Journal of Southeast University (English Edition) Vol. 24, No. 4, pp. 444 – 449 Dec. 2008 ISSN 1003—7985

SSABC: a super-peer selection algorithm based on capacity

Zhao Shenghui[1,2] Qian Ning[1] Wu Guoxin[1] Chen Guilin[2]

([1]Key Laboratory of Computer Network and Information Integration of Ministry of Education, Southeast University, Nanjing 210096, China)
([2]Department of Computer Science and Technology, Chuzhou University, Chuzhou 239012, China)

Abstract: Combining the characteristics of peer-to-peer (P2P) and grid, a super-peer selection algorithm—SSABC is presented in the distributed network merging P2P and grid. The algorithm computes nodes capacities using their resource properties provided by a grid monitoring and discovery system, such as available bandwidth, free CPU and idle memory, as well as the number of current connections and online time. when a new node joins the network and the super-peers are all saturated, it should select a new super-peer from the new node or joined nodes with the highest capacity. By theoretical analyses and simulation experiments, it is shown that super-peers selected by capacity can achieve higher query success rates and shorten the average hop count when compared with super-peers selected randomly, and they can also balance the network load when all super-peers are saturated. When the number of total nodes changes, the conclusion is still valid, which explains that the algorithm SSABC is feasible and stable.

Key words: peer to peer (P2P); grid; super-peer; capacity selection; random selection

The convergence system of P2P and grid can efficiently and reasonably implement resource utilization, which provides a new platform for sharing digital resources. Using P2P scalability and dynamic properties, designing resource location and discovery protocols for querying grid resources to improve the query success rate and fault tolerance of the system have been involved in network systems merging P2P and grids[1-6]. These hybrid systems can manage general nodes concentrated within a certain range by super-peers (also called cluster-heads), and integrate limited resources. At the same time, super-peers collaborate with each other to form a decentralized P2P network at a higher layer. Yang et al. studied pure P2P networks based on super-peer architectures[7]. KaZaA and Gnutella are two typical existing super-peers-based P2P systems. Furthermore, Montresor et al. [8-11] proposed opinions on super-peer selection in their systems, but did not do further research on how to select them. Pasquale et al. presented a model for job assignments across the grid exploiting an underlying super-peer topology[12]. Mastroianni et al. devised grid information services according to a super-peer model[13-14]. A hybrid and unstructured network model based on P2P and grid was introduced in Ref. [15]. But these models only gave or applied the concept of a super-peer. Although they have agreed on opinions of how to construct super-peer architecture on the basis of P2P and grid, they have not discussed super-peer selection and its formation in detail.

Furthermore, because of the super-peer's special contribution, its performance will directly influence the whole network efficiency. Generally, not only the configuration of the super-peer itself, such as the number of CPUs, the memory size and the I/O speed, but also some dynamic factors such as network load, available memory or free CPUs and so on, impact its functions. How to select a super-peer reasonably has been recognized as a new hot research topic.

At present, an open-sources software, Globus Toolkit (GT)[16], has been accepted as the mature software toolkit for deploying grid. GT includes resources monitoring, discovery, management, and some software services about security communication and file management and correlative lib functions. By deploying GT, computers can share computing power, databases and other secure online tools with the characters of across-organization distributed geographically.

In Ref. [15], Chen et al. designed a model of integrating P2P and grid which has three levels. Since the construction of a grid usually has a clear applied target, the corresponding resources organization is called a VO (virtual organization). The model consists of different VOs forming the first level. Each VO has some super-peers, which construct the second level. A super-peer function for replacing its sub-nodes to submit jobs is the same as that described in Ref. [7]. How to select nodes as super-peers from numerous nodes in one VO is a key problem. This paper designs a super-peer selection algorithm based on capacity. In order to compute a node's capacity, we make use of the properties provided by the monitoring and discovery services in GT. Then we select the node with the highest capacity as a super-peer.

1 Monitoring and Discovery Service on Grid

Resources in grids include all kinds of an application data and corresponding software and hardware, for example, computing and storage resources. Along with the running of applications, the available resources in the VO will continually change. From the management point of view, grid resources are divided into static and dynamic ones. The properties of static resources include the operating system name and version, processor type and physical memory size, etc. And dynamic properties involve available disk space, delay, network bandwidth, idle physical memory, free CPU and so on.

The current version of GT is 4. 2. The monitoring and discovery system (MDS)[16] is one of the GT4. 2 components. MDS is a suite of web services used to monitor and discover resources and services on grids. Web service used for collecting the above information is called an index service. The index service is a registry similar to UDDI[17], and collects

Received 2008-03-12.

Biography: Zhao Shenghui (1970—), female, graduate, associate professor, shzhao@ah.edu.cn.

Foundation items: The National High Technology Research and Development Program of China (863 Program) (No. 2007AA01Z422), the Natural Foundation of Anhui Provincial Education Department (No. 2006KJ041B, KJ2007B073).

Citation: Zhao Shenghui, Qian Ning, Wu Guoxin, et al. SSABC: a super-peer selection algorithm based on capacity[J]. Journal of Southeast University (English Edition), 2008, 24(4): 444 – 449.

static and dynamic information and publishes them as resource properties provided to clients by a web service interface. Clients may query and subscribe resource properties from indices. The index service not only saves local useful data, but also caches the remote data, and maintains data updating by a lifetime management mechanism. In a large scale grid, indices can register each other in a hierarchical fashion in order to aggregate data at different levels.

On the basis of the model presented in Ref. [15], each VO sets an aggregated node (AN, such as the contact node defined in Ref. [15]) and each node registers to an AN in the form of a service when it joins the system. The registered nodes are also called grid nodes. Index services in an AN is used for collecting the resources status information of registered nodes. All registered grid nodes probably become super-peers; meanwhile, each registered service has a lifetime, and it updates information periodically.

2 Strategy for Selecting Super-Peer

Super-peers are selected from grid nodes, and they connect to each other to form an overlay network at a higher level using a P2P mechanism. In a VO, the number of super-peers is limited. If the number is fewer, it will make the super-peers' loads increase and bring about a bottleneck; conversely, it will cause greater traffic between super-peers while forwarding queries. We refer to a super-peer and its managed sub-nodes as a cluster.

2.1 Normalization of resource properties

We assume that there are n nodes in a VO, each node has t properties. We can obtain the following matrix $\boldsymbol{L}$. Each row in $\boldsymbol{L}$ represents a node, while each column represents one of nodes properties.

$$\boldsymbol{L}=\begin{bmatrix} q_{1,1} & q_{1,2} & \cdots & q_{1,t} \\ q_{2,1} & q_{2,2} & \cdots & q_{2,t} \\ \vdots & \vdots & & \vdots \\ q_{n,1} & q_{n,2} & \cdots & q_{n,t} \end{bmatrix}$$

Each property's value has different types and ranges. For the sake of obtaining a uniform measurement of a property, it should be normalized. The property normalization computing method is Eq. (1).

$$m_{i,j}=\frac{q_{i,j}}{\max(\{q_{i,j}\mid j=1,2,\ldots,t\})} \tag{1}$$

Then the range of values can be adjusted to a uniform scope $[0,1]$. Each property $q_{i,j}$ has a weight w_j in order of precedence. The sum of w_j is 1, that is

$$\sum_{j=1}^{t} w_j = 1 \qquad w_j \in [0,1]; j = 1,2,\ldots,t$$

Set

$$h_{i,j}=w_j m_{i,j}$$

Then matrix $\boldsymbol{L}$ is converted into

$$\boldsymbol{N}=\begin{bmatrix} h_{1,1} & h_{1,2} & \cdots & h_{1,t} \\ h_{2,1} & h_{2,2} & \cdots & h_{2,t} \\ \vdots & \vdots & & \vdots \\ h_{n,1} & h_{n,2} & \cdots & h_{n,t} \end{bmatrix}$$

Set

$$C_i = \sum_{j=1}^{t} w_j h_{i,j} \qquad i = 1,2,\ldots,n \tag{2}$$

where C_i is the capacity of node e_i.

2.2 Super-peer selection algorithm based on capacity

In order to describe the algorithm conveniently, we define two sets. I represents the set of super-peers. R is a set of nodes capacities registered to an AN. After a grid node registers into an AN, the AN obtains its information, such as free CPU, idle memory, available bandwidth, number of current connections and online time, and stores this information in its index table. A node's capacity means how many resources a node owns and whether or not it has been selected as a super-peer. We use the above five properties to measure a node's capacity, because CPU, memory and bandwidth are all dynamic in the grid, and they play important roles in computing power. The number of current connections declares computer bearing ability, and online time explains computer stability and reliability.

Assuming M is the maximal value of a cluster size. When the number of nodes in one cluster is fewer than M, the super-peer in this cluster is called as non-saturated. C_i is the capacity of node e_i, and S_{j_conn} represents the number of nodes connected to a node e_j, while T_s represents the number of saturated super-peers. D is the number of sub-nodes which would be released by all saturated super-peers when a new super-peer joins the system.

In order to balance network load, when adding a new super-peer, the AN would demand the existing super-peers to release D sub-nodes and make DT_s sub-nodes connect to the new super-peer.

The SSABC algorithm is as follows:

1) Initialization: $I \leftarrow \varnothing$, $R \leftarrow \varnothing$.
2) If the first node e_1 joins the network, then
   Compute e_1's capacity C_1 using Eq. (2);
   $I \leftarrow I \cup \{e_1\}$;
   $R \leftarrow R \cup \{C_1\}$;
3) If the i-th node $e_i (i \geqslant 2)$ joins the network, then
   Compute e_i's capacity C_i using Eq. (2);
   $R \leftarrow R \cup \{C_i\}$;
4) /* If there are non-saturated super-peers in I, then e_i connects to the nearest node among them. The distance is computed using hop counts. */
   if $\exists S_{j\_\mathrm{conn}} < M$, $\forall e_j \in I$ and e_j is the nearest
      e_i connects to e_j;
   endif
   go to 3);
5) /* If super-peers in I are all saturated, then select a new super-peer. */
   if $C_i \geqslant C_j, j \neq i, \forall C_j \in R$
   /* C_i is the maximum capacity, and then e_i is selected as a new super-peer. */
      $I \leftarrow I \cup \{e_i\}$;
      $R \leftarrow R - \{C_i\}$;
   endif
   if C_i is not the maximum capacity, then
   Query a node e_k from R whose capacity is the maxi-

mum;

$I \leftarrow I \cup \{e_k\}$;

$R \leftarrow R - \{C_k\}$;

endif

e_k disconnects its super-peer and acts as a new super-peer itself.

6) / * Let every existing super-peer release D nodes, balancing the load. * /

if e_i or e_k acts as super-peer then

$D = M - MT_s/(T_s + 1)$;

DT_s sub-nodes connect to e_i or e_k;

endif

The SSABC algorithm is executed by AN in a VO.

Without loss of generality, it is assumed that there are at most N nodes in the system. Obviously, computing overhead of the SSABC algorithm focuses on steps 4) and 5). Either of them is $o(N)$, other steps are $o(1)$, so the whole overhead is $o(N)$.

2.3 Theoretical analysis of query performance in SSABC

We assume that Y is the required capacity of a user resource application, and then Y is a random variable and follows exponential distribution. Y's distribution function is $F(y) = 1 - e^{-\lambda y}$.

Therefore, the probability that Y is less than or equal to super-peer capacity (set it as y) is

$$p(0 < Y \leqslant y) = F(Y) - F(0) = 1 - e^{-\lambda y}$$

Suppose that y_1 is a super-peer's capacity when using SSABC. Then p_1 is the query success rate; y_2 is another super-peer capacity by random selection, and its query success rate is p_2. Obviously, the super-peer capacity selected by SSABC is greater than that of random selection. So, $\bar{y}_1$ is greater than $\bar{y}_2$, viz. $\bar{y}_1 > \bar{y}_2$. Then

$$P_1(0 < Y \leqslant \bar{y}_1) - P_2(0 < Y \leqslant \bar{y}_2) = e^{-\lambda \bar{y}_2} - e^{-\lambda \bar{y}_1} > 0$$

The above formula shows that when carrying out one query in a hop, success probability for capacity-based selection is greater than that for random selection, viz. $p_1 > p_2$. When we do a one-time resource application, the total query number of times is n and the success query number of times is X, then X is a random variable. $X = \{0, 1, 2, \ldots, n\}$. Each query follows the Bernoulli distribution and p is the query success probability; then n queries follow a binomial distribution, viz. $X \sim B(n, p)$. X's mathematical expectation is

$$E(X) = \sum_{i=0}^{n} ip(i) = \sum_{i=0}^{n} i\binom{n}{i} p^i (1-p)^{n-i} = np \quad (3)$$

Owing to $p_1 > p_2$, we can deduce $E(X_1) > E(X_2)$ according to Eq. (3). X_1 and X_2 represent the success query number of times under the capacity-based selection and random selection, respectively. It indicates that the frequency of the average query success based on capacity selection is higher than that based on random selection under the same total query times (n). Apparently, the success rate of capacity-based selection is greater than that of random selection.

As a result of the fact that success rate based on capacity selection is higher than that of random selection under one hop, it is apparent that the success rate is still higher under multi-hop. In contrast, the hop counts based on capacity selection is less than that based on random selection under the same success rate.

3 Experiments and Evaluation

When a node comes into a system, if it registers to an AN, it can be regarded as a grid node, and an AN's MDS can find its current resources status. If it is selected as a super-peer, it can deal with user resource application and it will provide resources located in it or cluster to clients. Each super-peer has an index table, which saves its neighbor super-peers status information of available resources. Neighbors can gain update-registering information from each other by periodic announcements.

When a client applies for resources, its node will submit an application to its super-peer. If the super-peer cannot satisfy a client's resources application, it will search its index table to find a neighbor super-peer to satisfy the client's requirements. If it still cannot find the target super-peer, the original super-peer will randomly select a neighbor super-peer on behalf of its client and continue the same query. In the process of the query, we set TTL to limit the query to be carried out in a VO. When TTL decreases to zero, the query will fail. Our experiments simulate the behaviors of client querying file resources and other resources such as free CPU, available bandwidth, memory, etc.

3.1 Experimental environments

The simulation program is made using VC++ 6.0. The network topology in our experiments adopts the random graph model based on WAXMAN. Making use of BRITE[18], an Internet topology generator, we generate multi-different network topology files aimed at various numbers of nodes. Simulation parameters and corresponding values used in our analysis are listed in Tab. 1. Each node bandwidth and the number of current connection nodes are generated in topology files. The number of file resources follows a geometric distribution, while the number of CPUs, memory and online time all follow a uniform distribution. These values are produced by Matlab random functions. Ref. [19] pointed out that cluster size in the KaZaA system was between 60 and 100, while ultra-peers in the Guntella system reached 30 to 40. So we set the proportion (γ) having super-peers from 1% to 5% in one VO. On the basis of experience, the weights for the properties of CPUs, bandwidth, memory, online time and number of connections are defined as 0.25, 0.25, 0.1, 0.2 and 0.2, respectively.

Tab. 1 Simulation parameters

Parameters	Value
Kinds of files	100
Number of idle CPU	{1, 2, 3, 4}
Available bandwidth/(Mbit·s^{-1})	10 to 1 024
Online time/h	4 to 40
Total nodes	6 000 to 10 000
TTL	2 to 7

Success rate is equal to X divided by n. Average hop count is total hop count divided by n on the condition of query suc-

cess. A successful query means that a super-peer's capacity can satisfy the user resources application. In order to explain the advantage of super-peers based on capacity selection (CS), we compare it with that of random selection (RS).

3.2 Experimental results

The total number of nodes is 8 000 in Figs. 1 to 3. The three figures are comparisons of super-peers based on capacity selection alone, where query resources are of a file type. The γ in both Fig. 1 and Fig. 2 is 3%. The total number of times of queries is 1 000. In Fig. 1, while the TTL is fixed, the success rate appears as an ascending trend as the duplicators of files rise from 80 to 120. File duplicate copies vs. success rates closely present a linear relationship. This is obvious and demonstrates that our super-peer on the basis of capacity selection is correct and feasible.

Success rate and average hop count increase with the

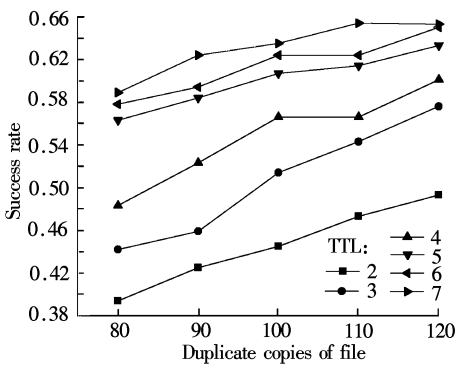

Fig. 1 Duplicate copies vs. success rate

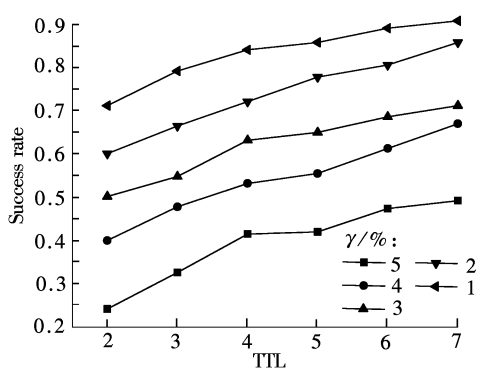

Fig. 2 TTL vs. success rate

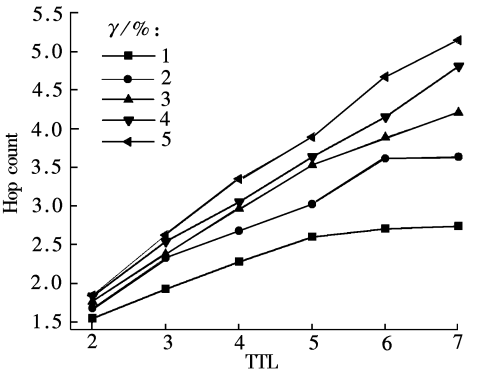

Fig. 3 TTL vs. hop count

value of the TTL shown in Fig. 2 and Fig. 3, respectively, while γ varies from 1% to 5%. As the cluster size steadily rises from 20 to 100, the success rate increases but the average hop count decreases; i. e., the average response time drops. Because the query mainly focuses on a small number of super-peers, increasing cluster size results in response time decreasing. However, it is not a good thing if the cluster size is very large or the number of super-peers is very small. Otherwise, it will add to the super-peer workload, and degrade network performance.

The success rate of CS and RS with different TTLs and different nodes are reported in Tab. 2 and Tab. 3 in which γ changes from 1% to 5% and query resources are of other types excepting file types. For the purpose of convenience and simplicity, we carry out resource queries using capacity. This method does not affect the performance analysis, because comparisons of the query success rate and the hop count are all based on the same capacity, TTL and γ. Capacity values in the simulation program are calculated according to Eq. (2), where each parameter generated is described in section 3.1.

In Tab. 2, the total number of nodes is 8 000. For a fixed number of super-peers, the success rate in CS is higher than that in RS. When γ changes from 1% to 5%, the success rate of CS becomes smaller and smaller as indicated in Fig. 2. However, the number of super-peers hardly makes any difference to success rates under random selection. Evidently, the success rates of both CS and RS are gradually higher with a rising TTL.

Tab. 2 Comparison of success rate between CS and RS with different TTLs

TTL	$\gamma=5\%$		$\gamma=4\%$		$\gamma=3\%$		$\gamma=2\%$		$\gamma=1\%$	
	CS	RS	CS	RS	CS	RS	CS	RS	CS	RS
2	0.242	0.160	0.401	0.124	0.502	0.125	0.601	0.131	0.712	0.111
3	0.326	0.184	0.478	0.168	0.548	0.160	0.665	0.134	0.792	0.194
4	0.415	0.232	0.532	0.191	0.632	0.239	0.721	0.220	0.841	0.236
5	0.420	0.278	0.555	0.243	0.650	0.238	0.778	0.269	0.858	0.278
6	0.474	0.281	0.613	0.273	0.686	0.272	0.806	0.330	0.891	0.346
7	0.477	0.300	0.670	0.362	0.712	0.289	0.858	0.340	0.908	0.358

Tab. 3 Comparison of success rate between CS and RS with different nodes

Number of nodes	$\gamma=5\%$		$\gamma=4\%$		$\gamma=3\%$		$\gamma=2\%$		$\gamma=1\%$	
	CS	RS	CS	RS	CS	RS	CS	RS	CS	RS
6 000	0.462	0.352	0.623	0.246	0.720	0.255	0.807	0.263	0.936	0.254
7 000	0.415	0.269	0.511	0.354	0.656	0.283	0.688	0.240	0.796	0.264
8 000	0.455	0.261	0.554	0.263	0.692	0.214	0.717	0.217	0.874	0.233
9 000	0.395	0.257	0.521	0.250	0.595	0.214	0.622	0.269	0.735	0.254
10 000	0.412	0.250	0.581	0.260	0.650	0.206	0.740	0.234	0.842	0.212

From Tab. 3, it is observed that the success rates of both CS and RS do not clearly change at the same γ. Considering the same nodes, the success rate of CS still increases with cluster size dropping, while the success rate of RS does not do any better, which explains that the super-peers based on capacity selection have better stability.

Assuming the cluster size is 40, the number of connections represents load. When all super-peers are saturated and a new super-peer is selected, super-peer load has been balanced using SSABC. Fig. 4 demonstrates that the load on each super-peer has decreased.

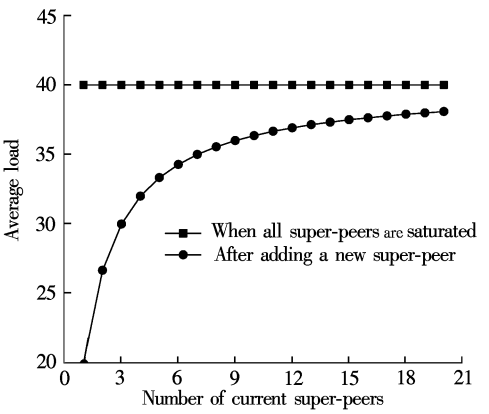

Fig. 4 Load balance when all super-peers are saturated

4 Conclusion

It has been acknowledged that super-peer structure can be applied to a hybrid system consisting of P2P and grid as well as pure P2P, and how to select high performance super-peers is a key problem in these models. However, researches on selecting super-peers have mostly been done in pure P2P systems. Using MDS4. 2 index services to discover dynamic available resource and using resource properties to compute node capacity, we present an algorithm for selecting super-peers according to nodes capacities in P2P and grid-based hybrid systems. Experimental results demonstrate that the success rate of capacity selection is obviously higher than that of random selection, and the average hop count is lower than that of random selection. Therefore, selecting higher capacity nodes as super-peers in the hybrid system merging P2P and grid can improve query success rate and decrease query time, which provides a good platform for sharing resources. But it needs to be determined further that the value of γ which can achieve the biggest success rate and the smallest delay as well as a more rational TTL number. Moreover, future work will study how node joining and quitting affect super-peer selection and system stability.

References

[1] Li Deng, Liu Hui, Chen Zhigang, et al. IPBGA: a hybrid P2P based grid architecture by using information pool protocol [C]//*ICA3PP* 2007. Hangzhou, China, 2007: 356 − 367.

[2] Andrade N, Costa L, Germóglio G, et al. Peer-to-peer grid computing with the OurGrid community [C]//*Proc of the* 23*rd Brazilian Symposium on Computer Networks* (*SBRC*)—4*th Special Tools Session*. Fortaleza, Brazil, 2005.

[3] Amoretti M, Reggiani M, Zanichelli F, et al. SP2A: enabling service-oriented grids using a peer-to-peer approach [C]// *Proc of* 14*th IEEE International Workshops on Enabling Technologies*: *Infrastructure for Collaborative Enterprise*. Linköping, Sweden, 2005: 301 − 304.

[4] Uppuluri P, Jabisetti N, Joshi U, et al. P2P grid: service oriented framework for distributed resource management [C]// *IEEE International Conference on Services Computing*. Chicago, USA, 2005: 347 − 350.

[5] Adriana I, Ian Foster. A peer-to-peer approach to resource location in grid environments [C]//*Proc of* 11*th IEEE International Symposium on High Performance Distributed Computing*. Edinburgh: IEEE Computer Society, 2002: 419 − 435.

[6] Talia D, Truno P. A P2P grid services-based protocol: design and evaluation [C]//*Proc of the European Conference on Parallel Computing*. Pisa, Italy, 2004: 1022 − 1031.

[7] Yang Beverly, Garcia-Molina H. Designing a super-peer network [C]//*Proc of the* 19*th International Conference on Data Engineering* (*ICDE*). Bangalore, India, 2003: 49 − 60.

[8] Montresor A. A robust protocol for building super-peer overlay topologies [C]//*Proc of the International Conference on Peer-to-Peer Computing*. Zurich, Switzerland, 2004: 202 − 209.

[9] Li Juan, Son Vuong. An efficient clustered architecture for P2P networks [C]//*Proc of* 18*th International Conference on Advanced Information Networking and Applications*. Fukuoka, Japan, 2004, **1**: 278 − 283.

[10] Lo Virginia, Zhou Dayi, Liu Yuhong, et al. Scalable supernode selection in peer-to-peer overlay networks [C]//*Second International Workshop on Hot Topics in Peer-to-Peer Systems*. San Diego, California, USA, 2005: 18 − 27.

[11] Min Su-Hong, Holliday Joanne, Cho Dong-Sub. Optimal super-peer selection for large-scale P2P system [C]//*Proc of International Conference on Hybrid Information Technology*. Cheju Island, Korea, 2006, **2**: 588 − 593.

[12] Pasquale C, Domenico T, Carlo M, et al. A superpeer model for multiple job submission on a grid[C]//*Euro-Par* 2006 *Workshops*: *Parallel Processing*. Dresden, Germany, 2006: 116 − 125.

[13] Mastroianni C, Talia D, Verta O. A super-peer model for building resource discovery services in grids: design and simulation analysis [C]//*European Grid Conference* (*EGC* 2005). Amsterdam, the Netherlands, 2005: 132 − 143.

[14] Puppin D, Moncelli S, Baraglia R, et al. A grid information service based on peer-to-peer [C]//*Proc* 11*th Euro-Par Conf*. Lisboa, Portugal, 2005: 454 − 464.

[15] Chen Guilin, Zhao Shenghui, Hou Zhenfeng. A model of integrating P2P technology and grid technology [J]. *Journal of Hefei University of Technology*: *Natural Science*, 2007, **30** (6): 676 − 680. (in Chinese)

[16] Globus Project. Information services (MDS) [EB/OL]. (2006-08-20) [2007-06-20]. http://www. globus. org/toolkit/docs/4. 2/info/key-index. html.

[17] UDDI, Version 3. 0. 2 [EB/OL]. (2004-10-19) [2007-10-20]. http://www. oasis-open. org/committees/uddi-spec/doc/spec/v3/uddi-v3. 0. 2-20041019. htm.

[18] Boston University. Internet topology generator-BRITE [EB/OL]. (2001-04-12) [2007-10-20]. http://www. cs. bu. edu/brite/.

[19] Liu Qiong, Xu Peng, Yang Haitao, et al. Research on measurement of peer-to-peer file sharing system [J]. *Journal of Software*, 2006, **17**(10): 2131 − 2140. (in Chinese)

SSABC:一种基于能力的超级节点选择算法

赵生慧[1,2] 钱 宁[1] 吴国新[1] 陈桂林[2]

([1] 东南大学计算机网络和信息集成教育部重点实验室,南京 210096)

([2] 滁州学院计算机科学与技术系,滁州 239012)

摘要:结合 P2P 和网格的特点,提出了从 P2P 与网格混合的分布式网络中选择超级节点的算法 SSABC. 算法使用网格信息监控系统(MDS)获取节点资源的动态属性信息,如可用带宽、空闲 CPU、可用内存、当前连接数及在线时间等,根据以上属性计算节点的能力. 当有新节点加入且超级节点均饱和时,从新节点或已加入节点中选择能力最高的作为新的超级节点. 通过理论分析和仿真实验表明,基于能力选择的超级节点与随机选择的超级节点相比,提高了资源的查询成功率,缩短了平均查询跳数,并能够在超级节点饱和时均衡网络负载. 当网络中节点数发生变化时,以上结论依然成立,说明了算法的可行性和稳定性.

关键词:对等网;网格;超级节点;能力选择;随机选择

中图分类号:TP393