COM SCI 122 Week 1

Aidan Jan

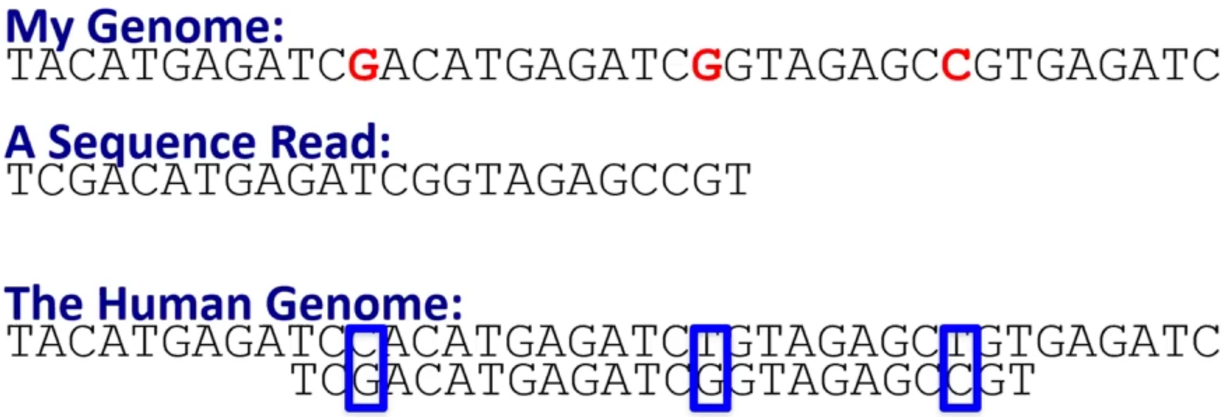
January 6, 2025

Short Read Sequencing Problem

Suppose we have my full DNA sequence of 3B base pairs, and are given reads about 10 base pairs in length. How do you figure out where in the full DNA sequence each read comes from?

- A major issue is that there are only four types of base pairs (A, T, C, G), and therefore only $4^{10} \approx 1000000$ different sequences of length 10. This means each sequence will appear approximately 3000 times in the genome!
- To counter this, we use a *reference*. This takes a long time to compute.

We know that *my* genome is very close to the Human genome.



Now, we need a **map** to speed up the computation time, essentially a table listing positions where sequence reads may be located.

A Trivial Mapping Algorithm

- Suppose we are given a human genome:

```
TACATGAGATCCACATGAGATCTGTAGAGCTGTGAGATC
```

- and a read:

```
TCGACATGAGATCGGTAGAGCCGT
```

- We can simply loop every possible alignment location and find the position under a threshold of mismatches. If the position is below the threshold, we count it as a match.

```
TACATGAGATCCACATGAGATCTGTAGAGCTGTGAGATC
TCGACATGAGATCGGTAGAGCCGT
 || |||  |||||| ||||| ||  = 18 mismatches

TACATGAGATCCACATGAGATCTGTAGAGCTGTGAGATC
 TCGACATGAGATCGGTAGAGCCGT
 | ||| |||||| |||   | |   = 15 mismatches

... eventually ...

TACATGAGATCCACATGAGATCTGTAGAGCTGTGAGATC
         TCGACATGAGATCGGTAGAGCCGT
           |          |      |    = 3 mismatches, below threshold
```

- Such an algorithm is easy to implement, but will take forever to run.

Complexity of Trivial Algorithm

- 3,000,000,000 length genome (N)
- 300,000,000 reads to map (M)
- Reads are of length 30 (L)
- Number of mismatches allowed is 2 (D)
- Each comparison of match vs. mismatch takes 1/1,000,000 seconds (t)
- Total time = N * M * L * t = 27,000,000,000,000 seconds = 854,164 years!

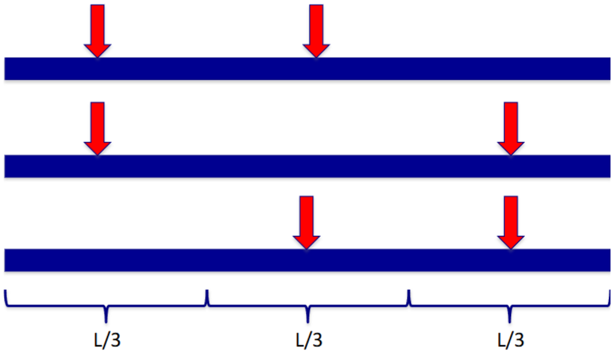
How to improve

Some observations we can make are:

- Most positions in the genome match very poorly.
- We are looking for only a few mismatches (D is small).
- Therefore, a substring of our read will match perfectly.

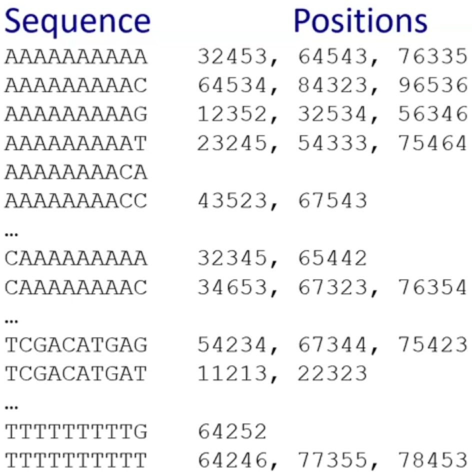
Perfect Matching Read Substrings

The three "worst" possible cases for placement of mutations.



Importantly, in each case, there's a perfect match of L/3.

Intuition: Create an index for the whole genome that maps every sequence of length L/3 to the positions it appears.



Sequence	Positions
AAAAAAAAAA	32453, 64543, 76335
AAAAAAAAAC	64534, 84323, 96536
AAAAAAAAAG	12352, 32534, 56346
AAAAAAAAAT	23245, 54333, 75464
AAAAAAAACA	
AAAAAAAACC	43523, 67543
…	
CAAAAAAAAA	32345, 65442
CAAAAAAAAC	34653, 67323, 76354
…	
TCGACATGAG	54234, 67344, 75423
TCGACATGAT	11213, 22323
…	
TTTTTTTTTG	64252
TTTTTTTTTT	64246, 77355, 78453

Now, for each read, we look at each third chunk in the index.

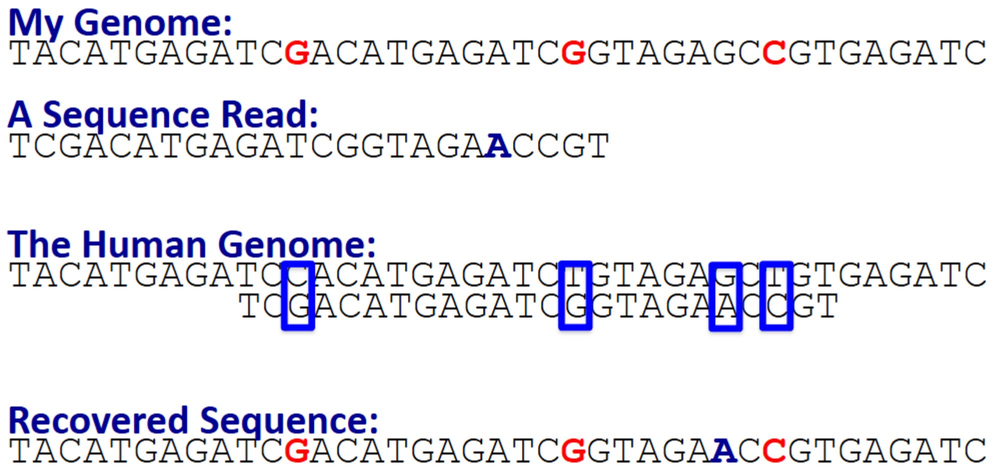
Complexity of Indexing Algorithm

- We need to look up each third of the read in the index.
- For L = 30, our index will contain entries of length 10. Each entry will contain on average $N/(4^{L/3})$ or 3,000 positions.
- For each position, we need to compute the number of mismatches.
- Our running time is $L \cdot M \cdot 3 \cdot N/(4^{L/3}) \cdot T = 81,000,000$ seconds, or 937 days.
- This is a massive improvement from the 854,164 years! If we make L longer, for example, if L = 45, then the time is 81,000 seconds, or 22.5 hours.

More Problems

Read Errors

- Each sequence read can have some random errors, on top of mutations of the genome.



My Genome:

TACATGAGATCGACATGAGATCGGTAGAGCCGTGAGATC

A Sequence Read:

TCGACATGAGATCGGTAGAACCGT

The Human Genome:

```
TACATGAGATCCACATGAGATCTGTAGAGCTGTGAGATC
         TCGACATGAGATCGGTAGAACCGT
```

Recovered Sequence:

TACATGAGATCGACATGAGATCGGTAGAACCGTGAGATC

- The issue here is that we cannot differentiate between errors and mutations.
- To fix this, we collect redundant data. Then, errors may only appear in one of the reads, while mutations would appear in most of the reads (not all because there is always a chance an error was made at the same location as the mutation).

How much coverage do we need to prevent errors?

- If error rate is ε, and we are going to predict the consensus sequence, what is the error rate if the coverage is 3?
- We will make a prediction with an error if two out of our three reads have an error in the same place.
- This is approximately $3\varepsilon^2$.

Repeated Regions

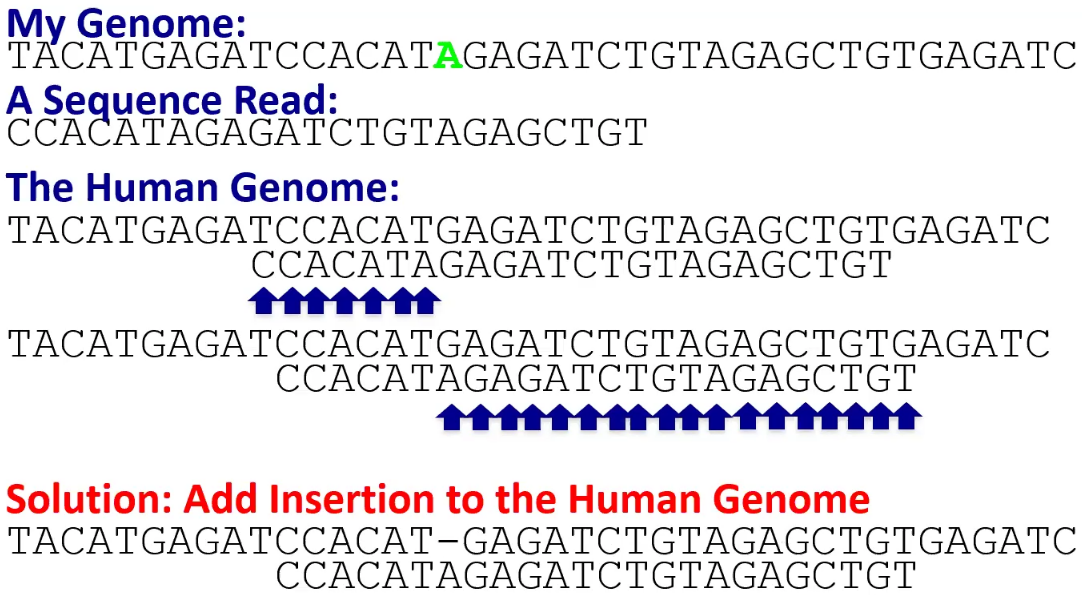
- Sometimes, we may encounter two regions where the sequence is repeated.
- If the reads have a mutation, which region is the mutation at?
- This is a difficult problem that can sometimes be solved by using longer reads. However, in cases where the two regions are long and far apart, there is not a good way to determine the correct region.

Many Mutations

- What happens if there are a lot of mutations right next to each other?
- Too many mismatches to match the read to the reference, Since we don't know where it came from, we can't identify the difference in the target sequence.

Insertions

- Suppose in your genome, there was a insertion. Then, compared to the human genome reference, all the bases before the insertion may be incorrect, or all the bases after the insertion may be incorrect.



My Genome:
TACATGAGATCCACATAGAGATCTGTAGAGCTGTGAGATC

A Sequence Read:
CCACATAGAGATCTGTAGAGCTGT

The Human Genome:

```
TACATGAGATCCACATGAGATCTGTAGAGCTGTGAGATC
          CCACATAGAGATCTGTAGAGCTGT

TACATGAGATCCACATGAGATCTGTAGAGCTGTGAGATC
           CCACATAGAGATCTGTAGAGCTGT
```

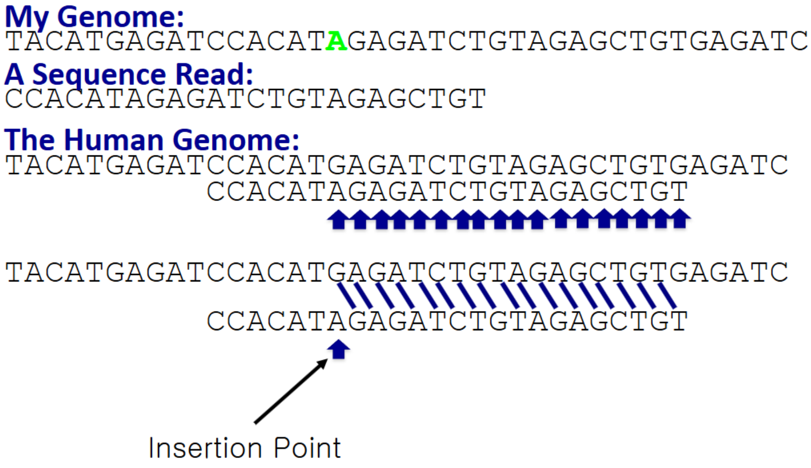
Solution: Add Insertion to the Human Genome

```
TACATGAGATCCACAT-GAGATCTGTAGAGCTGTGAGATC
           CCACATAGAGATCTGTAGAGCTGT
```

- To fix this, we add an insertion to the human genome.
- How do we do this using the thirds technique we discussed earlier?

Finding Insertions

- We can assume that the indel is always in the middle third of a read. This is because of the redundancy of reads; it is very likely at least one read will have the insertion in the middle.
- Therefore, both outside regions of size L/3 will match perfectly
- Since the middle read has an insertion, the middle distance will be L/3 + 1 or L/3 - 1.



My Genome:
TACATGAGATCCACATAGAGATCTGTAGAGCTGTGAGATC

A Sequence Read:
CCACATAGAGATCTGTAGAGCTGT

The Human Genome:

```
TACATGAGATCCACATGAGATCTGTAGAGCTGTGAGATC
          CCACATAGAGATCTGTAGAGCTGT

TACATGAGATCCACATGAGATCTGTAGAGCTGTGAGATC

          CCACATAGAGATCTGTAGAGCTGT
```

Indel Algorithms

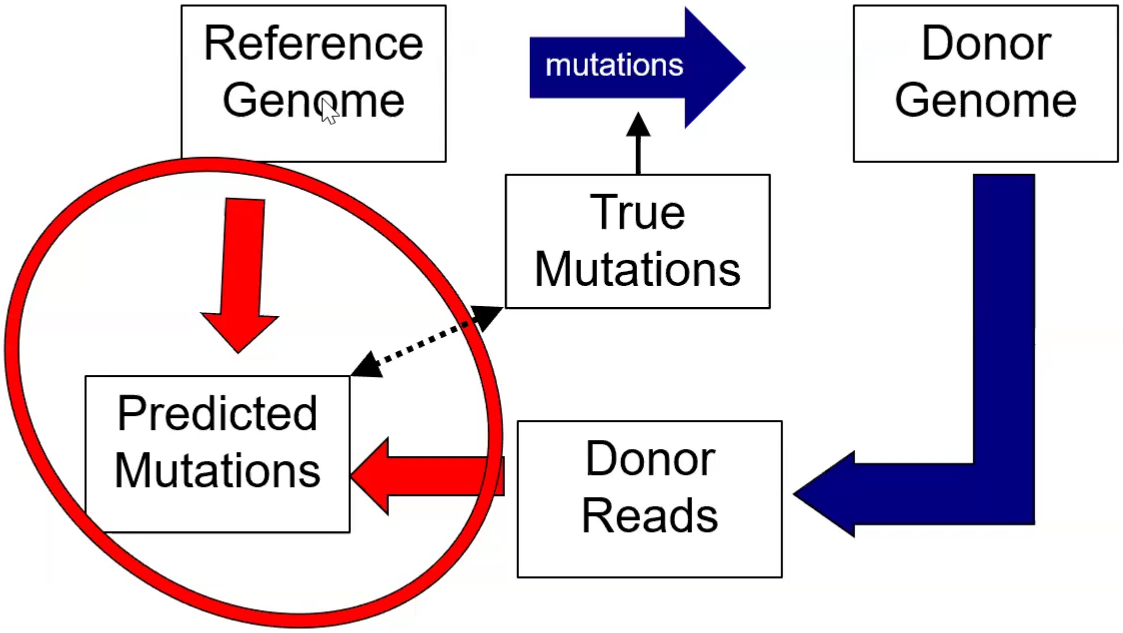
- Trivial Algorithm
  - Try all insertion points for a read
  - If read matches (with insertion) below number of mismatches, then we declare a match and identify and indel

- More Efficient Algorithm
  - Look for perfect match in first part of read
  - Try insertion point at point of first mismatch
  - More complicated but faster
- More Accurate Algorithm
  - Perform alignment between read and reference
- Extremely Accurate Algorithm
  - Align all reads with indel together.
  - Multiple Sequence Alignment

Other Challenges

- **Coverage of sequence reads is not uniform**: Some places we have many reads, while some we have fewer. How do we design an approach so we can always recover the sequence?
- **Large memory requirements**: We need to fit our index into RAM. Often tens of Gigabytes or greater.

Read Mapping Project



- Project predicts mutations using donor reads and reference genome
- Evaluated using true mutations

Mutations Format

- ">" then reference position
- "S" - Substitution (original new)

- "I" - Insertion (new)
- "D" - Deletion (deleted original)
- Also format for prediction

For example,

```
>S125 A A
>S369 C T
>S625 C C
>S630 G A
>S812 T A
>S841 T T
>S845 G C
>S880 A T
>S937 A A
>I447 G
>D633 G
```

Reference / Read Format

These are stored in FASTA files.

- Reference:

```
>genome_1000
TCCCTACACTTGGCGATTGAACGGAGACACTGTTCATGCCACCCCGTGCCCTAGCCTGCTTTACCTTGCTGGCGCCCCCT
CACTTAAATTATAATCTTAGCCCCTTCTCCTCTCCCAGTCGTGTCAGCGTTTTCGGTGAGGACCCGGGGTAAGTTCACGT
GCGTTGTCTAACTTGAAACAACTTTCTTTCTTGCCGCATCCGTACTCATTCGCAGTCGCTGTGTCTCTAACCCAACTTCC
TAAGTGCTTGCAGCTAAATCTGAACAGCATTGCCTATTTCTCAGTTAATCTAGCAGTTTAGGTAAGTTAGTACCACTTCC
```

- Reads:

```
>read_0
GGTAAGTTAGTACCACTTCCAATAACAAGCTGATAGACATGGACTTGAAC
>read_1
CAAGATTCGGTATCTTACAAACCTTTCCCGAATTGTTCGATTTGGGACAC
>read_2
CGAAGTTCTTGCCGCGCATTTGCGAGAAGCAGATAGAGCGACTCCTGGCT
>read_3
GAATCTGTCACTCCGATCGATAGCTTATTGCCCAACACGATCCCGGTCGG
```

Paired reads

- 2 reads separated by "insert"
- "insert" size is drawn from a distribution
- Paired reads:

```
>read_0/1
TTCCTAAGTGCTTGCAGCTAAATCTGAACAGCATTGCCTATTTCTCAGTT
>read_0/2
ATAGACATGGACTTGAACGATTTTAGACAGATATACCTCGATGTAAGGGA
```

```
>read_1/1
TTCCCAGCTGCAAACGATTTGATTCGCTGTCAGGTTACGTCAACGCGGGA
>read_1/2
GCGGCCGTAATTGTCATAGCAACGTATGTTTGCCGGCAGTCGTAATCTCT
```

Simulator Details

- Mutation Process
  - Substitution Rate
  - Insert Rate
  - Deletion Rate
- Read sampled uniformly from genome
- Insert sizes drawn uniformly
  - Minimum Insert
  - Maximum Insert
- Error rates for reads
  - Substitution Rate
  - Insert Rate
  - Deletion Rate

Read Mapping Sample Genome

- Small genome is available with correct answers
- Reads are available with and without errors:
  - Paired no errors
  - Paired with errors

Assignment Details

- Starter - 10,000 length genome
- Assignment:
  - 1,000,000 length genome for undergrad
  - 100,000,000 length genome for grad (or extra credit)
- Scoreboard will evaluate F-score for substitutions, insertions, deletions

Sequence Mapping Coverage

- If a genome is length N (human is 3,000,000,000), and the total length of all sequence reads collected is M, the coverage ratio is defined at M / N.
- Often written with an "x". For example, 10x or 20x coverage.
- Number of reads spanning a specific location follows a distribution with a mean of the coverage.
- Most positions close to coverage.

Sequencing Coverage Statistics

- If length of the genome is N the probability of the event that a single read position starts at a single position in the genome is 1/N (very small).
- If the number of reads is K, the total number of read positions that start at a single genome position is the number of times that an event with probability 1/N happens out of K trials.
- Poisson distribution.