DVSF: 가상 검색을 위한 분산 시스템
(DVSF: A Distributed System for Virtual Screening)

황 석 찬 † 최 종 선 † 이 상 근 †† 임 재 훈 †
(Seogchan Hwang) (Jongsun Choi) (Sangkeon Lee) (Jaehoon Lim)
최 재 영 ††† 노 경 태 †††† 이 지 수 †††††
(Jaeyoung Choi) (Kyoung Tai No) (Jysoo Lee)

요 약 오늘날 사람들의 많은 관심을 끌고 있는 생명 공학 분야에서, 가상 검색은 컴퓨터에서 가상적으로 시뮬레이션을 수행함으로서 특정 질병에 관여하는 물질을 보다 빠르게 추출하여 신약을 설계하는 방법이다. 본 논문에서는 지역적으로 분산된 유휴 또는 전략적 자원을 이용하여 가상 검색을 효과적으로 수행할 수 있는 분산 시스템인 DVSF(Distributed Virtual Screening Facilities)를 소개한다. DVSF는 중소규모의 실험실이 가지고 있는 유휴 자원을 이용하여 최소한의 비용으로 효과적인 신약 개발을 위한 환경을 구성할 수 있도록 개발되었다.

키워드 : 그리드, 그리드 미들웨어, 가상 검색

Abstract As one of a new generation of industries, biotechnology is currently spotlighted, and is creating new research and industrial areas that can be readily combined with information technology. Among them, virtual screening is a method for new drug design which uses computer simulation to virtually design active drug candidates for special disease. In this paper, we present a distributed system for virtual screening, called DVSF(Distributed Virtual Screening Facilities). DVSF is designed and developed to perform virtual screening efficiently on logically distributed idle or strategic resources in a small or medium scale laboratory.

Key words : Grid, Grid Middleware, Virtual Screening

1. 서 론

현재의 생명공학 분야는 IT를 결합한 새로운 패러다임으로 빠르게 변화하고 있다. 인간 유전자 지도와 생물정보학(Bio-Informatics)을 이용하여 특정한 질병에 관여하는 유전자와 단백질에 대한 정보를 얻을 수 있는데, 그 정보를 이용하여 특정한 질병에 생리 활성을 보일 것으로 예상되는 신약 후보 물질을 설계할 수 있다. 최근에는 조합 화학(Combinatorial Chemistry)과 화학 게놈(Chemical Genomic)에 고속 대량 처리 검색(High Throughput Screening) 기술이 접목되어, 소규모 실험실 규모에서도 컴퓨터를 이용하여 신약 후보 물질을 도출할 수 있다. 이러한 방법을 가상 검색(Virtual Screening)이라고 한다[1, 2, 3]. 가상 검색은 특정 질환에 관여하는 단백질이 활성화된 곳에, 화합물 데이타베이스(실험적으로 합성된 화합물들과 가상적으로 만들어진 화합물들을 모두 포함)에 있는 화합물들을 하나씩 결합시켜 보는 도킹(Docking) 과정을 통하여, 강한 상호 작용을 할 것으로 기대되는 신약 후보 물질을 가상적으로 도출해내는 방법이다.

· 이 논문은 한국과학기술정보연구원의 국가 그리드 기반 구축 사업과 교육부 BK-21 핵심분야 사업의 지원으로 수행되었습니다.
† 비 회 원 : 숭실대학교 컴퓨터학과
seogchan@dreamwiz.com
goodboy3@ss.ssu.ac.kr
suplim@ss.ssu.ac.kr
†† 학생회원 : 숭실대학교 컴퓨터학과
seventy9@ss.ssu.ac.kr
††† 종신회원 : 숭실대학교 컴퓨터학부 교수
choi@comp.ssu.ac.kr
†††† 비 회 원 : 숭실대학교 교수/분자설계기술혁신센터 소장
ktno@camd.ssu.ac.kr
††††† 비 회 원 : 한국과학기술정보연구원 슈퍼컴퓨팅 연구실장
jysoo@hpcnet.ne.kr
논문접수 : 2002년 6월 11일
심사완료 : 2002년 9월 16일

한편, 그리드는 지역적으로 분산되어 있는 계산적 자원을 통합하여, 병렬 처리 혹은 분산 처리 형태의 대규모 계산이나 데이타 중심적인 응용 프로그램을 수행하기 위한 확장된 컴퓨팅 플랫폼을 제공한다[4]. 그리드는 많은 데이타를 처리해야 하는 과학이나, 공학, 상업적 연구에서 발생할 수 있는 대량의 계산문제에서 새로운 해결책을 이끌어낼 수 있는 역할을 수행하는데 적합하다. 가상 검색을 이용해서 신약을 설계하는 연구에도 그리드의 이러한 계산적 능력을 사용할 수 있다.

본 논문에서는 분산된 네트워크 자원을 이용하여 가상 검색을 지원하는 분산 시스템인 DVSF(Distributed Virtual Screening Facilities)의 시스템 구성과 구현에 대하여 기술한다. 2장에서는 도킹 소프트웨어인 Auto Dock 프로그램을 중심으로 가상 검색과 함께, 이를 분산 시스템에 적용시키기 위한 과정을 분석한다. 3장은 DVSF의 설계 및 구현을 기술하고, 4장은 관련 연구 및 동향에 대해서 논의한다. 마지막으로 5장에서는 결론을 맺는다.

2. 가상 검색

가상 검색은 특정 질환에 관여하는 단백질이 활성화된 곳에, 화합물을 하나씩 결합시켜 보는 도킹(Docking) 과정을 통하여 신약 후보 물질을 가상적으로 도출해내는 방법이다. 따라서 한 번의 도킹 작업을 수행하기 위해서는 단백질 하나와 화합물 하나가 필요하다. 전체 가상 검색 작업에서 단백질은 특정한 하나만 필요하므로, 작업 컴퓨터들은 그 단백질을 한 번만 복사해서 저장해 놓는다. 작업 컴퓨터들은 필요할 때마다 데이타베이스로부터 화합물을 하나씩 가져오는데, 데이타베이스에 저장된 화합물의 크기는 대부분 10K를 넘지 않는다.

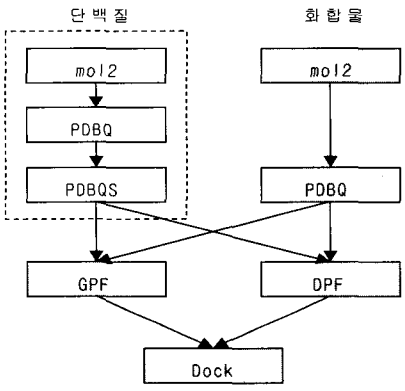

그림 1 AutoDock 실행 과정

가상 검색은 계산량과 비교하여 데이타의 이동과 저장을 포함한 통신이 아주 작은 부분을 차지하는 계산 집중적인 작업이기 때문에, 널리 분산되어 있는 분산 컴퓨팅 자원을 이용하는 시스템에 적합하다.

AutoDock[5]은 도킹 작업을 수행하는 프로그램으로 그림 1의 단계들을 수행한다. 그림에서 보듯이 Auto Dock은 도킹에 필요한 단백질과 화합물의 처리 과정이 분리되어 수행된다. 각 단계별 단백질과 화합물의 출력 형식(format)을 사각형 안에서 나타내었다. 단백질과 화합물은 처음에 mol2 형식으로 저장되어 있다. mol2 형식의 단백질을 partial atomic charges와 atomic solvation parameters를 포함하는 고분자(macro molecule) 구조로 만들기 위해, 점선으로 된 사각형에서 보듯이, PDBQ를 거쳐 PDBQS의 형태로 변환시킨다.

한편 mol2 형식으로 데이타베이스에 저장되어 있는 화합물은 앞서 생성한 단백질 고분자 구조에 도킹시키기 위해 PDBQ 형식으로 변환시킨다(torsion). 준비된 각각의 단백질과 화합물 구조를 3차원 형태의 퍼즐 모양으로 구성되는 GPF(Grid Parameter File) 형식과 도킹에 필요한 관련 파라미터를 계산하는 DPF(Docking Parameter File) 형식으로 변환시킨다. 변환된 결과를 이용하여 그리드 맵과 최종 에너지 값을 계산한다.

그림 1에서 점선으로 된 사각형 안은 단백질과 관련한 명령으로서 한번만 수행하여 저장해 놓았다가, 필요할 때에 이미 계산된 결과를 사용한다. 나머지 음영 부분은 각 화합물마다 실행하여야 하는 명령으로, 각각의 작업을 파라미터 변형(parameter sweep)을 이용하는 배치 형태의 작업으로 구성한다.

위와 같은 분산 시스템을 이용한 가상 검색 방법은 기존의 보유자원을 이용할 수 있는 효율적인 방법을 제시하며 신약 후보 물질을 도출하는 과정에서도 그 시간과 비용을 절감할 수 있다. 그러나 분산 시스템을 설치하는 과정에서는 DVSF 구성 요소와 함께 가상 검색에서 필요로 하는 도킹 소프트웨어인 AutoDock을 각 실행 노드마다 설치해야 하는 번거로운 작업이 뒤따라야 한다.

3. DVSF 시스템

가상 검색을 위한 분산시스템인 DVSF(Distributed Virtual Screening Facilities)의 작업 과정을 그림 2에서 나타내었다. DVSF는 작업 노드(Working Node)에 설치되어 해당 노드의 자원 상태를 관리하고, 도킹 프로그램인 AutoDock을 수행시키는 노드 관리자(Node Manager와, 자원 working node, DB, etc)의 정보를 관리하

는 정보 관리자(Information Center), 그리고 작업과 자원을 연결해주고 스케줄을 관리하는 작업 관리자(Job Manager)로 구성된다.

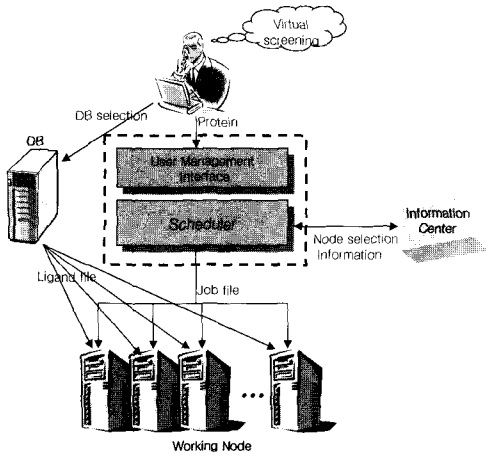

그림 2 가상 검색 작업 과정

사용자는 PDBQS 파일 형태로 미리 계산되어 있는 단백질 구조와, 이 단백질에 결합하여 화학적 반응값을 계산할 화합물을 저장하고 있는 데이타베이스를 선택한다. 단백질 구조와 선택한 데이타베이스, 기타 작업에 필요한 파라미터를 가지고 있는 작업 파일은 점선안의 작업 관리자에게 전달된다. 작업 관리자는 사용자의 작업 요청을 분석하여, 정보 관리자를 이용하여 계산에 필요한 자원들에 대한 정보를 가져오고, 해당 노드를 관리하는 노드 관리자와 자원 사용을 위한 정보를 교환한다. 작업 관리자는 선정된 작업 노드들에 대한 큐를 생성하고 작업을 분배한다. 작업 노드는 데이타베이스로부터 필요할 때마다 화합물을 가져와 작업을 수행하고, 작업이 종료되면 작업 관리자의 결과 저장소에 그 결과를 저장한다. 전체 작업이 완료된 후에 사용자 요구대로 결과를 정렬하여 사용자에게 보낸다.

3.1 작업 관리자(Job Manager)

작업 관리자는 사용자 관리(User Management Interface), 자원 관리(Resource Scheduler), 작업 관리(Job Scheduler)의 세 가지 일을 담당한다. 사용자 관리는 사용자가 처음 접속하면서부터 종료할 때까지의 모든 사용자 요구를 처리하는 부분으로, 패스워드를 사용하는 사용자 인증, 작업 분석기(Job Analyzer)를 통한 사용자가 제출한 작업 분석, 콜렉터(Collector)를 통한 결과 취합 등의 기능을 포함하고 있다. 자원 관리는 분석된 작업이 필요로 하는 자원들을 관리하는 부분으로서, 정보 관리자(Information Center)를 통한 자원의 선택과 추가되는 자원의 재할당, 자원 모니터링 등의 기능을 포함한다. 작업 관리는 자원 관리 부분으로부터 배정받은 자원을 이용하여 사용자 작업을 실제 배분하는 기능을 수행한다. 작업 노드에 대한 큐 생성, 각 노드에 성능별 작업 스케줄링, 다른 노드로의 작업 이관 등의 기능을 포함한다.

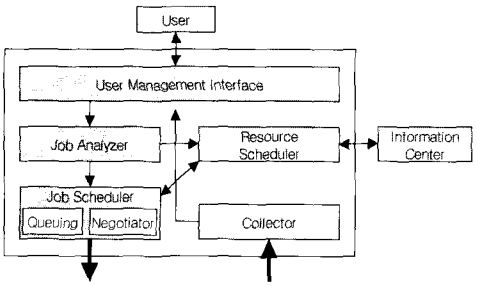

그림 3 작업 관리자 구성도

작업 관리자의 내부 구성은 그림 3과 같이 사용자 관리자(User Management Interface), 작업 분석기(Job Analyzer), 자원 관리자(Resource Scheduler), 작업 관리자(Job Scheduler), 콜렉터(Collector)로 구성된다. 사용자가 요청한 작업을 처리하기 위해서, 먼저 작업 분석기를 이용하여 작업에 필요한 요구 사항을 분석한다. 도킹에 필요한 단백질 구조와 데이타베이스의 선택, 작업 수행을 위한 파라미터 처리, 필요한 자원 형태와 규모 등을 분석한다. 분석된 데이타를 바탕으로, 자원 관리자는 작업을 수행시키는데 필요한 자원을 찾기 위해 외부의 정보 관리자(Information Manager)와의 정보를 교환한다. 찾아진 자원 정보는 작업 관리자에게 통보된다.

이때 작업 관리자는 선택된 작업 노드와 정보를 교환하면서, 노드가 실제로 작업이 수행될 수 있는 환경을 갖추었는지, 그리고 해당 작업 노드의 정책에 따라 자원의 예약이 가능한지를 확인하는 교섭(negotiation) 과정을 거친다. 정보 관리자로부터 받은 자원(작업 노드)의 정보는 일정한 주기로 갱신되는 정적인 상태의 정보이므로, 실제로 사용되는 시점에서는 노드 상태가 다를 수 있기 때문에, 교섭 과정이 필요하다. 이러한 과정을 거쳐서 부적합하다고 판정될 경우, 자원 관리자를 통해 다른 자원을 할당받는다. 작업에 필요한 자원이 충족될 때까지 이러한 과정을 반복한다. 그림 4는 자원 교섭 프로토콜을 나타내었다. 음영 안의 점선으로 표시된 프로토

콜은 노드 관리자로부터 자원의 사용이 거부되었을 때, 정보 관리자로부터 새로운 자원에 대한 정보를 받아 다시 교섭하는 부분을 보여준다.

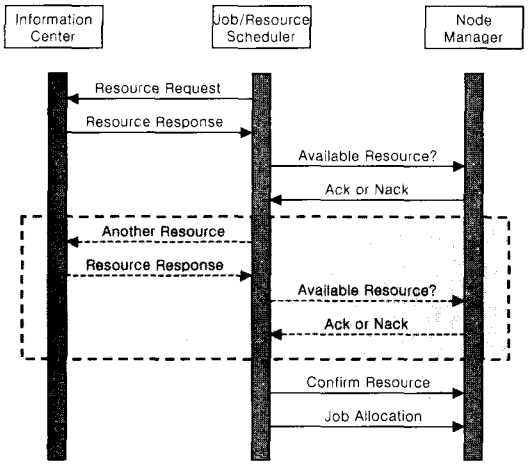

그림 4 자원 교섭 프로토콜

자원의 예약이 완료되면 각각의 작업 노드에 대해서 작업 큐를 생성한다. 큐는 각 작업 노드의 로컬 작업 큐에 직접 매핑된다. 각 노드는 작업 관리자로부터 기본 큐의 크기만큼 작업을 할당받아 수행한다. 할당된 작업의 수행이 끝나면 콜렉터로 결과를 보낸 후에 작업 관리자에게 작업의 재할당을 요구한다. 한편 콜렉터는 작업노드로부터 받은 결과를 모아 사용자의 요구에 따라 정렬하고, 그 결과를 사용자에게 보낸다.

3.2 노드 관리자(Node Manager)

노드 관리자는 작업을 실행시키는 로컬 스케줄러(Local scheduler), 외부 시스템(DB, File server, 작업 노드 등)과의 데이타 통신을 위한 UGDS(Universal Grid Data System)[6], 시스템의 자원과 프로세스를 모니터링 하는 도구로 구성되어 있다.

노드 관리자는 유휴 상태 또는 여유 자원을 활용하기 위해 미리 설정된 정책에 따라 자신의 노드 상태를 외부의 정보 관리자(Information Center)에 등록한다. 이후에 작업 관리자(Job Manager)로부터의 작업 교섭을 요청받으면 현재 노드의 상태를 보고하고, 조건이 만족되면 작업을 할당받는다. 해당 노드에 있는 로컬 스케줄러는 요청된 작업을 수행된다. 로컬 스케줄러는 PBS (Protable Batch System) 등과 같은 큐잉 시스템과 연동하며, 큐잉 시스템이 없거나 단일 노드인 경우에는 fork 시스템 콜과 스레드를 이용하여 자체적으로 작업을 수행시킨다.

수행 중인 작업은 모니터 시스템을 통하여 지속적으로 관찰한다. 그리고 수행 중에 발생한 에러를 점검하여 중앙의 작업 관리자에 보고하여 정지, 재실행, 다른 노드로의 작업 이관 등을 준비한다. 작업 수행 중에 필요한 mol2 파일과 같은 외부 데이타는 UGDS를 통하여 가져온다. UGDS는 DB 서버, FTP 서버, 파일 시스템 등 여러 형태의 저장소에 산재된 데이타를 단일 인터페이스를 이용하여 읽어온다[6].

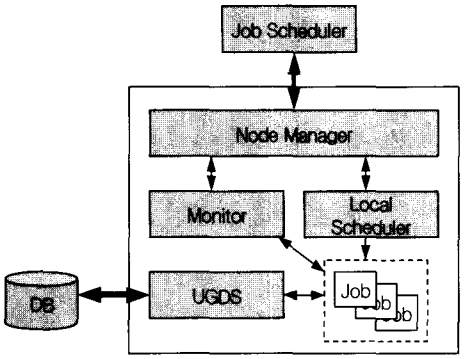

그림 5 작업 관리자의 구성도

3.3 정보 관리자(Information Center)

정보 관리자는 자원을 소유하고 있는 자원 제공자(Node Manager)와 자원을 사용하고자 하는 자원 소비자(Job Manager) 사이에서 중계 역할을 한다. 각 작업 노드가 유휴 상태이거나 혹은 임의의 정책에 따라서 노드 풀(node pool)에 참여를 희망하면, 노드 관리자는 해당 노드의 정보(CPU, 메모리, 디스크, 가용률, 설치 프로그램 유무 등)를 정보 관리자에 등록한다. 정보 관리자는 이러한 정보를 기반으로 작업 관리자의 요청에 맞는 자원을 선별하여 전해준다.

정보 관리자의 메타 정보 시스템은 그림 6과 같은 Producer-Consumer 모델로 구성되며 디렉토리 기반의

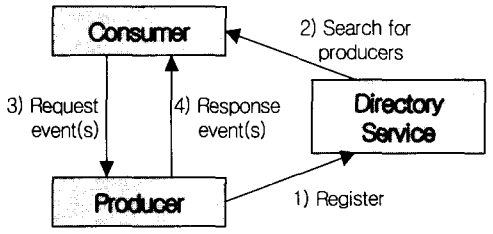

그림 6 정보 관리자 모델

서비스를 위해 LDAP를 이용하여 그림 7의 구조로 구현하였다. LDAP는 IP 프로토콜 기반의 표준화된 소프트웨어 프로토콜로서 정보를 디렉토리 서버에 계층적인 방식으로 표현하여 저장한다. 이러한 특징은 관계형 데이타베이스보다 데이타 양이 증가하더라도 검색에 대한 성능이 일정하다는 장점이 있다[7].

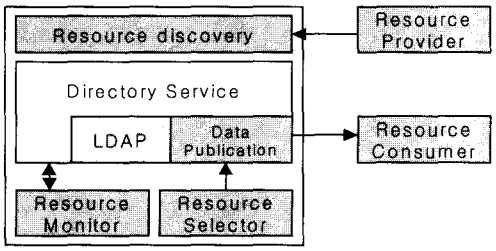

그림 7 정보 관리자의 시스템 구조

정보 관리자의 시스템 구조는 디렉토리 서비스와 이를 지원하는 자원 발견(discovery), 배포(publication), 감시(monitoring), 선택(selector) 서비스로 구성되어 있다. 자원 발견 서비스는 새로운 자원의 정보를 갱신하기 위한 기능으로, 사용 가능한 자원에 대한 메타 정보를 저장하도록 한다. 노드 관리자가 해당 자원을 등록하면, 자원 발견 서비스는 정보를 LDAP 스키마에 맞는 형식으로 변환하여 저장한다. 자원의 목록은 분산 환경에 적합하도록 디렉토리로 구성하며, 그림 8과 같은 LDAP 스키마 구조를 정의하고 있다[8].

자원 할당 서비스나 작업 관리자(Job Manager) 등과 같이 자원에 대한 정보를 필요로 하는 자원 소비자가 자원에 대한 정보를 요청하면, 자원 배포 서비스는 LDAP 서버에 저장되어 있는 적합한 자원의 메타 정보를 찾아 그 정보를 제공한다. 최적으로 자원들을 관리하고 그 자원들을 최대로 활용하기 위해서, 자원 감시 서비스는 주기적으로 등록된 자원의 변동 상황을 점검하여 정보를 갱신한다.

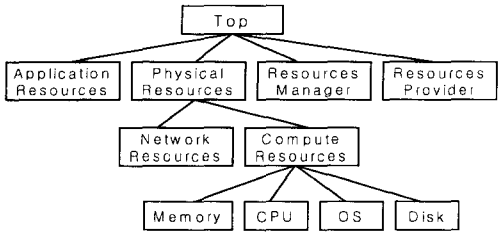

그림 8 정보 관리자의 LDAP 스키마 구조

3.4 구현 및 테스트

DVSF는 리눅스에서 C로 구현되었으며, 정보 관리자에서 사용되는 디렉토리 서버로 openLDAP 2.0을 사용하였고, 도킹 소프트웨어로는 AutoDock for linux v3.0을 사용하였다. 테스트 플랫폼으로는 600 MHz Penti um III로 구성된 4 노드 PC 클러스터와 함께, Pentium 4(1.6 GHz, 512 MB)와 Pentium III(500 MHz, 256 MB)의 일반 데스크탑 컴퓨터 2대를 이용하여 구성하였다. Pentium III PC에는 작업 관리자와 정보 관리자를 설치하고, 다른 Pentium 4 PC에는 샘플 화합물 데이타를 위해 리눅스용 오라클을 설치하였다. 클러스터는 작업 노드로 사용하며 모두 리눅스를 운영체제로 사용한다. 각 컴퓨터는 100 Mbps Ethernet으로 연결되었고, 작업 노드의 스케줄러는 모두 fork를 이용하여 수행하였다. 화합물 데이타베이스는 1000개의 샘플 데이타를 이용하여 구성하였으며 도킹 작업의 특성상 화합물을 복사하는 네트워크 부하는 배제할 수 있을 정도로 미비하므로 같은 작업 노드와 같은 네트워크 안에 포함시켰다.

DVSF의 목적은 중소규모 실험실 수준에서의 장비를 이용하여 최대한의 활용을 이끌어내기 위한 것이므로 기존의 컴퓨터가 가지고 있는 성능을 그대로 활용한다. 때문에 직접적으로 성능을 비교하기는 어렵지만, 많은 컴퓨터를 동시에 이용할 수 있다는 활용성에 초점을 맞추어 실험 결과를 서술한다.

테스트에 사용되는 모든 설정은 기본값을 사용했고 샘플 데이타 200개를 기준으로 하여 각각 하나의 노드부터 차례로 4개까지의 노드를 증가시켜서 실험을 수행하였다. 그림 9에서 나타내듯이 계산 참여 노드가 증가할수록 작업 수행 시간이 감소하는 반비례 관계를 보여준다. 이는 가상 검색 프로그램의 특성상 프로세스간의 통신을 수행하지 않는 독립적인 작업이기 때문에 동시 수행 프로세스가 늘어나도 통신과 관련한 오버헤드를 가지지 않고 오직 노드의 성능에 의존적이기 때문에 계

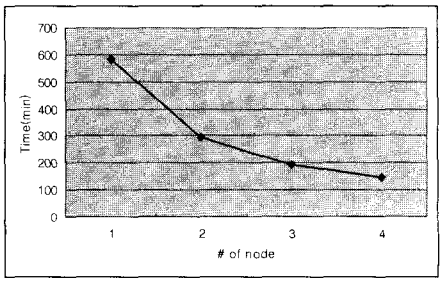

그림 9 DVSF의 성능 평가

산에 참여하는 노드수가 증가하는 만큼 전체적인 성능이 이에 비례하여 좋아진다.

그림 10은 작업 노드 상태와 진행 중인 전체 작업에 대한 정보를 나타내는 화면이다. 작업에 참여하는 4대의 노드는 모두 같은 성능을 갖는 PC로 큐의 초기값이 변동이 없는 상태로 작업이 수행 중(status -> 1)인 것을 나타내고 있다. 작업 정보는 전체 200개의 작업 중 81번째의 작업이 큐에 대기상태인 것을 보여주고 있으며 기타 화합물 데이타베이스 정보와 단백질에 관련한 정보를 포함하고 있다. 그림 11은 작업 수행후의 결과를 ViewerLite[9]를 이용하여 단백질과 화합물의 결합 구조를 보여준다.

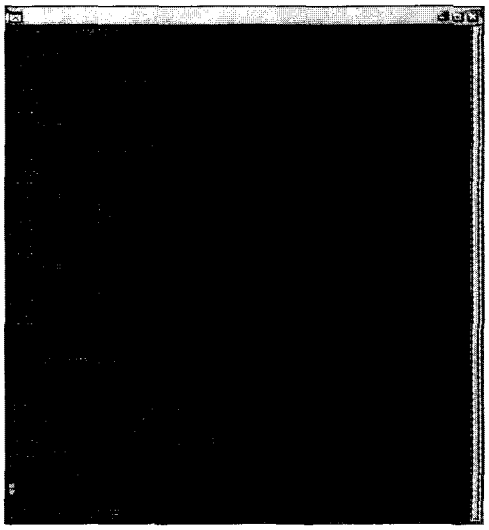

그림 10 작업 중의 노드 상태와 작업 정보

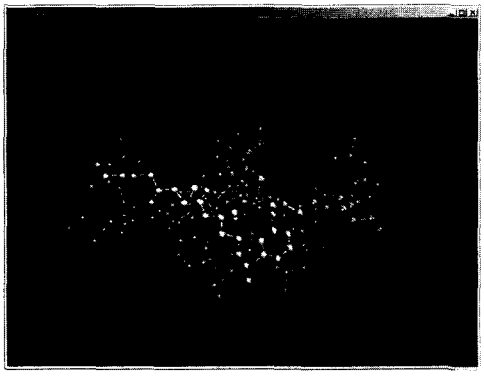

그림 11 예제 실행 화면

4. 관련 연구

현재 그리드 관련 연구는 세계적으로 많은 곳에서 진행되고 있으며, 또한 이와 관련된 응용 프로젝트도 활발히 이루어지고 있다. 그리드 미들웨어는 대표적인 Globus Toolkit[10]을 비롯하여 Condor[11], GRACE[12] 등이 있다. 또 이를 응용하는 다양한 그리드 관련 프로젝트가 진행 중이다.

Globus는 그리드를 구축하기 위해 미국 아르곤 국립연구소, Chicago 대학, USC 등의 합작 프로젝트로서 응용 프로그램이 분산된 이기종 컴퓨팅 자원들을 하나의 가상 컴퓨터처럼 사용하기 위한 소프트웨어 인프라이다. 연구의 핵심은 Globus Toolkit이다. 이것은 그리드에서 요구되는 서비스와 기능들을 제공하기 위한 도구들로서 자원 할당 및 프로세스들을 관리하는 GRAM, 서로 다른 통신 시스템간의 통신 기능을 제공하는 Nexus, 사용자 인증을 위한 GSI, 분산된 자원들의 상태와 정보를 제공하는 MDS, 프로그램 인터페이스를 이용하여 원격지의 데이타를 접근할 수 있는 GASS 등의 도구들로 구성되어 있다.

미국 Wisconsin 대학에서 개발한 Condor는 분산된 워크스테이션들의 유휴 자원을 통합하여 사용할 수 있게 하는 고속 대량 처리(High Throughput) 컴퓨팅 환경이다. Condor는 내부의 같은 도메인(intra-domain)의 유휴 자원들을 활용하기 위한 컴퓨팅 환경으로서, ClassAd라는 기술자를 이용하여 자원 제공자와 자원 수요자를 연결하여 분산 작업을 수행하게 된다. Condor-G는 Globus Toolkit을 이용하기 위한 상위 계층의 응용 도구로서, 기존의 Condor는 해당 도메인 내부 자원만을 사용하지만, Condor-G는 서로 다른 도메인(inter-domain)의 자원들도 함께 사용할 수는 멀티 도메인의 작업 환경을 제공할 수 있도록 개선되었다.

FightAIDS@Home 프로젝트는 Entropia의 분산 컴퓨팅 네트워크를 기반으로 수행된다[13]. 이 프로젝트에 참여하기를 원하는 지원자는 스크린 세이버 형태의 프로그램을 다운로드 받아 설치한다. 해당 컴퓨터가 유휴 상태가 되면, 서버에 자동적으로 접속해서 관련 데이타를 내려받은 후에, 도킹 작업을 수행한다. 작업을 마치면 그 결과를 서버에게 보내게 된다. 이러한 수행 방법은 전형적인 인터넷 컴퓨팅 모델로서, 스케쥴러가 직접 작업을 할당하여 수행하고 자원을 관리하는 본 연구와는 차이점이 있다.

호주 Monash 대학에서 수행 중인 Virtual Laboratory [14]은 분산 데이타와 컴퓨팅 자원을 이용하여 신

약을 설계를 위한 분자 모델링 환경을 구성하는 프로젝트로서GRACE(GRid Architecture for Computational Economy)구조 위에 구현되었다. GRACE는 Globus를 기반 구조로 채택하였으며 분산된 자원의 연결과 관련한 방식으로 Economy/Market 모델을 기반으로 연구되었다. 본 연구와 같은 응용 연구로서, 도킹 소프트웨어의 사용은 다르지만 분산자원을 이용하여 가상 검색을 위한 시스템 환경을 구축하였다는 점에서 많은 공통점을 가지고 있다. 그러나 Globus를 기반으로 구현되었으며, 자원 브로커에서 마켓 모델을 이용한 경제적인 관점에서 연구가 수행되었다.

광역의 분산된 자원을 이용하기 위한 대표적인 미들웨어로서 Globus는 글로벌(multi-domain) 규모의 자원을 이용하는 프로젝트와 관련하여 표준적인 서비스를 구성할 때에 핵심적인 미들웨어로서의 역할을 담당한다. 그러나 Globus는 각 구성요소의 서비스 정의를 이용하여 다른 서비스와의 - Globus에서 제공하지 않는 응용에서 필요한 부가 서비스와의 - 융합을 위해 필요한 추가적인 노력이 글로벌한 규모의 연구와 비교해서 더 크기 때문에 일반적인 중소규모의 실험실에서의 연구 환경을 위한 요소로서는 너무 무겁다고 할 수 있다.

5. 결 론

본 논문의 DVSF는 중소규모의 실험실에서 유휴 자원을 이용하여 최소한의 비용(소프트웨어 설치, 운영 등의 용이성)으로 효과적인 신약 개발을 위한 환경을 구성하기 위해 개발되었다. 또한 글로버스와 같은 모듈별로 구성하고 각 구성 요소간의 의존 관계를 배제하여 확장성을 갖도록 구현하였으며, 향후 글로버스와 연계하여 다른 응용을 쉽게 구성할 수 있는 기술을 축적할 수 있는 기반을 구축하였다. 그리고 DVSF를 실험실 환경에서 쉽게 설치해서 사용할 수 있도록 하기 위해서 작업 노드에 설치되는 요소의 크기를 200K 미만으로 하여 가벼운 구조를 가지며, 노드 관리자를 설치하는 것만으로 쉽게 다른 노드에 확장할 수 있다. DVSF에서 사용되는 도킹 소프트웨어인 AutoDock은 Virtual Laboratory에서 사용되는 Dock[15]과 비교하여, 도킹 중에 화합물 구조를 변환시킬 수는 없지만, 보다 빠른 계산을 수행할 수 있어서 가상 검색을 이용한 신약 후보 물질을 빠른 시간 안에 찾아낼 수 있는 장점이 있다.

DVSF를 이용한 가상 검색 방법은 적은 비용의 실험실 수준에서의 연구를 촉진시킬 수 있는 환경을 제공하여 생명공학 산업에서의 신약 개발을 활성화시키는데 기여할 수 있을 것이다. 향후 연구로는 좀 더 편리한 그래픽 방식의 사용자 인터페이스 구현과 자원 활용도, 작업상태를 실시간으로 모니터링 하는 기능을 추가할 것이며 가상 검색에 관련한 응용 연구를 통합할 수 있는 포털 시스템에 관한 연구를 수행할 예정이다.

참 고 문 헌

[1] Hugo Kubinyi, Gerd Folkers and Yvonne C. Martin, 3D QSAR in Drug Design, Kluwer/Escom, Vol.2, pp.355-380, 1998.

[2] G. M. Morris, D. S. Coodsell, R. S. Halliday, R. Huey, W. E. Hart, R. K. Belew, and A. J. Olson, "Automated Docking Using a Lamarckian Genetic Algorithm and Empirical Binding Free Energy Function," *Journal of Computational Chemistry*, Vol.19, pp.1639-1662, 1998.

[3] R. A. Lewis and A. R. Leach, *J. Comput.-Aided Mol. Design*, Vol.8, pp.467, 1994.

[4] Ian Foster, Carl Kesselman, *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann Publishers, 1999.

[5] AutoDock v3.0, http://www.scripps.edu/pub/olson-web/autodock/.

[6] 이상근, 황석찬, 최재영, "범용 그리드 데이타 시스템의 구현", 정보과학회 봄 학술발표논문집, 제 29권, 제 1호, pp.619-621, 2002.

[7] H. Johner, L. Brown, F.-S. Henner, W. Reis, J. Westman, *Understanding* LDAP, http://www.redbooks.ibm.com/, 1998.

[8] 임재훈, 황석찬, 최재영, "분산 환경을 위한 메타정보 시스템의 구축", 정보과학회 봄 학술발표논문집, 제29권, 제 1호, pp.622-624, 2002.

[9] ViewerLite, http://www.accelrys.com/viewer/.

[10] Foster, C. Kesselman, "Globus: A Metacomputing Infrastructure Toolkit," *International Journal of Supercomputer Applications*, Vol. 11, No. 2, pp.115-128, 1997.

[11] J. Frey, T. Tannenbaum, I. Foster, M. Livny, and S. Tuecke, "Condr-G,: A Computation Mana-gement Agent for Multi-Institutional Grids," *Proceedings of the Tenth IEEE Symposium on High Performance Distributed Computing* (HPDC10), 2001.

[12] R. Buyya, D. Abramson, J. Giddy, "Nimrod/G: An Architecture for a Resource Management and Scheduling System in a Global Computational Grid," *The 4th International Conference on High Performance Computing in Asia-Pacific Region*, May 2000.

[13] Entropia, Fight AIDS at Home Project: A joint effort of Entropia and Scripps Research Institute,

http://www.fightAIDSatHome.org/.
[14] R. Buyya, K. Branson, J. Giddy, and D. Abramson, "The Virtual Laboratory: A Toolset to Enable Distributed Molecular Modeling for Drug Design on the World Wide Grid," *The Journal of Concurrency and Computation: Practice and Experience*, Wiley Press, 2002.
[15] B. Shoichet, D. Dodian, and I. Kuntz, "Molecular Docking Using Shape Descriptors," *Journal of Computational chemistry*, Vol. 13, No. 3, pp.380-397, 1992.

황 석 찬
1998 숭실대학교 컴퓨터학과(석사). 1998 ~ 현재 숭실대학교 컴퓨터학과 박사과정. 관심분야는 분산/병렬 컴퓨팅, 인터넷 컴퓨팅, 시스템 소프트웨어

최 종 선
2000 숭실대학교 컴퓨터학부(학사). 2002 숭실대학교 컴퓨터학과(석사). 관심분야는 병렬컴퓨팅, 그리드 기반 응용연구

이 상 근
2001 숭실대학교 컴퓨터학과(학사). 2001 ~ 현재 숭실대학교 컴퓨터학과(석사과정). 관심분야는 그리드, 초고속 컴퓨팅(HPC), 전자상거래

임 재 훈
2001 숭실대학교 컴퓨터학과(학사). 2001 ~ 현재 숭실대학교 컴퓨터학과(석사과정). 관심분야는 그리드, CBD, XML, 데이타베이스

최 재 영
1984 서울대학교 제어계측공학과(학사) 1986 미국 남가주대학교 전기공학과(석사). 1991 미국 코넬대학교 전기공학과(박사). 1992.1 ~ 1994.2 미국 국립 오크리지연구소 연구원. 1994.3 ~ 1995.2 미국 테네시 주립대학교 연구교수. 2001.8 ~ 2002.8 미국 국립 수퍼컴퓨팅 응용센터(NCSA) 초빙연구원 1995.3 ~ 현재 숭실대학교 컴퓨터학부 부교수. 관심분야는 초고속 컴퓨팅(HPC), 병렬/분산처리, 시스템 소프트웨어

노 경 태
1978 연세대학교 화학과 졸업. 1980 한국과학기술원 화학과(석사). 1983 한국과학기술원 화학과(박사). 1983 ~ 현재 숭실대학교 화학과/생명정보학과 교수 1997 ~ 현재 숭실대학교 분자설계연구센터 소장. 2002 ~ 현재 사단법인 분자설계기술혁신센터 소장. 관심분야는 컴퓨터를 이용한 신약개발, 생물정보 관련 S/W 개발

이 지 수
1985 서울대학교 물리학과(학사). 1986 University of Pittsburgh 물리학과(석사) 1992 Boston University, 물리학과(박사) 1992 ~ 1993 HLRZ-KFA(German National Supercomputing Center), 연구원. 1993 ~ 1997 City College of New York, 연구원. 1997~1998 서울대학교, 초빙조교수. 1998~2001 고려대학교, 연구조교수. 2001 ~ 현재 한국과학기술정보연구원 슈퍼컴퓨팅 연구실장. 관심분야는 초고속 컴퓨팅(HPC), 전산물리