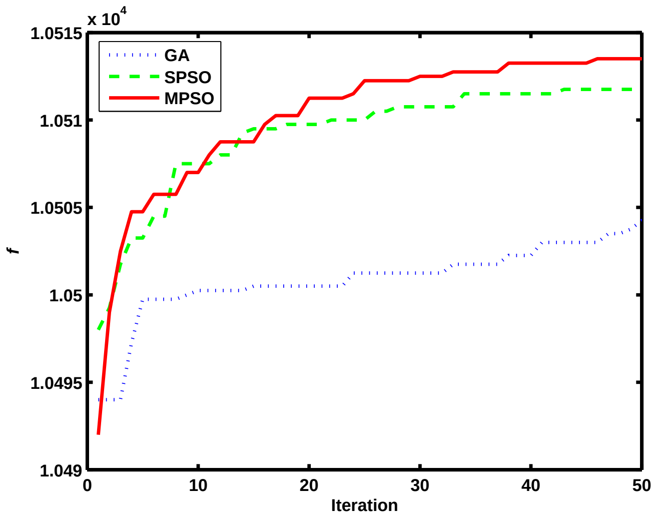

Fig. 7. Performance for the NS $(30, 1400, 15)$

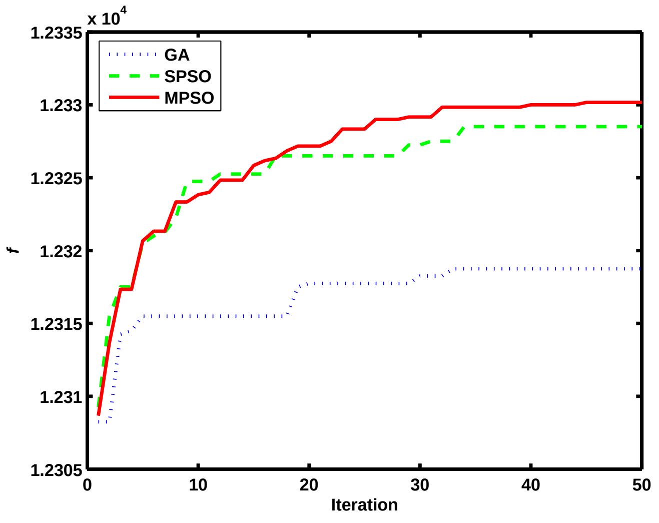

Fig. 8. Performance for the NS $(35, 1400, 17)$

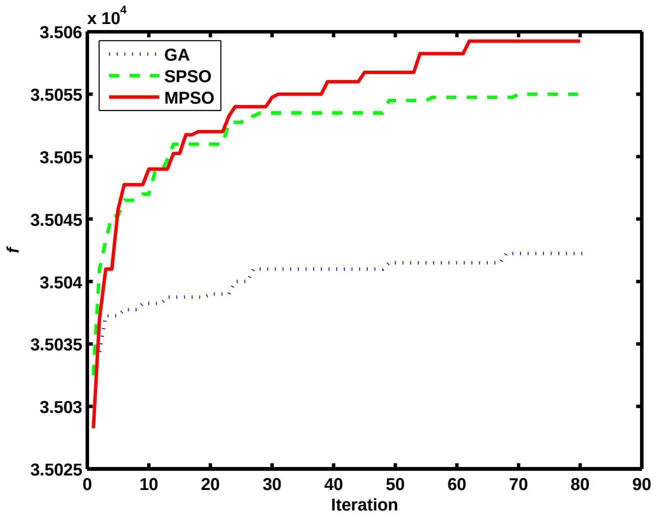

Fig. 9. Performance for the NS $(100, 1400, 20)$

6 Conclusion

In this chapter, we investigated to solve the class of the neighbor-selection problem in peer-to-peer networks by using a swarm intelligence approach. We encoded the particles using the upper half matrix of the peer connection through the undirected graph, through which we accomplished the mapping between the problem and the particle. It is feasible to reduce the dimension of the particle's search space. Since particle swarm shares some common characteristics with P2P in the dynamic socially environment, a multi-swarm interactive pattern was introduced to match the corresponding mechanism. We designed a crossover neighborhood multi-swarm algorithm based on discrete particle swarm optimization for the neighbor-selection problem in peer-to-peer networks. We analyzed the dynamic characteristic of the single particle in the swarm. The multi-swarm algorithm performance was illustrated theoretically that it converges with a probability of 1 towards the global optimum. We evaluated the performance of the proposed approach and compared it with Genetic Algorithm (GA) and SPSO (standard PSO). The results indicated that multi-swarm approach usually obtained better results much faster than GA and SPSO, specially for large scale problems and the multi-swarm algorithm offered the advantages of steady performance. The crossover neighborhood multi-swarm algorithm could be an ideal approach for solving the neighbor-selection problem in peer-to-peer networks. Compared to the previous algorithms proposed in P2P file sharing systems, the proposed algorithm has a low communication overhead.

Acknowledgment

The authors would like to thank Drs. Shichang Sun, Mingyan Zhao for their scientific collaboration in this research work. This work is supported partially by NSFC Grant 60873054 and DLMU Grant DLMU-ZL-200709.

References

1. Lua E K, Crowcroft J, Pias M, Sharma R and Lim S (2005) A Survey and Comparison of Peer-to-Peer Overlay Network Schemes. IEEE Communications Surveys & Tutorials, 7(2):72–93
2. Kwok S (2006) P2P Searching Trends: 2002-2004. Information Processing and Management, 42:237–247
3. Huang X, Chang C and Chen M (2006) PeerCluster: A Cluster-Based Peer-to-Peer System. IEEE Transactions on Parallel and Distributed Systems, 17(10):1110-1123
4. Belmonte M V, Conejo R, Díaz M and Pérez-de-la-Cruz J L (2006) Coalition Formation in P2P File Sharing Systems. Lecture Notes in Artificial Intelligence, CAEPIA'05, 4177:153–162

5. Idris T and Altmann J (2006) A Market-managed Topology Formation Algorithm for Peer-to-Peer File Sharing Networks, Lecture Notes in Computer Science, 4033:61–77
6. Cho H (2007) An Update Propagation Algorithm for P2P File Sharing over Wireless Mobile Networks, Lecture Notes in Computer Science, ICCS'07, 4490:753–760
7. Pianese F, Perino D , Keller J and Biersack E W (2007) PULSE: An Adaptive, Incentive-Based, Unstructured P2P Live Streaming System. IEEE Transactions on Multimedia, 9(8):1645–1660
8. Sigurdsson H M, Halldorsson U R and Hasslinger G (2007) Potentials and Challenges of Peer-to-Peer Based Content Distribution. Telematics and Informatics, 24:348–365
9. Yang S and Chen I (2008) A Social Network-based System for Supporting Interactive Collaboration in Knowledge Sharing Over Peer-to-Peer Network. International Journal of Human-Computer Studies, 66:36–50
10. Kim J K, Kim H K and Cho Y H (2008) A User-oriented Contents Recommendation System in Peer-to-Peer Architecture. Expert Systems with Applications, 34:300–312
11. Sen S and Wang J (2004) Analyzing Peer-to-Peer Traffic Across Large Networks. IEEE/ACM Transactions on Networking, 12(2):219–232
12. Leung A and Kwok Y (2005) An Efficient and Practical Greedy Algorithm for Server-Peer Selection in Wireless Peer-to-Peer File Sharing Networks. Lecture Notes in Computer Science, MSN'05, 3794:1016-1025
13. Ardizzone E, Gatani L, La Cascia M, Lo Re G and Ortolani M (2007) Enhanced P2P Services Providing Multimedia Content. Advances in Multimedia, 1–12
14. Androutsellis-theotokis S and Spinellis D (2004) A Survey of Peer-to-Peer Content Distribution Technologies. ACM Computing Surveys, 36(4):335–371
15. Clerc M (2006) Particle Swarm Optimization, ISTE Publishing Company, London
16. Abraham A, Guo H and Liu H (2006) Swarm intelligence: foundations, perspectives and applications. Swarm Intelligent Systems, Studies in Computational Intelligence, 3–25
17. Schollmeier R (2001) A Efinition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications. Proceedings of the First International August Conference on Peer-to-Peer Computing, 101–102
18. Ghosal D, Poon B K and Kong K (2005) P2P contracts: a framework for resource and service exchange. Future Generation Computer Systems, 21:333–347
19. Koo S G, Kannan K and Lee C S (2006) A Genetic-algorithm-based Neighbor-selection Strategy for Hybrid Peer-to-Peer Networks. Future Generation Computer Systems, 22:732–741
20. Surana S, Godfrey B, Lakshminarayanan K, Karp R and Stoica I (2006) Load Balancing in Dynamic Structured Peer-to-Peer Systems. Performance Evaluation, 63:217–240
21. Merrer E, Kermarrec A, and Massoulié L (2006) Peer to Peer Size Estimation in Large and Dynamic Networks: A Comparative Study. Proceedings of 15th IEEE International Symposium on High Performance Distributed Computing, 7–17
22. Meo M and Milan F (2008) QoS Content Management for P2P File-sharing Applications. Future Generation Computer Systems, 24:213–221
23. Risson J and Moors T (2006) Survey of Research Towards Robust Peer-to-Peer Networks: Search Methods. Computer Networks, 50:3485–3521

24. Habib A and Chuang J (2006) Service Differentiated Peer Selection: An Incentive Mechanism for Peer-to-Peer Media Streaming. IEEE Transactions on Multimedia, 8(3):610–623
25. Lo V, Zhou D, Liu Y, GauthierDickey C S, and Li J (2005) Scalable Supernode Selection in Peer-to-Peer Overlay Networks. Proceedings of the Second IEEE International Workshop on Hot Topics in Peer-to-Peer Systems, 18–27
26. Kothapalli K and Scheideler C (2005) Supervised Peer-to-Peer Systems. Proceedings of the 8th International Symposium on Parallel Architectures, Algorithms and Networks, 188–193
27. Koulouris T, Henjes R, Tutschku K and de Meer H (2004) Implementation of adaptive control for P2P overlays. Lecture Notes in Computer Science, 2982:292–306
28. Liu Y, Xiao L, Esfahanian A and Ni L M (2005) Approaching Optimal Peer-to-Peer Overlays. Proceedings of the 13th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 407–414
29. Leung A K and Kwok Y (2008) On Localized Application-Driven Topology Control for Energy-Efficient Wireless Peer-to-Peer File Sharing. IEEE Transactions on Mbile Computing, 7(1):66–80
30. Mastronarde N, Turaga D S and van der Schaar M (2007) Collaborative Resource Exchanges for Peer-to-Peer Video Streaming Over Wireless Mesh Networks. IEEE Journal on Selected Areas in Communications, 25(1)108–118
31. Fenner T, Levene M, Loizou G, and Roussos G (2007) A Stochastic Evolutionary Growth Model for Social Networks. Computer Networks, 51:4586-4595
32. Sacha J, Dowling J, Cunningham R, and Meier R (2006) Discovery of Stable Peers in a Self-Organising Peer-to-Peer Gradient Topology. Lecture Notes in Computer Science, 4025:70–83
33. Bisnik N and Abouzeid A A (2007) Optimizing Random Walk Search Algorithms in P2P Networks. Computer Networks, 51(6):1499–1514
34. Kersch P, Szabo R, Cheng L, Jean K, and Galis A (2007) Stochastic Maintenance of Overlays in Structured P2P Systems. Computer Communications, `doi:10.1016/j.comcom.2007.08.017`
35. Krishnamurthy B and Wang J (2001) Topology Modeling via Cluster Graphs. Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement, 19–23
36. Padmanabhan V N and Subramanian L (2001) An Investigation of Geographic Mapping Techniques for Internet Hosts. Proceedings of the ACM conference on Applications, technologies, architectures, and protocols for computer communications, 173–185
37. Nakao A, Peterson L and Bavier A (2003) A Routing Underlay for Overlay Networks. Proceedings of the ACM conference on Applications, technologies, architectures, and protocols for computer communications, 11–18
38. Xu X (2007) ABC: A Cluster-based Protocol for Resource Location in Peer-to-Peer Systems. Journal of Parallel and Distributed Computing, `doi:10.1016/j.jpdc.2005.02.004`
39. Ramaswamy L, Gedik B and Liu L (2005) A Distributed Approach to Node Clustering in Decentralized Peer-to-Peer Networks. IEEE Transactions on Parallel and Distributed Systems, 16(9):814–829

40. Tewari S and L. Kleinrock L (2007) Optimal Search Performance in Unstructured Peer-to-Peer Networks With Clustered Demands. IEEE Journal on Selected Areas in Communications, 25(1):84–95
41. Kurmanowytsch R, Kirda E, Kerer C and Dustdar S (2003) OMNIX: A topology-independent P2P middleware. Proceedings of The 15th Conference on Advanced Information Systems Engineering
42. Gupta R, Sekhri V, and Somani A K (2006) CompuP2P: An Architecture for Internet Computing Using Peer-to-Peer Networks. IEEE Transactions on Parallel and Distributed Systems, 17(11):1306–1320
43. Zeinalipour-Yazti D, Kalogeraki V and Gunopulos D (2007) pFusion: A P2P Architecture for Internet-Scale Content-Based Search and Retrieval. IEEE Transactions on Parallel and Distributed Systems, 18(6):804–817
44. Ghanea-Hercock R A, Wang F and Sun Y (2006) Self-Organizing and Adaptive Peer-to-Peer Network. IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics, 36(6):1230-1236
45. Biersack E W, Rodriguez P, and Felber P (2004) Performance Analysis of Peer-to-Peer Networks for File Distribution. Lecture Notes in Computer Science, QofIS'04, 3266:1–10
46. Carchiolo V, Malgeri M, Mangioni G and Nicosia V (2007) Emerging structures of P2P networks induced by social relationships, `doi:10.1016/j.comcom.2007.08.016`
47. Zhuge H and Li X (2007) Peer-to-Peer in Metric Space and Semantic Space. IEEE Transactions on Knowledge and Data Engineering, 19(6):759–771
48. Wang S, Chou H, Wei D and Kuo S (2007) On the Fundamental Performance Limits of Peer-to-Peer Data Replication in Wireless Ad hoc Networks. Journal on Selected Areas in Communications, 25(1):211–221
49. Qiu D and Sang W (2007) Global Stability of Peer-to-Peer File Sharing Systems. Computer Communications, `doi:10.1016/j.comcom.2007.08.012`
50. Salman A, Ahmad I, Al-Madani S (2002) Particle Swarm Optimization for Task Assignment Problem. Microprocessors and Microsystems, 26:363-371
51. Clerc M and Kennedy J (2002) The Particle Swarm - Explosion, Stability, and Convergence in A Multidimensional Complex Space. IEEE Transactions on Evolutionary Computation, 6(1):58–73
52. Cristian T I (2003) The Particle Swarm Optimization Algorithm: Convergence Analysis and Parameter Selection. Information Processing Letters, 85(6):317–325
53. van den Bergh F and A.P. Engelbrecht A P (2006) A Study of Particle Swarm Optimization Particle Trajectories. Information Sciences, 176:937–971
54. Liu H, Abraham A and Clerc M (2007) Chaotic Dynamic Characteristics in Swarm intelligence. Applied Soft Computing, 7:1019–1026
55. Kennedy J and Eberhart R (2001) Swarm Intelligence, Morgan Kaufmann, CA
56. Liu H, Li B, Ji Y and Sun T (2006) Particle Swarm Optimisation from lbest to gbest. Applied Soft Computing Technologies: The Challenge of Complexity, 537–545
57. Grosan C, Abraham A, Nicoara M (2005) Search Optimization Using Hybrid Particle Sub-swarms and Evolutionary Algorithms. International Journal of Simulation Systems, Science & Technology, 6(10):60–79
58. Jiang C W and Etorre B (2005) A Hybrid Method of Chaotic Particle Swarm Optimization and Linear Interior for Reactive Power Optimisation. Mathematics and Computers in Simulation, 68:57–65

59. Liu H and Abraham A (2007) An Hybrid Fuzzy Variable Neighborhood Particle Swarm Optimization Algorithm for Solving Quadratic Assignment Problems. Journal of Universal Computer Science, 13(7):1032–1054
60. Liang J J, Qin A K, Suganthan P N, Baskar S (2006) Comprehensive Learning Particle Swarm Optimizer for Global Optimization of Multimodal Functions. IEEE Transactions on Evolutionary Computation, 10(3):281–295
61. Elshamy W, Emara H M, and Bahgat A (2007) Clubs-based Particle Swarm Optimization, Proceedings of the IEEE International Conference on Swarm Intelligence Symposium, 1:289–296
62. Guo C and Tang H (2001) Global Convergence Properties of Evolution Stragtegies. Mathematica Numerica Sinica, 23(1):105–110
63. He R, Wang Y, Wang Q, Zhou J, Hu C (2005) An Improved Particle Swarm Optimization Based on Self-adaptive Escape Velocity. Journal of Software, 16(12):2036–2044
64. Weisstein E W (2007) Borel-Cantelli Lemma, From MathWorld – A Wolfram Web Resource, `http://mathworld.wolfram.com/Borel-CantelliLemma.html`
65. Xu Z, Cheng G and Liang Y (1999) Search Capability for An Algebraic Crossover. Journal of Xi'an Jiaotong University, 33(10):88-99
66. Whitley L D (1991) Fundamental Principles of Deception in Genetic Search. Foundation of Genetic Algorithms, California: Morgan Kaufmann Publishers, 221–241
67. Mastrolilli M and Gambardella L M (2002) Effective Neighborhood Functions for the Flexible Job Shop Problem. Journal of Scheduling, 3(1):3–20
68. Holland J H (1975) Adaptation in Natural and Artificial Systems, Ann Arbor: University of Michigan Press
69. Goldberg D E (1989) Genetic Algorithms in Search, Optimization and Machine Learning. Reading, MA: Addison-Wesley
70. Abraham A (2005) Evolutionary computation. Handbook for Measurement Systems Design, 920–931