韓國 컴퓨터情報學會 論文誌
第10卷 第6號, 2005. 12.

2005-10-6-1-20

Etherboot 기반의 CGRID 구축과 서열분석에의 적용

김 태 경*, 조 완 섭**

CGRID construction based on Etherboot technology and its utilization to sequence analysis

Tae-Kyung Kim*, Wan-Sup Cho**

요 약

최근 생물학 분야에서 실험 도구의 발달 및 컴퓨터 기술의 도입으로 생물 데이터가 폭발적으로 증가하고 있다. 대량의 생물 데이터로부터 의미 있는 정보를 추출하는 것은 매우 중요한 문제이다. 서열비교는 유전자 및 단백질 기능 예측을 하기 위해 사용되는 가장 기본적인 분석방법이다. 하지만, 급격히 증가하는 대량의 서열데이터에 대하여 처리시간 또한 많이 소요된다. 본 논문에서는 이러한 성능상의 한계를 극복하고 기존 미들웨어 방식의 그리드를 보완하기 위하여 하드웨어 기반의 그리드인 CGRID (Chungbuk national university GRID)를 제안하고 서열비교에 적용한다. 하드웨어 기반의 그리드 방식은 기존의 방식과는 달리 모든 작업노드에 반복적으로 프로그램 설치를 할 필요가 없으므로 그리드 구축, 유지 및 관리가 용이하다. 27대의 PC로 구성된 CGRID에서 89종의 오솔로그 데이터베이스 구축 시간을 33주에서 1주일로 단축하였다. 또한, 실험을 통하여 CGRID에서 PC의 수가 증가함에 따라 시스템의 성능이 비례하여 향상됨을 확인하였다.

Abstract

Recently, amount of the data such as sequences is being increased rapidly due to deploying computational technique and advance of experiment tools in the biological areas. In bioinformatics, it is very significant to extract the knowledge from such huge biological data. Sequence comparisons are most frequently used to predict the function of the genes or proteins. However, it takes so much time to process the persistently increasing data. In this paper, we propose hardware-based grid, CGRID(Chungbuk National University GRID), to improve performance and complement existing middleware-only approach and apply it in the sequence comparison. Hardware-based approach is easy to construct, maintain, and manage the grid as not requiring the software installation individually for every node. We reduce orthologous database construction time from 33 weeks to just a week. Furthermore, CGRID guarantees that the performance increases proportionally as adding the nodes.

▸ Keyword : 그리드 컴퓨팅(Grid Computing), 바이오 인포메틱스(Bioinformatics), 서열분석(Sequence Analysis)

• 제1저자 : 김태경
• 접수일 : 2005.11.12, 심사완료일 : 2005.12.19
* 충북대학교 정보산업공학, ** 충북대학교 경영정보학과
※ 본 연구는 산업자원부의 지역혁신 인력양성사업의 연구결과로 수행되었음.

Ⅰ. 서 론

최근 생물학 분야에서 실험 도구의 발달 및 컴퓨터 기술의 도입으로 데이터의 양이 폭발적으로 증가하고 있다. 이 데이터는 현재 NCBI[40], DDBJ[29], KEGG[35] 등의 생물정보 데이터베이스에 지속적으로 저장되고 있다. 그 중 대표적인 생물정보 데이터베이스인 NCBI 서열 데이터베이스를 살펴보면 염기서열의 수가 2004년 4월 38,989,342,565개를 넘어 그 양이 15개월 간격으로 2배가량 증가하고 있다.

생물정보학 분야에서 유전자의 서열비교를 통해 기능을 예측하는 연구[4, 5, 6, 13], 단백질 서열을 이용하여 2차 구조를 예측하는 연구[21], 그리고 분자 모델링 디자인 연구 등은 매우 중요한 문제이다. 하지만 이러한 생물 데이터 처리는 시간이 많이 필요하고, 급격하게 증가하고 있는 데이터의 양으로 인하여 단일 컴퓨터의 능력으로는 해결하기에는 어려운 문제가 되었다. 또한 하나의 문제를 해결하기 위하여 다양한 경우의 수를 고려해야 하므로 처리시간이 매우 길게 소요된다[12]. BLAST 연산의 경우 그 처리 시간이 데이터의 양과 비례하는데 급격한 데이터양의 증가로 인하여 처리 성능의 한계에 부딪히고 있다[1].

현재 생물정보학 분야에서 복잡하고 어려운 문제를 빠르게 해결하기 위하여 SMP (Symmetric Multi-Processors) 컴퓨터[22, 27]를 이용하는 방법, 클러스터 컴퓨팅 방법을 이용하는 연구[7, 17, 22]가 진행되고 있다. 하지만 이러한 방법은 지역적 한계를 극복할 수 없고 지속적인 확장이 불가능하며 시스템 구축비용 측면에서도 부담이 크다. 또한 시스템을 유지하고 관리하는 번거로움이 있으며 지속적인 확장이 어렵다.

현재 인터넷에 존재하는 많은 컴퓨터 자원들을 활용하는 그리드 컴퓨팅 (Grid computing) 기법[8, 9, 11, 15, 18, 32, 33, 36, 38, 39]으로 처리 성능을 높이려는 노력을 하고 있다. 그리드 컴퓨팅은 인터넷의 발전과 네트워크의 확산으로 인하여 생겨난 개념으로 인터넷에 있는 많은 컴퓨터 자원을 마치 단일 컴퓨팅 자원처럼 활용하자는 것이다. 따라서 SMP 컴퓨팅과 클러스터링 컴퓨팅의 지역적인 한계점을 극복할 수 있을 뿐만 아니라, 확장 측면에서도 유리하다. 향후 네트워크의 확산과 더불어 네트워크 속도의 향상은 더욱 그리드 컴퓨팅 기술의 발전을 진일보 시킬 것으로 기대된다.

기존의 그리드는 각 지역별로 미들웨어 서버를 구축해야 하고 작업 노드에도 Globus 툴킷을 모두 설치해야 하는 번거로움이 있다. 또한 일을 처리할 경우 각 노드들의 자원을 100% 활용하는 것이 아니라 일부분의 자원을 활용하므로 시스템 성능을 극대화하는 것은 불가능하다. 이러한 접근 방법에서는 각 노드에 해당하는 컴퓨터 시스템을 다시 설치하는 경우 그리드 소프트웨어가 삭제됨으로 인하여 노드의 기능을 할 수 없게 된다. 인터넷에 존재하는 많은 시스템 자원을 활용할 수 있는 장점은 있으나 소프트웨어 적인 방법을 활용한 기법으로 인한 그리드 유지의 어려움이 있다.

본 논문에서는 대량의 바이오 데이터 처리를 위한 그리드 환경인 CGRID (ChungBuk national university computing GRID)를 제안한다. CGRID는 학교의 전산실 PC들을 이용하여 그리드 컴퓨팅 환경을 구축하고, 이 기반에서 고성능 바이오 데이터 처리 서비스를 제공한다. CGRID는 최소의 비용으로 저녁시간과 주말에 사용하지 않는 학교 전산실 PC 자원들을 활용함으로써 바이오 데이터를 분석하기 위한 인프라를 구성하는데 그 목적이 있다.

CGRID는 GNU EtherBoot 프로젝트[30]를 기반으로 만들어진 LanLinux 시스템[37]을 활용하여 구축한다. 작업노드는 마스터 노드에서 시스템 자원을 읽어 와서 시스템을 부팅하면서 그리드를 구성한다. 이후 PC는 디스크를 사용하지 않고 운영되므로 관리 및 유지가 용이하고 비용이 적게 든다. 또한 평소 PC의 활용에 전혀 영향을 끼치지 않는다. 단지 부트롬을 장착함으로써 간편하게 지속적으로 PC들을 추가하여 확장이 가능하다.

기존의 그리드는 소프트웨어적으로 그리드를 형성하였으며 소극적으로 시스템 자원을 사용하는 반면 본 논문의 접근법은 리눅스 운영체제[3]를 기반 하에 하드웨어적으로 그리드를 구성하며 적극적으로 시스템 자원을 활용하므로 성능이 크게 증가하는 장점이 있다.

본 논문의 구성은 다음과 같다. 제2장에서는 기존의 관련 연구에 대하여 언급하고, 제3장에서는 CGRID의 구성 및 환경에 대하여 설명한다. 그리고 제4장에서는 CGRID를 실제로 적용한 사례 및 성능평가 결과를 살펴보고 제5장에서 결론을 맺는다.

II. 관련 연구

본 장에서는 현재 바이오 데이터 분석에 대한 관련 연구 및 대량의 데이터 처리 및 연산을 위한 컴퓨팅 기법들에 대해 알아본다.

2.1 바이오 데이터 분석

본 절에서는 바이오 데이터 분석에 있어서 가장 많이 사용되고 있는 서열비교 분석과 단백질 2차 구조 예측에 대하여 살펴본다.

2.1.1 서열 비교 분석

단백질과 DNA의 서열 비교는 생물정보학의 가장 많이 사용되는 작업이다. 신속하고 자동화된 서열 비교 능력은 새로운 서열에 대한 기능을 알아내는 작업에서부터 모델 단백질의 구조 예측 및 구성 그리고 유전자 발현 실험의 설계 및 분석에 이르기까지 모든 작업에 필요하다.

서열 정렬(sequence alignment)은 서열 유사도를 찾기 위한 과정으로 서열에서 동일한 순서에 있는 개별적 특징이나 패턴을 검색함으로써 두 개 이상의 서열들과 비교하는 것이다[4, 5, 6].

서열 비교는 기능이 밝혀지지 않은 서열에 대한 기능 정보를 알아내기 위한 필수 연산이며 이를 위하여 다양한 서열비교 프로그램들이 개발되었다. 그 중 가장 대표적인 프로그램은 BLAST이다[4]. 서열 분석 도구들로 인하여 서열 분석이 용이하고 다양한 분석이 가능하지만 대량의 서열 정보를 분석하는 데에는 한계에 이르고 있다. 이러한 한계를 극복하기 위하여 병렬처리 컴퓨팅, 클러스터 컴퓨팅을 적용하고 있다.

2.1.2 단백질 구조와 기능예측

단백질의 아미노산 서열에는 유용한 정보가 담겨져 있으며 그 자체로도 정보로서의 의미가 있다. 단백질 서열 분석은 다른 단백질과의 상관성을 위해서 필요한 작업이다. 이미 알려진 단백질과의 서열 비교를 통하여 기능에 대한 정보를 얻을 수 있으며 생화학적 기능의 진화에 대한 정보도 얻을 수도 있다. 하지만 단백질의 기능을 이해하려면 서열 분석보다는 단백질의 2차 및 3차원 구조를 분석하는 것이 훨씬 중요하다[34]. 단백질이 여러 가지 다양한 생화학적 기능을 수행하게 하는 데 가장 중요한 특성들은 단백질 체인을 만드는 아미노산 서열에 의해 결정되며 이 서열은 단백질의 3차원 구조를 결정하는 데도 큰 역할을 한다. 20개의 아미노산은 수많은 조합이 가능하므로 아주 작은 단백질 서열을 만들 수 있는 조합의 수도 매우 많다. 이는 주어진 시간 동안 생물체가 다양한 목적에 맞는 기능을 수행하는 단백질을 진화시킬 수 있다는 것을 의미한다[12].

2차 구조 예측은 단백질 구조를 예측하는 첫 단계이고 단백질 서열의 아미노산들을 예측되는 로컬 구조에 따라 분류하는 작업이다. 2차 구조는 크게 3 가지로 나뉘는데 알파 나선구조, 베타 평면구조, 코일 구조가 그것이다. 하지만 실제 사용되는 모델에 따라 그 숫자가 결정된다.

2.2 대용량 데이터 처리 방법론

본 절에서는 대용량 데이터 처리를 위해 사용되는 컴퓨팅 기법인 클러스터 기법과 그리드 방식에 대하여 살펴본다.

2.2.1 클러스터 컴퓨팅

'클러스터(Cluster) 시스템'이란 PC 또는 워크스테이션을 고속 네트워크로 연결하여 고성능과 고가용성을 얻을 수 있도록 하는 기술이나 시스템을 말하며 NOW(Network Of Workstation) 또는 COW(Cluster Of Workstations)라고도 한다[2].

클러스터 시스템의 장점은 저가의 상용제품을 사용함으로써 기존 고성능 서버보다 수배에서 수십 배 작은 비용으로 동일한 성능의 시스템 구현이 가능하여 가격 대 성능비가 높다는 것이다. 그리고 사용자가 직접 상용부품을 사용하여 즉각적인 업그레이드가 가능함으로 시스템 유지비용이 감소하고, 사용이 편리한 PC 및 워크스테이션(Workstation)의 개발 환경을 그대로 사용할 수 있기 때문에 프로그램 개발이 용이하다.

(그림 1)은 클러스터 컴퓨팅의 구조를 보여준다. 제한적인 지역 내에서, 하나의 마스터 노드와 여러 개의 작업 노드들이 연결되어 있다. PC와 SMP (Symmetric Multi-Processors) 컴퓨터를 작업 노드로 적용하여 고성능(high-throughput) 및 대용량(large scale)데이터 처리한다. 네트워크는 노드들 간의 통신을 위한 통로가 되고 이 속도는 처리 성능에 커다란 영향을 끼친다. (그림 1)에서 볼 수 있듯이 클러스터 컴퓨팅은 병렬처리와 분산시스템을 합쳐 놓은 하나의 새로운 컴퓨팅이라 할 수 있다.

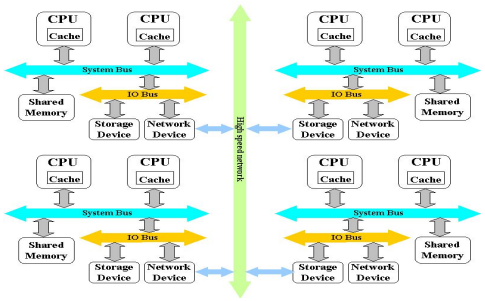

그림 1. 단백질 서열 비교를 위해 4개의 SMP를 연결한 클러스터
Fig 1. Cluster constructed with 4 SMPs to perform protein sequence comparison

클러스터 컴퓨팅의 경우 (그림 2)와 같이 일이 들어왔을 때 일을 작은 단위로 나누고 난 다음 각 작업 노드에 할당하여 나누어진 일이 동시에 여러 개의 프로세스에서 처리된다. 이때 작업할당 스케줄 방식과 노드간의 네트워크 속도는 처리 성능에 많은 영향을 끼친다.

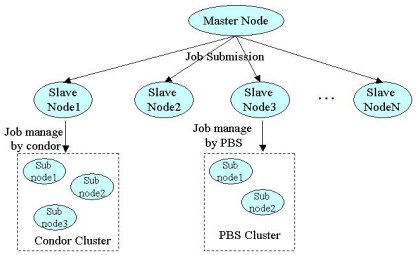

그림 2. 클러스터 컴퓨팅의 MASTER-SLAVE 작업구조
Fig 2. Master-Slave mechanism in cluster computing

현재 클러스터 컴퓨팅 기술은 기후 및 강수량 예측, 유전 탐사 분석, 자동차 및 비행기 시뮬레이션, 그리고 생물의학에서의 분자 모델링 등에 사용되고 있다. 현재 관련 연구[23]는 리눅스 클러스터를 형성하여 단백질 서열 분석에 적용 하였으며 뛰어난 성능을 보였다.

2.2.2 그리드 컴퓨팅

90년대 중반에 등장한 그리드는 기존의 병렬 및 분산 컴퓨팅이 가지고 있는 성능과 확장성을 극복하기 위하여 인터넷에서 상호-연동된 컴퓨팅 자원들을 안전하게 사용할 수 있는 환경이다[8]. 기존의 병렬 및 분산 시스템 분야는 작은 컴퓨팅 자원을 여러 개 연결하여 보다 큰 컴퓨팅 성능을 제공하기 위한 방법으로 발전해 왔지만, 범용성이나 단일 시스템 이미지 등을 제공하기에는 많은 한계가 있었다. 그리드 컴퓨팅은 병렬/분산 컴퓨팅의 연장선상에서 이러한 한계를 극복하기 위하여 태동하고 발전해 온 기술이다.

Globus 툴킷[10]은 그리드 환경을 구축하는데 사용되는 대표적인 그리드 구축 도구로, 그리드 컴퓨팅에서 사실상의 표준으로 자리 잡고 있다. Globus 툴킷은 슈퍼컴퓨터들을 연결하여 활용하기 위한 목적으로 미국 아르곤 연구소, 남가주 대학교의 정보과학센터(ISI), 시카고 대학 등이 참여하여 1996년부터 개발이 진행되었다.

Globus 툴킷은 그리드와 그리드 응용프로그램을 지원하는 서비스 및 소프트웨어 라이브러리로서 보안, 정보 발견, 자원 관리, 데이터 관리, 통신오류 감지, 이식성 등 그리드에서 필요한 서비스들을 독립적인 요소로 제공한다. Globus 툴킷에서 제공하는 주요 서비스로는 TLS 기반의 인증인 GSI(Grid Security Infrastructure), 원격에서 작업을 제출할 수 있게 해주는 GRAM, 대용량의 데이터를 빠르고 안정적으로 제공해주는 GridFTP, 시스템의 정보를 검색할 수 있는 MDS 등이 있다. 1998년에 처음 공개된 Globus 툴킷 1.0 버전에서는 분산 컴퓨팅 기능을 손쉽게 구현할 수 있는 기본적인 기능들을 제공하였으며, 2002년에 공개된 Globus 툴킷 2.0 버전(GT2)은 대규모 데이터를 처리할 수 있도록 성능과 안정성을 크게 개선하였다.

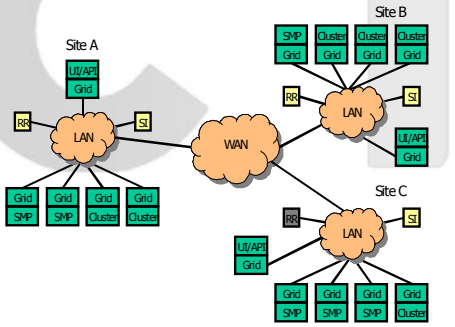

그림 3. 그리드 컴퓨팅을 위한 환경
Fig 3. Grid Environment

(그림 3)은 그리드 컴퓨팅의 환경을 보여준다. 클러스터 컴퓨팅과는 달리 지역적 한계를 뛰어넘어 인터넷으로 연결되는 모든 곳의 자원들을 활용할 수 있다. 기존의 자원들을 사용함으로 인한 비용 절감과 자원의 지속적인 확장이 가능하여 시스템 성능의 획기적인 향상을 기대할 수 있다.

하지만 기존의 그리드 방식에서는 각 지역별로 서버뿐만 아니라 일을 처리할 시스템(Worker Node)에도 Globus

툴킷을 설치해야 하므로 구축 및 유지비용이 많이 든다. 또한 일을 처리할 경우 각 노드들의 자원을 100% 활용하는 것이 아니라 일부분의 자원을 활용하므로 시스템 성능을 급격히 높이기는 불가능하다. 현재 각 노드가 컴퓨터 시스템을 다시 설치하는 경우 그리드 소프트웨어가 삭제됨으로 인하여 작업을 부여받지 못하는 한계점이 있다. 인터넷에 존재하는 많은 시스템 자원을 활용하는 장점은 있으나 소프트웨어적인 방법을 활용한 기법으로 인한 그리드 유지 및 관리에 어려움이 있다.

본 논문에서는 Globus 툴킷을 이용하여 소프트웨어적으로 그리드를 구성하는 것과는 달리 작업 노드에 부트롬을 장착하여 네트워크로 시스템을 부팅한 다음 각 노드에 일을 부여하는 하드웨어적인 방법론을 사용하였다. 이 방식은 부트롬을 시스템에 장착한 다음 별도의 관리가 필요 없으며 마스터 노드에서 자유롭게 제어하고 통제할 수 있으므로 가용성 및 확장성 측면에서 우수하다. 또한 평상시에는 컴퓨터 실습실 용도로도 아무런 불편 없이 사용할 수 있다.

III. CGRID 환경 및 구성 요소

제 3 장에서는 CGRID의 구성 및 환경에 대하여 살펴보고 작동 알고리즘을 상세히 살펴본다.

3.1 CGRID 시스템 구성도

CGRID는 충북대학교 컴퓨터 실습실에 있는 컴퓨터들을 이용하여 구성한다. 두 개의 전산 실습실의 PC들을 이용하여 그리드를 구성하였으며, 향후 지속적인 확장이 가능하다.

그림 4. 부트롬을 장착한 랜카드
Fig 4. Ethernet Card for BootRom to be fixed

작업 노드 PC의 하드웨어적 설정은 (그림 4)와 같은 Lanlinux[37]사에서 특수 제작된 부트롬을 랜카드에 장착하고, 메인보드에 설치하는 것으로 모든 것이 준비가 완료된다. 다음으로 마스터 노드에 부트롬을 장착한 랜카드의 MAC 어드레스를 등록하면 된다. 설정이 끝나면 노드들은 CGRID 상의 작업 노드로서 역할을 하게 된다. 작업 노드들을 부팅할 때 로컬디스크를 이용하여 부팅할 것인지 네트워크로 부팅할 것인지에 대해 선택할 수 있게 되며 네트워크로 부팅할 경우 CGRID의 작업 노드가 된다.

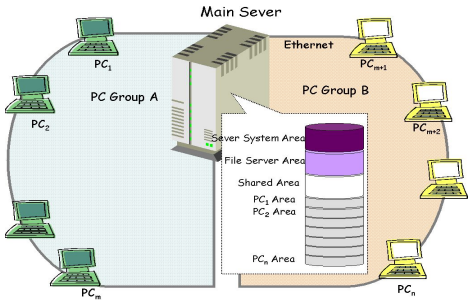

그림 5. CGRID 시스템 구성
Fig 5. CGRID System Organization

(그림 5)은 CGRID의 전체적인 구성을 보여준다. 각 노드에 부트롬을 장착한 PC들은 자신의 로컬 디스크를 이용하지 않고 마스터 노드로부터 자신의 고유 영역을 읽어서 리눅스 또는 FreeBSD 운영체제를 부팅하게 된다. 시스템이 부팅된 이후부터는 시스템 유지에 필요할 때만 마스터 노드와 통신하게 되며 마스터 노드에는 거의 부담을 주지 않는다. 각 작업 노드는 독자적인 서버로서의 기능도 할 수 있고, 마스터 노드로부터 일을 부여받아 처리한다. 이와 같은 방식은 각 노드에 대하여 별도의 시스템 환경설정과 소프트웨어를 설치할 필요가 없으므로 시스템 관리 및 유지비용이 줄어들게 된다.

본 논문에서 구축한 CGRID에서 사용하는 시스템 자원은 <표 1>과 같다.

표 1. CGRID 시스템 구성
Table 1. CGRID System Organization

	Master Node	PC Group A	PC Group B
CPU	PIII800Mhz	PIII800Mhz	PIII800Mhz
Main Memory	256MB	256MB	256MB
HDD	120GB	Nothing	
노드 수	1	15	12
Network Bandwidth	10Mbps		

현재 두 개의 PC 그룹을 이용하고 있으며, 각 작업 노드들은 Group A에 15대 Group B에 12대를 활용하고 있다. 그리고 각 그룹의 노드들을 관리하는 마스터 노드는 1대를 이용하고 있다. 네트워크 대역폭은 10Mbps이며 PC Group A의 네트워크 속도가 PC Group B의 네트워크 속도보다 2배정도 빠르다.

3.2 CGRID 구성 요소 및 작동 알고리즘

3.2.1 CGRID 구성 요소

3.2.1.1 프로토콜 정의

마스터 노드와 작업 노드간의 신뢰성 있고, 안정적인 일을 처리하기 위해서는 상호간의 명령과 필요한 정보를 표현할 수 있는 프로토콜을 설계하는 것이 필수이다. (그림 6)은 CGRID에서 마스터 노드와 작업 노드와의 통신에 사용되는 프로토콜 명세이다.

[정의 1] CGRID 프로토콜(P)

$$P = < N, j, c, s, d, r >$$

N : node 정보

$N = \{ n_{cpu}, n_m, n_{os}, n_{os_v}, n_{role} \}$

n_{cpu} : node의 CPU 속도

n_m : node의 메모리크기

n_{os} : node의 운영체제

n_{os_v} : node의 커널 버전

n_{role} : node의 역할(Master/Slave)

j : 작업 식별자

c : 명령어

s : 네트워크 속도

d : 전송방향

r : 부가정보

그림 6. CGRID 프로토콜 표기
Fig 6. Protocol Specification used in CGRID

마스터 노드와 작업 노드간의 일을 주기 위한 명령의 전달 매개로써 변수 c를 사용한다. 즉 작업 노드가 해야 할 일 또는 마스터 노드가 해야 할일에 대한 명확한 정보를 전달한다. 작업 노드들의 시스템 사양 및 마스터 노드와의 통신 속도를 알려 주기 위한 노드정보(N), 네트워크 속도(s) 변수를 사용한다. 또한 확장을 위하여 노드의 정보에 n_{role} 변수를 두어 노드가 작업 노드인지 마스터 노드인지를 구분한다. n_{slave} 변수를 통해 관리 노드가 관리하는 작업 노드의 수를 알 수 있다. 운영체제에 대한 정보를 알기 위하여 n_{os}와 커널정보를 나타내는 n_{os_v}변수를 두었다. 현재 프로토콜의 방향성을 표시하기 위하여 d 변수를 이용한다. 마지막으로 처리해야 할 작업을 주고받기 위하여 j변수를 이용하고 변수 r을 통해 지금 주고받는 일의 속성을 표현한다.

3.2.1.2 일을 처리하기 위한 테이블

대량의 일이 들어왔을 때 이 일들을 체계적으로 관리하는 것이 필요하다. 이 테이블의 정보를 바탕으로 마스터 노드에 있는 스케줄러가 일을 부여한다. 기본적으로 <표 2>와 같이 그리드에서 현재 일을 할 수 있는 노드를 확인할 수 있는 테이블이 필요하다.

표 2. 작업 노드 테이블 정보 : 작업 노드의 정보를 기록하는 테이블
Table 2. Work node table schema

Working_Node Table

속성	설명
node_id	고유 번호
ip	노드의 IP
node_state	노드의 현재 상태 0 : 노드가 off 되어 있는 상태 1 : 일을 부여받지 않은 상태 2 : 일을 처리하고 있는 상태
start_time	일처리 시작 시간
finish_time	일처리 끝 시간

node_id는 노드의 고유 번호이고 ip는 작업 노드의 ip 주소를 나타낸다. 또한 node_state는 현재 노드의 작업 상태를 나타낸다. 그리고 현재 진행 중인 일의 시작 시간과 끝 시간을 start_time과 finish_time 필드에 저장한다.

<표 3>의 Master 테이블의 경우 해야 할 일들의 목록을 보관하기 위한 것이다.

표 3. 마스터 테이블 정보 : 일의 목록을 저장하는 테이블
Table 3. Master table schema

Master Table

속 성	설 명
job_id	일의 고유 번호
arg1	첫 번째 인자
arg2	두 번째 인자
job_state	일의 처리 상태 0 : 아직 처리 되지 않음 1 : 처리 중 2 : 처리 완료
st_time	일을 처리하기 시작한 시간
end_time	일을 종료한 시간
ip	일을 처리한 노드의 IP 주소

job_id는 처리해야 할 일의 고유 번호이고 arg1과 arg2는 처리에 필요한 인자이며 job_state는 현재 이 일의 처리 상태를 나타낸다. 또한 st_time과 end_time은 일을 처리하기 시작한 시간과 종료 시간을 표현하며 ip는 일을 처리한 노드의 ip 주소를 저장한다. Working_Node 테이블과 Master 테이블을 이용하여 스케줄러는 일을 체계적으로 노드에 부여하며 노드의 상태와 일의 처리 상태를 관리하게 된다.

3.2.2 CGRID 처리 과정

CGRID 상에서 마스터 노드와 작업 노드 간의 일을 주고받기 위한 전체적인 과정은 (그림 7)과 같다. 마스터 노드의 Master 테이블에 있는 작업 정보와 Working_node 테이블의 노드 정보를 바탕으로 CGRID 상의 노드들에게 작업을 부여하고 그 결과를 다시 마스터 노드의 디스크로 저장한다.

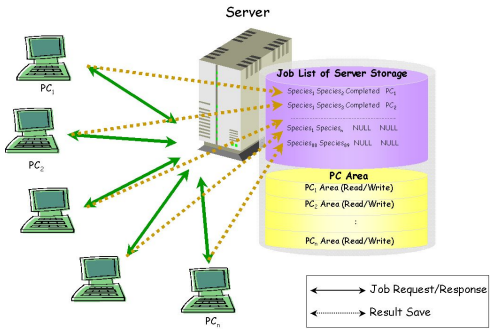

그림 7. CGRID에서의 작업 처리 원리
Fig 7. The principle to process jobs on CGRID

각 노드는 마스터 노드의 디스크의 일부를 사용하므로 데이터 처리 결과를 별도로 모으거나 처리할 필요가 없다. (그림 8)은 이러한 과정에서 보다 신뢰성을 높이기 위하여 작업 노드와 마스터 노드간의 작업을 처리하기 위한 규칙을 제시하고 있다.

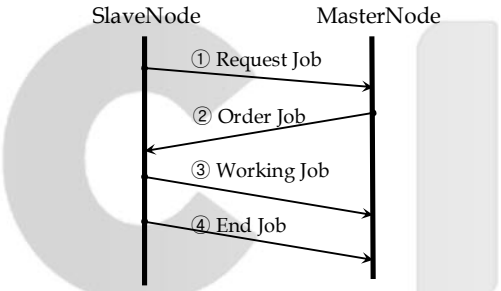

그림 8. 마스터 노드와 작업 노드 간의 통신 방법
Fig 8. Communication between a master node and slave nodes

3.2.1 절에서 정의한 프로토콜의 cmd 멤버 변수를 이용하여 상호간에 처리하고자 하는 일의 속성을 서로 확인한다. 우선 작업 노드들이 시스템을 부팅하거나 일을 마쳤을 경우 마스터 노드에 자신이 일을 할 수 있음을 알린다(100). 이러한 보고를 받은 마스터 노드는 Master 테이블에 있는 일들 중 하나를 선택하여 작업 노드에게 부여한다. 이때 Master 테이블의 job_state 필드와 ip 필드에 처리 상태를 변경(Idle → Working)하고 처리하고 있는 노드의 ip 주소와 처리시작 시간을 기록한다.

일을 부여받은 작업 노드는 자신이 일을 하고 있다고 다시 마스터 노드에게 신호를 보내줌으로써(200) 정상적인 작업이 진행되고 있음을 알린다. 마지막으로 작업 노드는 자

신의 일을 마친 후 마스터 노드에게 일을 종료했다는 신호를 보내게 되며 이때 Master 테이블의 end_time 과 Working_node 테이블의 node_state를 변경한다. 전체작업을 마칠 때까지 이러한 과정을 반복하여 일을 처리한다.

Ⅳ. CGRID 적용사례 및 성능분석

본 장에서는 CGRID를 이용하여 오솔로그 데이터베이스 구축하기 위한 유전자 서열 분석을 위하여 제안된 HPBS(High Performance BLAST System)을 설명하고 이를 바탕으로 한 성능평과 결과를 제시한다.

4.1 오솔로그 데이터베이스 구축을 위한 유전자 서열 비교 분석

4.1.1 개요

분자 생물학에서 필수적 단계는 유전체 분석이라고 할 수 있다. 유전체를 분석하기 위해서는 먼저 유전체 시퀀싱(Sequencing) 작업이 선행 되어야 하며 시퀀싱된 유전자를 통해 기능을 예측하는 주해(Annotation) 과정을 거치게 된다. 기술의 발달로 인해 과거에 비해 빠르게 많은 양의 시퀀싱된 데이터를 얻을 수 있게 되었고 정보기술을 활용하여 기능을 예측하고 분석하는 많은 도구와 방법이 현재 널리 쓰이고 있다. 유전자의 기능을 예측하는데 활용되는 가장 보편적인 방법은 BLAST와 같은 프로그램을 이용해 기존의 서열 데이터베이스와 서열의 상동성 비교를 통한 기능 예측 방법이 주로 사용된다. 그러나 사용되는 데이터베이스의 신뢰성이 떨어 질 뿐만 아니라 그 데이터베이스를 이용하여 생물학자들이 다양한 분석을 하기엔 까다로운 것이 사실이다.

이러한 단점을 보완하기 위해 진화적으로 보전적인 유전자들의 서열의 상동성에 따른 분류를 통한 오솔로그 데이터베이스[17]를 구축하고, 구축된 데이터베이스를 바탕으로 유전자의 다차원적인 분석을 위하여 89종의 원핵생물(Prokaryote)들에 대하여 유전자 간의 상동성 비교를 한다. 이 작업은 처리 시간이 많이 걸리므로 성능을 높이기 위하여 CGRID상에서 HPBS(High Performance Blast System)을 구현하였다.

여기서 오솔로그는 공통의 조상으로부터 종분화(Speciation) 되어 서로 다른 유전체에 있는 직접적으로 관련된 유전자들의 집합이라고 정의한다[24, 26]. 일반적으로 같은 오솔로그 관계에 있는 유전자들은 서열의 유사성과 함께 같은 기능을 갖게 된다. 따라서 오솔로그 관계를 분류한 2차 데이터베이스의 구축은 계통 발생적(Phylogenetic) 분석에 있어서, 서로 다른 종에서 나타나는 공통의 필수 유전자 파악 및 정확한 기능 예측과 다양한 분석을 위한 기본모델이라고 말할 수 있다.

HPBS에서는 89 종에 대하여 오솔로그 데이터베이스를 구축하기 위해서는 89C2 만큼의 유전체 간의 비교가 필요하며 유전체 간의 비교는 평균 8,400,000번 이상의 유전자 상동성 비교가 필요하다. 이는 단일 서버에서 33주 이상의 많은 처리 시간을 필요로 한다. 본 실험에서는 CGRID 상에서 27대의 PC를 이용한 서열비교를 통하여 33주의 시간을 1주로 줄이는 성과를 얻었다.

4.1.2 수학적 분석

이 절에서는 수학적 분석을 통하여 오솔로그 데이터베이스 구축에 있어서의 비교 횟수 및 처리 시간을 예측한다. 다음 <표 4>는 수학적 분석을 위한 기호를 정한다.

표 4. 서열비교에 대한 수학적 분석을 위한 기호 정의
Table 4. Mathematical Notation for analyzing sequence comparisons

	Notation
Gi :	i 번째 종의 유전체
G :	유전체들의 집합, (Gi∈G)
GNi :	Gi의 유전자 개수
gj :	Gi 의 j 번째 유전자
N :	종의 개수
Tij :	Gi와 Gj의 유전자 비교 시간

N종을 대상으로 유전체간의 상호 비교 회수는 NC2가 된다. 또한 각 유전체 간의 비교에 있어서 유전자 간의 비교 횟수는 다음 식(1) 과 같다.

$$f(G_p, G_q) = \sum_{i=1}^{GN_p} \sum_{j=1}^{GN_{q\,v}} Compare(g^i_p, g^j_q) \quad \cdots\cdots \text{식(1)}$$

따라서 전체 유전체에서 유전자간의 비교회수는 식(2)와 같이 된다.

$$F = \sum_{i=1}^{N-1} \sum_{j=1}^{GN_i} \sum_{k=1}^{GN_{i+1}} Compare(g^j_i, g^k_{i+1}) \quad \cdots\cdots\cdots \text{식(2)}$$

우리는 89종에 대하여 유전자 비교를 하므로, 유전체간의 비교회수는 89C2=3916 만큼 일어나고, 전체 유전자 비교 회수는 아래 <표 5>의 데이터를 바탕으로 식(2)을 적용하면 32,933,560,000 만큼 발생한다.

표 5. 실험 유전체 및 유전체 내에 있는 유전자의 개수
Table 5. The number of genes on the 89 genomes

i	종 이름	GN_i
1	A. tumefaciens	4,554
2	A. aeolicus	1,529
3	B. subtilis	4,112
4	B. thetaiotaomicron VPI-5482	4,778
...	...	...

유전체 간의 비교 횟수는 대략 20분~4시간정도 걸리므로 전체 처리 시간을 다음 식(3)에 적용하면 하나의 서버로 2~21개월 정도 소요됨을 알 수 있다.

$${}_NC_2 * \min T_{ij}(i \geq 1, j \leq N) \leq T_{total} \leq {}_NC_2 * \max T_{ij}(i \geq 1, j \leq N)$$
$$\cdots\cdots\cdots \text{식(3)}$$

4.1.3 실험적 분석

4.1.3절에서는 CGRID 상에서 HPBS를 이용한 실제 실험 결과를 기술하고 그에 따른 분석결과를 설명한다.

HPBS는 89종의 실제 유전체를 <표 1>과 같이 펜티엄 III급 27대의 PC를 가지고 실험하였다. (그림 8)는 유전체 데이터(a) 와 처리하는 BLAST 연산(b)들을 시각화 하여 보여준다.

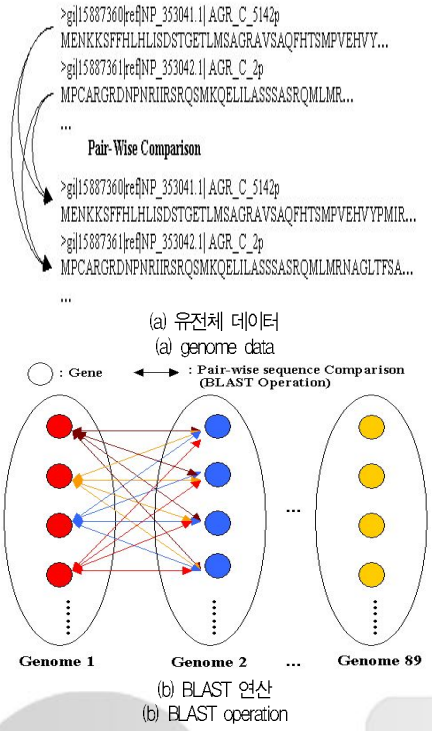

그림 8. 유전체 데이터와 BLAST 연산
Fig 8. Genome data and BLAST operation

89종의 유전체에 대하여 오솔로그 데이터베이스 구축을 위해서는 89C2 = 3,916만큼의 유전체간의 비교가 필요하며 각 유전체간의 비교는 평균적으로 약 8,410,000 정도의 BLAST 연산이 필요하다.

<표 6>은 CGRID에 있는 PC들이 1주일간 처리한 일의 양을 나타낸다. 이 결과에서 Group A가 Group B보다 두 배 이상의 일을 처리했음을 알 수 있다. 이러한 결과는 마스터 노드와 Group A의 PC들 간의 통신 속도가 Group B의 통신 속도보다 두 배 이상 빠르기 때문이다.

표 6. 각 PC 그룹에서 처리한 일의 양
Table 6. The amount of jobs completed in each group

Group A		Group B	
PC ID	완료된 작업 수	PC ID	완료된 작업 수
1	244	13	95
2	224	14	93
3	241	15	90
4	209	16	95
5	234	17	96
6	217	18	94
...	...	...	...

다음 <표 7>은 마스터 노드와 Group A의 PC들과 Group B의 PC들 간의 통신 속도를 보여준다. 이 결과는 ping 명령어를 이용하여 측정하였으며 마스터 노드로부터 각 그룹의 PC들에게 1,000의 ICMP ECHO_REQUEST를 보내어 응답시간의 최소, 평균, 최대를 보여주고 있다. 그 결과 마스터 노드와 Group A PC들의 네트워크 속도가 Group B PC들과의 속도보다 실제로 두 배 이상 빠름을 확인할 수 있다.

표 7. PC Group 노드들과 마스터 노드와의 통신 속도
Table 7. Network Speed between master node and slave nodes in each group

Time(ms)	Minimum	Average	Maximum
Group A	0.248	0.253	0.258
Group B	0.509	0.522	0.576

그림 9는 PC 수와 전체 처리 시간과의 상관관계를 보여주고 있다. 이 결과를 통하여 PC의 수를 증가시킬수록 전체 처리 시간이 33주에서 1주 이내로 처리됨을 알 수 있다. 이 실험결과는 CGRID 상에서 PC를 증가시킬수록 성능이 비례하여 증가함을 보여준다.

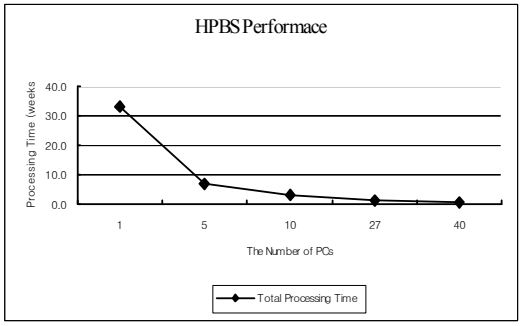

그림 9. PC 수와 전체 처리시간과의 상관관계
Fig 9. Correlation between the number of PCs and overall processing time.

그림 10은 PC의 수가 증가함에 따른 마스터 노드의 CPU 이용률 변화를 보여준다. PC의 수가 증가됨에 따라 메인 컴퓨터의 CPU 이용률이 증가하기는 하지만 급격하게 증가하지 않음을 볼 수 있다. 1대의 마스터 노드에 40대의 작업노드 PC를 이용하는 경우 20% 정도의 CPU를 사용하고 있음을 확인할 수 있다.

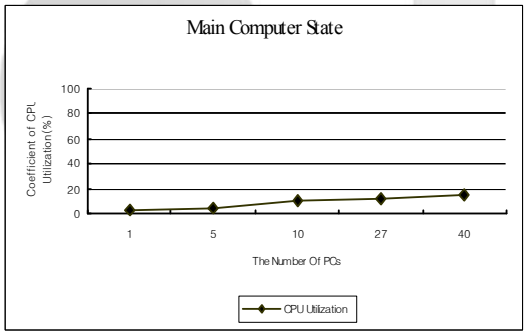

그림 10. PC 수와 마스터 노드의 CPU 이용률 간의 관계
Fig 10. The correlation between the number of PCs and CPU utilization

그림 10의 결과를 통해 노드 수가 증가함에 따라 마스터 노드의 부하 정도는 급격히 증가하지 않음을 알 수 있지만, 한 대의 마스터가 지원할 수 있는 노드의 수에는 한계가 있을 것이다. 이러한 어려움을 극복하기 위하여 중간 단계의 관리 노드(Management Node)를 둘 예정이다.

V. 결론 및 향후 연구

본 논문에서는 PC를 하드웨어 기반의 그리드인 CGRID를 제안하였고, 오솔로그 데이터베이스 구축을 통해 검증하였다. CGRID는 기존의 소프트웨어적인 방식으로 구축된 그리드와는 달리 부트롬을 이용하는 하드웨어적인 접근법을 활용하였다. 이 접근법은 그리드를 효율적으로 구축할 수 있고 유지 및 관리 측면에서 편리하다. 평상시에는 그리드의 노드를 일반적인 용도로 사용할 수 있으므로 가용성이 높다. 또한 유휴시간에 저비용으로 기존의 PC들의 자율성을 보장하면서 고성능 컴퓨팅 환경을 지원한다.

CGRID 상에서 이루어진 오솔로그 데이터베이스 구축 실험에서 약 33주 정도의 BLAST 연산을 27대의 PC를 이용하여 1주로 단축할 수 있었다. CGRID 상에서 처리 성능은 노드를 추가함으로써 비례하여 증가하였다. 또한, 노드를 추가함에 따라 마스터 노드의 부하 정도는 급격하지 않음을 확인하였다. 40대의 노드가 있을 경우 마스터 노드는 CPU의 약 20% 점유율을 차지하였다.

향후 연구로는 CGRID 상에서 수 백 또는 수 만개 그 이상의 노드가 확장됨에 따른 마스터 노드의 부하를 분산하기 위한 중간 단계의 관리 노드를 추가할 예정이다. 또한 많은 수의 노드를 안정성 있게 운영할 뿐만 아니라 원활한 관리를 위한 노드 운영 정책이나 프로토콜 설계를 보다 발전시킬 예정이다. CGRID 구축 및 관리에 대한 연구와 함께 실제 구축된 그리드에서 처리될 수 있는 다양한 응용분야들을 적용할 것이다.

참고문헌

[1] 김홍숙, “Hyper-BLAST : A Parallelized BLAST on Cluster System,” A Dissertation of the Degree of Philosophy in ICU, 2003.

[2] 강명주, “클러스터 기반의 멀티캐스트 라우팅 문제 해법을 위한 유전자 알고리즘,” 한국컴퓨터정보학회 논문지 제8권 제3호, 2003.9.

[3] 김성락, “실시간 Linux 환경에서 효율적인 스케쥴링을 위한 선택 알고리즘의 구현,” 한국컴퓨터정보학회 논문지 제7권 제2호, 2002.1.

[4] S. F. Altschul, et al., “Basic Local Alignment Search Tool,” Journal of Molecular Biology, 215:403-410, 1990.

[5] S. F. Altschul and W. Gish, “Local alignment statistics,” Methods in Enzymology, 266:460-480, 1996.

[6] S. F. Altschul et al., “Gapped BLAST and PSI-BLAST: A new generation of protein database search programs,” Nucleic Acids Research, 25:3389-3402, 1997.

[7] Etzold, T. and Argos, P. SRS: An Indexing and Retrieval Tool for Flat File Data Libraries, Computer Applications in the Biosciences, 9, 49-57, 1993.

[8] I. Foster, C. Kesselman, and S. Tuecke, “The Anatomy of the Grid : Enabling Scalable Virtual Organizations,” International J. Supercomputer Application, 15(3), 2001.

[9] I. Foster, “The Grid: A New Infrastructure for 21st Century Science,” Physics Today, 55(2), pp. 42-47, 2002.

[10] I. Foster and C. Kesselman. “Globus: A Metacomputing Infrastructure Toolkit”, Intl J, Supercomputer Applications, 11(2), pp. 115-128, 1997.

[11] I. Foster and C. Kesselman, eds., The Grid: Blueprint for a New Computing Infrastructure, Morgan Kaufmann; 1st edition, Jan. 1999.

[12] Cynthia Gibas, Per Jambeck, Developing Bioinformatics Computer Skills, O'REILLY, 2002

[13] W. Gish and D. States. "Identification of protein coding regions by database similarity search," Nature Genetics, 3:266-272, 1993.

[14] Higgins, D. and Sharp, P. "CLUSTAL: A Package for Performing Multiple Sequence Alignment on a Microcomputer," Gene, 73, 237-244, 1988.

[15] Heath A. James, Scheduling in Metacomputing Systems, BSc(Ma&Comp Sc) (Hons). 1999

[16] Minoru Kanehisa, editor. Post-Genome Informatics. Oxford University Press, 1998.

[17] Yu-Lun Kuo and Chao-Tung Yang, "Apply Parallel Bioinformatics Applications on Linux PC Clusters," Tunghai Science Vol. : 125?141 125 July, 2003.

[18] Y. L. Kuo, et al., "Construct a Grid Computing Environment for Bioinformatics," In Proc. of the International Symposium on Parallel Architectures, Algorithms and Net-works(ISPAN'04), 1087-4089, 2004.

[19] Laskowski, R., MacArthur, M., Moss, D. and Thornton, J. "PROCHECK: A Program to Check the Stereochemical Quality of Protein Structures," Journal of Applied Crystallography, 26, 283-291, 1993.

[20] Lavery, R. and Sklenar, H. "The Definition of Generalized Helicoidal Parameters and of Axis Curvature for Irregular Nucleic Acids," Journal of Biomolecular Structure and Dynamics, 6, 63-91,1988.

[21] Soojin Lee, et al., "Exploring protein fold space by secondary structure prediction using data distribution method on Grid platform," Bioinformatics, Advance Access published on July 29, 2004.

[22] A. Mark Baker and Geoffery C. Fox, "Metacomputing: Harnessing Informal Supercomput-ers," In High Performance Cluster Computing, Prentice-Hall, ISBN 0-13-013784-7, May 1999.

[23] Siepel, A., Farmer, A., Tolopko, A., Zhuang, M., Mendes, P., Beavis,W. and Sobral, B. "ISYS: A Decentralized, Component based Approach to the Integration of Heterogeneous Bioinformatics Resources," Bioinformatics, 17(1), 83-94, 2001.

[24] Roman L. Tatusov, et al., "The COG Database: A Tool for Genomic-Scale Analysis of Protein Function and Evolution", Nucleic Acids Res. 28, 33-36, 1999.

[25] L. Wang, et al., "Biogrid Computing Platform: Parallel computing for protein alignment analysis," HPC Asia'02, Bangalore, India, 2002.

[26] Yamanishi, Y., Akiyasu C. Yoshizawa, Itoh, M., Katayama, T., Kanehisa, M., "Extraction of Organism Groups from Whole Genome Comparisons," Genomic Information 14, 438-439, 2003.

[27] Chao-Tung Yang and Chi-Chu Hung, "High-performance computing on low-cost PCs based SMPs clusters," In Proc. of the 2001 National Computer Symposium (NCS 2001), Taipei, Taiwan, pp. 149-156, Dec. 2001.

[28] Beowulf Cluster, http://www.beowulf.org/

[29] DDBJ, http://www.ddbj.nig.ac.jp/

[30] EtherBoot Project, http://etherboot.sourceforge.net/

[31] FightAIDSAtHome, http://www.fightaidsathome.org/

[32] Global Grid Forum, http://www.ggf.org

[33] The Globus Project, http://www.globus.org/

[34] JPred, http://www.compbio.dundee.ac.uk/~www-jpred/new.html

[35] KEGG, http://www.genome.jp/kegg/

[36] KISTI Grid Testbed, http://gridtest.hpcnet.ne.kr/

[37] LanLinux, http://www.lanlinux.com

[38] LHC - The Large Hadron Collider Home Page, http://lhc-new-homepage.web.cern.ch/

[39] NASA Launchpad Portal, http://portal.ipg.nasa.gov

[40] NCBI, http://www.ncbi.nlm.nih.gov/

저 자 소 개

김 태 경

2005년 2월 충북대학교
정보산업공학과 석사
2005~현재 정보산업공학과 박사
과정 재학 중
<관심분야> 바이오인포메틱스,
그리드 컴퓨팅, XML,
데이터웨어하우스

조 완 섭

2005년 11월 충북대학교 경영정보과
부교수
<관심분야> 바이오인포메틱스,
데이터 웨어하우스
OLAP,데이터베이스