GOVP1200511507

M1-0224-01-0002

국가유전체정보 DB 구축 및 기반기술개발 사업

IT 기반 바이오인포매틱스 인프라구축 및 응용연구

Infrastructure Establishment for IT based Bioinformatics and Collaborative Research for its Application

한국과학기술정보연구원

과학기술부

제 출 문

과학기술부 장관 귀하

본 보고서를 "국가유전체정보 DB 구축 및 기반기술개발 사업"과제 (세부과제 "IT기반 바이오인포매틱스 인프라구축 및 응용연구") 의 보고서로 제출합니다.

2004. 6. 17

주관연구기관명 : 한국과학기술정보연구원

주관연구책임자 : 홍 순 찬

연 구 원 : 박형선, 이상주, 이식, 김진숙, 안부영, 오충식, 안설아, 안인성, 안성수, 김세훈, 황미넝, 손강렬, 김재성, 김지현, 박종미, 유석종, 조용성, 김형진, 김태환, 권민경, 윤단규, 허인애

보고서 초록

과제관리번호	M1-0224-01-0002	해당단계 연구기간	2002.12.1 - 2004.6.30	단계 구분	(1단계) / (총단계)
연구사업명	중 사 업 명	국책연구개발사업			
	세부 사업명	국가유전체정보 DB 구축 및 기반기술개발 사업			
연구과제명	중 과 제 명	국가유전체정보 DB 구축 및 기반기술개발			
	세부(단위)과제명	IT 기반 바이오인포매틱스 인프라구축 및 응용연구			
연구책임자	홍 순 찬	해당단계 참여연구원수	총 : 23 명 내부 : 14 명 외부 : 9 명	해당단계 연구비	정부: 천원 기업: 천원 계: 천원
연구기관명 및 소속부서명	한국과학기술정보연구원		참여기업명		
국제공동연구	상대국명 :		상대국연구기관명 :		
위 탁 연 구	연구기관명 :		연구책임자 :		

요약(연구결과를 중심으로 개조식 500자이내) | 보고서 면수 287

유전체 연구의 진전은 BT와 IT의 융합으로 가능했으며, 이러한 융합화는 포스트 게놈 시대에 더욱 가속화될 전망이다. 바이오인포매틱스는 유전정보의 체계적 해석과 정보화 및 컴퓨터 시뮬레이션을 통한 효율성 증대 및 부가가치 창출을 목표로 한다. 이를 위해서 연구결과로 얻어진 유전체 정보를 DB화할 필요가 있으며, 이러한 일은 국내의 IT 인프라와 인력들을 활용하여 수행할 수 있다. 본 연구과제에서는 수행한 연구내용은 다음과 같다.

- ○ 국가 유전체정보센터 통합홈페이지 구축
- ○ 국내외 생물정보 DB 유지보수 및 신규 구축
- ○ 클러스터시스템기반 생물정보분석시스템의 고도화
- ○ 생물정보검색 시스템(Bio-KRISTAL) 개발
- ○ 3차원 비교가시화 소프트웨어 개발
- ○ 유전체정보센터 인프라 고도화

본 연구과제에서는 기존 DB의 인터페이스를 보완하고 REBASE, dbSNP, Bind, DIP를 비롯한 신규 생물정보 DB를 구축하였으며, 생물정보검색시스템(Bio-KRISTAL)을 개발하여 유전정보 검색에 활용하였다. 또한 클러스터시스템기반 생물정보분석시스템의 고도화를 실현하고자 국내 BT분야에서 많이 사용되는 Parallel BLAST, ClustalW, InterProScan 등을 신규로 구축하여 서비스를 제공하고 있다.

색 인 어 (각 5개 이상)	한 글	생물정보학, 정보기술, 인프라, 정보검색시스템, 유전자, 단백질
	영 어	Bioinformatics, Information Technology, Information Retrieval System, Gene, Protein, genomic Database, Protein database

요 약 문

Ⅰ. 제 목

IT 기반 바이오인포매틱스 인프라구축 및 응용연구

Ⅱ. 연구개발의 목적 및 필요성

- 대용량 유전체 정보저장 및 관리, 유통서비스를 위한 기반기술 확보 및 IT 기반 바이오인포매틱스 국가 인프라 구축
- 유전체 연구의 진전은 BT와 IT의 융합화로 가능했으며, 이러한 융합화는 포스트 게놈 시대를 맞아 더욱 가속화될 전망임.
- 국내에서도 유전체 연구가 활성화되고 21C 프론티어사업 등을 통한 정부 투자가 늘어남에 따라 유전체 연구결과를 체계적이고 효율적으로 축적하고 활용할 필요성 있음.
- 국내에 풍부한 IT 인력을 BT와 연계 및 활용하여 선진국과의 격차를 최소화할 필요성 있음.
- 국내외 유전체 연구결과를 통합한 DB를 구축하여 공동 활용하고, 국내외 유전체정보 연구 기관간의 유기적인 네트워크를 구축하여 생명공학 연구의 시너지 효과를 도모할 필요성 부각

Ⅲ. 연구개발의 내용 및 범위

- 국가 유전체정보센터 통합홈페이지 구축
- 국내외 생물정보 DB 유지보수 및 신규 구축
- 클러스터시스템기반 생물정보분석시스템의 고도화
- 생물정보검색 시스템(Bio-KRISTAL) 개발
- 3차원 비교가시화 소프트웨어 개발
- 유전체정보센터 인프라 고도화

Ⅳ. 연구개발결과

- 국가유전체정보센터 통합 홈페이지 구축
  - 생물정보 데이터베이스 검색 및 유전체 분석 서비스 체제 구축
- 신규 DB 구축(REBASE, dbSNP, Bind, DIP 등) 및 기존 DB의 GUI 강화
- 클러스터시스템 기반 생물정보분석시스템 고도화
  - 국내 BT 분야에서 사용빈도가 높은 분석도구의 신규 구축
    . Parallel BLAST, ClustalW, InterProScan, FASTA
  - 클러스터시스템기반 생물정보분석시스템의 고속화 서비스 실시
  - 생물정보 자동 마이닝 웹 서비스 구축
  - 클러스터시스템 관리시스템 구축
- 생물정보검색시스템(Bio-KRISTAL) 개발

- 단백질 아미노산 서열 검색기법 개발
- 색인기반 단백질 Superfamily 분류시스템 개발
- 단백질서열 n-gram 빈도 데이터베이스 구축

• 3차원 비교가시화 소프트웨어 개발

• 대용량 유전체자원 인프라 고도화

V. 연구개발결과의 활용계획

• BT와 IT의 융합으로 유전체, 단백질체 관련 국내 BT 산업 및 새로운 drug target을 찾는 제약업계와 생명공학기업의 국제 경쟁력 증대

• 인간, 동·식물, 미생물의 유전체 종합정보의 통합 DB 구축 및 공동활용 기반 구축으로 고부가가치형 생물정보의 인프라 서비스 가능

• 생명체속의 유전자 네트워크 및 대사경로를 분석함으로써 컴퓨터상에서 세포생리 및 병리현상을 모형화 할 수 있어, 실험과 병행된 연구를 진행하는 physiome 시대로의 패러다임 변화에 대비 가능

• 바이오인포매틱스 연구를 수행하기 위한 분석도구 개발 및 이의 보급을 통한 IT 분야의 기술집적도 향상 및 고성능 컴퓨터 활용 기술 개발의 기반 마련

• 연구과제의 수행 및 개발과정을 통하여 바이오인포매틱스 분야의 고급인력을 양성하여 산·학·연 전반으로 바이오인포매틱스 인식 확산 및 발전에 기여

SUMMARY

Ⅰ. Title

Infrastructure Establishment for IT based Bioinformatics and Collaborative Research for Its Application

Ⅱ. Objective of the study and its importance

- Securing the essential technology for archiving, managing, and distributing of huge genome data, and constructing IT-based national bioinformatics infrastructure
- Fusion of BT and IT makes the advance in genome research possible and will speed up the days of post-genome
- Activation of domestic genome research and increasement of government's investment (e.g. 21C frontier project) leeds to the necessity of systematic and effective archiving and/or application of genome research
- It is necessary to minimize the gap between Korea and advanced countries through liaison of abundant IT manpower with BT
- It is necessary to bring the synergy effect through [1)]constructing and co-utilizing of unified database of domestic/foreign genomic research and [2)]building the close network among domestic/foreign genome research institution.

Ⅲ. Content and scope of the study

- Constructing the unified homepage of NGIC
- Constructing and Maintaining the domestic/international bioinformatics databases
- Enhancement of cluster-based bioinformatics analysis system
- Development of bioinformatics search system (Bio-KRISTAL)
- Development of 3-D visualization software
- Advancement of NGIC infrastructure

Ⅳ. Result of the study

- Constructing the homepage of NGIC
  - Constructing the bioinformatics database search and genome analysis service
- Constructing new databases (REBASE, dbSNP, Bind, DIP, etc) and improvement of GUI
- Enhancement of cluster-based bioinformatics analysis system
  - Constructing new analysis tools most requested from domestic biotechnology community
    . Parallel BLAST, ClustalW, InterProScan, and FASTA
  - Enhancement of cluster-based bioinformatics analysis service
  - Building the automatics mining web services for

bioinformatics

- Constructing the management system for cluster system

• Development of bioinformatics information retrieval system (Bio-KRISTAL)

- Developing the indexing method of amino acid sequence

- Developing index-based protein superfamily classification system

- Constructing the n-gram frequency database of protein

• Development of 3-D visualization software

• Advancement of infrastructure for massive genomic data

V. Application plan of the result

• Fusion of BT and IT increases the competitiveness of pharmaceutical and bio-technological companies who are interested in genomics, proteomics, and new drug target.

• Construction and co-utilization of unified genetic database of human, animal, plant and microorganism make the service of high value-added bioinformatics infrastructure possible

• Analyzing the genetic network and metabolic pathway in living cells causes the *in silico* research for cell physiology and pathology. It is possible to prepare for the paradigm shift to physiome era

• Preparing the technology for high-performance application and degree of integration at IT area through the development of

analyzing tools for bioinformatics research and its spread

Through this development project high talented experts in bioinformatics can be reared. And it will be contributed to the diffusion of bioinformatics to industry, academia, and research institution, and to the advancement of bioinformatics.

목 차

제 1 장 연구개발과제의 개요

1. 제 목

IT 기반 바이오인포매틱스 인프라구축 및 응용연구

2. 연구개발의 목적 및 필요성

2.1. 목적

- IT 기반 바이오인포매틱스 인프라 구축 및 서비스
- 대용량 유전체정보 저장, 관리, 유통 서비스 제공을 위한 기반기술 확보
- 국가차원의 유전체정보 유통시스템 보유 및 관련 연구 기반환경 구축

2.2. 필요성

- 유전체 연구의 진전은 BT와 IT의 융합화로 가능했으며, 이러한 융합화는 포스트 게놈 시대에 더욱 가속화될 전망이다.
- 국제 컨소시엄으로 추진된 Human Genome Project의 완성으로 인하여 국내외적으로 유전체 연구에 대한 관심과 투자 증대에 의해 생산된 유전체 모의 분석 및 처리에 대한 수요가 급증하고 있다.
- 유전자원으로부터 바이오제품을 개발하는 생명공학 과정에서 바이오인포매틱스의 역할은 유전정보의 체계적 해석과 정보화 및 컴퓨터 시뮬레이션을 통한 효율성 증대 및 부가가치 창출에 있다.

- 유전자 및 유전체 서열정보, 발현정보, 상호작용정보 등을 DB화하고 체계적으로 분석하여 유전자의 기능과 세포내 유전자 네트워크 경로 등을 파악한 시스템을 개발하는 것이 중요한 분야로 부상하고 있다.

- 국내에서도 유전체 연구가 활성화되고 21C 프론티어 사업 등을 통한 정부 투자가 늘어남에 따라 유전체 연구를 체계적이고 효율적으로 추진하기 위해 국가유전체정보센터가 지정되었으나 국내의 연구는 초기단계이므로 진행을 서두르지 않을 경우 선진국과의 기술격차가 심화될 우려가 있다.

- 국내에 풍부한 IT 인력을 BT에 연계하여 활용하면 선진국과의 격차를 최소화할 수 있다.

- 국내외 유전체 연구결과를 통합한 DB를 구축하여 공동 활용하고, 국내 유전체 연구기관간의 유기적인 네트워크를 구축하여 생명공학 연구의 시너지 효과를 도모할 필요성이 있다.

3. 연구개발의 내용 및 범위

목표	내 용	연구범위
생물정보검색시스템 개발	생물정보검색 기반시스템 구축	· 생물정보검색기반시스템 구축 · 생물정보 색인시스템 개발
	국내외 DB 유지보수 및 신규구축	· 기존 DB 업데이트 및 인터페이스 개선 · 검색속도 : 5초 이내 · 사용자 요구 분석을 통한 신규 DB 구축
	통합서비스시스템 구축에 관한 연구	· NCBI의 Entrez 등과 같은 통합서비스 시스템의 구조 분석을 및 서비스 동향 파악 · Annotation 시스템 분석 · 생명공학관련 문헌정보서비스시스템 설계 · FTP 서비스 시스템 확대 설계 · 통합DB 검색시스템 설계
생물분자 3차원 비 교 가 시 화 S/W 개발	3차원 가시화 S/W 개발	· 2개 이상의 단백질 분자에 대한 구조를 비교 분석할 수 있는 비교가시화 도구 개발 · HTTP 프로토콜 기반의 데이터베이스 연동 기능 제공
IT 기반 바이오 인포매틱스 응용연구	색인기반 단백질서열검색시스템 개발	· PIR-NREF 단백질 데이터베이스를 대상으로 하는 단백질 서열 검색 시스템 개발 · BLAST보다 빠른 검색을 지원하는 서열 검색시스템 개발
	서열분류시스템 개발	· 단백질 분류 시스템 · IT 분야의 문서분류시스템을 생물정보 데이터로 확장함으로써 BLAST 기반의 분류 시스템을 서열검색기반의 단백질 기능 분류 시스템으로 전환하여 고속 분류시스템 구현
홈페이지 구축	국가유전체 통합 홈페이지 구축	· 서비스 자원의 one-stop 활용체제 · 사용자 통계 분석
유전체정보센터 인프라 고도화		· 컴퓨팅자원증설을 통한 기반인프라고도화
주요 생물정보 DB구축 및 유지보수	신규 데이터베이스 구축	· SRS에서 지원하지 않는 DB 중 선별하여 구축 (dbSNP, Unigene, PDB, CATH, SCOP, Ensembl, DIP, BIND)
	데이터 최신성 유지	· 데이터 업데이트 전략 수립 및 자동화 프로그램 개발 확대
	기존 데이터베이스의 GUI 및 기능 강화	· 기존 데이터베이스의 사용자 인터페이스 개선 및 기능 업그레이드
	한글화에 의한 사용 환경 개선	· 일반 사용자도 이용이 용이하도록 한글화 혹은 한글/영문 이원화 추진
클러스터시스템 기반 생물정보 분석시스템의 고도화	국내 BT분야에서 사용빈도 가 높은 분석도구의 신규 구축	· Blast, FASTA, ClustalW, InterProScan 등 구축
	클러스터 시스템 기반 생물 정보분석시스템의 고속화 서비스	· 시스템 병렬화 및 BLAST 프로그램 성능 향상
	생물정보 자동 마이닝 웹서비스	· 주요 사용자를 위한 자동 배치 처리 시스템 개발/서비스(Parasol, PBS, LSF 적용)
	클러스터시스템 관리시스템 구축	· 사용자별 작업 관리 환경 구축 및 시스템 이용환경 모니터링 체제 구축

4. 연구개발결과

내 용	결 과
생물정보검색시스템 개발	- 대용량의 데이터에 대한 서비스를 지원할 수 있는 생물정보 주석검색 시스템 개발 - Hot Search/Cold Search 개념을 도입한 초고속 검색 시스템 개발
생물정보 색인시스템	- 단백질 아미노산 서열 색인추출 시스템 개발 및 생물정보검색기반시스템에 이를 적용
서열분류시스템 개발	- iProClass의 기능성 단백질 분류인 superfamily 분류체계에 따라 단백질 서열에 대한 기능 분류 서비스인 ProFaC (protein family classification) 단백질 분류 시스템 구축
3차원 가시화 S/W 개발	- 2개 이상의 단백질 분자에 대한 구조를 비교 분석할 수 있는 비교가시화 도구 개발 - HTTP 프로토콜 기반의 데이터베이스 연동 기능 제공
컴퓨팅 자원 증설을 통한 기반 인프라 고도화	- 네트워크 성능향상 및 파일서버 증설로 안정된 인프라 환경 구축 - 시스템 이용정책을 수립하여 체계적인 지원 환경 조성함
FTP 서비스 시스템 확대 설계	- 스토리지의 도입으로 대용량의 데이터 저장하고 또한 동시 이용자수를 증가시켰음 - Anonymous 사용자에게 서비스를 오픈하여 일반 사용자도 이용할 수 있도록 함
통합 검색 시스템 구축	- 국가유전체정보센터 통합 홈페이지 - 해외 유명 홈페이지 (e.g, EMBL, NCBI, RSCB, etc)의 분석을 통해 주요 생물정보 콘텐츠들의 국내 서비스 체제를 구축하였으며 분산된 생물정보 자원을 통합함으로써 국내외 생물정보의 one-stop 활용체제 마련 - 데이터베이스 및 콘텐츠들의 통계 및 분석 시스템의 개발을 통한 양질의 서비스 제공 기반 마련
컴퓨팅 자원 증설을 통한 기반 인프라 고도화	- 네트워크 성능향상 및 파일서버 증설로 안정된 인프라 환경 구축 - 시스템 이용정책을 수립하여 체계적인 지원 환경 조성함 - FTP 서비스 시스템 확대 설계 - 스토리지의 도입으로 대용량의 데이터 저장하고 또한 동시 이용자수를 증가시켰음 - Anonymous 사용자에게 서비스를 오픈하여 일반 사용자도 이용할 수 있도록 함

내 용	결 과
신규 데이터베이스 구축	- dbSNP : 2004년 3월 18일자(Build 120)자료를 이용하여 NCBI의 MSSQL로 제작된 데이터베이스를 MySQL로 재구축함. 데이터 확인 작업 완료 - REBASE : 최신 자료인 2004년 4월 27일자(405 build) 데이터베이스를 이용하여 "enzyme", "reference", "commercial", "comm_enz" 등 4개 테이블 생성 및 구축 완료 - DIP 최신버전으로 DB 구축 (6월 완료) - BIND 최신 버전으로 DB 구축 (6월 완료) - CATH 최신버전으로 DB 구축 (6월 완료)
데이터 최신성 유지	- Ensemble, PDB, PIR, Swiss-Prot, GenBank 데이터를 최신 버전으로 상시 업데이트 - SCOP은 정식 미러 사이트로 등록하고 1.65버전 서비스 개시 - FTP 사이트의 정보 최신성 유지
기존 데이터베이스의 GUI 및 기능 강화	- 직관적으로 사용하기 쉽게 홈페이지 디자인을 개선 - 분석결과를 원 클릭으로 ftp 할 수 있게 처리
한글화에 의한 사용 환경 개선	- HELP 버튼을 활용하여 한글 설명 추가 - 국영 혼용으로 사용자 편의 도모
국내 BT분야에서 사용빈도가 높은 분석도구의 신규 구축	- FASTA, ClustalW, InterProScan 프로그램을 설치하고 국내 사용자의 환경에 맞는 웹 인터페이스 구축
클러스터 시스템 기반 생물정보분석시스템의 고속화 서비스	- BLAST 프로그램의 최적화로 10~33%의 성능 향상 - MPI를 이용한 프로그램 병렬화로 사용하는 노드 수에 비례하는 성능 향상
생물정보 자동 마이닝 웹서비스	- SRS의 배치처리용 스크립트 개발로 원하는 클러스터로 작업을 요청할 수 있도록 제작 - 클러스터 컴퓨터를 제작하여 다량의 작업요청을 처리 할 수 있도록 함 - mpiBLAST를 이용 최적의 실행 환경 설정 (10초미만) - Web Services를 이용하여, 차세대 인터넷 작업요청에 대한 서비스 기능 제공.
클러스터시스템 관리시스템 구축	- PHP를 이용하여 웹상에서 클러스터시스템의 작업 상황을 시간과 장소에 구애받지 않고 확인 가능 - 실행내용을 기간별로 검색하여 사용자가 원하는 통계처리 기능 제공 - 막대그래프를 이용하여 작업 상황을 표시함

5. 연구개발결과의 활용계획

o BT와 IT의 융합으로 유전체, 단백질체 관련 국내 BT 산업 및 새로운 drug target을 찾는 제약업계와 생명공학기업의 국제 경쟁력을 증대시킨다.

o 인간, 동·식물, 미생물의 유전체 종합정보의 통합 DB 구축 및 공동 활용 기반 구축으로 고부가가치형 생물정보의 인프라 서비스가 가능하다.

o 생명체속의 유전자 네트워크 및 대사경로를 분석함으로써 컴퓨터상에서 세포 생리 및 병리현상을 모형화할 수 있어, 실험과 병행된 연구를 진행하는 physiome 시대로의 패러다임 변화에 대비할 수 있다.

o 바이오인포매틱스를 수행하기 위하여 IT 분야에서 핵심기반 S/W개발을 하도록 유도하며 이의 보급을 통한 IT 분야의 기술집적도 향상 및 고성능 컴퓨터 활용 기술개발을 가져올 수 있다.

o 연구과제 수행 및 개발과정을 통하여 바이오인포매틱스 분야의 고급인력을 양성하여 산·학·연 전반으로 생물정보학의 저변 확대를 가져올 수 있다.

제 2 장 국내외 기술개발 현황

1. 세계 기술개발 현황

가. 미국

○ 세계 생명공학 시장을 주도하고 있으며 생물정보학 분야에서도 가장 앞서나가고 있다.

○ 국립보건원 산하의 NCBI (National Center for Biological Information)가 공공기관으로서는 대표적인 생물정보 연구기관이며 인간 및 동·식물, 미생물을 망라한 각종 생물정보를 제공하고 있다. 이와 더불어 스탠포드 대학과 NCGR (National Center for Genome Research)이 주축이 된 TAIR(The Arabidopsis Information Resource)에서는 식물의 데이터를 중심으로 한 연구 활동과 서비스를 하고 있으며, TIGR(The Institute of Genome Research)에서도 각종 미생물 및 동·식물의 Gene Indexes를 제공하고 있다.

○ IBM, SUN, Motorola, HP 등의 거대 IT기업과 Affymatrix, Celera Genomics, Double Twist, Incyte, Millenium, Rosetta등의 많은 벤쳐기업이 주축이 되어 생물정보학 분야에 투자를 급속히 증대시키고 있다.

나. 유럽연합

○ EU은 EBI (European Bioinformatics Institute)와 같은 공공기관을 설립하고 공동투자를 하고 있다.

○ EBI를 비롯한 EMBL의 생물정보학 관련 연구 개발결과는 주로 생물정보학 벤쳐기업인 Lion을 통해 사업화되고 있다.

○ EBI에서 2001년부터 시작하여 3년간 수행될 DESPRAD(Development and Establishing Standards and Prototype Repository for DNA-array Data)프로젝트에는 4천8백만 유로(약 60 억원)가 투자되며, 데이터의 국제표준화를 확립하고 데이터 검색 및 분석 소프트웨어를 개발할 예정이다.

○ 영국의 Sanger Center에서는 데이터베이스 제공이외에도 연구자들이 생성한 데이터를 분석할 수 있는 프로그램들도 같이 개발하여 제공하고 있다.

다. 일본

○ 국립유전학연구소 산하의 CIB(Center for Information Biology)라는 기관이 있어 미국 NCBI의 GenBank같은 DDBJ를 서비스하고 있다.

○ 문부성, 상공부, 과학청이 중심이 되어 지원 및 투자를 하고 있으며 실용화를 위해 대기업의 투자가 이루어지고 있다.

○ 정부가 주도하는 밀레니엄 프로젝트(2000~2005)와 민간이 주도하는 Helix프로젝트(1996~2002) 두 개의 거대한 프로젝트가 수행 중이다.

○ 일본 정부는 생물정보학 분야에서 2001년까지 SNP (Single Nucleotide Polymorphism) 데이터베이스 완성, 2003년까지 네트워크화, 2004년까지 통합 데이터베이스를 구축한다는 계획을 가지고 있다.

2. 국내 기술개발 현황

○ 국내 생물정보관련 데이터베이스는 현재 포항공대 생물학연구정보센터 (BRIC)에서 유전자 데이터베이스와 서열 기반의 검색엔진을 운영중이고, 한국생명공학연구원 및 연세대 프로테옴연구센터를 비롯한 일부대학 등에서 해외 데이터베이스 미러링 서비스 등이 운영되고 있으나 해외에 비하여 데이터의 양이 극히 적으며, 몇몇 벤처회사에서 프로테옴 테이터베이스, 서열 분석프로그램들을 개발하고 있는데, 규모나 통합성 측면에서 세계 기술과 격차를 보이고 있다.

○ 단백질 DB 구축의 경우, 유전자 DB에 비해 더욱 낙후되어 있다.

○ 국내에서도 유전체 연구가 활성화되고 21C 프론티어 사업 등을 통한 정부 투자가 늘어남에 따라 유전체 연구를 체계적이고 효율적으로 추진하기 위해 국가유전체정보센터가 지정되어 운영중이다.

○ 국가차원에서 산·학·연의 연구결과를 완벽하게 공유할 수 있는 체제를 갖추고 있지 못한 실정이다.

○ 소수의 벤처기업과 학교에서 일부 소프트웨어의 연구 개발이 진행되고 있을 뿐 대부분의 소프트웨어 관련 원천기술을 해외에 의존하고 있음. 전반적으로 우리나라의 강점인 IT 역량이 BT 산업에 적절히 활용되지 못하고 있다.

3. 국내외 표준화 현황(또는 향후 기술 발전 추세)

○ 미국은 NCBI를 중심으로, EU는 EBI 주도로 국제 표준을 만들려고 노력하고 있으나, 각각 기관의 이익이 걸려 있기 때문에 표준화에 어려움이 있고, 이에 더하여 기업들은 자체적인 데이터베이스를 구축하는 경향이 뚜렷하다.

○ 국내에는 표준화의 노력이 전혀 없는 실정으로 국가 기관 주도로 시급히 표준을 정할 필요성이 있다고 사료된다.

제 3 장 연구개발 수행내용 및 결과

제 1 절 국가 유전체 정보센터 통합 홈페이지 구축

1. 연구배경

가. 연구의 필요성 및 목적

생물과 생명의 특징을 결정짓는 유전정보를 완전히 밝혀, 그 특성을 분석/규명하고자 하는 인간게놈프로젝트 (HGP: Human Genome Project)의 결과, 전산처리가 가능한 각종 생물체의 유전체 염기서열에 대한 정보가 쏟아져 나오고 있다. 일반적으로 생명체의 유전정보는 수 기가바이트에 이르는 대용량 데이터이며 HGP이후 이 데이터의 양은 단 기간에 급격히 증가하고 있다. 바이오인포매틱스는 유전자 연구와 같은 대규모 생물학 연구로부터 얻어진 천문학적 분량의 데이터를 체계적으로 수집/정리하고, 분석/유통함으로써 연구자들의 효율적인 연구 및 정보 창출 활동에 결정적인 기여를 하고 있다.

인터넷 환경의 확산과 정보기술의 발달 등 정보환경의 변화와 더불어 바이오인포매틱스 관련 산업분야 역시 급격한 성장세에 있다. 국내에서도 국가적인 지원 하에 바이오인포매틱스 분야의 많은 연구들이 진행되고 있으며 점차 그 결과들이 성과를 보이고 있다. 이렇게 빠르게 증가하고 있는 바이오인포매틱스 관련 연구결과에 힘입어 대량의 생물정보들이 쏟아져 나오고 있어 이러한 정보들을 체계적으로 수집하고 신속히 분배/확산하는 것이 더욱 중요해지고 있는 추세이다. 그러나 대부분의 생물정보 서비스 시스템들이 유전 정보의 가공 및 응용에 대한 연구에 그 초점이 맞추어져 있는 반면 양질의 분석 서비스 제공 및 유전 정보의 효율적인 분배/확산을 위한 정보 유통에 관한 연구와 시스템 개발은 상대적으로 미비한 실정이다.

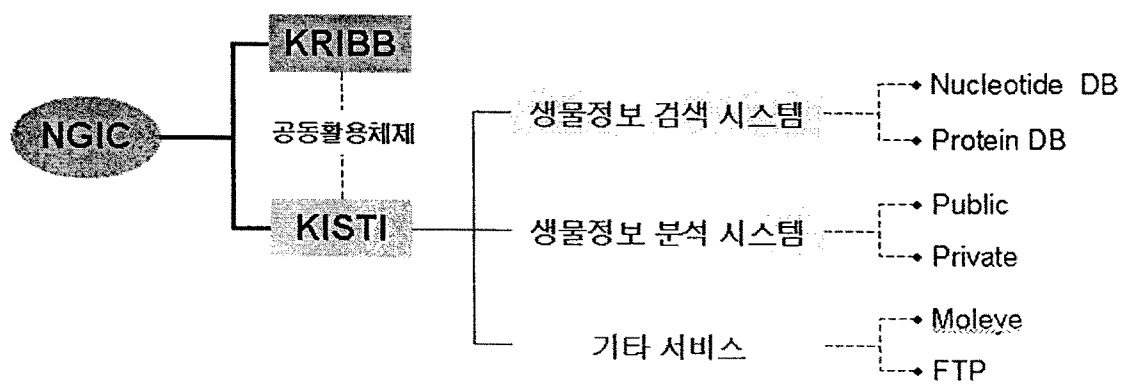
따라서 국내 유전체 연구기관간의 유기적인 연구 네트워크를 구축함으로써 범국가적으로 증대되는 다양한 분야의 생물학 연구 결과들을 체계적으로 수집/관리/유통함으로써 국내 연구자들의 효율적인 연구지원을 도모할 수 있는 통합적인 바이오인포매틱스 유통 서비스 체제의 구축이 요구된다. 또한 시간이 흐를수록 급격히 증가하는 생물학 데이터와 바이오인포매틱스 관련 리소스들의 효율적인 유통 및 정

보 제공 방법의 선진화를 통해 생물정보 유통의 다각화를 지속적으로 추진할 필요가 있다.

이러한 필요성에 기반하여 본 연구에서는 국내 유전체 관련 연구 주체간의 유기적 연구 네트워크를 구축하고 각 기관이 보유하고 있는 국내외 생물정보 데이터베이스, 분석 시스템 및 관련 리소스들을 공동으로 활용하기 위한 통합적인 바이오인포매틱스 포탈서비스 체제를 구축하고자 한다.

나. 연구 내용 및 추진체계

바이오인포매틱스에서 주로 다루는 유전체 관련 연구는 데이터베이스 분야와 분석도구의 개발 관련 등으로 나눌 수 있다. 생물데이터와 같이 대용량 데이터에 대한 서열 분석이나 서열정보의 검색과 같은 작업들을 수행하기 위해서는 높은 수준의 시스템 사양이 요구된다. 이러한 분석 작업을 효율적으로 지원하기 위해서는 사용자의 요구에 대한 시간적 지연을 최소화하여 실시간으로 분석 결과를 도출하는 실용성이 필수적이다. 본 연구에서는 본 기관(KISTI)에 기 구축된 고성능 생물정보 분석 서버를 활용한 생물정보 검색 및 분석 시스템의 서비스 체제를 구축하고 이를 국가유전체정보센터(National Genome Information Center) 홈페이지에 연계함으로써 유사 기관간의 관련 시스템의 통합적인 활용 체제를 구축한다. 이를 위해 본 연구에서는 CCBB(Center for Computational Biology and Bioinformatics)에서 기 구축한 생물정보 데이터베이스의 검색 인터페이스의 개선, KRISTAL 2000을 기반으로 하는 데이터베이스의 구축 및 유전체 분석시스템의 분석 인터페이스의 개선으로 나누어진다.

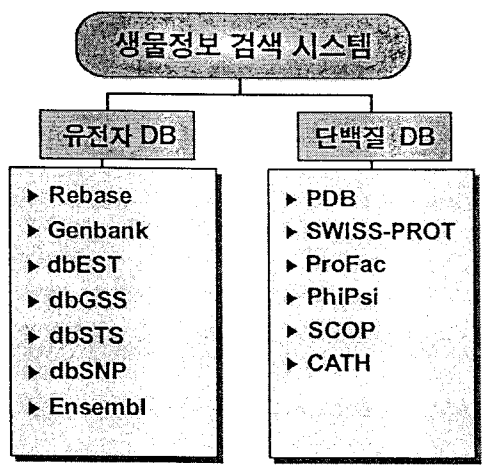


<figure 1-1> 추진체계

본 연구는 한국생명과학연구원(Korea Research Institute of Bioscience and Biotechnology)의 국가유전체정보센터와 공동으로 수행되었으며 연구의 추진체계는 <figure 1-1>과 같다.

2. 생물정보 데이터베이스 검색 서비스 체제 구축

본 연구를 통해 구축된 생물정보 데이터베이스 검색 서비스 체제는 <figure 1-2>와 같은 데이터베이스들로 구성된다. 생물정보 검색 시스템은 크게 유전자 DB와 단백질 DB로 구성되며 유전자 DB에는 Rebase, GenBank, dbEST, dbGSS, dbSTS, dbSNP, Ensembl 데이터 베이스의 검색 서비스가 제공된다. 단백질 DB에서는 PDB, SWISS-PROT, PIR, ProFac, PhiPsi, SCOP, CATH 데이터베이스의 검색서비스가 제공된다. 단백질 DB들 중 활용도가 높은 PDB, SWISS-PROT, PIR 데이터베이스는 KISTI에서 개발한 KRISTAL 2000 검색엔진을 이용한 데이터베이스 검색서비스를 제공함으로써 사용자의 검색에 소요되는 시간과 비용을 최소화하였다.



<figure 1-2> 생물정보 검색 시스템

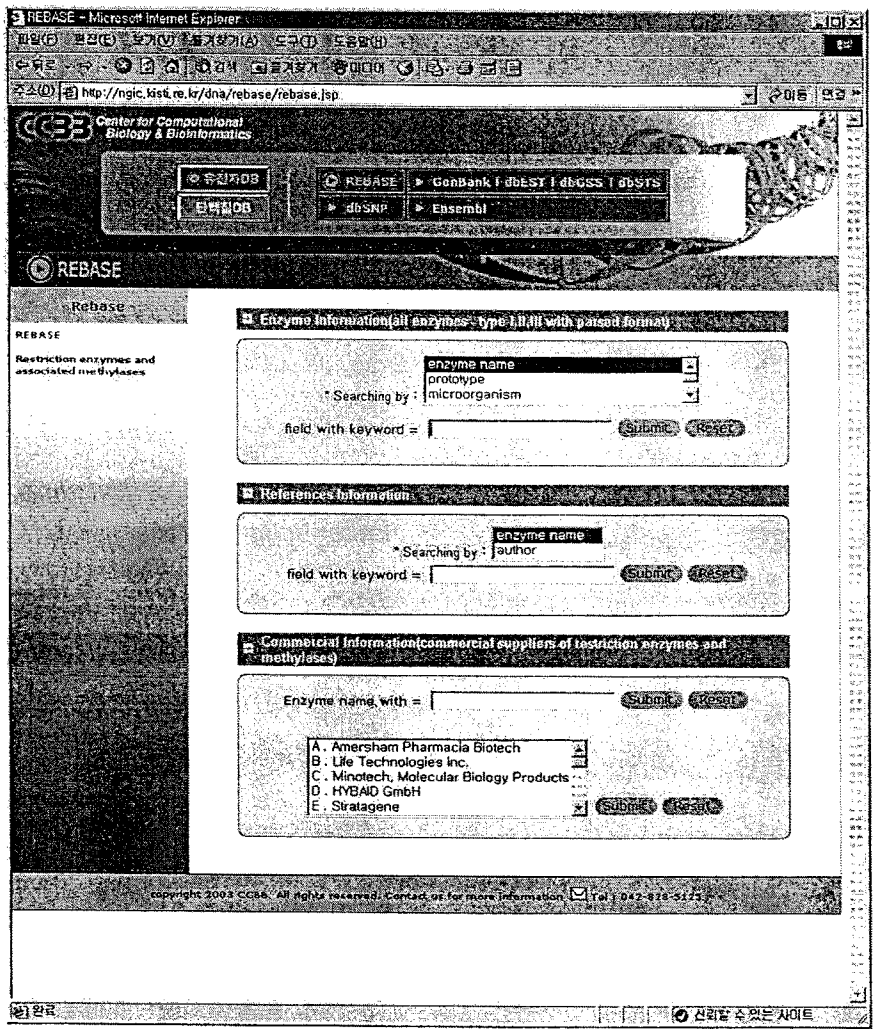
가. 유전자 데이터베이스

(1) Rebase

Rebase (Restriction Enzyme data BASE)는 제한효소와 이와 관련된 단백질에 대한 정보를 제공하는 데이터베이스이다. 또한 Rebase에서는 제한효소과 관련된 문

헌정보(reference), recognition site, methylation sensitivity 및 관련 상업적 정보를 제공한다. Rebase 데이터베이스 검색 프로그램은 기존의 PHP에서 Java/JSP로 변환함으로써 평균 20%의 검색 성능 향상을 보이고 있다.

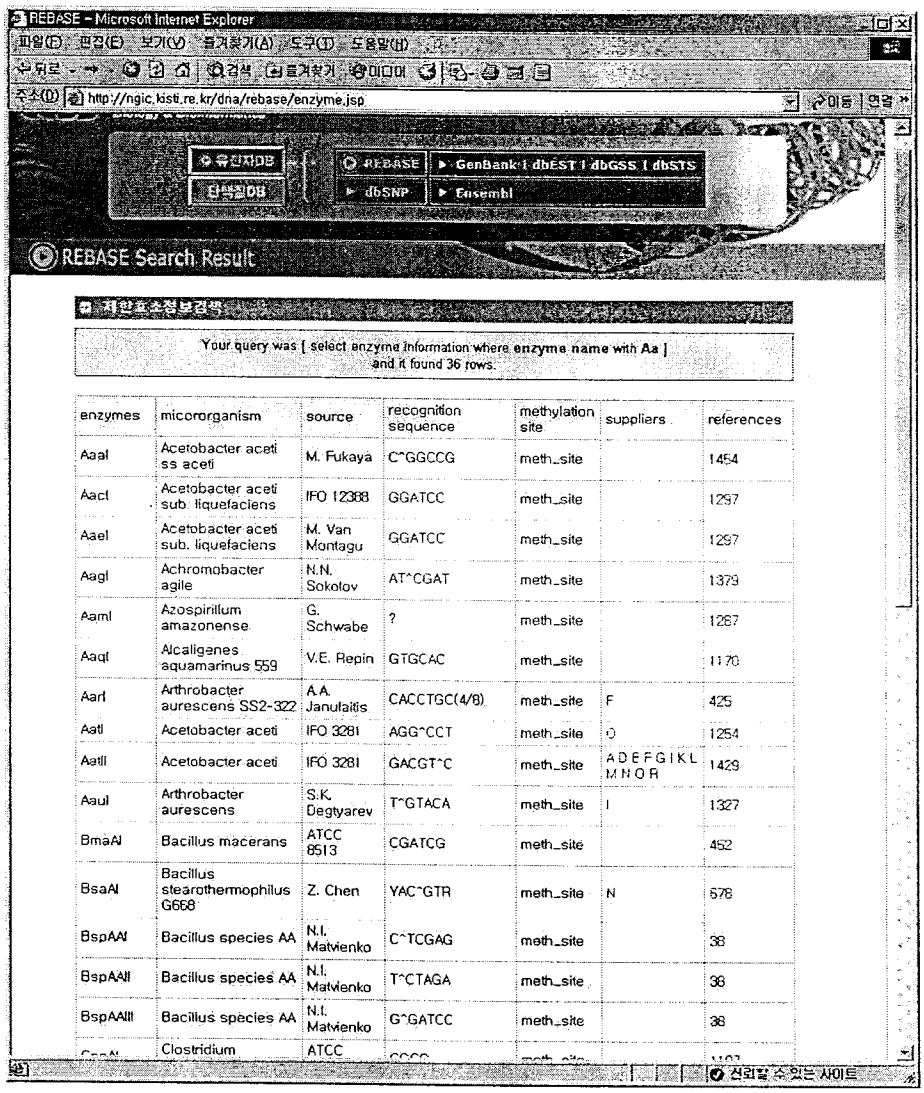
<figure 1-3>은 Rebase 데이터베이스의 검색 페이지를 보이고 있다. Rebase에서의 검색은 제한효소 정보 검색, 관련 참고 문헌 검색, 관련 상업적 정보 검색으로 구성되며 각 검색은 해당정보에 대해 keyword 검색이 가능하도록 하였다.



<figure 1-3> Rebase 검색 페이지

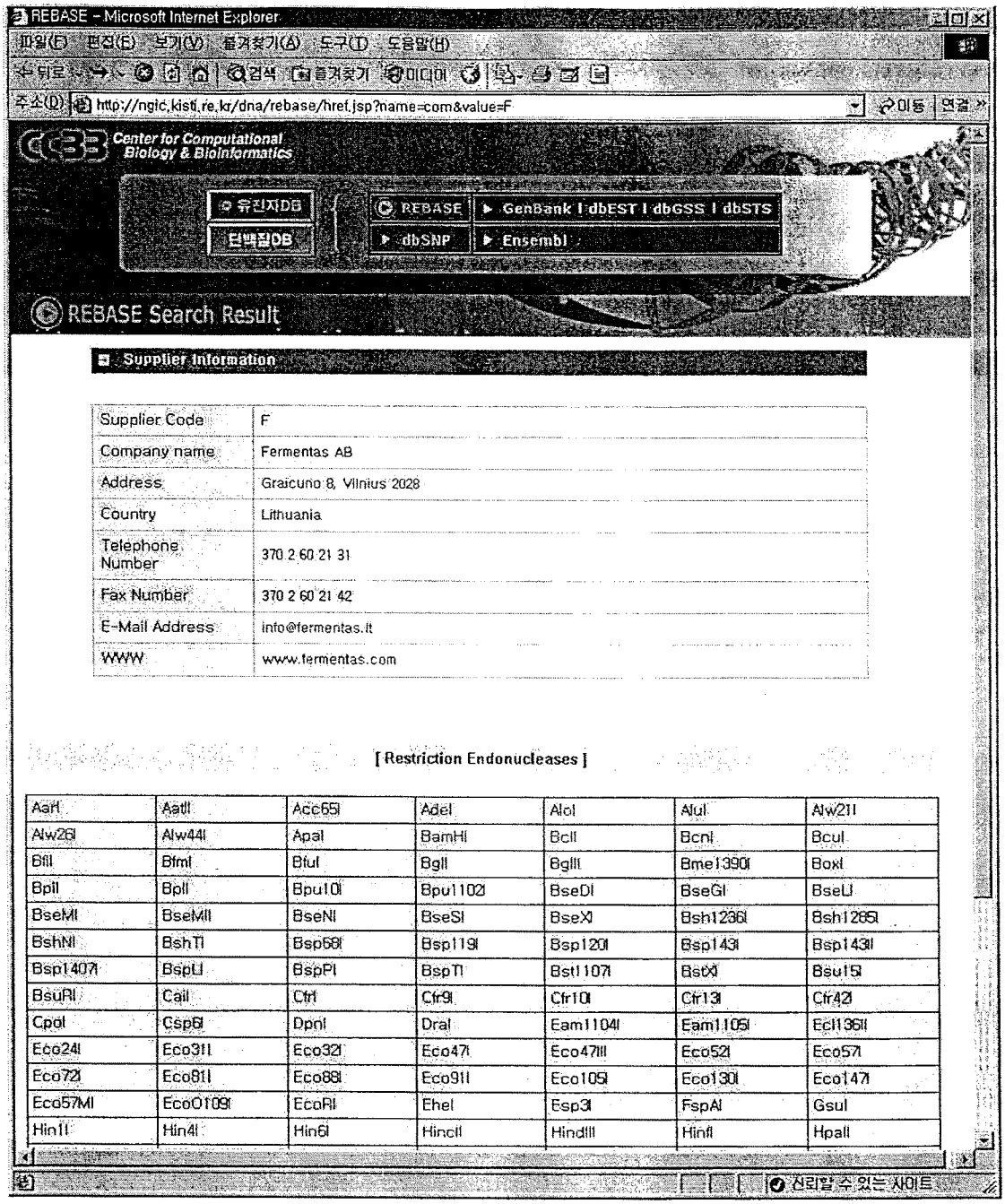
<figure 1-4>는 제한 효소 정보검색에서 효소이름 중 'Aa'를 가지는 제한 효소에 대한 keyword 검색 결과 화면을 보이고 있다. <figure 1-4>에서는 'Aa'라는 keyword를 이름으로 가지는 총 35개의 제한효소와 이와 관련된 recognition

sequence, methylation site, supplier, reference등의 정보를 보이고 있다. <figure 1-5>는 제한효소 AarI의 supplier F에 대한 정보를 보이고 있으며 <figure 1-6>은 F (Fermentas AB)가 제공하는 제한효소 Bsp119I에 대한 정보를 보여주고 있다. <fiugre 2-7>은 Bsp119I의 reference 정보를 보이고 있다.

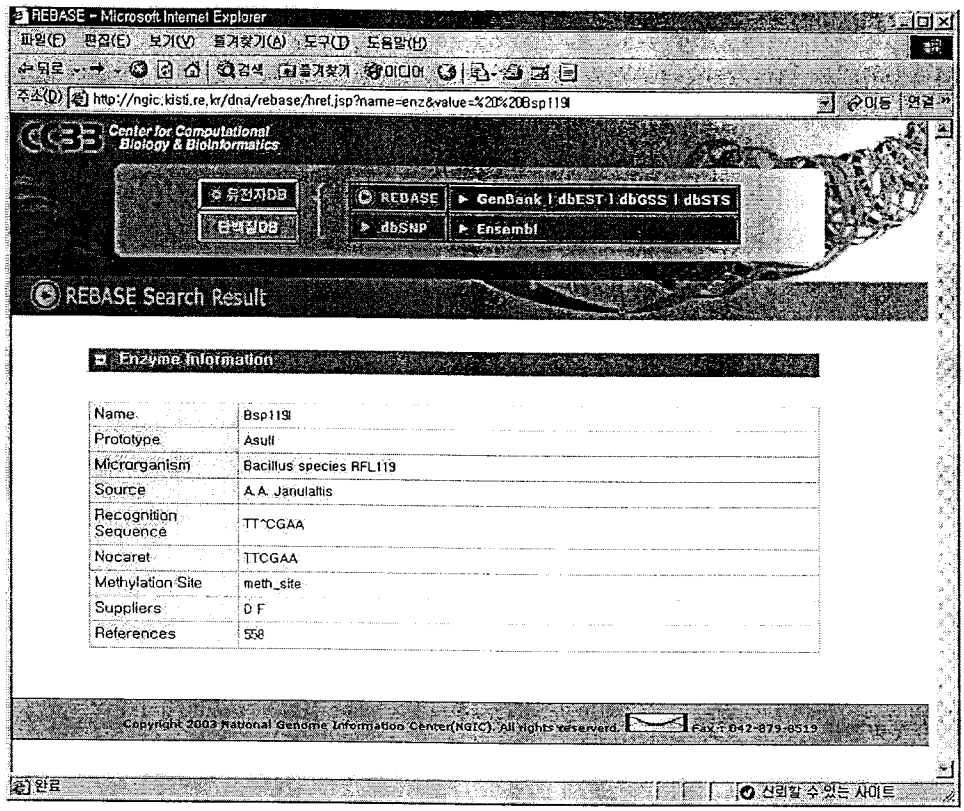


enzymes	micororganism	source	recognition sequence	methylation site	suppliers	references
AaaI	Acetobacter aceti ss aceti	M. Fukaya	C^GGCCG	meth_site		1454
AacI	Acetobacter aceti sub. liquefaciens	IFO 12388	GGATCC	meth_site		1297
AaeI	Acetobacter aceti sub. liquefaciens	M. Van Montagu	GGATCC	meth_site		1297
AagI	Achromobacter agile	N.N. Sokolov	AT^CGAT	meth_site		1379
AamI	Azospirillum amazonense	G. Schwabe	?	meth_site		1287
AaqI	Alcaligenes aquamarinus 559	V.E. Repin	GTGCAC	meth_site		1170
AarI	Arthrobacter aurescens SS2-322	A.A. Janulaitis	CACCTGC(4/8)	meth_site	F	425
AatI	Acetobacter aceti	IFO 3281	AGG^CCT	meth_site	O	1254
AatII	Acetobacter aceti	IFO 3281	GACGT^C	meth_site	A D E F G I K L M N O R	1429
AauI	Arthrobacter aurescens	S.K. Degtyarev	T^GTACA	meth_site	I	1327
BmaAI	Bacillus macerans	ATCC 8513	CGATCG	meth_site		452
BsaAI	Bacillus stearothermophilus G668	Z. Chen	YAC^GTR	meth_site	N	578
BspAAI	Bacillus species AA	N.I. Matvienko	C^TCGAG	meth_site		38
BspAAII	Bacillus species AA	N.I. Matvienko	T^CTAGA	meth_site		38
BspAAIII	Bacillus species AA	N.I. Matvienko	G^GATCC	meth_site		38
CspAI	Clostridium	ATCC	CCGC			

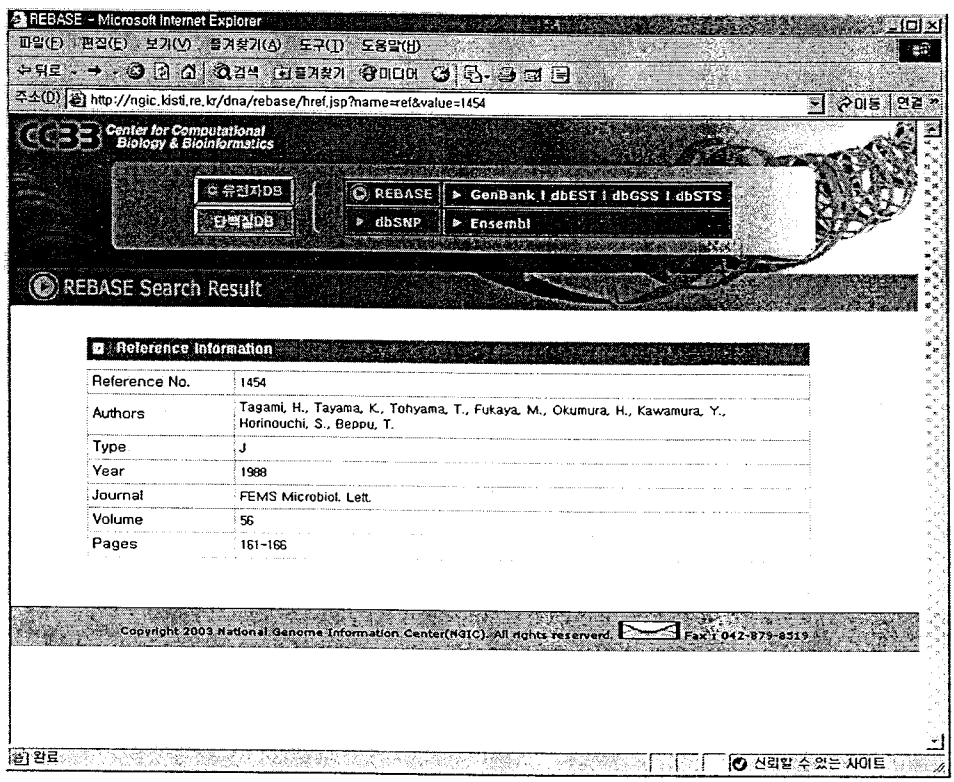
<figure 1-4> Rebase: 검색 결과 페이지



<figure 1-5> Supplier information: Fermentas AB and its restriction Endonucleases



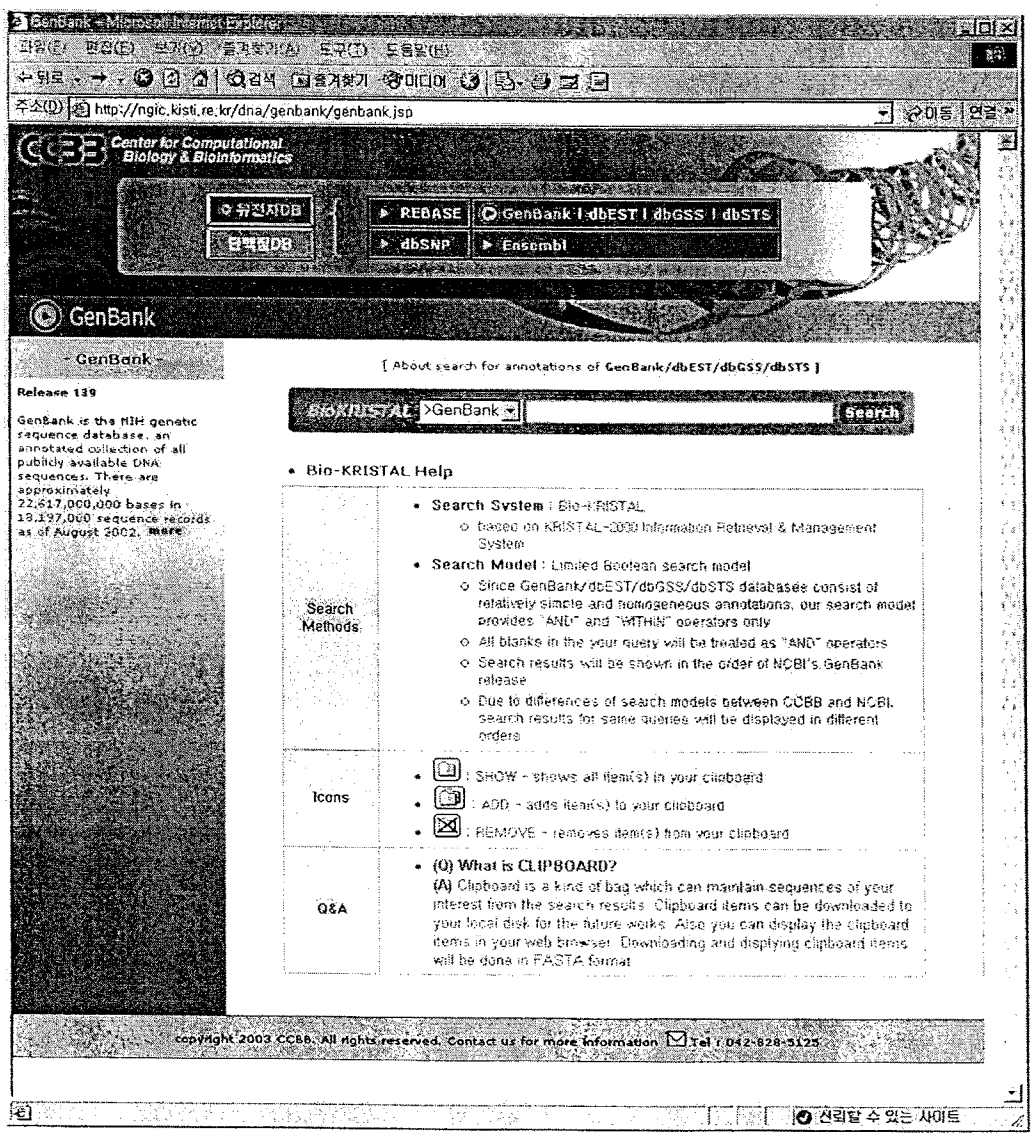
<figure 1-6> Enzyme information: Bsp119I



<figure 1-7> Reference information: #1454

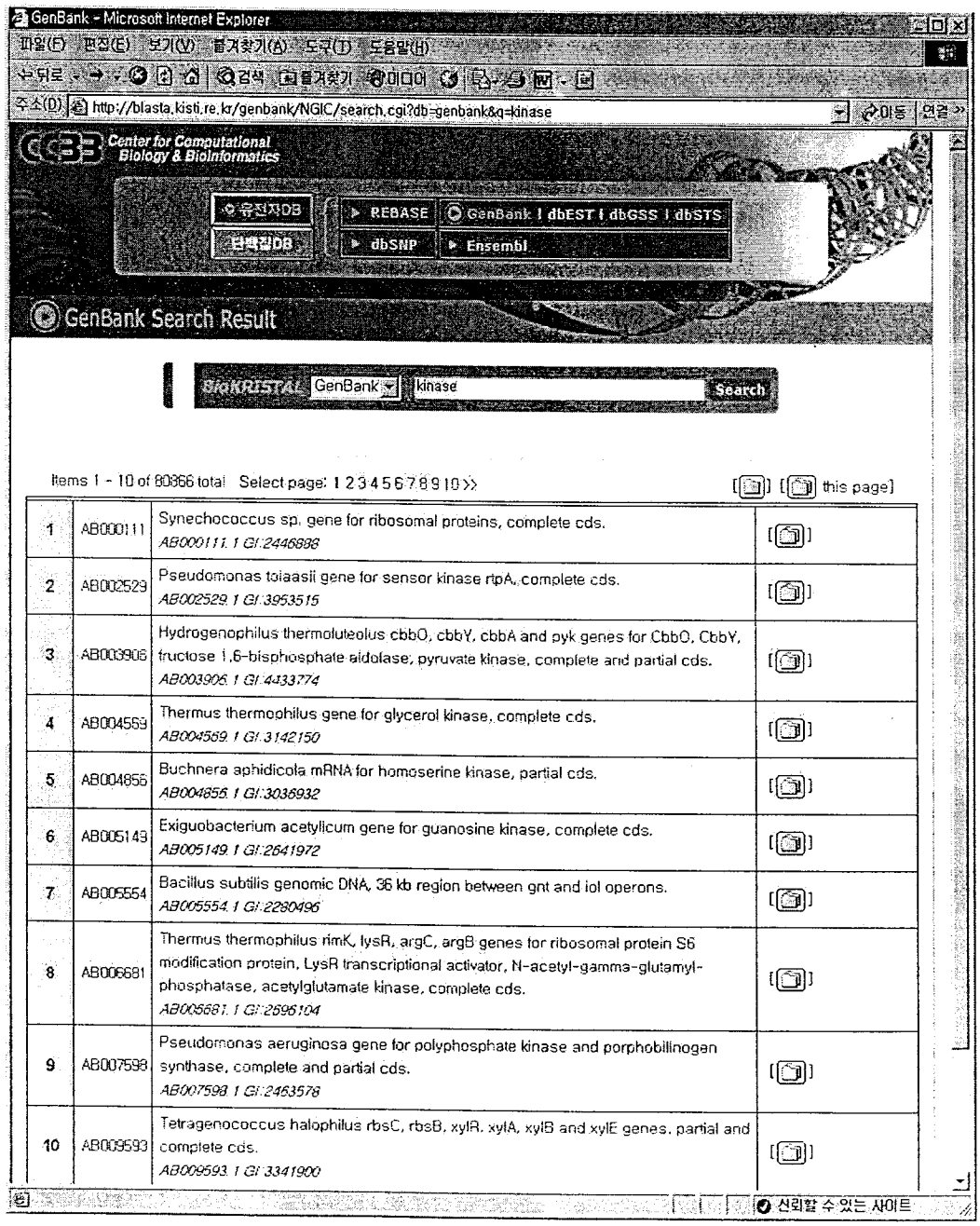
(2) GenBank: dbEST, dbGSS, dbSTS

GenBank (Gene Bank)는 미국의 NCBI와 유럽 분자생물학 실험실인 EMBL(Europena Molecular Biology Laboratory)과 일본의 유전자 데이터베이스인 DDBJ(DNA DataBank of Japan)이 공동으로 구축한 최대의 유전자 데이터베이스로 DNA의 염기서열에 관한 각종 정보와 자료를 제공하고 있다. 본 연구에서는 CCBB에서 구축한 KRISTAL 기반 GenBank 데이터베이스의 검색 인터페이스를 기존의 PHP에서 Java/JSP로 변환함으로써 평균 20%정도의 검색 성능 향상을 보이고 있다. <figure 1-8>은 GenBank의 검색 페이지를 보이고 있다.



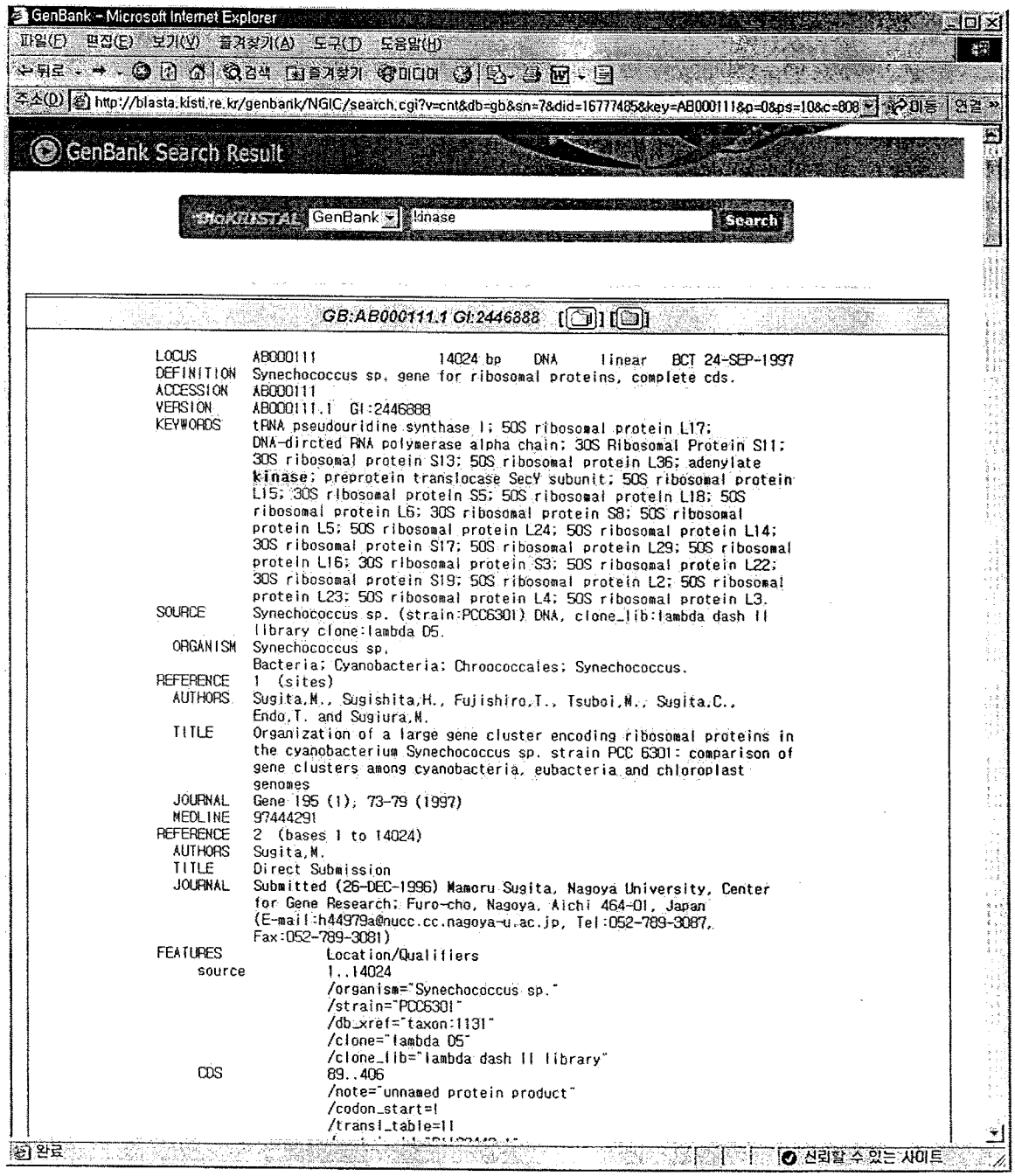
<figure 1-8> GenBank: 검색 페이지

GenBank 데이터베이스의 검색은 검색 화면을 단순화함으로써 사용자가 검색에 관한 전문적인 지식이 없어도 손쉽게 검색을 할 수 있도록 하였다. 그림에서 볼 수 있듯이 사용자는 자신이 찾고자 하는 염기서열에 대한 keyword 만을 나열함으로써 원하는 검색 결과를 얻을 수 있다. <figure 1-9>는 'kinase' 검색 keyword에 대한 GenBank 데이터베이스의 검색 결과를 보이고 있다.



<figure 1-9> GenBank: 검색 결과

<figure 1-10>은 <figure 1-9>의 검색 결과들 중 Entry AB003906의 상세 정보를 보이고 있다.



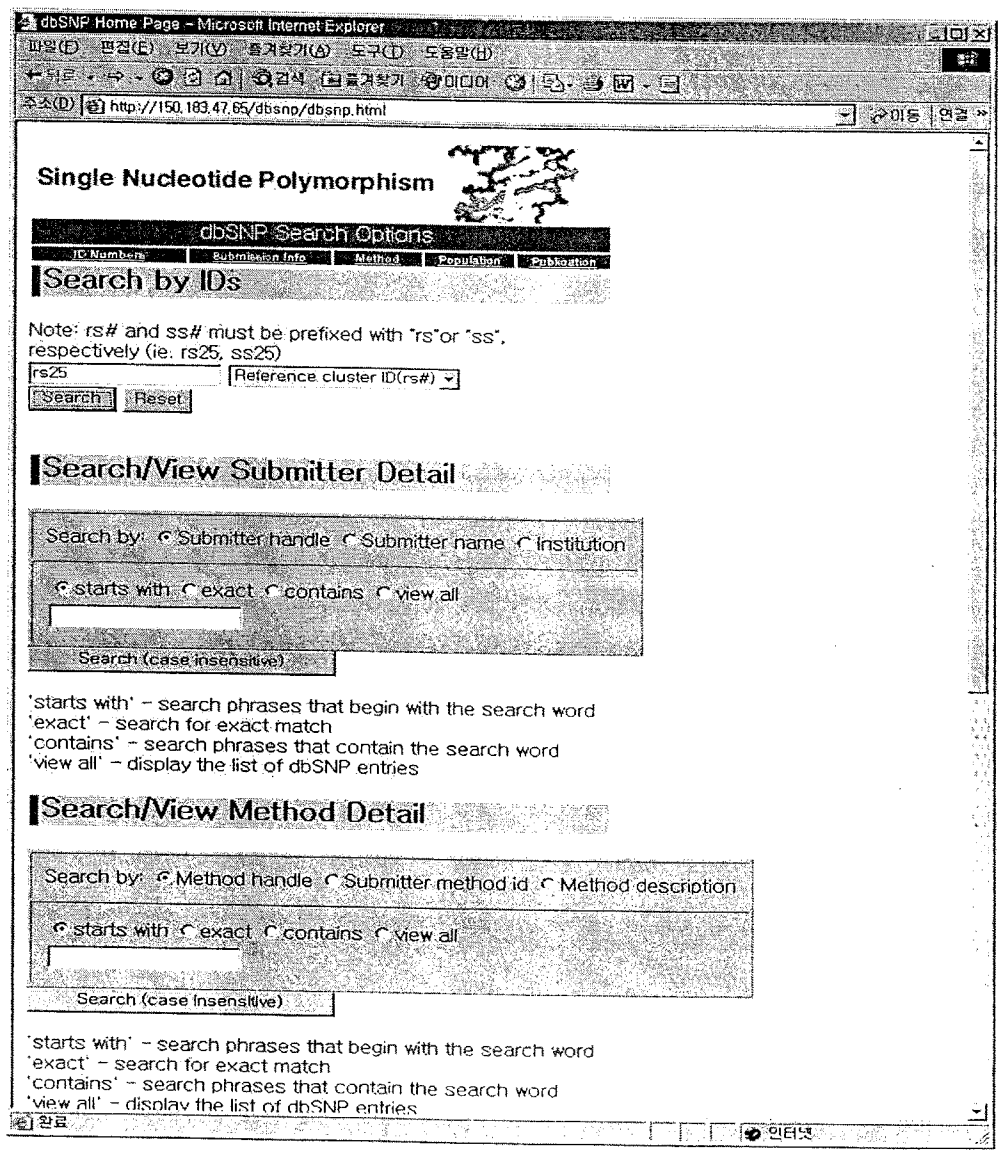
<figure 1-10> GenBank-상세 정보 보기

이밖에도 사용자의 편이한 검색 활동을 지원하기 위해 검색 결과를 클립보드에 저장하는 기능, 저장된 결과의 염기서열 정보를 파일 혹은 브라우저로 다운받는 기능 등을 제공한다.

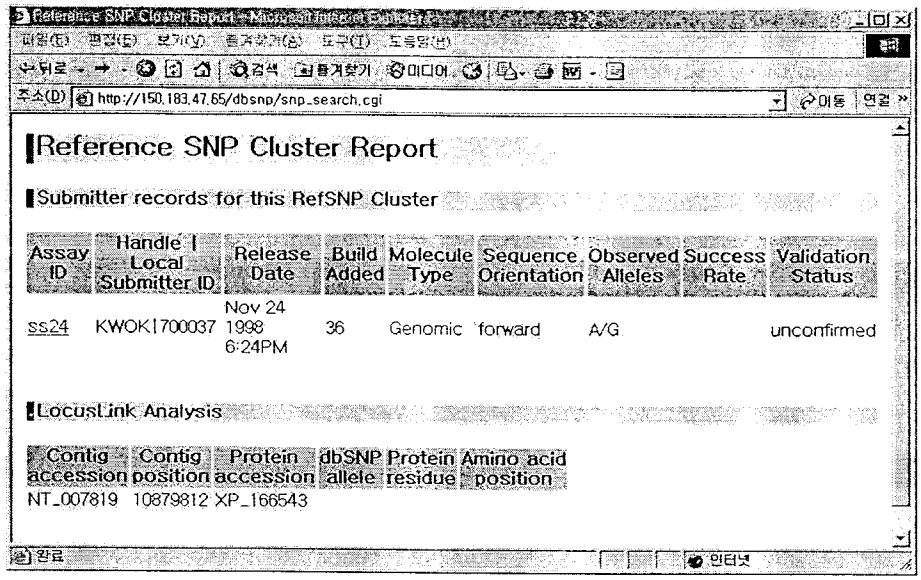
(3) dbSNP과 Ensembl

(가) dbSNP

SNP는 단일핵산다형성(Single Nucleotide Polymorphism)에 대한 데이터베이스로 유전적인 표현형을 가진 서열 변종과의 결합, 질병과 모집단에서의 특정 차이(SNP)간의 결합을 검색하여 질병 유전자의 발견하는 연구를 가능케 한다. <figure 1-11>과 <figure 1-12>는 각각 dbSNP의 검색 페이지와 검색 결과 페이지를 보이고 있다. 자세한 사항은 2절을 참조 (p 84)



<figure 1-11> dbSNP: 검색 페이지

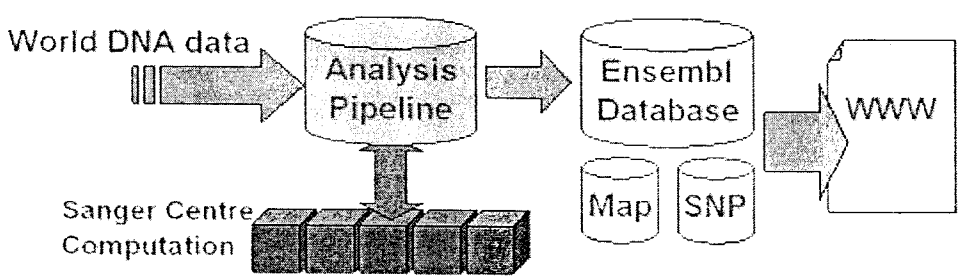


<figure 1-12> dbSNP: 검색 결과 페이지

Ensembl 데이터베이스는 Sanger Institute와 EMBL-EBI 공동 프로젝트로 서비스되고 있는 서열데이터의 자동화된 annotation을 지원하는 데이터베이스 시스템으로서, human genome의 시퀀스 조각들을 자동으로 찾아내어 전체 긴 strand로 맞추고, 자동으로 gene을 찾아내어 annotation하여 보여주는 데이터베이스이다. 아래의 <figure 1-14>, <figure 1-15>, <figure 1-16>는 각각 Ensembl 데이터베이스의 메인화면과 검색 페이지와 검색 결과페이지를 보이고 있다.

(나) Ensembl 데이터베이스

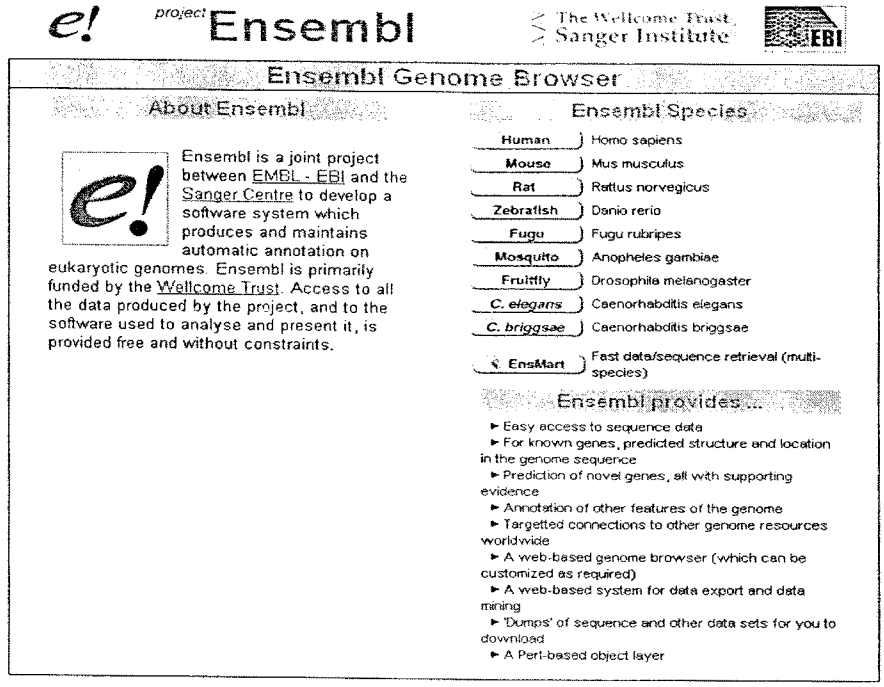
1) 개요



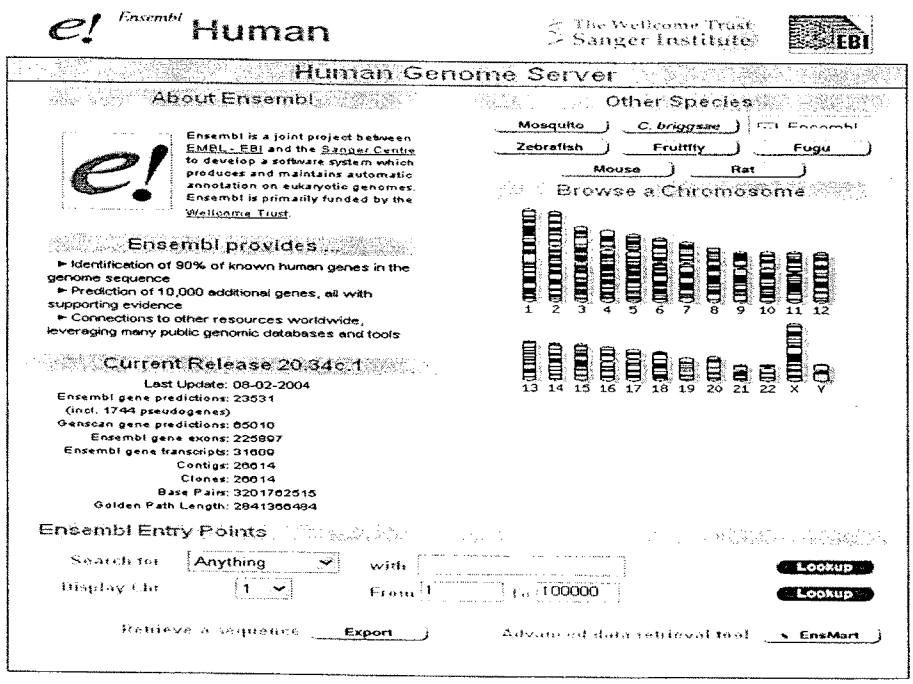
<figure 1-13> Ensembl 데이터 유통 과정

Ensembl 데이터베이스는 현재 human을 비롯한 mouse, fly, zebrafish, mosquito 등의 진핵생물의 유전체정보를 담고 있으며, 추후 더 많은 생물 종으로 그 영역을 확대하고

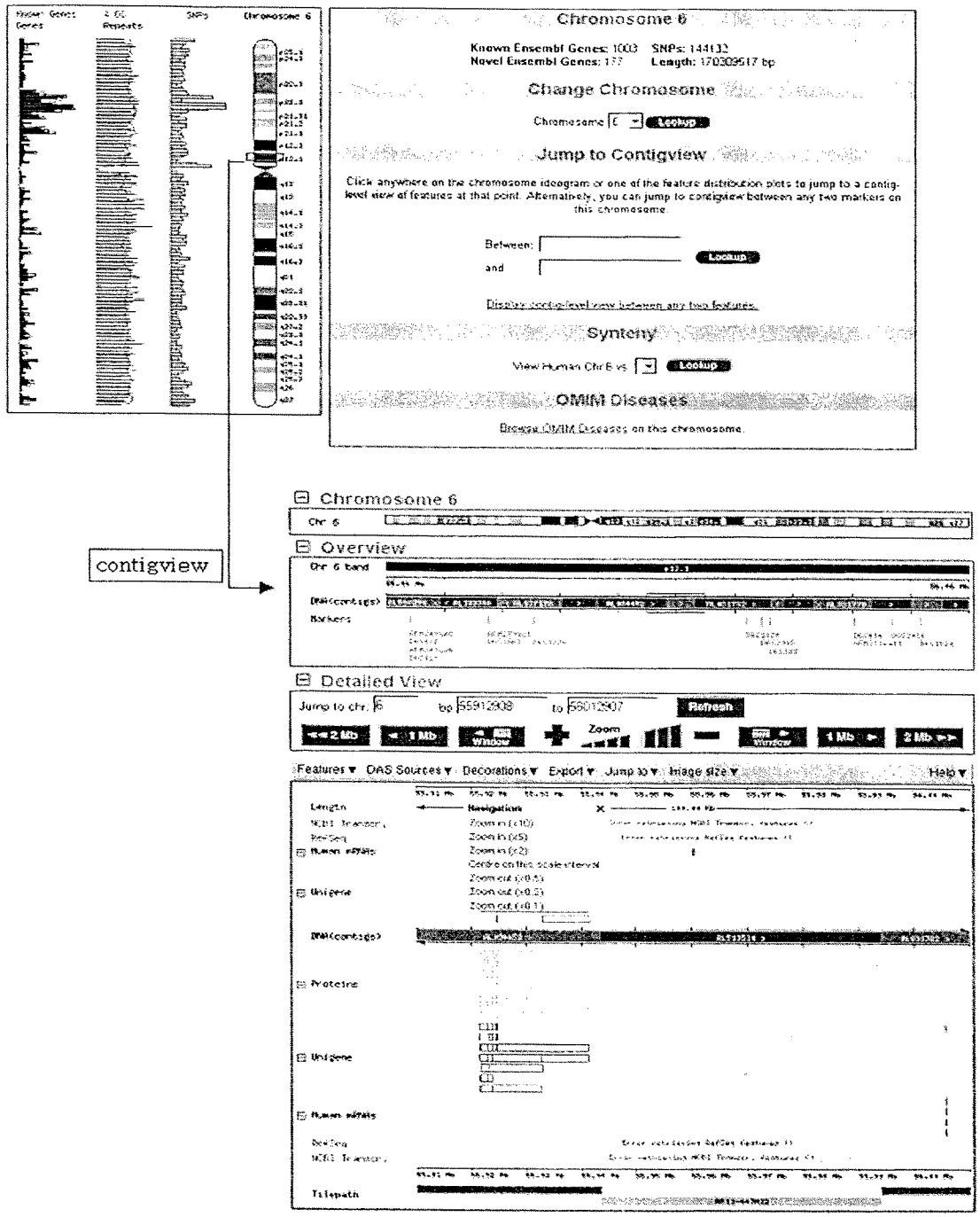
있다.



<figure 1-14> Ensembl 메인화면



<figure 1-15> Ensembl: 검색 페이지



<figure 1-16> Human chromosome6 검색결과화면

<필요 사양>

- UNIX 계열의 OS (예 Tru64, Solaris, Linux 등)
- MySQL 데이터베이스의 완전한 세트를 수용하기 위한 약 35 GB의 하드디스크 용량 및 자료를 다운로드하고 풀어놓기 위한 약 35 GB의 하드디스크 용량

2) 설치방법

가) 사이트 구조

구축할 사이트 구조는 서버의 루트 디렉토리 내부에 완비되어 있다. 이 서버 루트 디렉토리가 이용자 시스템의 어디에 위치하는지는 중요하지 않다. 서버 루트 디렉토리 안의 파일들이 아래 설명된 것처럼 위치하도록 지정해야 한다.

사이트 구조는 다음과 같다(서버 루트의 예로서 "usr/local/ensembl "을 이용함)

/usr/local/ensembl

|-- **bioperl-live** Ensembl이 기초한 생물 모형화 도구 모음

|-- **ensembl** Ensembl core 모듈 모음

|-- **ensembl-compara** 교잡종(cross-species) 비교를 다루는 코드

|-- **ensembl-draw** 웹사이트에서의 그리기 코드

|-- **ensembl-lite** 비정규화 자료 액세스용 코드

|-- **ensembl-map** 지도자료를 다루는 코드

|-- **ensembl-mart** 시장 자료-검색 시스템 액세스용 코드

|-- **perl** 웹사이트 mod_perl 모듈

|-- **modules** 웹사이트 액세서리 모듈 모음

|-- **conf** 웹사이트 환경 파일 모음

|-- **htdocs** 웹사이트 HTML 과 이미지 파일 모음

|-- **utils** 유틸리티 스크립트 모음

나) 설치

① Non-Ensembl 절

: 아파치나 MySQL처럼 Ensembl 사업의 일부분은 아니지만 웹사이트의 운영을 위해 필요한 응용프로그램/모듈의 설치를 내용으로 한다.

<Non-Ensembl 응용프로그램 구축/설치>

이 응용 프로그램들은 특정 Ensembl 버전에 구애받지 않는다. 즉 새로운 Ensembl 버전이 나왔을 때 Ensembl 설치를 새로운 버전으로 향상시킨다고 해도 이 응용프로그램들을 다시 설치할 필요는 없다.

이 소프트웨어들은 Ensembl처럼 모두 공개소스 소프트웨어이며 무료로 다운로드할 수 있고 사용할 수 있다. 그러나 상업적인 환경에서 Ensembl을 설치할 경우에는 각각의 응용프로그램이 어떤 라인센스 하에서 배포되었는지 기록을 확인하여야 한다.

(아래 설명은 설치 머신의 루트 권한을 소유를 가정한 것이다. 소유하지 않았을 경우, 이 소프트웨어를 설치하도록 한다.)

이 소프트웨어의 일부 혹은 전부를 설치하였을 경우, 먼저 설치된 소프트웨어(특히 RPM으로부터 설치된 mod_perl과 결부된 아파치)가 운영되고 있는 사이트를 얻는 데 문제가 있으면 다음의 설명에 따라 최신판을 설치할 것을 권장한다.

ⓐ CVS

CVS는 Ensembl로의 소스코드를 저장할 때 사용하는 소프트웨어 판 제어시스템이다. Ensembl 소스코드를 다운로드받고 싶다면 CVS를 설치해야한다.

<설치방법>

1. http://ftp.cvshome.org/LATEST/ 로부터 최신 소스를 다운로드 한다. 현재 최신

판의 안정 버전은 1.11.11이며 다운로드 받을 파일은 cvs-1.11.11.tar.gz이다.
2. 작업 디렉토리에서 소스의 압축을 다음처럼 푼다:

```
$ gunzip < cvs-1.11.11.tar.gz | tar xvf -
$ cd cvs-1.11.11
$ ./configure
$ make
$ make install
```

ⓑ Perl

웹사이트를 실행하려면 Perl5 혹은 5.6.0 판 또는 더 이상의 판이 필요하다. 버전을 알고 싶거나 Perl이 깔려있는지를 알고 싶으면 아래 구문을 실행한다.

```
$ perl -v
```

Perl이 깔려 있지 않거나 업그레이드를 하고 싶으면 http://www.cpan.org/에 가서 'source code' install을 선택한 후 웹 사이트의 설치 설명을 따르도록 한다.

ⓒ MYSQL

MySQL은 매우 대중적인 공개소스의 관계형 데이터베이스 시스템이다. 미리 컴파일된 바이너리파일을 http://www.mysql.com/downloads/에서 받거나 설명에 따라 소스를 설치할 수 있다.

MySQL의 설치는 다른 설치보다는 좀 더 복잡하지만 아주 상세한 설명이 MySQL 사이트(http://www.mysql.com/documentation/mysql/bychapter)에 있다.

<설치방법>

1. http://www.mysql.com/downloads/mysql-3.23.html로부터 소스압축파일을 다운로드받는다. 현재의 안정화된 버전을 받는다. 현재 안정화된 버전은 3.23.52이며 받을 파일은 mysql-3.23.52.tar.gz이다.

2. MySQL 서버의 시스템 사용자와 그룹을 생성한다. 예를 들면 설명에 따라 mysqldba를 사용할 것이다. mysqldba를 선택한 사용자들로 바꾼다.
3. MySQL이 설치될 디렉토리를 생성한다. 디렉토리의 하부디렉토리가 데이터베이스를 보유할 것이므로 충분한 공간(적어도 35GB)을 가지는 위치를 선택한다. 한 예로써 /data/mysql을 사용할 것이다. 이 설명을 따를 때에는 다시 한번 /data/mysql을 선택하는 다른 경로로 바꾼다.
4. 압축파일을 /data/mysql로 복사한다.
5. 압축된 소스를 다음처럼 푼다.

```
$ gunzip < mysql-3.23.52.tar.gz | tar xvf -
$ ./configure --prefix=/data/mysql --localstatedir=/data/mysql/var
$ make
$ make install
$ chown -R root /data/mysql
$ cd /data/mysql
$ ./bin/mysql_install_db
$ chown -R mysqldba /data/mysql/var
$ chgrp -R mysqldba /data/mysql
```

이제 다음 명령어로 MySQL 서버를 시동할 수 있다.

```
$ /data/mysql/bin/safe_mysqld --user=mysqldba &
```

또한 MySQL 콘솔(직접 데이터베이스에 질의할 수 있는 곳)을 다음 명령어로 불러낼 수 있다.

```
$ /data/mysql/bin/mysql --user=mysqldba
```

ⓓ 아파치 & mod_perl

아파치는 Ensembl 사이트가 실행되는 웹 서버이다. mod_perl은 아파치를 위한 모듈로서 perl 스크립트를 요청될 때마다가 아니라 한번만 컴파일하게 함으로써 모든 것이 더

욱 빠르게 실행되게 한다.

아파치의 최신판은 버전 2 계열이지만 아파치 2를 위한 mod_perl은 아직 개발 중이다. 따라서 아파치 2를 사용하지 않을 것을 권장한다. Ensembl은 아파치와 mod_perl의 2판에서 완벽하게 잘 실행될 것이지만 아직 시도해 보지 않았으므로, 다음의 설치설명은 아파치와 mod_perl의 2판에서는 안 맞을 수도 있다.

<설치방법>

1. http://httpd.apache.org/dist/httpd/로부터 아파치 소스의 압축파일을 다운로드한다. 현재의 안정된 버전을 가져온다. 현재 안정된 버전은 1.27이며 다운로드할 파일은 apache-1.3.27.tar.gz이다.
2. http://www.cpan.org/modules/by-module/Apache/로부터 mod_perl 소스를 다운로드한다. 현재 최신판은 1.27이고 다운로드할 파일은 mod_perl-1.27.tar.gz이다.
3. 작업 디렉토리에서 다음처럼 모든 소스의 압축을 푼다.

```
$ gunzip < apache_1.3.27.tar.gz | tar xvf -
$ cd apache_1.3.27
$ ./configure
$ cd ../mod_perl-1.27
```

4. Perl 메이크 파일을 만든다

```
$ perl Makefile.PL APACHE SRC=../apache_1.3.27/src \
  DO_HTTPD=1 USE_APACI=1 EVERYTHING=1
```

5. 'make' 유틸리티를 실행한다.

```
$ make
```

6. 설치를 시작한다.

```
$ make install
```

7. 아파치 설치 디렉토리로 되돌아간다.

```
$ cd ../apache_1.3.27
```

8. 아파치 설정 스크립트를 실행한다.

```
$ ./configure --enable-module=include \
  --activate-module=src/modules/perl/libperl.a \
  --enable-module=perl
```

9. 'make'를 실행하고 설치한다.

```
$ make
$ make install
```

ⓔ **Perl** 모듈

Ensembl 웹 사이트를 실행시키기 위해서는 아주 적은 수의 Perl 모듈만을 설치하면 된다. 이 모듈들은 모두 www.cpan.org 로부터 다운로드 받아 설치할 수 있다. 모듈 압축 파일을 다운로드한 후 작업 디렉토리에서 압축을 풀고 모듈을 설치한다.

```
$ gunzip < module.tar.gz | tar xvf -
$ cd module
$ perl Makefile.PL
$ make
$ make test
$ make install
```

필요한 모듈과 해당 URL의 목록은 다음과 같다. 항상 최신 모듈을 설치해야한다.

1. Apache::DBI아파치가 속도 때문에 데이터베이스를 연결한다.
   http://www.cpan.org/modules/by-module/Apache/ApacheDBI-0.89.tar.gz
2. CGIPerl 스크립트가 손쉽게 cgi를 파싱할 수 있도록 한다.
   http://www.cpan.org/modules/by-module/CGI/CGI.pm-w.89.tar.gz
3. Compress::Zlib DAS를 압축하기 위한 모듈
http://www.cpan.org/modules/by-module/Compress/Compress-Zlib-1.16.tar.gz
4. DBIPerl을 위한 공용 데이터베이스 인터페이스
   http://www.cpan.org/modules/by-module/DBI/DBI-1.30.tar.gz
5. DBD::MysqlDB 인터페이스를 위한 MySQL 드라이버 모음

http://www.cpan.org/modules/by-module/DBD/Msql-Mysql-modules-1.2219.tar.gz
6. GD그래픽 라이브러리
   (주의: 추가 모듈이 필요할 수도 있다. 설치 설명을 읽도록 한다.)
   http://www.cpan.org/modules/by-module/GD/GD-2.041.tar.gz
7. Digest::MD5파일을 위한 고유 체크섬 계산
   http://www.cpan.org/modules/by-module/Digest/Digest-MD5-2.20.tar.gz
8. Storable데이터 구조를 저장하고 복구하는데 쓰인다.
   http://www.cpan.org/modules/by-module/Storable/Storable-2.05.tar.gz
9. LWPDAS 소스와 통신하기 위해 DAS에 의해 이용된다.
   (주의: 추가 모듈이 필요할 수도 있으므로 설치 기록을 읽는다.)
   http://www.cpan.org/modules/by-module/LWP/libwww-perl-5.65.tar.gz
10. XML::Parser DAS에 의해 DAS 소스를 파싱하는데 이용된다.
    http://www.cpan.org/modules/by-module/XML/XML-Parser-2.31.tar.gz

(주의: 이 모듈은 첫 번째로 설치되어야 할 **expat** 라이브러리를 둘러싸는 포장지와 같다. expat 소스를 http://sourceforge.net/projects/expat/으로부터 다운로드한다. 최신판 (현재는 expat-1.95.5.tar.gz)을 받는다.)

다음의 명령어에 따라 실행한다.

```
$ gunzip < expat-1.95.5.tar.gz | tar xvf -
$ cd expat-1.95.5
$ ./configure
$ make
$ make install
```

11. Parse-Excel 내보내기 기능에 의해 사용된다.
    RecDescent http://www.cpan.org/modules/by-module/Parse/Parse-Rec Descent-1.80.tar.gz
12. Spreadsheet::Excel 스프레드시트를 내보내는데 쓰인다.
    WriteExcel
    http://www.cpan.org/modules/by-module/Spreadsheet/Spreadsheet-WriteExcel-0.39.tar.gz
13. HTML::MartView에 의해 쓰인다.
    Template http://www.cpan.org/modules/by-module/HTML/HTML-Template -2.6.tar.gz
14. File::TempMartView에 의해 쓰인다.
    http://www.cpan.org/modules/by-module/FILE/FILE-Temp-0.12.tar.gz
15. Mail::MailerMartView에 의해 쓰인다.
    http://www.cpan.org/modules/by-module/MAIL/MailTools-1.51.tar.gz

② Ensembl 절

: Ensembl 자료, 모듈, 웹사이트 코드를 설치를 내용으로 한다.

<Ensembl 구축/설치>

이 절은 Ensembl 데이터, perl 모듈, 웹 코드의 설치방법 및 BioPerl 설치방법이다.

ⓐ 버전 관리

외부에서 유지시키는 소프트웨어에 더하여 Ensembl 웹사이트는 세 가지의 차별화된 구성요소들로 이루어진다.

- Ensembl API 코드
- Ensembl 데이터베이스
- Ensembl 웹 코드

Ensembl API는 데이터베이스 스키마에 따른 정수 버전 번호를 부여받는다. 현재의 API 배포판은 버전 9이고 'branch-ensembl-9'이라는 CVS 태그를 가지고 있다.

Ensembl 데이터베이스의 이름은 종, 데이터베이스 유형, 데이터베이스 버전으로 이루어져 있다. 데이터베이스 버전은 API 판(데이터베이스 스키마와 같음), 자료 개정판의 추적을 가능하게 하는 추가 증분 정수로 이루어져 있다. 현재의 human 'core' 데이터베이스는 homo_sapiens_core_9_30으로 명명되어 있으므로 즉, API 버전 9, 데이터베이스 버전 30 (NCBI 어셈블리에 기초한)이다. 이 어셈블리에서 차후의 데이터 배포판은 소문자(a, b, 등)로 첨자 될 것이다.

Ensembl 웹 코드 버전은 API 버전과 API 개정판 사이에서 웹 코드의 개정판을 가능하게 하는 추가 증분 정수로 이루어져 있다. 현재의 웹 코드 배포판은 버전 9.1이며 그에 대응하는 'branch-ensembl-9-1'이라는 CVS 태그가 있다.

여기서 중요한 것은 API 버전의 구성요소들이 같이 맞물려야 한다. 즉 '버전 9 ' API, ' 버전 9 ' 웹 코드,' 버전 9 ' 데이터베이스로 이루어진 웹사이트는 올바로 작동할 것이다.

ⓑ Ensembl 데이터 설치하기

Ensembl 데이터는 MySQL로 가져오기 위한 탭으로 구분된 텍스트 파일 형식으로 Ensembl FTP 사이트에서 제공된다. ftp://ftp.ensembl.org/pub 에는 각각의 종에 따른 각각의 배포판의 디렉토리가 있다. 최근 버전들은 current_species, 즉 current_human, current_mouse 등으로 명명되어 있다. 아래의 디렉토리 구조들은 다음과 같다.(current_human을 예로 들었을 때)

ftp.ensembl.org/pub/current_human

|--data

|--fastacDNA, DNA와 펩타이드 덤프

|--flatfilesEMBL과 GenBank 포맷 덤프

|--golden_path masked되거나 unmasked된 염색체 배열 덤프

|--mysql데이터베이스 덤프

Ensembl 데이터를 설치하기 위해서 mysql 디렉토리 내용을 주목해야한다. 다음은 설치하고자 하는 각각의 데이터베이스가 들어갈 디렉토리를 포함한다.

ftp.ensembl.org/pub/current_human/data/mysql

|-- **homo_sapiens_disease_21_30** 질병 데이터베이스

|-- **homo_sapiens_embl_21_30** Embl유전자데이터베이스

|-- **homo_sapiens_core_21_30** core Ensembl 데이터베이스

...etc...

각각의 데이터베이스 디렉토리는 데이터가 바르게 다운로드 되었는지 검증할 수 있는 체크섬 파일 (UNIX의 "sum" 유틸리티를 이용함), 데이터베이스의 각각의 표를 위한 데이터 파일, 데이터베이스의 표 구조를 구축하는데 필요한 SQL 명령어들이 들어있는 SQL 파일이 들어있다.

▶ **multi-species 데이터**

각각의 종 데이터에 더하여 multi-species 데이터베이스 - 즉 사이트에 전체적으로 영향을 끼치는 데이터베이스(예를 들면 ensembl_web_user_db)나 multi-species에 관한 메타정보를 설명하는 데이터베이스(예를 들면 ensembl_compara) - 가 있다. 이 데이터베이스들은 여러 버전으로 존재할 것이므로 디렉토리 안의 README 파일을 참고하면 어느 데이터베이스가 어느 배포판과 호환되는지 알 수 있을 것이다.

ftp.ensembl.org/pub/multi-species/data/mysql

|-- **README**

|-- **ensembl_compara_21_1**

|-- **ensembl_help_21_1**

|-- **ensembl_mart_21_1**

|-- **ensembl_web_user_db**

...etc...

어느 종을 설치하던 간에 multi-species 데이터베이스를 같이 설치해야한다. 즉

compara, help, mart, web_user_db 모두가 설치되어야한다.

(주의) 설명된 것처럼 FTP 사이트가 이상적으로 구축될 것이다. 만약 혹시라도 공간이나 유지능력 때문에 파일들은 설명대로 위치하지 않을 수도 있으므로 ftp 사이트에서 데이터의 위치를 설명해 주는 README 파일을 찾아서 확인한다.

<설치방법>

1. ftp.ensembl.org/pub/current_*organism*/data/mysql에서 설치하고 싶은 생물체의 디렉토리를 다운로드한다. 주목할 점은 Ensembl 디렉토리에는 DNA와 특성표에 대한 여러 가지 파일들이 있는데 이들은 매우 큰 표들이라 뭉치 파일은 다운로드를 쉽게 하기 위해 좀 더 작은 덩어리들로 나뉘어져 있다는 것이다.
2. 각각의 표 파일은 gzip으로 압축되어 있으므로 각각의 데이터베이스가 디렉토리에 있는 구조를 유지시키면서 작업 디렉토리에 데이터의 압축을 푼다. 각각의 데이터베이스를 다운로드 할 때 데이터베이스 디렉토리로 들어가서 다음의 3-5 과정을 수행한다. 한 예로써 homo_sapiens_core_21_1 데이터베이스를 사용한다. 이것을 설치하고 싶은 데이터베이스에 대하여 적당히 바꾸면 된다. 또한 multi-species 데이터베이스도 설치해야한다.
3. MySQL 콘솔 세션을 시작한다. (필요하면 위의 MySQL 설치 장을 본다.) 그리고 다음 명령어를 입력한다.

```
> create datatbase homo_sapiens_core_21_1
```

4. 콘솔 세션에서 빠져 나온 후 다음의 명령어를 입력하여 ensembl SQL 파일 - 다운로드한 데이터의 압축을 푼 디렉토리에 있을 - 을 실행시킨다. 이는 과정 3에서 만든 빈 데이터베이스를 위한 스키마를 만들 것이다. 설치 디렉토리로서 /data/mysql의 MySQL 세팅을, 또 데이터베이스 사용자로서는 mysqldba를 예로 들고 있다는 점에 주목한다. 주의할 점은 여기서 mysqldba는 MySQL 계정이며 시스템 사용자와는 다르다는 것이다. 사용자의 생성/관리에 대한 설명은 MySQL 설명서를 참조한다.

```
$ /data/mysql/bin/mysql -u mysqldba homo_sapiens_core_21_1 < homo_sapiens_core_21_1.sql
```

5. 다음의 명령어를 사용하여 위에서 만든 데이터베이스 구조 안으로 데이터를 적재한다.

$ /data/mysql/bin/mysqlimport -u mysqldba homo_sapiens_core_21_1 *.txt.table

핵심 Ensembl 데이터베이스를 생성하고 적재하였다.

ensembl_web_user를 제외한 모든 데이터베이스에서는 웹 사이트의 작동을 위하여 단지 읽기 엑세스만을 필요로 한다는 것에 주목한다. ensembl_web_user_db는 MySQL 사용자에게 삭제/삽입/갱신 퍼미션(접근허가)를 요구한다. 웹 사이트가 데이터를 입력하는 데이터베이스는 오직 이것뿐이므로 ensembl_web_user_db에는 다운로드할 .table (data) 파일들이 없다는 것에도 또한 주목한다.

(주의) MySQL은 데이터베이스를 적재하기 위하여 꽤 많은 임시 공간을 필요로 한다. /tmp 디렉토리(MySQL이 디폴트로 사용함)이 너무 작을 수도 있는데 이 경우에는 Error 28(이 오류코드의 뜻을 알려면 MySQL 도구인 perror를 사용한다.)이 뜰 것이다. MySQL에게 임시 파일들을 다른 장소로 옮겨 쓰도록 명령할 수 있다. 더 자세한 것은 다음의 MySQL 설명을 참조한다.

http://www.mysql.com/doc/T/e/Temporary_files/html.

가장 간단한 해결책은 mysqld를 시작할 때 --tmpdir my_spacious_tmp_location 인자를 쓰는 것이다.

▶ GO 자료

Ensembl 웹사이트는 유전자 온톨로지 데이터베이스(Gene Ontology Database: GO)에 둘 중의 한 방법으로 연결될 수 있다. 연결들은 (예를 들면 GeneView로부터) 다음 둘 중의 하나로 만들어진다.

1. 외부 GO 데이터베이스 브라우저 (디폴트 설치는 유럽 생명정보 연구소(European Bioinformatics Institute)의 QuickGo를 사용한다.)

2. GO 데이터베이스의 로컬 카피.

위의 선택 (1)은 새로 Ensembl을 설치할 경우의 디폴트 상태이다. www.ensembl.org 에서 사용되듯이 로컬 GO 설치를 원하시면 데이터를 설치한 후 데이터베이스를 추가하기 위하여 사이트를 구성해야 한다.

만약 선택 (2)를 택하면 다음으로부터 MySQL GO 데이터베이스를 다운로드 해야한다.

http://www.godatabase.org/dev/database/archive/latest/go_XXXXX-tables.tar.gz

(XXX는 가장 최근 배포 일자이다.) 설치 설명을 따라 로컬 MySQL GO 데이터베이스를 생성하고 다운로드한 데이터 파일들로 표를 채운다. 즉시 사용 가능한 완전한 GO 데이터베이스를 가지게 될 것이다. GO 데이터베이스를 인식하도록 Ensembl 웹 사이트를 설정하려면 3절을 본다.

▶ Ensembl, 웹, BioPerl 모듈 설치

위 설명의 사이트 구조 편을 보면 사이트는 하나의 서버-루트 디렉토리에 기초하고 있다는 것을 기억하게 될 것이다. Ensembl, BioPerl, 웹 모듈은 이 디렉토리 안에 모두 설치된다. 적당한 위치를 택한 후 서버-루트 디렉토리를 생성한다. 예로써 /usr/local/ensembl을 사용하겠다. 이 설명서를 따를 때는 /usr/local/ensembl을 서버-루트 디렉토리와 바꾸도록 한다.

<Ensembl 모듈 설치방법>

1. 서버-루트 디렉토리로 간다.
   $ cd/usr/local/ensembl
2. Sanger CVS 서버로 접속한다. (비밀번호: CVSUSER)
   $ cvs -d :pserver:cvsuser@cvsro.sanger.ac.uk:/cvsroot/CVSmaster login
3. ensembl API와 CVS로부터의 웹 코드를 검사한다. (Sanger CVS 서버로부터 지역 컴퓨터로 코드를 내려받기를 한다.)

$ cvs -d :pserver:cvsuser@cvsro.sanger.ac.uk:/cvsroot/CVSmaster co -r \ branch-ensembl-21 ensembl-api

$ cvs -d :pserver:cvsuser@cvsro.sanger.ac.uk:/cvsroot/CVSmaster co -r \ branch-ensembl-21-1 ensembl-web

서버-루트 디렉토리의 목록은 다음처럼 보일 것이다.

conf/ensembl-draw/ensembl-map/modules/ensembl/ensembl-external/ensembl-mart/perl/ensembl-compara/ensembl-lite/htdocs/utils/

<BioPerl 모듈 설치>

1. 서버-루트 디렉토리로 간다.
   $ cd /usr/local/ensembl
2. BioPerl CVS 서버로 접속한다. (비밀번호: cvs) :
   $ cvs -d :pserver:cvs@cvs.open-bio.org:/home/repository/bioperl login
3. BioPerl 코드를 점검한다.
   $ cvs -d :pserver:cvs@cvs.open-bio.org:/home/repository/bioperl co -r branch-07-ensembl-120 bioperl-live

모든 Ensembl 웹 사이트 코드와 데이터를 설치하였으며 설정할 준비가 되었다.

다) 설정

이번 절은 Ensembl이 로컬 설치에서 실행될 수 있도록 설정 변화를 설명한다.

① 웹 사이트를 위한 파일 설정

(1) 사이트 공통 설정(아파치 confgi, 전역) - SiteDefs.pm에 저장되어 있다.
(2) 종별 설정(데이터베이스 이름 등)]
- *species_name*.ini 파일 (예를 들면 Homo_sapiens.ini) 안에 들어 있다.
- Ensembl 웹사이트에서 보여주고 싶은 각각의 종에 대하여

species-specific.ini 파일이 필요하다.

이 배치 파일들에 더하여 DEFAULTS.ini 파일과 MULTI.ini 파일이 있다. DEFAULTS.ini는 종별 .ini 파일들을 위한 디폴트 값들을 가지고 있으므로 이 설정을 특정한 종류의 ini 파일에 오버라이트 하지 않는 이상 모든 종에 적용될 것이다. 이것은 multi-species ini 파일들에서 같은 설정을 만들기 위하여 소유물을 줄이는 것이다.

이 배치 파일들은 서버-루트의 "conf" 서브디렉토리에서 모두 찾을 수 있다.

SiteDefs.pm

이 파일을 vi와 같은 텍스트 편집기에서 연다면 이는 사실 Perl 스크립트라는 것을 볼 수 있을 것이다. 이 스크립트의 첫 부분은 사이트의 운영방법을 제어하는 많은 변수들을 내보낸다. 편집하고 싶은 부분은 다음 라벨 뒤의 절이다.

```
##############################################################
########################################  LOCAL  CONFIGURATION
VARIABLES  ####################
##############################################################
##############
```

이 절은 변수 목록($ENSEMBL_OOO 의 꼴), 변수값, 변수에 대한 주석 (#기호 뒤)으로 이루어져 있다.

설치와 맞추기 위하여 몇몇 변수들의 값을 바꾸어야 할 것이다.

② 일반 설정

$ENSEMBL_SERVERROOT값을 서버-루트로 바꾼다. 예를 들면 Ensembl 사이트를 /usr/local/ensembl에 설치하였다면 SiteDefs의 다음 행을 바꾸어야한다.

```
$ENSEMBL_SERVERROOT = '/usr/local/ensembl';
```

(작은 따옴표 안의 값만 바꾸면 된다.)

③ 아파치 웹서버의 설정

$ENSEMBL_SERVERNAME을 서버의 웹 이름(예를 들면 "www.yoursite.org ")으로 바꾼다. $ENSEMBL_USER와 $ENSEMBL_GROUP을 아파치 웹서버를 운영할 시스템 사용자와 집단으로 바꾼다. 보통 보안 때문에 특수 사용자("nobody"같은 경우)는 거의 권한이 없다.

④ 메일 설정 - 오류 메시지

오류가 자동적으로 이메일로 보고 되길 원한다면 $ENSEMBL_MAIL_ERRORS값을 1로 바꾸고 $ENSEMBL_ERRORS_TO값을 이메일 주소로 바꾼다. 오류가 이메일로 보고 되지 않도록 하려면 $ENSEMBL_MAIL_ERORRS값을 0으로 바꾼다.

⑤ 사용자 데이터베이스 - 데이터베이스와 쿠키 설정

$ENSEMBL_USERDB_NAME,$ENSEMBL_USERDB_HOST,$ENSEMBL_USERDB_USER,$ENSEMBL_USERDB_ PASS의 값을 웹 사용자 데이터베이스에 대한 값들로 바꾼다. 이 데이터베이스는 갱신/삽입/삭제 권한을 가진 사용자들을 필요로 한다는 것을 기억한다. 또한 쿠키를 보호하기 위한 암호화키 - $ENSEMBL_ENCRYPT_0는 6자리의 16진수이어야 하며 $ENSEMBL_ENCRYPT_1, 2, 3은 각각 두 개의 영숫자의 문자가 포함되어야 한다 - 를 바꾸고 싶어 할 수도 있다. 쿠키를 바꾸는 사람들에 대해 특별히 신경 쓰지 않는 한 디폴트 값들이 적당하다.

⑥ 종 활성화

$ENSEMBL_SPECIES_ALIASES는 종 명칭에 대응하는 종 얼라이어스 해시를 가지고 있다. 예를 들면:

$ENSEMBL_SPECIES_ALIASES = 'human'=> 'Homo_sapiens',
'mouse'=> 'Mus_musculus',
'mosquito'=> 'Anopheles_gambiae',;

해시의 값(오른쪽의 종 명칭)은 이 웹사이트에서 유효한 종들의 목록을 이룬다. URL에서는 해시의 키들(왼쪽의 얼라이어스)이 긴 종 명칭들 대신에 쓰일 수 있다.

즉, http://my.ensembl.site/mouse/contigview는 http://my.ensembl.site/Mus_musculus/contigview와 동일하다.

한 종을 설정하기 위해서는 그것은 $ENSEMBL_SPECIES_ALIASES 해시에 한 값으로 존재하여야 하며 알맞게 이름지어진 .ini 파일이 conf 디렉토리에 있어야 한다.

$ENSEMBL_PERL_SPECIES - 이것은 역사적인 이유로 하여 http://my.ensembl.site/perl/configview 라는 URL의 특별한 얼라이어스가 되었다. - 를 바꾸고 싶어할 수도 있다. 이것은 나중에 multi-species의 쪽/함수의 디폴트 종들을 구성하는데 쓰일 것이다.

⑦ 임시 파일

배치될 수 있는 임시 파일 위치는 세 군데이다.

$ENSEMBL_TMP_DIR 임시파일의 일반 기억장소
$ENSEMBL_TMP_DIR_IMG 이미지 파일의 기억장소
$ENSEMBL_TMP_DIR_BLAST blast 파일의 기억장소

이 값들은 알맞은 파일시스템 경로로 지정되어야 한다.

어떤 임시 파일들은 웹 페이지로부터 URL에 의해 참조되어야 한다. 그러므로 위의 처음 두 tmp 디렉토리는 URL 얼라이어스들을 가지며 또한 SiteDefs.pm 안에서 배치된다. 이것을 편집해서는 안 된다.

$ENSEMBL_TMP_URL $ENSEMBL_TMP_DIR을 위한 URL 얼라이어스
$ENSEMBL_TMP_URL_IMG $ENSEMBL_TML_DIR_IMG를 위한 URL 얼라이어스

사용가능한 임시 파일 배치의 옵션은 두 가지가 더 있다.

$ENSEMBL_TMP_CREATE 지정되면 아파치 시동 시 서버는 배치되었던 어떤 임시 디렉토리라도 생성하려고 시도할 것이지만 그것은 이미 존재하지 않을 것이다. 이것은 또한 $ENSEMBL_USER.$ENSEMBL_GROUP의 소유권을 바꿀 것이다.

$ENSEMBL_TMP_DELETE 지정되면 아파치 시동 시 서버는 $ENSEMBL_TMP_DIR과 $ENSEMBL_TMP_DIR_IMG의 내용을 지우려고 시도할 것이다.

[Species_name].ini

종별 .ini 파일들의 형식은 표준적인 윈도우즈 .ini 파일들과 비슷하여 사각 괄호([ ])안의 헤더에 의해 구별되는 절로 나뉘어 지며, 각 절은 key = 값 항들로 이루어진다. 아파치 서버가 시동될 때 이 파일들은 파싱 되고 conf 디렉토리 안의 config.packed라는 파일 안에 저장된다. 이 config.packed 파일은 서버가 재 시동될 때마다 재생된다.

⑧ 데이터베이스 설정

[general] 절에서 ENSEMBL_DBUSER와 ENSEMBL_DBPASS의 값을 MySQL을 엑세스할 사용자와 비번으로 바꾼다. ENSEMBL_HOST와 ENSEMBL_HOST_PORT를 MySQL이 실행되고 있는 데이터베이스와 포트로 지정한다.

이것들은 디폴트 설정이므로 데이터베이스를 위한 절을 추가시킴으로써 오버라이트할 수 있다. 예를 들면, 다음과 같은 절을 추가하면 Help 데이터베이스를 위한 디폴트가 오버라이트될 것이다.

```
[ENSEMBL_HELP]
USER=mysqluser2
PASS=helppass
HOST=mysqlsever2
PORT=4444
```

⑨ 데이터베이스 이름

[database] 절에서 ENSEMBL_DB, ENSEMBL_EST, ENSEMBL_LITE 등의 값을 core, EST, 돌(lite) 등의 생성한 데이터베이스의 이름에 맞게 바꾼다.

DAS 프록시

외부 DAS 소스를 ensembl 설치에서 표시하고 싶은데 방화벽 뒤에 있다면 ENSEMBL_DAS_PROXY의 값을 설정해야한다. 프록시는 모든 종에 대해 같기 쉬우므로 이것은 아마도 DEFAULTS.ini에 설정되어 있을 것이다. 그 값은 웹 프록시 설정과 같을 것이다. 즉,

ENSEMBL_DAS_PROXY=http://webproxy.mycompany.com:80

MULTI.ini

종별 ini 파일에서처럼 MySQL 서버 접속을 위해 [general] 절을 설정한다. [database] 절에서 ENSEMBL_COMPARA와 ENSEMBL_MART의 값을 생성시킨 ensembl_compara 와 ensembl_ mart 데이터베이스의 이름에 맞게 바꾼다.

또한 로컬 GO 데이터베이스의 설치를 선택하면 다음의 항을 [databases] 절에 추가함으로서 그것을 설정할 수 있다.

ENSEMBL_GO = your_go_database_name, ENSEMBL_COMPARA, ENSEMBL_MART.

다른 species.ini 파일들에서와 같이 다음과 같은 절을 추가함으로써 특정 데이터베이스를 위한 데이터베이스 연결 설정을 오버라이트할 수 있다.

[ENSEMBL_MART]
USER=mysqluser2
PASS=dbpass
HOST=mysqlserver2

PORT=4444

라) Ensembl 웹 사이트 검색시 주의점

Ensembl 웹사이트는 AltaVista (http://www.altavista.com/) 검색 엔진을 사용한다. 이 소프트웨어는 사용자 인증을 요구하며 Ensembl 웹 코드의 일부분으로서 배포될 수 없다. AltaVista를 지역적으로 설치하여 사용할 경우 AltaVista 지원(ensembl@av.com)에 연락한다. AltaVista는 현재 Ensembl/AV 검색의 학계 사용자들을 위한 우선적 인증 조건(preferential licensing terms)을 도입할 것을 고려 중이다.

AltaVista에 대한 대안으로서 두 외부 공헌자인 Chen Peng과 Dyfed Lloyd-Evans가 MySQL4의 새로운 풀 텍스트 색인 기능을 사용한 시스템을 구축하였다. 이 코드와 사용 설명은 contrib/mysql_indexer 디렉토리에서 찾을 수 있다. (싱가포르의 Ensembl Fugu 사이트에서 운영되고 있다.)

기본적으로 Ensembl의 지역 설치는 데이터베이스에 대하여 SQL 검색을 하는 Unisearch라는 단순 검색 페이지를 사용한다. (이 방법은 좀 느린 경향이 있다.)

① 사이트 준비

서버 루트로 들어가서 다음의 명령어를 실행한다.

```
$ mkdir logs
$ chown -R $ENSEMBL_USER: $ENSEMBL_GROUP
```

여기서 $ENSEMBL_USER와 $ENSEMBL_GROUP은 SiteDef에 배치시킨 웹 서버의 사용자와 그룹이다.

이제 Ensembl 사이트가 시동할 준비가 되었다.

② 공지사항

아파치, mod_perl, MySQL, 모든 Ensembl 모듈에 대한 제시된 설정은 높은 수준의 보

안을 제공할 의도가 아니라는 것에 주의해야한다. 시스템 관리자가 타인들에게 제공하는 시스템을 점검하도록 강력하게 권하는 바이다. 특히 conf 디렉토리의 perl.startup 파일은 루트로서 실행될 것이므로 이 파일의 권한에 대해 주의해야한다. Ensembl 웹사이트 실행으로부터 생기는 데이터의 손상/손실에 대하여 책임지지 않는다.

마) 유지(Maintenance)

이 절은 Ensembl 웹사이트의 상시운영에 대해 개략적인 설명을 하고 있다.

① Ensembl 웹사이트 시동

1. MySQL이 실행되고 있는지 확인한다. 실행되고 있지 않으면 다음과 같은 명령으로 시동한다.

```
$/data/mysql/bin/safe_mysqld --user=mysqldba &
```

위의 명령의 /data/mysql과 mysqldba는 MySQL 설치시의 설치경로와 사용자로 바꾸어 주어야 한다.

2. 웹 서버를 시동한다.

```
$ /usr/local/apache/bin/httpd -d /usr/local/ensembl
```

여기서 /usr/local/ensembl은 서브-루트이다.

웹사이트를 시동시키기 전에 MySQL을 시동시키는 것이 중요하다. 사이트를 위한 시동 스크립트는 이용 가능한 데이터베이스들에 달려 있는데 사용 가능한 데이터의 목록을 컴파일하기 때문이다.

② Ensembl 웹사이트 정지

1. 웹 서버 정지시키기:

$ kill 'cat /usr/local/ensembl/logs/HOSTNAME.httpd.pid'

여기서 /usr/local/ensembl은 서버-루트이며 HOSTNAME은 서버를 실행시키고 있는 컴퓨터의 호스트명이다. 여기서 구두점은 인용점이 아니라 '이라는 것을 주의해야한다.

2. MySQL 정지시키기:

$ /data/mysql/bin/mysqladmin shutdown

바) 문제해결

가장 좋은 문제해결 방법은 아파치의 error_log를 보고 문제가 무엇인지 찾는 것이다. 다음 명령어에 의해서 서버-루트로부터 이를 실행할 수 있다.

$ tail -f logs/error_log

특히 파일 권한이나 Perl 모듈 분실 등의 오류를 찾도록 해준다.

이메일 오류가 생기면 그 경로를 통하여 어떤 추가 정보를 받을 것이다. 도움이 필요하면 helpdesk@ensembl.org로 이메일을 보낸다. 오류의 세밀 정보와 error_log에 관련된 절 등은 문제를 추적하는데 도움이 될 것이다.

나. 단백질 데이터베이스

본 연구를 통해 서비스되는 단백질 데이터베이스 시스템은 PDB, PIR, SWISS-PROT, ProFac, PhiPsi, CATH, SCOP의 7가지 데이터베이스들로 이 중 활용도가 높은 해외 데이터베이스들인 PDB, PIR, SWISS-PROT은 KISTI에서 개발한 KRISTAL 2000 검색 엔진을 활용하여 데이터베이스를 재구축함으로써 빠르고 정확한 검색이 가능하도록 개선하였다. 단백질 상과를 분류해주는 ProFac (Protein supterFamily Classification)과 PhiPsi데이터베이스는 CCBB에서 자체개발하여 서비스 되는 콘텐츠들이다.

본 연구에서는 앞서 언급한 데이터베이스 검색 시스템의 구축과 더불어 각 데

이터베이스의 이용실태를 파악함으로써 향후 양질의 검색 서비스 제공을 위한 데이터베이스 이용실태 통계 기능을 구축하였다.

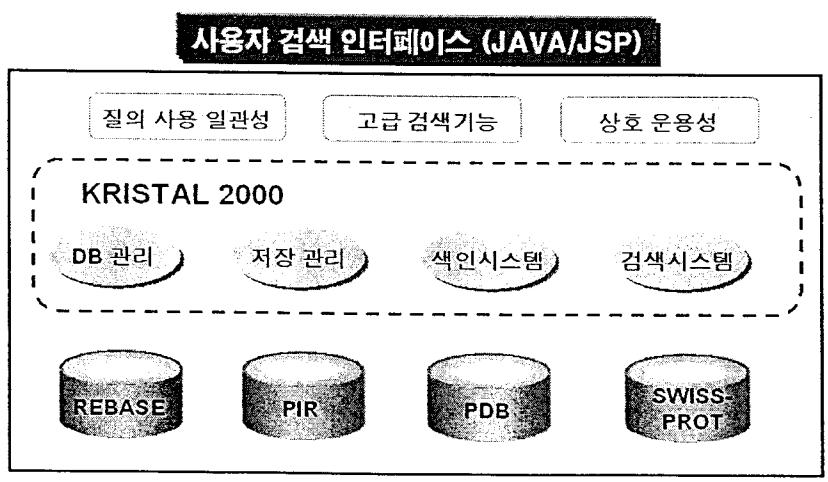
(1) KRISTAL 기반 단백질 데이터베이스 구축

기 구축된 생물 정보 데이터베이스를 본 기관에서 자체 개발한 KRISTAL 2000으로 교체하고 (<table 1-1> 참조) 이를 기반으로 한 검색 인터페이스를 개발함으로써 검색 시스템의 성능 향상을 도모하였다. <figure 1-17>는 본 연구에서의 데이터베이스 검색 시스템 구축의 전체적인 개요를 나타내고 있다.

- KRISTAL 2000을 통한 빠르고 정확한 검색
- 정보 검색 질의 사용의 일관성 확보
- 데이터베이스 검색 인터페이스를 PHP에서 Java/JSP로 conversion
- Oracle 데이터베이스와 MySQL 데이터베이스를 통합하여 사용할 수 있도록 두 데이터베이스간의 Connection Manager 개발

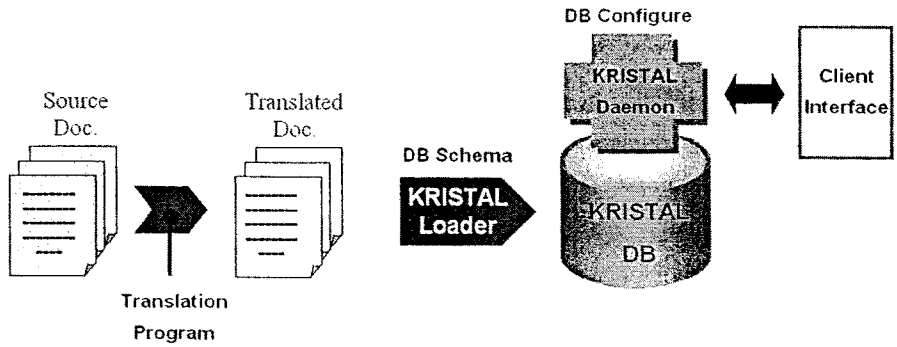
<table 1-1> KRISTAL 2000으로의 변환 대상 DB

	KRISTAL 2000 대상 DB	저장형태
단백질 DB	PDB	Oracle
	SWISS-PROT	Oracle
	PIR	Oracle



<figure 1-17> 데이터베이스 검색 시스템 구축 개요

<figure 1-18>은 KRISTAL 2000을 이용한 검색시스템 구축의 개요를 보이고 있다.



<figure 1-18> KRISTAL 2000을 이용한 검색시스템 구축 개요

그림에서 보는 바와 같이 KRISTAL 2000을 이용하여 생물정보 검색시스템을 구축하기 위해서는 먼저 데이터베이스를 이루는 원시데이터의 문서구조를 KRISTAL 2000이 다룰 수 있는 문서구조로 바꿔주는 변환 프로그램의 개발이 필요하다. 변환 프로그램을 통해 얻어진 변환문서(translated document)들에 대한 정보를 기반으로 데이터베이스의 스키마를 작성한 후 KRISTAL Loader를 이용해 변환문서들을 KRISTAL 데이터베이스를 적재하게 된다. 구축된 데이터베이스는 KRISTAL Deamon과 client interface를 통해 정보검색에 참여하게 된다.

(2) PDB

PDB (Protein Data Bank)는 생물학적인 단백질 3차원 고분자 결정을 위한 데이터베이스로서 1971년 부룩헤이번 국립 연구소(BNL)에 의해서 공개된 대표적인 단백질 공용 데이터베이스이다. PDB 데이터는 단백질 원자들의 3차원 공간상의 좌표, 서열정보, 실험정보 및 참조 정보 등이 포함되어있다.

(가) Database Construction

PDB 데이터베이스를 이루는 원시자료는 RSCB의 FTP 사이트인 ftp://ftp.rscb.org/pdb/data/structures/all/pdb/에서 *.ent.Z 파일을 다운받아 사용할 수 있다. PDB 1개의 데이터가 몇백KB의 데이터에 이르며 ID, Compound, Authors, Classification, Source, Primary Citation 필드에 대해서 KRISTAL 2000 검색을 위한 Index File을 생성하게 된다. <table 1-2>는 PDB 원시파일의 KRISTAL

2000으로의 Indexing 정보를 나타내고 있다.

<table 1-2> PDB data indexing

LABEL	SECTION NAME	INDEX TYPE	검색 구분
ID	ID	INDEX_AS_IS	ID=
HEADER	CLASSFICATION	INDEX_BY_TOKEN	HEADER=
AUTHOR	AUTHOR	INDEX_BY_TOKEN	AUTHOR=
CRYST1	CRYST1	INDEX_BY_TOKEN	CRYST1=
COMPND	COMPND	INDEX_BY_TOKEN	COMPND=
EXPDTA	EXPDTA	INDEX_BY_TOKEN	EXPDTA=
HETNAM	HETNAM	INDEX_BY_TOKEN	HETNAM=
JRNL	JRNL	INDEX_BY_TOKEN	JRNL=
KEYWDS	KEYWDS	INDEX_BY_TOKEN	KEYWDS=
LINK	LINK	INDEX_PROTEIN	LINK=
REVDAT	REVDAT	INDEX_PROTEIN	REVDAT=
SOURCE	SOURCE	INDEX_PROTEIN	SOURCE=
SEQRES	SEQRES	INDEX_PROTEIN	SEQRES=
TITLE	TITLE	INDEX_BY_TOKEN	TITLE=
SUMMARY	SUMMARY	DO_NOT_INDEX	SUMMARY=
TEXT	TEXT	DO_NOT_INDEX	TEXT=

<table 1-3>은 PDB 데이터베이스의 Schema를 보이고 있다.

<table 1-3> PDB Schema

```
// KRISTAL version
KRISTAL_VERSION="2000"
KRISTAL_DIRECTORY='/data7/pdb/K2000'
DATABASE_DIRECTORY='/data7/pdb/pdb/volumes'
DATABASE_GROUP_NAME="PDB"
DATABASE_COMPRESS=TRUE

CREATE_SCHEMA
{
    SECTION_DEFINITION
    {
    (1) LABEL="ID"
      SECTION_NAME=ID
      INDEX_TYPE="INDEX_AS_IS",
    (2) LABEL="HEADER"
      SECTION_NAME=CLASSIFICATION
      INDEX_TYPE="INDEX_BY_TOKEN",
    (3) LABEL="AUTHOR"
      SECTION_NAME=AUTHOR
      INDEX_TYPE="INDEX_BY_TOKEN",
    (4) LABEL="CRYST1"
      SECTION_NAME=CRYST1
      INDEX_TYPE="INDEX_BY_TOKEN",
    (5) LABEL="COMPND"
      SECTION_NAME=COMPND
      INDEX_TYPE="INDEX_BY_TOKEN",
```

```
    (6) LABEL="EXPDTA"
      SECTION_NAME=EXPDTA
      INDEX_TYPE="INDEX_BY_TOKEN",
    (7) LABEL="HETNAM"
      SECTION_NAME=HETNAM
      INDEX_TYPE="INDEX_BY_TOKEN",
    (8) LABEL="JRNL"
      SECTION_NAME=JRNL
      INDEX_TYPE="INDEX_BY_TOKEN",
    (9) LABEL="KEYWDS"
      SECTION_NAME=KEYWDS
      INDEX_TYPE="INDEX_BY_TOKEN",
    (10) LABEL="LINK"
      SECTION_NAME=LINK
      INDEX_TYPE="INDEX_BY_TOKEN",
    (11)  LABEL="REVDAT"
      SECTION_NAME=REVDAT
      INDEX_TYPE="INDEX_BY_TOKEN",
    (12)  LABEL="SOURCE"
      SECTION_NAME=SOURCE
      INDEX_TYPE="INDEX_BY_TOKEN",
    (13)  LABEL="SEQRES"
      SECTION_NAME=SEQRES
      INDEX_TYPE="INDEX_PROTEIN",
    (14)  LABEL="TITLE"
      SECTION_NAME=TITLE
      INDEX_TYPE="INDEX_BY_TOKEN",
    (15)  LABEL="SUMMARY"
      SECTION_NAME=SUMMARY
      INDEX_TYPE="DO_NOT_INDEX",
    (16)  LABEL="TEXT"
      SECTION_NAME=TEXT
      INDEX_TYPE="DO_NOT_INDEX"

  };

  PRIMARY_KEY_DEFINITION
  {
        PRIMARY_SECTIONS = (ID)
  };

  UNION_SECTION_DEFINITION
  {
    (1) LABEL="BASIC SEARCH FIELDS"
      SECTION_NAME=BASIC

SECTIONS=(ID,CLASSIFICATION,AUTHOR,CRYST1,COMPND,EXPDTA,HETNAM,JRNL,KEYWDS,LINK,REVDAT,S
OURCE,SEQRES,TITLE)

  };

}

DEFINE_DOCUMENT_STRUCTURE
{
 STRUCTURE_DEFINITION = ALL_DOCUMENTS {
  // 문서시작 태그 정의
  (1)          TAG_NAME="@PDB" ACTION=DISCARD NEW_DOCUMENT_FLAG=TRUE,
  // 섹션 태그 및 액션 정의
  (2)          TAG_NAME="ID=" ACTION=COPY SECTION_NAME=ID,
  (3)          TAG_NAME="HEADER=" ACTION=COPY SECTION_NAME=CLASSIFICATION,
  (4)          TAG_NAME="AUTHOR=" ACTION=COPY SECTION_NAME=AUTHOR,
```

```
  (5)                TAG_NAME="CRYST1=" ACTION=COPY SECTION_NAME=CRYST1,
  (6)                TAG_NAME="COMPND=" ACTION=COPY SECTION_NAME=COMPND,
  (7)                TAG_NAME="EXPDTA=" ACTION=COPY SECTION_NAME=EXPDTA,
  (8)                TAG_NAME="HETNAM=" ACTION=COPY SECTION_NAME=HETNAM,
  (9)                TAG_NAME="JRNL=" ACTION=COPY SECTION_NAME=JRNL,
  (10)      TAG_NAME="KEYWDS=" ACTION=COPY SECTION_NAME=KEYWDS,
  (11)      TAG_NAME="LINK=" ACTION=COPY SECTION_NAME=LINK,
  (12)      TAG_NAME="REVDAT=" ACTION=COPY SECTION_NAME=REVDAT,
  (13)      TAG_NAME="SOURCE=" ACTION=COPY SECTION_NAME=SOURCE,
  (14)      TAG_NAME="SEQRES=" ACTION=COPY SECTION_NAME=SEQRES,
  (15)      TAG_NAME="TITLE=" ACTION=COPY SECTION_NAME=TITLE,
  (16)      TAG_NAME="SUMMARY" ACTION=COPY SECTION_NAME=SUMMARY,
  (17)      TAG_NAME="TEXT=" ACTION=COPY SECTION_NAME=TEXT

  }
};

// 데이터베이스 생성
CREATE_DATABASE
{
// 데이터베이스 이름과 크기를 정의
          (1)          DATABASE_NAME=PDB1
                       DATABASE_SIZE=2000,
          (2)          DATABASE_NAME=PDB2
                       DATABASE_SIZE=2000,
          (3)          DATABASE_NAME=PDB3
                       DATABASE_SIZE=2000,
          (4)          DATABASE_NAME=PDB4
                       DATABASE_SIZE=2000,
          (5)          DATABASE_NAME=PDB5
                       DATABASE_SIZE=2000,
          (6)          DATABASE_NAME=PDB6
                       DATABASE_SIZE=2000,
          (7)          DATABASE_NAME=PDB7
                       DATABASE_SIZE=2000,
          (8)          DATABASE_NAME=PDB8
                       DATABASE_SIZE=2000,
          (9)          DATABASE_NAME=PDB9
                       DATABASE_SIZE=2000,
          (10)         DATABASE_NAME=PDB10
                       DATABASE_SIZE=2000
};

// 문서 그룹 정의
DEFINE_DOCUMENT_GROUP
{
    (1)   pdb1=('/data7/pdb/pdb/data/pdb_*1.kst'),
    (2)   pdb2=('/data7/pdb/pdb/data/pdb_*2.kst'),
    (3)   pdb3=('/data7/pdb/pdb/data/pdb_*3.kst'),
    (4)   pdb4=('/data7/pdb/pdb/data/pdb_*4.kst'),
    (5)   pdb5=('/data7/pdb/pdb/data/pdb_*5.kst'),
    (6)   pdb6=('/data7/pdb/pdb/data/pdb_*6.kst'),
    (7)   pdb7=('/data7/pdb/pdb/data/pdb_*7.kst'),
    (8)   pdb8=('/data7/pdb/pdb/data/pdb_*8.kst'),
    (9)   pdb9=('/data7/pdb/pdb/data/pdb_*9.kst'),
          (10)   pdb10=('/data7/pdb/pdb/data/pdb_*0.kst')
};

// 문서 적재
LOAD_DATABASE
{
          (1)          FROM=pdb1
```

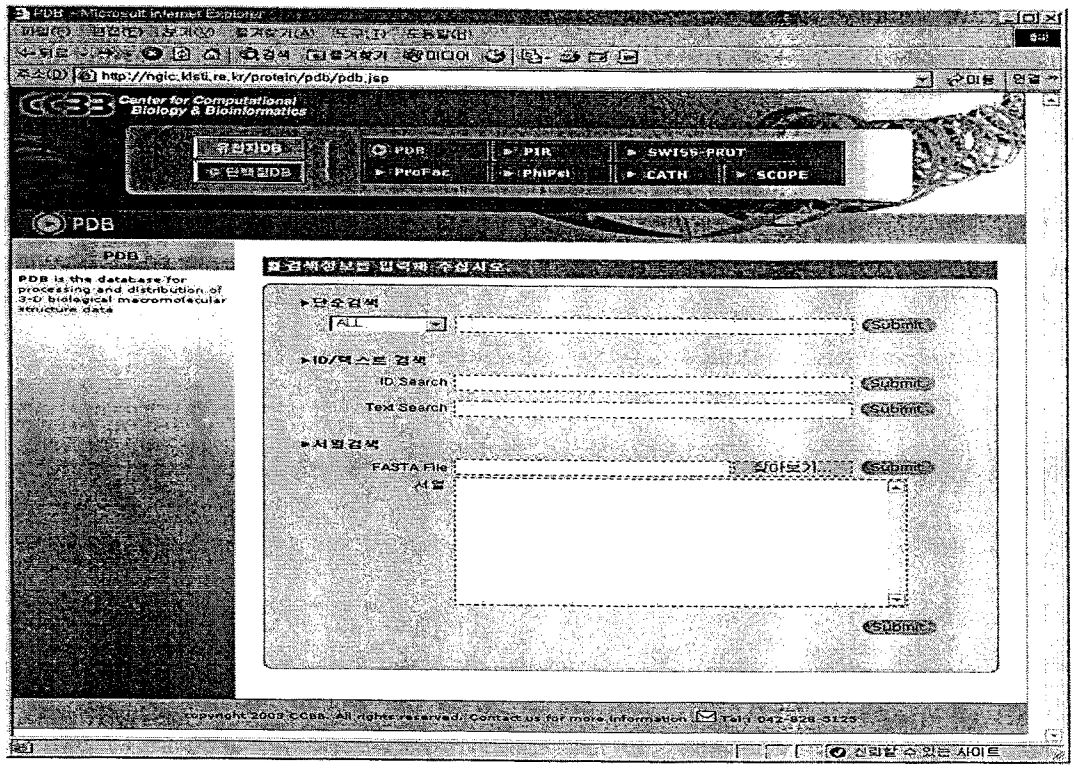
```
                TO=PDB1
                WITH=ALL_DOCUMENTS,
        (2)     FROM=pdb2
                TO=PDB2
                WITH=ALL_DOCUMENTS,
        (3)     FROM=pdb3
                TO=PDB3
                WITH=ALL_DOCUMENTS,
        (4)     FROM=pdb4
                TO=PDB4
                WITH=ALL_DOCUMENTS,
        (5)     FROM=pdb5
                TO=PDB5
                WITH=ALL_DOCUMENTS,
        (6)     FROM=pdb6
                TO=PDB6
                WITH=ALL_DOCUMENTS,
        (7)     FROM=pdb7
                TO=PDB7
                WITH=ALL_DOCUMENTS,
        (8)     FROM=pdb8
                TO=PDB8
                WITH=ALL_DOCUMENTS,
        (9)     FROM=pdb9
                TO=PDB9
                WITH=ALL_DOCUMENTS,
        (10)    FROM=pdb10
                TO=PDB10
                WITH=ALL_DOCUMENTS
};
```

(나) 검색 인터페이스

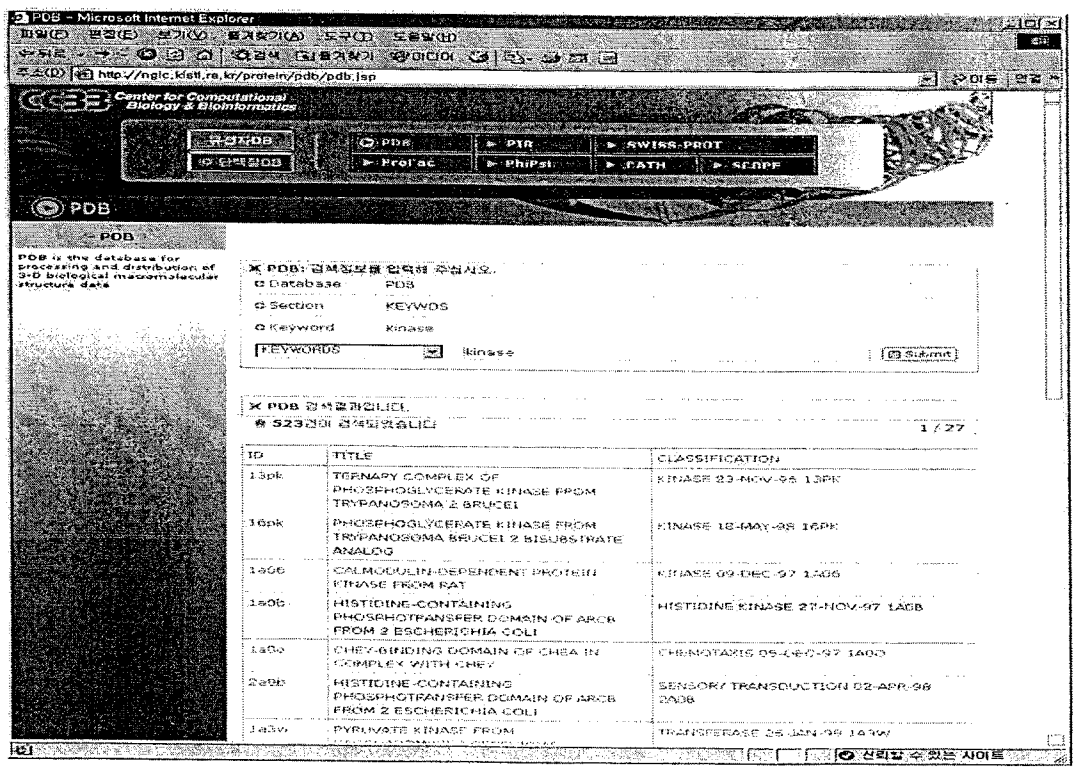
<figure 1-19>는 PDB의 검색 페이지를 보이고 있다. PDB 검색은 원하는 데이터 섹션 (ALL, ID, Author, Compound, Keyword, Source, Link)에 대한 keyword 검색의 단순검색, ID/Text 검색, 그리고 서열검색으로 구성된다.

<figure 1-20>은 단순검색에서 Keyword 'kinase"를 가지는 PDB 파일에 대한 검색 결과 페이지를 보이고 있다. 그림에서 알 수 있듯이 총 865건(2004년 5월말 기준)이 검색되었으며 이들에 대한 ID, Title, Classification의 간략 정보를 보이고 있다. 각 해당 ID를 클릭하면 원시 데이터에 대한 상세 정보를 살펴볼 수 있다.

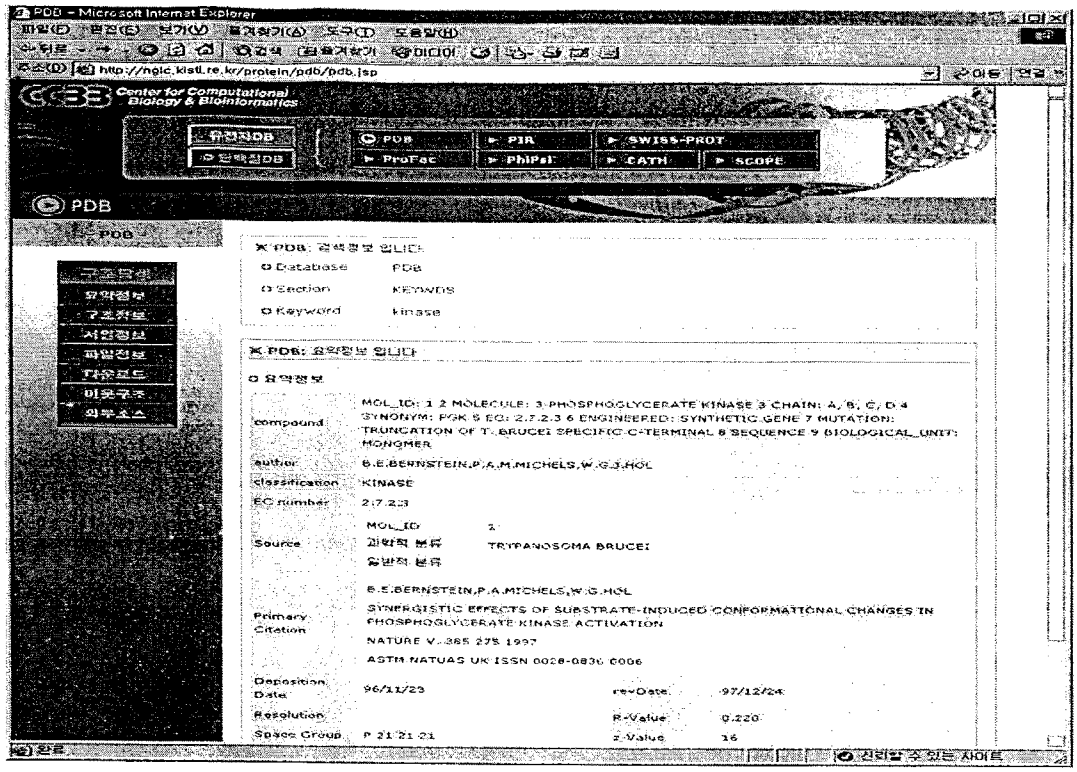
<figure 1-21>은 ID 13pk의 상세 정보를 보이고 있다. PDB 데이터의 상세정보는 해당 ID의 요약정보, 구조정보, 서열정보, 파일정보, 다운로드, 이웃구조, 외부소스들로 구성된다. <figure 1-22>은 ID 13pk의 서열정보를 보이고 있다.



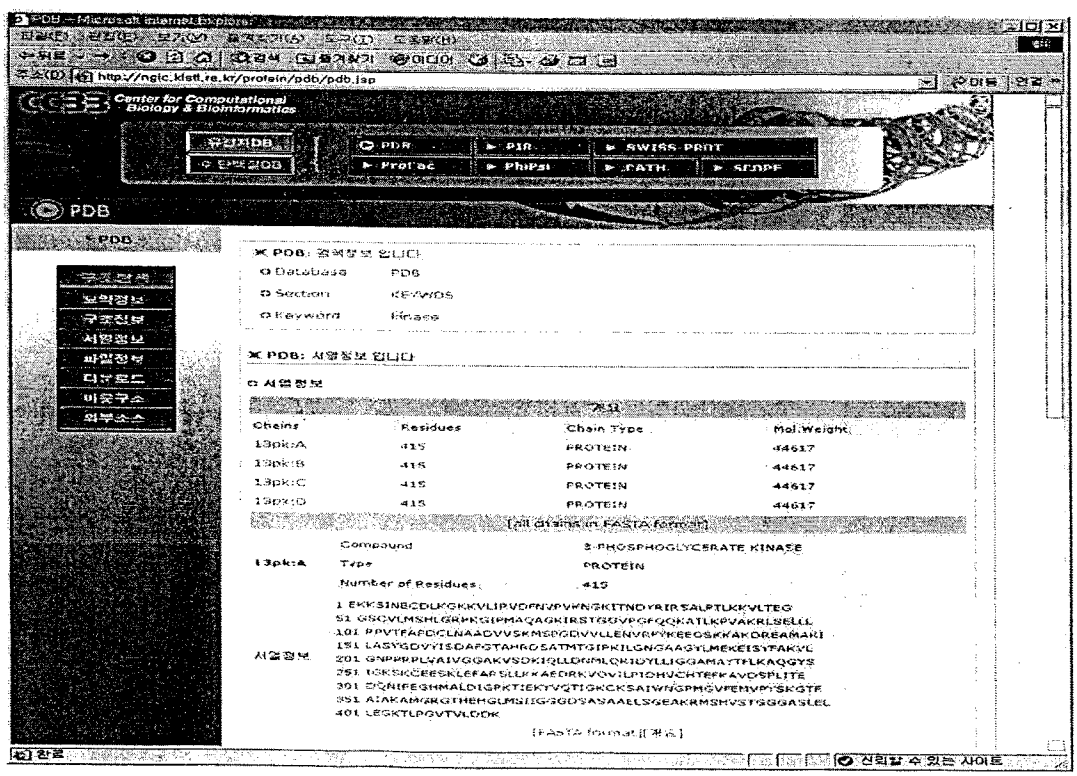
<figure 1-19> PDB: 검색 페이지



<figure 1-20> PDB: 검색 결과 - 간략 정보 보기



<figure 1-21> PDB: 상세 정보 - ID 13pk



<figrue 2-22> PDB: 상세정보- Sequence

(3) PIR

PIR (Ptotein Information Resource)는 조지타운 의과대학 (GUMC)에서 운영되는 공용 단백질 데이터베이스로써 2004년 5월 기준으로 283,416개의 단백질 서열정보와 단백질의 기능과 관련된 taxonomy를 제공하고 있다.

(가) Database Construction

<table 1-4>는 PIR-PSD Database 원시자료에 대한 내용을 보여 주고 있으며 ftp://ftp.ebi.ac.uk/pub/databases/pir/ 에 접속하여 pir1.dat.Z ~ pir4.dat.Z 파일을 다운 받을 수 있다. <table 1-5>는 <table 1-4>에서 보여진 PIR Data에 대한 Sequence Data를 보여준다.

<table 1-4> PIR-PSD 원시 데이터

>P1;CCHU
cytochrome c [validated] - human
C;Species: Homo sapiens (man)
C;Date: 24-Apr-1984 #sequence_revision 30-Sep-1991 #text_change 28-Jul-2000
C;Accession: A31764; A05676; I55192; A00001
R;Evans, M.J.; Scarpulla, R.C.
Proc. Natl. Acad. Sci. U.S.A. 85, 9625-9629, 1988
A;Title: The human somatic cytochrome c gene: two classes of processed pseudogenes demarcate a period of rapid molecular evolution.
A;Reference number: A31764; MUID:89071748; PMID:2849112
A;Accession: A31764
A;Molecule type: DNA
A;Residues: 1-105 <EVA>
A;Cross-references: GB:M22877; NID:g181241; PIDN:AAA35732.1; PID:g181242 R;Matsubara, H.; Smith, E.L.
J. Biol. Chem. 238, 2732-2753, 1963
C;Superfamily: cytochrome c; cytochrome c homology
C;Keywords: acetylated amino end; chromoprotein; electron transfer; heme; iron; metalloprotein; mitochondrion; oxidative phosphorylation; polymorphism; respiratory chain
F;2-105/Product: cytochrome c #status experimental <MAT>
F;5-99/Domain: cytochrome c homology <CYC>
F;2/Modified site: acetylated amino end (Gly) (in mature form) #status experimental
F;15,18/Binding site: heme (Cys) (covalent) #status experimental
F;19,81/Binding site: heme iron (His, Met) (axial ligands) #status predicted

<table 1-5> Sequence information of Entry 'CCHU'

```
>P1;CCHU
cytochrome c [validated] - human
MGDVEKGKKIFIMKCSQCHTVEKGGKHKTGPNLHGLFGRKTGQAPGYSYTAANKNKGIIWGEDTLMEYLENPKKY
IPGTKMIFVGIKKKEERADLIAYLKKATNE*
```

<table 1-6>은 PIR 원시파일에 나타나는 각 레코드와 그 의미 및 발생 횟수를 보여준다.

<table 1-6> The records, sub-records and their occurrences and contents in an PIR entry

Record	Occurrence	Content	Sub-record	Occurrence
Entry	Once	*Identification*		Once
		Type	#type	Optional
TITLE	Once	*Protein Name*		Once
ALTERNATE_ NAMES	Optional	*Alternative name of the protein*		Optional
CONTAINS	Optional	*Contains*		Optional
ORGANISM	Once	*Organism Name*		Optional
		Formal Name	#formal_name	Optional
		Common Name	#common_name	Optional
		Variety	#variety	Optional
		Note	#note	Optional+
DATE	Optional	*Creation Date*		Optional
		Last Sequence Modification	#sequence_revision	Optional
		Last Text Modification	#text_change	Optional
ACCESSION	Optional	Accession Number(s)		Once
REFERENCE	Once+	*Reference Number*		Once
		Authors	#authors	Once
		Journal	#journal	Optional
		Book	#book	Optional
		Submission	#submission	Optional
		Title	#title	Optional
		DB Cross-Reference	#cross-references	Optional
		Contents	#contents	Optional
		Notes	#notes	Optional
		Accession	#accession	Optional
		- Status	##status	Optional
		- Molecule Type	##molecule_type	Optional
		- Residues	##residues	Optional

		- Label	##label	Optional
		- Cross-Reference	##cross-reference	Optional
		- Experimental Source	##experimental_source	Optional
		- Genetics	##genetics	Optional
		- Note	##note	Optional+
COMMENT	Optional+	*Comment(s)*		Once
		Label	#label	Optional
		Link	#link	Optional
GENETICS	Optional+	*Gene Expression of the PRT*		Optional
		Gene	#gene	Optional
		Map Position	#map_position	Optional
		Genome	#genome	Optional
		Gene Origin	#gene_origin	Optional
		Genetic Code	#genetic_code	Optional
		Start Codon	#start_codon	Optional
		Intron	#intron	Optional
		- Status	##status	Optional
		Other Products	#other_products	Optional
		Note	#note	Optional+
COMPLEX	Optional	*Protein Complex*		Once
FUNCTION	Optional	*Function of the Protein*		Optional+
		Description	#description	Optional
		Pathway	#pathway	Optional
		Note	#note	Optional+
CLASSIFICTATION	Optional	*Classification of the Protein*		
		Superfamily	#superfamily	Optional
		Group	#group	Optional
KEYWORD	Optional	*keywords*		Once
FEATURE	Optional+	*Feature of the Protein*		
		Product	#product	Optional
		Domain	#domain	Optional
		Region	#region	Optional
		Bond	#bond	Optional
		Cleavage Site	#cleavage_site	Optional
		Modified Site	#modified_site	Optional
		Site	#site	Optional
		Binding Site	#binding_site	Optional
		Active Site	#active_site	Optional
SUMMARY	Once	*Summary of the Sequence*		
		Length	#length	Once
		Molecular Weight	#molecular_weight	Optional
		Checksum	#checksum	
SEQUENCE	Once	*Sequence Data*		Once

<table 1-7>은 PIR 원시데이터의 KRISTAL 2000로의 Indexing 정보이다.

<table 1-7> PIR Indexing

LABEL	SECTION NAME	INDEX TYPE	검색 구분
Entry_Name	Entry_Name	INDEX_AS_IS	ENTRY_NAME=
Accession_number	Accession_number	INDEX_BY_TOKEN	ACCESSIONS=
ALTERNATE_NAMES	ALTERNATE_NAMES	DO_NOT_INDEX	ALTERNATE_NAMES=
Classification	Classification	INDEX_BY_TOKEN	CLASSIFICATION=
COMMENT	COMMENT	DO_NOT_INDEX	COMMENT=
COMPLEX	COMPLEX	DO_NOT_INDEX	COMPLEX=
CONTAINS	CONTAINS	DO_NOT_INDEX	CONTAINS=
Date	Date	INDEX_BY_TOKEN	DATE=
Identification	Identification	INDEX_PROTEIN	ENTRY=
Features	Features	INDEX_BY_TOKEN	FEATURE=
FUNCTION	FUNCTION	DO_NOT_INDEX	FUNCTION=
Genetics	Genetics	INDEX_BY_TOKEN	GENETICS=
Keywords	Keywords	INDEX_BY_TOKEN	KEYWORDS=
Organism_Name	Organism_Name	INDEX_BY_TOKEN	ORGANISM=
Protein_Name	Protein_Name	INDEX_BY_TOKEN	TITLE=
References	References	INDEX_BY_TOKEN	REFERENCE=
Sequence	Sequence	INDEX_PROTEIN	SEQUENCE=
Summary	Summary	DO_NOT_INDEX	SUMMARY=
TEXT	TEXT	INDEX_BY_TOKEN	TEXT=

<table 1-8>은 KRISTAL로 재구축된 PIR 데이터베이스의 Schema를 보이고 있다.

<table 1-8> PIR Indexing

```
// KRISTAL version
KRISTAL_VERSION="2000"
KRISTAL_DIRECTORY='/data5/pir/K2000'
DATABASE_DIRECTORY='/data5/pir/pir/volumes'
DATABASE_GROUP_NAME="PIR"
DATABASE_COMPRESS=TRUE

CREATE_SCHEMA
{
    SECTION_DEFINITION
    {
    (1)  LABEL="Entry_Name"
         SECTION_NAME=Entry_Name
         INDEX_TYPE="INDEX_AS_IS",
    (2)  LABEL="Accession_number"
         SECTION_NAME=Accession_number
         INDEX_TYPE="INDEX_BY_TOKEN",
    (3)  LABEL="ALTERNATE_NAMES"
         SECTION_NAME=ALTERNATE_NAMES
```

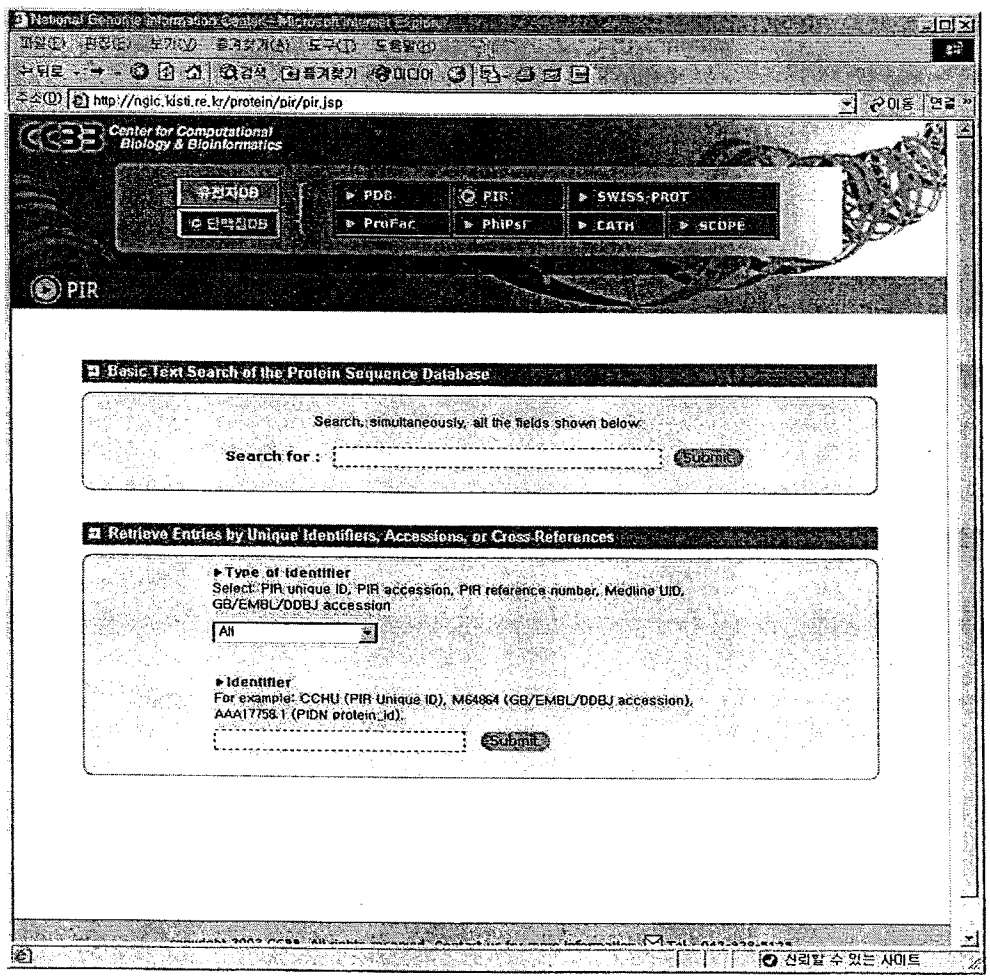
```
    INDEX_TYPE="DO_NOT_INDEX",
(4)  LABEL="Classification"
    SECTION_NAME=Classification
    INDEX_TYPE="INDEX_BY_TOKEN",
(5)  LABEL="COMMENT"
    SECTION_NAME=COMMENT
    INDEX_TYPE="DO_NOT_INDEX",
(6)  LABEL="COMPLEX"
    SECTION_NAME=COMPLEX
    INDEX_TYPE="DO_NOT_INDEX",
(7)  LABEL="CONTAINS"
    SECTION_NAME=CONTAINS
    INDEX_TYPE="DO_NOT_INDEX",
(8)  LABEL="Date"
    SECTION_NAME=Date
    INDEX_TYPE="INDEX_BY_TOKEN",
(9)  LABEL="Identification"
    SECTION_NAME=Identification
    INDEX_TYPE="INDEX_BY_TOKEN",
(10)  LABEL="Features"
    SECTION_NAME=Features
    INDEX_TYPE="INDEX_BY_TOKEN",
(11)  LABEL="FUNCTION"
    SECTION_NAME=FUNCTION
    INDEX_TYPE="DO_NOT_INDEX",
(12)  LABEL="Genetics"
    SECTION_NAME=Genetics
    INDEX_TYPE="INDEX_BY_TOKEN",
(13)  LABEL="Keywords"
    SECTION_NAME=Keywords
    INDEX_TYPE="INDEX_BY_TOKEN",
(14)  LABEL="Organism_Name"
    SECTION_NAME=Organism_Name
    INDEX_TYPE="INDEX_BY_TOKEN",
(15)  LABEL="Protein_Name"
    SECTION_NAME=Protein_Name
    INDEX_TYPE="INDEX_BY_TOKEN",
(16)  LABEL="References"
    SECTION_NAME=References
    INDEX_TYPE="INDEX_BY_TOKEN",
(17)  LABEL="Sequence"
    SECTION_NAME=Sequence
    INDEX_TYPE="INDEX_PROTEIN",
(18)  LABEL="Summary"
    SECTION_NAME=Summary
    INDEX_TYPE="DO_NOT_INDEX",
(19)  LABEL="TEXT"
    SECTION_NAME=TEXT
    INDEX_TYPE="INDEX_BY_TOKEN"
}
```

```
  PRIMARY_KEY_DEFINITION
    {
        PRIMARY_SECTIONS = (Entry_Name)
    }

  UNION_SECTION_DEFINITION
  {
    (1) LABEL="BASIC SEARCH FIELDS"
      SECTION_NAME=BASIC
      SECTIONS=(TEXT)
  }
};

DEFINE_DOCUMENT_STRUCTURE
{
 STRUCTURE_DEFINITION = ALL_DOCUMENTS {
  // 문서시작 태그 정의
  (1)   TAG_NAME="@PIR" ACTION=DISCARD NEW_DOCUMENT_FLAG=TRUE,
  // 섹션 태그및 액션 정의
  (2)   TAG_NAME="ENTRY_NAME=" ACTION=COPY SECTION_NAME=Entry_Name,
  (3)   TAG_NAME="ACCESSIONS=" ACTION=COPY SECTION_NAME=Accession_number,
  (4)   TAG_NAME="ALTERNATE_NAMES=" ACTION=COPY SECTION_NAME=ALTERNATE_NAMES,
  (5)   TAG_NAME="CLASSIFICATION=" ACTION=COPY SECTION_NAME=Classification,
  (6)   TAG_NAME="COMMENT=" ACTION=COPY SECTION_NAME=COMMENT,
  (7)   TAG_NAME="COMPLEX=" ACTION=COPY SECTION_NAME=COMPLEX,
  (8)   TAG_NAME="CONTAINS=" ACTION=COPY SECTION_NAME=CONTAINS,
  (9)   TAG_NAME="DATE=" ACTION=COPY SECTION_NAME=Date,
  (10)  TAG_NAME="ENTRY=" ACTION=COPY SECTION_NAME=Identification,
  (11)  TAG_NAME="FEATURE=" ACTION=COPY SECTION_NAME=Features,
  (12)  TAG_NAME="FUNCTION=" ACTION=COPY SECTION_NAME=FUNCTION,
  (13)  TAG_NAME="GENETICS=" ACTION=COPY SECTION_NAME=Genetics,
  (14)  TAG_NAME="KEYWORDS=" ACTION=COPY SECTION_NAME=Keywords,
  (15)  TAG_NAME="ORGANISM=" ACTION=COPY SECTION_NAME=Organism_Name,
  (16)  TAG_NAME="TITLE=" ACTION=COPY SECTION_NAME=Protein_Name,
  (17)  TAG_NAME="REFERENCE=" ACTION=COPY SECTION_NAME=References,
  (18)  TAG_NAME="SEQUENCE=" ACTION=COPY SECTION_NAME=Sequence,
  (19)  TAG_NAME="SUMMARY=" ACTION=COPY SECTION_NAME=Summary,
  (20)  TAG_NAME="TEXT=" ACTION=COPY SECTION_NAME=TEXT
  }
};

// 데이터베이스 생성
CREATE_DATABASE
{
// 데이터베이스 이름과 크기를 정의
        (1)     DATABASE_NAME=PIR1
                DATABASE_SIZE=2000,
        (2)     DATABASE_NAME=PIR2
                DATABASE_SIZE=2000,
        (3)     DATABASE_NAME=PIR3
                DATABASE_SIZE=2000,
```

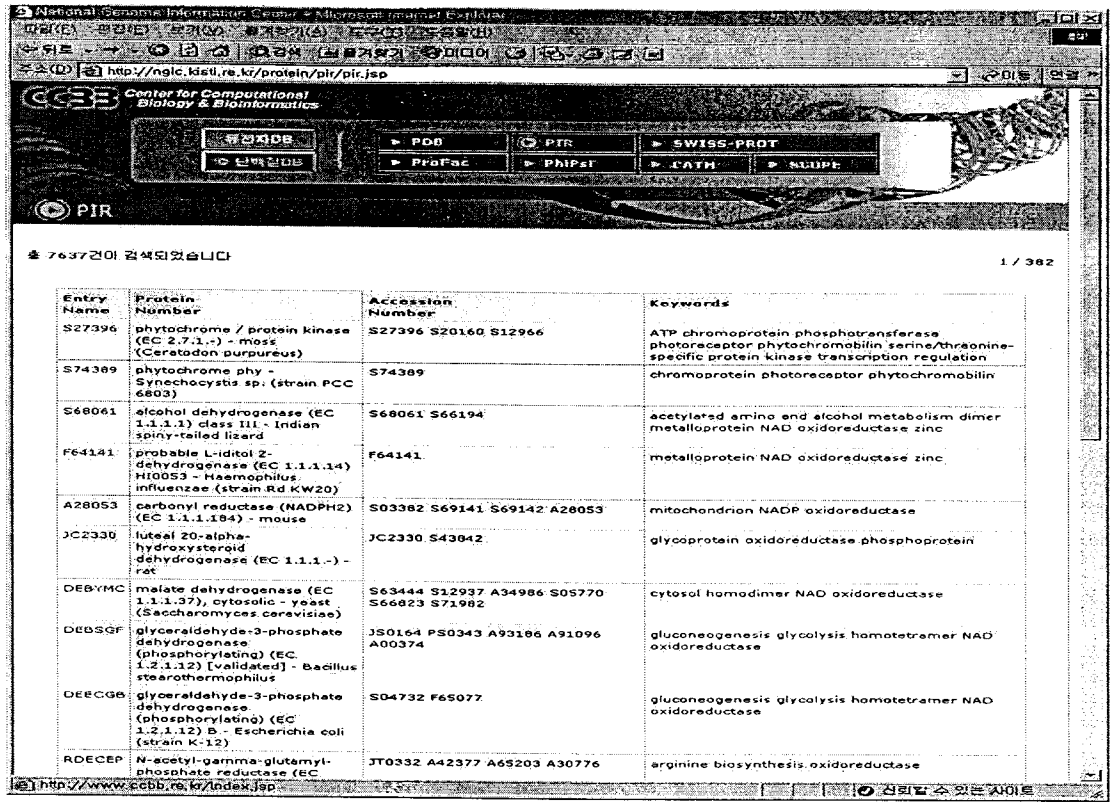
```
        (4)     DATABASE_NAME=PIR4
                DATABASE_SIZE=2000,
        (5)     DATABASE_NAME=PIR5
                DATABASE_SIZE=2000,
        (6)     DATABASE_NAME=PIR6
                DATABASE_SIZE=2000,
        (7)     DATABASE_NAME=PIR7
                DATABASE_SIZE=2000
}

// 문서 그룹 정의
DEFINE_DOCUMENT_GROUP
{
        (1)  pir1=('/data5/pir/pir/data/pir_1.kst'),
        (2)  pir2=('/data5/pir/pir/data/pir_2.kst'),
        (3)  pir3=('/data5/pir/pir/data/pir_3.kst'),
        (4)  pir4=('/data5/pir/pir/data/pir_4.kst'),
        (5)  pir5=('/data5/pir/pir/data/pir_5.kst'),
        (6)  pir6=('/data5/pir/pir/data/pir_6.kst'),
        (7)  pir7=('/data5/pir/pir/data/pir_7.kst')
}

// 문서 적재
LOAD_DATABASE
{
        (1) FROM=pir1
                TO=PIR1
                WITH=ALL_DOCUMENTS,
        (2) FROM=pir2
                TO=PIR2
                WITH=ALL_DOCUMENTS,
        (3) FROM=pir3
                TO=PIR3
                WITH=ALL_DOCUMENTS,
        (4) FROM=pir4
                TO=PIR4
                WITH=ALL_DOCUMENTS,
        (5) FROM=pir5
                TO=PIR5
                WITH=ALL_DOCUMENTS,
        (6) FROM=pir6
                TO=PIR6
                WITH=ALL_DOCUMENTS,
        (7) FROM=pir7
                TO=PIR7
                WITH=ALL_DOCUMENTS
};
END
```

(나) 검색 인터페이스

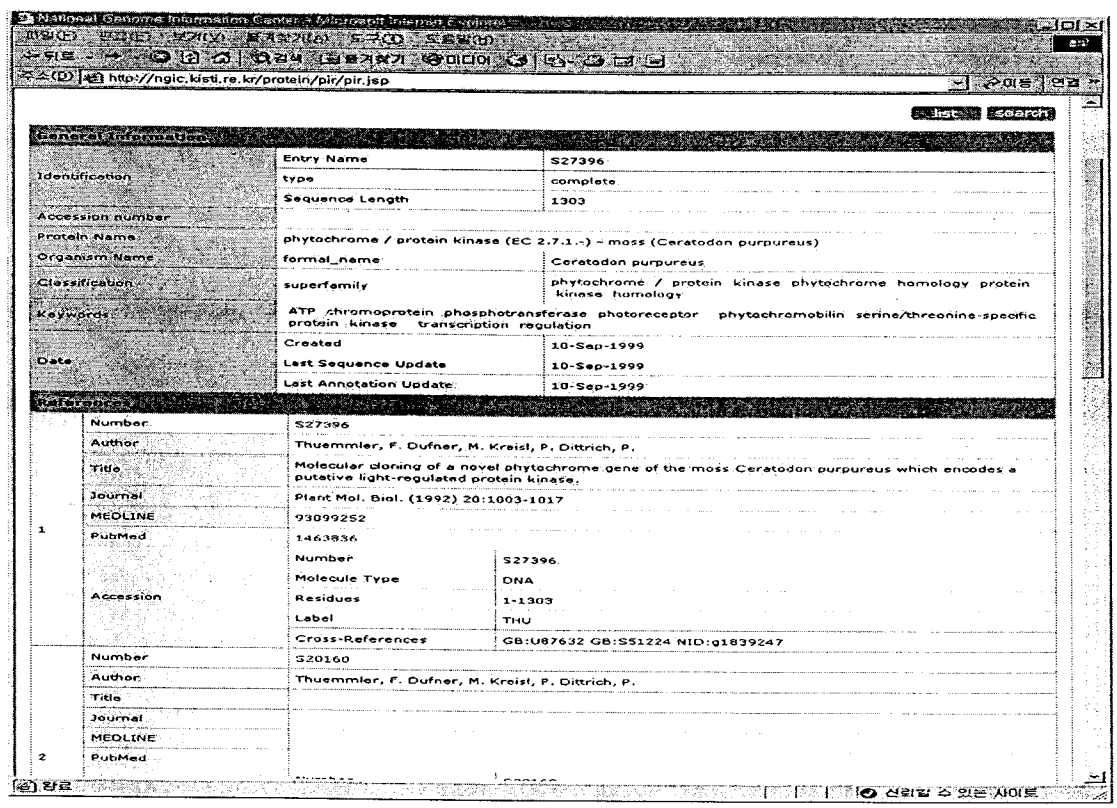
<figure 1-23>은 PIR 데이터베이스의 검색 페이지를 보이고 있다. PIR 데이터베이스의 검색은 모든 검색 필드에 대한 keyword검색 기능과 KRISTAL에 의해 나누어진 각 Section (All, Entry Name, Protein Name, Accession Number, Keyword)에 대한 keyword 검색 기능을 제공한다. <figure 1-24>는 모든 Section에 대한 keyword 'kinase'의 검색 결과 페이지를 보이고 있다. 그림에서 알 수 있듯이 총 7637개의 entry가 검색되었으며 각 entry의 name, protein name, accession number, kyeowrds의 간략 정보를 보이고 있다. 각 entry의 상세 정보는 해당 entry name을 클릭함으로써 얻을 수 있다. <figure 1-25>는 entry 'S27396'의 상세정보를 보이고 있다.



<figure 1-23> PIR: 검색 페이지



<figure 1-24> PIR: 검색 결과 - 간략 정보 보기



<figure 1-25> PIR: 상세 정보 보기 - Entry 'S27396'

(4) SWISS-PROT

SWISS-PROT 은 1986년 Swiss Institute for Bioinformatics (SIB)와 European Bioinformatics Institute (EBI)에서 공동으로 운영되는 공용 단백질 데이터베이스로서, 높은 수준의 annotation과 자료들간의 중복이 거의 없는 단백질 정보를 제공한다.

(가) Database Construction

SWISS-PROT Database 원시자료는 EBI의 FTP 사이트인 ftp://ftp.ebi.ac.uk/pub/databases/swissprot/release/ 에 접속하여 sprot##.dat.Z 파일을 다운받을 수 있다. <table 1-9>는 SWISS-PROT 원시 자료의 예를 보이고 있다.

<table 1-9> SWISS-PROT 원시 자료

```
ID   108_LYCES      STANDARD;      PRT;   102 AA.
AC   Q43495;
DT   15-JUL-1999 (Rel. 38, Created)
DT   15-JUL-1999 (Rel. 38, Last sequence update)
DT   15-JUL-1999 (Rel. 38, Last annotation update)
DE   Protein 108 precursor.
OS   Lycopersicon esculentum (Tomato).
OC   Eukaryota; Viridiplantae; Streptophyta; Embryophyta; Tracheophyta;
OX   NCBI_TaxID=4081;
RN   [1]
RP   SEQUENCE FROM N.A.
RC   STRAIN=cv. VF36; TISSUE=Anther;
RX   MEDLINE=94143497; PubMed=8310077;
RA   Chen R., Smith A.G.;
RT   "Nucleotide sequence of a stamen- and tapetum-specific gene from
RT   Lycopersicon esculentum.";
RL   Plant Physiol. 101:1413-1413(1993).
CC   -!- TISSUE SPECIFICITY: STAMEN- AND TAPETUM-SPECIFIC.
DR   EMBL; Z14088; CAA78466.1; -.
DR   SMART; SM00499; AAI; 1.
KW   Signal.
FT   SIGNAL        1     30       POTENTIAL.
FT   CHAIN         31    102      PROTEIN 108.
FT   DISULFID      41    77       BY SIMILARITY.
FT   DISULFID      79    99       BY SIMILARITY.
SQ   SEQUENCE   296 AA;  34077 MW;  B0D7CD175C7A3625 CRC64;
     FNSNMLRGSV CEEDVSLMTS IDNMIEEIDF YEKEIYKGSH SGGVIKGMDY DLEDDENDED
     EMTEQMVEEV ADHITQDMID EVAHHVLDNI THDMAHMEEI VHGLSGDVTQ
     IKEIVQKVNV  AVEKVKHIVE TEETQKTVEP EQIEETQNTV EPEQTEETQK TVEPEQTEET
     QNTVEPEQIE  ETQKTVEPEQ TEEAQKTVEP EQTEETQKTV EPEQTEETQK TVEPEQTEET
     QKTVEPEQTE ETQKTVEPEQ TEETQKTVEP EQTEETQKTV EPEQTEETQN TVEPE?TQET
```

<table 1-10>은 SWISS-PROT 원시파일(entry)에서 나타나는 레코드의 의미, 순서 및 발생횟수를 나타낸다.

<table 1-10> The records, content and occurrence in an SWISS-PROT entry

Record	Content	Occurrence in an entry
ID	Identification (대문자, 10char. 이하)	Once; start of the entry
AC	Accession Number(s)	Once or more
DT	Date	Three times
DE	Description	Once or more
GN	Gene Name(s)	Optional
OS	Organism Species	Once or more
OG	Organelle	Optional
OC	Organism Classification	Once or more
OX	Taxonomy Cross-reference(s)	Once or more
RN	Reference Number	Once or more
RP	Reference Position	Once or more
RC	Reference Comment(s)	Optional
RX	Reference Cross-Reference(s)	Optional
RA	Reference authors	Once or more
RT	Reference Title	Optional
RL	Reference Location	Once or more
CC	Comments or Notes	Optional
DR	Database Cross-references	Optional
KW	Keywords	Optional
FT	Feature Table Data	Optional
SQ	Sequence Header	Once
	(blanks)Sequence Data	Once or more
//	Termination line	Once; end of the entry

SWISS-PROT Database의 원시자료에는 많은 양의 필드를 가지고 있지만 사용자가 많이 이용하는 필드를 선택하여 KRITSTAL 2000에 Index File 생성한다. Swiss-prot ID, Access Number, Protein Name, Source Organism, Reference ID, Amino Sequence, Description Text 필드의 데이터를 추출하여 Database를 구축, 검색에 이용할 수 있게 한다. <table 1-11>은 SWISS-PROT 데이터베이스의 KRISTAL 2000로의 Indexing 정보를 나타내고 있다.

<table 1-11> SWISS-PROT Indexing

LABEL	SECTION NAM	INDEX_TYPE	구분
Entry_Name	Entry_Nam	INDEX_AS_IS	EID=
Identification	Identification	INDEX_BY_TOKEN	ID=
Accession_number	Accession_numbe	INDEX_AS_IS	AC=
Date	Date	DO_NOT_INDEX	DT=
Description	Description	INDEX_BY_TOKEN	DE=
Gene_Name	Gene_Name	DO_NOT_INDEX	GN=
Organism_Species	Organism_Species	INDEX_BY_TOKEN	OS=
Organism_Taxonomy_Cross_Reference	Organism_Taxonomy_Cross_Reference	DO_NOT_INDEX	OX=
Organism_Classification	Organism_Classification	DO_NOT_INDEX	OC=
Reference	Reference	INDEX_BY_TOKEN	RX=
Comments	Comment	DO_NOT_INDEX	CC=
Database_Cross_Reference	Database_Cross_Reference	INDEX_BY_TOKEN	DR=
Keywords	Keywords	INDEX_BY_TOKEN	KW=
Features	Feature	DO_NOT_INDEX	FT=
Sequence_Information	Sequence_Information	INDEX_PROTEIN	SQ=
TEXT	TEXT	INDEX_BY_TOKEN	TEXT=

<table 1-12>는 KRISTAL 2000로 재구축된 SWISS-PROT 데이터베이스 Schema를 보이고 있다.

<table 1-12> SWISS-PROT Schema

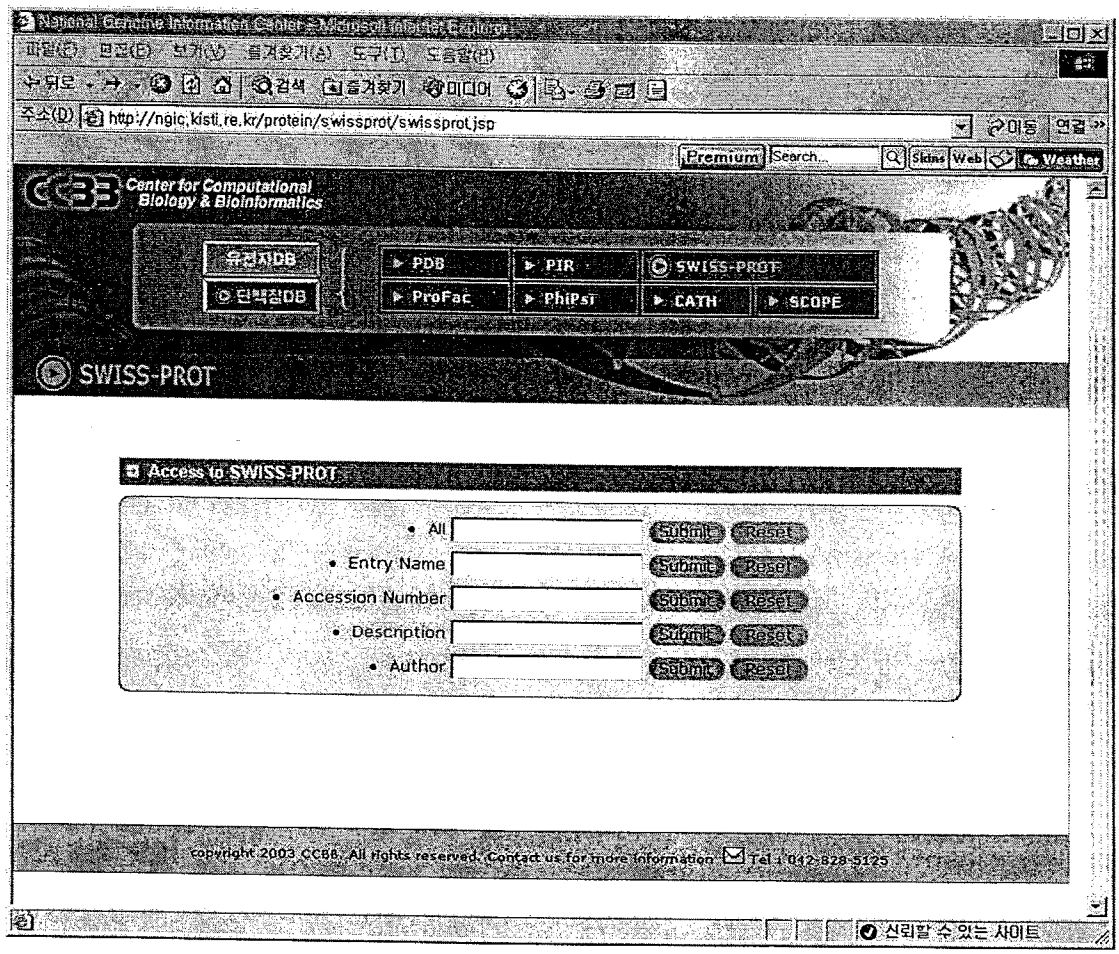
```
// KRISTAL version
KRISTAL_VERSION="2000"
KRISTAL_DIRECTORY='/data4/swiss/K2000'
DATABASE_DIRECTORY='/data4/swiss/swissprot/volumes'
DATABASE_GROUP_NAME="SWISSPROT"
DATABASE_COMPRESS=TRUE

CREATE_SCHEMA
{
    SECTION_DEFINITION
    {
    (1) LABEL="Entry_Name"
      SECTION_NAME=Entry_Name
      INDEX_TYPE="INDEX_AS_IS",
    (2) LABEL="Identification"
      SECTION_NAME=Identification
      INDEX_TYPE="INDEX_BY_TOKEN",
```

```
  (3) LABEL="Accession_number"
    SECTION_NAME=Accession_number
    INDEX_TYPE="INDEX_AS_IS",
  (4) LABEL="Date"
    SECTION_NAME=Date
    INDEX_TYPE="DO_NOT_INDEX",
  (5) LABEL="Description"
    SECTION_NAME=Description
    INDEX_TYPE="INDEX_BY_TOKEN",
  (6) LABEL="Gene_Name"
    SECTION_NAME=Gene_Name
    INDEX_TYPE="DO_NOT_INDEX",
  (7) LABEL="Organism_Species"
    SECTION_NAME=Organism_Species
    INDEX_TYPE="INDEX_BY_TOKEN",
  (8) LABEL="Organism_Taxonomy_Cross_Reference"
    SECTION_NAME=Organism_Taxonomy_Cross_Reference
    INDEX_TYPE="DO_NOT_INDEX",
  (9) LABEL="Organism_Classification"
    SECTION_NAME=Organism_Classification
    INDEX_TYPE="DO_NOT_INDEX",
  (10)  LABEL="Reference"
    SECTION_NAME=Reference
    INDEX_TYPE="INDEX_BY_TOKEN",
  (11)  LABEL="Comments"
    SECTION_NAME=Comments
    INDEX_TYPE="DO_NOT_INDEX",
  (12)  LABEL="Database_Cross_Reference"
    SECTION_NAME=Database_Cross_Reference
    INDEX_TYPE="INDEX_BY_TOKEN",
  (13)  LABEL="Keywords"
    SECTION_NAME=Keywords
    INDEX_TYPE="INDEX_BY_TOKEN",
  (14)  LABEL="Features"
    SECTION_NAME=Features
    INDEX_TYPE="DO_NOT_INDEX",
  (15)  LABEL="Sequence_Information"
    SECTION_NAME=Sequence_Information
    INDEX_TYPE="INDEX_PROTEIN",
  (16)  LABEL="TEXT"
    SECTION_NAME=TEXT
    INDEX_TYPE="INDEX_BY_TOKEN"
}

PRIMARY_KEY_DEFINITION
  {
      PRIMARY_SECTIONS = (Entry_Name)
  }

UNION_SECTION_DEFINITION
{
```

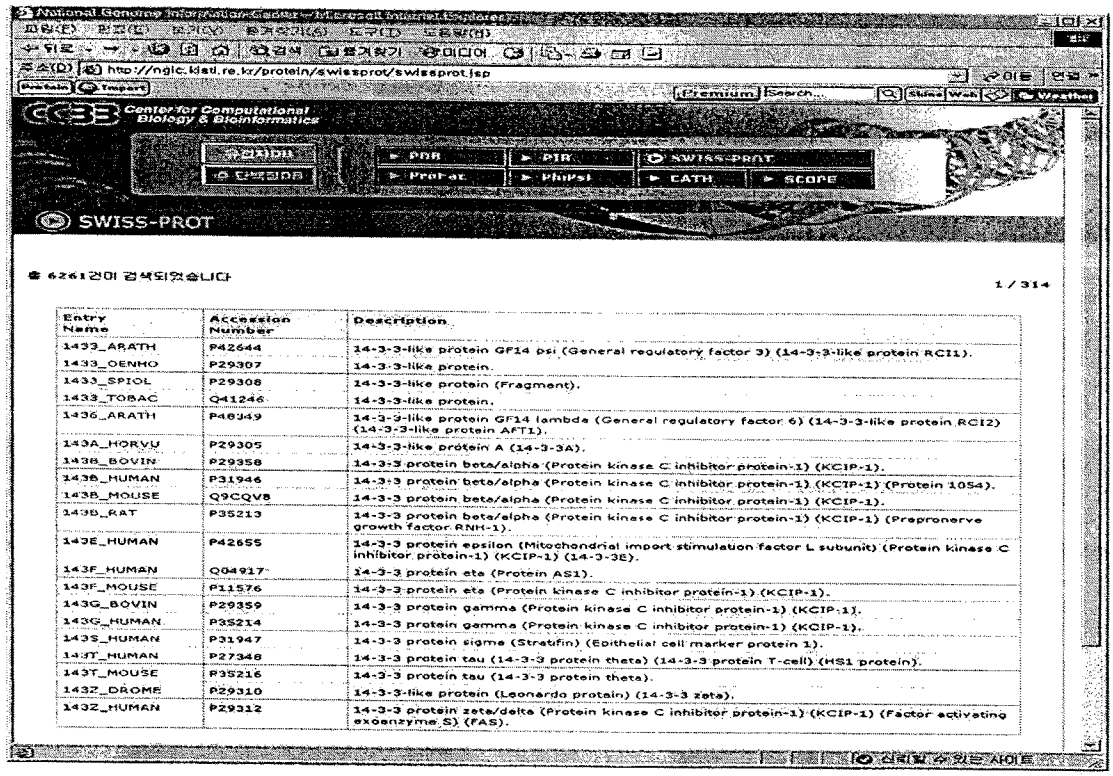
```
    (1) LABEL="BASIC SEARCH FIELDS"
        SECTION_NAME=BASIC
        SECTIONS=(TEXT)
  }
}

DEFINE_DOCUMENT_STRUCTURE
{
 STRUCTURE_DEFINITION = ALL_DOCUMENTS {
  // 문서시작 태그 정의
  (1) TAG_NAME="@swissprot" ACTION=DISCARD NEW_DOCUMENT_FLAG=TRUE,
  // 섹션 태그및 액션 정의
  (2) TAG_NAME="EID=" ACTION=COPY SECTION_NAME=Entry_Name,
  (3) TAG_NAME="ID=" ACTION=COPY SECTION_NAME=Identification,
  (4) TAG_NAME="AC=" ACTION=COPY SECTION_NAME=Accession_number,
  (5) TAG_NAME="DT=" ACTION=COPY SECTION_NAME=Date,
  (6) TAG_NAME="DE=" ACTION=COPY SECTION_NAME=Description,
  (7) TAG_NAME="GN=" ACTION=COPY SECTION_NAME=Gene_Name,
  (8) TAG_NAME="OS=" ACTION=COPY SECTION_NAME=Organism_Species,
  (9) TAG_NAME="OX=" ACTION=COPY SECTION_NAME=Organism_Taxonomy_Cross_Reference,
  (10)  TAG_NAME="OC=" ACTION=COPY SECTION_NAME=Organism_Classification,
  (11)  TAG_NAME="RX=" ACTION=COPY SECTION_NAME=Reference,
  (12)  TAG_NAME="CC=" ACTION=COPY SECTION_NAME=Comments,
  (13)  TAG_NAME="DR=" ACTION=COPY SECTION_NAME=Database_Cross_Reference,
  (14)  TAG_NAME="KW=" ACTION=COPY SECTION_NAME=Keywords,
  (15)  TAG_NAME="FT=" ACTION=COPY SECTION_NAME=Features,
  (16)  TAG_NAME="SQ=" ACTION=COPY SECTION_NAME=Sequence_Information,
  (17)  TAG_NAME="TEXT=" ACTION=COPY SECTION_NAME=TEXT
  }
}

// 데이터베이스 생성
CREATE_DATABASE
{
// 데이터베이스 이름과 크기를 정의
  (1) DATABASE_NAME=SWISSPROT1
      DATABASE_SIZE=1000
}

// 문서 그룹 정의
DEFINE_DOCUMENT_GROUP
{
    (1)   swiss1=('/data4/swiss/swissprot/data/swissprot_1.kst')
}

LOAD_DATABASE
{
  (1) FROM=swiss1
      TO=SWISSPROT1
      WITH=ALL_DOCUMENTS
} END
```

(나) 검색 인터페이스

<figure 1-26>는 SWISS-PROT 데이터베이스의 검색 페이지를 보이고 있다. SWISS-PROT 데이터베이스의 검색은 KRISTAL 데이터베이스로 변환시 생성된 각 검색 Section에 대한 keyword 검색을 제공한다. <figure 1-27>는 All Section에서 'kinase' keyword에 대한 검색 결과를 보이고 있다. 그림에서 알 수 있듯이, 총 6261개의 entry가 검색되었으며 각 entry의 name, accession number, description 정보를 제공한다. 각 entry의 상세정보는 entry name을 클릭함으로써 얻을 수 있다. <figure 1-28>은 entry '1433_ARATH'의 상세 정보를 보이고 있다.

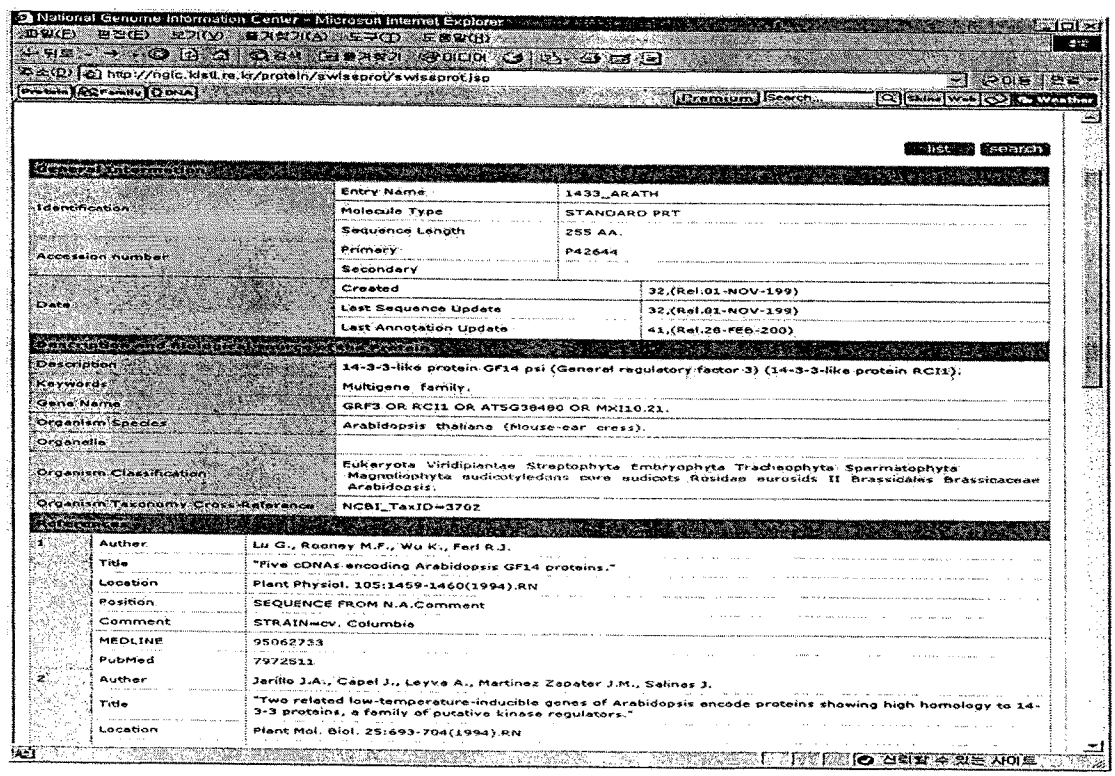


<figure 1-26> SWISS-PROT: 검색 페이지



Entry Name	Accession Number	Description
1433_ARATH	P42644	14-3-3-like protein GF14 psi (General regulatory factor 3) (14-3-3-like protein RCI1).
1433_OENHO	P29307	14-3-3-like protein.
1433_SPIOL	P29308	14-3-3-like protein (Fragment).
1433_TOBAC	Q41246	14-3-3-like protein.
1436_ARATH	P48349	14-3-3-like protein GF14 lambda (General regulatory factor 6) (14-3-3-like protein RCI2) (14-3-3-like protein AFT1).
143A_HORVU	P29305	14-3-3-like protein A (14-3-3A).
143B_BOVIN	P29358	14-3-3 protein beta/alpha (Protein kinase C inhibitor protein-1) (KCIP-1).
143B_HUMAN	P31946	14-3-3 protein beta/alpha (Protein kinase C inhibitor protein-1) (KCIP-1) (Protein 1054).
143B_MOUSE	Q9CQV8	14-3-3 protein beta/alpha (Protein kinase C inhibitor protein-1) (KCIP-1).
143B_RAT	P35213	14-3-3 protein beta/alpha (Protein kinase C inhibitor protein-1) (KCIP-1) (Prepronerve growth factor RNH-1).
143E_HUMAN	P42655	14-3-3 protein epsilon (Mitochondrial import stimulation factor L subunit) (Protein kinase C inhibitor protein-1) (KCIP-1) (14-3-3E).
143F_HUMAN	Q04917	14-3-3 protein eta (Protein AS1).
143F_MOUSE	P11576	14-3-3 protein eta (Protein kinase C inhibitor protein-1) (KCIP-1).
143G_BOVIN	P29359	14-3-3 protein gamma (Protein kinase C inhibitor protein-1) (KCIP-1).
143G_HUMAN	P35214	14-3-3 protein gamma (Protein kinase C inhibitor protein-1) (KCIP-1).
143S_HUMAN	P31947	14-3-3 protein sigma (Stratifin) (Epithelial cell marker protein 1).
143T_HUMAN	P27348	14-3-3 protein tau (14-3-3 protein theta) (14-3-3 protein T-cell) (HS1 protein).
143T_MOUSE	P35216	14-3-3 protein tau (14-3-3 protein theta).
1432_DROME	P29310	14-3-3-like protein (Leonardo protein) (14-3-3 zeta).
1432_HUMAN	P29312	14-3-3 protein zeta/delta (Protein kinase C inhibitor protein-1) (KCIP-1) (Factor activating exoenzyme S) (FAS).

<figure 1-27> SWISS-PROT: 검색 결과 페이지 - 간략 정보 보기



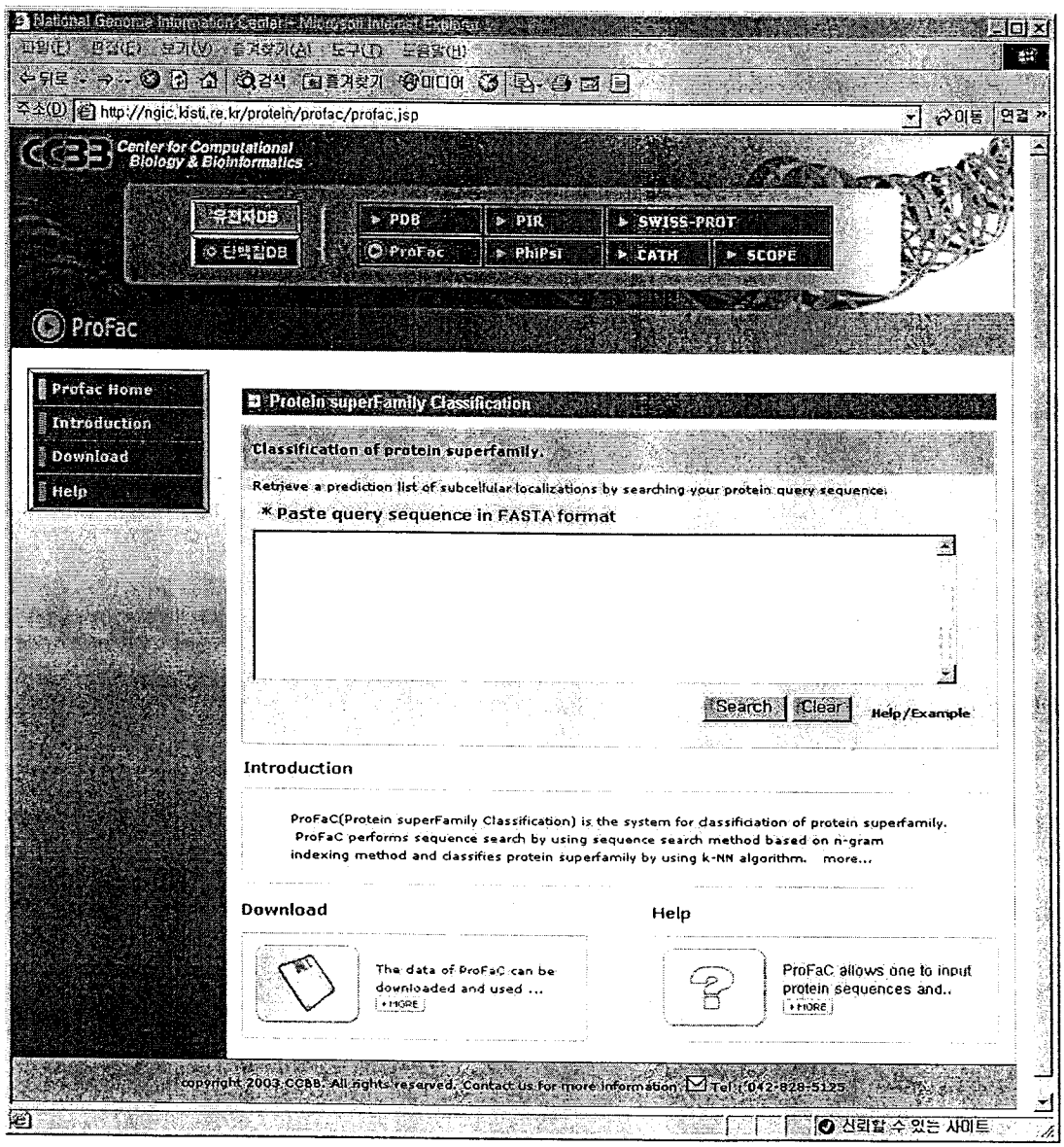
Identification	Entry Name	1433_ARATH
	Molecule Type	STANDARD PRT
	Sequence Length	255 AA.
Accession number	Primary	P42644
	Secondary	
Date	Created	32,(Rel.01-NOV-199)
	Last Sequence Update	32,(Rel.01-NOV-199)
	Last Annotation Update	41,(Rel.26-FEB-200)
Description		14-3-3-like protein GF14 psi (General regulatory factor 3) (14-3-3-like protein RCI1).
Keywords		Multigene family.
Gene Name		GRF3 OR RCI1 OR AT5G38480 OR MXI10.21.
Organism Species		Arabidopsis thaliana (Mouse-ear cress).
Organelle		
Organism Classification		Eukaryota Viridiplantae Streptophyta Embryophyta Tracheophyta Spermatophyta Magnoliophyta eudicotyledons core eudicots Rosidae eurosids II Brassicales Brassicaceae Arabidopsis.
Organism Taxonomy Cross-Reference		NCBI_TaxID=3702

1	Author	Lu G., Rooney M.F., Wu K., Ferl R.J.
	Title	"Five cDNAs encoding Arabidopsis GF14 proteins."
	Location	Plant Physiol. 105:1459-1460(1994).RN
	Position	SEQUENCE FROM N.A.Comment
	Comment	STRAIN=cv. Columbia
	MEDLINE	95062733
	PubMed	7972511
2	Author	Jarillo J.A., Capel J., Leyva A., Martinez Zapater J.M., Salinas J.
	Title	"Two related low-temperature-inducible genes of Arabidopsis encode proteins showing high homology to 14-3-3 proteins, a family of putative kinase regulators."
	Location	Plant Mol. Biol. 25:693-704(1994).RN

<figure 1-28> SWISS-PROT: 상세정보 보기 - entry '1433_ARATH'

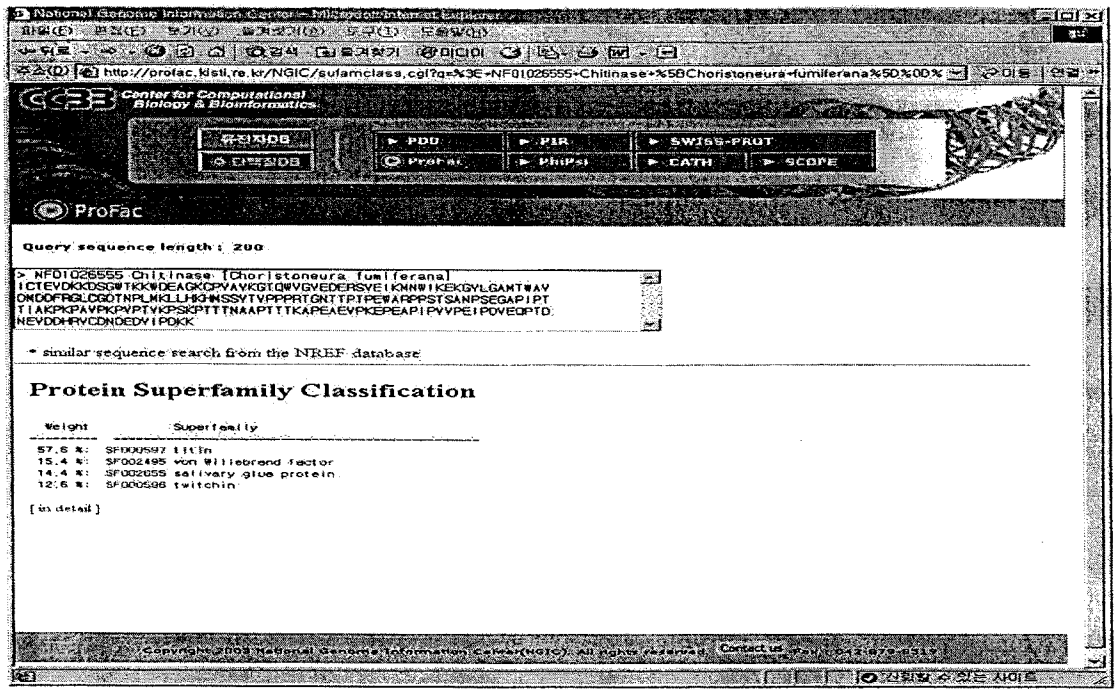
(5) ProFac

ProFac (Protein superFamily Classification)은 CCBB에서 자체 구축한 2차 데이터베이스로서, 단백질의 superfamily 레벨에서의 단백질 서열로부터 단백질 기능에 대한 정보를 제공하는 데이터베이스이다. ProFac은 단백질 서열 검색을 위해 자체 개발한 N-gram indexing 방법을 사용하여 단백질의 superfamily를 분류하기 위해 k-NN 알고리즘을 활용하였다. <figure 1-29>은 ProFac 데이터베이스의 검색 페이지를 보이고 있다. 검색 페이지에서는 FASTA 포맷의 단백질 서열을 활용한 검색 기능을 제공하며 ProFac에 대한 간단한 소개와 관련 프로그램을 다운받는 채널을 제공하고 있다.

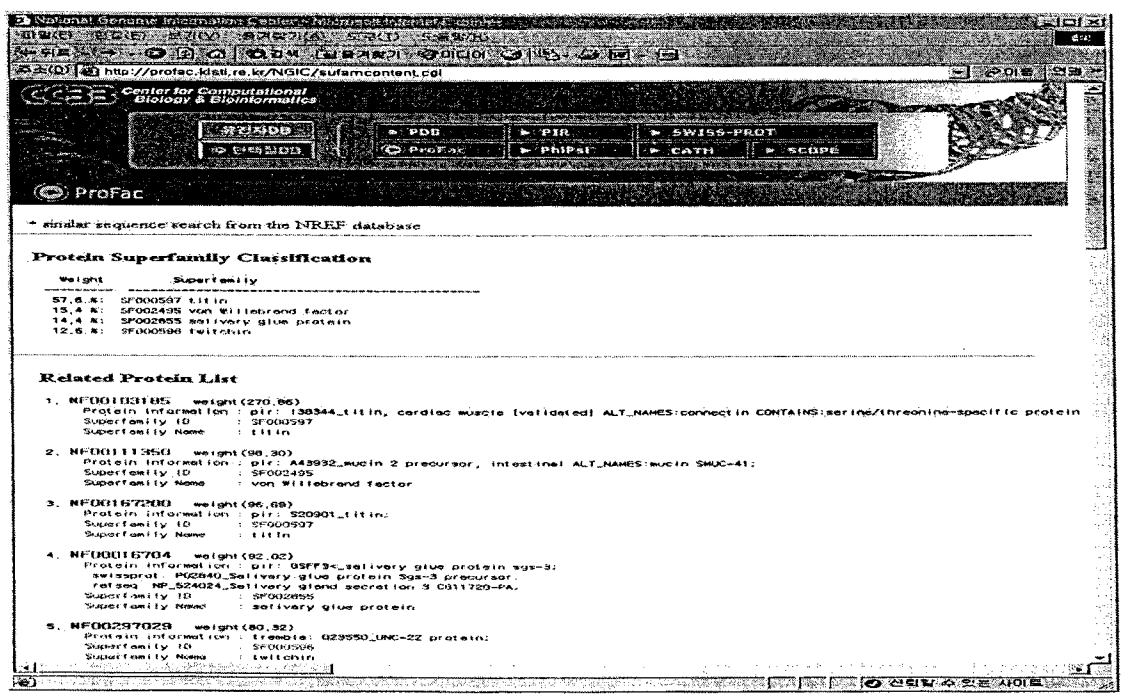


<figure 1-29> ProFac: 검색 페이지

<figure 1-30>은 NF0102655 sequence에 대한 분류 결과의 간략정보를 보이고 있다. <figure 1-31>는 얻어진 각 superfamily의 상세정보를 보이고 있다. 그림에서 얻어진 각 entry의 상세정보는 PIR 데이터베이스와 하이퍼링크 되어 있다.



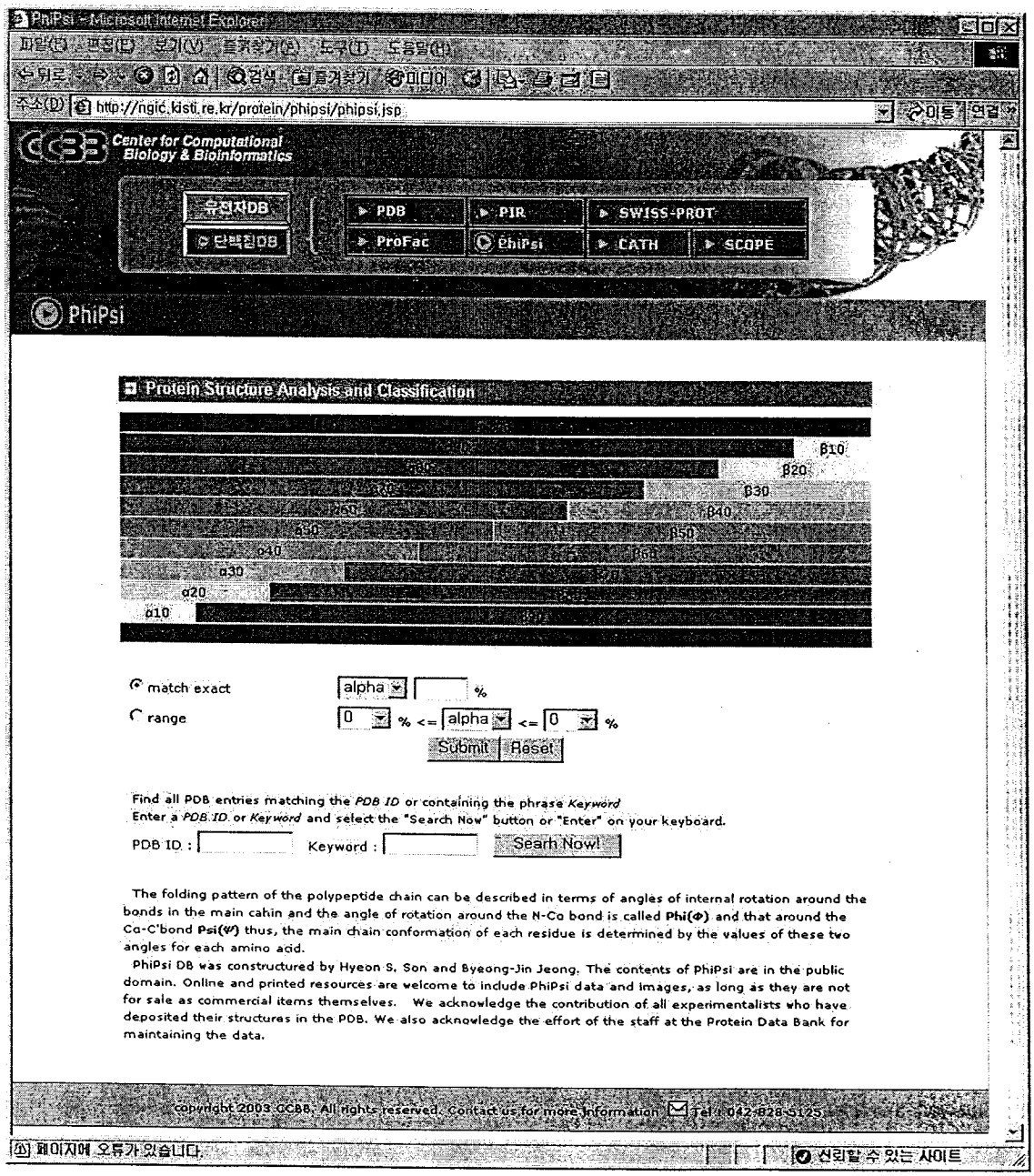
<figure 1-30> ProFac: 검색 결과 -간략 정보 보기



<figure 1-31> ProFac: 검색 결과 - 상세 정보 보기

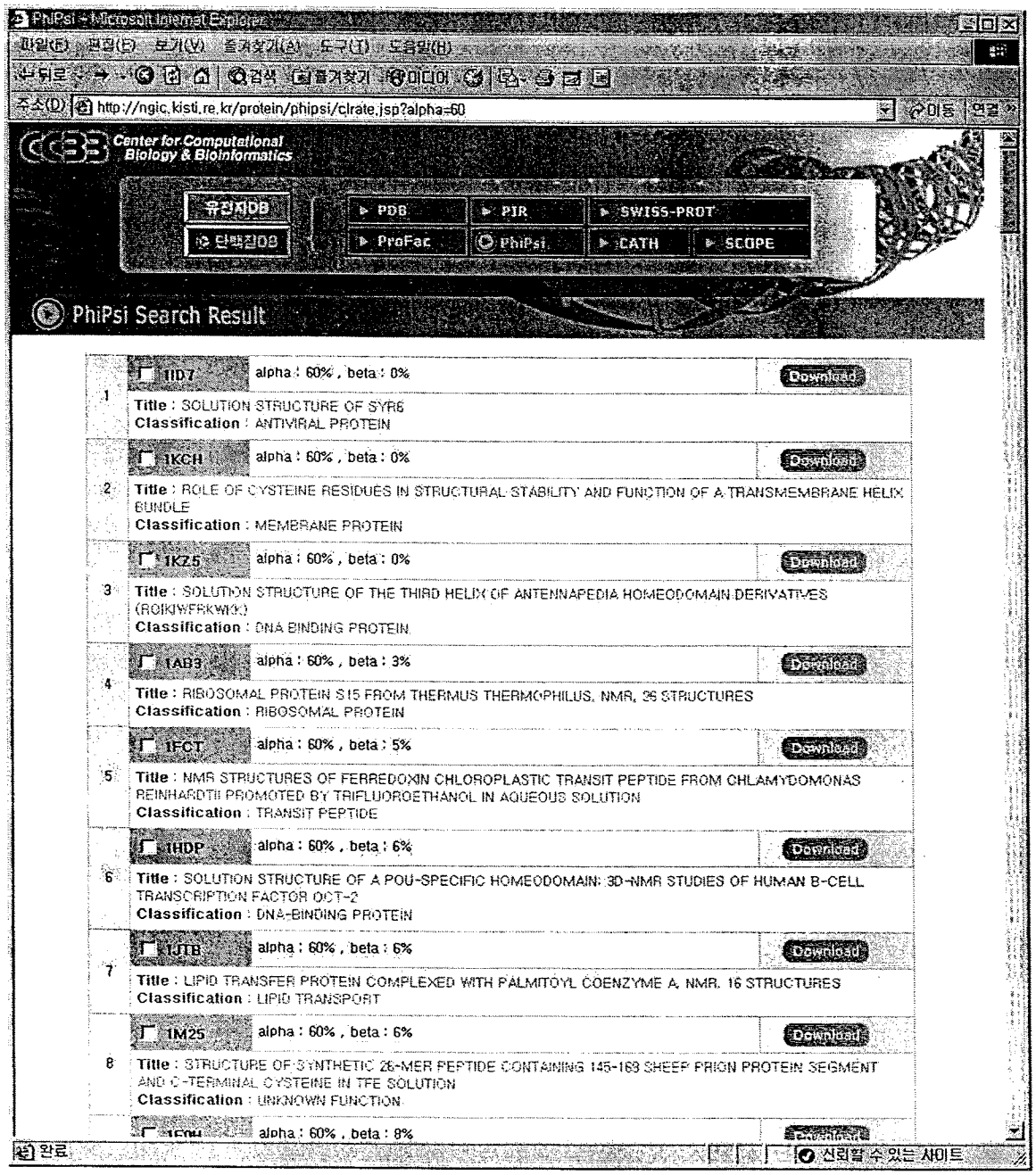
(6) PhiPsi

PhiPsi 데이터베이스는 기존의 PHP에서 Java/JSP로 검색 인터페이스를 변환함으로써 평균 20% 정도의 검색 성능 향상을 보이고 있다. <figure 1-32>과 <figure 1-33>은 PhiPsi 데이터베이스의 검색 페이지와 알파 60에 대한 검색 결과를 보이고 있다.

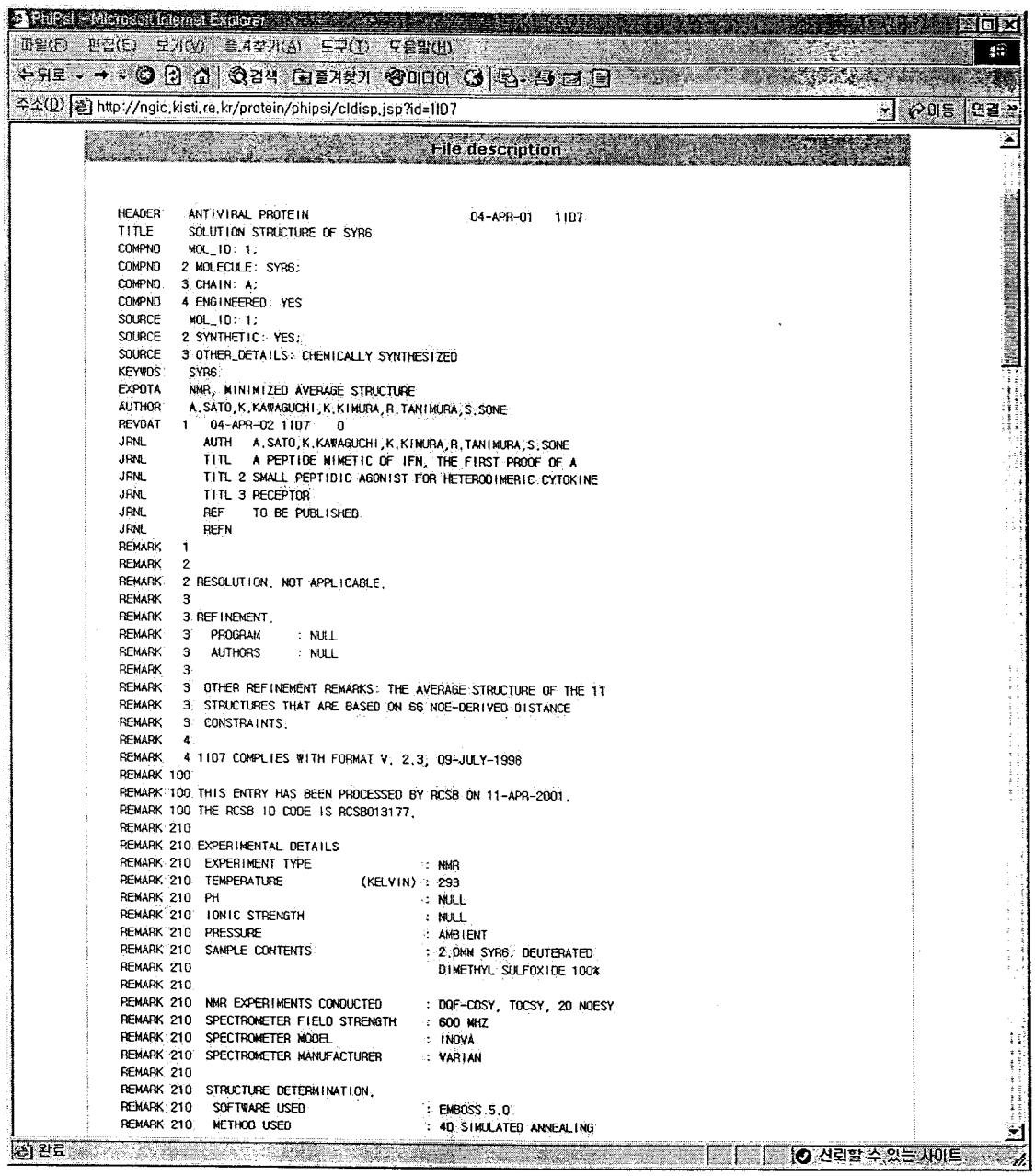


<figure 1-32> PhiPsi: 검색 페이지

검색된 결과는 각 결과를 파일로 저장하거나 원하는 결과들을 선택하여 압축된 형태로 저장할 수 있다. <figure 1-34>은 ID 1ID7의 파일 정보를 보이고 있다.



<figure 1-33> PhiPsi: 검색 결과 페이지



<figure 1-34> PhiPsi: 검색 결과 - 파일 정보

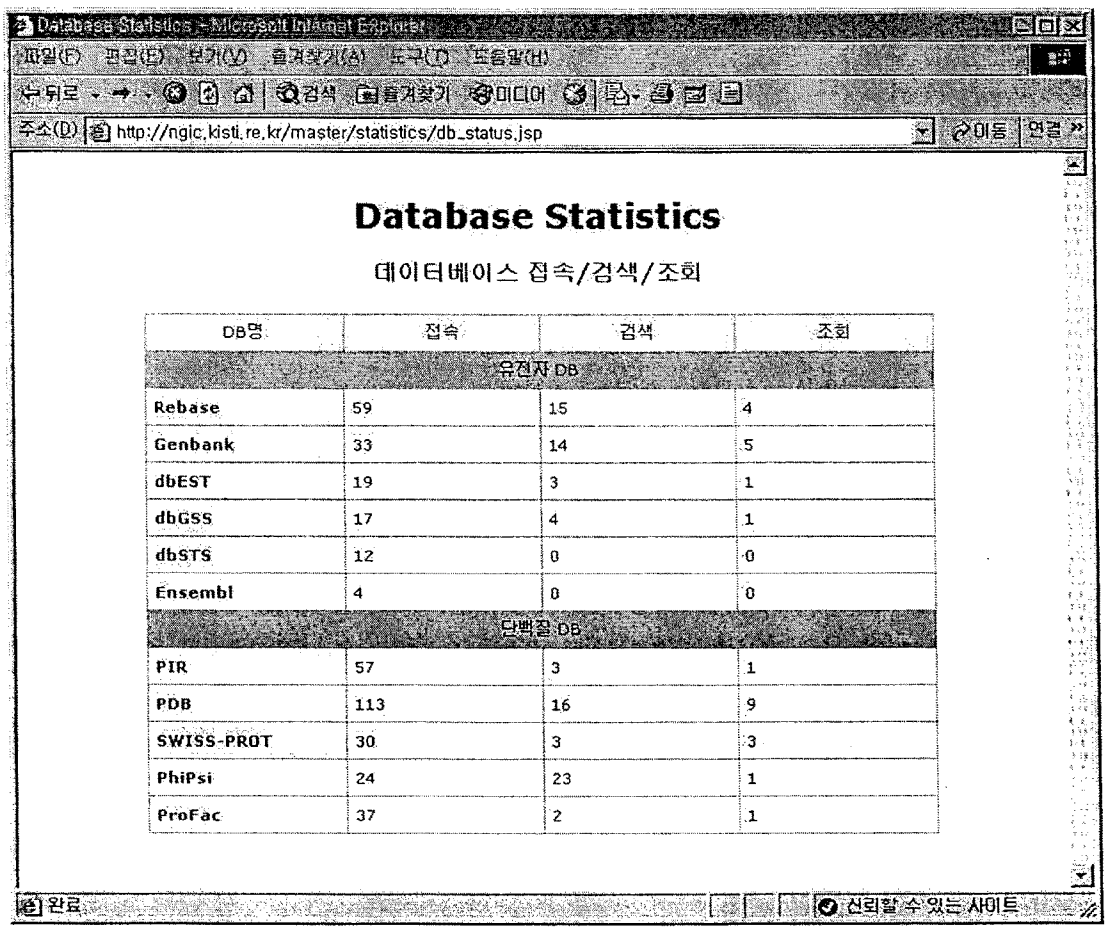
(7) 데이터베이스 통계 서비스

본 연구에서는 앞서 언급한 데이터베이스 검색 시스템의 구축과 더불어 각 데이터베이스의 이용실태를 파악함으로써 향후 양질의 검색 서비스 제공을 위한 데이터베이스 이용실태 통계 기능을 구축하였다. 데이터베이스 통계는 사용자들의 각 데이터베이스의 접속, 검색, 조회 횟수를 조사한다. 데이터베이스의 통계에서 접속,

검색, 조회가 가지는 의미는 다음과 같다.

- 접속통계: 사용자가 원하는 데이터베이스에 접속한 횟수
- 검색통계: 사용자가 검색 단어를 입력한 횟수
- 조회통계: 사용자가 검색 결과 중 단위 레코드를 열람한 횟수

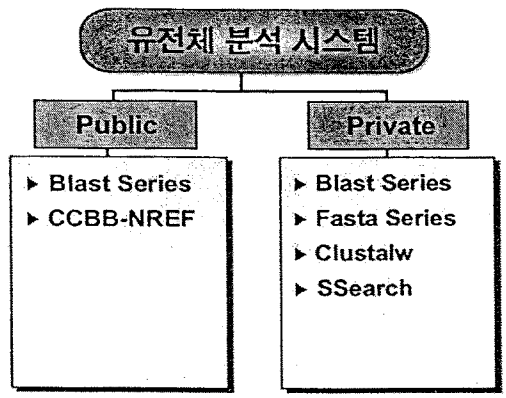
<figure 1-35>는 데이터베이스 검색 시스템에서 제공되는 각 데이터베이스의 2003.10.1일자 통계 현황을 보이고 있다.



<figure 1-35> 데이터베이스의 통계현황

3. 유전체 분석 서비스 체제 구축

유전체 분석 서비스는 CCBB에 구축된 고성능 생물 분석 서버를 활용하여 국내 연구자들의 생물 정보 분석 서비스를 제공한다. 본 서비스에서는 사용자의 다양한 분석 요구도 중 대표적인 분석 서비스를 제공하며 사용자의 분석 질의에 대한 분석 결과의 지연을 최소화함으로써 분석 요구를 원활히 지원할 수 있도록 하였다. <figure 1-36>는 본 서비스에서 제공하는 유전체 분석 시스템의 개요를 나타내고 있다. 유전체 분석 시스템은 Public 서비스와 Private 서비스로 나누어진다.

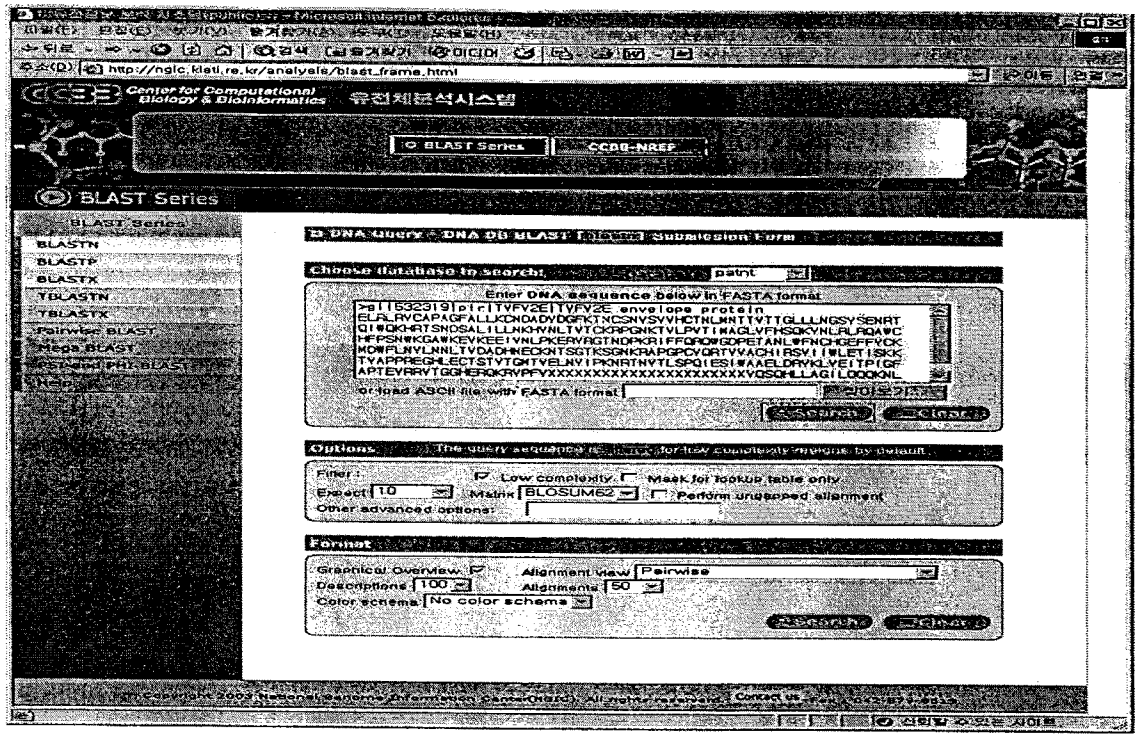


<figure 1-36> 유전체 분석 시스템

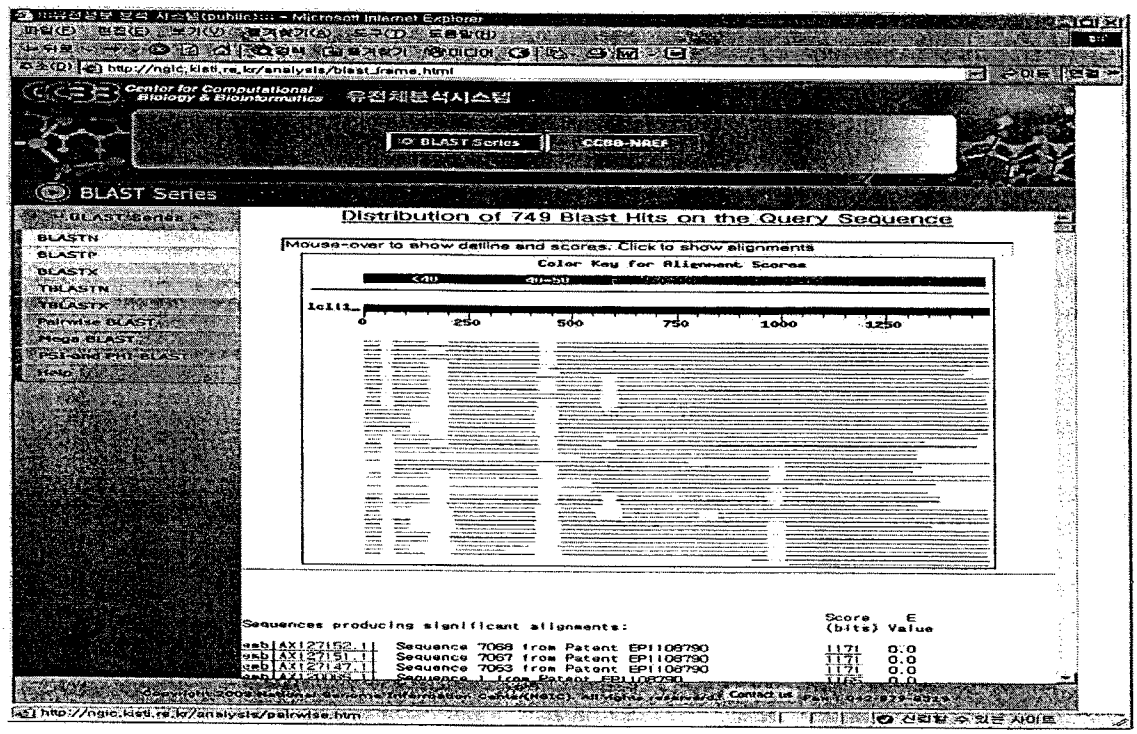
Public 서비스에서는 유전체 분석에 가장 많이 활용되는 8가지 Blast 분석 도구들 (BlastN, BlastP, BlastX, TBlastN, TBalstX, Pairwise Blast, Mega Blast, Psi-and Phi-Blast)과 CCBB에서 자체 개발한 CCBB-NREF 분석 도구를 제공한다. Private 분석 서비스에서는 Public에서 제공하는 분석 도구들외에 Fasta Series와 ClustalW, SSearch 분석 도구를 서비스한다. Public과 Private 분석 서비스 모두 CCBB의 고성능 분석 서버를 활용하지만 Private 분석 서비스는 Public과 달리 분석에 사용되는 사용자의 실험 데이터의 보안을 유지할 수 있도록 하였다.

<figure 1-37>는 Public 분석 서비스의 BlastN 분석 페이지를 보이고 있다. Blast 프로그램 사용에 있어서 개별 option이나 프로그램에 대한 도움말은 해당 option이나 프로그램 이름을 클릭하면 상세한 설명 화면이 나타나도록 하였으며 별도의 help 파일을 만들어 Blast 프로그램 수행에 대한 전반적인 설명서를 볼 수 있

으며 보다 심도 깊은 사용 설명을 원하면 개별 entry에 대한 외부 접속 링크를 만들어 두었다. <figure 1-38>은 BlastN 의 분석 결과 페이지를 보이고 있다.

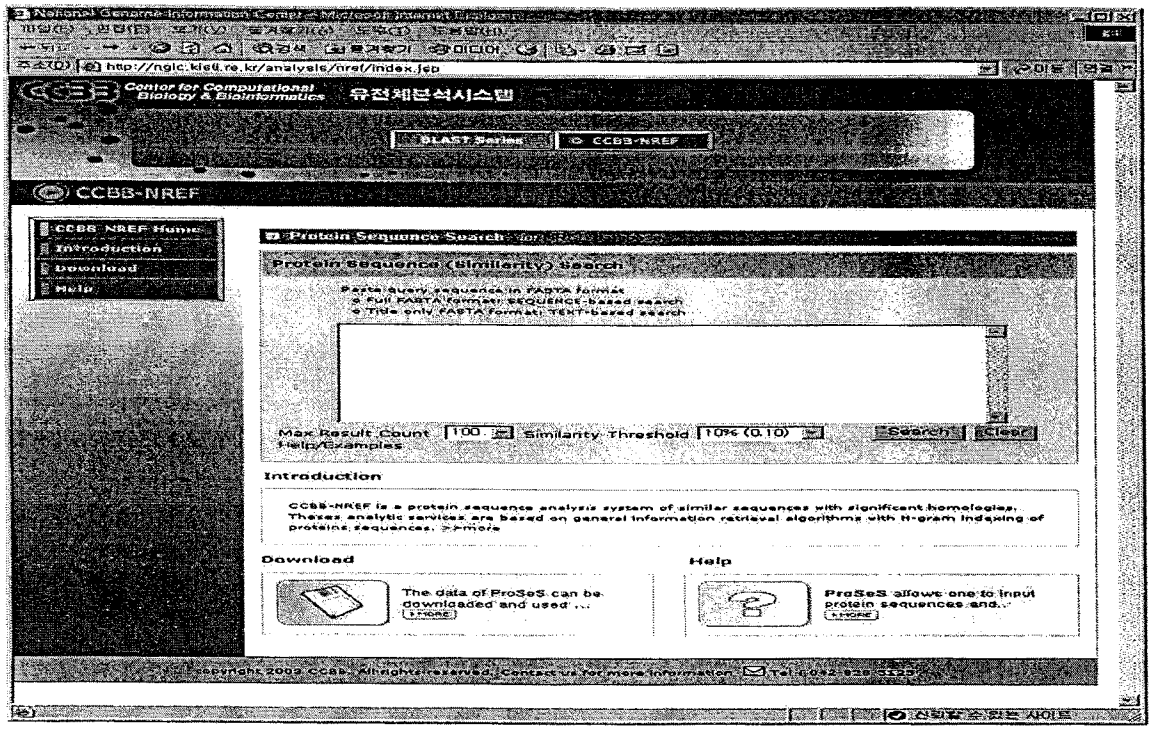


<figure 1-37> BlastN 분석 페이지

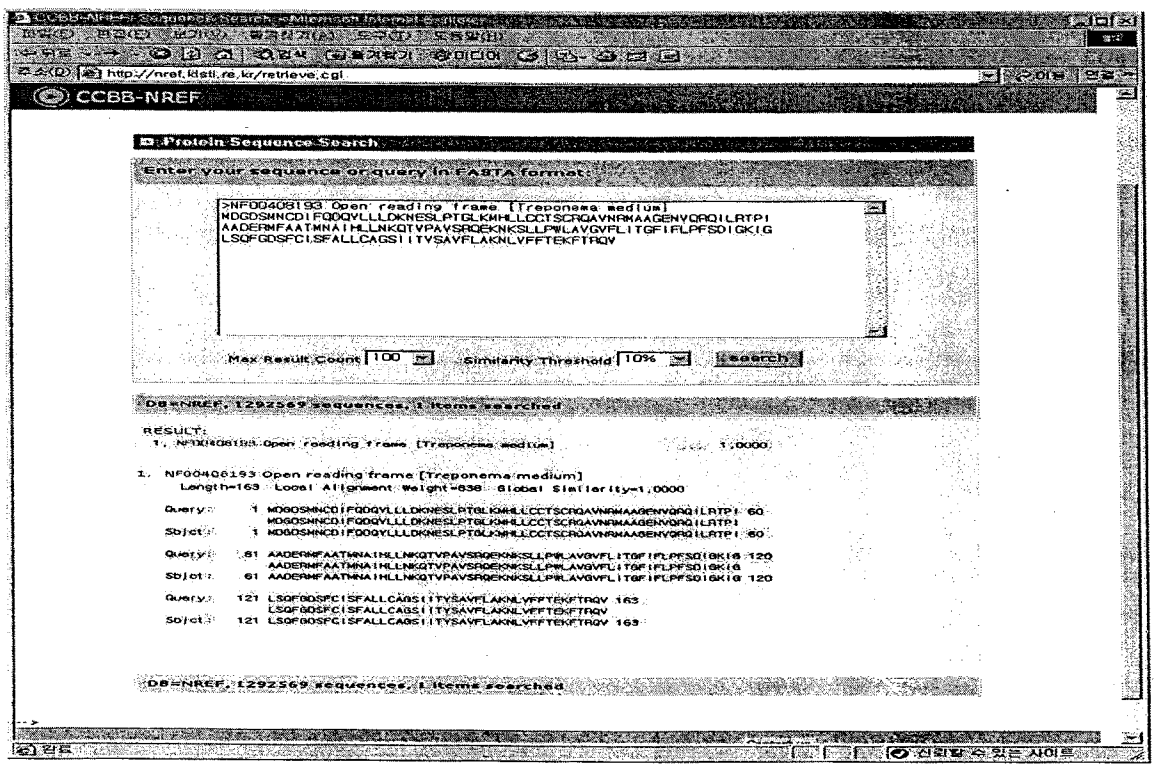


<figure 1-38> BlastN 분석 결과 페이지

<figure 1-39>는 CCBB-NREF 분석 프로그램의 분석 페이지를 <figure 1-40>은 분석 결과 페이지를 보이고 있다.



<figure 1-39> NREF: 분석 페이지

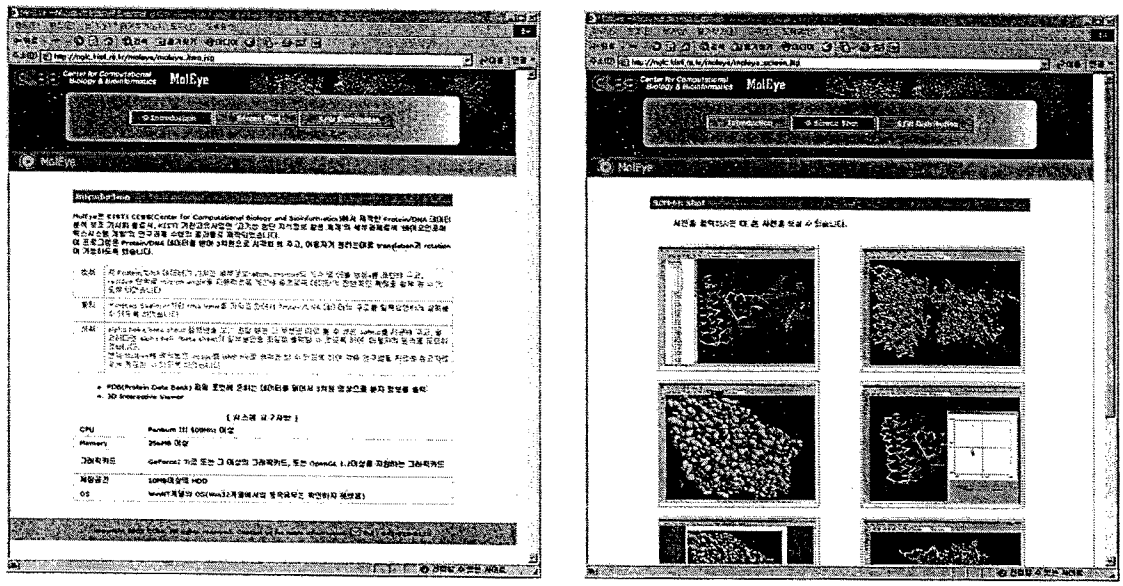


<figure 1-40> CCBB-NREF: 분석 결과 페이지

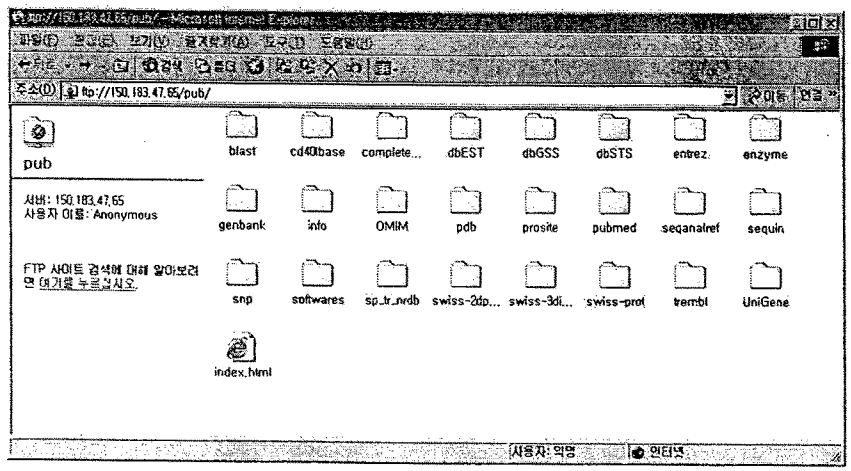
4. 기타 서비스

생물정보 데이터베이스 검색과 분석 서비스 외에 효율적인 연구활동과 원할한 데이터의 공유를 위해 Dummy Analyzer <figure 1-41>와 대용량 FTP 서비스 체제를 구축하였다 (<figure 1-42>).

Dummy Analyzer는 KISTI CCBB에서 자체 제작한 Protein/DNA 데이터 분석 보조 가시화 툴로서, 자세한 사항은 5절에 정리하였다 (p 214)



<figure 1-41> Dummy Analyzer



<figure 1-42> FTP service

제 2 절 국내외 생물정보 DB 유지보수 및 신규 구축

1. 신규 데이터베이스 구축

가. REBASE

(1) 개요

REBASE는 제한효소에 대한 정보를 제공하고 있는 데이터베이스로서 최근에는 methyltransferase와 nicking enzymes등도 포함되어지고 있다. 현재 약 3,576개의 제한효소정보를 제공하고 있으며, 이 안에는 12개의 새로운 TypeII 제한효소가 추가되었으며, 이들 3516개의 TypeII 제한효소에는 588개의 판매 제품이 있다. 또한 15개의 DNA methltransferase와 5개의 homing endonuclease 그리고 3개의 nicking enzymes이 판매되고 있다. (Roberts *et al.*, 2003)

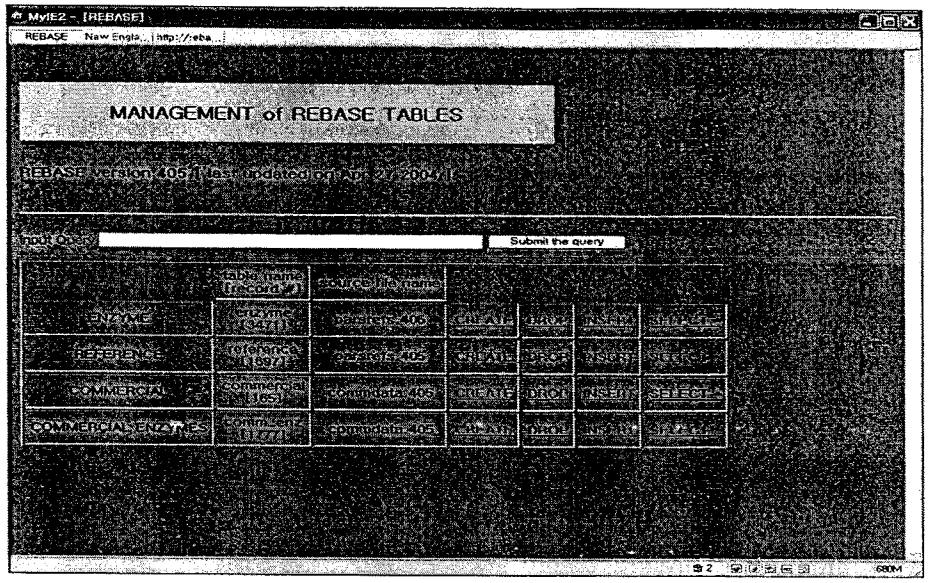
현재 REBASE는 New England BioLabs, Inc.에서 제공하고 있으며, 웹사이트는 http://rebase.neb.com/rebase/rebase.html 이다.

특히 제한효소정보는 분자생물할 실험에서 주로 이용되고 있으며 정보로 사용자들이 자신의 염기서열에서 필요한 제한효소를 손쉽게 찾을 수 있도록 정보를 제공하고 있다.

(2) 데이터베이스 구축

현재 최신버전의 REBASE는 405 build이고, 2004년 4월 27일자로 공개되었다. 제공되는 자료중 commdata.405 와 parsrefs.405를 사용하여 "enzyme", "reference", "commercial", "comm_enz"의 4개의 테이블을 재구축하였다.

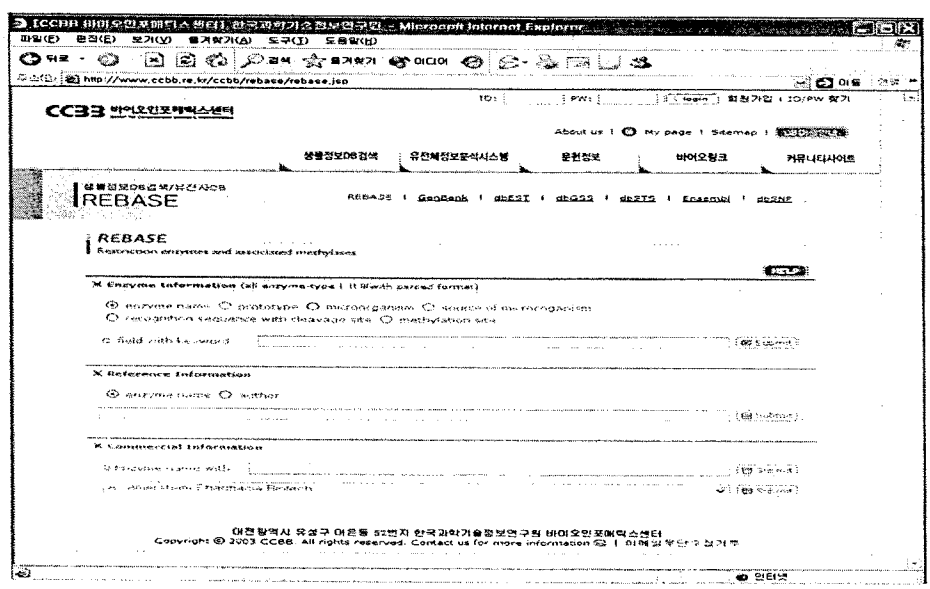
재구축 작업을 위해 REBASE의 텍스트 파일은 분석하고, 이를 데이터베이스로 옮기는 작업을 지속적으로 진행하기위해 웹 인터페이스 상에서 자료를 업데이트 할 수 있는 데이터베이스 관리용 웹 프로그램을 제작하였다. (<figure 2-1>)



<figure 2-1> REBASE 관리용 웹프로그램

(3) REBASE 서비스

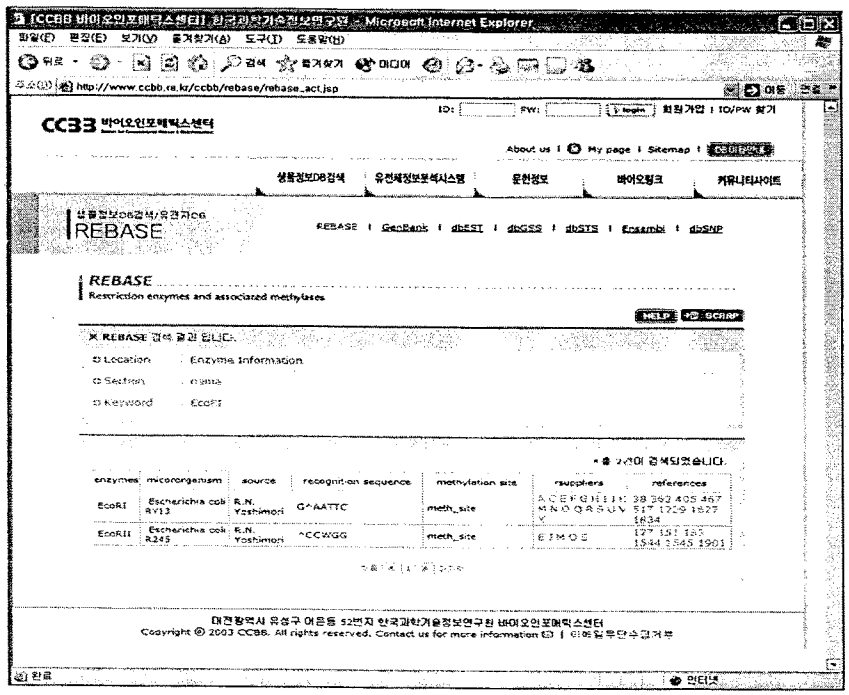
사용자들이 손쉽게 제한효소를 검색할 수 있도록 9가지의 항목에 대한 검색이 첫 페이지에서 가능하도록 구성하였다. 이를 통해 사용자가 하나의 화면에서 필요한 모든 항목을 검색하여 불필요한 네비게이션 시간을 최소화 하려 하였다. (<figure 2-2>)



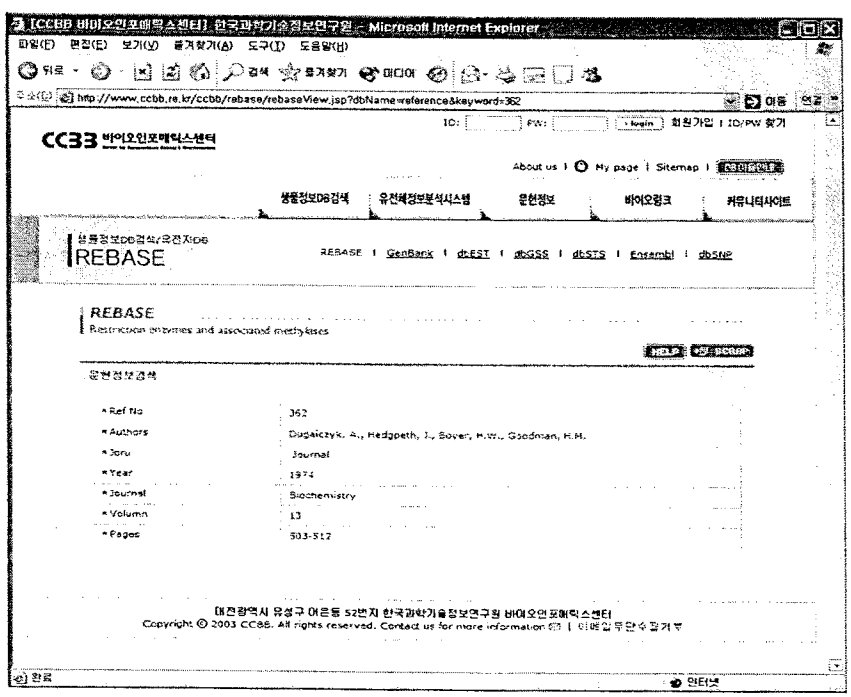
<figure 2-2> REBASE 관리용 웹프로그램

사용자가 EcoRI에 대해 궁금해 한다면 enzyme name의 항목으로 검색 해 볼 수 있다. 결과는 <figure 2-3>에서처럼 발견된 제한효소의 목록을 볼 수 있으며, 제한효소의 인식부위

에 대한 정보를 곧바로 볼 수 있도록 하였다. 뿐 만 아니라 제한효소를 제공하는 회사정보와 참고문헌정보에 대한 링크를 제공하여 필요하다면 제한효소를 제공하는 회사 또는 참고논문에 대한 정보를 찾아갈 수 있도록 하였다. (<figure 2-4>)

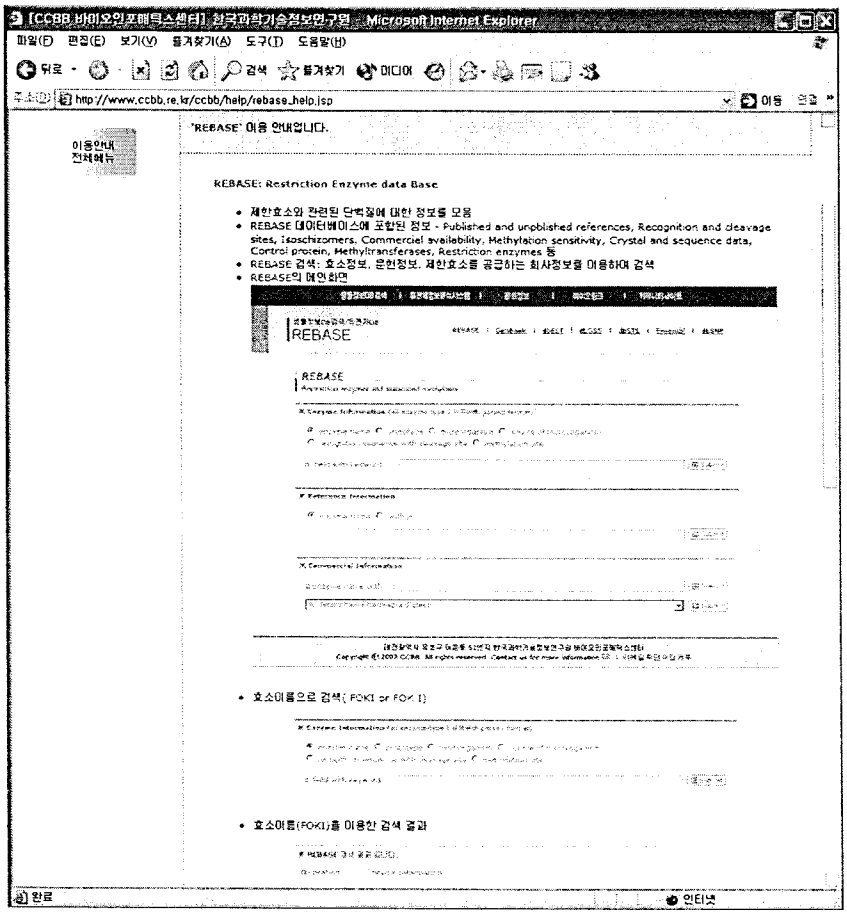


<figure 2-3> 제한효소 검색 결과



<figure 2-4>참고문헌 검색 결과.

특히 사용자들의 편의를 도모하기 위해 상세한 설명을 첨부하여 사용자가 처음으로 사용하더라도 설명내용만 따라가기만 하면 원하는 검색 결과를 얻을 수 있도록 배려하였다. (<figure 2-5>)



<figure 2-5> REBASE 사용자 설명서

(4) 결과 활용 및 기대효과

REBASE는 분자생물학 실험에 가장 많이 사용하고 있는 제한 효소의 정보를 제공함으로써 실험자들이 손쉽게 자신의 실험에 적합한 효소를 필요한 회사와 참고논문까지 한 번에 찾을 수 있도록 하여 실험 설계 시 효율성을 높여 줄 수 있을 것이며, 더 나아가 주기적으로 데이터베이스가 업데이트 되므로 이에 대한 편리한 데이터베이스 업데이트용 웹프로그램을 제작하여 사용자에게 최신의 정보를 꾸준히 제공 할 수 있을 것이다.

나. dbSNP

(1) 개요

SNP는 단일 핵산 다형성(Single Nucleotide Polymorphism)에 대한 데이터베이스로서, 현재 약 천백만개의 SNP가 보고 되어져 있다 (<table 2-1>). SNP들은 가장 일반적인 유전자 변종이며 매 100~300 염기에서 일어난다. 유전학에서 주요 연구 관점은 유전적인 표현형을 가진 서열 변종과의 결합이며 질병과 모집단에서 특정 차이(SNP) 간의 결합을 검색하여 SNP가 질병 유전자의 발견을 하며 이러한 다형성(polymorphism)은 특정 실험 조건에 대한 서열 정보를 이용한 연구를 가능하게 한다.

(2) dbSNP 데이터베이스 구축

(가) dbSNP 최신자료

dbSNP는 NCBI(ftp://ftp.ncbi.nlm.nih.gov/dbsnp)에서 제공하고 있으며, 현재 Build 120이 2004년 3월 18일 공개되었다.

제공되는 데이터베이스 자료는 Microsoft사(http://www.microsoft.com/)의 MSSQL을 기반으로 제공되고 있으며, 이에 대한 데이터베이스 스키마도 제공되고 있다.

(나) MSSQL용 데이터베이스를 MySQL로 전환

상용으로 판매되고 있는 MSSQL용 데이터베이스를 자유롭게 사용가능한 MySQL(http://www.mysql.com)로 전환하기 위해 데이터베이스 스키마정보를 MySQL에 맞도록 수정작업을 거쳤다. 수정은 전체 198개의 테이블에 대해 이루어졌으며, 최종적으로 MySQL에 해당 테이블을 만들고 자료를 입력해 봄으로서 이상이 없음을 확인 할 수 있었다.

(다) 자료의 입력

현재 120 빌드의 전체 자료는 473,065,541개였으며, MSSQL에서 제공하는 bcp파일 포맷을 사용하기 위해 탭분리용 파일을 MySQL데이터베이스에 넣을 수 있도록 입력용 프로그램을 제작하였다. 또한 입력이 정확히 이루어지었는지 확인할 수 있도록 입력 작업이 끝난 다음, 입력용 bcp파일과 데이터베이스를 비교해주는 관리용 프로그램도 개발하였다. 이는 모두 perl을 이용하여 개발되어졌다.

(라) MySQL의 환경 세팅

MySQL에서 기본적으로 제공하는 최대 파일크기는 4Gb이다. 하지만 dbSNP의 경

우 최대 약 9Gb까지 자료를 입력해야하는 문제점이 발생하였다. 이 문제를 해결하기 위해 MySQL에서는 데이터베이스의 크기를 변경할 수 있는 기능을 최근 제공하고 있다. 본 과제에서도 대량의 데이터베이스 정보의 입력을 위해 최대행의 크기를 증가시킴으로 하여 원활히 데이터베이스가 설치 될 수 있도록 최적화 하였다.

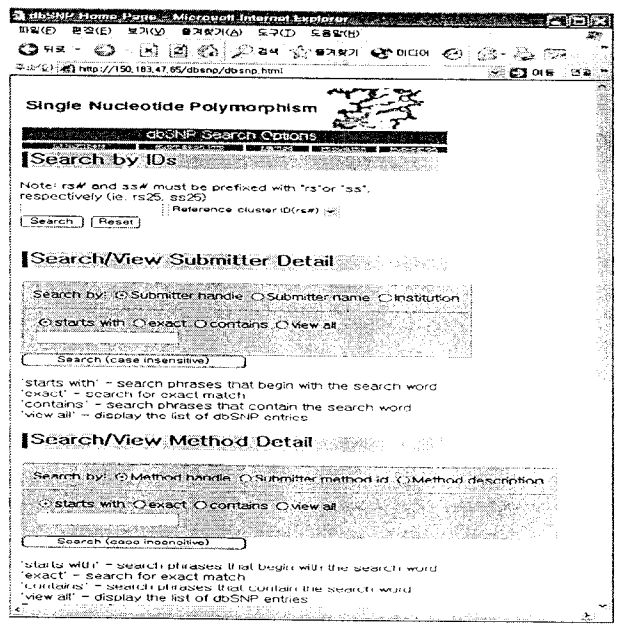
<table 2-1> 현재 등록된 SNP 통계

Organism	Number of Submissions (ss#'s)	Number of RefSNP Clusters (rs#'s) (# validated)	Number of (ss#'s) with frequency	Number of (ss#'s) with genotype
HOMO SAPIENS	17176850	9098790 (4267639)	561124	224830
ANOPHELES GAMBIAE	1368805	1139947 (0)		
MUS MUSCULUS	553539	498463 (468328)		501732
Canis familiaris	975400	975400 (0)		
Saccharum hybrid cultivar	42853	42853 (0)		
RATTUS NORVEGICUS	45103	31971 (35)		182
GALLUS GALLUS	11796	11678 (112)		
DANIO RERIO	2031	2025 (1903)		2031
SUS SCROFA	1545	1521 (24)		
CAENORHABDITIS ELEGANS	1065	1065 (0)		
Cooperia oncophora	426	426 (96)	425	
OVIS ARIES	680	614 (66)		
GLYCINE MAX	281	278 (3)		281
ARABIDOPSIS THALIANA	184	184 (184)		184
PLASMODIUM FALCIPARUM	203	203 (0)		199
ZEA MAYS	148	146 (80)		90
BOS TAURUS	54	32 (29)	54	
PINUS PINASTER	132	32 (0)		
FICEDULA ALBICOLLIS	37	37 (18)	37	
FICEDULA HYPOLEUCA	28	20 (10)	28	
HYDROIDES ELEGANS	8	7 (1)		
Schistosoma mansoni	8	(0)	2	
G. GORILLA	4	4 (0)		
PAN TROGLODYTES	2	2 (0)	2	

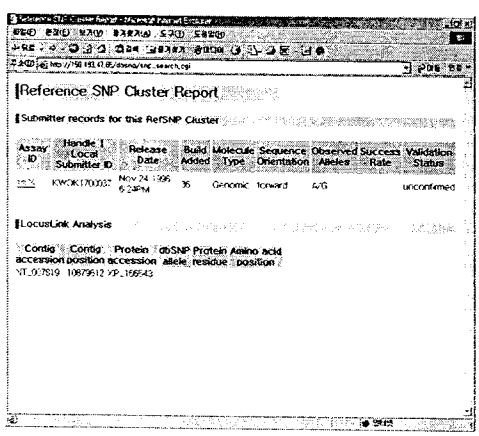
(3) dbSNP 서비스

dbSNP 데이터베이스 검색방법으로는 제출자 이름(handle)로 검색, SNP를 찾은 제출자에 의해 사용된 방법으로 검색, 연구된 모집단 형태를 통한 검색 , 문헌 제목으로 검색하는 방법이 있다. <figure 2-6>은 dbSNP 초기 검색 서비스 사용자 인터페이스를 보여주고 있다.

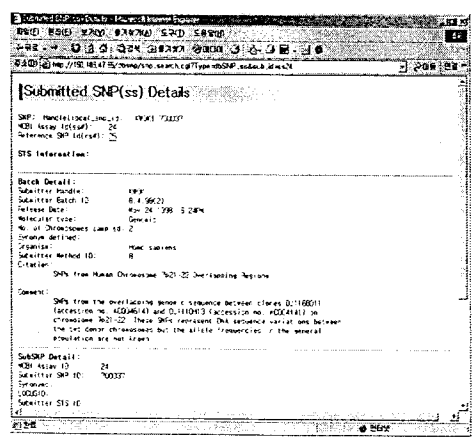
<figure 2-6> dbSNP main page 화면



<figure 2-7>은 ID를 이용한 검색 결과를 보여주고 있고, <figure 2-8>은 해당 SNP의 상세 정보를 보여주고 있다.

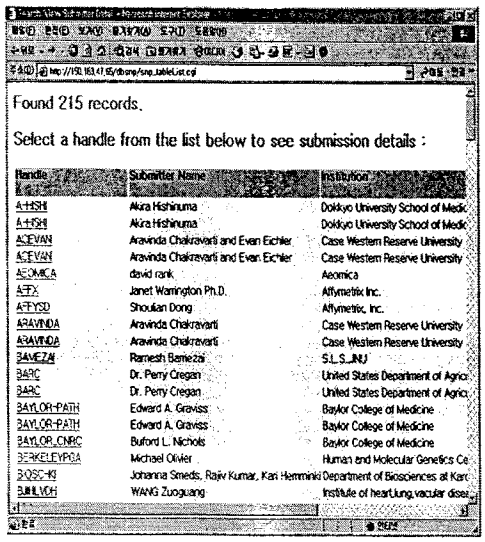


<figure 2-7> Search result by ID

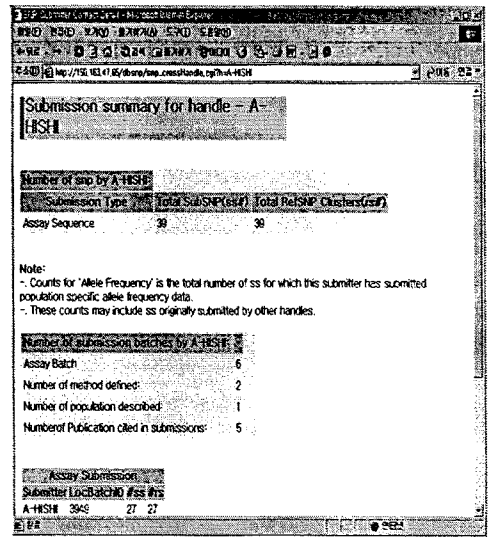


<figure 2-8> Detail Information

<figure 2-9>는 제출자에 의한 검색 결과를 보여주고 있고, <figure 2-10>은 관련된 제출요약 정보를 보여주고 있다

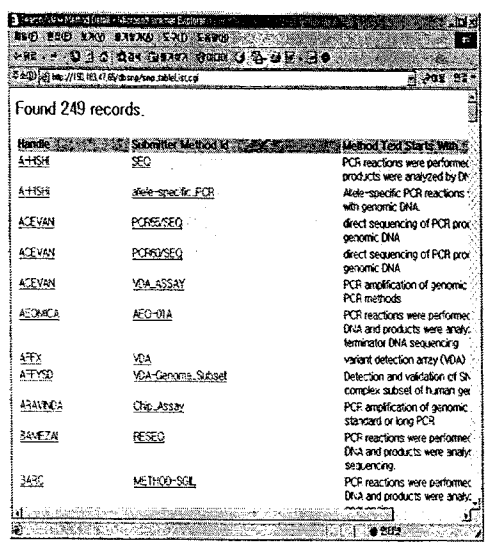


<figure 2-9> Search result by submitter

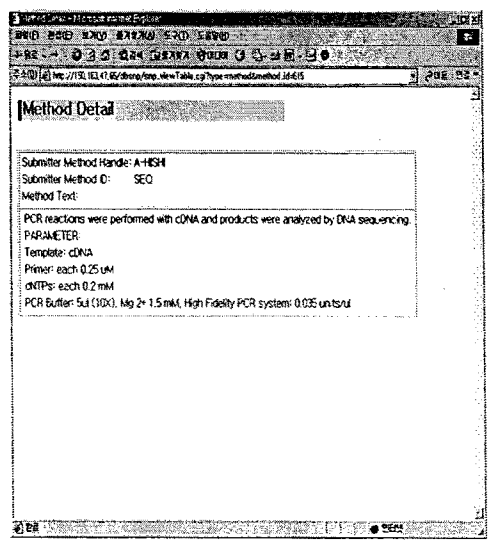


<figure 2-10> Submission Summary

<figure 2-11>은 사용된 방법에 의한 검색결과를 보여주고, 즉 어떤 실험방법이 이용되었는지 확인할 수 있다. <figure 2-12>는 해당 방법의 상세한 내용을 보여주고 있다.



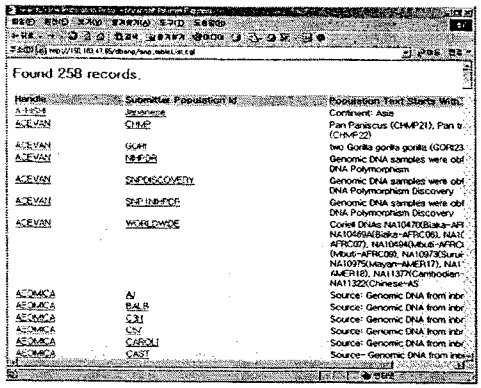
<figure 2-11> Search result by method



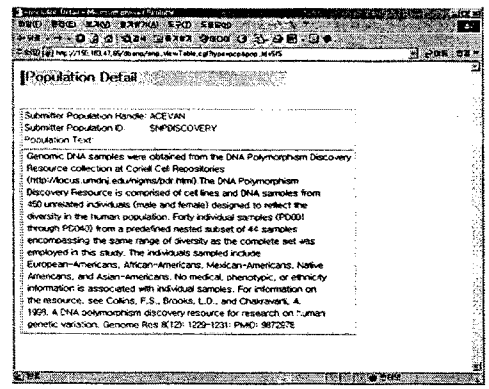
<figure 2-12> Method Detail information

<figure 2-13>는 연구된 모집단 형태를 통한 검색결과 보여주고, <figure 2-14>는 해당

모집단의 상세 정보를 보여주고 있다.

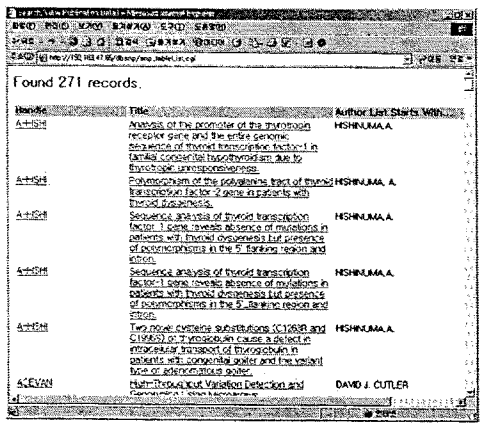


<figure 2-13> Search result by population

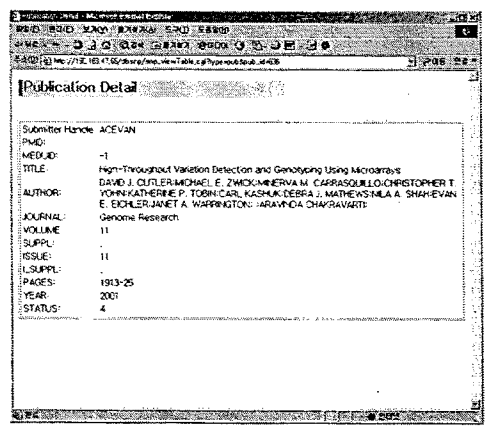


<figure 2-14>. Population detail information

<figure 2-15>은 문헌 제목으로 검색한 결과를 나타내고 <figure 2-16>은 해당 문헌의 상세정보를 보여주고 있다.



<figure 2-15> Search result by publication title



<figure 2-16> Publication detail information

라. 결과 활용 및 기대효과

SNP는 유전변이 비율이 낮고, 인체 게놈 상에서의 발생 빈도가 높아 고효율 유전자형 분석에 매우 유용하다. 따라서, physical mapping 및 genetic mapping을 위한 marker로서 사용될 수 있으며 최근에는 'complex genetic traits을 연구하기 위한 marker로서 각광 받고 있다. 이론적인 모델에 따르면, 'Linkage Disequillibrium' 현상에 기초하여 병에 걸린 집단과 병에 걸리지 않은 집단사이에 유전자형을 비교할 경우 특정 유전자형이 병에 걸린 집단의 유전자형과 연관성이 보여지게 되는데, 이러한 현상의 연구를 통하여 특정 질병과

연관된 돌연변이 근처에 marker를 mapping할 수 있게 됨으로써 질병과 관련된 유전자를 발견하는 연구를 수행할 수 있다.

다. BIND (Biomolecular Interaction Network Database)

(1) 개요

최근 많이 사용하고 있는 high-throughput 방법 즉, mass-spectrometers, gene chips, two-hybrid systems 으로 세포내에서 일어나는 단백질들의 interaction이나 단백질의 기능을 확인하는 Proteomics는 결국 Human Genome Project 완성 때 보다 더 많은 생물학적 데이터를 만들어 낼 것이다. 이렇게 많은 데이터들의 정확성을 높이고 오류데이터를 줄이기 위해서는 알고리즘 적으로 해결하려는 computational 방법이 필요하게 되었고 그 외에도 다양한 방법들이 시도 되고 있다.

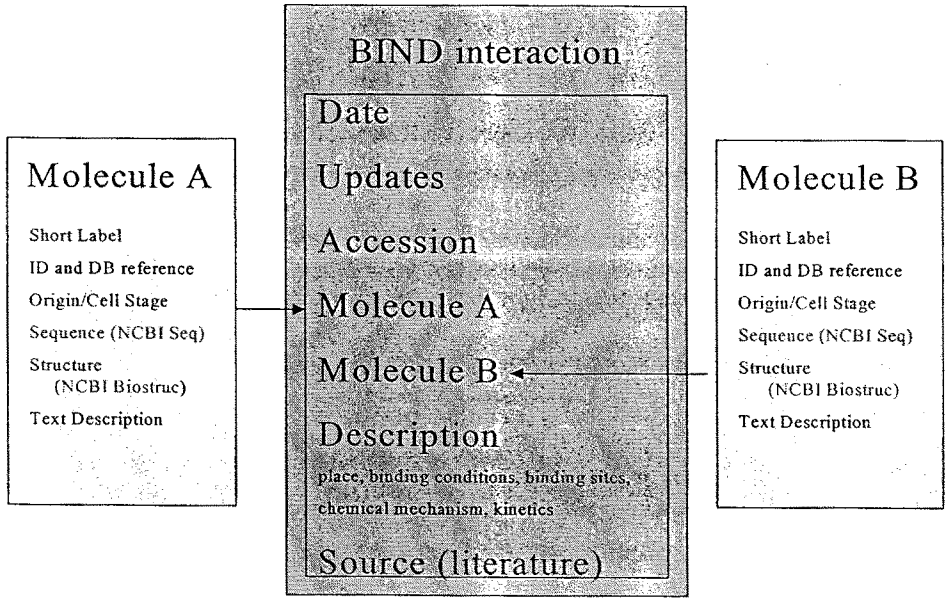
세포내에서 발현된 각각의 단백질들은 기능면에서 여러 가지 다른 단백질과 분자들 사이에서 interaction 할 수 있는데 이러한 정보들은 연구자들에게 중요하므로 DB로 구축하여 interaction 정보를 많은 곳에서 제공하고 있다. 다른 데이터베이스와 다르게 BIND(Biomolecular Interaction Network Database)에서는 DNA, RNA, Protein 각각의 기능을 확인하면서 DNA에서 RNA, RNA에서 Protein으로 변화하는 생물학적 대사경로를 확인할 수 있을 뿐만 아니라 저널에 발표된 논문을 링크함으로서 실험적인 기법을 소개하고 있다.

BIND는 interactions, molecular complexes, pathways 등의 유용한 정보들을 텍스트 형식으로 제공하고 있다. 분자 수준의 interaction, high-throughput data를 포함하며 논문에서 수집된 자료를 전문가들이 직접 확인한 검증된 데이터베이스이다. 각 자료들은 분자 interaction과 그것들의 관련된 중요성을 나타내고 연결된 자료들의 집합에 의해 표현된다.

분류 : Interactions, Molecular Complexes, Pathways

(가) Interactions

두 개 이상의 DNA, RNA, protein, ligand, molecular complex, gene, photon or an unclassified biological entity들 사이의 interaction을 표현한다. interaction에는 분자의 위치, 화학적인 반응, 화학적인 상태, 열역학, 동역학 및 interaction을 관찰하기 위해 사용되는 실험적인 조건과 세포내의 위치가 포함되어 있다. 다음 <figure 2-17>는 Molecule A와 Molecule B의 interaction을 나타내고 있는 레코드 형식이다.



<figure 2-17> Molecular A, B의 interaction record 형식

(나) Molecular Complexes

한 Organism에서 function unit의 형성과 관련하여 2개 이상의 분자들의 결합 내용 등을 정리하였고, interaction에 관련된 subunits의 수와 복합 토폴로지에 관한 정보가 구현되어 있다.

(다) Pathways

한 Organism에서 순서가 있는 interaction에서 발생하는 2개 이상의 interaction의 결합 내용 등을 DB화 하였다. 또한, pathway가 실제 질병과 관련이 있는지와 존재하는 세포 주기가 어떤 단계인지에 관한 정보가 구현되어 있다.

(2) 데이터 서비스 시스템

BIND DB 구축을 위해 http://www.blueprint.org/에서 ftp로 제공하는 데이터를 다운받아 구현하였다. 데이터는 fasta, xml 형식 등으로 제공하며 서비스에 필요한 요소를 추출하였고, MySQL DBMS를 사용하여 데이터를 저장하였다. 또한 MySQL 접근 클라이언트 및 웹 어플리케이션 등을 개발하여 Keywords, ID 검색 등으로 서비스 할 수 있도록 하였다.

(가) 데이터 수집 및 가공

DB구축은 xml 데이터를 사용하였고, 데이터 처리에 필요한 프로그램을 개발하였

다. 데이터 처리를 위해 BIND.dtd 문서를 분석하였으며, 필요한 요소를 추출하여 스키마를 정의 하였다. 다음 <table 2-2>는 BIND 데이터를 정의 해놓은 DTD문서의 일부분을 나타내고 있다.

<table 2-2> BIND DTD 문서

```
<!-- NCBI DTD                    -->
<!-- NCBI ASN.1 mapped to XML -->

<!-- Entities used to give specificity to #PCDATA -->
<!ENTITY % INTEGER '#PCDATA'>
<!ENTITY % ENUM 'EMPTY'>
<!ENTITY % BOOLEAN 'EMPTY'>
<!ENTITY % NULL 'EMPTY'>
<!ENTITY % REAL '#PCDATA'>
<!ENTITY % OCTETS '#PCDATA'>

<!ELEMENT BIND-Submit-id ( %INTEGER; )>

<!-- Definition of BIND-accession-number -->
<!ELEMENT BIND-accession-number (
                BIND-accession-number_interaction |
                BIND-accession-number_complex |
                BIND-accession-number_pathway )>

<!ELEMENT BIND-accession-number_interaction ( Interaction-id )>
<!ELEMENT BIND-accession-number_complex ( Molecular-Complex-id )>
<!ELEMENT BIND-accession-number_pathway ( Pathway-id )>
          ..... 중략 .....
<!ELEMENT BIND-Interaction-set (
                BIND-Interaction-set_date? ,
                BIND-Interaction-set_database? ,
                BIND-Interaction-set_interactions )>

<!ELEMENT BIND-Interaction-set_date ( Date )>
<!ELEMENT BIND-Interaction-set_database ( BIND-Database-site )>
<!ELEMENT BIND-Interaction-set_interactions ( BIND-Interaction+ )>
          ..... 이하 생략 .....
```

데이터 파싱 프로그램은 각 해당 테이블에 로딩할 수 있도록 모듈별로 구현하여 데이터를 처리하였다. 데이터 테이블과 스키마들은 다음 <table 2-3>~<table 2-10>와 같이 설계 하였고, Bind ID로 Primary Key를 정의하여 테이블간의 관계를 설정하였다. BIND DB 구축에 필요한 테이블은 총 8개로 정의하였으며 향후 추가정보에 따라 늘어날 수 있다.

<table 2-3> Interaction 데이터의 기본정보 테이블

Column	Data Type	Description
bid	integer	BIND ID(Primary Key)
label_a	varchar(30)	molecular A의 short label
label_b	varchar(30)	molecular B의 short label
names_a	varchar(255)	molecular A의 Alias 명
names_b	varchar(255)	molecular B의 Alias 명
descr_a	text	molecular A에 대한 설명
descr_b	text	molecular B에 대한 설명
gi_a	integer	molecular A의 GI정보
gi_b	integer	molecular B의 GI정보
molecular_a	text	GO Annotation 링크 정보
molecular_b	text	〃
cellular_a	text	〃
cellular_b	text	〃
biological_a	text	〃
biological_b	text	〃
experiment	text	〃

<table 2-4> Interaction 데이터의 기타정보 테이블

Column	Data Type	Description
bid	integer	BIND ID(Primary Key)
date	varchar(12)	등록날짜
descr	text	Interaction 설명정보
history	char(1)	flag
type_a	varchar(30)	Molecular Type
type_b	varchar(30)	〃
domain_a	varchar(255)	Domain 정보
domain_b	varchar(255)	〃
small_a	varchar(255)	complex small 정보
small_b	varchar(255)	〃
dbname_a	varchar(255)	기타 DB 링크정보
dbname_b	varchar(255)	〃
intp_a	varchar(255)	
intp_b	varchar(255)	
strp_a	varchar(255)	
strp_b	varchar(255)	

<table 2-5> BIND 데이터의 업데이트 정보 테이블

Column	Data Type	Description
bid	integer	BIND ID(Primary Key)
date	varchar(12)	업데이트 날짜
descr	text	업데이트 내용

<table 2-6> Complex 데이터의 기본정보 테이블

Column	Data Type	Description
bid	integer	BIND ID(Primary Key)
unit	integer	Subunit ID
label	varchar(30)	Molecular label
names	varchar(255)	Molecular names
gi	varchar(10)	GI id
descr	text	Complex subunit의 설명정보
molecular	text	GO Annotation 링크 정보
cellular	text	〃
biological	text	〃

<table 2-7> Complex 데이터의 기타정보 테이블

Column	Data Type	Description
bid	integer	BIND ID(Primary Key)
date	varchar(15)	등록날짜
descr	text	Complex의 설명정보
history	char(1)	flag
unit	char(3)	Subunit 갯수

<table 2-8> Pathway 데이터의 기본정보 테이블

Column	Data Type	Description
bid	integer	BIND ID(Primary Key)
date	varchar(12)	등록날짜
descr	text	Pathway의 설명정보
history	char(1)	flag

<table 2-9> Publication 데이터의 기본정보

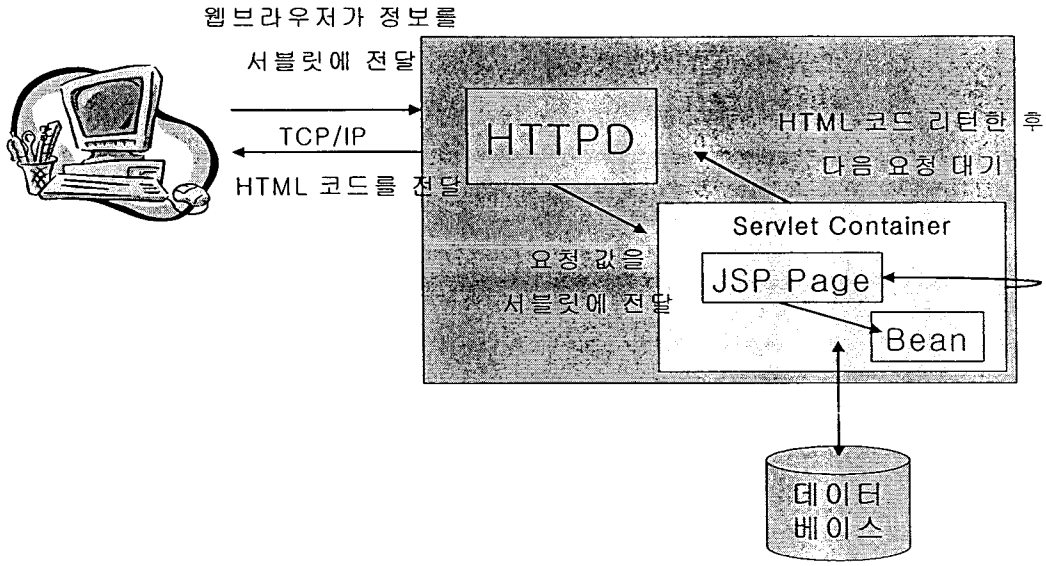
Column	Data Type	Description
bid	integer	BIND ID
pmid	integer	PubMed ID
descr	text	간략설명정보
dispute	varchar(10)	기타
opinion	varchar(10)	기타

<table 2-10> Interaction과 complex, pathway 데이터의 관련성 정보

Column	Data Type	Description
cpid	integer	complex, pathway ID
interaction	integer	interaction ID

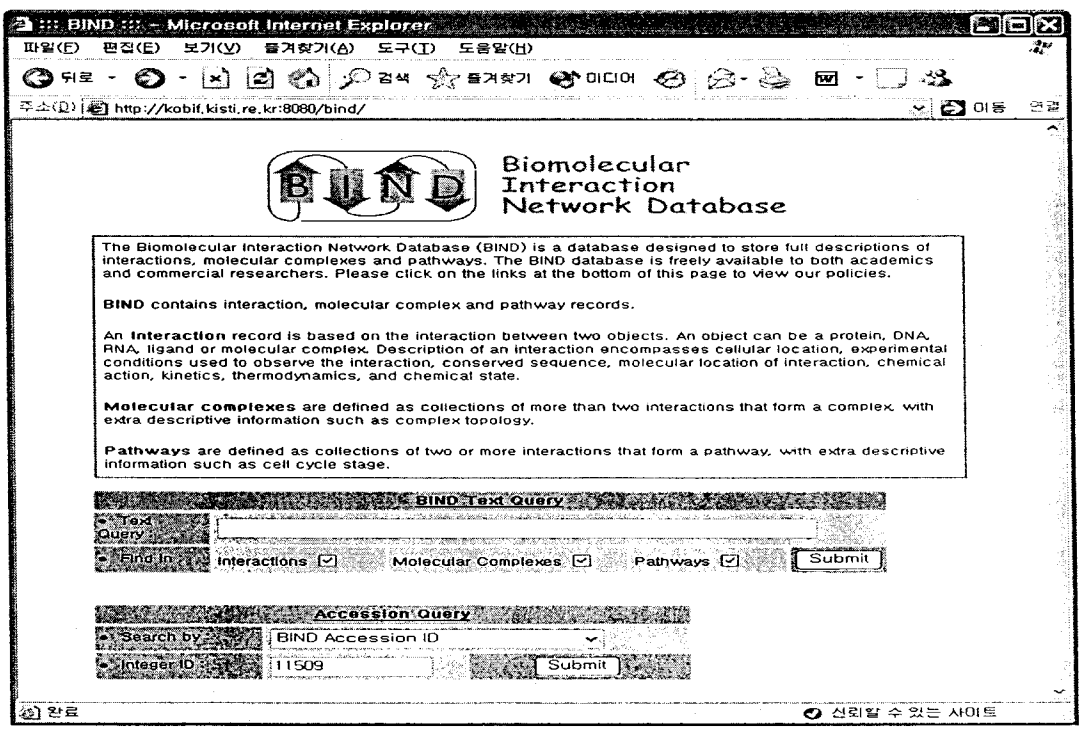
(3) 검색 인터페이스

데이터베이스 구축 시스템은 <figure 2-18>과 같으며, MySQL 접근 클라이언 및 정보검색 인터페이스를 개발하였다. 사용자는 웹 브라우저를 이용하여 BIND 데이터를 쉽고 간편하게 검색결과를 얻을 수 있다.



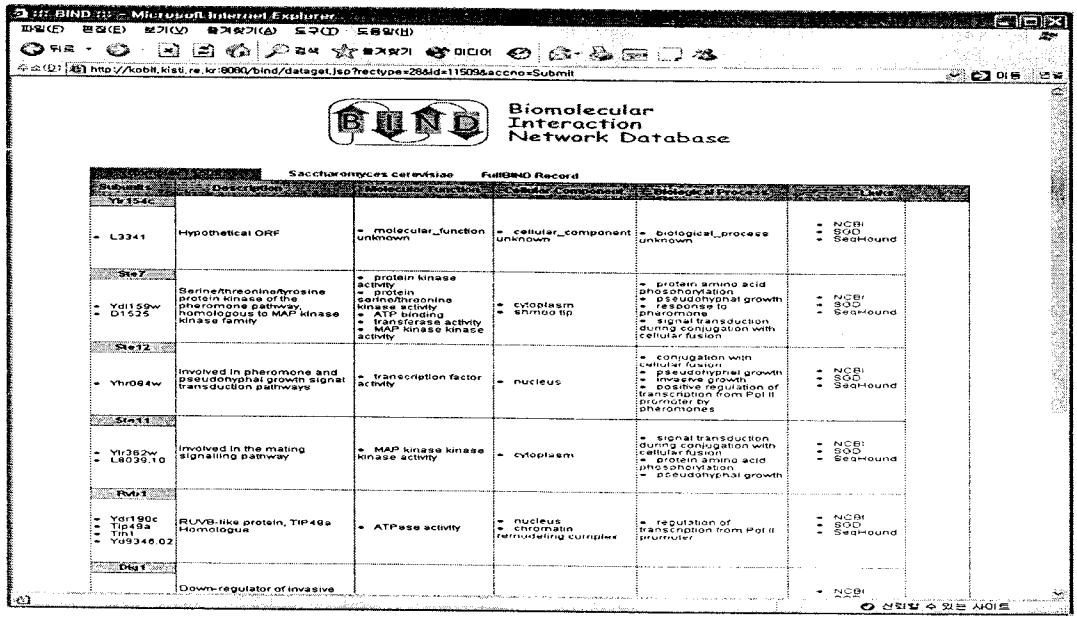
<figure 2-18> BIND DB 구축 시스템 구조

<figure 2-19>은 BIND 데이터베이스의 홈페이지를 보여주며, Keyword와 ID로 검색할 수 있다. ID검색은 선택적으로 BIND, GI, PubMed ID로 데이터에 접근 할 수 있으며, 타 데이터베이스로 연결하여 많은 정보를 얻을 수 있다. 검색서비스 화면은 사용자 편의성을 위해 단순화하여 검색에 관한 전문적인 지식이 없어도 손쉽게 검색할 수 있도록 하였다.

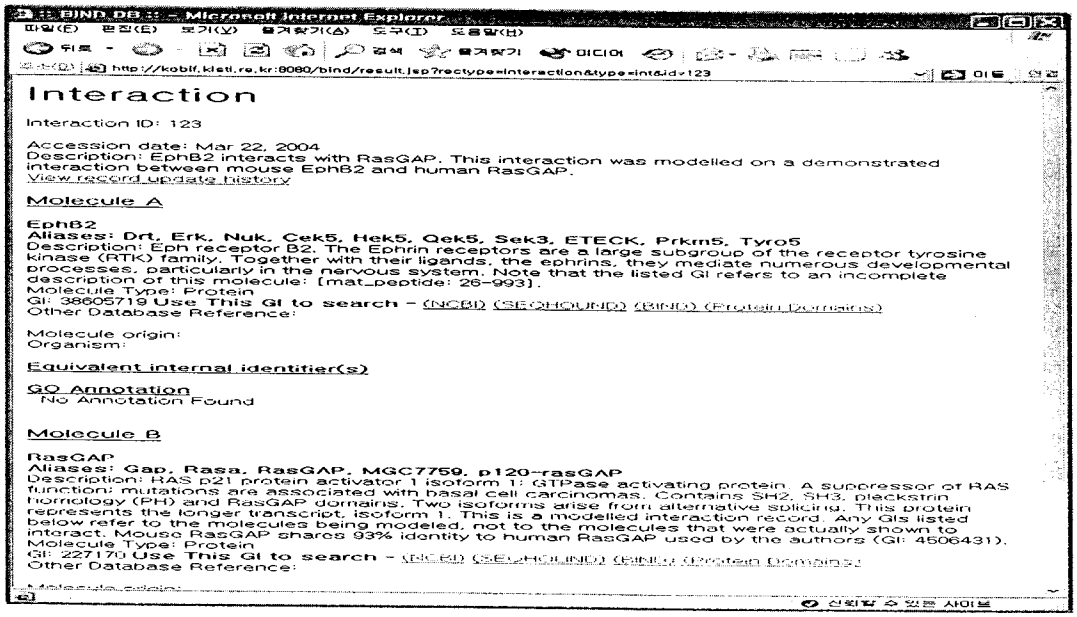


<figure 2-19> BIND DB 서비스 메인 페이지

<figure 2-20>는 BIND 데이터베이스 검색결과에 대한 간략정보를 보여주는 화면이다. 사용자는 찾고자 하는 Keyword 또는 ID를 입력하여 검색을 하면 간략정보를 얻을 수 있다. 간략정보는 BIND 데이터에서 중요한 정보를 한눈에 보여주고, 원하는 정보가 있다면 선택적으로 상세정보를 얻을 수 있다. <figure 2-21>>은 한 Interaction에 대해서 상세정보를 보여주는 화면이다.

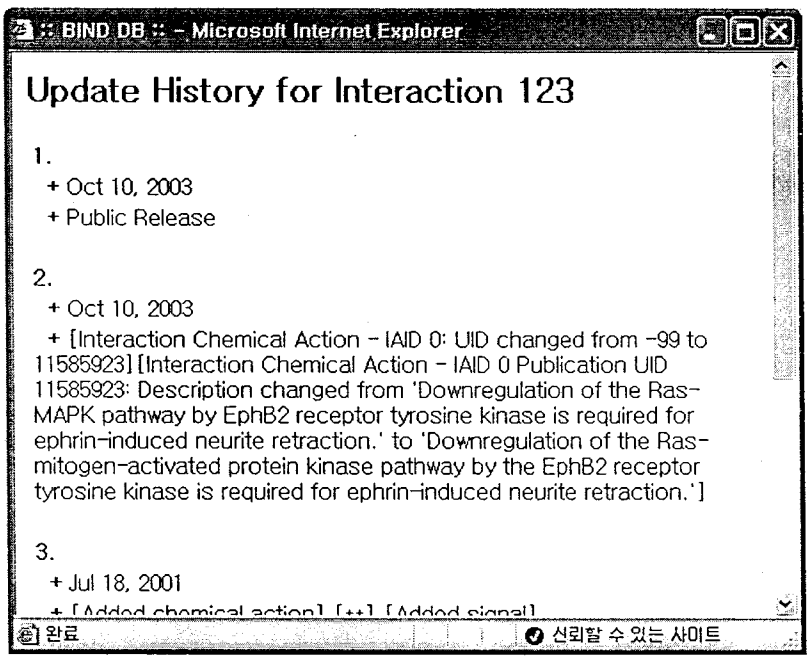


<figure 2-20> 검색결과의 간략정보

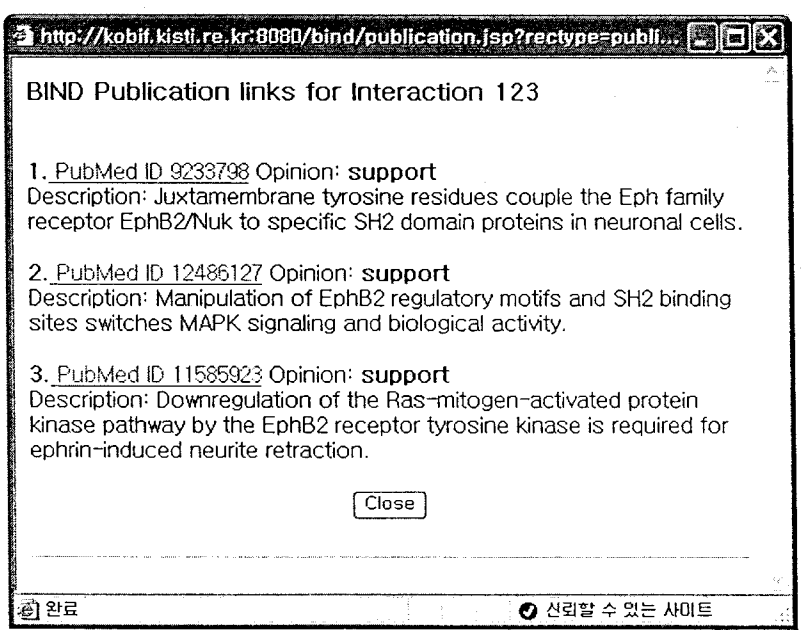


<figue 3-21> 검색결과 전체 내용

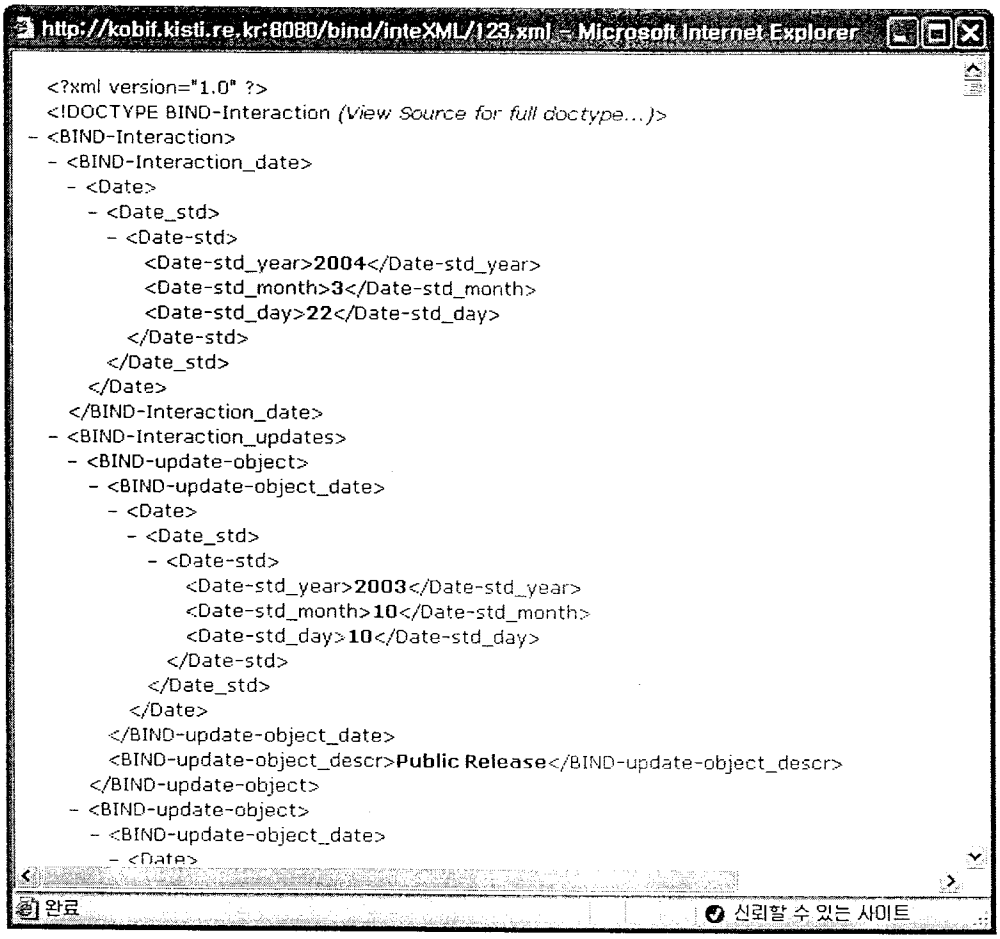
상세정보는 날짜, Interaction 설명정보, Update history, Molecular A, Molecular B, GI ID 등을 나타내고 있다.
<figure 2-22>~<figure 3~24>은 상세정보 외에 기타정보를 보여주는 화면이다. 이와 같은 구성으로 Interaction, Molecular complex, Pathway 에 대한 내용을 웹을 통하여 정보들을 제공하도록 구축하였다.



<figure 2-22> BIND Update 내용



<figure 2-23> Publication 정보



<figure 2-24> XML 문서

라. DIP (Database of Interacting Proteins)

(1) 개요

DIP 데이터베이스는 '상호작용'한다고 알려진 단백질을 나열하는 것으로, 여기에서. 상호작용(interact)은 실험적으로 식별된 2개의 아미노산이 서로 바인드(bind)하는 것을 뜻한다. DIP은 특정의 PPI(Protein-Protein Interaction, 단백질간 상호작용) 연구, 조절과 신호 체계, 세포 레벨의 단백질 상호작용 네트워크의 조직과 복잡성의 연구에 도움을 주고 있다. DIP 데이터베이스에 저장된 데이터는 전문가가 직접 확인하고 단백질 상호작용 네트워크 지식을 이용하는 계산적인 접근방법을 사용하여 검증된 것이다.

DIP 데이터베이스는 노드(nodes)와 에즈(edges)로 구성된다.

- DIP 노드(단백질)

DIP 상호작용에 관여하는 각각의 단백질은 <DIP:*nnn*N> 형태의 유일한 식별자를 가지고 적어도 중요 단백질 DB - PIR, SWISSPROT 그리고 GBNBANK에 상호 참조를 하고 있다. 또한 상호참조 DB를 접근할 수 없을 경우에는 각 단백질의 기본 정보(이름, 기능, 국부세포 지역화와 다른 생물데이터베이스의 상호참조)가 저장된다 (<table 2-11>).

- DIP 에즈(상호작용)

DIP의 상호작용 정보는 <DIP:*nnn*E> 형태의 유일한 식별자로 구별되고 에즈는 상호작용에 관련된 영역, 분해 상수, 상호작용을 식별하고 특징지우는 실험방법 등을 제공한다 (<table 2-12>).

<table 2-11> DIP 노드 테이블

항 목	설 명
DIP node ID	DIP 데이터베이스에서 각 단백질에 대한 유일한 식별자로 <DIP:nnnN>의 형태를 가진다.
Description/Name	단백질의 일반 이름과 이것의 짧은 서술
Primary Database Reference(s)	적어도 하나의 PIR, SWISSPROT 그리고 GENBANK 엔트리(entry)가 제공된다.
Cross-references	관심 단백질의 다른 데이터베이스 엔트리 참조 정보. 상호참조의 리스트는 (P)rotein, (D)omain, (F)eature (eg motifs)로 나뉜다.
Superfamily	PIR에 명시된 superfamily
Organism	단백질을 생산하는 생물체 TaxonID 데이터베이스의 참조정보가 제공된다.
Function	세포내 단백질 기능의 설명
Localization	세포내 단백질의 지역화
Keywords	단백질과 연관된 키워드로 단백질 구조, 서열 특징, 생물학적 행동, 긴응, 세포 지역화 등의 정보이다.

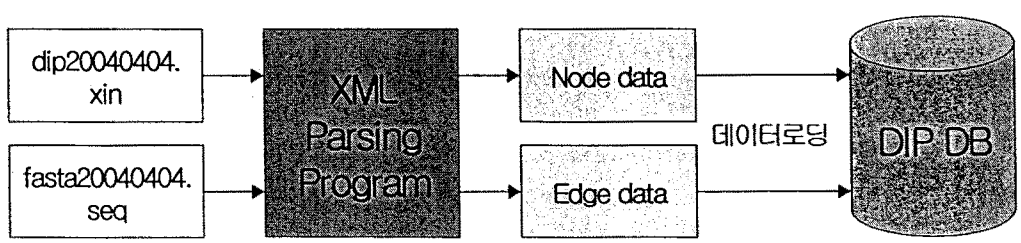
<table 2-12> DIP 에즈 테이블

항 목	설 명
DIP interaction ID	DIP 데이터베이스에서 서술되는 각 상호작용에 대한 유일한 식별자.
Residue Ranges	영역은 상호작용에서 단백질의 어느 부분이 더욱 관련되어 있는지 더욱 정확하게 명시하는 역할을 한다.
Protein Domain	상호작용하는 도메인 이름
Dissociation Constant	분리 상수
Experimental Method(s)	상호작용을 구별 또는 특징지우는데 사용되어진 실험 방법
Reference(s)	상호작용 연구 실험에 대한 참고문헌
Comment(s)	상호작용을 지정한 사람의 추가 설명.

(2) 구축

DIP 데이터베이스는 http://dip.doe-mbi.ucla.edu에서 서비스 되고 있고 본 과제에서는 이 데이터베이스를 CCBB에서 구축하여 서비스하는 것을 목표로 하고 있다.

데이터베이스를 구축하고 서비스하기 위해서 DIP에서 제공하는 검색서비스를 분석하였는데 DIP의 검색 서비스는 4개의 검색 서비스(Node, BLAST, Motif, Article)를 제공하고 있다. 위의 서비스를 하기 위하여 DIP 웹 사이트에서 관련된 파일을 다운로드 받아 목적에 맞게 가공하였다. DIP에서는 xin, fin의 두 가지 XML 파일 형식을 지원하는데 본 과제에서는 xin 파일 형식을 사용하였다.
본 과제에서는 MySQL DBMS를 사용하였고 데이터베이스(dip)에 필요한 테이블(NODE, EDGE)의 스키마를 구성하였다. 다음으로 DIP에서 제공하는 FULL 데이터 집합(dip20040404.xin)과 DIP 서열 파일(fasta20040404.seq)을 다운받아 각각의 스키마에 맞도록 레코드를 생성하였다. 이 때 원본파일의 XML 데이터를 파싱하기 위해 Java SAX 프로그래밍을 사용하였다. 마지막으로 데이터베이스(MySQL) 2개의 테이블에 각각의 파일을 적재하여 검색이 가능하도록 하였다. 전체적인 프로그램의 과정은 <figure 2-25>와 같다.



<figure 2-25> 데이터 처리과정

DIP 데이터베이스는 Node 테이블과 Edge 테이블로 구성되는데 각 스키마를 아래와 같이 구성하였다.

<table 2-13> Node 테이블의 구성요소

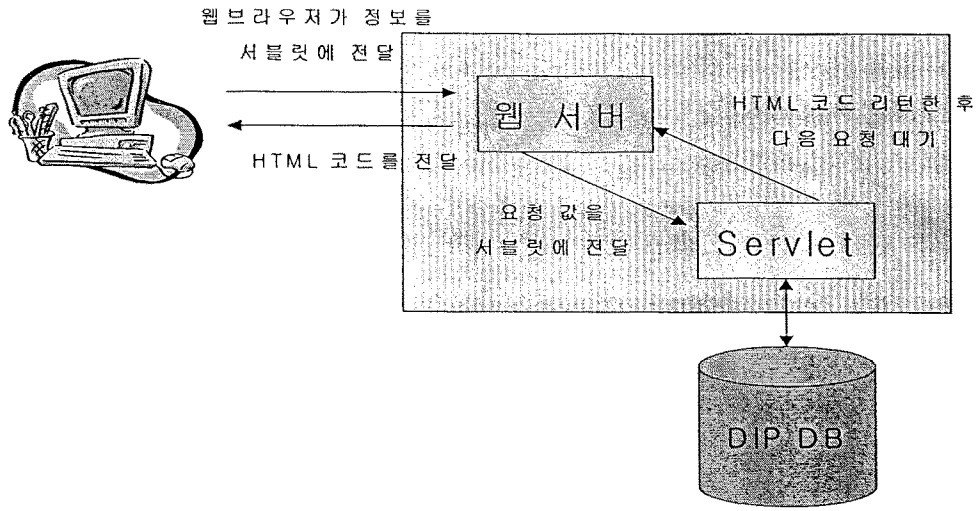
```
CREATE TABLE NODE (
        uid             VARCHAR(255)    NOT NULL,
        id              VARCHAR(255)    NOT NULL,
        name            VARCHAR(255)    NOT NULL,
        class_          VARCHAR(255)    NOT NULL,
        SWP             VARCHAR(255)    NOT NULL,
        PIR             VARCHAR(255)    NOT NULL,
        GBK             VARCHAR(255)    NOT NULL,
        descr           VARCHAR(255)    NOT NULL,
        taxon           VARCHAR(255)    NOT NULL,
        organism        VARCHAR(255)    NOT NULL,
        shn             VARCHAR(255)    NOT NULL,
        sequence        TEXT,

        PRIMARY KEY (uid),
        INDEX name (name),
        INDEX PIR (PIR),
        INDEX descr (descr),
        INDEX organism (organism)
);
```

<table 2-14> Edge 테이블의 구성요소

```
CREATE TABLE EDGE (
        uid             VARCHAR(200)    NOT NULL,
        id              VARCHAR(200)    NOT NULL,
        from_           VARCHAR(255)    NOT NULL,
        to_             VARCHAR(255)    NOT NULL,
        class_          VARCHAR(255)    NOT NULL,
        fUid            VARCHAR(255)    NOT NULL,
        fClass          VARCHAR(255)    NOT NULL,
        pmid            VARCHAR(255)    NOT NULL,
        val             VARCHAR(255)    NOT NULL,

        PRIMARY KEY (uid, fUid),
        INDEX uid (uid),
        INDEX pmid (pmid)
);
```

(3) 서비스

내부적으로 DIP의 정보를 검색하는 방법은 Java Servlet을 이용하였고 시스템의 구조는 <figure 2-26>과 같다.

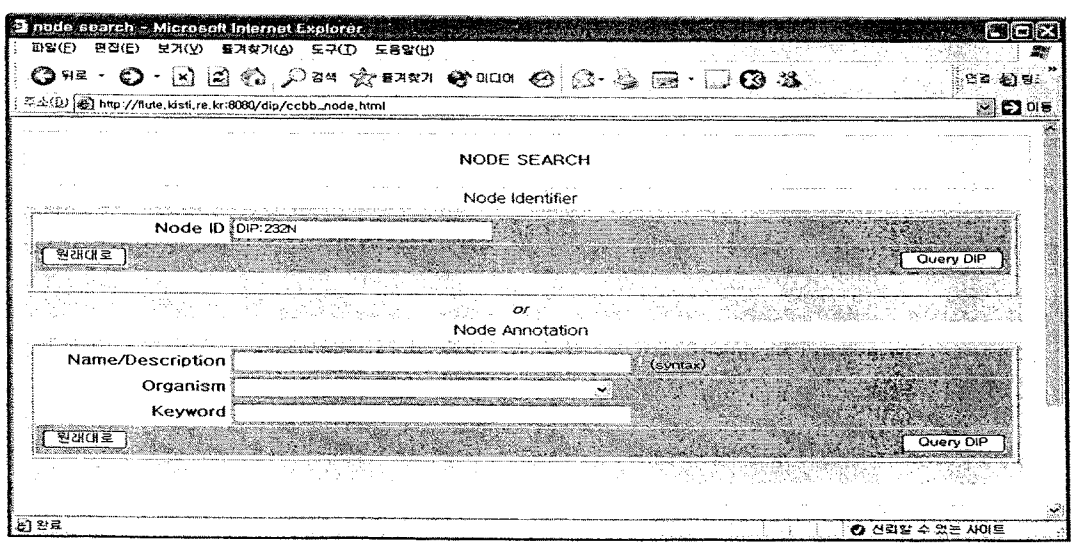


<figure 2-26> DIP 데이터베이스 서비스 시스템

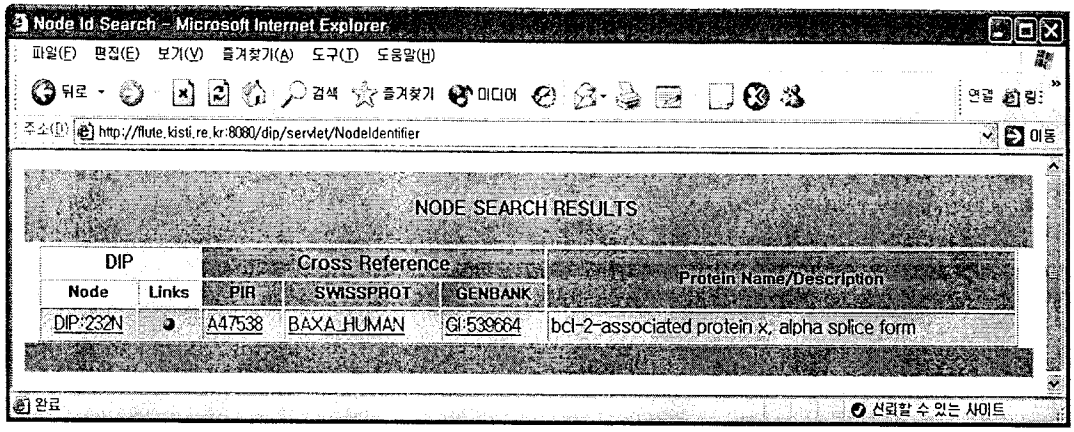
(가) Node 검색

DIP의 노드정보를 검색하는 서비스이다. 사용자는 노드 ID, 이름, 생물체, 키워드 등으로 검색할 수 있다.

다음 <figure 2-28>은 사용자가 DIP의 노드 ID로 검색을 한 경우의 결과를 보여주고 있다. Node, Links, PIR, SWISS-PROT, GenBank, Name/Description의 정보뿐만 아니라 각 사이트에 링크가 있어 더욱 자세한 정보를 참조할 수 있다.



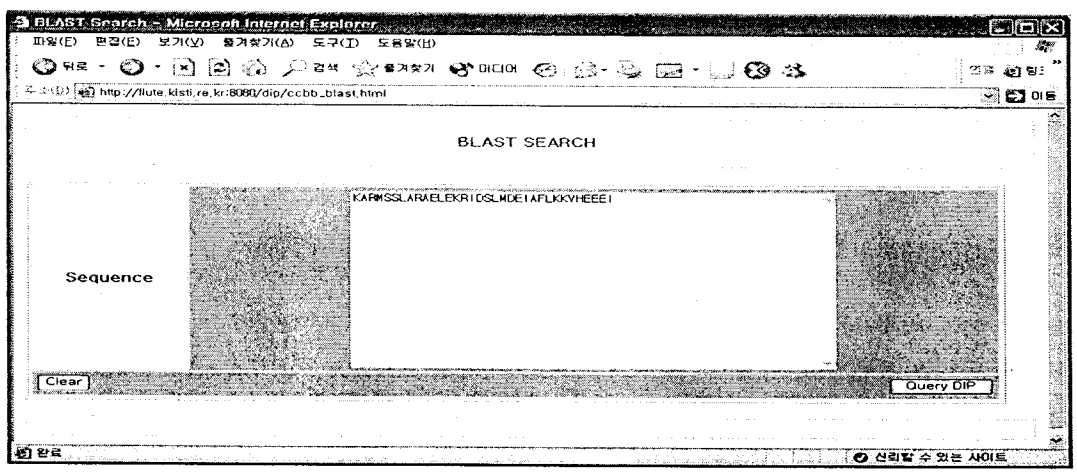
<figure 2-27> Node 검색



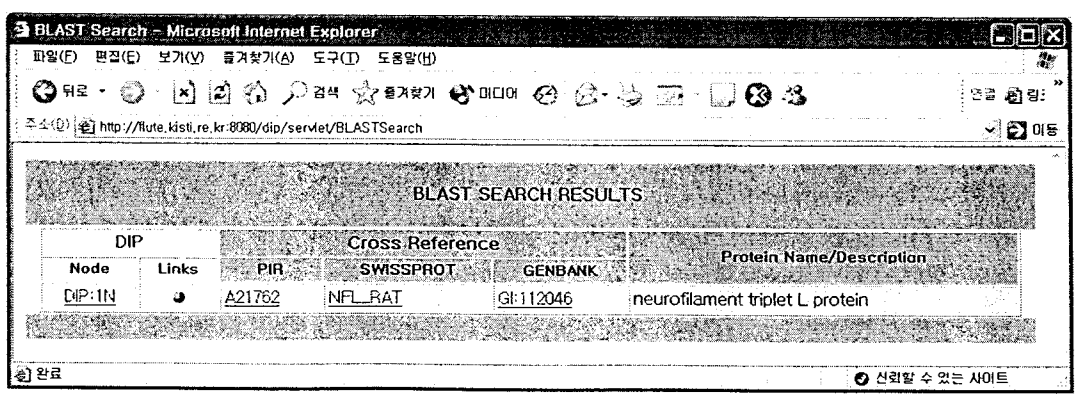
<figure 2-28> Node 검색 결과

(나) BLAST 검색

DIP 노드 서열의 부분 일치 검색을 할 수 있는 서비스이다. 검색된 결과는 Node 검색과 같다.



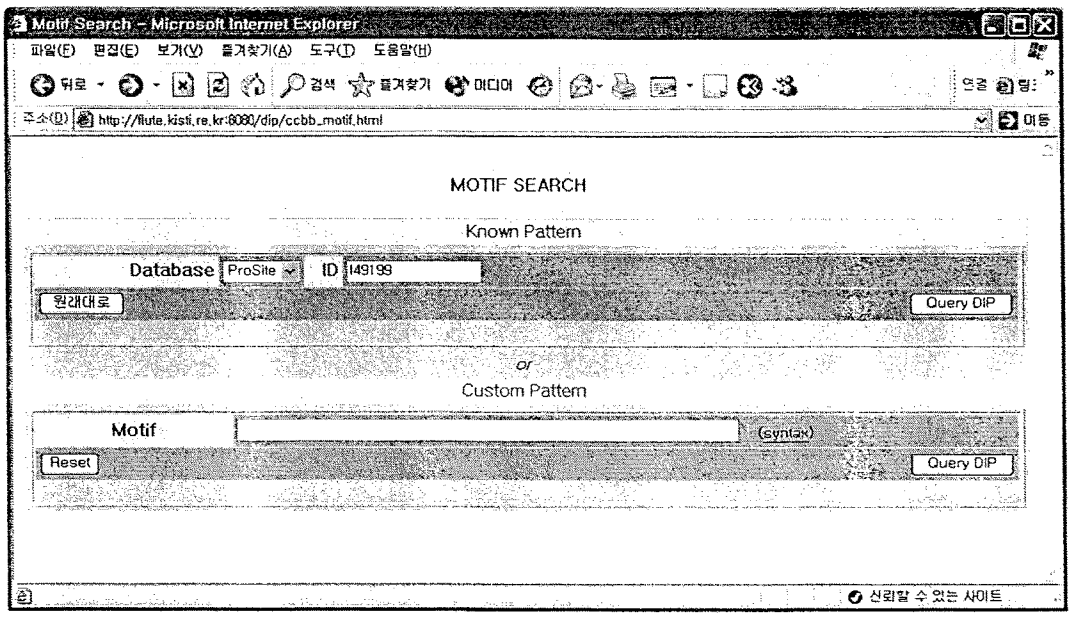
<figure 2-29> BLAST 검색



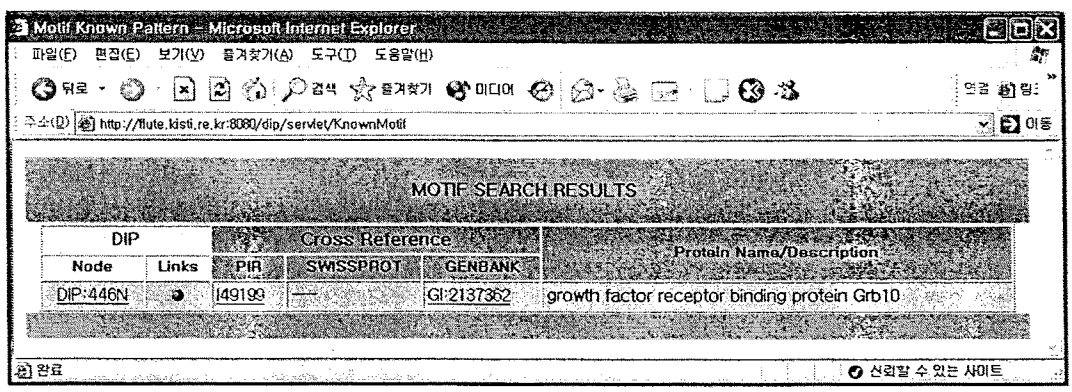
<figure 2-30> BLAST 검색 결과

(다) Motif 검색

Protesite ID 또는 사용자가 입력한 정규 표현식의 모티프를 검색하는 서비스이다.



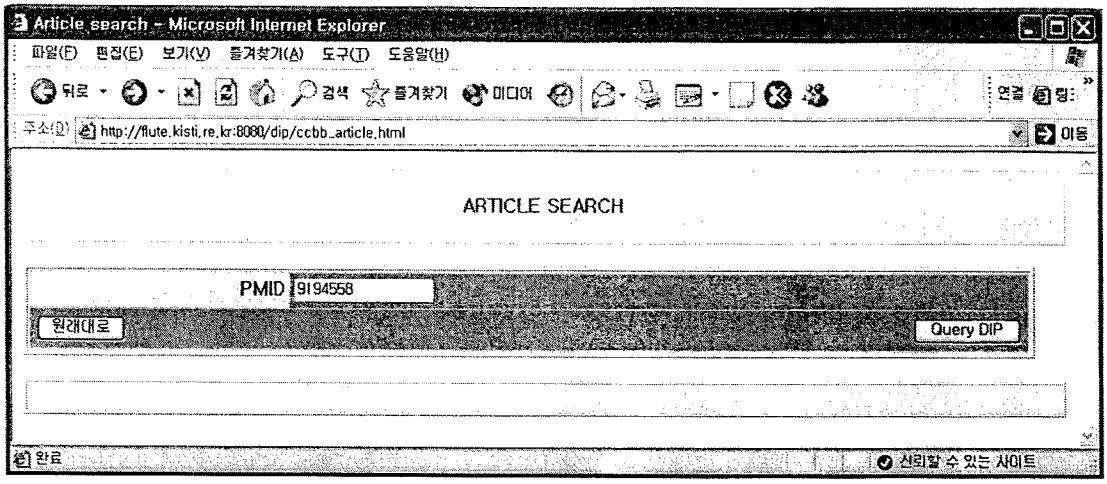
<figure 2-31> Motif 검색



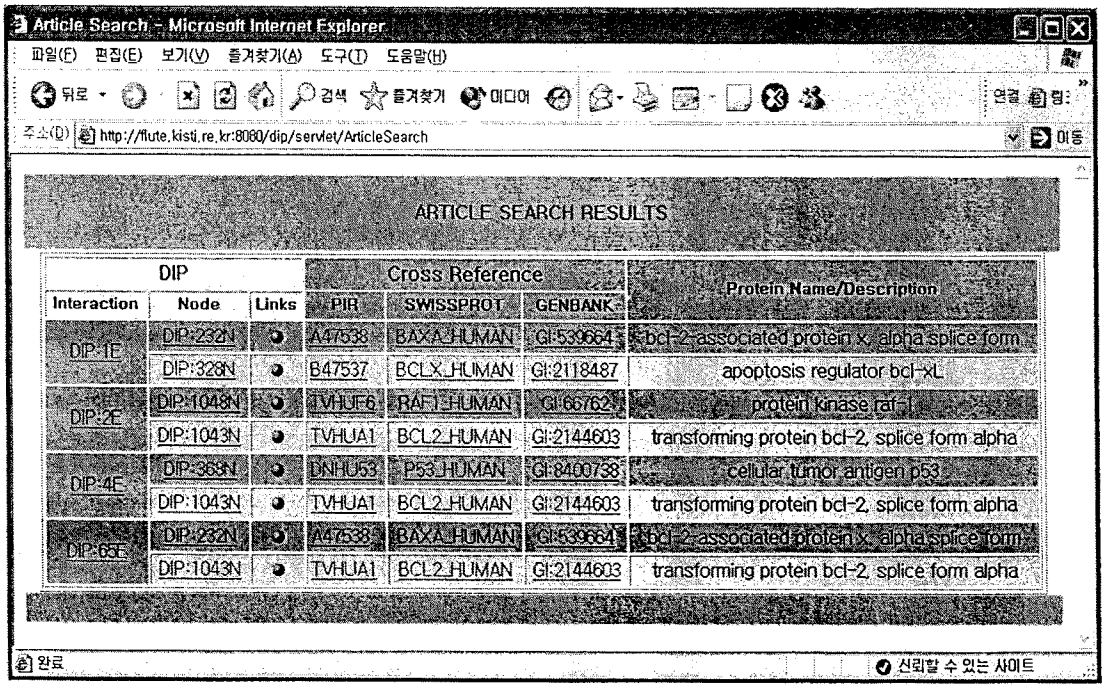
<figure 2-32> Motif 검색 결과

(라) Article 검색

DIP의 노드, 에즈 정보를 PubMed ID로 검색할 수 있는 서비스이다. 결과화면은 PubMed에 출간된 문헌에서 서술된 상호 작용하는 단백질의 리스트를 보여준다. DIP 웹사이트에는 PubMed ID이외에 저자, 타이틀, 저널, 볼륨, 년도로 검색할 수 있지만 본 과제의 서비스에는 현재 PubMed ID만을 검색할 수 있다. 그 이유는 다운로드받을 수 있는 DIP의 파일에 저자, 타이틀 등의 관련 정보가 없기 때문이다.



<figure 2-33> Article 검색



<figure 2-34> Article 검색 결과

(4) 기대효과

현재 DIP 데이터베이스의 레코드 건수는 NODE 테이블에 17,049건, EDGE 테이블에 48,903건이고 앞으로 원본 데이터 파일이 갱신될 때마다 CCBB의 DIP 데이터베이스도 갱신할 예정이다.. 단백질의 상호작용에 대한 정보를 저장하는 DIP 데이터베이스를 본 과제에서 구축, 서비스함으로써 국내의 연구자들이 신속하게 관련 정보를 검색하여 응용할 것으로 기대된다.

2. 데이터의 최신성 유지

가. 데이터의 최신성 유지

데이터의 최신성이 매우 중요하기 때문에 기 구축 데이터베이스를 매주 확인하여 실시간 업데이트가 이루어지도록 하고 있음. SCOP의 경우는 Cambridge MRC를 직접 미러링하고 있음.

<table 2-15> CCBB 데이터베이스의 최신성 유지 (2004년 5월 기준)

	해외 최신 버전	비고
GenBank dbEST dbGSS dbSTS	v 141 (2004. 4. 15)	
dbSNP	(2003. 10. 25)	
Ensembl	v 21 (2004. 5. 11)	
REBASE	v 405 (2004. 5. 3)	
PDB	(2004. 5. 11)	
PIR	v 79.02 (2004. 5. 9)	
SIWSS-PROT	v 43.2 (2004. 4. 24)	
PhiPsi	-	2003년 PDB 데이타[1)]
ProSeS ProFaC ProSLP	N/A	
CATH	v 2.5.1 (2004. 1)	
SCOP	v 1.65 (2003. 12)	정식 미러링
DIP	(2004. 1)	2004. 6월 완성
BIND	(2004. 3. 22)	2004. 6월 완성

1) 전체 PDB에서 얻어진 구조데이타를 통계처리하는 PhiPsi 데이터베이스의 특성상 최신 버전의 데이터베이스를 사용해도 거의 대동소이한 결과를 준다.

나. 데이터베이스 최신성 유지를 위한 FTP 모듈 개발

(1) 서론

생명정보 데이터의 끈임 없는 증가로 인하여 최신의 데이터베이스를 유지하고 관리하는 일은 쉬운 문제가 아니다. 새로운 데이터가 생성될 때마다 스키마 추가, 데이터의 가공, 저장 등을 통하여 생명정보 연구자들에게 신뢰성 있는 최신의 정보를 제공하여야 함은 물론, 기 구축된 데이터의 갱신도 바로바로 이루어져야 할 문제이다. 이 모든 것을 수작업으로 처리하는 것은 늦은 데이터의 갱신, 소요인력, 시간 등의 낭비라 할 수 있다. 따라서 본 과제에서는 이와 같은 문제점을 보완하고 원활한 유지보수를 위해 데이터베이스 갱신을 자동으로 처리할 수 있는 시스템을 개발하였다.

본 시스템의 목적은 기 구축된 DB를 자동으로 업데이트하여 빠르고 정확한 검색 서비스 제공을 위함이다. 이에 데이터의 최신성 유지에 필요한 CCBB FTP 모듈을 개발하여 자동으로 데이터를 다운받고 가공하여 DBMS에 로딩할 수 있도록 하였다.

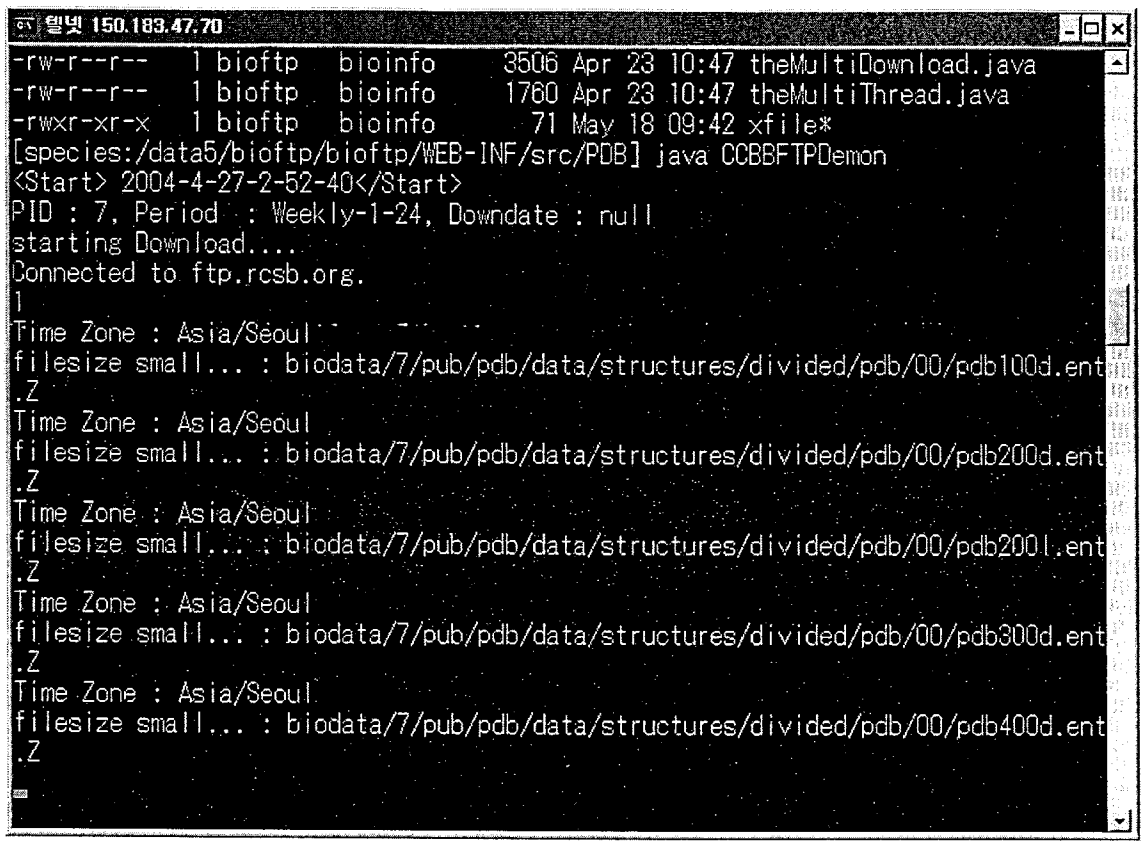
(2) 데이터관리자 GUI

데이터 최신성 유지에 필요한 모듈 개발의 목적은 수작업이 아닌 자동으로 처리할 수 있어야 한다. 이에 개발 환경으로 운영체제에 독립적이며 Network 처리에 많이 사용하고 있는 JAVA로 개발 하였으며 데이터를 저장할 수 있는 대용량 저장장치와 빠른 네트워크 환경으로 구성하였다. 또한 쉽고 간편한 관리를 위하여 웹 기반 인터페이스를 제공하도록 하였다.

<figure 2-35>은 CCBB FTP 모듈의 실행 화면이다. 프로그램은 데몬 형태로 실행되며 해당 사이트에 접속하여 로그정보를 체크하여 필요한 데이터를 다운받아 온다. CCBB FTP모듈 실행에 필요한 정보는 MySQL DB에 저장되며 다운받은 데이터는 특정 디렉토리에 저장된다.

데이터는 크기에 따라 다운로딩 속도가 나오는데 작은 용량의 데이터는 빠른 반면 큰 용량의 데이터는 오랜 시간 다운받아야 하는 문제가 있다. 이런 문제를 해결하기위해 Remote 사이트에 multi connection을 맺어 큰 용량의 데이터를 나눠서 받도록 처리하였다. 또한 데이터의 갱신날짜를 체크하여 이미 받아온 데이터와 새로

받아야 할 데이터를 결정하도록 하여 가장 빠르게 처리하도록 개발하였다.

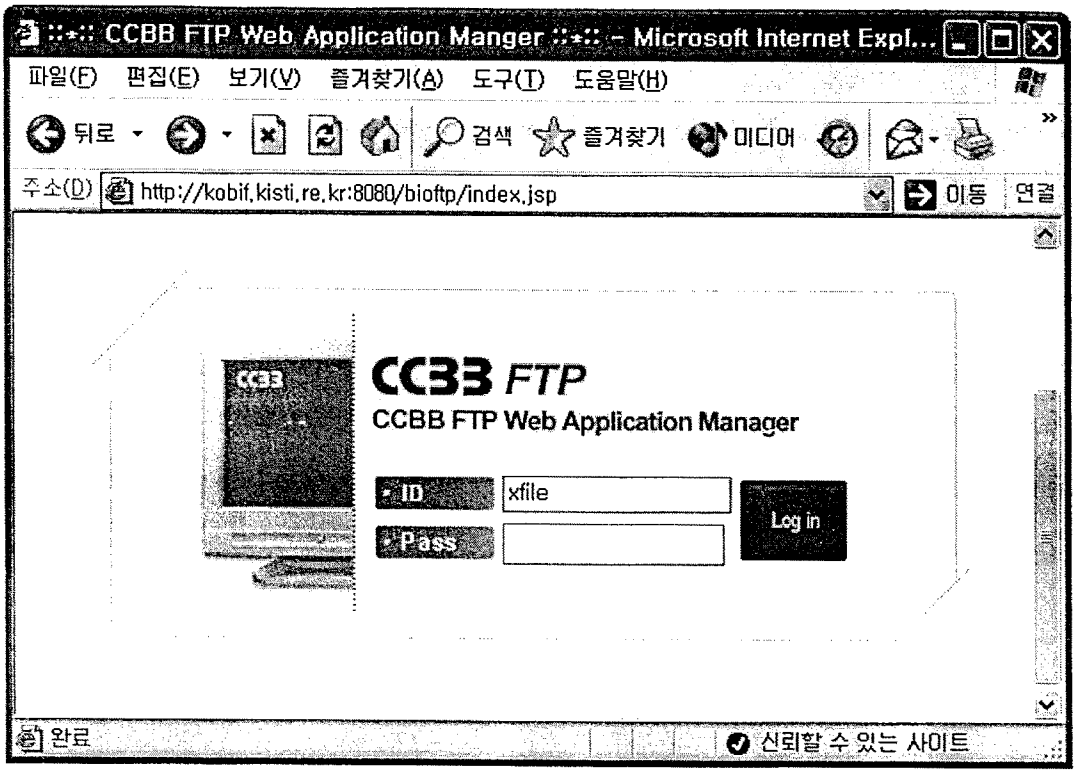


<figure 2-35> ccbb ftp 모듈 실행화면

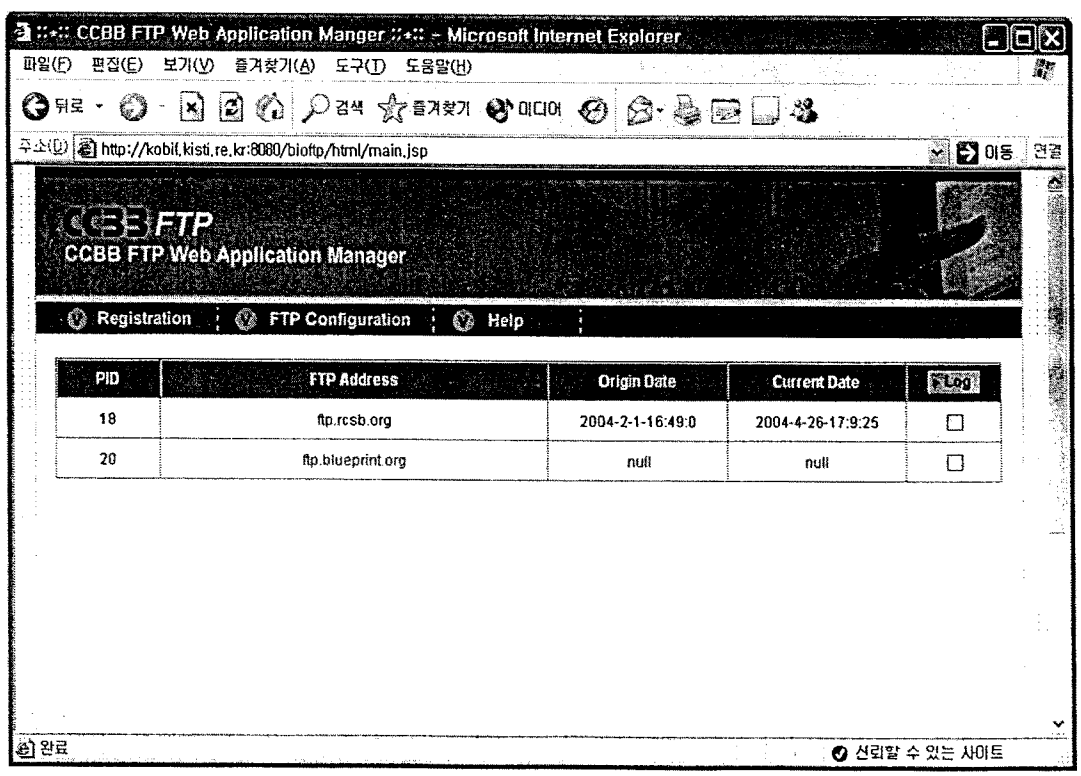
<figure 2-36>는 웹 브라우저를 이용하여 손쉽게 관리할 수 있는 관리자 로그인 화면이다. 관리자는 로그인을 함으로서 환경설정과 데이터 업데이트 상황정보를 관리할 수 있다 (<figiure 3-37> ~ <table 2-40>).

웹 기반으로 제공하는 내용은 데이터에 대한 업데이트 상황정보(파일명, 업데이트 날짜, 접근경로, 소스 URL, 업데이트 주기 등)을 게시판 형태로 보여준다. 관리자는 FTP 사이트, 디렉토리, 파일명, 확장자, 업데이트 주기 등의 정보를 입력할 수 있게 되며, 이러한 조건에 따라 데이터를 다운로딩 하게 된다. 데이터에 대한 Log 정보와 Configuration 관리정보 등은 데이터베이스에 저장하도록 하였다. 또한 실행시킬 수 있는 스크립트 명령어를 처리하도록 하였으며, 필요시에는 추가로 스크립트를 등록 하도록 하였다.

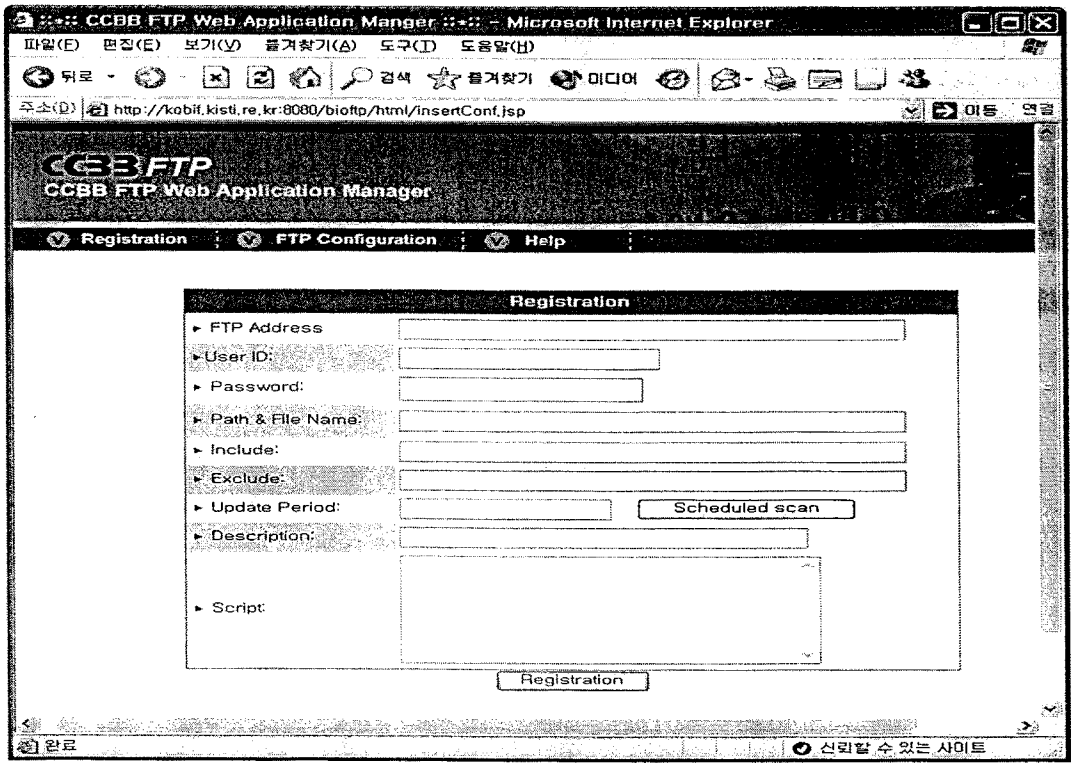
본 CCBB FTP 모듈은 그동안 많은 번거로운 일련의 작업들을 자동으로 처리하고자 함이며, 앞으로 좀더 강화된 기능 추가를 통해 완벽한 데이터베이스 유지보수가 될 수 있도록 할 예정이다.



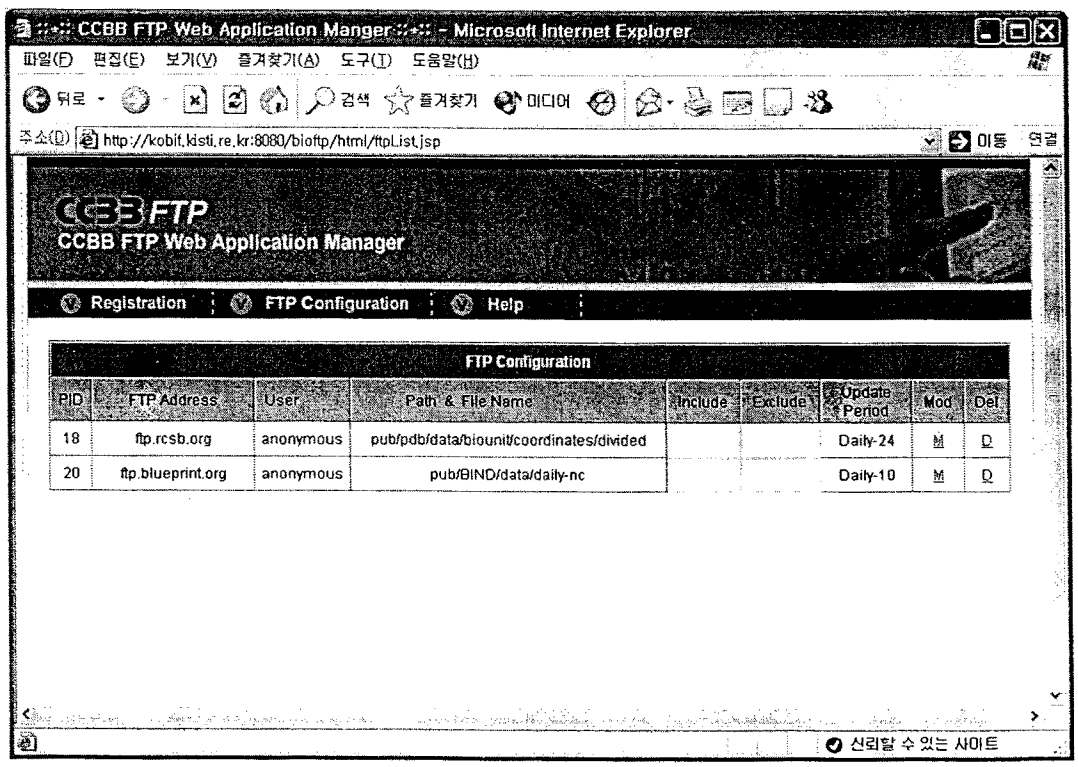
<figure 2-36> CCBB FTP 웹 기반 관리자 로그인



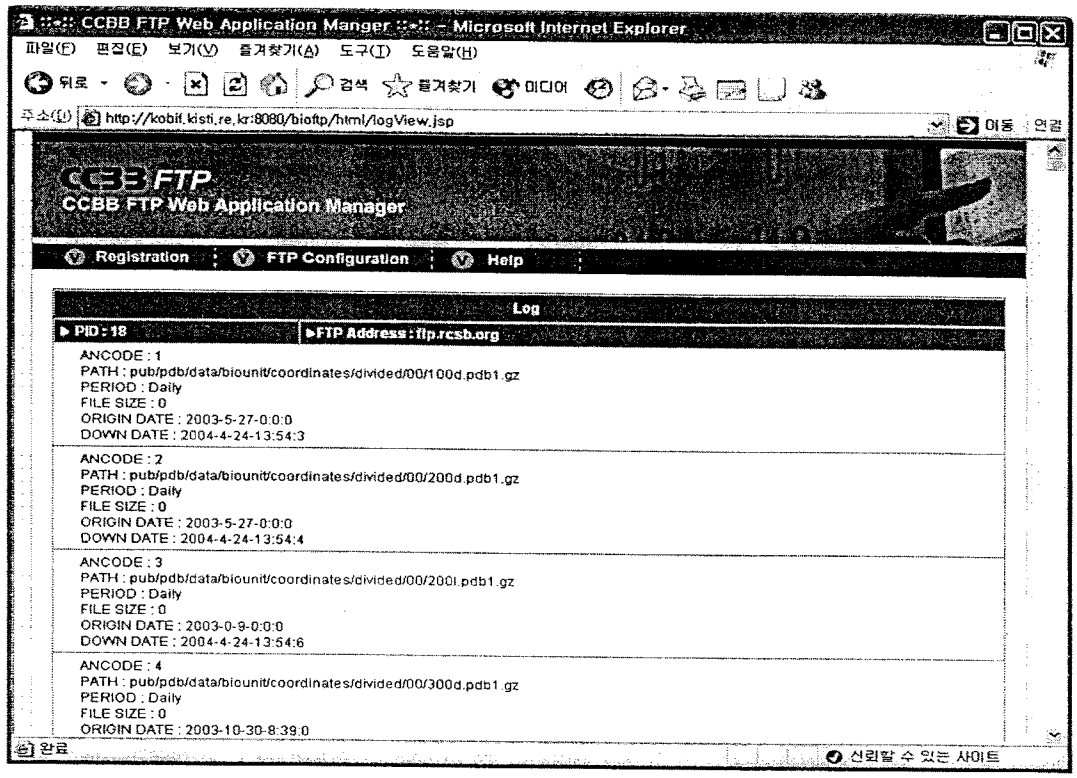
<figure 2-37> CCBB FTP 웹 기반 메인 화면



<figure 2-38> CCBB FTP 환경설정



<figure 2-39> Configuration 정보 리스트



<figure 2-40> 데이터 로그정보

3. 기존 데이터베이스의 GUI 및 기능 강화

가. GenBank

(1) GenBank 소개

생명체의 유전정보를 담고 있는 DNA는 4개의 염기로 구성된다. 이 4개의 염기의 배열을 해당 유전자의 염기서열이라고 하는데, 염기서열을 결정하는 기술은 1977년에 영국의 프레드릭 생거(Frederick Sanger) 박사에 의해 처음으로 개발되었다. 생거 박사의 이름을 딴 "생거 염기서열 분석법"이 확립된 이래, DNA 염기서열 결정 기술은 거듭 발전했으며, 이 기술을 이용하여 생명체의 신비를 담고 있는 유전자의 염기서열을 밝히고자 하는 노력이 전 세계적으로 확산되었다.

전 세계적인 염기서열 결정 작업에서 밝혀진 많은 DNA 염기서열을 체계적으로 정리하고 저장하여야만 생물학 연구자들이 정보를 공유해야하고 불필요한 중복 연구를 막아야 할 필요성이 생겼다. 이러한 요구에 부응하여 1981년에 미국국립보건원(NIH)의 지원

을 받아 로스 알라모스 국립연구소(Los Alamos National Laboratory)가 DNA 염기서열들을 모아 유전자은행(Gene Bank)이라는 뜻의 GenBank라 명명한 유전자정보 데이터베이스를 구축하기 시작했다. 이후 GenBank 구축 작업은 1992년에 미국국립보건원 국립의학도서관(NLM, National Library of Medicine) 산하의 미국국립생물공학정보센터(NCBI, National Center for Biotechnology Information)로 이관되어 현재까지 NCBI가 구축 및 관리를 담당하고 있다. NCBI는 전 세계의 과학자들이 제출하는 모든 염기서열에 대한 정보를 수정을 통해 보다 정확한 형태로 가공하여 데이터베이스를 구축하고 있으며, 모든 사람들이 이 정보를 공유할 수 있는 검색서비스를 제공하고 있다.

NCBI의 GenBank는 유럽 분자생물학 실험실인 EMBL(European Molecular Biology Laboratory)과 일본의 유전자 데이터베이스인 DDBJ(DNA DataBank of Japan)와 함께 '국제 염기서열 데이터베이스 협의체(International Nucleotide Sequence Collaboration)"를 구성하여 염기서열에 관한 각종 정보와 자료를 교환 및 공유하고 있다.

현재 GenBank에는 DNA 염기서열 정보뿐만 아니라, 염기서열로부터 해독한 단백질의 아미노산 서열 정보 및 RNA 염기서열도 실려 있다. NCBI에서는 약 2개월에 한번씩 Release라는 이름으로 GenBank를 배포하며, Release와 Release 사이에는 정기적으로 갱신된 정보를 배포하고 있다. 2004년 4월 15일 현재 GenBank Release 141이 배포되었으며, 이 Release는 3천4백만(33,676,218)개의 염기서열을 담고 있는데, 이는 대략 390억(38,989,342,565)개의 염기에 해당한다. GenBank의 공식적인 서비스는 NCBI의 홈페이지(http:// www.ncbi.nlm.nih.gov/)에서 볼 수 있다

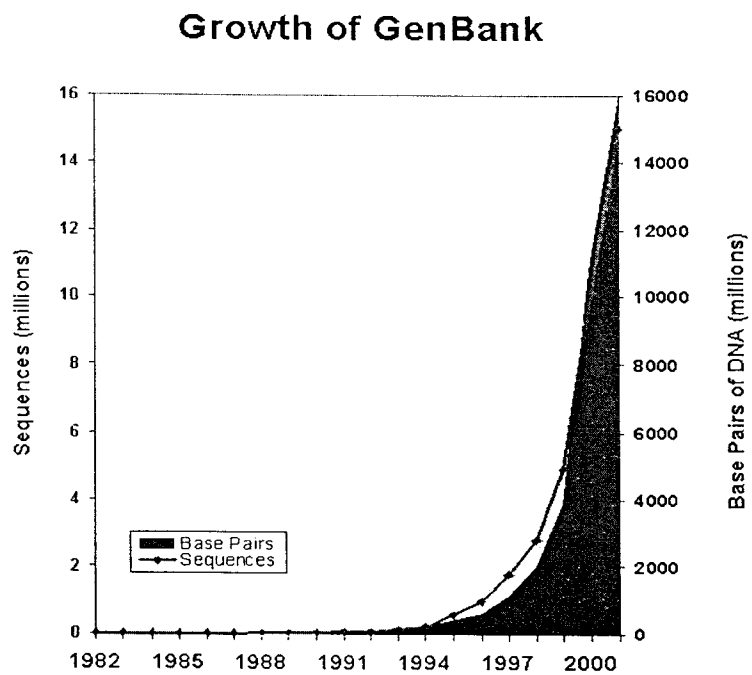
(2) GenBank 자료의 기하급수적인 증가

1985년도 GenBank에는 주로 생의학 잡지의 서열 자료를 스캐닝하여 확보한 5,700개가 약간 넘는 자료가 있었다. 1998년도 GenBank는 거의 2백만개의 레코드와 25,000 종류 이상의 생물체에 대한 10억개를 훨씬 초과하는 염기쌍에 대한 유전 정보가 포함하게 되었다. 2004년 4월 현재에는 3,360만개 이상의 염기서열과 약 390억개에 달하는 염기쌍에 대한 유전 정보를 포함하는 방대한 데이터베이스로 발전하였으며, 확장 추세는 기하급수적이라고 할 수 있다 (<figure 2-41> 참조).

<figure 2-41>에서 GenBank의 데이터베이스 확장 추세를 보면 대략 18개월마다 자료의 크기가 두 배로 증가하고 있음을 알 수 있다. 이러한 폭발적인 데이터의 증가는 주로

EST 서열에 기인하는 것인데, 최근에는 데이터베이스 서열 자료의 70% 이상이 EST이다. EST와 유전체 사업(Genome Project)의 강력한 추진으로 GenBank의 배가속도는 더욱 가속될 것으로도 예측된다.

이러한 GenBank의 기하급수적인 자료 증가속도는 조만간 일반적인 데이터베이스 시스템으로는 검색을 지원하기 어려워진다. 본 사업에서는 GenBank의 자료 구조를 분석하고, 새로운 검색서비스 시스템을 설계함으로써 향후 연구자들에게 보다 빠른 검색서비스를 제공하고자 한다.



<figure 2-41> GenBank의 기하급수적인 증가 추이

(3) GenBank의 자료구조

<table 2-16>은 미국국립보건원 산하의 국립생물공학정보연구소(NCBI)에서 배포하는 GenBank 원시자료의 하나이다. 그림에서 보는 바와 같이 원시자료는 유전자위(LOCUS), 정의(DEFINITION), 제어번호(ACCESSION), 키워드(KEYWORDS), 생물체(SOURCE), 참고자료(REFERENCE), 부연설명(COMMENTS), 특징(FEATURES) 등과 같이 해당 염기서열을 기술하기 위한 각각의 세부항목들로 이루어져 있으며, 마지막으로 A, G, C, T의 4개 염기로 구성된 염기서열이 수록되어 있다.

<table 2-16>의 예에서 알 수 있듯이, GenBank의 각 항목이 가지는 형식적인 구조는 다음과 같은 특징을 가진다.

ㅇ항목의 각 줄(line)은 1-10열에 기재되는 키워드 부분과 13열부터 80열까지 기록되는 내용(Content)으로 구성된다. 유전자 염기 서열이나 단백질의 아미노산 서열은 11열부터 75열까지 표시되는데, 각 줄당 10개 단위로 60개를 표시한다. GenBank에서 사용하는 키워드들은 표시 형식에 따른 종류별로 나열하면 다음과 같이 구분할 수 있다.

ㅇ키워드(Keyword) : 첫 번째 열부터 시작한다. 키워드는 모두 대문자로 표시한다. 예1: LOCUS, DEFINITION, ACCESSION, SOURCE, REFERENCE, FEATURES, BASE COUNT, ORIGIN 등

<table 2-16> NCBI에서 배포하는 GenBank의 원시자료 예제

```
LOCUS       AAURRA          118 bp ss-rRNA              RNA
16-JUN-1986
DEFINITION  A.auricula-judae (mushroom) 5S ribosomal RNA.
ACCESSION   K03160
KEYWORDS    5S ribosomal RNA; ribosomal RNA.
SOURCE      A.auricula-judae (mushroom) ribosomal RNA.
  ORGANISM  Auricularia auricula-judae
            Eukaryota; Fungi; Eumycota; Basidiomycotina; Phragmobasidiomycetes;
            Heterobasidiomycetidae; Auriculariales; Auriculariaceae.
REFERENCE   1  (bases 1 to 118)
  AUTHORS   Huysmans,E., Dams,E., Vandenberghe,A. and De Wachter,R.
  TITLE     The nucleotide sequences of the 5S rRNAs of four mushrooms and
            their use in studying the phylogenetic position of basidiomycetes
            among the eukaryotes
  JOURNAL   Nucleic Acids Res. 11, 2871-2880 (1983)
FEATURES             Location/Qualifiers
     rRNA            1..118
                     /note="5S ribosomal RNA"
BASE COUNT       27 a    34 c    34 g    23 t
ORIGIN      5' end of mature rRNA.
        1 atccacggcc ataggactct gaaagcactg catcccgtcc gatctgcaaa gttaaccaga
       61 gtaccgccca gttagtacca cggtggggga ccacgcggga atcctgggtg ctgtggtt
//
```

ㅇ부키워드(Subkeyword): 세번째 열부터 시작하며, 바로 위에 나타나는 키워드에 종속된 정보임을 표시한다. 키워드와 마찬가지로 부키워드는 모두 대문자로 표시한다. 예를 들어 ORGANISM은 SOURCE 키워드에 종속된 부키워드로 생물체의 학명 및 분류학적인 계통을 표시한다.

ㅇ공백: 공백문자로 시작하는 열은 앞줄의 내용과 연결됨을 의미한다.

○ 코드(Code): 6번째 열부터 시작하며, 상위 키워드의 하부 코드임을 표시한다. 소문자로 표시하나, rRNA, mRNA 등과 같은 경우에는 대소문자를 혼용한다. 예를 들어 rRNA, source 등은 FEATURES의 하부코드이다.

○ 숫자(Number): 숫자의 10단위 자리수에 따라 시작점이 다르나 9번째 열에서 모두 끝난다. 유전자 염기서열이나 단백질 아미노산 서열의 번호를 나타낸다.

○ 이중사선(//): 1,2열에 표시, 각 항목의 끝을 나타낸다.

다음은 각 키워드와 이에 해당하는 내용에 대해 자세히 설명한다. "필요"로 표시된 항목은 반드시 기입되어 있는 항목이며, "선택"으로 표시된 항목은 있을 수도 있고 없을 수도 있다.

○ LOCUS: [필요] 짧고 기억하기 좋은 해당항목의 고유한 명칭이다. LOCUS 이름은 영문자 대문자와 숫자로 구성된다. 첫 자는 영문자이고 10자를 초과할 수 없다. LOCUS 이름 뒤에는 서열의 크기(50 - 350000 bp), 서열이 확인된 분자의 형태(DNA, RNA등), 분류하기 위한 GenBank division code, 자료가 공개된 날짜가 표시된다. 각 항목의 위치 및 자료 형식은 아래와 같다.

<table 3-17> LOCUS 항목의 각 Column당 표기방법

Column	내 용
13-22	Locus name
23-29	Length of sequence, right-justified
31-32	bp
34-36	Blank, ss-(single-stranded), ds-(double-stranded), ms-(mixed-stranded)
37-40	Blank, DNA, RNA, tRNA(transfer RNA), rRNA(ribosomal RNA), mRNA (messenger RNA), or uRNA (small nuclear RNA)
43-52	Blank (implies linear) or circular
53-55	The division code (see Section 3.3)
63-73	Date, in the form dd-MMM-yyyy (e.g., 15-MAR-1991)

○ DEFINITION: [필요] 서열에 대한 간단한 설명이다. 속명, 종명, 생산물 또는 유전자 이름, 전체 또는 부분 서열 등을 명시한다.

○ ACCESSION: [필요] GenBank에 보관된 서열의 고유부호이며, NCBI에서 고유하게 부여한다. 자료를 인용할 때는 이 부호를 사용한다. 이차적인 고유번호가 동시에 표시되기도 한다.

○ VERSION: [필요] 해당 서열의 ACCESSION 번호와 수정 번호로 구성는 고유번호이

다. GI(geninfo) 번호라고도 하며 NCBI가 부여한다.

○NID: [선택] VERSION과 동일하나, 사용되지는 않는다. 하위버전과의 호환성 때문에 표기하는 항목이다.

○KEYWORDS: [필요] 서열에 대한 정보를 설명할 수 있는 단어나 구로 표현된다.

○SEGMENT: [선택] 같은 분자에서 불연속적으로 위치하는 서열들의 순서에 대한 정보이다.

○SOURCE: [필수] 서열이 추출된 생물의 일반이름을 표시한다. 학명 및 분류학적 데이터를 기록하는 ORGANISM의 부키워드(subkeyword)가 있다. ORGANISM에는 첫줄에 생물의 공식적인 학명과 두 번째 줄부터 분류단계별 이름 목록을 나열한다. ORGANISM 부키워드는 반드시 기입되어 있는 항목이다.

○REFERENCE: [필요] 서열 정보에 포함된 자료의 인용문헌을 표시하며, 복수로 기재 가능하다. AUTHORS, TITLE, JOURNAL, MEDLINE, REMARK 등의 부키워드(subkeyword)가 있다. 각 부키워드에 대한 설명은 다음과 같다.

<table 2-18> REFERENCE 항목의 부키워드 설명

AUTHORS	인용문헌의 저자들 이름
TITLE	인용문헌의 제목
JOURNAL	인용문헌이 게재된 학술지명, 권, 연도, 페이지
MEDLINE	인용문헌의 Medline 고유번호
REMARK	인용문헌에 대한 기타사항

○COMMENT: [선택] 다른 염기/아미노산 서열에 대한 상호참조 정보나 GenBank 이외의 다른 데이터베이스와의 참조 정보, LOCUS 이름의 변경 및 기타 설명할 내용 따위를 기록한다.

○FEATURES: [선택] 단백질 또는 RNA를 암호화하는 부분에 대한 정보나 실험으로 증명된 생물학적으로 중요한 부분에 대한 정보 등을 수록한다. 하부 코드(code)로 source, gene, CDS, rRNA 등과 같은 것들이 있다.

○BASE COUNT: [필요] 핵산서열을 구성하는 각 염기의 빈도수에 대한 요약 정보이다.

○ORIGIN: [필요] 다음 줄부터 표시되는 핵산서열의 첫 번째 염기의 genome내 위치에 대한 정보를 표시한다. Sequence의 시작을 알리는 keyword이므로 내용이 없더라도 반드시 필요하다.

○//: [필요] 항목의 끝을 표시하는 기호

(4) GenBank 데이터베이스 설계

NCBI의 GenBank는 2004년 4월 현재 3,400만 건에 이를 만큼 방대한 양을 자랑한다. 더불어 이 수는 기하급수적으로 증가하고 있는 형편이다. 따라서 올바른 데이터베이스 구조를 선택하지 않으면, 사소한 잘못으로도 검색시스템의 성능에 큰 지장을 초래할 수 있다. 본 사업에서는 이러한 점을 고려하여 다음과 같은 방향으로 GenBank 검색시스템을 설계하였다.

- 원시 자료 : 원시자료는 NCBI에서 공개적으로 배포하는 GenBank 데이터베이스를 사용하였다. 2004년 4월에 배포된 GenBank Release 141은 559개의 파일로 배포되고 있다. 본 사업에서는 이 파일들을 대상으로 하여 GenBank 검색서비스를 구성하였다. 최대 100만개씩의 서열을 하나의 볼륨으로 묶어서 KRISTAL-2002 정보검색시스템의 볼륨(Volume)으로 구성하였으며, 그 결과 GenBank 검색 서비스에서는 45개의 볼륨이 사용되고 있다.

- 색인 범위 : 그림 1.1에서 볼 수 있듯이, 각 항목을 KRISTAL-2002 시스템의 섹션으로 정의함으로써 해당 필드(Field)만으로도 검색이 가능하도록 하였다. 반면, 사용자의 편의를 위해 모든 필드를 동시에 색인할 수 있는 색인섹션을 별도로 구성하여 검색 편의성과 검색 성능을 높일 수 있도록 하였다. 색인섹션은 유전자위(LOCUS), 정의(DEFINITION), 제어번호(ACCESSION), 키워드(KEYWORDS), 생물체(SOURCE), 참고자료(REFERENCE) 등을 포함한다. 부연설명(COMMENTS), 특징(FEATURES) 등은 색인섹션에서 제외하였다. 이것은 많은 데이터를 분석한 결과 이 두 부분은 색인섹션에 포함된 다른 부분들과 내용이 대부분 중복되었기 때문이다.

- 색인 방법 : 색인 방식은 KRISTAL-2002 정보검색시스템의 INDEX_BY_TOKEN 방식을 사용하였으며, 영어 어간 추출(Stemming)이나 불용어(Stopwords) 제거 처리는 하지 않았다. 이렇게 처리한 이유는 유전좌, 유전자 이름 등에 불용어에 해당할 수 있는 많은 단어들이 포함되어 있으므로 정확한 검색을 할 수 없기 때문이다. KRISTAL-2002 시스템이 제공하는 INDEX_BY_TOKEN 색인 방식을 이용하면, 어절 단위로 키워드를 추출하여 색인으로 사용한다. 이 경우 어간추출을 이용하면 재현율은 높아지나, 정확도가 떨어지는 단점이 있어서 본 시스템에서는 어간추출을 하지 않았다. 또한 GenBank에 등록된 염기서열 정보의 경우 다양한 약어들을 사용한다. 대부분 이러한 약어들은 기능어(Functional Word)라고도 하는 불용어들과 중복되는 경우가 많았다. 따라서 불용어를 색인에서 제거할 경우 검색의 속도는 향상되나 정확도를 크게 떨어뜨릴 위험이 있으므로 불용어를 제거하지 않고 시

스템을 구축하였다.

이러한 점들을 반영하여 만들어진 KRISTAL-2002 스키마(Schema) 파일은 그림1.2와 같다. KRISTAL-2002 시스템은 데이터베이스의 대부분의 설정을 스키마 파일에 정의할 수 있도록 하고 있다. GenBank 원시자료 분석에 따라 본 시스템에서는 원시자료를 GenBank 제어번호(ACCESSION), 편집 버전(VERSION), 분자정보(MOLECULE), 출처 기관(ORGANELLE), 최종수정일(DATE), 색인(INDEX), 원본문서(TEXT) 등으로 분할하였다. 대부분의 섹션은 섹션값 자체를 색인으로 삼았으나 보다 상세한 검색을 위해서 색인 섹션(INDEX)을 INDEX_BY_TOKEN 색인방식으로 색인하였다. 각 섹션의 색인 방식은 <table 2-19>에서 볼 수 있다.

<table 2-19> 각 섹션별 색인 정보

섹션	색인 방식	색인 방법	비고
ACCESSION	DO_NOT_INDEX	색인하지 않음	PRIMARY KEY로 접근
DEFINITION	INDEX_BY_TOKEN[2]	어절별 색인	염기서열정보
VERSION	INDEX_AS_IS[3]	섹션별 색인	편집버전정보
MOLECULE	INDEX_AS_IS	섹션별 색인	분자형태
ORGANELLE	INDEX_AS_IS	섹션별 색인	출처 기관
DATE	INDEX_AS_IS	섹션별 색인	등록일
INDEX	INDEX_BY_TOKEN	어절별 색인	주 검색 섹션

```
<?xml version="1.0"?>

<!-- ######################################################## -->
<!-- Descriptions for KRISTAL-2002 Database Schema -->
<!-- ######################################################## -->
<!-- data-type: KINT, KFLOAT, KBOOL, KDATE, KSTRING, KBLOB, KCHAR -->
<!-- index-type: INDEX_AS_IS, INDEX_BY_TOKEN, INDEX_AS_NUMERIC,  -->
<!--                INDEX_BY_MA, DO_NOT_INDEX  -->
<!-- All value is case-insensitive.  -->
```

2) INDEX_BY_TOKEN 색인방식은 TOKEN 즉 어절별로 색인어를 추출하는 방식이다. 예를 들어 "Bush attacked Iraq"라는 내용을 가지는 문서에서는 'Bush', 'attacked', 'Iraq'라는 세 단어가 색인어로 추출된다.

3) INDEX_AS_IS 색인방식은 섹션값 자체를 하나의 색인어로 취급하는 방식으로, 주로 제어번호, 날짜 등과 같이 섹션값 자체가 색인어로서 유효한 값을 가질 때 사용된다. 예를 들어 "Bush attacked Iraq"라는 내용을 가지는 문서에서는 'Bush attacked Iraq'라는 문자열 자체가 하나의 색인어로 추출된다.

```
<!-- ########################################################## -->

<!-- Start of KRISTAL-2002 Database Schema file -->

<DatabaseSchema>
    <CreateDatabase database-name="GENBANK"
        volume-dir="/disk2/genbank/gb_volumes" />
    <!-- Table Schema Definitions -->
    <CreateTableSchema name="schema01">
        <!-- Primary Key Definition -->
        <!-- PrimaryKey sections="ACCESSION"/ -->

        <!-- Stopword File Definition -->
        <Stopword file="/disk4/genbank/dict/swords-eng"/>

        <!-- Basic Section Definitions -->
        <BasicSection  name="ACC" data-type="KSTRING"
            index-type="DO_NOT_INDEX"/>
        <BasicSection  name="LOC" data-type="KSTRING"
            index-type="DO_NOT_INDEX" />
        <BasicSection  name="DEF" data-type="KSTRING"
            index-type="DO_NOT_INDEX" />
        <BasicSection  name="VER" data-type="KSTRING"
            index-type="INDEX_BY_TOKEN" />
        <BasicSection  name="KEY" data-type="KSTRING"
            index-type="DO_NOT_INDEX" />
        <BasicSection  name="MOL" data-type="KSTRING"
            index-type="DO_NOT_INDEX" />
        <BasicSection  name="ORG" data-type="KSTRING"
            index-type="DO_NOT_INDEX" />
        <BasicSection  name="REF" data-type="KSTRING"
            index-type="INDEX_BY_TOKEN" remove-stopword="yes" />
        <BasicSection  name="AUT" data-type="KSTRING"
            index-type="INDEX_BY_TOKEN" remove-stopword="yes" />
        <BasicSection  name="DAT" data-type="KCHAR[8]"
            index-type="DO_NOT_INDEX" />
        <BasicSection  name="IDX" data-type="KSTRING"
            index-type="INDEX_BY_TOKEN" />
        <!-- Union Section Definitions -->
        <UnionSection name="BASIC"
```

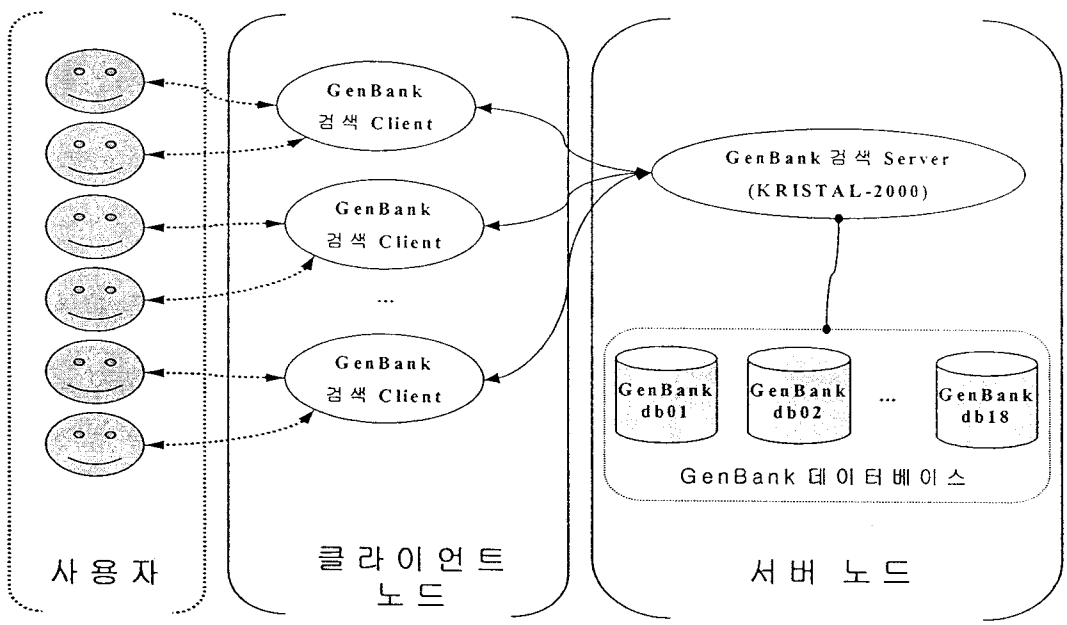
```
                include-sections="IDX REF AUT" />
        <!--            include-sections="IDX REF AUT" / -->

    </CreateTableSchema>


<CreateTable table-name="bct001" with-schema="schema01"/>
<CreateTable table-name="con001" with-schema="schema01"/>
<CreateTable table-name="htc001" with-schema="schema01"/>
<CreateTable table-name="htg001" with-schema="schema01"/>
<CreateTable table-name="inv001" with-schema="schema01"/>
<CreateTable table-name="mam001" with-schema="schema01"/>
<CreateTable table-name="pat001" with-schema="schema01"/>
<CreateTable table-name="pat002" with-schema="schema01"/>
<CreateTable table-name="pat003" with-schema="schema01"/>
<CreateTable table-name="phg001" with-schema="schema01"/>
<CreateTable table-name="pln001" with-schema="schema01"/>
<CreateTable table-name="pri001" with-schema="schema01"/>
<CreateTable table-name="rod001" with-schema="schema01"/>
<CreateTable table-name="sts001" with-schema="schema01"/>
<CreateTable table-name="syn001" with-schema="schema01"/>
<CreateTable table-name="una001" with-schema="schema01"/>
<CreateTable table-name="vrl001" with-schema="schema01"/>
<CreateTable table-name="vrt001" with-schema="schema01"/>
<CreateTable table-name="est001" with-schema="schema01"/>
<CreateTable table-name="est002" with-schema="schema01"/>
<CreateTable table-name="est003" with-schema="schema01"/>
<CreateTable table-name="est004" with-schema="schema01"/>
<CreateTable table-name="est005" with-schema="schema01"/>
<CreateTable table-name="est006" with-schema="schema01"/>
<CreateTable table-name="est007" with-schema="schema01"/>
<CreateTable table-name="est008" with-schema="schema01"/>
<CreateTable table-name="est009" with-schema="schema01"/>
<CreateTable table-name="est010" with-schema="schema01"/>
<CreateTable table-name="est011" with-schema="schema01"/>
<CreateTable table-name="est012" with-schema="schema01"/>
<CreateTable table-name="est013" with-schema="schema01"/>
<CreateTable table-name="est014" with-schema="schema01"/>
<CreateTable table-name="est015" with-schema="schema01"/>
<CreateTable table-name="est016" with-schema="schema01"/>
```

```
<CreateTable table-name="est017" with-schema="schema01"/>
<CreateTable table-name="est018" with-schema="schema01"/>
<CreateTable table-name="est019" with-schema="schema01"/>
<CreateTable table-name="est020" with-schema="schema01"/>
<CreateTable table-name="gss001" with-schema="schema01"/>
<CreateTable table-name="gss002" with-schema="schema01"/>
<CreateTable table-name="gss003" with-schema="schema01"/>
<CreateTable table-name="gss004" with-schema="schema01"/>
<CreateTable table-name="gss005" with-schema="schema01"/>
<CreateTable table-name="gss006" with-schema="schema01"/>
<CreateTable table-name="gss007" with-schema="schema01"/>

</DatabaseSchema>

<!-- End of KRISTAL-2002 Database Schema file -->


<?xml version="1.0" ?>

<LoaderSchema database-name="GENBANK"
    volume-dir="/disk2/genbank/gb_volumes"
    kristal-root="/home/k2002/K2002">

    <DocStructure name="struct01" border-string="@GENBANK">
        <Tag name="#ACC=" mapping-section="ACC" />
        <Tag name="#LOC=" mapping-section="LOC" />
        <Tag name="#DEF=" mapping-section="DEF" />
        <Tag name="#VER=" mapping-section="VER" />
        <Tag name="#KEY=" mapping-section="KEY" />
        <Tag name="#MOL=" mapping-section="MOL" />
        <Tag name="#ORG=" mapping-section="ORG" />
        <Tag name="#REF=" mapping-section="REF" />
        <Tag name="#AUT=" mapping-section="AUT" />
        <Tag name="#DAT=" mapping-section="DAT" />
        <Tag name="#IDX=" mapping-section="IDX" />
    </DocStructure>


<LoaderMap table="bct001" doc-structure="struct01"
file="/disk4/genbank/data/bct001*.seq" encoding="UTF-8" />
```

<LoaderMap table="con001" doc-structure="struct01"
file="/disk4/genbank/data/con001*.seq" encoding="UTF-8" />
<LoaderMap table="htc001" doc-structure="struct01"
file="/disk4/genbank/data/htc001*.seq" encoding="UTF-8" />
<LoaderMap table="htg001" doc-structure="struct01"
file="/disk4/genbank/data/htg001*.seq" encoding="UTF-8" />
<LoaderMap table="inv001" doc-structure="struct01"
file="/disk4/genbank/data/inv001*.seq" encoding="UTF-8" />
<LoaderMap table="mam001" doc-structure="struct01"
file="/disk4/genbank/data/mam001*.seq" encoding="UTF-8" />
<LoaderMap table="pat001" doc-structure="struct01"
file="/disk4/genbank/data/pat001*.seq" encoding="UTF-8" />
<LoaderMap table="pat002" doc-structure="struct01"
file="/disk4/genbank/data/pat002*.seq" encoding="UTF-8" />
<LoaderMap table="pat003" doc-structure="struct01"
file="/disk4/genbank/data/pat003*.seq" encoding="UTF-8" />
<LoaderMap table="phg001" doc-structure="struct01"
file="/disk4/genbank/data/phg001*.seq" encoding="UTF-8" />
<LoaderMap table="pln001" doc-structure="struct01"
file="/disk4/genbank/data/pln001*.seq" encoding="UTF-8" />
<LoaderMap table="pri001" doc-structure="struct01"
file="/disk4/genbank/data/pri001*.seq" encoding="UTF-8" />
<LoaderMap table="rod001" doc-structure="struct01"
file="/disk4/genbank/data/rod001*.seq" encoding="UTF-8" />
<LoaderMap table="sts001" doc-structure="struct01"
file="/disk4/genbank/data/sts001*.seq" encoding="UTF-8" />
<LoaderMap table="syn001" doc-structure="struct01"
file="/disk4/genbank/data/syn001*.seq" encoding="UTF-8" />
<LoaderMap table="una001" doc-structure="struct01"
file="/disk4/genbank/data/una001*.seq" encoding="UTF-8" />
<LoaderMap table="vrl001" doc-structure="struct01"
file="/disk4/genbank/data/vrl001*.seq" encoding="UTF-8" />
<LoaderMap table="vrt001" doc-structure="struct01"
file="/disk4/genbank/data/vrt001*.seq" encoding="UTF-8" />
<LoaderMap table="est001" doc-structure="struct01"
file="/disk4/genbank/data/est001*.seq" encoding="UTF-8" />
<LoaderMap table="est002" doc-structure="struct01"
file="/disk4/genbank/data/est002*.seq" encoding="UTF-8" />
<LoaderMap table="est002" doc-structure="struct01"
file="/disk4/genbank/data/est002*.seq" encoding="UTF-8" />

```
<LoaderMap table="est003" doc-structure="struct01"
file="/disk4/genbank/data/est003*.seq" encoding="UTF-8" />
<LoaderMap table="est004" doc-structure="struct01"
file="/disk4/genbank/data/est004*.seq" encoding="UTF-8" />
<LoaderMap table="est005" doc-structure="struct01"
file="/disk4/genbank/data/est005*.seq" encoding="UTF-8" />
<LoaderMap table="est006" doc-structure="struct01"
file="/disk4/genbank/data/est006*.seq" encoding="UTF-8" />
<LoaderMap table="est007" doc-structure="struct01"
file="/disk4/genbank/data/est007*.seq" encoding="UTF-8" />
<LoaderMap table="est008" doc-structure="struct01"
file="/disk4/genbank/data/est008*.seq" encoding="UTF-8" />
<LoaderMap table="est009" doc-structure="struct01"
file="/disk4/genbank/data/est009*.seq" encoding="UTF-8" />
<LoaderMap table="est010" doc-structure="struct01"
file="/disk4/genbank/data/est010*.seq" encoding="UTF-8" />
<LoaderMap table="est011" doc-structure="struct01"
file="/disk4/genbank/data/est011*.seq" encoding="UTF-8" />
<LoaderMap table="est012" doc-structure="struct01"
file="/disk4/genbank/data/est012*.seq" encoding="UTF-8" />
<LoaderMap table="est013" doc-structure="struct01"
file="/disk4/genbank/data/est013*.seq" encoding="UTF-8" />
<LoaderMap table="est014" doc-structure="struct01"
file="/disk4/genbank/data/est014*.seq" encoding="UTF-8" />
<LoaderMap table="est015" doc-structure="struct01"
file="/disk4/genbank/data/est015*.seq" encoding="UTF-8" />
<LoaderMap table="est016" doc-structure="struct01"
file="/disk4/genbank/data/est016*.seq" encoding="UTF-8" />
<LoaderMap table="est017" doc-structure="struct01"
file="/disk4/genbank/data/est017*.seq" encoding="UTF-8" />
<LoaderMap table="est018" doc-structure="struct01"
file="/disk4/genbank/data/est018*.seq" encoding="UTF-8" />
<LoaderMap table="est019" doc-structure="struct01"
file="/disk4/genbank/data/est019*.seq" encoding="UTF-8" />
<LoaderMap table="est020" doc-structure="struct01"
file="/disk4/genbank/data/est020*.seq" encoding="UTF-8" />
<LoaderMap table="est020" doc-structure="struct01"
file="/disk4/genbank/data/est020*.seq" encoding="UTF-8" />
<LoaderMap table="gss001" doc-structure="struct01"
file="/disk4/genbank/data/gss001*.seq" encoding="UTF-8" />
```

<LoaderMap table="gss002" doc-structure="struct01"
file="/disk4/genbank/data/gss002*.seq" encoding="UTF-8" />
<LoaderMap table="gss003" doc-structure="struct01"
file="/disk4/genbank/data/gss003*.seq" encoding="UTF-8" />
<LoaderMap table="gss004" doc-structure="struct01"
file="/disk4/genbank/data/gss004*.seq" encoding="UTF-8" />
<LoaderMap table="gss005" doc-structure="struct01"
file="/disk4/genbank/data/gss005*.seq" encoding="UTF-8" />
<LoaderMap table="gss006" doc-structure="struct01"
file="/disk4/genbank/data/gss006*.seq" encoding="UTF-8" />
<LoaderMap table="gss007" doc-structure="struct01"
file="/disk4/genbank/data/gss007*.seq" encoding="UTF-8" />
</LoaderSchema>



<figure 2-42> GenBank 데이터베이스 서비스를 위한 KRISTAL-2002
스키마1.3.3 GenBank 검색 서비스를 위한 서버/클라이언트

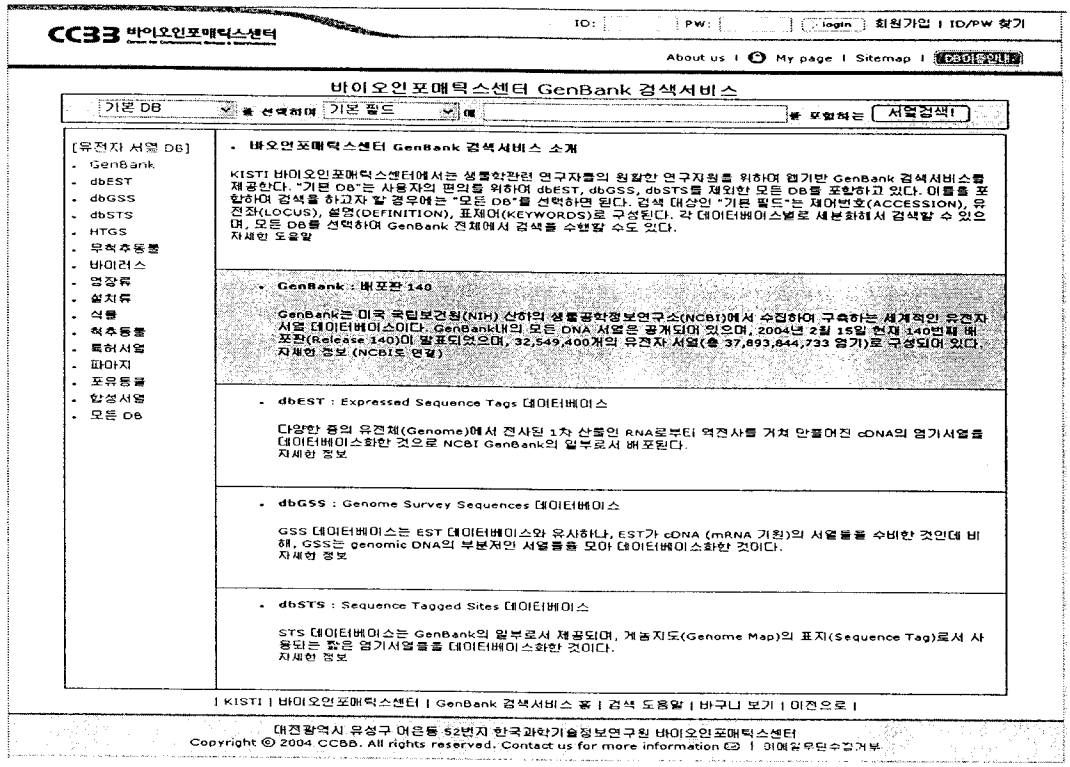
<figure 2-42>는 GenBank 데이터베이스의 서비스 구조도를 보여주고 있다. 먼저 사용자들은 웹을 통해 GenBank 클라이언트 노드에 접속하여 질의를 보낸다. 질의를 받은 클라이언트는 소켓(Socket) 통신을 통해 서버노드로 검색요청을 보낸다. 검색서버는 사용자의 질의에 적합한 염기서열 정보를 검색하여 그 결과를 소켓 통신을 통해 다시 클라이언트로 되돌려 보내고, 클라이언트는 이 결과를 정리하여 웹을 통해 사용자에게 제공한다.

GenBank 검색서비스에서는 검색서버와 사용자를 위한 클라이언트를 하드웨어적으로 분리하는 Clinet/Server 구조를 도입하여 보안성 및 안정성을 높였다. 즉 클라이언트 노드는 검색을 위한 client만을 가지고 있으며, client는 CGI(Common Gateway Interface) 방식으로 작성되었다. 클라이언트는 CGI 변수처리 모듈, KRISTAL-2002 서버와의 통신 모듈 및 웹 인터페이스 모듈 등의 비교적 간단한 모듈로만 구성되어 있어 단일 클라이언트는 클라이언트 노드에 부담을 주지 않는다. 클라이언트는 CGI 방식으로 작동하므로 하드웨어가 지원하는 범위 내에서 무제한적인 접속이 가능하다. 현 GenBank 검색 클라이언트 노드는 2개의 CPU와 1GB의 메모리를 가진다. 반면 서버는 KRISTAL-2002 검색 서버가 GenBank 데이터베이스 검색을 담당한다. 검색 서버는 2개의 CPU와 2GB의 메모리를 가진다.

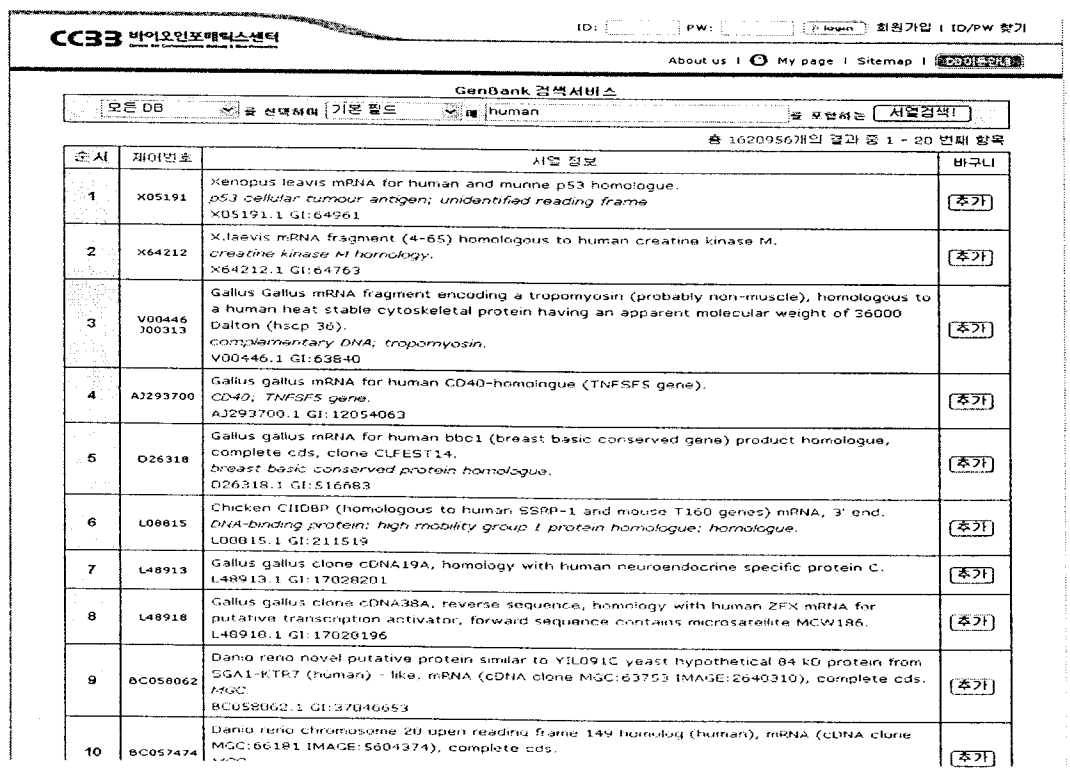
앞서 기술한 바와 같이, GenBank 데이터베이스 검색 서비스 시스템에서는 검색 서버로서 KRISTAL-2002 정보검색시스템을 채택하였으며, 클라이언트는 GenBank 검색을 위해 최적화하는 방식으로 새로 구성되었다. 검색 클라이언트의 설계 주안점은 클라이언트 소스(Source)와 웹 인터페이스를 분리하는 방식을 채택하여 프로그램의 수정을 최소화하고도 웹 인터페이스를 수정할 수 있도록 하였다. 클라이언트는 CGI 방식으로 구현되었으며, C++ 표준 라이브러리, CGICC 모듈, 및 KRISTAL-2002 Client 라이브러리로 구성되어 있다. 또한 검색의 성능을 높이기 위해, 검색 모델을 최적화하는 부분은 검색 클라이언트에서 수행하도록 설계하였다. 이렇게 함으로써 KRISTAL-2002이라는 범용 정보검색관리시스템을 수정하지 않고도 GenBank 검색을 최적화할 수 있다. GenBank 검색시스템에서 채택한 검색모델은 불리안(Boolean)검색모델이며, GenBank 검색 클라이언트에서는 이 검색모델을 보다 최적화하여 불리안 연산자들 중 AND 및 WITHIN 연산자를 사용자의 질의에 따라 최적화된 형식으로 재구성하여 KRISTAL-2002 검색서버에 전달할 수 있도록 하였다.

(5) GenBank 웹서비스 인터페이스

<figure 2-43>에서는 GenBank 검색서비스의 주화면을 보여주고 있다. 검색 서비스 개발의 주안점은 사용자 편의성에 주안점을 두었다. 즉, 검색 화면을 단순화함으로써 사용자가 검색에 관한 전문적인 지식이 없이도 손쉽게 검색을 할 수 있도록 하였다. 위 그림에서 볼 수 있듯이 GenBank 데이터베이스를 검색하고자 하는 사용자는 자신이 찾고자하는 염기서열에 대한 키워드(Keyword) 만을 나열함으로써 원하는 검색 결과를 얻을 수 있다. 참고로 NCBI의 GenBank 검색 서비스에서도 기본적인 검색 페이지는 이와 유사한 방식을 취하고 있어서, 기존 사용자들도 쉽게 사용할 수 있다는 장점이 있다.



<figure 2-43> GenBank 검색서비스 기본 인터페이스

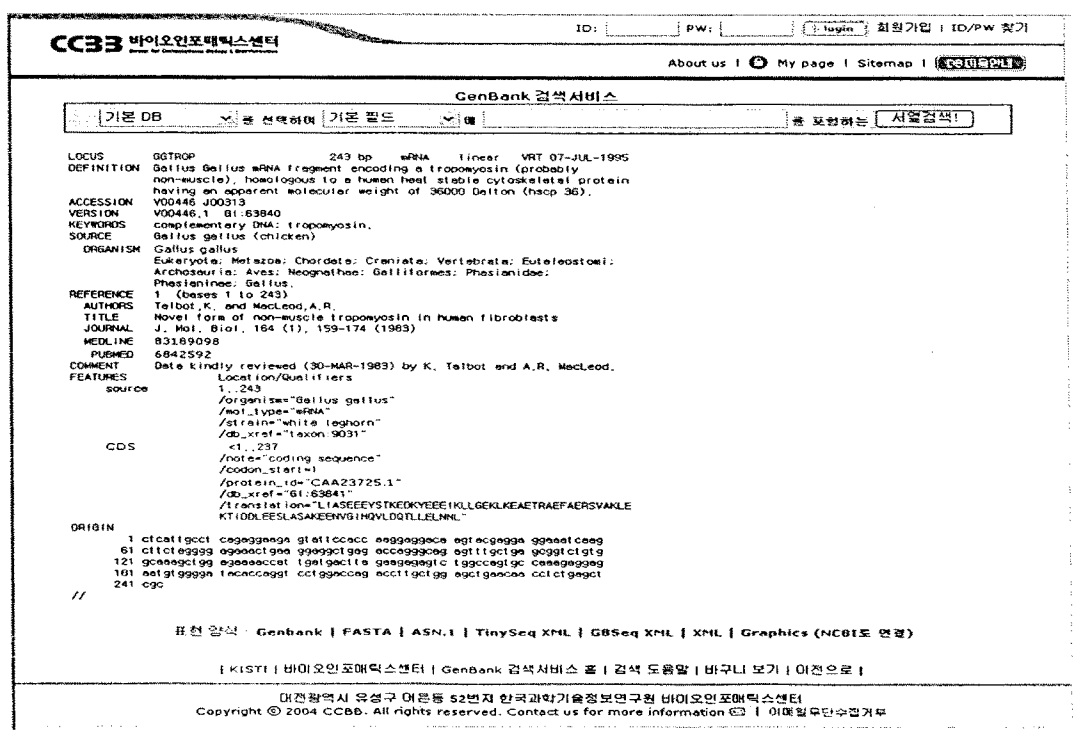


순서	제어번호	서열 정보	바구니
1	X05191	Xenopus laevis mRNA for human and murine p53 homologue. *p53 cellular tumour antigen; unidentified reading frame* X05191.1 GI:64961	추가
2	X64212	X.laevis mRNA fragment (4-65) homologous to human creatine kinase M. *creatine kinase M homology.* X64212.1 GI:64763	추가
3	V00446 J00313	Gallus Gallus mRNA fragment encoding a tropomyosin (probably non-muscle), homologous to a human heat stable cytoskeletal protein having an apparent molecular weight of 36000 Dalton (hscp 36). *complementary DNA; tropomyosin.* V00446.1 GI:63840	추가
4	AJ293700	Gallus gallus mRNA for human CD40-homologue (TNFSF5 gene). *CD40; TNFSF5 gene.* AJ293700.1 GI:12054063	추가
5	D26318	Gallus gallus mRNA for human bbc1 (breast basic conserved gene) product homologue, complete cds, clone CLFEST14. *breast basic conserved protein homologue.* D26318.1 GI:516683	추가
6	L00815	Chicken CIIDBP (homologous to human SSRP-1 and mouse T160 genes) mRNA, 3' end. *DNA-binding protein; high mobility group 1 protein homologue; homologue.* L00815.1 GI:211519	추가
7	L48913	Gallus gallus clone cDNA19A, homology with human neuroendocrine specific protein C. L48913.1 GI:17028201	추가
8	L48918	Gallus gallus clone cDNA38A, reverse sequence, homology with human ZFX mRNA for putative transcription activator, forward sequence contains microsatellite MCW186. L48918.1 GI:17028196	추가
9	BC058062	Danio rerio novel putative protein similar to YIL091C yeast hypothetical 84 kD protein from SGA1-KTR7 (human) - like, mRNA (cDNA clone MGC:63753 IMAGE:2640310), complete cds. *MGC* BC058062.1 GI:37046653	추가
10	BC057474	Danio rerio chromosome 20 open reading frame 149 homolog (human), mRNA (cDNA clone MGC:66181 IMAGE:5604374), complete cds.	추가

<figure 2-44> GenBank 검색 간략보기 인터페이스

<figure 2-44>는 GenBank 데이터베이스 검색결과에 대한 간략한 정보를 출력하는 화면이다. 결과화면에서도 사용자의 질의를 그대로 유지함으로써 재검색 등과 같이 사용자의 편이성을 유지하려고 노력하였다. 결과에는 총 검색결과의 수와 현재 보여주고 있는 페이지에 표현된 결과의 위치, 및 결과 페이지와 페이지 사이를 이동할 수 있는 링크들로 구성하였다. 그리고 클립보드 보기 및 클립보드 추가 단추들을 제공한다. 그리고 마지막으로 염기서열에 대한 정보들을 출력한다.

염기서열에 대한 정보를 표시하는 첫 번째 항목은 검색된 순서, 두 번째 항목은 GenBank의 제어번호(ACCESSION code), 세 번째 항목에서는 유전자에 대한 설명(DESCRIPTION) 및 버전 정보를 출력하도록 하여 사용자의 요구에 맞는 결과인지 알기 쉽게 판단할 수 있도록 하였다. 이와 더불어 마지막 항목으로는 클립보드 추가 단추를 두어서 사용자가 따로 원하는 서열들을 저장할 수 있도록 하였다 - 이미 클립보드에 추가되어 있는 항목에 대해서는 클립보드에서 제거할 수 있는 단추가 제공된다. 이와 더불어 검색결과의 상단에 보여주는 클립보드 추가 기능은 간략 보기에서는 현재 페이지의 모든 항목들을 클립보드에 추가할 수 있도록 하였다. 이렇게 함으로써 사용자는 모든 항목에 대해 일일이 클립보드 추가를 할 필요는 없으며, 한 번의 클릭으로 모든 항목이 클립보드에 추가되도록 하였다.



<figure 2-45> GenBank 데이터베이스 검색결과 상세보기 화면

<figure 2-45>는 GenBank 데이터베이스 검색결과에 대한 상세한 정보를 출력하는 화면이다. 상세보기 화면에서도 사용자의 질의를 유지하여 향후 검색에 편의를 도모하였다. 결과 화면의 상단에는 이 염기서열의 버전 정보 및 클립보드 추가 및 클립보드 보기 기능 단추를 제공하였다. 이어서 해당 항목에 대한 자세한 정보를 출력한다.

상세보기에서는 사용자가 보고자 하는 염기서열에 대한 정보를 원본에 충실하게 제공한다. 즉 상세보기 항목의 정보는 미국국립보건원 산하의 NCBI에서 배포하는 원시자료 형태와 동일하게 출력하도록 하였다.

<figure 2-46> GenBank 데이터베이스 클립보드 간략보기

<figure 2-46>은 GenBank 데이터베이스에서 사용자가 추가한 클립보드(Clipboard) 항목들을 간략하게 볼 수 있는 화면이다. 클립보드(Clipboard)란 사용자의 장바구니와 같은 기능으로 원하는 항목을 검색결과로부터 별도로 저장할 수 있는 공간이다. 본 검색 시스템에서는 쿠키(cookie)를 이용하여 사용자의 웹 브라우저에 염기서열의 ID를 저장하도록 하였다. 클립보드에는 최대 500개까지 항목을 저장할 수 있으며, 사용자가 현재의 웹 브라우저(Web Browser)를 닫기 전까지는 클립보드가 유지되도록 하였다. 즉 사용자가 웹 브라

우저를 종료시키면, GenBank의 클립보드도 자동적으로 비워지도록 하였다.

클립보드 간략보기에서는 기본적으로 FASTA 포맷으로 보기 등과 같은 기능을 제공한다. 각 항목에 대한 설명에서는 각 항목별 클립보드 제거 기능 단추가 제공된다.

내려받기란 클립보드내의 모든 항목을 사용자의 로칼 디스크로 내려받아 저장하는 기능이다. 내려받기에서 사용자가 받는 파일은 FASTA 형식으로 된 텍스트 파일이다. 화면에서 보기(View) 기능은 내려받기 기능과 동일한 인터페이스를 가지지만, 내려받기에서는 사용자의 디스크로 저장하는 반면 View에서는 웹 브라우저에 그 결과를 출력한다. View의 출력 형식 역시 내려받기에서와 마찬가지로 FASTA 형식이다. FASTA 형식에 대해서는 다음 페이지에서 보다 자세히 설명한다.

간략보기에서 출력하는 정보는 검색결과 간략보기에서의 출력정보와 동일하다. 다만 여기서 보여주는 모든 항목은 클립보드에 존재하는 것이므로 마지막으로 제공하는 모든 단추는 클립보드로부터 각 항목을 제거할 수 있는 단추가 제공된다. 클립보드 상세보기는 검색시의 상세보기와 동일한 인터페이스로 구성되어 있다. 다만 클립보드 상세보기에서는 사용자의 질의가 어떠한 것인지 판단하기 어려우므로 질의어 돋보임(highlighting) 기능은 제공되지 않는다.

<figure 2-47> Genbank데이터베이스 클립보드 FASTA 포맷으로 보기

<figure 2-47>은 GenBank 데이터베이스로부터 사용자가 클립보드에 저장한 항목들을 웹브라우저 상에서 보기(View) 기능을 눌렀을 때의 출력화면이다. 출력 양식은 1개 이상의 염기서열을 FASTA 포맷으로 변환하여 사용자에게 제공한다. 내려받기(Download)는 보기와 동일하나 그림에서 보여주는 바와 같은 결과를 사용자의 디스크에 파일로 저장하도록 하였다. FASTA 포맷으로 출력할 때는 GenBank 버전 정보, 염기서열에 관한 기술 및 염기서열을 제공한다.

FASTA 형식의 파일은 생물정보 분석 프로그램들의 입력 양식 중 간단하면서도 가장 일반적으로 사용되는 것으로 FASTP/FASTN에서 사용되다가 FASTA에서도 사용되고 있는 파일 양식이다. FASTA 포맷 파일은 한 줄의 설명 라인(description line)과 서열 데이터(sequence data)로 구성된다. 설명라인은 서열 데이터(sequence data)와 달리 첫 번째 열에 있는 ">" 문자로 시작한다. 모든 줄은 대개 줄당 80행 이하로 작성되는데, 본 시스템에서는 70행으로 출력한다. 다음은 간단한 FASTA 포맷 파일의 예이다. 이 예에서 FASTA 파일이 나타내는 것은 GenBank 제어번호가 AA041505이며 Homo sapiens의 cDNA clone에서 추출한 염기서열이라는 것을 나타낸다.(예제2 참조)

예제2:

>|GB:AA041505.1 GI:1517739| Homo sapiens cDNA clone
agggaaagcatcaggaggaatagctannngntgctgggcttaatacctaggcaatggattgatctgtgca
gtaaaccaccatggcacacgtttacctatgtaacanacccgcacatcctacacatgtaccccggaactga
agatnanagttgaagaaaaaaaggccaggcatggtggctcatgctgcantctcggcactttgggaggcca
aggcaggaggattgcccgagctcaaaagtttgagaccagcctgggcaacacagtgaaaccccatctccac
tanaatacanaaaatgtatggccaagaatttccaattccagnaaagtagtaggtcacttctattattctt
ttccataggc

4.. 한글화에 의한 사용자 환경 개선

사용자 편의성 증진을 위해 dbSNP, PIR, SWISS-Prot, PhiPsi 등의 사용자 환경을 한글화하였다. HELP 버튼을 추가하여 상세한 설명과 예제를 추가함으로써, 학생이나 초보자들도 쉽게 사용할 수 있도록 하였다.

제 3 절 클러스터시스템기반 생물정보분석시스템의 고도화

1. 국내 BT 분야에서 사용빈도가 높은 분석도구의 신규 구축

가. 연구의 중요성

1990년대 초에 미국에서 시작된 유전체 프로젝트는 2001년 초에 미국과 유럽의 주요 두 그룹에서 경쟁적으로 진행된 인간 유전체 프로젝트 분석 결과가 각각 Science와 Nature에 발표되면서 인간 유전체 프로젝트에 대한 주요한 단락이 지어졌다. 이를 기반으로 하여 인간에 대한 연구 외에도 다양한 고등생물에 대한 유전체 프로젝트의 착수에 가속이 붙고 있다.

국내의 유전체 연구도 2000년을 시작으로 본격화되었으며, 미생물의 유전체 서열결정을 수 주 이내에 완료할 수 있는 시설을 갖추고 있는 벤처기업도 이미 등장하였다. 한편, 이들 과제의 산물인 유전체정보로부터 유용한 학문적 결과와 산업적 결과를 얻기 위해 유전체 후속(post-genome) 프로젝트들이 최근에 진행되고 있다. 유전정보분석을 위한 생물정보학 연구와 개발은 2000년도 이후부터 비교적 활발히 진행되고 있으나, 현재 절대적인 연구개발 인력의 부족으로 대부분의 유전체연구 관련 바이오벤처와 대학 및 연구소에서 필요로 하는 정보 분석과 개발 수요를 맞추지 못하고 있는 실정이다. 최근 유전체 연구가 활발해지고 대학과 연구소 및 기업 등에서 자체 인력을 양성하고 자체적으로 유전정보 분석을 위한 연구개발을 시작하고 있다. 현 단계에서는 국내 일부 팀의 서열 분석에 대한 연구가 전부라고 할 수 있지만, 수 년 내에 이 분야서 활발한 연구결과가 나올 것으로 기대된다.

국내의 유전정보분석 공공서버는 공공 연구소에 의해 시도된 바 있지만, 소규모 인력과 단기간의 예산 지원에 의한 프로젝트여서 지속적으로 이루어지지 못했다. 많은 유전정보 분석 시스템의 프로그램들의 웹을 통해서 제공된다는 특징으로 인해, 자체 시스템의 구축 없이 국내외의 많은 시스템을 활용할 수 있다는 논리가 국내에서의 이러한 기반 시스템 구축 논의에서 장벽이 되어 왔고, 이 분야 과학기술자 양성과 기술력 향상에 장애로서 작용했다. 당연한 결론이지만, 이러한 기반시스템 구축을 위해 경쟁적으로 기술적인 투자를 해 온 기관이나 국가는 이제 선도그룹이 가지는 구축경험과 운영에 대한 방법론(knowhow)을 비롯한 기술적인 성과를 활용하여 사용화의 주요한 기반을 제공하고 있고, 후발 국가는 적지 않은 대가를 지불하면서 이들 노하우와 기술을 따라가야 하는 구도가

형성된 것이다.

국내에서도 생물정보학자들의 연구에 필요한 분석도구의 신규개발에 있어서, 새로운 이론으로 새로운 분석도구를 만들기 이전에 이미 많은 사람들이 사용하고 있는 분석 도구를 국내 사용자들의 편의에 맞도록 시스템화하는 작업이 선행되어야 한다. 사용빈도가 높은 분석도구의 구축이 선 지원되고 난 후에 이를 기반으로 하여 신규 분석 도구의 개발이 이루어져야한다.

나. 연구의 내용

(1) Parallel BLAST

BLAST는 데이터베이스 대상의 서열 상동성 검색을 지원해주는 가장 많이 사용되는 분석도구이다. 거대한 데이터베이스를 대상으로 빠른 검색 결과를 지원해주기 위해서 KISTI에 설치되어있는 리눅스 클러스터의 46개의 노드를 동시에 사용한다. 사용자 질의어의 병렬 처리 작업을 통하여 수행 시간의 단축과 함께 웹 서비스에서의 다중 사용자 지원도 효율적으로 가능케 해준다. 현재 http://www.ccbb.re.kr에서 서비스되고 있다.

<table 3-1> BLAST 프로그램의 종류

·blastp : 아미노산 서열 검색 ·blastn : 핵산 서열 검색 ·blastx : 핵산 서열을 모든 해독틀(RF;reading frame)1에 대하여 번역(translation) 후 단백질 서열 DB를 대상으로 검색 ·tblastn : 입력 서열이 aa 이고 모든 reading frame으로 전사되는 nt DB를 대상으로 검색 ·tblastx nt DB의 모든 6 frame 을 대상으로 입력 쿼리인 nt 서열의 모든 전사된 6 frame을 비교함 tblastx 는 blast 웹에서 nr DB와 함께 사용할 수 없음을 유의.

※자세한 내용은 "국가유전체정보 DB 구축 및 기반기술 개발 " 2002년 12월 보고서(과학기술부) p135~166 참고

(2) ClustalW

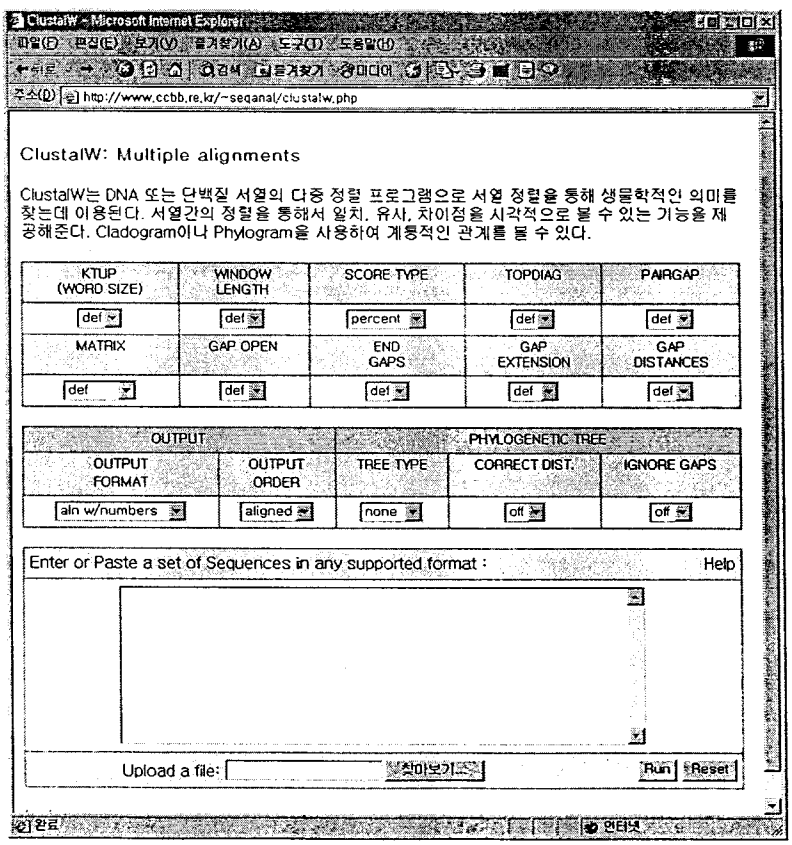
(가) 개요

ClustalW는 단백질 서열이나 DNA 서열들 사이의 다중 정렬을 분석하기 위해서 일

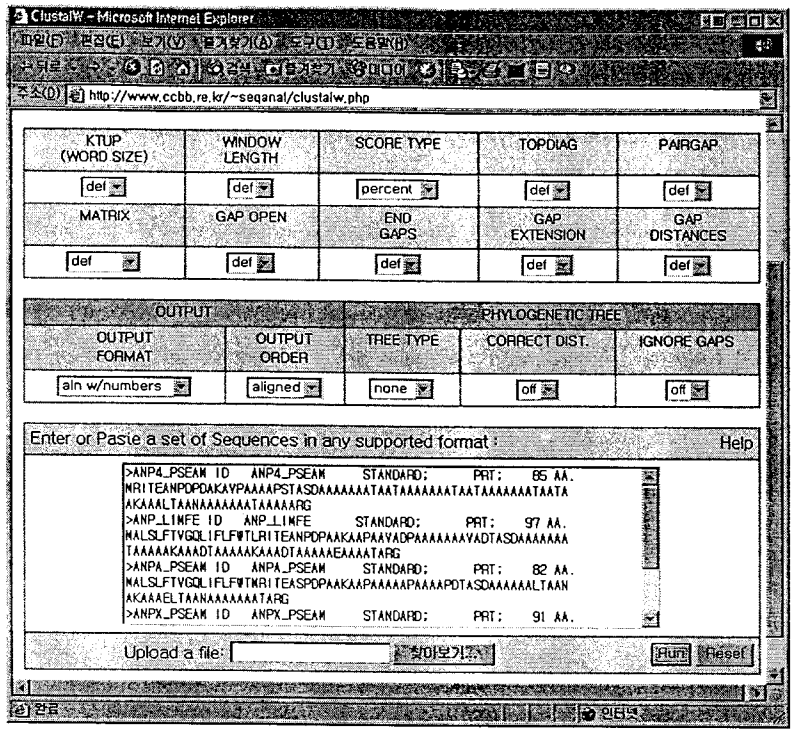
반적으로 많이 사용하는 프로그램이다. 각 서열 간의 동일하거나 유사한 부분들을 분석하기 위해서 사용하고 좀 더 시각적으로 확인하기 위해서는 계통 분석 프로그램을 이용한다.

(나) 구축

EBI(European Bioinformatics Institute)에서 배포하는 ClustalW 파일을 내려받은 후 하드웨어에 맞도록 설치한다. 설치 후에는 명령 라인의 이 프로그램을 사용자가 웹 화면을 통해서 질의어를 입력하고 그 결과를 보기 위해서 웹 인터페이스를 PHP 언어를 사용하여 구축하였다. 스크립트 언어인 PHP를 사용함으로해서 시스템 부하가 적고 인터페이스와 ClustalW의 시스템 간의 연동을 유연하게 해 줄 수 있다. 사용자 인터페이스에서는 ClustalW에서 사용되는 각 옵션을 처리해 줄 수 있게 하며, 단백질 또는 DNA 서열을 직접 입력하는 것 외에 파일 업로드도 가능하도록 만들었다. 입력 서열의 형식은 NBRF/PIR, EMBL, UniProt/SwissProt, Pearson(Fasta), GDE, ALN/ClustalW, GCG/MSF, RSF 등의 7가지를 지원해준다.

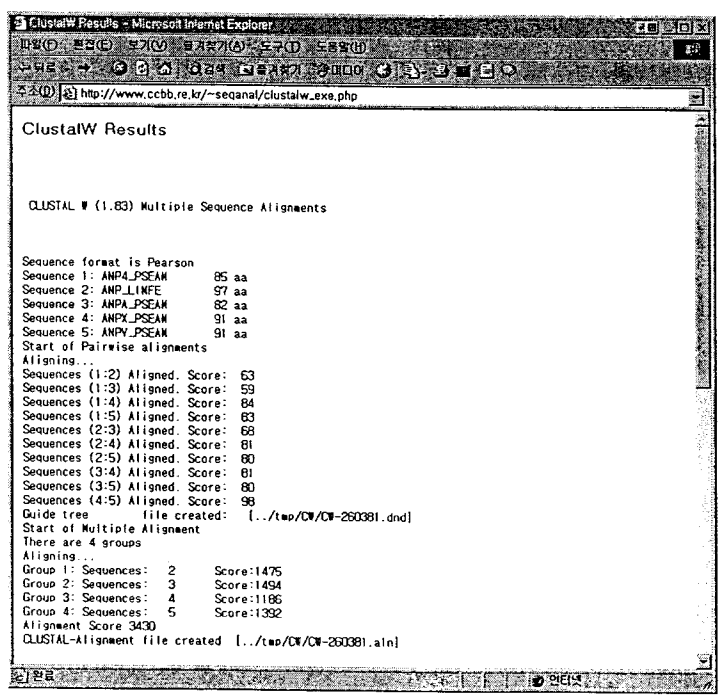


<figure 3-1> ClustalW main page



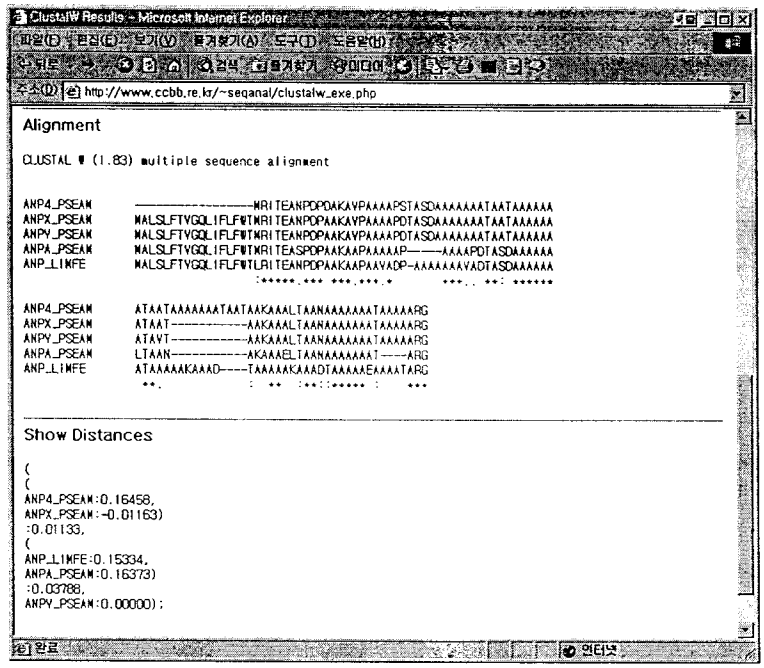
<figure 3-2> Insert query sequence for Searching

원하는 옵션을 체크한 후에 실행 버튼을 누르면 다중 정렬 결과를 볼 수 있다



<figure 3-3> Results of ClustalW: alignment score

가장 상단에는 다중 정렬 결과를 서열별 점수표로 보여준다. 그 하단에는 서열들 간의 다중 정렬 상태를 시각적으로 쉽게 확인할 수 있는 가시화 기능을 제공해준다. http://www.ccbb.re.kr/~seqanal/clustalw.php에서 서비스되고 있다.



<figure 3-4> Results of ClustalW : Alignment and Show Distance

(2) InterProScan

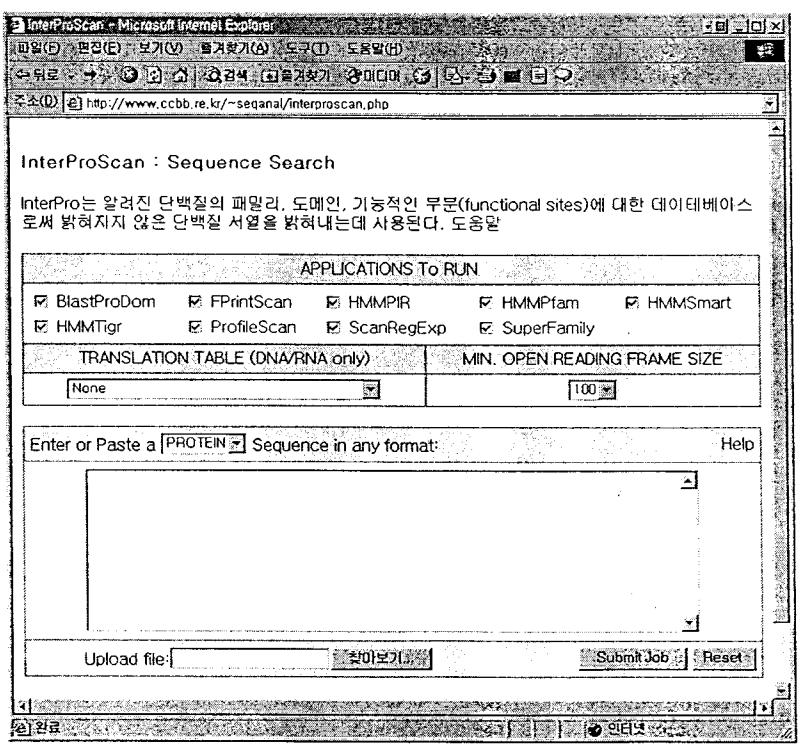
(가) 개요

InterProScan은 단백질 도메인과 기능적인 위치(functional sites) 정보를 모아놓은 데이터베이스로써 신규 단백질의 기능을 예측하는데 널리 사용되고 있다. 지금까지 알려진 단백질 관련 데이터베이스 PROSITE, PRINTS, Pfam, ProDom. SMART, TIGRFAMs등의 데이터베이스를 통합해 놓았기 때문에 한번의 단백질 서열 검색으로 다양한 결과를 찾아내어주는 편리한 기능을 제공한다. 관련 검색 모듈은 Perl로 구현되어 있으며 유닉스 시스템에 적합하도록 배포된다.

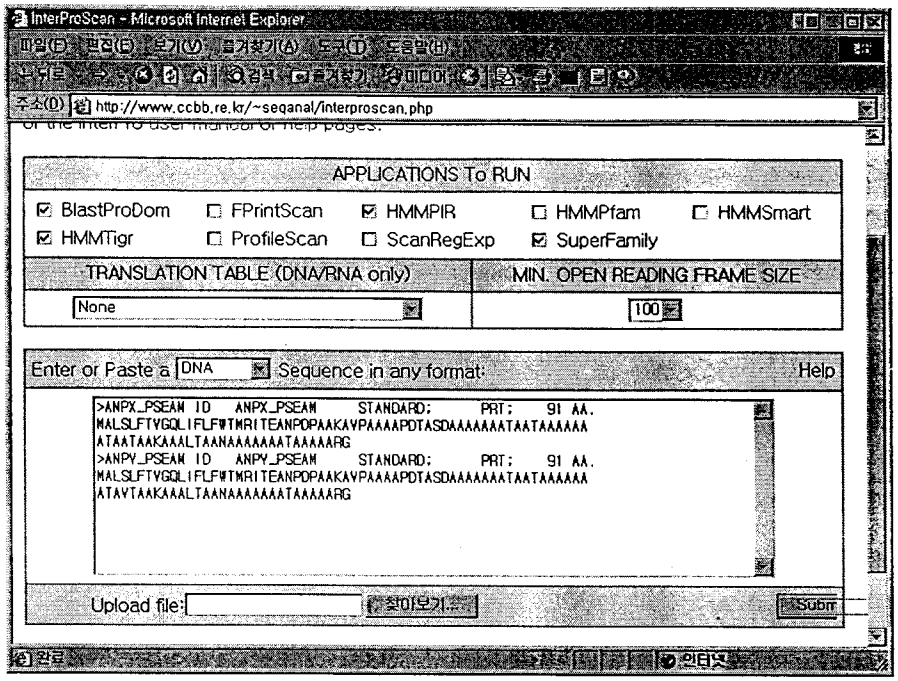
(나) 구축

EBI(European Bioinformatics Institute)에서 배포하는 ClustalW 파일을 내려받은 후 하드웨어에 맞도록 설치한다. 설치 후에는 명령 라인의 이 프로그램을 사용자가 웹 화면을 통해서 질의어를 입력하고 그 결과를 보기 위해서 웹 인터페이스를 PHP 언어를 사용하여 구축하였다. 스크립트 언어인 PHP를 사용하면 시스템 부하가 적고 인터페이스와 InterProScan의 시스템 간의 연동을 유연하게 해 줄 수 있다. 사용자 인터페이스에서는 InterProScan에서 사용되는 각 옵션을 처리해 줄 수 있게 하며, 단백질 또는 DNA 서열을 직접 입력하는 것 외에 파일 업로드도 가능하도록 만들었다. 입력 서열의 형식은 NBRF/PIR, EMBL, UniProt/SwissProt, Pearson(Fasta), GDE, ALN/ClustalW, GCG/MSF, RSF 등의 7가지를 지원해준다.

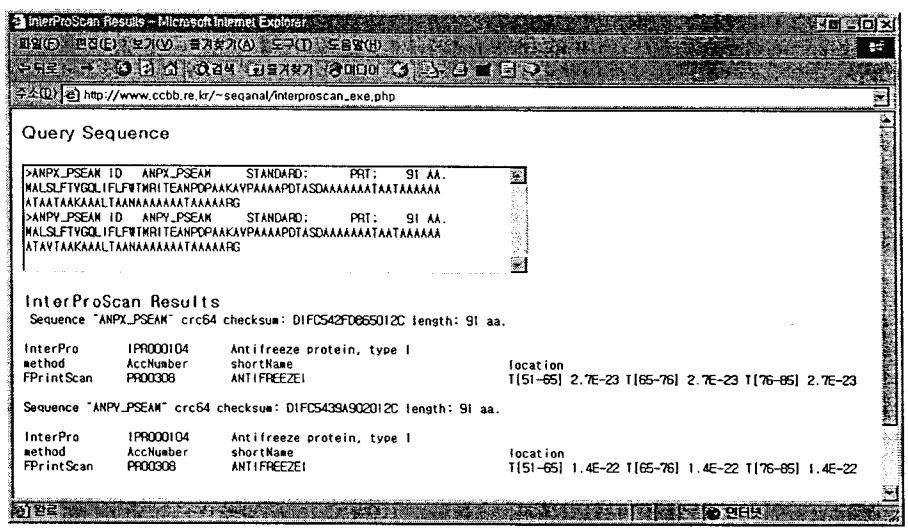
사용자 질의어가 단백질 서열인 경우에는 검색하고자 하는 데이터베이스를 선택한 후에 실행버튼을 누르면 된다. DNA 서열을 질의어로 입력할 때에는 TRANSLATION TABLE의 옵션을 눌러서 원하는 변역 표를 선택한 후에 검색을 시작하면 된다. DNA를 단백질로 번역하는 과정 한 단계가 추가된 후에 각 단백질 서열들에 대한 검색 과정은 동일하게 진행된다. InterProScan 시스템은 http://www.ccbb.re.kr/~seqanal/interproscan.php 에서 사용할 수 있다.



<figure 3-5> InterProScan main page



<figure 3-6> Insert protein sequence for searching



<figure 3-7> Result of Searching

(3) FASTA

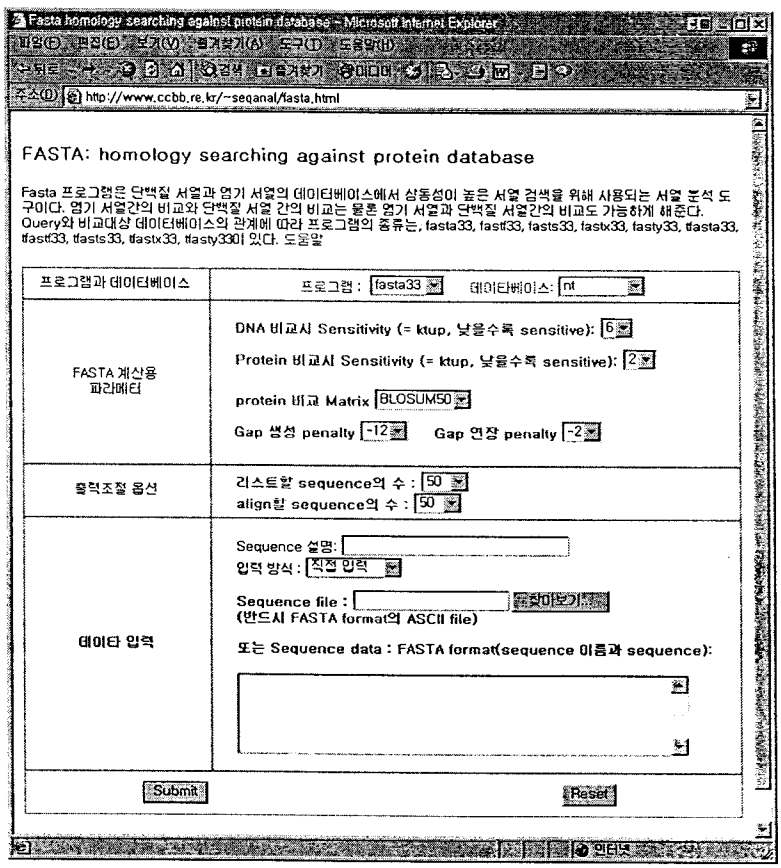
(가) 개요

Fasta 프로그램은 단백질 서열과 염기 서열의 데이터베이스에서 상동성이 높은 서열 검색을 위해 사용되는 서열 분석 도구이다. 염기 서열간의 비교와 단백질 서열 간의 비교

는 물론 염기 서열과 단백질 서열간의 비교도 가능하게 해준다. Query와 비교대상 데이터베이스의 관계에 따라 프로그램의 종류는, fasta33, fastf33, fasts33, fastx33, fasty33, tfasta33, tfastf33, tfasts33, tfastx33, tfasty33이 있다

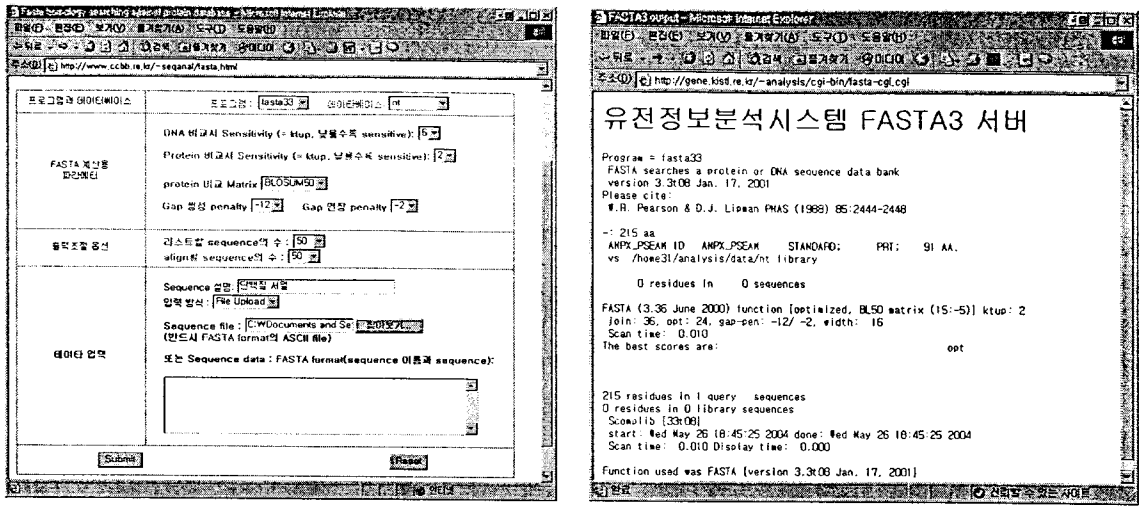
(나) 구축

EBI(European Bioinformatics Institute)에서 배포하는 ClustalW 파일을 내려받은 후 하드웨어에 맞도록 설치한다. 설치 후에는 명령 라인의 이 프로그램을 사용자가 웹 화면을 통해서 질의어를 입력하고 그 결과를 보기 위해서 웹 인터페이스를 CGI를 이용하여 구현하였다. Fasta 분석 도구의 각 옵션 설정을 웹화면에서 지원해주도록 하였다. Fasta 시스템은 http://www.ccbb.re.kr/~seqanal/fasta.html에서 사용가능하다.



<figure 3-8> Fasta main page

사용하고자 하는 Fasta 프로그램의 종류와 데이터베이스를 선택한 후 Fasta 계산용 변수들을 체크한 후에 서열 데이터를 직접 입력하거나 파일 업로드를 통하여 검색을 진행하도록 한다. 검색 결과 화면도 초기화면에서 사용자가 원하는 사양에 맞도록 변경가능하다.



<figure 3-9> Insert the protein sequence and result of searching

2. 클러스터 시스템 기반 생물 정보분석시스템의 고속화 서비스

가. 연구범위 및 연구수행 방법

연구범위	연구수행방법 (이론적,실험적 접근방법)	구체적인내용
시스템 병렬화 및 BLAST 프로그램 성능 향상	- 프로그램 프로파일링으로 성능상 병목 지점을 찾고 이를 최적화 - 클러스터 시스템에서의 프로그램 수행 병렬화	- gprof등의 프로파일링 도구를 사용하여 성능상 병목 지점 찾아 최적화 - 컴파일러의 최적화 옵션을 적절히 사용하여 성능 개선 - MPI를 이용하여 프로그램의 수행을 병렬화

나. 연구수행 내용 및 결과

프로그램의 성능을 향상시키는 과정에서 처음 할 일은 프로그램의 어느 지점에서 시스템의 자원을 가장 많이 소모하고 수행 시간이 오래 걸리는지를 알아내는 것이다. GNU의 gprof를 사용하여 프로파일링한 결과, 검색하려는 염기서열을 일정 크기로 나누어 자른 염기서열 조각을 DB의 염기서열과 비교하고 여기서 일치하는 조각들에 대해 한 염기씩 늘여가며 비교해보는 BlastNtWordFinder()라는 함수와 이 함수에서 호출하는 몇몇 개의

함수들이 전체 프로그램 수행 시간의 80~90%를 소모한다는 사실을 알 수 있었다. 따라서 이 부분의 알고리듬을 개선하는 것이 성능 향상을 위해 중요한 작업이다.

그렇지만 이 부분은 BLAST의 핵심 알고리듬이고 전체적인 구조를 바꾸기는 어려웠다. 그래도 전체 프로그램의 구조를 살펴본 바로는 많은 부분이 반복 작업이어서 일부분의 성능을 개선하는 것으로도 전체 성능을 크게 향상시킬 수 있을 것으로 판단되었다. 특히 loop 구문을 최적화하고 서로 독립적인 계산을 한 번에 수행할 수 있도록 한다면 성능 향상을 이룰 수 있다. 이러한 작업은 굳이 수작업으로 할 필요 없이 최신의 컴파일러들의 최적화 옵션을 이용하여 시스템의 아키텍처에 최적화된 실행 파일을 얻을 수 있었다. 성능 향상을 시험하는데 사용한 프로그램과 시스템, 검색에 사용한 염기서열은 다음과 같다.

* 프로그램

BLAST: NCBI version 2.2.8, 2004-02-10

Intel C++ compiler: version 8.0 for IA32 Build 20031016Z

* 시스템

시스템 1:

Pentium 4 3.0 GHz (HyperThreading), RAM 2 GB

Gentoo Linux (kernel 2.6.3smp), gcc 3.3.2 20031218, glibc 2.3.2

시스템 2: 리눅스 클러스터 (40 노드)

Dual Pentium III 1.266 GHz, RAM 1 GB

RedHat Linux 7.3 (kernel 2.4.18smp), gcc 2.96 20000731, glibc 2.2.5

* 염기서열

염기서열 1: NCBI에서 제공하는 BLAST DB 중 other_genomic에서 임의로 추출한 염기서열, 길이 50, 100, 500, 1000, 5000 각각 2 개씩

염기서열 2: other_genomic에서 임의로 추출한 염기서열, 길이 50, 100, 500, 1000, 5000

각각 1 개씩

염기서열 3: H.vulgare에서 임의로 추출한 염기서열, 길이 546, 719, 1154, 1180, 1311, 1344, 1556, 1831, 2263, 3790 각각 1개씩

수행 시간은 프로그램의 수행이 끝날 때까지의 실제 시간을 측정하였다. 명령 행은

'time blastall -i (검색하려는 염기서열) -p (검색에 사용하는 프로그램, blastn/blastx 등) -d (검색하려는 DB) -o (결과 파일)'

으로 하여 이외의 모든 옵션은 기본값을 사용하도록 하였다. 검색할 DB를 바꾸면 그 첫 번째 실행 시에는 저장장치로부터 DB 파일을 읽으므로 그 결과는 측정값에 포함시키지 않았다. 네 번 실행해서 첫 번째 실행 결과를 제외한 평균값을 수행시간으로 기록하였다.

시스템 1은 Intel사의 Pentium 4 프로세서에 기반한 시스템으로 최신 버전의 컴파일러와 SIMD(Single Instruction Multiple Data; MMX, SSE, SSE2 등)를 이용할 수 있었다. 아래와 같은 최적화 옵션들을 이용하여 컴파일하고 각각의 프로그램 수행시간을 측정하였다.

* 최적화 옵션

gcc: gcc -O3 -mcpu=pentium4 (gcc: GNU C/C++ compiler)

gcc2: gcc -O3 -mcpu=pentium4 -funroll-loops -fomit-frame-pointer

gcc3: <gcc2> -mmmx -msse -msse2

gcc4: gcc -O2 -mcpu=pentium4 -funroll-loops -fomit-frame-pointer

gcc5: <gcc5> -mmmx -msse -msse2

gcc6: gcc -O3 -march=pentium4 -funroll-loops -fomit-frame-pointer

-mmmx -msse -msse2

gcc7: <gcc6> -mfpmath=sse

gcc8: <gcc6> -falign-functions -falign-jumps -falign-loops -falign-labels

-finline-functions

(manual에서 -O3일 경우 사용하는 옵션이라고 한 것과 실제로 'gcc -v -Q'로 확인한 것과 비교하여 빠진 것 포함시킴)

icc1: icc -xN (icc: Intel C++ compiler, -xN: pentium 4에 최적화)

icc2: icc -xN -ip -parallel -unroll

icc3: icc -xN -ipo -ipo_obj -parallel unroll

ncbi: NCBI에서 제공하는 실행 파일을 사용

('gcc -O3 -mcpu=pentiumpro'와 같음)

* 수행 시간 비교

gcc	gcc2	gcc3	gcc4	gcc5	gcc6
147.41	129.84	126.36	134.51	131.02	126.74
gcc7	gcc8	icc1	icc2	icc3	
127.46	125.83	108.70	106.61	107.65	

(단위: 초, 프로그램/DB/Query: blastn, est_human, 염기서열 1)

최적화 옵션 중 '-O2'의 경우 프로그램에 따라 '-O3'보다 성능이 좋은 경우도 있으나 BLAST는 '-O3'의 경우가 더 좋은 결과를 나타내었다. 이를 기본으로 여러 가지 최적화 옵션을 시험해 본 결과 세 가지의 옵션이 성능 향상에 크게 도움이 되었다. 하나는 loop의 수행 속도를 높이기 위해 컴파일 시 loop unroll을 지시하는 '-funroll-loops'이고 두 번째는 함수 호출시의 오버헤드를 감소시키는 '-fomit-frame-pointer' 이다. 마지막은 Pentium 4의 SIMD 명령어를 이용하여 서로 독립적인 계산들은 한 번에 계산할 수 있도록 하는 '-mmmx -msse -msse2'이다. 이를 사용해서 수행 성능을 14% 향상시킬 수 있었다(gcc3의 경우).

Intel 사의 C++ compiler는 자사의 CPU들을 사용한 시스템에 대해 gcc보다 더 최적화된 실행 파일을 산출한다고 하여 이를 이용하여서도 시험하였다. gcc의 경우와 마찬가지로 loop unroll 옵션인 '-unroll'과 SIMD 명령어를 사용하도록 하는 '-parallel' 옵션이 가장 성능에 많은 영향을 주었고 그 외의 여러 옵션들은 거의 영향이 없었다. Intel C++ compiler로는 28%의 성능 향상이 있었다. 검색하려는 염기 서열과 비교하는 DB등에 따라

서 수행 성능이 차이가 나지만 최소 10%에서 최대 33%까지의 성능 향상을 볼 수 있었다.

시스템 1은 프로세서가 HyperThreading(TM)이라는 기술을 이용하여 dual processor 시스템처럼 동작한다. 그래서 BLAST 프로그램이 두 개 이상의 CPU를 사용하도록 지시하여 thread를 이용한 병렬 처리가 성능 향상에 얼마나 도움이 되는지 알아보았다. 예상대로 2개의 CPU를 사용하도록 했을 때 35%의 성능 향상이 있었다.

* CPU의 수에 따른 수행 시간 비교

	x	1	2	4	8	16
ncbi	128.31	128.79	84.33	82.19	82.75	85.66
gcc	127.39	127.27	79.43	80.14	81.20	82.26
gcc3	111.26	110.13	83.63	85.32	84.30	85.86
icc3	85.12	84.93	70.69	72.38	71.29	72.30

(단위: 초, 프로그램/DB/Query: blastn, nt, 염기서열 2)

* x: -a option 지정하지 않은 경우, 나머지는 사용하는 CPU 개수

시스템 2는 시스템 1보다는 구형의 시스템이고 운영체제 및 컴파일러가 이전 버전이지만 여러 개의 계산 노드를 가지는 클러스터 시스템이다. 후에 언급할 시스템 병렬화와 관련하여 이용하기 위해 사용하였다. 하나의 노드를 이용하여 컴파일러의 최적화 옵션을 달리하며 수행 성능을 측정하였다. 이 시스템에 설치된 gcc는 SIMD에 관련된 최적화 옵션이 없다.

* 최적화 옵션

gcc: gcc -O3 -mcpu=pentiumpro

gcc2: gcc -O3 -mcpu=pentiumpro -funroll-loops -fomit-frame-pointer

gcc3: gcc -O3 -mcpu=pentiumpro -funroll-loops

gcc4: gcc -O3 -mcpu=pentiumpro -fomit-frame-pointer

gcc5: gcc -O2 -mcpu=pentiumpro

gcc6: gcc -O2 -mcpu=pentiumpro -funroll-loops -fomit-frame-pointer

gcc7: gcc -O3 -mcpu=i686 -funroll-loops -fomit-frame-pointer

icc1: icc -xK (-xK: pentium III에 최적화)

icc2: icc -xK -ip -parallel -unroll

icc3: icc -xK -ipo -ipo_obj -parallel -unroll

ncbi: NCBI에서 제공하는 binary

* 수행 시간 비교

	gcc	gcc2	gcc3	gcc4	gcc5
blastn, nt.01, seq.1	196.10	187.02	194.73	188.27	195.49
	gcc6	gcc7	icc1	icc2	icc3
	189.42	187.13	161.94	150.86	150.84

(단위: 초, 프로그램/DB/Query = blastn, est_human, 염기서열 1)

시스템 1의 경우와 같은 옵션들을 사용했을 때 성능이 가장 좋았으나 시스템 1에 비해서는 향상 정도가 적었다(gcc, icc 각각 4%, 23%). 이는 프로세서가 제공하는 SIMD 명령어의 종류와 수의 차이(Pentium II의 MMX, SSE와 Pentium 4의 SSE2의 차이)에 기인한 것도 있겠지만 Intel C++ compiler를 사용한 경우가 gcc의 경우보다 성능 향상이 큰 것은 컴파일러에서 이러한 명령을 사용하도록 최적화하는 방법의 차이에서 비롯하는 것이라 생각된다.

사용하는 CPU의 수에 따른 수행시간은 시스템 1과 마찬가지로 2개 이상으로 지정하였을 경우 성능이 향상되었고 시스템 1이 35% 정도 향상된 데 비해 45% 정도 향상되었다. 결과에는 나타내지 않았지만 검색하려는 염기서열과 DB를 바꾸어가며 성능을 측정하였을 때 시스템 1보다는 시스템 2에서의 성능 향상 정도가 두드러졌다. Pentium 4의 HyperThreading(TM)에 의한 병렬처리보다는 물리적으로 2개의 CPU에 의해 수행되는 병렬처리가 더 효과적인 것으로 나타났다.

* CPU의 수에 따른 수행 시간 비교

	x	1	2	4	8	16
ncbi	87.20	87.19	45.71	45.70	45.72	47.31
gcc2	85.30	85.15	45.16	45.19	45.19	45.32
icc3	69.09	69.10	37.62	37.63	37.69	37.72

(단위: 초, 프로그램/DB/Query = blastn, nt.01, 염기서열 2)

* x: -a option 지정하지 않은 경우, 나머지는 a 사용할 CPU 개수

프로그램의 성능을 향상시키는 다른 방법으로는 여러 개의 컴퓨터를 이용하여 같은 작업을 병렬화하는 것이다. 이렇게 하는 것도 여러 가지 방법이 있지만 BLAST에 대해서는 MPI(Message Passing Interface)를 이용하여 병렬 처리를 구현한 mpiBLAST(http://mpiblast.lanl.gov/index.html)가 있어 이를 이용하였다. 40개의 계산 노드로 구성된 시스템 2에 LAM/MPI 버전 7.0.4과 mpiBLAST 버전 1.2.1를 설치하여 성능을 측정하였다.

mpiBLAST는 검색할 DB를 비슷한 용량의 여러 조각으로 나누어 이를 각 계산 노드에 배분하고 검색하고자 하는 염기서열을 각 노드에서 검색함으로써 전체적인 수행 성능을 향상시킨다. 결과에서 보듯이 노드 수에 정비례하지는 않지만 사용하는 노드 수를 늘이면 수행 시간이 짧아짐을 알 수 있다.

mpiBLAST는 MPI에 관련된 부분과 결과를 취합하는 부분을 새로 작성하고 실제 BLAST 검색은 NCBI에서 제공하는 BLAST 프로그램의 함수들을 호출하여 실행하도록 구현되었다. 따라서 이 부분은 앞에서 수행한 최적화를 적용할 수 있다. 결과를 보면 최적화를 하지 않은 경우보다는 빠르게 실행되나 MPI를 이용하지 않았을 경우 28% 정도인데 비해 10% 미만의 성능 향상을 보였다. 이는 MPI에 의한 오버헤드가 있기 때문인 것으로 생각된다. MPI를 이용한 프로그램도 프로파일링을 해서 최적화를 할 수 있으므로 이에 대한 연구도 필요하다고 본다. 그리고 현재 구현된 mpiBLAST는 개별 노드에서 여러 CPU를 이용할 수 없다. 앞서의 결과에서 보듯이 개별 노드에서 SMP의 장점을 살린다면 좀 더 성능을 향상시킬 수 있으므로 이에 대한 연구도 추가로 진행되어야 하겠다.

* 최적화 옵션

gcc: gcc -O3 -mcpu=pentiumpro

icc: icc -xK -ip -parallel unroll : ncbi toolbox만 이것으로 컴파일

* 수행 시간 비교 1

프로그램/DB/염기서열	MPI 사용 안함	MPI 사용
blastx, nr, 염기서열 1	1013.25	164.02
blastx, nr, 염기서열 3	1070.96	183.91
blastn, est_human, 염기서열 1	287.38	41.61
blastn, est_human, 염기서열 3	313.54	39.50

(단위: 초, nr은 7개로, est_human은 10개로 분할)

* 수행 시간 비교 2

프로그램/DB/DB 분할 개수	gcc	icc
blastx, nr, MPI를 사용 안함	850.20	-
blastx, nr, 10개로 나눔	292.07	-
blastx, nr, 20개로 나눔	181.88	-
blastn, nt, not MPI	976.55	868.34
blastn, nt, 20개로 나눔	92.12	79.74
blastn, nt, 39개로 나눔	67.07	61.28

(단위: 초, 염기서열 3 사용)

다. 연구개발목표의 달성도 및 자체평가

(1) 연구개발목표의 달성도

목표	달성도	내용
클러스터 시스템 기반 생물정보분석시스템의 고속화 서비스	95 %	- 프로그램의 최적화로 10~33%의 성능 향상 - 프로그램의 병렬화로 클러스터 시스템에서 사용하는 노드 수에 비례하는 성능 향상 달성

(2) 평가의 착안점에 따른 목표달성도에 대한 자체평가

고속화 신장률	프로그램의 최적화로 10~33%의 성능 향상과 MPI를 이용한 프로그램 병렬화로 사용하는 노드 수에 비례하는 성능 향상을 이루어 대용량의 유전체 정보 데이터를 빠르게 분석할 수 있게 됨

3. 생물정보 자동 마이닝 웹서비스

생명정보 통합 데이터베이스는 21세기 생명정보시대의 꽃으로 등장할 것이다. 생명정보 분야의 급속한 발전은 생명과학자들에게 많은 지식과 정보를 빠른 시간 안에 제공하려 하고 있다. 하지만 전 세계의 다양한 기관에서 생산되어지는 정보들을 매일 정확히 파악하기란 여간 힘든 일이 아니다. 특히 생명과학자들은 자신의 연구와 최근의 정보를 비교하면서 지속적인 연구개발을 하기에는 많은 시간과 노력이 필요한 실정이다.

최근에 자료에 의하면 현재 NCBI의 GenBank는 기하급수적으로 증가해 년 간 5백4십만 건의 새로운 염기서열이 보고 되고 있는 실정이다. 또한 영국의 EMBL 과 일본의 DDBJ 도 그 자료가 계속적으로 증가하고 있다. (Dennis *et al.*, 2003)

이러한 상황 속에서 과학자들의 정보에 대한 정확하고 신속한 서비스를 요구하고 있

는 실정이다. 이 같은 문제점을 해결하기 위해 국내뿐 만 아니라 전 세계적으로 많은 연구 개발이 이루어지고 있다.

특히 최근에는 동일한 유전자에 대한 종합적인 정보의 요구가 증가하고 있다. 즉, 기존에는 단순한 유전자의 기능만을 분석하려는 반면, 최근에는 유전자의 염기서열, 3차원구조, 유전자기능 뿐 만 아니라 질병과의 관계, 유전체 내의 위치, 발현양상, 돌연변이정보, domain의 특징 등등 하나의 유전자에 대한 다양한 정보를 필요하게 되고 있다.

이 같은 필요성은 통합 검색 데이터베이스의 등장을 가져왔으며, 현재 NCBI의 Entrez, EBI의 SRS등이 널리 사용되고 있다. 점점 더 빨라지고 있는 현대 과학기술 속에서 정보의 신속한 검색은 앞으로 중요한 주제가 될 것이다.

가. SRS와 병렬컴퓨터의 연계방법 개발

(1) SRS 도입

(가) SRS의 소개

SRS(Sequence Retrieval System)는 한번의 검색으로 다양한 데이터베이스에 검색결과를 받아볼 수 있다. 처음에는 생물학적 서열정보데이터베이스의 손쉬운 접근을 위해 개발되어졌다 (Etzold and Argos, 1993). 하지만 최근에는 약 400개 이상의 데이터베이스를 지원하고 있는 통합 데이터베이스로 각광 받고 있다. 특히 다양한 종류의 데이터베이스를 보유하고 있음에도 불구하고, 동일한 사용자 화면을 제공하는 특징이 있다.

특히 단순한 자료 검색뿐만 아니라 생물정보에 대한 분석 프로그램과 연계시켜 다양한 분석도 가능하게 하는 종합시스템이다. 처음에는 EBI (http://srs.ebi.ac.uk)에서 개발되어 졌지만 1999년 이후 LION Bioscience AG (http://www.lionbioscience.com)에 의해 개발되어지고 있으며, 최근 7.1.3 버젼이 배포되고 있다. (Evgeni *et al.*, 2002)

SRS의 특징은 상용 데이터베이스 프로그램을 기반으로 하는 것이 아니라 텍스트문서를 기반으로 직접 데이터베이스를 구축하는 시스템으로 어떤 종류의 컴퓨터시스템에도 이식이 가능하다. SRS가 다른 시스템과 구별되는 핵심적인 사항은 객체지향적인 설계에 있다.

기존 데이터베이스의 한계는 서로 다른 데이터들을 통합적으로 정리하기 복잡하다는 것인데, SRS는 이 문제를 객체지향적 설계를 도입함으로서 극복할 수 있었다.

특히 Icarus라는 고유의 스크립트언어를 이용하여 다양한 종류의 객체정의와 분석이

가능해 졌다.

이 같은 이유로, 최근 정보기술의 발달로 많이 사용되어지고 있는 관계형 데이터베이스 관리 시스템 (RDBMS)에 비해 약 10-100배의 속도 증가를 가져왔다. 또한 직접 데이터 파일에 접근하는 방식으로 기존 RDBMS보다 약 2 - 5배적은 저장 공간을 차지한다. 마지막으로 새로운 데이터 형태가 등장 했을 때 RDBMS는 그 자료형태의 구현이 복잡한 반면, SRS는 자료의 정의 및 분석저장을 Icarus를 이용하여 손쉽게 적용할 수 있는 장점이 있다.

(나) SRS에서 병렬 컴퓨터 환경 지원

SRS는 현재 Sun 사의 Sun Grid Engine을 비롯해서 LSF, DQS, CODINE을 지원하고 있는 상태이다. SRS는 최근 데이터베이스와 분석 프로그램간의 연동을 지원하는 시스템으로 개발되고 있는 상황에서 이 같은 병렬시스템의 지원은 필수 불가결한 사항이 앞으로 될 것이기 때문에 지속적으로 추가적인 병렬시스템의 지원이 확대될 것으로 기대된다.

본 과제에서는 SRS에서 지원하는 병렬시스템 중에 Sun 사가 개발한 Sun Grid Engine을 이용하여 병렬 컴퓨팅 환경을 구성하였다.

(다) 통합데이터베이스 연계

데이터는 다른 데이터와의 연계 시 더욱 가치를 지니게 된다. SRS는 이 같은 데이터베이스간의 연계가 가능하도록 각 데이터베이스에서 상호 참조가 가능한 항목을 추출하여 이를 사전처럼 찾기 쉽도록 인덱스화 함으로서 데이터 검색 시 검색하려는 데이터베이스뿐 만 아니라 인접한 데이터베이스의 정보까지 부수적으로 찾아볼 수 있도록 설계되었다.

이 같은 특징은 앞으로 지식기반의 2차원 데이터베이스의 구축을 가능하게 하였다.

(2) 병렬컴퓨터 관리 프로그램인 Sun Grid Engine 셋팅

(가) 그리드 컴퓨팅 (Grid Computing)

그리드 컴퓨팅이란 여러 곳에 분산된 컴퓨터들을 연결하여 하나의 거대한 컴퓨터시스템을 구축하고자 하는 방법입니다. 즉 단일 시스템부터 수 천개의 프로세서를 이용하는 슈퍼 컴퓨터급까지 컴퓨터의 종류와 무관하게 하나의 거대 시스템을 구축할 수 있습니다.

이 같은 방식의 컴퓨팅 환경을 지원하고자 개발된 것이 Sun 사의 Sun Grid Engine입니다. 본 프로그램은 다양한 종류의 컴퓨터들을 수용하여 사용하고자 자원 및 정책 관리를 통해 작업을 관리하며, 이를 기반으로 사용자에게 서비스를 제공할 수 있습니다.

즉 컴퓨터들이 위치, 그리고 종류와 상관없이 대형 슈퍼컴퓨터를 구현할 수 있는 시스템입니다.

(나) 시스템 구축

현재 KISTI에 구축된 Cluster 시스템은 총 112 개의 노드로 구성되어져 있으며, 각각의 노드는 사양은 펜티엄4 1.7G CPU를 장착하였고, RAM 1G와 하드디스크 20G, Gigabit 네트워크로 구성되어져 있다.

(다) Sun Grid Engine의 구성

Sun Grid Engine은 크게 4가지의 종류의 호스트로 구분된다 (그림 1).

① 마스터 호스트

전체 클러스트의 중심이 되는 호스트로 모든 활동의 중심이 된다. 마스터 데몬은 sge_qmaster 및 sge_schedd을 실행하며, 대기열과 작업을 제어하고 있다.

기본적으로 마스터 호스트는 관리호스트와 제출호스트 기능도 수행하고 있다.

② 실행 호스트

실행 호스트는 Sun Grid Engine에 의해 작업을 실행 할 수 있는 권한을 갖는 노드이다. 그러므로 실행 호스트는 대기열에 따라 작업내용을 실행한다. 이 같은 실행을 관리하는 데몬은 sge_execd 이다.

③ 관리 호스트

Sun Grid Engine의 모든 종류의 관리 활동을 수행할 수 있는 호스트로 관리 활동은 작업열의 수정 및 작업처리 규칙의 관리가 가능한 호스트이다.

④ 제출 호스트

제출 호스트는 일괄처리작업의 제출 및 제어만 허용하는 호스트로 작업자가 qsub를

통해 작업을 제출하고 qstat를 통한 작업상태의 확인이 가능한 호스트이다.

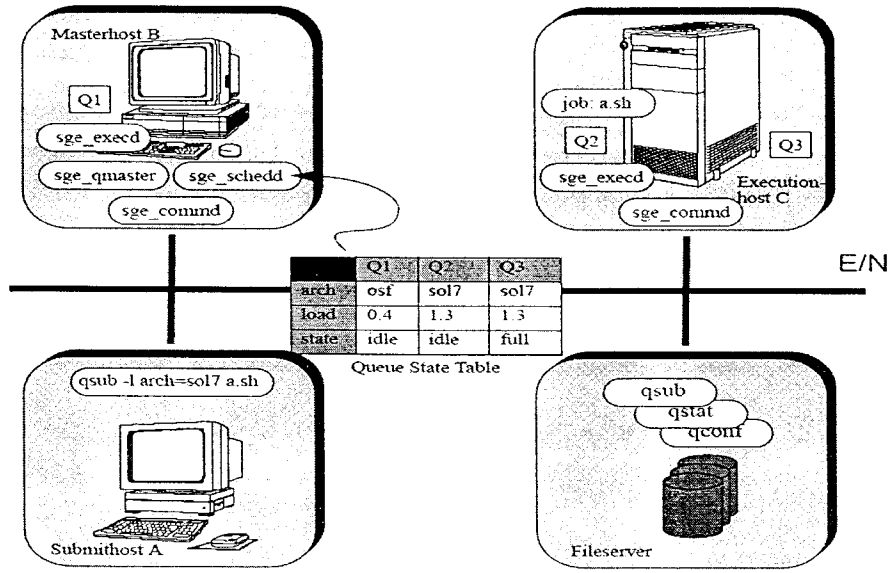


FIGURE 1-1 Component Interaction in the Sun Grid Engine System

<figure 3-10> Sun Grid Engine system내의 컴포넌트간의 상호작용 구성도

(라) 프로그램 다운로드

현재 Sun Grid Engine은 http://gridengine.sunsource.net 에서 자유롭게 다운 받을 수 있으며, 최신 안정화버전은 5.3p6 이며, 개발 버전은 6.0 beta2이다.

또한 다양한 OS에서 설치할 수 있도록 프로그램을 제공하고 있다.

(마) 프로그램의 설치 및 환경설정

① 서비스 포트 추가

설치하려는 클러스터의 모든 노드의 /etc/services 파일에 sge_commd를 위한 포트를 지정한다. 가능하면 1024번 이하의 포트를 지정한다. 모든 노드가 같은 포트번호를 사용하여야 한다.

/etc/services 파일에 아래 사항을 추가 한다

sge_commd 536/tcp # Sun Grid Engine

본 sge_commd 데몬은 마스터 호스트와 실행 호스트간의 TCP/IP 통신으로 정보를 교환하며, 작업의 실행 및 실행된 결과들의 교환작업에 사용되어진다.

② 작업 설치 디렉토리

Sun Grid Engine은 공용으로 사용할 수 있도록 /usr/local/sge 에 설치하였다. 설치 시 사용자들이 <Sun Grid Engine Root>/<Cell>/common/ 에 접근할 수 있도록 하였다. 또한 설치시 구성하는 클러스트 환경이 단일 클러스트인지 서브클러스터의 집합인지를 결정하여 기본 SGE_CELL환경을 셋팅하였다.

현재 KISTI내의 112노드는 단일 클러스터로 구성되어져 있다.

③ 설치 계정 및 파일 접근 권한

Sun Grid Engine은 설치시 root 권한이 필요하지만 실제 사용시 관리자는 root가 아닌 별도의 사용자이여야 한다. 즉 설치시 root가 관여하지만 사용시에는 Sun Grid Engine용 관리자의 권한으로 모든 관리가 가능하기 때문이다.

특히 sge_commd 데몬을 제외한 나머지 데몬들은 root가 아닌 Sun Grid Engine관리자의 권한으로 실행되어진다.

그리고 일반사용자들은 각 노드에서 사용자의 디렉토리를 사용하여야 하므로 NFS를 이용하여 모든 노드에서 공통으로 마운트하여 사용할 수 있도록 하였다.

이 같은 환경은 단순한 웹프로그램에서의 사용뿐만 아니라 일반 사용자들의 이용도 가능할 수 있도록 셋팅하였다.

④ 설치 스크립트

Sun Grid Engine은 간편한 설치를 위해 설치용 스트립트를 제공한다. 마스트 호스트 설

치를 위해서는 Sun Grid Engine프로그램을 다운받아 압축을 풀은 다음 아래의 간단한 명령으로 가능하다.

./inst_sgee -m

또한 실행호스트의 설치는 아래의 명령과 같다.

./inst_sgee -x

이상의 명령으로 마스터 호스트와 실행호스트들을 설정할 수 있다.

이후의 설정은 qconf 라는 명령에 의해 환경설정이 가능하다.

⑤ 관리 및 제출 호스트 설정

마스터 호스트는 기본적으로 관리업무를 실행하고, 작업을 제출하며, 모니터 및 삭제 등의 작업을 지원한다. 하지만 사용자가 마스터 호스트 이외의 별도 호스트에서 이 같은 관리 및 제출 작업을 수행하려 한다면 별도의 설정이 필요하다.

관리호스트 추가

% qconf -ah <관리용 호스트 이름>

제출호스트 추가

% qconf -as <제출용 호스트 이름>

(바) SRS에서 Batch Queue System 셋팅

① SRS에 batch queue system 추가

SRS에서는 ICARUS에서 $BatchQueueSystem 을 정의하고 있다. 이에 상응하는 내용으로 객체를 생성한다.

해당 파일 : template $SRSDB/srsgen.i

setting file : $SRSSITE/srsgen.i

해당파일중에 Sun grid engine에 대한 셋팅부분을 찾아서 자신의 서버에 실행파일 위치가 맞도록 수정한다.

```
$SunGridEngine=$BatchQueueSystem:[
  name:'Sun Grid Engine'
  sub_command:
   |. /usr/local/sge/default/common/settings.sh ; . ./sge_cell.sh ; qsub  \
  interactive_sub_command:
   |. /usr/local/sge/default/common/settings.sh ; . ./sge_cell.sh ; qsh \
  del_command:
   |. /usr/local/sge/default/common/settings.sh ; . ./sge_cell.sh ; qdel \
  stat_command:
   |. /usr/local/sge/default/common/settings.sh ; . ./sge_cell.sh ; qstat
  acct_command:
   |. /usr/local/sge/default/common/settings.sh ; . ./sge_cell.sh ; qacct -j \
  #----------------
  #Option specifiers
  #----------------
```

```
  job_name_spec:N
  priority_spec:p
  queue_name_spec:q
  #----------------------------------------
  #Regular expressions for state identification
  #----------------------------------------
  runStateIndicatorRE:"( r | t )"
  waitStateIndicatorRE:"( w | h | S | s | T | qw )"
]
```

② Local Batch queue 셋팅

$BatchQueueSystem에 대한 셋팅을 마친 후 해당 서버에 맞는 환경정보를 셋팅해야 한다. 이것은 $SRSSITE/site.i에 $QueueConfig 객체를 만들면 된다.

```
$queueconfiguration=$QueueConfig:[
        batchQueueingSystem:$SunGridEngine
        indexingQueue:"myblast"
        srsServer:mysystem
        remoteHosts: {
                # List of remote systems, if any.
                $Host:[system2 execDir:"/tmp"]
        }
        applicationLaunchers: {
                $ApplLaunchers:[{ BlastN BlastP }
                queueNames:{"kisti.q" "ngic.q"}
```

```
    }
]
```

여기에 batchQueueingSystem 을 $SunGridEngine으로 한다.

③ 실행 script 셋팅

본 프로그램은 어떤 종류의 클러스터와도 연결이 가능하도록 설계되었다.

해당 서버에 명령을 실행하도록 셋팅된 파일은 $ SRSICA/appl.i이다.

여기에는 submit에 담당하는 부분과 launch를 담당하는 부분, indexing을 담당하는 부분에 해당하는 script들이 있다.

이중에서 $submitScript 부분은 작업 제출 시에 사용되는 부분으로 현재 국가유전체 정보센터 (NGIC)에서 한국 과학기술 정보연구원 (KISTI)의 클러스터를 사용할 수 있도록 script를 개발하였다.

```
|#!/bin/bash
|
|getDBInfo(){
|  COUNT=0
|  while read junk dbname; do
|    NAME=\$dbname
|
|    let COUNT=\$((\$COUNT+1))
|    if [ \$COUNT = 2 ]; then
```

```
|       DBNAME='echo \$NAME \| cut -d '/' -f6'
|       if [ \$DBNAME = est_human ]; then
|         server='cluster'
|         echo 'SGE_CELL=default' > sge_cell.sh
|         echo 'SPLITED_DB_NUM=5' >> sge_cell.sh
|         echo 'export SGE_CELL SPLITED_DB_NUM' >> sge_cell.sh
|       else
|         server='local'
|       fi
|     fi
|   done
|}
|getDBInfo < ./($p3)_command
|
|if [ \$server = cluster ]; then
|
|$p1 \
|-$p2 $p3 \
|$p8 \
|  -masterq master.q -S /usr/bin/perl -pe ngic \$SPLITED_DB_NUM
($p3)_launch.csh \
|($Alt:[$p6==0 t:" > ($p7).jobId"] )
|elif [ \$server = local ]; then
|echo "your job 111 ("pseudo.sh") has been submitted"
|./($p3)_command
```

|./($p3)_index.sh

|fi

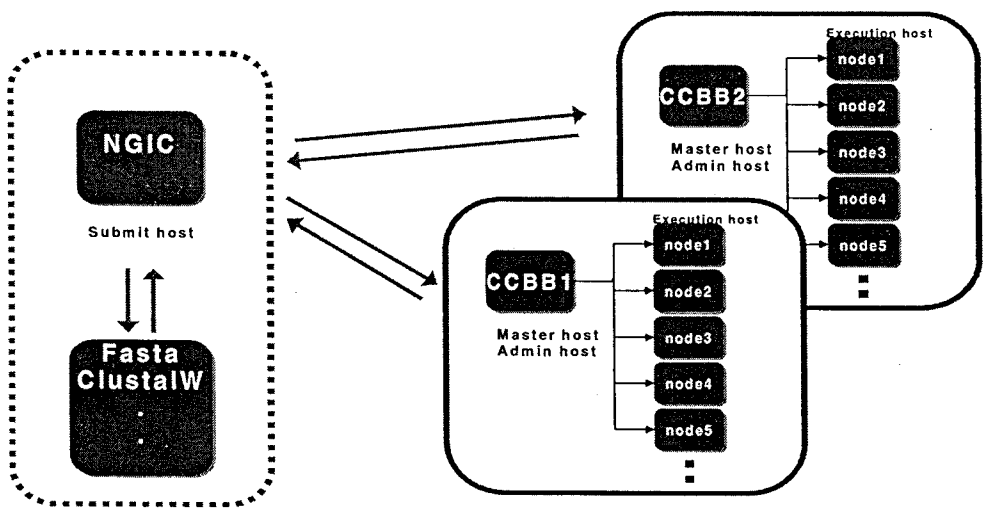
이 과정에서 단일 서버에서 실행하기 힘든 데이터베이스를 선별하여 KISTI의 클러스터로 작업을 의뢰할 수 있도록 프로그램을 수정하였다.

또한 이 과정에서 단순히 KISTI의 클러스터뿐 만 아니라 향후 추가되는 클러스터를 이용 할 수 있도록 하기 위하여 현재 프로그램 내에서 실행하고자 하는 클러스터를 변경할 수 있도록 하여 향후 확장성을 고려하였다 (그림 2).

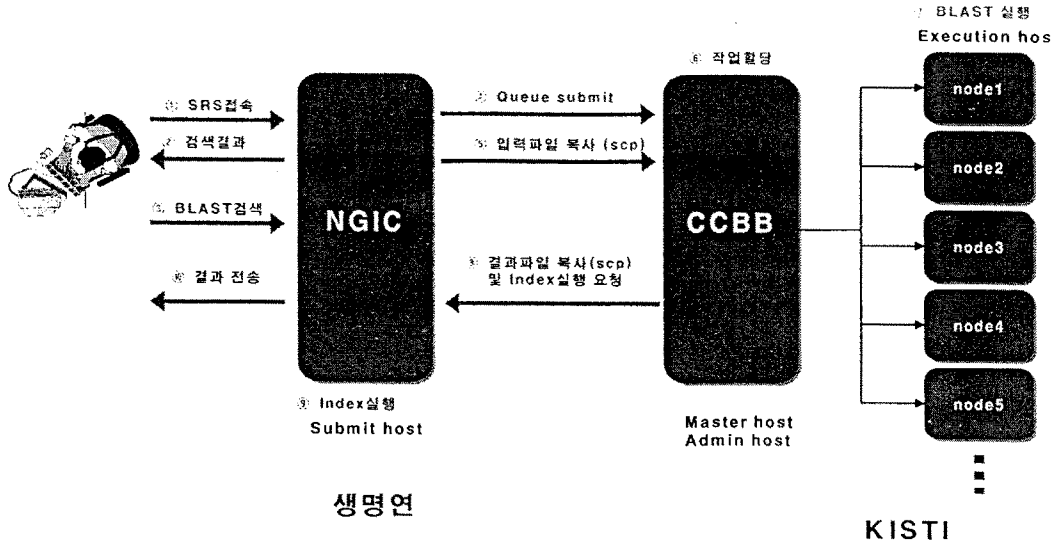
뿐만 아니라 현재 서비스와는 별도로 다른 생명정보 분석 프로그램을 설치할 것을 대비하여 어떠한 종류의 프로그램으로도 사용이 가능하도록 해당 command내에 응용프로그램을 정의할 수 있도록 하였다.

이 과정을 거치게 되면 SRS에서 KISTI의 클러스터로 작업을 의뢰하게 되면 접수된 작업들은 작업대기열을 거쳐 순서에 따라 실행된다. 실행된 결과는 보안을 고려하여 안전하게 NGIC의 서버로 내용이 전달된다.

위 사항은 아래 <figure 3-11>에 도식화하였다.



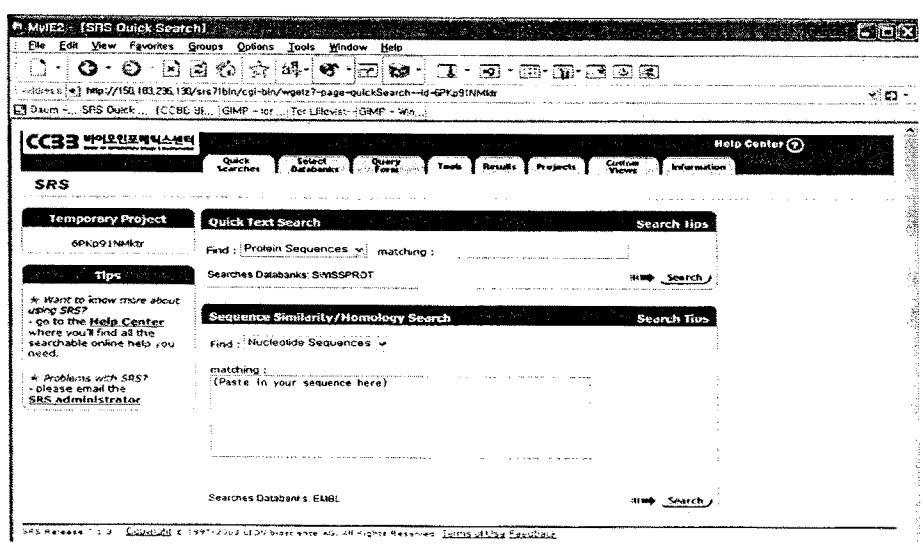
<figure 3-11>다중 클러스터이용이 가능한 확장성



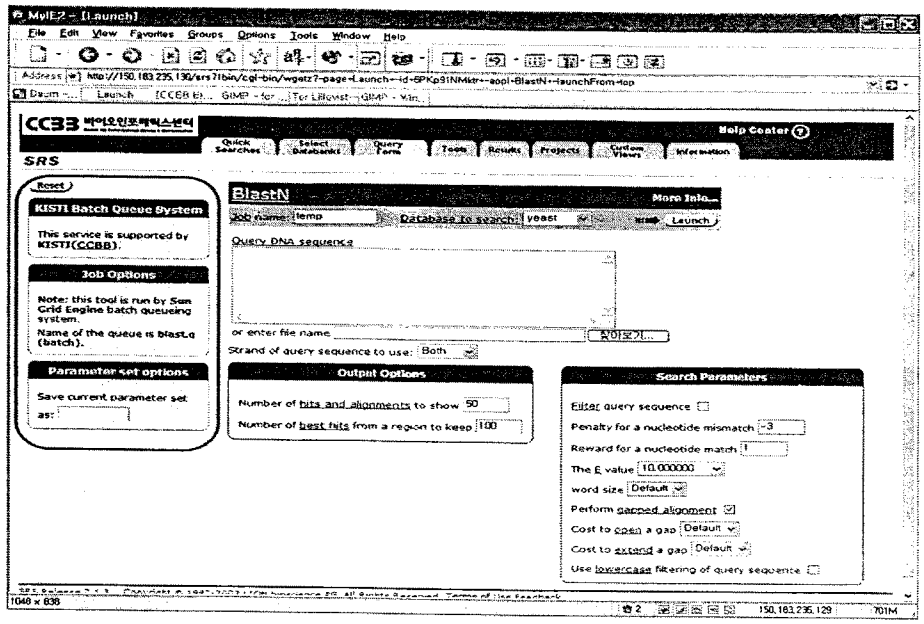
<figure 3-12> 작업 흐름도

작업의 흐름을 간략히 요약하면 사용자는 NGIC 홈페이지를 통해 SRS에 접속한다. 그리고 사용자가 궁금해 하는 내용을 검색한 후에 이를 Blast등 생명정보 분석 프로그램에 작업을 요청하게 된다. 요청된 내용은 NGIC에서 KISTI의 바이오인포메틱스센터(CCBB)의 마스터 호스트에 접수 되며, 접수된 사항은 대기열에 따라 각 실행 호스트에서 실행된다. 작업이 실행되면, 마스터 호스트는 NGIC에 작업에 필요한 입력 파일을 보안셸을 이용하여 복사하고, 이를 실행한다. 이 같은 실행결과는 다시 보안셸을 통해 NGIC로 이동하며, 인덱싱을 거쳐 사용자에게 결과를 전달하게 된다.

(사) SRS 실행 화면



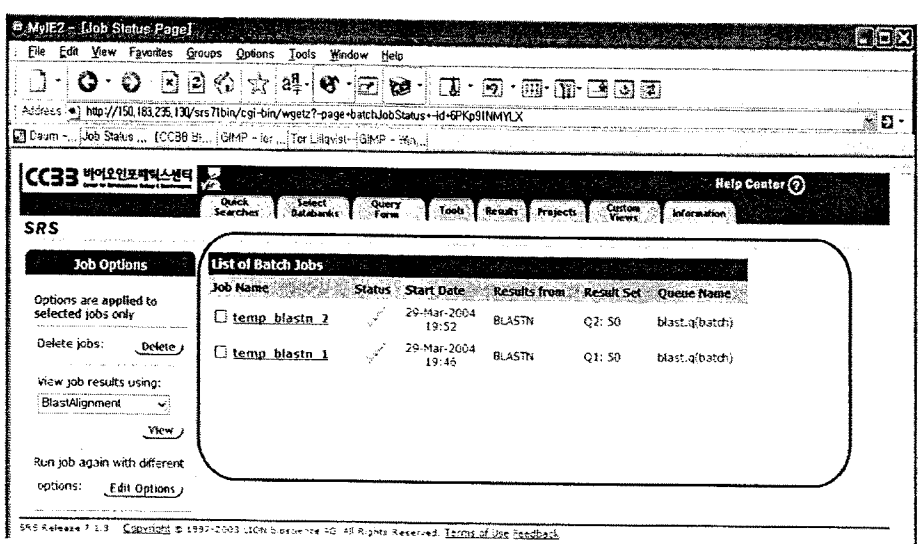
<figure 3-13> SRS의 첫화면



<figure 3-14> 작업 제출 화면

SRS를 실행하면(<figure 3-13>) 사용자는 다양한 상단 메뉴를 볼 수 있다 이중에 Tool을 선택하여 Blast를 선택하면 사용자의 작업 제출용 화면을(<figure 4.13>))을 볼 수 있다.

이곳에 자신이 요청하는 입력 내용 혹을 미리 준비한 입력파일을 첨부하여 작업을 요청하면 사용자는 자신이 요청한 작업의 상황 화면(<figure 3-15>)으로 이동하여 작업내용을 파악할 수 있다.

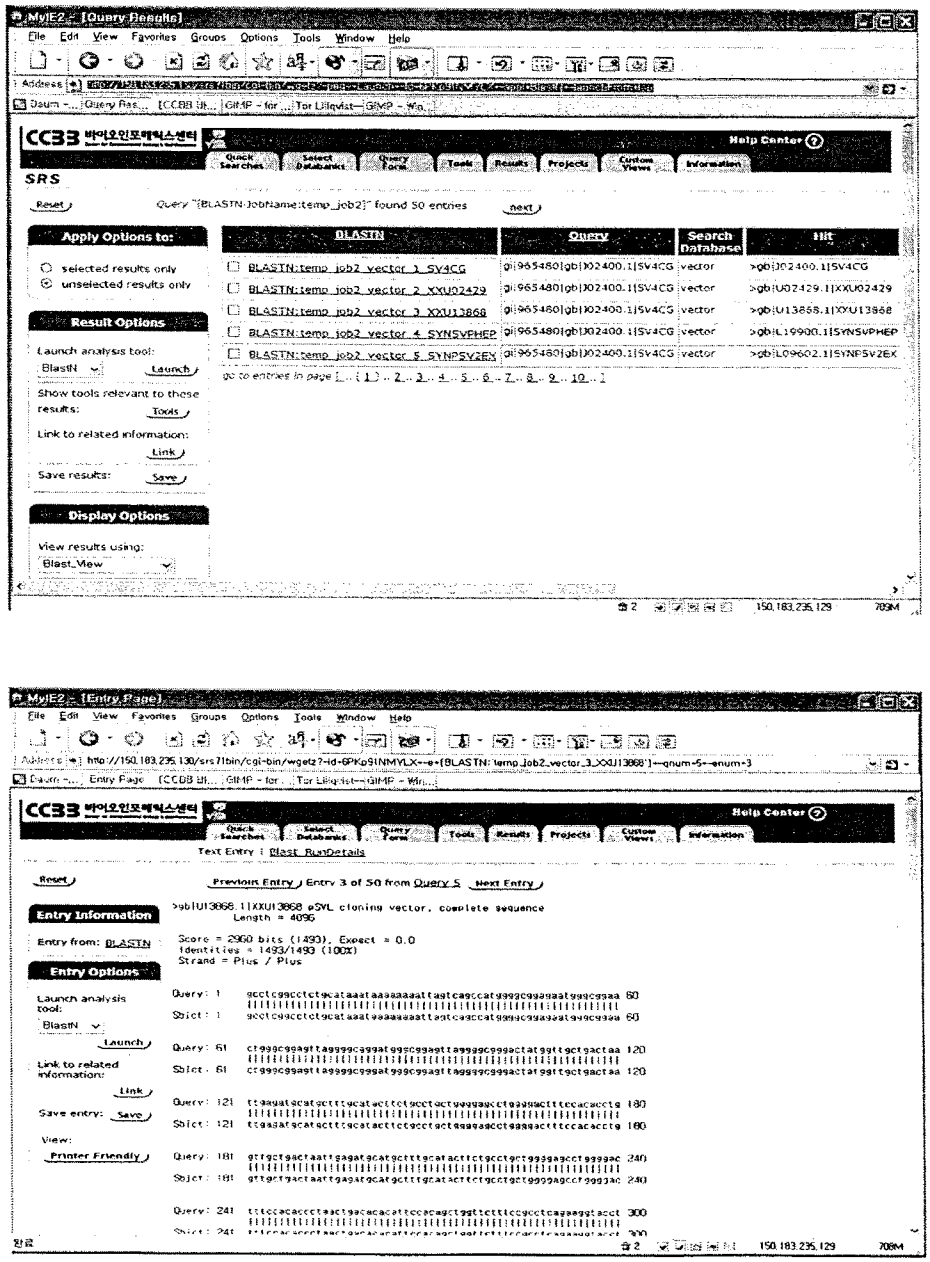


<figure 3-15> 작업 상황 조회 화면

여기서 현재 작업의 상황 그리고 결과를 확인해 볼 수 있으며, 이 같은 내용들은 상단의 모래시계아이콘을 클릭하면 계속적으로 최신 정보를 볼 수 있다.

또한 결과가 완료 된 경우와 실행중인 경우 그리고 실패한 경우를 아이콘을 통해 일목요연하게 파악할 수 있는 장점이 있다.

정상적으로 실행된 결과는 녹색 체크모양으로 표시되면 결과를 선택하면 세부적인 결과내용을 볼 수 있다 (<figure 3-16>).



<figure 3-16>결과 조회 내용

(아) mpiBLAST의 최적화

mpiBLAST는 BLAST프로그램을 병렬화한 것으로 Los Alamos National Laboratory에서 개발하여 배포하고 있다 (http://mpiblast.lanl.gov). 현재 intel계열(x86)의 리눅스 뿐만 아니라 AMD (x86_64) 에서도 사용이 가능하도록 개발되어졌다. 현재 1.2.1버전이 공개되었다.

본 과제에서는 구축된 클러스터를 기반으로 대용량의 생물정보 분석 서비스를 제공하기 위해 mpiBLAST를 각 클러스터에 설치하여 사용하고 있다.

BLAST를 병렬화 하기 위해서는 가장 필요한 부분이 바로 데이터베이스를 여러 개로 조각내어 각 노드에서 실행시키는 것으로 최적화를 위해 크기별로 데이터베이스를 조각내어 실행속도를 측정해 보았다.

① 데이터베이스 크기별 실행속도.

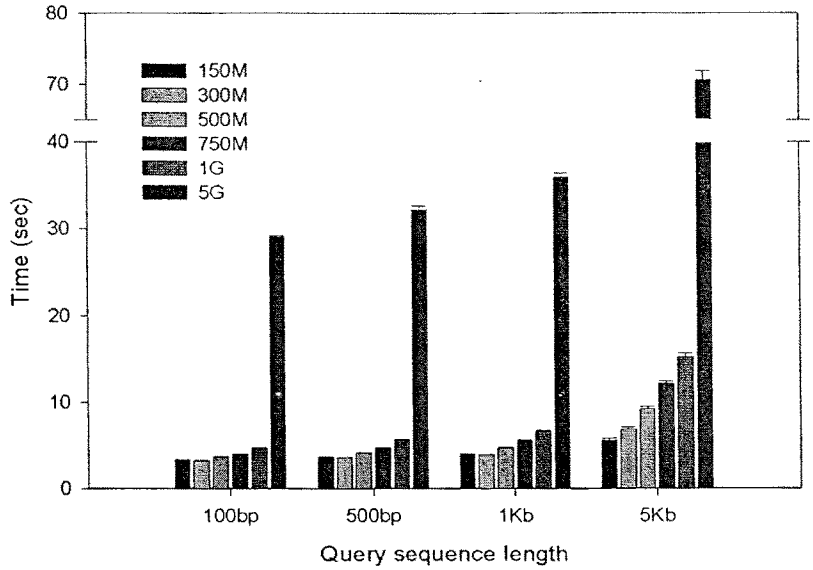
현재 가장 많이 사용하고 있는 데이터베이스인 NT를 이용하여 테스트에 이용하였다. NT의 전체 크기는 약 10Gb였으며, 이를 각각 150M, 300M, 500M, 750M, 1G, 5G로 조각내었다. 이때 사용한 프로그램은 mpiformatdb이다. 이것은 mpiBLAST에서 제공하고 있으며, 기본적으로 BLAST의 formatdb를 이용하여 인덱싱하고, 이를 원하는 크기로 나누어주는 역할을 한다.

② 실행결과 분석

실행 속도 테스트를 위해 입력염기서열을 각각 100bp, 500bp, 1000bp, 5000bp을 각기 50개씩 준비하였다. 준비된 염기서열은 각각 조각난 데이터베이스에 대해 실행 속도 테스트를 실시하였다. 실행시 LAM-MPI를 이용하여 mpi 통신을 수행할 수 있도록 하였으며, 실행 시 각각의 조각난 데이터베이스만큼의 실행 노드를 부여하였다 (<figure 3-17>).

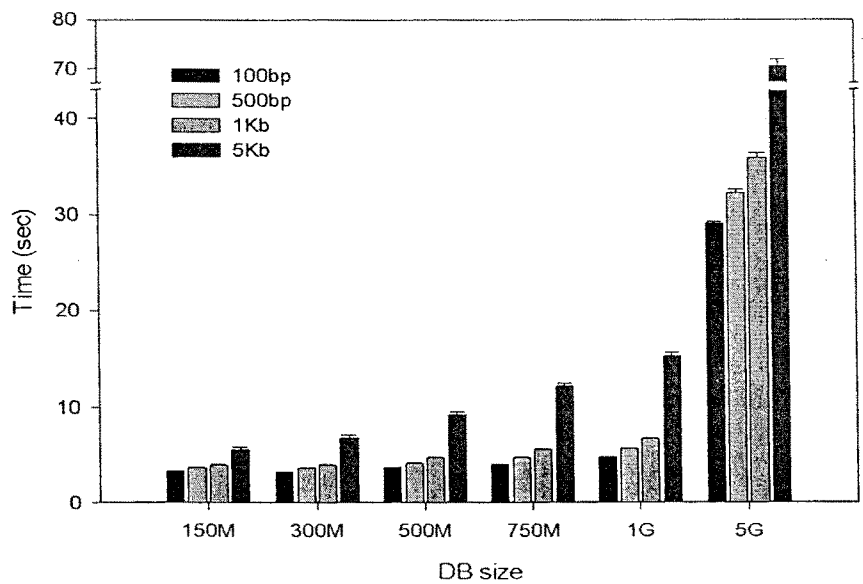
먼저 입력염기서열의 크기별 실행속도를 살펴보면, 입력크기가 증가할수록 실행 속도가 증가함을 관찰할 수 있으며, 특히 1000bp에서 5000bp로 증가했을 때 가장 많은 변화를 보였다. 이는 BLAST의 알고리즘 상 입력 크기에 의해 데이터베이스를 검색하는 검색횟수가 증가함에 의해 발생하는 것이다 (<figure 3-18>).

일반적으로 입력염기서열의 크기는 500bp에서 1000bp가 많으므로 데이터베이스의 크기가 5G를 제외하고는 모두 10초 이내의 실행시간을 관찰 할 수 있었다.



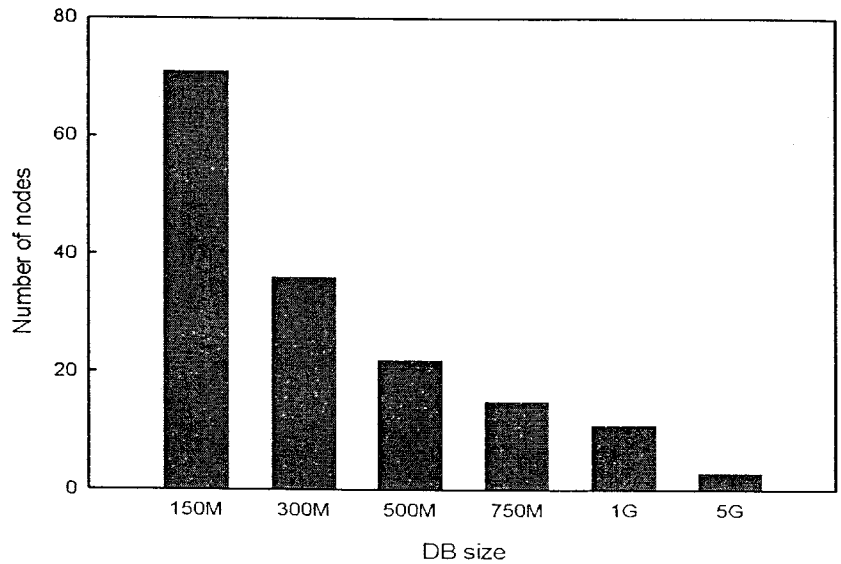
<figure 3-17> 입력염기서열 크기별 mpiBLAST 실행시간

이번에는 입력염기서열의 길이가 아닌 각 노드의 데이터베이스의 크기에 따른 실행 속도를 관찰 해 보았다. 그 결과 1G byte에서 5G byte 사이에서 급격한 속도 저하가 발생함을 확인 할 수 있었으며, 이 같은 이유는 각 노드의 메모리 크기가 1G byte로 메모리의 한계를 벗어나 스웹을 사용함에 따라 하드디스크에 입/출력시간이 증가함에 기인하는 것이다. (<figure 3-18>)



<figure 3-18> 데이터베이스 크기별 mpiBLAST 실행시간

특히 입력염기서열의 크기가 5kb를 제외 한다면 150M에서 1G까지 거의 동일한 실행 시간을 나타내고 있었다. 이는 각 노드에서 작업이 실행될 때 기본적으로 작업을 할당하는 MPI통신에서 발생하는 시간이 기본적으로 거의 동일하며, 대부분을 차지하는 것으로 판단된다.



<figure 3-19>조각난 데이터베이스크기에 따른 실행 노드의 개수

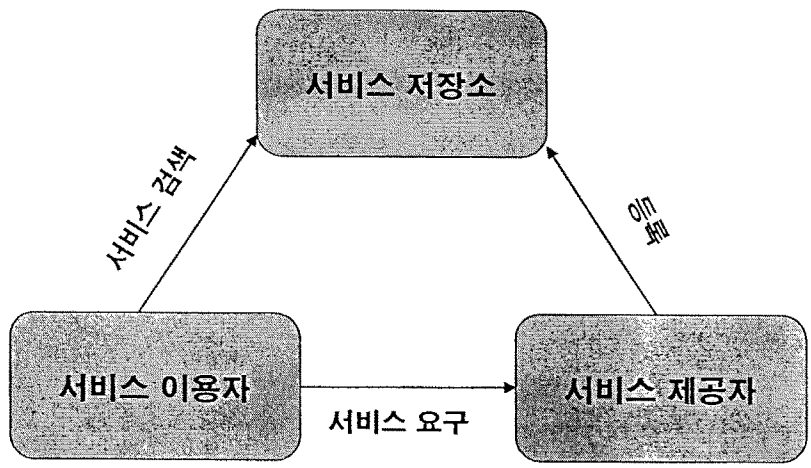
위의 결과를 바탕으로 KISTI 클러스터 시스템에서 서비스할 mpiBLAST의 데이터베이스 크기는 750Mb로 조각내었으며, 16개의 노드에서 수행할 수 있도록 하였다. 이같이 750Mb로 결정한 이유는 1Kb 미만의 일반적인 입력요청염기서열에 대해 10초 미만의 응답속도를 가지면서 실행 가능한 클러스터의 노드 숫자를 최소화 하려고 했기 때문이다. 이렇게 노드숫자를 줄임으로 인해 한번에 약 5개까지 작업까지 동시에 수행할 수 있기 때문이다.

(자) BLAST WebServices

웹서비스는 네트워크 상에서 접근 가능한 소프트웨어 기능 단위로, 플랫폼, 프로그래밍 언어 및 컴포넌트 모델에 독립적인 기술로 만들어진 소프트웨어를 말한다. 웹서비스는 SOA(Service-Oriented Architecture)에 기반을 두고 있으며, SOA에서는 소프트웨어의 기능이 서비스의 집합으로 분류된다. SOA영역 내에 서비스가 상주하려면 서비스를 기술, 검색, 호출하는데 사용될 공통 메커니즘이 필요하다.

아래 <figure 3-20>은 SOA내에서의 역학과 역할간 상화작용을 나타내고 있다.

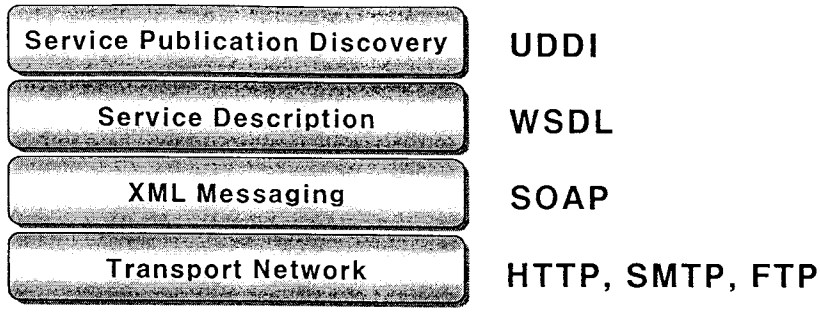
즉 서비스 제공자는 웹서비스를 수행할 기능을 구현하고, 이 기능에 대한 인터페이스를 표준에 맞추어 기술한다. 또한 이 인터페이스를 레지스트리에 공개함으로서 소비자가 웹서비스를 찾아갈 수 있게 한다. 또한 서비스 레지스트리는 이러한 웹서비스의 창고역할을 하는 곳으로, 웹서비스 제공자가 등록한 내용을 사용자가 찾아서 해당서비스를 사용할 수 있도록 저장소 역할을 한다. 서비스 소비자는 서비스 레지스트리에서 원하는 서비스를 검색하며, 웹서비스 제공자가 공개한 인터페이스를 통해 서비스를 사용한다.



<figure 3-20> SOA(Service-Oriented Architecture)의 역할과 상호작용

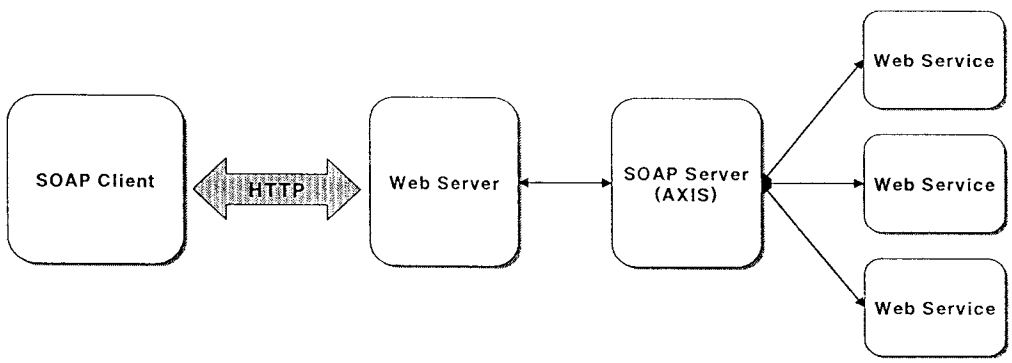
이상의 정의는 SOA의 추상화 모델이다. 이런 모델을 운용 가능한 웹서비스 스택은 IBM이나 마이크로소프트 같은 몇몇 회사에서 정의했다. 이런 웹서비스 스택은 특정 플랫폼이나 제조사에 의존하지 않는 상호 운용성이 필수적이다.

최하위 계층은 전송계층으로 종단점간의 통신을 담당한다. 웹서비스 기술은 HTTP와 같이 널리 사용되고 있는 전송 메커니즘을 사용함으로써 기존 네트워크를 충분히 활용할 수 있다. 한 단계 위의 메시지계층은 서비스 소비자에 의한 웹서비스 호출을 담당한다. 이 계층에서 서비스 소비자는 SOAP메시지를 웹서비스에 의해 공개된 메소드를 호출하는데 사용하고, 서비스 제공자는 WSDL(Web Services Description Language)을 사용해서 웹서비스 인터페이스를 표준 방식으로 기술한다. 최상위 계층은 탐색계층이며, UDDI(Universal Description and Discovery Interface)로 구현돼 있다.



<figure 3-21> 웹서비스 스택과 관련기술

SOAP클라이언트는 HTTP를 통해 웹서비스로 SOAP메시지를 전송한다. 웹 서버는 메시지를 받아서 그 처리를 SOAP서버에게 위임하며, SOAP서버는 맞는 웹서비스를 차례로 호출한다. 본 과제에서 사용한 AXIS는 SOAP의 서버의 한 종류이며, SOAP 클라이언트의 호출을 받아서 SOAP메시지의 형태와 그 응답을 전송하는 역할을 수행한다.



<figure 3-22> 기본 SOAP시스템

① SOAP 메시지 설계

BLAST Web Services를 위해 사용자가 BLAST실행을 위해 요청하는 SOAP 메세지는 XML형태의 문서이다. 아래는 사용자가 신청하는 SOAP 메세지의 일부이다.

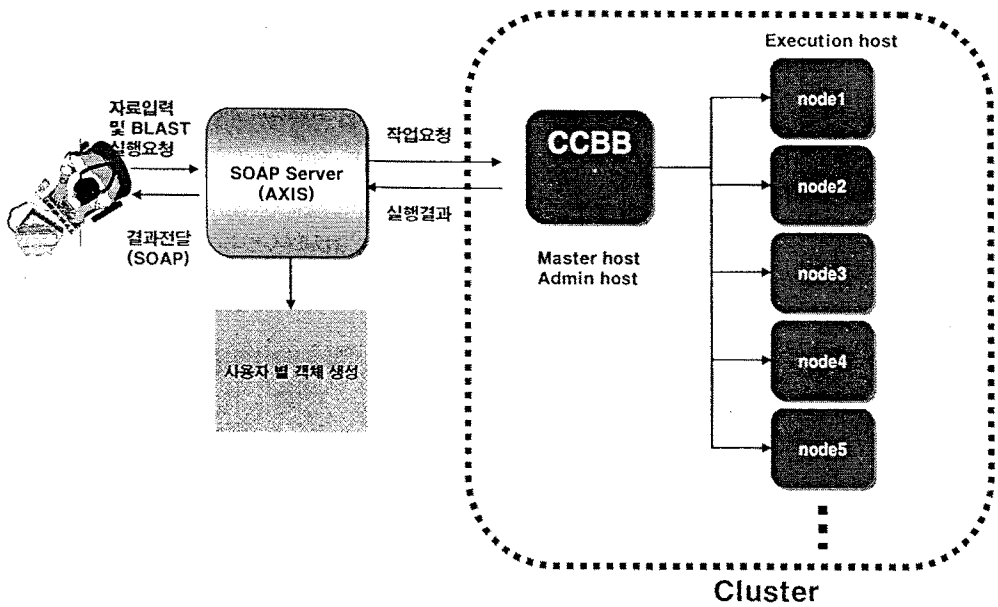
사용자는 Body내에 자신의 요청하는 작업의 정보를 포함하여 서버에 내용을 제출하게 된다.

② BLAST 실행을 위한 서버 클래스 설계

사용자로부터 요청을 받은 웹서비스 서버는 사용자의 요청 정보를 분석하게 된다. 이를 위해 QueryHander클래스와 Queue클래스가 사용되어진다. 분석된 정보를 바탕으로 BLAST실행을 담당하는 BlastHander 클래스가 있다. 이 클래스는 Sun Grid Engine으로

작업요청을 실행하며, 이후 작업의 접수를 확인하고 해당하는 작업접수 번호를 사용자에게 전달해 준다.

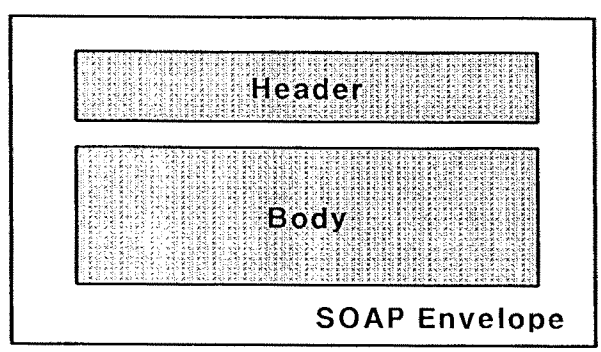
사용자는 자신의 접수한 작업접수번호를 바탕으로 작업의 수행 과정을 파악할 수 있다. 작업이 완료되면 사용자는 결과 파일의 전송을 서버에 요청하며, 실행된 결과는 SOAP에 포장되어 XML형태로 사용자에게 전달된다.



<figure 3-23> 작업 흐름도

③ 작업 요청 및 실행 후 결과 메시지

작업 요청과 실행 결과는 SOAP에 포장되어 아래와 같은 형태로 제공된다. 사용자는 body내의 작업실행 결과를 XML형태로 받아볼 수 있다.



<figure 3-24> SOAP내용

④ WSDL

BLAST Web Services의 WSDL(Web Services Description Language)은 웹서비스를 기술하는 문법이다. 즉 사용자가 어떻게 이 서비스를 이용할 수 있는지 설명하는 매뉴얼이라고 할 수 있다. 이 안에는 웹서비스가 수행하는 작업, 호출가능한 메소드, 메소드에 전달해야 하는 파라미터 및 타입, 사용되는 바인팅 프로토콜등이 포함되어있다.

아래 <figure 3-25>는 BLAST의 Web Services를 WSDL로 나타낸 것이다.

본 서비스는 메시지 방식을 취하고 있으며, 크게 runBlastResponse, runBlastRequest로 나뉘어진다.

```
<?xml version="1.0" encoding="UTF-8" ?>
<wsdl:definitions targetNamespace="http://localhost:8080/axis/services/WSBlast"
xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:apachesoap="http://xml.apache.org/xml-soap"
xmlns:impl="http://localhost:8080/axis/services/WSBlast"
xmlns:intf="http://localhost:8080/axis/services/WSBlast"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:tns1="http://WSBlast" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<wsdl:types>
<schema targetNamespace="http://WSBlast"
xmlns="http://www.w3.org/2001/XMLSchema">
<element name="runBlast" type="xsd:anyType" />
</schema>
<schema targetNamespace="http://localhost:8080/axis/services/WSBlast"
xmlns="http://www.w3.org/2001/XMLSchema">
<element name="runBlastReturn" type="xsd:anyType" />
</schema>
</wsdl:types>
<wsdl:message name="runBlastResponse">
<wsdl:part element="impl:runBlastReturn" name="runBlastReturn" />
</wsdl:message>
<wsdl:message name="runBlastRequest">
```

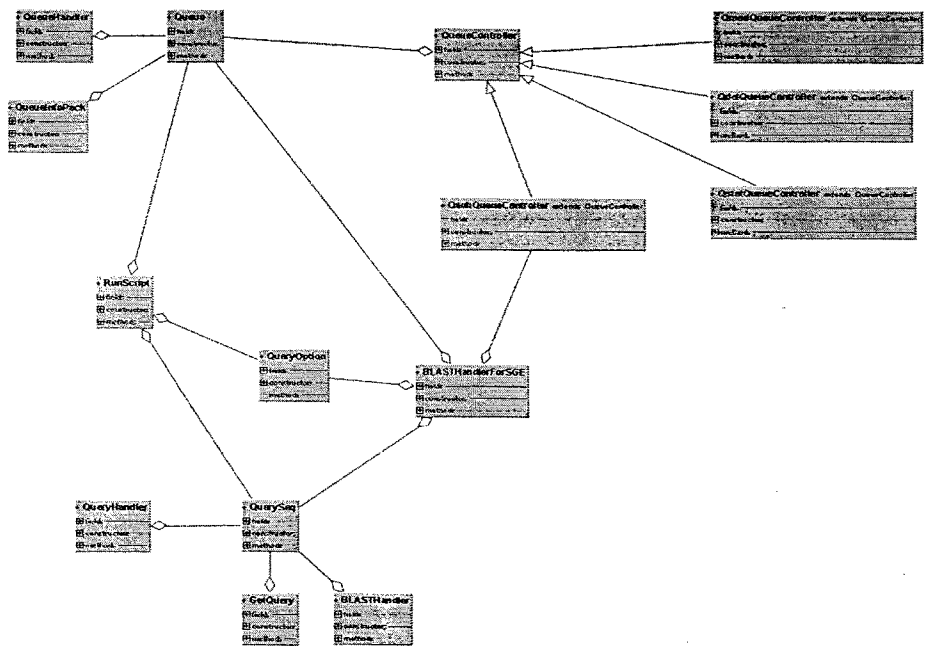
```
<wsdl:part element="tns1:runBlast" name="part" />
</wsdl:message>
<wsdl:portType name="BLASTServices">
<wsdl:operation name="runBlast">
<wsdl:input message="impl:runBlastRequest" name="runBlastRequest" />
<wsdl:output message="impl:runBlastResponse" name="runBlastResponse" />
</wsdl:operation>
</wsdl:portType>
<wsdl:binding name="WSBlastSoapBinding" type="impl:BLASTServices">
<wsdlsoap:binding style="document'
transport="http://schemas.xmlsoap.org/soap/http" />
<wsdl:operation name="runBlast">
<wsdlsoap:operation soapAction="" />
<wsdl:input name="runBlastRequest">
<wsdlsoap:body namespace="http://WSBlast" use="literal" />
</wsdl:input>
<wsdl:output name="runBlastResponse">
<wsdlsoap:body namespace="http://localhost:8080/axis/services/WSBlast
use="literal" />
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="BLASTServicesService">
<wsdl:port binding="impl:WSBlastSoapBinding" name="WSBlast">
<wsdlsoap:address location="http://localhost:8080/axis/services/WSBlast" />
</wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

<figure 3-25>. BLAST Web Services의 WSDL

⑤ 개발클래스

Web Services를 위해 사용자가 요청한 정보를 분석하는 QueryHandler클래스와 해당 정보를 저장하는 QuerySeq, 그리고 Blast를 실행하는 BlastHandlerForSGE 및 QsubQueueController가 주된 역할을 수행하며, 그 밖에 작업처리상황을 지니고 있는 Queue클래스, QueueHandler 그리고 QstatQueueController클래스는 사용자가 진행상황을 요청 시 처리하는 클래스들이다.

그 밖에 실행스크립트생성에 필요한 QueryOption클래스와 RunScript가 있다.



<figure 3-26> 클래스 구성도

⑥ 향후 전망

SOAP서비스는 앞으로 웹서비스의 대안으로 다양한 정보를 전달할 수 있을 뿐만 아니라 사용자가 원하는 정보들을 연계하여 분석할 수 있는 한차원 높은 서비스로 계속 발전하고 있으며, 생명정보학 분야에서도 이 같은 웹서비스 기술을 이용한 다양한 서비스가 개발되어 질것으로 예상된다. 본 과제에서 수행한 BLAST 웹서비스는 다양한 사용자에게 KISTI의 클러스터를 웹서비스를 통해 사용할 수 있는 기회를 제공할 수 있을 것이며, 대량은 분석작업을 위해 사용자가 SOAP을 이용하여 다양한 플랫폼에서 이용가능하리라 예상된다.

나. 연구결과 활용 및 기대효과

일반 연구자들이 값비싼 대단위 클러스터컴퓨터를 사용할 수 있게 되었다. SRS와 대규모 클러스터 컴퓨터를 연결함으로 인해 생명연과 KISTI간 업무의 연계가 가능하게 되었으며, 생명공학 연구자들이 생명연의 SRS을 통해 KISTI의 대단위 클러스터컴퓨터를 사용할 수 있게 되었다. 이 연구를 기반으로 앞으로 대단위의 분석시스템에 대한 병렬화 작업의 모델이 될 수 있을 것으로 기대된다.

특히 SRS은 앞으로 확장 가능한 플랫폼으로 다양한 정보의 추가 및 분석이 가능할 수 있는 시스템이므로, 일반 사용자들도 본 시스템을 이용하여 다양한 종류의 대단위의 분석작업을 수행할 수 있을 것으로 기대된다.

4. 클러스터시스템 관리시스템 구축

가. 연구의 중요성

Sun Grid Engine은 앞으로 많은 클러스터 시스템에서 사용되어질 그리드 컴퓨터시스템용 대기열 관리시스템으로 각광 받고 있다. 하지만 시스템 운영과는 달리 운영되는 상황파악과 실행 통계 조사 등의 기능은 지원하고 있지 못한 실정이다.

더욱이 다량의 사용자와 복잡한 작업이 지속적으로 수행되게 되면, 이 같은 운영 상황의 파악이 중요한 문제로 대두될 것으로 생각되며, 이를 위한 관리용 프로그램이 필요한 실정이다.

하지만 관리용 모니터링 프로그램은 현재 개발되어져 있는 것이 없는 상황으로 이러한 기능을 수행할 수 있는 프로그램개발은 본 과제 뿐 만 아니라 다른 시스템 운영자들에게 많은 도움을 줄 수 있을 것이다.

나. 연구 내용

(1) 웹기반 Sun Grid Engine모니터링용 프로그램 설계.

Sun Grid Engine은 작업대기열에 대한 실행 결과를 파일로 관리 하고 있다. 이 파일은 $SGE_ROOT/default/common/accounting이란 파일로 저장되며, 이 안에는 작업의 이름, 사용자, 작업번호, 계정, 접수시간, 실행 시작시간, 종료시간 등 다양한 정보를 저장하고 있다.

본 프로그램에서는 이와 같이 Sun Grid Engine에서 저장하는 파일을 이용하고 별도의 데이터베이스를 이용하지 않았다. 이는 파일로 저장되는 결과들을 데이터베이스로 전환 하는 작업이 불필요한 부분이며, 파일의 분석만으로 충분히 결과 통계처리 및 검색이 가능 할 수 있기 때문이다.

결과 파일의 분석작업은 웹기반으로 수행하기 위해 PHP라는 언어를 사용하여 개발하였으며, 객제지향형으로 개발하기 위해 총 6개의 객체로 구성되도록 설계하였다. (<table 3-2>)

특히 파일의 정보를 객체화 하여 별도의 데이터베이스를 만들지 않고 검색할 수 있도록 설계하였다.

또한 결과파일을 객체화하고 검색된 내용을 분석하여 Bar그래프로 표시할 수 있도록 별도의 그래프생성용 객체를 만들어 사용하였다.

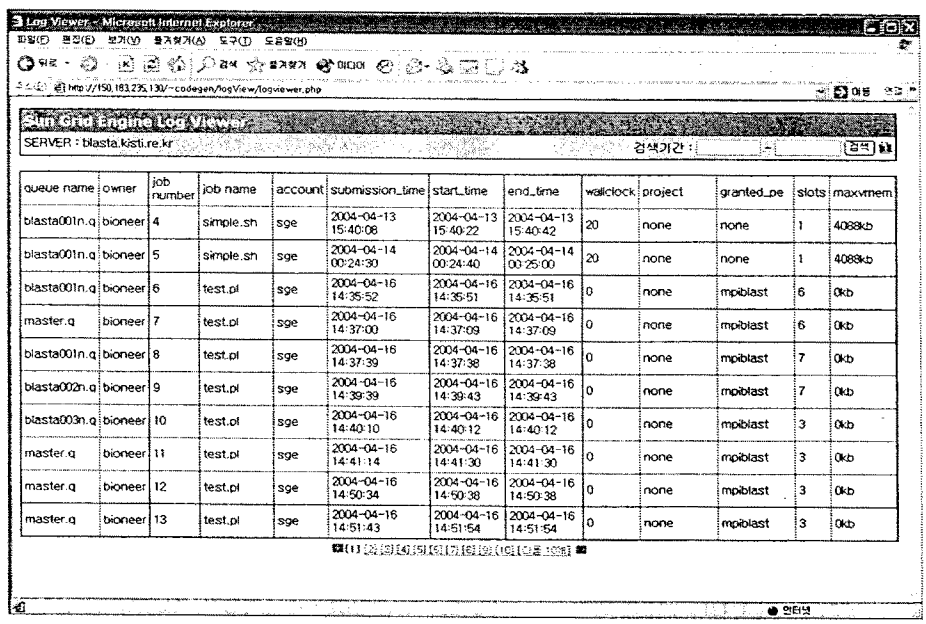
<table 3-2> 구성 객체 목록

Class 이름	기능	비고
Log	1개의 실행단위의 기록에 대한 정보를 저장하는 객체	accounting 파일의 정보자료
SearchOptions	실행내용 검색시 참조하는 객체	
Statistic	통계처리 수행용 객체	
ViewerControl	사용자에게 결과를 출력시 출력내용을 관리하는 객체	
BarGraph	Bar 차트 생성 및 출력에 사용하는 객체	
File	Sun Grid Engine에서 출력한 결과를 불러들여 관리하는 객체	Log 객체에서 사용

(2) 웹 기반 프로그램의 개발 결과

설계된 Class를 기반으로 실행된 작업내용을 출력하는 테이블 형태의 로그내용을 볼 수 있는 웹 페이지를 제작하였다 (<figure 3-27>). 이는 실행 내용의 요약 및 필요한 경우 작업기간을 검색 할 수 있도록 하였으며, 보고 있는 결과내용을 Bar그래프로 나타날 수 있도록 링크를 설정하였다.

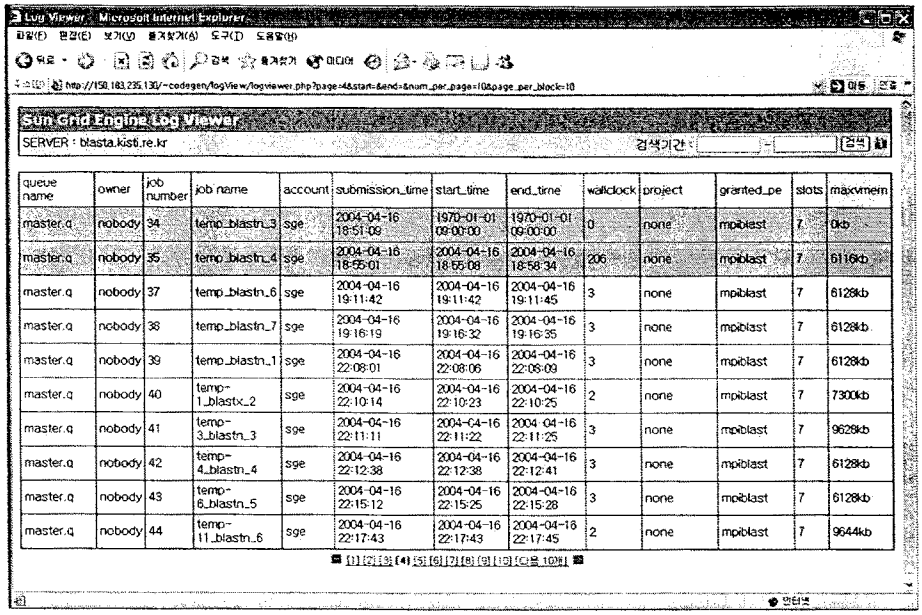
특히 사용자의 편의를 도모하기 위해 결과 내용은 10개 단위로 잘라 화면에 나타나며, 필요에 따라 사용자가 한 페이지 당 볼 수 있는 로그 정보를 늘릴 수 있도록 하였다.



queue name	owner	job number	job name	account	submission_time	start_time	end_time	wallclock	project	granted_pe	slots	maxvmem
blasta001n.q	bioneer	4	simple.sh	sge	2004-04-13 15:40:08	2004-04-13 15:40:22	2004-04-13 15:40:42	20	none	none	1	4088kb
blasta001n.q	bioneer	5	simple.sh	sge	2004-04-14 00:24:30	2004-04-14 00:24:40	2004-04-14 00:25:00	20	none	none	1	4088kb
blasta001n.q	bioneer	6	test.pl	sge	2004-04-16 14:35:52	2004-04-16 14:35:51	2004-04-16 14:35:51	0	none	mpiblast	6	0kb
master.q	bioneer	7	test.pl	sge	2004-04-16 14:37:00	2004-04-16 14:37:09	2004-04-16 14:37:09	0	none	mpiblast	6	0kb
blasta001n.q	bioneer	8	test.pl	sge	2004-04-16 14:37:39	2004-04-16 14:37:38	2004-04-16 14:37:38	0	none	mpiblast	7	0kb
blasta002n.q	bioneer	9	test.pl	sge	2004-04-16 14:39:39	2004-04-16 14:39:43	2004-04-16 14:39:43	0	none	mpiblast	7	0kb
blasta003n.q	bioneer	10	test.pl	sge	2004-04-16 14:40:10	2004-04-16 14:40:12	2004-04-16 14:40:12	0	none	mpiblast	3	0kb
master.q	bioneer	11	test.pl	sge	2004-04-16 14:41:14	2004-04-16 14:41:30	2004-04-16 14:41:30	0	none	mpiblast	3	0kb
master.q	bioneer	12	test.pl	sge	2004-04-16 14:50:34	2004-04-16 14:50:38	2004-04-16 14:50:38	0	none	mpiblast	3	0kb
master.q	bioneer	13	test.pl	sge	2004-04-16 14:51:43	2004-04-16 14:51:54	2004-04-16 14:51:54	0	none	mpiblast	3	0kb

<figure 3-27> 로그 결과 보기 화면

특히 사용자가 직관적으로 사용내역의 문제점을 파악하기 위해 에러가 발생한 부분을 붉은색으로 나타내어 에러의 원인을 손쉽게 파악할 수 있도록 하였다. (<figure 3-28>)



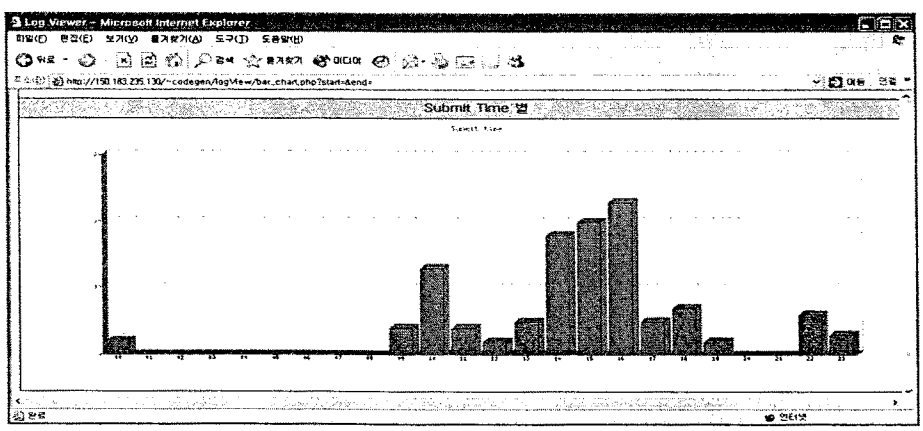
queue name	owner	job number	job name	account	submission_time	start_time	end_time	wallclock	project	granted_pe	slots	maxvmem
master.q	nobody	34	temp_blastn_3	sge	2004-04-16 18:51:09	1970-01-01 09:00:00	1970-01-01 09:00:00	0	none	mpiblast	7	0kb
master.q	nobody	35	temp_blastn_4	sge	2004-04-16 18:55:01	2004-04-16 18:55:08	2004-04-16 18:58:34	206	none	mpiblast	7	6116kb
master.q	nobody	37	temp_blastn_6	sge	2004-04-16 19:11:42	2004-04-16 19:11:42	2004-04-16 19:11:45	3	none	mpiblast	7	6128kb
master.q	nobody	38	temp_blastn_7	sge	2004-04-16 19:16:19	2004-04-16 19:16:32	2004-04-16 19:16:35	3	none	mpiblast	7	6128kb
master.q	nobody	39	temp_blastn_1	sge	2004-04-16 22:08:01	2004-04-16 22:08:06	2004-04-16 22:08:09	3	none	mpiblast	7	6128kb
master.q	nobody	40	temp-1_blastx_2	sge	2004-04-16 22:10:14	2004-04-16 22:10:23	2004-04-16 22:10:25	2	none	mpiblast	7	7300kb
master.q	nobody	41	temp-3_blastn_3	sge	2004-04-16 22:11:11	2004-04-16 22:11:22	2004-04-16 22:11:25	3	none	mpiblast	7	9628kb
master.q	nobody	42	temp-4_blastn_4	sge	2004-04-16 22:12:38	2004-04-16 22:12:38	2004-04-16 22:12:41	3	none	mpiblast	7	6128kb
master.q	nobody	43	temp-6_blastn_5	sge	2004-04-16 22:15:12	2004-04-16 22:15:25	2004-04-16 22:15:28	3	none	mpiblast	7	6128kb
master.q	nobody	44	temp-11_blastn_6	sge	2004-04-16 22:17:43	2004-04-16 22:17:43	2004-04-16 22:17:45	2	none	mpiblast	7	9644kb

<figure 3-28> 작업 에러에 대한 표시

(3) 통계 내역에 대한 Bar그래프.

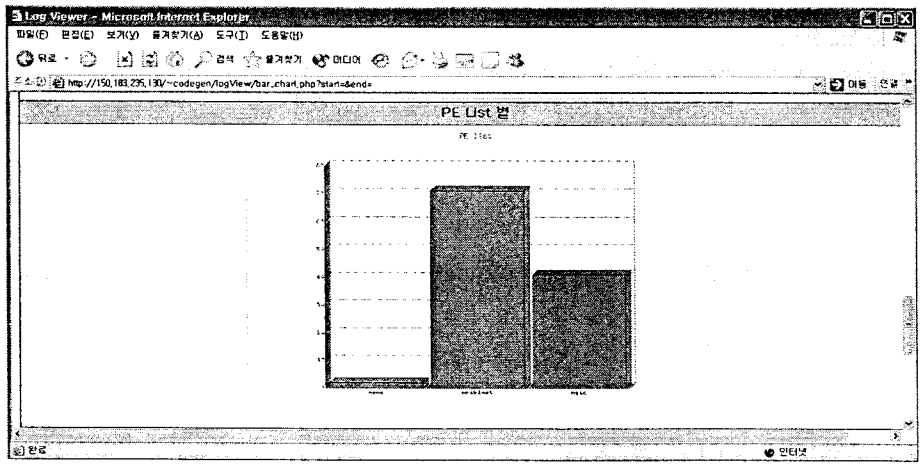
그림11에서 검색된 내역에 대한 통계상황을 파악하기 위해 Bar그래프를 사용하여 사용자가 직관적으로 이용내용을 파악할 수 있도록 개발하였다. 이를 위해 분석 빈도가 높은 항목 순으로 정리하여 작업 내용을 출력 하도록 하였다.

먼저 작업 제출시간에 대한 분석화면을 제공 하고 있다 (<figure 3-29>). 이는 사용자들이 작업의 재출 시간에 패턴을 분석하여 작업대기열의 축적 상황을 분석하고 이에 적당한 대기열 개수를 관리자가 변경할 수 있는 데이터로 활용 할 수 있다.



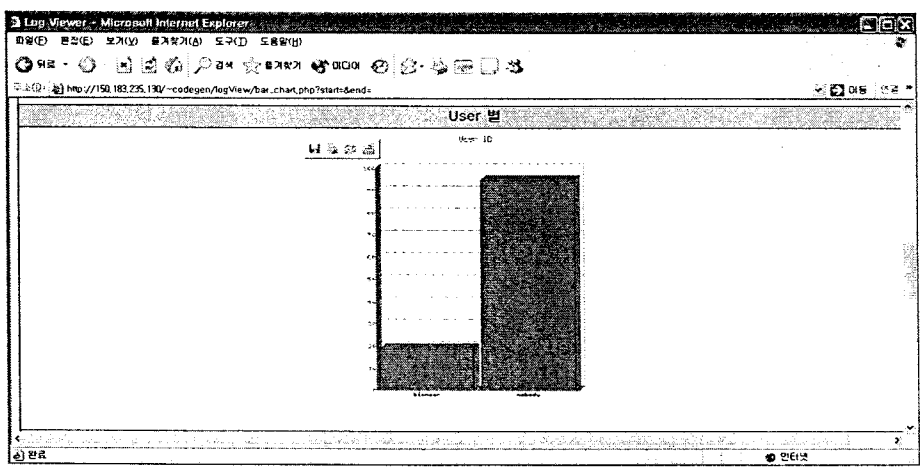
<figure 3-29> 제출시간별 작업누적 상황

또한 작업의 요청이 어떤 종류의 parallel environment(PE)를 사용하지는 파악하기 위해 Sun Grid Engine에 설정된 PE 목록에 따른 제출 작업요청 수를 확인 할 수 있도록 하였다. (<figure 3-30>)



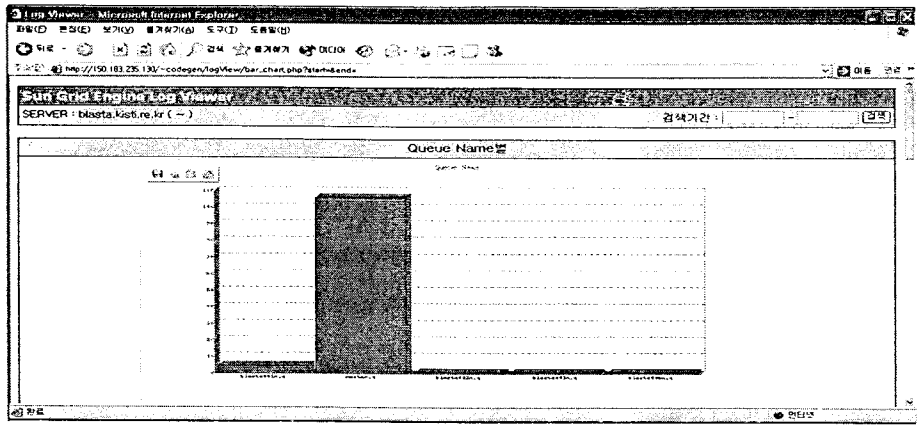
<figure 3-30> parallel environment별 작업 요청상황

특히 어떤 사용자의 요청이 많은지 파악할 수 있도록 하기위해 사용자별 작업요구량을 분석 할 수 있게 하였다. 이는 작업자의 사용량에 따라 앞으로의 운영 정책을 설정 시 많은 도움이 될 수 있을 것이다 (<figure 3-31>)



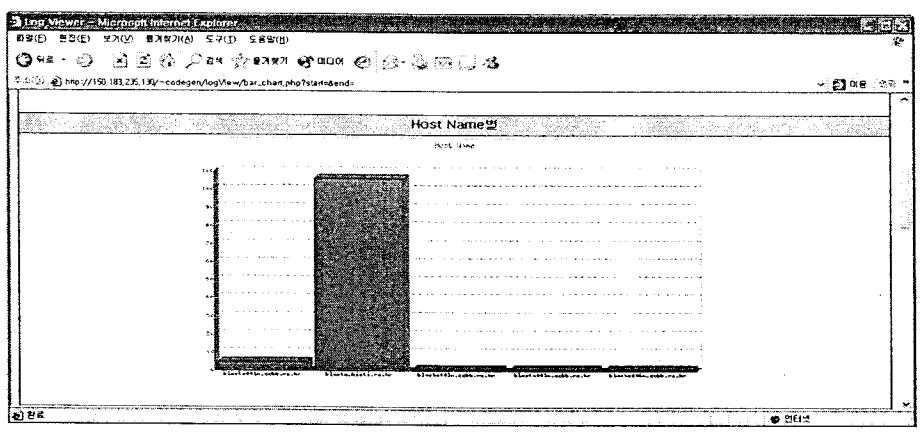
<figure 3-31> 사용자별 작업 요청 상황

다음으로 Queue 이름별로 분석하여 자주 사용되는 작업대기열의 상황을 파악할 수 있도록 하였다. (<figure 3-32>)



<figure 3-32> Queue 이름별 작업 상황

또한 작업을 요청한 컴퓨터별로 분석 할 수 있도록 하였다. 이는 사용자뿐만 아니라 작업을 요청하는 컴퓨터를 파악함으로 네트워크의 사용빈도를 종합적으로 분석할 수 있는데 도움을 줄 수 있을 것이다 (<figure 3-33>).



<figure 3-33> 작업 요청 컴퓨터별 상황

다. 연구결과 활용 및 기대효과

본 시스템을 이용함으로 인해 Sun Grid Engine을 운용함에 있어 작업 및 에러의 상황 작업의 요청빈도 등을 웹상으로 종합적으로 관리 분석 할 수 있을 것으로 기대된다.

특히 이 같은 웹 기반의 분석 프로그램은 아직까지 제작된 적이 없으므로 본 과제를 통해 다양한 분야에서 본 프로그램을 응용할 수 있을 것이며, 이를 기반으로 유사한 시스템 개발에 응용될 수 있을 것으로 기대된다.

제 4 절 생물정보검색시스템(Bio-KRISTAL) 개발

1. 단백질 아미노산 서열 색인기법 개발

단백질은 20개의 아미노산으로 구성된 일차원 서열이다[4]. 단백질서열 데이터베이스에서 출현할 수 있는 아미노산과 추가적인 코드에 대한 설명이 <table 4-1>에 수록되어 있다.

<table 4-1> 단백질 서열의 아미노산 코드

세글자 코드	한글자 코드	영어이름	한글이름
Ala	A	Alanine	알라닌
Arg	R	Arginine	아르기닌
Asn	N	Asparagine	아스파라긴
Asp	D	Aspartic acid	아스파르산
Cys	C	Cysteine	시스테인
Gln	Q	Glutamine	글루타민
Glu	E	Glutamic Acid	글루탐산
Gly	G	Glycine	글리신
His	H	Hidtidine	히스티딘
Ile	I	Isoleucine	이소루신
Leu	L	Leucine	루신
Lys	K	Lysine	리신
Met	M	Methionine	메티오닌
Phe	F	Phenylalanine	페닐알라닌
Pro	P	Proline	프롤린
Ser	S	Serine	세린
Thr	T	Threonine	스레오닌
Trp	W	Tryptophan	트립토판
Tyr	Y	Tyrosine	티로신
Val	V	Valine	발린
Asx	B	Asn or Asp	
Glx	Z	Gln or Glu	
Sec	U	Selenocysteine	셀레노시스테인
Unk	X	Unknown	알지못함

4) 실제 단백질 서열에서 출현하는 아미노산의 수는 90년대 들어 새로이 1개(Selenocysteine)가 추가됨으로써 21개가 되었다. 그러나 Selenocysteine의 경우 그 수가 매우 적어서 실제 단백질 데이터베이스에서 출현하는 빈도를 조사하면 거의 무시할 수 있을 수준으로만 나타난다. 이외에 B, Z, X 등과 같은 와일드카드 문자가 있으나 이들 또한 전체 데이터베이스에서는 낮은 빈도로 나타나기 때문에 무시할 수 있다.

단백질 서열 자체를 자연어 처리에 의한 색인을 추출하는 것은 매우 어려운 일이다. 또한 구조적인 특징을 가지는 서열이나 도메인 등과 같은 특수한 서열을 색인으로 추출할 수도 있으나 이는 모든 단백질에서 범용으로 사용하기는 어렵다는 단점이 발생한다. 본 연구에서는 단백질 서열을 중복되는 *n*-그램(*n*-gram)으로 분할하여 색인으로 추출하는 방식을 제 1차 색인기법으로 선정하였다. 예를 들어서, 자르는 단위 *n*-gram을 *n*이 4인 테트라그램(tetragram)인 경우에, 단백질 서열 "ACEPITCH"은 "ACEP", "CEPI", "EPIT", "PITC", "ITCH"로 색인이 추출된다. 같은 단백질 서열을 *n*이 5인 펜타그램(pentagram)으로 색인을 추출하게 되면 "ACEPI", "CEPIT", "EPITC", "PITCH"가 된다. 정보검색의 측면에서 볼 때 데이터베이스 전체에서 출현할 수 있는 색인어의 수는 검색 성능에 중요한 영향을 미친다. 단백질의 아미노산 서열을 *n*-gram으로 절단하여 색인할 경우 발생할 수 있는 고유한 색인어의 수는 아래 표와 같다. 표에서 각 *n*-gram의 예로 사용된 원래 단백질 서열은 "ACEPITCH"이다.

<table 4-2> 단백질 서열의 n-grams

n-gram	절단단위	고유 색인어 수	n-gram 예
tri-gram	3	203 = 8,000	ACE, CEP, EPI, PIT, ITC, TCH
tetra-gram	4	204 = 160,000	ACEP, CEPI, EPIT, PITC, ITCH
penta-gram	5	205 = 3,200,000	ACEPI, CEPIT, EPITC, PITCH
hexa-gram	6	206 = 64,000,000	ACEPIT, CEPITC, EPITCH
hepta-gram	7	207 = 128,000,000	ACEPITC, CEPITCH

앞서 언급한 바와 같이 일반적으로 색인어의 수는 수십만에서 수백만 사이일 때에 가장 좋은 성능을 보여준다. 따라서 Table 2에서 나타난 고유한 색인어의 수로부터 단백질의 아미노산 서열 색인에는 n을 4 ~ 5정도로 지정하는 것이 가장 큰 검색 효과가 있을 것으로 예측된다. 참고적으로 BLAST의 단백질 서열 검색에서는 검색 수행 전에 데이터베이스에 미리 n = 3인 n-gram(tri-gram)으로 일종의 색인을 저장하고 이를 대상으로 비교 대상으로 선별한다.

제시된 n-gram 방식들에 대해서 n-gram 색인기를 구현하고 검색 성능 및 속도를 비교한 결과 penta-gram이 가장 우수한 성능을 보였으며, 검색 속도 또한 가장 빨랐다. 이러한 결과를 바탕으로 당해연도에 구현한 단백질서열 색인기에는 기본 색인모드로 penta-gram 추출 방식을 채택하였다. 서열 색인방법을 적용하여 단백질 서열 검색 시스템

과 색인기반 단백질 superfamily 분류 시스템을 개발하였다. 색인방법에 대한 보다 자세한 내용은 단백질 서열검색 시스템 개발과 색인기반 단백질 superfamily 분류 시스템 개발에서 확인할 수 있다.

2. 단백질 서열검색시스템 개발

가. 연구의 중요성

○ 대상 데이터베이스 및 유사 소프트웨어의 현황

- 인간게놈프로젝트(Human Genome Project) 등의 결과로 게놈(Genome) 및 단백질 데이터베이스의 용량이 기하급수적으로 증가하고 있으나, 이를 검색할 수 있는 서열 비교 소프트웨어는 시간소모적인 알고리즘(exhaustive algorithm)을 사용하고 있으므로 온라인 서비스에 차질을 빚고 있는 형편이다. 이제 하드웨어로는 이를 극복할 수 있는 한계를 넘어섰으며, 따라서 소프트웨어 알고리즘 상의 개선이 필요하나 대안에 대한 연구가 미흡한 실정이다.

○ 기존 서열 비교 시스템의 한계 극복

- 기존의 서열 비교 소프트웨어들은 $O(mn)$의 알고리즘으로 수행되는 시간소모적인 알고리즘의 소프트웨어들이나(m은 비교할 대상 서열의 크기, n은 전체 데이터베이스의 크기), 본 연구에서는 서열 비교 시스템의 알고리즘을 대략 $O(mlogn)$으로 감소시킬 수 있는 방안을 연구 개발하고자 한다.

○ 색인기반 단백질 서열검색시스템 개발

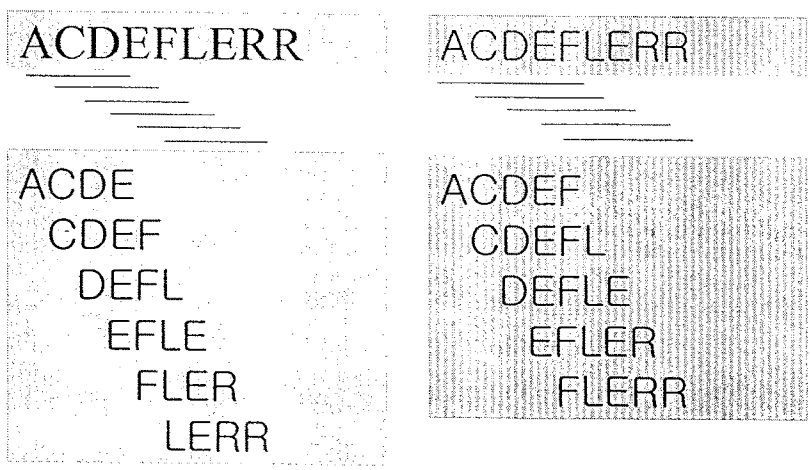
- 위에서 개발된 새로운 서열 비교 및 검색 시스템을 단백질 데이터베이스에 적용하여 신규 단백질 서열과 가장 유사한 서열을 가지는 단백질을 검색할 수 있는 시스템을 개발하는 것이 목표이다.

나. 연구 내용

(1) 색인 기반 검색 기법

(가) n-gram 색인 기법

단백질 서열은 20개의 아미노산 서열로 이루어져 있다. 이 아미노산 서열을 자연어로 생각을 하면, n-gram의 토큰들로 단백질 서열을 분리해낼 수가 있게 된다. 예를 들어서, 자르는 단위 n-gram을 4인 테트라그램(tetragram)인 경우에, 단백질 서열 "ACDEFLERR"은 "ACDE", "CDEF", "DEFL", "EFLE", "FLER", "LERR"로 색인어가 추출된다. 같은 단백질 서열을 N이 5인 펜타그램(pentagram)으로 색인어를 추출하게 되면 "ACDER", "CDEFL", "DEFLE", "EFLER", "FLERR"가 된다. 단백질 서열에 대해서 이 같은 토큰들이 추출되고 난 후에는 역파일(inverted file)에 그 결과들을 저장하게 된다.



<figure 4-1> Examples of n-gram Indexing method: tetragram (left figure) and pentagram (right figure)

역파일에는 각 색인어가 단백질 서열에 나타나는 위치와 빈도를 기록하게 된다. 예를 들어,

"ACEP" 36 (2), 127(3), 1074(1), ···

여기에서 테트라그램 "ACEP"는 36번째 서열에서 두 번 나타나고, 127번째 서열에서 세 번, 1074번째 서열에서 한 번 나타남을 의미한다. 검색을 위한 색인 사용에서 디스크 작업을 줄이기 위해서 저장 위치 리스트(posting list)는 범용 압축 알고리즘인 gzip으로 압축된다. 각기 이런 정보를 이용하여 색인어의 실제 단백질 서열의 위치를 역으로 찾아가게 된다.

N-gram 색인 방법에서 n의 숫자가 커질수록 고유 색인어의 수가 증가하게 된다. 실제로 현재 파일 시스템 관리 체계에서는 헥사그램을 넘어서게 되면, 실제적으로 어렵게 된다. 본 연구에서는 n = 3, 4, 5, 6 일 때 각각 색인과 검색을 거쳐 성능 비교 후에 현재 서비스 되고 있는 ProSeS 시스템에서는 n = 5 인 펜타그램으로 색인화 과정을 거치게 되었다.

(나) 서열 유사도 측정

정보 검색 분야에서 질의어와 대상 문서의 유사성을 측정하기 위한 모델은 여러 가지가 있다. 이들 중에서 가장 많이 연구되고, 널리 사용되는 모델이 벡터 공간 모델이다. 이 모델에서는 질의어와 검색 대상 문서를 각각 고차원 공간의 벡터로 나타낸다. 그 후에 이들 벡터의 내적을 문서 길이와 관계된 정규화 계수로 나누어 줌으로써 질의어와 문서의 유사성을 측정할 수 있다. 질의 서열 q와 대상 서열 d의 유사성 $Sim(q,d)$는 다음과 같이 정의한다.

$$Sim(q,d) = \frac{1}{W_d} \cdot \sum_{t \in q \wedge d} (w_{q,t} \cdot w_{d,t}) \quad \text{에서}$$

$$w_{q,t} = \log(f_{q,t} + 1) \cdot \log(\frac{N}{f_t} + 1)$$

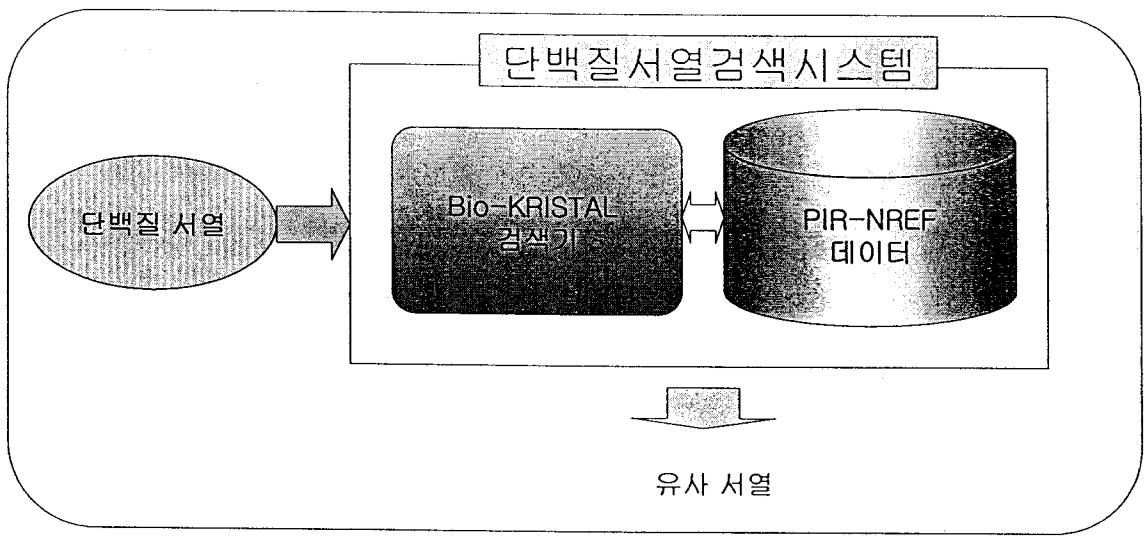
$$w_{d,t} = \log(f_{d,t} + 1) \cdot \log(\frac{N}{f_t} + 1)$$

$$W_d = \log(1 + \sum_{t \in d} f_{d,t})$$

여기에서 $f_{s,t}$는 서열 s안의 n-gram 토큰의 빈도를, N은 데이터베이스 안의 총 서열의 개수를, f_t는 n-gram 토큰 t가 한 번 이상 나타나는 서열들의 개수를, $w_{s,t}$는 질의 서열 또는 대상 서열 s안의 토큰 t의 가중치를 의미하고, W_d는 대상 서열 d의 길이를 나타낸다.

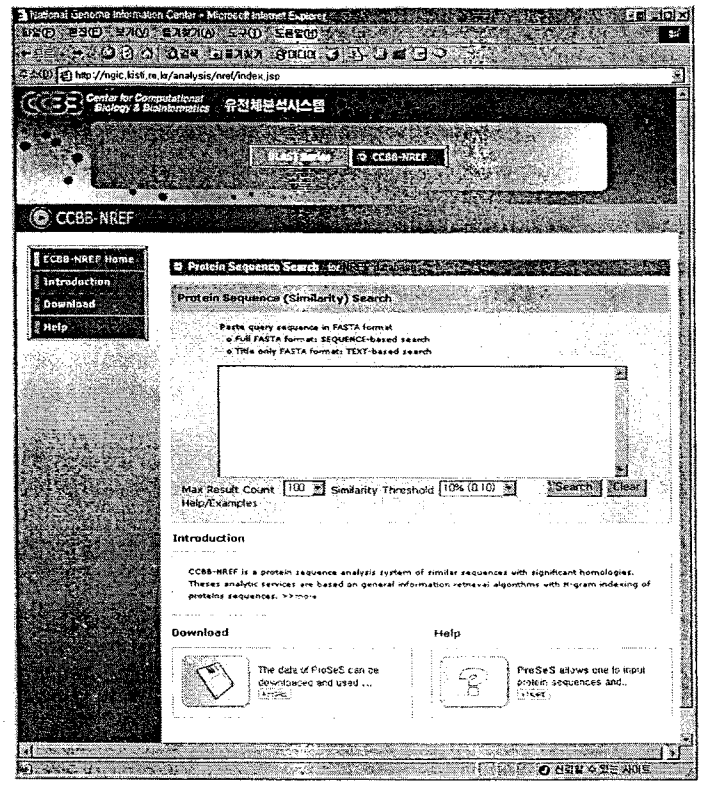
(2) 시스템 구성과 인터페이스

단백질 검색 시스템 CCBB-NREF 은 XEON 듀얼 CPU 2.4GHz 프로세서와 3GB RAM으로 구성된 리눅스 운영체제 환경에서 구현되어서 현재 http://www.ccbb.re.kr 에서 서비스되고 있다. CCBB-NREF 시스템을 도식적으로 다타나면 다음과 같다.

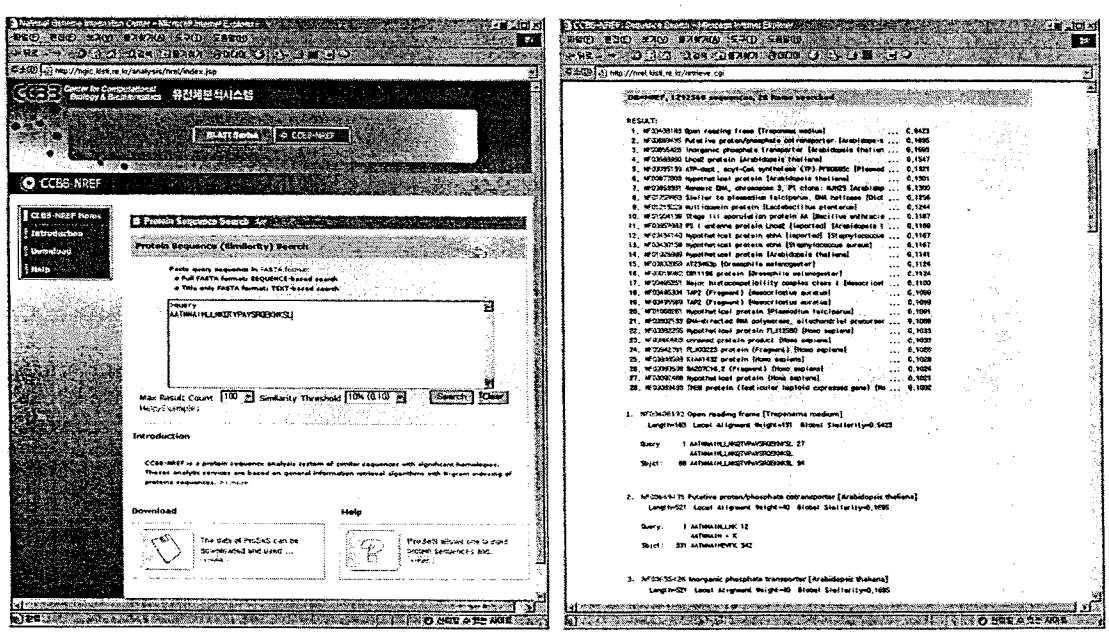


<figure 4-2> A schematic diagram of CCBB-NREF system

또한 CCBB-NREF 시스템의 웹 인터페이스는 다음과 같다. 서열 검색을 위한 화면을 제공하며 간략한 소개와 도움말이 제공된다.



<figure 4-3> Interface of CCBB-NREF



<figure 4-4> ProSeS result page containing similar protein sequences retrieved: the upper part (left) and the lower part (right)

입력서열을 FASTA 형식으로 입력을 하면 상동성이 높은 서열들을 PIR-NREF 데이터베이스에서 검색하여 그 결과를 보여준다.

3. 색인기반 단백질 Superfamily 분류 시스템 개발

가. 서론

생체 세포내에서의 단백질의 기능에 대한 지식은 생물학적 과정의 이해에 있어서 아주 중요하다. 인간유전체사업을 비롯한 거대한 염기서열분석사업의 결과로 급격히 서열정보들이 쏟아지면서 단백질 DB의 서열정보 또한 빠르게 축적되었다. 이와 함께 기능이 밝혀지지 않은 새로운 단백질서열 또한 급격히 증가하게 되었다. 이러한 서열데이터의 급격한 축적에 따른 미지의 서열데이터의 증가로 인해 새로운 단백질의 기능 규명이라는 새로운 문제가 대두되었다.

서열정보의 축적과 함께 서열로부터 단백질의 기능 예측 방법들에 관심이 집중되어왔다. 이러한 방법들은 직접적인 단백질의 유사도 (similarity) 분석 또는 보존되어진 서열 모티프 (motif) 검색에 기초한 특별한 단백질 패밀리 (protein families) 또는 기능적 분류 (functional classification)에 의해 단백질의 기능을 결정한다. 다양한 직접적인 실험에 의한 단백질의 기능 예측이 가능하지만 많은 시간과 비용이 요구되므로, 서열의 유사도 분석 또는 모티프 검색에 기초한 자동화된 컴퓨터를 이용한 예측방법들이 개발되어 왔다.

미지의 단백질 기능을 결정하기 위한 일반적인 방법들에서는 Swiss-Prot (Boeckmann *et al.*, 2003)과 PIR-PSD (Wu *et al.*, 2002) 같은 주석화된 단백질서열 DB로부터 단백질서열의 유사도에 기초하여 단백질의 기능을 추론한다. 이들은 단백질의 기능결정에 있어 종종 중요한 역할을 하지만 기본적인 상동성 (homology) 검색도구가 전체서열에 대한 정렬 (global alignment) 방법에 기초하므로 매우 복잡하며, 많은 시간이 소요된다.

정보검색과 기계학습(machine learning)분야에서 문서범주화 연구는 이미 성숙되었으며 자연어 문서나 HTML 문서의 분류에 성공적으로 적용되어 왔다(Yang, 1994; Yang, 1999; Yang *et al.*, 2002; Sebastiani, 2002; Kim *et al.*, 2004). 20개의 아미노산 코드들로 쓰여진 문서로 단백질 서열을 간주함으로써, 자질추출 (Feature extraction) 방법이 적절하다면 단백질서열에 문서범주화 시스템을 적용할 수 있다. 이와 같은 분류 시스템이 개발되어 단백질서열들의 분류정보를 예측하여 줄 수 있다면, 알려지지 않은 단백질의 성질을 규명하는 초기단계에서 효율적으로 단백질의 기능에 대한 중요한 단서를 제공할 수 있을 것이다.

단백질 sueprfamily 분류 (Protein superFamily Classification) 시스템은 *n*-gram 방법에 기초한 단백질서열에 대한 자질추출 방법과 *k*-NN 범주화 방법을 이용하여 개발한 단백질 superfamily에 대한 예제기반 예측 시스템이다. 기능이 알려지지 않은 단백질 그룹에

대한 정보 예측 기능을 통하여 단백질의 기능 규명에 대한 효율적인 작업을 가능하게 할 것이라 기대한다.

나. 시스템 구성 및 실험 방법

(1) 시스템 환경

C++ STL (Standard Template Libraries)과 Berkerly DB (Loverso *et al.*, 2002) 시스템을 가지고 단백질 서열 데이터베이스를 위한 예제 기반 문서 범주화 방법을 구현하였다. ProFaC 시스템은 FASTA 형식으로 작성된 단백질 서열 데이터로부터 n-gram 자질을 추출하고 Berkerly DB 시스템에 의해 제공된 B+ 트리로 자질 정보를 저장하였다.

하드웨어 시스템은 펜티엄-IV 2.4GHz의 듀얼 CPU, 2 GB 시스템 메모리, 그리고 RAID-5 SCSI 하드 드라이브를 가진 리눅스 시스템을 사용한다.

(2) 자료 구축

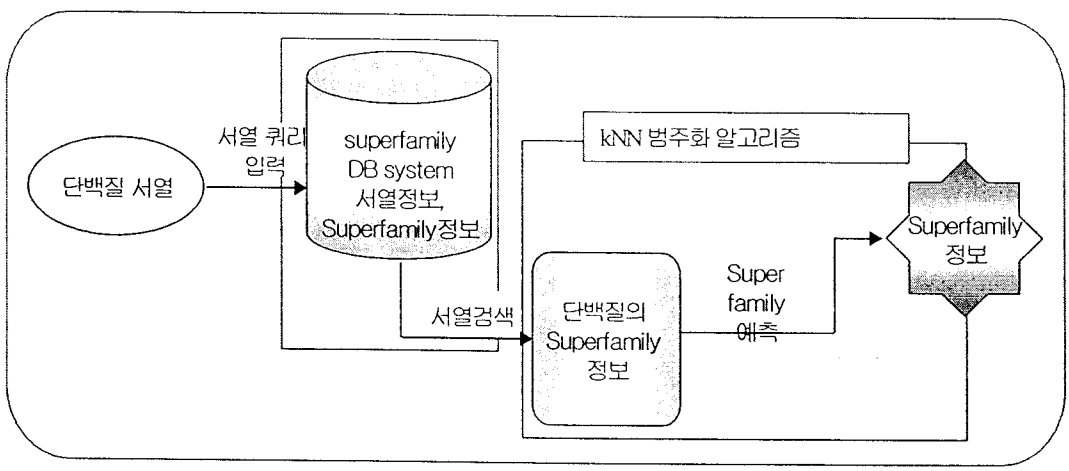
단백질 superfamily 분류 (classification) 시스템을 구축하기 위하여 사전 분류 범주를 가진 서열 데이터베이스를 이용하여 시스템을 위한 데이터를 구축하였다. PIR(Protein Information Resource)의 수퍼패밀리(superfamily)/패밀리(family) 데이터베이스인 iProClass는 서열 유사도(similarity)에 의해 분류된 superfamily 범주에 기초한 단백질 서열의 클러스터 자료를 제공한다. iProClass 배포용 데이터 2.29로부터 superfamily 정보를 가진 서열데이터를 추출하여 36,277개의 superfamily로 분류되어지는 181,576개의 단백질 서열 데이터를 구축하였다. 이 자료를 PIRSF라고 명명하였다.

시스템의 최적화 및 성능 시험을 위하여 구축된 데이터를 학습용 데이터와 시험용 데이터로 나누었다. 두 가지 기준에 의해 두 가지의 학습용 데이터와 시험용 데이터를 가지는 데이터 셋을 구성하였다. 첫 번째 데이터 셋은 PIRSF 서열데이터의 열 번째에 해당하는 엔트리를 시험용 데이터로 분류하고 나머지는 학습용 데이터로 분류하는 거의 무작위 추출로 데이터를 구성하였다. 이 데이터를 SF1 이라고 명한다. 163,419개의 학습용 데이터와 18,157개의 시험용 데이터가 만들어졌다. SF2라고 명한 두 번째 데이터는 7개 이상의 서열 엔트리를 가지는 superfamily 범주만을 선택하여 일곱 번째에 해당하는 서열을 시험용 데이터로 분류하고 나머지는 학습용 데이터로 분류하였다. 두 번째 데이터 셋은 164,542개의

학습용 데이터와 17,034개의 학습용 데이터로 분류되었다.

(3) 시스템 개요

단백질 기능의 해석은 세포의 작용, 생화학적 반응경로, 각 단백질의 독립적인 기능을 이해하기 위한 중요한 단계로서 단백질 서열을 이용한 superfamily 정보를 구함으로써 단백질 기능에 대한 좀 더 정확한 정보를 얻을 필요성이 있다. 이러한 필요성에 입각한 단백질 superfamily 분류서비스 시스템은 Bio-KRISTAL의 단백질 서열 검색기능을 기반으로 서열 수준에서 단백질의 기능을 분류할 수 있는 서비스를 지향한다. 본 시스템은 웹 인터페이스를 통하여 superfamily 분류를 알고자 하는 질의 서열을 입력받는다. 입력된 질의 서열과 유사도가 높은 서열을 유사도 비교를 통하여 DB로부터 검색하고, 검색된 서열들이 가지는 superfamily 범주 정보는 범주기를 통하여 분류된 후, 가능한 superfamily 정보가 사용자에게 제공되어진다. <figure 4-5>은 단백질 superfamily 시스템을 도식적으로 나타낸 것이다.



<figure 4-5> Schematic diagram of *k*NN sequence classifier

(4) 데이터베이스 설계

단백질 superfamily 분류 시스템의 데이터베이스는 KISTI에서 개발한 정보검색엔진인 KRISTAL-2000 시스템을 이용하여 구축되었다. 각 엔트리가 갖는 서열정보는 펜타그램(pentagram) 단위로 자질 추출하여 이를 역파일 형태로 저장하였다. 검색방법은 vector 공간모델을 사용하였다.

(5) 자질 추출 (Feature Extraction)

자연어로 작성된 문서들은 공백에 의해 구분되어지는 단어들로 만들어져 있다. 정보검색 또는 문서 범주화에 있어 대부분 문서로부터 추출된 이러한 단어들은 문서들의 자질로 추출되어지고 역파일로 저장된다 (Salton *et al.*, 1983; Witten *et al.*, 1999). 그러나, DNA와 단백질 서열과 같은 생물학 서열들은 공백과 같은 구분자가 없는 문자열이다. 더구나 서열의 스트링으로부터 어떠한 의미 있는 부분을 구분한다는 것은 어려운 일이다. 이를 위한 적합한 방법들이 개발되어왔고 *n*-gram 토큰 (token) 방법은 그 방법 중의 하나이다. 정보검색 분야에서 *n*-gram을 이용한 문서들의 색인과 검색이 중국 문서 (Wilkinson, 1998)와 OCR 문서(Harding *et al.*, 1997) 같은 문서 집합들에 대하여 성공적으로 적용되어 왔다. 이러한 문서들은 단어의 경계가 명확하지 않은 방법으로 작성되어 있다. 위에서 언급한 생체의 서열 또한 경계가 분명하지 않다. 따라서 *n*-gram 색인 방법은 훌륭히 적용될 수 있을 것이다.

n-gram은 각 서열에 발생하는 간격으로 정의될 수 있다. 그 간격은 고정된 길이 *n*의 부분 스트링으로 부분적으로 겹치게 된다. 예를 들어 *n*이 4이고 "ACEPITCH"의 단백질 서열이 있다면 최종 *n*-gram은 "ACEP", "CEPI", "EPIT", "PITC", 그리고 "ITCH"가 된다.

최적화된 결과를 주는 *n*을 선택하기 위하여 3에서 7까지 *n*의 값을 증가시키면서 실험을 하였다. 각각의 *n*-gram이 가지는 특징이 <figure 4-3>에 정리되어 있다. 'Alphabet #'은 자질 추출에 사용되는 구분 가능한 아미노산 코드의 수이고 'Unique Term #'은 단백질 서열 데이터베이스에 발생될 수 있는 유일한 자질들의 이론적인 빈도이다.

<table 4-3> Characteristic of n-grams from 3-gram to 7 gram

Symbol	Length	Alphabet #	Unique Term #
3-gram	3	20	8,000(20^3)
4-gram	4	20	160,000(20^4)
5-gram	5	20	3,200,000(20^5)
6-gram	6	20	64,000,000(20^6)
7-gram	7	20	128,000,000(20^7)

단백질 서열은 20개의 아미노산으로 구성된 스트링이다. *n*-gram은 단백질 서열로부터 추출되어진 후 역파일 형태로 데이터베이스 시스템에 저장된다. 역파일의 각 엔트리는 검

색할 수 있는 기호와 그것의 빈도정보를 가지고 있다. 빈도정보는 서열 ID의 리스트와 서열에서 발생하는 횟수를 포함하고 있다. 예를 들면 다음과 같다.

$$"ACEP"\ 36\,(2), 127\,(3), 1074\,(1), \ldots$$

이것은 tetragram인 ACEP가 36번째 서열에 2번, 127번째 서열에 3번, 1074번째 서열에 1번 발생한다는 것을 의미한다.

(6) 유사도 측정 및 범주 상관성 측정

정보 검색 분야에서 질의 (query)와 대상 (target) 문서 사이의 유사도를 계산하기 위한 몇 가지 모델이 있다. 그들 중 벡터 공간 모델 (vector space model)은 가장 잘 연구되어왔고 가장 널리 사용되어왔다 (Witten *et al.*, 1999). 이 방법에서는 쿼리와 대상 문서들은 유일한 항의 벡터로서 표현되어지고 쿼리와 대상 문서간의 유사도는 그들간의 내적에 의해 계산되어지고 문서 길이와 관계된 표준화 요소로 나누어진다. 쿼리 서열 q와 대상 서열 d간의 유사도를 계산하는 식은 다음과 같이 정의된다.

$$Sim(q,d) = \frac{1}{W_d} \cdot \sum_{t \in q \wedge d} (w_{q,t} \cdot w_{d,t}) \qquad \text{---------(Equation 1)}$$

$$where$$

$$w_{q,t} = log(f_{q,t}+1) \cdot log(\frac{N}{f_t}+1)$$

$$w_{d,t} = \log(f_{d,t}+1) \cdot log(\frac{N}{f_t}+1)$$

$$W_d = log(1+\sum_{t \in d} f_{d,t})$$

$f_{s,t}$는 서열 s에서 n-gram 토큰 t의 빈도, N은 데이터베이스에서 서열의 총수, f_t는 n-gram 토큰 t가 한번 이상 발생하는 서열의 수, $w_{s,t}$는 쿼리 또는 대상 서열 s에서 토큰 t의 가중치, 그리고 W_d는 대상 서열 d의 길이를 표현한다.

문서 d가 범주 $c_j \in C = \{c_1, c_2, c_3, \ldots, c_{|C|}\}$에 속한다는 것을 결정하기 위하여 ProFaC 분류기는 문서 d와 가장 유사한 k개의 학습문서를 검색하고 문서 d와 검색된 문서

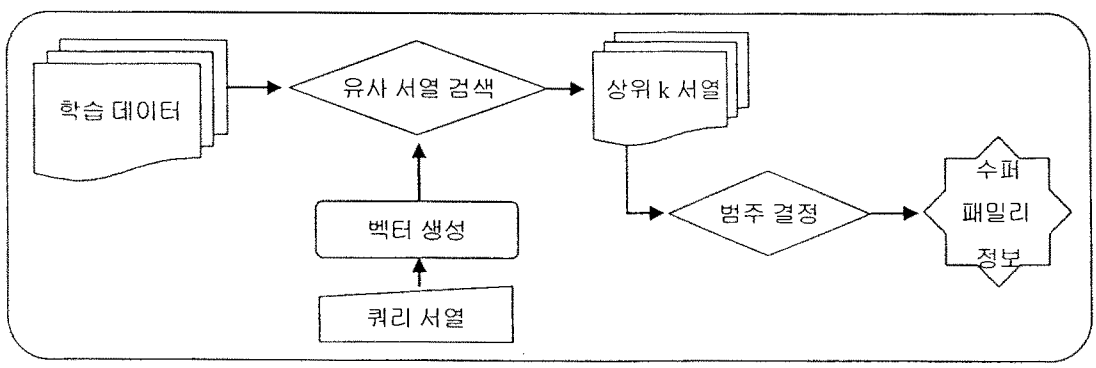
간의 유사도를 더함으로써 c_j의 가중치를 계산한다. 가중치가 충분히 크다면 이 결정은 선택되는 것이고 그렇지 않다면 선택이 되지 않는다. 문서 d에 대한 범주 c_j의 가중치는 'category relevance score, $Rel(c_j, d)$'라고 부르고 다음과 같이 계산된다.

$$Rel(c_j, d) = \sum_{d' \in R_k(d) \cap D_j} Sim(d', d) \qquad \text{--------(Equation 2)}$$

$R_k(d)$는 문서 d와 가장 유사한 상위 k개의 검색된 학습 문서들의 집합이다. D_j는 범주 c_j에 할당된 학습문서의 집합, 그리고 $Sim(d',d)$는 Equation Ⅲ-5에 의해 계산되어진 문서간의 유사도이다. 각 문서들에 대해 주어진 한계값 보다 더 큰 상관 점수를 가진 범주가 문서에 할당되어진다.

(7) *k*NN 방법을 이용한 서열 범주기

단백질 superfamily 분류 시스템은 *k*NN(*k*-nearest neighbor) 범주기(Yang, 1994; Kim *et al.*, 2004)를 이용한 superfamily 범주화 시스템이다. FASTA 포맷의 서열을 입력받아 상위 k개의 서열을 검색한다. 검색된 상위 k개의 서열이 가지는 superfamily 정보들을 기반으로 하여 가능한 superfamily의 가중치를 계산한다. 상위 k개의 서열을 검색하고 증명하는 과정은 위에서 설명한 유사도 측정과 범주의 상관성 측정 방법에 의하여 이루어진다. 이 과정을 거쳐 가장 합당하다고 판단되는 superfamily의 정보를 사용자들에게 제공하게 된다.



<figure 4-6> Schematic diagram of *k*NN sequence classifier

(8) 성능 측정

각각의 n-gram 범주기의 효율을 측정하기 위하여 정확률(p)과 재현율(r)의 표준 정의를 사용하였다.

$$p = \frac{Categories relevant \text{and} retrieved}{Categories retrieved}$$

$$r = \frac{Categories relevant \text{and} retrieved}{Categories relevant}$$

효율성 측정을 위하여 많은 연구자들이 정확률과 재현율과 함께 F_1 값을 사용한다. F_1 값 (van Rijsbergen, 1979)은 정확률과 재현율의 조화평균이고 다음과 같이 정의한다.

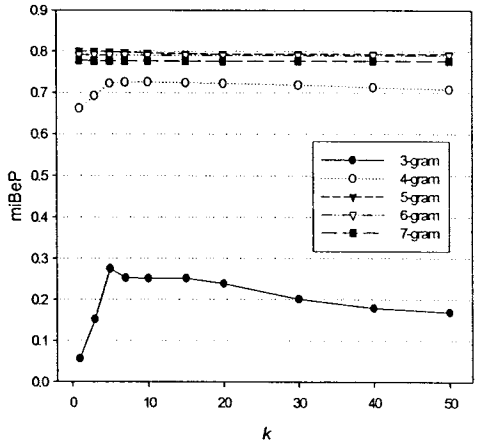
$$F_1 = \frac{2pr}{p+r}$$

정확률과 재현율이 같아지는 F_1 값을 break-even point (BeP)라고 한다. BeP는 F_1 보다는 항상 작거나 같다. BeP는 다른 종류의 분류방법 또는 범주화 방법간의 효율성을 비교하기 위하여 주로 사용되어 왔다. 단백질 superfamily 분류시스템의 최적화와 효율성을 측정하기 위하여 BeP를 계산하였고 BeP를 구하기 힘든 포인트에서는 F_1 값으로 대신하였다. 범주들의 정확률과 재현율의 평균을 구하기 위하여 micro-averaging 방법을 적용하였다.

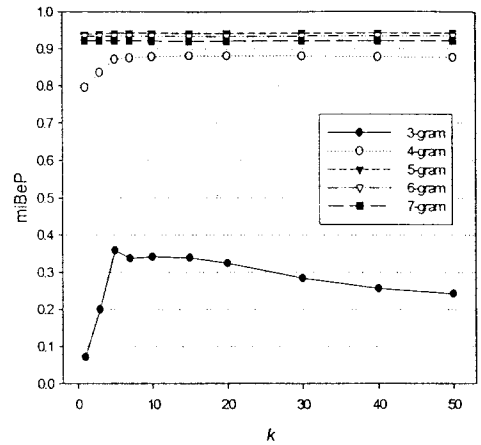
다. 시스템 효율성 측정

(1) 분류 효율성

<figure 4-7>과 <figure 4-8>은 SF1과 SF2의 데이터 집합에 적용한 다섯 가지의 n-gram 방법 (n = 3, 4, 5, 6, 7) 의 k값에 따른 micro-averaged BeP값을 보여준다. 이 두 그래프로부터 k값은 3-gram 방법과 4-gram 방법을 제외하면 분류 효율성에 크게 영향을 주지 않는다는 것을 알 수 있다. 심지어 3-gram 방법과 4-gram 방법의 경우도 k값이 5이상에서는 효율성에 큰 영향을 미치지 않는다는 것을 알 수 있다. 따라서 경험적으로 시스템과 이후 효율성 실험에서는 k값은 10으로 사용하였다.

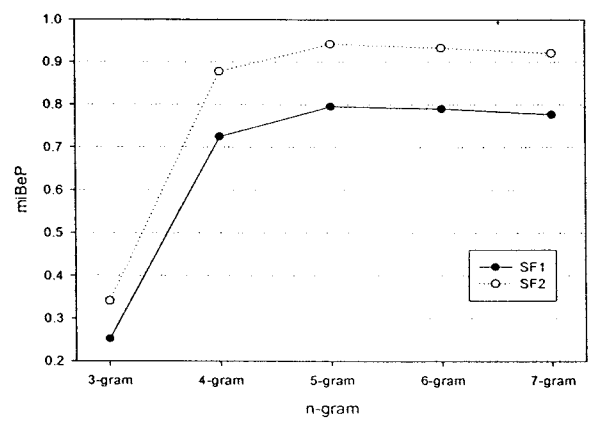


<figure 4-7> Precision and recall break-even point against k for SF1 division. "miBeP" means micro- averaged break-even point and k is the number of top-ranked similar sequences used in superfamily categorization



<figure 4-8> Precision and recall break-even point against k for SF2 division. "miBeP" means micro- averaged break-even point and k is the number of top-ranked similar sequences used in superfamily categorization

다섯 가지의 n-gram 방법의 효율성과 관련하여 <figure 4-7>, <figure 4-8>, <figure 4-9>, 그리고 <table 4-4>를 살펴보면 3-gram 방법의 효율성이 가장 떨어진다는 것을 알 수 있고, 5-gram 방법이 가장 좋은 성능을 보여주는 것을 알 수 있다. k값은 10으로 설정하고 SF1과 SF2 두개의 데이터 집합의 효율성을 측정한 실험결과 5-gram 방법은 각각 0.796과 0.942의 BeP값을 보여 준다(<figure 4-9>, <table 4-4>). 6-gram과 7-gram의 방법 또한 좋은 성능의 결과를 보여주지만 <figure 4-9>의 그래프에서 보여주듯 5-gram이후 분류시스템의 성능이 점점 줄어드는 것을 알 수 있다.



<figure 4-9> Micro averaged precision and recall break-even point against the length of n-grams for SF1 and SF2 divisions at fixed k = 10. Penta-gram shows the best BeP values, 0.796 and 0.942 for SF1 and SF2 data sets, respectively. See also Table Ⅲ-14 for detailed values

<table 4-4> Classification effectiveness for five n-gram methods at k = 10. The maximum BeP values for SF1 and SF2 are underlined. This result is also plotted in Figure Ⅲ-28

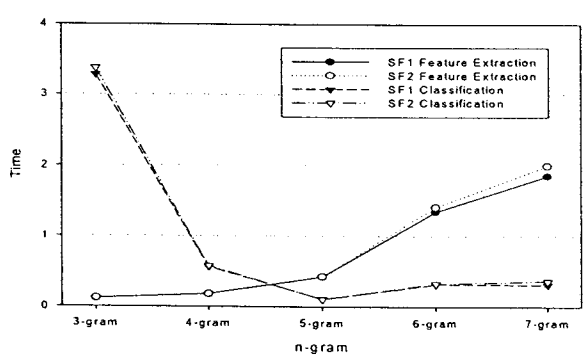
N-gram methods	BeP for SF1	BeP for SF2
3-gram	0.2510	0.3408
4-gram	0.7251	0.8777
5-gram	0.7955	0.9421
6-gram	0.7900	0.9330
7-gram	0.7765	0.9206

위의 실험결과들을 통하여 단백질 서열에 대한 가장 적합한 자질 추출 방법은 5-gram 이라는 것을 알 수 있다. 현재 단백질 superfamily 분류 시스템의 자질 추출 방법은 이 실험결과에 따라 5-gram을 적용하였다.

(2) 단백질 서열 분류 시스템의 속도측정

분류 속도에 있어서 또한 5-gram 방법이 가장 빠른 분류 속도를 보여주었고, 초당 10개의 서열에 대한 분류 정보를 제공하였다. Figure Ⅲ-29에서 각 n-gram의 분류 속도 측정 그래프를 볼 수 있다. 분류 속도 실험 그래프를 통하여 흥미로운 경향성을 볼 수 있다. n-gram의 n값이 3에서 5로 증가할수록 분류 속도가 반비례한다는 것을 알 수 있다. 그러나 5이후의 6과 7에서는 다소 증가하는 것을 볼 수 있다.

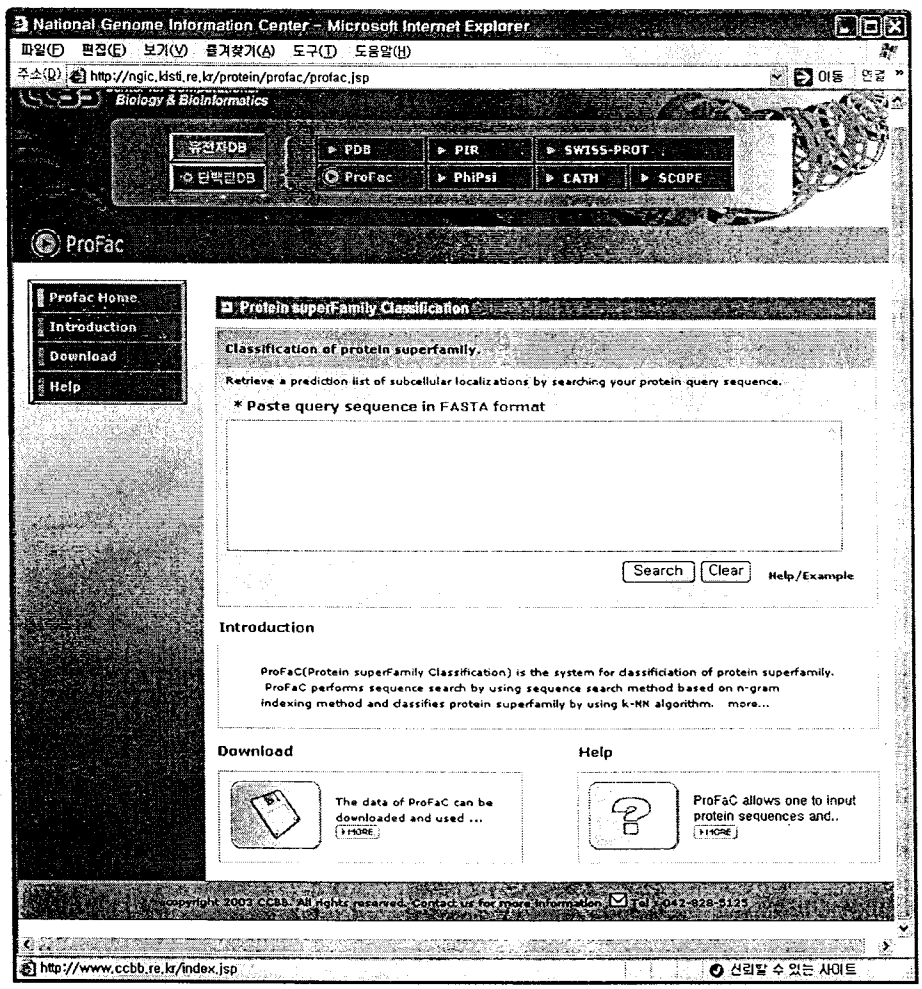
자질 추출 시간은 <figure 4-10>에서 볼 수 있듯이 n값이 증가함에 따라 함께 증가한다는 것을 알 수 있다. SF1의 학습데이터에 대한 3-gram의 자질 추출 시간은 7분정도가 소요되었고, 4-gram방법은 11분, 5-gram방법에서는 25분, 6-gram은 80분, 7-gram에서는 111분이 소요되었다. 자질 추출 시간이 적용된 n-gram방법에 의존하는 것은 사실이지만 이것은 시스템 구축 시 관리자의 몫으로 시스템을 사용하는 사용자와는 무관하다. 또한 가장 효율성이 좋다고 판단된 5-gram의 경우 25분정도의 자질 추출 시간을 필요로 한다.



<figure 4-10> Features extraction and classification time for SF1 and SF2 divisions. Feature extraction time (hr) is the total hours to extract and store whole feature information into inverted files. Classification time (sec/seq) is total seconds to classify one test sequence averaged all over the whole test set of SF1 and SF2 data sets

라. 웹 인터페이스 (Web Interface)

<figure 4-11>은 단백질 superfamily 분류 시스템의 주 화면을 보여주는 그림이다. FASTA 형식의 서열 쿼리를 입력받아 단백질 서열의 분류정보를 검색할 수 있고, 또한 텍스트 검색창을 두어 키워드에 의한 superfamily 정보를 검색하는 것이 가능하다.

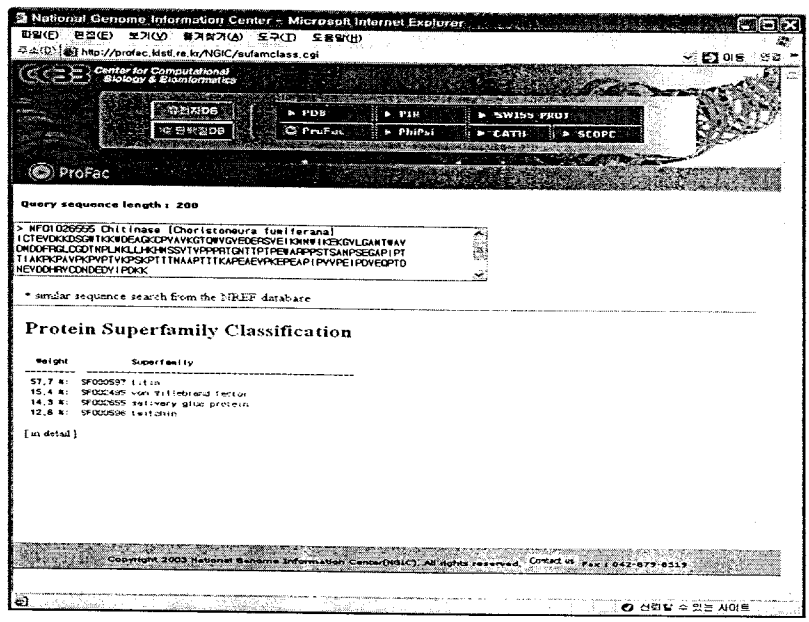


<figure 4-11> Main service page of ProFaC

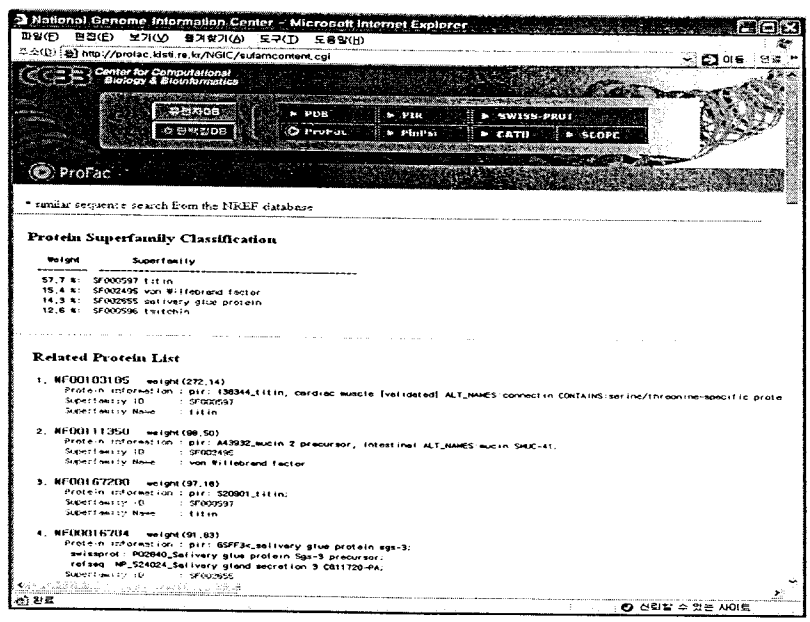
다운로드 페이지를 두어 단백질 superfamily 분류 데이터베이스에 올려진 superfamily 서열데이터를 다운받을 수 있고 Help 페이지를 통하여 익숙치않은 사용자를 위한 예제와 FASTA 형식에 대한 설명을 제공한다.

<figure 4-12>은 서열검색을 통하여 나온 결과 화면이다. 입력된 서열 쿼리에 대한 정보를 길이와 함께 보여주고 아래 superfamily의 분류정보를 가중치와 함께 보여준다. 각각

의 superfamily 아이디는 iProClass 데이터베이스로의 링크 정보를 담고 있다. 또한 쿼리 서열의 정보를 보여주는 텍스트 박스 아래 서열 검색을 할 수 있도록 ProSeS 서열검색서비스로의 링크정보를 준다. 'in detail'을 통하여 관련된 단백질의 서열들의 상세정보를 볼 수 있다. <figure 4-13>은 상세보기의 화면이다.



<figure 4-12> Classification result on query sequence



<figure 4-13> Classification information and related protein list

4. 단백질 서열 N-Gram 빈도 데이터베이스 (Protein N-Gram Frequency: ProNGF) 구축 및 활용

가. 서론

단백질 서열 N-Gram 빈도 데이터베이스 시스템 (Protein N-Gram Frequency, 이하 약칭 ProNGF)이란 단백질 서열 내의 길이가 N인 연속된 아미노산 서열(N-Gram)이 출현하는 출현빈도 (Frequency)를 단백질 서열 데이터베이스에 대해서 측정하는 것이다. 단백질 서열의 특정 구조에서 발견되는 N-Gram의 빈도, 2차 구조, 출현 단백질 정보 등을 알아봄으로써 특정 N-Gram의 기능을 예측할 수 있다. KRISTAL-2000의 단백질 서열 검색 기능을 이용하여 조지타운대학교에서 제공하는 단백질 서열 데이터베이스인 PIR-NREF 데이터베이스에 대한 N-Gram 출현빈도 검색 시스템을 구축하였다. 또한 RCSB (Research Collaboratory for Structural Bioinformatics)에서 제공하는 PDB 데이터베이스에 대해 단백질 2차구조를 추출하여 이에 대한 N-Gram 출현빈도 검색 시스템을 구축하였다.

나. PIR-NREF를 이용한 단백질 서열 N-Gram 빈도 데이터베이스(Protein N-Gram Frequency: ProNGF) 구축 및 활용

(1) 개요

N-Gram 빈도를 측정하는 단백질 서열 데이터베이스로서 미국의 조지타운대학교의학센터 (GUMC)의 PIR (Protein Infromation Resource)에서 제공하는 단백질 서열 DB중 최대규모인 PIR-NREF (http://pir.georgetown.edu/pirwww/search/pirnref.shtml)를 이용하였다. PIR-NREF는 Protein Information Resource Non-Redundant Reference Protein Database의 약칭으로서 PIR-PSD(Protein Sequence Database), Swiss-Prot(Curated Protein Sequence Database), TrEMBL, RefSeq, GenPept, PDB의 모든 단백질 서열 데이터를 포함하며 2주일마다 갱신된다. 2004년 4월 말 현재 1,597,470 개의 단백질 서열 데이터를 가지고 있으며 서열 ID와 종 수준의 분류에 의해 수집되었으므로 비중복성(Non-Redundancy)이 있다는 것이 가장 큰 특징이다.

PIR-NREF 데이터베이스를 기반으로 하여 N-Gram 빈도 데이터베이스를 무한히 만들 수 있다. 이는 N-Gram이 무엇인지를 알면 쉽게 이해할 수 있다.

N-Gram이란 단백질 서열 내의 길이가 N인 연속된 아미노산 서열이다. 길이 N의 크기에 따라서 Uni-Gram, Bi-Gram, Tri-Gram, Tetra-Gram, Penta-Gram, Hexa-Gram 등으로 칭한다. 각 N-Gram을 서열 검색 시스템의 색인어로 사용할 때의 특징은 다음과 같다.

◦ Uni-Gram : 서열을 1개씩 끊어서 색인어로 추출

- 단백질 아미노산: 총 24개 종류

- THFYYEC -> T, H, F, Y, Y, E, C

- 고유 색인어 수 = 24

◦ Bi-Gram : 서열을 2 개씩 중첩하여 끊어서 색인어로 추출

- THFYYEC -> TH, HF, FY, YY, YE, EC

- 고유 색인어 수 = 242= 576

◦ Tri-Gram : 서열을 3 개씩 중첩하여 끊어서 색인어로 추출

- THFYYEC -> THF, HFY, FYY, YYE, YEC

- 고유 색인어 수 = 243= 13,824

- 고유 색인어의 수가 적음

◦ Tetra-Gram : 서열을 4 개씩 중첩하여 끊어서 색인어로 추출

- THFYYEC -> THFY, HFYY, FYYE, YYEC

- 고유 색인어 수 = 244= 331,176

- 고유 색인어의 수가 적절함

◦ Penta-Gram : 서열을 5 개씩 중첩하여 끊어서 색인어로 추출

- THFYYEC -> THFYY, HFYYE, FYYEC

- 고유 색인어 수 = 245= 7,962,624

- 고유 색인어의 수가 너무 많음 -> 검색 시 디스크 I/O 에 대한 부담이 큼

◦ N-Gram : 서열을 N 개씩 중첩하여 끊어서 색인어로 추출

- 고유 색인어 수 = 24N

PIR-NREF의 단백질 서열들에 대해 위의 각 N-Gram의 존재하는 모든 색인어를 추출한 후 각 N-Gram 색인어의 출현 빈도(N-Gram Frequency)를 계산하여 N-Gram 빈도 데이터베이스를 구축한다. 이 때 출현 빈도는 TF (Term Frequency), 즉 하나의 N-Gram이 하나의 서열 내에 등장하는 횟수를 모든 서열에 대하여 합친 것과 SF (Sequence Frequency 또는 Document Frequency), 즉 하나의 N-Gram이 총 서열 중 등장하는 서열 개수의 두 가지를 고려한다. 또, N-Gram이 출현하는 확률을 존재하는 모든 색인어에 대해 구하는 Probability of Total Occurrences (이하 REAL로 통칭)와 N-Gram 내의 아미노산들이 서로 떨어져서 독자적으로 존재할 확률, 즉 Probability of Random Distribution (이하 RANDOM으로 통칭)을 구하여 그 비율인 REAL/RANDOM Ratio를 본다. 이 때 REAL/RANDOM 값이 1보다 훨씬 작으면 실제로 존재할 확률이 매우 낮은 것이고, REAL/RANDOM 값이 1보다 훨씬 크면 실제로 존재할 확률이 매우 높은 것이다.

(2) 데이터베이스 설계

조지타운대학교의 PIR-NREF는 2004년 4월 말 현재 1,597,470 개의 단백질 서열 정보를 포함하고 있다. 또한 새롭게 발견되거나 만들어지는 단백질 서열들의 증가에 의해 매월 약 1만 개 이상 증가하고 있다. 이를 기본으로 하여 새로이 만들어지는 N-Gram 빈도 데이터베이스는 Penta-Gram까지만 생각하더라도 최소 약 $245 \simeq 7{,}000{,}000$ 개에 달하는 방대한 색인어 수를 자랑하게 된다. 따라서 올바른 데이터베이스 구조를 선택하지 않을 경우에는 사소한 잘못으로도 검색시스템의 성능에 큰 지장을 초래할 수 있다. 이러한 점을 고려하여 다음과 같이 검색시스템을 설계하였다.

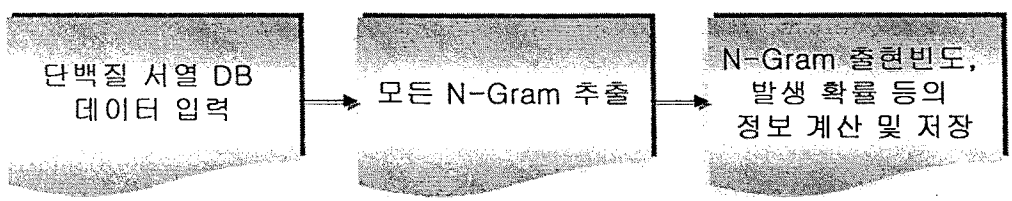
○ 원시 자료 :

원시자료는 조지타운대학교에서 공개적으로 배포하는 PIR-NREF 데이터베이스를 기반으로 한 단백질 서열 N-Gram 빈도 데이터베이스(ProNGF)를 사용하였다. PIR-NREF 데이터베이스는 FASTA 및 XML 포맷으로 되어 있는데 본 시스템에서는 이 중 FASTA 포맷 파일(<figure 4-14>)을 대상으로 하여 새로 단백질 서열 N-Gram 빈도 데이터베이스를 구성하였다. 이를 위해 작성한 N-Gram 분석 프로그램(<figure 4-15>)을 이용하여, FASTA

포맷 파일 안의 단백질 서열 정보로부터 Uni-Gram부터 Penta-Gram까지의 색인어를 추출하고, 각 N-Gram에 대한 TRM(Term Frequency), DOC(Sequence Frequency), PRR(Probability of Total Occurrences), PRI(Probability of Random Distribution), RAT(REAL/RANDOM)의 5가지 정보를 구하였다 (<figure 4-16>). 이 때 검색 시의 선택사항으로 24가지의 아미노산 중 출현빈도가 현저히 낮은 B (Aspartate 또는 Asparagine, D 또는 N), Z (Glutamate 또는 Glutamine, E 또는 Q) , X (무관독) 의 세 가지 아미노산을 뺀 나머지 아미노산만을 대상으로 검색할 수 있도록 하기 위하여, N-Gram 빈도 데이터베이스를 B, Z, X를 포함한 것과 B, Z, X를 포함하지 않은 2가지로 나누었다. 각 N-Gram 빈도 데이터베이스를 ProNGF 시스템의 볼륨(Volume)으로 구성하였으며, 그 결과 ProNGF 검색 서비스에서는 10개의 볼륨이 사용되고 있다[5].

```
>NF00716896 transposase tnpB [imported] [Shigella flexneri]
MKYVFIENHRAEFSIKAMCRVLRVARSGWYVWLRRRHQMSLRQQFR
LTCDAAVHKAFFEAXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

<figure 4-14> PIR-NREF 데이터베이스의 단백질서열 데이터(FASTA 포맷)



<figure 4-15> N-Gram 분석 프로그램

```
Total Tokens: 363312325
Total Sequences: 1235044
#SEQ=AAAAA
#TRM=37413
#DOC=14091
#PRR=1.029775e-04
#PRI=2.549652e-06
#RAT=40.388854
```

<figure 4-16> PIR-NREF 데이터베이스에 기반한 N-Gram 빈도 데이터베이스의 N-Gram 데이터(Penta-Gram의 예)

5) 현재까지는 N = 5 d인 Penta-Gram 까지의 검색만 가능하기 때문이다.

○ 색인 범위 :

<table 4-7>에서 볼 수 있듯이, 각 항목을 ProNGF 시스템의 섹션으로 정의함으로써 해당 필드(Field)만으로도 검색이 가능하도록 하였다. 각 섹션은 N-Gram 서열(NGRAM), 용어 빈도(TERMF), 서열 빈도(SEQUF), 실제 통계(REALP), 이상 통계(IDEALP), 실제/이상 비율(RATIO)로 이루어진다.

○ 색인 방법 :

색인 방식은 주검색 섹션인 N-Gram 서열(NGRAM)에 대해 KRISTAL-2000의 INDEX_AS_IS 방식을 사용하였다.

○ 검색 모델 :

KRISTAL-2000 정보검색시스템은 불리안(Boolean) 검색모델 및 벡터공간(Vector Space) 검색모델을 제공한다. ProNGF 데이터베이스 시스템에서는 불리안 검색모델을 기본 검색모델로 채택하였으며, 자료 분석을 통해 ProNGF 데이터베이스에 가장 알맞도록 불리안 모델을 최적화하였다. 즉, 불리안 모델의 OR, NOT 연산자를 제외하고, AND 및 WITHIN 연산자만을 지원하도록 하였다. 이는 ProNGF의 자료가 문서순위를 결정(document ranking)할 필요가 없는 속성을 가지고 있기 때문이며, 또한 자료의 특성상 일반 사용자에게는 OR 검색의 필요가 없다고 판단했기 때문이다.

위와 같은 사항을 토대로 만든 KRISTAL-2000 스키마(Schema) 파일의 내용은 <table 4-5>, <table 4-6> 와 같다. KRISTAL-2000에 기반한 검색 시스템은 데이터베이스의 대부분의 설정을 스키마 파일에 정의할 수 있도록 하고 있다. 검색 시의 선택사항으로 24가지의 아미노산 중 출현빈도가 현저히 낮은 B (Aspartate 또는 Asparagine, D 또는 N), Z (Glutamate 또는 Glutamine, E 또는 Q) , X (무판독) 의 세 가지 아미노산을 뺀 나머지 아미노산만을 대상으로 검색할 수 있도록 하기 위하여, N-Gram 빈도 데이터베이스를 B, Z, X를 포함한 것과 B, Z, X를 포함하지 않은 2가지로 나누었다. ProNGF 원시자료 구조에 따라 본 시스템에서는 원시자료를 N-Gram 서열(NGRAM), 용어 빈도(TERMF), 서열 빈도(SEQUF), 실제 통계(REALP), 이상 통계(IDEALP), 실제/이상 비율(RATIO)로 구분하였다. 섹션은 N-Gram 서열은 섹션값 자체를 색인으로 삼았고, 나머지는 색인하지 않거나 보다 상세한 검색을 위해서 용어빈도(TERMF) 및 실제/이상 비율(RATIO)를 INDEX_BY_NUMERIC 색인방식으로 색인하였다. 각 섹션의 색인 방식은 <table 4-7> 에서 볼 수 있다.

<table 4-5> PIR-NREF 데이터베이스에 기반한 ProNGF 데이터베이스 검색을 위한 KRISTAL-2000 스키마

```
// KRISTAL version
KRISTAL_VERSION="2000"
KRISTAL_DIRECTORY='/home/k2000/K2000'
DATABASE_DIRECTORY='/home/prongf/VOLs'
DATABASE_GROUP_NAME="PRONGF"

CREATE_SCHEMA
{
  SECTION_DEFINITION
  {
        (1)     LABEL="N-Gram Sequence"          SECTION_NAME=NGRAM
INDEX_TYPE="INDEX_AS_IS",
        (2)     LABEL="Term Frequency"           SECTION_NAME=TERMF
INDEX_TYPE="INDEX_AS_NUMERIC",
        (3)     LABEL="Sequence Frequency"               SECTION_NAME=SEQUF
INDEX_TYPE="DO_NOT_INDEX",
        (4)     LABEL="Real Statistics"          SECTION_NAME=REALP
INDEX_TYPE="DO_NOT_INDEX",
        (5)     LABEL="Ideal Statistics"         SECTION_NAME=IDEAP
INDEX_TYPE="DO_NOT_INDEX",
        (6)     LABEL="Real to Ideal Ratio"      SECTION_NAME=RATIO
INDEX_TYPE="INDEX_AS_NUMERIC"
  }
};

// 데이터베이스 생성
CREATE_DATABASE
{
// 데이터베이스 이름과 크기를 정의
        (1)     DATABASE_NAME=PRONGF1            DATABASE_SIZE=20,
        (2)     DATABASE_NAME=PRONGF2            DATABASE_SIZE=20,
        (3)     DATABASE_NAME=PRONGF3            DATABASE_SIZE=20,
        (4)     DATABASE_NAME=PRONGF4            DATABASE_SIZE=80,
        (5)     DATABASE_NAME=PRONGF5            DATABASE_SIZE=900,
        (6)     DATABASE_NAME=PRONGF1woBZX       DATABASE_SIZE=20,
        (7)     DATABASE_NAME=PRONGF2woBZX       DATABASE_SIZE=20,
        (8)     DATABASE_NAME=PRONGF3woBZX       DATABASE_SIZE=20,
        (9)     DATABASE_NAME=PRONGF4woBZX       DATABASE_SIZE=80,
        (10)    DATABASE_NAME=PRONGF5woBZX       DATABASE_SIZE=900
};

DEFINE_DOCUMENT_STRUCTURE
{
  STRUCTURE_DEFINITION = ALL_DOCUMENTS {
        (1)     TAG_NAME="#SEQ="     ACTION=COPY   SECTION_NAME=NGRAM
NEW_DOCUMENT_FLAG=TRUE,
        (2)     TAG_NAME="#TRM="     ACTION=COPY   SECTION_NAME=TERMF,
        (3)     TAG_NAME="#DOC="     ACTION=COPY   SECTION_NAME=SEQUF,
        (4)     TAG_NAME="#PRR="     ACTION=COPY   SECTION_NAME=REALP,
        (5)     TAG_NAME="#PRI="     ACTION=COPY   SECTION_NAME=IDEAP,
        (6)     TAG_NAME="#RAT="     ACTION=COPY   SECTION_NAME=RATIO
  }
};
```

<table 4-6> PIR-NREF 데이터베이스에 기반한 ProNGF 데이터베이스 검색을 위한 KRISTAL-2000 스키마

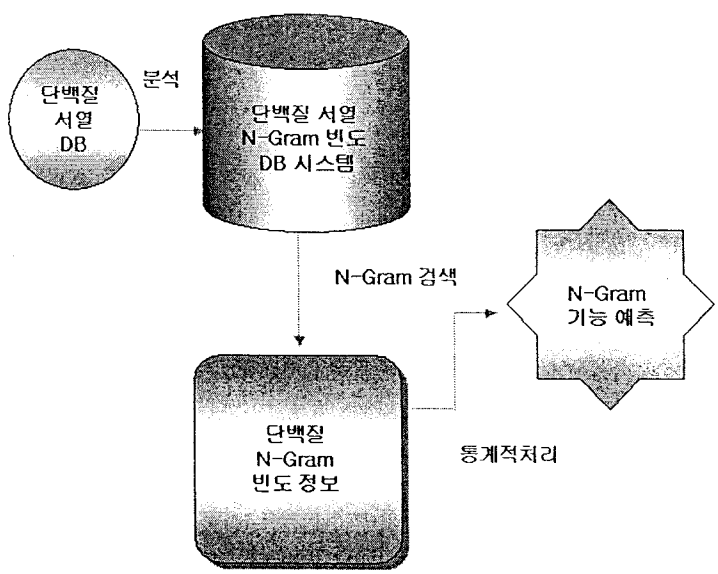
```
// 문서 그룹 정의
DEFINE_DOCUMENT_GROUP
{
        (1)     prongf1=('/home/prongf/data/kst/01_monogram_.kst'),
        (2)     prongf2=('/home/prongf/data/kst/02_digram___.kst'),
        (3)     prongf3=('/home/prongf/data/kst/03_trigram__.kst'),
        (4)     prongf4=('/home/prongf/data/kst/04_tetragram.kst'),
        (5)     prongf5=('/home/prongf/data/kst/05_pentagram.kst'),
        (6)     prongf1woBZX=('/home/prongf/data/kst/01_monogram__woBZX.kst'),
        (7)     prongf2woBZX=('/home/prongf/data/kst/02_digram____woBZX.kst'),
        (8)     prongf3woBZX=('/home/prongf/data/kst/03_trigram___woBZX.kst'),
        (9)     prongf4woBZX=('/home/prongf/data/kst/04_tetragram_woBZX.kst'),
        (10)    prongf5woBZX=('/home/prongf/data/kst/05_pentagram_woBZX.kst')

};
// 문서 적재
LOAD_DATABASE
{
        (1)     FROM=prongf1            TO=PRONGF1
WITH=ALL_DOCUMENTS,
        (2)     FROM=prongf2            TO=PRONGF2
WITH=ALL_DOCUMENTS,
        (3)     FROM=prongf3            TO=PRONGF3
WITH=ALL_DOCUMENTS,
        (4)     FROM=prongf4            TO=PRONGF4
WITH=ALL_DOCUMENTS,
        (5)     FROM=prongf5            TO=PRONGF5
WITH=ALL_DOCUMENTS,
        (6)     FROM=prongf1woBZX            TO=PRONGF1woBZX
WITH=ALL_DOCUMENTS,
        (7)     FROM=prongf2woBZX            TO=PRONGF2woBZX
WITH=ALL_DOCUMENTS,
        (8)     FROM=prongf3woBZX            TO=PRONGF3woBZX
WITH=ALL_DOCUMENTS,
        (9)     FROM=prongf4woBZX            TO=PRONGF4woBZX
WITH=ALL_DOCUMENTS,
        (10)    FROM=prongf5woBZX            TO=PRONGF5woBZX
WITH=ALL_DOCUMENTS
};
END
```

<table 4-7> 각 섹션별 색인 정보

섹션	색인 방식	색인 방법	비고
NGRAM	INDEX_AS_IS	섹션별 색인	PRIMARY KEY로 접근, 주검색 섹션
TERMF	INDEX_AS_NUMERIC	숫자 색인	용어 빈도
SEQUF	DO_NOT_INDEX	색인하지 않음	서열 빈도
REALP	DO_NOT_INDEX	색인하지 않음	실제 통계
IDEALP	DO_NOT_INDEX	색인하지 않음	이상 통계
RATIO	INDEX_AS_NUMERIC	숫자 색인	실제/이상 비율

(3) 데이터베이스 검색 구조

<figure 4-17>에서는 ProNGF 데이터베이스의 검색 구조를 보여주고 있다. ProNGF 데이터베이스 검색 시스템에서는 검색 서버로서 KRISTAL-2000 정보검색시스템을 채택하였으며, 클라이언트는 ProNGF 검색을 위해 최적화하는 방식으로 새로 구성되었다. 검색 클라이언트의 설계 주안점은 클라이언트 소스(Source)와 웹 인터페이스를 분리하는 방식을 채택하여 프로그램의 수정을 최소화하고도 웹 인터페이스를 수정할 수 있도록 하였다. 클라이언트는 CGI 방식으로 구현되었으며, C++ 표준 라이브러리, CGICC 모듈, 및 KRISTAL-2000 Client 라이브러리로 구성되어 있다. 또한 검색의 성능을 높이기 위해, 검색 모델을 최적화하는 부분은 검색 클라이언트에서 수행하도록 설계하였다. 이렇게 함으로써 KRISTAL-2000라는 범용 정보검색관리시스템을 수정하지 않고도 ProNGF 검색을 최적화할 수 있다. ProNGF 데이터베이스 검색시스템에서 채택한 검색모델은 불리안(Boolean)검색모델이며, GenBank 검색 클라이언트에서는 이 검색 모델을 보다 최적화하여 불리안 연산자들 중 AND 및 WITHIN 연산자를 사용자의 질의에 따라 최적화된 형식으로 재구성하여 KRISTAL-2000 검색서버에 전달할 수 있도록 하였다.

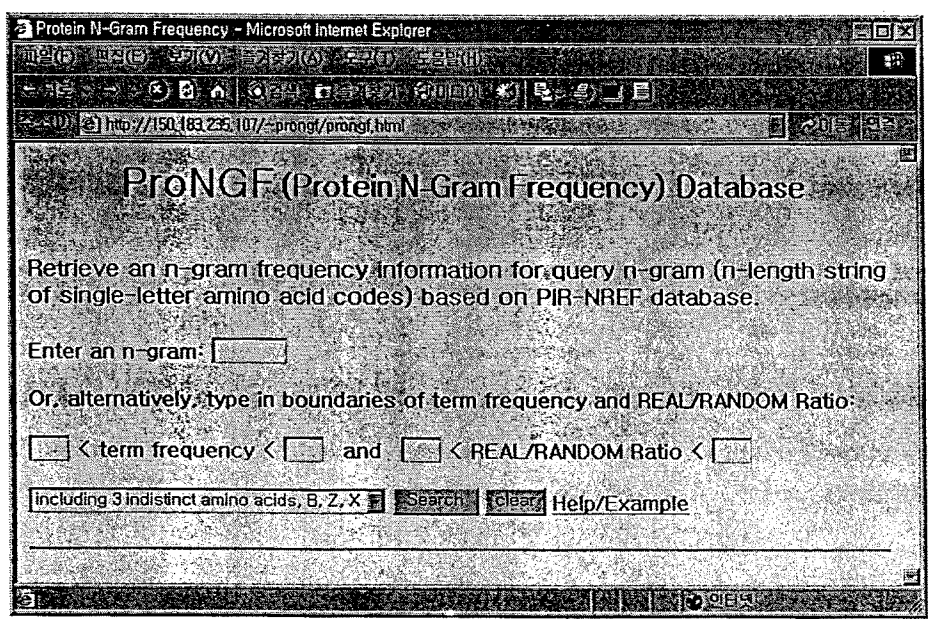


<figure 4-17> ProNGF 데이터베이스 검색 구조

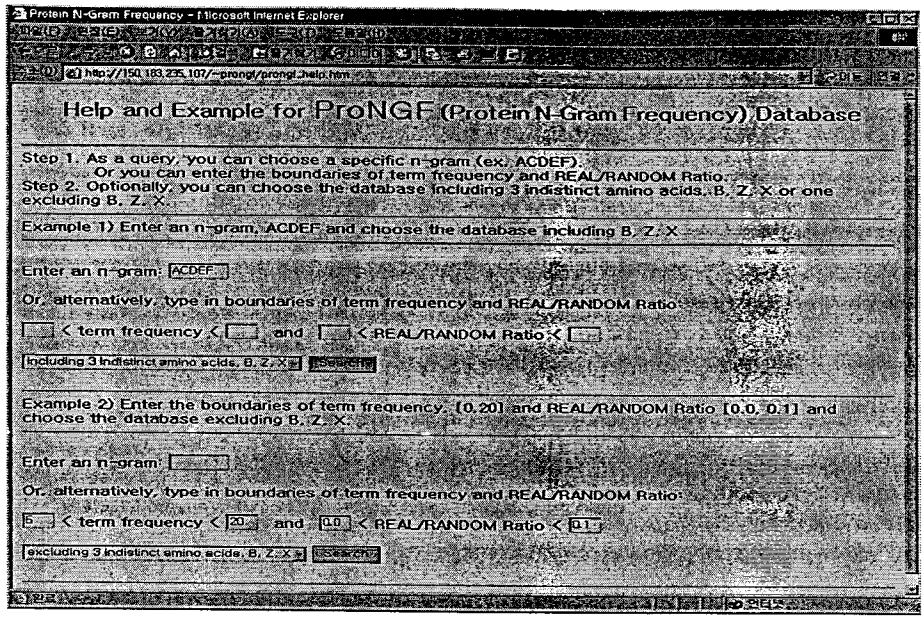
(4) 웹 검색 인터페이스

<figure 4-18>에서는 ProNGF 검색 서비스의 초기화면(http://150.183.235.107/~prongf/prongf.html)을 보여주고 있다. 검색 서비스 개발의 주안점은 사용자 편의성 및 기능성에 주안점을 두었다. 아래 그림에서 볼 수 있듯이 ProNGF 데이터베이스를 검색하고자 하는 사용자는 단순히 자신이 찾고자하는 N-Gram만을 입력함으로써 원하는 검색 결과를 얻을 수 있다. 또는 참고로 PIR-NREF 에서도 목적은 다르지만 이와 비슷하게 서열 조각만을 입력함으로써 원하는 서열을 찾는 메뉴가 있으므로, 기존 PIR-NREF 사용자들도 쉽게 접근할 수 있다는 장점이 있다. 또한 term frequency, 즉 N-Gram의 출현 빈도의 범위 및 REAL/RANDOM Ratio 의 범위에 따른 검색이 가능하도록 하였다. 이 검색법으로는 저빈도 서열이나 다빈도 서열이 나타나는 한계를 정하여 검색할 수 있다. 마지막으로 검색 시 선택사항으로 24가지의 아미노산 중 출현빈도가 현저히 낮은 B (Aspartate 또는 Asparagine, D 또는 N), Z (Glutamate 또는 Glutamine, E 또는 Q), X (무판독) 의 세 가지 아미노산을 뺀 나머지 아미노산만을 대상으로 검색할 수 있도록 하였다.

검색에 도움을 주기 위하여 도움말 및 검색 예를 보여주는 페이지가 링크되어 있다. (<figure 4-19>)

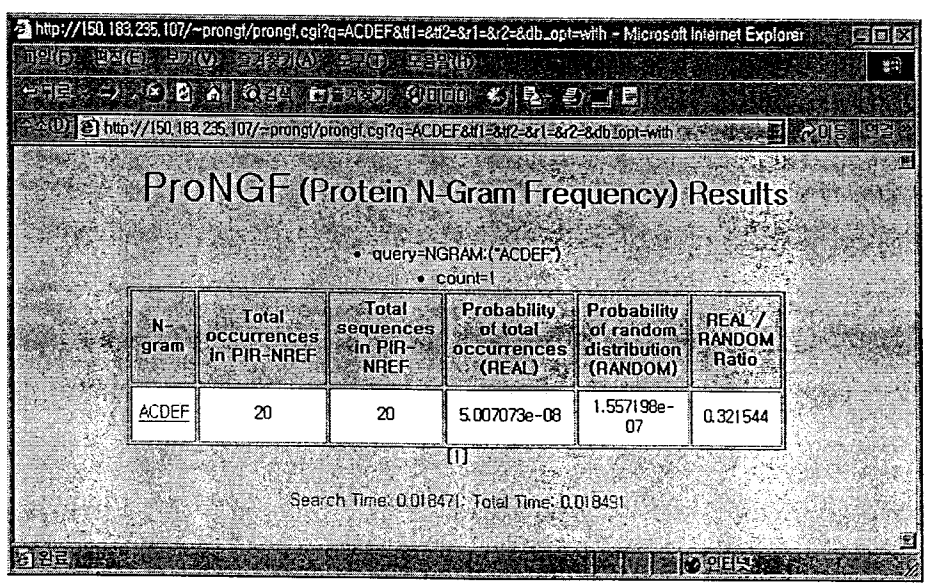


<figure 4-18> ProNGF 검색 초기 화면

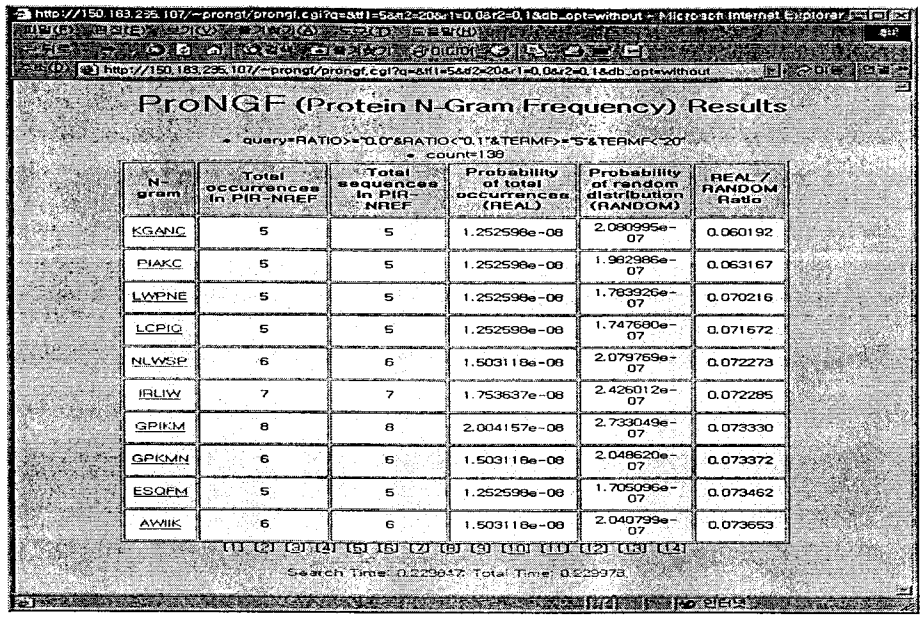


<figure 4-19> 검색 help/Example

검색 결과로는 질의로 N-Gram을 주고 검색에 B,Z,X를 포함시켰을 때 <figure 4-20>과 같이 검색질의어로 주어진 N-Gram 및 용어 빈도, 서열 빈도, 실제 통계, 이상 통계, 실제/이상 비율이 표시된다. 또 질의어로서 REAL/RANDOM RATIO 의 범위를 최소 0.0과 최대 0.1로 한정시키고, term frequency의 범위를 5이상 20 이하로 한정시킨 경우의 결과를 <figure 4-21>에서 볼 수 있다. REAL/RANDOM ratio 값에 따른 오름차순으로 정렬되어 있으며, 검색건수에 관계없이 최대 2000건까지 보여준다. 두 검색 결과 모두 PIR-NREF의 peptide match 결과가 링크되어 있다.



<figure 4-20> ProNGF 검색결과화면: 특정 N-Gram 질의어에 의한 검색



ProNGF (Protein N-Gram Frequency) Results

query=RATIO>="0.0"&RATIO<"0.1"&TERMF>="5"&TERMF<"20"

count=136

N-gram	Total occurrences in PIR-NREF	Total sequences in PIR-NREF	Probability of total occurrences (REAL)	Probability of random distribution (RANDOM)	REAL / RANDOM Ratio
KGANC	5	5	1.252598e-08	2.080995e-07	0.060192
PIAKC	5	5	1.252598e-08	1.982986e-07	0.063167
LWPNE	5	5	1.252598e-08	1.783326e-07	0.070216
LCPIG	5	5	1.252598e-08	1.747680e-07	0.071672
NLWSP	6	6	1.503118e-08	2.079769e-07	0.072273
IRLIW	7	7	1.753637e-08	2.426012e-07	0.072285
GPIKM	8	8	2.004157e-08	2.733049e-07	0.073330
GPKMN	6	6	1.503118e-08	2.048620e-07	0.073372
ESQFM	5	5	1.252598e-08	1.705096e-07	0.073462
AWIIK	6	6	1.503118e-08	2.040799e-07	0.073653

[1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14]

Search Time: 0.229847 Total Time: 0.229978

<figure 4-21> ProNGF 검색 결과 화면 : 출현빈도의 범위 및 실제 일어날 확률/임의로 발견될 확률의 범위에 따른 검색

(5) 검색 결과 분석

질의어로 단순히 N-Gram만을 주었을 경우의 한 예를 보면 질의어 ACDEF에 대해 20개의 단백질 서열이 검색되었다. 이 서열들은 REAL/RANDOM Ratio가 약 0.32 정도인 저빈도 서열 (희소 서열) 로서 그 중 4개의 hydroxylase, 1개의 monooxygenase에서 "GRACDEFL"의 보존 서열 (conserved sequence) 가 발견되었다. 반면 질의어가 AAAAA인 경우는 서열빈도가 37,313이고 REAL/RANDOM Ratio가 약 40인 다빈도 서열이며 나타나는 단백질의 종류도 천차만별이다. 이에 가설로서 "다빈도 서열은 구조적인 역할 (structural role)을 하는 부분이 많을 것이고, 저빈도 서열 (희소 서열) 은 기능적인 역할 (functional role)을 할 것이다"를 세울 수 있다.

또 질의어로 REAL/RANDOM RATIO 의 범위를 주었을 때의 결과를 보면 REAL/RANDOM RATIO의 범위가 최소 0.0, 최대 0.5일 경우 약 61만건이 검색되었다. Real/Random ratio << 1 일 경우에는 저빈도 서열로서 기능적 역할을 할 것이며, Real/Random ratio >> 1 일 경우에는 다빈도 서열로서 구조적 역할을 할 것임을 앞의 검색예에 비추어 가정할 수 있다.

이로써 PIR-NREF 데이터베이스에 대한 N-Gram 빈도 분석을 통하여 ProNGF 데이터베이스의 단백질의 구조 및 기능 예측에의 사용가능성을 탐구하였다.

다. 단백질 내의 특정 n-gram의 이차 구조 예측 : PDB 데이터베이스의 *α*-helix, *β*-strand 데이터에 대한 N-Gram 분석

(1) 개요

RCSB (Research Collaboratory for Structural Bioinformatics) 에서 운영하는 PDB (Protein Data Bank) 는 단백질과 핵산으로 이루어진 거대분자들의 3차원 구조 데이터를 처리하고 배포하기 위한 세계적인 단일 저장소이다. 단백질 서열 내의 특정 n-gram의 이차 구조를 예측하기 위하여, PDB의 단백질 서열 데이터만을 따로 추출하여 α-helix, β-strand[6] 데이터를 얻었다. α-helix, β-strand 데이터에 나타나는 n-gram들의 특징을 알아내기 위하여 n-gram 데이터베이스를 다음 4가지로 구성하였다.

- PDB 단백질 서열 데이터에서 추출한 n-gram 데이터베이스
- PDB 단백질 서열 중 α-helix 또는 β-strand 데이터에서만 추출한 n-gram 데이터베이스
- PDB 단백질 서열 중 α-helix에서만 추출한 n-gram 데이터베이스
- PDB 단백질 서열 중 β-strand 데이터에서만 추출한 n-gram 데이터베이스

위의 4가지 데이터베이스에서 나타나는 특정 n-gram의 성질(예: term frequency 등)을 비교하면 특정 2차 구조에서 주로 나타나는 n-gram들을 예측할 수 있다.

(2) 데이터베이스 설계

PDB는 2004년 3월 말 현재 24,785개의 구조를 포함하고 있다. 이를 기본으로 하여 새로이 만들어지는 N-Gram 빈도 데이터베이스는 Penta-Gram까지만 생각하더라도 최소 약 $245 = 7{,}000{,}000$ 개에 달하는 방대한 색인어 수를 자랑하게 된다. 따라서 올바른 데이터베이스 구조를 선택하지 않을 경우에는 사소한 잘못으로도 검색시스템의 성능에 큰 지장을 초래할 수 있다. 이러한 점을 고려하여 다음과 같이 검색시스템을 설계하였다.

---

6) CCBB의 김세훈 저 *α*-helix, *β*-strand 추출 프로그램 이용

○ 원시 자료 :

원시자료는 RCSB에서 공개적으로 배포하는 PDB 데이터베이스를 기반으로 한 단백질 서열 N-Gram 빈도 데이터베이스(ProNGF)를 사용하였다. PDB 데이터베이스의 데이터는 PDB의 고유한 포맷으로 되어 있는데 본 시스템에서는 이 중 서열 정보 (<figure 4-22>) 만을 추출하여 새로 단백질 서열 N-Gram 빈도 데이터베이스의 일부분으로 구성하였다. 또한 PDB 데이터베이스의 α-helix, β-strand 데이터 (<figure 4-23>) 로부터도 N-Gram을 추출하여 단백질 서열 N-Gram 빈도 데이터베이스의 나머지 부분을 구성하였다. PDB 포맷 파일 안의 단백질 서열 정보 및 PDB 데이터베이스의 α-helix, β-strand 데이터로부터 Uni-Gram부터 Penta-Gram까지의 색인어를 추출하여 각 N-Gram에 대한 TRM(Term Frequency), DOC(Sequence Frequency), PRR(Probability of Total Occurrences), PRI(Probability of Random Distribution), RAT(REAL/RANDOM)의 5가지 정보를 구하였다. (<figure 4-24>) 각 N-Gram을 ProNGF 시스템의 볼륨(Volume)으로 구성하였으며, 그 결과 ProNGF 검색 서비스에서는 20개의 볼륨[7]이 사용되고 있다.

```
SEQRES   1  154  MET VAL LEU SER GLU GLY GLU TRP GLN LEU VAL LEU HIS
SEQRES   2  154  VAL TRP ALA LYS VAL GLU ALA ASP VAL ALA GLY HIS GLY
SEQRES   3  154  GLN ASP ILE LEU ILE ARG LEU PHE LYS SER HIS PRO GLU
SEQRES   4  154  THR LEU GLU LYS PHE ASP ARG VAL LYS HIS LEU LYS THR
SEQRES   5  154  GLU ALA GLU MET LYS ALA SER GLU ASP LEU LYS LYS HIS
SEQRES   6  154  GLY VAL THR VAL LEU THR ALA LEU GLY ALA ILE LEU LYS
SEQRES   7  154  LYS LYS GLY HIS HIS GLU ALA GLU LEU LYS PRO LEU ALA
SEQRES   8  154  GLN SER HIS ALA THR LYS HIS LYS ILE PRO ILE LYS TYR
SEQRES   9  154  LEU GLU PHE ILE SER GLU ALA ILE ILE HIS VAL LEU HIS
SEQRES  10  154  SER ARG HIS PRO GLY ASN PHE GLY ALA ASP ALA GLN GLY
SEQRES  11  154  ALA MET ASN LYS ALA LEU GLU LEU PHE ARG LYS ASP ILE
SEQRES  12  154  ALA ALA LYS TYR LYS GLU LEU GLY TYR GLN GLY
```

<figure 4-22> PDB 데이터베이스의 단백질서열 데이터 (PDB 포맷)

7) 20개의 볼륨은 PDB 서열 데이터로부터 추출한 n-gram 데이터베이스의 볼륨 5개(uni~pentagram)에 PDB 단백질 서열 중 α-helix 또는 β-strand 데이터에서만 추출한 n-gram 데이터베이스의 볼륨 5개, PDB 단백질 서열 중 α-helix에서만 추출한 n-gram 데이터베이스의 볼륨 5개, PDB 단백질 서열 중 β-strand 데이터에서만 추출한 n-gram 데이터베이스의 볼륨 5개를 더한 것이다.

```
Filename : ./01/pdb101m.ent

Chain : (154)<129, 9>
ALPHA : EGEWQLVLHVWAKVEA [5-20]
ALPHA : VAGHGQDILIRLFKS [22-36]
ALPHA : PETLEK [38-43]
ALPHA : EAEMKA [53-58]
ALPHA : EDLKKHGVTVLTALGAILKK [60-79]
```

<figure 4-23> PDB 데이터베이스의 α-helix, β-strand 데이터

```
Total Tokens: 4131804
Total Sequences: 18805
#SEQ=AAAAA
#TRM=179
#DOC=63
#PRR=4.332248e-05
#PRI=2.040299e-05
#RAT=2.123340
```

<figure 4-24> PDB 데이터베이스의 단백질 서열중 α-helix데이터로 구축한 N-Gram 빈도 데이터베이스의 N-Gram 데이터 (Penta-Gram의 예)

○ 색인 범위 :

<table 4-8>에서 볼 수 있듯이, 각 항목을 ProNGF 시스템의 섹션으로 정의함으로써 해당 필드(Field)만으로도 검색이 가능하도록 하였다. 각 섹션은 N-Gram 서열(NGRAM), 용어 빈도(TERMF), 서열 빈도(SEQUF), 실제 통계(REALP), 이상 통계(IDEALP), 실제/이상 비율(RATIO)로 이루어진다.

○ 색인 방법 :

색인 방식은 주검색 섹션인 N-Gram 서열(NGRAM)에 대해 KRISTAL-2000의 INDEX_AS_IS 방식을 사용하였다.

○ 검색 모델 :

KRISTAL-2000 정보검색시스템은 불리안(Boolean) 검색모델 및 벡터공간(Vector Space) 검색모델을 제공한다. ProNGF 데이터베이스 시스템에서는 불리안 검색모델을 기본 검색모델로 채택하였으며, 자료 분석을 통해 ProNGF 데이터베이스에 가장 알맞도록 불리안 모델을 최적화하였다. 즉, 불리안 모델의 OR, NOT 연산자를 제외하고, AND 및 WITHIN 연산자만을 지원하도록 하였다. 이는 ProNGF의 자료가 문

서순위를 결정(document ranking)할 필요가 없는 속성을 가지고 있기 때문이며, 또한 자료의 특성상 일반 사용자에게는 OR검색의 필요가 없다고 판단했기 때문이다.

위와 같은 사항을 토대로 만든 KRISTAL-2000 스키마(Schema) 파일은 <table 4-8>과 같다. KRISTAL-2000에 기반한 검색 시스템은 데이터베이스의 대부분의 설정을 스키마 파일에 정의할 수 있도록 하고 있다. ProNGF 원시자료 구조에 따라 본 시스템에서는 원시자료를 N-Gram 서열(NGRAM), 용어 빈도(TERMF), 서열 빈도(SEQUF), 실제 통계(REALP), 이상 통계(IDEALP), 실제/이상 비율(RATIO)로 구분하였다. 섹션은 N-Gram 서열은 섹션 값 자체를 색인으로 삼았고, 나머지는 색인하지 않거나 보다 상세한 검색을 위해서 용어 빈도(TERMF) 및 실제/이상 비율(RATIO)를 INDEX_BY_NUMERIC 색인방식으로 색인하였다. 각 섹션의 색인 방식은 <table 4-9> 에서 볼 수 있다.

<table 4-8> 단백질 2차구조의 n-gram DB 검색을 위한 KRISTAL-2000 스키마 파일

```
// KRISTAL version
KRISTAL_VERSION="2000"
KRISTAL_DIRECTORY='/home/k2000/K2000'
DATABASE_DIRECTORY='/home/prongf/PDB_ab/VOLs'
DATABASE_GROUP_NAME="PDB_ALPHA_BETA"
// DATABASE_COMPRESS=TRUE

CREATE_SCHEMA
{
  SECTION_DEFINITION
  {
        (1)LABEL="N-Gram Sequence"          SECTION_NAME=NGRAM  INDEX_TYPE="INDEX_AS_IS",
        (2)LABEL="Term Frequency"           SECTION_NAME=TERMF  INDEX_TYPE="INDEX_AS_NUMERIC",
        (3)LABEL="Sequence Frequency"       SECTION_NAME=SEQUF  INDEX_TYPE="DO_NOT_INDEX",
        (4)LABEL="Real Statistics"          SECTION_NAME=REALP  INDEX_TYPE="DO_NOT_INDEX",
        (5)LABEL="Ideal Statistics"         SECTION_NAME=IDEAP  INDEX_TYPE="DO_NOT_INDEX",
        (6)LABEL="Real to Ideal Ratio"      SECTION_NAME=RATIO  INDEX_TYPE="INDEX_AS_NUMERIC"
  }
};
// 데이터베이스 생성
CREATE_DATABASE
{
// 데이터베이스 이름과 크기를 정의
        (1)     DATABASE_NAME=PDB1          DATABASE_SIZE=20,
        (2)     DATABASE_NAME=PDB2          DATABASE_SIZE=20,
        (3)     DATABASE_NAME=PDB3          DATABASE_SIZE=20,
        (4)     DATABASE_NAME=PDB4          DATABASE_SIZE=80,
        (5)     DATABASE_NAME=PDB5          DATABASE_SIZE=900,
        (6)     DATABASE_NAME=PDB_A1        DATABASE_SIZE=20,
        (7)     DATABASE_NAME=PDB_A2        DATABASE_SIZE=20,
        (8)     DATABASE_NAME=PDB_A3        DATABASE_SIZE=20,
        (9)     DATABASE_NAME=PDB_A4        DATABASE_SIZE=80,
        (10)    DATABASE_NAME=PDB_A5        DATABASE_SIZE=900,
```

<table 4-8> 단백질 2차구조의 n-gram DB 검색을 위한 KRISTAL-2000 스키마파일 (<table 4-8>의 계속)

```
        (11)    DATABASE_NAME=PDB_B1        DATABASE_SIZE=20,
        (12)    DATABASE_NAME=PDB_B2        DATABASE_SIZE=20,
        (13)    DATABASE_NAME=PDB_B3        DATABASE_SIZE=20,
        (14)    DATABASE_NAME=PDB_B4        DATABASE_SIZE=80,
        (15)    DATABASE_NAME=PDB_B5        DATABASE_SIZE=900,
        (16)    DATABASE_NAME=PDB_AB1       DATABASE_SIZE=20,
        (17)    DATABASE_NAME=PDB_AB2       DATABASE_SIZE=20,
        (18)    DATABASE_NAME=PDB_AB3       DATABASE_SIZE=20,
        (19)    DATABASE_NAME=PDB_AB4       DATABASE_SIZE=80,
        (20)    DATABASE_NAME=PDB_AB5       DATABASE_SIZE=900
};

DEFINE_DOCUMENT_STRUCTURE
{
  STRUCTURE_DEFINITION = PDB_AB_DOC_STRUCTURE {
        // 문서시작 태그 정의
        (1)TAG_NAME="#SEQ=" ACTION=COPY     SECTION_NAME=NGRAM   NEW_DOCUMENT_FLAG=TRUE,
        // 섹션 태그 및 액션 정의
        (2)TAG_NAME="#TRM=" ACTION=COPY     SECTION_NAME=TERMF,
        (3)TAG_NAME="#DOC=" ACTION=COPY     SECTION_NAME=SEQUF,
        (4)TAG_NAME="#PRR=" ACTION=COPY     SECTION_NAME=REALP,
        (5)TAG_NAME="#PRI=" ACTION=COPY     SECTION_NAME=IDEAP,
        (6)TAG_NAME="#RAT=" ACTION=COPY     SECTION_NAME=RATIO
  }
};
// 문서 그룹 정의
DEFINE_DOCUMENT_GROUP
{
        (1)     pdb1=('/home/prongf/PDB_ab/kst/0/01_monogram_.kst'),
        (2)     pdb2=('/home/prongf/PDB_ab/kst/0/02_digram___.kst'),
        (3)     pdb3=('/home/prongf/PDB_ab/kst/0/03_trigram__.kst'),
        (4)     pdb4=('/home/prongf/PDB_ab/kst/0/04_tetragram.kst'),
        (5)     pdb5=('/home/prongf/PDB_ab/kst/0/05_pentagram.kst'),
        (6)     pdb_a1=('/home/prongf/PDB_ab/kst/a/a_01_monogram_.kst'),
        (7)     pdb_a2=('/home/prongf/PDB_ab/kst/a/a_02_digram___.kst'),
        (8)     pdb_a3=('/home/prongf/PDB_ab/kst/a/a_03_trigram__.kst'),
        (9)     pdb_a4=('/home/prongf/PDB_ab/kst/a/a_04_tetragram.kst'),
        (10)    pdb_a5=('/home/prongf/PDB_ab/kst/a/a_05_pentagram.kst'),
        (11)    pdb_b1=('/home/prongf/PDB_ab/kst/b/b_01_monogram_.kst'),
        (12)    pdb_b2=('/home/prongf/PDB_ab/kst/b/b_02_digram___.kst'),
        (13)    pdb_b3=('/home/prongf/PDB_ab/kst/b/b_03_trigram__.kst'),
        (14)    pdb_b4=('/home/prongf/PDB_ab/kst/b/b_04_tetragram.kst'),
        (15)    pdb_b5=('/home/prongf/PDB_ab/kst/b/b_05_pentagram.kst'),
        (16)    pdb_ab1=('/home/prongf/PDB_ab/kst/ab/ab_01_monogram_.kst'),
        (17)    pdb_ab2=('/home/prongf/PDB_ab/kst/ab/ab_02_digram___.kst'),
        (18)    pdb_ab3=('/home/prongf/PDB_ab/kst/ab/ab_03_trigram__.kst'),
        (19)    pdb_ab4=('/home/prongf/PDB_ab/kst/ab/ab_04_tetragram.kst'),
        (20)    pdb_ab5=('/home/prongf/PDB_ab/kst/ab/ab_05_pentagram.kst')
};
```

<table 4-8> 단백질 2차구조의 n-gram DB 검색을 위한 KRISTAL-2000 스키마파일 (<table 4-8>의 계속)

```
// 문서 적재
LOAD_DATABASE
{
        (1)     FROM=pdb1       TO=PDB1         WITH=PDB_AB_DOC_STRUCTURE,
        (2)     FROM=pdb2       TO=PDB2         WITH=PDB_AB_DOC_STRUCTURE,
        (3)     FROM=pdb3       TO=PDB3         WITH=PDB_AB_DOC_STRUCTURE,
        (4)     FROM=pdb4       TO=PDB4         WITH=PDB_AB_DOC_STRUCTURE,
        (5)     FROM=pdb5       TO=PDB5         WITH=PDB_AB_DOC_STRUCTURE,
        (6)     FROM=pdb_a1     TO=PDB_A1       WITH=PDB_AB_DOC_STRUCTURE,
        (7)     FROM=pdb_a2     TO=PDB_A2       WITH=PDB_AB_DOC_STRUCTURE,
        (8)     FROM=pdb_a3     TO=PDB_A3       WITH=PDB_AB_DOC_STRUCTURE,
        (9)     FROM=pdb_a4     TO=PDB_A4       WITH=PDB_AB_DOC_STRUCTURE,
        (10)    FROM=pdb_a5     TO=PDB_A5       WITH=PDB_AB_DOC_STRUCTURE,
        (11)    FROM=pdb_b1     TO=PDB_B1       WITH=PDB_AB_DOC_STRUCTURE,
        (12)    FROM=pdb_b2     TO=PDB_B2       WITH=PDB_AB_DOC_STRUCTURE,
        (13)    FROM=pdb_b3     TO=PDB_B3       WITH=PDB_AB_DOC_STRUCTURE,
        (14)    FROM=pdb_b4     TO=PDB_B4       WITH=PDB_AB_DOC_STRUCTURE,
        (15)    FROM=pdb_b5     TO=PDB_B5       WITH=PDB_AB_DOC_STRUCTURE,
        (16)    FROM=pdb_ab1    TO=PDB_AB1      WITH=PDB_AB_DOC_STRUCTURE,
        (17)    FROM=pdb_ab2    TO=PDB_AB2      WITH=PDB_AB_DOC_STRUCTURE,
        (18)    FROM=pdb_ab3    TO=PDB_AB3      WITH=PDB_AB_DOC_STRUCTURE,
        (19)    FROM=pdb_ab4    TO=PDB_AB4      WITH=PDB_AB_DOC_STRUCTURE,
        (20)    FROM=pdb_ab5    TO=PDB_AB5      WITH=PDB_AB_DOC_STRUCTURE
};

END
```

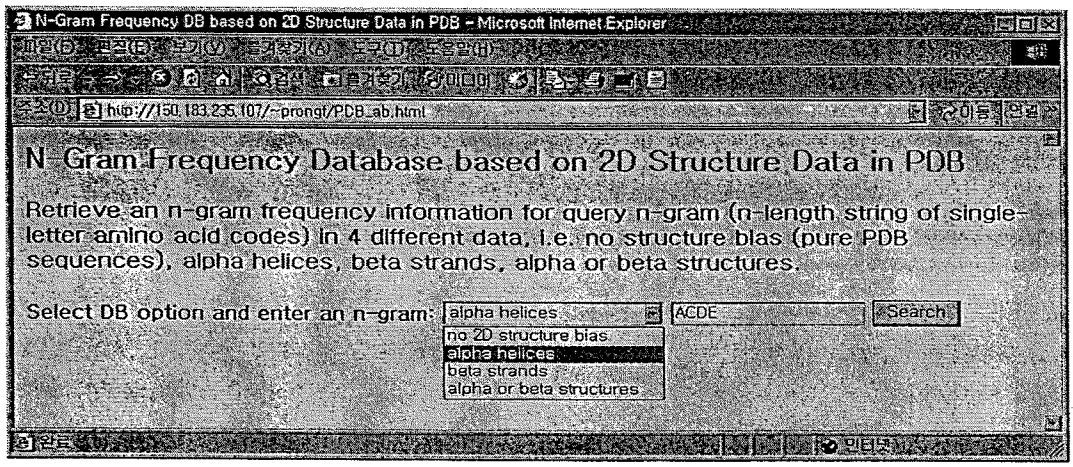
<table 4-9> 각 섹션별 색인 정보

섹션	색인 방식	색인 방법	비고
NGRAM	INDEX_AS_IS	섹션별 색인	PRIMARY KEY로 접근, 주검색 섹션
TERMF	INDEX_AS_NUMERIC	숫자 색인	용어 빈도
SEQUF	DO_NOT_INDEX	색인하지 않음	서열 빈도
REALP	DO_NOT_INDEX	색인하지 않음	실제 통계
IDEALP	DO_NOT_INDEX	색인하지 않음	이상 통계
RATIO	INDEX_AS_NUMERIC	숫자 색인	실제/이상 비율

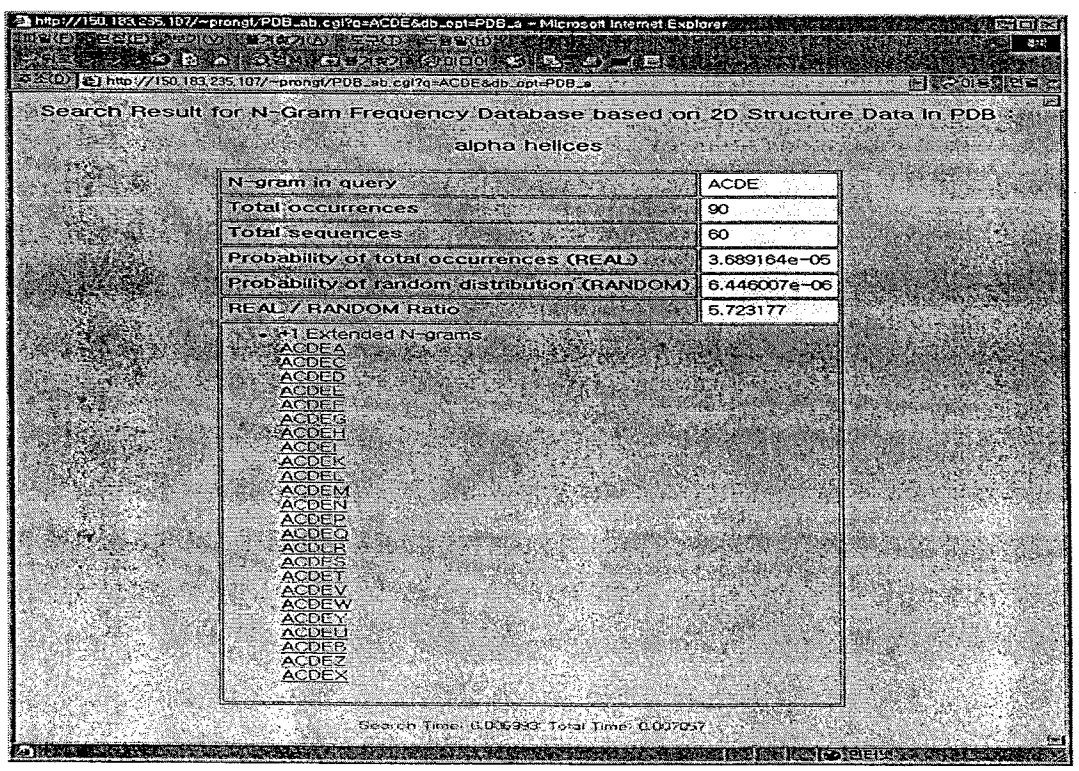
(3) 웹 검색 인터페이스

<figure 4-25>에서는 단백질 2차구조의 n-gram 빈도 데이터베이스 검색서비스의 메인화면을 보여주고 있다. 서로 다른 4가지의 2차원 구조 데이터, 즉 PDB 서열 데이터, PDB 단

백질 서열 중 α-helix 데이터, PDB 단백질 서열 중 β-strand 데이터, PDB 단백질 서열 중 α-helix 또는 β-strand 데이터에서 1가지를 검색대상 데이터베이스로 선택할 수 있으며, 질의로 n-gram을 입력받는다. 검색 결과에서는 <figure 4-26>과 같이 검색질의어로 주어진 N-Gram 및 용어 빈도, 서열 빈도, 실제 통계, 이상 통계, 실제/이상 비율이 표시된다. 또 검색한 N-Gram의 끝에 아미노산 1개를 더 추가한 "+1 Extended N-Grams"에 대한 검색도 링크시켜서 검색한 N-Gram의 확장성 여부를 알아볼 수 있도록 하였다.



<figure 4-25> 단백질 2차구조의 n-gram DB 검색 초기 화면



<figure 4-26> 단백질 2차구조의 N-Gram DB 검색 결과 화면

(4) 검색 결과 분석

Tetra-Gram 과 Penta-Gram에서 같은 종류의 아미노산으로 이루어진 것들 (예: AAAA)의 데이터베이스 종류에 따른 빈도(TF: Term Frequency)를 찾아 보았다. 몇몇 경우(예: HHHHH)를 제외하고 TF(PDB) > TF(α 또는 β 구조) > TF(α 구조) > TF(β 구조)의 순으로 빈도 수가 높게 나왔다. 대부분의 경우에서 TF(α 또는 β 구조) = TF(α 구조) +TF(β 구조) 였다. 이는 하나의 서열 조각은 동시에 α-helix 와 β-strand 구조를 가질 가능성이 거의 없다는 것을 말해 준다. 나선과 평면의 성질을 동시에 가지는 경우는 거의 없을 것이므로, 이는 당연한 결과인 듯이 보인다.

라. 결론

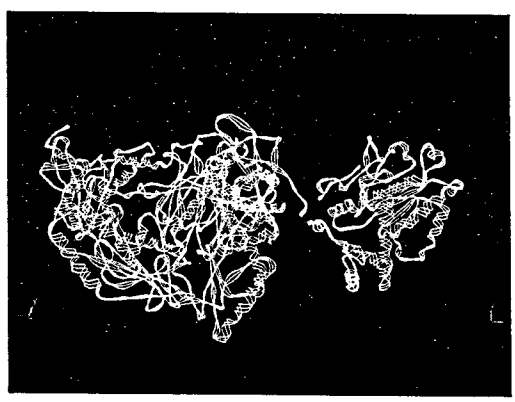
단백질 서열 N-Gram 빈도 데이터베이스 시스템(Protein N-Gram Frequency, 이하 약칭 ProNGF)이란 단백질 서열 내의 길이가 N인 연속된 아미노산 서열인 N-Gram이 출현하는 빈도(Frequency)를 단백질 서열 데이터베이스에 대해서 측정하는 것이다. 단백질 서열에서 발견되는 특정 N-Gram의 빈도, 2차 구조, 출현 단백질 정보 등을 알아봄으로써 그 N-Gram의 기능을 예측할 수 있다. PIR-NREF 데이터베이스에 대하여 저빈도 N-Gram과 고빈도 N-Gram의 예를 찾아보고 그 특징들을 연구하였다. 이에 따라 저빈도 N-Gram 은 단백질의 기능 예측에, 또한 고빈도 N-Gram 은 단백질의 2차 구조 예측에 적용될 수 있을 가능성을 탐구하였다. 또한 ProNGF 데이터베이스를 단백질 내의 특정 n-gram의 이차 구조 예측에 활용하기 위하여 PDB 데이터베이스의 α-helix, β-strand 데이터에 대한 N-Gram 빈도 데이터베이스를 구축하여 분석하였다. 분석은 진행 중이며, 앞으로 검색 시 서로 다른 4가지 N-Gram 정보, 즉 PDB 서열 데이터, PDB 단백질 서열 중 α-helix 또는 β-strand 데이터, PDB 단백질 서열 중 α-helix 데이터, PDB 단백질 서열 중 β-strand 데이터에서의 각각의 N-Gram 정보들을 모두 검색하여 한 화면에 보여 줄 예정이다. 그럼으로써 특정 N-Gram이 α-helix 또는 β-strand에 흔히 존재하는지의 여부를 한눈에 볼 수 있을 것이다. 또한 가능한 한 PDB 데이터베이스의 서열에서 나타나는 모든 N-Gram에 대하여 2차 구조에 나타나는 빈도를 분석하여, 통계적 분포를 탐구할 것이다.

제 5 절 3차원 비교가시화 소프트웨어 개발

1. 소개

유전체 분석을 위해서는 다양한 기능 분석과 연구가 필요하지만, 그 기본이 되는 것은 단백질/DNA 구조이다. 이를 얻는 방법은 수치적인 실험을 통해서나 NMR 사진을 통해서 얻은 각 원자들의 중심좌표를 추출하는 것을 통해서인데, 이것을 수치 데이터로만 보는 것보다는 실제 3차원 공간상에 표현을 해 주는 것이 연구자들에게 도움이 될 것이다. 이를 위해서 RasMol, MoE, AmiraMol과 같은 다양한 3차원 가시화 툴이 개발되어 있다.

Dummy Analyzer는 이러한 툴들 중 하나로 시각/통계적인 비교가시화를 통해 연구자들의 단백질/DNA 파일구조 분석을 돕기 위한 툴이다. 기본적인 wireframe, ball and stick, sticks, CPK, backbone, strand 그리고 ribbon 스타일로 분자구조를 시각적으로 표현하는 것 이외에도 2)에서 언급할 기능들을 첨가하였다. 아래의 그림은 실제 Dummy Analyzerr에서 1hnv와 1aua를 불러들여서 strand, ribbon 스타일로 비교할 수 있도록 분자구조를 표현한 것을 나타낸다.



<figure 5-1> Strand modeling

<figure 5-2> Ribbon modeling

2. 개발환경

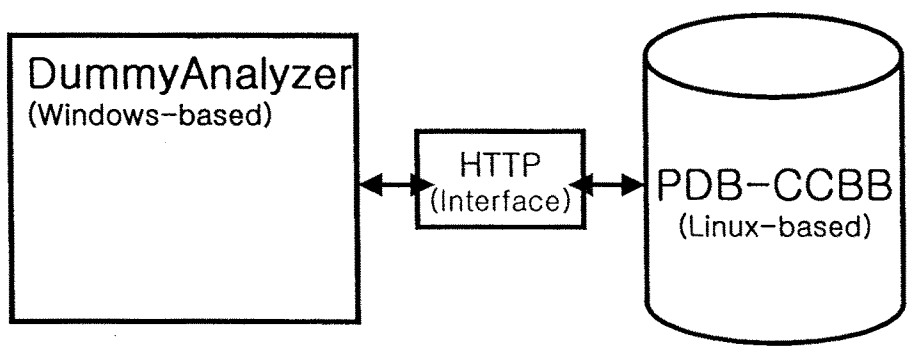
이 프로그램은 Microsoft Windows 2000 Professional Edition(with service pack 3) 환경과 SGI OpenGL spec 1.3, 그리고 MFC(Microsoft Foundation Class) ver 4.2를 기반으로 개발되었다. Microsoft Windows 2000, XP를 비롯한 Windows NT계열의 모든 운영체제에서 실행 가능하며 OpenGL을 하드웨어적으로 지원하는

그래픽카드를 장착한 시스템에서는 좀더 효과적으로 프로그램을 실행시킬 수 있다.

3. 구현 결과

가. PDB-CCBB와의 연동

PDB 파일을 import할 때 기존에는 Local 하드디스크에 있는 파일들만 가능했으나, 2003년 3월말에 만들어진 PDB-CCBB 검색시스템이 개발이 되었고, Dummy Analyzer 자체 인터넷 브라우저를 내장하여, Dummy Analyzer와 PDB-CCBB 검색 시스템간의 연계가 가능해졌다. 그러므로 인터넷이 연결된 환경이라면 어디서든지 PDB 파일을 검색해서 찾을 수 있게 되었다.



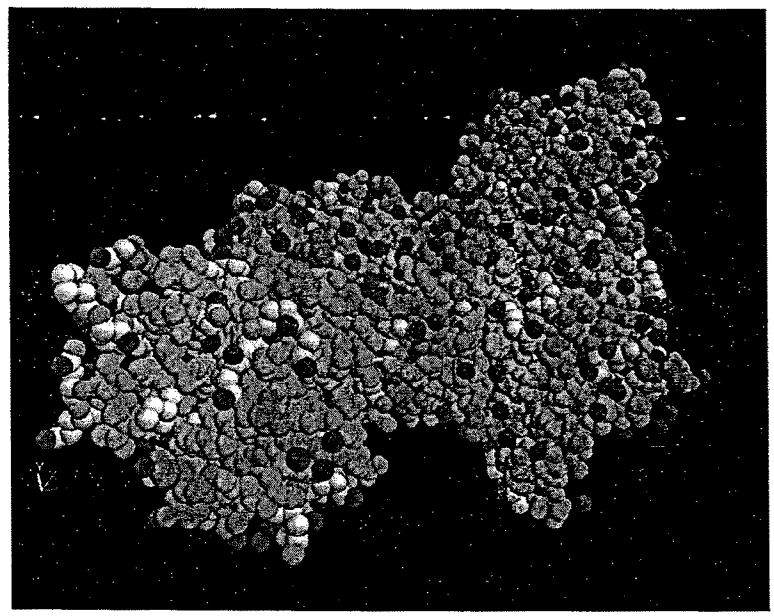
<figure 5-3> Dummy Analyzer와 PDB-CCBB와의 연동

그림에서처럼 서로 다른 운영체제 환경(Dummy Analyzer - Windows-based, PDB-CCBB - Linux-based)이라 검색 시스템을 embed할 수 없어 HTTP(HyperText Transfer Protocol)라고 불리는 표준 프로토콜을 활용, 다른 운영체제에서 동작하는 어떤 프로그램도 검색 시스템을 이용할 수 있도록 하여 시스템의 유연성을 높였다.

나. 단백질의 2차 구조 표현

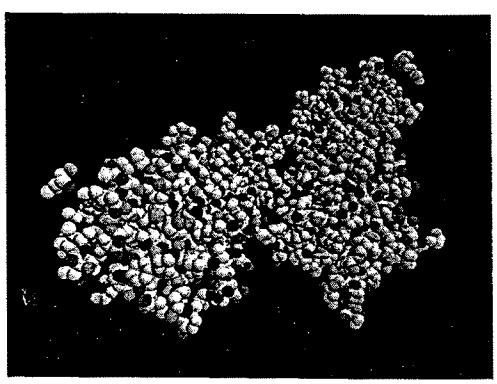
Dummy Analyzer에서는 단백질의 2차 구조를 표현하고, PDB(Brookhaven Protein Data Bank file format) 파일에서 그 구성 성분만을 추출할 수 있는 기능을 첨가하였다. 시각적인 요소에서는 크게 전체 구조에서 색깔로 alpha helix, beta

sheet의 영역을 표시할 수 있도록 하는 것과, alpha helix 또는 beta sheet만 별도로 표현하도록 하는 것으로 사용자가 쉽게 분자 구조를 파악할 수 있도록 하였다. Dummy Analyzer로 불러들인 파일이 단백질일 경우에는 alpha helix 또는 beta sheet만을 별도로 PDB format으로 export가 가능해서 Dummy Analyzer 뿐만 아니라 다른 소프트웨어에서도 자유롭게 사용이 가능하다.

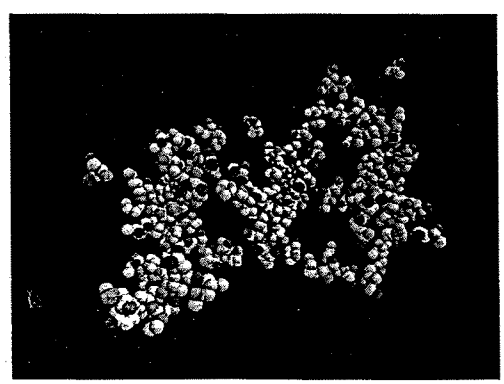


<figure 5-4> 1jix에서의 alpha helix, beta sheet 부분 표현

위의 그림은 1jix에서 alpha helix와 beta sheet의 위치를 색깔로 표현한 것이다. 녹색은 alpha helix, 하늘색은 beta sheet을 나타내며, 그 이외의 것은 물분자 또는 loop구조를 나타낸다. 이 그림을 통해 개략적으로 2차 구조의 구성비율을 짐작할 수 있다.



<figure 5-5> 1jix의 alpha helix



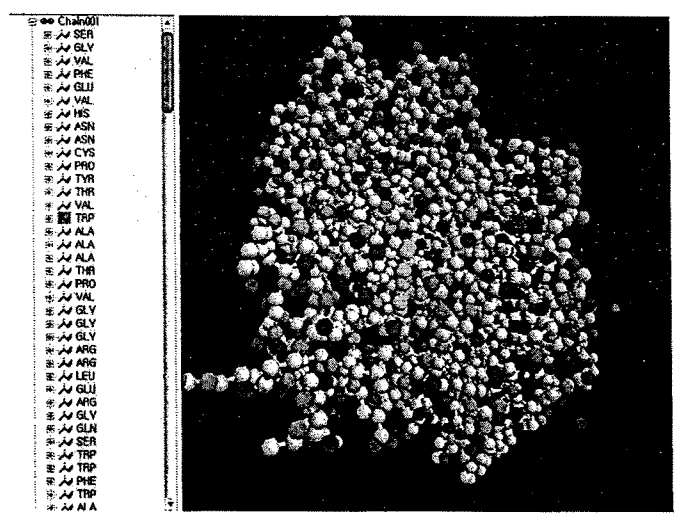
<figure 5-6> 1jix의 beta sheet

위의 그림은 1jix에서 alpha helix(좌)와 beta sheet(우)의 구조만 별도로 표현한 것을 나타낸다. 이 이외의 것들에 대해서는 옵션을 줌으로써 pruning이 가능하며, 이 내용을 별도의 PDB 파일로 저장이 가능하게 된다.

다. Treeview

Dummy Analyzer는 단백질/DNA 분자의 전체 구조를 3차원 공간에 화면으로만 보여줄 뿐만 아니라, 디스크-디렉토리(또는 폴더)의 계층구조를 표현할 때 주로 쓰이는 treeview 형태를 지원한다. 이를 통해 단백질/DNA을 이루는 원자 또는 원자단이 속한 그룹이 실제 3차원 공간상에 어떤 위치에 존재하는지 파악할 수 있도록 하였다.

Root node에는 단백질/DNA 정보를 가지는 파일을 위치시켰고, 그 아래 node에는 단백질/DNA을 이루는 chain을 표현하고, 그 아래 node에는 원자단을 표시하였으며, 가장 아래 node에는 원자 정보를 표시하였다. 마우스 커서로 각 node를 클릭하면 노란색으로 우측 3차원 공간 화면에 해당 node를 표현해 주게 된다.



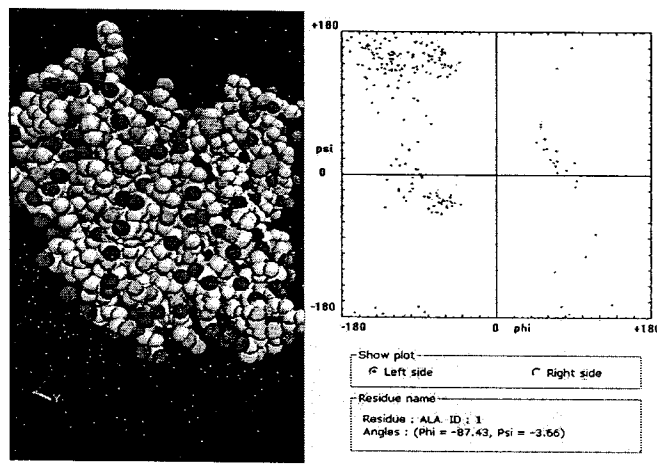
<figure 5-7> Treeview

위의 그림은 treeview를 이용해서 분자의 일부분을 선택한 모습이다.

라. Ramachandran Plot

Dummy Analyzer는 단백질 분자의 2차 구조 분포 파악을 위해 Ramachandran Plot을 쓸 수 있는 기능을 제공한다. 이를 통해 3차원 공간상의 부분을 지정하고 그 곳에 해당하는 원자단의 정보를 얻을 수 있다.또한 분포도 파악을 통해 구성 비율을 짐작할 수 있다.

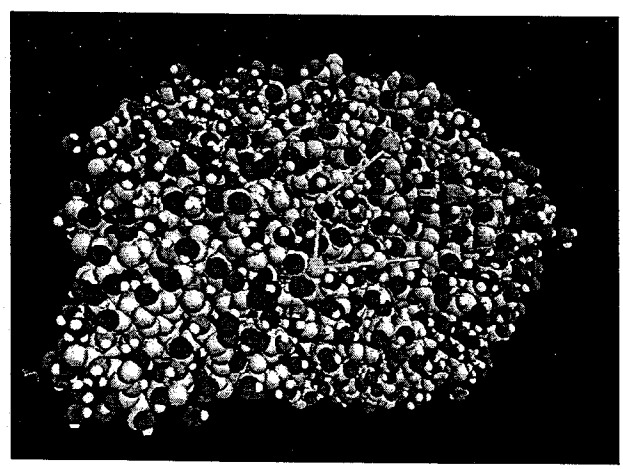
아래의 그림은 Ramachandran Plot을 이용해서 원자단 하나를 지정한 모습을 나타낸다.



<figure 5-8> Ramachandran Plot

마. Torsion angle

제한된 구조 표현모드(Ball and stick, CPK)에서 torsion angle을 구할 수 있는 방법을 제공한다. 4개의 원자를 지정해 주면, 그것들이 이루는 torsion angle값을 계산해서 그 결과를 사용자에게 알려준다. 아래의 그림은 torsion angle값을 구하는 모습을 보여준다.



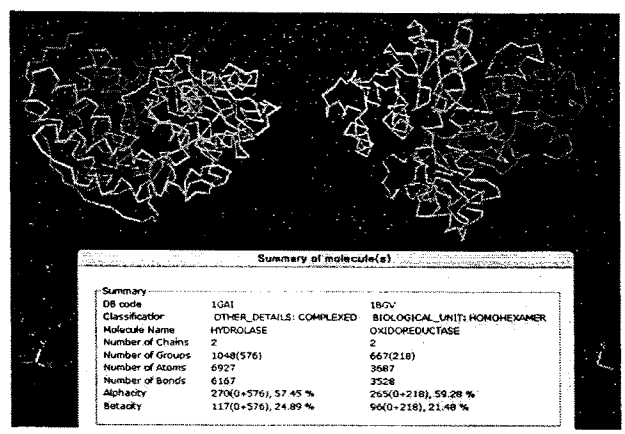
<figure 5-9> Torsion angle 계산

Torsion angle값은 좌측 하단에 표시하도록 하였다.

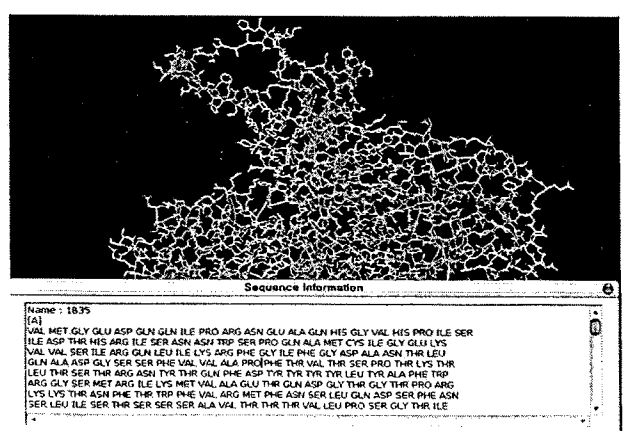
바. 그 이외의 기능

화면을 bitmap 이미지로 출력할 수 있는 기능과 분자데이터 관련 정보에 대한 요약, 서열구조를 텍스트로 표현하는 기능을 제공하고 있다. 아래의 그림은 각각, 분자데이터 관련 정보에 대한 요약, 서열구조를 텍스트로 표현하는 기능을 나타낸다.

이를 통해 두개의 서로 다른 분자데이터를 비교, 분석 및 가시화를 할 수 있는 환경을 구축하였다.



<figure 5-10> 두개의 분자 데이터의 비교 분석. 이를 통해 두 분자데이터의 특성을 시각적인 데이터와 같이 확인해 볼 수 있다.



<figure 5-11> 서열정보 출력

4. 벤치마킹

가. 개요

Dummy Analyzer의 성능이 어느 정도 되는지 여부를 판별하기 위해 현재 가장 빠르게 이미지를 생성해주는 RasMol과 비교를 하였다. OpenGL을 기반으로 한 프로그램 중에서는 가장 빠르게 이미지를 생성해 주는 Dummy Analyzer의 성능을 검증해 보고자 RasMol의 소스를 일부 수정, 성능 체크를 할 수 있도록 하였다.

나. Bottleneck

Dummy Analyzer와 RasMol은 다소 다른 렌더링 환경을 제공한다. Dummy Analyzer의 경우에는 범용 그래픽 라이브러리인 OpenGL을 사용하는 반면, RasMol은 GDC(Graphics Device Context)를 직접 사용하는 방식(소위 RasMol Engine이라고 불리는 독자적인 렌더링 엔진을 가지고 있다)을 취하고 있다. 그렇기 때문에 각각의 프로그램은 다음의 조건일 때 bottleneck이 발생한다.

① 원자의 개수가 많을 경우 : 많은 원자들을 그려야 하므로 각각의 원자를 그 릴 때 폴리곤을 이용하는 Dummy Analyzer가 성능에 제약을 받게 된다. 왜냐하면 원자의 개수만큼 폴리곤의 수도 비례해서 증가를 하기 때문이다.

② 프로그램의 해상도가 높을 경우 : 많은 점들을 찍어내야 하므로 RasMol이 불리하게 된다. 한편, Dummy Analyzer의 경우에는 하드웨어 가속기능이 점을 찍는 부분에서 효과적으로 지원을 하기 때문에 해상도의 크기에 그렇게 크게 영향을 받지 않는다.

본 벤치마킹에서는 이 두개의 bottleneck을 체크하기 위해서 테스트 셋을 4와 같이 설정하였고, 결과는 원자 개수에 대해서 표현을 하였다.

다. 테스트 환경

테스트를 위해 제공된 환경은 다음과 같다.

① OS : Microsoft Windows 2000 Professional
② CPU : P4 1.6A
③ Memory : 512MB
④ 그래픽카드 : GeForce3 Ti 500

라. 테스트 셋

(1) 데이터

대체로 원자의 개수에 비례하여 성능이 나빠지므로 데이터 셋의 기준을 원자개수 1000개 이하, 10000개 이하, 10000개 이상으로 나누어 테스트를 실시하였다.

(2) 프로그램 설정

해상도에 프로그램의 성능이 영향을 받는지 판단하기 위해 400 * 400, 800 * 600 해상도 모드에서 세 프로그램의 성능을 체크하였다.

(3) 데이터 로딩시간 측정

각각의 데이터 셋에 대해서 평균적인 데이터 로딩시간을 측정하였다.

(4) 구조 표현모드 변경

Wireframe, stick, ball and stick, CPK, backbone에 대해서 각각 속도 측정을 하였다.

마. 테스트 결과

테스트 결과는 아래와 같다. 참고로 로딩시간은 작으면 작을수록 좋으며, frame rate(fps)는 크면 클수록 좋은 것이다. Frame rate는 1초당 몇 번의 화면을 바꿀 수

있는지를 나타내는 것인데, 이 값이 크면 클수록 움직임이 좀더 자연스런 화면을 만들 수 있게 되므로 성능 측정에 중요한 요소가 된다.

(1) 원자의 개수가 1,000개 이하일 때

해상도	400 * 400		800 * 600	
프로그램명	RasMol	Dummy Analyzer	RasMol	Dummy Analyzer
로딩시간(ms)	132.8	72.2	132.8	72.2
Wireframe(fps)	166.7	1000	75.054	1000
Sticks(fps)	65.682	117.382	30.434	116.184
Ball and stick(fps)	78.276	121.874	37.034	120.146
CPK(fps)	47.33	180.05	15.428	176.182
Backbone(fps)	116.178	740	50.562	1000

(2)원자의 개수가 10,000개 이하일 때

해상도	400 * 400		800 * 600	
프로그램명	RasMol	Dummy Analyzer	RasMol	Dummy Analyzer
로딩시간(ms)	388.6	461.4	388.6	461.4
Wireframe(fps)	88.888	241.666	43.506	305.554
Sticks(fps)	17.17	10.422	7.882	10.23
Ball and stick(fps)	30.708	9.654	13.3	9.864
CPK(fps)	13.204	16.244	6.772	12.806
Backbone(fps)	48.418	76.388	28.039	94.34

(3) 원자의 개수가 10,000개 이상일 때

해상도	400 * 400		800 * 600	
프로그램명	RasMol	Dummy Analyzer	RasMol	Dummy Analyzer
로딩시간(ms)	2602.5	1834	2602.5	1834
Wireframe(fps)	18.385	37.855	8.335	30.4
Sticks(fps)	2.765	1.555	0.775	1.13
Ball and stick(fps)	3.765	1.005	1.26	1.15
CPK(fps)	2.49	3.08	1.125	1.105
Backbone(fps)	12.8	10.795	4.915	13.105

바. 결과해석

두 프로그램의 렌더링 구조 차이 때문에 두 프로그램이 강점을 갖는 부분이 차이가 두드러지게 났다. RasMol의 경우에는 해상도가 낮고 원자의 개수가 많을때 Dummy Analyzer에 비해서 비교우위를 가졌다. 한편 Dummy Analyzer는 해상도에는 상관없이 원자의 개수가 적을 때 RasMol에 비해 비교우위를 가졌다. 그 이외의 부분에서는 엇비슷한 성능을 나타내었으며, 해상도가 높은 부분에서는 비교적 Dummy Analyzer가 좋은 성능을 나타내었다. 데이터 로딩하는 시간은 Dummy Analyzer가 좀더 빨랐는데, 저 수치는 millisecond단위로 나타낸 것이고 실제로는 1~2초 이내에 PDB 데이터는 다 메모리 상에 올릴 수 있으므로 이것의 차이는 크게 의미를 둘 필요는 없으며, 두 소프트웨어가 빠르게 데이터를 로딩한다고 판단을 해도 무방하다.

이를 통해 Dummy Analyzer는 같은 OpenGL기반의 소프트웨어보다 빠르고, 다른 엔진을 이용하는 RasMol에 필적하는 성능을 가졌다고 볼 수 있다. 하드웨어 성능에 제약을 받지 않는 RasMol에 비해서 Dummy Analyzer가 우위를 가질 수 있다면 계속적으로 비약적인 성능개선을 이루고 있는 하드웨어의 도움을 받아 더 우수한 성능을 내는 프로그램이 될 수 있다는 것이다.

5. 결론

3차원 비교가시화 소프트웨어는 기본적인 가시화 기능 이외에 비교가시화 및 데이터 분석 부분을 첨가하였고, HTTP를 활용한 검색 시스템과의 연동을 하여 보다 이용자가 쉽게 PDB 데이터를 쓸 수 있도록 편의를 제공하였다.

제 6 절 국내 유전체 정보센터 인프라 고도화

1. 대용량 계산용 시스템 구축

가. 연구의 필요성

○ KISTI에서 수행중인 유전자/단백질 DB 구축에 연계한 시스템의 확대 발전 필요

○ 시시각각으로 증대되는 해외 DB에 맞춘 국내 DB의 지속적인 업그레이드를 위한 시스템의 유지보수필요

○ 생물정보 저장 및 해석에 필요한 시스템의 도입 및 관리 필요

○ 국가유전체정보센터의 기능을 확대 보완할 필요성 대두

○ 실질적인 생물정보시스템 연구사업 담당기관으로서의 능력 배양을 위한 연구 사업 기능 수행 필요

○ 슈퍼컴퓨팅 자원의 효과적인 활용을 위한 생물정보학 관련 연구 유도 및 응용기술의 관련 기관 및 산업체에 제공

나. 연구의 목적

○ 생명공학분야의 원활한 연구활동과 이에 필요한 신속, 정확한 자료처리 결과를 얻기 위해 필요한 최첨단 시스템의 구성과 제반 운영환경 조성

○ Bioscience/Biotechnology와 관련된 해외 주요 공공 Database와의 연계체계 수립 및 자체 Database 구축

○ 첨단 Bioscience/Biotechnology와 관련된 포괄적인 컴퓨팅 서비스 지원체제 구축

○ Sequence, Genomics, Proteomics, Physiomics 등의 바이오인포매틱스 연구를 주도하기 위한 입지 조성과 시스템 기술환경의 단계적 증강을 위한 기반환경 조성 및 관련기술의 국제 경쟁력확보와 국내 Bioscience/Biotechnology 관련 기술 선도를 위한 환경 구축

○ Bioinformatics 분야의 연구주제를 적절히 지원할 수 있는 클러스터링 등의 개념에 근거한 대규모 확장 가능한 시스템 지원 환경 조성

2. 바이오인포매틱스 시스템 지원

가. SMP Cluster 시스템 도입 및 설치

○ 활용방안 수립 및 기종 선정 지원

○ 하드웨어와 소프트웨어 설치, 품목 및 안정성 승인시험

(1) 도입 목적

○ 생명공학분야의 원활한 연구활동과 이에 필요한 신속, 정확한 자료처리 결과를 얻기 위해 필요한 최첨단 시스템의 구성과 제반 운영환경 조성

○ Bioscience/Biotechnology와 관련된 해외 주요 공공 Database와의 연계체계 수립 및 자체 Database 구축

○ 첨단 Bioscience/Biotechnology와 관련된 포괄적인 컴퓨팅 서비스 지원체제 구축

○ Sequence, Genomics, Proteomics, Physiomics 등의 바이오인포매틱스 연구를 주도하기 위한 입지 조성과 시스템 기술환경의 단계적 증강을 위한 기반환경 조성 및 관련기술의 국제 경쟁력확보와 국내 Bioscience/ Biotechnology 관련 기술 선도를 위한 환경 구축

○ Bioinformatics 분야의 연구주제를 적절히 지원할 수 있는 클러스터링 등의 개념에 근거한 대규모 확장 가능한 시스템 지원 환경 조성

(2) 도입 및 설치 일정

○ <table 6-1> 및 <table 6-2>에 나타난바와 같이 2001년 하반기부터 국내 관련 전문가로 바이오인포매틱스 기반 시스템 도입 위원회를 구성하여 4개 업체(Compaq, HP, IBM, SUN)에서 제안한 시스템을 2차례의 시스템 선정 과정을 통해 최종적으로

Compaq사의 시스템으로 결정하였다.

○ 2002년 6월말경 Compaq사의 SMP(Symmetric Multi-Processor) Cluster 시스템인 SC45를 선정하여 11월초부터 WEB을 통해서 일반 사용자에게 DB 서비스를 하였다.

(3) 시스템 사양 및 구성도

○ 하드웨어 사양 및 시스템 구성도

- 도입된 SMP 클러스터 시스템은 <table 4-3> 및 <figure 4-1>과 같으며 Front-End의 역할을 담당하는 DS20E 시스템과 WEB/FTP 서버, 개발서버, DB서버, 백업서버, 계산노드들로 구성되어있다.

- DS20E 시스템은 사용자가 계산노드로 접근하기 위한 로그인 서버의 역할을 담당하고 있다.

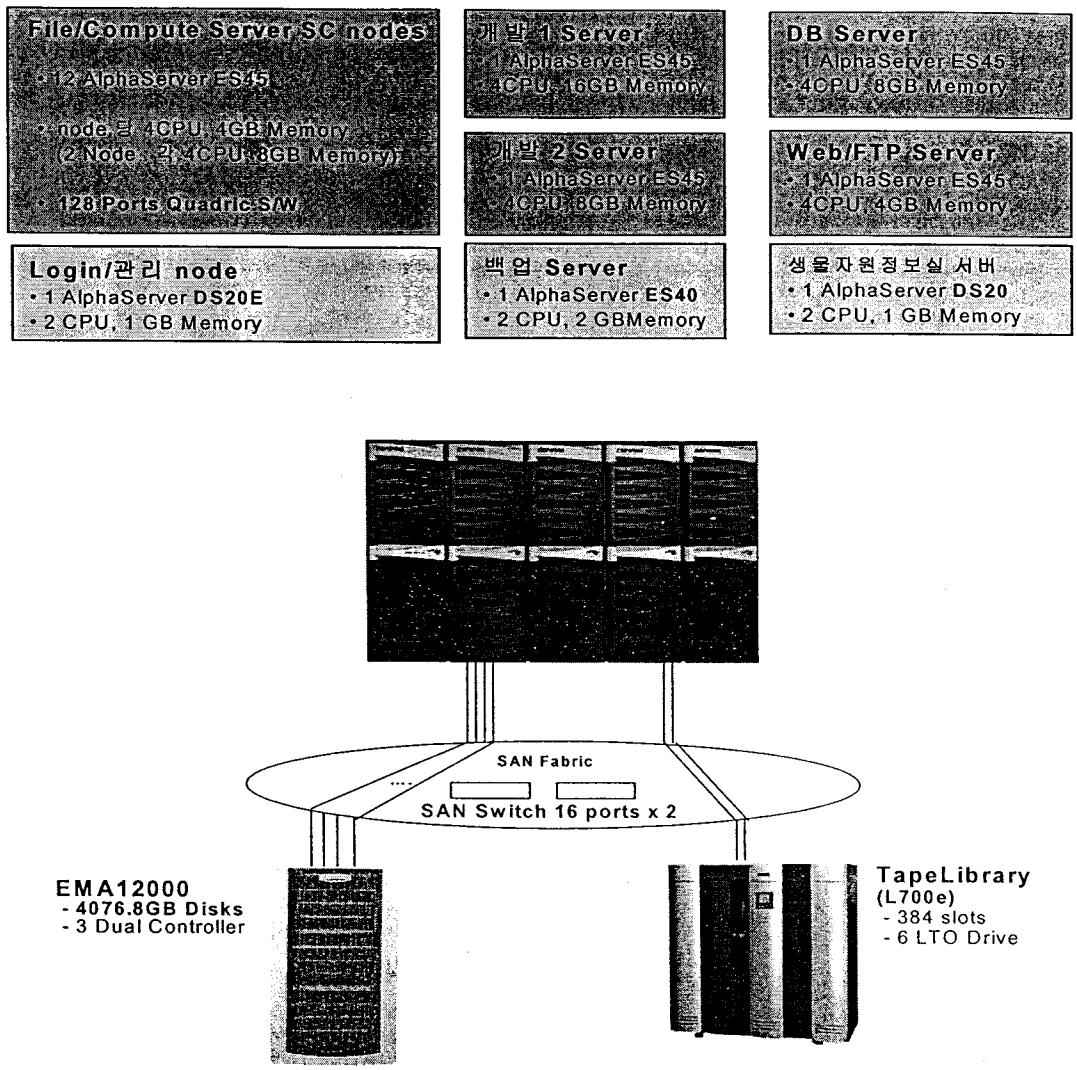
- 노드 0, 노드 1은 파일서버 역할과 계산노드 역할을 동시에 수행하도록 구성되어져 있으며 이 노드들의 Alpha CPU는 4개씩, 메모리는 각 8GB 이다. 그 외 노드 2부터 노드11까지의 노드들은 각각 Alpha CPU 4개씩과 메모리 4GB로 구성되었다.

- 각 시스템들은 독립적인 내장 디스크를 가지고 있으며, 또한 모든 노드에서 SAN을 이용하여 EMA 12000을 사용하고 있다.

- SMP 클러스터의 외부연결 네트워크는 기본적으로 Gigabit Ethernet으로 연결되어 있고, 클러스터 내부 네트워크는 Quadric Switch(200MB/sec)를 이용한다.

<table 6-1> 바이오인포매틱스 기반 시스템 하드웨어 사양

모델	CPU 수	메모리	이론성능	디스크
SC45	64개	92GB	128Gflops	4TB(외장)
DS20E	2	1	2.68Gflops	36GB(내장)
ES40	2	2	2.68Gflops	36GB(내장)
DS20	1	1	1.34Gflops	36GB(내장)



<figure 6-1> 바이오인포매틱스 기반 시스템 구성도

o 소프트웨어 사양

- SC45 시스템에 설치된 운영체제는 Tru64 Unix V5.1a를 사용한다.

- 작업 관리 프로그램으로는 Platform사의 LSF(Load Sharing Facilities)를 사용한다.

- 백업을 관리하기 위한 프로그램으로는 Atepo사의 TiNa(Time Navigate)를 사용한다.

- 오라클 9i등 다양한 어플리케이션 소프트웨어를 설치하였다.

나. Linux Cluster 시스템 도입 및 설치

○ 활용방안 수립 및 입찰에 의한 시스템 도입

○ 하드웨어와 소프트웨어 설치, 품목 및 안정성 검사 시행

(1) 도입 목적

○ 대량의 서열 데이터 유사성 검색 및 유전체정보 분석

○ 유전체정보 데이터베이스 유통 및 분석 서비스 제공

○ 검색 에러율을 줄이고 검색 속도 개선

○ 리눅스 클러스터링 시스템 구축에 의한 안정적 서비스 제공

○ 유전체 정보의 분석을 통한 원활한 연구 활동과 이에 필요한 신속, 정확한 자료처리 결과를 얻기 위해 필요한 PC Cluster 시스템의 구성과 제반 운영환경 조성

○ 유전체 정보 분석을 지원할 수 있는 PC 클러스터링에 근거한 대규모 확장 가능한 리눅스 기반 시스템 지원 환경 조성

(2) 도입 및 설치 일정

○ <table 6-2>에 나타난바와 같이 리눅스 기반 클러스터 시스템 도입을 경쟁 입찰방식으로 추진하여, 주식회사 포스데이터가 낙찰되어 7월초부터 서비스를 시행하였다.

<table 6-2> 리눅스 기반 클러스터 시스템 도입 일정

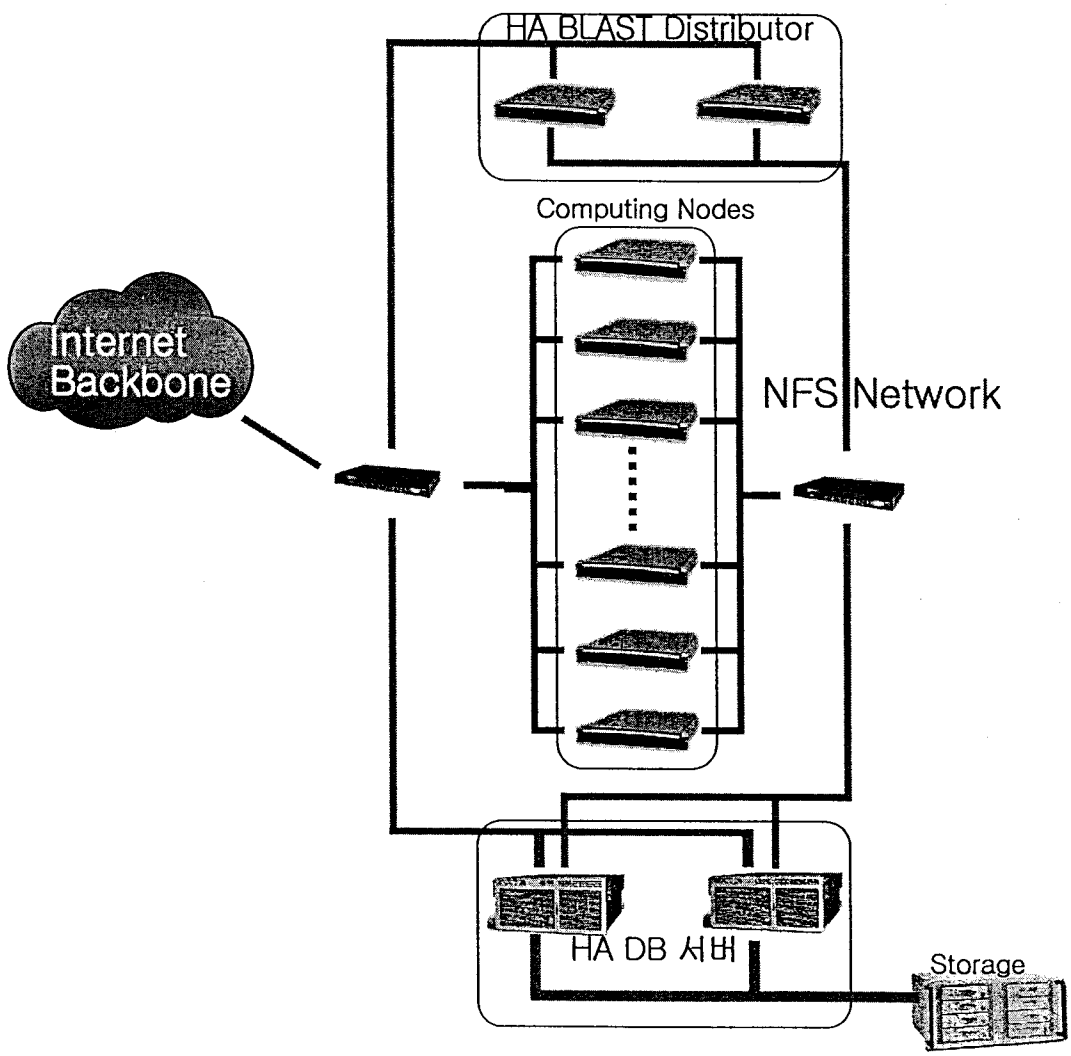
일 정	주 요 내 용
5월15일	입찰 사양 설명회
5월22일	입찰
5월28일	계약 체결
계약 후 2주 이내	시스템 설치 완료
계약 후 5주 이내	승인 시험 실시 완료
계약 후 6주 이내	대외 공개 서비스 개시

(3) 시스템 사양 및 구성도

○ 하드웨어 사양 및 시스템 구성도

<table 6-3> 리눅스 기반 클러스터 시스템 하드웨어 사양

모델	CPU 수	메모리	이론성능	디스크
계산 노드	96개	48GB	96Gflops	1TB(외장)
로그인 노드	4	2	4Gflops	
NFS 노드	4	2	2.68Gflops	



<figure 6-2> 리눅스 기반 클러스터 시스템 하드웨어 구성도

○ 소프트웨어 사양

<table 6-4> 리눅스 기반 클러스터 시스템 소프트웨어 사양

구 분	세 부 사 항
Operating System	Linux Ver 7.3
	Message Passing Libray
Remote File Access	NFS 이용
Parallel BLAST	Super BLAST Ver2.0 Parallel BLAST Distributor
Bio관련 소프트웨어	FASTA HMMER등

다. Linux Cluster 시스템 증설

(1) 시스템 사양

<table 6-5> 리눅스 기반 클러스터 시스템 사양 - 클러스터 노드

품명	세부사양			수량
클러스터 노드	프로세서	Intel XeonDP 2.8GHZ/518KB	12Set	1
	시스템 버스	533MHz		
	메모리	4GB(1GB X 4) DDR ECC SDRAM		
	하드디스크	146GB U320 10K HDD 3개		
	네트워크 카드	Onboard, 10/100/1000Mbps Dual NIC		
	RAID 컨트롤러	Onboard, SCSI U320용 2채널, battery backed cache 내장		
	비디오컨트롤러	Onboard		
	샤시	1U Rackmountable Chassis, 전면마우스, 키보드, 모니터, USB 포트		
	I/O 확장슬롯	100MHz 64bit PCI-X 슬롯 2개		
	전원장치	Redundant Power Supply		
	CD-ROM	24배속		
	FDD	3.5" 1.44MB		
	외부인터페이스	Serial 1Port 이상, USB 3Port		
	관리 S/W, O/S	포함, Redhat Linux V9.x professional		
	유지보수	3년이상		
	랙	42U, TRAY 방식, KVM(16포트), KEYBOARD WITH TRACKBALL 14" TFT LCD 평면모니터		

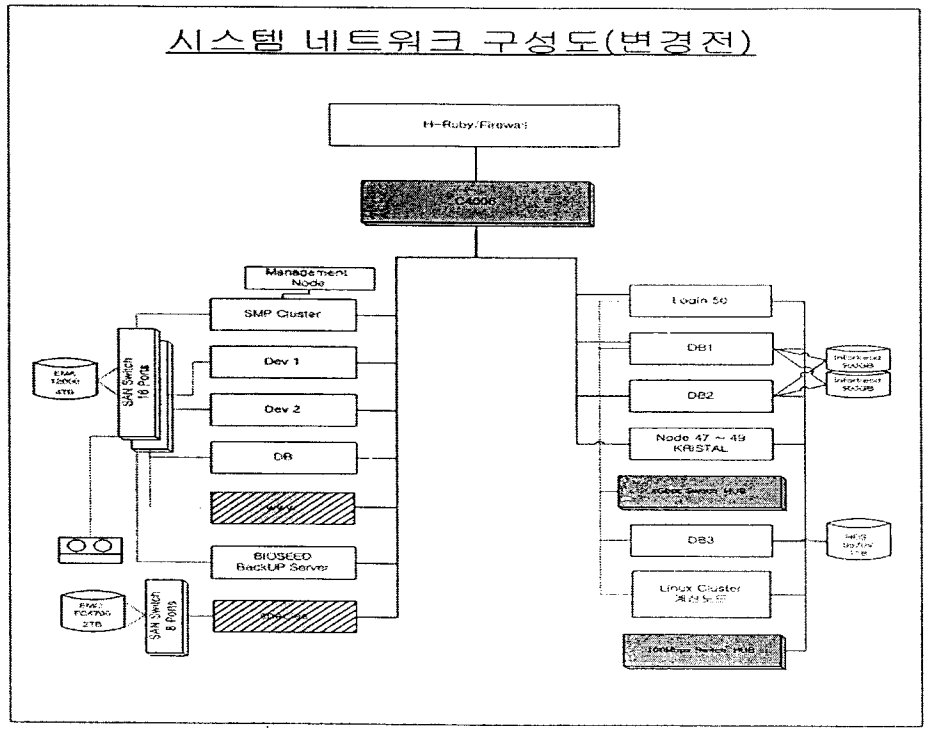
<table 6-6> 리눅스 기반 클러스터 시스템 사양 - 스토리지

품명	세부사양	수량
스토리지	- 랙 : 스토리지와 동일사 정품 (35U ~ 42U의 black/blue color 톤으로 제공) - 제안용량 : 1TB이상/16TB이상 확장가능(70GB 기준) - 컨트롤러 : Dual - 호스트 인터페이스 : 4포트이상(FC-2Gbps/포트) - 드라이브 인터페이스 : 4포트이상(FC-2Gbps/포트) - 캐쉬 : 1GB 이상 - 드라이브 : 70GB ~ 80GB(10,000 rpm 이상) - RAID 0/1/3/5/0+1 지원 가능 - LUN 생성 : 512개 이상 - 내부볼륨복제 기능 제공 - 자동채널장애 복구기능 제공 - 이중화 및 Hot swappable 지원 : 컨트롤러, 파워 등 주요 구성 요소 - 배터리 저장시간 : 72시간 이상 - HBA 2개(4포트) 이상(FC-2Gb 지원) - Full Fibre 제공 - 스토리지 관리 프로그램 제공(GUI 환경에서의 구성, 모니터링, 온라인 관리) - Fibre Cable 8개 이상 포함	1

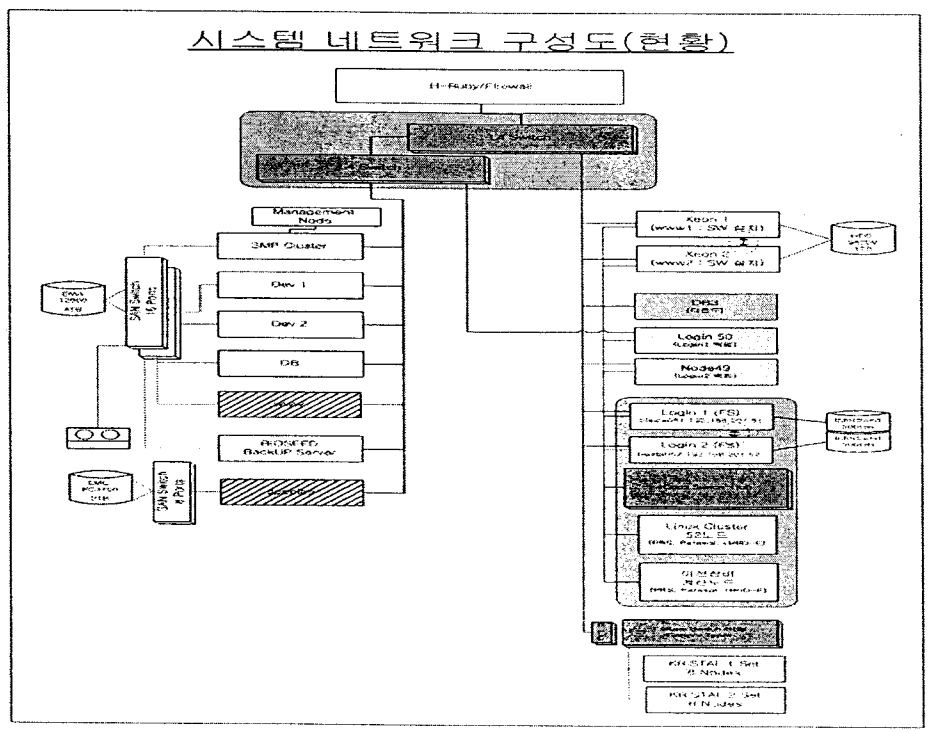
<table 6-7> 리눅스 기반 클러스터 시스템 사양 - 네트워크 스위치외

품명	세부사양		비고
네트워크 스위치	※ 랙 타입 ※ Fast Spanning Tree 프로토콜 지원 ※ 전체 클러스터 시스템의 내부 네트워크를 1Gbps 이상으로 연결할 수 있는 2 Level 구조로 구성해야 하며 전체 클러스터를 운영하는데 문제가 없어야 한다. ※ 네트워크 관리가 CLI 및 WEB을 통해서 가능해야한다.		
	- 계산용 스위치 * 24Port 이상의 TX Giga Ports * 4Ports 이상의 SX Giga Ports for Uplink * 4Gbps 이상의 Full Duplex Trunk 지원 * 48Gbps 이상의 Backplane Switching Capacity 지원 * 128MB 이상의 메모리와 16MB 이상의 플래쉬 메모리를 제공한다.	9 대 이상	
	- 백본 스위치 * 계산용 네트워크의 각 4Gbps Uplink를 수용하고 또한 각 서버들을 수용할 수 있는 백본 스위치 * 128MB 이상의 메모리와 16MB 이상의 플래쉬 메모리를 제공한다.	1 대 이상	
기타	- Mini G-Big 모듈(8개) - Intel Gigabit NIC 32 bit(180개) - Intel Gigabit NIC 64 bit(4개) - 전체 190노드 이상을 수용 가능해야 한다. - 별도의 18포트 이상의 여유 포트 제공		

(2) 네트워크 구성도 및 서버 이전설치



<figure 6-3> 네트워크 구성도 - 변경전



<figure 6-4> 네트워크 구성도 - 변경후

(3) 서버 이전 작업

(가) 작업 일정

번호	내용	비고
1	- 서버 인수(128노드) - 이전 서버(128노드) 네트워크 구성 - 이전 서버 OS 설치 - www1, www2 : OS 설치 - 백업(각 노드 : /usr/local, /etc - DNS, IP 등록 및 방화벽 설정 변경	
2	- 52노드 OS Upgrade(기존상태 유지) --> 향후 추진 - 기존 DB1, DB2 : Login1, Login2로 OS 설치 (임시로 WEB, MySQL DB 설치 및 기동) - 네트워크 통합(128노드 + 52노드) *기존 52노드의 서비스 기능 유지 (--> www 작업 완료 후 기능 이전) * 로그인 및 파일 서버 기능 수행 - www1, www2 : 데이터 이전용으로 서버 기동 - KRISTAL 데이터 이전	
3	- SMP 데이터 이전(-> www1, www2) * www --> www1, www2 * ftp --> bioftp	
4	- SMP(WEB, FTP, Ensembl) 및 리눅스(WEB, MySQL) 기능 이전	
5	- 랙 정비 * 리눅스 서버를 용도별로 랙에 장착 * 스위치 정리 및 랜 백업 환경 구축 - 서버 이전 작업 마무리	

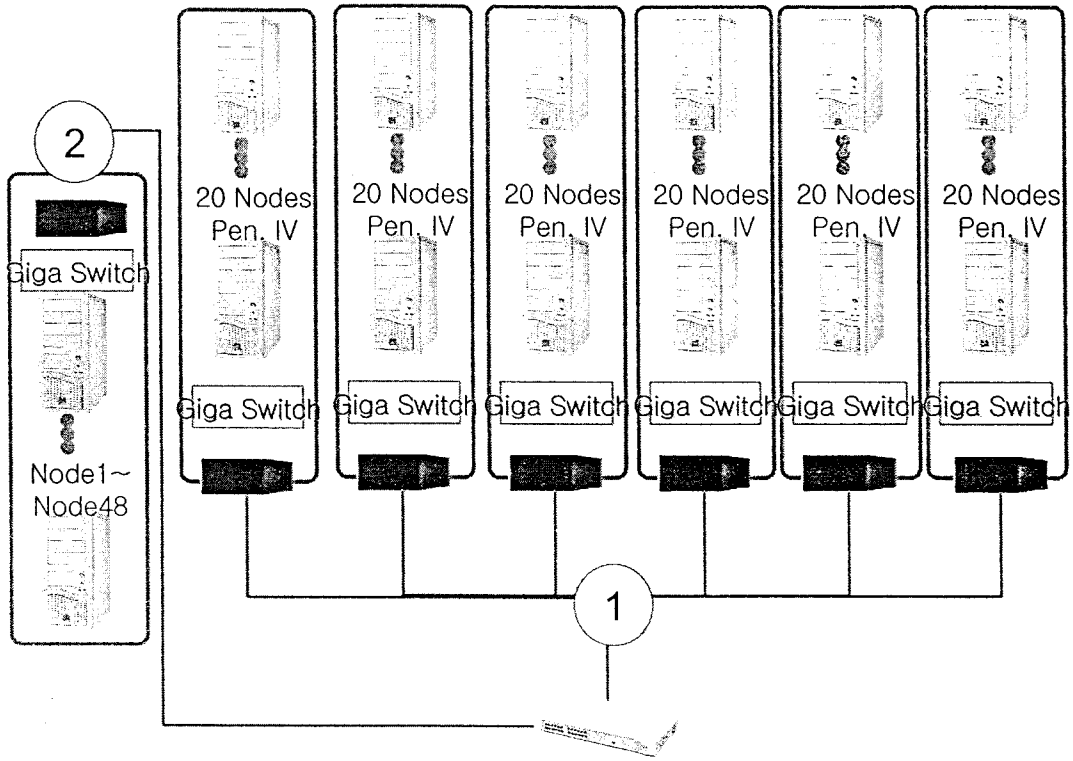
(나) 작업 순서

서버	순서 및 내용	비고
리눅스	1. 서버통합 - 128 노드 네트워크 구축 - 128 노드 OS 및 각종 SW 설치 - 기존 52노드 데이터 백업 * blasta001 ~ blasta046 (-> db3:/home2/'hostname') ** 대상 : /etc, /usr/local * blasta047 ~ blastadb2 (-> db3:/home3/'hostname') ** /etc, /usr/local, var/www(login 서버) - 52 노드 OS Upgrade - 기존 52노드와 네트워크 연계	
	2. 기능별 서버 분리 구축 - KRISTAL 전용 서버 구축(8노드 x 2세트) - 신규 WEB 서버 구축(2노드) * OS 및 SW 설치	
	3. 네트워크 IP 변경 - 150.183.47.0 --> 150.183.48.0	
	4. 데이터 이전 - 기존 리눅스의 서비스는 그대로 유지 - 기존 데이터를 이전 * WEB용 데이터 이전(-> www1, www2로 이전) * MySQL 데이터 이전(-> DB 서버로 이전)	
	5. KRISTAL 이전	
	6. 데이터 이전 작업 완료 후 - www 기능 완전 이전(-> www1, www2)	
SMP	1. 리눅스로 데이터 이전 - www --> www1, www2 - ftp 데이터 --> bioftp	
	2. www, bioftp, ensembl 네트워크 변경 - 기능 분리	
	3. 전체 서비스 기능 이전 완료	

(다) 도메인/IP 변경

현재 사용 IP	DNS	비고
150.183.47.65	www.ccbb.re.kr web.ccbb.re.kr ccbb.kisti.re.kr	데이터 이전 후 작업
150.183.47.66 ftp.ccbb.re.kr	www1.ccbb.re.kr	
신규	www2.ccbb.re.kr	
신규	elsembl.ccbb.re.kr	서버 : ES45
150.183.47.70	ftp.ccbb.re.kr bioftp.ccbb.re.kr uddi.ccbb.re.kr species.ccbb.re.kr	
150.183.47.100 blasta.kisti.re.kr	cluster.ccbb.re.kr linux.ccbb.re.kr blasta.ccbb.re.kr	가상 IP
	login1.ccbb.re.kr	Real IP
	login2.ccbb.re.kr	
150.183.47.104	www.physiomekorea.org	데이터 이전 후 작업
신규	kristal1.ccbb.re.kr	Real IP
신규	kristal2.ccbb.re.kr	
신규	kristal.ccbb.re.kr	가상 IP

(라) 리눅스 서버 통합



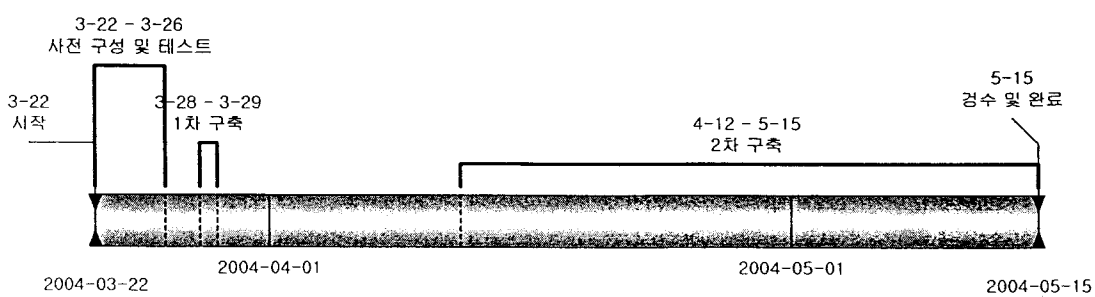
<figure 6-5> 리눅스 서버 통합 시스템 구성도

① 128대의Pentium IV장비가 20대씩 edge Iron24G에 연결되어 있고 이는 다시 uplink를 위한 스위치로 연결된다.

② 48대의 Pentium III 장비는 uplink를 통하여 나머지 장비들과 연동된다.

라. 스위치 및 취약성 점검툴 설치

(1) 수행 기간



(2) 설치 내역

모델명	상세리스트	수량	비고
멀티레이어 스위치			
Alteon 184 #1	Chassis	1	S/N:
	S/W	1	10.0.28.5
	10/100/1000 포트	9	00:0e:62:f7:1f:f0
Alteon 184 #2	Chassis	1	S/N:
	S/W	1	10.0.28.5
	10/100/1000 포트	9	00:0e:62:f7:16:50
취약성 점검도구			
ISS 인터넷 스캐너	S/W	1	
	라이센스	1	

(3) 작업 내용

2004.03.22 ~ 2004.03.26	사전 구성	각 장비 사전 설정
	테스트	장비간 연결 테스트
2004.03.28 ~ 2004.03.29	장비 장착	랙에 실장 및 장비간 연결
	장비 구성	SLB 설정, 라우팅 설정
	테스트	장비간 연결 테스트
2004.04.12 ~ 2004.05.25	장비 구성	SLB 커스터마이징, L4 추가 설정
		연결 테스트
	IS 설치	취약점 점검 소프트웨어 설치
2004.05.27	완료 보고	사업 종료 보고

마. 스토리지 도입

(1) 도입 목적

○ 국내 생물종자원(Species)에 대한 Catalogue정보 구축

○ 내용정보, 이미지정보, 동영상정보, GIS정보, 관련 정보 등과 같은 방대한 양의 멀티미디어 정보를 실시간(Real-Time)으로 운용

○ 대용량 데이터 처리로 원활한 정보 운용이 가능한 네트워크 체제 구축

<table 6-8> 스토리지 하드웨어 및 소프트웨어 사양

구분	세부 내역
하드웨어	FC4700-2 : FC4700 2GB DPE(Factory Install)
	PWCAB-USL : DUAL CAB Power Cord US TwistLock
	EMPTY-RACK : CLARiX 39U RACK
	DAE : ARRAY EXPANSION(Factory Install)
	FM-LL10MD : 25M FIBRE CABLE
	FC-31-73 : FC 73GB Drive 10k RPM
	Redundant & Power Path Support Support OS : Tru64
소프트웨어	Navisphere Manager NT/W2K, T6
	Navisphere Agent SUN
	Navisphere ATF-SUN Attach
	Access Logix
용량	2TB

(2) 활용분야

○ 생물자원정보의 정보화·지식화를 위한 관련 정보DB 구축

○ 디지털화를 위한 작업 공간 제공 및 기본적인 인프라 제공

○ 국내 기 구축된 정보와 신규 구축될 정보들간의 네트워크 체제 구축을 위하여 관련 데이터베이스들 간의 미러 사이트 운영 및 통합 서비스 제공을 위한 작업공간

○ 정보의 데이터베이스화 및 관련정보의 생성

3. 바이오인포매틱스 시스템 운영

○ 서비스 개발 지원을 위한 기반 구축, 운영 모니터링 및 보안 관리(IDS)

○ 사용자 계정 관리 및 사용자 지원

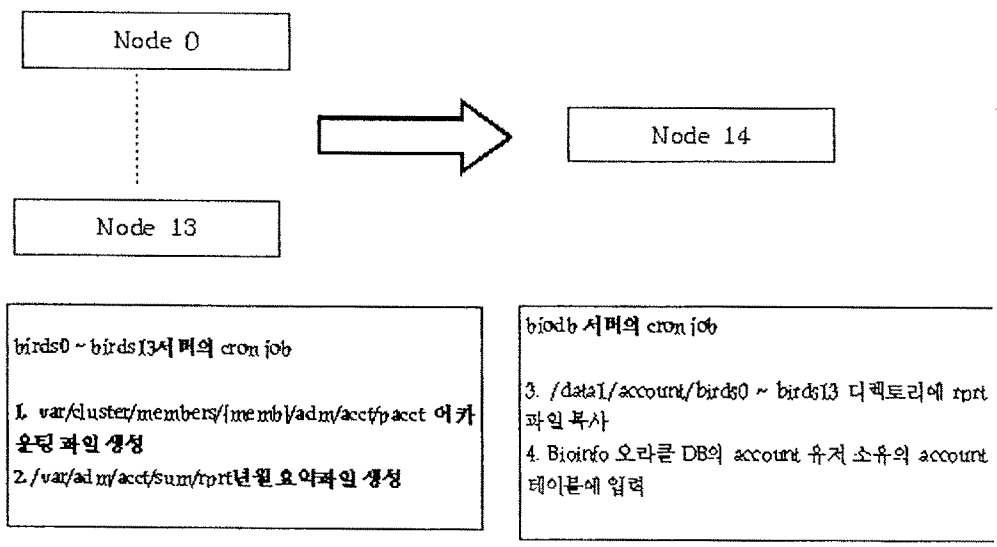
○ 시스템의 운영을 위하여 제반 주변 환경 구축

→ 슈퍼컴퓨팅센터 공조, 냉난방, 전원 및 통신시설 이용

가. 서비스 개발 지원을 위한 기반 구축

(1) 시스템 Account 구성

○ 구축된 어카운팅 시스템의 구조



- 어카운팅 대상 : SC45 시스템을 구성하는 서버중 bird0 부터birds13까지의 14대의 서버가 어카운팅 대상임

- 각각의 서버에 산출되는 어카운팅 파일 : 어카운팅 대상으로부터 산출되는 어카운팅 데이터 파일은 1차적으로 각각의 서버에 산출됨. 디렉토리 및 파일명 : /var/cluster/memb ers/{memb}/adm/acct/pacct 여기서 {memb}는 각 서버별로 birds0부터 birds13으로 대체 됨.

- 각각의 서버에서 산출된 어카운팅 파일을 가공 : 어카운팅파일을 매일 한번씩 읽어들여 각 서버의 /var/adm/acct/sum 디렉토리에 매일 rprt월일의 이름으로 (예 : 11월 1일의 데이터라면 rprt1101이라는 파일이름을 갖게 됨.) 어카운팅 요약 파일을 생성 함. 이 작업은 각각의 서버의 cron 작업에 의해 db서버로의 전송작업까지 함께 일괄적으로 매일 1회 수행됨.

- 어카운팅 요약 파일의 DB서버로의 전송 : 이렇게 생성된 rprtxxxx파일을 rcp를 통하여 각각의 서버별로 biodb서버의 /data1/account/birds0 ~ birds13 디렉토리로 복사한 후 각각의 서버에 있는 /var/adm/acct/sum/backup 디렉토리에 옮겨 놓음.

- DB 서버에서 수신한 어카운팅 요약 파일을 데이터베이스에 입력 : biodb라는 이름을 갖는 오라클 데이터베이스서버의 /data1/account/birds0부터 /data1/account/birds13 까지의 디렉토리에 rprt라는 이름으로 도착한 어카운팅 파일은 biodb의 cron 작업에 의해 1일 1회 데이터베이스에 입력되고 (bioinfo 데이터베이스의 account유저가 소유하는 account라는 이름의 테이블에 입력됨) 해당 어카운팅 요약 파일들은 1일동안만 /data1/account/1day_backup/birds0 ~ 13까지의 디렉토리에 나누어 저장됨. 1일이 지나면 cron 작업에 의해 새롭게 수신된 파일로 대체됨

o DB에 의한 어카운팅 데이터 저장

- 어카운팅 데이터 저장 시스템은 biodb서버의 Oracle Database에 저장 한다.

o 어카운팅 데이터 저장 시스템의 어카운팅 구조

- 어카운팅 데이터 저장 시스템인 BIODB서버의 다음의 디렉토리에 어카운팅 레포트 파일인 rprt 파일이 매일 1회씩 각각의 소스 서버로부터 전송되어 저장된다.

/data1/account/birds0 ~ birds13의 총 14개의 어카운팅 디렉토리가 biodb 서버의 /data1/account 디렉토리 아래에 존재하며, 여기에 rprt라는 이름의 텍스트 파일이 들어간다.

- 이렇게 도착한 어카운팅 파일은 오라클 PL/SQL 프러시져가 읽어들여 BIOINFO 데이터베이스의 account 유저의 account 테이블에 입력하게 된다.

- 그런 다음 각각의 레포트 파일은 백업 디렉토리로 옮겨져 1일간 보관되게 되며, 만일 1일 이전의 데이터가 필요하게 될 경우는 각각의 소스 서버에 보관되는 백업 파일을 이용하면 된다.

o 어카운팅 테이블

- biodb서버의 boiinfo 오라클 데이터베이스 인스턴스의 account 유저의 account 테이블의 생성 한다.

- 각 칼럼에 들어가는 데이터의 내용은 pri_cpu 칼럼부터 순서대로 다음과 같다.

server_name : 어카운팅 대상 시스템 명칭
acct_date : 어카운팅 대상 날자, 1일 단위임.
user_id : 리소스를 사용한 유저 아이디
login_name : 리소스를 사용한 유저의 이름
group_id : 리소스를 사용한 유저가 속한 그룹의 아이디
group_name : 리소스를 사용한 유저가 속한 그룹의 이름
pri_cpu, npri_cpu : Cumulative CPU time in minutes. pri는 평일 npri는 휴일의 데이터임.
pri_mem, npri_mem : Cumulative K-core time in minutes.
pri_chrblrw, npri_chrblrw : Cumulative number of characters transferred in blocks of 512 bytes.
pri_blockio, npri_blockio : Cumulative number of blocks read and written.
pri_connect, npri_connect : Cumulative connect time in minutes.
disk_time: Cumulative disk-usage time in minutes.
print_fee : Queuing system (printer) fee in number of pages.
special_fee : Special services fee expressed in units.
count_login: A count of the number of login sessions.
count_disk : A count of the number of disk samples.

○ 어카운팅을 위해 필요한 기타 설정 사항

① 어카운팅을 위한 프로그램이 설치되었는지 확인함

/usr/sbin/setld -i | grep Accounting => 설치되었으면 설치된 항목이 표시됨.

② 리부팅시에도 어카운팅이 자동으로 되도록 설정함.

rcmgr set ACCOUNTING YES

③ 어카운팅 파일의 권한을 꼭 다음과 같이 해주어야 함.

chmod 664 /var/cluster/members/{memb}/adm/acct/pacct

ls -al /var /cluster/members/{memb}/adm/acct/pacct

④ 지금 바로 어카운팅이 시작되도록 함.

/usr/sbin/acct/startup

⑤ 만약 중단하려면 다음의 명령을 함.

/usr/sbin/acct/shutacct

⑥ 어카운팅이 되면 파일의 크기가 명령시마다 계속 증가함.

ls -al /var/cluster/members/{memb}/adm/acct/pacct

ls -al /var/cluster/members/{memb}/adm/acct/pacct (크기 증가 확인)

⑦ 다음의 파일의 권한이 꼭 4755여야 됨.

chmod 4755 /usr/sbin/acct/accton

ls -al /usr/sbin/acct/accton (권한 확인)

⑧ 이제 생성되는 pacct파일을 매일 한번씩 처리하여 요약정리파일인 rprtmmdd 및 tacctmmdd가 생성되도록 adm유저의 크론탭에 등록함.

crontab -e adm

59 23 * * * /usr/sbin/acct/kisti_acct0 > /usr/adm/acct/nite/fd2log &

kisti_acct0 ~ 13 파일에 여러 내용이 있는데 그중에서

/usr/sbin/acct/runacct 라인이 실제 어카운팅 작업을 수행하는 명령줄이며 runacct 가 이런 요약작업을 함.

⑨ 어카운팅 파일의 필드는 모든 필드가 다 되도록 함.

vi /usr/sbin/acct/prtacct

위의 명령 후 FIELDS=항목을 아래와 같이 만들어 줌.

FIELDS="1-18"

이 작업은 클러스터내의 아무서버 하나만 해주면 모두 적용된다.

(2) Oracle Setup

o 커널 파라미터 설정

- /etc/sysconfigtab을 수정

ipc: shm_max = 4278190080

shm_mni = 256

shm_seg = 128

proc: per_proc_stack_size = 33554432

per_proc_data_size = 201326592

vm: new_wire_method = 0

SGA가 4GB 이상일 경우 오라클9i 설치 참조

o Mount Point 생성

software와 db file용을 분리하여 작성

db file용은 3개 권장

o DBA용 UNIX그룹 생성

osdba 그룹을 생성 : osdba로 생성했음

orainventory 그룹을 생성 : orainven으로 생성했음

o Oracle Universal Installer Inventory용 UNIX 그룹 생성 :생성치 않았음

o Oracle용 UNIX Account 생성

umask 022 지정

name : oracle

primary GID : orainven

secondary GID : osdba

home directory : oracle home과는 다음 일반 사용자와 비슷하게 설정
/usr/users/oracle

Machine: OSF1 web.ccbb.re.kr V5.1 1885 alpha

HOME=/usr/users/oracle

○ Oracle HTTP Server용 UNIX Account 생성 : 만들지 않았음.

해당 유저는 orainventory 그룹의 member여야 설치 가능

HTTP Server의 start는 root user로 해야 root용 port가 있어야하며, application에 가용 하게된다.

○ File 생성을 위한 Permission 설정

오라클 유저가application을 설치하고 database 파일을 생성할 수 있도록 미리 해당 디렉토리에 대한 사용 권한을 부여한다.

○ oracle 유저로 할 일

▸ DISPLAY 변수 설정

IP는 S/W 설치하는 서버의 경우 Xwindow 콘솔을 이용, IP는 S/W 설치하는 서버가 아닌 곳에서 화면을 볼 때만 지정

BOURNE, KORN Shell의 경우.

$xhost +server_name (자기의 컴퓨터에서)

$DISPLAY=workstation_name:0.0 (서버에서)

$export DISPLAY

C Shell의 경우

xhost +server_name (자기의 컴퓨터에서)

setenv DISPLAY workstation_name:0.0 (서버에서)

▸ ORACLE_BASE 설정

오라클 S/W의 최고위 directory. 'Mount_point/app/oracle'을 권장함
ORACLE_BASE=/data1/app/oracle

▸ ORACLE_SID 설정 : 데이터베이스 인스턴스의 이름 ORACLE_SID=BIOINFO

▸ ORACLE_HOME 설정

특정 release용 근거 디렉토리. $ORACLE_BASE/product/release 권장. ORACLE_HOME=/data1/app/oracle/product/9.2.0.1.0

▸ PATH변수 설정 : 다음을 포함. $ORACLE_HOME/bin, /usr/bin, /etc, /usr/bin/X11, /usr/local/bin

PATH=/data1/app/oracle/product/9.2.0.1.0/bin:/usr/users/oracle/bin:/data1/app/oracle/product/9.2.0.1.0/bin:/usr/bin:/etc:/usr/bin/X11:/usr/local/bin:/usr/i18n/bin:/usr/bin:.

▸ ORA_NLS33 설정 : 기본 위치가 아닌 곳에 .nlb 파일이 있을 경우

▸ Env 실행

bourne 또는 korn shell의 경우 : cd, $HOME/.profile

C shell의 경우 : cd, source $HOME/.login

▸ pro C/C++ : path에 Pro C/C++ compiler executable이 들어가도록 함.

pro C 사전요건 5.1A Patch Kit1의 경우 v6.4-014, C++ v6.3-008

5.1 Patch Kit 4의 경우 v6.3-029, C++ v6.3-008

▸ SHELL=/usr/bin/csh

Oracle Universal Installer실행시 주의 요건

- oracle user로 실행
- JDK 1.3.1 사전 설치 (/usr/opt/java131)
- runInstaller 실행

○ 설치 정보

Oracle OS User Account : oracle

DB SID : bioinfo

Global DB Name : bioinfo.ccbb.re.kr

Oracle Home : /data1/app/oracle/product/9.2.0.1.0

Edition : Enterprise Edition

Language : KO16KSC5601

SGA size : 3.7GB (database buffer 2.3GB, shared pool 1.4GB)

(3) LSF Scheduling Policies 모델

o Preemptive scheduling

Pending된 High-priority 작업이 running 되고 있는 low-priority의 resource를 점유하는 정책. LSF는 자동으로 low-priority 작업을 suspend시키고 high-priority 작업에게 resource를 넘겨준다. 그리고 suspend된 low-priority 작업은 resource가 있으면 다시 실행된다.

o Exclusive Scheduling

작업이 실행 될 때, 그 작업이 끝날 때까지는 다른 어떠한 작업도 그 host에서 돌아가지 못 하게 하는 정책

o FCFS Scheduling

First Come First Service Scheduling

o Fairshare Scheduling

특정 사용자 및 그룹에게 동적인 Priority를 제공하는 정책

o Deadline Constraint Scheduling

특정 시간이 되면 running되고 있는 작업을suspend 하는 scheduling.

Run window : Queue level에서 정의하며 running되고 있는 작업을 suspend시킨다

RUN_WINDOW = time

o Time based configuration

특정 시간이 되면 LSF의 configuration을 자동으로 바꾸어 정책을 변화시킬 수 있는 환경을 지원한다.

○ 병렬 작업을 위한 Queue정책

Processor reservation :

CPU를 n에 MBD_SLEEP_TIME을 곱한 시간동안 예약한다.

Memory reservation :

Memory을 n에 MBD_SLEEP_TIME을 곱한 시간동안 예약한다.

Backfill Scheduling :

예약된 작업 slot을 사용할 수 있게 끔 하는 정책.

○ Resource Usage Limits

Resource에 제한을 두어 작업을 제어하는 방법으로 usage가 제한범위를 넘어서면 작업을 삭제하거나 dispatch를 하지 않는 방법.

○ Load Thresholds

작업이 실행되는 system의 load index에stop condition을 설정하여 이 수치를 넘어서게 되면 priority가 낮은 작업부터 suspend를 거는 방법. 또한 load가 start condition 이하가 되면 다시 running 상태로 됨.

○ Requeue and Rerun

Requeue :

작업이 종료된 후 특정 exited code가 발생된 경우, 작업을 다시 queue에 넣는 방법.

Rerun :

작업이 실행되는 동안 system에 문제가 있어 작업이 중단된 경우 작업을 재실행하는 방법

○ Queue Priority

Queue마다 Priority를 다르게 하여 높은 priority의 큐에 있는작업을 우선적으로 dispatch 시키는scheduling.

○ Sample Queue

- Sample A:

Queue	Priority	Max Job	Threshold	Preemption	기타
Realtime	80	4	ut - r1m -	Preemptive [express normal economy]	NEW_JOB_SCHED_DELAY=0
express	60	-	ut 0.7/0.9 r1m 0.9/1.0	Preemptive [normal economy] Preemptable [realtime]	
Normal	40	-	ut 0.6/0.8 r1m 0.7/1.0	Preemptable [realtime express]	
economy	20	-	ut 0.5/0.8 r1m 0.6/1.0	Preemptable [realtime express]	

- 각 queue의 특징

realtime 큐의 특징은 priority가 가장 높아 dispatch순위가 가장 높을 뿐만 아니라, 나머지 큐의 resource를 preemptive할 수 있다. queue limit에 도달하면 사용자가 작업을 submit할 때 Queue limit에 도달하였다는 메시지를 바로 보여주어 사용자로 하여금 다른 queue에 작업을 던질 수 있게 하여준다.

또한 Job schedule time을 0로 setting 하여 dispatch time을 최소로 줄임.

```
% ./pbsubreal1.pl
Job <656> is submitted to queue <realtime>.
% ./pbsubreal2.pl
LSF is rejecting your job
QJOBLIMIT is reached
Request aborted by esub. Job not submitted.
%
```

Express 큐의 특징은 resource를 realtime에게 줄 수 있고, normal과 economy 큐에서 resource를 가져올 수 있다.

- Sample B

Queue	Priority	Max Job	Threshold	CPULIMIT	JL/U
4h	80	10	ut - r1m -	CPULIMIT=4:00	4
24h	60	6	ut 0.7/0.9 r1m 0.9/1.0	CPULIMIT=24:00	3
120h	40	6	ut 0.6/0.8 r1m 0.7/1.0	CPULIMIT=120:00	3
unlimit	20	4	ut 0.5/0.8 r1m 0.6/1.0		2

- 각 queue의 특징

각 Queue는 CPU limitation이 있고, CPU time이 낮게 걸리는 작업부터 dispatch가 된다.

- Sample C

Queue	Priority	Max Job	JL/U	기타
abaqus	60	4	4	RERUNNABLE=Y
gaussian	60	6	3	RERUNNABLE=Y
cerius	60	6	3	RERUNNABLE=Y
normal	40	10	2	RERUNNABLE=Y

- 각 queue의 특징

Application별로 queue를 설정한 sample.

Application license가 그 queue에서 돌릴 수 있는 최대 작업수로 setting되어 있음.

각 queue는 Rerun할 수 있게 끔 설정하여 놓는다.

o 기타

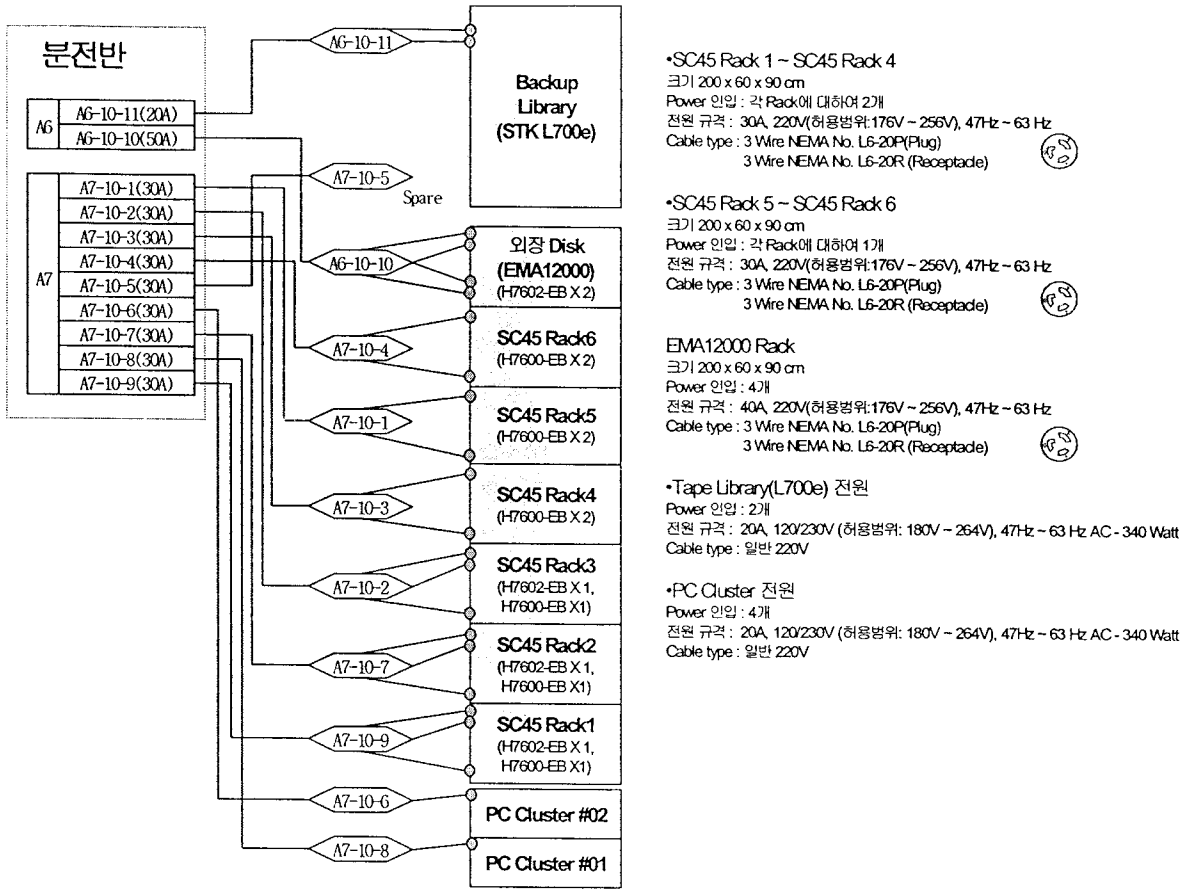
queue정책은 언제든지 사이트의 상황에 맞게끔 변경이 가능하며, 여러 개의 Schedule 정책을 복합적으로 설정하는 것도 또한 가능하다.

사이트의 특성과 일정기간동안의 작업형태를 보고 tuning을 할 수 있다

(4) 시스템 운영 환경

o 시스템의 운영을 위하여 제반 주변 환경 구축

→ 슈퍼컴퓨팅센터 공조, 냉난방, 전원 및 통신시설 이용



<figure 6-6> 전기 계통도

<table 6-9> SMP 클러스터 전력 용량 및 Cable 사양

장비명	모델	수량 (EA)	구분	전원			주파수		전력				전선		브레커 스위치
				상	정격(V)	범위(V)	정격(Hz)	범위(Hz)	용량		전류		가닥수	굵기 (mm2)	용량(A)
									단위 (Watt/EA)	합계 (Watt)	단위운전 (A/EA)	합계(A)			
AlphaServer SC45	ES45 5node rack	5	Rack	1	208	+-10	55	+-10	750 (1node)	12,000 (16nodes)	5 (1nodes)	80 (16nodes)	3	5.5	30A
AlphaServer SC45	CM--S128 & Network rack	1	Rack	1	208	+-10	55	+-10	1,000	1,000	6	6	3	2.6	15A
Storage	EMA 12000	1	Rack	1	208	+-15	55	+-10	1,800	1,800	16	16	3	8	40A
Backup	L700e	1	Rack	1	208	10%	60	+-10	150	150	10	15	3	-	20A

나. IDS(Intrusion Detection System) 설치

(1) 침입 탐지 시스템의 배치

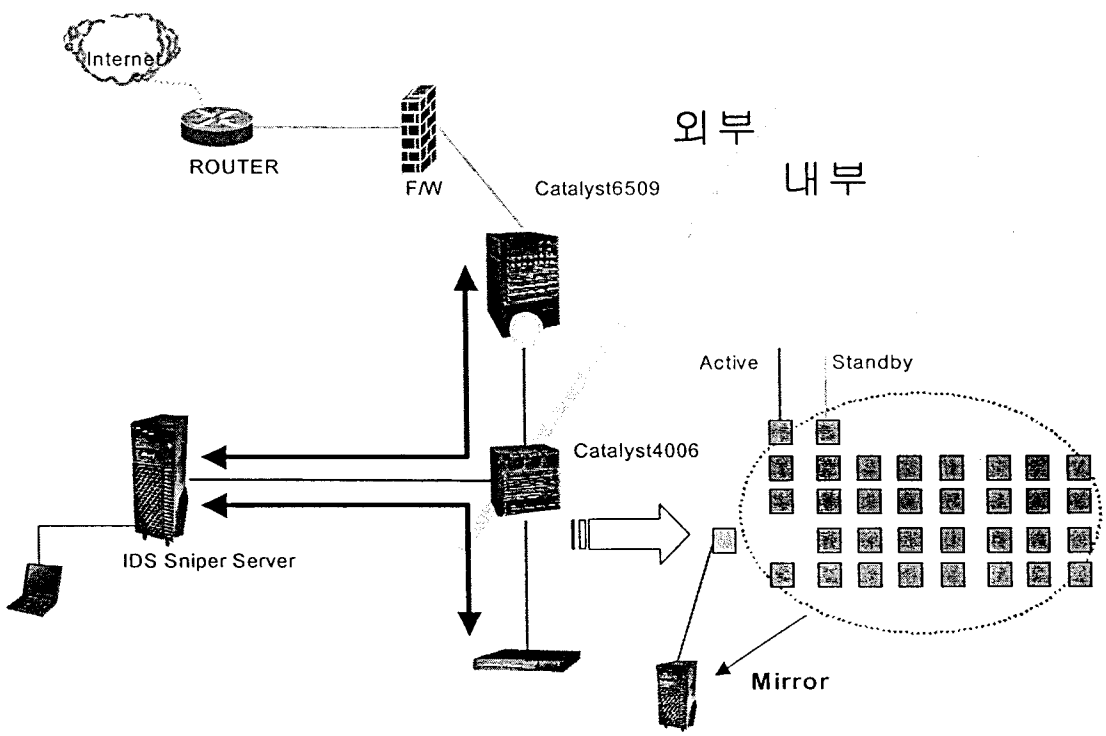
o IDS는 전산실의 Catalyst 4006 스위치에 연결하고, Catalyst 4006 스위치의 포트 미러링을 이용하여, 상단의 Catalyst 6509를 통해 들어오는 패킷 및 하단에서 올라오는 패킷에 대해서 모니터링을 함.

o Catalyst 4006 하단에 연결된, 기가 스위치에 연결 된 호스트 및 서버군들의 자체 통신을 감시하기 위해서는 1개의 네트워크 인터페이스를 IDS 서버에 연결하여, 멀티 모드의 패킷 검출을 해야 함.

o 접근 방법

㉠ 내부의 10/100 UTP 네트워크 포트를 통하여(Cross 연결), 자체 비공인 IP로 맞추어 암호화 통신을 이용하여 웹 브라우져로 접근한다.

㉡ 항상 내부 서버로 접근하여(콘솔) 인증을 거친 후에 접근하여야 함



<figure 6-7> 침입 탐지 시스템 배치도

(2) IDS 설치

○ IDS 서버의 하드웨어 사양

① 기종 및 중앙 처리 장치 : SUN BLADE 1000, ULTRA SPARC Ⅲ * 2EA

② 주 메모리 : 2048MB

③ 주 하드 디스크 드라이브 : 36GB

④ 자체 내장 10/100 LAN Adapter

⑤ 확장 기가비트1000BaseTP LAN Adapter

⑥ Operating System : Solaris 8

○ IDS 서버의 네트워크 연결 현황

① 자체 내장 보드 10/100 Interface

㉠ NIC 1개 할당

㉡ Private IP Address 할당

㉢ 시스템 자체의 내부 사용자를 위한 콘솔 포트로 이용

② 확장 Gigabit Ethernet 1000BaseTP Interface

㉠ NIC 1개 할당

㉡ Public IP Address 할당

㉢ SNIPER Enterprise V2.0 서버를 위한 인터페이스로 이용

㉣ Catalyst 4006 스위치와 기가 링크

○ IDS Software (SNIPER Enterprise V 2.0)의 사양

① 동시 처리 세션 수 : 10240개 이상

② 실 시간 네트워크 모니터링 및 탐지

③ 해킹 탐지 및 추적, 로깅, 방어 가능

④ E-Mail 및 웹 메일 로깅 및 바이러스 탐지

⑤ 스텔스 기능 내장

⑥ 웹 방식을 통한 원격 관리 기능

다. PBS 큐잉 시스템

(1) 개요

PBS (Portable Batch System) 는 일종의 Batch Queuing System으로, NASA Ames 연구센터의 NAS(National Aerospace Simulation)와Lawrence Livermore 국립연구소의 NERSC(National Energy Research Supercomputer Center)의 합작 프로젝트의 결과물이다. 현재 이 소스 코드는 Altair Engineering사에서 무료 배포 있으며, 그 소스 코드는 이 회사에서 사용자 등록을 하면 다운로드할 수 있으며, 저작권은 몇 가지 주의를 담은 부분을 소스코드에 첨가하면 소스코드의 변경과 배포를 할 수 있게 되어 있다.

PBS의 기능을 요약하면 다음과 같다.

① job 실행 정지 기능 : 가장 기본적인 Batch 시스템의 기능으로 job을 실행 정지뿐만 아니라, 그 우선도(priority)도 설정할 수가 있다.

② 보안 및 사용 제한 기능 : 특정한 job을 실행할 수 있는 서버나 그룹, 사용자를 지정할수 있으며, 특정한 자원도 특정한 서버나 그룹, 사용자만 사용하도록 제한할 수 있다.

③ 계산서(account) log 기능 : 각 사용자별 사용시간이나 사용 자원에 따른 상세한 사용 log가 자동으로 발급된다.

④ 병렬 job 처리 기능 : 리눅스를 포함한 다양한 플랫폼에서 병렬화를 위한 패키지인 MPI나 MPL, PVM, HPF(High Performance Fortran)등을 지원한다.

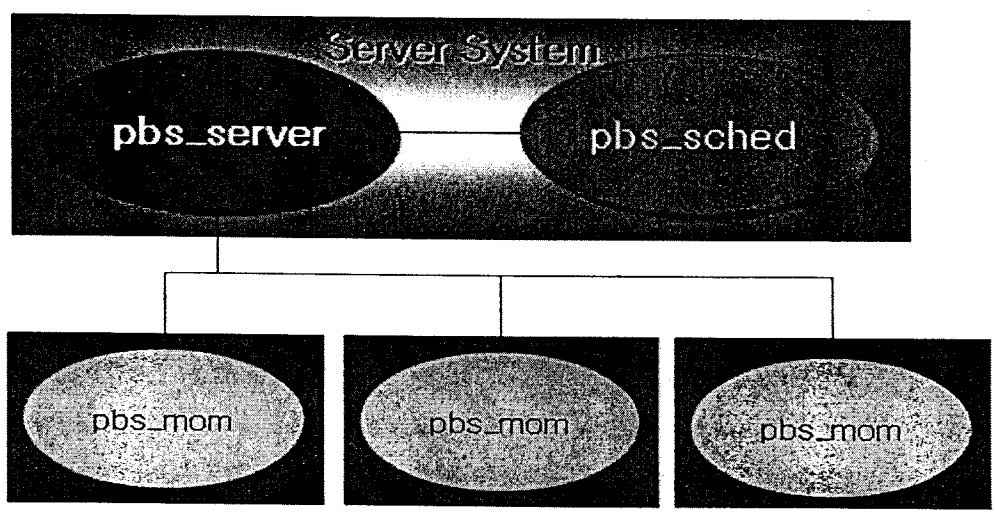
⑤ 표준의 API제공 : POSIX 표준의 API를 지원함으로써 다른 프로그램에서 PBS와 연동하는 기능을 구현할 수 있게 하였다.

PBS는 아래 그림과 같이 3개의 모듈 즉, 서버(server), 모체(mom), 스케줄링 데몬(sched)으로 구성되어 있다. 또 이들 각각은 priv라는 설정 디렉토리와 logs라는 로그를 보관하는 디렉토리로 구성되어 있다. 특히 서버의 환경 설정이 중요한데, server_priv에서 job 컨트롤과 계산서 발행, 사용자 사용 제한(acl)를 지정하고 있다. 또한 PBS서버는 각종 job들이 순차적으로 들어 있는 queue들을 관리하고, 서버와 사용자들의 보안을 책임지는 핵심적인 기능을 하고 있다.

- PBS server: 사용자로부터 Queuing 작업 을 받아서 수행하는 역할을 관장한다. 만일 작업을 수행할 경우에는 필요한 resource 및작업환경을 pbs scheduler와의통신을 통하여 할당 받는다.
- PBS scheduler: pbs server에서 필요한 resource 요청시 이에 응답하는 역할과, client의 현재 상태를 모니터링 하는 역할을 수행한다. Client의 상태를 주기적으로 확인하여 만일 이상이 발생하거나 client의 상태가 변할 경우 이를 즉시 DB화하여 보관한다.
- PBS_MOM: PBS server에서 받은 명령어를 직접 수행하는 역할을 한다. 작업이 끝나면 항상 server에 이를 보고한다. 그리고 PBS scheduler와 항상 통신을 하여 자신의 상태를 점검 받는다.

(2) OpenPBS 설치

OpenPBS는 Altair Engineering사에서 무료 배포하는 큐잉 시스템의 일종이다. 이를 설치하기 위해서는 gcc2.96이 필요하며, 'gcc -v' 명령어로 버전 확인 후 컴파일을 수행한다.



<figure 6-8> PBS 구조도

① PBS의 소스코드 (OpenPBS_2_6_18.tar.gz)를 적당한 디렉토리에 압축을 푼다.

```
% tar zxvf OpenPBS_2_6_18.tar.gz
% cd OpenPBS_2_6_18
```

② PBS의 job 스케줄러를 선택한다.

3가지를 선택할 수 있는데, C 언어와 Tcl 스크립트 언어와 PBS를 위해 확장된 C 언어인 BaSL(Batch Scheduling Language)가 있다. 디폴트로 C언어가 선택된다.

③ configure로 시스템에 필요한 정보를 알아내고 setting한다.

```
# ./configure
```

- 디폴트로 사용자 명령어들은 /usr/local/bin에 오며,
- 데몬이나 관리자를 위한 명령어들은 /usr/local/sbin에 오며,
- 데몬의 워킹 디렉토리(PBS_HOME)는 /usr/spool/pbs가된다.

④ PBS를 컴파일한다.

```
# make
```

⑤ PBS를 인스톨한다.

```
# make install
```

PBS 설치가 끝난 후 아래와 같은 순서대로 각 설정 화일들을 수정한다.

- /etc/services 화일에 아래와 같은 내용 삽입한다.

```
pbs             15001/tcp
pbs_mom         15002/tcp
pbs_resmom      15003/tcp
pbs_resmom      15003/udp
pbs_sched       15004/tcp
```

- /etc/hosts_equiv에 모든 서버 노드와 계산 노드의 호스트명을 삽입한다.

⑥ 만약에 PBS 클러스터에 몇 개의 호스트들이 있을때는 서버에 node 파일을 만든다.

PBS_HOME 인 /usr/spool/PBS를 보면 다음과 같이 여러 가지 파일들이 있다.

```
# cd /usr/spool/PBS
# ls -l
total 12
drwxr-xr-x  2 root    root       1024 Apr 27 01:14 aux
drwx------  2 root    root       1024 Apr 27 01:14 checkpoint
drwxr-xr-x  2 root    root       1024 Apr 27 01:14 mom_logs
drwxr-x--x  3 root    root       1024 Apr 27 01:14 mom_priv
-rw-r--r--  1 root    root         27 Apr 27 01:14 pbs_environment
drwxr-xr-x  2 root    root       1024 Apr 27 01:14 sched_logs
drwxr-x---  2 root    root       1024 Apr 27 01:14 sched_priv
drwxr-xr-x  2 root    root       1024 Apr 27 01:14 server_logs
-rw-r--r--  1 root    root          6 Apr 27 01:14 server_name
drwxr-x---  9 root    root       1024 Apr 27 01:14 server_priv
drwxrwxrwt  2 root    root       1024 Apr 27 01:14 spool
drwxrwxrwt  2 root    root       1024 Apr 27 01:14 undelivered
# cd server_priv
```

위 디렉토리 중 server_priv에 가서 nodes라는 파일을 하나 만들고, 여기에다가 각가의 호스트 이름들을 적어주면 된다. 여기서 호스트 이름을 적을 때 도메인 이름은 빼고 적는다. 즉, 만약 clust1.sait.samsung.co.kr이라는 호스트가 있다면, clust1만 적는다.

현재 nodes 파일의 내용은 다음과 같다.

```
Blasta001
Blasta002
....
Blasta008
```

만약에 위 node들이 timesharing node들이면 다음과 같이 호스트 이름뒤에 :ts를 붙인다.

```
Blasta001:ts
Blasta002:ts
```

Blasta008:ts

⑦ 이제 PBS를 위한 서버 환경 설정을 한다.

PBS를 위한 환경 설정을 하기 위해서는 일단 /usr/local/sbin에 있는 pbs_server를 다음과 같이 처음으로 실행해야 한다.

/usr/local/sbin/pbs_server -t create

그리고, qmgr를 사용하여 다음과 같이 queue를 설정해야 한다.

```
# qmgr
Max open servers: 4
Qmgr: #
```

⑧ 이제 PBS를 위한 데몬과 서버를 부팅할 때 자동 실행시켜주도록 한다.

자동 실행되어야 하는 데몬과 서버는 pbs_server, pbs_sched, pbs_mom 으로 /etc/rc.d 디렉토리에 있는 rc3.d에 다음과 같이 S100pbsd 이라는 파일을 만들어 시스템이 다시 부팅했을 때 자동실행되도록 만든다.

```
#!/bin/sh
#
# Startup script for the PBS
#
# config: /usr/local/PBS/serv_priv/node
#
/usr/local/sbin/pbs_mom
/usr/local/sbin/pbs_sched
/usr/local/sbin/pbs_server -t hot
```

(가) PBS Configuration

PBS의 환경 설정은 qmgr에 의해서 수행된다. 일단 PBS의 환경을 설정하기 위해서는 PBS의 설치 후 처음으로 pbs_server를 실행할 때는 다음과 같이 -t 이라는 서버 타입을 지정하는 옵션에 create의 타입으로 pbs_server를 실행해야 한다. (처음이 아닐 때는 hot, warm, cold라는 옵션을 쓸 수 있다.)

① pbs_server를 실행한다. (처음으로 실행할 경우)

/usr/local/sbin/pbs_server -t create

위의 경우는 처음으로 pbs_server를 실행할 경우이고, 보통 서버 셋팅이 끝나면 타입을 hot으로 하는 경우가 많다. cold일 경우는 queue에 있는 모든 job들을 죽이고 다시 서버를 실행하는 것이고, hot은 서버가 죽었다 다시 살아나도 queue에 잇는 모든 job들을 다시 실행할 수가 있다.

② 이제 qmgr를 실행한다.
그리고 다음과 같이 Queue를 관리하는 툴인 qmgr를 사용하여 환경 설정을 한다.
/usr/local/bin/qmgr
Max open servers: 4
Qmgr:

③ list server로 서버의 디폴트 환경을 본다.
이제 qmgr에서 디폴트로 셋팅된 서버 환경을 보려면 다음과 같이 list server를 실행하면 된다. (l s로 줄여써도 된다.)
Qmgr: l s
Server alpha
server_state = Idle
total_jobs = 0
state_count = Transit:0 Queued:0 Held:0 Waiting:0 Running:0 Exiting:0
log_events = 511
mail_from = adm
scheduler_iteration = 600
pbs_version = 2.1p18

Qmgr:

④ create queue로 디폴트 queue를 만든다.

```
Qmgr: c q dque queue_type=e
Qmgr: s s default_queue=dque

Qmgr: l s
Server alpha
    server_state = Idle
    total_jobs = 0
    state_count = Transit:0 Queued:0 Held:0 Waiting:0 Running:0 Exiting:0
    default_queue = dque
    log_events = 511
    mail_from = adm
    scheduler_iteration = 600
    pbs_version = 2.1p18
```

위와 같이 create queue로 dque라는 디폴트 큐를 만들고, set server로 서버의 디폴트 큐를 dque로 셋팅했다. list server로 확인해 보면 위와 같이 default_queue가 dque가 됨을 볼 수 있다.

⑤ 만들어진 queue를 활성화한다.

```
Qmgr: s q dque enabled=true
Qmgr: s q dque started=true
Qmgr: l q dque
Queue dque
    queue_type = Execution
    total_jobs = 0
    state_count = Transit:0 Queued:0 Held:0 Waiting:0 Running:0 Exiting:0
    enabled = True
    started = True
```

이제 이미 만들어진 큐인 dque를 set queue를 사용하여 위와 같이 활성화해주고, list queue로 확인해 보면, 큐가 사용가능하고 이미 시작되었음을 알수 있다.

⑥ 이제 서버를 활성화하면 된다.

```
Qmgr: s s scheduling=true
Qmgr: l s
Server alpha
    server_state = Active
    scheduling = True
    total_jobs = 0
    state_count = Transit:0 Queued:0 Held:0 Waiting:0 Running:0 Exiting:0
    default_queue = dque
    log_events = 511
    mail_from = adm
    scheduler_iteration = 600
    pbs_version = 2.1p18
```

위와 같이 set server로 스케쥴링을 활성화하면 서버가 활동을 하게 된다. list server로 확인해 보면server_state가 Idle에서 Active로 바뀐 것을 볼 수 있다.

⑦ qsub로 job을 넣고 결과를 본다.

```
% more test.sh
#!/bin/sh
ls -l
```

위와 같이 간단한 쉘 스크립트를 만든다. 그리고 다음과 같이 qsub로 서버에 job를 던지면 된다.

```
% qsub test.sh
0.alpha
% qstat
Job id        Name          User          Time Use S Queue
0.alpha       test.sh       bsjung          00:00:00 R dque
```

확인은 위와 같이 qstat로 job의 지금 현재 상황을 볼 수 있다. 지금은 Use를 보면 실행 중임을 볼 수 있다. 결과는 test.sh.o0라는 결과 파일과 test.sh.e0라는 에러 파일을 볼 수 있다. test.sh.o0은 표준 출력파일이고, test.sh.e0는 표준 에러 파일이다

• PBS 서버 호스트에서 /usr/pbs/sbin/pbs_server와 /usr/pbs/sbin/pbs_sched 실행하고, 재부팅할 때마다 수행될 수 있도록 /etc/rc.local에 삽입한다.

PBS 설치와 설정이 완료된 후 qsub와 같은 PBS 명령어로 테스트 해 본다. 간단한 테스트는 아래와 같다.

```
$ qsub hostname
```

라. 사용자 지원 시스템 구축

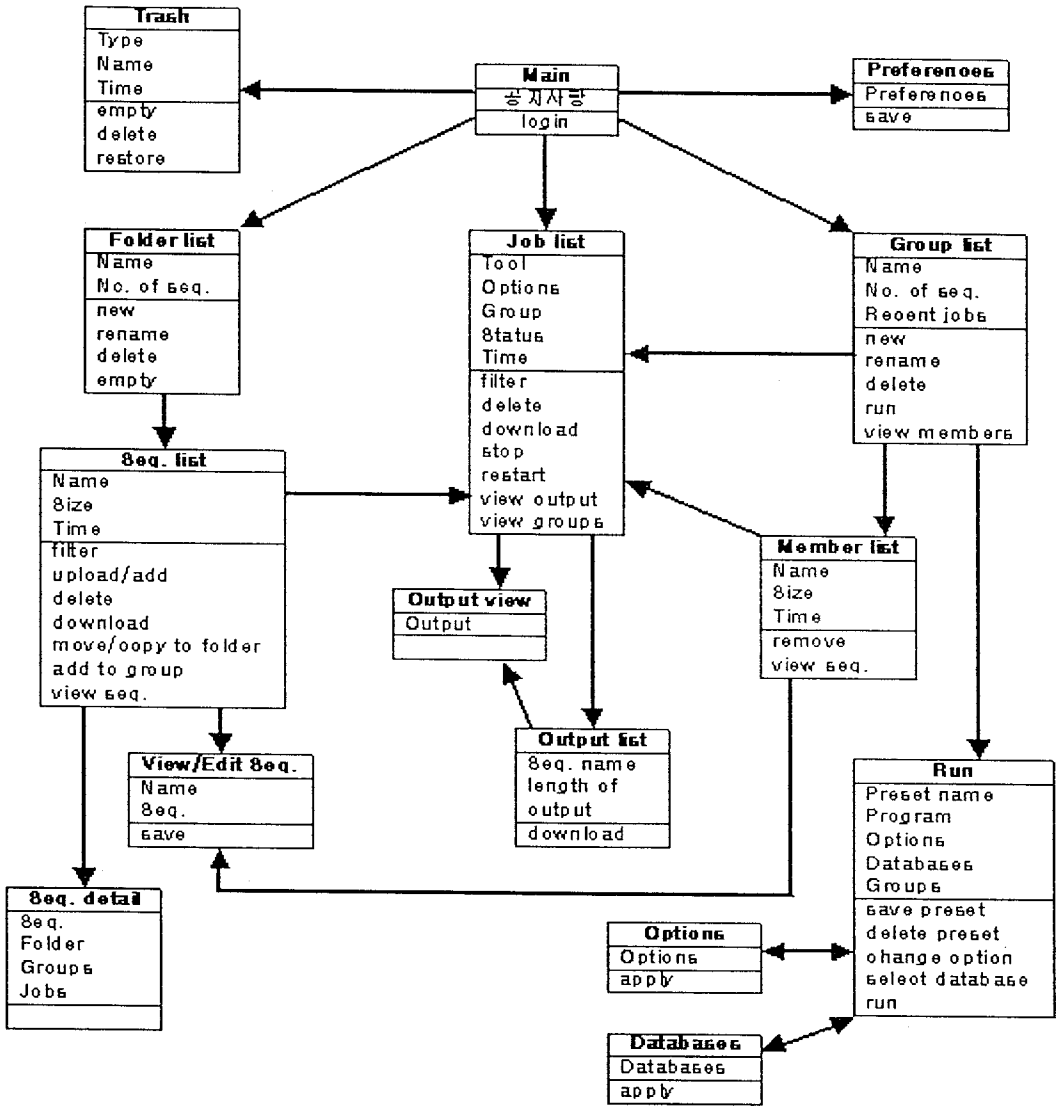
(1) 시스템 구성

사용자들이 바이오인포매틱스 서비스를 편리하게 이용할 수 있도록 웹 인터페이스로 개발하였다. Linux(PC Cluster)와 Tru64(Compaq SMP Cluster) 운영체제에 웹 서버로 Apache를 사용하였고, 주 개발 언어로 PHP(PHP: Hypertext Preprocessor)와 JavaScript, 데이터베이스로 MySQL을 이용하였다. 그러나 여러 웹 서버에 사용할 수 있는 PHP로 개발하였고 데이터베이스에 대한 질의는 다른 데이터베이스와 호환되도록 하여 특정 웹 서버나 OS, 데이터베이스에 종속되지 않도록 하였다. Cluster에서 실행되는 생물정보학 도구들은 사용자 데이터베이스에서 자료를 가져오고 다시 그 데이터베이스에 결과를 저장하도록 수정하였다.

(2) 웹 인터페이스 구성

사용자의 웹 인터페이스는 <figure 4-5>과 같이 구성되어 있다. 사용자의 필요에 따라 폴더를 생성하고 각각의 폴더에 자료를 입력하도록 하였다. 생물정보학 도구에서 사용할 자료들을 선택하여 그룹으로 묶은 다음 그룹 단위로 도구를 이용한다. 생물정보학 도구는 도구별로 선택사항을 고를 수 있도록 하고 자주 사용하는 옵션은 이름을 붙여 저장하였다가 나중에 다시 사용할 수 있도록 하였다. 또한 여러 그룹을 같은 선택사항으로 일괄적으로 작업시킬 수 있다. 작업 결과는 '자료 목록'이나 '그룹 목록' 또는 '작업 목록' 등에서 확인할 수 있다. 필요 없는 자료, 그룹, 작업 결과 등을 삭제

하면 임시로 휴지통으로 이동시켜, 사용자가 확인한 후 완전히 삭제하도록 하였다. 그 밖에 사용자의 편의를 위한 설정이나 개인 정보 등을 변경할 수 있도록 하였다.



<figure 6-9> Web Page Diagram

'폴더 목록' 페이지에서는 폴더 목록을 볼 수 있고 폴더 생성, 삭제, 이름 변경 등의 기능을 수행할 수 있다. 특정 폴더를 선택하면 폴더에 속한 염기서열 자료를 목록으로 볼 수 있고 자료 입력, 삭제, 검색, 이동, 복사 등의 기능을 수행할 수 있다. 폴더에 저장된 염기서열 데이터를 특정 조건을 주어 검색하여 선택하거나 목록에서 직접 선택하여 하나의 그룹으로 묶어서 작업을 수행하므로 이 화면에서 자료를 선택하여 그룹으로 묶고 그룹 목록 화면에서 작업을 실행한다. 염기서열 자료를 저장하고자할 때에는 사용자의 컴퓨터에 저장된 파일을 upload하거나 이름과 함께 염기서열을 입

력하면 새로 추가하여 저장할 수 있다.

'그룹 목록' 페이지에서는 그룹 목록과 각 그룹을 이용하여 최근에 실행한 작업 목록을 보여준다. 이 페이지에서 그룹을 생성, 삭제하고 이름을 변경 할 수 있다. 또한 특정 그룹을 선택하면 그룹에 속한 자료들의 목록을 볼 수 있다. 사용하려는 생물정보학 도구와 선택사항, public database 등을 선택하고 하나의 작업을 몇 개의 노드에 나누어(또는 자료를 몇 개 단위로 나누어) 병렬로 실행시킬 것인가를 정한 뒤 작업을 실행시킬 수 있다.

'작업 목록' 페이지에서는 수행한 작업들의 목록을 볼 수 있으며 작업 검색, 클라이언트 PC로의 자료 저장 등의 기능이 있다. 실행중인 작업은 사용자의 필요에 따라 정지시키거나 정지된 작업을 재시작할 수도 있다.

'휴지통' 페이지는 지운 폴더, 그룹, 작업등을 볼 수 있고 원하는 경우 지운 자료를 복원하거나 데이터베이스에서 실제 자료를 삭제할 수 있다.

'설정' 페이지에서는 한 페이지에 보여 줄 자료 개수나 정렬 순서 등의 각종 설정 값 및 개인 정보 등을 변경한다.

(3) 작업 관리

데이터베이스는 사용자마다 별도로 유지된다. 사용자가 웹 인터페이스를 이용하여 입력한 자료는 [그림 5.4]에서와 같이 사용자 데이터베이스의 자료(seq) 테이블에 저장된다. 폴더(folder)는 자료를 구분하기 위한 가상의 저장 공간에 해당한다. 자료들에 대해 생물정보학 도구를 이용하여 정보를 얻고자 할 때에는 선택한 자료들을 하나의 그룹(grp)으로 모아서 작업을 실행하는데, 이 때 어떤 자료가 어느 그룹에 속하는지는 seqgrp 테이블을 이용하여 나타낸다.

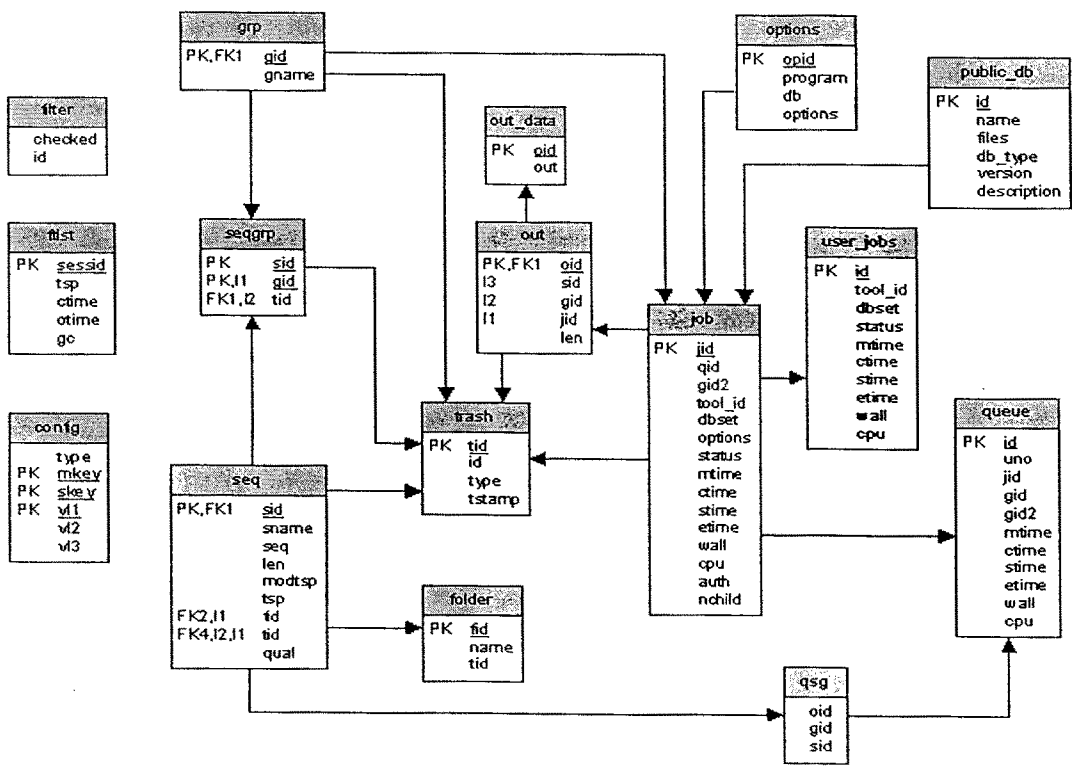
작업(job) 테이블에는 사용자가 생물정보학 도구를 이용할 때 사용한 그룹, 선택사항, 시간정보 등을 저장한다. 도구를 실행시키면 이를 자료 개수나 cluster의 노드 수에 따라 여러 개의 하위 작업으로 나누어 별도의 데이터베이스(queue)에 필요한 정보를 저장한다. 하위 작업으로 자료들을 나누기 위해 seqgrp와 비슷한 qsg 테이블을 이용하여 여러 개의 하위 그룹으로 자료들을 분산시킨다. Job management server가 생물정보학 도구를 cluster system의 각 노드에서 실행시키면 이 하위 작업에 대한 정보를 읽어 필요한 작업을 하게 된다. 작업 결과는 사용자의 결과(out, out_data) 테이블에 저장된다.

모든 하위 작업이 끝나게 되면 사용자의 작업(job) 테이블과 전체 사용자들의 작업에 대한 데이터베이스(user_jobs)에 정보를 갱신한다. 관리자는 수행중인 작업은 queue 데이터베이스로부터, 이미 끝난 작업은 user_jobs 데이터베이스로부터 그 정보를 얻을 수 있다.

'휴지통'에 해당하는 trash 테이블은 각 자료, 그룹, 작업등이 지워질 때 실제 자료를 지우는 대신 각 테이블에 지워졌는지 여부를 나타내는 tid 값을 설정하고 이에 대한 정보를 저장한다. 복원할 경우에는 이 정보를 토대로 원래대로 되돌리고, 완전히 지우는 경우에는 실제 자료와 trash에 저장한 정보를 함께 삭제한다.

그 밖에 생물정보학 도구들의 선택사항(options)을 저장하는 테이블, 웹 인터페이스에서 한 화면에 표시할 정보의 개수나 정렬 순서 등의 설정 정보(config)를 저장하는 테이블이 있다. 'filter' 테이블은 웹 인터페이스의 테이블들에서 선택한 항목을 나타내기 위해 사용한다.

각 사용자에 대한 정보는 전체 사용자에 대한 테이블(p__info 데이터베이스의 user_info)에서 얻는다. 사용자의 로그인 정보는 웹 서버의 session 정보와 함께 ftlst 테이블에 저장되고 로그인할 때마다 정보가 정리된다.



<figure 6-10> Relationship Diagram

(4) 기대 효과

바이오인포매틱스 서비스 지원 시스템은 컴퓨터 지식이 부족한 생물학 분야 사용자에게 일관되고 편리한 사용자 인터페이스를 제공함으로써 PC Cluster는 물론 Compaq SMP Cluster를 별도의 전문지식 없이 쉽게 사용할 수 있다. 또한 사용자 관리 기능과 효율적인 시스템 운영 기능을 제공함으로써 시스템 관리 부담을 크게 줄일 수 있다. 국내외 바이오인포매틱스 연구자들이 BLAST, FASTA, ClustalW 등의 프로그램을 이용하여 대량의 데이터를 처리하는 데 유용하게 사용될 수 있으며, 염기서열 데이터 및 작업 결과 저장소 및 browser로도 활용될 수 있어 바이오인포매틱스 연구의 효율을 크게 증대시킬 수 있을 것으로 기대된다.

마. 어카운팅 시스템

어카운팅 시스템은 네트워크 서비스 시스템 자원에 대한 상태 및 사용여부, 사용자에 대한 인증 및 접근통제 그리고 서비스 자원에 대한 비용산출을 계산하는 시스템을 말하는데 이 과제에서는 리눅스 기반의 어카운팅 시스템의 프로토 타입을 개발 적용하였다.

(1) 분류

account 디렉토리 : 클러 스터 노드의 정보를 수집 하기위한 프로그램

psa.pl 사용자별 사용량을 수집 /var/account/년/월/pacct_일 형식으로 암호화 된다.
real_psa.pl 실시간 사용자별 사용량을 수집 /var/account/년/월/real_pacct_일 형식으로 암호화 된다.
disk_usage.pl 사용자별 디스크 사용량을 수집 /var/account/년/월/hdd_used_일 형식으로 암호화 된다.
real_disk_usage.pl 실시간 사용자별 디스크 사용량을 수집 /var/account/년/월/real_hdd_used_일 형식으로 암호화 된다.
con.pl 사용자별 접속 사용량을 수집 /var/account/년/월/con_일 형식으로 암호화 된다.

real_con.pl 실시간 사용자별 접속 사용량을 수집 /var/account/년/월/real_con_일 형식으로 암호화 된다.

pbs-account batch job 의 사용량을 수집

- /var/account/년/월/SEEDQ_LOG.log 형식으로 암호화 된다.
- /var/account/년/월/SEEDS_LOG.log 형식으로 암호화 된다.
- /var/account/년/월/SEEDD_LOG.log 형식으로 암호화 된다.
- /var/account/년/월/SEEDE_LOG.log 형식으로 암호화 된다.

rpbs-account 실시간 batch job 의 사용량을 수집

- /var/account/년/월/SEEDRQ_LOG.log 형식으로 암호화 된다.
- /var/account/년/월/SEEDRS_LOG.log 형식으로 암호화 된다.
- /var/account/년/월/SEEDRD_LOG.log 형식으로 암호화 된다.
- /var/account/년/월/SEEDRE_LOG.log 형식으로 암호화 된다.

agent 디렉토리

node 의 account 에서 수집된 사용량을 server 로 옮겨준다.

server 디렉토리

node에서 account를 수집해서 mysql db(manager) 에 정보를 담아 둔다.
client 의 요청에 의해서 mysql db(manager)에서 정보를 빼서 보내준다.

client 디렉토리

server 에 요청하여 정보를 받아 와서 mysql db(account) 에 정보를 저장 한다.

ui 디렉토리

mysql db(account) 에 들어 있는 정보를 가져와서 암호화를 풀고 웹 화면에 정보를 보여 준다.

dbschema 디렉토리

mysql db (manager) 와 mysql db (account)의 db 스키마가 들어 있다.

(2) 설치

ACCT.tar.gz를 /usr/local/ 에 압축을 푼다
/usr/local/ACCT/servd를 /etc/rc.d/ini.d/ 로 옮긴다.

/etc/syslog.conf를 열어서
local5.* /var/log/servd.log을 추가 한다.
/etc/services를 열어서
servd 7777/udp
servd 7777/tcp 를 추가 한다.

/etc/rc.d/init.d/syslog stop
/etc/rc.d/init.d/syslog start

/etc/rc.d/init.d/servd start

관리툴 server가 작동한다.

crontab -e
00 23 * * * /usr/local/ACCT/account/disk_usage.pl { server ip }
00 23 * * * /usr/local/ACCT/account/con.pl { server ip }
00 23 * * * /usr/local/ACCT/account/psa.pl { server ip }
* 1-24/4 * * * /usr/local/ACCT/account/real_disk_usage.pl { server ip }
1-60/2 * * * * /usr/local/ACCT/account/real_con.pl { server ip }
1-60/2 * * * * /usr/local/ACCT/account/real_psa.pl { server ip }

pbs server 가 있는 노드에서는
/usr/local/ACCT/account/pbs-account 와 rpbs-account를 열어서 1475 라인의 서버 아이피를 현제 서버의 아이피에 맞게 고친다.

00 23 * * * /usr/local/ACCT/account/pbs-account { server ip }
* 1-24/2 * * * /usr/local/ACCT/account/pbs-account { server ip }

NODE 에는 /usr/local/ACCT 안의 account , agent , seed 가 있어야 한다.
MANAGER 서버에서는 /usr/local/ACCT/ 안의 server 가 있어야 한다.
ACCOUNT 서버에서는 /usr/local/ACCT/ 안의 client , ui, seed 가 있어야 한다.

(3) DB 스키마

MANAGER 서버의 스키마는 /usr/local/ACCT/dbschema/ 안의 Mcreate.sql 파일에 정의 되있다.
ACCOUNT 서버의 스키마는 /usr/local/ACCT/dbschema/ 안의 Acreate.sql 파일에 정의 되있다.

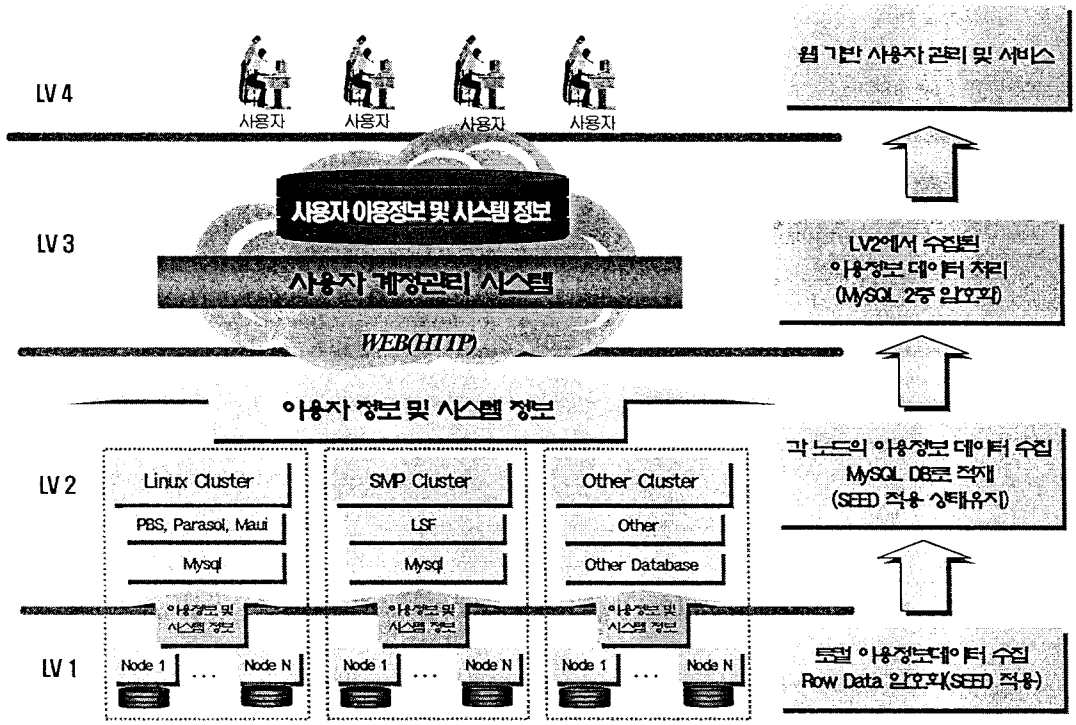
ACCOUNT 서버에서 /usr/local/ACCT/client/cli를 작동 시켜 서버의 자료를 가져와서 account 서버의 mysql db account 에 넣기 위해서
crontab -e
59 23 * * * /usr/local/ACCT/client/cli { server ip } CPU_INFO
59 23 * * * /usr/local/ACCT/client/cli { server ip } DISK_INFO
59 23 * * * /usr/local/ACCT/client/cli { server ip } CON_INFO
59 23 * * * /usr/local/ACCT/client/cli { server ip } PBS_QINFO
59 23 * * * /usr/local/ACCT/client/cli { server ip } PBS_SINFO
59 23 * * * /usr/local/ACCT/client/cli { server ip } PBS_DINFO
59 23 * * * /usr/local/ACCT/client/cli { server ip } PBS_EINFO

1-60/5 * * * * /usr/local/ACCT/client/cli { server ip } REAL_CPU_INFO
* 1-24/4 * * * /usr/local/ACCT/client/cli { server ip } REAL_DISK_INFO
1-60/1 * * * * /usr/local/ACCT/client/cli { server ip } REAL_CON_INFO
* 1-24/4 * * * /usr/local/ACCT/client/cli { server ip } PBS_RQINFO
* 1-24/4 * * * /usr/local/ACCT/client/cli { server ip } PBS_RSINFO
* 1-24/4 * * * /usr/local/ACCT/client/cli { server ip } PBS_RDINFO
* 1-24/4 * * * /usr/local/ACCT/client/cli { server ip } PBS_REINFO

(4) 최종 구성

이 과제에서는 시스템 구조를 크게 3가지로 나누어 고려하였다. 첫째, 타인의 시스템 자원을 임의로 사용 방지를 통해 형평성 있는 요금부과와 실시간 시스템 자

원 사용 현황 파악 및 통계 자료 작성을 위해 클러스터 정보를 수집, 둘째, 시스템에 대한 신뢰성 저하로 인한 사용자 감소 및 정보 유출의 우려를 최소화하기 위해 국내 표준 암호화 알고리즘 SEED를 적용, 셋째, 사용자 접근의 편의성과 자원 관리의 편의성을 고려한 웹 기반 어카운팅 시스템으로 크게 세부분으로 나누어 설계해서 최종적으로 <그림 7-7>과 같은 시스템 구성을 가질 수 있도록 하였다.



<feagure 6-11> 어카운팅 시스템 구성도

4. 기대효과

가. 시스템 구성 바이오인포매틱스 기반 시스템 도입으로 생물정보 분석시스템의 병렬화에 따른 속도 및 성능 향상

나. 대량의 염기서열 데이터 처리 및 사용자의 시스템 사용 편이성 향상

다. 국내 유일의 바이오인포매틱스 종합 정보기반 구축으로 산·학·연에 연구개발 인프라 제공

라. 생물정보 전용 시스템 구축으로 시스템 이용 효율 증대

마. BT와 IT 기술의 접목으로 새로운 IT 산업의 응용 분야 창출

제 4 장 목표달성도 및 기대효과

1. 목표달성도

목 표	달성도 (%)	내 용
생물정보검색 기반 시스템 개발 - GenBank 검색시간: 평균5초	100	- 대용량 주석처리 시스템 개발 - 결과집합 관리시스템 개발 - GenBank DB에 적용시 평균 1.7초
단백질 아미노산서열 색인기법 개발	100	- 아미노산 서열색인기 개발 - 서열 검색시스템 개발
단백질 서열 검색시스템 개발 - NREF DB 구축 110만건 - 평균 검색시간 10초	100	- NREF 데이터베이스 대상의 NREF-CCBB 서비스 구축 - NREF DB 130만건/평균검색시간 6.0초
색인기반 단백질 분류 시스템 개발 - Superfamily 데이터 50만건	100	- ProFaC(PROtein FAmily Classification) 서비스: 단백질 Superfamily 분류데이터 서비스 시스템 구축 - Superfamily 분류데이터 60만건 구축
3차원비교가시화 S/W 개발 - 대형구조 초기화시간: 2초	100	- 비교가시화 도구 개발 - HTTP를 통한 PDB 데이터베이스 연동 - 대형 구조 초기화 시간: 1.8초
국가유전체정보센터 통합 홈페이지 구축	100	- 주요 생물정보 콘텐츠들의 국내 서비스 체제 구축 - 데이터베이스 및 콘텐츠 통계 시스템 구축
유전체정보센터 인프라 고도화	100	- 내부 네트워크 성능 향상 - 스토리지 증설 - 클러스터 시스템 증설
신규 데이터베이스 구축	100	- dbSNP는 총 473,065,541개 입력 완료 - REBASE는 총 7,110개 자료 입력 완료 - BIND DB 구축 - DIP DB 구축
데이터 최신성 유지	95	- Ensemble, GenBan, PIR, Swiss-Prot, REBASE, dbSNP 등의 최신성 유지 탁월 - SCOP의 실시간 미러링 - CATH 업데이트
기존 DB GUI 및 기능 강화	95	- 텍스트 위주의 디자인으로 빠른 응답이 가능함
한글화에 의한 사용 환경 개선	95	- 한글설명 추가로 초보자도 용이한 사용가능 - 한글 HELP 기능 추가
국내 BT 분야에서 사용빈도가 높은 분석 도구의 신규 구축	95	- 각 시스템을 설치하고 한글화된 웹 인터페이스를 구축하여 국내 사용자들에게 편의성 제공
클러스터 시스템 기반 생물정보 분석시스템의 고속화 서비스	100	- 프로그램의 최적화로 10~33%의 성능 향상 - 프로그램의 병렬화로 클러스터 시스템에서 사용하는 노드 수에 비례하는 성능 향상 달성
생물정보 자동 마이닝 웹서비스	100	- KISTI의 128 node Cluster 셋팅 - SRS의 배치처리 프로그램 제작완료 - Web Services를 이용한 Blast서비스 제작 완료
클러스터시스템 관리시스템구축	100	- 작업상황을 표로 정리하여 관찰가능 - 에러상황 표시 - 검색기능 제공으로 원하는 기간동안 작업 통계를 손쉽게 파악할 수 있음 - 그래프를 이용하여 시각적인 상황 파악 용이함

2. 기대 효과

2.1 기술적 효과

- 바이오인포매틱스 핵심 및 기반기술의 축적을 통해 생명공학기술 및 바이오인포매틱스 인프라 구축으로 바이오인포매틱스 전 분야의 발전에 기여할 수 있다.
- 현재 생명현상에 중요한 역할을 하는 유전자는 특허를 내는 추세이므로, 국내에서 개발한 검색시스템을 사용하여 국외의 다른 연구자들보다 새로운 유전자 또는 신약 후보물질 발견에 앞설 수 있으므로 제약 및 생명공학 분야에 막대한 기여가 가능하다.
- 초고속네트워크 구축 및 운영의 노하우를 가지고 있는 KISTI내 초고속연구망부와 연계하여 빠르고 안정적인 대용량의 FTP 사이트 구축 및 운영으로 이용자 만족도를 극대화할 수 있다.
- 국내 유전체 연구 기관간의 정보 네트워크를 구축하여 효율적인 유전체 연구기반을 제공할 수 있다.
- 유전자 분석 기술 개발로 생명공학 연구발전에 기여할 수 있다.
- 유전체 연구관련 국내외 연구동향의 체계적인 수집, 분석 및 지원으로 효율적인 기술개발 기반 마련의 효과가 크다.
- 국내 고유의 유전자원 관리 및 개발에 대한 독자적인 시스템 구축을 통해 기술력 축적은 물론 물적 자원의 해외유출을 막을 수 있다.
- 기존의 생물정보 데이터베이스 검색 시스템의 단점을 보완한 새로운 개념의 통합 검색 관리 시스템 개발로 기존 연구자의 연구능력 향상을 야기함과 동시에 세계적으로 독창적인 검색 시스템으로의 자리 매김이 가능하다.
- 연구중인 유전자/단백질과 유사한 유전자/단백질 연구 결과를 쉽게 검색할 수 있어 연구의 효율성이 증대된다.

2.2 사회경제적 효과

- 외국에 비해 연구 능력이 미비한 바이오인포매틱스 전반적인 분야의 핵심 및 기반 기술의 축적과 전문인력 양성, 이를 통한 미래 바이오인포매틱스 연구의 선도적인 역할을 수행할 수 있다.
- 대부분을 외국에 의존하고 있는 데이터베이스와 분석 소프트웨어중 상업화 가능성이 크고 핵심 기술을 포함하는 것들을 국산화하여 수입대체 효과를 거둘 수 있다.
- 관련 기초과학 분야(생물학, 물리, 화학, 수학, 통계학 등)와 응용과학 및 보건분야(의학, 약학) 발전에 기여할 수 있다.
- 유전체 정보 DB 구축 및 서비스로 지속적인 연구기반을 제공하여 국가 경쟁력을 높일 수 있다.
- 국내외 유전체 연구자들의 유기적인 네트워크 구축에 따른 국내 유전체 연구의 국제 경쟁력 강화와 공동연구 및 정보교환, 소재 공동 활용등에 기여할 수 있다.
- 인간, 동식물, 미생물의 유전체 종합정보의 통합 DB 구축 및 공동 활용 기반 구축으로 산·학·연 유전체 연구의 발전기반 제공한다.
- 개발 중인 바이오인포매틱스 정보검색 프로그램 각각은 개별적으로 상품화가 가능하므로 국내 산업 경쟁력을 높이는데 기여할 수 있다.
- 상용 데이터베이스를 사용할 경우에 비해 연구비용 절감효과가 있다
- 국내에서 생산되는 바이오인포매틱스 데이터들을 데이터베이스화 할 수 있으므로 우리나라 특성에 맞는 바이오인포매틱스 연구 수행에 기여할 수 있다.
- 외국 생물정보 데이터베이스들이 유료화될 경우 업무협력시 유리한 여건에서 협상이 가능하다.
- 국내에서 산출된 데이터를 이용하여 특허 및 기술개발시 외국의 데이터베이스를 이용하지 않아도 되므로 데이터의 보안과 안전성 및 연구효율성을 증대시킨다.
- 신약 개발대상 생물자원, 신약후보물질, 신약후보물질 탐색시스템의 라이센싱 및 독자적인 신약개발로 경제적인 수익과 기업 경쟁력을 강화할 수 있다.

제 5장 연구개발결과의 활용계획

o BT와 IT의 융합으로 유전체, 단백질체 관련 국내 BT 산업 및 새로운 drug target을 찾는 제약업계와 생명공학기업의 국제 경쟁력을 증대시킨다.

o 인간, 동·식물, 미생물의 유전체 종합정보의 통합 DB 구축 및 공동 활용 기반 구축으로 고부가가치형 생물정보의 인프라 서비스가 가능하다.

o 생명체속의 유전자 네트워크 및 대사경로를 분석함으로써 컴퓨터상에서 세포 생리 및 병리현상을 모형화할 수 있어, 실험과 병행된 연구를 진행하는 physiome 시대로의 패러다임 변화에 대비할 수 있다.

o 바이오인포매틱스를 수행하기 위하여 IT 분야에서 핵심기반 S/W개발을 하도록 유도하며 이의 보급을 통한 IT 분야의 기술집적도 향상 및 고성능 컴퓨터 활용 기술개발을 가져올 수 있다.

o 연구과제 수행 및 개발과정을 통하여 바이오인포매틱스 분야의 고급인력을 양성하여 산·학·연 전반으로 생물정보학의 저변 확대를 가져올 수 있다.

제 6장 연구개발과정에서 수집한 해외과학기술정보

"해당사항 없음"

제 7장 참고문헌

1. Ki-Bong Kim, Hwajung Seo, Hyeweon Nam, Hongseok Tae, Pan-Gyu Kim, Daesang Lee, Haeyoung Jeong, and Kiejung Park, 2001, An Integrated Sequence Data Management and Annotation System For Microbial Genome Projects. ISMB 2001.

2. Altschul, S.F., W. Gish, W. Miller, E.W. Myers and D.J. Lipman, 1990. Basic local alignment search tool. J. Mol. Biol. 215, 403-410.

3. Altschul, S.F., T.L. Madden, A.A. Schaffer, J. Zhang, Z. Zhang, W. Miller and D.J. Lipman, 1997. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Research. 25, 3389-3402

4. Bansal, A.K., P. Bork, and P.J. Stuckey, 1998. Automated pairwise comparisons of microbial genomes. Math. Modelling and Sci. Computing. 9, 1-23.

5. Bansal, A.K., 1999. An automated comparative analysis of 17 complete microbial genomes. Bioinformatics. 15, 900-908.

6. Bateman, A., E. Birney, L. Cerruti, R. Durbin, L. Etwiller, S.R. Eddy, S.G. Jones, K.L. Howe, M. Marshall, and E.L.L. Sonnhammer. 2002 The Pfam Protein Families Database Nucleic Acids Research. 30, 276-280.

7. Blattner, F.R., G. Plunkett, C.A. Bloch, N.T. Perna, Y. Shao, et al. 1997. The Complete Genome Sequence of Escherichia coli K-12. Science. 277, 1453-1462.

8. Delcher, A.L., S Kasif, R.D. Fleischmann, J. Peterson, O. White, and S.L. Salzberg 1999. Alignment of whole genomes. Nucleic Acids Research. 27, 2369-2376.

9. Delcher, A.L., A. Phillippy, J. Carlton and S.L. Salzberg, 2002. Fast algorithms

for large-scale genome alignment and comparison. Nucleic Acids Research. 30, 2478-2483.

10. Fleischmann, R.D., M.D. Adams, O. White, R.A. Clayton, E.F. Kirkness, A.R. Kerlavage, C.J. Bult, J.F. Tomb, B.A. Dougherty, and J.M. Merrick, 1995. Whole-genome random sequencing and assembly of Haemophilus influenzae Rd. Science. 269, 496-512.

11. Folrea, L., C. Riemer, S. Schwartz, Z. Zhang, N. Stojanovic, W. Miller and M. McClelland, 2000. Web-based visualization tools for bacterial genome alignments. Nucleic Acids Research. 20, 3486-3496.

12. Hayashi, T., K. Makino, M. Ohnishi, K. Kurokawa, K. Ishii, K. Yokoyama, C.G. Han, E. Ohtsubo, K. Nakayama, T. Murata, M. Tanaka, T. Tobe, T. Iida, G. Takami , T. Honda, C. Sasakawa, N. Ogasawara, T. Yasunaga, S. Kuhara, T. Shiba, M. Hattori, H. Shinagawa. 2001. Complete genome sequence of enterohemorrhagic Escherichia coli O157:H7 and genomic comparison with a laboratory strain K-12. DNA Res. 8, 11-22.

13. Heidelberg J.F., J.A. Eisen, W.C. Nelson, R.A. Clayton, M.L. Gwinn, R.J. Dodson , D.H. Haft, E.K. Hickey, J.D. Peterson, L. Umayam, S.R. Gill, K.E. Nelson, T.D. Read, H. Tettelin, D. Richardson, M.D. Ermolaeva, J. Vamathevan, S. Bass, H. Qin, I. Dragoi, P. Sellers, L. McDonald, T. Utterback, R.D. Fleishmann, W. C. Nierman, O. White, S.L. Salzberg, H.O. Smith, R.R. Colwell, J.J. Mekalanos, J.C. Venter, C.M. Fraser. 2000. DNA sequence of both chromosomes of the cholera pathogen Vibrio cholerae. Nature. 406. 477-483.

14. Mayor, C., M. Brudno, J.R. Schwartz, A. Poliakov, E.M. Rubin, K.A. Frazer, L.S. Pachter, and I. Dubchak. 2000. VISTA : visualizing global DNA sequence alignments of arbitrary length. Bioinformatics. 16, 1046-1047.

15. Pearson, W.R., 1990. Rapid and Sensitive Sequence Comparison with FASTP and FASTA. Methods Enzymol. 183, 63-98.

16. Pearson,W.R., 2000. Flexible similarity searching with the FASTA3 program package. Methods Mol. BioI. 132, 185-219.

17. Roten, C..H., P. Gamba, J. Barblan, and D Karamata. 2002. Comparative Genometrics (CG): a database dedicated to biometric comparisons of whole genomes Nucleic Acids Research. 30. 142-144.

18. Schwartz S., Z. Zhang, K.A. Frazer, A. Smit, C. Riemer, J. Bouck, R. Gibbs, R. Hardison, and W Miller. 2000. PipMaker. A Web Server for Aligning Two Genomic DNA Sequences. Genome Research. 10, 577-586.

19. Takami, H., K. Nakasone, Y. Takaki, G. Maeno, R. Sasaki, N. Masui, F. Fuji, C. Hirama, Y. Nakamura, N. Ogasawara, S. Kuhara, and K. Horikoshi. 2000 Complete genome sequence of the alkaliphilic bacterium Bacillus halodurans and genomic sequence comparison with Bacillus subtilis. Nucleic Acids Research. 28, 4317-4331.

20. Tatusov R.L., M.Y. Galperin, D.A. Natale, E.V Koonin. 2001 The COG database: a tool for genome-scale analysis of protein functions and evolution. Nucleic Acids Research. 29, 22-28.

21. Walker, M., V. Pavlovic and S. Kasif. 2002 A comparative genomic method for computational identification of prokaryotic translation initiation sites. Nucleic Acids Research, 30, 3181-3191.

22. Xie, H., A. Wasserman, Z. Levine, A. Novik, V. Grebinskiy, A. Shoshan, and L Mintz. 2002. Large-Scale Protein Annotation through Gene Ontology. Genome Research. 12, 785-794.

23. Zdobnov, E.M., C. Mering, I. Letunic, D. Torrents, M. Suyama, R.R. Copley, G.K. Christophides, D. Thomasova, R.A. Holt, G.M. Subramanian, H.M. Mueller, G. Dimopoulos, J.H. Law, M.A. Wells, E.Birney, R. Charlab, A.L. Halpern, E.

Kokoza, C.L. Kraft, Z. Lai, S. Lewis, C. Louis, C. Barillas-Mury, D. Nusskern, G.M. Rubin, S.L. Salzberg, G.G. Sutton, P. Topalis, R. Wides, P. Wincker, M. Yandell, F.H. Collins, J. Ribeiro, W.M. Gelbart, F.C. Kafatos, and P. Bork. 2002. Comparative Genome and Proteome Analysis of Anopheles gambiae and Drosophila melanogaster. Science. 298, 149-159.

특정연구개발사업 연구결과 활용계획서

사업명	중사업명	국책연구개발사업		
	세부사업명	국가유전체정보 DB 구축 및 기반기술개발 사업		
과제명		IT 기반 바이오인포매틱스 인프라 구축 및 응용연구		
연구기관		한국과학기술정보연구원	연구책임자	홍 순 찬
총연구기간		2002년 12월 1일 ~ 2004년 6월 30일 (19개월)		
총 연구비 (단위 : 천원)		정부출연금	민간부담금	합계
		1,700,000	0	1,700,000
기술분야		100 정보산업분야 /400 생명과학분야		
참여기업		*해당사항 없음*		
공동연구기관		*해당사항 없음*		
위탁연구기관		*해당사항 없음*		
연구결과활용 (해당항목에(∨) 표시)		1. 기업화 ( )	2. 기술이전( )	3. 후속연구추진(∨)	4. 타사업에 활용( )
		5. 선행 및 기초연구( )	6. 기타목적활용 (교육,연구)( )	7. 활용중단(미활용)( )	8. 기타( )

특정연구개발사업 처리규정 제 31조(연구개발결과의 보고) 제 2항에 의거 연구결과 활용계획서를 제출합니다.

첨부 : 1. 연구결과 활용계획서 1부.
2. 기술요약서 1부

2004년 6월 15 일

연구책임자 : 홍 순 찬
연구기관장 : 조 영 화

과학기술부장관 귀하

[첨부1]

연구결과 활용계획서

1. 연구목표 및 내용

유전체 연구의 진전은 BT와 IT의 융합으로 가능했으며, 이러한 융합화는 포스트 게놈 시대에 더욱 가속화될 전망이다. 바이오인포매틱스는 유전정보의 체계적 해석과 정보화 및 컴퓨터 시뮬레이션을 통한 효율성 증대 및 부가가치 창출을 목표로 한다. 이를 위해서 연구결과로 얻어진 유전체 정보를 DB화하고 분석 툴을 개발할 필요가 있으며, 이러한 일은 국내의 IT 인프라와 인력들을 활용하여 수행할 수 있다. 본 연구과제에서는 수행한 연구내용은 다음과 같다.

- ○ 국가 유전체정보센터 통합홈페이지 구축
- ○ 국내외 생물정보 DB 유지보수 및 신규 구축
- ○ 클러스터시스템기반 생물정보분석시스템의 고도화
- ○ 생물정보검색 시스템(Bio-KRISTAL) 개발
- ○ 3차원 비교가시화 소프트웨어 개발
- ○ 유전체정보센터 인프라 고도화

2. 연구수행결과 현황(연구종료시점까지)

가. 특허(실용신안) 등 자료목록

발명명칭	특허공고번호 출원(등록)번호	공고일자 출원(등록)일자	발명자 (출원인)	출원국	비고

나. 프로그램 등록목록

프로그램 명칭	등록번호	등록일자	개발자	비고

다. 노하우 내역

라. 발생품 및 시작품 내역

마. 논문게재 및 발표 실적

○논문게재 실적(필요시 별지사용)

학술지 명칭	제목	게재연월일	호	발 행 기 관	국 명	SCI게재 여부
		년 월 일				
계: 건수						

주: 국제 SCI 저널에 두 편을 투고하여 현재 심사중임.

○학술회의 발표 실적(필요시 별지사용)

학술회의 명칭	제목	게재연월일
2003 생물정보학심포지움	Protein Sequence Search based on N-gram Indexing	2003년 9월 17일
한국생물정보학회 (KOSBI)	ProSeS: Protein Sequence Search	2003년 10월 31일
한국생물정보학회 (KOSBI)	A Novel Predictor of Protein Subcellular Localization based on N-gram Features	2003년 10월 31일
KOSTI	N-gram Indexing for Protein Sequence Database	2003년 12월 5일
KOSTI	ProSLP:penta-gram 기반 단백질의 세포내 위치 예측 시스템	2003년 12월 5일
2004 생물정보학심포지움	정보기술을 이용한 단백질 서열분석	2004년 6월 4일
계: 건수		총 6건

3. 연구성과

본 연구과제에서는 기존 DB의 인터페이스를 보완하고 PIR, REBASE, dbSNP, Bind, DIP, CATH, dbSNP, Ensembl 등의 신규 생물정보 DB를 구축하였으며 NCBI와 SIB에서 제공되는 여러 DB들을 동시에 서비스하고 있다. 생물정보검색시스템(Bio-KRISTAL)을 개발하여 유전정보 검색에 활용하고 있다. 또한 클러스터시스템기반 생물정보분석시스템의 고도화를 실현하고자 국내 BT분야에서 많이 사용되는 Parallel BLAST, ClustalW, InterProScan 등을 신규로 구축하여 검색 서비스를 제공하고 있다. Dummy Analyzer라는

3차원 분자구조 가시화 소프트웨어를 개발하였다. 이외에도 FTP 사이트와 해외 DB의 미러 사이트를 운영하고 있으며, 바이오인포매틱스 연구자들에게 대용량 슈퍼컴퓨터와 PC 클러스터의 CPU 시간을 제공하고 있다.

4. 기술이전 및 연구결과 활용계획

가. 당해연도 활용계획

DB의 사용환경을 더욱 개선하고 사용자 편의성을 증진하여 연구자들이 외국 사이트보다 더 우선적으로 사용할 수 있도록 지원할 예정임. 이를 위해 해외의 중요 DB를 추가적으로 미러링할 계획임.

나. 활용방법

연구자들이 연구에 실질적인 도움이 되는 것은 물론이고, 바이오인포매틱스를 처음 접하는 초보자와 학생들도 쉽게 사용할 수 있고 강의/실습에도 사용할 수 있도록 기본적인 분석 툴과 자료를 추가할 예정임.

다. 차년도 이후 활용계획(6하원칙에 따라 구체적으로 작성)

KISTI에서는 2004년 하반기부터 SRS를 대체할 한국형 통합환경을 구축할 예정이다. 기존 구축 DB와 분석 툴들을 새로운 환경에 포함시킴으로써 사용자들이 더욱 쉽게 사용할 수 있도록 지원할 것이다.

5. 기대효과

유전체 종합정보 통합 DB 구축 및 공동활용기반의 구축으로 국내에서도 고부가가치형 생물정보인프라 서비스가 가능하게 되었다. 기존의 해외 DB 이용을 국내로 전환시킴으로써 중요한 연구정보의 해외유출을 막을 수 있고, 본격적인 포스트지놈 연구와 physiome 연구 등에도 크게 도움이 될 것으로 기대된다. 또한, BT 연구를 위한 IT 분야에서의 핵심 기반 소프트웨어 및 DB 관련 기술과 인력을 축적할 수 있었고, 이는 융합기술의 시대인 21세기 국가경쟁력 제고에 크게 기여할 수 있을 것으로 기대된다.

6. 문제점 및 건의사항(연구성과의 제고를 위한 제도·규정 및 연구관리 등의 개선점을 기재)

[첨부2]

기술 요약서

■ 기술의 명칭

정보검색기술을 기반으로 하는 대용량 생명정보 검색기법

■ 기술을 도출한 과제현황

과제관리번호	M1-0224-01-002			
과제명	IT 기반 바이오인포매틱스 인프라구축 및 응용연구			
사업명	국책연구개발사업			
세부사업명	국가유전체정보 DB 구축 및 기반기술개발 사업			
연구기관	한국과학기술정보연구원		기관유형	연구소
참여기관(기업)	*해당사항 없음*			
총연구기간	2002년 12월 1일 - 2004년 6월 30일			
총연구비	정부(1,700,000)천원 민간( )천원 합계(1,700,000)천원			
연구책임자 1	성명	홍 순 찬	주민번호	
	근무기관 부서	KISTI 생명정보시스템지원실	E-mail	schong@hpcnet.ne.kr
	직위/직급	책임연구원	전화번호	042-869-0537
실무연락책임자	성명	김 진 숙	소속/부서	KISTI 생명정보시스템개발실
	직위/직급	선임연구원	E-mail	jinsuk@kisti.re.kr
	전화번호	042-828-5144	FAX	042-828-5179
	주소	(305-333) 대전시 유성구 어은동 52번지		

■ 기술의 주요내용

[기술의 개요]

- GenBank 등과 같이 수천만 건에 이르는 방대한 데이터베이스에 대한 원활한 검색서비스를 제공하기 위해 결과 집합 관리기, 대용량 저장시스템, 불리안 검색모델 등을 이용한 생명정보검색 기법 개발

- 단백질 아미노산 서열 및 DNA 염기서열을 자연어 문서와 동일하게 처리할 수 있는 색인 기법 및 검색 기술 개발

<기술적 특징>

(1) 결과집합 관리기

미리 검색된 결과를 집합으로 저장하여 관리함으로써 대용량 검색시 검색 부하를 줄이면서 결과집합 연산 등과 같이 사용자 만족도를 높일 수 있는 기법 개발

(2) 대용량 저장시스템

GenBank 등과 같은 방대한 데이터베이스의 원문 및 색인 등을 저장할 수 있는 테라 바이트 급 데이터 저장용 저장시스템 구축

(3) 변형 불리안 검색 모델 개발

대용량 데이터베이스에서의 원활한 검색과 사용자 만족도를 충족시킬 수 있는 AND 기반 불리안 검색 모델

(4) 자연어 처리 기반 생물정보 서열 색인기법 및 검색기법

생물 서열을 자연어 텍스트와 동일한 방식으로 처리할 수 있는 색인 추출 알고리즘 및 검색 모델에 대한 연구 및 개발

[용도 · 이용분야]

(1) 향후 구축될 국내 대용량 생명정보 데이터베이스에 대한 고속 검색서비스 가능

(2) 생물정보 서열 색임 및 검색 기법을 이용한 고성능 신규 단백질 screening 시스템 구축

■ 기술의 분류

[기술코드] D 6 1 (KISTEP 홈페이지 기술요약서용 기술분류표 참조)

[기술분야] (1개만 선택(∨로 표시)하여 주십시오)

☐ 정보산업 ☐ 기계설비 ☐ 소재 ☐ 정밀화학·공정 ■ 생명과학
☐ 원자력 ☐ 자원 ☐ 에너지 ☐ 항공·우주 ☐ 해양
☐ 교통 ☐ 보건·의료 ☐ 환경 ☐ 기초·원천 ☐ 기타

[기술의 활용유형] (1개만 선택(∨로 표시)하여 주십시오)

☐ 신제품개발 ■ 신공정개발 ☐ 기존제품개선 ☐ 기존공정개선
☐ 기 타 ( )

[기술의 용도] (복수 선택(∨로 표시)가능합니다)

☐ 기계설비 ☐ 부품소자 ☐ 원료재료 ■ 소프트웨어
☐ 가공처리기술 ☐ 자동화기술 ☐ 불량률 감소 등 현장애로기술
☐ 제품설계기술 ☐ 공정설계기술 ☐ 기 타 ( )

■ 산업재산권 보유현황(기술과 관련한)

권리유형	명 칭	국가명	출원단계	일자	등록번호

* '권리유형'란에는 특허, 실용신안, 의장, 컴퓨터프로그램, 노하우 등을 선택하여 기재
* '출원단계'란에는 출원, 공개, 등록 등을 선택하여 기재

■ 기술이전 조건

이전형태	□ 유상 □무상	최저기술료	천원
이전방식	□ 소유권이전 □ 전용실시권 □ 통상실시권 □ 협의결정 □ 기타( )		
이전 소요기간	년 개월	실용화예상시기	년도
기술이전시 선행요건			

* 기술이전시 선행요건 : 기술이전을 위한 사전준비사항(필수 설비 및 장비, 전문가 확보 등)을 기술

* 실용화예상시기 : 기술을 활용한 대표적인 제품이 최초로 생산이 시작되는 시기를 기재

■ 기술의 개발단계 및 수준

[기술의 완성도] (1개만 선택(V로 표시)하여 주십시오)

	① **기초, 탐색연구단계** : 특정용도를 위해 필요한 신 지식을 얻거나 기술적 가능성을 탐색하는 단계
	② **응용연구단계** : 기술적 가능성의 실증, 잠재적 실용화 가능성의 입증 등 실험실적 확인 단계
√	③ **개발연구단계** : Prototype의 제작, Pilot Plant Test 등을 행하는 단계
	④ **기업화 준비단계** : 기업화에 필요한 양산화 기술 및 주변 기술까지도 확보하는 단계
	⑤ **상품화 완료단계**

[기술의 수명주기] (1개만 선택(V로 표시)하여 주십시오)

	① **기술개념 정립기** : 기술의 잠재적 가능성만 있는 단계
	② **기술실험기** : 기술개발에 성공했으나 아직 실용성, 경제성 등이 확실치 않은 단계
√	③ **기술적용 시작기**: 최초의 기술개발국에서만 활용되고 있는 단계
	④ **기술적용 성장기**: 기술개발국 및 일부 선진국에서 활용되고 있는단계
	⑤ **기술적용 성숙기**: 선진국사이에서 활발한 기술이전이 일어나며, 기술의 표준화가 되어가는 단계
	⑥ **기술적용 쇠퇴기**: 선진국에서 개도국으로 기술이전이 활발하게 일어나고, 선진국에서는 기술의 가치가 저하되나, 개도국에서는 아직 시장의 가치가 높은 기술

[기술발전 과정상의 기술수준] (1개만 선택(∨로 표시)하여 주십시오)

	① 외국기술의 모방단계 : 이미 외국에서 개발된 기술의 복제, reverse Eng.
√	**② 외국기술의 소화·흡수단계** : 국내시장구조나 특성에 적합하게 적응시킴
	③ 외국기술의 개선·개량단계 : 성능이나 기능을 개선시킴
	④ 신기술의 혁신·발명단계 : 국내 최초로 개발

■ 본 기술과 관련하여 추가로 확보되었거나 개발중인 기술

[기술개요]

기술명	
개발단계	**□ 연구개발 계획 □ 연구개발 중 □ 연구개발 완료**
기술개요	

[기술을 도출한 과제현황]

과제관리번호				
과제명				
사업명				
세부사업명				
연구기관			기관유형	
참여기관(기업)				
총연구기간				
총연구비	합계 : ( )백만원 - 정부 : ( )백만원 민간 : ( )백만원			
연구책임자	소속		성명	
	전화번호		E-mail	
연구개발 주요내용				