ISSN(Print): 2287-5905

http://dx.doi.org/10.3745/KTSDE.2013.2.2.123

Genome Analysis Pipeline I/O Workload Analysis

Kyeongyeol Lim[†] · Dongoh Kim[††] · Hongyeon Kim[†††] · Geehan Park[†]
Minseok Choi[††††] · Youjip Won[†††††]

ABSTRACT

As size of genomic data is increasing rapidly, the needs for high-performance computing system to process and store genomic data is also increasing. In this paper, we captured I/O trace of a system which analyzed 500 million sequence reads data in Genome analysis pipeline for 86 hours. The workload created 630 file with size of 1031.7 Gbyte and deleted 535 file with size of 91.4 GByte. What is interesting in this workload is that 80% of all accesses are from only two files among 654 files in the system. Size of read and write request in the workload was larger than 512 KByte and 1 Mbyte, respectively. Majority of read write operations show random and sequential patterns, respectively. Throughput and bandwidth observed in each processing phase was different from each other.

Keywords : Bioinformatics, Workload Analysis, SSD

유전체 분석 파이프라인의 I/O 워크로드 분석

임 경 열[†] · 김 동 오[††] · 김 홍 연[†††] · 박 기 한[†] · 최 민 석[††††] · 원 유 집[†††††]

요 약

최근 유전체 데이터의 급격한 증가로 인해 이를 처리하기 위한 고성능 컴퓨팅 시스템이 필요로 하게 되었으며 대량의 유전체 데이터를 저장 관리할 수 있는 고성능 저장 시스템이 필요하게 되었다. 본 논문에서는 대략 5억 개 정도의 시퀀스 리드 데이터를 분석하는 유전체 분석 파이프라인의 I/O워크로드를 수집 및 분석하였다. 실험은 86시간 동안 수행되었다. 1031.7 GByte 크기의 630개 파일이 생성되었으며 91.4 GByte 크기의 535개의 파일이 삭제되었다. 전체 654개의 파일 중 0.3%인 2개의 파일이 전체 접근 빈도의 80%를 차지하여 전체 파일 중 일부분의 파일이 대부분의 I/O를 발생시킨다는 것을 알 수 있다. 요청 크기 단위로는 읽기에서 주로 512 KByte 크기 이상의 요청이 발생했고 쓰기에서 주로 1 MByte 크기 이상의 요청이 발생했다. 파일이 열려있는 동안의 접근 패턴은 읽기와 쓰기 연산에서 각각 임의와 순차패턴을 보였다. IOPS와 대역폭은 각 단계마다 고유한 패턴을 보였다.

키워드 : 바이오인포매틱스, 워크로드 분석, SSD

1. 서 론

생명공학 기술 발달로 예측할 수 없을 정도의 빠른 속도로 대량의 바이오 데이터가 획득되고 있다. 획득되는 바이오 데이터의 관리, 분석 및 처리를 위해 페타플롭스(Petaflops)급의 고성능 컴퓨팅 시스템이 필요하게 되었으며 페타바이트(Petabyte) 급으로 급증하고 있는 바이오데이터를 저장 관리 할 수 있는 저장 시스템이 필요하게 되었다.

오래전부터 저장 장치는 컴퓨팅 시스템의 병목지점이 되어왔지만 최근 플래시 메모리를 사용한 SSD가 사용되면서 저장장치 기술이 빠르게 발전하고 있다. SSD가 빠른 접근 속도, 저전력, 무소음, 뛰어난 내구성 등 많은 장점을 갖고 있지만 HDD와 비교했을 때는 아직도 단위 저장 공간 당 가격이 높고 플래시 메모리의 특성상 제자리 쓰기(in-place-update)가 불가능한 점, 블록의 지움 횟수 제한과 같은 단점이 있다[1].

최근 두 장치의 성능의 차이를 극복하고 각 장치의 장점을 활용하여 더 낳은 성능을 제공하기 위한 기술이 개발되고 있고 두 장치의 최적 구성 방안을 찾기 위한 연구가 활발히 진행되고 있다. 본 논문에서는 유전체 분석 파이프라인에 최적의 저장 시스템을 구성하기 위해 워크로드의 I/O 특성을 분석하고자 한다.

※ 본 연구는 지식경제부 및 한국산업기술평가관리원의 IT산업원천기술 개발사업의 일환으로 수행하였음.
† 준 회 원 : 한양대학교 전자컴퓨터통신과 석사과정
†† 정 회 원 : 한국전자통신연구원 선임연구원
††† 정 회 원 : 한국전자통신연구원 책임연구원
†††† 준 회 원 : 한양대학교 전자컴퓨터통신학과 박사
††††† 정 회 원 : 한양대학교 전자컴퓨터통신공학과 부교수
논문접수 : 2013년 1월 8일
수 정 일 : 1차 2013년 1월 17일
심사완료 : 2013년 1월 17일
* Corresponding Author : Youjip Won(yjwon@hanyang.ac.kr)

2 장에서는 유전체 분석에 대한 소개와 유전체 분석 파이프라인의 단계별 특성을 설명하고 사용 어플리케이션에 대해 서술한다. 3 장에서는 본 실험에서 사용한 바이오 워크로드 분석 도구에 대해 간단하게 소개한다. 4 장에서는 본 실험에서 사용한 인간유전체 시퀀스 데이터와 참조인간 유전체 데이터 및 실험을 수행한 테스트 베드 구성을 서술한다. 5 장에서는 유전체 분석 파이프라인의 I/O워크로드를 수집하고 그 결과를 IOPS, 접근 패턴, 파일의 생성과 삭제, 접근 빈도에 대해 분석한다. 마지막으로 6 장에서는 결론 및 본 논문 이후의 향후 진행 방향을 간단히 소개한다.

2. 유전체 분석 파이프라인 소개

본 논문에서 사용하는 유전체 분석 파이프라인[2, 3, 4]은 인간 유전체 데이터로부터 SNP 구조 분석[5, 6] 에 의한 질병 관련 정보를 추출하기 위한 유전체 분석을 수행하는 파이프라인이다. 유전체 분석 파이프라인은 Fig. 1과 같이 크게 3가지 과정으로 나뉜다.

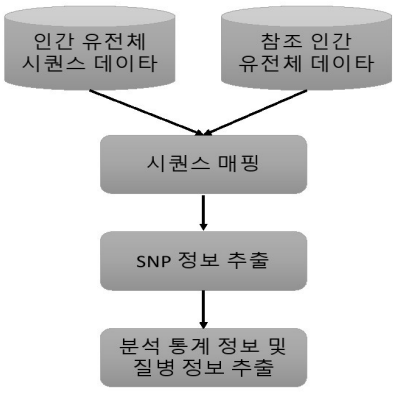


Fig. 1. Genome Analysis Pipeline phase

시퀀스 리드 매핑 단계는 참조 인간 유전체 인덱스에 일종의 검색 단계인 alignment를 수행하는 단계로써 수행시간 단축을 위해 병렬처리로 수행 가능하다.

SNP 정보 추출은 SNP calling 단계로써 시퀀스 매핑의 결과를 SAM(Sequence Alignment/Map)형식[7]으로 변환하는 역할을 담당한다. 이 단계에서 Genomic 위치에 따른 정렬 과정이 포함된다.

마지막으로 분석 통계 정보 및 질병 정보 추출 단계는 밝혀진 SNP에 대해서 synonymous, non-synonymous, insertion, deletion, protein amino acid의 frame-shift 가능성에 대해서 분석 수행한다.

유전체 분석에 사용되는 다양한 어플리케이션이 존재한다. BWA(Burrows-Wheeler Aligner)는 BWT(Burrows-Wheeler Transform)를 사용 하여 짧은 시퀀스 리드뿐만 아니라 인간 유전체와 같은 커다란 참조 시퀀스까지도 효율적으로 정렬한다[8].

SAM 포맷은 정렬된 리드를 저장하기 위해 사용되는 일반적인 포맷이다. samtools는 정렬된 SAM 포맷에 대해서 sorting, merging, indexing등 후처리를 위한 다양한 유틸리티를 제공한다[9].

3. 바이오 워크로드 분석 도구

분산 컴퓨팅 환경 및 고속 I/O환경에서 유전체 분석 애플리케이션의 I/O워크로드를 분석하기 위해 유전체 분석 워크로드에 특화된 분석 도구를 개발하였다. Fig. 2는 바이오 워크로드 분석 도구의 시스템 구성을 보여준다.

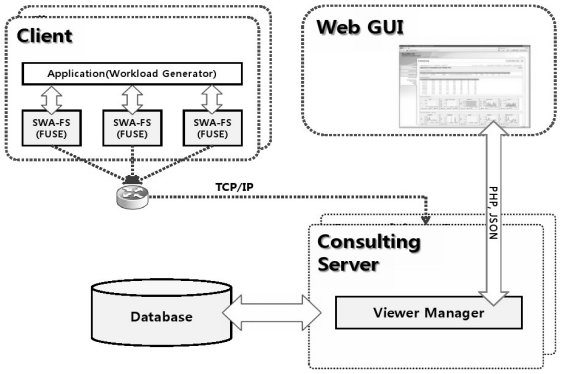


Fig. 2. System Diagram

Fig. 2에서 보듯이 바이오 워크로드 분석 도구는 수집된 워크로드 데이터를 저장, 관리, 분석하기 위한 바이오 워크로드 분석 서버, 바이오 워크로드 응용에서 발생되는 I/O 워크로드를 수집하기 위한 바이오 워크로드 분석 클라이언트, 분석된 결과를 사용자가 이해하기 쉽게 보여주기 위한 바이오 워크로드 분석 웹 GUI 3가지 컴포넌트로 구성된다.

특히, 바이오 워크로드 분석 클라이언트 컴포넌트는 분산 컴퓨팅 환경을 지원하기 위해 각 클라이언트에 에이전트 형태로 존재한다. 바이오 워크로드 분석 클라이언트는 FUSE (Filesystem in Userspace)[10]기반으로 구현되어 있다.

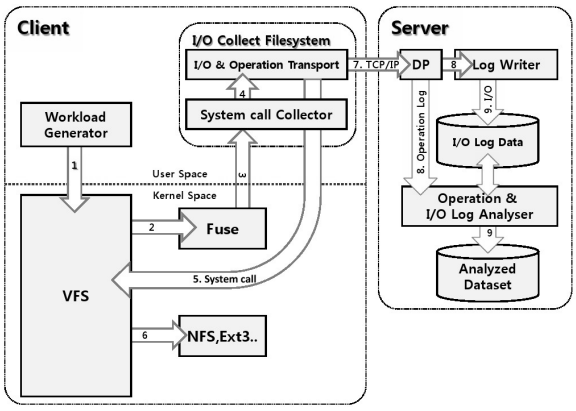


Fig. 3. I/O Capture Flow Chart

Fig. 3은 I/O 정보를 수집하는 흐름도를 보여준다. Fig. 3에서 보듯이 각 클라이언트에서 발생한 I/O 정보를 파일시스템 수준에서 수집하여 TCP/IP를 이용해 바이오 워크로드 분석 서버로 전송한다.

바이오 워크로드 분석 서버에서는 전송 받은 I/O 정보를 취합 후 분석 저장 한다. 분석된 워크로드 정보는 데이터베이스에 저장 되며 웹 GUI를 통해 관리 및 모니터링 한다.

바이오 워크로드 분석 도구는 실시간 모니터링, 워크로드 분석 두 가지 기능을 제공한다.

실시간 모니터링 기능은 대상 시스템의 IOPS, 대역폭, 각 파일시스템 연산의 발생 횟수 등 I/O 벤치마킹 정보를 실시간으로 모니터링 한다.

워크로드 분석기능은 대상 워크로드가 종료된 이후 해당 워크로드에 대해 각 유전체 분석 파이프라인 단계별로 실행 시간, 파일별 접근 패턴, 파일이 열려있는 동안의 접근 패턴, IOPS, 대역폭, 파일의 접근 빈도, 요청 크기별 횟수, CPU 사용율 등의 I/O분석 정보를 제공한다.

바이오 워크로드 분석 웹 GUI 컴포넌트는 분석된 결과를 사용자가 보기 쉽게 보여주며, Fig. 4A는 웹 GUI의 워크로드 분석 화면을 Fig. 4B는 워크로드 분석 화면 중 대역폭와 요청크기 화면의 예를 보여준다.

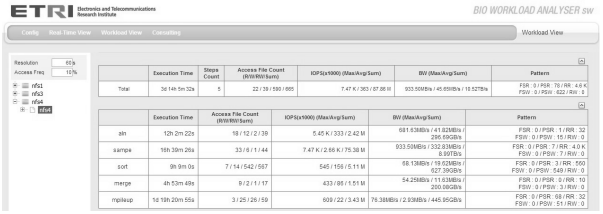


Fig. 4A. Workload Overview

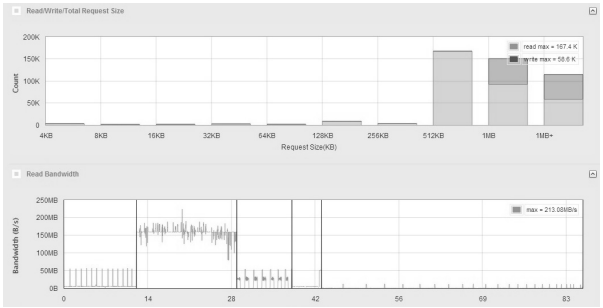


Fig. 4B. BW and Request size View

Fig. 4. Web GUI

4. 워크로드 분석 환경

보통 인간 유전체의 30억 염기쌍에 대해서 10~40배의 coverage로 시퀀싱하게 되는데, 이러한 과정에서 전체 300억에서 1200억 개의 염기쌍이 생성된다. 본 실험에 사용되는 데이터는 인간 유전체의 시퀀스 데이터 중 대략 5억 개 정도의 시퀀스 리드 데이터를 분석하였다. 또한, 인간 유전체 시퀀스 데이터 분석을 위해 인간 참조 유전자 데이터가 사용되었다. 본 실험에서 사용된 File Set의 크기는 Table 1과 같다.

Table 1. File Set

	갯수	크기
Human reference genome	10	11GB
Human whole genome sequencing data	14	218GB

본 논문의 실험 환경은 유전체 분석 파이프라인을 실행할 클라이언트 1대, NFS 스토리지 서버 1대, 분석 서버 1대를 10G 네트워크로 연결하여 Fig. 5와 같은 테스트 베드를 구성하였다. 실험에 필요한 데이터 및 처리 과정에서 생성되는 파일들은 NFS 서버에 저장된다.

Fig. 5. Test Bed Setup

5. 워크로드 분석 결과

유전체 분석 파이프라인의 I/O 워크로드를 분석하기 위해서 본 논문에서는 바이오 워크로드 분석 도구를 사용했다.

실험 환경은 Fig. 5와 같은 환경에서 실험의 정확성을 위해 하나의 클라이언트에서 수행하였다. 유전체 분석 파이프라인 도구는 BWA(Burrows-Wheeler Aligner)와 samtools를 사용하였다.

유전체 분석 파이프라인은 크게 시퀀스 매핑, SNP 정보 추출, 분석 통계 정보 및 질병 정보 추출 3단계로 나뉜다. 본 실험에서는 유전체 분석 파이프라인 도구의 사용에 따라 스퀀스 매핑은 bwa aln단계로 SNP 정보 추출은 bwa sampe와 samtools view | sort & index & flagstat단계 로 나누었다. 마지막으로 분석 통계 정보 및 질병 정보 추출은 samtools merge & index, samtools mpileup 단계로 나누었다. 이와 같이 전체 단계를 총 5단계로 나누어 비교 분석 하였다.

각 단계는 bwa aln, bwa sampe, samtools view, samtools merge, samtools mpileup으로 지칭하도록 하겠다.

5.1 파일의 생성과 삭제

Table 2는 유전체 분석 파이프라인 워크로드를 실험하면서 얻어진 파일의 생성 및 삭제 정보를 요약해 보여주고 있다.

Table 2. File Creation and Deletion

	Create / Delete/ Open	Create	Delete
bwaaln	14 / 0 / 2979	91.0	0
bwasampe	7 / 0 / 1647	320.8	0
samtoolsview	556 / 535 / 12	227.0	91.4
samtoolsmerge	1 / 0 / 10	69.4	0
samtoolsmpileup	52 / 0 / 108	323.4	0
Total	630 / 535 / 4756	1031.7	91.4

Table 2에서 보듯이 전체적으로 총 1031.7 GByte 크기의 630개 파일이 생성됐다. 총 91.4 GByte 크기의 535개의 파일이 삭제되었으며 4756번 열렸다.

각 단계별로 보면 이중 bwa sampe와 samtools mpileup 단계에서 각각 320.8 GByte와 323.4 GByte로 가장 많은 크기의 파일이 생성되었다. bwa aln단계에서는 91 GByte의 파일을 생성하며 2079번 열려 작은 크기의 I/O가 빈번하게 발생했음을 알 수 있다.

5.2 접근 빈도

Fig. 6에서는 파일의 접근 빈도를 랭크 순으로 정렬하여 상위 2%를 보여주고 있다. 본 논문에서는 접근 빈도를 열린 횟수(Fig. 6A), I/O 발생 횟수(Fig. 6B, 6C), I/O 요청 크기(Fig. 6D, 6E)로 나누어 정의하였다.

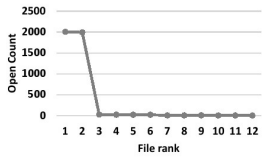


Fig. 6A. Open

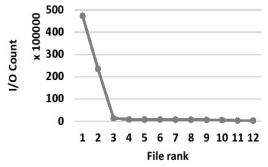


Fig. 6B. Read Count

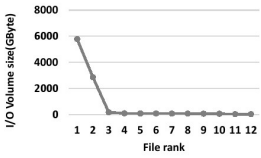


Fig. 6C. Read Volume size

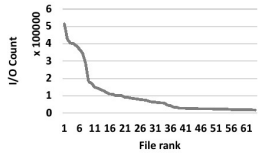


Fig. 6D. Write Count

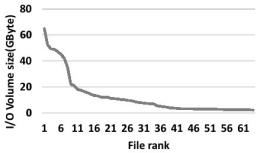


Fig. 6E. Write Volume size

Fig. 6. Access Frequency

읽기 연산(Fig. 6B, 6D)에서는 상위 랭크 5%의 파일(31개의 파일)이 전체 읽기 요청의 99%를 차지하고 있다.

상위 랭크 0.3%의 파일(2개의 파일)이 읽기 요청의 빈도의 80%를 차지하고 있어 전체 파일 중 일부분의 파일이 대부분의 읽기 요청을 발생시킨다는 것을 알 수 있다.

상위 랭크 2개의 파일을 고성능의 SSD 스토리지에 저장한다면 스토리지의 성능 향상 및 소모전력 감소에 영향을 줄 것으로 예상된다.

쓰기 연산의 경우(Fig. 6C, 6E) 전체 파일의 1.3%인 9개의 파일이 쓰기 요청의 빈도를 75%를 차지해 읽기와 마찬가지로 전체 파일 중 일부분의 파일이 대부분의 쓰기 요청을 발생시킨다는 것을 알 수 있다. 쓰기 요청이 많이 발생한 상위 5%의 파일들 중 15개의 파일은 읽기 연산에서도 상위 5%의 파일에 속해 있다.

5.3 접근 패턴

Fig. 7은 유전체 분석 파이프라인 단계별 순차적으로 발생한 전체 I/O 요청 크기들의 횟수를 나타내고 있다. x축은 하나 또는 다수의 I/O요청들이 순차적으로 발생한 경우 해당 요청들의 크기를 합한 값이다. y축은 I/O 요청 크기의 발생 횟수를 나타낸다.

bwa aln단계와 bwa sampe단계 에서는 1 MByte 이상의 크기로 발생한 요청이 각각 전체 중 96%와 83%로 대부분의 요청이 1 MByte 이상의 크기로 발생했음을 알 수 있다. samtools view단계에서 쓰기의 경우 1 MByte 이상의 크기로 발생한 요청이 전체 중 97%로 전체 중 대부분을 차지하지만 읽기의 경우 4 KByte, 128 KByte, 1 MByte 이상에서 유사한 수치를 보이고 있다. samtools merge단계에서 쓰기의 경우 대부분 1 MByte 이상의 크기로 요청이 발생 했다. 읽기의 경우 대부분의 요청이 512 KByte와 1 MByte 이상의 크기로 순차적인 요청이 발생 했다. samtools mpileup 단계에서 쓰기의 경우 대부분 1 MByte의 요청이 발생하였고 읽기의 경우 대부분 512 KByte의 요청이 발생 했다.

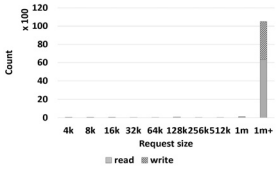


Fig. 7A. bwa aln

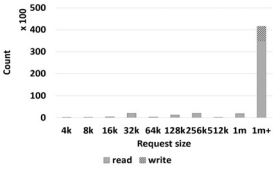


Fig. 7B. bwa sampe

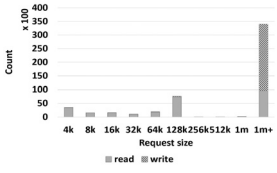


Fig. 7C. samtools view

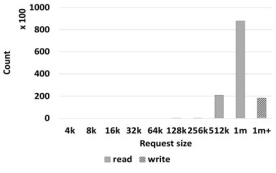


Fig. 7D. samtools merge

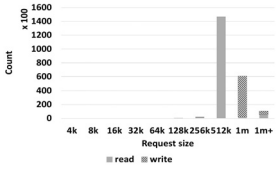


Fig. 7E. samtools mpileup

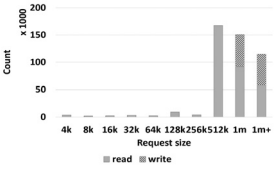


Fig. 7F. Total

Fig. 7. Request Size Count

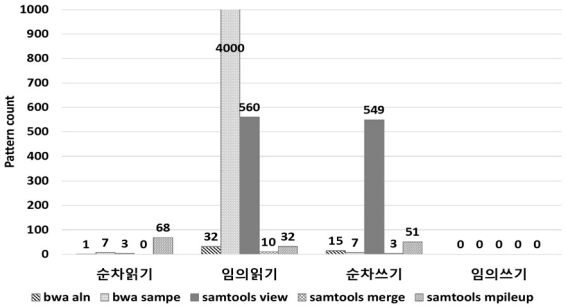


Fig. 8. Access Pattern During File Open

Table 3. File Access Pattern

	Read	Write	R/W
bwa aln	18	12	2
bwa sampe	33	6	1
samtools view	7	14	542
samtools merge	9	2	1
samtools mpileup	3	25	26
Total	22	39	590

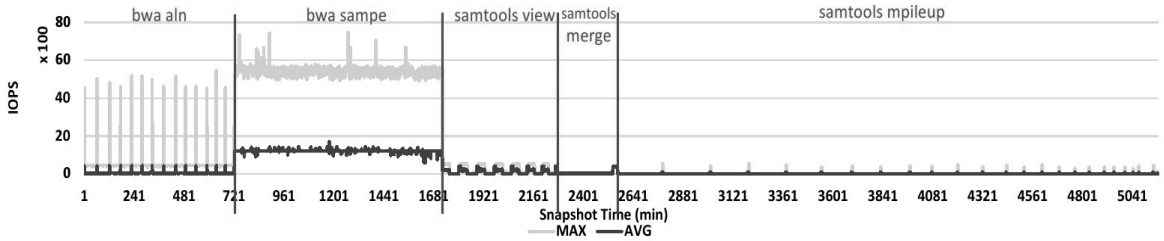


Fig. 9A. Read IOPS

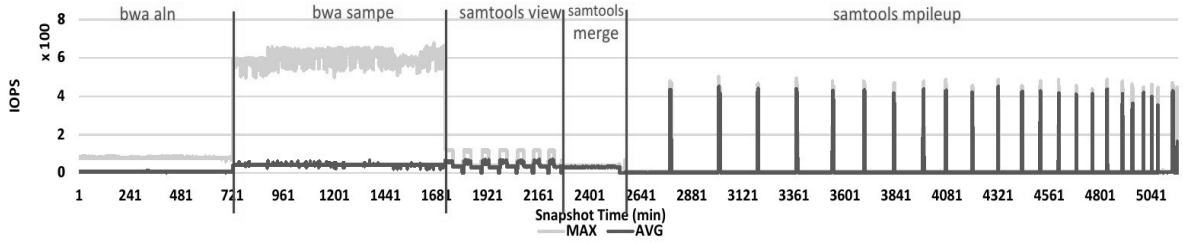


Fig. 9B. Write IOPS

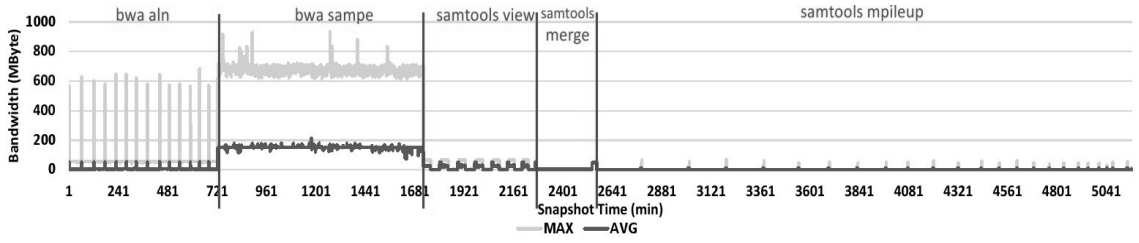


Fig. 9C. Read Bandwidth

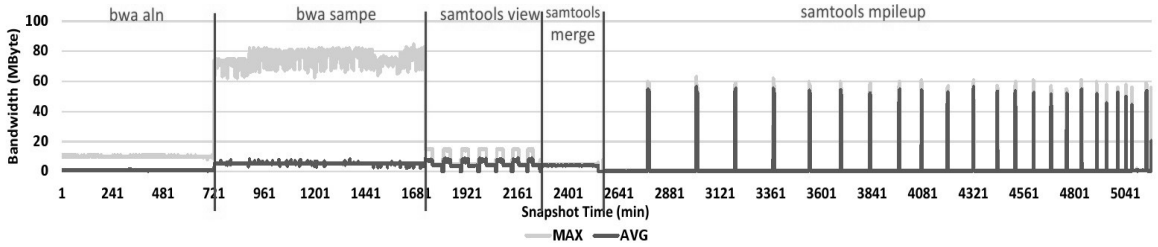


Fig. 9D. Write Bandwidth

Fig. 9. IOPS And Bandwidth

마지막으로 전체 단계에 걸쳐 512 KByte 이상의 요청이 전체 중 94%로 대부분의 요청이 512 KByte 이상의 크기로 발생했다는 것을 알 수 있다.

Fig. 8은 파일이 open되고 close될 때까지의 I/O 접근 패

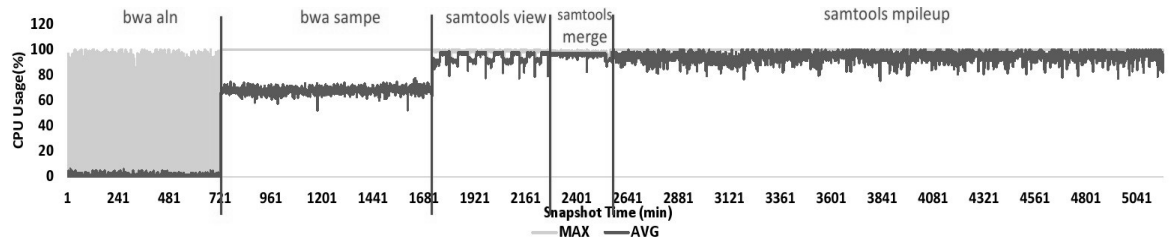


Fig. 10A. user

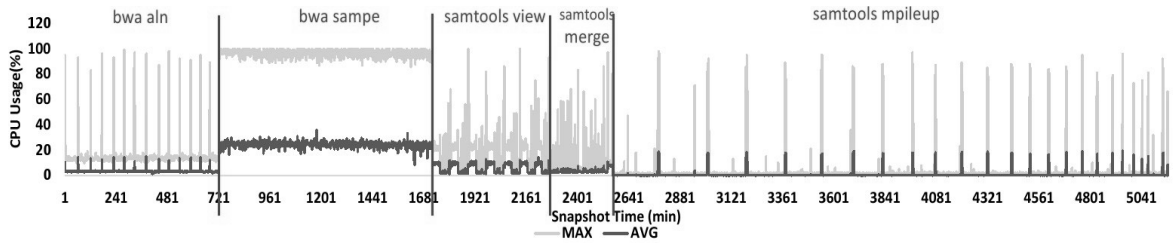


Fig. 10B. system

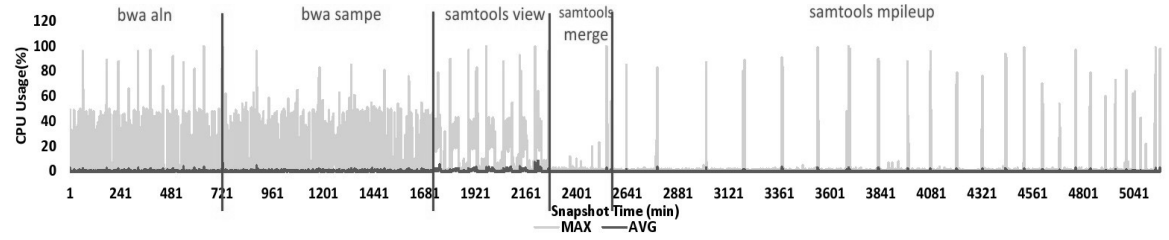


Fig. 10C. iowait

Fig. 10. CPU 사용율

턴을 순차 읽기 쓰기, 임의 읽기 쓰기로 구분했을 경우 읽기는 주로 임의패턴을 보였고 쓰기는 주로 순차패턴을 보였다(Fig. 8).

Table3 에서는 각 파일단위의 접근 패턴을 분석하기 위해 해당 파일에 읽기만 발생한 경우와 쓰기만 발생한 경우, 읽기 쓰기 모두 발생한 경우로 나누었다.

bwa aln단계에서는 읽기와 쓰기 패턴을 보이는 파일이 균등하게 나타났다. bwa sampe단계 에서는 전체 파일 중 82%가 읽기만 한 파일로 대부분 파일이 읽기만 한 것으로 나타났다. samtools view단계에서는 전체 파일 중 92%가 읽기 쓰기가 동시에 잃어난 파일이었다. samtools merge단계에서는 읽기만 한 파일이 9개, 쓰기만 한 파일이 2개, 읽고 쓴 파일이 1개 존재했다. samtools mpileup 단계에서는 쓰기만 한 파일과 일기 쓰기가 동시에 일어난 파일이 균등하게 나타났다. 마지막으로 전체 워크로드에 걸쳐 패턴을 판단했을 때는 읽고 쓰기가 모두 발생한 파일이 많은 것으로 나타났다.

5.4 IOPS

Fig. 9A, 9B에서는 유전체 분석 파이프라인 워크로드 전체 읽기 쓰기 IOPS의 1분간의 최대값과 평균값을 그래프로 보여주고 있다. 각각 단계마다 고유한 패턴을 보임을 알 수 있다.

bwa aln 단계 에서는 읽기 집중적인 패턴을 보인다. 주기적으로 IOPS가 값이 치솟는 모습을 보이는데 모두 전체 읽기 요청 크기 중 상위랭크 2개 파일에서 발생한다. bwa sampe 단계에서는 평균값과 최대값이 차이가 크게 나는 모습을 보여 bwa sampe 단계의 전 구간에 걸쳐 짧은 시간동안 많은 양의 데이터를 지속해서 읽고 쓰는 패턴을 보였다. samtools view 단계에서는 지속해서 읽고 쓰는 패턴이 주기적으로 나타난다. samtools merge 단계에서는 대부분 지속해서 쓰는 패턴을 보이며 단계의 마지막구간에서 짧은 시간동안 집중적인 읽기 패턴을 보인다.

마지막으로 samtools mplieup 단계에서는 읽기 연산이 매우 적은 반면 쓰기 연산에서 주기적으로 피크 되는 모습을 보여 쓰기 집중적인 패턴을 보임을 알 수 있었다. 대부분의쓰기 연산이 bwa sampe 단계와 samtools mplieup 단계에서 발생함을 알 수 있었다.

Fig. 9C, 9D에서는 전체 워크로드의 읽기 쓰기 대역폭을 1분간의 최대값과 평균값으로 보여주고 있다. Fig. 9A, 9B와 아주 유사한 모습을 보이는 것을 알 수 있다. 이는 FUSE를 사용하기 때문이다. FUSE는 실제 요청 크기를 일정한 크기로 잘라서 처리하기 때문에 각 IOPS들과 각 대역폭들은 동일한 비율을 가지게 된다.

Fig. 10에서는 전체 워크로드에 대한 user, system, iowait CPU 사용율을 1분 동안의 최대, 평균값으로 보여주고 있다. Fig. 10A에서는 Fig. 9B와 비교해 보면 samtools mpileup 단계에서 가장 많은 CPU 작업을 처리 했고 주기적으로 파일에 쓰는 모습을 보인다. Fig. 10B는 Fig. 9A와 Fig. 9B의 읽기와 쓰기 추세 중 각각 높은 수치와 매우 유사한 패턴을 보인다.

6. 결론 및 향후 연구

본 논문은 유전체 분석 파이프라인을 위한 효율적인 저장 시스템을 구축하고 I/O성능을 극대화하기 위해 유전체 분석 파이프라인의 I/O특성을 바이오 워크로드 분석 소프트웨어를 사용해 분석하였다. 그 결과 유전체 분석 파이프라인의 각 단계별로 고유한 I/O특성을 가지고 있는 것으로 나타났다.

86시간 동안 총 1031.7 GByte 크기의 630개 파일이 생성되었으며 총 91.4 GByte 크기의 535개 파일이 삭제되었다.

전체 654개 파일 중에 0.3%인 2개의 파일이 전체 80%의 접근 빈도를 보임을 알 수 있었다. 2개의 파일은 모두 Human refernce genome File Set에 속해 있다.

본 워크로드에서 접근 패턴을 요청 크기단위로 보면 읽기와 쓰기 각각의 경우 대부분 1 MByte와 512 KByte 이상의 크기로 읽기 쓰기를 요청한다. 워크로드의 전체적인 읽기와 쓰기 패턴을 분석하면 읽기의 경우 대부분 임의패턴을 보이고 쓰기의 경우 순차패턴을 보인다.

IOPS에서는 각각 단계마다 고유한 패턴을 보임을 알 수 있다. bwa aln단계 에서는 읽기 집중적인 패턴을 보이며 주기적으로 IOPS가 피크 되는 모습을 보인다.

bwa sampe 단계에서는 전 구간에 걸쳐 짧은 시간동안 많은 양의 데이터를 지속해서 읽고 쓰는 패턴을 보였다. samtools view 단계에서는 지속해서 읽고 쓰는 패턴이 주기적으로 나타난다. samtools merge 단계에서는 대부분 지속해서 쓰는 패턴을 보이며 단계의 마지막구간에서 짧은 시간동안 집중적인 읽기 패턴을 보인다. samtools mplieup 단계에서는 집중적인 패턴을 보인다.

향후에는 본 연구에 이어 바이오 워크로드 분석 SW에서 도출된 정보를 바탕으로 최적화된 저장 시스템을 구성해 줄 수 있는 컨설팅 알고리즘 및 시스템을 개발하고자 한다.

참 고 문 헌

[1] J. Kang, H. Jo, J. Kim, and J. Lee, “A superblock-based flash translation layer for nand flash memory,” pp.161-170, 2006.

[2] C. Bell, R. Dixon, A. Farmer, R. Flo-res, J. Inman, R. Gonzales, M. Harri-son, N. Paiva, A. Scott, J. Weller, et al., “The medicago genome initiative: a model legume database,” Nucleic Acids Research, Vol.29, No.1, pp.114-117, 2001.

[3] L. Matukumalli, J. Grefenstette, D. Hyten, I. Choi, P. Cregan, and C. Van Tassell, “Snp-phage - high throughput snp discov-ery pipeline,” BMC bioinformatics, Vol.7, No.1, pp.468, 2006.

[4] Seon-Hee Park, “IT based Bioinformatics,” kiise, Vol.21, No.6, pp.20-26, 2003.

[5] Ik-Young Choi, “A review of the technology of genome & expression analysis,” TiBMB, Vol.30, No.2, pp.25-35, 2010.

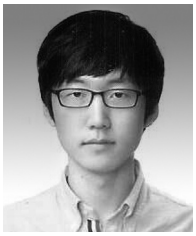
[6] E. Lander, L. Linton, B. Birren, C. Nus-baum, M. Zody, J. Baldwin, K. Devon, K. Dewar, M. Doyle, W. FitzHugh, et al., “Initial sequencing and analysis of the hu-man genome,” Nature, Vol.409, No.6822, pp.860-921, 2001.

[7] A. McKenna, M. Hanna, E. Banks, A. Sivachenko, K. Cibulskis, A. Kernytsky, K. Garimella, D. Altshuler, S. Gabriel, M. Daly, et al., “The genome analysis toolkit: a mapreduce framework for an-alyzing next-generation dna sequencing data,” Genome research, Vol.20, No.9, pp.1297-1303, 2010.

[8] H. Li and R. Durbin, “Fast and accu-rate short read alignment with burrows - wheeler transform,” Bioinformatics, Vol.25, No.14, pp.1754-1760, 2009.

[9] H. Li, B. Handsaker, A. Wysoker, T. Fen-nell, J. Ruan, N. Homer, G. Marth, G. Abecasis, R. Durbin, et al., “The se-quence alignment/map format and sam-tools,” Bio-informatics, Vol.25, No.16, pp.2078-2079, 2009.

[10] FUSE, “Filesystem in userspace.” http://fuse.sourceforge.net/.



임 경 열

e-mail : lkyeol@hanyang.ac.kr

2011년 백석대학교 정보통신학부(학사)

2011년～현 재 한양대학교 전자컴퓨터통신과 석사과정

관심분야 : Operating System, File system

김 동 오

e-mail : dokim@etri.re.kr

2000년 건국대학교 컴퓨터공학(학사)

2002년 건국대학교 컴퓨터·정보통신공학(석사)

2006년 건국대학교 컴퓨터·정보통신공학(박사)

2006년～2009년 건국대학교 강의교수

2009년～현 재 한국전자통신연구원 선임연구원

관심분야 : Database, Parallel File System, Spatial data management

김 홍 연

e-mail : kimhy@etri.re.kr

1992년 인하대학교 통계학과(학사)

1994년 인하대학교 전자계산학과(석사)

1999년 인하대학교 전자계산학과(박사)

1999년～현 재 한국전자통신연구원 책임연구원

관심분야 : Storage System, File system, Database System

박 기 한

e-mail : pghsky@hanyang.ac.kr
2011년 백석대학교 정보통신학부(학사)
2011년~현 재 한양대학교
전자컴퓨터통신과 석사과정
관심분야 : Embedded Software Systems

최 민 석

e-mail : 0310cms@hanyang.ac.kr
2003년 방송통신대학교(독학사)
컴퓨터과학과(학사)
2009년 한양대학교 전자컴퓨터통신학과
(석사)
2009년~현 재 매크로임팩트 SI본부/SCF
2011년 방송통신대학교 정보통계학과(학사)
2012년~현 재 한양대학교 전자컴퓨터통신학과 박사
관심분야 : Operating System, Distributed File System, Storage System

원 유 집

e-mail : yjwon@hanyang.ac.kr
1990년 서울대학교 계산통계학과(학사)
1992년 서울대학교 계산통계학과(석사)
1997년 University of Minnesota(박사)
1997년~1999년 Intel 연구원
1999년~현 재 한양대학교
전자컴퓨터통신공학과 부교수
관심분야 : Operating System, Storage System, High Speed Internet, Wireless Communication, Internet Model