Experimental Demonstration of a Self-organized Architecture for Emerging Grid Computing Applications on OBS Testbed

Lei Liu, Xiaobin Hong, Jian Wu, and Jintong Lin

P.O. Box 55#, Key Laboratory of Optical Communication and Lightwave Technologies, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing, China
liulei.bupt@gmail.com,{xbhong,jianwu,ljt}@bupt.edu.cn

Abstract. As Grid computing continues to gain popularity in the industry and research community, it also attracts more attention from the customer level. The large number of users and high frequency of job requests in the consumer market make it challenging. Clearly, all the current Client/Server(C/S)-based architecture will become unfeasible for supporting large-scale Grid applications due to its poor scalability and poor fault-tolerance. In this paper, based on our previous works [1, 2], a novel self-organized architecture to realize a highly scalable and flexible platform for Grids is proposed. Experimental results show that this architecture is suitable and efficient for consumer-oriented Grids.

Keywords: optical Grid, self-organization, optical burst switching (OBS).

1 Introduction

Optical networking for Grid computing is an attractive proposition offering huge amount of affordable bandwidth and global reach of resources. Optical burst switching (OBS) [3], which combines the best of optical circuit switching (OCS) and optical packet switching (OPS), is widely regarded as a promising technology for supporting future Grid computing applications [4].

In recent years, the Grid over OBS architectures have been extensively studied [5-7]. The latest research proposed to use Session Initiation Protocol (SIP) to construct optical Grid architecture [8]. But to the best of the authors' knowledge, these architectures are all conventional C/S model, in which the roles are well separated. All the grid resources publish their available resource information to the corresponding server (or SIP proxy). The job request is firstly sent to a server (or SIP proxy) for resource discovery, and if none satisfied resource is found, the request will then transferred to other server (or SIP proxy) for further handling.

The Grid will open to the consumer market in the future, which is a challenge by the potentially large number of resources and users (perhaps millions), high frequency of job requests and considerable heterogeneity in resource types. Under this background, C/S-based optical Grid architecture will become unfeasible due to its poor scalability and poor fault-tolerance. In order to address this issue, a P2P-based architecture is proposed in [1]. Compared with the conventional C/S solution, the significant improvement of the P2P-based architecture is the resource discovery scheme is fully decentralized. Experimental results in [1] show the P2P-based architecture can improve the network scalability and it is a better choice for large-scale Grid applications. However, the shortcoming of the P2P-based solution is only the non-network resource is considered for resource discovery. So in the work of [2], we propose a self-organized resource discovery and management (SRDM) scheme, in which both the network resource and non-network resource are taken into account for resource discovery. But the resource reservation is not efficient enough in [2] and additional time delay will be introduced. So in this paper, based on [1] and [2], a novel self-organized architecture for optical Grid is investigated. Experimental results show that this architecture is suitable and efficient for Grid applications. Moreover, it outperforms our previous works [1, 2] for supporting future large-scale consumer-oriented Grid computing applications.

The rest of this paper is organized as follows. Section 2 proposes the self-organized optical Grid architecture. Section 3 investigates the signaling process in this architecture. Section 4 is the performance evaluation and experimental demonstration of the proposed architecture. Section 5 concludes this paper.

2 Self-organized Network Architecture

Fig.1 shows the self-organized architecture for Grid over OBS. The architecture is separated into 2 layers: transport layer and protocol layer.

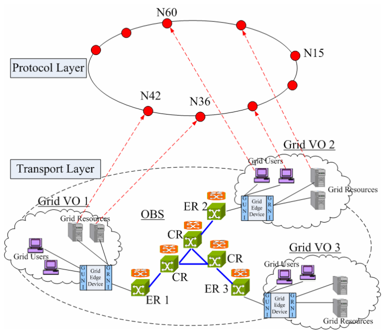
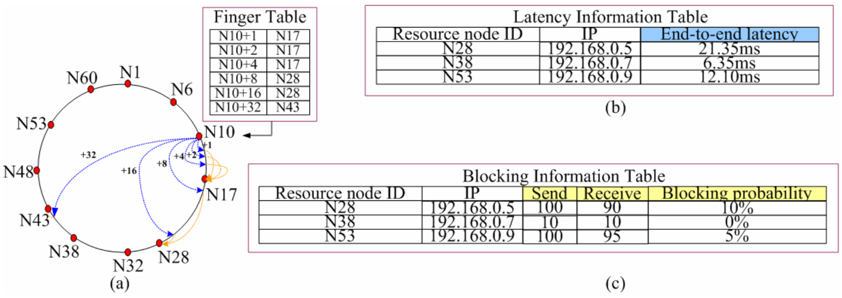


Fig. 1. Self-organized architecture for Grid over OBS

The transport layer is the actual Grid network. Fig.1 shows a typical Grid over OBS infrastructure [4], that is, the Grid users/resources are divided into several Grid virtual organizations which are connected through OBS network. Two interfaces, Grid User Network Interface (GUNI) and Grid Resource Network Interface (GRNI) are used to connect Grid and OBS network.

The protocol layer is used for implementing the resource discovery scheme. It is composed of the virtual nodes mapping from the Grid users/resources in the transport layer. The consistent hash function assigns each node in the protocol layer an m-bit identifier (node ID) using SHA-1[9] as a base hash function. A node's identifier is chosen by hashing the node's IP address. In the rest of this paper, the term "node" will refer to either the actual node in transport layer or the virtual node in protocol layer, as will be clearly distinguished from context.

In the self-organized architecture, each node maintains three tables, including Finger Table (FT), Latency Information Table (LIT) and Blocking Information Table (BIT). FT is a routing table (Fig.2(a)), which includes both the identifier and the IP address of the relevant node. Note that the first finger of n is the immediate successor of n on the protocol layer; for convenience we often refer to the first finger as the successor. The generation process and algorithm for FT is introduced in [1] in detail.



Finger Table

N10+1	N17
N10+2	N17
N10+4	N17
N10+8	N28
N10+16	N28
N10+32	N43

Latency Information Table

Resource node ID	IP	End-to-end latency
N28	192.168.0.5	21.35ms
N38	192.168.0.7	6.35ms
N53	192.168.0.9	12.10ms

Blocking Information Table

Resource node ID	IP	Send	Receive	Blocking probability
N28	192.168.0.5	100	90	10%
N38	192.168.0.7	10	10	0%
N53	192.168.0.9	100	95	5%

Fig. 2. An example of (a) FT, (b) LIT and (c) BIT of node 10 (N10)

LIT and BIT are used for storing the end-to-end latency and blocking probability from the current user to different resources respectively. Such information is obtained by self-learning mechanism. Once a resource is discovered, the IP address of this resource will be saved in LIT and BIT. For each resource, user periodically sends "Hello" signaling to it to get the end-to-end latency and save it in LIT. Meanwhile, user records the job history (success or failure), calculates job blocking probability for each resource and saves it in BIT. BIT is cleared in every T minutes to eliminate the out-of-data information. Fig.2(b) and Fig.2(c) show examples of the LIT and BIT of node 10.

3 Signaling Process

Fig.3(a) shows the signaling process for Grid applications in the self-organized architecture, which can be divided into 3 steps: preparation, resources discovery and job execution. A novel P2P protocol based on Chord [10] is integrated in this process.

3.1 Preparation

This step is used to generate Resource Publication Message (RPM), which can be further illustrated as Fig.3(b). In computational Grids, Grid resources can be divided into two types: static resources, including operation system configuration, system version, service types that this resource can provide, etc. and dynamic resources,

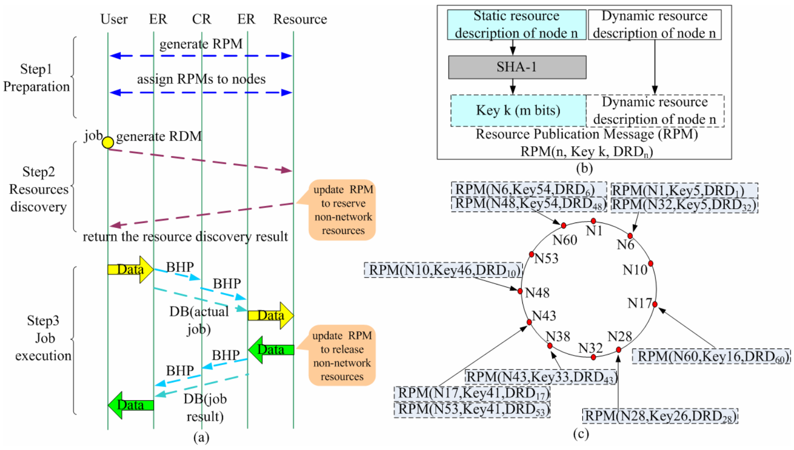


Fig. 3. (a) Signaling process (b) Resource Publication Message (RPM) generation (c) Protocol layer consisting of 11 virtual nodes storing 10 RPMs

including idle CPU cycles, available disk space, free RAM, etc. The dynamic resource will be changed in the process of Grid job execution while the static resource will remain unchanged.

Each resource in the Grid network is required to describe its available static resource and dynamic resource in the same description method and format which are needed to be negotiated by all the Grid users. This description format should be a structured naming or description, such as [11-12]. The RPM is composed of two parts. The top m bits are the key which is generated by hashing the static resource description using SHA-1, the rest bits are the dynamic resource description (DRD). The dynamic resource description is reversible and can be analyzed by every Grid user while the key is irreversible due to the SHA-1. The RPM of node n will be denoted by RPM (n, Key k, DRD_n) in the remainder of this paper.

After the RPMs are generated, they will be published to nodes residing in the protocol layer, which is an identifier circle modulo 2^m. As shown in Fig.3(c), RPM (n, Key k, DRD_n) is assigned to the first node whose identifier is equal to or follows key k in the identifier space. This node is called the successor node of key k, denoted by successor(k). If identifiers are represented as a circle of numbers from 0 to 2^m-1, then successor(k) is the first node clockwise from k.

3.2 Resource Discovery, Reservation and Release

The process of resource discovery, reservation and release is described in this section. Firstly, user can specify the job requirements and job characteristic (i.e. loss-sensitivity or delay-sensitivity) through a web portal in which dynamic Web Service technology is implemented. After that, a job to be executed remotely will generate a Resource Discovery Message (RDM) according to its job requirements. The top m bits of RDM are also the key which is the hashing of the static resource requirements and the rest bits are the description of dynamic resource requirements (DRR) and the network information (NI). The input and output for generating NI can be described as the following program. Based on the end-to-end latency/blocking information, the IP addresses of the resources ($IP_1,IP_2,...,IP_n$), which are stored in LIT and BIT, are sorted into increasing order ($IP'_1,IP'_2,...,IP'_n$). After that, the reordering permutation ($IP'_1,IP'_2,...,IP'_n$) is saved in NI. These are several sorting algorithm, which is investigated in detail in [13]. Fig.4 (a) shows the generation process of RDM. The RDM with key k will be denoted by RDM (Key k, DRR, NI) in the rest of this paper.

Program of the generation of network information (NI) in RDM

```
program
Input: (1)A sequence of IP addresses stored in LIT and
BIT, $IP_1,IP_2,...,IP_n$ and (2)their corresponding end-to-
end latency $L_1,L_2,...,L_n$ (Fig.2(b)) and (3)blocking
probability value $P_1,P_2,...,P_n$ (Fig.2(c))and (4)job
characteristic specified by the user (i.e. loss-
sensitive, delay-sensitive)

Output: A permutation (reordering) $IP'_1,IP'_2,...,IP'_n$ of
the input sequence such that $L'_1 \le L'_2 \le ... \le L'_n$ (delay-
sensitive case)or $P'_1 \le P'_2 \le ... \le P'_n$(loss-sensitive case)
end.
```

The resource discovery process can be described as follows: an operation, *find_successor*, is firstly invokes at node n to find the *key* of RDM. If *key* falls between n and its successor, *find_successor* is finished and node n returns its successor. Otherwise, n searches its finger table for the node n' whose ID most immediately precedes *key*, and then invokes *find_successor* at n'. The reason behind this choice of n' is that the closer n' is to *key*, the more it will know about the identifier circle in the region of *key*. When the *key* is found, the dynamic resource description will be compared to find out which node can meet the dynamic resource requirements of the job. After a list of candidate resources satisfying the specified requirement is obtained, i.e. list *L*, the ($IP'_1,IP'_2,...,IP'_n$) in NI will be compared to the IP addresses in *L* in order to choose a best resource (i.e. least latency or least blocking). First-fit mechanism is used here since the ($IP'_1,IP'_2,...,IP'_n$) in NI have been already sorted. If there is no relevant record in NI, an IP address is randomly selected from list *L* as resource discovery result. After a resource is chosen, non-network resource can be reserved by updating RPM. Together with OBS bandwidth reservation protocols (e.g. Just-Enough-Time (JET) [3]), the proposed self-organized architecture enables a more flexible end-to-end reservation (compared with [2]) of both network and non-network resources in a fully decentralized manner. Fig. 4(b) is

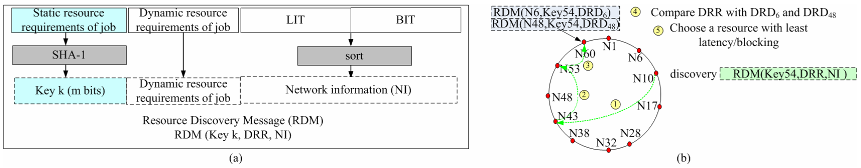

Static resource requirements of job
Dynamic resource requirements of job
LIT
BIT
SHA-1
sort
Key k (m bits)
Dynamic resource requirements of job
Network information (NI)
Resource Discovery Message (RDM)
RDM (Key k, DRR, NI)
(a)
RDM(N6,Key54,DRD6)
RDM(N48,Key54,DRD48)
4 Compare DRR with DRD6 and DRD48
5 Choose a resource with least latency/blocking
discovery RDM(Key54,DRR,NI)
N1
N6
N10
N17
N28
N32
N38
N43
N48
N53
N60
1
2
3
(b)

Fig. 4. (a) Resource Discovery Message (RDM) generation (b) Path of a query for RDM (Key54, DRR, NI) starting at node 10 an example that node 10 wants to find the successor of RDM(Key54, DRR, NI). The number (1~5) in Fig.4(b) shows the resource discovery procedures.

3.3 Job Execution

Once the resource discovery result is obtained, the user will send the actual job to Edge Router (ER) for transmission to the resource. ER aggregates the job into optical bursts which then are sent to the reserved resources by utilizing JET bandwidth reservation scheme. The edge router is able to send data from different users to different reserved resources across the network. After the job is successful executed, the job results will be returned to the user, at the same time, updating the RPM to release the reserved non-network resources.

4 Experimental Setup, Results and Discussions

The experimental setup is shown in Fig.5. Grid users and resources were connected through JET-OBS testbed [14]. Various latency and blocking from users to resource 1 and 2 was introduced by injecting background traffic. The resource discovery signaling (e.g. Hello, PRS) were encapsulated into bursts for transmission for avoiding the O/E/O conversion and message processing delay. About 1000 jobs were randomly generated with random resource requirement, job characteristic and start time (Fig.7 (a)). OBS edge routers and core routers were connected with fibre links in which two DWDM data channels (1554.94nm, 1556.55nm) and one dedicated control channel (1.31μm) are included. Bit-rate for all channels is 1.25Gbps.

The experimental results (Fig.6) show that the proposed self-organized architecture can be well applied to optical Grid with different job burst size (Fig.6 (a, b)) and different job request frequency (Fig.6 (b, d)). The burst together with eye diagram (Fig.6(d, e)) show that 19.43dB extinction ratio can be achieved. In our experiment, the shortest resource discovery time is 31.25ms (Fig.7 (b)) and the longest resource discovery time is 640.625ms (Fig.7 (c)). For all the Grid jobs, the average resource discovery time is nearly 200 milliseconds and the lookup successful rate is 100%.

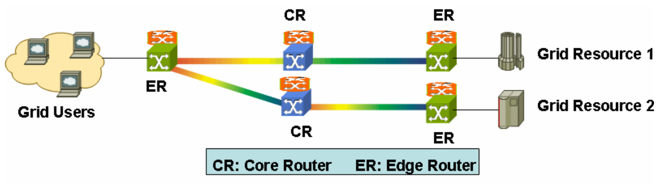


Fig. 5. Experimental setup

The results in Fig.7(d) show that self-organized architecture outperforms P2P-based architecture [1] in terms of job blocking and end-to-end latency since the resource discovery in the self-organized architecture is capable of consideration of both network and non-network resources. It can be seen that in the self-organized architecture, each user has its own intelligence to manage resource discovery requests

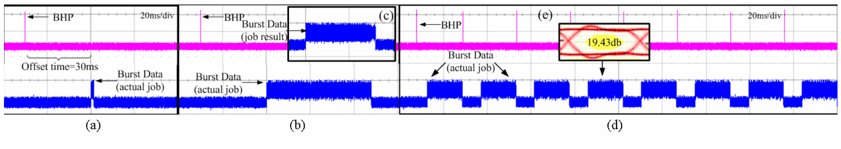
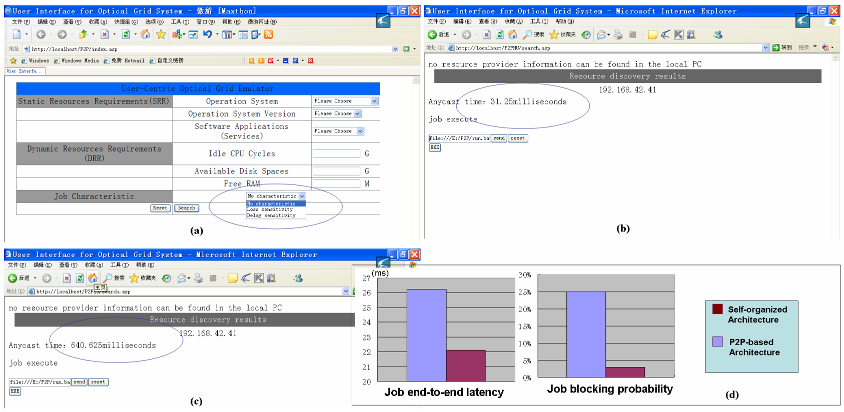


Fig. 6. Experimental results (a) a small job encapsulated in a burst to transmit (b) a large job encapsulated in a burst to transmit (c) job result sent back to the user (d) several job transmission when the job request frequency is high (e) eye-diagram of job bursts


User Interface for Optical Grid System - 傲游 [Maxthon]

http://localhost/P2P/index.asp

User-Centric Optical Grid Emulator		
Static Resources Requirements(SRR)	Operation System	Please Choose
	Operation System Version	Please Choose
	Software Applications (Services)	Please Choose
Dynamic Resources Requirements (DRR)	Idle CPU Cycles	G
	Available Disk Spaces	G
	Free RAM	M
Job Characteristic		No characteristic / No characteristic / Loss sensitivity / Delay sensitivity
Reset	Search	

(a)

User Interface for Optical Grid System - Microsoft Internet Explorer

http://localhost/P2PHU/search.asp

no resource provider information can be found in the local PC

Resource discovery results

192.168.42.41

Anycast time: 31.25milliseconds

job execute

file:///E:/P2P/run.ba send reset

EXE

(b)

User Interface for Optical Grid System - Microsoft Internet Explorer

http://localhost/P2PHU/search.asp

no resource provider information can be found in the local PC

Resource discovery results

192.168.42.41

Anycast time: 640.625milliseconds

job execute

file:///E:/P2P/run.ba send reset

EXE

(c)

Fig. 7. Experimental results (a) Web interface (b) shortest resource discovery time (c) longest resource discovery time (d) comparison of self-organized architecture and [1]

and make proper decision based on its own information about the whole Grid network. Clearly, by employing this distributed mechanism, it is not necessary to deploy powerful centralized servers for storing Grid resource information, which enables to construct a more scalable and fault-tolerant network for large-scale consumer Grid.

5 Conclusions

In this paper, a novel self-organized architecture for optical Grid is proposed and this solution is experimentally demonstrated on OBS testbed. By introducing self-organization in optical Grid, the disadvantages of the C/S-based Grid architecture are solved and many benefits are introduced to optical Grid due to the inherent advantages of self-organization (e.g. flexibility, scalability, fault-tolerance, etc.). The network resource and non-network resource are all taken into account for resource discovery in this architecture, which will result in better performance than our previous works. The experimental results verify that this architecture is suitable and efficient for future large-scale consumer-oriented Grid computing applications.

Acknowledgments. This work was supported by 863 Program (2007AA01Z248), MOST Program (No.2006DFA11040), PCSIRT (No.IRT0609) and 111 Project (B07005).

References

1. Liu, L., Hong, X.B., Wu, J., Lin, J.T.: Experimental Demonstration of P2P-based Optical Grid on LOBS Testbed. In: Optical Fiber Communication Conference (OFC), San Diego, USA (2008)
2. Liu, L., Guo, H., et al.: Demonstration of a Self-organized Consumer Grid Architecture. In: European Conference on Optical Communications (ECOC), Brussels, Belgium (accepted, 2008)
3. Qiao, C., Yoo, M.: Optical Burst Switching (OBS)－a New Paradigm for an Optical Internet. J. High Speed Netw. 8(1), 69–84 (1999)
4. Nejabati, R. (eds.): Grid Optical Burst Switched Networks (GOBS). Technical report, Open Grid Forum (OGF), GFD.128 (2008)
5. Zervas, G., Nejabati, R., Wang, Z., Simeonidou, D., Yu, S., O'Mahony, M.: A Fully Functional Application-aware Optical Burst Switched Network Test-bed. In: Optical Fiber Communication Conference (OFC). Anaheim, California, USA (2007)
6. Vokkarane, V.M., Zhang, Q.: Reliable Optical Burst Switching for Next-generation Grid Networks. In: IEEE/CreateNet GridNets, Boston, USA, pp. 505–514 (2005)
7. Farahmand, F., De Leenheer, M., Thysebaert, P., Volckaert, B., De Turck, F., Dhoedt, B., Demeestert, P., Jue, J.P.: A Multi-layered Approach to Optical Burst-switched Based Grids. In: International Conference on Broadband Networks (BroadNets), Boston, USA, vol. 2, pp. 1050–1057 (2005)
8. Zervas, G., Nejabati, R., Simeonidou, D., Campi, A., Cerroni, W., Callegati, F.: SIP Based OBS Networks for Grid Computing. In: Tomkos, I., Neri, F., Solé Pareta, J., Masip Bruin, X., Sánchez Lopez, S. (eds.) ONDM 2007. LNCS, vol. 4534, pp. 117–126. Springer, Heidelberg (2007)
9. FIPS 180-1, Secure Hash Standard. U.S. Department of Commerce/NIST, National Technical Information Service. Springfield, VA (1995)
10. Stoica, I., Morris, R., Karger, D., Kaashoek, F., Balakrishnan, H.: Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In: ACM SIGCOMM, San Diego, USA, pp. 149–160 (2001)
11. Globus Project, The Globus Resource Specification Language RSL v1.0, `http://www.globus.org/toolkit/docs/2.4/gram/rsl_spec1.html`
12. Smirnova, O.: Extended Resource Specification Language, reference manual, `http://www.nordugrid.org/documents/xrsl.pdf`
13. Cormen, T., Leiserson, C., Rivest, R., Stein, C.: Introduction to Algorithm, 2nd edn. The MIT Press, Cambridge (2001)
14. Guo, H., Lan, Z., Wu, J., Gao, Z., Li, X., Lin, J., Ji, Y., Chen, J., Li, X.: A Testbed for Optical Burst Switching Network. In: Optical Fiber Communication Conf. (OFC). Anaheim, California, USA (2005)