KIPS Trans. Comp. and Comm. Sys.
Vol.8, No.10 pp.231~238 pISSN: 2287-5891

https://doi.org/10.3745/KTCCS.2019.8.10.231

Parallelization of Genome Sequence Data Pre-Processing on Big Data and HPC Framework

Eun-Kyu Byun† · Jae-Hyuck Kwak† · Jihyeob Mun††

ABSTRACT

Analyzing next-generation genome sequencing data in a conventional way using single server may take several tens of hours depending on the data size. However, in order to cope with emergency situations where the results need to be known within a few hours, it is required to improve the performance of a single genome analysis. In this paper, we propose a parallelized method for pre-processing genome sequence data which can reduce the analysis time by utilizing the big data technology and the high-performance computing cluster which is connected to the high-speed network and shares the parallel file system. For the reliability of analytical data, we have chosen a strategy to parallelize the existing analytical tools and algorithms to the new environment. Parallelized processing, data distribution, and parallel merging techniques have been developed and performance improvements have been confirmed through experiments.

Keywords : Genome Sequence Data Preprocessing, NGS, Big Data, Hadoop, HPC, Parallelization

빅데이터 및 고성능컴퓨팅 프레임워크를 활용한 유전체 데이터 전처리 과정의 병렬화

변 은 규† · 곽 재 혁† · 문 지 협††

요 약

차세대 염기 서열 분석법이 생성한 유전체 원시 데이터를 기존의 방식대로 하나의 서버에서 분석하기 위해서는 데이터 크기에 따라 수십 시간이 필요할 수 있다. 그러나 응급 환자의 진단처럼 수 시간 내에 결과를 알아야 하는 상황이 존재하기 때문에 단일 유전체 분석의 성능을 향상시킬 필요가 있다. 본 연구에서는 빅데이터 기술의 병렬화 기법과 고속의 네트워크로 연결되고 병렬파일시스템을 공유하는 고성능컴퓨팅 클러스터를 적극적으로 활용하여 분석 시간을 크게 단축시킬 수 있는 유전체 데이터 분석의 전처리 프로세스의 병렬화 방법을 제안한다. 분석 데이터의 신뢰성을 위해 기존의 검증된 분석 도구 및 알고리즘을 새로운 환경에 맞게 병렬화 하는 전략을 선택하였다. 프로세스의 병렬화, 데이터의 분배 및 병렬 병합 기법을 개발하였고 실험을 통해 성능 향상을 확인하였다.

키워드 : 유전체 데이터 전처리, NGS, 빅데이터, Hadoop, 고성능컴퓨팅, 병렬화

1. 서 론

차세대 염기서열 유전체 분석법(NGS, Next Generation Sequencing)의 발전으로 인해 유전체 정보를 보다 저렴한 가격과 적은 시간을 들여 읽어 낼 수 있게 되었다[1]. 이렇게 얻은 유전체 데이터를 분석하여 질병의 진단, 예방 등에 활용할 수 있다. 이러한 분석 과정은 유전체 정보를 기계를 통해 읽어내는 작업 뿐 아니라 수백 기가바이트에 달하는 데이터를 분석하는 과정이 필요한데, 이러한 분석 기존의 단일 서버를 사용하는 방식으로 수행하면 수십 시간이 소요되는 경우도 일반적이다. 대규모의 컴퓨팅 클러스터를 소유한 경우, 이러한 분석을 동시에 여러 개를 수행하는 방식으로 효율을 높여왔다. 이러한 접근 방법은 일상적인 검사나 연구의 경우에는 크게 문제가 되지 않지만 응급 환자에게 적합한 처치를 해야 하는 상황처럼 빠른 분석이 필요한 경우에는 치명적일 수 있

※ 이 논문은 대한민국 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(No. 2015-0-00590, 빅데이터 처리 고도화 핵심기술개발 사업 총괄 및 고성능 컴퓨팅 기술을 활용한 성능 가속화 기술 개발)과 한국과학기술정보연구원 주요사업의 지원을 받아 수행된 연구임.
※ 이 논문은 2018년도 한국정보처리학회 추계학술발표대회에서 '고성능 클러스터와 분산 병렬 파일 시스템을 이용한 유전체 데이터 전처리 작업의 효율적인 병렬화 기법'의 제목으로 발표된 논문을 확장한 것임.
† 정 회 원 : 한국과학기술정보연구원 선임연구원
†† 비 회 원 : 한국과학기술정보연구원 선임연구원
Manuscript Received : December 10, 2018
First Revision : August 22, 2019
Accepted : September 9, 2019
* Corresponding Author : Eun-Kyu Byun(ekbyun@kisti.re.kr)

다. 따라서 더 많은 전산 자원을 활용하더라도 단일 분석의 시간을 단축 할 수 있는 기술이 필요하다. 본 논문에서는 많은 유전체 데이터 분석에서 공통적으로 쓰이는 데이터 전처리 프로세스를 병렬화하는 방법을 제안한다. 병렬화된 분석 프로세스는 고속의 네트워크로 연결되고 병렬 파일 시스템을 공유 하는 복수의 서버들 위에서 동작하여 실행시간을 크게 단축시킬 수 있다.

NGS기법을 통한 유전자 분석 과정은 여러 단계의 데이터 처리 프로세스로 구성된 파이프라인으로 이루어진다. 먼저 세포로부터 염기서열을 읽어야 한다. 각 염색체 혹은 RNA염기서열 전체를 한꺼번에 직접 읽어내는 것이 불가능하다. 따라서 시퀀서(Sequencer)라는 장비를 이용하여 유전체를 작은 다수의 리드(Read)로 조각내고 이를 증폭하여 화학 및 광학적인 방법을 사용하여 염기쌍(base pair)들을 순서대로 읽어내어 원시 데이터 파일을 생성한다. 분석을 위해서는 조각나기 전의 원래 염기서열 정보를 리드 데이터들로부터 재조합해야 한다. 이러한 과정을 데이터 전처리 과정이라 하며 대부분의 유전정보 분석 기법에서 공통적으로 필요로 한다. 전처리 결과의 기술 방법 또한 SAM, BAM이라는 파일 형식으로 표준화되어 있다[2].

전처리 과정의 첫 단계로 각 리드들이 실제로 유전자중 어느 위치에 해당하는지를 알아야 한다. 이를 위해 참고 유전체(reference genome)의 염기서열에 리드들의 데이터를 비교하여 높은 확률로 일치하는 위치를 찾는 과정을 거친다. 이 과정을 지역 정렬(alignment)이라 하며 많은 계산 시간과 메모리를 필요로 하지만 전처리 중 필수적인 단계이다.

다음으로 중복 검출 단계가 필요할 수 있다. 시퀀서에서 NGS데이터를 생성하는 과정 중 중합 효소 연쇄반응 단계에서 같은 데이터가 여러 차례 발생할 수 있다. 이러한 데이터가 분석에 왜곡을 가져올 수 있으므로 중복으로 판단되는 리드를 표시해 둘 필요가 있다. 데이터를 변경하는 것이 아니라 플래그를 추가하는 것이므로 분석 응용의 필요에 따라 활용 여부를 결정할 수 있다.

마지막으로 분석 도구가 원하는 위치의 유전체 정보에 빠르게 접근 할 수 있도록 색인화 하는 작업이 필요하다. 이를 위해서 각 리드를 참고 게놈 내에서의 위치를 나타내는 전역 좌표를 기준으로 정렬해야 한다. 그 이후에 문자열 형식으로 되어 있는 파일을 바이너리 형식으로 변경하고 압축하는 과정을 마친 후 색인 정보를 생성해야 한다.

본 연구에서는 가장 널리 쓰이는 유전체 원시 데이터인 ILLUMINA사의 시퀀서에서 생성하는 paired-end 리드 데이터를 대상으로 삼았다[3]. 이 원시 데이터를 지역 정렬, 중복 검출, 색인하는 전처리 과정 각각을 병렬화 하고 그에 필요한 데이터 전달 기법을 개발하였다. 데이터의 신뢰도를 높이기 위해 프로세스의 병렬화, 데이터의 분배 및 병합 기법은 추가로 개발한 반면 각 단위 프로세스에서의 데이터 처리는 기존의 검증된 도구들이 수행하도록 하였다. BWA (Burrow-Wheeler Aligner)[4], samblaster[5], samtools[6]가 각각의 단계에서 사용되었다. 작업 및 데이터 병렬 처리의 실제 구현에는 서버 간의 소켓 통신, 병렬 파일시스템인 LustreFS를 이용한 MPI-IO 및 Hadoop on Luster 기술 등을 활용하였다.

본 연구의 차별화 되는 가치는 데이터 전처리 과정 전체를 병렬화하여 더 많은 자원을 투입하면 단일 유전 정보 세트에 대해 더 빠른 수행이 가능한 방법을 제안했다는 점이다. 이는 멀티쓰레드 병렬화를 활용하였으나 단일 노드의 한계로 인해 성능 향상에 제한이 있었던 기존의 방식과 하둡 등을 이용하여 병렬화를 하였으나 파일 전처리의 병목 문제들로 인해 충분한 성능 향상을 얻지 못한 기존의 연구들의 문제점을 해결한 것이다. 또한 공통으로 쓰이는 전처리 과정에 집중하여 재사용이 가능하게 함으로써 특정 분석에서만 적용 가능한 병렬화를 제공했던 기존의 방법과 달리 모든 유전체 기반 분석에 활용 가능하다는 차이점이 있다.

논문의 나머지 부분은 다음과 같이 구성된다. 2장에서 관련연구를 간단히 소개하고, 3장에서 유전체 데이터 전처리 과정 병렬화 기법을 각 단계별로 상세히 소개한다. 추가로 전체 병렬 프로세스관리와 데이터 재분배 방식과 관련하여 실제 실험에 사용한 구현 방식과 대안들의 장단점에 대한 의견을 제시한다. 4장에서 테스트베드에서 수행한 성능 평가 결과를 제공한다.

2. 관련 연구

2009년 Burrow-Wheeler Aligner (BWA)가 개발된 이후 유전체 서열 데이터의 지역 정렬 알고리즘의 사실상 표준으로 사용되고 있다. 그러나 BWA는 상당히 많은 연산을 필요로 하는 알고리즘으로 멀티 코어 CPU에서 병렬 실행을 하더라도 전체 유전체 데이터를 정렬하는 데는 몇 시간 또는 며칠이 걸리는 경우도 발생한다. 유전체 분석 파이프라인을 활용하여 진단이나 치료를 돕기 위해서는 가능한 한 빨리 분석해야 하고 따라서 분산 처리 프레임 워크에서 구현되어야 가속화가 가능하다. 이러한 필요성에 의해 몇 가지 병렬 알고리즘[7-9]이 제안되었다. 그 중 하나는 BWA를 Hadoop 프레임 워크에서 병렬 실행시키기 위해 개발된 BigBWA이다[10]. Hadoop을 이용하여 입력 파일을 분할하여 다수의 Mapper가 병렬로 데이터를 처리한다. JNI를 이용하여 기존의 BWA응용을 수정 없이 호출하여 수행한다. 그러나 Hadoop의 스토리지 시스템인 HDFS는 데이터 전송, 데이터 변환 및 병합에 추가 시간을 발생시켜 예상보다 좋지 않은 성능을 보인다. 이는 기존의 응용 프로그램을 Hadoop 환경으로 이식 할 때 일반적으로 발생하는 문제이다.

또 하나의 NGS 데이터 분석의 병렬화 사례로는 유전체 분석 중 하나인 variant calling의 파이프라인 단계 전체를 병렬

화한 Halvade가 있다[11]. GATK Best Practice에서 제안한 알고리즘 세트를 빅데이터 프로그래밍 모델인 MapReduce를 이용하여 구현하였다. 이러한 여러 노드를 이용한 병렬화를 통해 싱글 노드만을 활용한 multi-thread 방식을 뛰어넘는 성능 향상도 이루었다. 그 외에도 HDFS 대신 Amazon S3와 Lustre를 HDFS 대신 활용하는 것도 가능하다. 그러나 Halvade는 variant calling이라는 특정한 유전체 분석 응용에만 사용하기 위해 설계되어서 다른 분석에 사용하기 위해서는 전처리 과정부터 다시 수행해야 한다는 한계점이 있다.

본 연구에서는 이러한 문제들을 해결하기 위해 분석에서 공통으로 쓰이는 NGS데이터의 전처리 과정 전체를 고성능 클러스터와 병렬파일시스템을 활용하여 병렬화하였다.

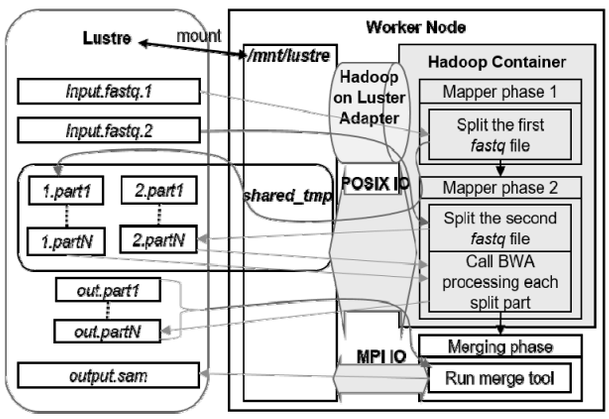


Fig. 1. Architecture and Execution Steps of KBigBWA

3. 유전체 데이터 전처리 과정 병렬화 기법

3.1 지역 정렬 단계 병렬화

지역 정렬 단계를 효율적으로 병렬화하기 위해 Hadoop을 이용하여 BWA를 병렬화한 기존의 BigBWA의 성능상의 문제점을 Hadoop on Lustre와 병렬 I/O 기술을 도입하여 개선한 KBigBWA를 개발하였다.

BigBWA의 성능 저하의 큰 부분은 HDFS를 사용하기 때문에 발생한다. Hadoop 클러스터에서 병렬로 정렬을 수행하려면 유전체 원시 파일을 HDFS에 복사해야 한다. 또한 BWA에서 생성 된 파일은 분석 파이프라인의 다음 단계에서 활용하기 위해 POSIX 호환 파일 시스템으로 다시 복사해야 한다. 본 연구에서는 Lustre on Hadoop을 도입하여 문제를 해결 하였다[13]. Hadoop 응용 프로그램은 HDFS에서 파일을 복사하거나 복사하지 않고 Lustre의 파일에 자유롭게 액세스 할 수 있다. HDFS의 특징인 데이터 지역성(data-locality)의 장점을 포기하지만 고성능컴퓨팅 자원을 활용하여 훨씬 큰 성능상의 이득을 얻을 수 있다.

이를 확인하기 위해 먼저 BigBWA를 변경하지 않고 단순히 HDFS를 Lustre로 교체 한 후에 성능 개선을 측정해 보았다. 실험을 위해 10개의 계산 노드로 이루어진 Hadoop 클러스터와 24개의 HDD로 구성된 Lustre 시스템을 구축하고 이를 Infiniband FDR과 연결하였다. 각 계산 노드에는 Seagate의 Hadoop on Lustre 어댑터와 연동한 Hadoop 2.8을 설치하였고 HDFS를 구성하는 로컬 디스크로는 SATA SSD를 사용하였다. Fig. 2에서 가장 왼쪽의 두 개의 막대는 HDFS와 Lustre를 사용하는 BigBWA를 각각 사용하여 30GB의 유전체 원시 데이터를 지역 정렬하는데 소요되는 시간을 보여준다. HDFS로의 복사에 소요되는 시간이 없어지기 때문에 Lustre에서 Hadoop을 사용하면 HDFS보다 30%가량 빠른 것을 확인할 수 있다.

Lustre에 Hadoop을 적용하는 것만으로도 실행 시간이 의미 있게 줄어들지만 BigBWA의 불완전한 병렬화로 인한 I/O 오버 헤드가 여전히 남아 있다. 우리는 BigBWA를 다시 설계하여 Lustre 특성을 적극적으로 활용하여 모든 I/O를 병렬화하였고 그 전체 수행 과정이 Fig. 1에 요약되어 있다.

기존의 BigBWA에서 가장 많은 시간을 소비하는 부분은 한 쌍의 fastq 파일을 Hadoop 호환 입력 파일로 병합하는 과정으로 단일 노드에서 수행되는 Python 스크립트로 구현되어 있다. 한 세트의 입력만 받아들일 수 있는 Hadoop Mapper들이 이 병합된 입력 파일을 여러 노드에서 분할하여 읽은 후 각 로컬 디스크의 임시 디렉토리에 저장하고 BWA를 호출하여 입력 데이터로 전달하여 지역 정렬을 수행한다.

KBigBWA에서는 이러한 시간소모적인 데이터 변환 단계를 추가적인 Mapper 단계로 대체하였다. 두 개의 Mapper 단계는 두 개의 fastq 파일을 각각 병렬로 분리한다. 분리된 데이터를 이용하여 BWA를 수행하기 위해서 각 BWA 인스턴스는 동일한 범위의 분할된 파일 쌍에 액세스 할 수 있어야 한다. 이를 위해서 두 Mapper 단계에서 사용되는 병렬 프로세스의 개수는 동일해야 한다. 또한 분할된 데이터를 로컬 디스크의 Hadoop temp디렉토리 대신 모든 노드가 공유하는 Lustre에서 마운트 된 shared_tmp 디렉토리에 저장하도록 하였다. 따라서 BWA 프로세스 모두가 shared_tmp에 있는 필요한 범위의 데이터 조각을 읽어서 처리할 수 있다. 또한 BWA의 출력 파일은 Lustre에 직접 기록된다.

각각의 BWA가 생성한 결과 파일을 분석 파이프라인의 다음 단계에서 사용하기 위해서는 하나의 sam 파일로 병합해야 한다. BigBWA에서 제공하는 Hadoop reduce 또는 FullSam python 스크립트를 사용하는 두 가지 방법 모두 비 병렬 방식을 사용하기 때문에 시간 소모가 크다. KBigBWA에서는 여러 노드가 동시에 Lustre에 대한 병렬 쓰기에 참여하는 MPIIO를 활용한 병합 도구를 개발하였다.

Fig. 2는 KBigBWA의 성능을 BigBWA와 비교한 결과를 보여준다. 입력 데이터 분할 및 출력 데이터 병합의 병렬화를 통해 실행 시간이 1/3 이하로 단축하였고 이는 HDFS를 사용하는 BigBWA과 비교해서는 1/5에 불과하다. 또한 단일 노드에서 수행한 BWA과 비교해도 시간을 크게 단축시켰음을 확인하였다.

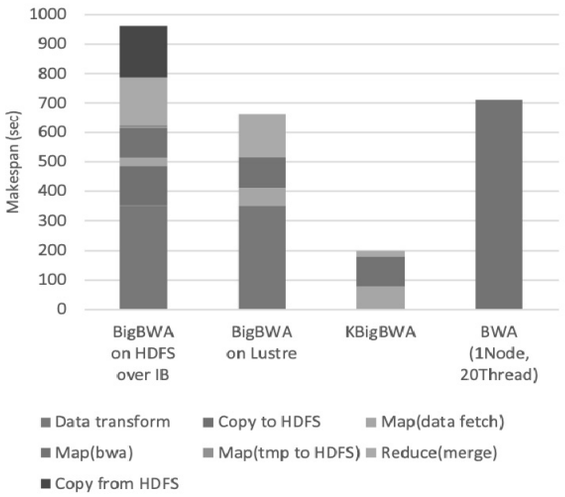


Fig. 2. Performance Improvement of KBigBWA with Hadoop on Lustre and Parallel I/O Functionality

3.2 중복 검출 단계 병렬화

지역 정렬 단계가 완료되면 SAM 형식의 파일이 생성된다. Fig. 3의 예제와 같이 SAM파일에는 참조 유전체의 염색체 이름과 길이 정보가 헤더에 나타나고 각 리드의 정렬(alignment) 정보가 매 줄 마다 기록된다. 이 중 RNAME 값과 POS값을 이용하면 해당 리드가 참조 유전체 중 어느 염색체의 어느 위치에 정렬되어 있는지 알 수 있다. 헤더에 적힌 참조 염색체의 순서와 길이 값을 누적하면 각 참조 염색체의 오프셋을 계산할 수 있고 이 값과 POS값을 더해서 정렬된 위치의 전역 좌표를 결정할 수 있다. 중복 검출과 색인 단계에는 이 전역 좌표 값을 사용한다.

Fig. 3에서 QNAME 값은 시퀀서에서 읽은 각 조각의ID를 나타낸다. Paired-end read의 경우 데이터는 각 유전자 조각을 양쪽에서 일정한 크기만큼 읽어내어 생성한 데이터이므로 동일한 QNAME에서 리드 정렬 정보가 최소한 2개 존재한다. 또한 참조 유전체에 정렬되는 위치가 하나 이상인 경우도 있을 수 있다. 이 경우 가장 확률이 높은 정렬 위치의 쌍을 Primary line이라고 한다. 이렇게 같은 QNAME을 가지는 여러 개의 리드 정렬 정보의 집합(이후 Read Group으로 표현

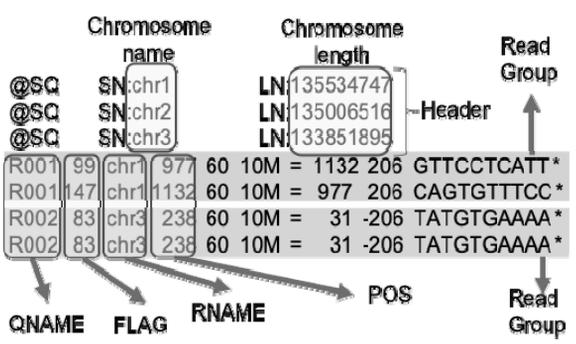


Fig. 3. An Example of SAM File

함)은 동일한 염기서열 조각에서 생성된 데이터들로 중복 검출시 함께 처리되어야 한다.

Samblaster는 SAM파일을 입력으로 받는데 이 때 같은 Read Group에 속한 리드들은 반드시 연속된 라인으로 함께 입력되어야 한다. Samblaster알고리즘은 Read Group의 primary line의 전역 좌표 값을 비교하여 중복을 판단한다. 따라서 중복 검출의 누락을 방지하기 위해 이 조건을 고려하여 병렬화를 위해 데이터를 재분배 기법을 설계하였다. 각각의 노드에서 병렬로 실행된 BWA는 SAM형식으로 결과를 출력하는데 Read Group이 연속되어 기록된다. 이 결과물을 읽어서 Read Group별로 구분하고 primary line의 전역 좌표 2개 중 작은 값을 Key값으로 read group의 정렬 정보 전체를 value값으로 결정한다. 그 Key값을 기준으로 데이터를 분배(partitioning)하여 samblaster를 수행하는 노드들에 전송한다. 이 과정을 통해 각각의 samblaster 프로세스에는 특정한 범위에 정렬된 유전자 조각들이 모이게 된다. 입력된 SAM형식의 데이터에서 중복으로 판단된 리드의 FLAG값만을 수정한 내용이 결과로 출력된다.

Primary line의 광역 좌표를 기준으로 파티션을 결정하였기 때문에, non-primary 리드들의 좌표가 파티션의 범위에 해당하지 않는 경우가 발생할 수 있다. Samblaster 수행 후 이러한 리드들에 대해 다시 한 번 광역 좌표를 기준으로 맞는 위치의 파티션에 전송하고, 현재 파티션과 일치하는 리드들은 로컬 노드 내에서 다음 단계의 입력으로 직접 전달될 수 있다. 몇 가지 데이터 셋을 통해 측정해 본 결과 이렇게 재분배 되는 비율은 1% 전후로 적은 양이어서 소요시간이 매우 짧았다. 설령 대부분이 데이터가 재분배되는 경우에도 samblaster병렬화를 위해 수행했던 재분배 프로세스에서 소요되는 시간과 동일한 시간 복잡도를 가지므로 전체 수행 시간에 큰 영향을 주지 않을 것으로 예상된다.

3.3 정렬, 압축 및 병합 단계 병렬화

다음으로 모든 리드 정보를 전역 좌표를 기준으로 오름차순 정렬(sort)을 해야 한다. 수억 개의 리드 정보를 한꺼번에 정렬하는 것은 엄청난 시간을 필요로 한다. 그러나 이미 samblaster 병렬화 단계에서 전역 좌표를 기준으로 데이터가 분배되어 있다. 이렇게 분배되어 있는 데이터 집합들 각각을 병렬로 정렬(sort)하고 난 후, 각각의 데이터 집합들을 순서에 맞게 연결시키기만 하면 전체 정렬(sort)이 완성된다.

Samtools의 sort 기능을 활용하면 sort, encoding, compress가 한꺼번에 수행되고 결과물로 BAM 파일이 생성된다. BAM 파일 형식은 ASCII 텍스트 형식인 SAM파일의 용량을 줄이는 목적으로 정의 되었다. binary encoding을 수행한 후 gzip알고리즘의 호환인 BGZF 포맷으로 압축을 진행한다. 이 때 데이터 전체를 한 번에 압축하지 않고 64KB보다 작은 크기의 블록으로 데이터를 나눈 후 각각을 압축하고 메타데이터에 블록의 크기, 전역 위치 정보들을 포함하여 향후 인덱싱

에 활용한다. gzip의 특성 상 독립적으로 압축된 파일인 각각의 블록을 단순히 연결해도 gzip파일이 되고 압축 해제 시 연결한 순서대로 복원되는 점을 이용하였다. SAM파일의 헤더 내용을 담은 헤더 블록과 EOF 블록이 앞뒤에 추가된다.

이렇게 각각의 samtools프로세스에서 생성된 BAM파일을 하나의 전체 BAM 파일로 병렬로 연결하는 기법을 개발하였다. 앞서 언급한 gzip 파일의 특성에 따라 각 파일 내에서 광역 좌표를 기준으로 정렬되어 저장된 BAM파일들을 순서에 맞게 단순히 연결하는 것만으로 전체 정렬된 파일을 얻을 수 있다. BAM 파일을 정상적으로 합치기 위해서는 연결부위의 헤더와 EOF블록을 잘라내야 한다. 이렇게 손질된 파일의 연결은 MPI-IO를 이용하여 N-to-1병렬 쓰기 작업을 통해 병합을 진행한. 따라서 최종 파일의 크기가 크더라도 여러 노드가 동시에 쓰기를 수행하기 때문에 소요 시간이 크지 않다.

마지막으로 합쳐진 BAM파일에 대하여 samtools의 index 기능을 이용하여 색인 파일을 생성한다. 이 과정은 100GB 데이터의 경우 수 분 정도 소요되는 작업으로 전체 수행시간 대비 비중이 크지 않고 samtools의 수정 없이 O(1)병렬화가 어렵기 때문에 본 연구에서는 따로 병렬화하지 않았다.

3.4 병렬 프로세스 관리 및 데이터 재배치 방법

Fig. 4에 본 논문에서 제안한 NGS 데이터의 전처리 병렬화 기법의 전체 수행 과정이 정리되어 있다. 파란 화살표는 데이터의 전달 과정을 나타내며, 회색 화살표는 MPI-IO를 이용한 병렬 쓰기를 나타낸다. 노란 박스로 표시된 부분이 병렬화 한 기존 도구들이며, 붉은 색으로 표시한 부분이 새롭게 개발한 기능들이다. 시퀀서에서 생성한 원시 데이터가 fastq 파일의 쌍으로 주어지면 KBigBWA가 Hadoop on Lustre를 이용해 지역 정렬 데이터를 각각의 노드에 생성한다. 이 데이터는 Read Group별로 primary line의 광역 좌표를 기준으로 재분배되고 각각의 노드에서 samblaster를 이용해 중복 리드를 검출한다. 이후 각 리드의 광역 좌표를 기준으로 위치 조정을 다시 수행한 후 각각의 노드에서 samtools sort를 이용해 정렬 압축된 BAM파일을 생성한다. 이들을 병렬 병합 기능을 이용해 하나의 BAM파일로 합치고 index파일까지 생성하면 NGA데이터의 전처리 과정이 완료된다. 결과물로 지역 정렬데이터 SAM파일 하나 지역 정렬, 중복 검출, 정렬 및 압축이 모두 완료된 BAM파일 하나와 그 index 파일 하나가 생성되고 이는 향후의 유전체 분석에 활용될 수 있다.

각각의 단계를 실제로 수행하기 위해서는 병렬 프로세스를 생성하고 데이터를 전달하는 관리자가 필요하다. KBigBWA에서는 Hadoop의 병렬 자원 및 프로세스 관리 기능을 사용하였다. Hadoop을 사용하면 자동으로 클러스터에 프로세스를 생성하고 병렬로 데이터 분석 작업을 시작 할 수 있다. 단, Java로 구현된 Hadoop API와 호환되도록 기존의 분석 도구를 수정할 필요가 있다. BigBWA가 이러한 기능을 이미 구현해 놓은 상황이었기 때문에 어렵지 않게 적용이 가능하였다.

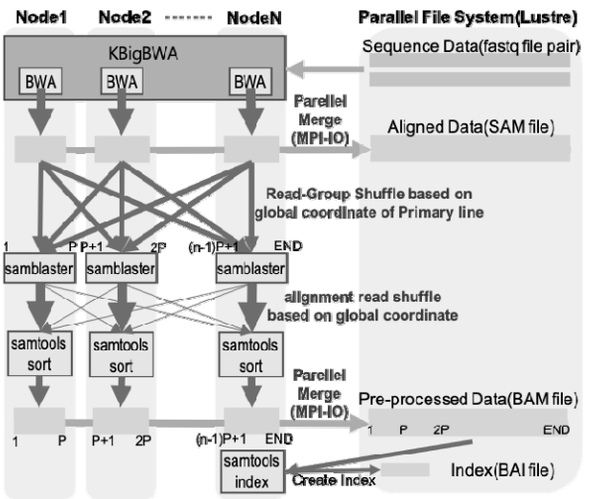


Fig. 4. Whole Process of NGS Data Preprocessing

Hadoop을 사용하지 않는 방법으로는 Spark을 사용하는 방법이 있다. 이 경우에도 분석 도구의 수정이 필요하다. Hadoop과 Spark과 같은 도구를 사용하는 방법은 병렬 작업의 관리 및 장애 복구 등에 장점이 있는 반면 실행 파일의 수정, 프레임워크와 API 단계로 인해 발생하는 성능 오버헤드 등의 단점이 있다. 이러한 단점을 가장 해결하기 위해 가장 가벼운 방법으로 작업을 병렬 수행시키는 방법은 GNU parallel을 사용하는 것이다. GNU parallel은 SSH로 연결된 클러스터에 병렬 작업을 손쉽게 실행할 수 있도록 해 준다. 간단히 Hadoop, Spark, GNU parallel을 사용하여 BWA의 병렬 수행의 시간을 비교해 본 결과 GNU parallel을 사용하는 것이 5% 이상 수행시간이 빠르다는 것을 확인하였다. 따라서 본 논문의 범위에서는 samblaster와 samtools를 이용하는 단계에서는 GNU parallel을 이용한 병렬 프로세스 수행 방식으로 구현을 하고 성능 분석을 진행하였다. 그러나 앞서 언급한 것처럼 빅데이터 자원 및 프로세스 관리 프레임워크의 장점들을 활용하는 것이 전체 소프트웨어의 안정성에 도움이 되기 때문에 전체 병렬화된 전처리 과정을 Spark으로 구현하는 작업을 함께 진행 중이다.

또 다른 중요한 고려사항은 데이터 재분배 과정의 효율적인 설계이다. Hadoop 프레임워크에서는 이 과정을 shuffle이라고 하고 기본 기능으로 포함되어 있다. 위에서 언급한 바와 같이 본 논문에서 구현한 방식에서는 GNU parallel을 사용하였기 때문에 데이터 전송 및 수집을 위한 가벼운 도구를 몇 가지 방식으로 개발하였다.

첫 번째는 OS에서 제공하는 파이프와 소켓을 사용하는 방식이다. 지역정렬 단계에서 중복 검출로 데이터를 전달하는 과정의 경우 BWA의 결과로 나오는 SAM포맷의 결과물을 파이프로 전달 받는 프로세스들이 실행되고 이들로부터 데이터를 TCP소켓 통신을 통해 전달받아 파이프를 통해 각 samblaster의 입력으로 전달하는 프로세스들이 실행되게 된다. N개의

BWA 인스턴스와 M개의 samblaster 인스턴스가 실행될 때 N+M개의 전달 프로세스와 N*M개의 연결이 필요하다. 이러한 방식의 장점은 데이터 전달 과정이 매우 간단하고 추가적인 저장 공간을 사용하지 않으며, 전 단계와 다음 단계가 동시에 수행되면서 파이프라이닝을 통해 자원 사용 효율을 극대화 시킨다는 점이다. 그러나 대규모 클러스터에서 아주 많은 인스턴스를 이용하여 병렬화를 하는 경우 소켓의 크기가 관리하기 어려울 정도로 많아져서 오히려 OS 및 메모리 부하로 인해 성능이 떨어질 우려가 있다.

이를 해결하기 위한 방법으로 ZeroMQ나 Kafka와 같은 메세징 소프트웨어를 사용하는 방법이 있다. 각 소프트웨어가 제공하는 프로토콜에 맞도록 데이터를 가공해야 하고 메세징 프레임워크를 설정해야 하는 단점이 있지만 보다 안정적인 데이터 전송이 가능하다는 장점이 있다.

또 다른 해결 방식은 공유 스토리지에 중간 생성 파일을 저장하여 단계 사이의 데이터를 전달하는 방식이다. 이 경우 본 논문에서 가정한 고성능컴퓨팅 환경의 병렬파일시스템을 적극적으로 활용하여 확장성이나 안정성이 확보된다는 장점이 있으나 임시 저장 공간이 필요하고, 전단계의 병렬 작업 끝난 이후에 다음 단계의 작업을 시작해야 하는 경우가 생길 수 있어서, 프로세스 사이에 직접 통신을 하는 방식에 비해 시간이 조금 더 걸릴 수 있다.

세 가지 방식을 8대 규모의 클러스터에서 100GB 크기의 데이터를 이용하여 테스트해 본 결과 파이프와 소켓을 사용하는 방식이 가장 시간이 적게 걸리는 것으로 나타났다. 다만, 이는 클러스터의 크기가 크지 않았기 때문에 단점이 드러나지 않았기 때문일 가능성이 있다. 그리고 공유 스토리지를 사용하는 방식과 시간 차이도 10% 이내로 크지 않았다. Kafka를 사용하였을 때에는 설정 값에 따라 성능과 안정성이 변동이 있었다.

마지막으로 고려해 볼 수 있는 방식은 Spark을 사용하는 것이다. 데이터 처리 프로세스가 병렬화 되면서 RDD 사이에 자동으로 데이터 재분배가 이루어진다. 특히 Spark-stream을 사용하면 간 단계가 파이프라이닝 되어 수행되므로 더 빠른 수행이 가능해 질 수 있다.

4. 성 능

본 연구의 목적은 복수의 노드로 이루어진 클러스터를 활용하여 수행시간을 단축시키는 방법을 제안하는 것이다. 따라서 실험을 통해 더 많은 노드로 사용하면 수행시간을 더 많이 단축시킬 수 있음을 확인하고자 하였다. 3장에서 설명한 전처리 프로세스를 실행 가능한 프로그램으로 구현하고 이를 병렬 클러스터 위에서 실제 유전체 원시 데이터를 이용하여 전처리를 수행하는 과정의 전체 및 각 단계의 시간을 측정하였다. 클러스터는 FDR Infiniband로 연결된 여덟 대의 계산 노드와 2대의 스토리지 노드로 구성된 Lustre파일시스템으로 구성하였다. 각 계산 노드에는 듀얼 소켓 10 core Xeon E5-2650 CPU와 80GB 메모리가 설치되어 있으며, 각각의 스토리지 노드에는 40개의 하드디스크가 4개의 RAID6 스토리지 타겟으로 구성되어 있다.

아래 Figs. 5, 6은 크기가 각각 102GB, 208GB인 유전체 리드 원시 데이터들의 전처리에 소요되는 시간을 보여주고 있다. 그래프의 맨 왼쪽 부분은 기존의 방법으로 하나의 노드에서 BWA, samblaster, samtools를 연속으로 실행할 때의 소요 시간을 나타낸다. 이 때, 멀티스레드 옵션을 활성화하여 20개의 코어를 최대한 활용하도록 하였다. 오른쪽의 두 결과는 본 논문에서 제시한 병렬화 기법을 이용하여 각각 4대, 8대의 노드에서 병렬로 수행하였을 때의 결과를 나타낸다. BWA는 KBigBWA를 통해 Hadoop on Lustre를 통해 병렬로 수행되며 samblaster와 samtools는 GNU parallel 을 이용하여 각각의 노드에서 작업을 나누어 처리한다. 각 단계 사이에서의 데이터 재배치 및 전송은 IPoIB 소켓 통신을 통해 이루어지고, 파일을 병합하는 과정은 MPI-IO를 이용해 병렬화하였다. 각 단계에

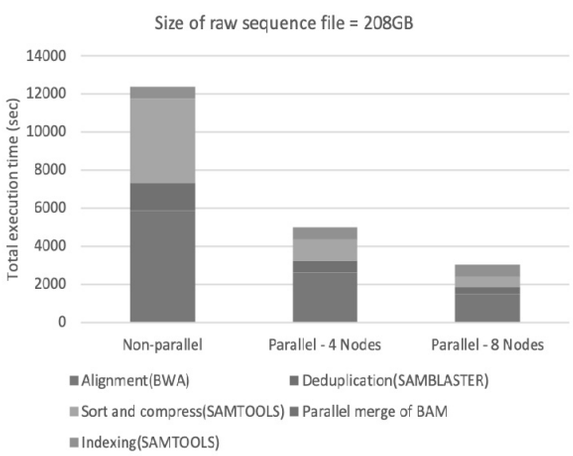


Fig. 5. Breakdown of Execution Time of Preprocessing 208GB NGS Data with Parallel Mechanism

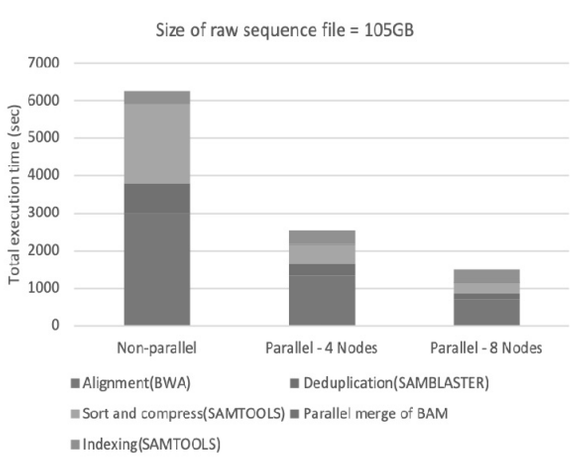


Fig. 6. Breakdown of Execution Time of Preprocessing 105GB NGS Data with Parallel Mechanism

서 노드 당 5개의 작업 프로세스가 병렬로 데이터 처리 작업을 처리한다. 멀티스레딩이 가능한 BWA와 samtools는 각 프로세스마다 4개의 스레드를 사용하여 모든 노드의 코어를 최대한 사용하였다.

실험을 통해 모든 전처리 단계를 병렬화 함으로써 더 많은 자원을 투입하면 전처리 속도를 유의미하게 향상 시킬 수 있음을 확인하였다. 데이터 재배치 과정에서 추가되는 시간은 병렬화를 통해 얻는 이득에 비해 크지 않다고 판단된다. 또한 두 방식으로 생성된 결과물을 비교하였을 때 데이터 처리 순서의 변경으로 인해 발생하고 최종 분석에 영향을 미치지 않는 오차를 제외하고는 동일한 결과물을 생성하는 것을 확인하였다. 분석의 종류에 따라서 실험에서 사용한 데이터 보다 수 배 이상 큰 데이터를 실제로 사용하는 경우도 존재하기 때문에 병렬 자원을 투입하여 소요시간을 단축시키는 것의 이점이 더 부각 될 수 있을 것이다.

5. 결 론

본 연구를 통해 Lustre, MPIIO 등 HPC 기술을 적극적으로 적용하여 병렬 자원을 활용하여 유전체 시퀀스 데이터의 전처리 과정을 소요 시간을 크게 감소시킬 수 있는 병렬화 하는 기법을 제안하고 구현 및 실험을 통해 입증하였다. 이 기법을 이용하여 응급 환자의 진단 등 시급성이 요구되는 상황에서 고성능의 병렬 자원을 이용하여 유전 변이 검출 프로세스의 효율성을 크게 증가시킬 수 있다. 유전 정보를 의료 진단 및 치료에 활용하고자 하는 요구사항 및 시장의 크기는 점점 더 커질 것으로 예상된다. 본 연구에서 제안한 병렬화를 기법을 통한 유전체 분석 시간의 단축이 대형 병원 등 의료 업계에서 유전체 분석 정보를 활용하는 서비스의 폭을 넓힐 수 있을 것으로 기대된다.

References

[1] S. Goodwin, J. D. McPherson, and W. R. McCombie, “Coming of age: ten years of next-generation sequencing technologies,” *Nature Review Genetics*, Vol.17, No.6, pp.333-351, May 2016.

[2] “Sequence Alignment/Map Format Specification”, The SAM/BAM Format Specification Working Group [Internet], https://samtools.github.io/hts-specs/SAMv1.pdf

[3] An introduction to Next-Generation Sequencing Technology, Illumina, Inc., [Internet], https://www.illumina.com/documents/products/illumina_sequencing_introduction.pdf

[4] H. Li and R. Durbin, “Fast and accurate long-read alignment with Burrows-Wheeler transform,” *Bioinformatics*, Vol.26, No.5, pp.589-595, 2010.

[5] Faust, Gregory G., and Ira M. Hall, “SAMBLASTER: Fast Duplicate Marking and Structural Variant Read Extraction,” *Bioinformatics*, Vol.30, No.17, pp.2503-2505, 2014.

[6] H. Li, B. Handsaker, A. Wysoker, T. Fennell, J. Ruan, N. Homer, G. Marth, G. Abecasis, R. Durbin, and 1000 Genome Project Data Processing Subgroup, “The Sequence Alignment/Map format and SAMtools,” *Bioinformatics*, Vol.25, No.16, pp.2078-2079, 2009.

[7] L. Pireddu, S. Leo, and G. Zanetti, “SEAL: a distributed short read mapping and duplicate removal tool,” *Bioinformatics*, Vol. 27, No.15, pp.2159-2160, Aug. 2011.

[8] T. Nguyen, W. Shi, and D. Ruden, “CloudAligner: A fast and full-featured MapReduce based tool for sequence mapping,” *BMC Res Notes*, Vol.4, No.1, pp.171, Jun. 2011.

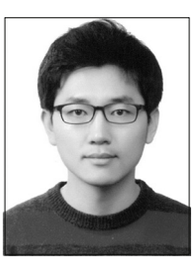
[9] M. C. Schatz, “CloudBurst: highly sensitive read mapping with MapReduce,” *Bioinformatics*, Vol.25, No.11, pp.1363-1369, Jun. 2009.

[10] J. M. Abuín, J. C. Pichel, T. F. Pena, and J. Amigo, “BigBWA: approaching the Burrows-Wheeler aligner to Big Data technologies,” *Bioinformatics*, Vol.31, No.24, pp.4003-4005, 2015.

[11] D. Decap, J. Reumers, C. Herzeel, P. Costanza, and J. Fostier, “Halvade: Scalable Sequence Analysis with MapReduce,” *Bioinformatics*, Vol.31, No.15, pp.2482-2488, 2015.

[12] E.-K. Byun, J. Lee, S. J. Yu, J.-H. Kwak, and S. Hwang, “Accelerating Genome Sequence Alignment on Hadoop on Lustre Environment,” *2017 IEEE 13th International Conference on E-Science*, pp.436-437, 2017.

[13] Lustre Hadoop Plugin, Seagate [Internet], https://github.com/Seagate/lustrefs



변 은 규

https://orcid.org/0000-0002-1811-9136

e-mail : ekbyun@kisti.re.kr

2003년 한국과학기술원 전산학과(학사)

2011년 한국과학기술원 전산학과(Ph.D., 석・박사통합)

2011년~2012년 SK텔레콤 매니저

2013년~현 재 한국과학기술정보연구원 선임연구원

관심분야 : 분산병렬시스템, 빅데이터, HPC, 스토리지 시스템

곽 재 혁

https://orcid.org/0000-0003-0058-8417

e-mail : jhkwak@kisti.re.kr

2001년 아주대학교 정보및컴퓨터공학부 (학사)

2003년 서울대학교 전기및컴퓨터공학부 (석사)

2003년~현 재 한국과학기술정보연구원 선임연구원

관심분야 : 분산컴퓨팅, 고성능컴퓨팅, 빅데이터컴퓨팅

문 지 협

https://orcid.org/0000-0002-5683-6021

e-mail : munji@kisti.re.kr

2009년 동국대학교 컴퓨터공학과(학사)

2011년 동국대학교 컴퓨터공학과(석사)

2017년 과학기술인연합대학원대학교 (UST) 기능유전체학(박사)

2017년~현 재 한국과학기술정보연구원 선임연구원

관심분야 : 차세대 염기서열 분석