Alignment of whole genomes using MUMmer

Presentation in Algorithms in Bioinformatics (TÖ111F)

Hannes Pétur Eggertsson

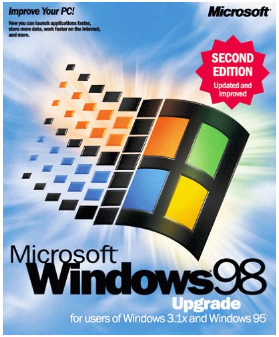
November 18, 2014

Motivation

First off. Let's travel back in time to 1999... (here are some things that will remind you of that wonderful time)

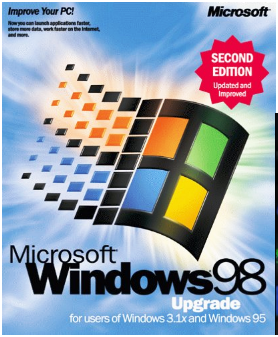
Motivation

First off. Let's travel back in time to 1999... (here are some things that will remind you of that wonderful time)


Improve Your PC!
Now you can launch applications faster, store more data, work faster on the Internet, and more.
Microsoft
SECOND EDITION
Updated and Improved
Microsoft Windows98
Upgrade
for users of Windows 3.1x and Windows 95

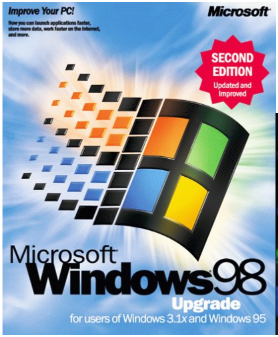
Motivation

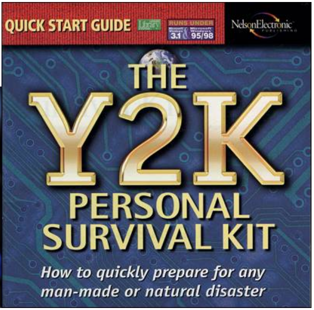
First off. Let's travel back in time to 1999... (here are some things that will remind you of that wonderful time)


Improve Your PC!
Now you can launch applications faster, store more data, work faster on the Internet, and more.
Microsoft
SECOND EDITION
Updated and Improved
Microsoft Windows98
Upgrade
for users of Windows 3.1x and Windows 95

Motivation

First off. Let's travel back in time to 1999... (here are some things that will remind you of that wonderful time)




QUICK START GUIDE
NelsonElectronic
THE Y2K PERSONAL SURVIVAL KIT
How to quickly prepare for any man-made or natural disaster

Motivation

Meanwhile at that particular time in bioinformatics...

- The number of sequenced genomes was very low but increasing very fast.
- Whenever a new genome is sequenced, one could ask himself:
  - How does this genome align to the other genomes we have sequenced?

Motivation

Meanwhile at that particular time in bioinformatics...

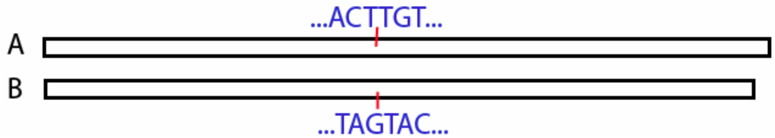
- The number of sequenced genomes was very low but increasing very fast.
- Whenever a new genome is sequenced, one could ask himself:
  - How does this genome align to the other genomes we have sequenced?

Problem:

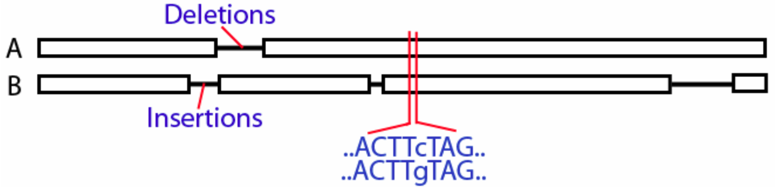
- We had algorithms that were made for single gene sequences.
- They didn't work well with whole genomes.
- It's simple: Size matters.
  - Require way too much memory or
  - take extremely long time to compute.

Problem description

In Two genomes, A and B. Both could be very large (millions of nucleotides)



Out Align the two genomes to maximize the number of matches using insertions and deletions.



Introduction to MUMmer

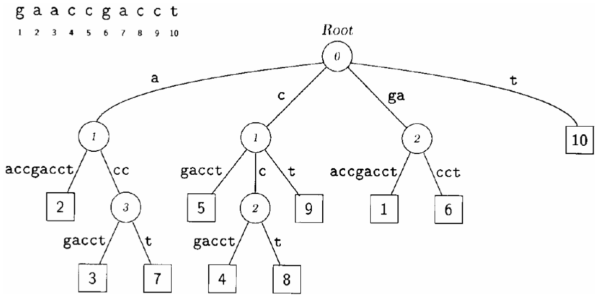
MUMmer:

- Was published in 1999.
- Is a system used to align whole genome sequences.
- Idea: Find large chunks of exact matches on both genomes in linear time.
- Doesn't guarantee the optimal solution, just a good one.

Step 1: Creating a suffix tree

A suffix tree stores all possible suffixes in a tree.

- Square nodes are leaves.
  - Store information about the starting position of the suffix.
- Circular nodes are internal nodes.
  - Store information about the length of the shared prefix.



Creating a suffix tree takes $O(n)$ time and space.

Step 2: MUM decomposition

MUM is a abbreviation for Maximal Unique Matches.

Definition

A sequence is a MUM if and only if:

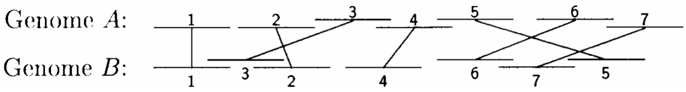
- The sequence has an exact match on both genomes
- and it is not a subsequence of another matched sequence
- and it is unique

Example

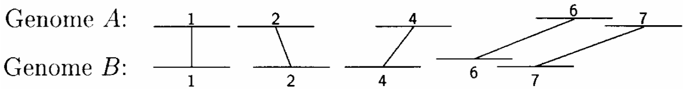
```
tcgatcGACGATCGCGGCCGTAGATCGAATAACGAGAGAGCATAAcgactta
gcattaGACGATCGCGGCCGTAGATCGAATAACGAGAGAGCATAAtccagag
```

Step 3: Sorting the matches found in the MUM alignment

We enumerate the found MUMs like this:

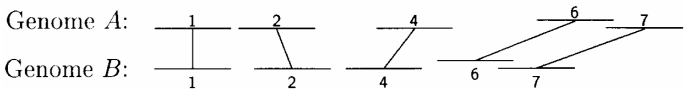


Problem: We cannot align all MUMs because they aren't in the same order in both genomes.
Solution: Align as many MUMs as we can:



Step 4: Closing the gaps

Everything in between the MUMs is called **gaps**.



- To find alignment for the gaps we can use any alignment algorithm.
- MUMmer uses the Smith-Waterman algorithm.

Diving deeper

How do we go from suffix trees to finding MUMs?

Example

In Two genomes

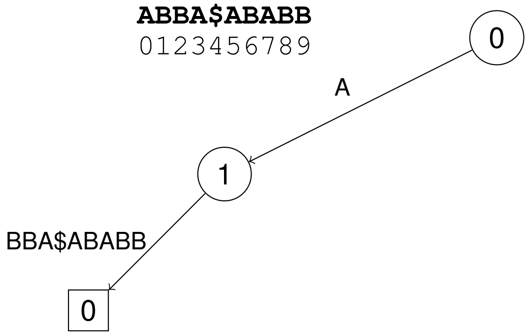
Genome A: `ABBA`

Genome B: `ABABB`

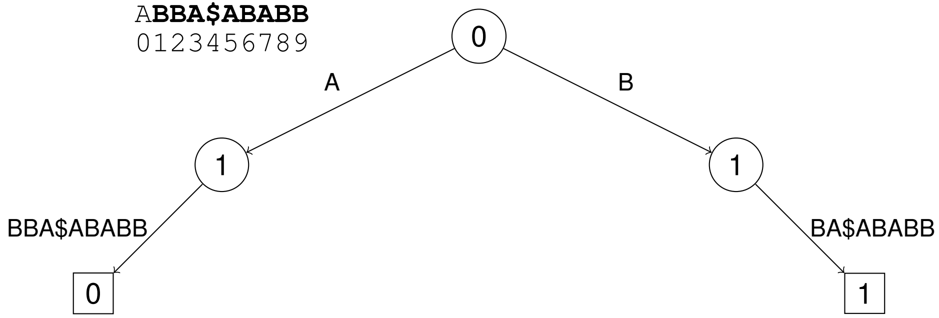
Out List of all MUMs.

Combined string: `ABBA$ABABB`

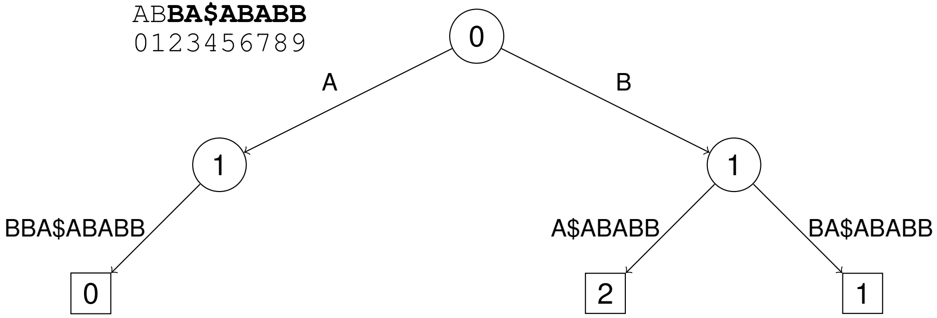
Creating a suffix tree



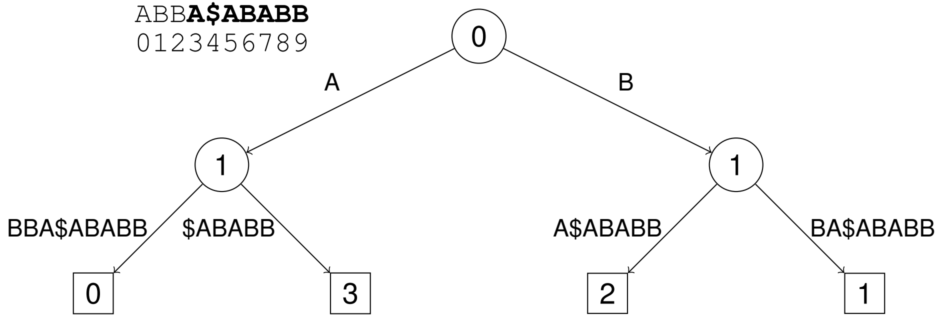
Creating a suffix tree



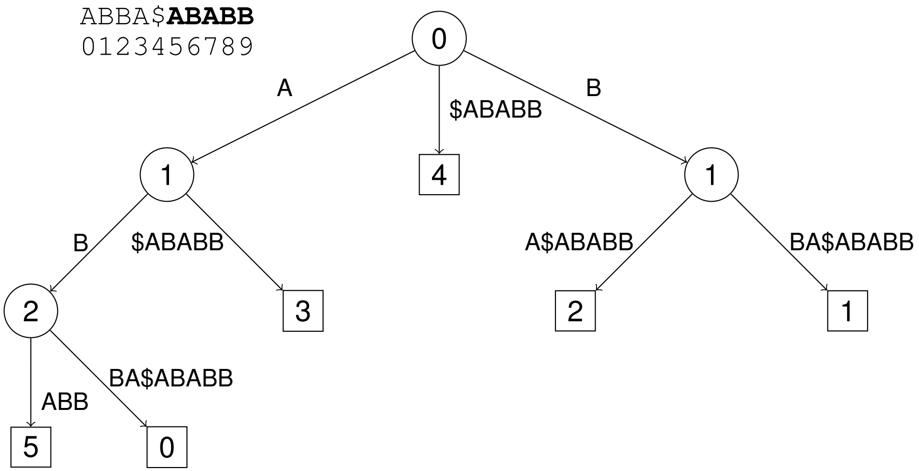
Creating a suffix tree



Creating a suffix tree

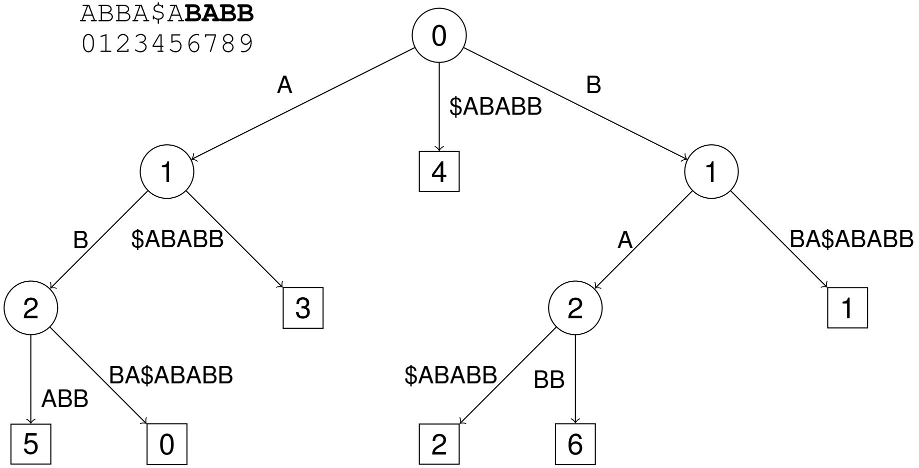


Creating a suffix tree

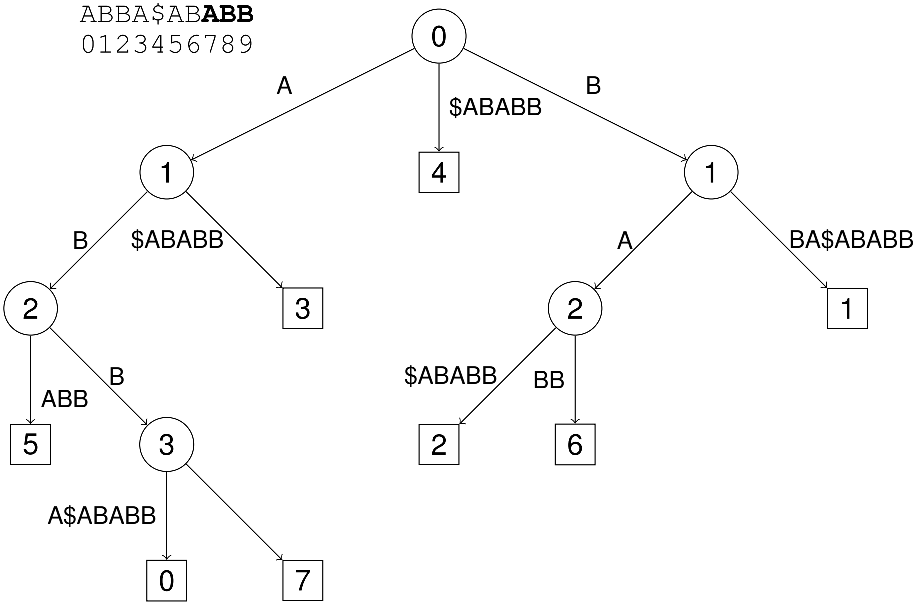


Creating a suffix tree

ABBA$A**BABB**
0123456789

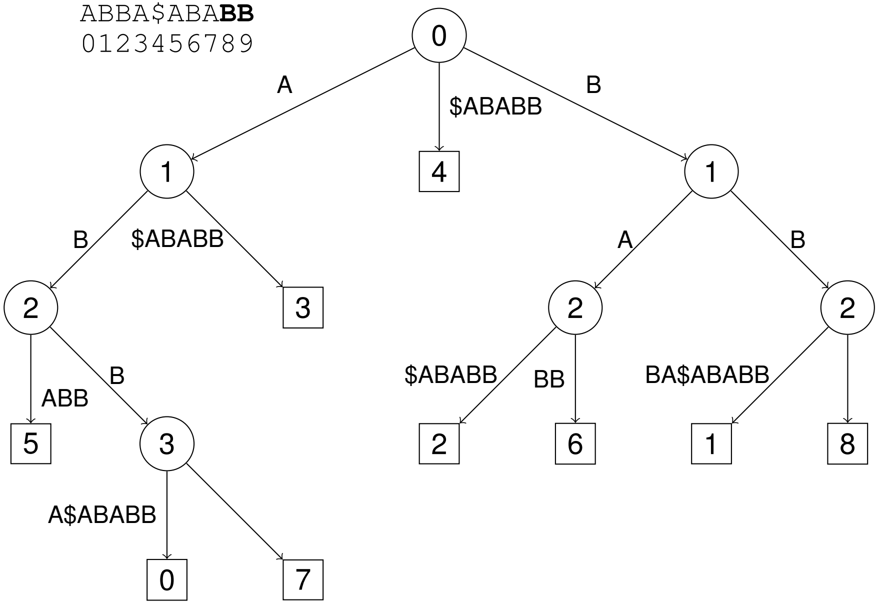


Creating a suffix tree

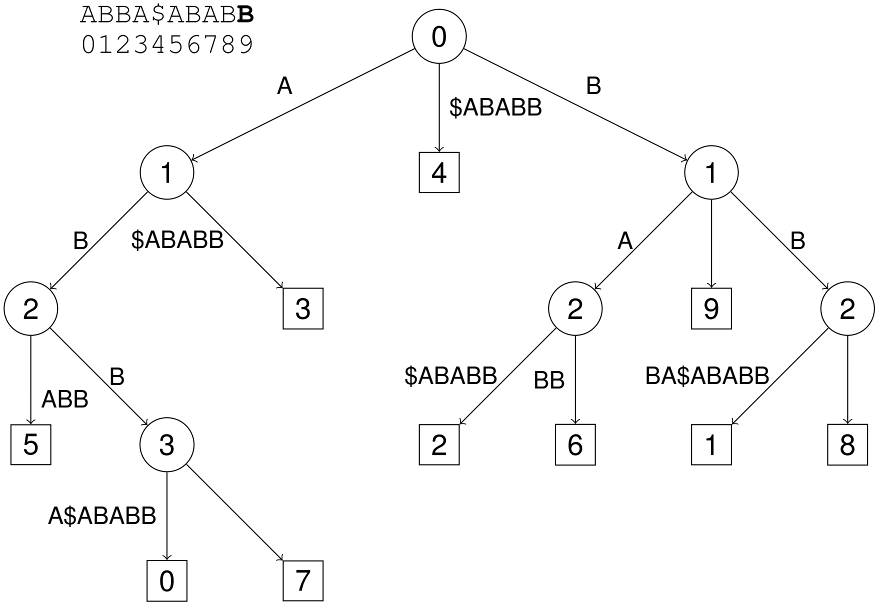


Creating a suffix tree

ABBA$ABA**BB**
0123456789



Creating a suffix tree



Finding MUMs from suffix tree

Let's recall what condition MUMs had to have:

1. Exact matches on both genomes.
2. Surrounded by mismatches.
3. Unique.

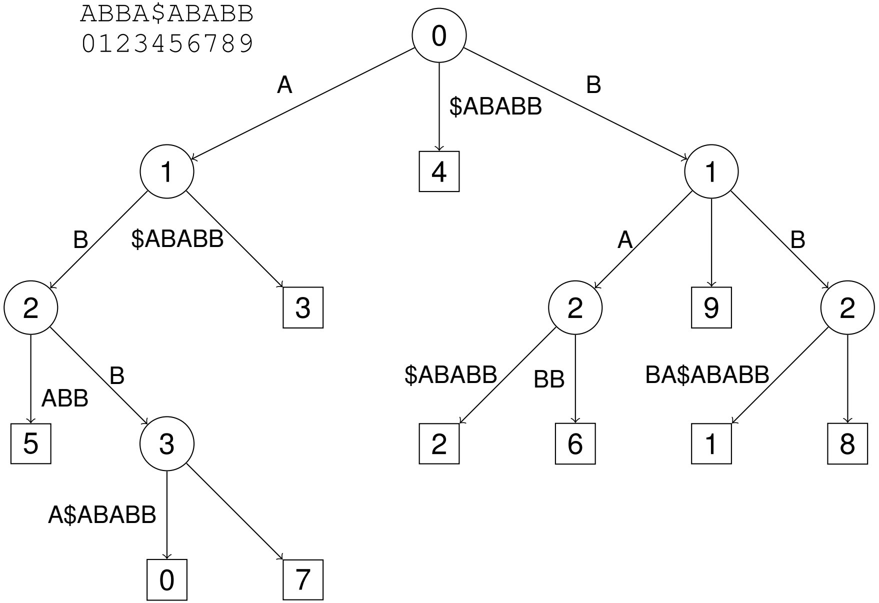
We can satisfy conditions 1 and 3 by searching the tree for a internal node with:

- Exactly two leafs.
- Both leaves' starting positions are on each side of the dollar sign.

We'll need to check for condition 2 afterwards.

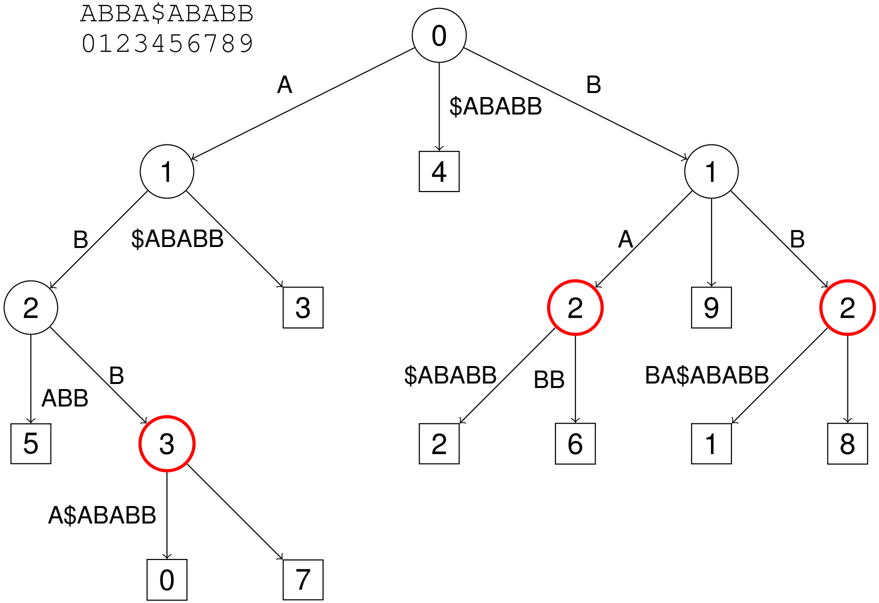
Finding potential MUMs

```
ABBA$ABABB
0123456789
```



Finding potential MUMs

```
ABBA$ABABB
0123456789
```



Potential MUMs to MUMs

We have the following potential MUMs:

- `ABB`.
- `BA`.
- `BB`.

Potential MUMs to MUMs

We have the following potential MUMs:

- `ABB`.
- `BA`.
- `BB`.

Our string: `ABBA$ABABB`. Let's check if condition 2 is satisfied

- `ABB` satisfies condition 2.
- `BA` satisfies condition 2.
- `BB` does NOT satisfy condition 2.

Resulting MUMs: **`ABB`** and **`BA`**.

Results and conclusion

- Two strains of tuberculosis that are >99% identical
  - 5 seconds to create the suffix tree.
  - 45 seconds to sort the MUMs.
  - 5 seconds to generate alignments of the gaps.

Results and conclusion

- Two strains of tuberculosis that are >99% identical
  - 5 seconds to create the suffix tree.
  - 45 seconds to sort the MUMs.
  - 5 seconds to generate alignments of the gaps.
- Two 'cousin' genomes. Genome of *M.genitalium* (580,074 nucleotides) and *M.pneumoniae* (816,394 nucleotides)
  - 6.5 seconds to create the suffix tree.
  - 0.02 seconds to sort the MUMs.
  - 116 seconds to generate alignments of the gaps.

Results and conclusion

- Two strains of tuberculosis that are >99% identical
  - 5 seconds to create the suffix tree.
  - 45 seconds to sort the MUMs.
  - 5 seconds to generate alignments of the gaps.
- Two 'cousin' genomes. Genome of *M.genitalium* (580,074 nucleotides) and *M.pneumoniae* (816,394 nucleotides)
  - 6.5 seconds to create the suffix tree.
  - 0.02 seconds to sort the MUMs.
  - 116 seconds to generate alignments of the gaps.

So in conclusion:

- If the two genomes are very similar, MUM sequences will...
  - ...be long and cover most of the genome.
  - ...rarely be a random match.
  - ...make the algorithm run fast.

Results and conclusion

- Two strains of tuberculosis that are >99% identical
  - 5 seconds to create the suffix tree.
  - 45 seconds to sort the MUMs.
  - 5 seconds to generate alignments of the gaps.
- Two 'cousin' genomes. Genome of *M.genitalium* (580,074 nucleotides) and *M.pneumoniae* (816,394 nucleotides)
  - 6.5 seconds to create the suffix tree.
  - 0.02 seconds to sort the MUMs.
  - 116 seconds to generate alignments of the gaps.

So in conclusion:

- If the two genomes are very similar, MUM sequences will...
  - ...be long and cover most of the genome.
  - ...rarely be a random match.
  - ...make the algorithm run fast.
- If the genomes are very different, MUM sequences will...
  - ...be short and cover a small part of the genome.
  - ...often be a random match.
  - ...make the algorithm run slow.

MUMmer 3

- MUMmer 3 is the latest version of MUMmer
- It was released in 2004 and is open-source.
- Requires less memory and is a lot faster than the initial MUMmer.
- Most notable features added since the initial MUMmer:
  - You can allow tolarence for mismatches when finding MUMs.
  - Can handle non-unique MUMs.
  - All sorts of visualization tools.

Thank you!

Feel free to ask any questions.

Original papers:
`http://mummer.sourceforge.net/MUMmer.pdf` by Arthur L. Delcher, Simon Kasif, Robert D. Fleischmann, Jeremy Peterson, Owen White and Steven L. Salzberg.

and

`http://mummer.sourceforge.net/MUMmer3.pdf` by Stefan Kurtz, Adam Phillippy, Arthur L. Delcher, Michael Smoot, Martin Shumway, Corina Antonescu and Steven L. Salzberg.