# Dynamic Multi-resource Advance Reservation in Grid Environment

Zhi-Ang Wu and Jun-Zhou Luo

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

**Abstract.** How to guarantee user's QoS (Quality of Service) demands becomes increasingly important in service-oriented grid environment. Current research on grid resource advance reservation, a well-known and effective mechanism to guarantee QoS, fails to adapt to dynamic variability of grid resource, and imprecise deny of user's reservation request often happens. For this, new system architecture for advance reservation is proposed. SNAP (Service Negotiation and Acquisition Protocol) is extended to support this architecture. Then VRC (virtual resource container) is adopted to alleviate negative effect resulted from resource performance variability, and QDD (QoS deviation distance) based logical resource selection algorithm is addressed to decrease imprecise reject rate of reservation. At last, this new architecture is deployed to campus grid, and two illustrative experiments are conducted in campus grid too. Preliminary results show that it can alleviate negative influence of grid resource dynamic fluctuation and avoid imprecise reject of advance reservation request effectively.

## 1 Introduction

Services provide higher-level applications for large-scale, open environment. Thus, services can be used to implement and configure software applications in a manner that improves productivity and application quality [1]. With the emergence of OGSA (Open Grid Services Architecture) and WSRF (WS-resource framework), grid begins to be a use case for Web Services [2, 3].

QoS (Quality of Service) encompasses important functional and non-functional service quality attributes, which is the key issue in service-oriented grid. Since delivering end-to-end QoS is a critical and significant challenge, how to guarantee QoS becomes a hot issue in this area. Advance reservation is a well-known and effective mechanism to guarantee QoS. GRAAP(Grid Resource Agreement and Allocation Protocol) work group of GGF(Global Grid Forum) has defined advance reservation as [4]: an advance reservation is a possibly limited or restricted delegation of a particular resource capability over a defined time interval, obtained by the requester from the resource owner through a negotiation process.

K. Li et al. (Eds.): NPC 2007, LNCS 4672, pp. 13–22, 2007.

In this study, we propose enabling system architecture for multi-resource advance reservation, which can adapt to dynamic variability of grid resource. Efficient logical resource selection algorithm is designed and implemented. Preliminary experimental results indicate that dynamic multi-resource advance reservation strategy proposed in this paper can avoid negative influence of grid resource dynamic fluctuation and imprecise reject of advance reservation request effectively.

The contribution of our research is as follows:

1. We overview related work of multi-resource advance reservation in grid environment. Concept and possible states of advance reservation are concluded. Advance reservation states transition is analyzed.
2. New system architecture for multi-resource advance reservation is proposed. VRC (virtual resource container) is proposed firstly and embedded in this architecture too. It aggregates same kind of logical resource and binds reservation request with resource dynamically. Consequently, it can solve negative influence of grid resource dynamic fluctuation effectively, and enhance stability of grid system.
3. QDD (QoS deviation distance) based logical resource selection algorithm is implemented in our architecture. QDD describes distance between user's desired QoS requirements and resource QoS properties, and QDD calculation method is improved newly in this paper. Then, QDD based logical resource selection algorithm is proposed. It avoids imprecise reject of reservation request, which exists widely in previous reservation negotiation strategies.

## 2 Related Work

The draft document of advance reservation established by GRAAP surveys advance reservation functionality in batch scheduling system [4]. It considers advance reservation from the client's perspective, where the client may be a user or a super-scheduler. Advance reservation is defined as a possibly limited or restricted delegation of a particular resource capability over a defined time interval, obtained by the requester from the resource owner through a negotiation process. Possible states of advance are also proposed in this document.

SNAP (Service Negotiation and Acquisition Protocol) is proposed in [5], which provides lifetime management and an at-most-once creation semantics for remote SLAs (Service Level Agreements). Three different types of SLAs are included in SNAP. They are Task service level agreements (TSLAs), Resource service level agreements (RSLAs) and Binding service level agreements (BSLAs). BSLA associates a TSLA with the RSLA and the resource service capabilities should satisfy the task's requirements. BSLA binds a task with a certain resource statically. This static binding is good for systems without supporting advance reservation, because this binding does not consider resource status at job's runtime. When applying this static binding to support advance reservation, the availability of this bond resource at

runtime cannot be guaranteed, for resource may join or quit dynamically and the workload also varies dynamically in grid environment. So, this performance fluctuation of grid resource will increase reject rate of advance reservation in great extent. Aiming at this deficiency, new architecture for advance reservation is proposed in this paper. VRC in this new architecture binds task with resource dynamically. And by using queuing theory, it is proved virtual resource container can improve service ability without degrading quality of service.

A lot of negotiation strategies for advance reservation have been presented in previous research. Backup resource selection is added to reservation to guarantee quality of service [6], which clusters same type resource and selects backup resource according to resource availability. Priority based reservation strategy is proposed in [7]. QoS and contention-aware multi-resource reservation is addressed in [8]. Though these reservation strategies mentioned above are able to solve some problems in grid environment, there are at least two deficiencies. First, user's QoS requirements are rarely considered. And these few existing consideration on QoS is short of good representation and classification of grid QoS. Second, the criterion of denying user's reservation request is coarse granularity. Imprecise deny of request often happens. In fact, not satisfying some provisional QoS requirements should not result in deny of user's request. Our proposed resource selection algorithm considers multi-dimensional QoS parameters and tries to avoid this imprecise deny of request based on detailed research on grid QoS in our early work.

R. Al-Ali, etc. classify the service QoS properties into five QoS domains in [9]. From performance perspective, grid QoS should include accounting QoS, service reliability and service security. And from guarantee mechanism perspective, grid QoS includes service QoS and provisional QoS. Our early research classifies grid QoS parameters newly from performance perspective and hierarchical structure model is proposed [10]. It can represent QoS parameters very well and reflect the intuition of the grid QoS. Grid QoS indeed can also be divided into service QoS and provisional QoS according to guarantee mechanism. Service QoS includes all the QoS parameters which must be guaranteed during reservation process. Its values can not be degraded. But QoS parameters belong to provisional QoS can be degraded within a range during negotiation process. Service QoS and provisional QoS are considered respectively to avoid imprecise deny of user's reservation request.

## 3 States of Advance Reservation

The possible states of advance reservation are addressed in [4] as table 1 shows. It is seen that the lifecycle of advance reservation starts from Requested and ends in either of following four kinds of state: Declined, Cancelled, Terminated and Completed. An advance reservation that ends in Completed state is a successful reservation and that ends other three state is unsuccessful reservation which may be resulted from scheduler's refuser or user's canceler.

**Table 1.** States of Advance Reservation

| State | Description |
|---|---|
| Requested or In Negotiation | A user has requested, or is negotiating, a set of resources for a reservation. If the reservation is accepted/agreed, it goes to being booked. Otherwise, it becomes declined. |
| Declined | The reservation is not successfully allocated for some reason. |
| Booked | A reservation has been made, and will be honoured by the scheduler. From here, the reservation can become active, or be cancelled by the user or system, or be altered. |
| Booked, change requested/in renegotiation | A user is trying to alter the resources for the reservation prior to its starting. Goes back to booked state on success or failure. |
| Cancelled | A user or scheduler cancels the reservation prior to beginning. |
| Active | The reservation has started, but not ended. |
| Terminated | A user, or possibly the system, terminates an active reservation before the end-time. |
| Completed | The reservation continued until its end-point. |
| Active, change requested/in renegotiation | A user is trying to alter the resources for the reservation after the reservation has become active. Goes back to active state on success or failure. |

## 4 Multi-resource Advance Reservation Architecture

In order to support dynamic advance reservation, enabling system architecture is introduced, which is illustrated in figure 1. Logical resource encapsulates service ability providing to remote tasks. This service ability varies with local workload, however, the capability of resources is fixed. Various logical resources can be divided into different kinds according to their function. VRC is proposed in this paper to aggregates the same kind of logical resource to be a computing pool. When user submits reservation request, advance reservation interface provided by VRC negotiates with user broker. If this negotiation is successful, advance reservation queue is established in VRC. VRC is responsible for maintaining and scheduling advance reservation queue, and renegotiating when negotiation is booked. It also collects available logical resource information and refreshes dynamically.

Figure 1 shows two kinds of logical resource: computing resource and device resource. Two kinds of VRC aggregate these two kinds of logical resource. User broker submits advance reservation request to advance reservation interface, and then advance reservation queue is established. When an advance reservation element in queue arrives at its start time, VRC selects appropriate logical resource by logical resource selection manager. And it binds this reservation with selected logical resource dynamically. Then this advance reservation process is invoked and switches to active state.

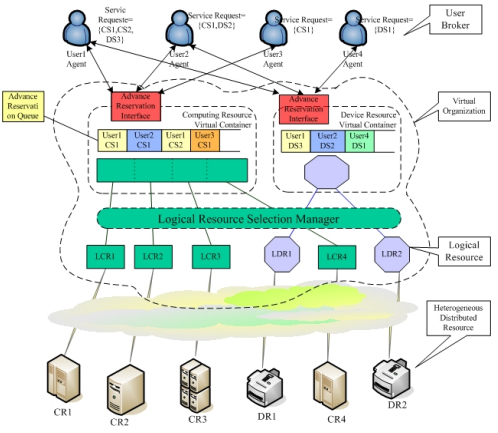


**Fig. 1.** Architecture for Advance Reservation

In this architecture, the key part is VRC, which aims to solve dynamic fluctuation of grid resource. So, the performance of VRC determines the performance of this advance reservation architecture.

By using of queuing theory, a computing pool including $c$ distributed supercomputers is proved that: while keeping the same average waiting time, the throughput of each supercomputer can be significantly improved by increasing the rate of receiving user's applications [11]. We assumed that the advance request arrival process is a Poisson process and a server has a service time which is an exponentially distributed random variable. A computing pool including $c$ distributed supercomputers is modeled as *M/M/c* queuing system. The average waiting time of *M/M/c* is deduced in queuing theory, and $Wq_c$ can be calculated by formula (1).

$$W_{q_c} = \frac{C(c,a)}{\mu_c(c-a)} \tag{1}$$

In formula (1) $a$ is workload ratio and $C(c,a)$ is the probability of all supercomputers are busy when an application arrives. These two parameters are derived from formula (2) and formula (3).

$$a = \frac{\lambda_c}{\mu_c} = \frac{c\lambda_1}{\mu_1} \tag{2}$$

$$(0 \leqq a<c)$$

$$C(c,a) = \frac{\frac{a^c}{c!(1-a/c)}}{\sum_{n=0}^{c-1}\frac{a^n}{n!} + \frac{a^c}{c!(1-a/c)}} \tag{3}$$

A *M/M/c* queuing system has the following properties: $Wq_c$ is the monotonic increasing function of $\lambda_1$, and $Wq_c$ is the monotonic decreasing function of $c$. So, a balanced condition exists: $Wq_c$ kept unchanged, while $\lambda_1$ can be increased with the growth of $c$. It is said that kept $Wq_c$ unchanged (means non-degraded quality of service), the rate of receiving user's application $\lambda_1$ (means service ability) will be increased with the growth of number of supercomputers aggregated by computing pool.

This result can be used to illuminate the VRC can improve service ability without degrading quality of service. So, the VRC can avoid negative influence of grid resource dynamic fluctuation effectively, but without degrading the performance of advance reservation.

## 5 Logical Resource Selection Manager

### 5.1 Calculation of QoS Deviation Distance

QoS deviation distance is used to quantify the extent that grid resource satisfy or dissatisfy the user's QoS requirements. It is the basis of negotiation and is calculated by formula (4) in [12].

$$\varepsilon = \sqrt{\sum_{i=1}^{m} (q_i - r_i)^2} \tag{4}$$

Where:
User request vector is : $Q=\{q_1,q_2,\ldots\ldots q_m\}$
Resource property vector is: $R=\{r_1,r_2,\ldots\ldots r_m\}$

This calculation by method reflects the distance between user's QoS requirements and resource QoS properties, but it is not appropriate for all kinds of QoS parameters. To provisional QoS, smaller QoS deviation distance denotes user is satisfied more. But to service QoS, once no resource can meet user's QoS requirement, this user's request should be denied, no matter how small QoS deviation distance is. When applying formula (4) to advance reservation negotiation algorithm, it selects resource with minimum QoS deviation distance. But once one service QoS parameter can not be satisfied, this resource can not satisfy this user's request actually, even QoS deviation distance is so minimum. Then renegotiation should be executed, and the efficiency of negotiation process will be decreased in great extent. So we have to improve on the calculation method of QoS deviation distance.

Grid application is represented as a set of grid service requests and grid resource in one VO can be abstracted to a set of logical resources [10]. Grid application $A=\{S_1,S_2,\ldots,S_m\}$, SNAP establishes TSLA for each grid service, which can be seen as a vector of QoS parameters, $S_i=\{q_1,q_2,\ldots,q_l\}$. Grid resource $R=\{R_1,R_2,\ldots,R_n\}$, RSLA is also established by SNAP, $R_i=\{r_1,r_2,\ldots,r_l\}$.We assume there exists $k$ service QoS parameters and namely $l$-$k$ provisional QoS parameters. So, TSLA and RSLA can be represented as : $S_i=\{q_1,q_2,\ldots,q_k,q_{k+1},\ldots,q_l\}$, $R_i=\{r_1,r_2,\ldots,r_k,r_{k+1},\ldots,r_l\}$.

QoS parameters are divided into negative criterion and positive criterion [13]. Negative criterion denotes the higher value the lower quality, such as response time. Positive criterion denotes the higher value the higher quality, such as CPU cycle and memory size. We define characteristic value $c_j$ to represent whether user's j-th QoS parameter is satisfied. $c_j$=1 represents it is satisfied perfectly, or $c_j$=0. $c_j$ of negative and positive criterion are calculated by formula (5) and (6) respectively.

$$c_j = \begin{cases} 1 & r_j - q_j \le 0 \\ 0 & r_j - q_j > 0 \end{cases} \tag{5}$$

$$c_j = \begin{cases} 1 & r_j - q_j \geq 0 \\ 0 & r_j - q_j < 0 \end{cases} \tag{6}$$

$$h_i = \prod_{j=1}^{k} c_j \tag{7}$$

$h_i$ denotes whether all service QoS parameters of the *i-th* grid service request are satisfied perfectly. We calculate $h_i$ by formula (7). If $h_i$=1, it shows that all $c_j$=1, $1 \leq j \leq k$, namely each service QoS parameter can be satisfied. And if $h_i$=0, it shows that at least one service QoS parameter can not be satisfied and this user request will be denied. After all service QoS parameters were satisfied, QoS deviation distance is calculated to determine the distance between resource performance and requirement. So, QoS deviation distance calculation formula is redefined as follows.

$$\varepsilon = \begin{cases} +\infty & h_i = 0 \\ \sqrt{\sum_{j=k+1}^{m} (q_j - r_j)^2} & h_i = 1 \end{cases} \tag{8}$$

### 5.2 QDD-Based Logical Resource Selection Algorithm

Logical resource selection manager is used to select current appropriate logical resource to complete reservation request in queue. Our selection algorithm is based on QoS deviation distance, which tries to select logical resource with minimal QoS deviation distance.

A successful reservation is abstracted to three tuple: advance reservation request ID, logical resource ID and a time interval in the future. We assume $Re_{mn}$ is reservation matrix, $Re_{mn}[i][j]=t_{ij}$ denotes the completed time $t_{ij}$ that resource $R_j$ is delegated to $S_i$. So, the main purpose of this selection algorithm is finding a near optimal reservation matrix. The input of this selection algorithm is advance reservation requests queue, named as *AR_Queue*, and logical resource set, named as *LR_Set*. The element of *AR_Queue* should include necessary parameters of advance reservation, and it is defined as a triple $< t_i^{start}, t_i^p, Q_i >$, in which $t_i^{start}$ is the start time of reservation, $t_i^p$ is duration of this reservation and $Q_i$ is QoS parameters of this reservation request. The output is reservation matrix $Re_{mn}$. The selection algorithm is described as follows.

```
(1)While AR_Queue is not null
(2)Si=AR_Queue.out()
(3)    If tcur==tstart
(4)      For j=1 to j=n
(5)         Determiningεij using formula (8)
(6)         Qdmn [i][j]=εij
(7)      End for
(8)         For each eligible resource in increased order
   according toε
(9)        Assume current resource is LRj
```

```
(10)        If ∩_{k=1}^{i-1} (t_kj < t_i^start)
(11)          Re_mn[i][j]=t_i^start+t_i^p
(12)          Break
(13)         End if
(14)     End for
(15)   End if
(16) End While
```

The advance reservation requests queue *AR_Queue* is ordered by start time of requests. When the queue head arrives at its start time, it calculates QoS deviation distance matrix $Qd_{mn}$, which denotes the QoS deviation distance between grid service request $S_i$ and logical resource $LR_j$. Then available resource for each request is of in increasing order according toε. And available resource is delegated to service request whose ε is small as possible as we can. Reservation matrix $Re_{mn}[i][j]$ records completed time of this reservation request, which is designed only to be convenient to confirm the availability of resource and start time can be gained from advance reservation request triple.

## 6 Experimental Results

Our experiment is conducted in SEUGrid(Southeast University Grid), which is developed based on Globus Toolkit. SEUGrid is designed for AMS-02(Alpha Magnetic Spectrometer) experiment. The AMS experiment is large-scale international collaborative project, and it is the only large physics experiment on the international space station [14]. Before AMS-02 is launched, SEUGrid is mainly used to process mini-type vast data in the MC (Monte Carlo) production. MC production is a kind of typical parameter sweep application, which is executed with a distinct set of parameters. There are also no inter-task communication or data dependencies in MC production.

Because to execute a MC production job requires large mount of computational and storage resource, resource advance reservation mechanism is needed to guarantee QoS. System architecture for advance reservation proposed in this paper is implemented in SEUGrid. And QoS oriented logical resource selection algorithm is applied to SEUGrid too. We use part of computational resources in SEUGrid to conduct our experiments.

In this experiment reservation requests include HPL (High Performance Linpack) testing jobs and MC production jobs. HPL jobs are much smaller than MC production jobs. According to adjusting HPL problem size, executing time is limited from 10min to 30min. And the executing time of MC production job is often limited from 1 hour to 4 hours. In order to simulate the fluctuation of grid resource, all grid nodes produce local HPL request randomly. We assumed that once CPU execute local HPL testing job, this CPU is seen as unavailable, which cancels accepted remote reservation request in this time interval and will not accepts any other remote reservation request at all. We produce 2400 reservation requests spreading over 24 hours, namely 50 requests per 30 minutes. The success rates of reservation with VRC and without VRC is compared in figure 2 – each point represents the success rate in a 30 minutes interval.

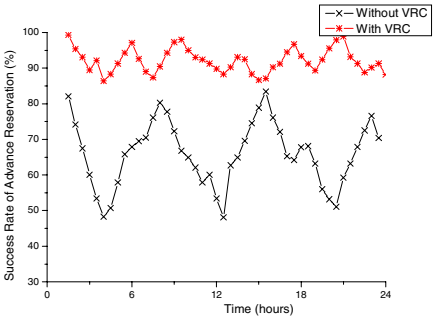


**Fig. 2.** Comparison of Reservation Success Rate

During one day, the average success rate with VRC reaches 92%, while without using VRC is only 66%. This result shows that VRC constantly achieves higher overall success rate, though resource performance varies dynamically in grid environment. And the range of curve without VRC is 35.3%, but using VRC reaches 12.9%. This result indicates that using VRC results in good stability of grid system.

## 7 Conclusion and Future Work

In the work described in this paper, main research focuses on alleviating negative influence of grid resource dynamic fluctuation and avoiding imprecise reject of advance reservation request. Two solutions, which are VRC and QDD-based selection, are embedded in advance reservation architecture. Performance evaluation and experimental results all indicate that dynamic multi-resource advance reservation strategy proposed in this paper can solves these two difficulties mentioned above effectively.

Workload balance and more flexible QoS supports for advance reservation will be researched and implemented in near future.

## Acknowledgement

This work is supported by National Natural Science Foundation of China under Grants No. 90412014 and 90604004 and Jiangsu Provincial Key Laboratory of Network and Information Security under Grants No. BM2003201.

## References

1. Huhns, M.N., Singh, M.P.: Service-oriented Computing: key concepts and principles. Internet Computing 9(1), 75–81 (2005)
2. Foster, I., Kesselman, C., Nick, J.M., Tuecke, S.: The physiology of the grid: An open grid services architecture for distributed systems integration (2002), http://www.gridforum.org/ogsi-wg/drafts/ogsa_draft2.9_2002-06-22.pdf

3. Czajkowski, K., Ferguson, D.F., Foster, I., Frey, J., Graham, S., Seduknin, I., Snelling, D., Tuecke, S., Vambenepe, W.: The WS-resource framework (Version 1.0) (2004), http://www-106.ibm.com/ developerworks/library/ws-resource/ws-wsrf.pdf
4. MacLaren, J.: Advance reservations: State of the Art. In: GGF GRAAP-WG (August 2003), See Web Site at: http://www.fz-juelich.de/zam/RD/coop/ggf/graap/graap-wg.html
5. Czajkowski, K., Foster, I., Kesselman, C., et al.: SNAP: A protocol for negotiating service level agreements and coordinating resource management in distributed systems. In: Feitelson, D.G., Rudolph, L., Schwiegelshohn, U. (eds.) JSSPP 2002. LNCS, vol. 2537, pp. 153–183. Springer, Heidelberg (2002)
6. Li, C., Xiao, N., et al.: Selection and Advanced Reservation of Backup Resources for High Availability Service in Computational Grid. In: Li, M., Sun, X.-H., Deng, Q.-n., Ni, J. (eds.) GCC 2003. LNCS, vol. 3033, pp. 26–33. Springer, Heidelberg (2004)
7. Min, R., Maheswaran, M.: Scheduling Co-Reservations with Priorities in Grid Computing Systems. In: Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID.02), May 2002, pp. 266–266. ACM Press, New York (2002)
8. Xu, D., Nahrstedt, K., Viswanathan, A., Wichadakul, D.: QoS and Contention-Aware Multi-Resource Reservation. In: High-Performance Distributed Computing, 2000. Proceedings, August 2000, pp. 3–10 (2000)
9. Al-Ali, R., ShaikhAli, A., Rana, O., Walker, D.: Supporting QoS-Based Discovery in Service-Oriented Grids. In: International Parallel and Distributed Processing Symposium (IPDPS'03), Nice, France (2003)
10. Wu, Z., Luo, J., Song, A.: QoS-Based Grid Resource Management. Journal of Software 17(11), 2264–2276 (2006)
11. Liu, P., Shi, Y., Li, S.: Computing Pool—a Simplified and Practical Computational Grid Model. In: GCC 2003. LNCS, Springer, Heidelberg (2003)
12. Zhu, S., Du, Z., Lam, W.K.: Design and Application of Contractual Computing Meta-Service in Grid Environment. Chinese Journal of Computers 28(4), 487–494 (2005)
13. Jin, H., Chen, H., Lu, Z., Ning, X.: QoS Optimizing Model and Solving for Composite Service in CGSP Job Manager. Chinese Journal of Computers 28(4) (April 2005)
14. Luo, J., Song, A., Zhu, Y., et al.: Grid Supporting Platform for AMS Data Processing. In: Pan, Y., Chen, D.-x., Guo, M., Cao, J., Dongarra, J.J. (eds.) ISPA 2005. LNCS, vol. 3758, pp. 276–285. Springer, Heidelberg (2005)