Spark 기반의 분산 병렬 DNA 중복제거 방법

Spark-based Distributed and Parallel DNA Deduplication Method

문 지 혜, 이 현 병, 송 석 일*
한국교통대학교

Jihye Moon, Hyeonbyeong Lee, Seokil Song*
Korea National University of Transportation

요약

이 논문에서는 DNA 분석단계 중 하나인 DNA 리드(Read)에 대한 중복제거 방법을 분산 병렬처리 기법을 적용하여 가속화하는 방법을 제안한다. 기존 제안된 중복제거 기법을 Spark을 기반으로 병렬처리 되도록 하는 접근방법을 취한다. 제안하는 기법은 실험을 통해서 기존 중복제거 기법과 비교하여 성능을 입증한다.

I. 서론

NGS(Next Generation Sequencing)는 차세대 염기서열 분석법으로, 유전체 논문의 핵심 기술이다. NGS는 하나의 유전체를 무수히 많은 리드(Read)로 분해하고 증폭하여 컴퓨터에 입력하고 다시 컴퓨터가 이를 조합하여 대량의 염기서열 정보를 생산한다. NGS는 일반적으로 유전체 정보 생성, 지역 정렬, 중복 제거, 유전자 변이 검지 등의 분석 과정들이 파이프라인(Pipeline) 형태로 구성되어 있다. 유전체 정보 생성 단계에서는 유전체 분석 장비에서 데이터를 읽어 컴퓨터로 입력한다. 지역 정렬 단계에서는 입력된 데이터를 참조 게놈(reference genome)에 사상(mapping)하는 작업을 수행한다. 이 결과를 다음 단계인 중복 제거 단계로 전달하면, 중복되는 리드를 삭제하거나 표시하는 작업을 수행한다. 이 과정을 거치게 된 유전체 정보는 유전자 변이 검지, 친자 확인 검사, 원산지 판별 등의 후속 단계로 전달된다.

이 논문에서는 분산 병렬 처리 플랫폼을 기반으로 NGS 파이프라인 단계 중 하나인 중복 제거 단계의 시간을 줄이기 위한 방법을 논문 개발한다. NGS 데이터를 생성하는 과정 중 중합 효소 연쇄 반응(PCR, Polymerase Chain Reaction) 단계에서 같은 리드가 중복해서 생성될 수 있다. 생성된 중복 리드는 이후 분석 과정에서 오류나 성능저하의 원인이 될 수 있어서 중복 리드를 삭제하거나 표시하는 과정은 NGS 파이프라인 단계중 하나이며 이를 중복 제거라고 한다.

기존에 NGS에서 리드에 대한 중복 제거 기법으로 Picard Picard, ParDRe, SAMBLASTER 등이 제안된 바 있다. 이중 Picard는 가장 일반적인 도구로 널리 사용되고 있다. 현재 일반적으로 사용되고 있는 도구이다. 하지만, 단일 프로세스로 중복 제거를 수행하므로, 대량의 데이터에 대해서는 빠른 처리가 어렵다. Fulcrum은 맵 리듀스(Map Reduce)를 이용하여 분산 병렬 형식으로 중복 제거를 수행하는 방법이다. 맵 리듀스를 통해 분산 병렬 처리를 할 수 있다는 장점이 있지만 맵 리듀스가 기본적으로 가지고 있는 중간 결과물 입출력을 위한 디스크 IO로 인한 성능 저하 문제가 있다.

ParDRe는 MPI(Message Passing Interface)를 기반으로 병렬 형태로 처리가 가능하다. Fulcrum보다 높은 성능을 낼 수 있지만, 이전 단계의 출력이 디스크에 저장된 후 이를 다시 읽어서 중복 제거를 수행하게 되어 디스크 IO로 인한 성능 저하가 발생한다. 이에 반해서 SAMBLASTER는 유닉스의 파이프(Pipe)를 통해서 이전 단계의 출력을 수신하므로 출력을 저장해서 다시 읽는 디스크 IO를 줄일 수 있다. 하지만, 단일 노드에서 데이터를 처리하기 때문에, 추가적인 성능 향상이 어렵다는 한계가 있다.

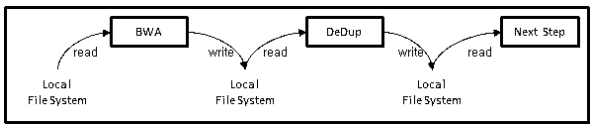
이 논문에서는 기존 중복 제거 기법들이 가지는 한계를 인-메모리, 분산 및 병렬 처리, 스트림 처리 기법을 적용하여 개선한다. 이 논문에서는 NGS에서 발생하는 중복 리드 제거를 분산 병렬처리 플랫폼인 스팍(Spark)및 스팍 스트리밍 (Spark Streaming)을 이용하여 빠르게 수행하는 방법을 개발한다. 스팍은 인-메모리 분산 병렬처리 플랫폼으로 RDD(Resilient Data Set) 모델을 이용하여 중간 결과물을 디스크에 저장하지 않고 처리가 가능한 기술이다. 또한, 스팍 스트리밍은 스팍의 프로그래밍 모델을 이용하여 마이크로 배치 (Micro Batch) 형태로 스트림을 처리하는 기술이다.

이 논문에서는 NGS 파이프라인에서 중복제거 앞 단계의 도구 (주로 BWA 또는 BigBWA) 들의 출력을 입력 스트림으로 받아서 점진적(Incremental)으로 중복제거를 수행하는 기술을 개발한다. 특히, 기존에 이미 검증된 중복제거 도구 (Picard 또는 Samblaster 등)를 분산 병렬처리 플랫폼상에서 활용할 수 있도록 하는 기술을 개발한다. 개발한 중복제거 도구를 기존 방식과 시간측면과 중복제거 효율 측면에서 비교하여 우수성을 입증한다.

Ⅱ. 제안하는 중복제거 기법

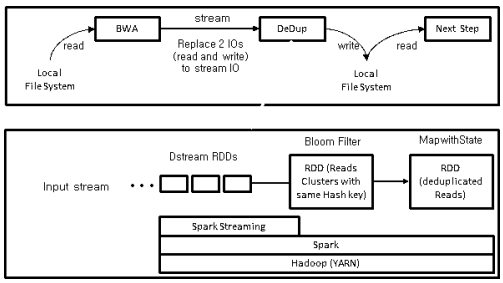
이 논문에서 개발하는 중복제거 기법의 기본 접근 방향이 그림 1과 그림 2에 나타 있다. 기존 NGS 파이프라인 도구간의 일반적인 데이터 입출력 방식은 그림 1과 같다. 즉, 매 단계 별로 2회의 디스크 입출력이 필요하다. 그림 2는 DeDup 도구와 앞단계 대표적인 도구인 BWA 간에 입출력 스트림을 통한 데이터 전달방법을 보여주고 있다. 그림에서와 같이 스트림 방식의 입출력은 디스크 IO 2회를 줄일 수 있다.

이 논문에서 개발하는 중복제거 방식은 이와 같은 기본 방향을 취한다. 그림 2의 아래에서처럼 인-메모리 분산 병렬처리 플랫폼인 스팍을 기반으로 대부분의 처리가 이루어진다. 특히, BWA와의 데이터 입출력을 위해 스팍 스트리밍 기술을 이용한다. 스팍 스트리밍은 BWA의 출력을 입력 스트림으로 받아서 마이크로 배치 형태로 점진적인 중복제거를 가능하게 한다.



▶▶ 그림 1. 기존 NGS 파이프라인 도구간 데이터 입출력 방식

그림 2의 하단에서는 모든 처리가 스팍을 기반으로 수행되는 예시이다. 모든 처리를 스팍을 기반으로 수행하므로 최적의 병렬성을 얻게 되어 처리 속도를 크게 향상할 수 있다. 하지만, 신규로 개발하는 NGS 도구가 인정을 받아서 널리 사용되게 하기 위해서는 상당한 검증 기간과 노력이 필요하다. 이를 어느 정도 해소하기 위해서는 기존의 검증된 NGS 도구를 분산 병렬처리 플랫폼상에서 동작할 수 있도록 변형하여 성능을 개선하는 방식이 있을수 있다. 기존에 리드의 지역정렬 도구로 널리 활용되고 있는 BWA를 맵 리듀스 분산 병렬처리 플랫폼에서 동작할 수 있도록 하여 처리 속도를 크게 향상 시킨 BigBWA가 그 대표적인 사례이다.



▶▶ 그림 2. 이 논문에서 제안하는 스트림 기반의 도구간 데이터 입출력 방식

이 논문에서도 이를 위해 기존의 중복제거 도구인 SAMBLASTER를 스팍 분산 병렬 처리 플랫폼 상에서 동작할수 있는 기술을 개발한다. 이 논문에서 개발하는 기술은 반드시 SAMBLASTER를 대상으로 하는 것은 아니며 일부 수정을 통해 다른 도구를 같이 사용할수 있다.

Ⅲ. 성능 평가

과제에서 개발한 SDedup의 성능을 검증하기 위해서 Samblaster 와 성능을 비교하였다. 성능 비교에 사용된 실험 환경은 표 1과 같다. 데이터는 18Gbyte의 크기를 사용하였다.

표 1. 실험 환경

구분	SDedup	Samblaster
클러스터	Amazon EC2 m4 8 Nodes 4 인텔 2.3GHz 제온® E5-2686 v4 CPUs, HDD 200G 16G RAM	Amazon EC2 R3 1 Node 4 인텔 2.3GHz 제온® E5-2686 v4 CPUs, HDD 200G 16G RAM
SW	Spark 2.1, Hadoop 2.7.3 Samblaster 0.1.24 , BigBWA 2.0, BWA 3.0	

다음 표 2는 성능평가 결과를 보여주고 있다. 각 대상별로 수행시간, 중복제거율을 측정하였다. 표에서처럼 제안하는 SDedup의 수행시간이 Samblaster에 비해 약 5분 32초 (62% 단축) 가량 빨랐으며 중복제거율은 1.65% 만큼 더 낮았다.

표 2. 실험 결과

구분	SDedup	Samblaster
수행 시간	9m10s	14m42s
중복 제거율(%)	3.37	5.02%

Ⅳ. 결론

이 논문에서는 기존 Samblaster를 Spark을 이용하여 분산 병렬로 처리하는 기법을 제안하였다. 향후 연구에서는 확장성을 보기 위해 제안하는 기법의 노드의 개수 및 데이터의 크기를 변경시켜 가면서 실험을 수행할 계획이다.

■ 참 고 문 헌 ■

[1] Picard, https://broadinstitute.github.io/picard/
[2] Burriesci, Matthew S., Erik M. Lehnert, and John R. Pringle. "Fulcrum: condensing redundant reads from high-throughput sequencing studies." Bioinformatics 28.10 (2012): 1324-1327.
[3] González-Domínguez, Jorge, and Bertil Schmidt. "ParDRe: faster parallel duplicated reads removal tool for sequencing studies." Bioinformatics (2016)
[4] Faust, Gregory G., and Ira M. Hall. "SAMBLASTER: fast duplicate marking and structural variant read extraction." Bioinformatics (2014)