Mapping Sequencing Reads to a Reference Genome

High Throughput Sequencing

Example applications:
- Sequencing a genome (DNA)
- Sequencing a transcriptome and gene expression studies (RNA)
- ChIP (chromatin immunoprecipitation)

Example platforms:
- 454
- Illumina
- SOLiD

Cost per Human Genome

Moore's Law
Sequencing Output

- Hundreds of millions of sequencing reads, each ~200 nts in length
- We need to map each read to the genome, i.e., determine the region of the genome each read corresponds to
**Nucleic Acid Sequencing**

```plaintext
>CGTAGTAGTCACAGTCTACGTATATGGGCTCAGCATATAGCGTATAGCGGACTTAGCCATCG
>GGCGTATAGTCTATATACGACTTATCGGCTCGGTCGCAGAGCAGATATATGCAGTTATATGCTAG
>CCTACGTTATATCGATACTACTAGCTCGTCATGAGCGAGTAGATAGTATGACGAGCGACGATCG
>CGATATTAGCCTAGCATCATTACGGCGAGACTCTCGGCTCGCTATATAGCGCTATAGCGAT
>CGGCTATAGCGCATATGCTCAGTAGCTATTAGCAGTATTACGATTATAGTCTCGGCGCATTAC
>TTTCGCGGGATAAGTCTTCGTCTTATGCGACGATTATACGGCCGTATATTTGCATTTAGCATTT
>GGCGTATGGCGGATATCGGCGCGTCATAGCGCCGATTAGGCTACGCCGATGCATCG
>CGCGATCGCGCGCGATCGCGTCAGTCGCGCAGTAGCGCGGCATAGTCGTATCGGCGCCG
TGACAGAAGCTATAAGAGTCAGTAGATCTGAGTATTAGCATTATCGGCGCGATGCGCGATAACG
GCGTATAGTCTATATACGACTTATCGGCTCGGTCGCAGAGCAGATATATGCAGTTATATGCTA
CGCGATCGCGCGCGATCGCGTCAGTCGCGCAGTAGCGCGGCATAGTCGTATCGGCGCCGATCGC
ATAGCAGCACGTGATAGGATATGCTGCTCGTTCGACTATCATATCTGCGTCTGGTCTAGCA
TGACAGAAGCTATAAGAGTCAGTAGATCTGAGTATTAGCATTATCGGCGCGATGCGCGATAACG
CCTACGTTATATCGATACTACTAGCTCGTCATGAGCGAGTAGATAGTATGACGAGCGACGATCC
CGTAGTAGTCACAGTCTACGTATATGGGCTCAGCATATAGCGTATAGCGGACTTAGCCATCG
```

**Mapping to Reference Genome**

**Reference Genome**

```plaintext
CGTAGTAGTCACAGTCTACGTATATGGGCTCAGCATATAGCGTATAGCGGACTTAGCCATCG
```

**Sequencing Read**

```plaintext
TCATCATATGCAGACGTCTAGAATGCTACCTGAGCAAGCAAGTATATGGCGGCGGGAGGAG
```

**Burrows-Wheeler Transform (BWT)**

```
ATCATTAAATCATG$

ATCATTAAATCATG$  0  $ATCATTAAATCATG  14
TCAATTAACATG$A  1  AAAATG$ATCAT$  6
CATTAATCGAGSAT  2  AAATG$ATCAT$  7
ATTTAACATG$ATC  3  ATCATG$ATCAT$  8
TTAAATCGAGTAAT  4  ATCATAATACATG$  0
AAATG$ATCAT$ATCAT$  5  ATG$ATCAT$ATCAT$  11
AAATG$ATCAT$ATCAT$  6  ATAAATG$ATATG$  3
AAATG$ATCAT$ATCAT$  7  CAG$ATCAT$ATCAT$  10
ATCATG$ATCAT$AAT  8  CATAATG$ATCAT$  2
TCG$ATCAT$ATAT$AAT  9  G$ATCAT$ATAT$AAT  13
CATG$ATCAT$AATAT  10  TAAATG$ATCAT$  5
ATG$ATCAT$AATAT$AAT  11  TATG$ATCAT$AATAT$AAT  9
TG$ATCAT$AATAT$AAT  12  TATG$ATCAT$AATAT$AAT  12
G$ATCAT$AATAT$AAT  13  TATG$ATCAT$AATAT$AAT  4
```

**Efficient Substring Search**

```
ATCATTAAATCATG$

TCA

$ATCATTAAATCATG  14
AAATG$ATCAT$  6
AAATG$ATCAT$  7
ATCATG$ATCAT$  8
ATCATAATACATG$  0
ATG$ATCAT$ATCAT$  11
ATG$ATCAT$ATCAT$  3
CAG$ATCAT$ATCAT$  10
CATAATG$ATCAT$  2
G$ATCAT$ATAT$AAT  13
TAAATG$ATCAT$  5
TCATG$ATCAT$AAT  9
TATG$ATCAT$AATAT$AAT  12
TATG$ATCAT$AATAT$AAT  4
```
Efficient Substring Search

**Range of rows starting with** \([1,7)\)

Thus, the substring is in the reference sequence at indices \(6, 7, 8, 0, 11, 3\).

Range of rows starting with \([1,7)\)

Thus, the substring is in the reference sequence at indices \(6, 7, 8, 0, 11, 3\).
Efficient Substring Search

Range of rows starting with A 
(1,7)

Range of rows starting with CA 
(7,9)

Thus, the substring is in the reference sequence at indices (10,2)

Range of rows starting with TCA 
(11,13)

Thus, the substring is in the reference sequence at indices (9,1)
Efficient Substring Search

ATCATTAAATCATG$  
GTAA$CCTTTTAAAA

TCA

Range of rows starting with A (1,7)

Range of rows starting with CA (7,9)

Range of rows starting with TCA (11,13)

Each step of the search must be fast!
O(1) time.

-----

Precompute Helper Information

GTAA$CCTTTTAAAA

Returns the number of nucleotide characters in the BWT that are lexicographically less than c.

public int getNumberCharactersLessThan(char c);
generateCharactersLessThan('A'); returns 0
generateCharactersLessThan('C'); returns 7
generateCharactersLessThan('T'); returns 10

Returns the number of occurrences of nucleotide character c in the BWT up to but not including index i.

public int getNumberOccurrencesPriorToIndex(char c, int i);
generateOccurrencesPriorToIndex('A', 5); returns 2
generateOccurrencesPriorToIndex('A', 13); returns 4
generateOccurrencesPriorToIndex('T', 8); returns 2

-----

Efficient Substring Search

ATCATTAAATCATG$  
GTAA$CCTTTTAAAA

TCA

Range of rows starting with A (1,7)

Range of rows starting with CA (7,9)

Range of rows starting with TCA (11,13)

How can we compute this?
Efficient Substring Search

A Substring Not in the Reference

Algorithm: Mapping read to genome
Algorithm: Mapping read to genome

Read is processed one character (NT) at a time, from right to left.

Iterative Case: Character (NT) at index \( i \) in read, \( ch \)

The range of rows where the substring from \( i \) onward in the read appears in the BWT is \([start, end]\) where

\[
start = \text{getNumberCharactersLessThan}(ch) + \text{getNumberOccurrencesPriorToIndex}(ch, \text{start for character at index } i+1)
\]

\[
end = \text{getNumberCharactersLessThan}(ch) + \text{getNumberOccurrencesPriorToIndex}(ch, \text{end for character at index } i+1)
\]

Efficient Substring Search with Errors

Suppose we tried to map a read of length 60 nts to a reference genome and found that the read did not map. Perhaps the read contains one or two errors from the sequencing process.

- Break the read up into three pieces and map each of the three pieces to the reference.
- If none of the three pieces map, then there are at least three errors in the read.
- If one or more of the pieces map, then we use that mapping to perform a fast alignment.