Linear Programming for Phylogenetic Reconstruction Based on Gene Rearrangements

Jijun Tang
jtang@cse.sc.edu

Department of Computer Science and Engineering
University of South Carolina

Acknowledgment

- **Joint work with Bernard Moret** (University of New Mexico).
- **Supported by National Science Foundation and U. of South Carolina.**

Overview

- Introduction to gene-order data
- GRAPPA and the computational challenge
- Linear programming setup
- Experimental design
- Experimental results
- Conclusions

What Is A Phylogeny?

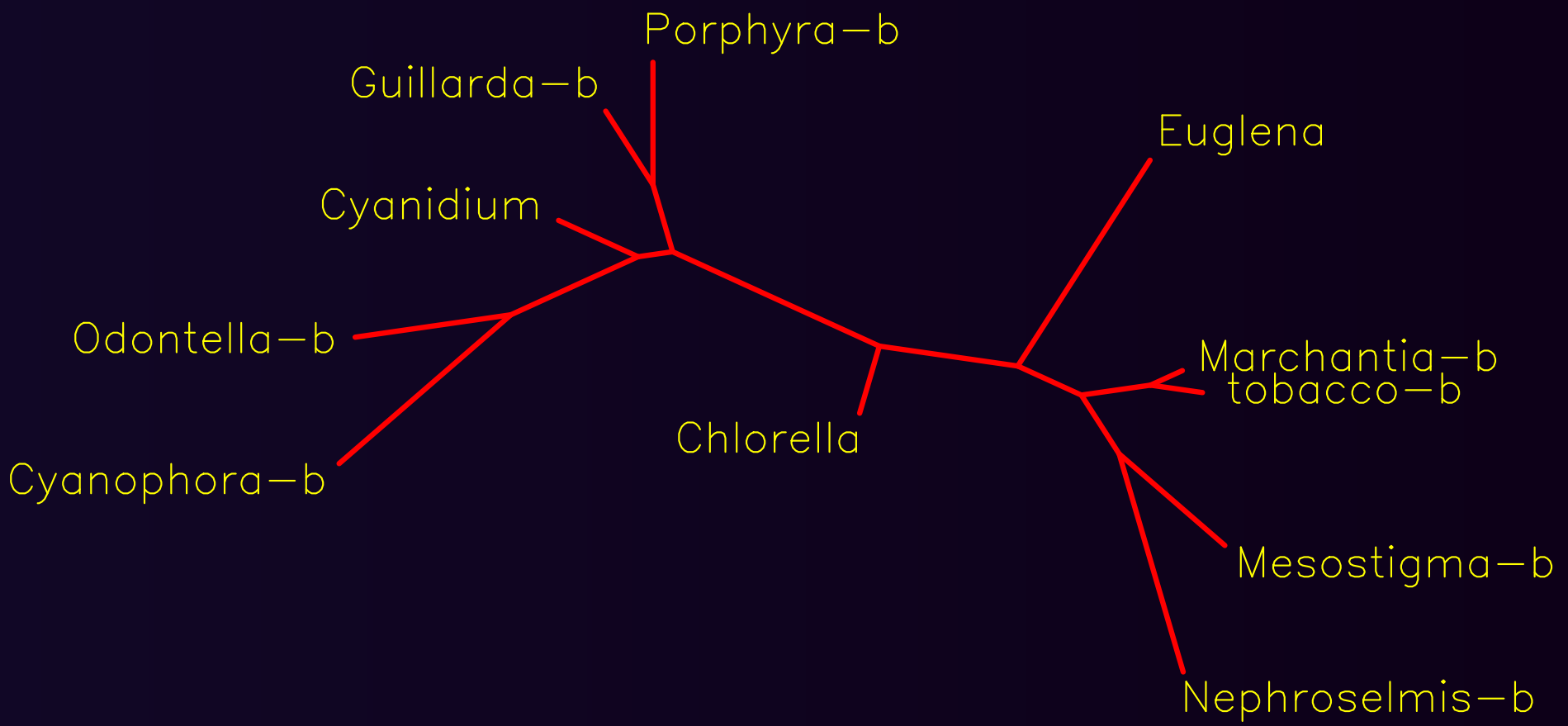
What Is A Phylogeny?

- **The evolutionary history of a group of organisms**

What Is A Phylogeny?

- **The evolutionary history of a group of organisms**
- **Usually takes the form of a tree:**
  - Modern organisms are placed at the leaves
  - Edges denote evolutionary relationships

Example


Porphyra−b
Guillarda−b
Euglena
Cyanidium
Odontella−b
Marchantia−b
tobacco−b
Chlorella
Cyanophora−b
Mesostigma−b
Nephroselmis−b

Gene-Order Data

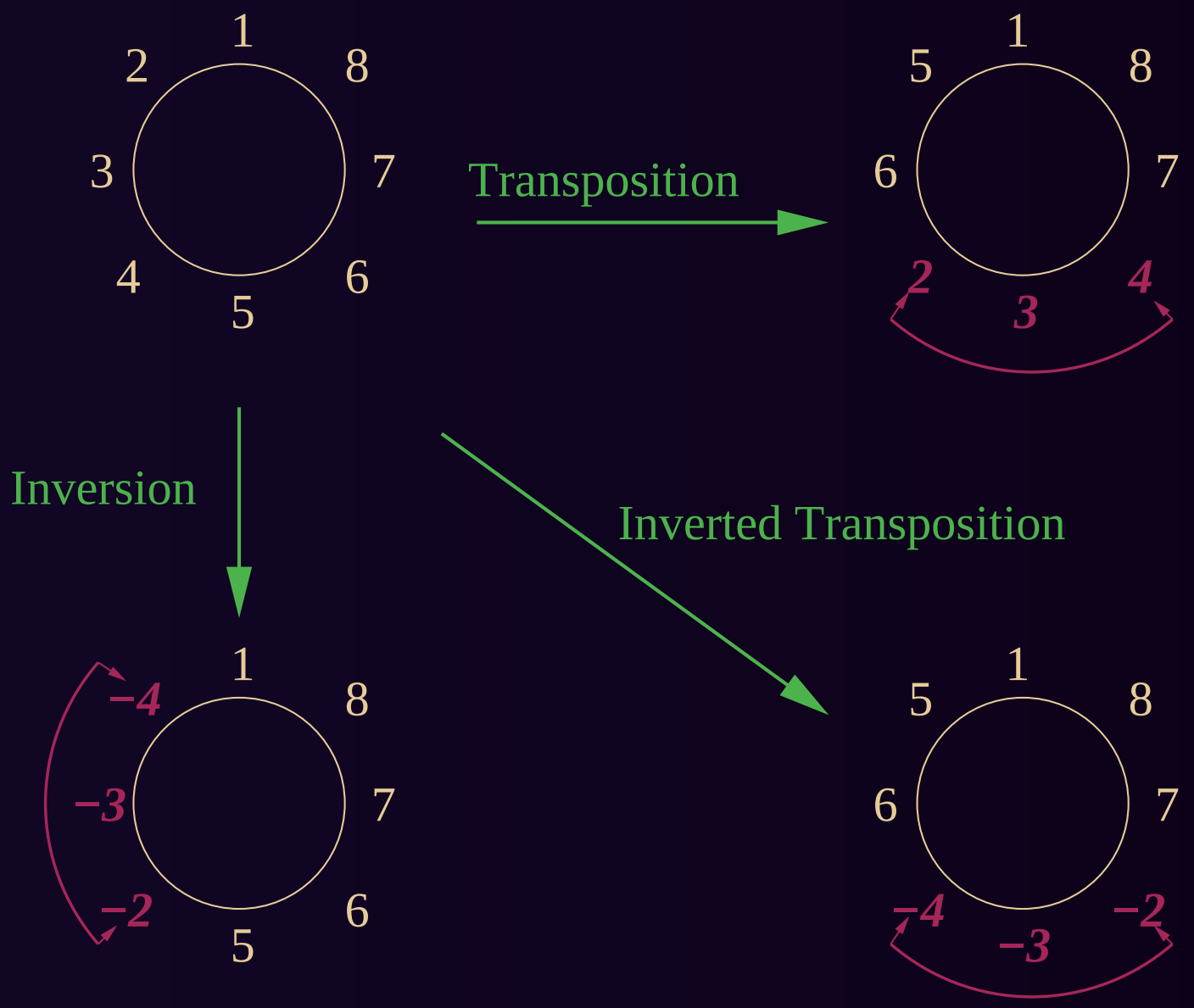
Gene-Order Data

- **Chromosome can be represented by an ordering of signed genes**
  - Linear or circular
  - Sign of a gene represents gene orientation

Gene-Order Data

- **Chromosome can be represented by an ordering of signed genes**
  - Linear or circular
  - Sign of a gene represents gene orientation
- **The gene order can be rearranged by evolutionary events such as:**
  - Inversion, transposition and inverted transposition
  - Deletion and insertion

Gene-Order Rearrangements



Reconstruction Methods

Reconstruction Methods

- **Distance based methods:**
  Neighbor-joining and its variants

Reconstruction Methods

- **Distance based methods:**
  Neighbor-joining and its variants
- **Bayesian method:**
  Badger

Reconstruction Methods

- **Distance based methods:**
  Neighbor-joining and its variants
- **Bayesian method:**
  Badger
- **Maximum parsimony based on encoding:**
  `MPBE`, `MPME`

Reconstruction Methods

- **Distance based methods:**
  Neighbor-joining and its variants
- **Bayesian method:**
  Badger
- **Maximum parsimony based on encoding:**
  `MPBE`, `MPME`
- **Direct optimization method:**
  `BPAnalysis`, `GRAPPA`, `MGR`

Direct Optimization Methods

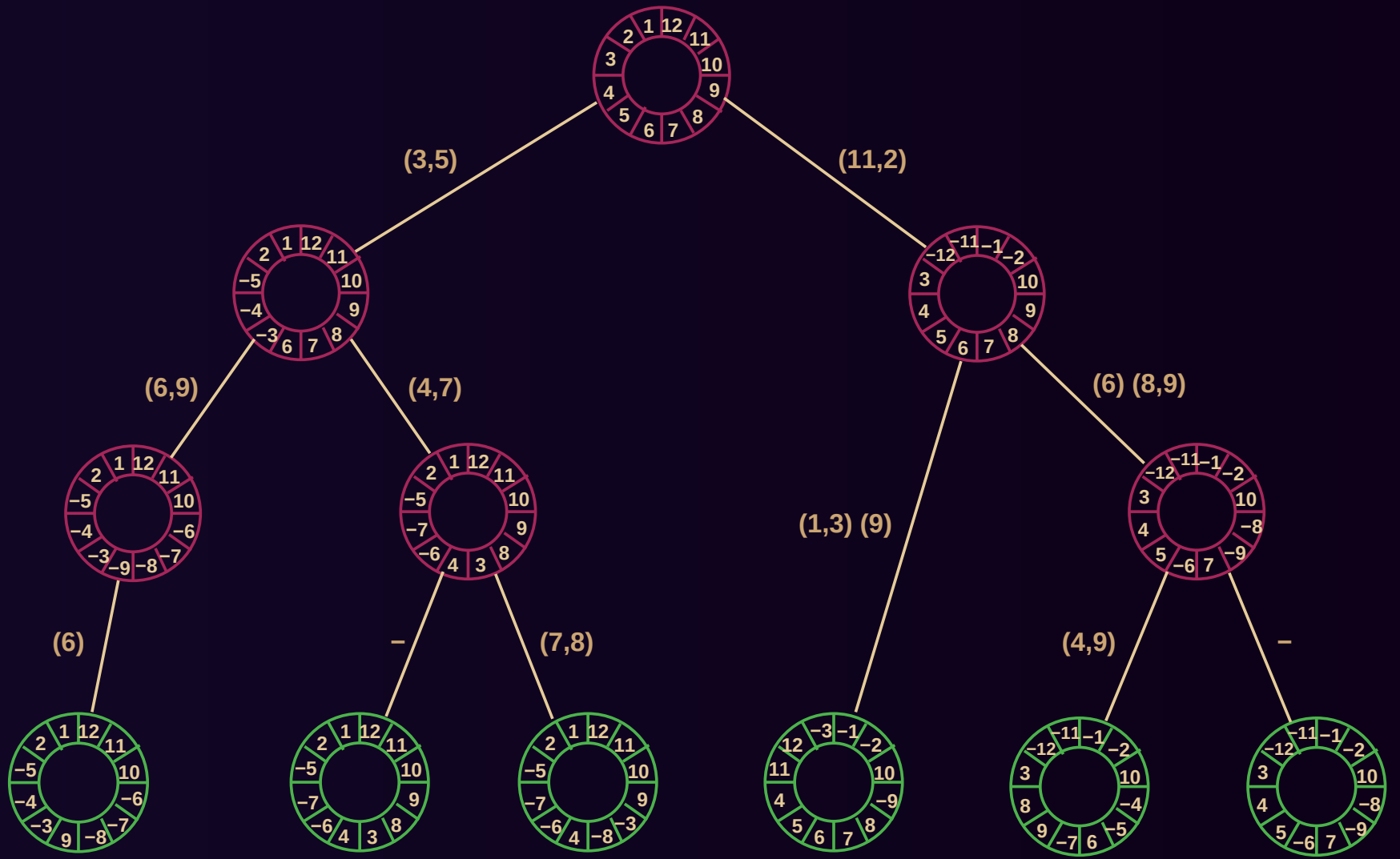
Direct Optimization Methods

- **Goal: to reconstruct phylogeny with *minimum* # of rearrangement events**

Direct Optimization Methods

- **Goal: to reconstruct phylogeny with *minimum* # of rearrangement events**
- **Computationally hard even for only three genomes**
  - Median problem for three is NP hard under general distance definition
  - Find the content of the median genome to minimize the sum of the distances from the median to the three genomes

Reconstruction Example



GRAPPA

GRAPPA

- **Genome Rearrangements Analysis under Parsimony and other Phylogenetic Algorithms**

GRAPPA

- **Genome Rearrangements Analysis under Parsimony and other Phylogenetic Algorithms**
- **Started as an effort to reimplement the BPAnalysis of Sankoff and Blanchette**

GRAPPA

- **Genome Rearrangements Analysis under Parsimony and other Phylogenetic Algorithms**
- **Started as an effort to reimplement the BPAnalysis of Sankoff and Blanchette**
- **Used algorithmic techniques to improve the speed**
  - A tightened lower bound to discard bad trees before scoring them
  - Profiling, cache awareness, etc

Algorithm Outline

Algorithm Outline

- **Consider each tree topology in turn**

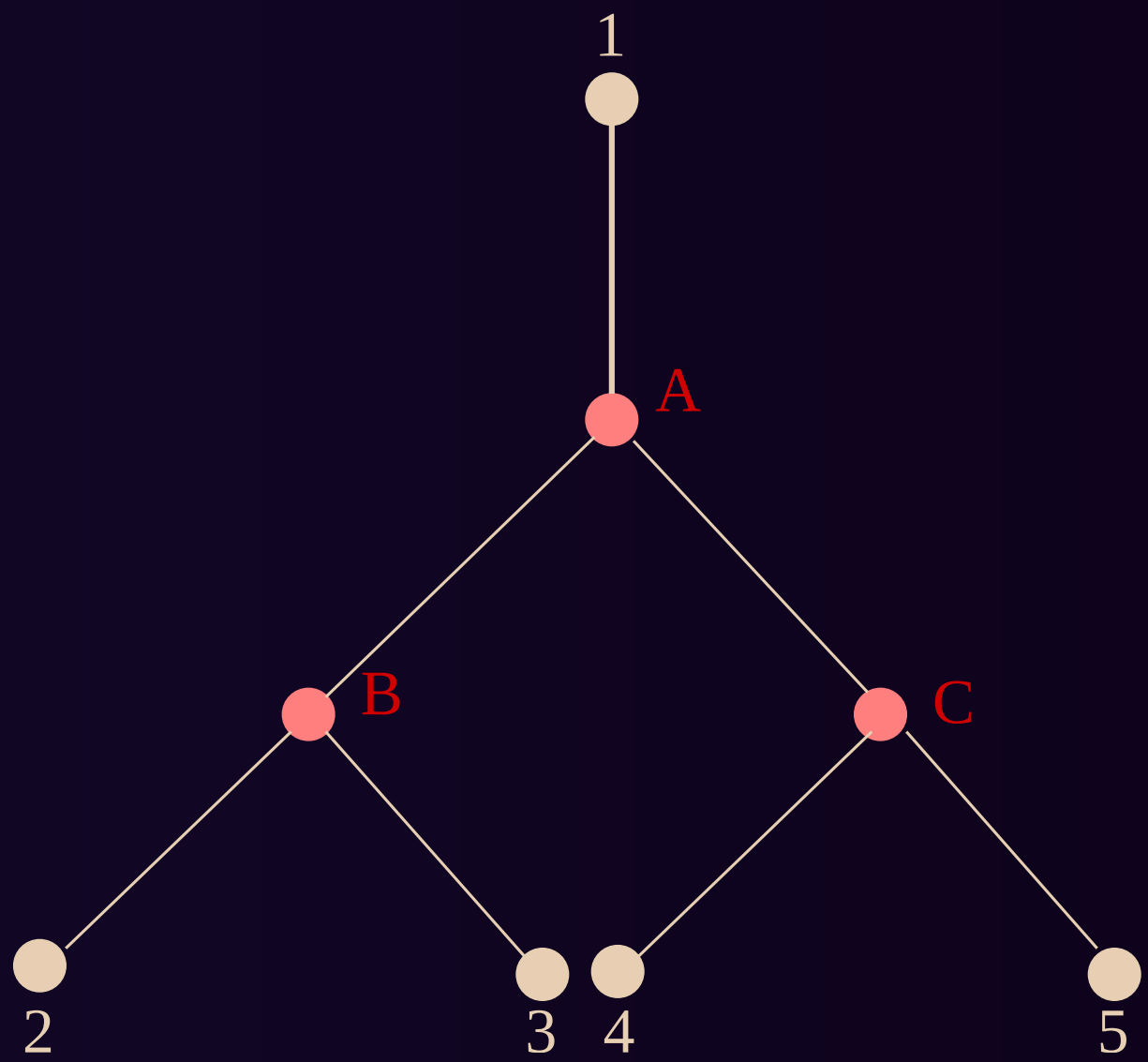
Algorithm Outline

- **Consider each tree topology in turn**
- **For each tree**
  - Test the lower bound, if it exceeds the best so far, continue to the next tree
  - Initialize the internal nodes by some means
  - Compute medians of three iteratively until no change occurs

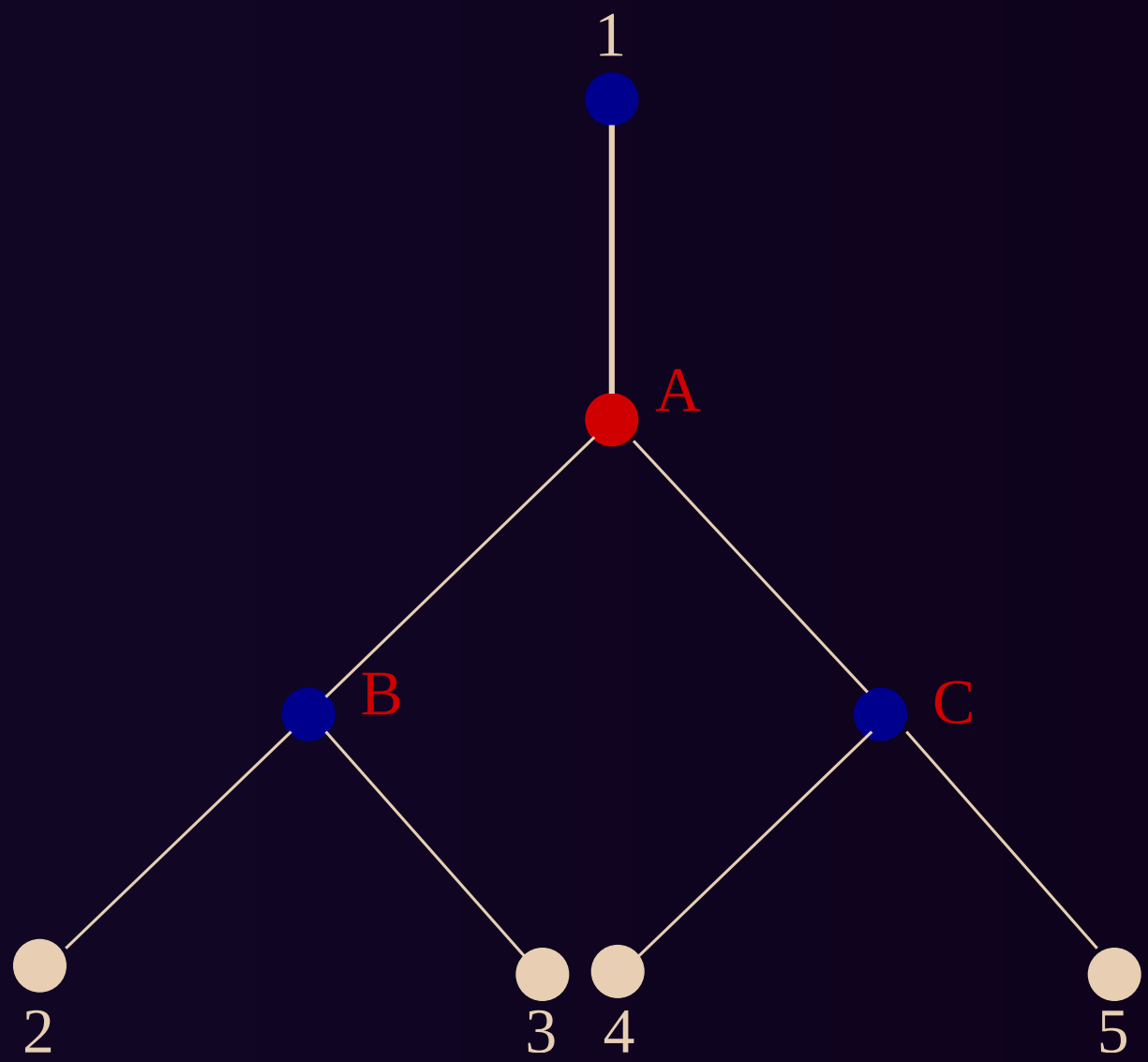
Algorithm Outline

- **Consider each tree topology in turn**
- **For each tree**
  - Test the lower bound, if it exceeds the best so far, continue to the next tree
  - Initialize the internal nodes by some means
  - Compute medians of three iteratively until no change occurs
- **Return the lowest score tree**

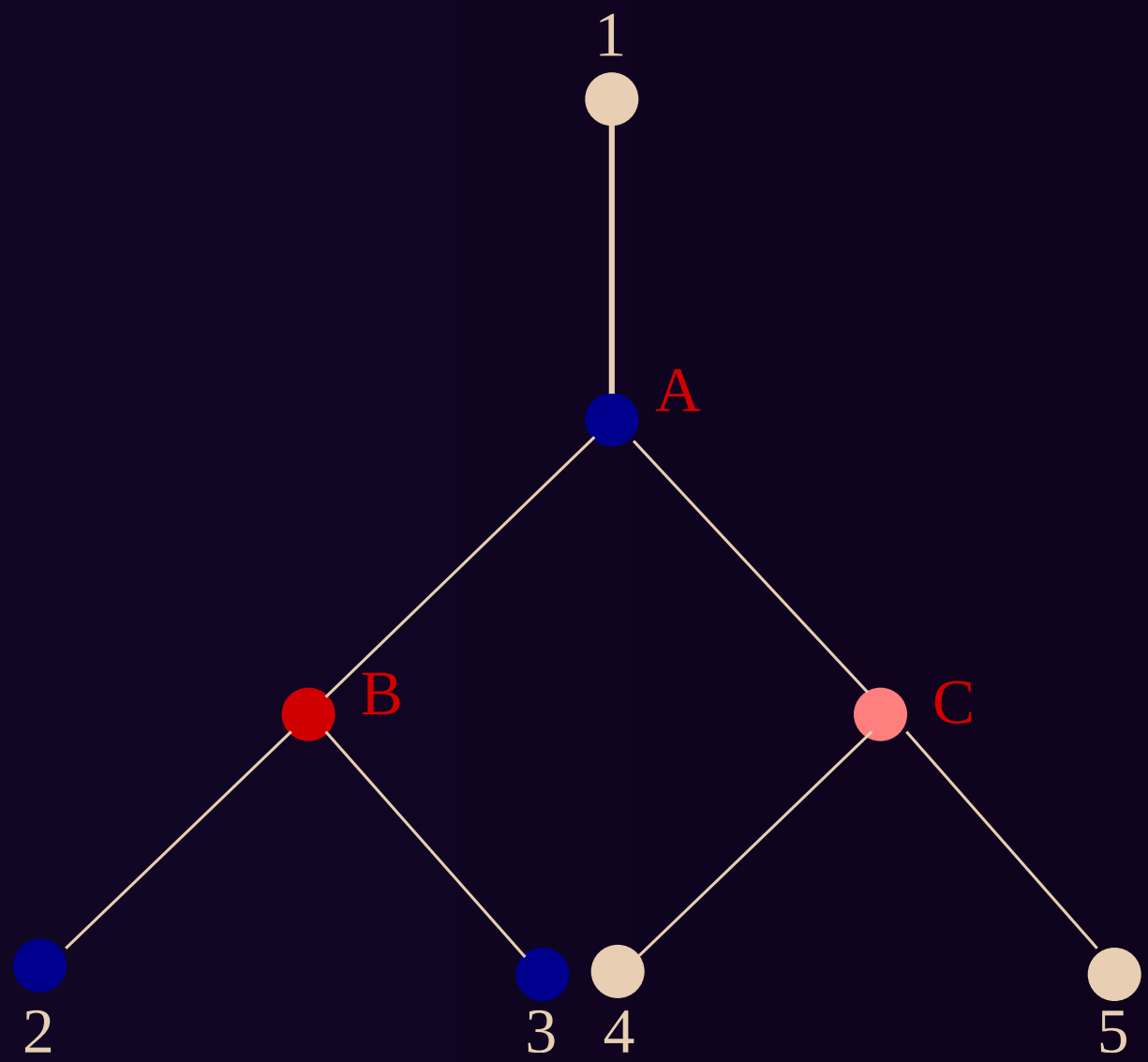
Scoring a Tree



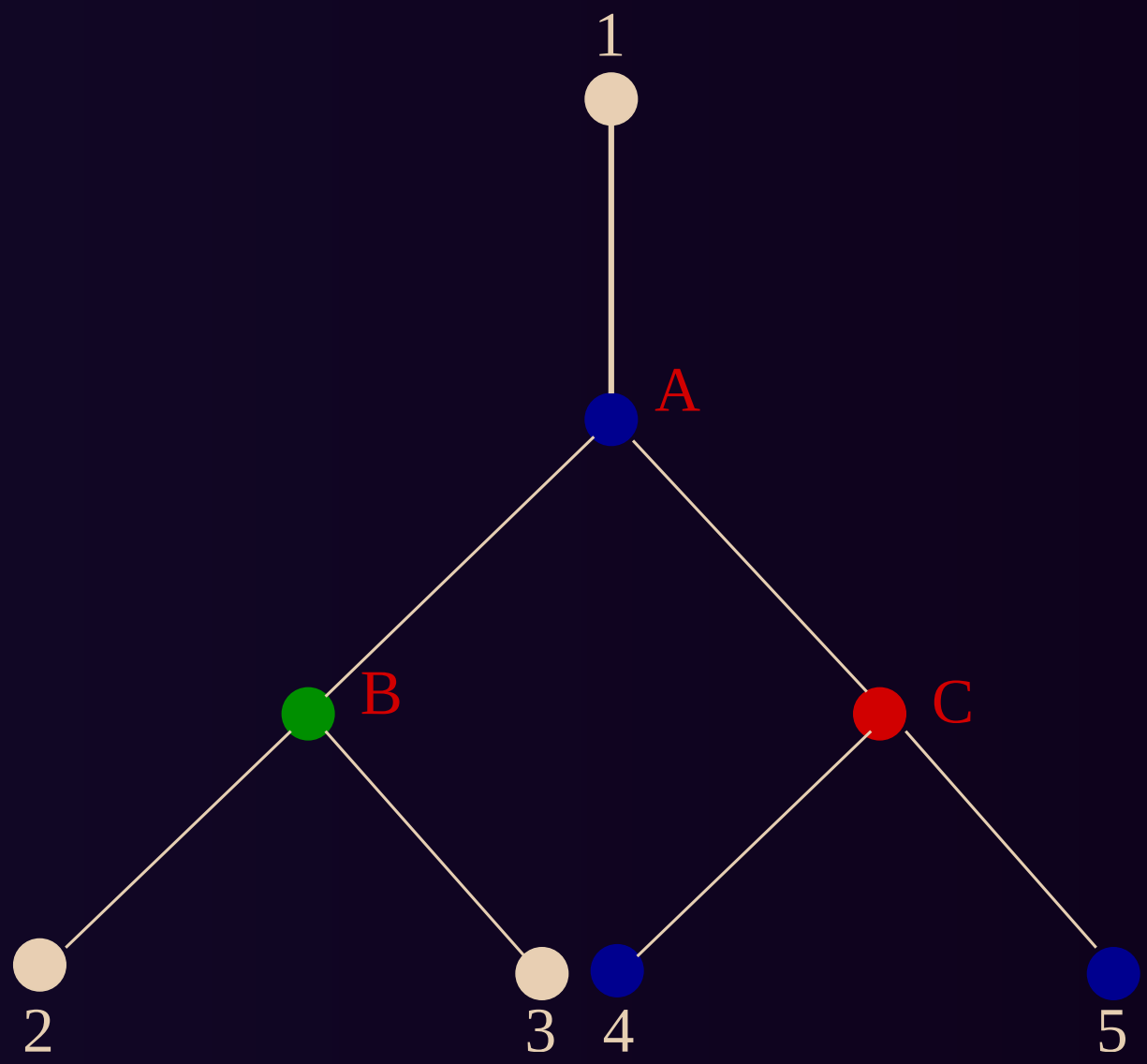
Scoring a Tree



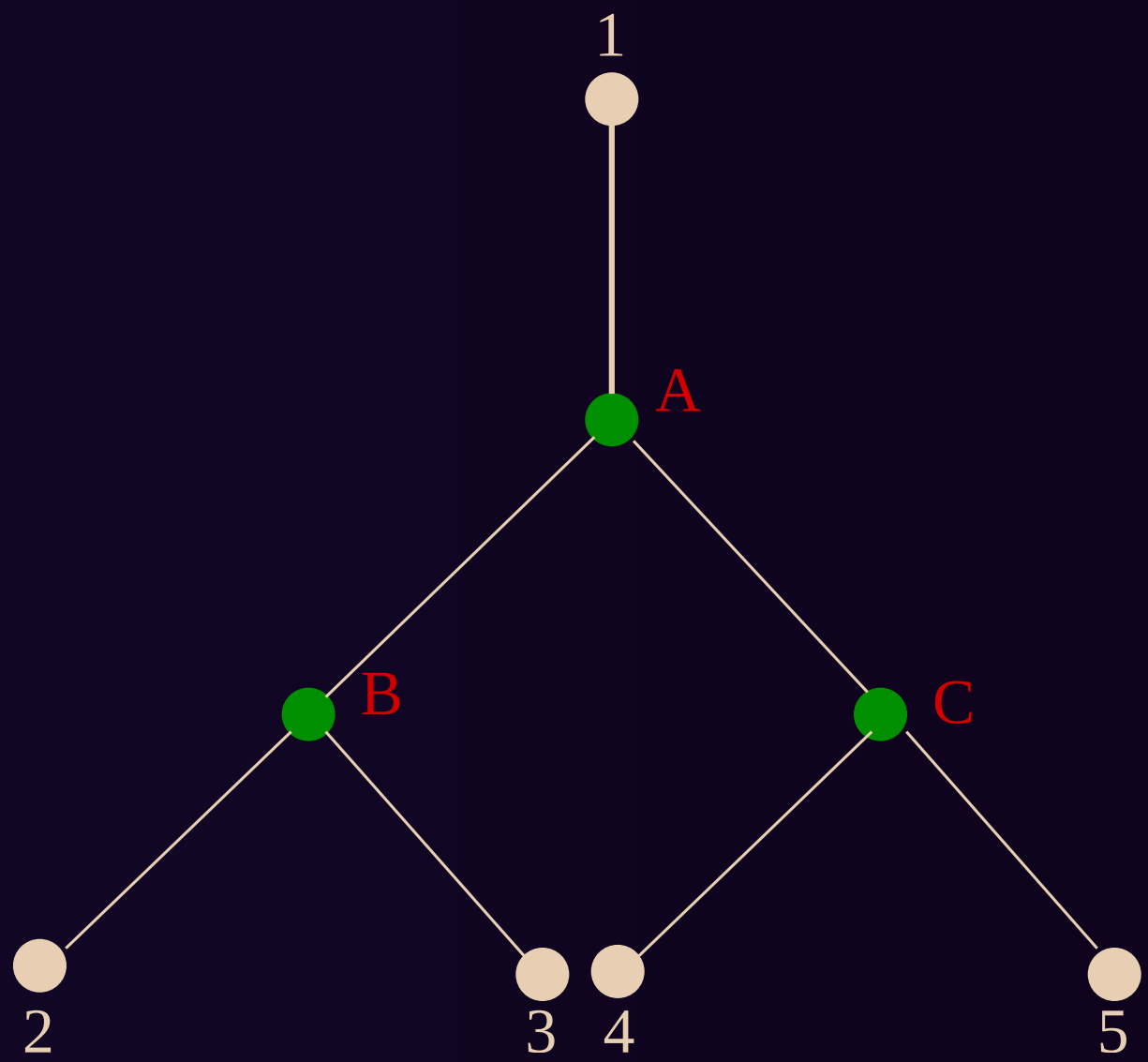
Scoring a Tree



Scoring a Tree



Scoring a Tree



Computational Challenge

Computational Challenge

- **Scoring a tree is very expensive**

Computational Challenge

- **Scoring a tree is very expensive**
- **When the genomes are distant, a median may take days or months to be solved**

Computational Challenge

- **Scoring a tree is very expensive**
- **When the genomes are distant, a median may take days or months to be solved**
- **It needs to solve the median problems iteratively**

Computational Challenge

- Scoring a tree is very expensive
- When the genomes are distant, a median may take days or months to be solved
- It needs to solve the median problems iteratively
- Can we find the tree score without solving the median problems?

Linear Programming Approach

Linear Programming Approach

- **Goal: minimize the tree length**

Linear Programming Approach

- **Goal: minimize the tree length**
- **What do we know?**

Linear Programming Approach

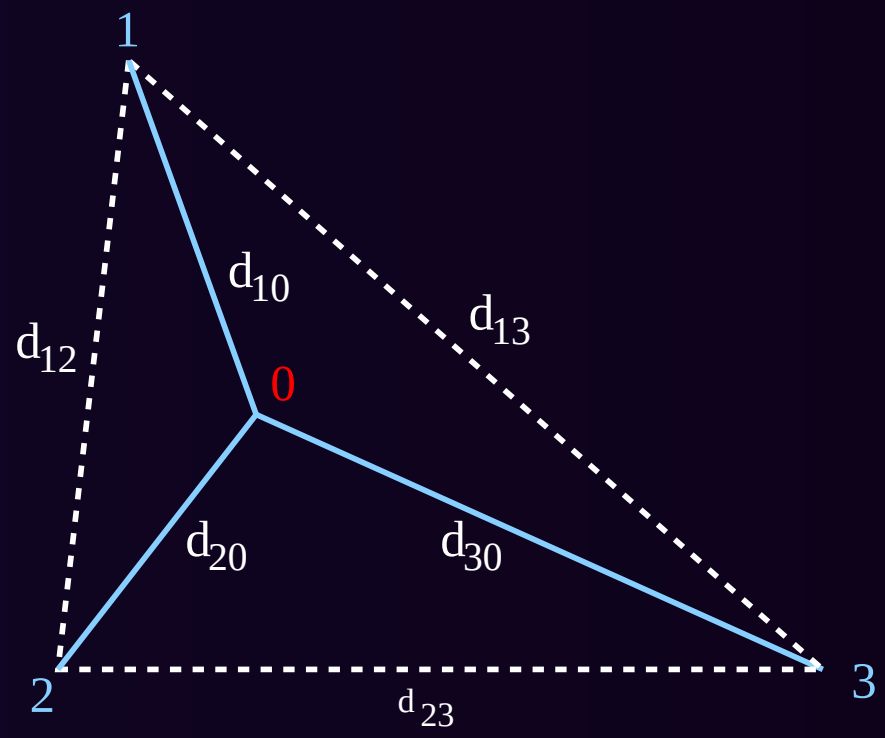
- **Goal: minimize the tree length**
- **What do we know?**
  - The pairwise distance matrix
  - A given tree topology

Linear Programming Approach

- **Goal: minimize the tree length**
- **What do we know?**
  - The pairwise distance matrix
  - A given tree topology
- **Approach:**
  - Finding useful constraints
  - Using linear programming method to minimize the tree length

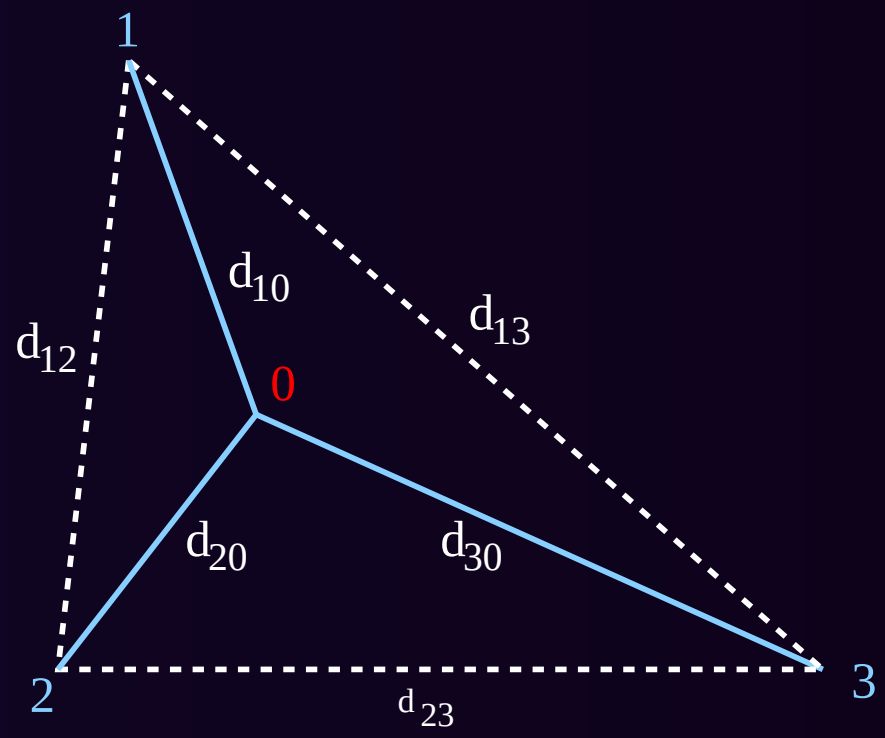
Median Problem

Median Problem



$$d_{01} + d_{02} + d_{03} \leq \frac{d_{12} + d_{23} + d_{13}}{2}$$

Median Problem

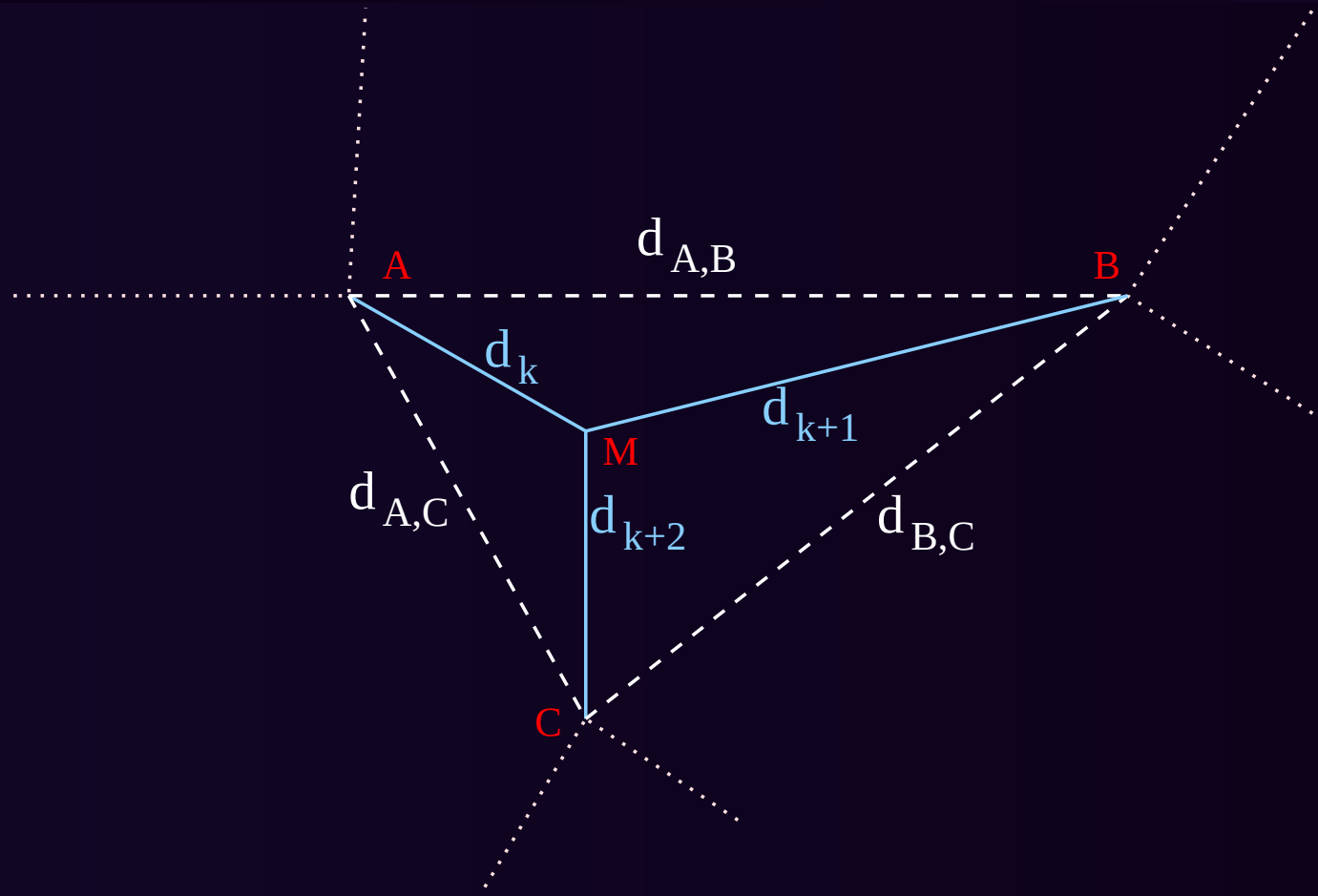


$$d_{01} + d_{02} + d_{03} \leq \frac{d_{12} + d_{23} + d_{13}}{2}$$

More than 98% cases we have

$$d_{01} + d_{02} + d_{03} = \frac{d_{12} + d_{23} + d_{13}}{2}$$

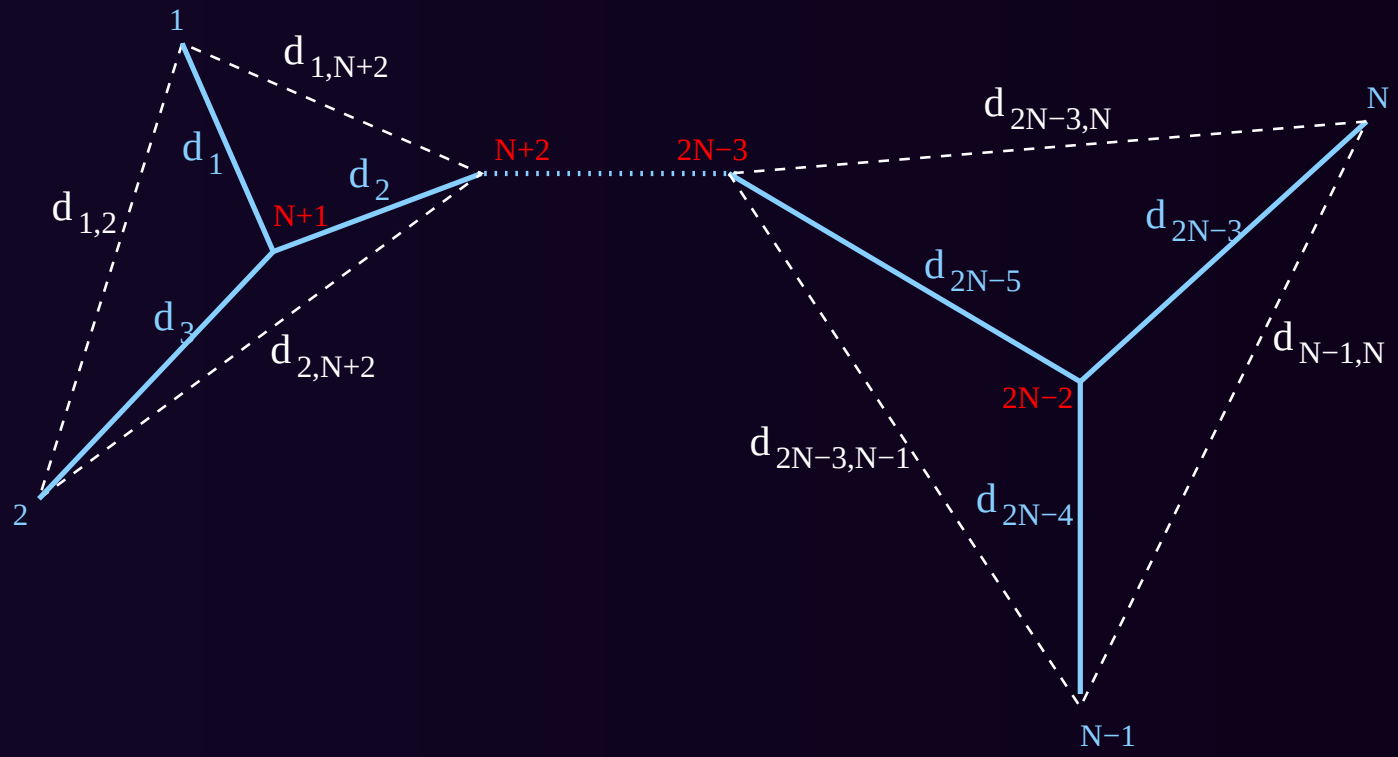
Constraint on Internal Node



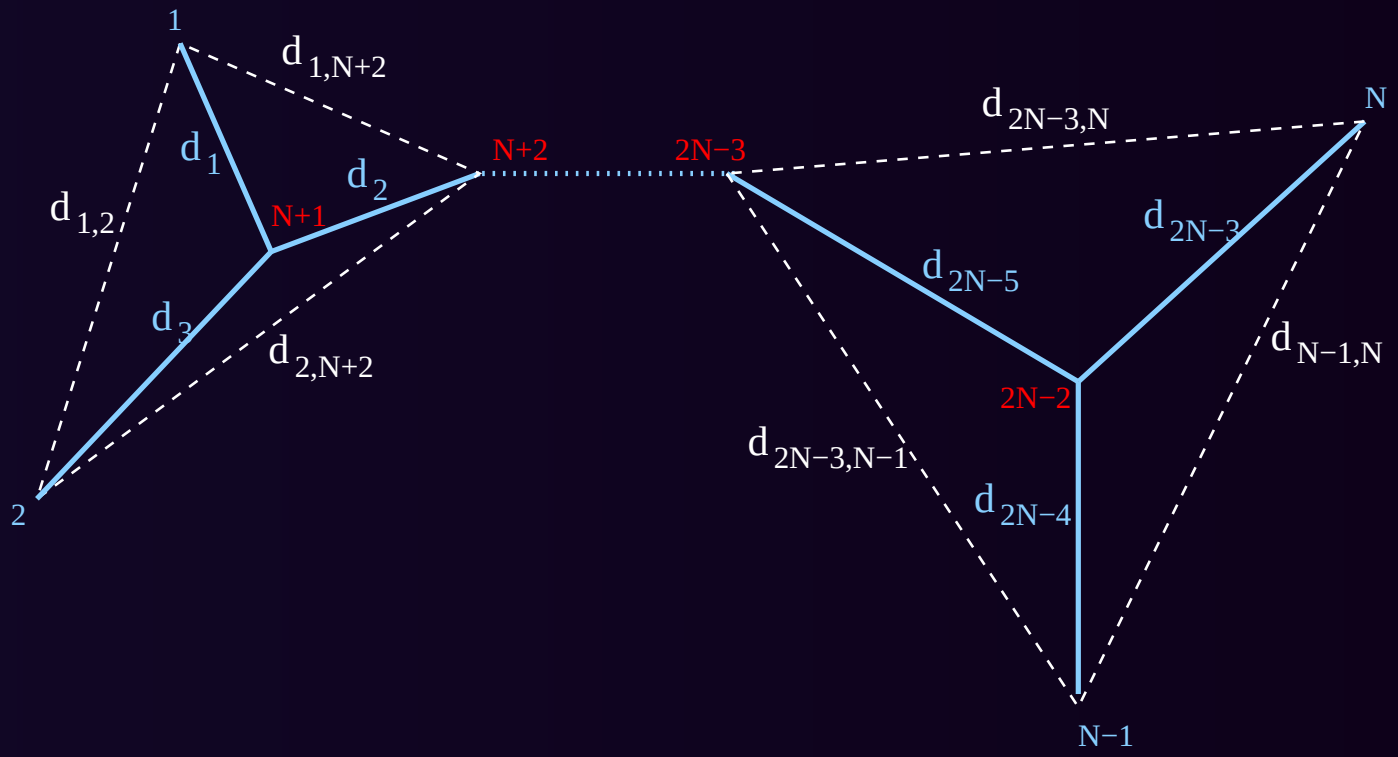
$$\forall M,\ d_k + d_{k+1} + d_{k+2} = \frac{d_{A,B} + d_{A,C} + d_{B,C}}{2}$$

Equations

Equations


1
d 1,N+2
d 1
d 2
d 1,2
N+1
N+2
2N−3
d 2N−3,N
N
d 2N−3
d 2N−5
d 3
d 2,N+2
d N−1,N
2N−2
d 2N−3,N−1
d 2N−4
2
N−1

Equations

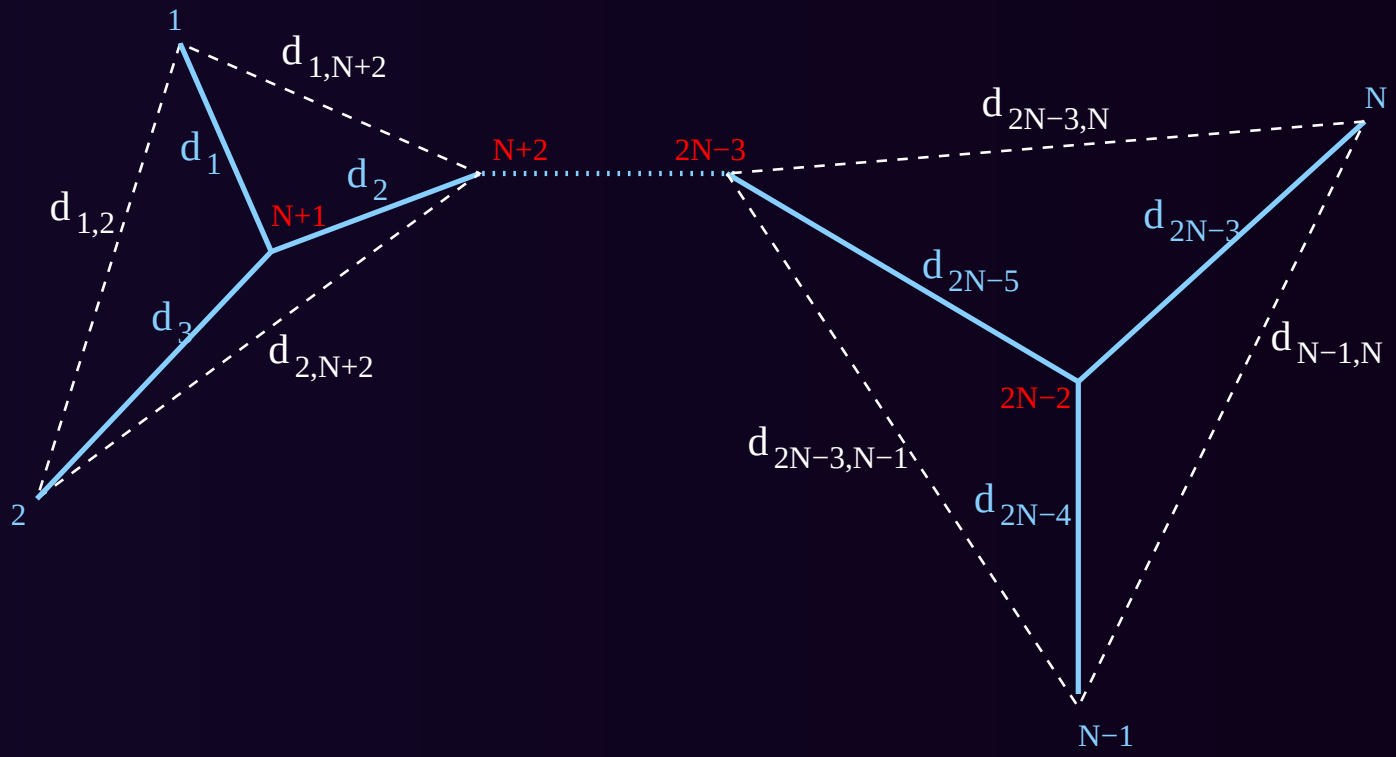


$$d_1 + d_2 + d_3 = \frac{d_{1,2} + d_{2,N+2} + d_{1,N+2}}{2}$$

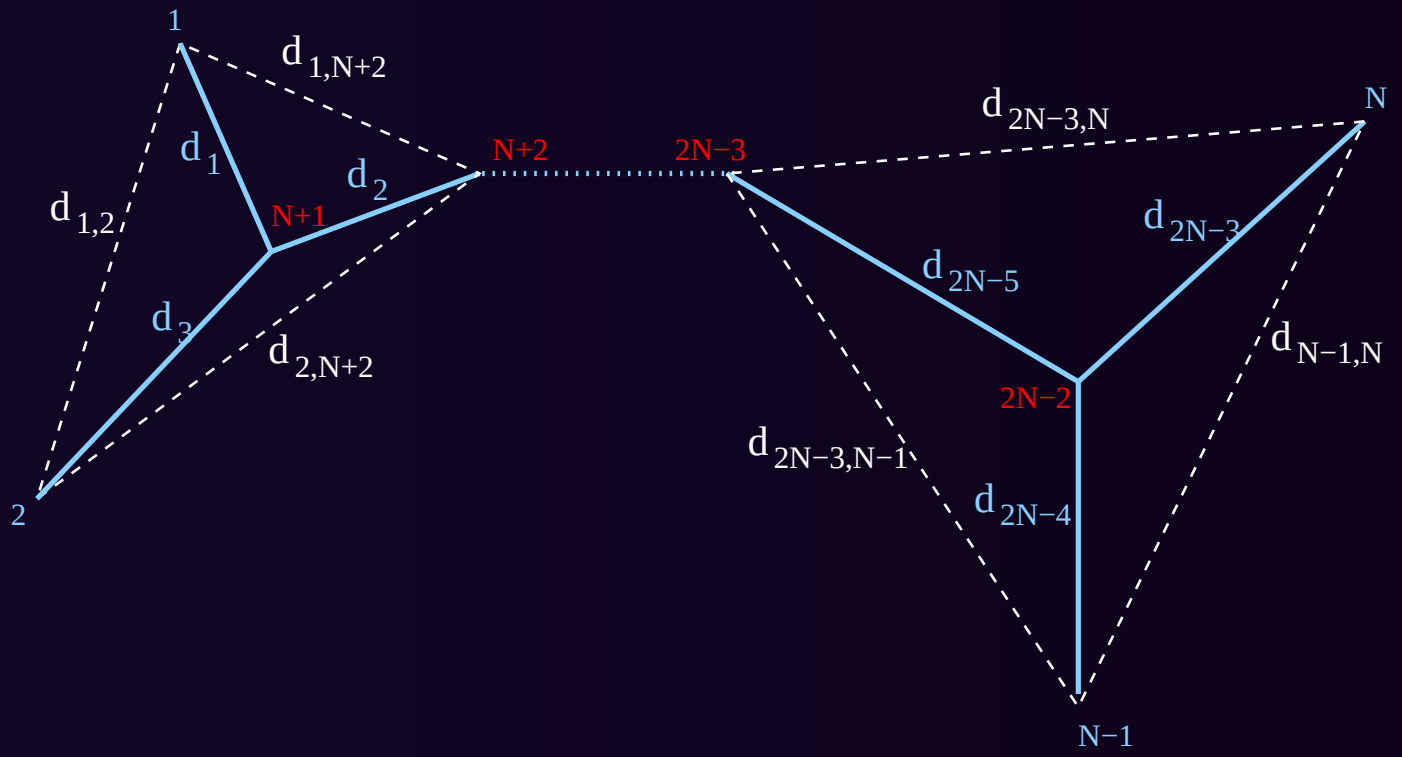
$$\dots$$

$$d_{2N-5} + d_{2N-4} + d_{2N-3} = \frac{d_{2N-3,N-1} + d_{N-1,N} + d_{2N-3,N}}{2}$$

Problems

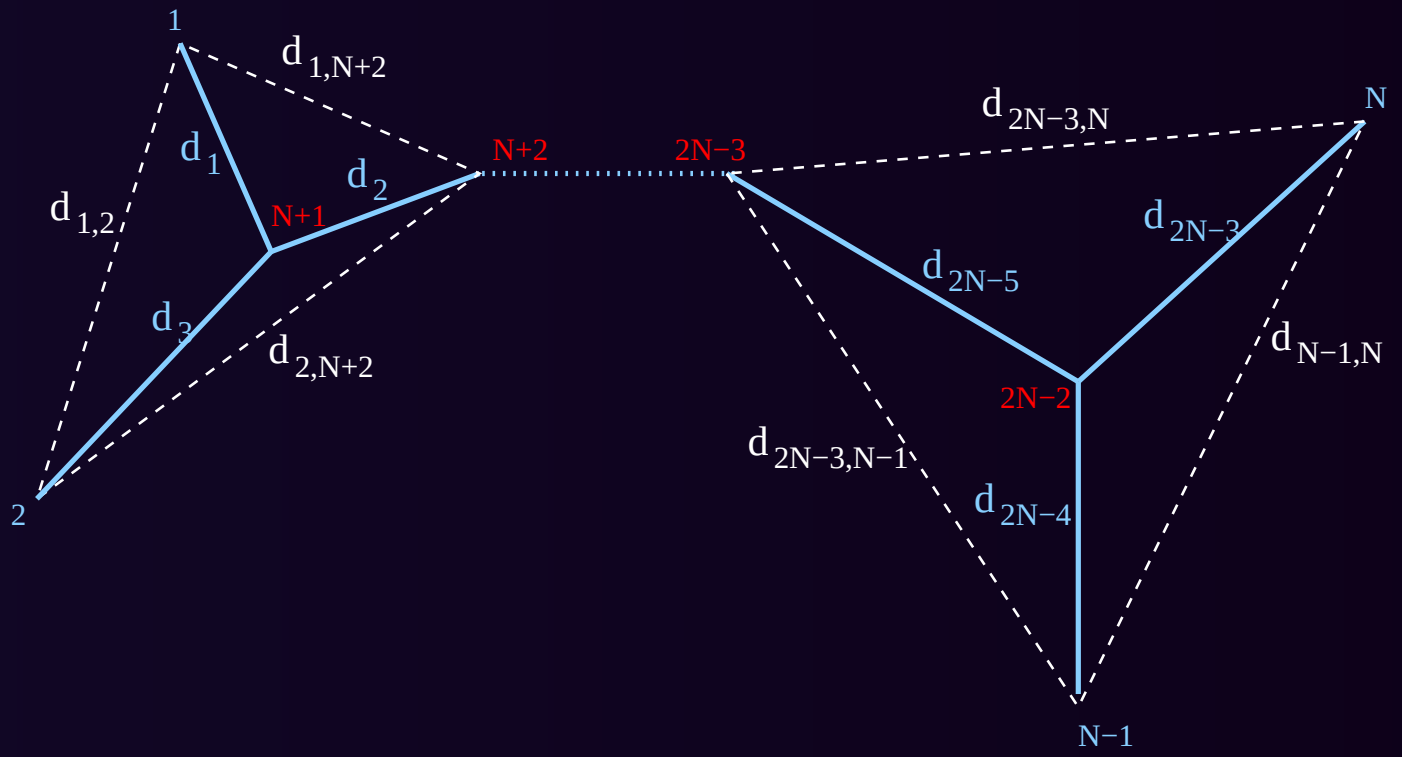

1
d 1,N+2
d 1
d 2
d 1,2
N+1
N+2
2N−3
d 2N−3,N
N
d 2N−3
d 2N−5
d 3
d 2,N+2
d N−1,N
2N−2
d 2N−3,N−1
d 2N−4
2
N−1

Problems



- **There are $\approx 5N$ variables, but only $N - 2$ equations $\cdots$**

Problems

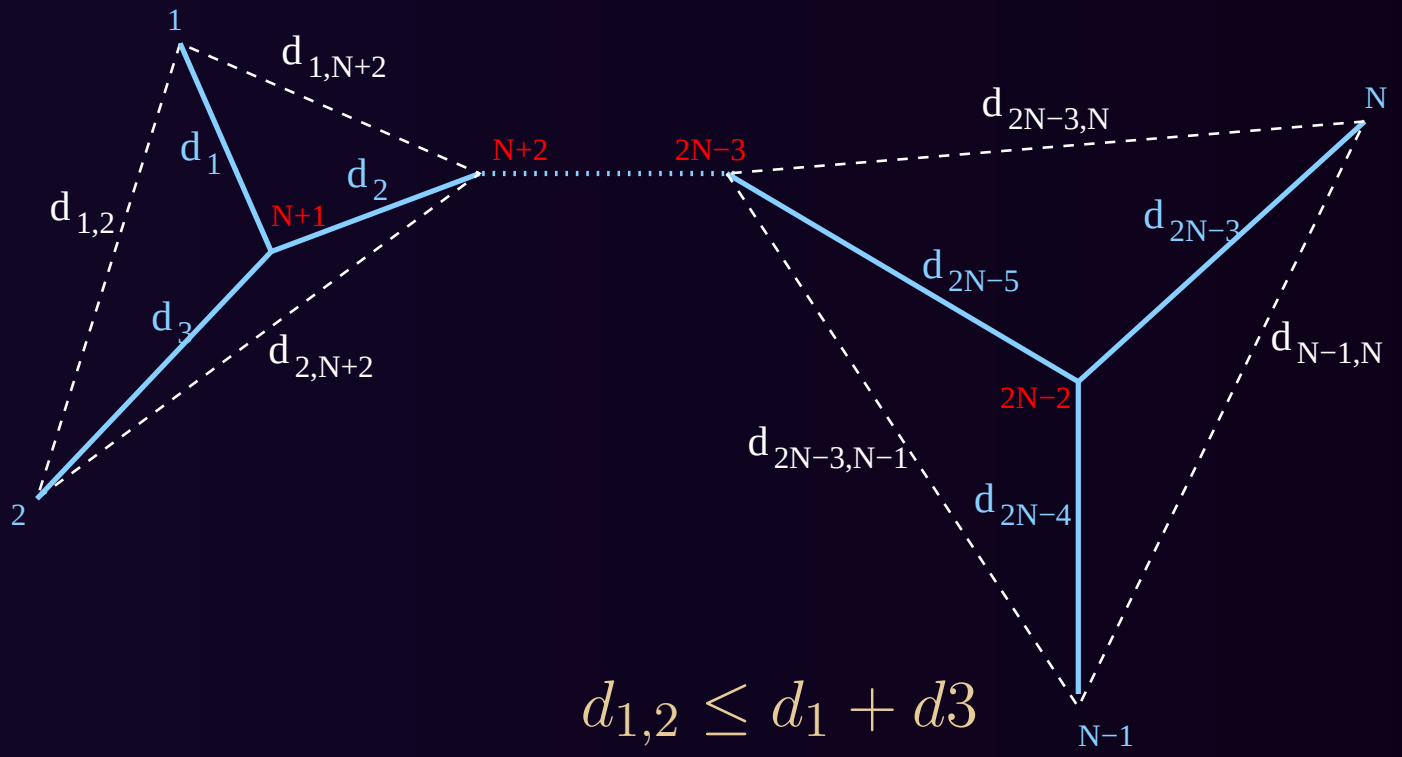


- **There are $\approx 5N$ variables, but only $N-2$ equations $\cdots$**
- **There are many (and redundant) triangular inequations**

Inequality Equations

- **We want to pick up a minimum number of inequations to cover all the variables**
- **We know only the distance matrix and tree topology**
- **Choices:**
  for each pair of genomes, find the two shortest paths from one to another, and build one inequation for each path

Inequality Equations



$$d_{1,2} \leq d_1 + d3$$

$$d_{N-1,N} \leq d_{2N-4} + d2N - 3$$

$$\ldots$$

$$d_{1,N-1} \leq d_{1,N+2} + \cdots + d_{2N-3,N-1}$$

$$d_{1,N-1} \leq d_{1,N+2} + \cdots + d_{2N-5}, + d_{2N-4}$$

Sum-up

- **Examine every tree**
- **For each tree (with N genomes)**
  - Minimize the sum of $2N - 3$ edge lengths
  - $\approx 5N$ variables total
  - $N - 2$ equations, $< 2N(N - 1)$ inequations
  - These numbers are relatively small if $N < 20$
- **Use `lp_solve` to find the length of the tree**
- **Return tree(s) with the minimum length**

Experimental Design

- **Real datasets—limited samples**
- **Simulation**
  - Generate a tree (*true tree*) from different topologies: uniform, birth-death, $\cdots$
  - Assign edge lengths based on the expected evolutionary rate
  - Assign gene content to each genome based on the edge length
  - Use GRAPPA to find a tree (*inferred tree*)
  - Compare inferred tree with true tree to determine the accuracy

Topological Accuracy

Topological Accuracy

- **False positive:**
  an edge is in the inferred tree,
  not in the true tree
- **False negative:**
  an edge is in the true tree,
  not in the inferred tree

Topological Accuracy

- **False positive:**
  an edge is in the inferred tree,
  not in the true tree
- **False negative:**
  an edge is in the true tree,
  not in the inferred tree

Goal: to minimize FP and FN

Simulation Details

- **Number of genomes (N): 12**

Simulation Details

- **Number of genomes (N): 12**
- **Number of genes (n): 200, 500 and 1000**

Simulation Details

- **Number of genomes (N): 12**
- **Number of genes (n): 200, 500 and 1000**
- **Expected # of events on each edge:** $0.05n - 0.15n$

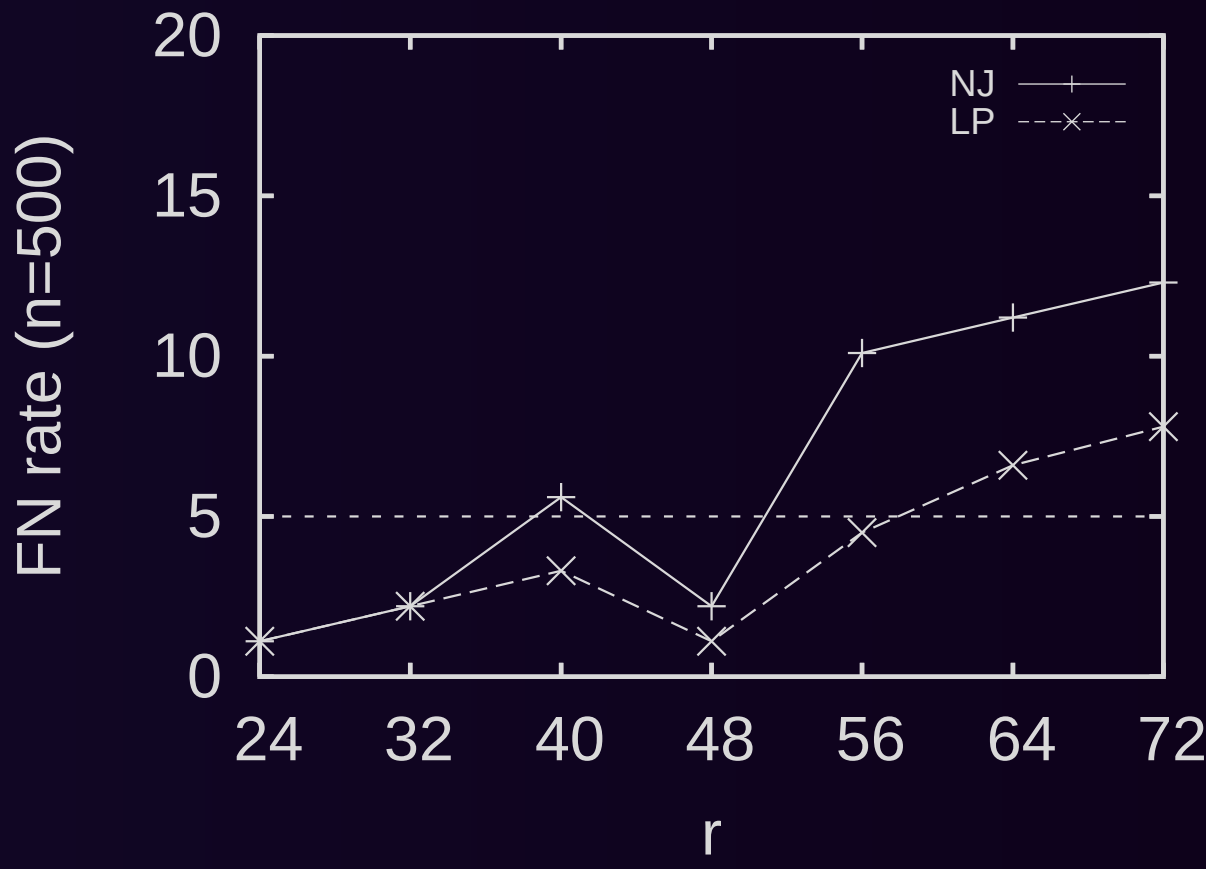
Simulation Details

- **Number of genomes (N): 12**
- **Number of genes (n): 200, 500 and 1000**
- **Expected # of events on each edge:** $0.05n - 0.15n$
- **Tree topologies: uniform and birth-death**

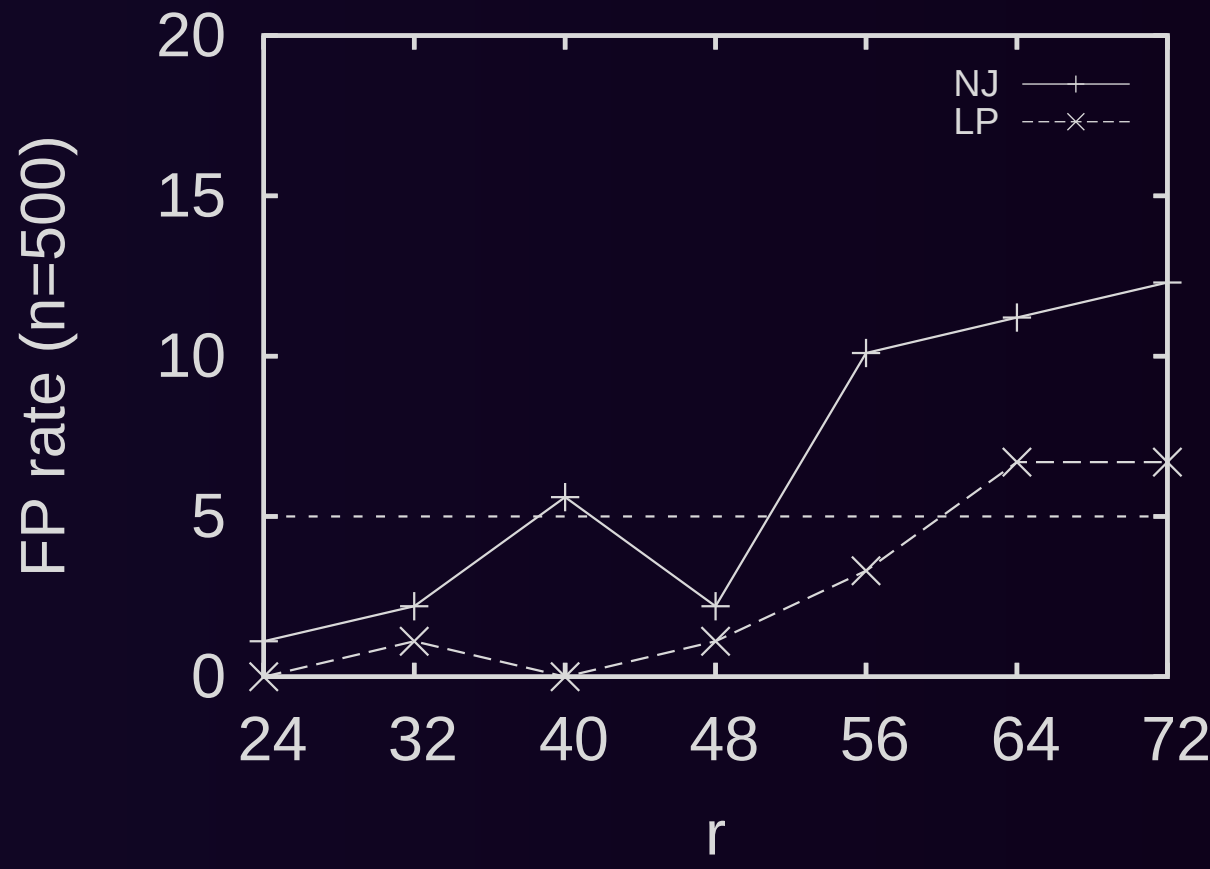
Simulation Details

- **Number of genomes (N): 12**
- **Number of genes (n): 200, 500 and 1000**
- **Expected # of events on each edge:** $0.05n - 0.15n$
- **Tree topologies: uniform and birth-death**
- **Datasets on each combination:** 10

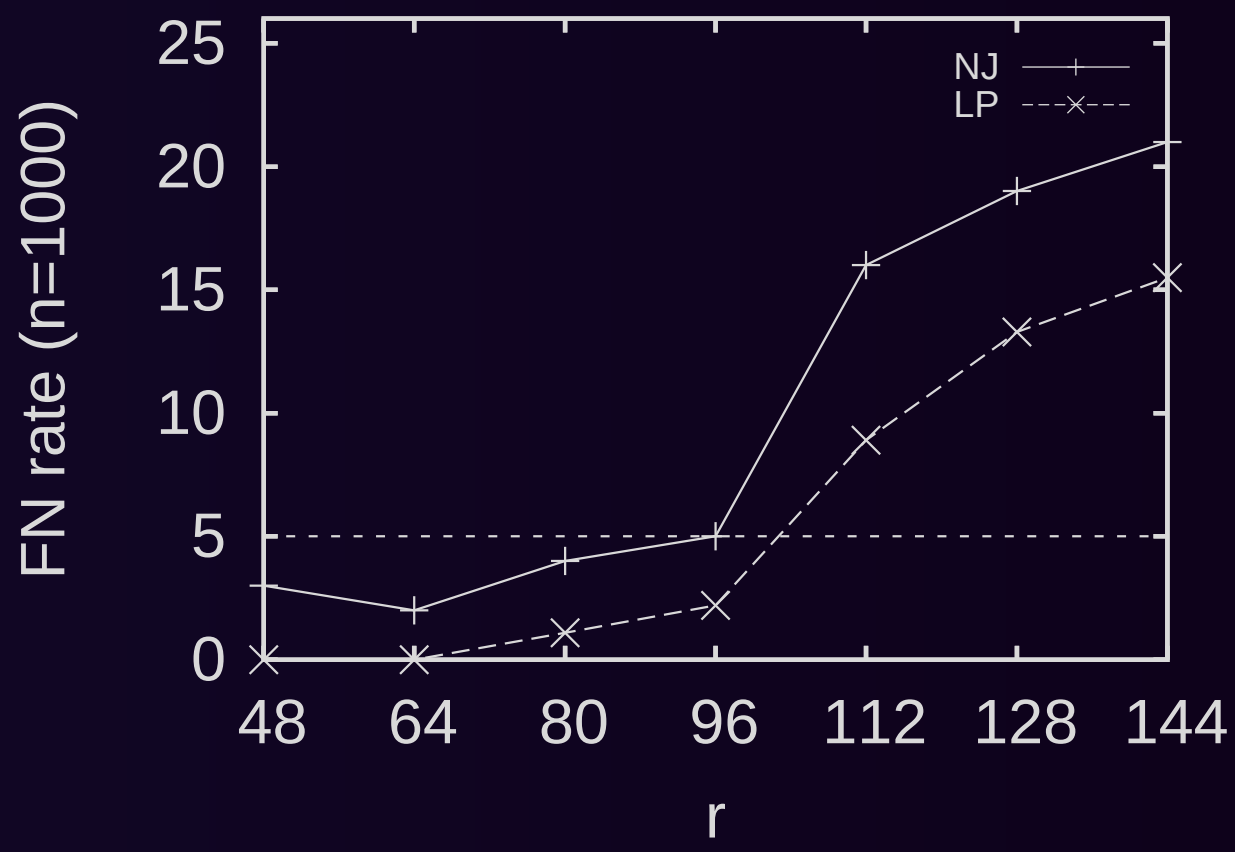
FN (500 genes, BD tree)



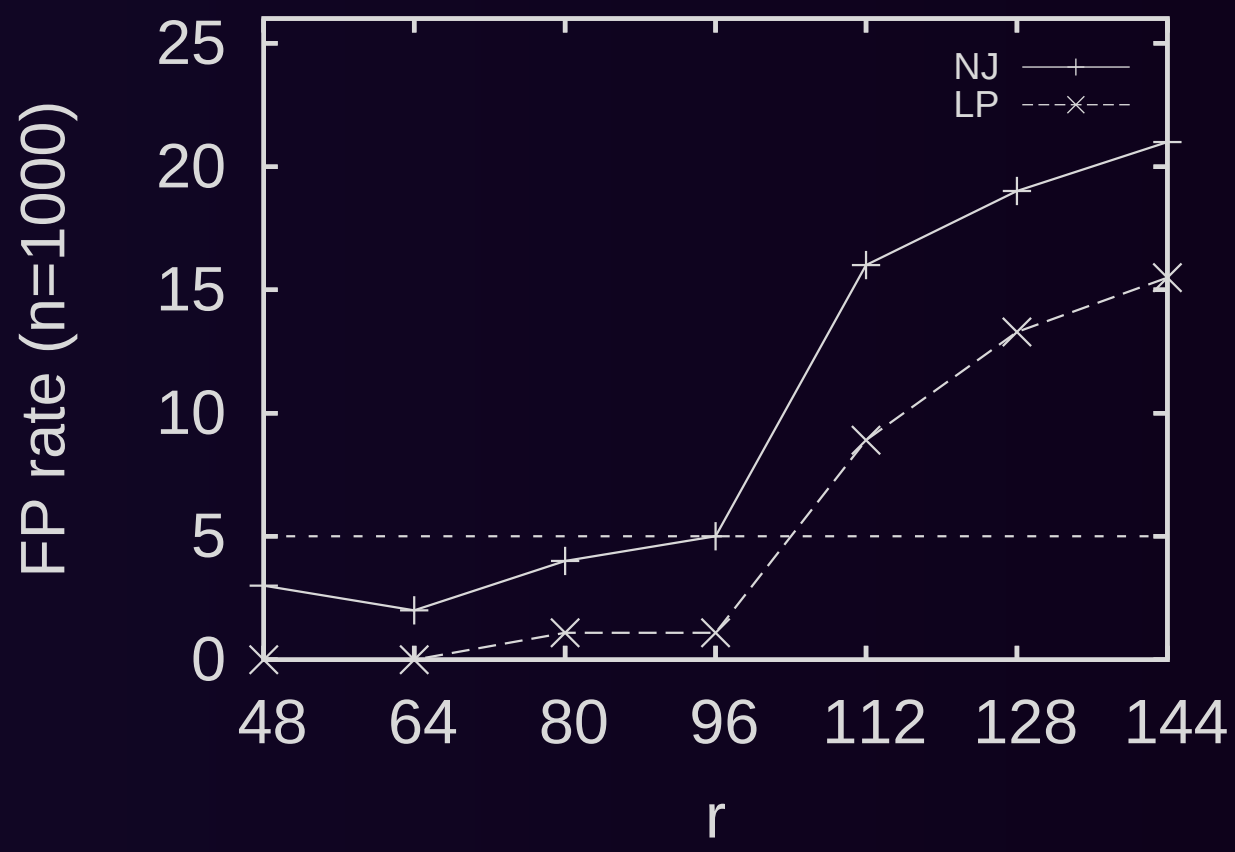
FP (500 genes, BD tree)



FN (1000 genes, uniform tree)



FP (1000 genes, uniform tree)



Conclusion

- **Linear programming gives us a new and accurate method for difficult datasets**
- **Can be applied to any distance**
- **Has potential to be used for large and complex genomes**
- **Can be extended to solve the median problems**