韓國컴퓨터情報學會 論文誌
第14卷 第2號, 2009. 2.

2009-14-2-3-8

고성능 BLAST 구현을 위한 E-Cluster 기반 데이터 분할 및 질의 라우팅 기법

김 태 경*, 조 완 섭**

A Physical Data Design and Query Routing Technique of High Performance BLAST on E-Cluster

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

요 약

BLAST는 생명정보학 분야에서 가장 많이 사용하는 도구이다. 이 도구는 입력서열을 기존 서열 데이터베이스와 신속히 비교하고 그 기능을 예측한다. 생물학자는 BLAST를 이용하여 실험의 범위, 시간과 비용을 줄일 수 있다. 하지만, 서열 데이터양이 급격히 증가함에 따라 그 처리 시간도 같이 증가하여 성능개선 방안이 필요하다. 본 논문에서는 대용량 BLAST 처리 성능 향상을 위한 PC 기반의 클러스터 인프라 (E-Cluster)를 제시하고 이 기반에서 데이터베이스 분할기법(Logical Partitioning)과 질의 라우팅 기법(Intra-Query)을 제안한다. 제안된 시스템을 평가하기 위해 다양한 길이의 서열들과 NR 데이터베이스와 비교하여 응답시간(Response Time), 성능 향상(Speedup), 효율(Efficiency) 관점에서 평가한다. 본 실험을 통해 기존 SMP, Cluster, 그리드 기반의 BLAST 시스템보다 성능, 효율이 뛰어남을 확인하였고, 특히 제안한 시스템의 최대 효율은 600%로 매우 높았다.

Abstract

BLAST (Basic Local Alignment Search Tool) is a best well-known tool in a bioinformatics area. BLAST quickly compares input sequences with annotated huge sequence databases and predicts their functions. It helps biologists to make it easy to annotate newly found sequences with reduced experimental time, scope, and cost. However, as the amount of sequences is increasing remarkably with the advance of sequencing machines, performance of BLAST has been a critical issue and tried to solve it with several alternatives. In this paper, we propose a new PC-Based Cluster system (E-Cluster), a new physical data design methodology (logical partitioning technique), and a query routing technique (intra-query routing). To verify our system, we measure response time, speedup, and efficiency for various sizes of sequences in NR (Non-Redundancy) database. Experimental result shows that proposed system has better speedup and efficiency (maximum 600%) than those of conventional approaches such as SMP machines, clusters, and grids.

▸ Keyword : BLAST, 서열분석, 바이오 인포메틱스 (Bioinformatics), 클러스터 시스템, 질의 라우팅(Query Routing)

• 제1저자 : 김태경 교신저자 : 조완섭
• 투고일 : 2009. 1. 14, 심사일 : 2009. 1. 22, 게재완료일 : 2009. 2. 23.
*충북대학교 정보산업공학과 **충북대학교 경영정보학과/u-Biz BK 사업팀
※이 논문은 2008년 교육과학기술부의 지원을 받아 수행된 연구임 (충북 BIT연구중심대학육성사업단)

Ⅰ. 서 론

생명공학 시퀀싱 기술의 발전으로 서열의 양이 1년에 2배 정도로 급격히 증가하고 있다[1]. 지금까지 916종의 시퀀싱 프로젝트가 완료되었고, 현재 3,454 종의 시퀀싱 프로젝트를 진행하고 있다[2]. GOLD 데이터베이스에 의하면 시퀀싱이 완료된 종은 전체 종의 5% 미만으로 앞으로 발생할 서열의 양은 막대할 것으로 예상하고 있다[2]. 현재 NR 데이터베이스는 4GB, NT 데이터베이스는 17GB 정도이며 매일 갱신되어 그 양이 지속적으로 증가하고 있다[1].

유전자 서열(아미노산, 단백질)은 유전체학, 단백체학, 시스템 생물학 등 생명 정보학 전 분야에서 가장 필수적인 데이터로 사용하고 있다[3]. 특히 새로 발견된 유전자 서열을 기존 서열과 비교하여 그 기능을 예측하여 생물학자에게 실험의 범위를 줄일 뿐만 아니라 시간과 비용을 절감할 수 있다.

BLAST (Basic Local Alignment Search Tool)는 컴퓨터를 이용하여 입력 서열을 대량의 서열과 신속하게 비교하는 생명정보학 전 분야에서 가장 많이 사용하는 도구이다[4,5]. 글로벌 정렬을 위한 Needleman-Wunsch 알고리즘[6]과 지역 정렬을 위한 Smith-Waternam[7]과 알고리즘을 기반으로 기존 다이내믹 프로그래밍 (Dynamic Programming) 보다 약 50배의 빠른 성능을 제공한다.

BLAST 개발로 말미암아 기존 수작업 보다 빨리 서열을 검색하게 되었지만, 서열 양의 증가로 그 성능 또한 급격히 저하되고 있다. 하루에 수십만 개의 서열을 대량의 데이터베이스 (NR, NT)와 비교하는 작업들이 지속적으로 발생하고 있지만, 컴퓨터 시스템이 사용자 요구를 수용하지 못하고 있다[9]. 현재 대부분의 생명공학 관련 연구소의 컴퓨팅 자원의 70%~90% 정도를 서열 비교용으로 사용하고 있으며, 처리 성능은 연구자의 요구를 만족시키지 못하고 있다[8].

이러한 문제를 극복하기 위해 BLAST의 성능 향상을 위한 다양한 연구가 진행되었다. ⅰ) 다수의 CPU를 가진 단일 SMP 컴퓨터를 이용하는 기법과 ⅱ) 다수의 컴퓨팅 노드를 LAN 내에서 묶은 클러스터를 이용하는 방식, 그리고 ⅲ) 인터넷상의 자원을 이용한 그리드 기반의 기법이 소개되었다. 각 접근 방식은 비용, 성능, 확장성, 관리, 안정성 측면에서 장단점이 있다.

SMP (Symmetric Multi-Processing) 또는 NUMA(None-Uniform Memory Access) 자원을 이용하면 단일 자원 내에서 성능을 높이기 위한 가장 최적의 방법으로 한정된 공간을 활용하여 안정적으로 서비스할 수 있다. 하지만, 자원 투입 대비 성능과 효율이 낮고 [9], 확장이 좋지 않으며, 비용이 많이 든다. 또한, 3~4년을 주기로 장비 교체가 필수적이다.

클러스터를 이용하는 방식은 독립적인 컴퓨팅 노드들을 고속 네트워크로 묶고 각 노드에서 데이터베이스를 물리적으로 복제한 뒤 질의 처리를 수행한다. 클러스터는 SMP에 비해 저렴하고, 성능과 효율 측면에서 뛰어나지만, 멀티 쓰레드 (Multi-thread) 기반의 BLAST 프로그램을 멀티 노드 (multi-node)에 적합한 형태로 변경해야 하며, 최신 버전의 BLAST 프로그램을 적용하는데 시간이 걸린다. 또한, 제한된 공간 내에 노드의 수를 늘리는데 한계가 있고, 3~4년 주기로 장비를 교체해야 한다. 다수의 노드에 대한 관리 비용이 많이 들어가며, 전력 소모와 발열 때문에 Green IT 관점에서 비효율적이다.

그리드를 이용한 방식은 인터넷상의 많은 자원을 이용하여 클러스터보다 확장이 좋고, 기존 유휴 자원을 활용하는 장점이 있다. 하지만, BLAST의 경우 인터넷상에서 주고받는 데이터의 양이 많고, 노드의 안정성이 떨어지므로 좋은 성능을 기대하기 어렵다. 특히, 그리드 미들웨어를 설치, 변경해야 하는 작업이 많아 성능 대비 관리 비용이 증가한다.

본 논문에서는 클러스터와 그리드 컴퓨팅의 기법의 장점을 결합한 PC 기반의 클러스터를 제시하고, 이 기반에서 BLAST 성능향상을 위해 데이터베이스 분할 기법과 질의 라우팅 기법을 제안한다.

제안한 클러스터 시스템은 대학 내의 PC 자원을 원격 부팅시켜 BLAST 연산에 활용하는 방법으로, 구축 및 관리 비용은 최소화하고 성능과 확장을 극대화한다. 또한, 내부 시스템 구성관점에서는 동적인 노드 환경에서 Logical Partitioning 기반의 Intra-Query를 통해 응답시간을 최소화하고 자원의 효율을 극대화한다. 응답속도, 성능향상, 효율을 통해 본 시스템의 우수성을 검증한다. 본 연구의 공헌은 다음과 같다.

첫째, 대학 내 기존 PC 자원에 오픈 소스 기반의 네트워크 부팅 기술[10]을 적용하여 BLAST 클러스터를 구축하였다. 저녁이나 주말 등의 유휴 시간에 Wake-up 기능을 통해 참여 PC를 부팅시키고 BLAST 클러스터로 활용한다. 본 방식은 인터넷상에 다수의 노드를 활용하는 그리드 방식과 노드를 적극적으로 활용할 수 있는 클러스터 방식의 장점을 결합한 것이다. 또한, 기존의 자원을 적극적으로 활용하여 비용을 줄이고, 관리 효율을 높인다. 또한, 별도의 공간이 필요 없고 전력 활용 측면에서 우수하다.

둘째, 캐쉬를 적극적으로 활용하는 Logical DB

Partitioning 기법을 제안하여 클러스터 성능과 자원 효율을 극대화한다. 하나의 물리적인 DB를 여러 개로 세그먼트로 나누고 참여 PC가 NFS로 접근하여 메모리에 캐쉬한 다음 각각 독립적으로 BLAST 연산을 수행한다. 특정 노드 개수 이후부터는 I/O가 발생하지 않아서 성능과 효율이 급격히 증가한다. 자체 개발한 미들웨어는 노드의 현황을 파악하며 참여 노드 수에 따라 동적으로 캐쉬(Cache)에 DB를 재배치할 수 있다. 논리적 분할 방식을 도입함으로써 데이터베이스 업데이트 부하를 최소화한다.

본 논문의 구성은 다음과 같다. 2장에서는 BLAST 성능을 높이기 위한 컴퓨팅 기법과 전략에 대해 자세히 살펴보고, 3장에서는 제안된 BLAST 시스템의 하드웨어와 소프트웨어적인 구성을 제시한다. 4장에서는 제안된 시스템과 다른 시스템과의 성능을 비교 분석하며, 5장에서는 결론 및 향후 연구를 제시한다.

II. BLAST 성능 향상을 위한 컴퓨팅 기법 및 전략

제2장에서는 BLAST 성능 향상을 위한 컴퓨팅 기법과 클러스터 환경에서 BLAST 구현 방안에 대해 살펴본다.

2.1 BLAST 성능 향상을 위한 컴퓨팅 기법

BLAST 성능 향상을 위한 컴퓨팅 기술로는 그리드 컴퓨팅 기술 (@HOME 방식 포함) [11], 클러스터 컴퓨팅, 고성능 SMP 장비를 이용하는 방식이 있다.

GridBLAST는 글로버스 툴킷 (Globus Toolkit)을 이용하여 인터넷상의 슈퍼컴퓨터, 고성능 클러스터와 같은 분산된 소수의 고성능 자원을 활용하여 BLAST 성능을 높였다 [12]. 그리드 방식이 이론적으로는 가장 이상적이나 현실적으로 네트워크에서 데이터베이스 및 BLAST 프로그램을 전달하는 과정에서 부하가 크고 자원 일부만 소극적으로 사용하므로 성능과 효율 측면에서 한계가 있다. 또한, 가변적인 시스템 환경에서 BLAST 연산을 안정적으로 처리하기 어렵다. 특히, 데스크톱 PC를 활용하는 @HOME 프로젝트[13]는 보안의 취약성, 낮은 안정성, 관리의 어려움, 이질성 때문에 안정적인 성능을 제공하지 못하고 있다.

Hyper-BLAST[14], MPI-BLAST[15,16], TurboBLAST [17] 는 단일 장비의 멀티 쓰레드 기반의 BLAST를 Multi-Node 기반의 클러스터 환경에 맞도록 변경하였다. Hyper-BLAST의 경우 기존 BLAST 명령어에 노드 설정 파일에 대한 옵션을 추가하고 마스터 노드와 작업 노드 간의 프로토콜 및 데이터 구조를 정의하였다 [14]. MPI-BLAST는 클러스터에서 물리적 데이터베이스 분할과 질의 분할을 하였고, 스래싱 현상을 최소화하여 성능과 효율을 높였다[15]. 하지만, 클러스터는 노드가 Tightly-Coupled 방식으로 연결되어 변경이나 확장이 쉽지 않고, 최신 버전의 BLAST 기능을 활

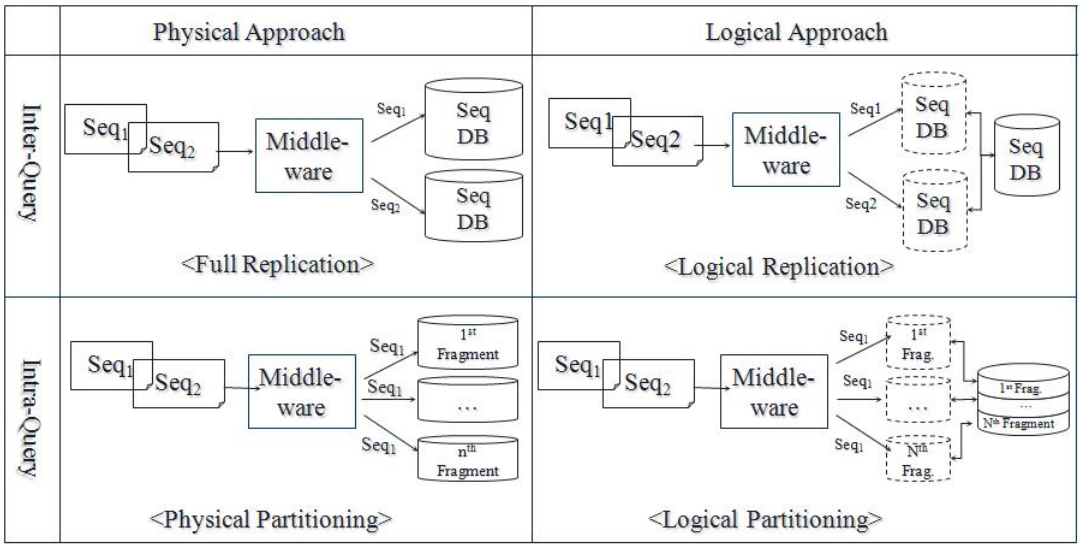


그림 1. BLAST 병렬 처리를 위한 데이터 배치와 질의 전략
Fig.1. Data Design and Query Strategy for Parallel BLAST Processing

용할 수 없다. 또한, 공간상의 한계 때문인 지속적인 확장이 힘들고 전력, 발열 등이 심각히 발생하여 Green IT 관점에서 비효율적이다.

SMP 또는 NUMA 장비를 이용하는 경우 단일 기계 내의 다수 CPU를 이용하는 것으로 BLAST를 소스코드 변경 없이 고성능 자원을 최대한 활용할 수 있다. 또한, 제한된 공간에서 안정적으로 최대의 성능을 제공할 수 있다. 하지만, 비용이 많이 들고, 투입 대비 효과가 낮으며 확장성이 낮다. 특히, SMP 장비는 3~5년 주기로 고가의 장비를 업그레이드 또는 교체해야 안정적인 성능을 보장할 수 있다.

2.2 데이터 디자인과 질의 라우팅

BLAST 성능을 높이기 위해서는 Physical DB Design [18]과 Query Routing 전략이 매우 중요한데 [그림 1]과 같이 크게 4가지로 분류할 수 있다.

첫째, Inter-Query와 물리적 복제 방식(Physical Replication)을 결합한 방식으로 데이터베이스를 여러 노드에 물리적으로 복제하고 나서 PBS, Condor와 같은 스케줄러로 각 질의를 노드에 분배하고 처리하는 방식이다. 이 방식은 구현이 간단하고 다수 질의를 동시에 처리하여 생산성을 극대화할 수 있다. 하지만, 단일 질의에 대한 응답시간을 줄일 수 없고, 대량의 여러 데이터베이스를 동시에 갱신하는 비용이 많이 든다. 또한, 각 데이터베이스 크기가 노드의 메모리보다 크면 스래싱(Thrashing) 현상이 발생하여 성능과 효율이 떨어진다.

둘째, Intra-Query와 물리적 분할 방식(Physical Partitioning)을 결합한 방식으로 각 노드에 분할된 데이터베이스를 저장하고 같은 질의를 수행하고서 그 결과를 병합하는 방식이다 [15]. 이 방식은 각 노드가 처리할 데이터베이스를 메모리에 완전히 캐쉬하여 I/O를 줄일 수 있으므로 성능과 효율을 높일 수 있다[15]. 반면, 분산된 다수 노드에 데이터베이스를 물리적으로 같은 크기로 분할하고 갱신하는 비용이 많이 든다. 또한, 한 노드에 장애가 발생하면 전체 결과의 일관성이 깨진다.

셋째, Inter-Query와 논리적 복제(Logical Replication)를 결합한 것으로 현재 대부분 바이오인포메틱스 관련 연구소에서 주로 활용하는 방식이다. 이 방식은 각 노드가 NFS(Network File System) 서버의 서열 데이터베이스에 접근하여 BLAST 연산을 수행한다. 물리적인 데이터베이스가 한 개 있으므로 업데이트가 편리하고 관리가 쉽다. 하지만, 데이터베이스가 대량인 경우(DB 〉 Cache) 네트워크상에서 대량의 데이터를 주고받는 병목과 각 노드가 전체 데이터베이스를 캐쉬하지 못하면 스래싱 현상이 발생하여 성능이 급격히 저하된다.

넷째, 본 논문에서 제안한 방식으로 Intra-Query와 논리적 분할(Logical Partitioning)을 결합하는 것이다. 분할된 데이터베이스를 각 노드에 완전히 캐쉬하여 BLAST 성능과 효율을 높이고 단일 데이터베이스를 유지하므로 관리가 쉽다. 자세한 내용은 3.2절에서 설명한다.

III. BLAST 성능 향상을 위한 클러스터 인프라와 소프트웨어 전략

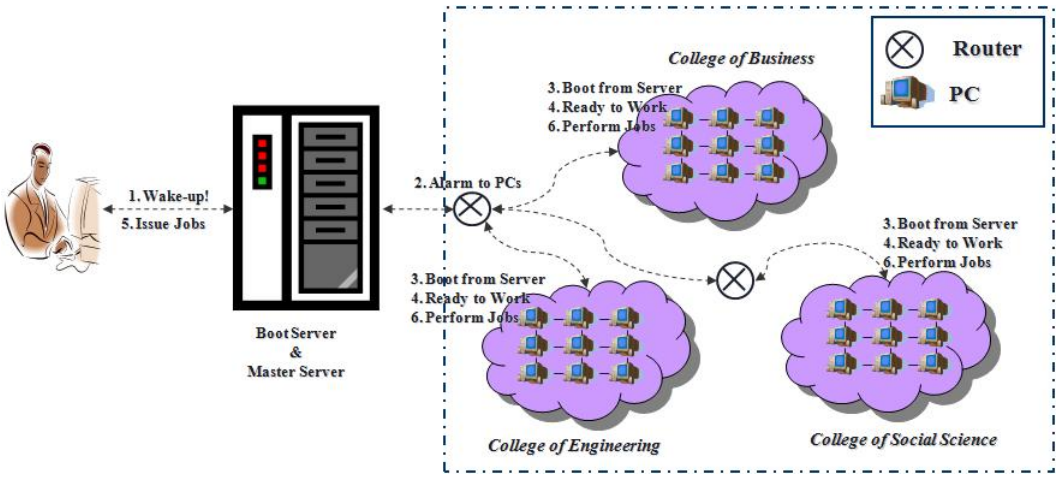


그림 2. E-Cluster 기반 BLAST 컴퓨팅 인프라
Fig.2. BLAST Infrastructure on E-Cluster

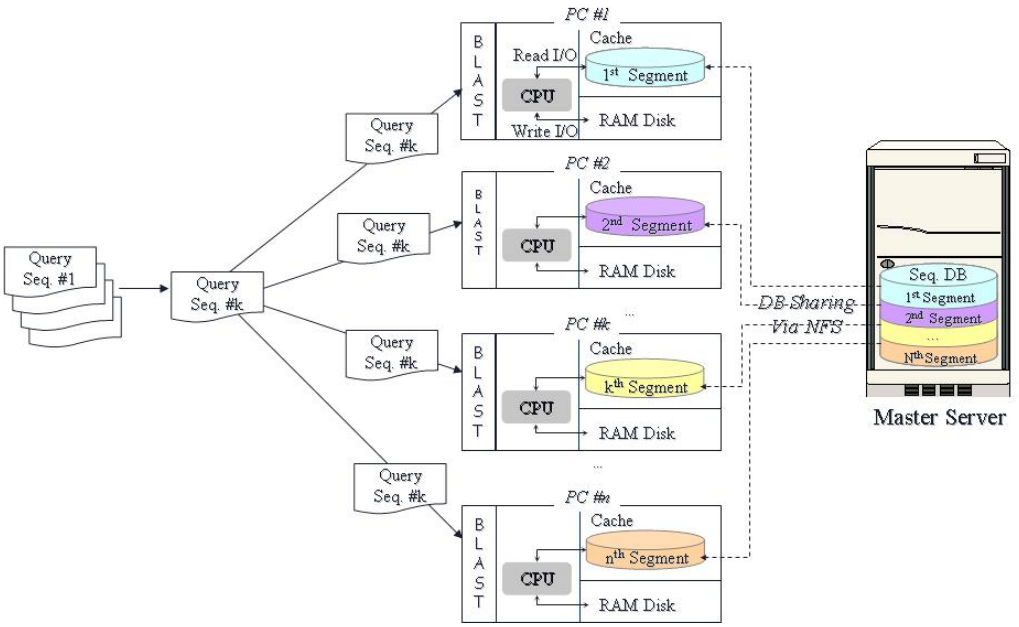


그림 3. E-BLAST 의 구조
Fig.3. Proposed E-BLAST Architecture

제3장에서는 본 연구에서 제안한 클러스터 시스템과 BLAST 시스템 구조에 대해 자세히 살펴본다.

3.1 E-Cluster : BLAST 컴퓨팅 인프라

E-Cluster (Extended Cluster)는 네트워크 부팅 기술 기반으로 인터넷상의 밀집된 유휴 PC 자원을 활용하여 제한된 시간 동안 고성능 컴퓨팅 파워를 제공하는 클러스터이다. 본 연구에서의 컴퓨팅 노드는 주로 대학 내에서 사용 중인 실습용 PC이다. 평소에는 교육용으로 사용하고 저녁 또는 주말 시간을 이용해 BLAST 전용 클러스터 노드로 활용한다. [그림 2]는 BLAST 시스템을 구현하는 E-Cluster의 개념도이다. 현재 대학 내에 두 개의 PC 실습실을 활용하고 있으며 지속적인 확장이 가능하다. 작동하는 순서는 다음과 같다.

관리자는 저녁 또는 주말의 정해진 시간에 웹을 통해 참여 PC들에 원격부팅 명령을 내린다(1,2). 각 PC 들은 부트 서버의 자신의 고유영역을 읽어서 리눅스로 부팅하고(3) 마스터 서버의 명령을 기다린다(4). 사용자가 명령을 내리면(5) 마스터 서버에서 각 노드에 작업을 분배하고 (6) 결과를 수집하여 최종 결과를 전달한다. 이 모든 과정에 각 참여 노드는 기본적으로 로컬 디스크를 사용하지 않으므로 PC의 재설치 환경 변화에 영향을 받지 않는다. 또한, 부팅 이후 운영체제와 처리할 데이터를 메모리에 캐쉬하고, 메모리 일부를 램 디스크(RAMDISK)로 활용하므로 부트 서버와의 통신을 최소화한다. 본 연구에서는 BLAST 프로그램과 분할된 데이터베이스를 메모리에 캐쉬하여 성능을 극대화하였다.

본 연구의 방식은 그리드 컴퓨팅 방식과 클러스터 컴퓨터의 장점을 결합한 방식으로 인터넷상의 다수 PC를 대상으로 클러스터처럼 적극적으로 자원을 활용할 수 있는 장점이 있다. 그뿐만 아니라 구축비용이 저렴하고 제한된 시간만 사용할 수 있다는 단점은 있지만, 단위 시간 내에 집중으로 컴퓨팅 자원을 활용하여 그 효율을 극대화한다. 4장에서 실험을 통해 이 효과를 검증한다.

실제 그리드 컴퓨팅은 비용이나 성능, 확장성 측면에서 가장 이상적인 방식이지만 자원의 이질성, 가변성 그리고 보안의 문제로 실용성이 떨어지는 단점이 있다. 클러스터는 구현 및 활용이 쉽고 적극적인 자원 제어가 쉽지만 비용 및 확장이 유연하지 않다. 하지만, 본 연구에서 제안하는 시스템은 이러한 단점을 극복한다.

3.2 물리적 데이터 설계와 질의 라우팅 기법

E-Cluster에서 사용하는 시스템 구조는 [그림 3]과 같다. 각 참여 노드는 부팅 하면서 원격 부트 서버에서 자신이 처리할 데이터베이스를 가져와서 메모리에 캐쉬 한다. (Logical Partitioning 기법) 각 노드의 메모리는 데이터베이스 캐쉬

표 1. 실험 환경
Table 1. Experiment Setup

구 분	설 명
CPU	Intel Pentium Ⅳ, Intel 3GHz. (double core)
Memory	1024 KB/Node
Network	100Mbps Switch
OS	Redhat Enterprise Linux 3.0, Kernel Version 2.4.18
BLAST	NCBI-BLAST Version 2.2.1 blastp
BLAST DB	NR (929420 Sequences: 291,584,220 total letters)
Query Sequence	Homo sapiens adenylate cyclase 2 (brain) (ADCY2) (total length : 6551 letters)

영역과 처리 결과를 저장하기 위한 램 디스크 영역으로 구분된다. 램 디스크에 처리 결과를 저장하므로 서버와의 I/O를 최소화 한다.

본 방식의 장점은 다음과 같다. 첫째, 각 노드가 직접 데이터를 관리하지 않으므로 전체 데이터베이스 업데이트 비용이 줄어든다. NCBI는 매주 새로운 서열을 제공하고 있으며 제안한 방식은 잦은 갱신주기에 가장 적합한 시스템이다. 둘째, 노드 수에 비례하여 데이터베이스가 분할되어 각 노드에 완전히 캐쉬하여 I/O를 최소화할 수 있다. 데이터베이스가 완전히 캐쉬되는 특정 개수 노드 이상에서 성능이 급격히 향상된다. 물리적 복제와 논리적 복제 방식은 데이터베이스 크기가 노드의 메모리보다 크면 스래싱과 네트워크 트래픽으로 성능이 급격히 저하된다. 셋째, Intra-Query 기반으로 질의 응답시간을 최소화할 수 있다. 데이터베이스 일부만을 검색하고 병합하는 방식으로 빠른 응답시간이 필요한 경우에 적합하다.

Ⅳ. 성능 평가

본 장에서는 제안된 시스템을 평가하기 위해 실험환경, 성능 평가 척도, 그리고 실제 실험 결과를 제시한다.

4.1 실험 환경과 성능 평가 척도

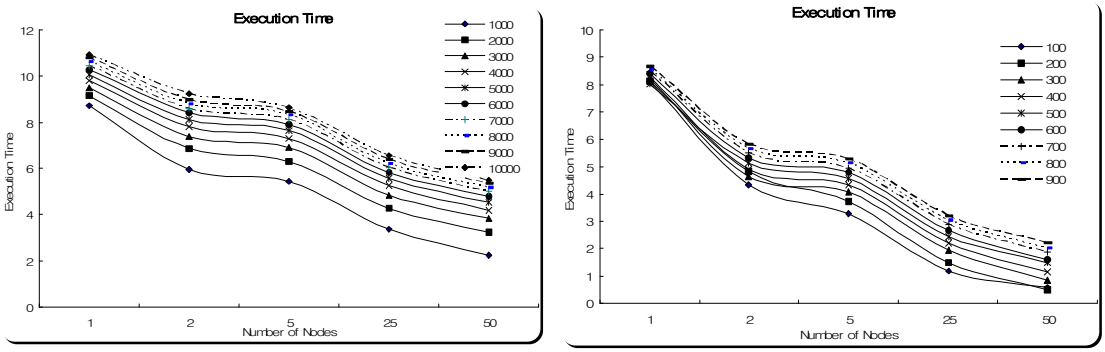
본 실험은 2개의 PC 실습실의 50대의 PC를 대상으로 수행하였다. [표 1]은 본 실험에서 사용한 각 PC의 사양과, 네트워크, 운영체제, BLAST 프로그램의 버전, 데이터베이스 구성을 제시한다.

본 실험의 데이터분할을 위해 *formatdb* 유틸리티를 이용하여 NR 데이터베이스를 50개로 분할하였으며, 질의 서열은 100 Base Pair에서 10000 Base Pair까지 다양 크기의 서열을 이용한다. 성능은 [표 2]와 같이 응답시간(Response Time), 성능향상(Speedup), 효율(Efficiency)로 평가한다.

질의 사이즈에 따른 응답시간의 편차를 줄이기 위해 Log_2를 이용하였다. N개의 노드에서 성능 향상(S_N)을 평가하기 위해 단일 시스템의 처리 시간(t_1)을 다수 노드에서 처리한 시간(t_N)으로 나눈다. N개의 노드에서의 효율(E_N)은 성능 향상

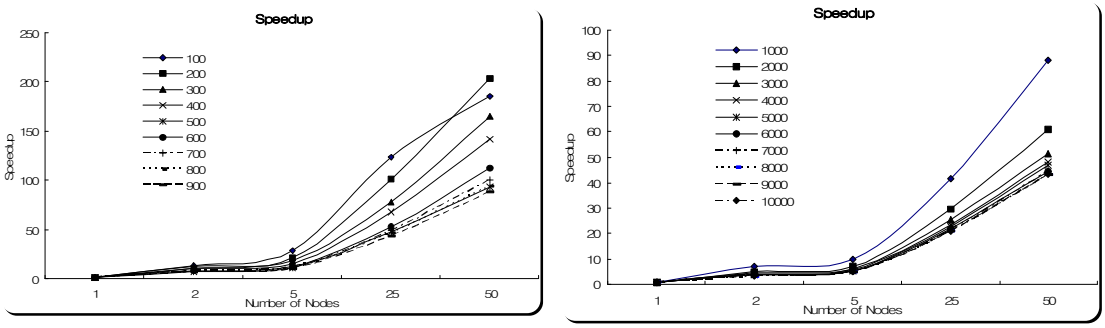
표 2. 성능 평가 척도
Table 2. Performance Evaluation Measures

평가척도	수식	설 명
응답시간 (Response Time)	$T_N = Log_2(t_N)$	T_N = N개의 노드에서 처리 시간
성능향상 (Speedup)	$S_N = \frac{t_1}{t_N}$	S_N = N 개의 노드에서 성능향상 t_1 = 1개 노드에서의 처리 시간 t_N = N개 노드에서의 처리 시간
효율성 (Efficiency)	$E_N = \frac{S_N}{N}$	E_N = N개 노드에서의 효율 S_N = N개 노드에서의 성능향상 N = 노드의 개수



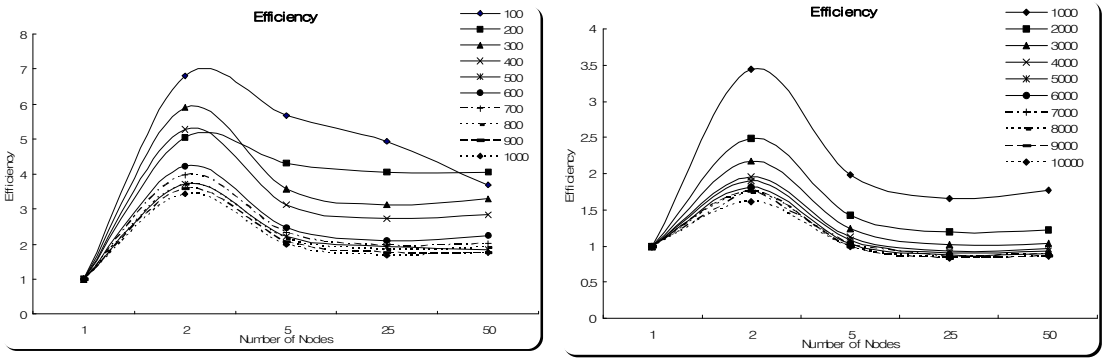
(a) (b)

그림 4. 짧은 서열(a)과 긴 서열(b)에 대한 BLAST 처리 응답 시간
Fig. 4. BLAST Response Time for Short (a) and Long(b) Sequences



(a) (b)

그림 5. BLAST 처리 성능 향상
Fig. 5. BLAST Speedup on E-Cluster



(a) (b)

그림 6. BLAST 처리 효율
Fig. 6. BLAST Efficiency on E-Cluster

을 노드 수로 나누어 계산한다.

4.2 실험 결과

〔그림 4〕는 본 실험에서 처리한 질의 응답시간을 보여준다. 예상한 바와 같이 노드 수가 증가함에 따라 응답시간이 급격히 좋아짐을 확인할 수 있다. 특히, 알고리즘의 특성상 입력 서열의 길이가 짧을수록 처리 시간이 빠르고, 성능 향상 정도가 높음을 확인할 수 있다.

〔그림 5〕는 좀 더 정확한 성능평가를 위해 노드 투입 대비 성능향상 정도를 보여준다. 노드의 수에 비례하여 성능이 급격히 증가함을 확인할 수 있다. 특히, 데이터베이스가 메모리에 모두 캐쉬 되는 2개의 노드에서 가장 급격한 성능향상을 보여준다. 이 지점은 각 노드의 메모리 크기에 따라 달라질 수 있다. 2개의 노드에서 S_N은 100 Base Pair에서 13.6, 10000 Base Pair에서 3.2이다. 〔14〕의 실험에 의하면 SMP 컴퓨터와 클러스터에서의 S_N은 각각 1.8~1.9, 1.9의 성능을 보여준다. 결과적으로 본 연구의 시스템 성능이 기존 시스템보다 1.5~7배 정도 우수함을 확인할 수 있다.

〔그림 6〕은 노드 투입 대비 효율을 제시한다. 2개의 노드에서 최대의 효율을 보여주며 노드가 증가함에 따라 효율이 떨어진다. 하지만, 50개의 노드에서도 최소 100%의 효율을 보여준다. 최대 50개의 노드까지 모든 경우에서 모두 100%를 넘는 효율을 보여준다. 본 실험에서 5개 노드에서의 효율은 서열의 크기에 따라 1~5.7이고, 〔14〕의 실험에 의하면 SMP 컴퓨터에서는 0.2~0.8, 클러스터에서는 0.6~0.7의 효율을 보여준다. 제안된 시스템의 성능평가를 수행한 결과 성능향상, 효율 측면에서 모두 뛰어남을 확인할 수 있다.

V. 결론 및 향후 연구

본 논문에서는 PC 기반 클러스터인 E-Cluster를 이용한 BLAST 시스템을 설계하고 구현하였다. E-Cluster 시스템은 대학 내의 PC를 유휴기간 동안 인터넷으로 연결하여, 본 연구에서는 BLAST 성능향상을 위해 사용하였다. 본 시스템은 Logical Partitioning 기법과 Intra-Query를 이용하여 성능과 효율을 개선하였다. Logical Partitioning 기법은 데이터베이스 여러 노드가 공유하므로 관리 비용이 저렴하고 최신 데이터 업데이트가 쉽다. Intra-Query는 데이터베이스의 일부를 접근하여 완선 캐쉬한 뒤 BLAST 연산을 수행하여 성능을 극대화하였다. 제안된 시스템은 기존 시스템에 비해 저렴한 비용과 우수한 확장성을 지원한다. 실험을 통해 기존 SMP, 클러스터, 그리드 보다 성능과 자원 활용 효율 측면에서도 우수함을 검증하였다. 향후 연구로는 노드 수에 따른 응답시간을 예측할 수 있는 함수 모델을 만들 예정이다. 또한 노드 성능이 다를 경우에 각 노드에서 균등한 응답시간을 얻을 수 있는 방안을 제시할 예정이다.

참고문헌

〔1〕 B. DA, et. al., "GenBank." *Nucleic Acids Res.* this issue. 2009.

〔2〕 GOLD database, http://www.genomesonline.org/

〔3〕 남성혁, 김태경, 김경란, 조완섭, "서비스 지향 구조 기반의 EST 서열 주해 시스템," *한국컴퓨터정보학회논문지*, 제13권, 제3호, 35-44쪽, 2008

〔4〕 J. Ye, et al., "BLAST: improvements for better sequence alignment," *Nucleic Acids Research*,34 6-9, 2006.

〔5〕 S. F. Altschul et al., "Gapped BLAST and PSI-BLAST: A new generation of protein database search programs," *Nucleic Acids Research*, 25:3389-3402, 1997.

〔6〕 S. B. Needleman, C.D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology*, 48 (3): 443-53, 1970

〔7〕 T. F. Smith, M.S. Waterman, "Identification of Common Molecular Subsequences," *Journal of Molecular Biology*147: 195 - 197, 1981

〔8〕 M. K. Gardner; Wu-chun Feng, H.J. Archuleta, "Parallel Genomic Sequence-Searching on an Ad-Hoc Grid: Experiences, Lessons Learned, and Implications," *The International Conference on High-Performance Computing, Networking, and Storage*, 2006

〔9〕 G. Amdahl, "Validity of the Single Processor Approach to Achieving Large-Scale Computing Capabilities," *AFIPS Conference Proceedings*, (30), pp. 483-485, 1967.

〔10〕 Etherboot Project, http://www.etherboot.org

〔11〕 I. Foster, "The Grid: A New Infrastructure for

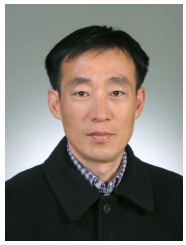
21st Century Science," *Physics Today*, 55(2), pp. 42-47, 2002.
[12] A. Krishnan, "GridBLAST: a Globus-based high-throughput implementation of BLAST in a Grid computing framework: Research Articles," *Concurrency and Computation*: Practice & Experience, Volume 17, Issue 13 Pages: 1607 - 1623, 2005
[13] SETI@HOME Project, http://setiathome.berkeley.edu/
[14] H. S. Kim, H. J. Kim and D. S. Han, "Hyper-BLAST: A Parallelized BLAST on Cluster System," *Lecture Note in Computer Science*, 265:213~222, 2003
[15] A. Darling, L. Carey, and W. Feng, "The Design, Implementation, and Evaluation of mpiBLAST," *International Conference on Linux Clusters*, 2003.
[16] 홍창범, 차정호, 이성훈, 신승우, 박근준, 박근용, "클러스터 환경에서의 MPI 기반 병렬 서열 유사성 검색에 관한 연구," *한국컴퓨터정보학회논문지* 제11권, 6호, 69-78쪽, 2006
[17] R. Bjornson, A. Sherman, S. Weston, N. Willard, and J. Wing. "TurboBLAST(r): A parallel implementation of BLAST built on the TurboHub," *International Parallel and Distributed Processing Symposium*2002.
[18] R. de Carvalho Costa and S. Lifschitz. "Database allocation strategies for parallel BLAST evaluation on clusters," *Distributed and Parallel Databases*, 13(1), 2003.

저 자 소 개

김 태 경

2002년 2월 : 충북대학교 경영정보학과 학사
2005년 2월 : 충북대학교 정보산업공학과 석사
2005년 2월 ~ 현재 : 충북대학교정보산업공학과 박사 과정
관심분야 : 바이오인포메틱스, 그리드 컴퓨팅, XML, 데이터웨어하우스, ERP

조 완 섭



현. 충북대학교 경영정보과 교수
1987년 2월 ~ 1990년 12월 : 전자통신연구원 연구원
2001년 ~ 2002년 : University of Florida, Post-Doc.
1996년 2월 : 한국과학기술원 전산학과 박사
관심분야 : SOA, BPM, DW&OLAP, 데이터마이닝, 바이오 정보시스템, CRM, 전자상거래