Grid Service Discovery Based on Cross-VO Service Domain Model

Jing-Ya Zhou, Jun-Zhou Luo and Ai-Bo Song

School of Computer Science and Engineering, Southeast University,
210096 Nanjing, P.R. China
{jyz, jluo, absong}@seu.edu.cn

Abstract. The diversity of grid service originates from heterogeneous and dynamic nature of grid, and it poses a great challenge to grid service discovery. How to discover services satisfying users' multiple requests meanwhile avoiding negative effect derived from requests and updates becomes increasingly important in grid environment. This paper proposes a Cross-VO (Visual Organization) service domain model for compensating deficiencies that traditional approaches exhibit in flexibility of discovery. Service domain is developed to make advantage of similarity among services. In this model each service domain is constructed through all services that have similar function in VOs and nodes in service domain connect according to unstructured P2P system. It breaks through resource restriction in a VO and satisfies users' requests in great extent while achieves favorable scalability and flexibility. Both theoretical analysis and experimental results indicate that this model performs efficiently in high discovery success ratio, low average hops and messages even with low density and small TTL. Compared with non-domain grid system via the same discovery success ratio, our model outperforms it in both average hops and messages.

Keywords: Service Discovery, Service Domain, P2P.

1 Introduction

OGSA (Open Grid Service Architecture) [1] is a service oriented grid architecture that derives from computational grid and combining with Web Services forms a grid service oriented hierarchical integration architecture. In OGSA the virtualization of resources is embodied in the form of grid services and these services distribute in large scale grid environment. Hence, how to discover grid service satisfying users' requirements effectively becomes key issue in grid study.

VO is defined as a set of individuals and/or institutions defined by sharing rules and they share resources and cooperate with each other through a way of under controlled [2]. According to the collaboration, the service type provided by local VO usually meet users' requirements in higher probability, but service instances may not satisfy requirements due to resource restriction in one VO or higher requirements addressed by users. There are a large number of grid services with same type or similar functional property across VOs, but traditional discovery approaches do not take this

case into account. Consequently, we propose Cross-VO service domain model for grid service discovery. Service domain is composed of many services with similar functions. Efficient discovery strategy is designed based on the model too. Both theoretical analysis and experimental results indicate that the model we proposed can reduce updating load effectively and increase discovery success ratio.

The rest of this paper is organized as follows: Section 2 gives related work on service/resource discovery in grid environment. Section 3 introduces Cross-VO service domain model, then service discovery strategy is described. In section 4 we make performance analysis on theory. The experimental results and analysis is present in section 5. In section 6 we conclude the paper and look forward to future work.

2 Related Work

Grid service/resource discovery is somewhat special because of high dynamics in grid environment. Many studies have made their efforts to solve this problem.

Globus [3] uses MDS (Monitoring and Discovery Service) [4] to realize tree-like metadata directory service based on LDAP [5]. MDS is in charge of monitoring and discovery of grid resources, however, it focuses on service data query while lack of support for service type discovery. UDDI (Universal Description Discovery and Integration) is a specification for distributed Web-based information registries for Web services [6]. It allows services to be published, and subsequently searched, based on their interface, but it does not an automatic mechanism for updating the registry as services change. A. ShaikhAli, etc. present UDDIe as an extension to UDDI, which supports QoS (Quality of Service) dynamic registry and enable discovery of services based on QoS [7]. Unfortunately, it is a centralized model, in which central server in charge of all queries and inclines to failure in case of overload. Meanwhile frequent QoS update results in huge network overhead due to dynamic nature of grid. In this paper, we suggest that similar services should be aggregated together in a service domain, and then multitude dynamic update is restricted within the range of domain.

P2P shares many common features with grid, for example, both of them are large scale system constructed for the purpose of resource sharing; resources or services in system exhibit characteristics of strong autonomy, heterogeneity and high dynamics; nodes may participate or withdraw at any moment. P. Trunfio, etc. propose that two systems be converged for the discovery research, and three kinds of P2P systems are also analyzed in [8]. As to unstructured P2P system, A. Iamnitchi, etc. propose a fully decentralized P2P architecture for resource discovery in grid environment. In this architecture all nodes are equivalent and no one act as central server. The discovery process is the execution of traversal among all nodes, because of no central server it avoids single-point failure, nevertheless, it will appear high latency as the growth of network size. Chord [9, 10] is the first structured P2P system to be proposed. The discovery process emulates the binary search, thus requires O (logN) hops and messages. Compared with unstructured P2P system, structured P2P system is more scalable in terms of traffic load, but need to have strong self-organization capabilities in order to be able to maintain rigid structure. Hybrid P2P system has been proposed to overcome the drawbacks of aforesaid two systems while retaining their benefits. Hy-

brid P2P system is composed of two kinds of nodes: ordinary nodes and super nodes, in which several ordinary nodes are administrated by one super node and super nodes constitute a fully decentralized structure. There is no central server storing index structure, so it is no need to worry about the appearance of server bottleneck. Compared with unstructured P2P system, it has much faster speed for synchronization of index information and does not result in large traffic. Y. Gong, etc. put forward VEGA resource discovery framework in [11]. In this framework, several resource routers constitute management domain and are connect to backbone through border router. VEGA constructs a hybrid-like hierarchical P2P structure, and uses layered clustering approach to aggregate resource information. Through interaction between layers resource information are updated continuously. This architecture brings enlightening significance to our study. The concept of management domain is similar to VO in management perspective, however, it lacks of consideration for clustering management of similar resources.

3 Cross-VO Service Domain Model

3.1 Introduction to the Model

Service domain aggregates many types of service with similar function. It is similar to the conception of VO in architectural perspective, whereas, other than VO the former pays more attention to clustering of service providers of specific application field. Cross-VO service domain model is a hybrid hierarchical P2P structure. In the model, VO can be composed of several service domains while single service domain may be covered by several VOs. There are a VOSR (VO Service Registry) and many LDSRs (Local Domain Service Registry) located in VO. LDSR takes charge of registry of service information belong to a certain type, so LDSR represents a kind of service type, and the service information here is the detailed service description including static and dynamic information. As to provisional services we use factory pattern for registry, namely providers only register service handle for activating factory to LDSR, but no context and resources are allocated. The service handle associate with service type, and create service instances when needed. For further description of similarity, we introduce service compatibility to depict the substitutable relationship between different service types. If service type A is compatible with service type B, it indicates that user's requirements for instance of A can also be satisfied by instance of B. Apparently the introduction of service compatibility enhances discovery performance. In addition, it is notable that compatibility has no reflexivity. VOSR takes charge of recording and maintaining service type etc. static information gathered from LDSRs in local VO. LDSRs belong to a service domain are collected together to constitute a complete service domain. In service domain, LDSRs as nodes connect with one another according to unstructured P2P system.

Figure 1 shows an example of Cross-VO service domain model. Service domain III is covered by three VOs namely VO A, B and C, while VO A is composed of part of three service domains namely service domain I, II and III. On the VO level of this model, VOSRs of all VOs constitute an unstructured P2P system, then they corres-

pond to super nodes in hybrid P2P system. In each VOSR we set a cache for recording service domain information published from neighbors and publish information of its own to neighbors periodically.

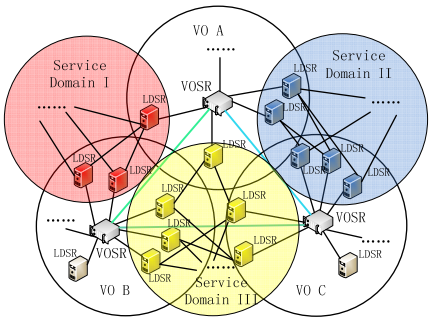

Fig. 1. Cross-VO Service Domain Model

3.2 Service discovery Strategy

In service discovery process, service request is dealt with according to distributed transmit strategy and describes as follows:

1. Users send request to local VOSR via LDSR.
2. VOSR receives request and makes some analysis, then lookup in service type list to determine if there exists item that match service domain that required service belongs to, if true, forward it to corresponding LDSR and go on, or else go to 5.
3. LDSR receives request and compares it with its own registry service type to determine whether they are the same or the registry service type is included in the compatibility list of required service, if not true, go to next step, or else continue to carry out service instance match in LDSR according to QoS etc. state information, if match success, then return discovery success and service information, otherwise, go to next step.
4. If request forwarding hops exceed TTL (Time To Live) return failure, otherwise, forward request to all its neighbors and go to 3.
5. Lookup cache for further match, if there exist item matching the required service domain, then forward request to the corresponding neighbor and go to 2, or else forward request to all neighbors and go to 2.

The above discovery strategy can be divided into two parts: discovery on VO level and discovery within service domain. The first part aims at finding service domain that required service belongs to. As service domain crosses VOs, each LDSR can be regarded as entrance from VO to service domain, it is equivalent to say that service domain has entrances among multiple VOs. It not only improves discovery success ratio, but avoids instability caused by node failure. After finding the service domain, it will go to the second part. The second part is responsible for finding satisfied service instance in service domain according to service type, state requirement etc.. It breaks through service resources restriction in a VO, meanwhile, it also solves the

problem of single-point failure and load balancing. When VOSR of VO A fails or overloads, LDSR 1 belonging to A will sends request to any of its neighbor LDSR 2 in the same domain instead of forwarding to VOSR of A, then LDSR 2 sends request to its own VO B and continue the following discovery process.

4 Performance Analysis

It is demonstrated that Internet topology follows power-law [12]. We assume that both inter-VO topology and intra-domain topology in Cross-VO service domain model obey power-law and theoretical analysis is given below.

Table 1. Symbols and Definitions

Symbol	Definition
N_{VO}	Number of VOs
N_D	Number of service domains
$N_{D\text{-}LDSR}$	Average number of LDSRs in a service domain
$P_{SUC}(h,t)$	Discovery success ratio within h+t hops
T(h,t)	Average hops under discovery success
M(h,t)	Average messages forwarded by single request
$P_{VO}(h)$	The probability of finding service domain that required within h hops
$P_D(t)$	The probability of finding satisfied service instance within t hops
T(h)	Average hops under service domain discovery success within h hops
T'(t)	Average hops in domain under service discovery success within t hops
M(h)	Average messages forwarded by single request within h hops on VO level
M'(t)	Average messages forwarded by single request within t hops in domain

Firstly, we take the first part of discovery process into consideration. According to lemma 2 in [12], the number of edges E on VO level, can be estimated as a function of N_{VO} and the rank exponent R: $E = \frac{1}{2(R+1)}(1-\frac{1}{N_{VO}^{R+1}})N_{VO}$, then substitute it into $d = \frac{2E}{N_{VO}}$, it goes into $d = \frac{N_{VO}^{R+1}-1}{(R+1)N_{VO}^{R+1}}$. d represents average degree on VO level, it can be seen that when $N_{VO} \to +\infty$, d tends to be constant $\frac{1}{R+1}$. Supposing that we search service with certain type x and x belongs to domain I. There are two possibilities to find domain I, let us donate by P_I the possibility that I is in the service type list, and P_{IC} the possibility of finding I in cache. Then, the possibility of finding I on VO level is shown to be:

$$P_{VO\text{-}I} = 1-(1-P_I)(1-P_{IC}) \quad (1)$$

Since I is covered by n_I VOs, we have: $P_I = \frac{n_I}{N_{VO}}$. The cache size of VOSR is set as its degree, we have got d, so $1-P_{IC} = \frac{\binom{N_D-1}{d}}{\binom{N_D}{d}} = \frac{N_D-d}{N_D}$, substitute it to equation (1), we obtain:

$$P_{VO\text{-}I} = 1-\frac{(N_{VO}-n_I)(N_D-d)}{N_{VO}N_D} \quad (2)$$

We use the following equation to calculate $P_{VO}(h)$:

$$P_{VO}(h) = 1-(1-P_{VO\text{-}I})(1-P_{VO}(h\text{-}1))^d = 1-(1-P_{VO\text{-}I})^{\frac{d^{h+1}-1}{d-1}} \quad (3)$$

Equation (3) shows that $P_{VO}(h)$ initially increases quickly as TTL h increases, then as $P_{VO}(h)$ approaches 1 the increase amplitude slow down gradually. Meanwhile, the increasement of $P_{VO\text{-}I}$ brings higher $P_{VO}(h)$, and we can improve $P_{VO\text{-}I}$ via heightening n_I, so n_I is also proportional to $P_{VO}(h)$. Let p_i donate the probability of finding service domain at exactly the *i*th hop, then $P_{VO}(h)$ is given by: $P_{VO}(h) = \sum_{i=0}^{h} p_i$. Now consider the probability of finding service domain at exactly the *i*th hop under service domain discovery success within h hops should be $p_i / P_{VO}(h)$, we have:

$$T(h) = \sum_{i=1}^{h} i\frac{p_i}{P_{VO}(h)} = \frac{1}{P_{VO}(h)}\left(\sum_{i=1}^{h-1} ip_i + hp_h\right) = \frac{1}{P_{VO}(h)}(P_{VO}(h-1)T(h-1) + hp_h)$$

Substitute $p_h = P_{VO}(h) - P_{VO}(h-1)$ to above equation and replace $P_{VO}(h)$ with equation (3), we obtain:

$$T(h) = h - \frac{h}{P_{VO}(h)} + \frac{1}{P_{VO}(h)}\sum_{i=1}^{h}(1-P_{VO\text{-}I})^{\frac{d^i-1}{d-1}} \quad (4)$$

By analyzing equation (4), we conclude that both h and $P_{VO\text{-}I}$ have relationship with T(h), increasing h exclusively may not always lead to continuous increase of T(h).

When the required service domain appears in service type list, discovery process go to the second part——intro-domain discovery, and then there is no messages generated on VO level, if matching in cache, a message is forwarded to corresponding neighbor. Otherwise, messages are forwarded to all neighbors. We let $a = 1-P_{VO\text{-}I}$, so M(h) is given by:

$$M(h) = 1\bullet(1-P_I)P_{IC} + (d+dm(h-1))a \quad (5)$$

where m(h-1) are messages generated within the following h-1 hops and can be calculated by following equations:

$$m(h-1) = 1 \bullet (1-P_I)P_{IC} + (d + dm(h-2))a = \sum_{i=0}^{h-1} (ad)^i (1+b) - 1$$

and we rewrite equation (5) as:

$$M(h) = b + \frac{ad((ad)^h - b - 1)}{ad - 1} \quad (6)$$

Equation (6) indicates that node degree corresponds to an exponential number of messages, and degree of node in WAN tends to constant, so decreasing messages requires reducing TTL h. But on basis of analysis on equation (3), reducing h may result in drop of success ratio largely. Therefore, we need to take both factors into consideration.

Given that we find service domain, then we reach intra-domain discovery process. Different from VO level, we do not set cache in LDSR for considering similarity of services in domain and update load. According to lemma 4 in [12], the average number of nodes within t hops is the function of hot-plot exponent H, where E' is the average number of edges in domain:

$$NN(t) = \frac{N_{D\text{-}LDSR} + 2E'}{N_{D\text{-}LDSR}} t^H - 1$$

Supposing that the required service type was S, and the number of nodes that provide this kind of service was N_S, then the density of type S is $D_S = \frac{N_S}{N_{D\text{-}LDSR}}$. We let m_c as the number of service types that are compatible with S. The probability that request can find at least one satisfied service instance within t hops is given by:

$$P_D(t) = 1 - (1 - P_M)^{NN(t) \sum_{i=1}^{m_c+1} D_i} \quad (7)$$

where P_M represents the probability of instance match. $P_D(t)$ shares the same change trend with $P_{VO}(h)$in equation (3). Combining equations (3) and (7) we obtain:

$$P_{SUC}(h,t) = 1 - (1 - P_{VO\text{-}I} P_D(t))^{\frac{d^{h+1}-1}{d-1}} \quad (8)$$

From equation (8), it is known that increasing match probability P_M, service density D_S, and number of compatible services m_c will increase $P_D(t)$, and further increase $P_{SUC}(h,t)$.

In terms of approaches for calculating average hops on VO level, we calculate intra-domain average hops as:

$$T'(t) = t - \frac{t}{P_D(t)} + \frac{1}{P_D(t)} \sum_{i=1}^{t} (1 - P_M)^{\frac{d^i - 1}{d-1}} \quad (9)$$

Average hops under discovery success are described as:

$$T(h,t) = h + T'(t) - \frac{1}{P_{SUC}(h,t)} \bullet \sum_{i=1}^{h-1} P_{SUC}(i,t) \quad (\mathbf{10})$$

T'(t) and t in equation (9) are not strictly inverse proportion relationship, T'(t) will keep stable on a range as $P_D(t)$ increases, and then as to T(h,t) we pay more attention to the impact of T(h).

When service request be satisfied, a success message will be returned, or else return failure message. Average messages within t hops are given by:

$$M'(t) = 1 \bullet P_M + (d' + d'm'(t-1))(1 - P_M)$$

Where m'(t-1) are messages generated within the following h-1 hops and d' is average degree in domain. Let $a' = 1 - P_M$, with a boundary of condition, we have:

$$m'(t-1) = (a'd')^{t-1} + \frac{(a'd')^{t-1} - 1}{a'd' - 1} \bullet (a'd' + 1 - a')$$

and then

$$M'(t) = 1 - a' + a'd'(1 + m'(t-1)) = \frac{2(a'd')^{t+1} - a'(a'd')^t - a'd' + a' - 1}{a'd' - 1} \quad (\mathbf{11})$$

Through decreasing a', namely increasing P_M, we can get smaller average messages. In terms of above approaches, overall average messages is calculated by:

$$M(h,t) = P_{VO\text{-}I} M'(t) + (1 - P_{VO\text{-}I})(d + dg(h-1))$$

where $g(h\text{-}1) = P_{VO\text{-}I} M'(t) + (1 - P_{VO\text{-}I})(d + dg(h-2))$

then

$$M(h,t) = P_{VO\text{-}I} M'(t) + \frac{ad((ad)^h - P_{VO\text{-}I} M'(t) - 1)}{ad - 1} \quad (\mathbf{12})$$

Above theoretical analysis indicates that discovery ratio, average hops and average messages are all mainly determined by three factors: node degree d or d', TTL h or t and probability of service instance match P_M. As network scale enlarge, node degree tends to be constant, if instance match probability keep unchanged, then it will be needed to choose suitable TTL to keep balance between discovery success ratio and average hops and messages.

5 Experiment

In this section, experimental environment is presented including our parameters setup. We also present metrics as well as the experimental results for performance evaluation.

5.1 Experimental Environment

SEUGrid is a grid system established for AMS-02 (Alpha Magnetic Spectrometer-02) project [13]. The AMS project is a large scale international collaborative project with the goal of searching in space for missing matter, antimatter and dark matter on the international space station. SEUGrid currently is used to deal with minitype vast data processing in MC (Monte Carlo) production. MC production aims at producing mass simulated data for particles analysis. Because there are many kinds of services is offered for different particles analysis, service discovery is needed to guarantee performance. All machines registered in SEUGrid are equipped with one or more type(s) of services. Cross-VO service domain model proposed in this paper is implemented in SEUGrid, and a service discovery strategy based on the model is also applied to it. We conduct our experiment in SEUGrid environment.

We divide experiment into two parts, the first part is used for performance comparison among different parameters in our model, only a kind of service type is considered; in the second part we compare our model with non-domain grid system, and requests are generated randomly without service type restriction. As to compatibility, parts of service types have one or two compatible service types.

Network topology affects performance of discovery strategy to some extent. We construct inter-VO topology according to power-law formula in [14] $f_d = \exp(8.03) * d^{-2.489}$ and intra-domain topology according to $f_d' = \exp(6.47) * d^{-2.489}$ and the node degree ranges from 2 to 10. Ten kinds of domains with two types of service in each one is registered to each VO, then there are 10*2=20 LDSRs included in each VO. Each LDSR is registered with a type of service instances, and the number of instances distributes in the range of 10 to 20.

$$N_{VO} = \sum_{d=2}^{10} f_d \approx 1000 \qquad N_{D\text{-}LDSR} = \sum_{d=2}^{10} f_d' \approx 200$$

The number of VOs is about 1000, and we set up 100 domains, hence, the average number of LDSRs:

$$N_{D\text{-}LDSR}=(1000*10*2)/100=200$$

We perform MC production on specified machines with high performance and divide generated data into many data blocks according to certain rule, then data blocks are transferred to several machines in each VO. Some of these machines are chosen randomly as request nodes every time.

5.2 Metrics

Three metrics are considered in the experiment. The former two are from user's perspective, while the latter is from the system's perspective.

1. Discovery Success Ratio: the percentage of satisfied requests of total requests, and can be divided into service domain discovery success ratio and intra-domain discovery success ratio respectively.

2. Average Hops: the mean of hops under service discovery success. We use average hops instead of response time as metrics to express search efficiency. It is divided into average hops on VO level and intra-domain average hops.
3. Average messages: the mean of messages generated by single request. We also divide it into average hops on VO level and intra-domain average hops.

5.3 Results

In order to avoiding influence of randomness, each group of experiment repeats for 100 times and all results are averaged. The discovery process is divided into two parts. Figure 3 shows that discovery success ratio of both parts initially increase quickly as TTL increases, when TTL reaches a certain value, the increase amplitude slow down and keep stable. This is because the number of domains arrived increases as TTL increases at initial time, afterward, the overlapping of service domains strengthened as TTL increases. Then the increase amplitude of number of domains slow down. When conduct intra-domain discovery, we set compatibility number as 1, and make comparisons between different densities. We find that the higher density the higher success ratio is, and success ratio become 1 when t is 3. Accordingly, when t comes to 3, overall success ratio is mainly dependent on service domain discovery success ratio.

The average hops in Figure 4 has the same trend with what Figure 3 reflects, and the points that change the trend are same too. This result is consistent with theoretical analysis. Compared with Figure 5 (a) and (b), we find that increasing density is one of the effective ways to reduce average messages, especially for large number of services and wide distribution.

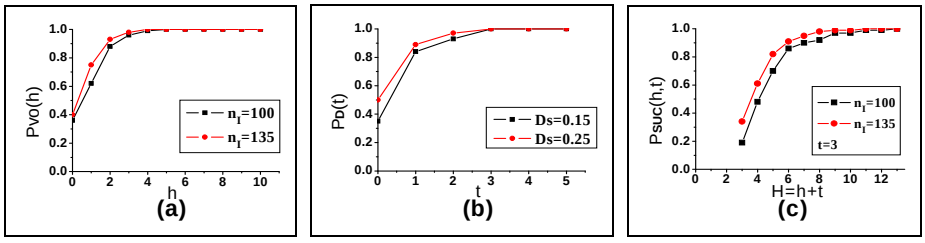

Fig. 2. Success Ratio and TTL

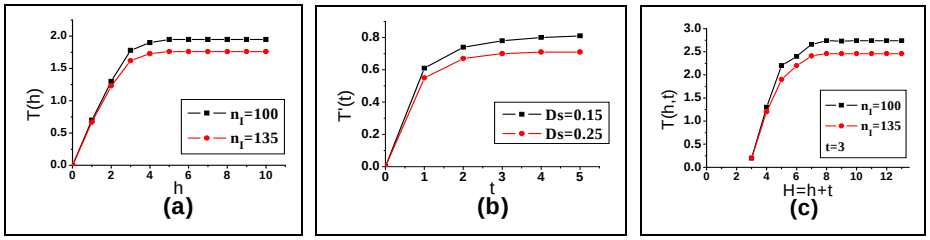

Fig. 3. Average Hops and TTL

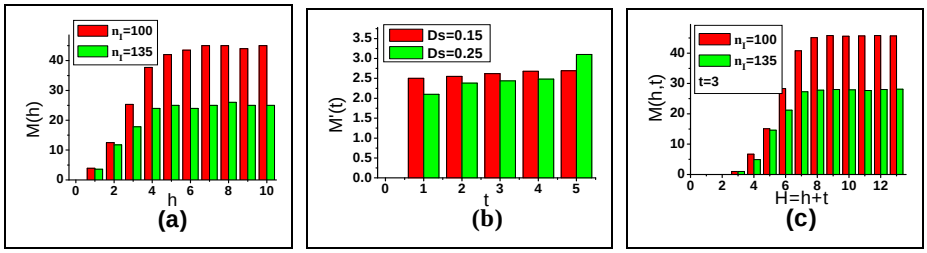

Fig. 4. Average Messages and TTL

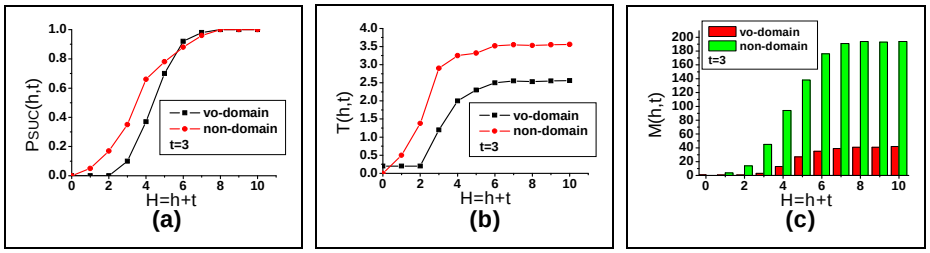

Fig. 5. Cross-VO Service Domain vs. Non-domain grid system

In the second part, as to non-domain grid system, we conduct searching by flooding on VO level and set cache size as node degree, but take no consideration of compatibility, in addition, we set t as 3. The other settings including topology and services information registered are same with service domain model. As Figure 6 describes, in (a) when H≤5, non-domain grid system keeps higher success ratio. The reason can be concluded that h is limited to 2 which do not arrive at equilibrium point, after exceeding the point, our model exhibits better performance. (b) indicates that non-domain grid system requires larger average hops under same discovery success ratio. Meanwhile, in service domain model, forwarding requests are restricted in a domain constructed by services with compatibility. This consequently reduces average messages greatly as (c) shown.

6 Conclusion and Future Work

This paper introduces service domain into grid system to make advantage of similarity among services as well as avoiding single-point failure and appearance of massive messages，and proposes Cross-VO service domain model. The whole discovery process is composed of service domain discovery and intra-domain discovery. The introduction of compatibility enhances discovery power of potential similar service resources, thus achieves favorable flexibility. We analyze factors on performance, and do experiment in SEUGrid to evaluate these factors, and compare it with non-domain

grid system. The experimental results show that Cross-VO service domain model we proposed can achieve high discovery success ratio, low average hops and messages.

Grid environment equipped with high dynamics requires updating frequently for correctness guarantee, especially for information in cache in our model. The performance impact of cache update will become our future work.

Acknowledgments. This work is supported by National Natural Science Foundation of China under Grants No. 90604004 and 60773103, Jiangsu Provincial Natural Science Foundation of China under Grants No. BK2007708, Jiangsu Provincial Key Laboratory of Network and Information Security under Grants No. BM2003201 and Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education under Grants No. 93K-9.

References

1. Foster, I., Kesselman, C., Nick, J. et al.: The physiology of the grid: An open grid services architecture for distributed systems integration (2002), http://www.globus.org/research/papers/ogsa.pdf
2. Foster, I., Kesselman, C., Tuecke, S.: The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal Supercomputer Applications 15(3), 200--222 (2001)
3. Globus Toolkit: http://www.globus.org
4. Czajkowski, K., Fitagerald, S., Foster, I., Kesselman, C.: Grid Information Services for Distributed Resource Sharing. In: International Symposium on High Performance Distributed Computing (HPDC'01), Proceedings, Redondo Beach (2001)
5. Howes, T., Smith, M.: A scalable, deployable directory service framework for the internet. Technical report, Michigan (1995)
6. Universal Description Discovery and Integration (UDDI). http://www.uddi.org/pubs/Iru UDDI Technical White Paper.pdf
7. ShaikhAli, A., Rana, O., Al-Ali R., Walker, D.: UDDIe: An extended registry for web services. In: the Workshop on Service Oriented Computing: Models, Architectures and Applications, pp. 85--89 (2003)
8. Trunfio, P., Talia, D., Papadskis, H. et al.: Peer-to-Peer resource discovery in Grids: Models and systems. Future Generation Computer Systems 23(7), 864--878 (2007)
9. Stoica, I., Morris, R., Karger, D. et al.: Chord: A scalable Peer-to-Peer lookup service for internet applications. In: ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM'01), pp. 149--160(2001)
10. Krishnamurthy, S., EI-Ansary, S., Aurell, E., Haridi, S.: A Statistical Theory of Chord Under Churn. In: IPTPS 2005. LNCS, Springer, Heidelberg (2005)
11. Gong, Y., Dong, F., Li, W., Xu, Z.: VEGA Infrastructure for Resource Discovery in Grids. Journal of Computer Science & Technology 18(4), pp. 413--422(2003)
12. Faloutsos, M., Faloutsos, P., Faloutsos, C.: On power-law relationship of the internet topology. In: ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM'99), pp. 251--262 (1999)
13. Fisher, P., Klimentov, A., Mujunen, A., Ritakari, J.: AMS Ground Support Computers for ISS mission. AMS Note 2002-03-01 (2002)
14. Lu, D., Dinda, P.: Synthesizing Realistic Computational Grids. In: ACM/IEEE Supercomputing Conference (SC'03), Phoenix (2003)