Journal of Digital Contents Society Vol. 10 No. 1 Mar. 2009(pp. 169-175)

그리드 네트워크에서의 QoS 보장방법 구현

김정윤,* 나원식**, 유인태***

요 약

그리드 컴퓨팅은 컴퓨팅 자원을 비용편익 측면에서 가장 효율적으로 만들 수 있는 능력을 가지고 있으며, 클러스터링 등의 기술로는 해결하기 어렵거나 시간이 오래 지체되어지는 대량의 컴퓨팅 능력을 요구하는 어려운 문제들을 풀기에 더 없이 좋은 방법 중에 하나이다. 이러한 그리드 컴퓨팅을 효율적으로 수행하기 위해서 지리적으로 분산되어 있는 고성능 컴퓨팅 자원을 실시간으로 상호 연결하기 위해 Grid Application이 필요하게 되었고, 이에 미국의 ANL(Argonne National Laboratory)를 주축으로 하여 여러 대학의 연구진에 의해 Globus가 만들어지게 되었다. 그러나, 이러한 Globus상에서 Grid Application을 실행시켜 일정의 Job을 네트워크를 통하여 주고받을 때, QoS가 보장되지 않는 문제점이 발견되었다. 그리하여 이러한 문제점을 해결하기 위하여 GARA(Globus Architecture for Reservation and Allocation)가 ANL에 의해서 개발 되었다. 본 논문에서는 이러한 GARA의 성능을 테스트하기 위하여 Testbed를 구축하여 GARA의 자원 예약 명령을 통해 자원을 예약을 수행하고, 그에 따른 적용 결과 및 추후 연구방향에 대하여 논하였다.

A Method on the Realization of QoS Guarantee in the Grid Network

Jungyun KIM*, Wonshin NA**, In-tae RYOO***

Abstract

Grid computing is an application to obtain the most efficient performance from computing resources in terms of cost and convenience. It is also considered as a good method to solve a problem that cannot be settled by conventional computing technologies such as clustering or is requiring supercomputing capability due to its complex and long-running task. In order to run grid computing effectively, it needs to connect high-performance computing resources in real-time which are distributed geographically. Answering to the needs of this grid application, researchers in several universities with Argonne National Laboratory in the USA (ANL) as the main axis have developed Globus. It is noticed, however, that the quality of service (QoS) is not guaranteed when certain jobs are exchanged through networks in the context of Globus. To tackle with this problem, the ANL has invented Globus Architecture for Reservation and Allocation (GARA). The researchers of this paper constructed a testbed for evaluating the ability to reserve resource in the GARA system and implemented the GARA code for it. We analyzed the applied results and discussed future research plans.

Keywords : Real-Time Scheduling, QoS Distributed System

1. 서론

그리드는 1990년대 미국의 슈퍼컴퓨팅 센터를 중심으로 슈퍼컴퓨팅 사이트들을 고성능 통신망으로 연결하여 사용하는 분산 시스템을 기반으로 제안되었다.

주로 Text 형태의 정보를 공유하고 있는 Web과는 달리, 고성능 컴퓨터와 대용량의 데이터베이스 등을 연동함으로써 데이터의 빠른 처리 및 협업 연산을 가능하게 한다.

그리드는 지구 관측, 천문, 기상정보 분석 등에 사용되어 질 수 있고, 대형 시뮬레이션등 무수한 많은 연구 분야에 사용되어 질 수 있다.

※ 제일저자(First Author) : 김정윤
접수일:2009.02.02 완료일:2009.03.30
* 경희대학교 컴퓨터공학과
inokyuni@khu.ac.kr
** 남서울대학교
*** 경희대학교 컴퓨터공학과

그리드는 슈퍼컴퓨터와 버금가는 성능을 낼 수 있으면서 가격이나 유지비용은 상대적으로 싸다. 클러스터링은 그리드와 비교하여 볼 때, 대용량의 연산을 수행할 때는 적합하지 않지만, 복잡하지 않는 시뮬레이션 등에서는 그리드 보다 훨씬 수월하게 시스템을 구성하여 사용할 수 있다.

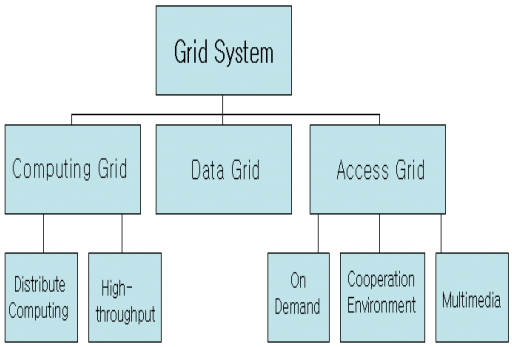
단순히, 그리드는 분산된 컴퓨터 이용에 있어서 거리의 문제점을 극복하기 위한 것이라고 표현할 수 있다.

그리드 기술은 On-demand 컴퓨팅을 가능하게 하여주고, 대규모 계산이나 정보를 수반하는 문제를 해결하기 위하여 필요로 하는 컴퓨팅 자원을 통합하여 활용하며, 컴퓨터와 정보, 가시화 센서 등을 동시에 이용하기기 위한 하나의 Single virtual laboratory 형태의 과학적 협동이 가능하게 하며, 이러한 과학적 협동은 인터넷의 급속한 발전에서 비롯되었다.

계산 그리드(Computing Grid)에 속하는 시스템들은 많은 자원들에 연결하여 작업의 전체 수행 속도를 줄이기 위해서 그 자원들을 최대한 활용하는 것이 목적이다.

데이터 그리드(Data Grid)는 분산된 자료들을 통합하여 분석할 수 이게 해주기 위한 그리드고, 액세스 그리드(Access Grid)는 실시간 멀티미디어 응용을 위한 인프라를 제공한다.

원격 및 화상강의 등에도 활용이 가능한 그리드 이다. 아래의 (그림 1)은 그리드를 수평적 분류 방법으로 분류한 그림이다.



(그림 1) 그리드 분류

2. Globus 및 Diffserv

2.1 Globus

글로버스는 Computational Grid를 구축하기 위한 연구, 소프트웨어 개발, 테스트 베드 구축, 응용분야 연구 등을 수행하고자 할 때 사용되는 소프트웨어 이며, 가장 많이 알려진 그리드 미들웨어 툴킷이다. 또한, 글로버스는 고성능 컴퓨터와 장비 등의 컴퓨팅 자원을 하나의 가상 컴퓨터처럼 사용할 수 있게 하기 위한 소프트웨어이며, 지역적으로 분산되어 있는 다수의 자원들을 활용해 컴퓨팅 작업을 수행 할 수 있게 하는 장점으로 인하여 특정 연구자나 연구소를 중심으로 하여 Grid 구축을 가능하게 하는 장점을 가지고 있고, 기본적으로 통신, 인증, 네트워크 정보, 자료 접근과 같은 기본적인 메커니즘들을 제공하고 있다.

글로버스 툴킷(Globus Toolkit)은 여러 모듈들로 집합되어져 있으며, 각 모듈들은 인터페이스를 정의하게 되고, 이러한 인터페이스들을 통하여, 여러 가지 서비스들이 각 모듈의 기능을 사용할 수 있게 된다.[1][6]

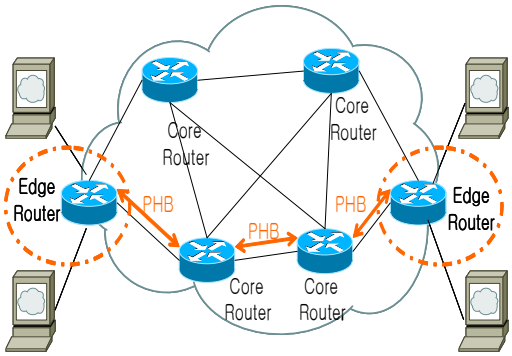
2.2 Diffserv의 기본구조 및 동작

Diffserv는 Traffic에 대해서 각각의 등급을 결정하여 서비스를 제공할 수 있는 기술로써 Int Serv의 단점인 확장성 문제를 해결한 모델이다.

Diffserv는 하나의 IP 패킷 흐름별로 서로 다른 QoS를 제공한다는 개념이 아니라 여러 흐름의 집합을 단위로 하여 각 집합별로 패킷을 전달함으로써 차별화를 통해 패킷을 전달하고, 패킷 분류와 같은 Traffic 조절의 기능들을 모두 망의 가장자리(Edge Router)에서 일어나게 하고 망의 내부는 아주 간단한 PHB(Per Hop Behavior)를 수행 되도록 한다.

다시 말해 경계 라우터(Edge Router)에서는 패킷을 중요도에 따라 3개의 Service Class, EF (Expected Forwarding), AF(Assured Forwarding), DE(Default)로 패킷을 구분하여 DSCP(DiffServ Code Point)값을 설정한 후 코어 라우터(Core Router)로 보내면 코어 라우터는 높은 우선순위의 DSCP값을 가진 패킷부터 PHB(Per Hop Behavior)를 수행하는 방식이다.

DSCP 값은 IP Header(IPv4의 경우)의 TOS (Type Of Service)byte 8bit중 6bit를 사용하여 DSCP값을 설정하고 IPv6의 경우 Class Field에 설정할 수 있으므로 차등 화된 서비스를 제공하는 것이 가능하다.[5]



(그림 2) DiffServ의 구조

Diffserv Network는 내부적으로 다른 PHB 그룹과 다른 코드 포인트와 PHB 매핑을 지원할 수 있으나, 영역 간에 걸친 서비스를 허용하기 위하여, 패킷을 주고받는 영역사이에 계약이 필요하다.

이러한 Diffserv Network 도메인간 경계에서 조절되는 방법을 TCA(Traffic Conditioning Agreement)라고 하며, 이러한 형태의 계약을 SLA(Service Level Agreement)이라 하며, 그림 2에서 보듯이 BB(Bandwidth Broker)는 Diffserv network에서 자원관리를 위하여 이용할 수 있는 자원과 입력 SLA에 의하여 할당된 대역폭에 관한 정보를 저장하고, 향후 할당을 결정하기 위하여 기초로 하는 정책 데이터베이스를 가진 소프트웨어이다.[2][3]

2.3. AF(Assured Forwarding) Class

AF PHB에 속하는 트래픽은 망의 혼잡 상태에서도 트래픽의 최소 전송속도를 보장하는 PHB이다. 이러한 AF PHB는 4개의 클래스로 나누어지고 각 클래스에는 3개의 다른 드롭 확률이 정용되는 세 가지 집합으로 다시 나눠진다. 각 클래스의 구분은 패킷의 손실, 지연, 지터와 같은 다양한 QoS등으로 구분될 수 있고, 각 클래스내의 계약에 정의된 트래픽 속도를 초과하는 지의 여부에 따라 각 집합에 해당하는 드롭 확률을 적용하여 각 집합간의 폐기 우선순위를 결정하게 된다.

예를 들자면 평균 속도와 최대 속도로 정의되는 특정 플로우의 트래픽이 특정 AF클래스로 서비스를 받기로 하였다면 평균 속도 이하로 입력되는 트래픽에 대해서는 가장 낮은 확률을 적용하고, 평균과 최대 속도 사이의 속도로 입력되는 트래픽에 대해서는 가장 높은 드롭 확률을 적용하고, 최대 속도 이상으로 입력되는 트래픽에 대해서는 가장 높은 드롭 확률을 적용하여 서비스를 해 줄 수 있다.

이렇게 하나의 특정 AF 클래스 내에는 t로 다른 드롭 확률이 적용하는 세 집합으로 나뉜다. PHB는 IP헤더 내의 DSCP에 해당 PHB를 나타내는 특정 값으로 엔코딩 되어야 한다.

일반적으로 AF PHB 그룹은 IP 패킷의 전달에 N개의 독립적인 AF클래스와 각 클래스 내부에 M개의 다른 'Drop precedence'를 제공한다. AF 클래스가 각 클래스 내부에 M개의 다른 'Drop precedence'가 j인 IP 패킷은 코드 값 AF_{ij}로 표시된다. 권장되는 방식은 4개의 클래스(N=4)와 각 클래스마다 3단계의 'Drop precedence'(M=3)이지만 더 많은 단계의 클래스와 'Drop precedence'도 가능하다.

Diffserv에서 다른 AF 클래스가 통합(Aggregate)되어서는 안 되고 각 AF 클래스에 가능한 최소한의 자원을 할당하고 추가 할당이 가능한 여분의 자원이 있을 경우에는 정해진 알고리즘에 의해 다시 할당 된다. 각 AF 클래스에서 'Drop precedence'가 p와 q인 패킷 AF_{xp}와 AF_{xq}가 있을 때 p < q 라면, Diffserv에서 패킷의 전달 확률

$P(AF) = P(AF_{xp}) \geq P(AF_{xq})$인 관계를 보장해야 한다. AF PHB의 전달 보장 등급은 세 가지의 의해서 결정된다.[7]

① IP 패킷이 속한 AF 클래스에 활당된 네트워크 자원 양
② AF 클래스의 사용 가능량
③ 패킷의 'Drop precedence' 이다.

다음의 <표 1> 은 AF PHB에 따른 Code Point값이다.

<표 1> AF PHB에 따른 Code Point 값

Drop preference	Class 1	Class 2	Class 3	Class 4
Low	001010	010010	011010	100010
Medium	001100	010100	011100	100100
High	001110	010110	011110	100110

3. GARA (Globus Architecture for Reservation and Allocation)

GARA는 DSRT(Dynamic Soft Real Time Cpu Scheduler)를 사용하여 글로버스 상의 시스템 CPU의 프로세스 스케줄링을 제어 할 수 있고, DPSS(Distributed Parallel Storage System)을 이용하여 원거리에서 DISK의 Input/Ouput에 관여할 수 있다.

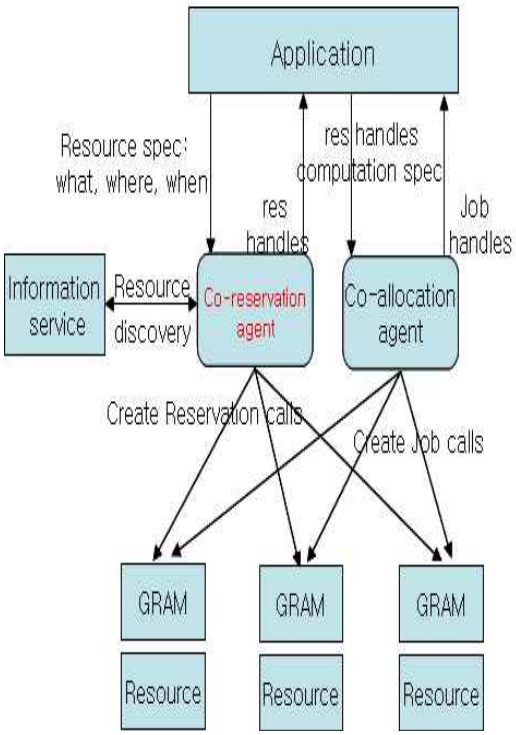
Network에 대한 QoS 보장방법으로는 Diffserv 방식을 사용하고 있으며, Network QoS를 보장받는 모든 패킷은 최우선순위의 PHB(Per Hop Behaviors)인 AF(Assured Forwarding)방식으로 패킷을 처리하게 된다.

즉, GARA는 QoS Reservation 메커니즘을 제공하는 것을 목적으로 한다.[4]

GARA의 특징은 다음과 같다.

① Multiple Reservations이 가능하다.
② Reservations에 대한 모니터링이 가능하다.
③ 누가 Reservations를 제어하고 있는지 알 수 있다.
④ DSRT로 CPU QoS를 제공할 수 있다.
⑤ DPSS로 Disk I/O에 제어할 수 있다.

Application에서 특정 자원을 사용하여 계산 컴퓨팅을 한다고 가정하면 먼저 GARA는 사용자가 원하는 해당 자원에 대한 예약을 수행하기 위하여 Co-reservation Agent가 Information Service를 통해 잉여자원에 대한 정보를 얻게 되고 이 정보를 바탕으로 하여 GRAM(Globus Resource Allocation Manager)을 통해 자원을 예약하게 된다. 물론 예약된 자원은 언제든지 취소와 수정이 가능하다.



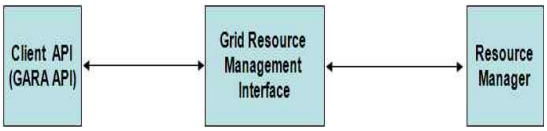
(그림 3) GARA 구조

GRAM은 자원에 대한 할당 및 모니터링과 관리 서비스를 제공한다.

이러한 자원 예약 과정이 끝나면 자원을 이용한 계산 컴퓨팅 과정은 기존의 Globus의 방식과 같다. GARA는 Globus가 설치된 시스템에서만 사용되어 질 수 있고, Globus가 설치되어 있지 않은 시스템에서는 사용할 수 없다.

참고로 위의 (그림 3)에서 Co-reservation agent가 빠지게 되면 글로버스 시스템의 구조가 된다. 따라서 아주 밀접한 관계를 가지고 있다고 말할 수 있다.

위와 같은 구조로 인하여 GARA는 QoS를 보장하게 된다.



(그림 4) GARA 연결구도

Client API는 Reservations을 생성하고, 수정하고 취소하며 모니터링 하는 기능을 가지고 있다.

4. GARA Testbed 구축

GARA의 성능을 테스트하기 위해서 아래의 그림3과 같이 테스트 베드를 구축하였다.

이 테스트 베드의 목적은 GARA의 여러 가지 기능중에 네트워크 자원에 관한 예약을 수행하는데 그 목적을 두었다.

테스트 베드는 2개의 라우터 및 스위치, 그리고 5대의 시스템을 통해서 이루어져 있고, 각각의 시스템에 글로버스와 GARA를 설치하였다.

GARA의 원활한 동작을 위하여 Diffserv_Manager.conf 파일과 setup_flow.cfg 파일을 (그림 5)와 (그림 6)과 같이 수정하였다.

GARA Configuration

1. diffserv_manager.conf

```
#A Configuration file for the Diffserv Manager
LoggingFileName diffserv_manager.log
Verbose True
Quantity 10000
PublicationMethod web
WEBPublishFile
/home/globus/diffserv_manager_slots.html
NoOfRouters 2
IPAddressesServed[1] 192.168.0.11,192.168.7.203
IPAddressesServed[2] 192.168.7.203
```

(그림 5) GARA의 Diffserv.manager.conf

여기서 Diffserv_manager_Slots.html을 설정하는 이유는 GARA를 통한 자원 예약상황을 웹으로 관찰하기 위해서 설정을 하였다.

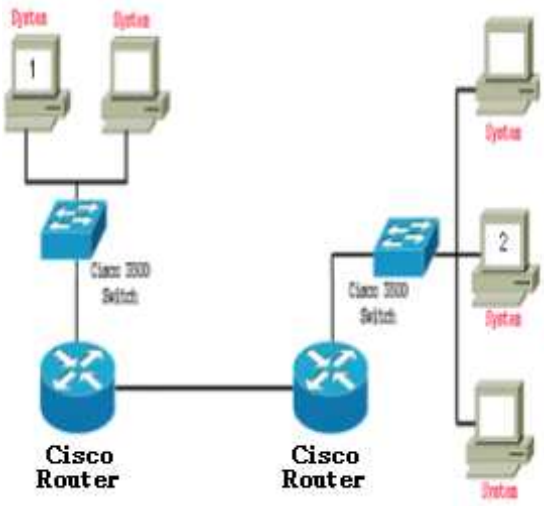
(그림 6) GARA의 Setup_flow.cfg

GARA Configuration

2. setup_flow.cfg

```
##############################
# Written by jungyun Kim
#
1_ROUTER 192.168.0.1
1_USERNAME USER_NAME
1_PASSWORD test
1_INTERFACE1 GigabitEthernet5/0
192.168.0.11,192.168.7.203
# the netx router
2_ROUTER 192.168.0.193
2_USERNAME USER_NAME
2_PASSWORD test
2_INTERFACE1 FastEthernet1/0
192.168.7.203
```

라우터가 2대 이므로 각각의 라우터 이름과 Password를 지정해 주고 라우터 인터페이스의 IP를 설정하였다.



(그림 7) GARA Testbed

위의 (그림 7)에서 보듯 Cisco 7200 라우터와 7500 라우터 까지 GARA를 통해 네트워크에 QoS를 설정하는 것이 목적이고 이와 관련된 GARA 명령어를 실행한 결과 아래의 (그림 8)과 같이 실행되는 결과를 얻을 수 있었다.

(그림 8) GARA의 실행 결과

```
[globus@gara2tests]$./gara_test -c
"localhost:32829:/O=Grid/O=Globus/OU=test.net
/CN=inokyuni" -1 192.168.0.11 -2 192.168.7.203
Reservation request = &(reservation-type=network) (duration=5) (endpoint-
a=192.168.0.11) (endpoint-b=192.168.7.203) (bandwidth=10) (protocol=tcp)

No error on reservation create.

reservation handle =
gara2.test.net:32829:/O=Grid/O=Globus/OU=test.net/CN=inokyuni*network*
diffserv*3
New reservation request for modify = &(reservation-type=network)
(duration=5) (endpoint-a=192.168.0.11) (endpoint-b=192.168.7.203)
(bandwidth=11) (protocol=tcp)

No error on reservation_modify.

after modify, new reservation handle =
gara2.test.net:32829:/O=Grid/O=Globus/OU=test.net/CN=inokyuni*network*
diffserv*3
No error on callback register.

Received callback:
   Status = "Not started"
   Handle =
gara2.test.net:32829:/O=Grid/O=Globus/OU=test.net/CN=inokyuni*network*
diffserv*3

Received callback:
   Status = "Not started, but bound"
   Handle =
gara2.test.net:32829:/O=Grid/O=Globus/OU=test.net/CN=inokyuni*network*
diffserv*3

No error on reservation bind.
.....
Received callback:
   Status = "Active"
   Handle =
gara2.test.net:32829:/O=Grid/O=Globus/OU=test.net/CN=inokyuni
*network*diffserv*3
.....
Received callback:
   Status = "Finished"
   Handle =
gara2.test.net:32829:/O=Grid/O=Globus/OU=test.net/CN=inokyuni
*network*diffserv*3
.....About to cancel reservation.
No error on reservation cancel.
```

위의 (그림 8)을 보면 알 수 있듯이, GARA는 Diffserv를 사용하여 네트워크 간의 QoS를 원하는 시간만큼 설정한 대역폭으로 보장하였고,

보장된 시간이 지나자 자동적으로 QoS 설정을 해지하여 종료되었음을 보여주었다.

Slot Table

Total Available to be reserved : 10000

There are 10 slots in the slot table as of Wed DEC 24 17:09:10 2008

(All times are GMT.)

	Start Time	Duration(Sec)	End Time	Quantity
Slot 1	Wed DEC 24 17:09:10 2008	5	Wed DEC 24 17:09:15 2008	100
Slot 2	Wed DEC 24 17:07:34 2008	10	Wed DEC 24 17:07:44 2008	100
Slot 3	Wed DEC 24 16:05:02 2008	3600	Wed DEC 24 17:05:02 2008	100
Slot 4	Wed DEC 24 15:02:10 2008	3600	Wed DEC 24 16:02:10 2008	100
Slot 5	Wed DEC 24 14:51:16 2008	5	Wed DEC 24 14:51:21 2008	1000
Slot 6	Wed DEC 24 14:47:12 2008	2	Wed DEC 24 14:47:14 2008	1000
Slot 7	Wed DEC 24 14:30:17 2008	2	Wed DEC 24 14:30:19 2008	1000
Slot 8	Wed DEC 24 14:21:17 2008	300	Wed DEC 24 14:26:17 2008	1000
Slot 9	Wed DEC 24 14:10:11 2008	5	Wed DEC 24 14:10:16 2008	1000
Slot 10	Wed DEC 24 13:05:13 2008	3600	Wed DEC 24 14:05:13 2008	1000

(그림 9) GARA의 Slot Table

또한, (그림 9)를 통하여 GARA는 네트워크 간에 QoS를 언제, 얼마동안 설정하고 유지하였는지에 대해서 Slot Table을 통해서 알 수 있었으며, Co-reservation Agent를 통하여 잉여자원에 대한 정보를 얻고, 글로버스 인증을 통하여 다른 시스템의 GRAM(Globus Resource Allocation Manager)에 접속 한 후, 필요에 따라 네트워크 자원들을 미리 예약할 수 있음도 확인 할 수 있었다.

5. 결론

그리드 컴퓨팅은 컴퓨팅 자원을 비용편익 측면에서 가장 효율적으로 만들 수 있는 능력을 가지고 있으며, 클러스터링 등의 기술로는 해결하기 어렵거나 시간이 오래 지체되어지는 대량의 컴퓨팅 능력을 요구하는 어려운 문제들을 풀기에 더 없이 좋은 방법 중에 하나이다.

이러한 그리드 컴퓨팅을 효율적으로 수행하기

위해서 지리적으로 분산되어 있는 고성능 컴퓨팅 자원을 실시간으로 상호 연결하기 위해 Grid Application이 필요하게 되었고 Grid는 수많은 가능성을 가지고 계속적으로 개발되어 지고 있고, 그 응용범위는 무궁무진하다.

GARA는 Globus 기반의 Grid Network에서 QoS 보장뿐만이 아니라, End-to-End Performance를 보장 할 수 있는 방법이다.

본 논문은 이러한 상황을 가정하여 테스트 베드를 구축하고 GARA를 통해 네트워크 QoS를 설정해 봄으로써, GARA와 관련된 Application 연구, Globus Application, DSRT(Dynamic Soft Real Time CPU Scheduler)응용 연구, DPSS(Disk Parallel Storage System)응용 연구 등의 기반을 마련하였다.

참 고 문 헌

[1] A Distributed Resource Management Architecture that Supports Advance Reservations and Co-Allocation. I. Foster, C. Kesselman, C. Lee, R. Lindell, K. Nahrstedt, A. Roy. (Int'l Workshop on Quality of Service, 1999).

[2] QoS as Middleware: Bandwidth Reservation System. G. Hoo, W. Johnston, I. Foster, A. Roy. Proceedings of the 8th IEEE Symposium on High Performance Distributed Computing. pg. 345-345, 1999.

[3] MPICH-GQ:Quality of Service for Message Passing Programs. A. Roy, I. Foster, W. Gropp, N. Karonis, V. Sander, B. Toonen. (Accepted to Supercomputing 2000).

[4] A Quality of Service Architecture that CombinesResource Reservation and Application Adaptation. I. Foster, A. Roy, V. Sander, (Published in the 8th International Workshop on Quality of Service (IWQOS 2000), pp. 181-188, June 2000.).

[5] A Differentiated Services Implementation for High-Performance TCP Flows. V. Sander, I. Foster, A. Roy, L. Winkler. (Accepted to The TERENA Networking Conference 2000).

[6] The Anatomy of the Grid: Enabling Scalable Virtual Organizations. I. Foster, C. Kesselman, S. Tuecke. International J. Supercomputer 15, 2001

[7] Linux Kernel Scheduling, May, 2002[1]

[8] Providing packet-loss guarantees in DiffServ Architectures S. Hassanein May, 2002

김 정 윤

2003년 경희대학교 정보통신대학 졸업(공학석사)
2007년 경희대학교대학원 컴퓨터 공학과 수료(박사 수료)
2003년~2004년 : 한국과학기술정보연구원 초고속 연구망 개발실 근무
2005년~현 재 : 경희대학교 정보통신연구실 연구원
관심분야 : 네트워크 보안 및 QoS, 차세대 통신

나 원 식

2005년 경희대학교대학원 컴퓨터 공학과 졸업(공학박사)
2006년~현재 남서울대학교 교양과 정부 교수(컴퓨터)
2001년~2003년 : (주)성신섬유 전산실장
관심분야 : 네트워크 보안, 무선랜, 의료정보, 전자제어

유 인 태

1987년 : 연세대학교 전자공학과 졸업(공학사)
1989년 : 연세대학교 대학원 전자공 과 졸업(공학석사)
1994년 : 연세대학교 대학원 전자공 과 졸업(공학박사)
1997년 : The University of Tokyo 졸업(공학박사)
1999년~현재 : 경희대학교 전자정보학부교수
관심분야 : 인터넷, 네트워크 보안, 무선 LAN