Seed기반의 short read aligner 구현에 관한 연구

지민근*, 김정규*, 이강만**
동국대학교 멀티미디어공학과
e-mail: *{mingeun, jkkim}@mme.dongguk.edu, **gangman@dongguk.edu

Mingeun Ji*, Jeongkyu Kim*, Gangman Yi**
Department of Multimedia Engineering, Dongguk University, Seoul, Korea
**Corresponding author

요 약

차세대 게놈 시퀀싱(NGS) 기술이 발전하면서 방대하게 축적된 유전체 데이터를 분석하기 위해 다양한 시퀀스 정렬 연구가 진행되고 있다. 시퀀스 정렬 중 잘 알려진 BLAST에서는 휴리스틱 기반의 시퀀스 정렬을 수행하여 긴 리드 시퀀스에 대해 속도와 안정성이 보장되지만 짧은 리드 시퀀스에 대해서는 성능이 저하되는 문제가 있다. 이 문제를 해결하기 위해 본 논문에서는 레퍼런스 시퀀스와 쿼리 시퀀스를 Seed 기반으로 분리하여 정렬을 수행한다. 최종적으로는 contig를 추출하고 레퍼런스-쿼리간 유효한 contig만 선별하여 빠르게 짧은 리드 시퀀스들의 정렬을 수행할 수 있는 정렬기를 구현하고자 한다.

1. 서론

차세대 게놈 시퀀싱(NGS) 기술이 발전함에 따라 microarray 데이터나 대사 경로 데이터베이스를 활용한 여러 생물의 게놈내 유전자들을 분석하는 연구가 활발해지면서 다양한 유기체의 게놈 데이터가 축적되고 있다. 유전체 데이터는 illumina등의 다양한 시퀀서(sequencer)를 통해 유기체의 DNA로부터 생성된 작은 contig 들을 병합하여 super-contig들을 생성하고 이를 반복하는 과정으로 생성되어 축적된다. 유전체 데이터를 생성하는 최소 단위인 리드 시퀀스부터 contig 또는 super-contig를 생성 및 병합하는 단계에서도 짧은 리드 시퀀스 정렬이 수행된다.

기존 시퀀스 정렬을 위한 잘 알려진 기존 연구로는 BLAST(Basic Local Alignment Search Tool)가 있으며 이는 시퀀스들에 대해 휴리스틱 기반의 시퀀스 정렬을 수행하여 속도 및 정확성을 보장하지만 짧은 리드 시퀀스들에 대해서는 정렬의 정확성이 떨어지는 문제가 있다.

위와 같은 문제를 해결하기 위해 본 연구에서는 짧은 리드 시퀀스들에 대한 시퀀스 정렬 속도를 향상시킬 수 있는 Seed기반 짧은 시퀀스 정렬기를 구현하고자 한다.

** 교신저자 : 이강만(gangman@dongguk.edu)
*** 이 논문은 교육부와 한국연구재단의 BK21플러스 사업의 장학지원을 받아 수행된 연구결과임.

2. 시스템 설계

본 연구에서 구현한 Seed기반 짧은 리드 시퀀스 정렬기는 사용자로부터 2개의 nucleotide 형태의 시퀀스와 Seed 값을 입력받아 각 시퀀스를 Seed만큼 짧은 리드 시퀀스 형태로 분리한다. 이를 적절하게 병합하여 더 긴 시퀀스로 생성한다. 생성된 시퀀스들을 이용하여 리드 시퀀스 정렬을 수행하고 최적화된 정렬 결과를 사용자에게 그래프 또는 파일 형태로 제공한다.

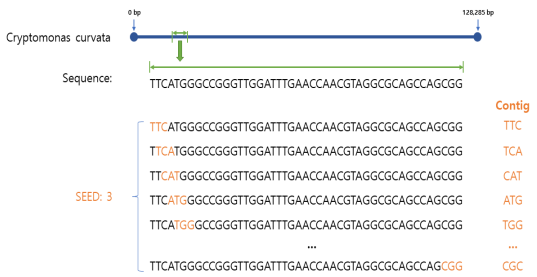
2.1 입력 파라미터

짧은 리드 시퀀스 정렬을 수행하기 위해서는 정렬을 수행할 레퍼런스 시퀀스와 쿼리 시퀀스 데이터의 Seed가 필요하며, 각 시퀀스는 FASTA 포맷형태로 구성되어 짧게는 몇 백 base-pair에서 길게는 수 십만 base-pair형태로 구성된다[그림 1].

Seed는 각 시퀀스에 대해 분리할 최소 길이를 지정하는 옵션으로 분리된 길이에 따른 짧은 리드 시퀀스 정렬을 수행할 수 있다. Seed에 따른 시퀀스 분리는 [그림 2]와 같다.

```
1  >NC_035720.1 Cryptomonas curvata strain CNUKR plastid, complete genome
2  TTCATGGGCCGGGTTGGATTTGAACCAACGTAGGCGCAGCCAGCGGATTTACAGTCCGCCCCCTTTAACC
3  CCTCGGGCACCGACCCAATATTATATTATTAATAGTATAGGATTAATTTAACATAGAAATAAAAACAATA
4  CAAGCGATTGTAGAAATGGATAGCCACAATTTGGGCTATCCAATCTTTTTAATCGAATCTTTGTTGTAAT
5  CTAGTAGCTTTACCAAATCGTAATCGAAGGTAGTATAATTTAGACCTTCTAACTGTGGAACTTCTTAAGA
6  TCTCTATACTTTTCAATTTAGGCGAGTTTATTAGATAAACCCTTTCAACTCCAACACCTTGCATAACACG
7  TCTTAAAGTTATTGTGGCATCTAATCCTGAACTTTTTTTTGCTATAACTACTCCTTCAGTAAATTGAACT
8  CTTTCTTTATTACCCTCAGTTATGGATACCCCGAGTTTTAGAGTATCCCCGACTCGGATTAGAGGTAAAT
9  TTCTATTTAAAAATTTATCTTCTAAATCTTTTTGGGCTAGTAAATGTAGGTTATTTTTTAATTTCATAAT
10 TTTATTTCGTTTAATGAGCCAGATATAGAAAACATTAAGTCTATTGATTCATATTAAATCAGTAAACTTC
11 TTAAGTCTATAAATATTTGAAATTTTTATTAGGTAAGATTATCTATACATTTAACATACTTAAAGGACTT
12 ACCCTAACTAATTATTATATAGACTATCAACGATAATAAATATTTAGTAGAACTATAAATATATGTTCGA
13 ACGTTTTACCGAAAAAGCTATAAAGGTTATCATGCTAGCTCAGGAAGAAGCTAGAAGATTAGGTCATAAT
14 TTTGTAGGTACTGAACAGATTTTATTAGGCCTTATAGGAGAAGGTACAGGGATTGCGGCAAAGGTTTTAA
15 AATCAATGGGAGTAAATTTAAAAGATGCCAGAGTTGAAGTTGAGAAAATTATAGGTAGAGGATCGGGTTT
16 TGTCGCGGTAGAAATTCCATTTACTCCAAGAGCAAAACGAGTTTTAGAACTATCTCTAGAAGAAGCTAGA
17 CAACTTGGTCACAATTATATAGGTACTGAACATTTGTTACTTGGATTAATAAGAGAAGGTGAAGGGGTAG
18 CTGCTAGAGTATTAGAGAATTTAGCATTAGATTTAACAAAGATAAGAACTCAAGTAATCAGGCTACTTGG
19 AGATACAGCAGAGGCTACTACTGGAAGTCCTCAAAATAAAGGAAAAACTCCAACCCTTGAAGAATTTGGG
20 AGTAATCTGACTCAAAAGGCTGCCGAAGGTAAATTAGACCCTGTAATTGGTAGACAAAAGGAAATTGAGA
21 GAGTTATCCAAATACTTGGTCGGAGAACAAAAAATAACCCAATATTAATTGGAGAGCCAGGGGTTGGTAA
22 AACGGCAATAGCTGAGGGTTTGGCCCAGAGAATAACCAATCGAGATGTGCCAGATATTTTAGAAGATAAA
```

[그림 1] 레퍼런스 및 쿼리 시퀀스 샘플 데이터 (Cryptomonas curvata)

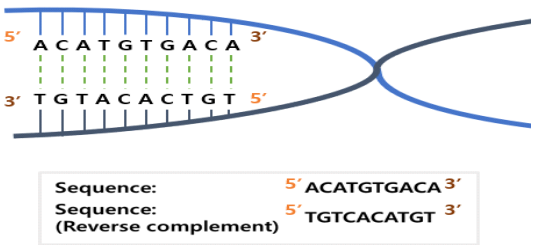


[그림 2] Seed에 따른 시퀀스 슬라이싱

2.2 시퀀스 데이터 전처리

사용자로부터 입력 받은 레퍼런스 시퀀스, 쿼리 시퀀스 데이터를 Seed만큼 슬라이싱하여 분리된 시퀀스를 추출하는 단계이다. 역방향 정렬을 보장하기 위하여 쿼리 시퀀스 데이터를 reverse complement한 데이터로의 변환이 필요하다[그림 3]. 기본 쿼리 시퀀스 데이터를 정방향 쿼리 시퀀스라 하며 reverse complementary를 통해 생성한 시퀀스 데이터를 역방향 쿼리 시퀀스라 한다.

전처리 단계는 레퍼런스 시퀀스, 정방향 쿼리 시퀀스, 역방향 쿼리 시퀀스에 대해 Seed의 크기로 분리하여, 각 시퀀스 데이터에 대한 분리된 리드 시퀀스들을 추출한다[그림 4]. 이 시퀀스들의 사이즌 짧은 bp로 구성되어 있어 시퀀스 내 중복되는 부분들이 발생할 수 있다. 중복되는 시퀀스들은 시퀀스내 위치를 값으로 하여 각각 구분한다.



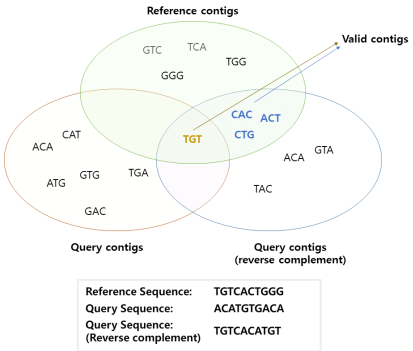
[그림 3] 시퀀스 및 reverse complement 시퀀스

```
>Reference data
GGGGTTAAAAGGGGGCCCTAAGTATAGACCAGGGGTTAAAAGGGGGCCCTAAGTATAGACCAGGGGTTAAAAGGGGGCCCTAAGTATAGACCA

>Query sequence data
GGGGTTAAAAGGGGGCCCTAAGTATAGACCA

>Query sequence data(reverse complement)
TGGTCTATACTTAGGGCCCCCTTTTAACCCC

--------------------------------------------------------------------------------
>Reference contig
['GGG', 'GGT', 'GTT', 'TTA', 'TAA', 'AAA', 'AAG', 'AGG', 'GGC', 'GCC', 'CCC', 'CCT', 'CTA', '
AGT', 'GTA', 'TAT', 'ATA', 'TAG', 'AGA', 'GAC', 'ACC', 'CCA', 'CAG']
>Query contig
['GGG', 'GGT', 'GTT', 'TTA', 'TAA', 'AAA', 'AAG', 'AGG', 'GGC', 'GCC', 'CCC', 'CCT', 'CTA', '
AGT', 'GTA', 'TAT', 'ATA', 'TAG', 'AGA', 'GAC', 'ACC', 'CCA']
<Query contig(reverse complement)
['CCC', 'CCA', 'CAA', 'AAT', 'ATT', 'TTT', 'TTC', 'TCC', 'CCG', 'CGG', 'GGG', 'GGA', 'GAT', '
TCA', 'CAT', 'ATA', 'TAT', 'ATC', 'TCT', 'CTG', 'TGG', 'GGT']
--------------------------------------------------------------------------------
```

[그림 4] 시퀀스 데이터 및 contig

2.3 유효 데이터 추출

짧은 리드 시퀀스 정렬을 빠르게 수행하기 위하여 각 시퀀스의 모든 분리된 리드 시퀀스 중에서 레퍼런스-정방향 쿼리, 레퍼런스-역방향 쿼리 간 교집합으로 존재하는 시퀀스들을 유효 데이터로 선별하여 각 시퀀스 유효 시퀀스들만 추출한다[그림 5].

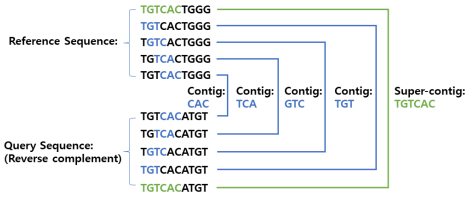


[그림 5] contig 벤다이어그램

각 시퀀스에서 유효한 contig들은 정방향으로 표시된 짧은 리드 시퀘스는 레퍼런스-정방향 쿼리 시퀀스간의 매칭된 시퀀스이고, 역방향으로 표시된 시퀀스는 레퍼런스-역방향 쿼리 시퀀스간의 매칭된 시퀀스이다.

2.4 데이터 병합

레퍼런스-정방향 또는 레퍼런스-역방향 시퀀스간의 유효한 시퀀스들의 nucleotide bp 중 일부분이 겹쳐져서 하나의 시퀀스로 확장하는 단계이다. 이 단계에서는 병합 가능한 시퀀스들을 하나의 시퀀스로 병합하여 더 긴 시퀀스를 생성한다. [그림 6]은 유효한 짧은 시퀀스들을 나열하고 병합 가능한 시퀀스들을 더 긴 시퀀스로 생성한다.



[그림 6] contig 및 super-contig

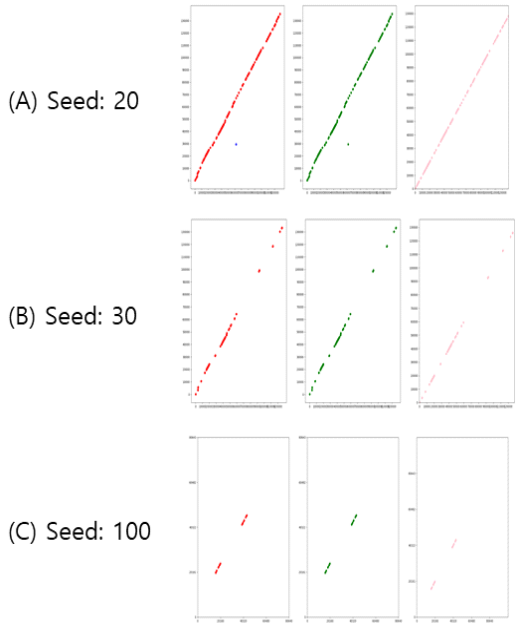
2.5 최적화

병합 단계에서 생성된 긴 리드 시퀀스 중 레퍼런스 시퀀스 기준으로 시퀀스의 길이가 길거나 레퍼런스 및 쿼리 시퀀스에서 비슷한 위치에 생성된 더 긴 리드 시퀀스를 선별하여 정렬을 최적화한다.

3. 최종 결과

본 논문에서 구현한 프로그램은 Nucleotide로 구성된 2개의 시퀀스(레퍼런스 & 쿼리 시퀀스)와 시퀀스를 슬라이싱할 값을 사용자로부터 입력받아 리드 시퀀스 정렬을 빠르게 수행하고 결과를 화면에 출력한다.

본 논문에서 구현한 정렬기를 약 100,000bp 크기의 레퍼런스 및 쿼리 시퀀스를 Seed값 20과 30, 100으로 하여 짧은 리드 시퀀스 정렬을 수행한 결과는 [그림 7]과 같다. 시퀀스 길이에 비해 너무 작은 Seed값을 사용할 경우 데이터 병합하는 단계에서 많은 시간이 소요된다. 반면, 높은 Seed 값을 사용할수록 출력되는 시퀀스의 개수가 줄어든다.



[그림 7] Seed 값에 따른 짧은 리드 정렬 결과

약 100,000bp 크기의 레퍼런스 및 쿼리 시퀀스에 대해 Seed 값을 100 이상으로 설정하여 리드 정렬을 수행할 겨우 각 시퀀스로부터 슬라이싱된 read 데이터가 정방향 또는 역방향으로 매칭되는 부분이 없어 화면에 매칭된 시퀀스가 없는 빈 결과가 출력된다. 따라서, 레퍼런스 및 쿼리 시퀀스의 bp 크기에 따라 Seed 값 조절이 필요하다.

본 논문에서 구현한 정렬기는 리드 정렬된 결과를 빠르게 시각화하여 정보를 제공하며 Seed기반으로 두 시퀀스에 대한 짧은 리드 정렬을 수행하고 즉시 그래프로 결과를 확인할 수 있다.

4. 개발 환경 및 구현

본 논문에서 구현된 프로그램은 Python3 언어기반으로 구현되었으며, 리드 시퀀스 정렬의 시각화를 위해 Python 라이브러리 matplotlib를 사용하였으며 Linux기반의 Mint OS가 탑재된 AMD Ryzen 3900X, 32GB RAM, 1TB NVMe M.2 SSD 사양의 데스크탑에서 개발을 수행하였다.

5. 결론

시퀀스 정렬 속도 향상을 위해 본 논문에서는 레퍼런스 시퀀스와 쿼리 시퀀스를 Seed기반의 짧은 리드 시퀀스 정렬기를 구현하였다. 레퍼런스 시퀀스와 쿼리 시퀀스의 bp 길이와 Seed값에 따라 결과가 상이하게 나타났으며 Seed 값에 따른 짧은 리드 시퀀스 정렬 보정이 필요하다. 추후 본 논문에서 구현된 정렬기의 성능 향상을 위한 알고리즘 개선이 필요하며 이를 위해 멀티 코어를 사용한 리드 시퀀스 정렬 알고리즘 구현 등의 연구를 수행하고자 한다.

참고문헌

[1] Kurtz S, Phillippy A, Delcher AL, et al. Versatile and open software for comparing large genomes. Genome Biol. 2004;5(2):R12. doi:10.1186/gb-2004-5-2-r12

[2] Leimeister CA, Boden M, Horwege S, Lindner S, Morgenstern B. Fast alignment-free sequence comparison using spaced-word frequencies. Bioinformatics. 2014;30(14):1991 - 1999. doi:10.1093/bioinformatics/btu177

[3] Langmead B, Salzberg SL. Fast gapped-read alignment with Bowtie 2. Nat Methods. 2012;9(4):357 - 359. Published 2012 Mar 4. doi:10.1038/nmeth.1923

[4] Li H, Durbin R. Fast and accurate short read alignment with Burrows-Wheeler transform. Bioinformatics. 2009;25(14):1754 - 1760. doi:10.1093/bioinformatics/btp324