계산 그리드에서의 시변 쿼럼 상태 예측을 기반으로 한 QoS 제약 자원 재구성 방법

(QoS-Constrained Resource Reconfiguration Scheme based on Temporal Quorum Status Estimation in Computational Grids)

김 병 상 † 윤 찬 현 †† 남 동 수 † 이 봉 한 †††
(Byungsang Kim) (Chan-Hyun Youn) (Dong Su Nam) (Bong Hwan Lee)

요 약 Quality of Service (QoS) 제약을 통한 자원관리 정책은 사용자에 의해 요구되는 서비스를 보장해줄 수 있는 장점을 가지고 있다. 쿼럼 시스템은 중복되어 저장되어 있는 데이타의 무결성과 가용성을 보장해 줄 수 있는 특성을 의미한다. 이 논문에서 우리는 사용자의 응용프로그램이 요구하는 시스템 자원 및 네트워크 자원의 QoS를 만족시킬수 있는 쿼럼 기반의 자원관리방법을 제안한다. 또한 우리는 쿼럼 시스템내의 자원에서 실행되고 있는 응용프로그램의 실행시간을 통하여 자원의 상태예측을 하고 기대되는 자원의 서비스 수준을 유지하기 위한 자원의 재구성기법을 제안하고 있다. 자원의 재구성은 현재의 가용한 쿼럼내의 자원의 집합을 요구되는 서비스 수준으로 새롭게 재구성하는 것을 의미한다. 자원 재구성 정책의 효율성을 평가하기 위해서 우리는 인공심장혈류해석 병렬 프로그램을 사용하였다. 실험결과는 자원의 재구성 방식이 자원의 재구성 방식을 채택하지 않은 경우와 비교해서 응용프로그램의 실행완료시간을 감소시킬 뿐 아니라 실행환경의 안정성을 증가시키고 있다.

키워드 : 계산 그리드, 자원의 재구성, 시변 쿼럼 상태 예측

Abstract Quality of Service (QoS)-constrained policy has an advantage to guarantee QoS requirements requested by users. Quorum systems can ensure the consistency and availability of replicated data despite the benign failure of data repositories. We propose a Quorum based resource management scheme, which includes a system resource and network resource, both of which can satisfy the requirements of application QoS. We also propose the resource reconfiguration algorithm based on temporal execution time estimation method. Resource reconfiguration means that reshuffling the current available resource set for maintaining the quality level of the resources. We evaluate the effectiveness of resource reconfiguration mechanism with a Heart Hemodynamics analysis. Our approach shows the increase of the stability of execution environment as well as decrease the completion time compare to the method that is not adapted the resource reconfiguration.

Key words : Computational Grids, Resource Reconfiguration, temporal Quorum status estimation

1. Introduction

Recently, Grid computing has distinguished itself from conventional distributed computing by focusing on large-scale resource sharing and innovative applications under the environment of widely-connected network. There are many architecture for the management of Grid networks and the existing ones are focused on some concrete aspects which do not cover the management of the Grid as a whole. (For example the Condor-G system[3,4] and Nimrod-G Grid resource broker[5,6].) Due to the complexity of Grid system, and the trend towards increased the complexity in both hardware/software and service requirements, the flexible management of the overall Grid system itself is becoming more and more important. Policy-based management

† 비 회 원 : 한국정보통신대학교 공학부
bskim@icu.ac.kr
dsnam@icu.ac.kr
†† 정 회 원 : 한국정보통신대학교 공학부 교수
chyoun@icu.ac.kr
††† 비 회 원 : 대전대학교 정보통신공학부 교수
blee@dju.ac.kr
논문접수 : 2004년 2월 4일
심사완료 : 2004년 7월 13일

(PBM)[7] shows a prominent approach and management architecture over previous traditional network management systems. Policy-based Management can guide the behavior of a network or distributed system through high-level declarative directives that are dynamically introduced, checked for consistency, refined and evaluated, resulting typically in a series of low-level actions [8]. The current PBM architecture has problems in the sense that it can only address a fairly limited set of issues and usually requires human intervention. Grid systems unfortunately suffer the same problems in terms of the scalability and autonomic management. It is time consuming and error-prone for a Grid administrator, resource manager or broker to configure their system manually. Furthermore it is extremely hard to configure their local resource while considering other domains in the whole Grid system.

Another important issue connected with autonomic management is that of high level satisfaction. Performance, reliability, and availability increase can be viewed as resource demanders and providers. A provider requires an efficient and effective use, such as a high utilization of the distributed components. To demanders, quantitative aspects reflect the Quality of Service (QoS) and Service Level Agreements (SLAs).

In this paper, we propose a resource reconfiguration algorithm based on the temporal deviation of the execution time. Resource reconfiguration means to reshuffle the current resources and application re-mapping from tasks to resources in order to support QoS. Resource reconfiguration provides a good solution for changing the degraded resources in an unpredictable environment. Our approach is to guarantee the resource QoS that is and hence issued from a user by monitoring the application execution time. Execution time reflects the current resource status. We use execution time perturbation analysis in order to estimate the resource status. We focus on the reconfiguration triggering point using the utility function which is the reward value of the estimated resource status.

2. Resource Quorum model and Resource State Estimation

To guarantee QoS in user's application, the Grid manager should assign the resources to each application. The Quorum based management system will define the QoS vector to each application from the profile at input. We use two resource management items, system resource and network resources, respectively. Each service requester must specify its QoS requirements for the Quorum Manager (QM) in terms of the minimum QoS properties for both the system and network resources. All resources are distinguished from others and ranked as a group of QoS level in terms of those resource descriptions. We present that resource description as a system resource vector $\overrightarrow{\theta_s}$ and a network resource vector $\overrightarrow{\theta_n}$.

We define the Resource Quorum Q_R which represents the current status of the resource. Each resource has its resource status vector which is represented both invariable and variable descriptions [10]. System resources can take the processor specification or memory size as invariables and processor utilization or available memory space as variable. Furthermore, network resource will have the link capacity with an invariable description and end-to-end available bandwidth, delay, data loss rate as variable specifications, such that

$$Q_R = \left\{ \left\langle \vec{\theta}_i, \vec{\theta}_{jk} \right\rangle \mid i, j, k = 1, \ldots\ldots, n \right\} \tag{1}$$

where $\overrightarrow{\theta_i}$ denotes the current available resource level of the system i and $\overrightarrow{\theta_{jk}}$ represents the current available resource level of the network j and k. A resource universe $R = \{r_1, \cdots, r_u\}$ assumes a collection of resources that can be taken in the administration domain. A resource, $r = <S,N>$, can be represented as undirected graph with in a system, S and their communication networks, N;

$$R = \{r_1, \ldots, r_u\} = \{< S_i, N_{ij} > \mid S_i = r_i, N_{ij} = r_i \times r_j \; i \neq j \quad i, j = 1, \ldots, u\} \tag{2}$$

R is an available resource universe which is a subset of resource universe U, and i and j are elements in the R. S_i is therefore a system resource that represents a computation node i and N_{ij} is a network resource that represents a communication network from system i to system j.

If we assume that each task has its QoS requirement issued from SLAs, all resources can be ranked by the performance or be grouped by its attribute in terms of the QoS level.

2.1 QoS-Quorum Model

A required QoS level represents the vector of the resource description and has the range between the minimum and maximum requirement. We denote it by q and Q, respectively. We define the QoS-Quorum, Q_A, which represents the required quality level for an application set A

$$Q_A = \{<[\vec{q}_i^k, \vec{Q}_i^k], [\vec{q}_{ij}^{kl}, \vec{Q}_{ij}^{kl}]> | i \neq j, i \text{ and } j = 1,\ldots,u \quad k \neq l, k \text{ and } l = 1,\ldots,m\} \tag{3}$$

where $\vec{q}_i^k$ and $\vec{Q}_i^k$ denote the minimum and maximum QoS level required for task k on the system resource i. $\vec{q}_{ij}^{kl}$ and $\vec{Q}_{ij}^{kl}$ represent the minimum and maximum QoS level required for communicating the task k in system resource i and task l in system resource j.

2.2 Available resource quorum and resource configuration

To achieve the reliability in resource management, we define an available resource Quorum, Q_{AR}, which is a selected from a resource set satisfying the QoS requirement from SLAs.

$$\mathbf{Q_{AR}} = \{ < S_i, N_{ij} > \subseteq R \mid \vec{q}_i^k \le \vec{\theta}_i \le \vec{Q}_i^k, \vec{q}_{ij}^{kl} \le \vec{\theta}_{ij} \le \vec{Q}_{ij}^{kl} \} \tag{4}$$

Where i, j = 1,...,n' and k, l = 1,...,m, and QAR is a set that satisfies a desired minimum QoS level of application. Then, we define the Resource Configuration Function, $F(A,Q_{AR})$, as follows:

$$F(\mathbf{A},\mathbf{Q_{AR}}) = \{< S_i^k, N_{ij}^{kl} >\} = \{< V^k, E^{kl} > \xrightarrow{Q} < S_i, N_{ij} >\}$$

where i, j = 1,...n' □ u and k, l = 1⋯m, (5)

where

$$S_i^k = \begin{cases} 1, \text{ if the } v^k \in A \text{ and allocated on } S_i \\ \phi, \text{otherwise} \end{cases}$$

$$N_{ij}^{kl} = \begin{cases} 1, \text{ if the } e^{kl} \in A \text{ and existed in} \\ \qquad\qquad \text{communication b/w } r_i \text{ and } r_j \\ \phi, \text{otherwise} \end{cases}$$

Resource configuration function is therefore different from general scheduling in term of QoS support. Resource configuration provides a more reliable scheduling because it assumes that the elements of the available resource quorum set (QAR) satisfy the user's desired quality level.

3. Temporal Quorum status estimation

An application execution time of v^k is defined as the computation time v^k on the system $S_i^k(\theta_i)$ and communication time $N_{ij}^{kl}(\theta_{ij})$ between v^k and v^l. The execution time of job v^k is given as follows:

$$Z_i^k = z\left[S_i^k(\theta_i)\right] + \sum_{\forall l} z\left[N_{ij}^{kl}(\theta_{ij})\right] \tag{7}$$

where $[\cdot]$ represent computation time for a system k and communication time of network k and l, respectively. Where l refers to all communication peers related to the application.

we obtain initial derivatives as follows.

$$\frac{d\theta}{dZ} = \frac{\theta}{Z} \text{ and } \Delta\theta = \frac{d\theta}{dZ} \cdot \Delta Z \tag{8}$$

As a result, we can obtain current resource status θ'

We estimate current resource utilization of each application activity. Assuming that the application represents its previous execution progress as an event, we can predict the current resource utilization on target application. Equation (7) shows the estimates of the execution time of the current resources. Current state can be modeled as previous state with temporal deviation.

Then, the state of the current system resource i is

$$\hat{S}_i^k(\theta_i)_{T_c} = S_i^k(\theta_i)_{T_c-1} + \frac{dS_i^k(\theta_i)}{dZ_i^k(\theta_i)} \Delta Z_i^k(\theta_i), \text{ where } \Delta Z_i^k(\theta_i) \text{ is } Z_i^k(\theta_i)_{T_c} - Z_i^k(\theta_i)_{T_c-1} \tag{9}$$

and the state of the current network between i and j is

$$\hat{N}_{ij}^{kl}(\theta_{ij})_{T_c} = N_{ij}^{kl}(\theta_{ij})_{T_c-1} + \frac{dN_{ij}^{kl}(\theta_{ij})_{T_c-1}}{dZ_{ij}^{kl}(\theta_{ij})_{T_c-1}} \Delta Z_{ij}^{kl}(\theta_{ij}),$$

$$\text{where } \Delta Z_{ij}^{kl}(\theta_{ij}) \text{ is } \Delta Z_{ij}^{kl}(\theta_{ij})_{T_c} - \Delta Z_{ij}^{kl}(\theta_{ij})_{T_c-1} \tag{10}$$

On the other hand, a resource configuration provides the logical topology that represents the computation and communication relation. To keep the application quality after resource allocation, the resource manager should guarantee the initial performance as a reward function from resource

allocation. After a time has elapsed, each reward function on resource allocation will be affected from the availability of the resources. To identify this thing described before, we require a useful measure. We define the utility functions, such as a system utility function and a network utility function, which present an availability of Quorum resources to guarantee the QoS. Utility functions of resource provide the current system and network performance including previous status. We denoted the minimum quality level of each task required for computation and communication in each system, $q_i^k(\theta_i)$ and network, $q_{ij}^{kl}(\theta_{ij})$. In the case of a system resource, $\hat{S}_i^k(\theta_i)$, the estimate of utility function at Tc is represented by,

$$\mu_i^k = \frac{\hat{s}_i^k(\theta_i) - q_i^k(\theta_i)}{q_i^k(\theta_i)} \tag{11}$$

If $\mu_i^k < 0$, we could determine that the system does not meet the QoS level. Similarly, in case of network resource, the estimate of network utility function, $\hat{N}_{ij}^{kl}(\theta_i)$, at Ti, is represented as,

$$\mu_{ij}^{kl} = \frac{\hat{s}_{ij}^{kl}(\theta_{ij}) - q_{ij}^{kl}(\theta_{ij})}{q_{ij}^{kl}(\theta_{ij})} \tag{12}$$

Furthermore if $\mu_{ij}^{kl} < 0$, we determine that the network does not guarantee the QoS level in the network. Therefore, the reward values on the resource configuration, sRC(A,Q_{AR}), is defined as follows.

$$sRC(A,Q_{AR}) = \left\{ \left\langle S_i^k, N_{ij}^{kl} \right\rangle \right\},$$

where

$$S_i^k = \begin{cases} \mu_i^k(T_c) = \sum_{t=0}^{T_c} \mu_i^k(t), & \text{if the } v^k \in A \text{ and allocated on } S_i \\ \phi, & \text{otherwise} \end{cases} \tag{13}$$

$$N_{ij}^{kl} = \begin{cases} \mu_{ij}^{kl}(T_c) = \sum_{t=0}^{T_c} \mu_{ij}^{kl}(t), & \text{if the } e^{kl} \in A \text{ and existed in communication b/w } r_i \text{ and } r_j \\ \phi, & \text{otherwise} \end{cases} \tag{14}$$

4. Two-stage resource reconfiguration algorithm

$\mu_i^k(T_c)$ and $\mu_{ij}^{kl}(T_c)$ are accumulated QoS reward values that are system and network resources. If the estimate of utility function is smaller than 0, it means that the resource does not match the desired QoS level. We should therefore identify the condition whether triggering the resource reconfiguration or not. The reconfiguration steps consist of System Resource Reconfiguration (SRR) and Network Resource Reconfiguration (NRR), respectively. When Resources violate the requirement of the desired QoS, the management system should decide how to configure the current resources. If the current resource configuration falls into a RR-condition, we reallocate the application and resource. Once triggered, the resource Quorum management system invokes the two-step resource Quorum reconfiguration procedure on sRC(A,Q_{AR},T_c) as shown in Fig. 1.

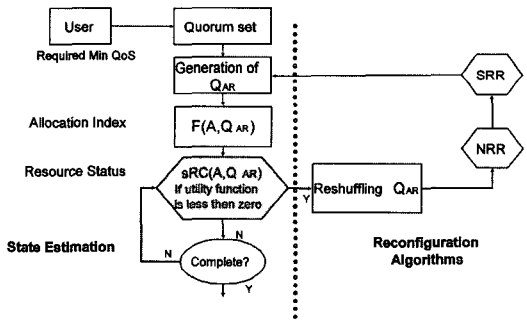


Figure 1 Operation procedure of the proposed a resource-Quorum reconfiguration scheme

In the reconfiguration step, after generating the initial QAR, it should be changed with the time. Before creating alternative configurations, we should obtain the pure QAR by updating QAR. For example, the reshuffling procedure governed by policy server is described as Table 1.

5. Experimental Results

We will evaluate the effectiveness of resource reconfiguration mechanism with a Heart Hemodynamics analysis application that is parallel application linked with each sub-job in order to examine the network reconfiguration as well as system reconfiguration. The parallel applications are connected to each other by the Grids. End-to-end communication quality of service in case of linked

Table 1 Resource reconfiguration algorithm

Stage 1.

If a state of current resource configuration is within reconfiguration condition Then reshuffling_procedure

① ***System should be joined in the Q_{AR}*** ***($S_{new} \leftarrow Q_{AR}$)***
S_{new} is newly discovered system that satisfy the desired QoS

② ***System should be removed in the Q_{AR} ($S_{corrupt} \rightarrow Q_{AR}$)***
$S_{corrupt}$ was allocated task k and this is degraded system that unsatisfied the desired QoS.

③ ***System should be removed after migrating the task on other systems in the Q_{AR} ($S_{corrupt} \rightarrow Q_{AR}$)***
$S^{k}_{corrupt}$ was allocated task k and this is degraded system that unsatisfied the desired QoS.
Go to Stage 2.

④ ***new $Q_{AR} = Q_{AR} + S_{new} - S_{corrupt}$***

Stage 2.

If NRR-condition (the network resource which utility function is less than zero).

① ***Choose the system resource that is reconfigured by comparing reward value of two systems.***
Go to SRR-condition ②

Else If SRR-condition (the system resource which utility function is less than zero).

② ***About unallocated system resource on the Q_{AR}***
If unallocated system resources exist, then Go to ③
Else if unallocated system resource does not exist, then resource reconfiguration is failed.

③ ***Search the maximum system resource***
If network quality level of selected system is satisfied the minimum QoS, and then select this system as alternatives. Go to ④
Else if does not satisfy 2-1, current candidate except from the alternative. Go to 2

④ ***Regenerate the resource configuration***
F(A, Q_{AR},Tc) and resource configuration status-**sRC(A,Q_{AR},Tc)**
on the RC(A, Q_{AR},Tc), $S_{candiate} = 1$, on the sRC(A,Q_{AR},Tc), $\mu_{candiate} = 0$

chains is more important issue because it is influenced on the entire performance of the application execution. Moreover the communication status has various reasons that cause degradation. In order to make point out the communication quality of service, we will examine the end-to-end bandwidth between each sub-job.

Blood flow in the sac of the Korean Artificial Heart (KAH) is numerically simulated by finite element analysis. A distributed computing algorithm

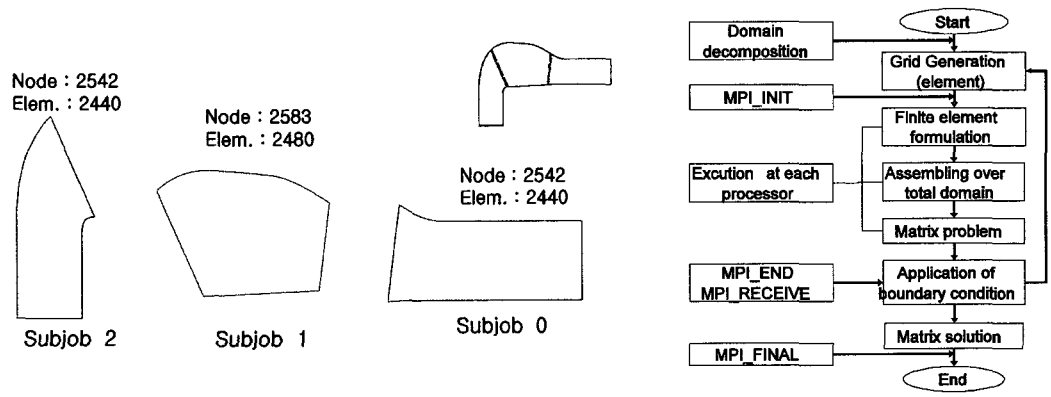


Figure 2 Flow chart of parallel computing program for the KAH

is employed to compute the hemodynamics of the sac using Globus-based MPI programming. Each sub-job of the KAH communicates with its neighbors in order to exchange the message for satisfying the boundary condition. Figure 2 shows the Boundary condition of the KAH and the flow chart for simulating the blood sac in the KAH. The program has the iteration characteristics that solve the velocity and pressure at each time step.

The Grid testbed for collaborating among multi-domain environments comprises Linux-based Globus platforms for Grid application. Figure 3 shows the testbed for this configuration between Information and Communications University (ICU) in Korea and MIT in US.

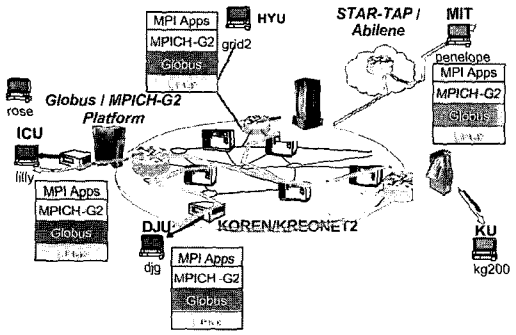


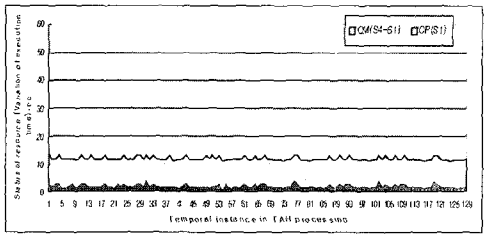
Figure 3 Grid Testbed for KAH modeling

Figure 3 shows the KAH is run on three domains, ICU, MIT and Hanyang Univeristy (HYU) and five machines (given in the Table 1). They are connected to the KOrea Research and Education Network (KOREN) and Science Technology And Research Transit Access Point (STAR-TAP). The system specification used in each university is shown in Table 2. In order to get the resource reconfiguration condition, we should set up minimum performance for execution and the resource requirement for executing the KAH.

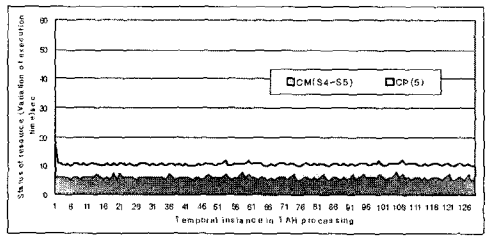
(a) and (c) in the Figure 4 present general execution time variation of system S1 and network S4-S5. We experiment with offered load at system S1 at the time instant T2 (time step 50) and network S4-S5 at the time step T3 (time step 100).

Table 2 Minimum requirement of available CPU ratio and available bandwidth

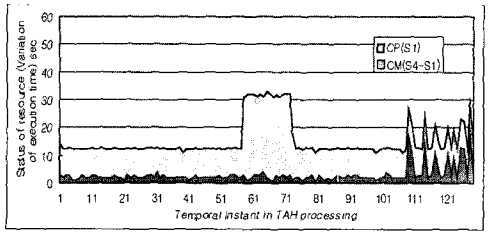
Resource numbering	Host name	Available CPU ratio (%)	Available BW (Mbps)
S1	grid2.hanyang.ac.kr	75%	0.1
S2	lilly.icu.ac.kr	30%	0.1
S3	tulip.icu.ac.kr	30%	0.1
S4	penelope.icu.ac.kr	35%	0.1
S5	violet.icu.ac.kr	30%	0.1



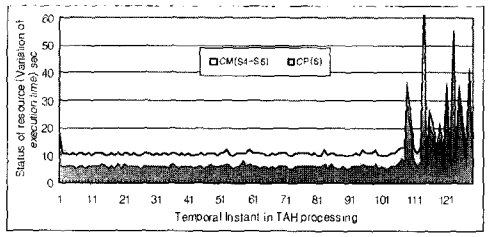
(a) General execution time variation at system S1



(b) General execution time variation at network S4-S5



(c) Offered load at system S1 at the time instant T2 (time step 50)



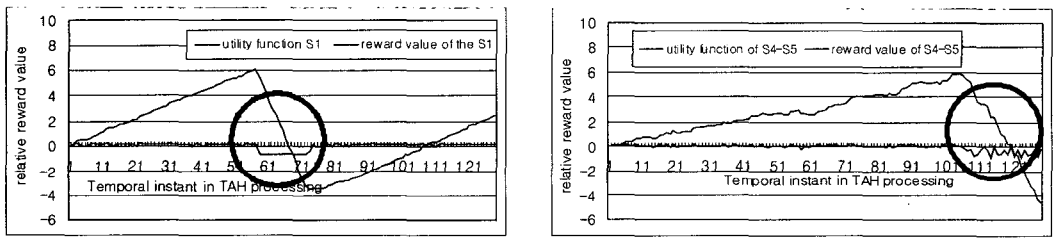
(d) Offered load at network S4-S5 at the time instant T3(time step 105)

Figure 4 Execution time of S1(a,c) and S5 (b,d)

T2 and T3 are the point of the time step 50 and time step 100 on the (b) and (d) of Figure 4 respectively. Figure 4 shows the execution time of system S1 and S5. On the (c) of Figure 4, system S1 increases the computation time on the time steps 50 to 75. Also, system S5 increases the communication time irregularly and rapidly between system S4 and system S5 between time steps 105 to 125 on the (d) of Figure5. The performance was evaluated according to each computation time (CP) and communication time (CM).

We calculate the reward value for the offered workload situation. Figure 5 shows the utility function and reward value of the system S1 and S5. As we can see, the reward value of the system S1 has a negative value about time-step 70 on the (a) of Figure 5. Also, the reward value of the network S4-S5 has negative value about time-step 115 on the (b) of Figure 5.

The zero point of the reward value of the resource is the reconfiguration triggering point. We take two reconfiguration points at time-step 70 and time-step 115 based on reward value from Figure 5. Figure 6 illustrates the reconfiguration policies that are generated by our reconfiguration algorithm. Initial topology has the system {S1, S4, S5}. After the reconfiguration of system S1 at T2, the system S1 replaced to S3 on the center part of the Figure 6. Also, on the reconfiguration of network S4-S5, the system S4 replaced to S1. The network topology has changed by the reconfiguration of the system. The network topology is shown in the



(a) utility function and relative reward value of the S1 (b) utility function and relative reward value of the network S4-S5

Figure 5 Utility function and the reward values (accumulated utility function value) of system S1 and network S4-S5

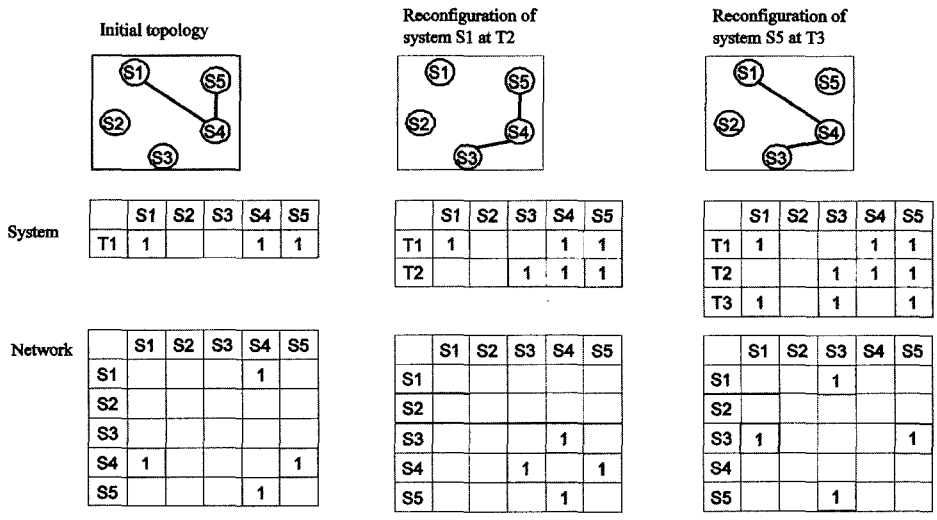
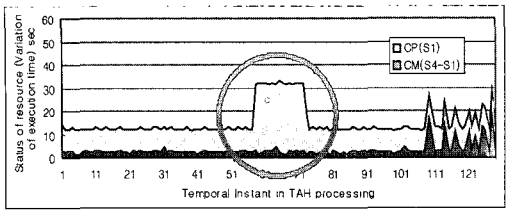
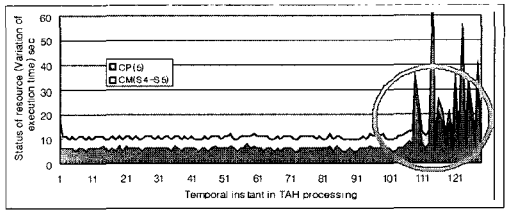


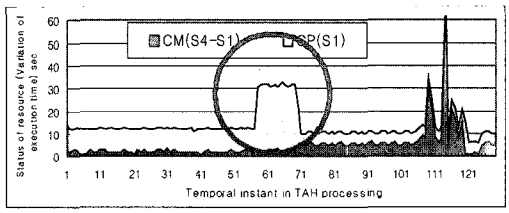
Figure 6 Two types of reconfiguration commitment on T2 and T3



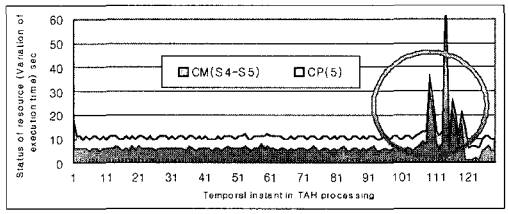
(a) Execution time of system S1 not committing resource reconfiguration



(b) Execution time of system S5 not committing resource reconfiguration



(c) Execution time of system S1 after committing resource reconfiguration



(d) Execution time of system S5 after committing resource reconfiguration

Figure 7 Execution time when proposed reconfiguration scheme has performed

lower part of Figure 6.

On the Figure 7, (a) illustrates the execution time in the case of not adapting resource reconfiguration and (b) is the execution time in the case of adapting resource reconfiguration at S1. also, (c) shows the execution time in the case of not adapting resource reconfiguration and (d) is the execution time in the case of adapting resource reconfiguration at system S5. Figure 7 shows that proposed reconfiguration scheme increase the stability of the execution time.

Figure 8 compares the total execution time between when not adapting resource reconfiguration and when adapting resource reconfiguration. As we can see, our resource reconfiguration scheme shows the total execution time reduction about 20%.

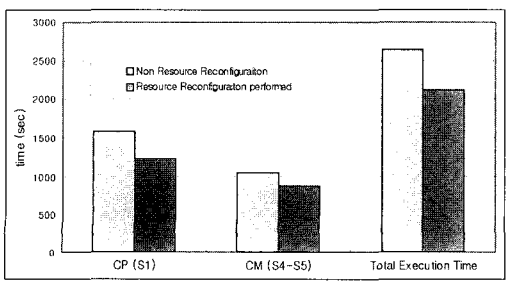


Figure 8 Comparison for Non Reconfiguration and Reconfiguration performed

Heart Hemodynamics Analysis Application shows that the effectiveness of system and network reconfiguration of the parallel application. As we compared, the proposed reconfiguration algorithm gives the stability of execution time as well as decreasing the completion time of the transaction of the application.

6. Conclusion

Computational Grids are focused primarily on high-performance distributed computing. In a wide area Grid environment, it is most important to guarantee the user's desired resource request. Reliable resource allocation should be maintained on the temporal and spatial change. Quorum based resource modeling and resource configuration scheme provides a more reliable resource scheduling. The proposed resource reconfiguration algorithm has two phases. One is the status estimation using temporal deviation. The other is resource reconfiguration based on estimated status. After triggering based on estimation, the reconfiguration algorithm tries to optimize the current system and network resource allocation. Our approach shows the increase in the stability of the execution environment as well as a decrease in the completion time compared to the method that is not adapted to

the resource reconfiguration.

References

[1] Foster, I. and Kesselman, C. (eds.). The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, 1999.

[2] [Foster, I. and Kesselman, C. The Anatomy of the Grid:Enabling Scalable Virtual Organizations. Intl J. Supercomputer Applications, 2001.

[3] Frey, J., Foster, I., Livny, M., Tannenbaum, T. and Tuecke, S. Condor-G: A Computation Management Agent for Multi-Institutional Grids, University of Wisconsin Madison, 2001.

[4] Raman, R., Livny, M., Solomon, M.: Matchmaking: Distributed resource management for high throughput computing. International Symposium on High Performance Distributed Computing (1998).

[5] Abramson, D., Sosic, R., Giddy, J. and Hall, B. Nimrod: A Tool for Performing Parameterized Simulations Using Distributed Workstations. In Proc. 4th IEEE Symp. On High Performance Distributed Computing, 1995.

[6] R. Buyya, D. Abramson, J. Giddy, and H. Stockinger, "Economic Models for Resource Management and Scheduling in Grid Computing," 2002, http://citeseer.nj.nec.com/buyya02economic.html.

[7] K. Yang, A. Galis, C. Todd "A Policy-based Active Grid Management Architecture," Proceedings of 10th IEEE International Conference on Networks (ICOIN02), pp. 243-248, IEEE Press. August 2002.

[8] P. Flegkas, P. Trimintzios, G. Pavlou, A. Liotta, Design and Implementation of a Policy-based Resource Management Architecture, Proceedings of IEEE/IFIP Integrated Management Symposium (IM'2003), Colorado Springs, Colorado, USA, pp. 215-229, Kluwer, March 2003.

[9] Franken, L.J.N. Haverkort, B.R, The performability manager, Network, IEEE performability manager.

[10] A.Leff, J.T.Rayfield, and D.M.Dias, "Service-Level Agreements and Commercial Grids," Internet Computing IEEE 2003, pp. 44-5.

[11] I. Cardei, S. Varadarajan, M. Pavan, M. Cardei, M. Min, "Resource Management for Ad-hoc wireless networks with cluster organization," Journal of Cluster Computing in the Internet, Kluwer Academic Publishers, Jan. 2004.

[12] S. R. Sarukkai and J. C. Yan. "Event-Based Study of the Effect of Execution Environments on Parallel Program Performance," Proceedings of the Forth International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOT '96), San Jose CA, February 1996. pages 257-261.

[13] Christos G, Cassandras "Discrete Event Systems, Modeling and Performance Analysis" IRWIN.

김 병 상

2002년 동국대학교 산업공학졸업(공학사) 2004년 한국정보통신대학교 전자통신공학 졸업(공학석사). 2004년~현재 한국정보통신대학교 전자통신공학과 박사과정 관심분야는 그리드 컴퓨팅, 유비쿼터스 컴퓨팅, 웹서비스 및 네트워크 관리

윤 찬 현

1981년 경북대학교 전자공학과졸업(공학사). 1985년 경북대학교 전자공학과졸업(공학석사). 1994년 일본 동북(東北)대학교 전자통신공학과 졸업(공학박사). 1986년~1997년 KT 초고속네트워크팀장. 1997년~현재 한국정보통신대학교 부교수. 관심분야는 그리드 미들웨어, 고성능 라우팅, 멀티캐스팅, 광인터넷 및 네트워크 성능 측정

남 동 수

2001년 홍익대학교 전기전자공학과 졸업(공학사). 2004년 한국정보통신대학교 전자통신공학 졸업(공학석사). 2004년~현재 국가보안연구소 연구원. 관심분야는 그리드 자원관리, 최적화 및 보안

이 봉 환

1985년 서강대학교 전자공학과 졸업(공학사). 1987년 연세대학교 전자공학과 졸업(공학석사). 1993년 Texas A&M 대학교 전기공학과 졸업(공학박사). 1987년~1995년 한국통신 연구개발원 연구원. 1995년~현재 대전대학교 정보통신공학과 부교수. 관심분야는 그리드컴퓨팅, 네트워크보안, 광인터넷