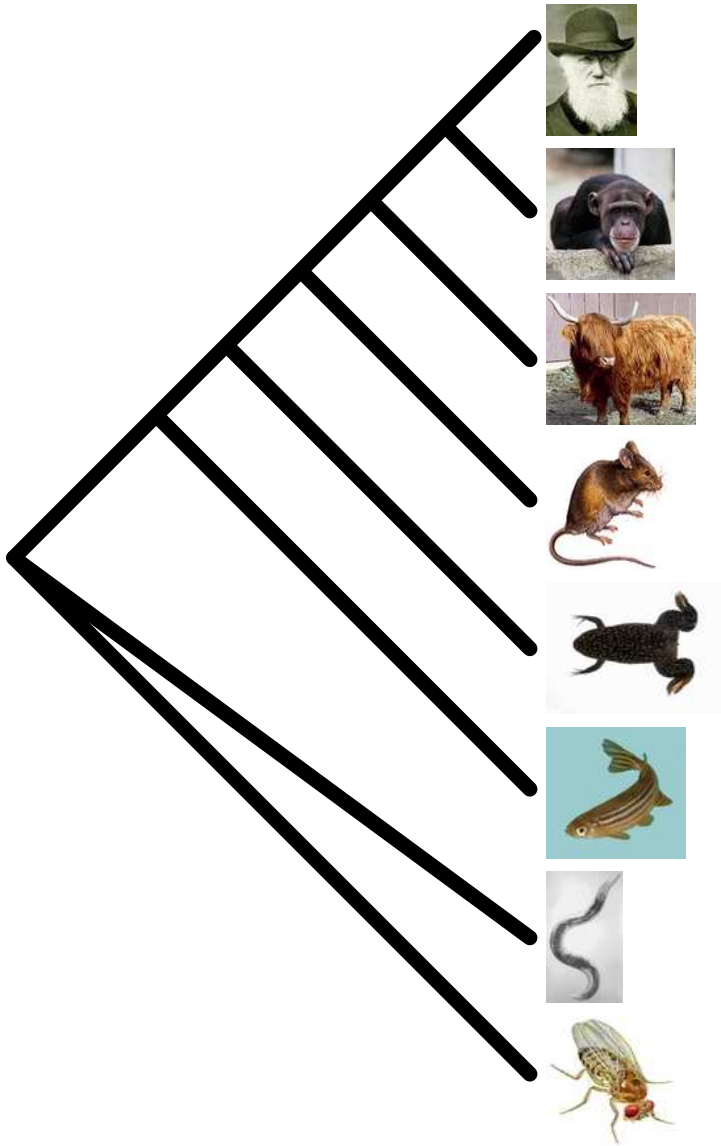
Distributed Computing in Evolutionary Tree Reconstruction

Heiko A. Schmidt

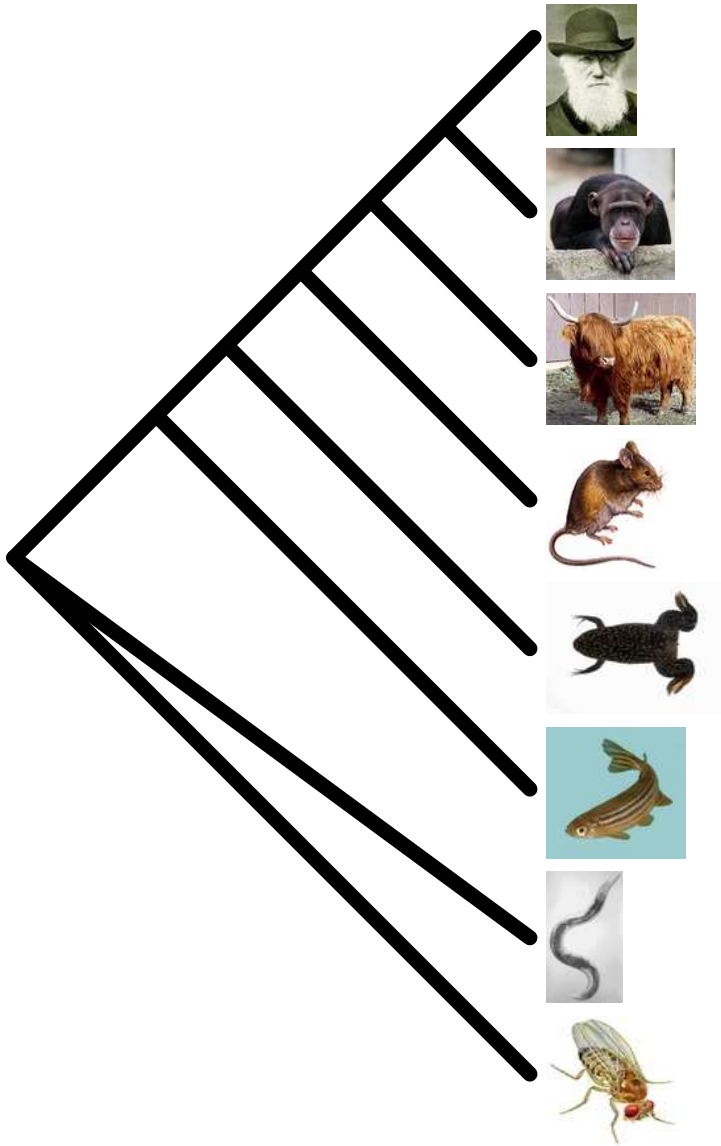
June 21, 2007

Introduction: Phylogenetic Reconstruction



Introduction: Phylogenetic Reconstruction

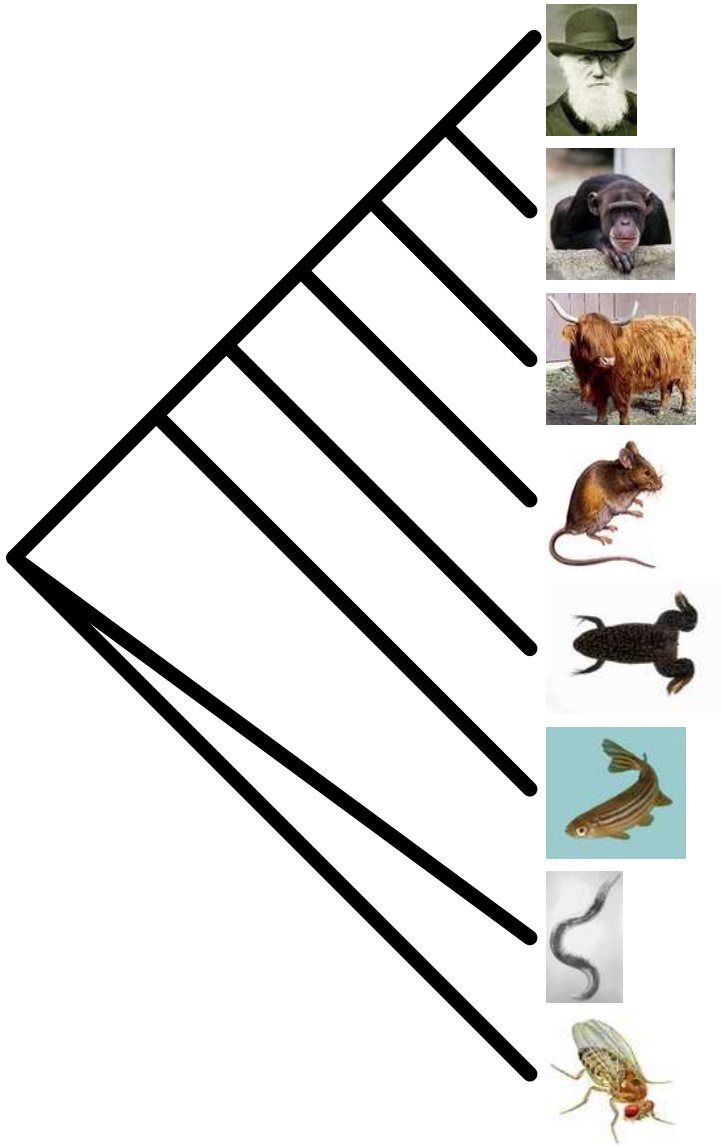
```
...CTCGG CTTAC TTCTC TTCCT TCTCT...
...CTTGG CTTAT TCCTT TTCCT CCTTA...
...TTAGG GGCCC TCTTA CTAAT TCTAG...
...TGAAA CATTG GAGTA CTTCT ACTGT...
...TGAAA TATTG GTGTG ATCCT CCTAT...
...TGAAA CATCG GAGTA GTCCT GTTCT...
...TGAAT GTCTG GTTTA ACAAT TTATT...
...TGATT AATTG GAGTA ATTAT TTTAT...
```



Introduction: Phylogenetic Reconstruction

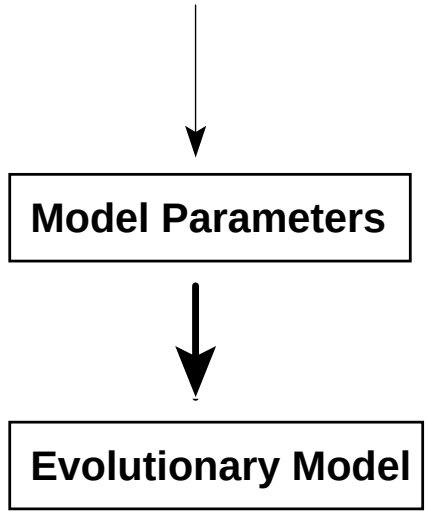
```
...CTCGG CTTAC TTCTC TTCCT TCTCT...
...CTTGG CTTAT TCCTT TTCCT CCTTA...
...TTAGG GGCCC TCTTA CTAAT TCTAG...
...TGAAA CATTG GAGTA CTTCT ACTGT...
...TGAAA TATTG GTGTG ATCCT CCTAT...
...TGAAA CATCG GAGTA GTCCT GTTCT...
...TGAAT GTCTG GTTTA ACAAT TTATT...
...TGATT AATTG GAGTA ATTAT TTTAT...
```

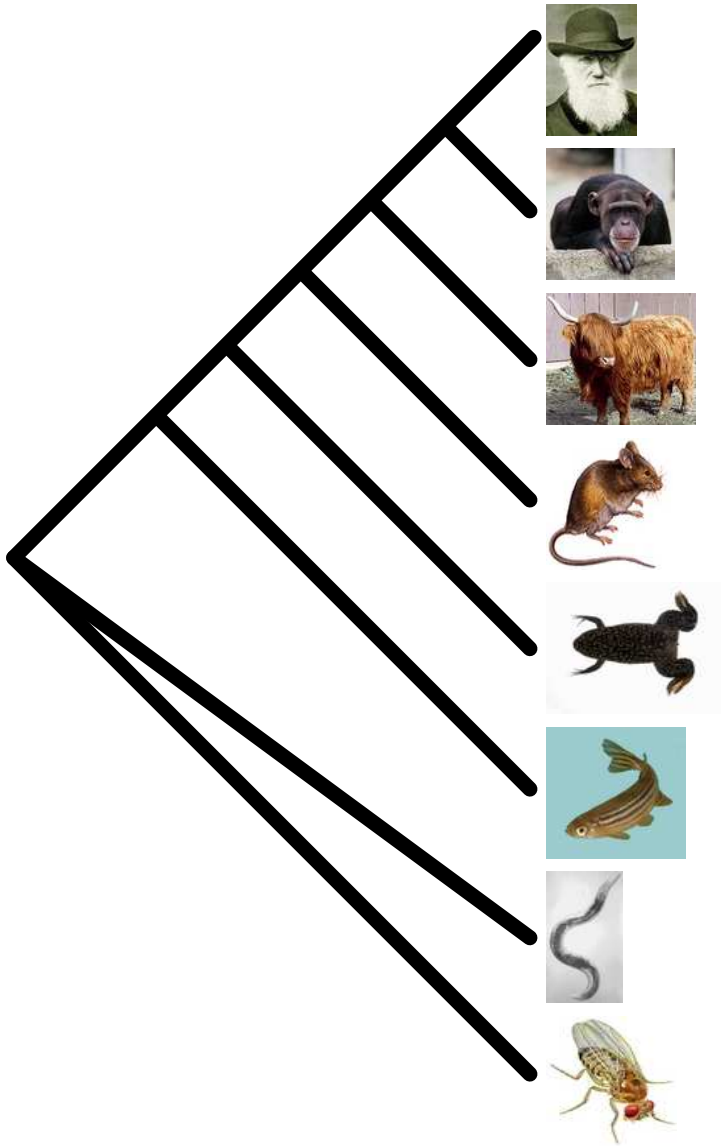
Evolutionary Model



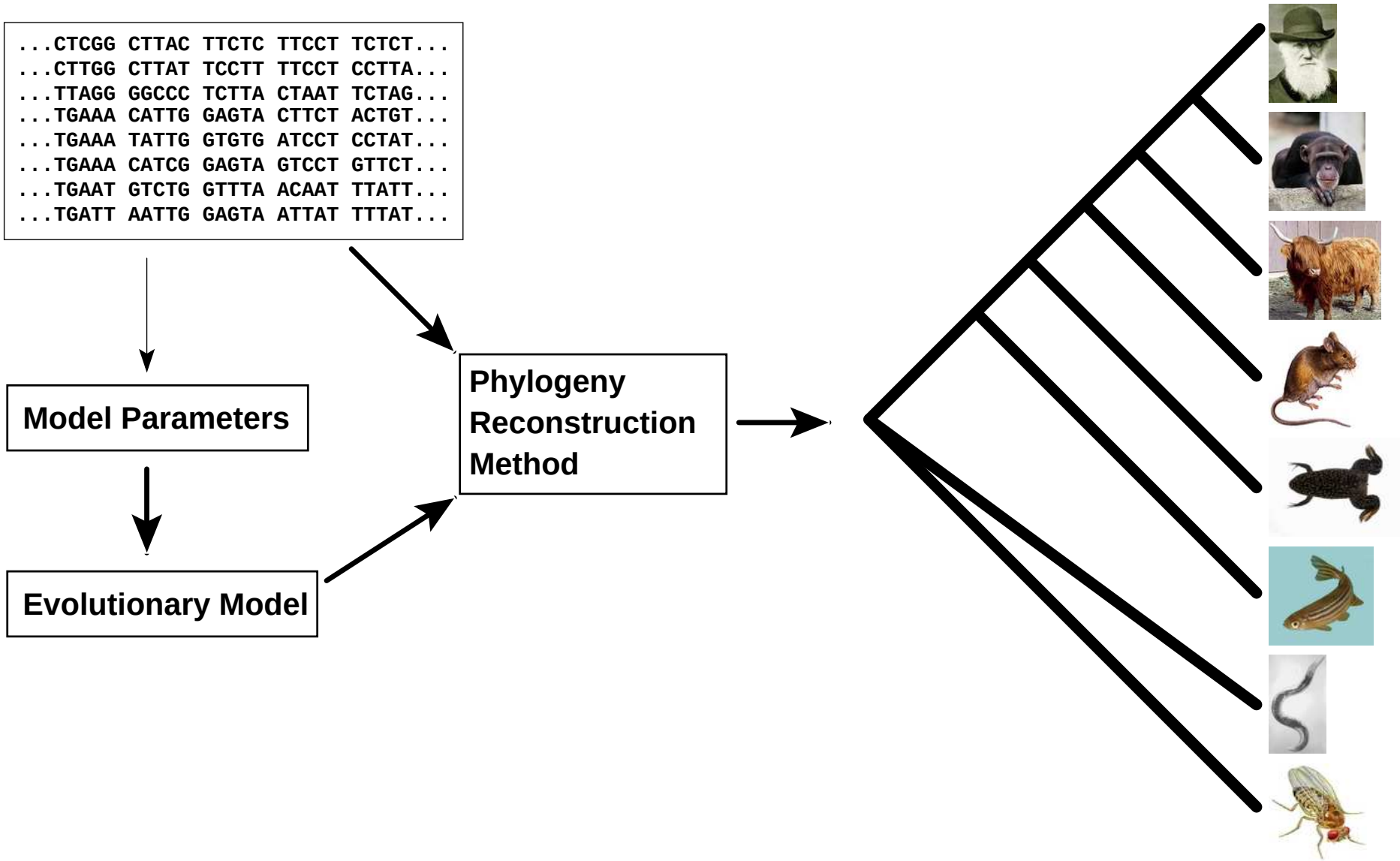
Introduction: Phylogenetic Reconstruction

```
...CTCGG CTTAC TTCTC TTCCT TCTCT...
...CTTGG CTTAT TCCTT TTCCT CCTTA...
...TTAGG GGCCC TCTTA CTAAT TCTAG...
...TGAAA CATTG GAGTA CTTCT ACTGT...
...TGAAA TATTG GTGTG ATCCT CCTAT...
...TGAAA CATCG GAGTA GTCCT GTTCT...
...TGAAT GTCTG GTTTA ACAAT TTATT...
...TGATT AATTG GAGTA ATTAT TTTAT...
```





Introduction: Phylogenetic Reconstruction



Main Types of Phylogenetic Methods

Data	Method	Evaluation Criterion
Characters (Alignment)	**Maximum Parsimony**	Parsimony
Characters (Alignment)	**Statistical Approaches: Likelihood, Bayesian**	Evolutionary Models
Distances	**Distance Methods**	Evolutionary Models

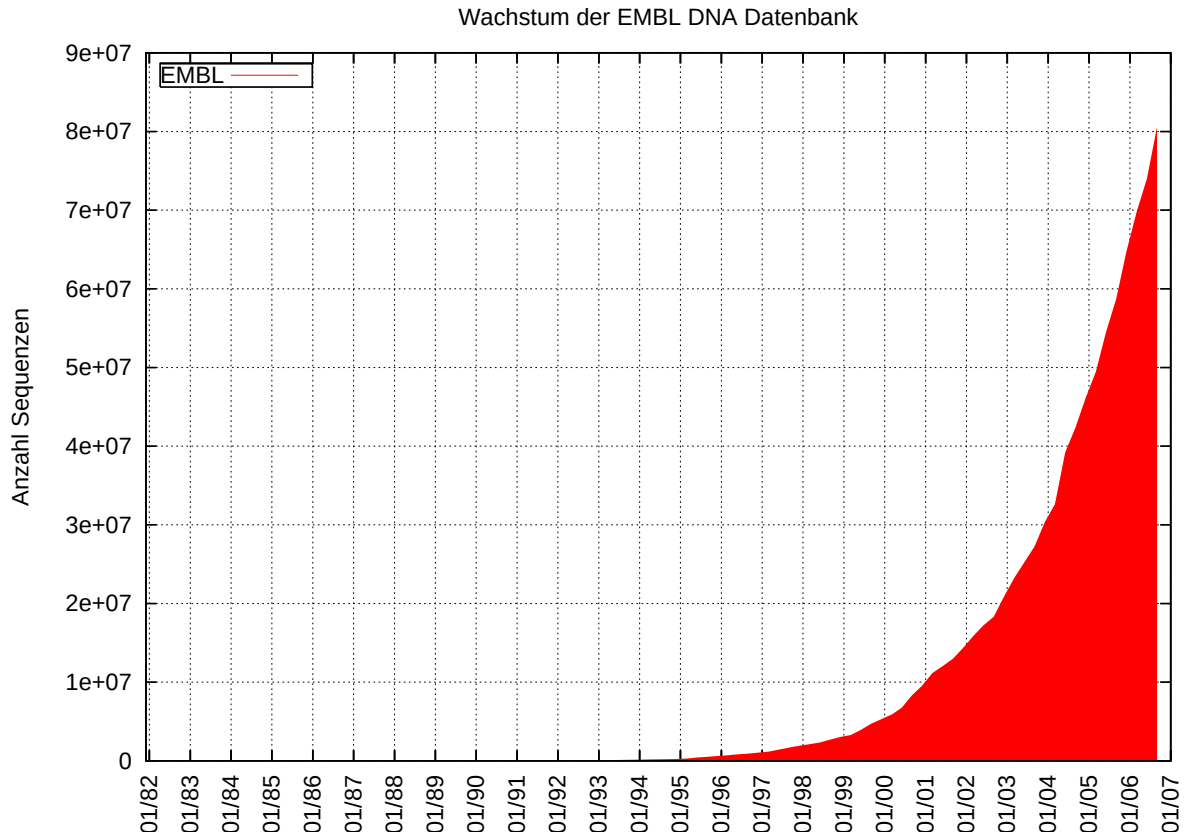
Main Types of Phylogenetic Methods

Data	Method	Evaluation Criterion
Characters (Alignment)	**Maximum Parsimony**	Parsimony
	Statistical Approaches: Likelihood, Bayesian	Evolutionary Models
Distances	**Distance Methods**	Evolutionary Models

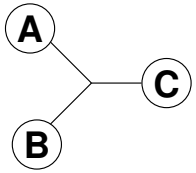
Complexity of phylogenetic methods

- Steiner tree problem = NP-complete (Foulds+Graham, 1982)
- Maximum parsimony = NP-complete (Day et al., 1986)
- Compatibilitiy trees = NP-complete (Day+Sankoff, 1986)
- Dissimilarity matrices = NP-complete (Day, 1987)
- Perfect trees = NP-complete (Bodlaender et al., 1992)
- Ancestral ML = NP-complete (Addario-Berry et al., 2004)
- Maximum likelihood trees = NP-hard (Chor+Tuller, 2005)

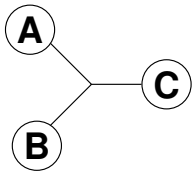
Data base growth

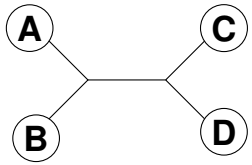


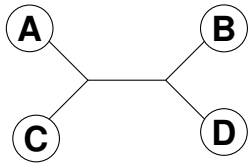
Number of Trees to Examine...

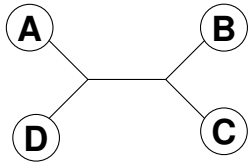


Number of Trees to Examine...

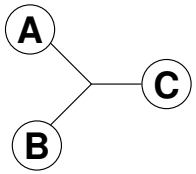


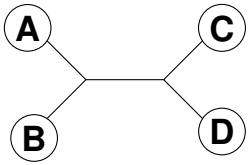


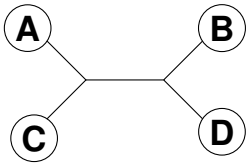


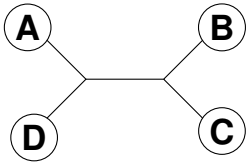


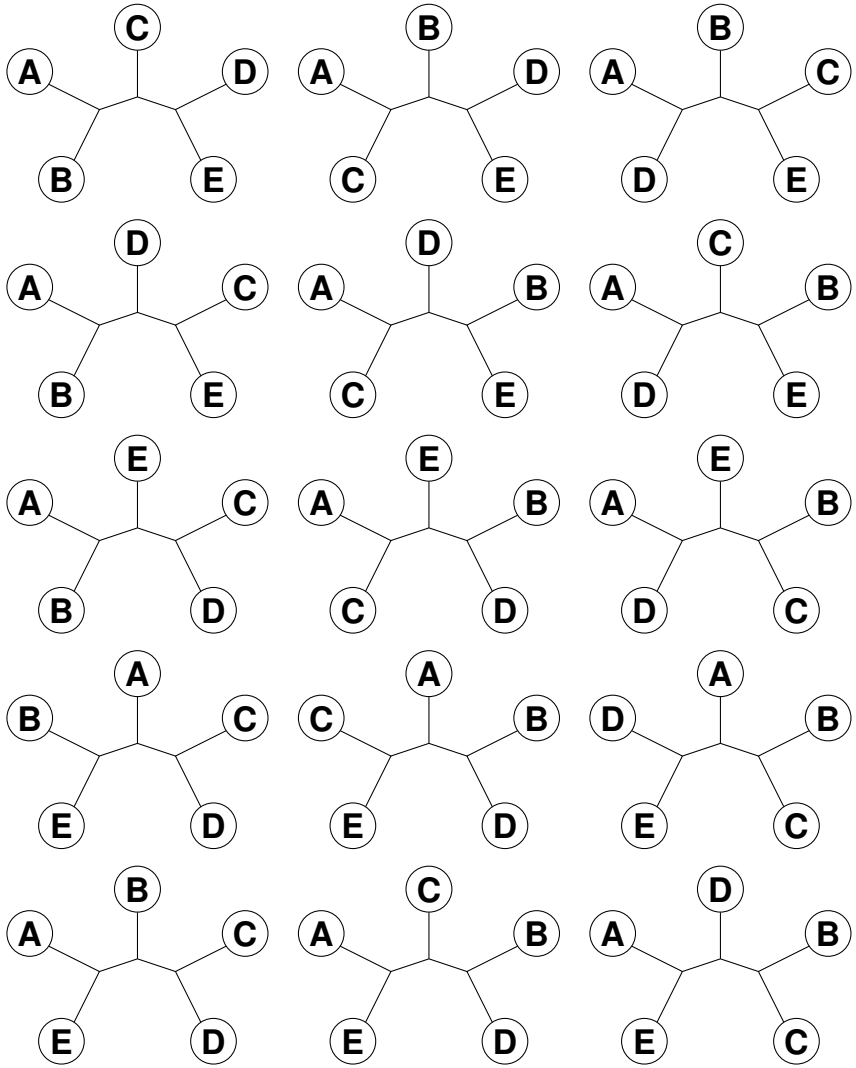
Number of Trees to Examine...



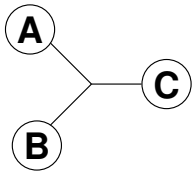


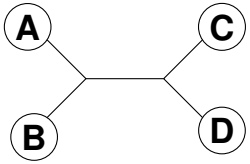


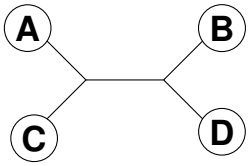


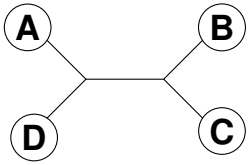


Number of Trees to Examine...

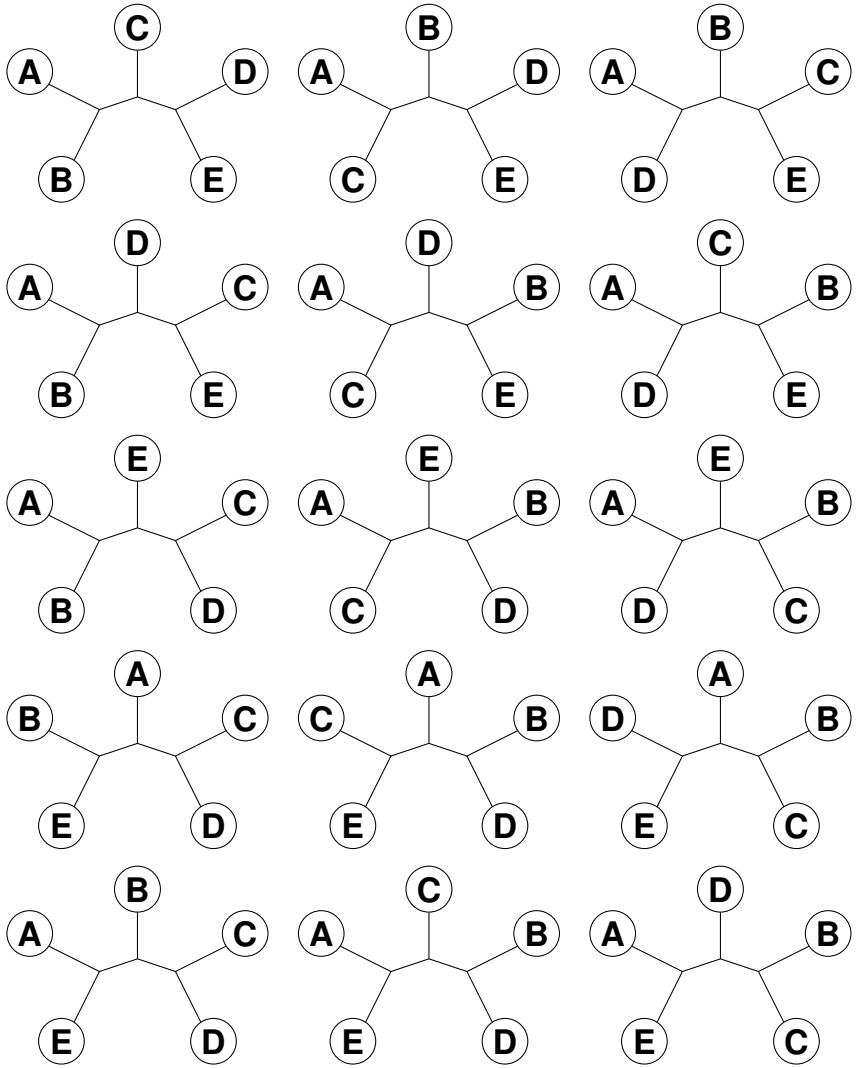




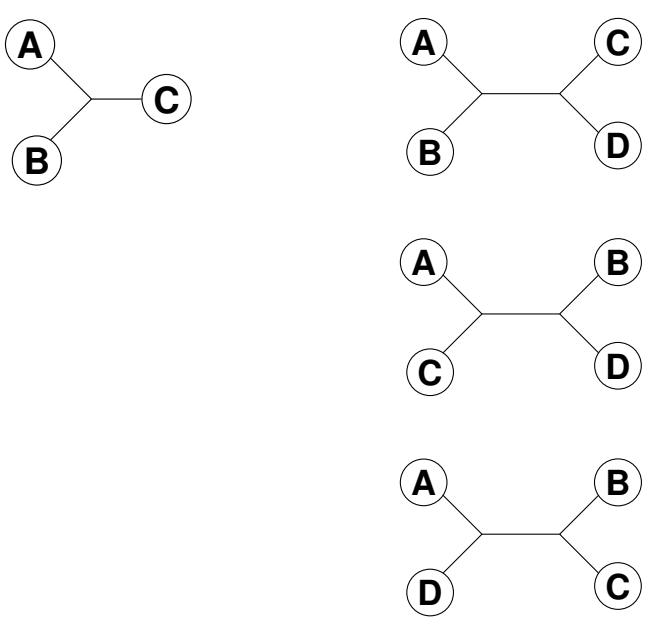


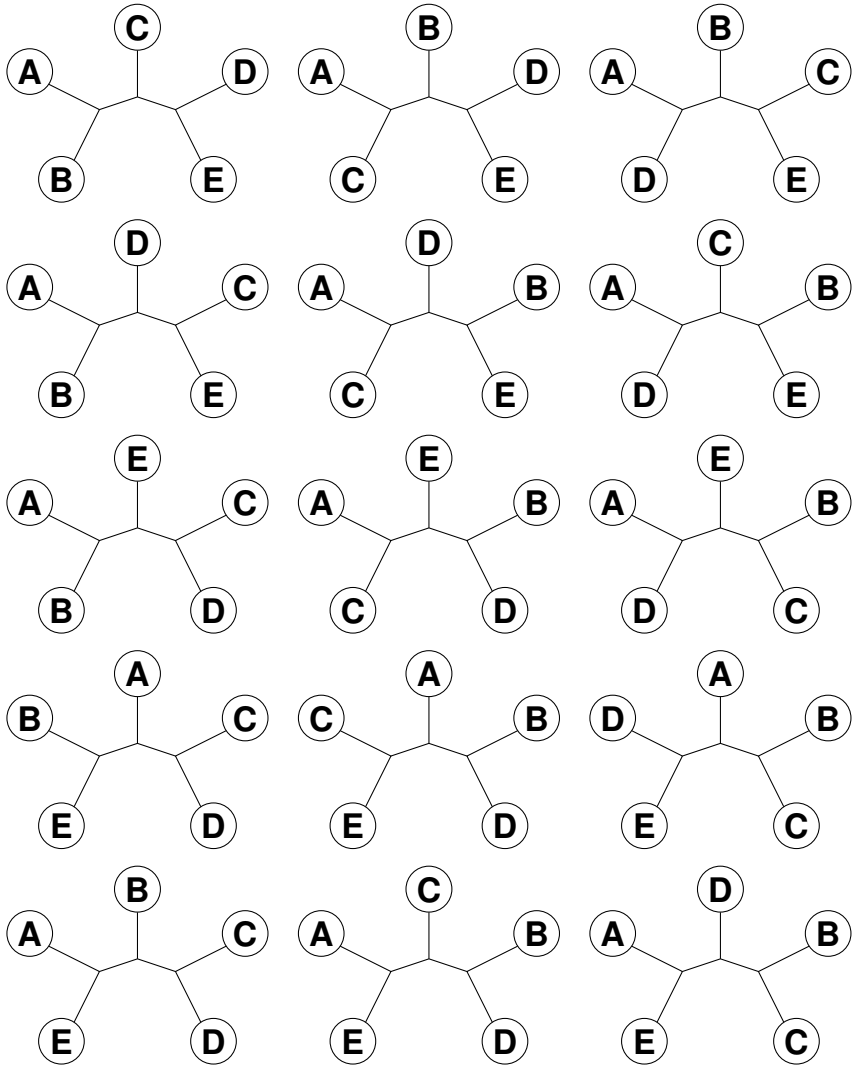


$$B(n) = \frac{(2n-5)!}{2^{n-3}(n-3)!}$$



Number of Trees to Examine...





$$B(n) = \frac{(2n-5)!}{2^{n-3}(n-3)!}$$

$$B(10) = 2027025$$

$$B(55) = 2.98 \cdot 10^{84}$$

$$B(100) = 1.70 \cdot 10^{182}$$

Finding the ML Tree

Exhaustive Search: guarantees to find the optimal tree, because all trees are evaluated, but not feasible for more than 10-12 taxa.

Finding the ML Tree

Exhaustive Search: guarantees to find the optimal tree, because all trees are evaluated, but not feasible for more than 10-12 taxa.

Branch and Bound: guarantees to find the optimal tree, without searching certain parts of the tree space – can run on more sequences, but often not for current-day datasets.

Finding the ML Tree

Exhaustive Search: guarantees to find the optimal tree, because all trees are evaluated, but not feasible for more than 10-12 taxa.

Branch and Bound: guarantees to find the optimal tree, without searching certain parts of the tree space – can run on more sequences, but often not for current-day datasets.

Heuristics: cannot guarantee to find the optimal tree, but are at least able to analyze large datasets.

Finding the ML Tree

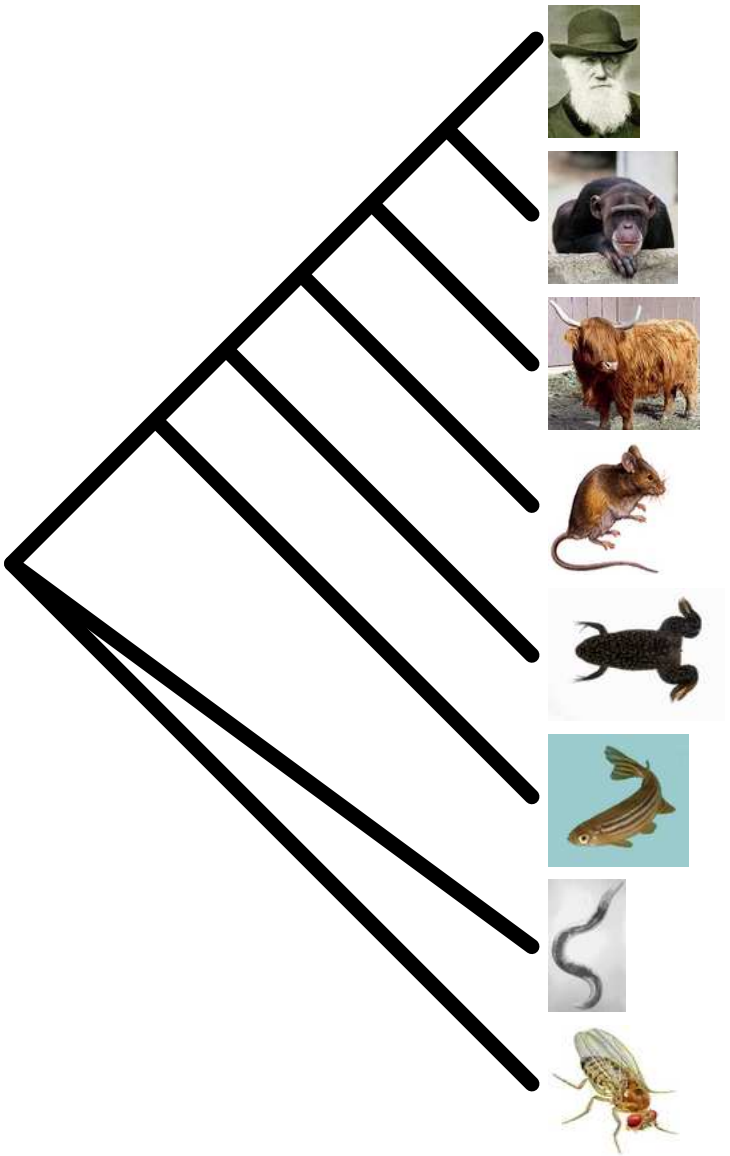
Exhaustive Search: guarantees to find the optimal tree, because all trees are evaluated, but not feasible for more than 10-12 taxa.

Branch and Bound: guarantees to find the optimal tree, without searching certain parts of the tree space – can run on more sequences, but often not for current-day datasets.

Heuristics: cannot guarantee to find the optimal tree, but are at least able to analyze large datasets.

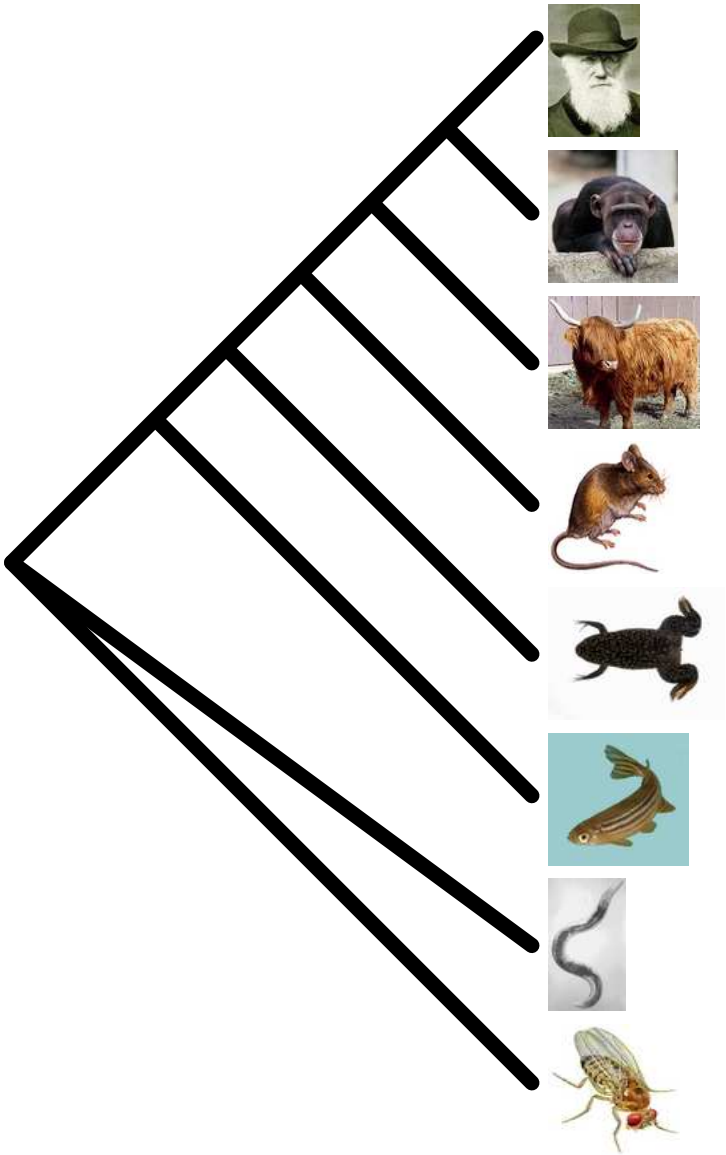
Parallel Heuristics: Like above, but need shorter running time.

How to parallelize?



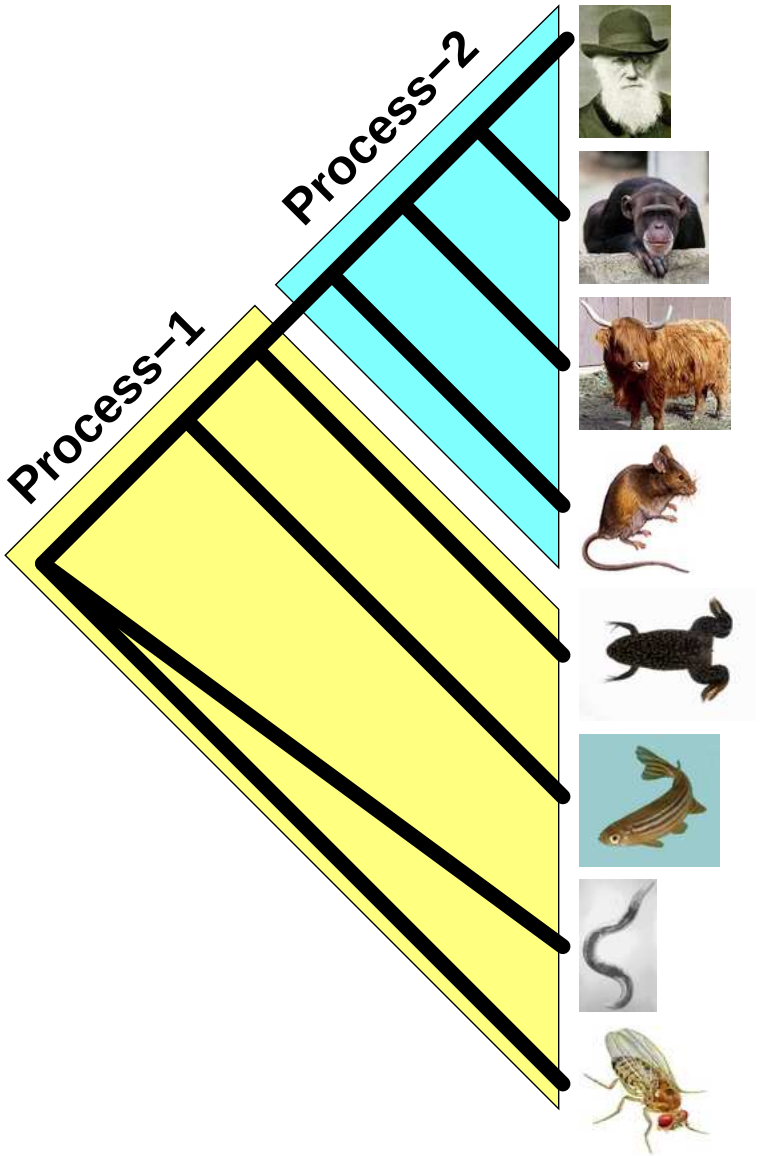
```
...CTCGG CTTAC TTCTC TTCCT TCTCT...
...CTTGG CTTAT TCCTT TTCCT CCTTA...
...TTAGG GGCCC TCTTA CTAAT TCTAG...
...TGAAA CATTG GAGTA CTTCT ACTGT...
...TGAAA TATTG GTGTG ATCCT CCTAT...
...TGAAA CATCG GAGTA GTCCT GTTCT...
...TGAAT GTCTG GTTTA ACAAT TTATT...
...TGATT AATTG GAGTA ATTAT TTTAT...
```

How to parallelize?



Process−1	Process−2
...CTCGG CTTAC TTCTC	TTCCT TCTCT...
...CTTGG CTTAT TCCTT	TTCCT CCTTA...
...TTAGG GGCCC TCTTA	CTAAT TCTAG...
...TGAAA CATTG GAGTA	CTTCT ACTGT...
...TGAAA TATTG GTGTG	ATCCT CCTAT...
...TGAAA CATCG GAGTA	GTCCT GTTCT...
...TGAAT GTCTG GTTTA	ACAAT TTATT...
...TGATT AATTG GAGTA	ATTAT TTTAT...

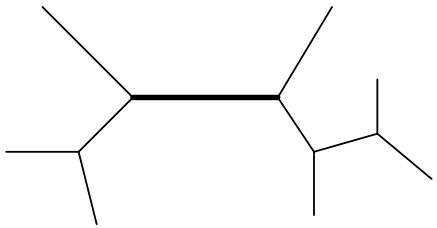
How to parallelize?




```
...CTCGG CTTAC TTCTC TTCCT TCTCT...
...CTTGG CTTAT TCCTT TTCCT CCTTA...
...TTAGG GGCCC TCTTA CTAAT TCTAG...
...TGAAA CATTG GAGTA CTTCT ACTGT...
...TGAAA TATTG GTGTG ATCCT CCTAT...
...TGAAA CATCG GAGTA GTCCT GTTCT...
...TGAAT GTCTG GTTTA ACAAT TTATT...
...TGATT AATTG GAGTA ATTAT TTTAT...
```

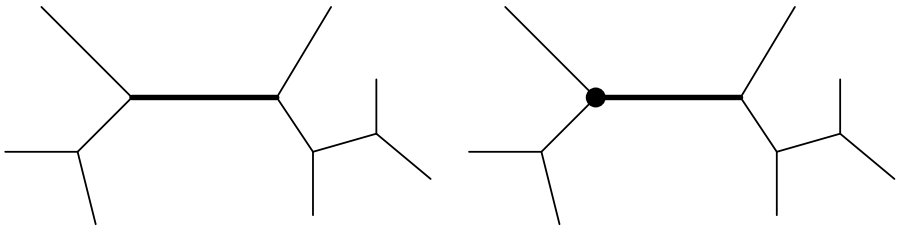
Compute the ML value for one tree

- Choose a branch (no time)



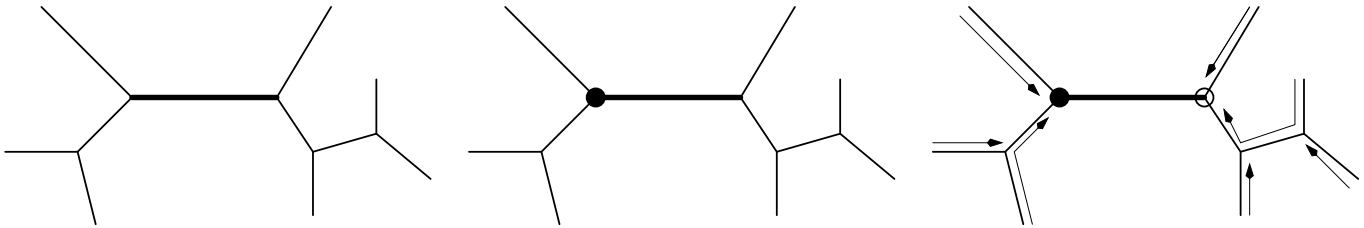
Compute the ML value for one tree

- Choose a branch (no time)
- Move the virtual root to an adjacent node (no time)



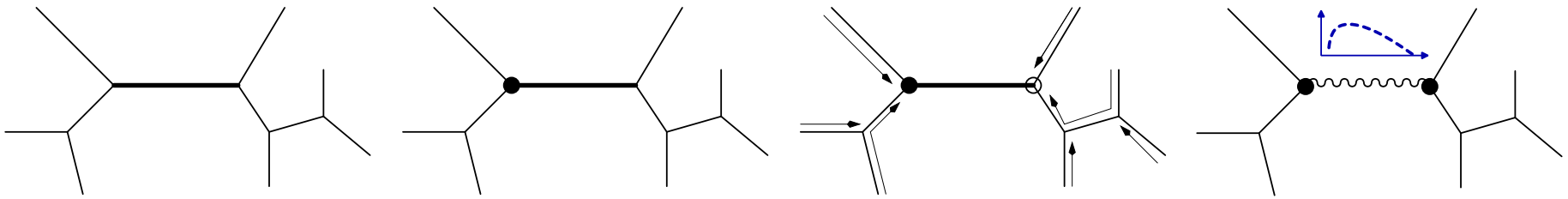
Compute the ML value for one tree

- Choose a branch (no time)
- Move the virtual root to an adjacent node (no time)
- Compute all partial likelihoods recursively (quick)



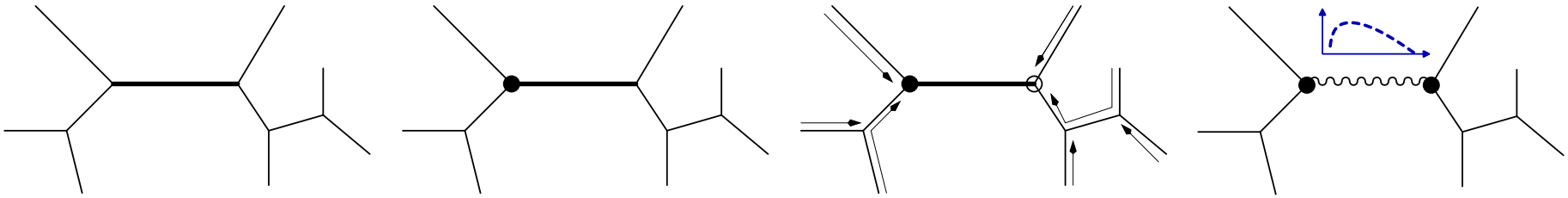
Compute the ML value for one tree

- Choose a branch (no time)
- Move the virtual root to an adjacent node (no time)
- Compute all partial likelihoods recursively (quick)
- Adjust length to maximize the likelihood (slow)

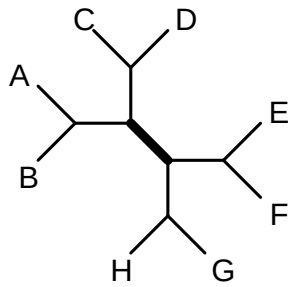


Compute the ML value for one tree

- Choose a branch (no time)
- Move the virtual root to an adjacent node (no time)
- Compute all partial likelihoods recursively (quick)
- Adjust length to maximize the likelihood (slow)
- Repeat for all branches and start over until convergence

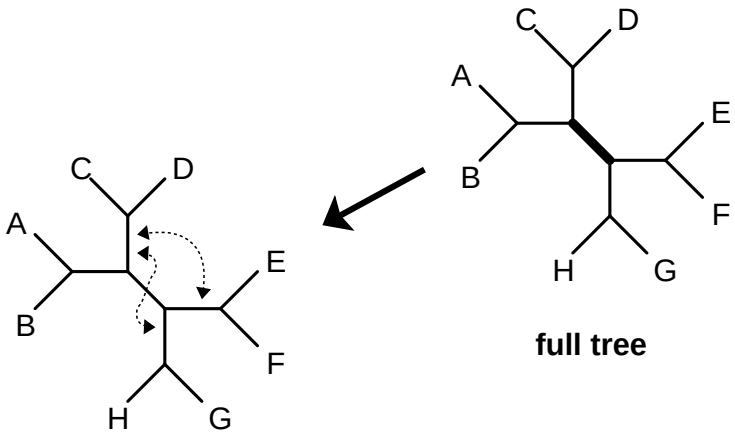


Tree search with topology rearrangements

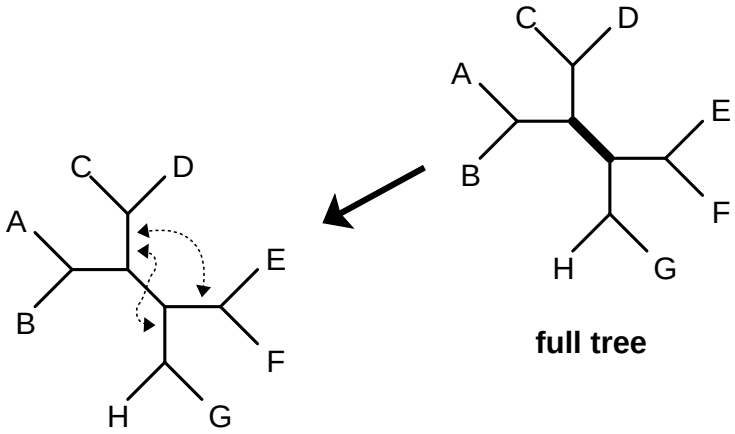


full tree

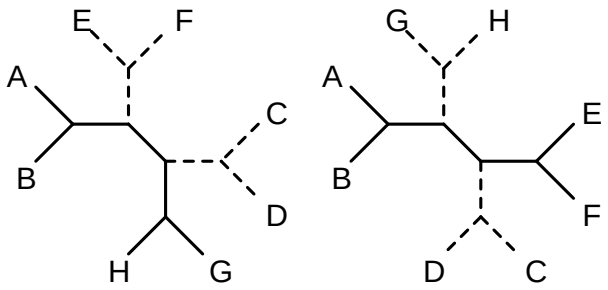
Tree search with topology rearrangements



Tree search with topology rearrangements



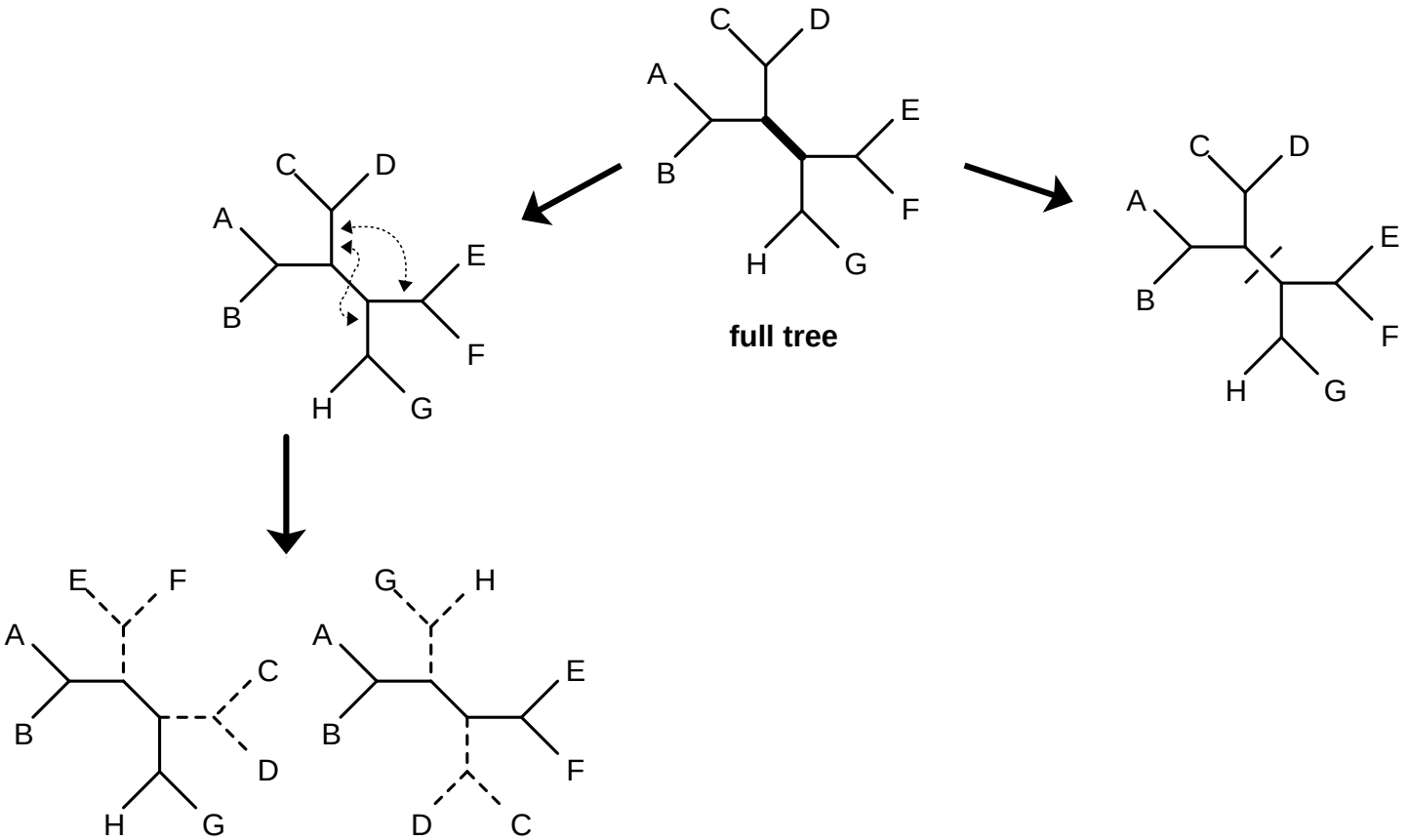
full tree



Nearest Neighbor Interchange (NNI)

O(n) NNI trees

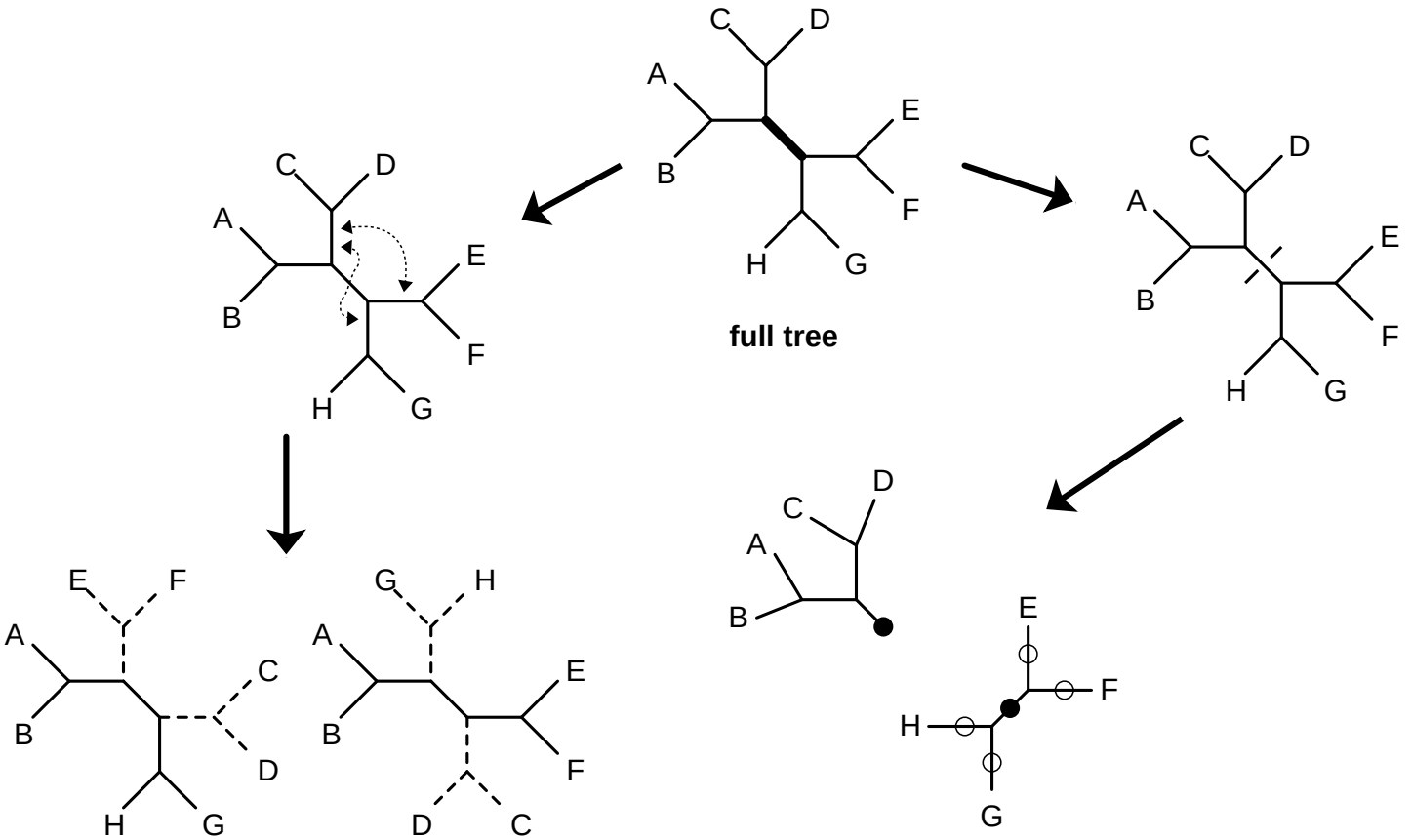
Tree search with topology rearrangements



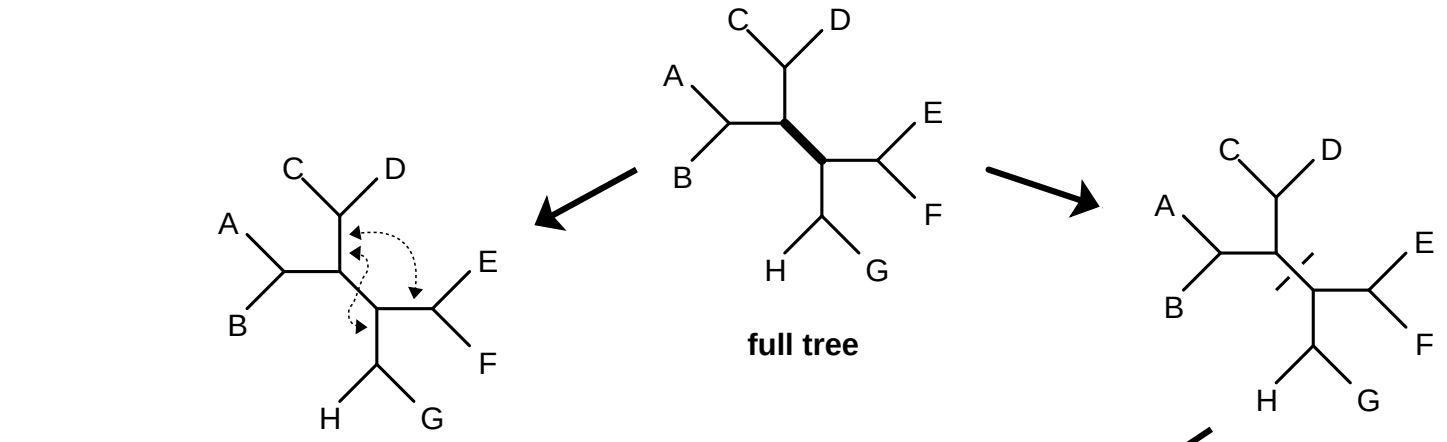
Nearest Neighbor Interchange (NNI)

O(n) NNI trees

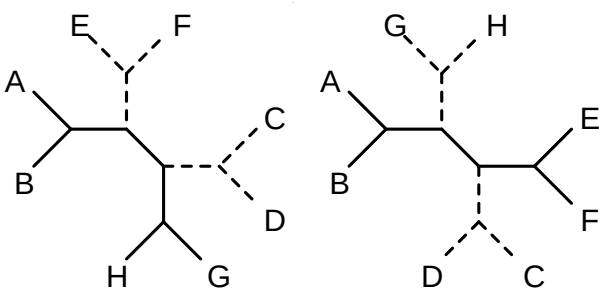
Tree search with topology rearrangements



Tree search with topology rearrangements

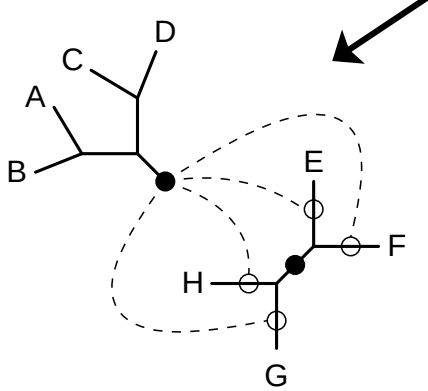


full tree



Nearest Neighbor Interchange (NNI)

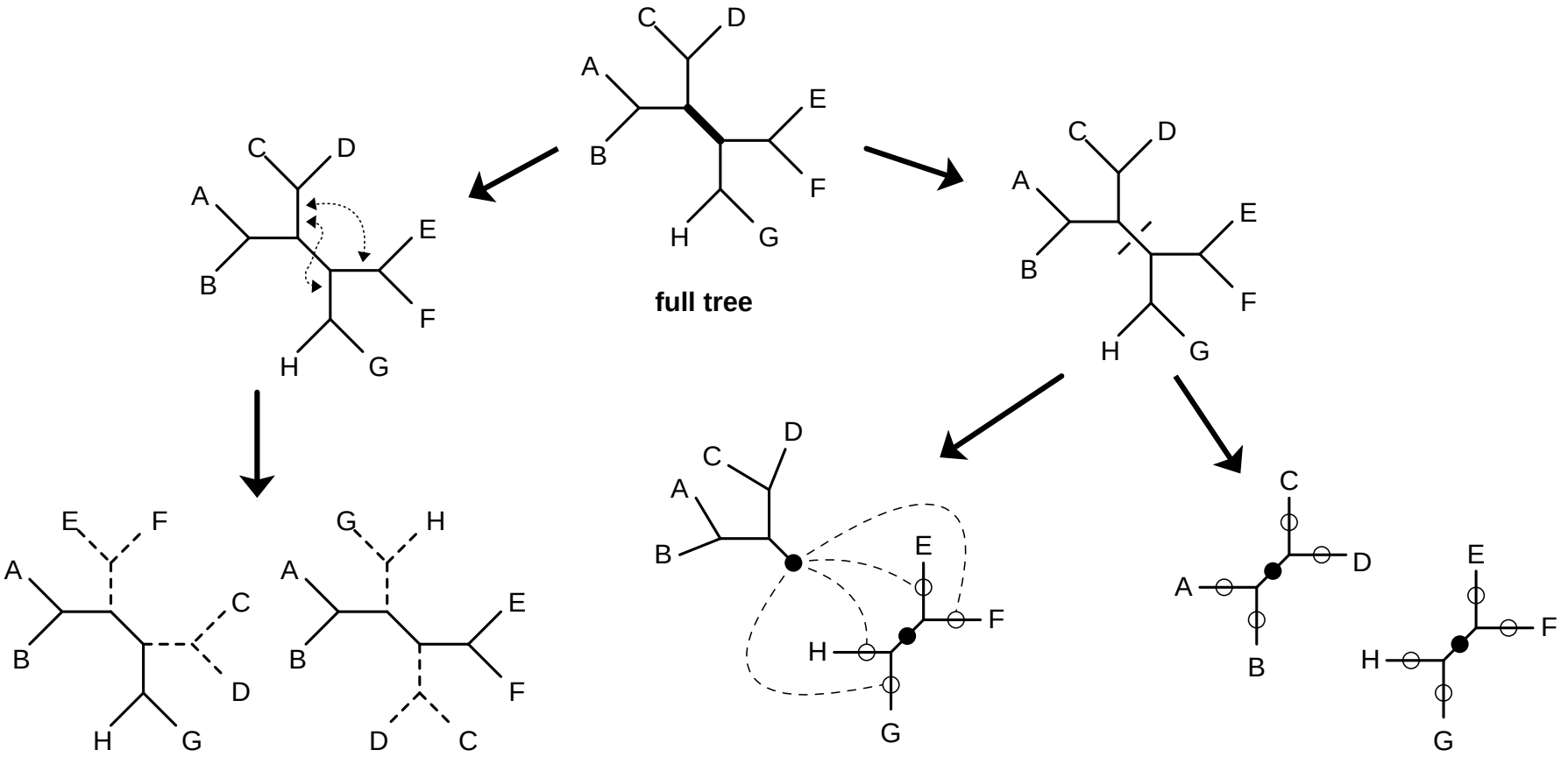
O(n) NNI trees



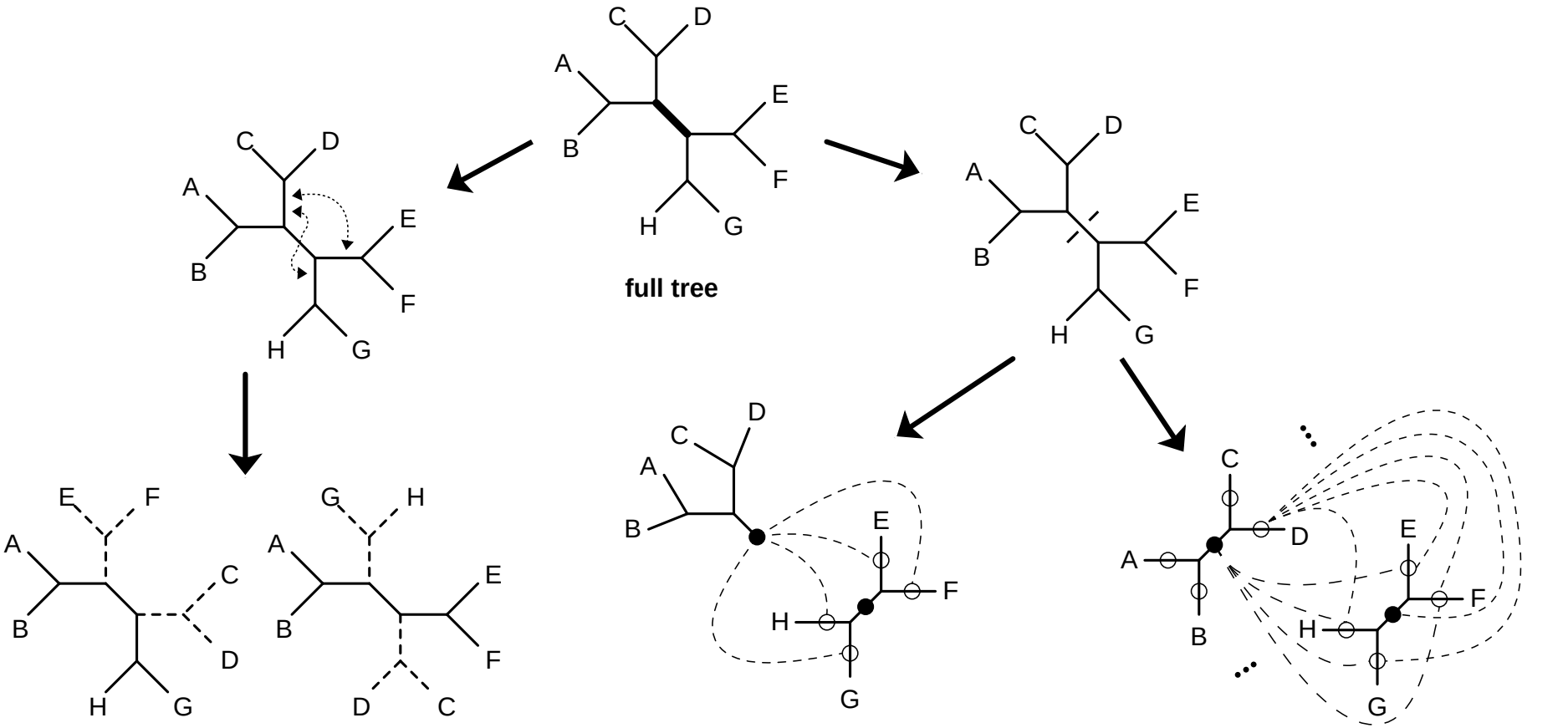
Subtree Pruning + Regrafting (SPR)

$O(n^2)$ SPR trees

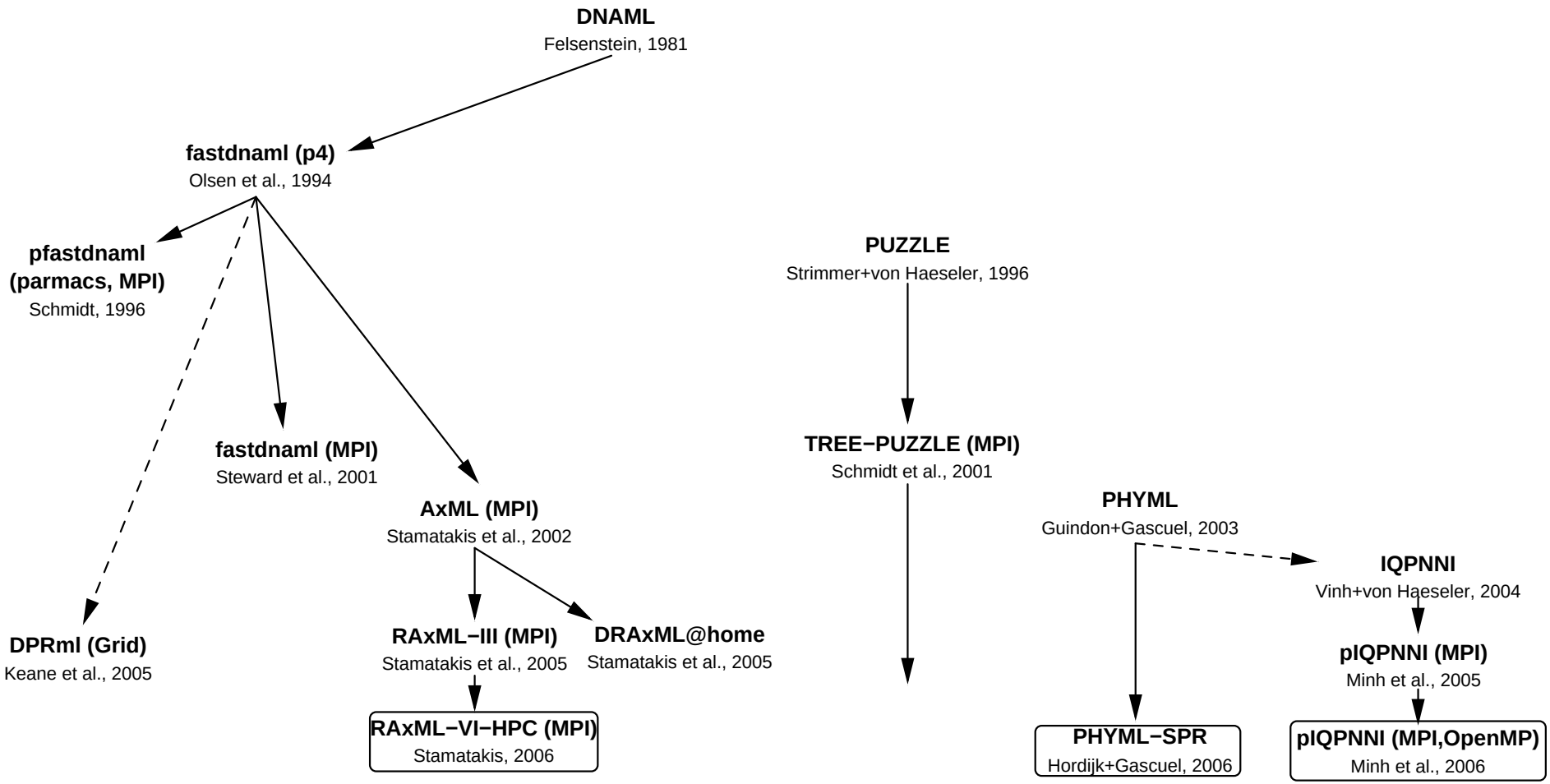
Tree search with topology rearrangements



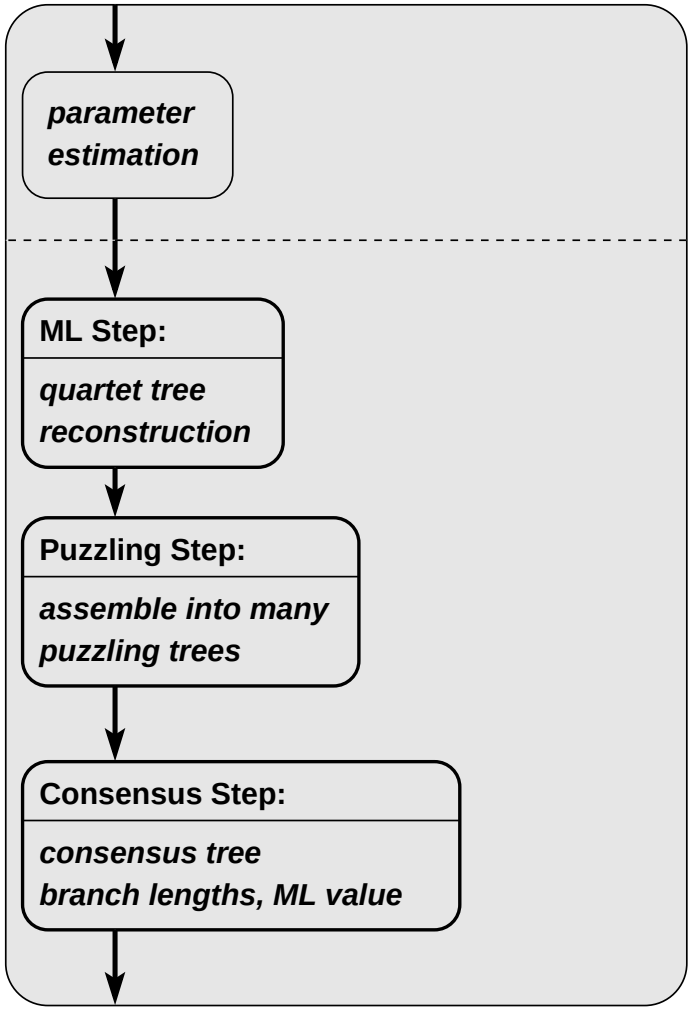
Tree search with topology rearrangements



ML Programs (Parallel and Sequential)



The *Quartet Puzzling* Algorithm

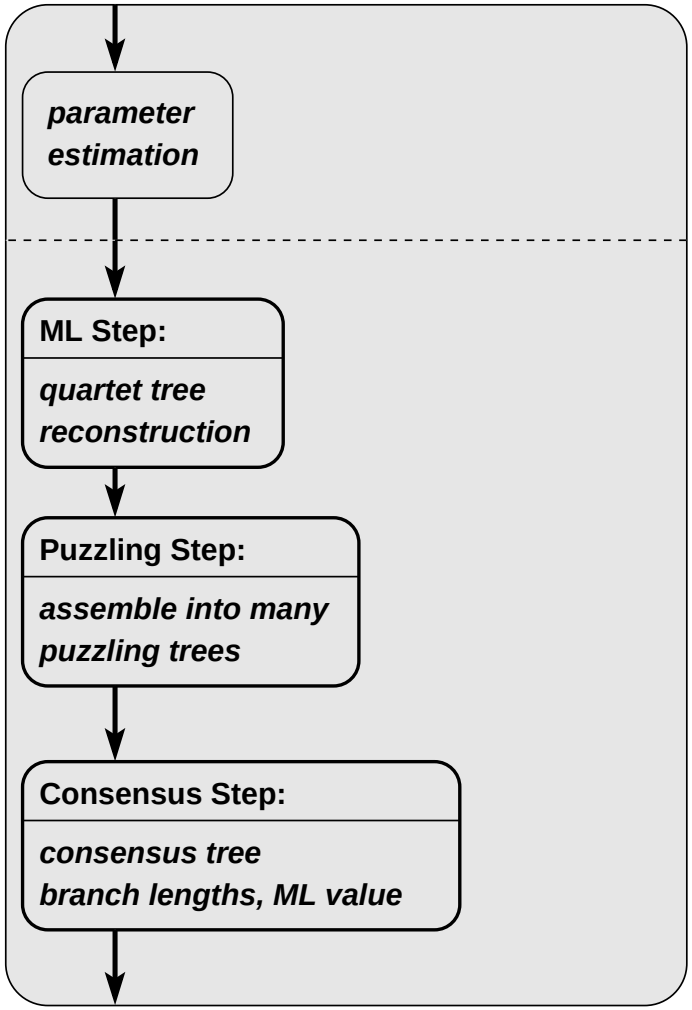


The *Quartet Puzzling* Algorithm

Runtime-Profile

64 sequences, 1000 bp

Parameters:	1.27%
ML Step:	43.15%
Puzzling Step:	55.53%
Consensus Step:	0.05%

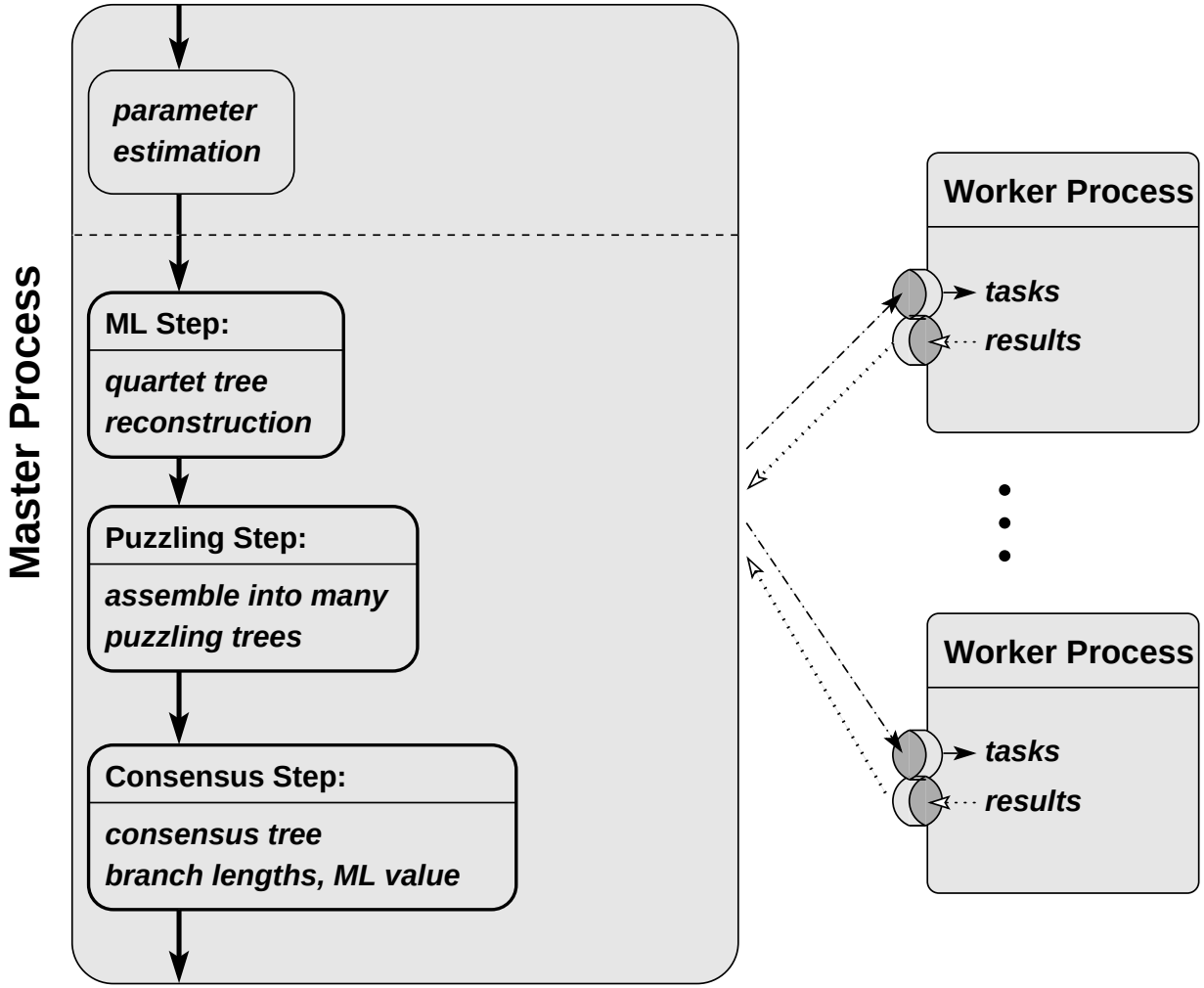


The *Quartet Puzzling* Algorithm

Runtime-Profile

64 sequences, 1000 bp

Parameters:	1.27%
ML Step:	43.15%
Puzzling Step:	55.53%
Consensus Step:	0.05%

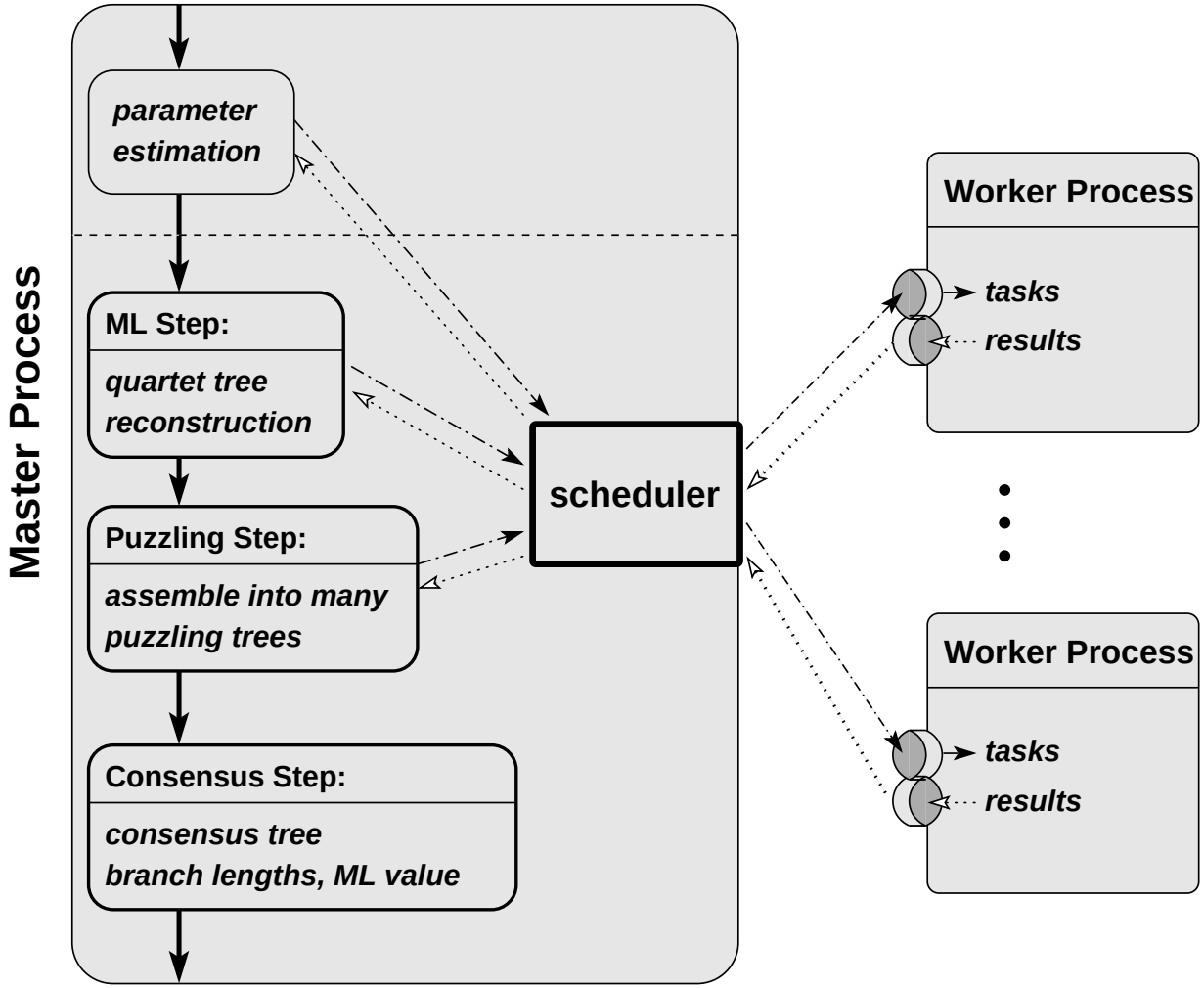


The *Quartet Puzzling* Algorithm

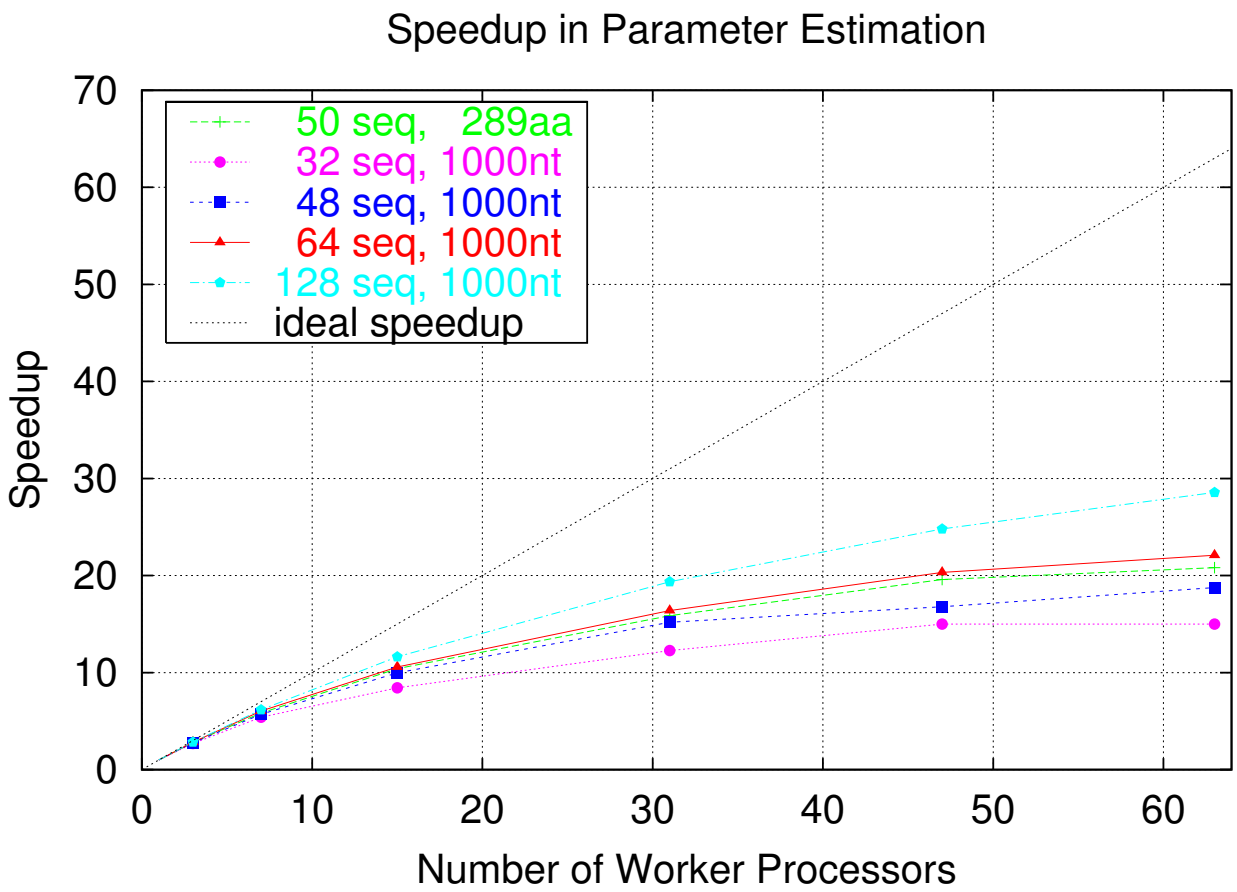
Runtime-Profile

64 sequences, 1000 bp

Parameters:	1.27%
ML Step:	43.15%
Puzzling Step:	55.53%
Consensus Step:	0.05%

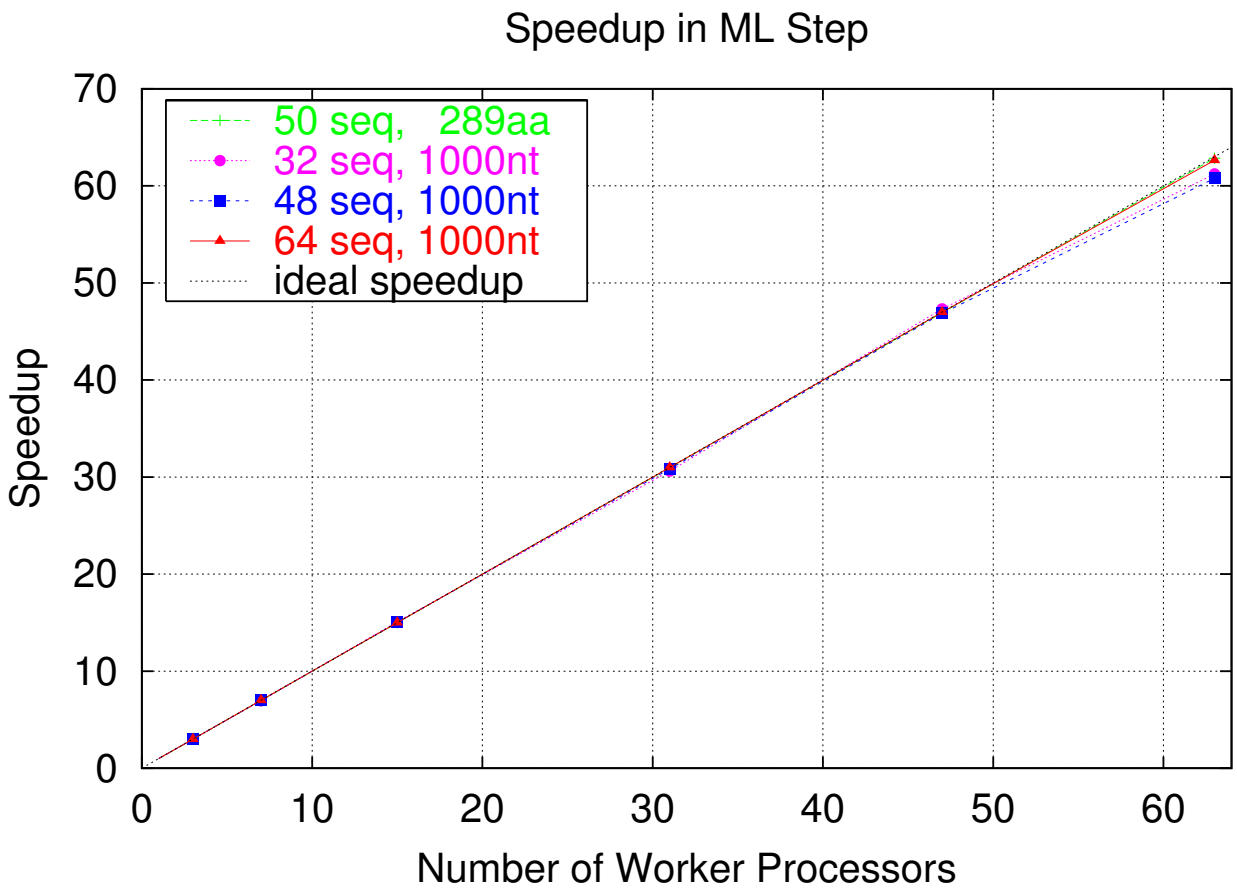


Parallel Parameter Estimation



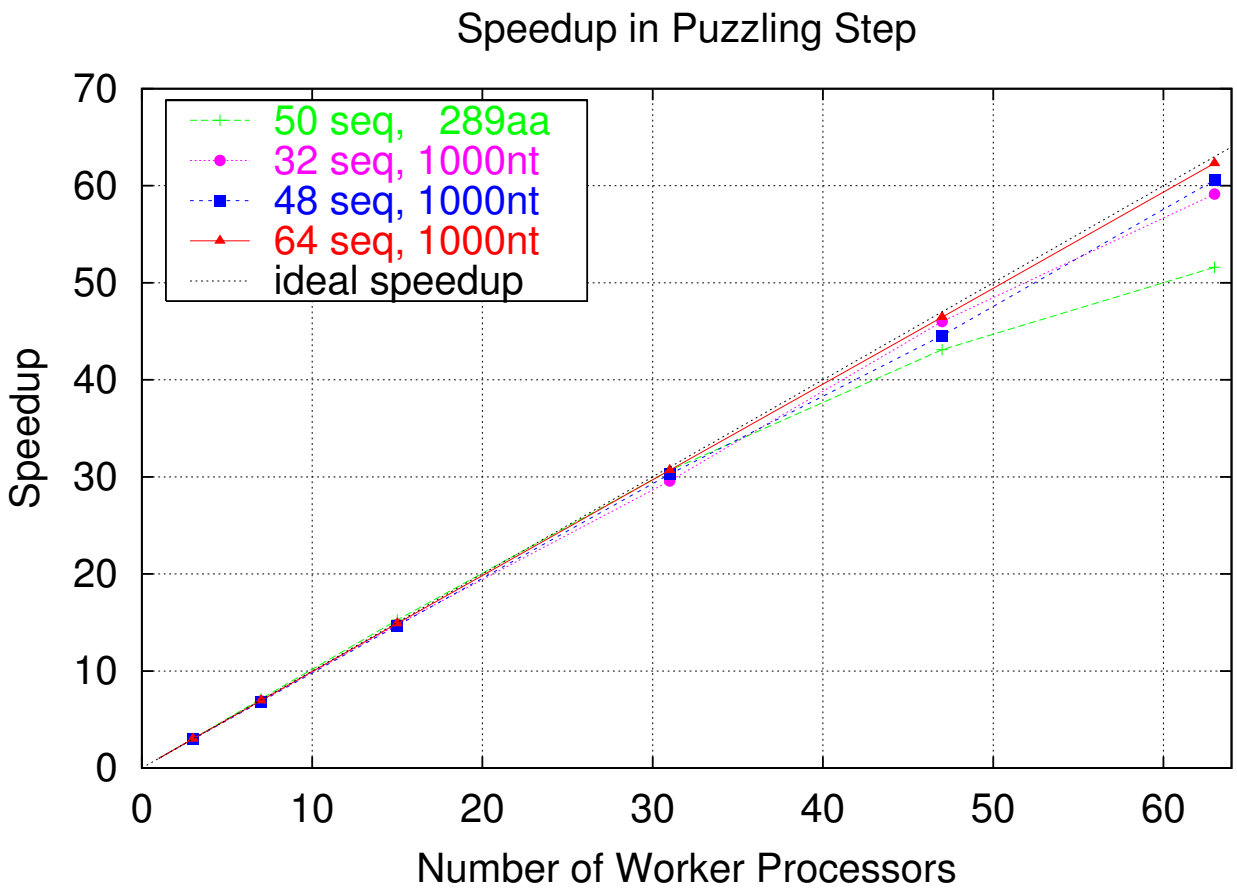
- Scheduling: Static chunking
- Granularity: fine grain
- Tasks: $(n^2 - n)/2$

Parallel ML Step



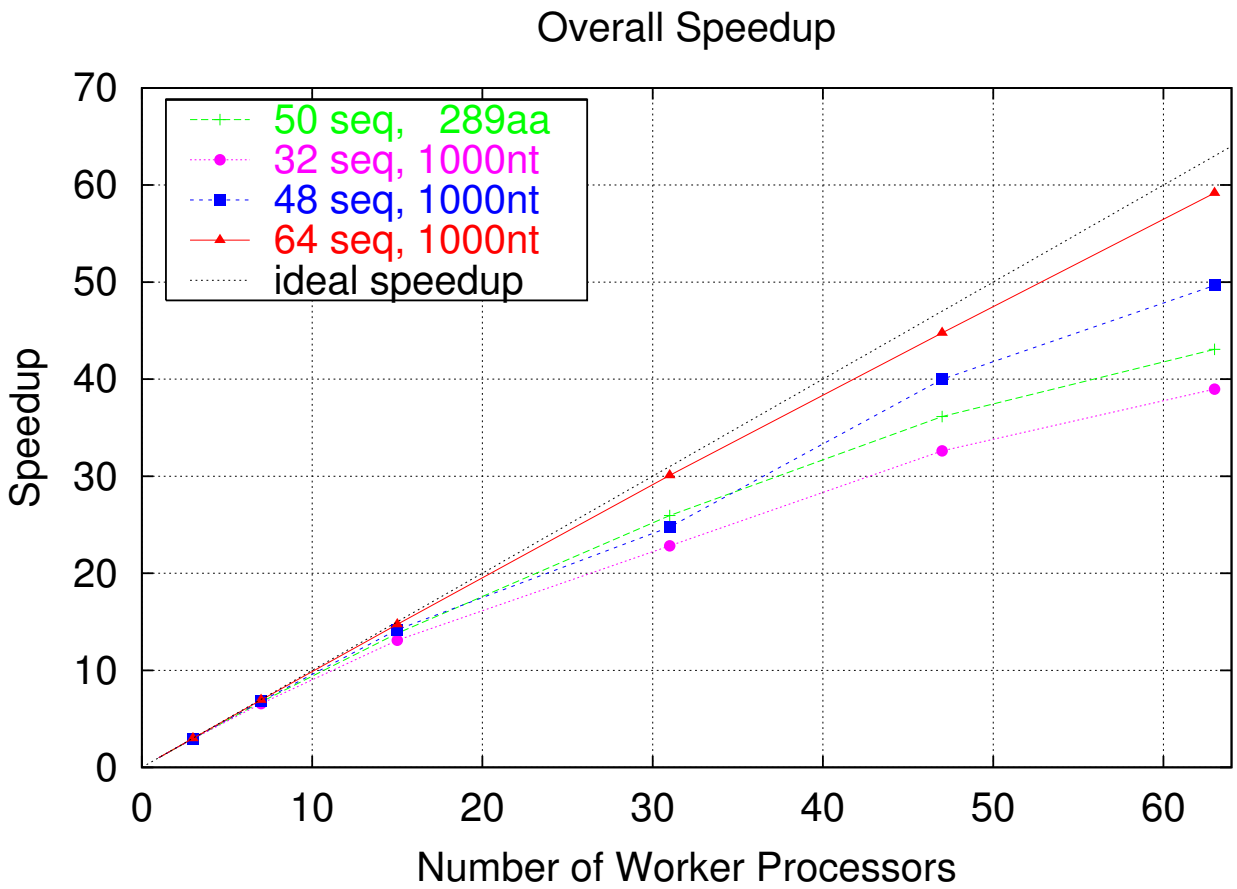
- Scheduling: dynamic, sGSS
- Granularity: coarse grain
- Tasks: $3 \times \binom{n}{4}$

Parallel Puzzling Step

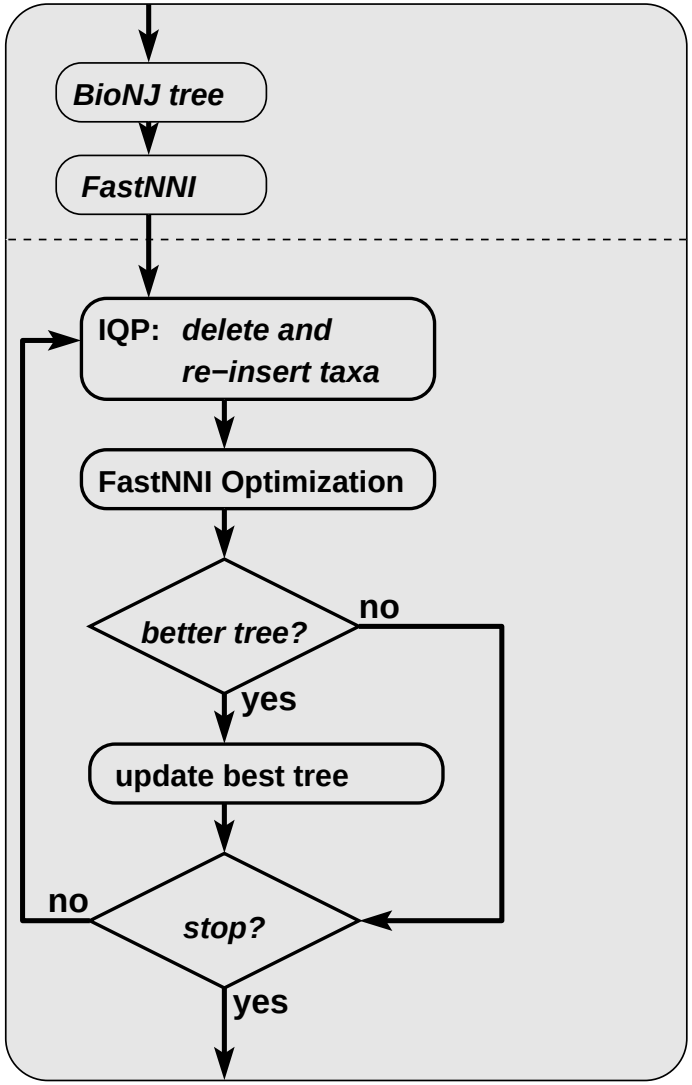


- Scheduling: dynamic, sGSS
- Granularity: coarse grain
- Tasks: 1000 – 25000

Total TREE-PUZZLE speedup



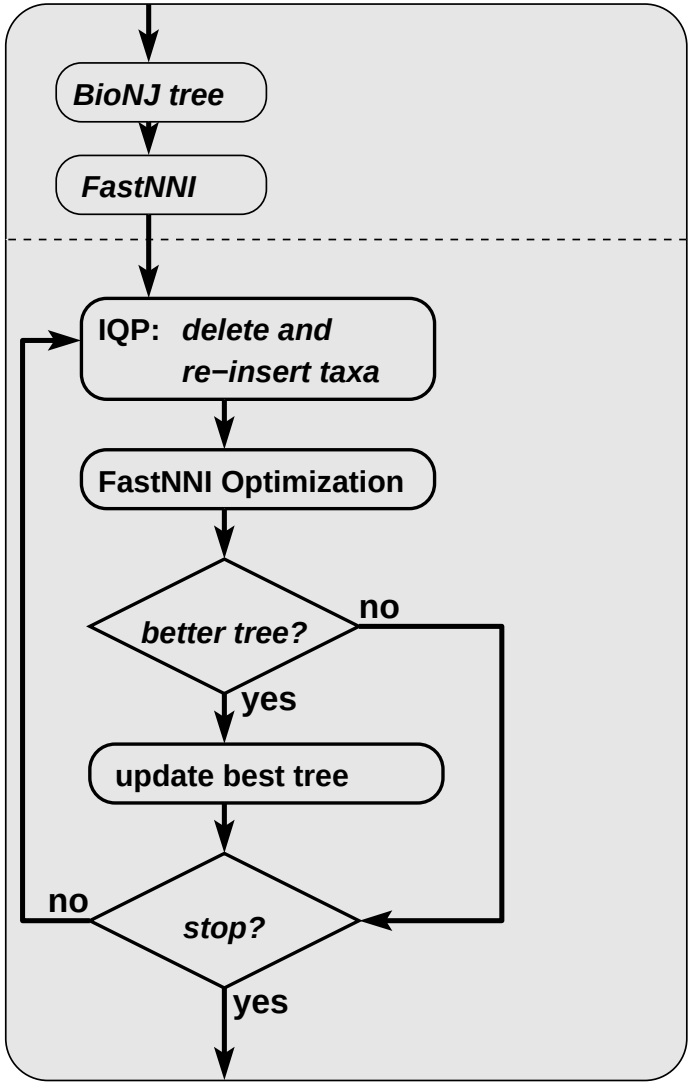
The *IQPNNI* Algorithm



The *IQPNNI* Algorithm

Runtime-Profile

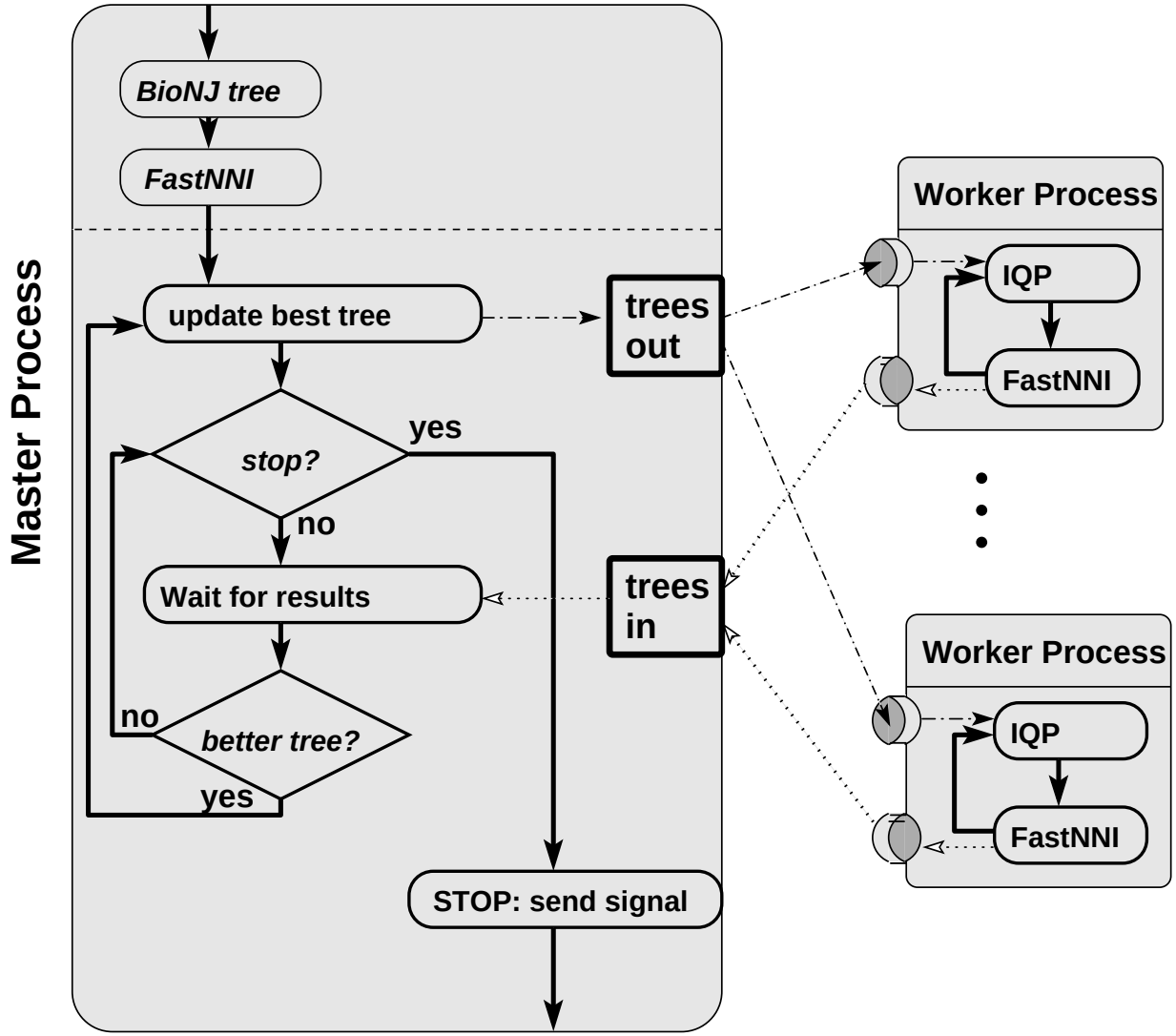
Initial step:	1-9%
Second part:	91-99%



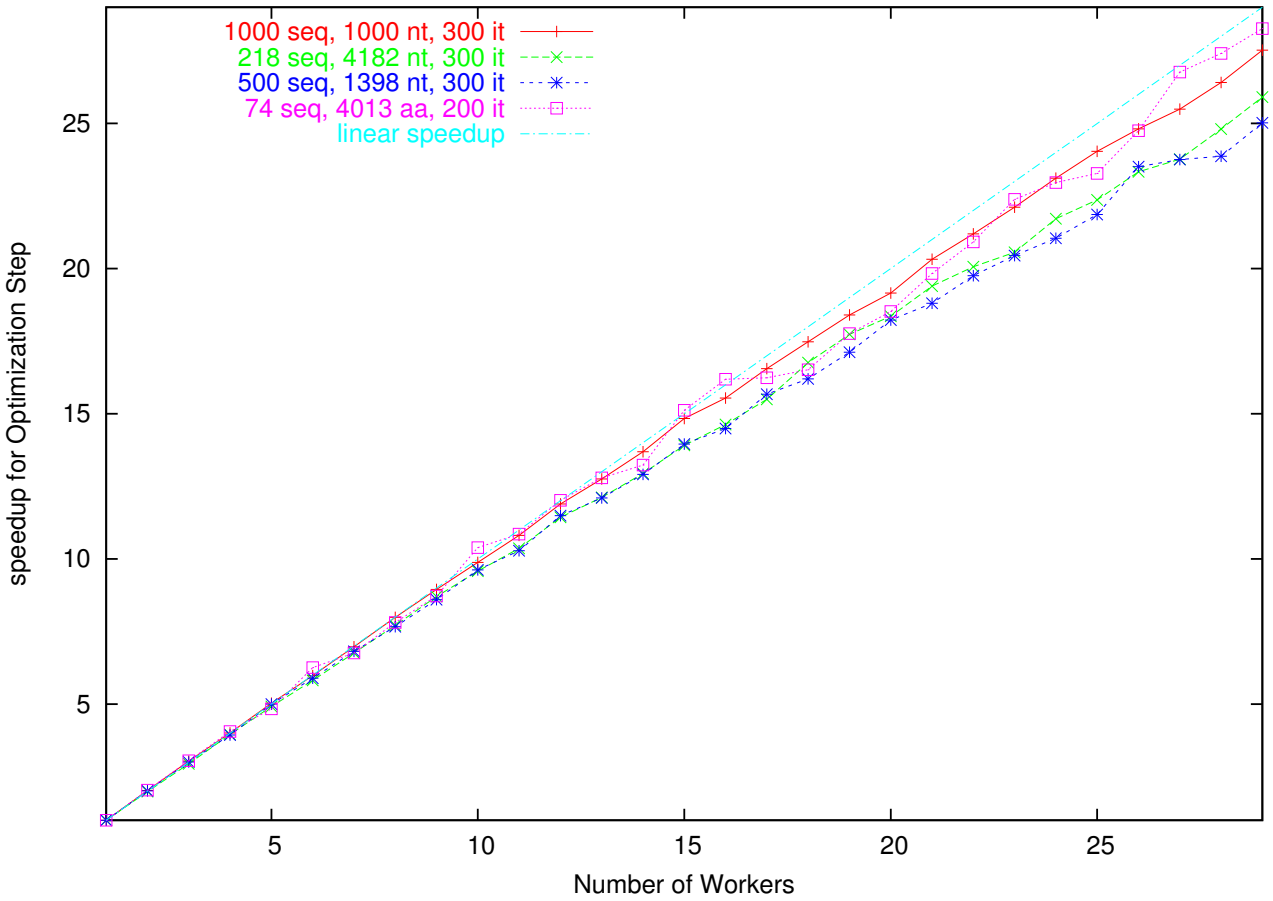
The *IQPNNI* Algorithm

Runtime-Profile

Initial step:	1-9%
Second part:	91-99%

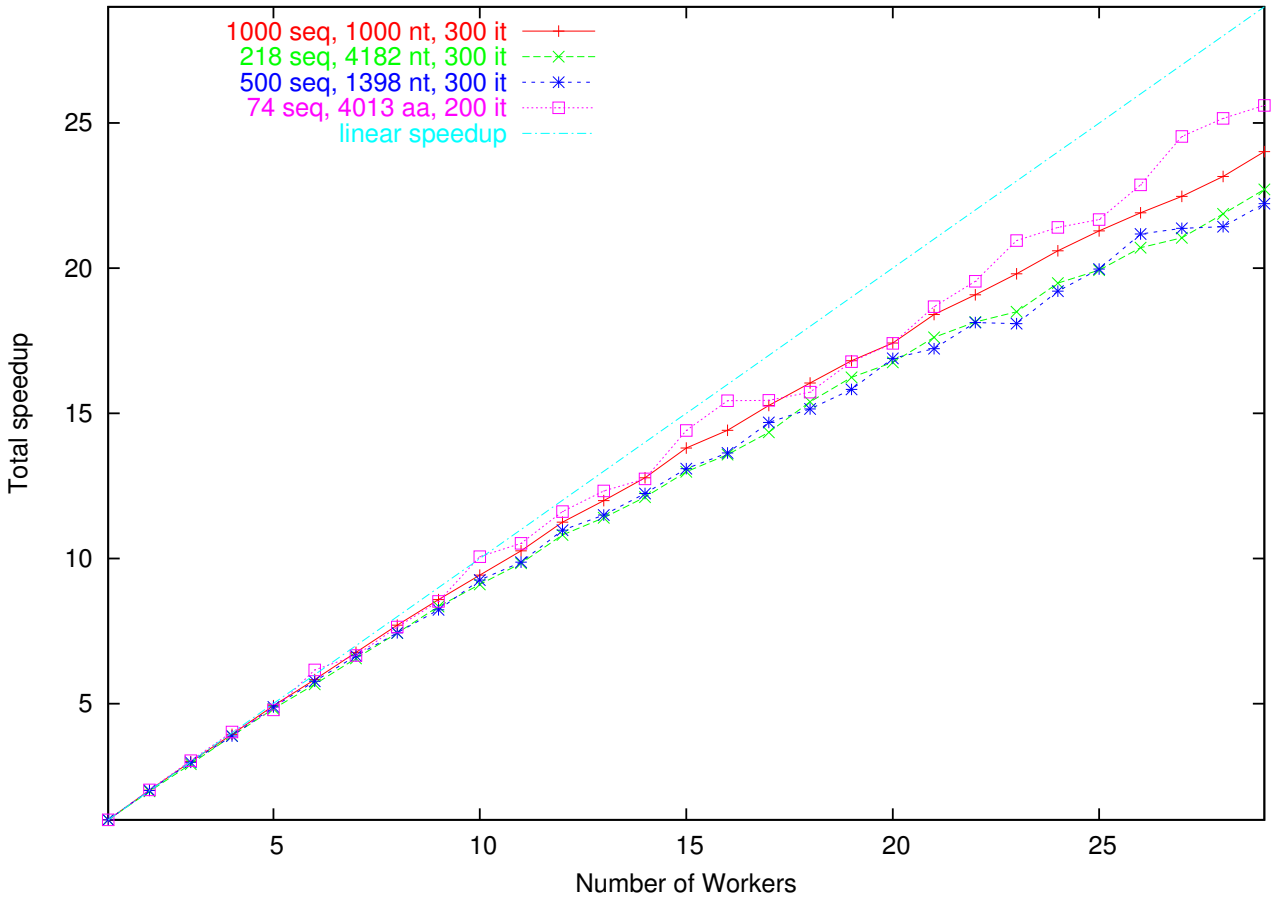


IQPNNI Optimization Step

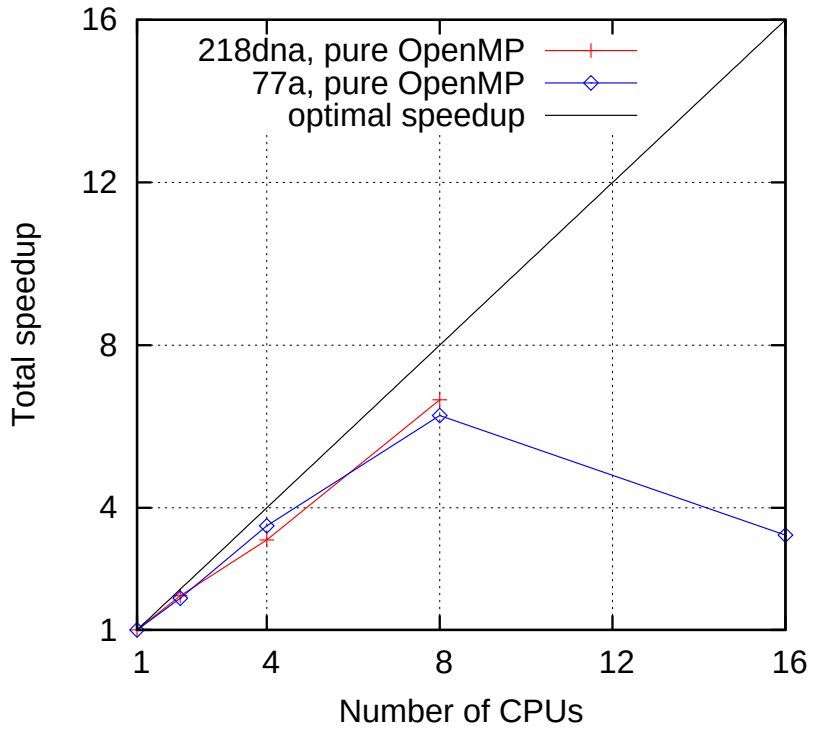


- Scheduling: tasks independent, workers are running in loops
- Granularity: coarse grain
- Tasks: many, at least $2n$

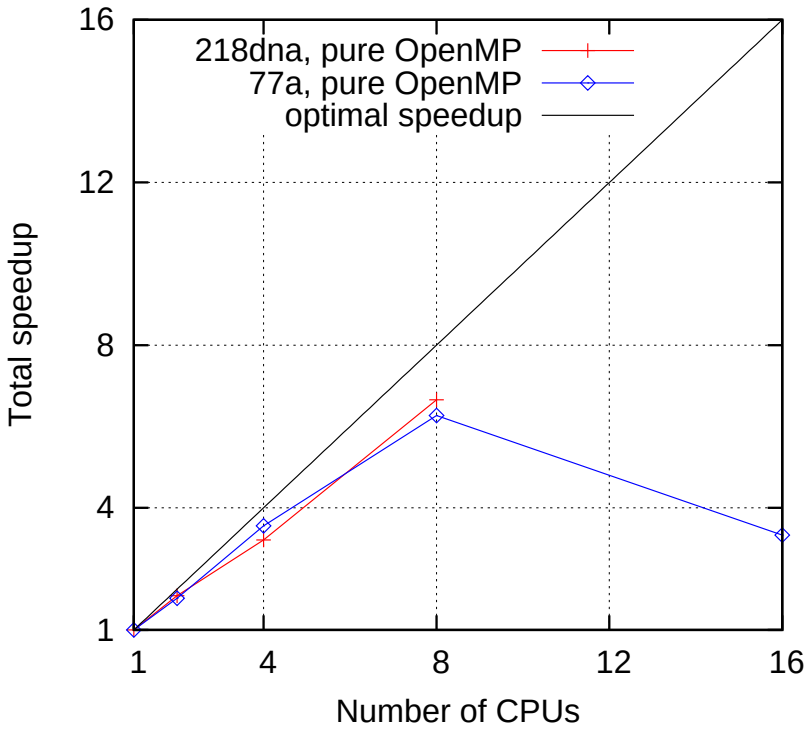
Total IQPNNI speedup

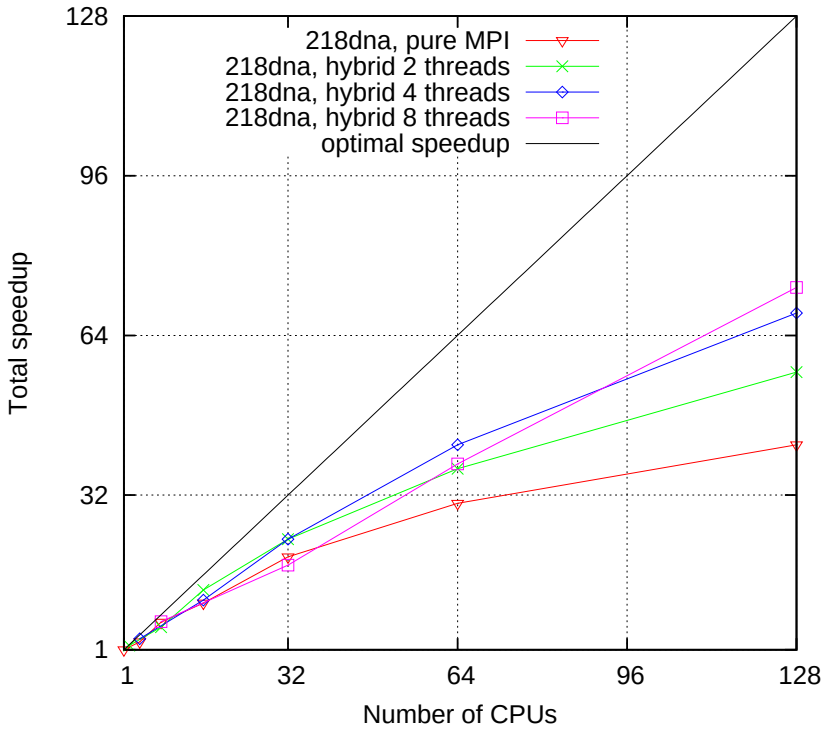


Hybrid MPI-OpenMP Approach to IQPNNI



Hybrid MPI-OpenMP Approach to IQPNNI





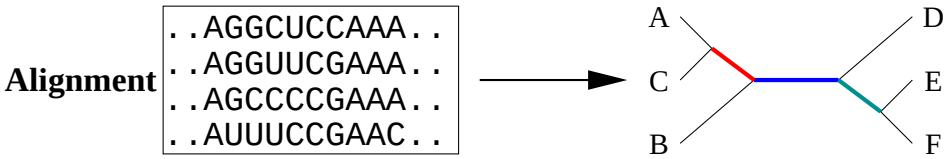
What about reliability of the trees?

The problem with those methods is that:

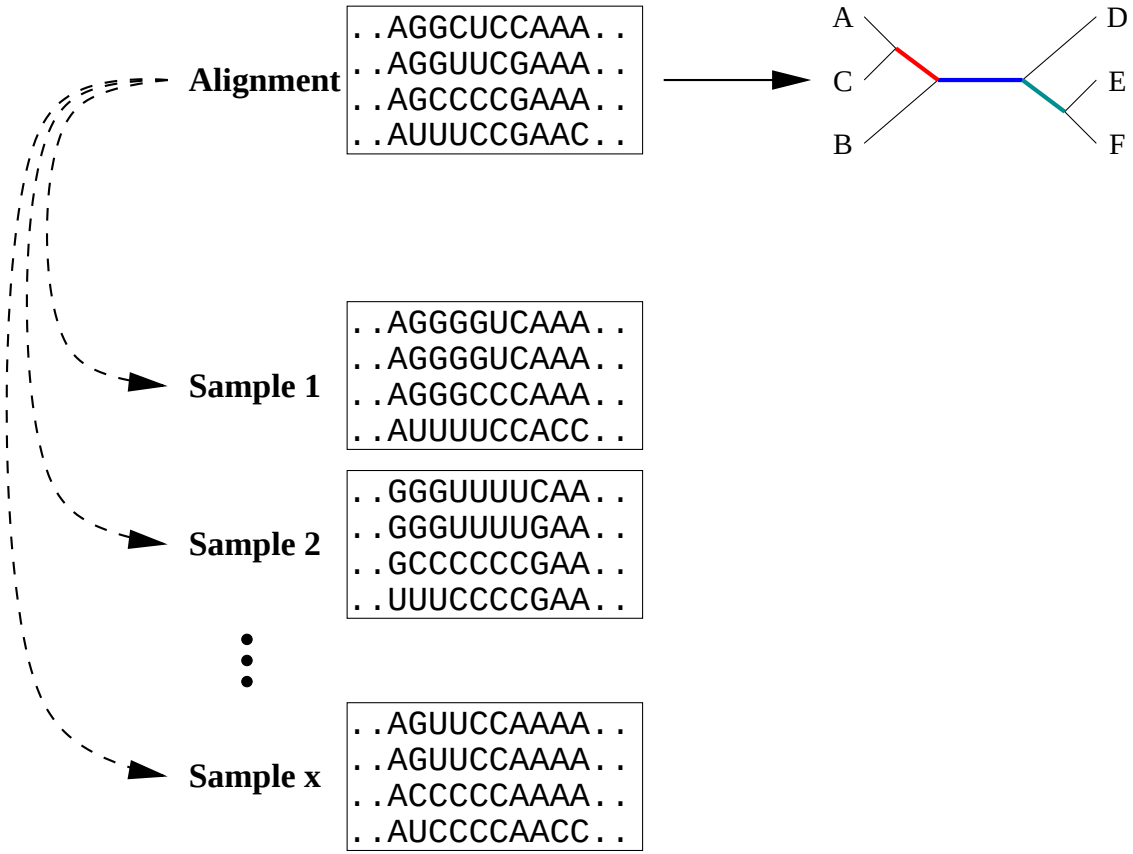
- one only gets a single tree with branch lengths,
- but not measure of reliability of the groupings in such a tree.

The usual way to overcome this is to draw bootstrap samples and to re-construct trees from these.

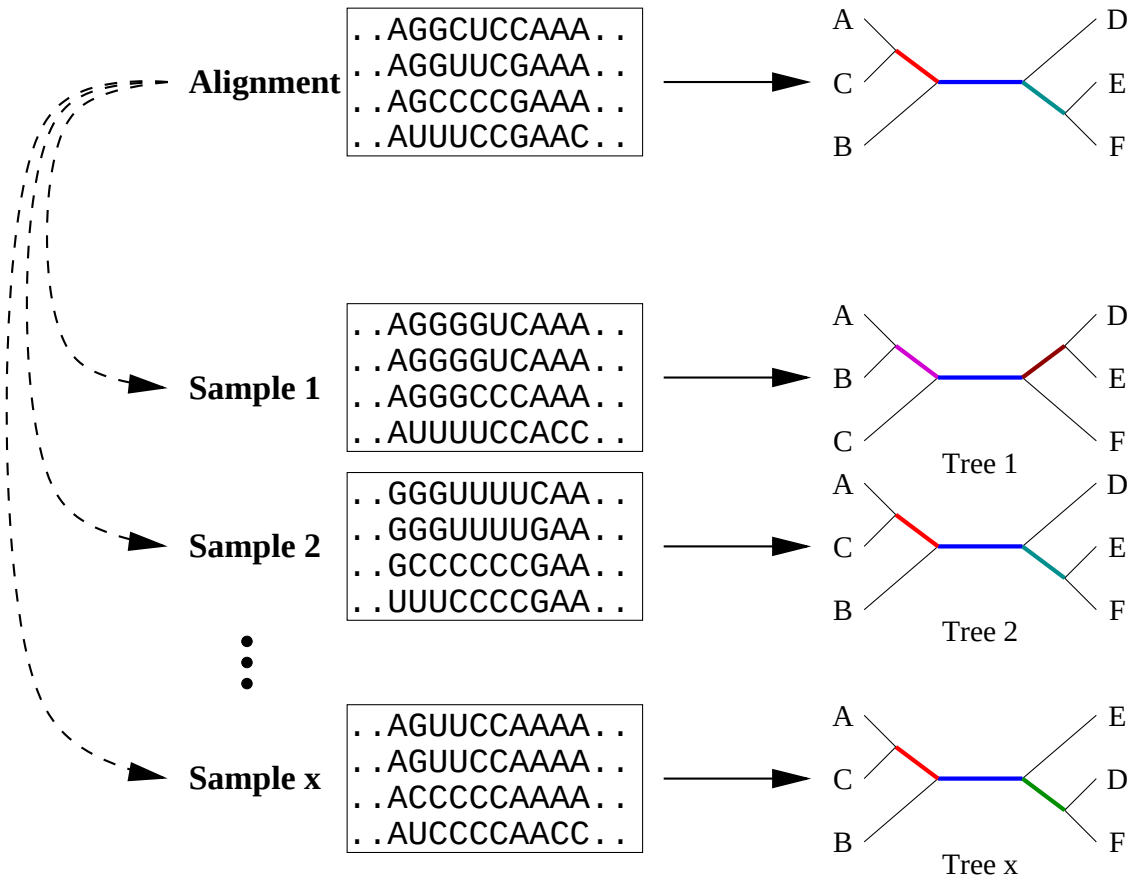
Bootstrapping using clusters or Grids



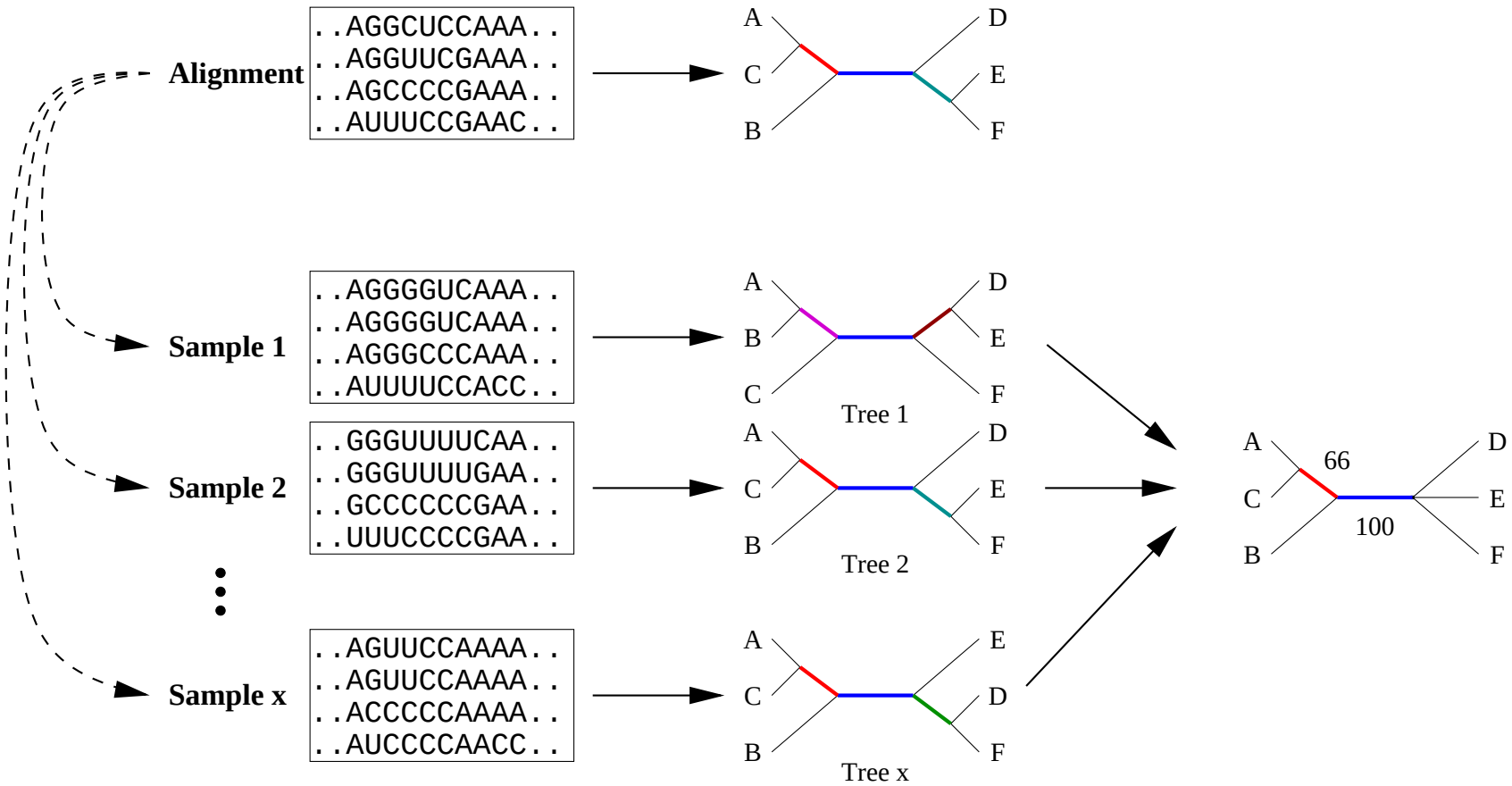
Bootstrapping using clusters or Grids



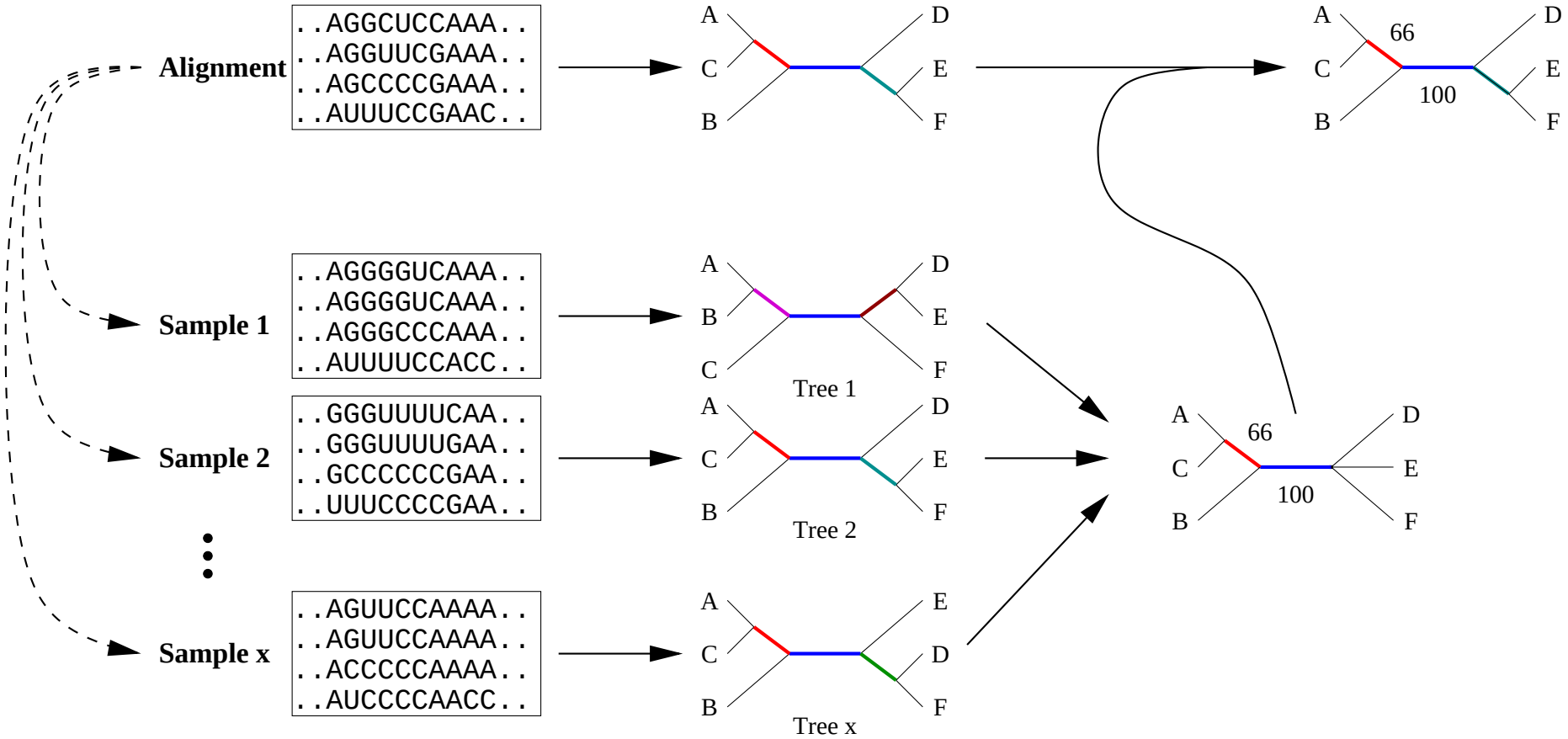
Bootstrapping using clusters or Grids



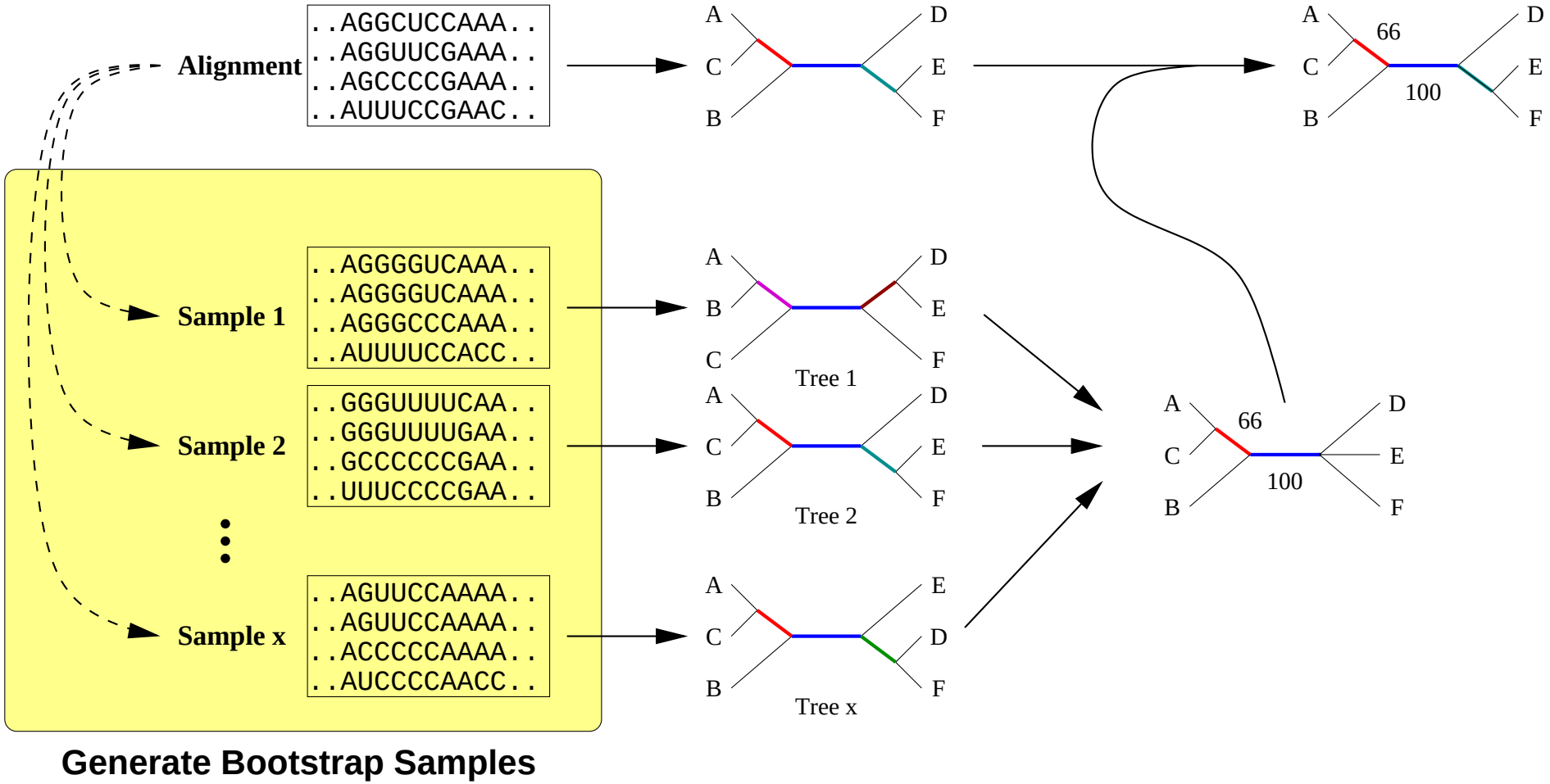
Bootstrapping using clusters or Grids



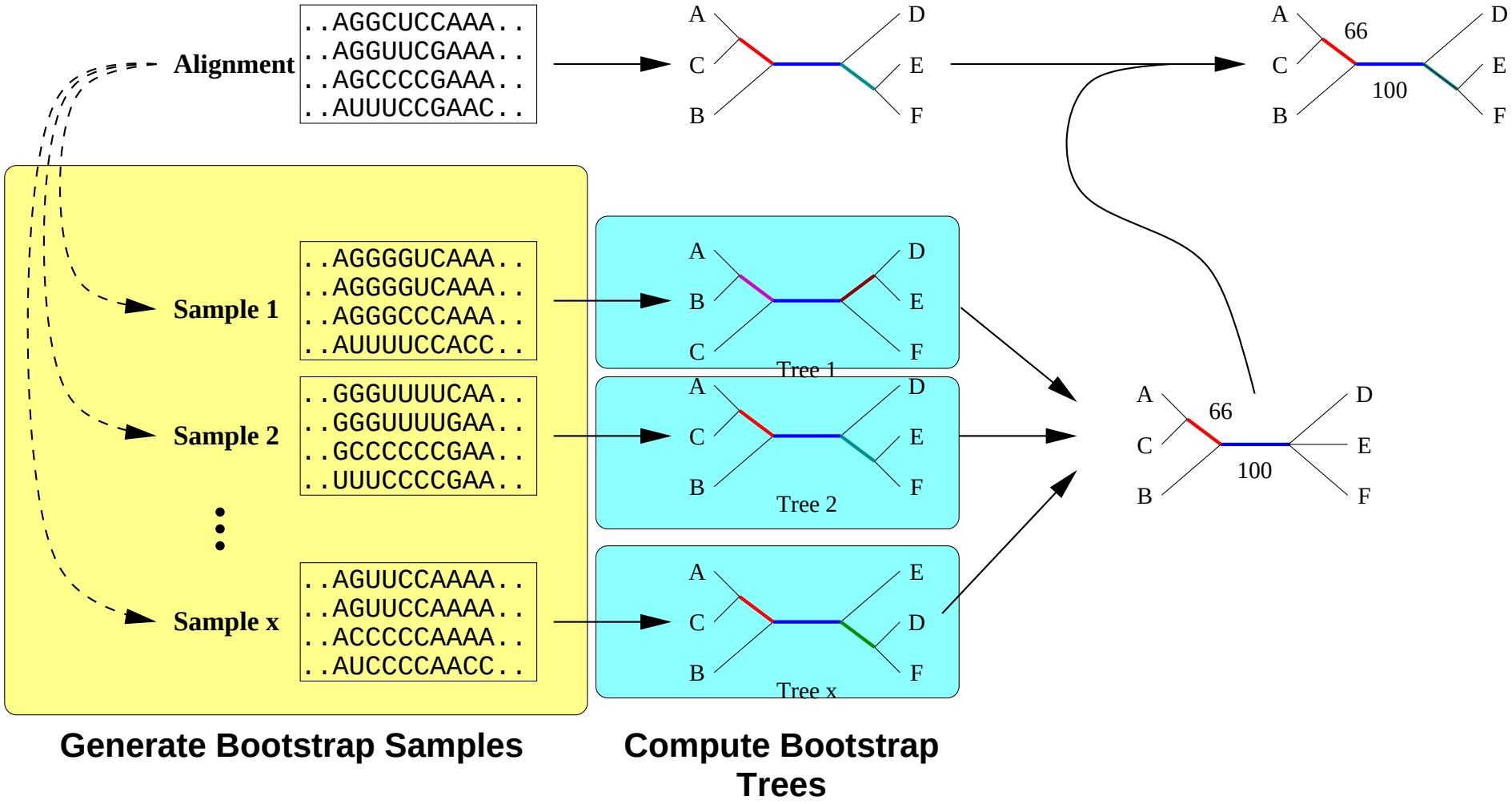
Bootstrapping using clusters or Grids



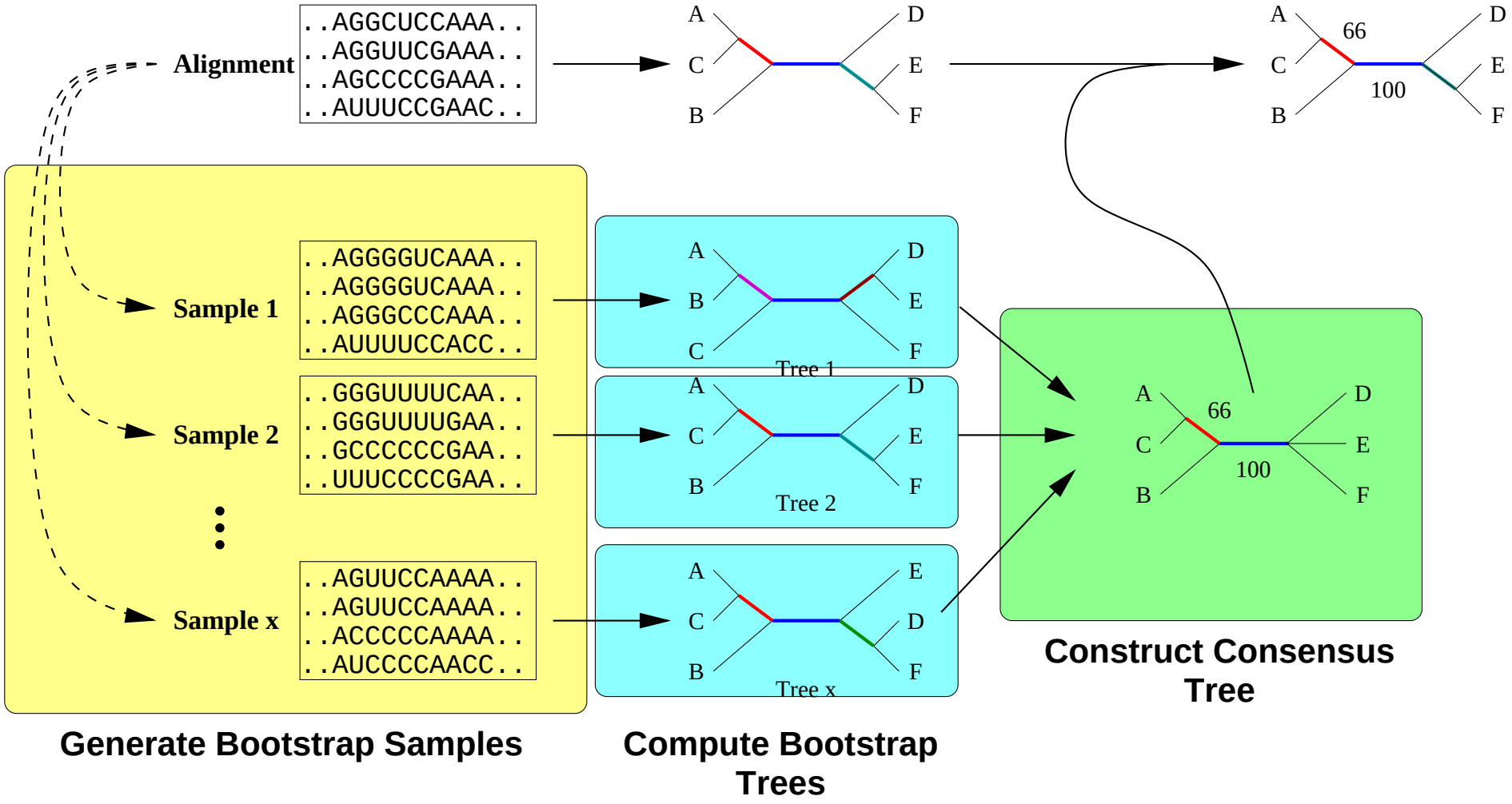
Bootstrapping using clusters or Grids



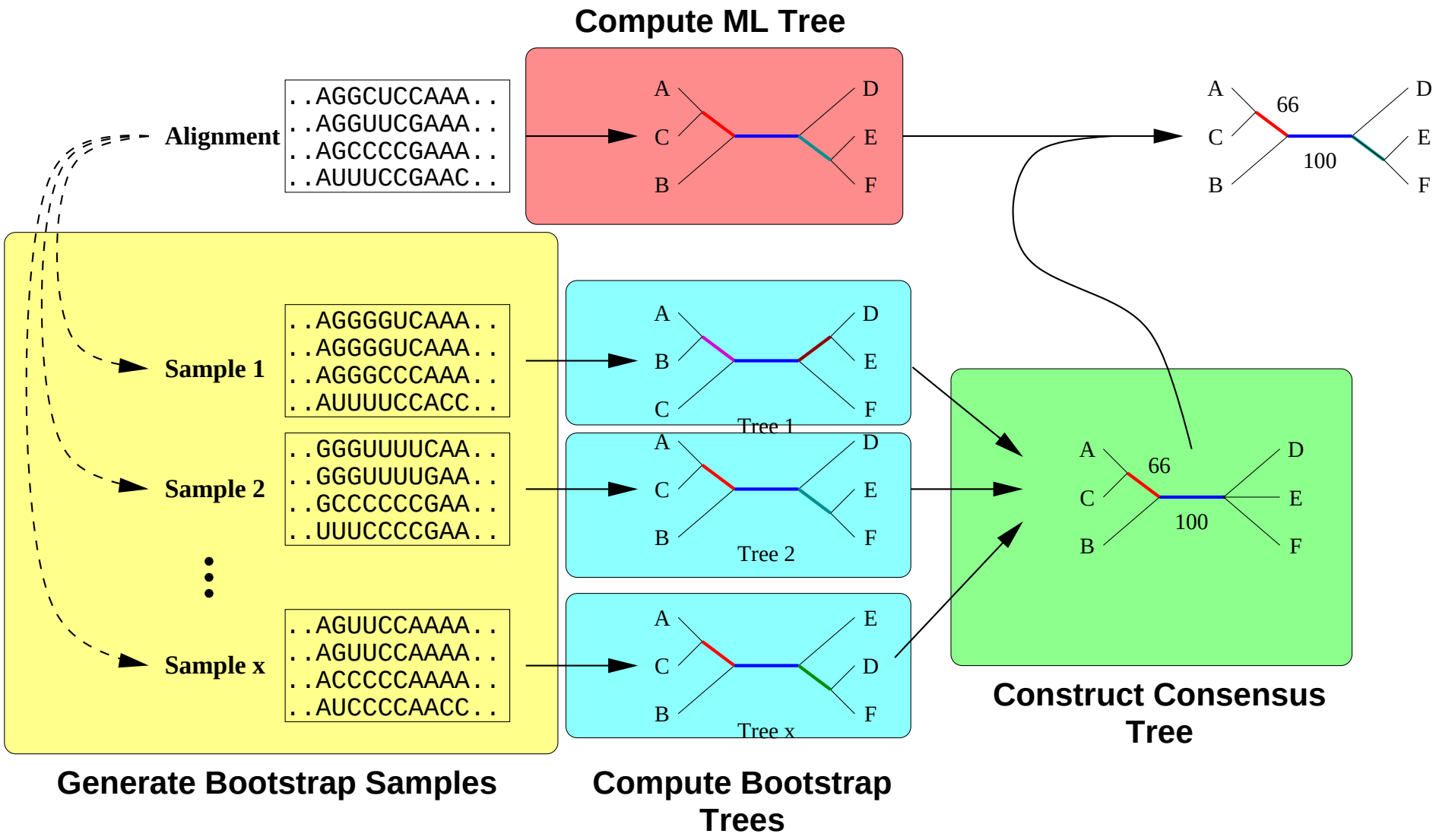
Bootstrapping using clusters or Grids



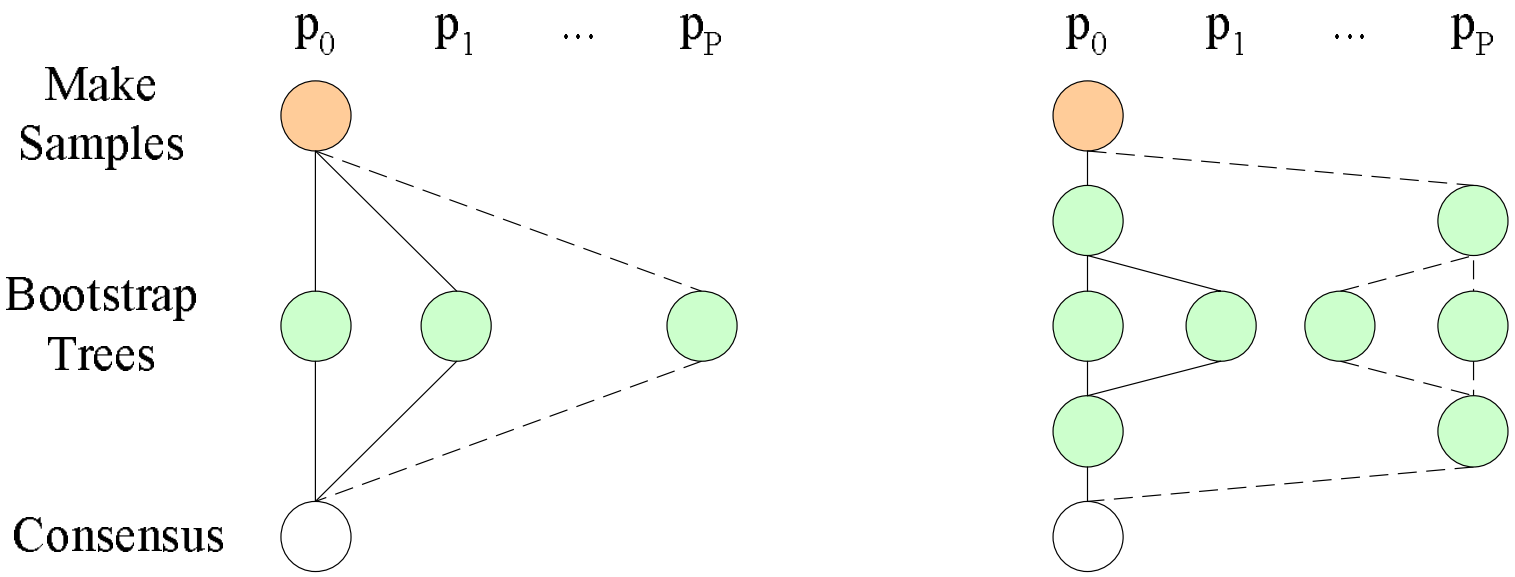
Bootstrapping using clusters or Grids



Bootstrapping using clusters or Grids



Future work



Parallel tasks within the workflow:

- Dynamically increasing the amount of parallel tasks in the workflow,
- should lead to a better load balancing,
- and should keep the CPUs equally busy until the end.
- The increased communication overhead should be made up for by the better usage of the hardware and the reduced running time.

Thanks

- Center for Integrative Bioinformatics Vienna (CIBIV), MFPL
  - BUI Quang Minh (CIBIV)
  - Arndt von HAESELER (CIBIV)
- Computer Science Dept., University of Vienna
  - Joachim ZOTTL (CPAMMS, UniVie)
  - Wilfried GANSTERER (CPAMMS, UniVie)
- Funding:
  - Wiener Wissenschafts- und Technologiefonds (WWTF), Vienna
  - Deutsche Forschungsgemeinschaft (DFG), Germany