Phylogenetic Inference

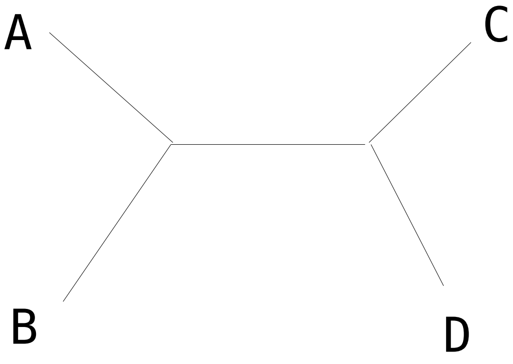
Alexandros Stamatakis[1,2,3]
1. Institute of Computer Science, Foundation for Research and Technology - Hellas
2. Heidelberg Institute for Theoretical Studies
3. Institute of Theoretical Informatics, Karlsruhe Institute of Technology
www.biocomp.gr (Crete lab)

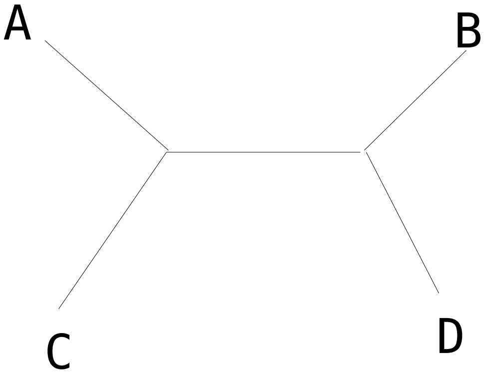
www.exelixis-lab.org (Heidelberg lab)

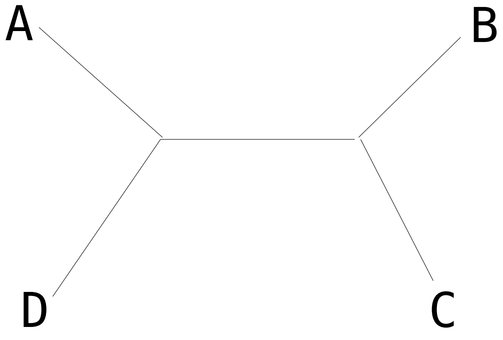
Plan for Today

- Phylogenetic Inference
- Phylogenetic scoring criteria
- Phylogenetic search algorithms

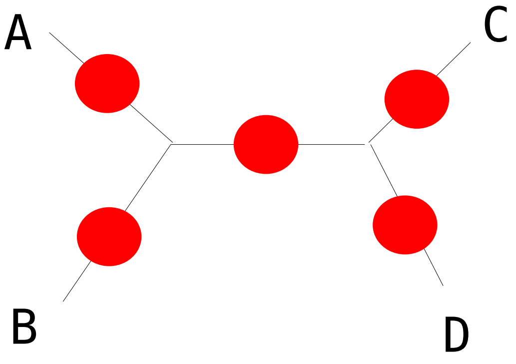
How many unrooted 4-taxon trees exist?

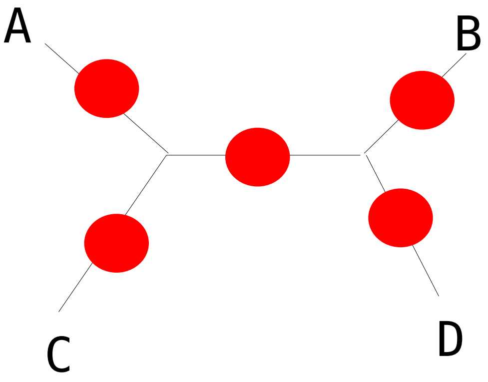


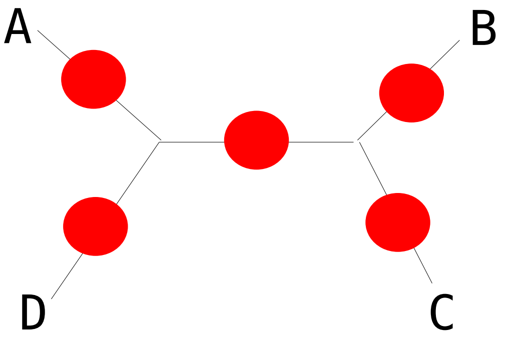




How many rooted 4-taxon trees exist?



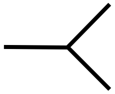




Tree Counts

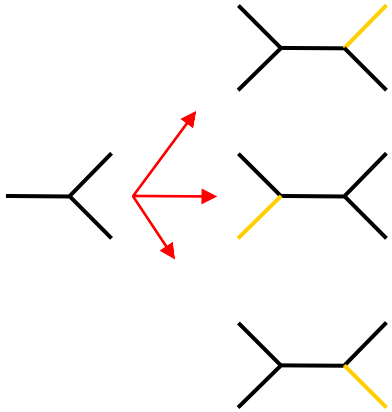
- Unrooted binary trees
  - *4* taxa → *3* distinct trees
  - A tree with *n* taxa has *n-2* inner nodes
  - And *2n-3* branches
- Rooted binary trees
  - *4* taxa → *3* unrooted trees * *5* branches each (rooting points) = *15* trees
  - *n-1* inner nodes
  - *2n-2* branches

The number of trees



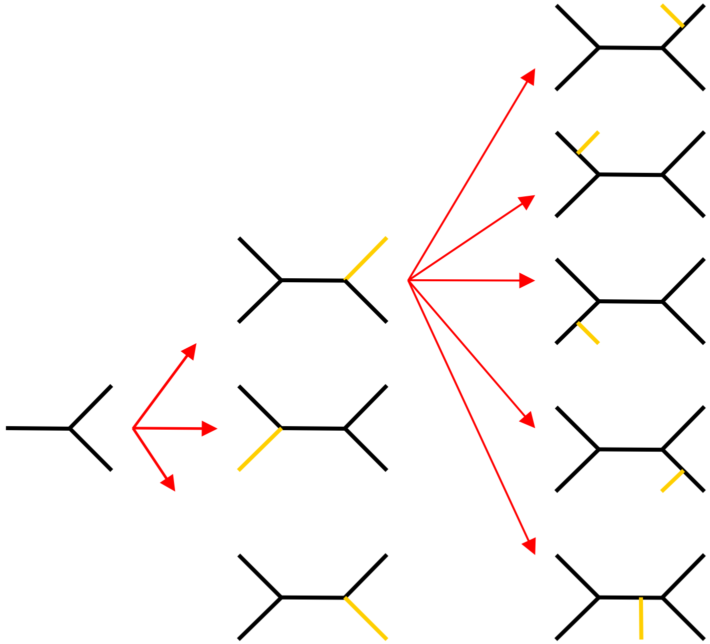
3 taxa = 1 tree

The number of trees



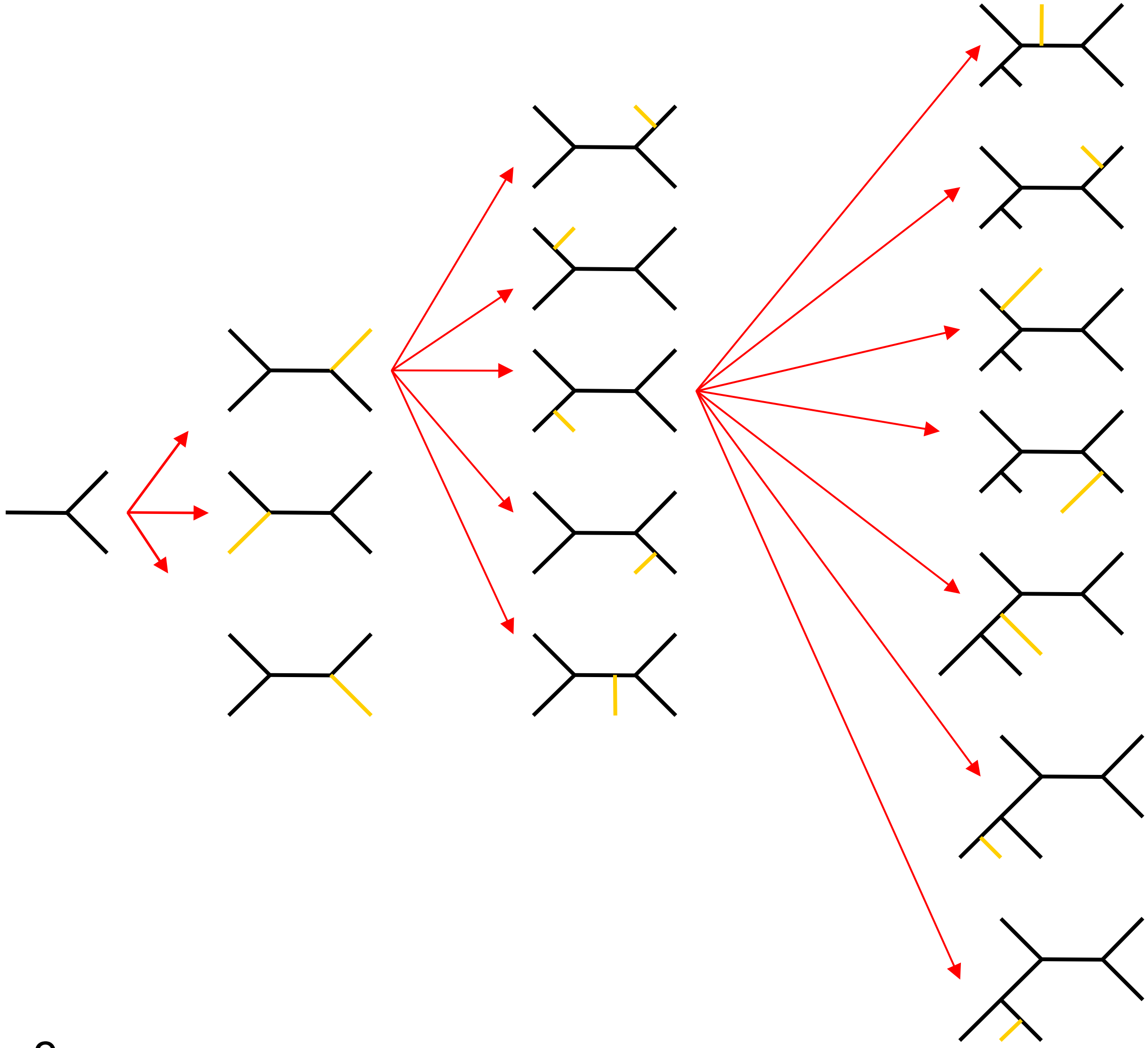
4 taxa: 3 trees
u: # trees of size 4-1 := 1
v: # branches in a tree of size 4-1 := 3
Number of unrooted binary trees with 4 taxa: u * v = 3

The number of trees



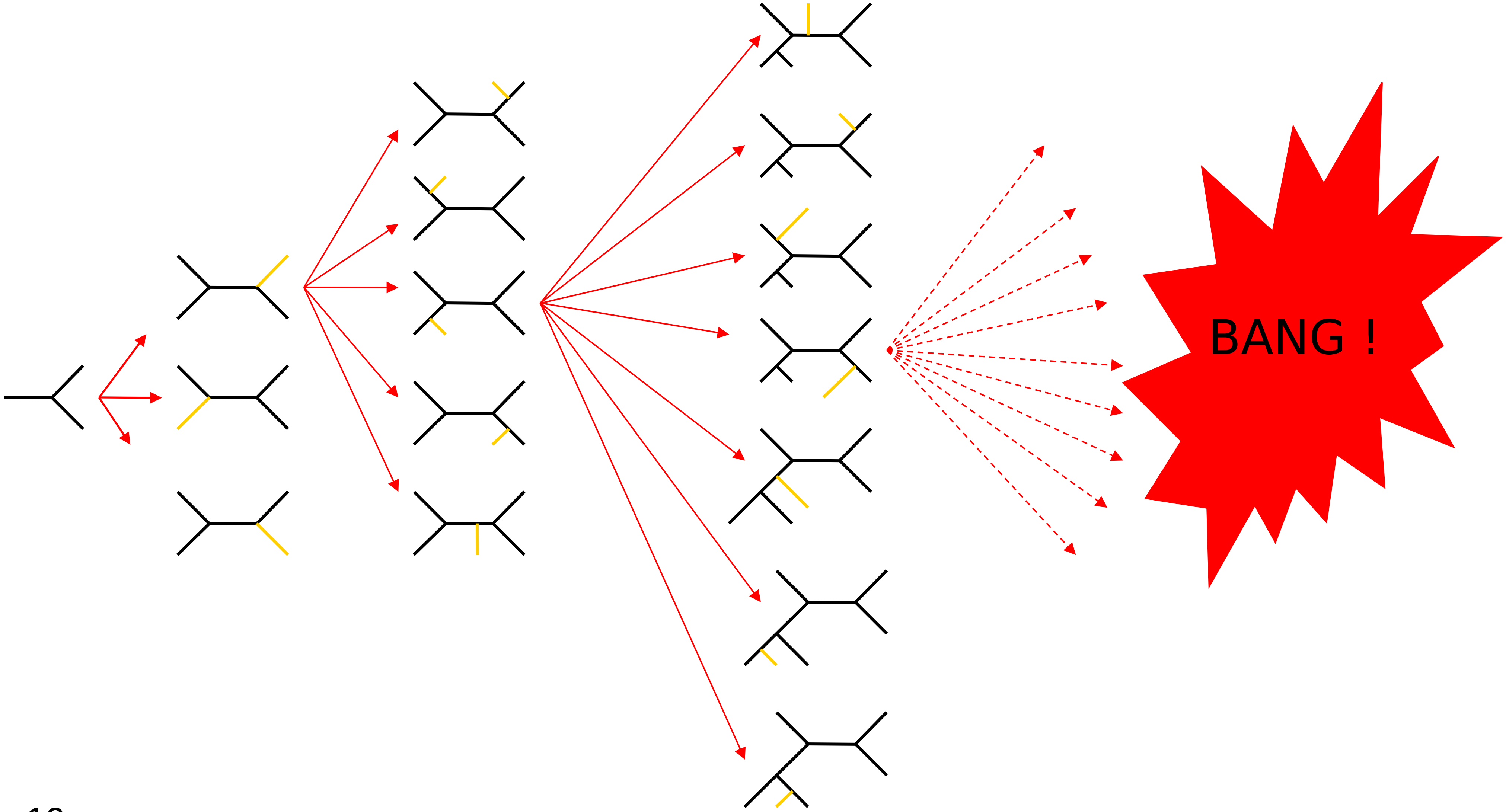
5 taxa: 15 trees
u = 3
v = 5
Number of unrooted trees with 5 taxa: 3 * 5 = 15

The number of trees



6 taxa: 105 trees
u = 15
v = 7
u * v = 105

The number of trees explodes!



Equation for the number of unrooted trees

- Simple proof via induction

$$\prod_{i=3}^{n}(2i-5)$$

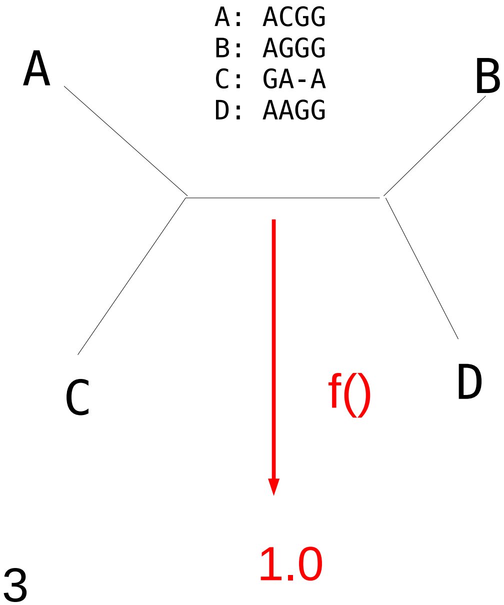
- The number of rooted trees for *n* taxa simply is the number of unrooted trees for *n+1* taxa
- The additional ($n+1^{th}$) taxon represents all possible rootings for all unrooted trees with *n* taxa

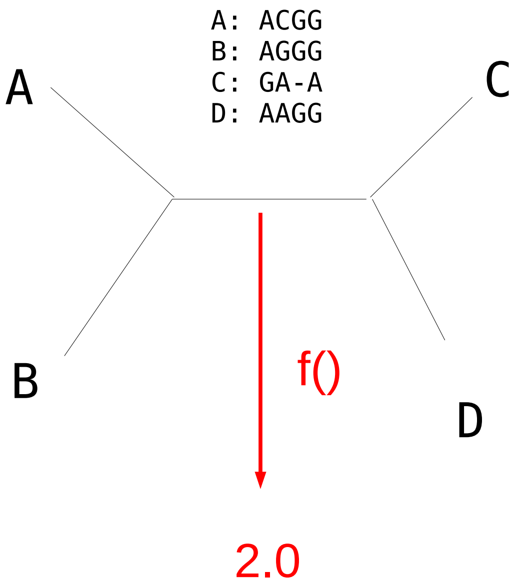
trees with 2000 tips

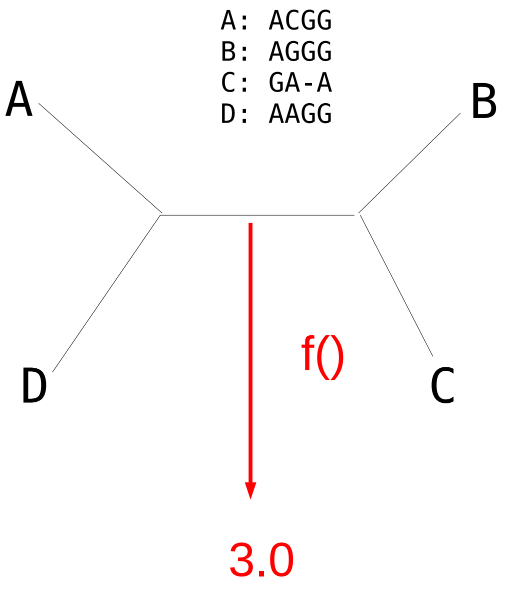
```
stamatak@exelixis:~/Desktop/GIT/TreeCounter$ ./treeCounter -n 2000


GNU GPL tree number calculator released June 2011 by Alexandros Stamatakis

Number of unrooted binary trees for 2000 taxa: 30049638174211656151632910065681814981377232074237013089504954043012636525258308210827685996688247000464352735214265634288295
8915023446000631493969130632970436056184861877465482277991223536809233455563199910834597693126756525012899867433187752811401960991631522367030609121735709762379847705467667
7795324797182614385273338226727784250737252849916669687584403510579587020686505817687044666318123742901021438506432471360934491667021135969756940300666252646479269124551031
4942366195542824118277625114848758254581227914289801132648902674033761294712745767036267579086843169660718609847941818865957214557044744572288661729053583520744253688123124
0106613156948861960941195646736200342575241335277575085829161096422575727699767991408283343210161327401652830993803904592327690690035972919709940739349563486203899010742687
2822975974655377102257672676842858011877224950106218117340523208265397342962227352536590515865631383272031119841987467599738646318290320383252308597997992216101227215780805
24814583120684401676062393060009711616729715504728487799634337531348994230372437347879131989085953764070134849446113877572576952408702461720107874297380462275052545706689372
3194182064407068918840038705902897721975164544959758216621306205064617761099485663734168183584989329076993382067801052437284614924034229611551826097782286191926720712951895
8936009959130974233072316382518428110330571017441156884305131865877544376308500311451110723837039707465182232040406154708273078629957549331031275208616700660791298014262230
05651235227180638195093358726517286235890205200161443617560756542864714221266130044348070840675015892476731663415395405750744749994909831496473031080411401891849735912811228
3787740498848340562102420566424463860093899650857429619472690543015281237526510965815284699797036792171129035568098180791695879516141592810495281798558472925344478644244359
9808531537204796814969465991768614533701051985928577157482455943377242369582576242663016946320482495182255939287403177623433881048604630975191556923871167513095213415098816
7154643078623526062377864068386804246902527491139319276802611515990582603886733172930713673903403618637463980605764836474670274446727880885337074254421922726677747003329403
32010382880351126890262551830967919483586789293701637681753048206338943871497931152353698229625111116307148294599211620803302684762013335690441089668145436150905155877581167
9770012563912151116237444170497371704604029481104114822286466131918821997571383368352072526055202769823974613218495249264897050790398360256255606289852288839561357874156576
64889992608732866126306425432602489792291135600716405739845163752452433769437558573847255455564397599604255914640112221144755235573176239973057747183956531217416532295986675
9012941161239240722093250369673124884491553759210650656015416720774159236240868667675348286512964888739059707578802473393463470848159011639772797747480417316268700916728735
6121642268468160683198959801260376485615312781611689587215123123308760063473381097253118423339640390937378395066835578735307886358646400563299499490631187424029092779272693
3003224453775957972248734568915114585570783850541681667667425811301958063621907500790295031088209097271748136436989473971079932777700676301730617566538739726037777173008441
3439405123669055544932486165082539957795036326704947844293498853172797348177797146567175151178876396434069332458076346110734214328195049909680874027397688914704517472055543
8969396668742601477241894693212902453317334188286773194653544113302100866575081713240342647580489218623663461607955943720516395154096949806486242309796947211169665961580041
7883992232264628498942352276926391124360767508997671789683459337890742346354557193530665615379900277916265636361861974048590938234062235459769733137213659343717585590664439
6461328300113672601934068706442339489199215304385281541659630118549423634863524585746642834609062916279564926584723003608555598989761916293248140094592489899468468862322586
0170551468905649837276003927487069550463788819741699429049102853180410776516007263384721638903782270013843599599730265727198804324664312359758555217196960513921010226596324
7887830977405333131551762978152880718652603176327264828094499370456258099938053058497699570080289379808014902901005293898472279947167804821689424159118284257696464786505731
5325178302336307298251692210346584265894447464916123854689718507968172909139032182834111184821384767728316548653212317382004131990510518967022201887049585687180509590730360
6930402937216038968917605587676955382318093705826257083898387409098468656634271397500013291835105943321729879825243707508272087959859437157667660155782699660343197752623308
8989962587800628009560944416932377944955441033696586261556256010669390303203878970983673786087056641433585106111658314520424513208508589994932364831689671194951671619567622
7070906973889588855579562466641536561723549301807394004760529801721771391686788000277851966173070061284517307582503735643102065112443730825229625040453160590741343881872563
4779138306605909318802522310085340176840261401539616989192075147108033757708849740141834599753972059878682064879116064969858177601153972058498222698907181349432691801821173
3188063653910893689811714891357456680542807485170175858266639633570189354449832669762835092657922201746372190273119641751489944010079636876017826747107019945473218887832742
6088966724371574713420600093704251309893630537459784279980403132989417266492290425730958368534416215640557290282066224003863237526380910233269897838860423759625601567975262
6950798639868104294832333160267216555178120899264677804935741326387137408423885546538336158643451305439624281397279559725995110706314305992615495622958320232708057681156690
4895866105220300573725298472118747827136713666058669271094875563974858489475910819727033878284439864486743456200958161930314727345961900499318424337975243662489363321244850
5971992523668529249305346252764137853413208943128901523738092556045987090912766662329678703328882059134949580074074473143388800724532321747309659741967114441453127132790205
1010047671014350638857953478447255389801541923317027519896180635152682543173193832925891931530164130548972311128666465492971930479296432829556719092881692091042334122007454
24204990087258504620805110487588305949599031118873666850941488217257345763552339640384813182131674083590069164000532622581847837650678044511777717328658189899215358309447765
350341796875

Approximately 3.00 times 10^6328
```

Scoring Trees

- Now we know how many **unrooted** candidate trees there exist for *n* taxa
- How do we chose among them?
  - → we need some scoring criterion f() to evaluate them
  - → finding the optimal tree under most of these criteria is NP-Hard







Scoring Criteria and Tree Inference Algorithms

Building Trees

- We distinguish between
  - *Distance-based methods*
    - → use MSA to compute a matrix of pair-wise distances
    - → build a tree using these distances
    - → Heuristics (essentially hierarchical clustering methods)
      - → *Neighbor Joining:* NJ
      - → *Unweighted Pair Group Method with Arithmetic Mean:* UPGMA
    - → least-squares method: explicit optimality criterion

Less accurate,
but faster

-
  - *Character-based methods*
    - → optimality criteria *f()* operate directly on the MSA & tree
      - → parsimony
      - → maximum likelihood
      - → Bayesian inference
    - → take the current tree topology & MSA to calculate a score
    - → the score tells us how well the MSA data fits the tree

Slow, but more
accurate

Building Trees

- We distinguish between
  - *Distance-based methods*
    - → use MSA to compute a matrix of pair-wise distances
    - → build a tree using these distances
    - → Heuristics (essentially hierarchical clustering methods)
      - → *Neighbor Joining:* NJ
      - → *Unweighted Pair Group Method with Arithmetic Mean:* UPGMA
    - → least-squares method: explicit optimality criterion

Less accurate,
but faster

  - *Character-based methods*
    - → optimality criteria *f()* operate directly on the MSA
      - → parsimony
      - → maximum likelihood
      - → Bayesian inference
    - → take the current tree topology & MSA to calculate a score
    - → the score tells us how well the MSA data fits the tree

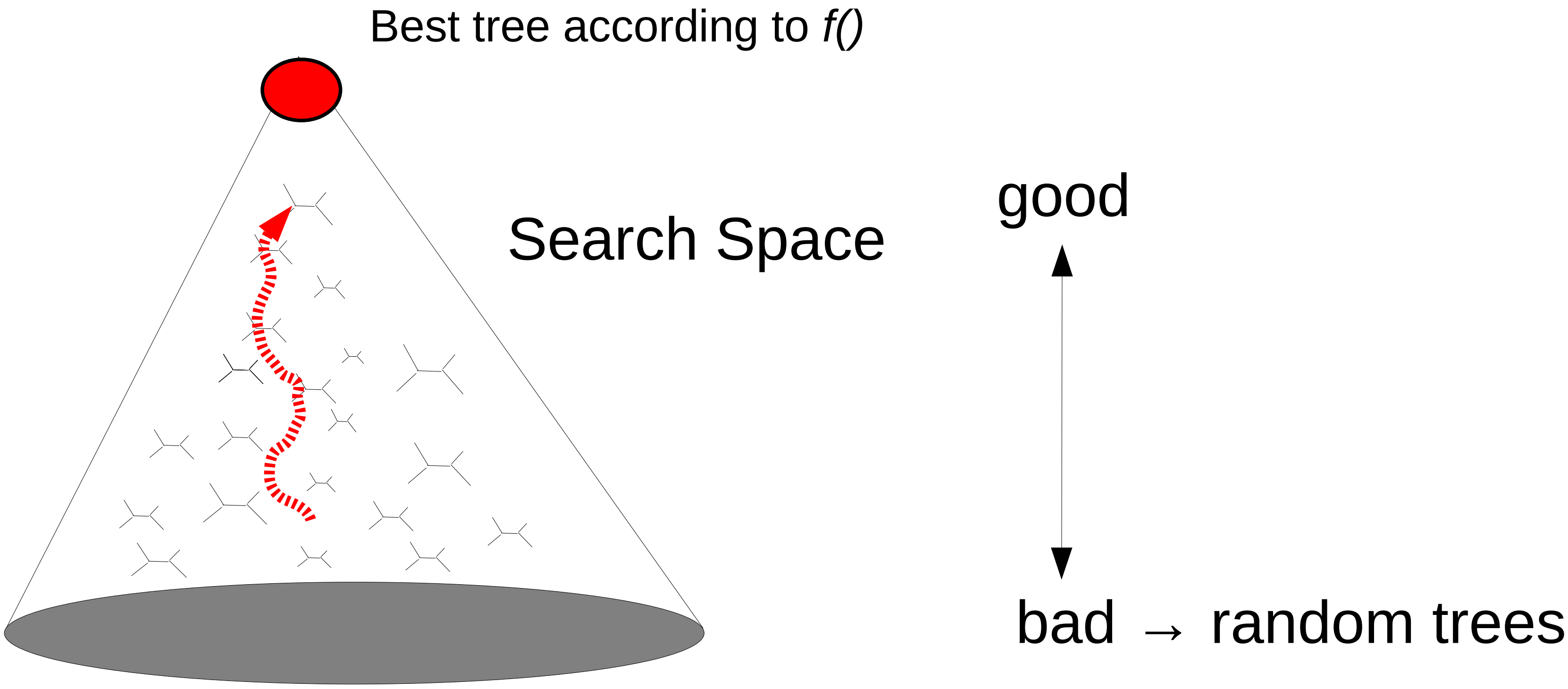
Memory-intensive!

Slow, but more
accurate

NP-Hardness

- Because of the **super-exponential increase** in the number of possible trees for *n* taxa ...
- all interesting optimality criteria on trees are NP-hard:
  - Least squares
  - Parsimony → discrete criterion
  - Likelihood → statistical criterion
  - Bayesian → integrate likelihood over entire tree space

Search Space



Best tree according to *f()*

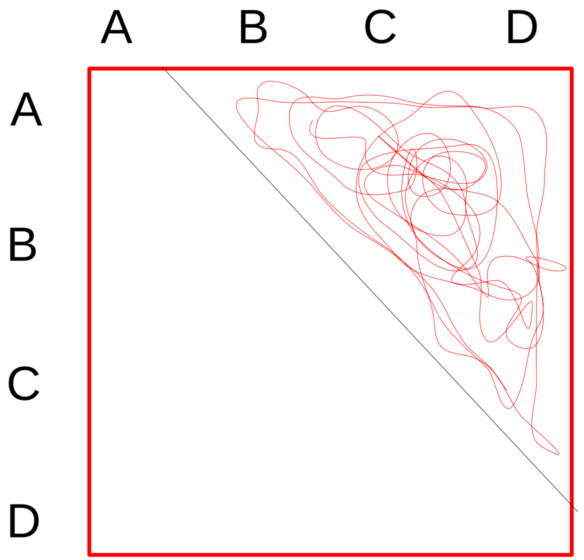
Search Space

good

bad → random trees

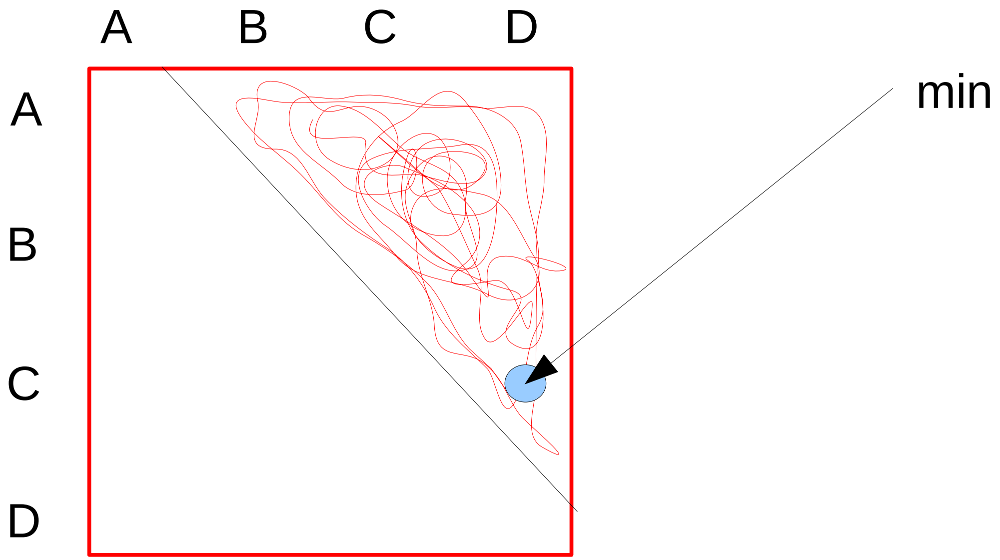
Let’s start with distance based methods/heuristics

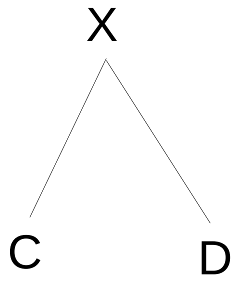
Neighbor Joining → Principle



Given a kind of distance matrix $D_{i,j}$ where $i,j=1...4$

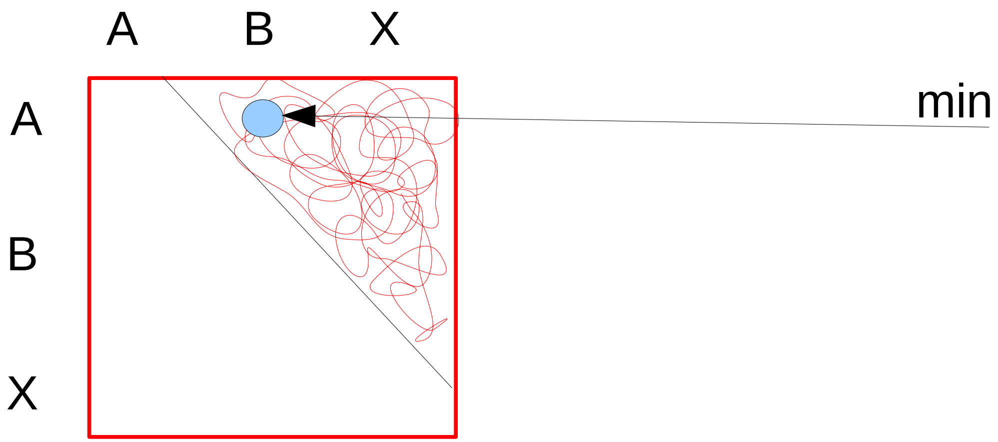
Neighbor Joining → Principle

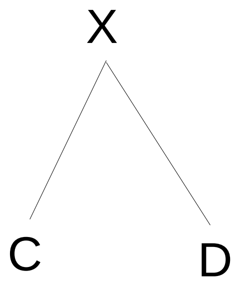




Given a kind of distance matrix $D_{i,j}$ where $i,j=1...4$
Find minimum and merge taxa

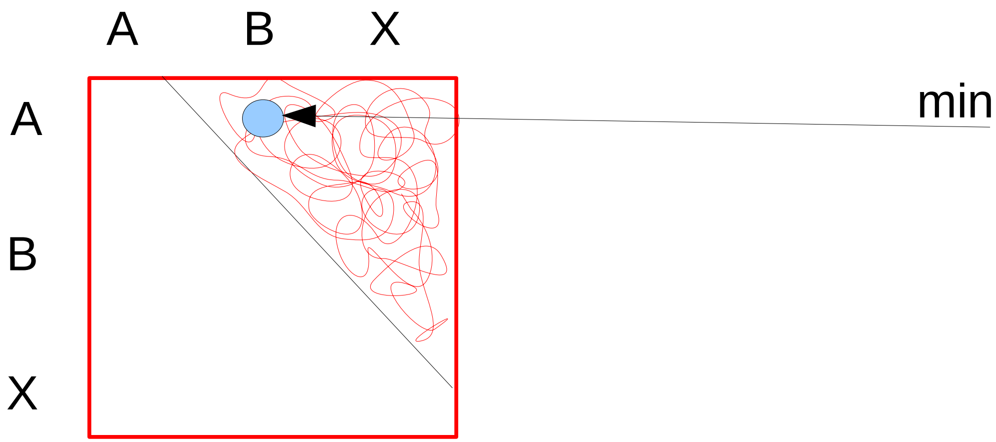
Neighbor Joining → Principle

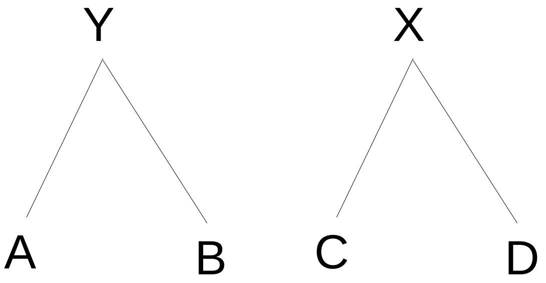




Given a kind of distance matrix $D_{i,j}$ where $i,j=1...4$
Find minimum and merge taxa
Compute a new distance matrix of size $n-1 = 3$
Find minimum

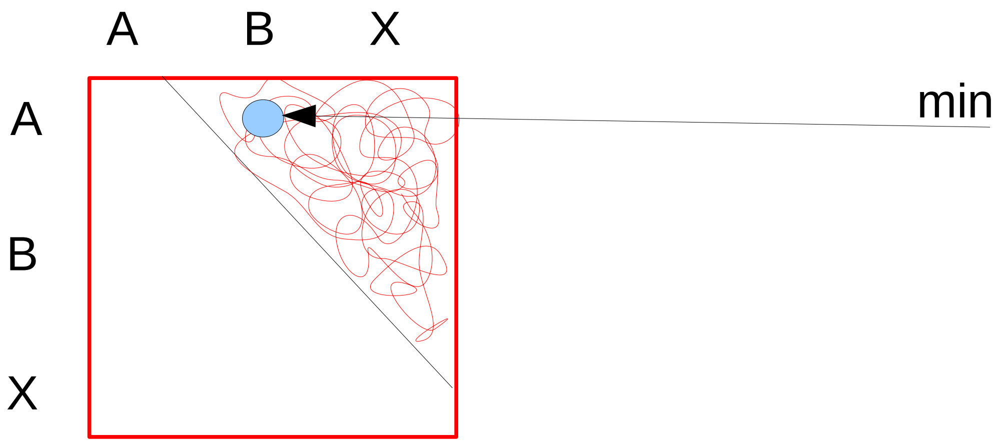
Neighbor Joining → Principle

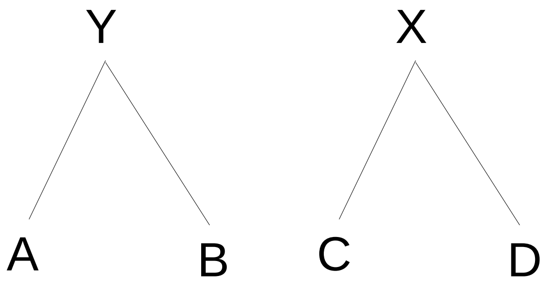




Given a kind of distance matrix $D_{i,j}$ where $i,j=1...4$
Find minimum and merge taxa
Compute a new distance matrix of size $n-1 = 3$
Find minimum and merge taxa

Neighbor Joining → Principle





Given a kind of distance matrix $D_{i,j}$ where $i,j=1...4$
Find minimum and merge taxa
Compute a new distance matrix of size $n-1 = 3$
Find minimum and merge taxa
Etc.
Space complexity: $O(n^2)$
Time complexity: $O(n^3)$
Key question: how do we compute distance between X and A or X and B respectively
→ for progressive alignment we may align the profile of X with all remaining sequences

Neighbor Joining Algorithm

- For each tip compute

  $u_i = \Sigma_j D_{ij}/(n-2)$

  → this is in principle the average distance to all other tips

  → the denominator is $n-2$ instead of n, see below why

- Find the pair of tips, (i, j) for which $D_{ij}-u_i-u_j$ is minimal
- Connect the tips (i,j) to build a new ancestral node X
- The branch lengths from the ancestral node X to i and j are:

  $b_i = 0.5\ D_{ij} + 0.5\ (u_i-u_j)$

  $b_j = 0.5\ D_{ij} + 0.5\ (u_j-u_i)$

- Update the distance matrix:
  → Compute distance between the new node X and each remaining tip as follows:

  $D_{ij,k} = (D_{ik}+D_{jk}-D_{ij})/2$

- Replace tips i and j by the new node X which is now treated as a tip
- Repeat until only two nodes remain
  → connect the remaining two nodes with each other

The UPGMA algorithm

- Usually introduced before Neighbor Joining *NJ* → it is simpler and older
- UPGMA is practically not used any more today for phylogeny reconstruction, but it is used for progressive multiple sequence alignment (see MUSCLE algorithm)
- In contrast to NJ it produces *ultrametric* trees!
- It produces rooted trees
- UPGMA stands for: *Unweighted Pair Group Method with Arithmetic Mean*
- Like NJ it uses a distance matrix *D* for clustering/joining nodes
- UPGMA can be used if we know that we have an ultrametric tree!

→ this is usually not the case!

UPGMA example

	A	B	C	D
A		0.4	0.6	0.6
B			0.6	0.6
C				0.2
D				

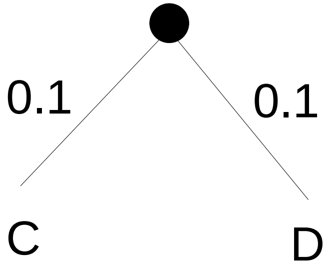
We will first walk through the algorithm and then look at the formal description!

UPGMA example

	A	B	C	D
A		0.4	0.6	0.6
B			0.6	0.6
C				0.2
D				

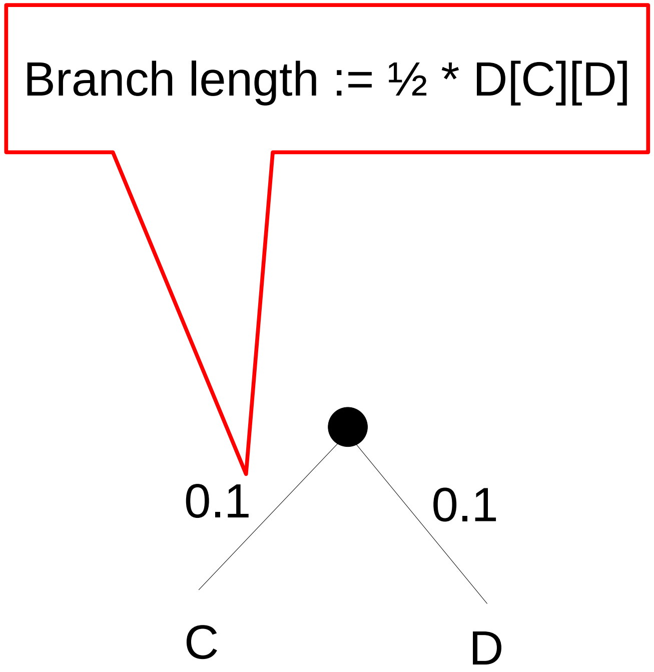
UPGMA example

	A	B	C	D
A		0.4	0.6	0.6
B			0.6	0.6
C				0.2
D				



UPGMA example

	A	B	C	D
A		0.4	0.6	0.6
B			0.6	0.6
C				0.2
D				

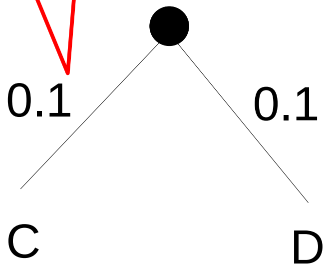


Branch length := ½ * D[C][D]

UPGMA example

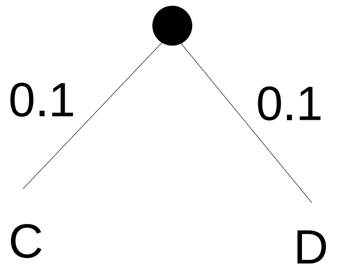
	A	B	C	D
A		0.4	0.6	0.6
B			0.6	0.6
C				0.2
D				

Branch length := ½ * D[C][D]
Ensures ultrametricity!



UPGMA example

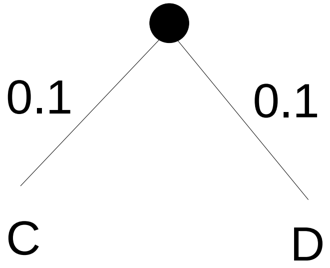
	A	B	C	D
A		0.4	0.6	0.6
B			0.6	0.6
C				0.2
D				



D [A][(C,D)] = ½ * 0.6 + ½ * 0.6
D [B][(C,D)] = ½ * 0.6 + ½ * 0.6

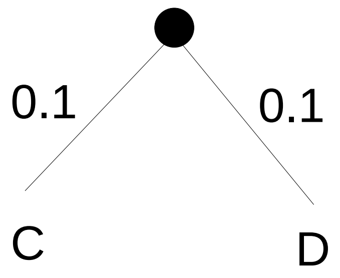
UPGMA example

	A	B	(C,D)
A		0.4	0.6
B			0.6
(C, D)			



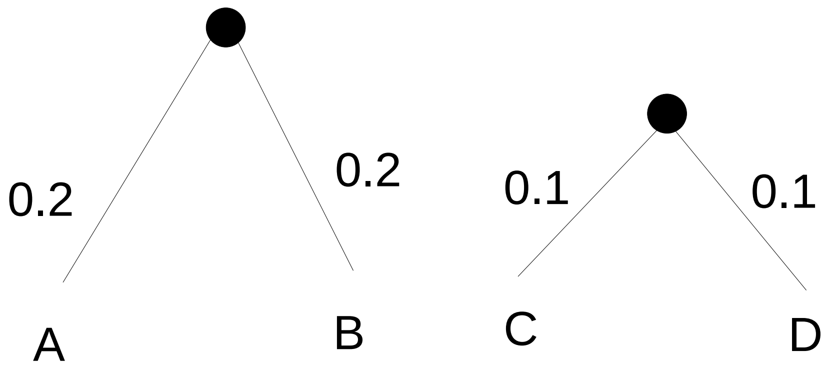
UPGMA example

	A	B	(C,D)
A		0.4	0.6
B			0.6
(C, D)			



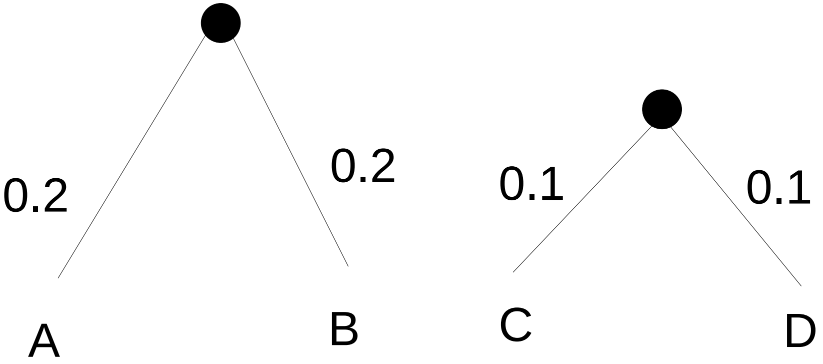
UPGMA example

	A	B	(C,D)
A		0.4	0.6
B			0.6
(C, D)			



UPGMA example

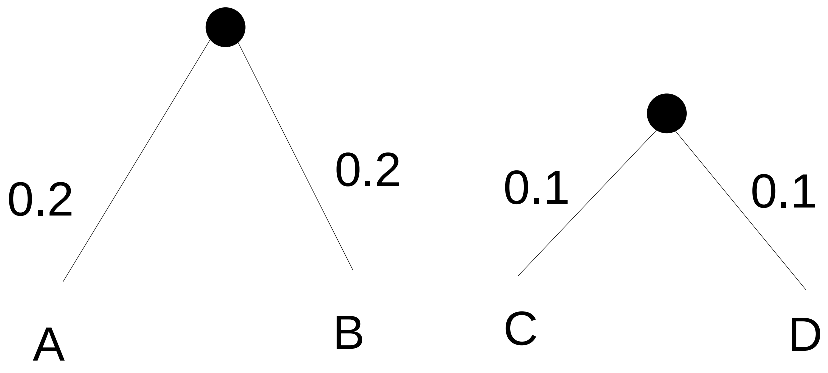
	A	B	(C,D)
A		0.4	0.6
B			0.6
(C, D)			



D[A,B][C,D] = ½ * 0.6 + ½ * 0.6

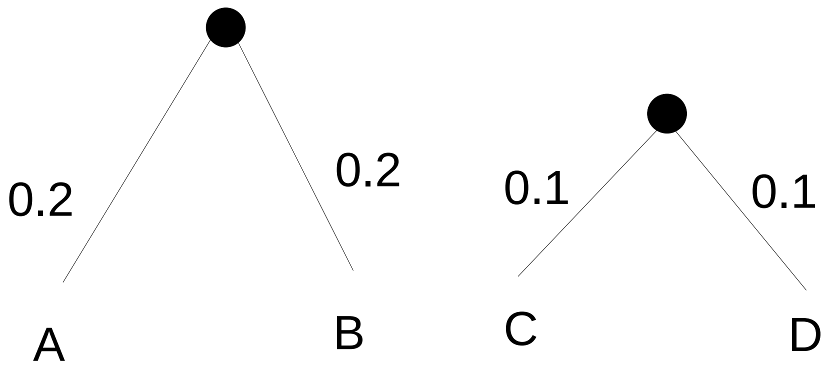
UPGMA example

	(A,B)	(C,D)
(A,B)		0.6
(C,D)		



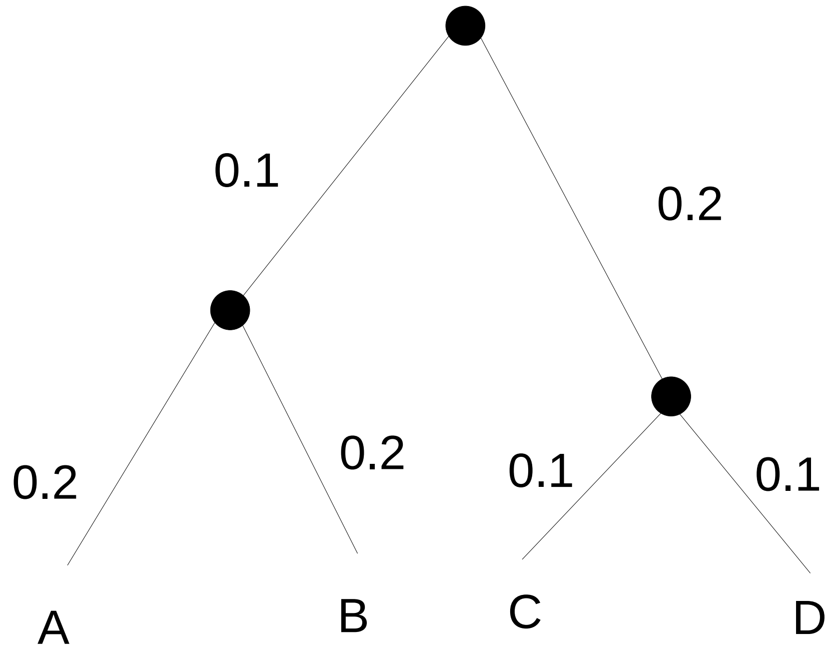
UPGMA example

	(A,B)	(C,D)
(A,B)		0.6
(C,D)		



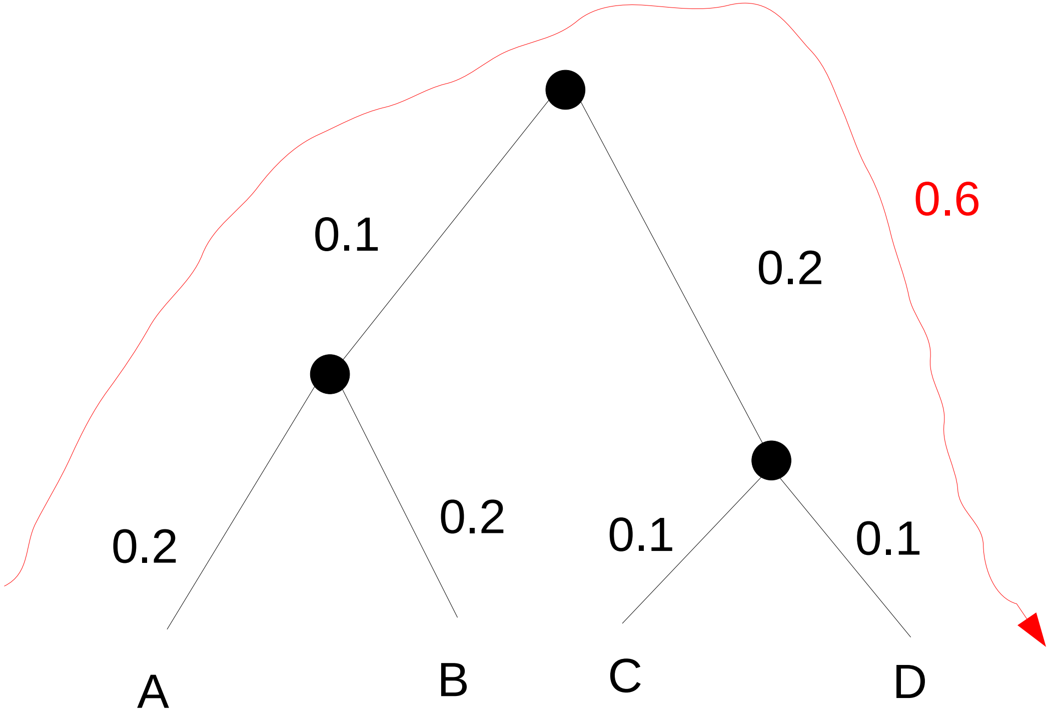
UPGMA example

	(A,B)	(C,D)
(A,B)		0.6
(C,D)		



UPGMA example

	(A,B)	(C,D)
(A,B)		0.6
(C,D)		



UPGMA Formal description

- Find the minimum $D[i][j]$
- Merge i and $j \rightarrow (i,j)$
- This new group has $n_{(i,j)}$ members, where $n_{(i,j)} := n_i + n_j$
- Connect i and j to form a new node (i,j)
- Assign the two branches connecting $i \rightarrow (i,j)$ and $j \rightarrow (i,j)$ the length $D[i][j]/2$
- Update the distances between (i,j) and all k, where $k \neq i$ and $k \neq j$ via $D[(i,j)][k] = (n_i/(n_i+n_j)) * D[i][k] + (n_j/(n_i+n_j)) * D[j][k]$
- Naive implementation: $O(n^3)$ → search for minimum in each instance of matrix D
- Maintain a list of per-column (or per-row) minima

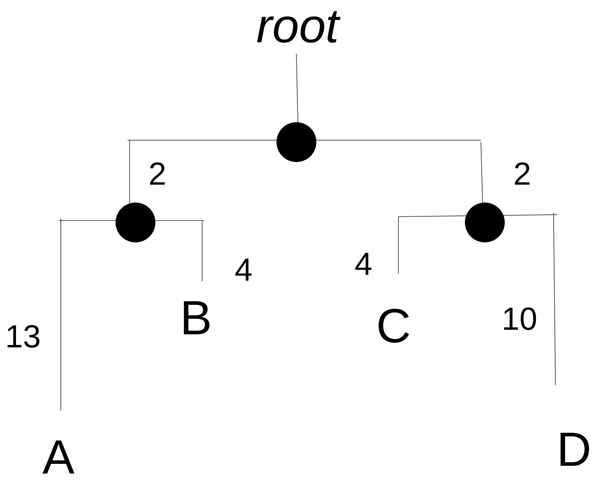
  → update list $O(n)$

  → look for minimum $O(n)$

  → $O(n^2)$
- In contrast to NJ we don't need to update the entire matrix each time, thus only $O(n^2)$

UPGMA on non-ultrametric trees

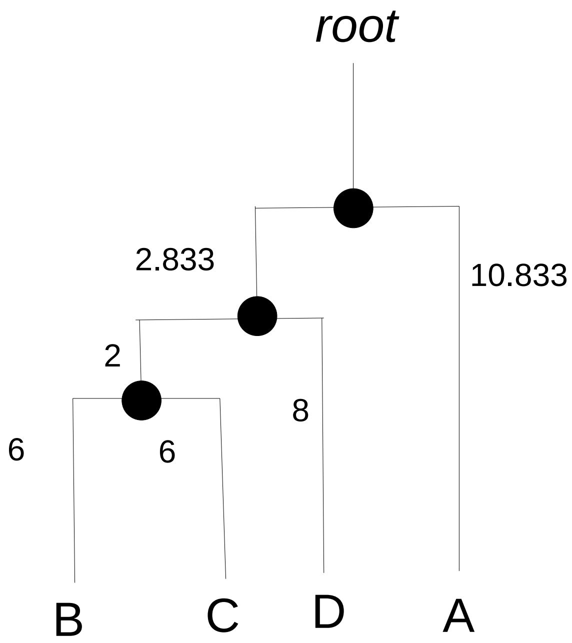
- Can yield misleading results
- Most trees are not ultrametric → do not have equal evolutionary rates among all lineages



True tree

	A	B	C	D
A	0	17	21	27
B		0	12	18
C			0	14
D				0

Patristic distance matrix

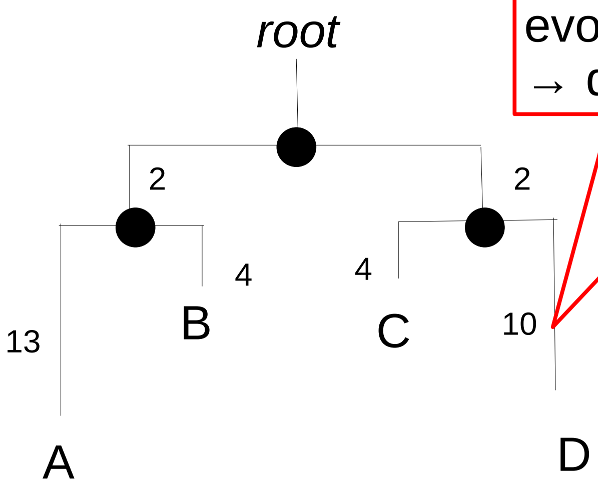


UPGMA tree

UPGMA on non-ultrametric trees

- Can yield misleading results
- Most trees are not ultrametric → do not have equal evolutionary rates among all lineages

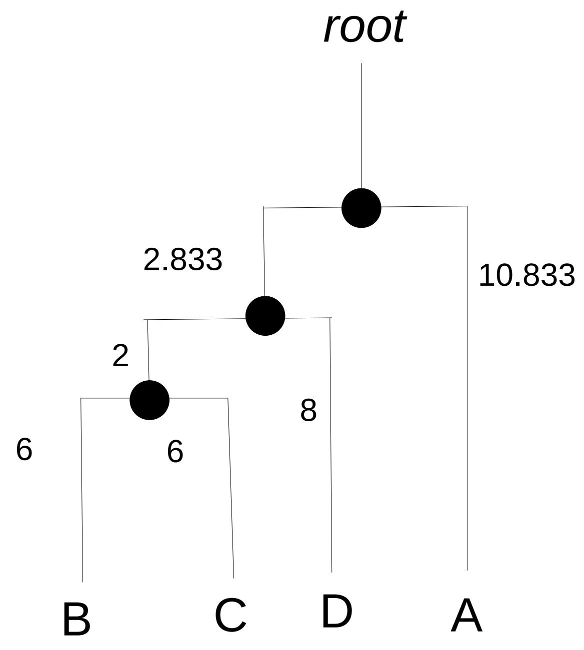
Imagine a higher evolutionary pressure!
→ difficult life conditions!



True tree

	A	B	C	D
A	0	17	21	27
B		0	12	18
C			0	14
D				0

Patristic distance matrix



UPGMA tree

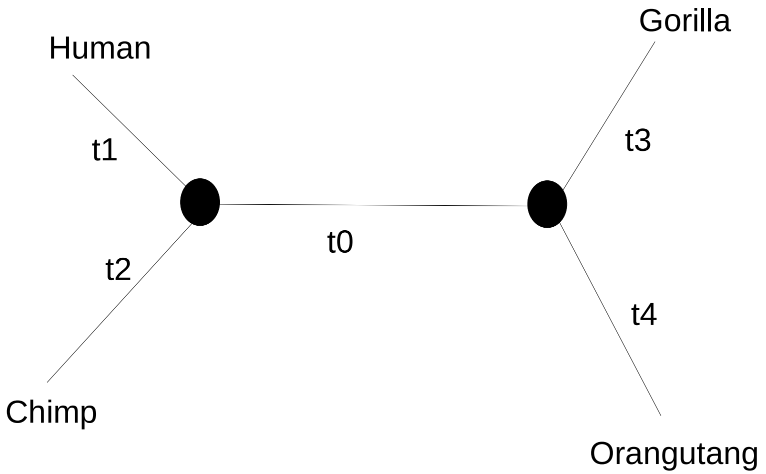
UPGMA and NJ

- Ad hoc heuristics
- Good for clustering any type of data on which you can define a reasonable distance
- Now let us look at explicit criteria

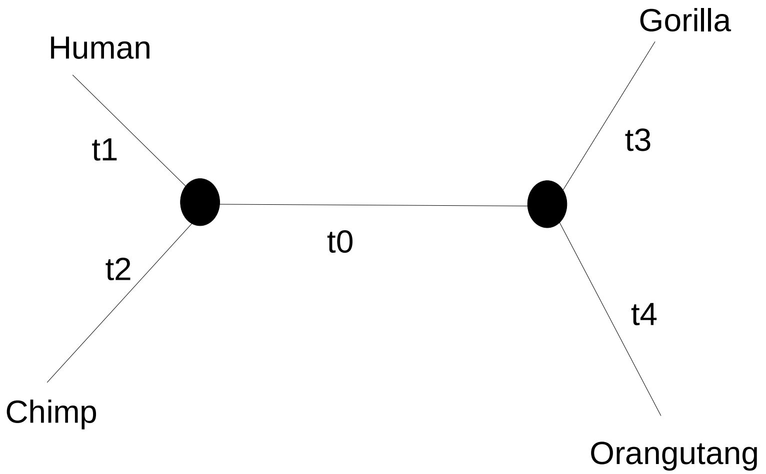
  → given a specific tree tolopology they yield a score

Let’s start with a distance-based criterion

Least Squares



Least Squares



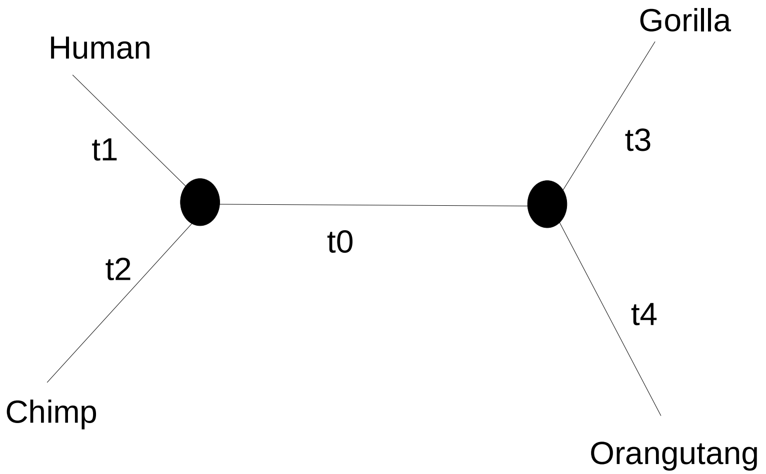
Patristic distances

d[H][C] = t1 + t2
d[H][G] = t1 + t0 + t3
d[H][O] = t1 + t0 + t4
d[C][G] = t2 + t0 + t3
d[C][O] = t2 + t0 + t4
d[G][O] = t3 + t4

Least Squares

Given distance matrix D

	H	C	G	O
H		0.0965	0.1140	0.1849
C			0.1180	0.2009
G				0.1947
O				

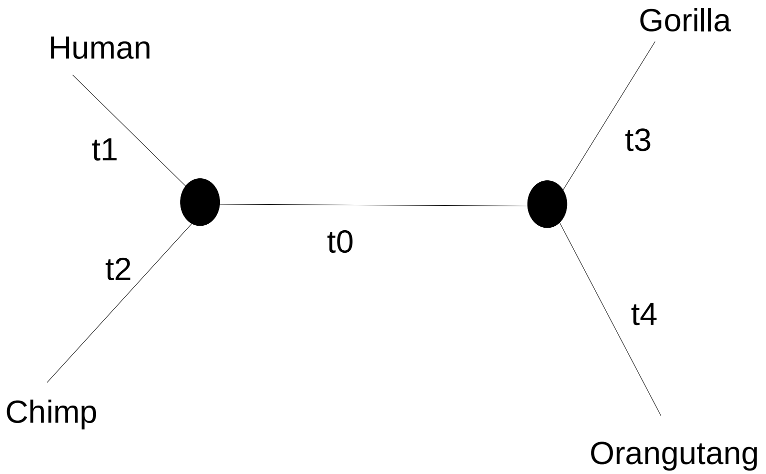


d[H][C] = *t1* + *t2*
d[H][G] = *t1* + *t0* + *t3*
d[H][O] = *t1* + *t0* + *t4*
d[C][G] = *t2* + *t0* + *t3*
d[C][O] = *t2* + *t0* + *t4*
d[G][O] = *t3* + *t4*

Least Squares

Given distance matrix D

	H	C	G	O
H		0.0965	0.1140	0.1849
C			0.1180	0.2009
G				0.1947
O				



Find $t0, t1, \ldots, t4$ such that deviation of $d[i][j]$ from $D[i][j]$ is minimized!

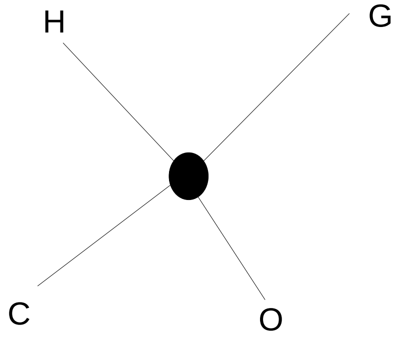
$$Q := (d[H][C] - D[H][C])^2 + (d[H][G] - D[H][G])^2 + (d[H][O] - D[H][O])^2 + (d[C][G] - D[C][G])^2 + (d[C][O] - D[C][O])^2 + (d[G][O] - D[G][O])^2$$

$d[H][C] = t1 + t2$
$d[H][G] = t1 + t0 + t3$
$d[H][O] = t1 + t0 + t4$
$d[C][G] = t2 + t0 + t3$
$d[C][O] = t2 + t0 + t4$
$d[G][O] = t3 + t4$

Least Squares Example

tree	t0	t1	t2	t3	t4	Q
((H,C),G,O)	0.008840	0.043266	0.053280	0.058908	0.135795	0.000035
((H,G),C,O)	0.000000	0.046212	0.056227	0.061854	0.138742	0.000140
((H,O),C,G)	As above	-	-	-	-	-

Least Squares Example



Star tree

tree	t0	t1	t2	t3	t4	Q
((H,C),G,O)	0.008840	0.043266	0.053280	0.058908	0.135795	0.000035
((H,G),C,O)	0.000000	0.046212	0.056227	0.061854	0.138742	0.000140
((H,O),C,G)	As above	-	-	-	-	-

Least Squares Optimization

- Given a fixed, fully binary, unrooted tree *T* with *n* taxa
- Given a pair-wise distance matrix *D*
- Assign branch lengths $t_1 , ..., t_{2n-3}$ to the tree such that:

  the sum of the squared differences between the pair-wise *patristic* (tree-based!) distances d_{ij} and the *plain* pair-wise distances D_{ij} is minimized

- In other words:

  $Q = \Sigma_{i<j} (D_{ij} – d_{ij})^2$ → find an assignment $t_1 ,..., t_{2n-3}$ to the tree such that Q is minimized

  Q can be minimized by taking the derivative and solving a system of linear equations in $O(n^3)$

  Minimization methods for Q that take into account the tree-like structure run in $O(n^2)$ or even $O(n)$

- Then, also find that tree topology *T* that minimizes *Q*
- Finding the minimal least squares tree is NP-hard

W.H.E. Day "Computational Complexity of Inferring Phylogenies from dissimilarity matrices", *Bulletin of Mathematical Biology* 49: 461-467, 1986.

Least Squares

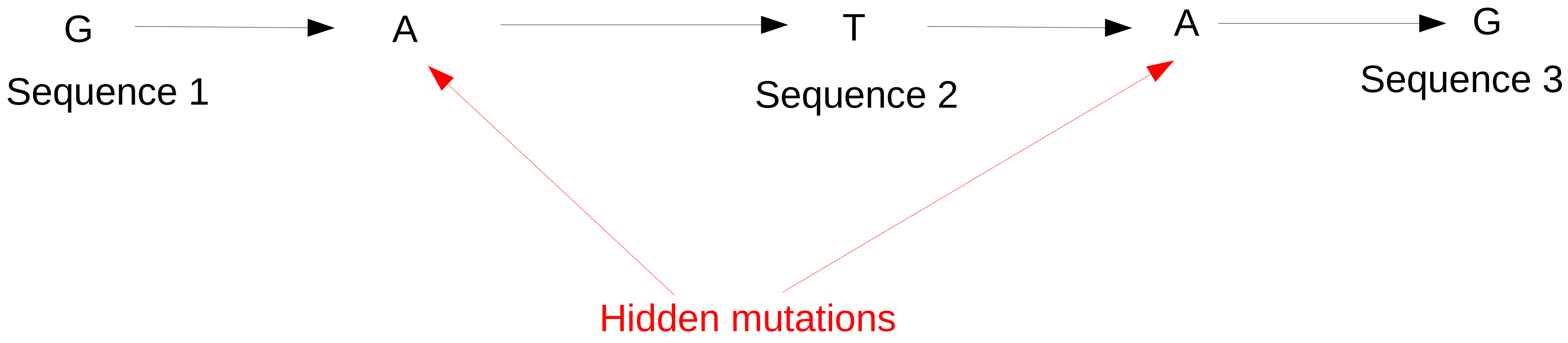
- *NP-hard* because of tree search problem
- Scoring a single tree takes time between $O(n)$ to $O(n^3)$
- There also exist weighted versions:

  $Q = \Sigma_{i<j} w_{ij}(D_{ij} - d_{ij})^2$

  where $w_{ij} := 1/D_{ij}$ *or* $w_{ij} := 1/D_{ij}^2$
- We will see how to search for trees a bit later-on
- Make sure you understand the difference between
  - Scoring a single tree
  - Searching for the tree with the best score

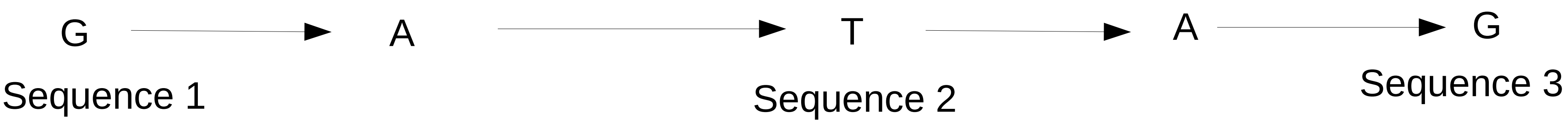
Distances

- A preview of the next lecture
- We need to accommodate multiple substitutions in the evolutionary history of sequences



Distances

- A preview of the next lecture
- We need to accommodate multiple substitutions in the evolutionary history of sequences



Simple edit distances will not be sufficient → we need statistical modes!

Minimum Evolution Method

- Similar to least squares
- Explicit Criterion → minimize total branch length (tree length) of the reconstructed tree
- Branch lengths are obtained using least-squares method → same time complexity
- Instead of searching for the tree that minimizes the squared difference between $D[i][j]$ and $d[i][j]$ that is denoted by Q we search for the tree where $t0 + t1 + t2 + t3 + t4$ is minimized

tree	t0	t1	t2	t3	t4	Q	Tree length
((H,C),G,O)	*0.008840*	*0.043266*	*0.053280*	*0.058908*	*0.135795*	0.000035	***0.240741***
((H,G),C,O)	*0.000000*	*0.046212*	*0.056227*	*0.061854*	*0.138742*	0.000140	*0.303035*
((H,O),C,G)	As above	-	-	-	-	-	

Distance-based Methods

- Clustering Algorithms/Heuristics
  - Neighbor Joining

    → Heuristic for Minimum Evolution Method
  - UPGMA
- Explicit criteria
  - least squares
  - minimum evolution
- All depend on the accuracy of the pair-wise distance matrix *D*
- The distance matrix needs to be an exact reflection of the tree

Character-based Methods

- Parsimony
- Maximum Likelihood
- Bayesian Inference

The Parsimony Criterion

- Directly operates on the MSA
- Find the tree that explains the data with the least amount of mutations
- Questions:
  - How do we count the least amount of mutations on a given tree?

    → dynamic programming algorithm
  - How do we find the tree topology that requires the least amount of mutations

    → requires a tree search!

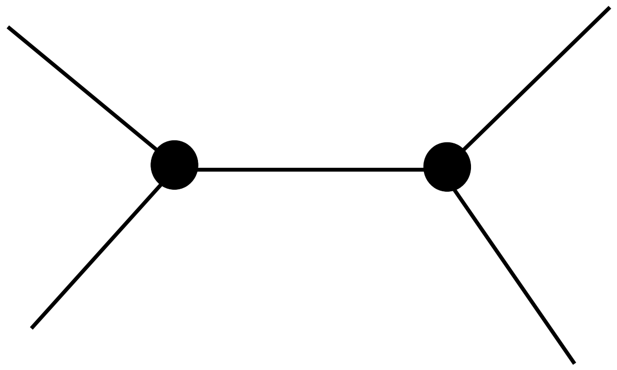
    → remember the number of trees!

    → this is also NP-hard!

Parsimony

MSA

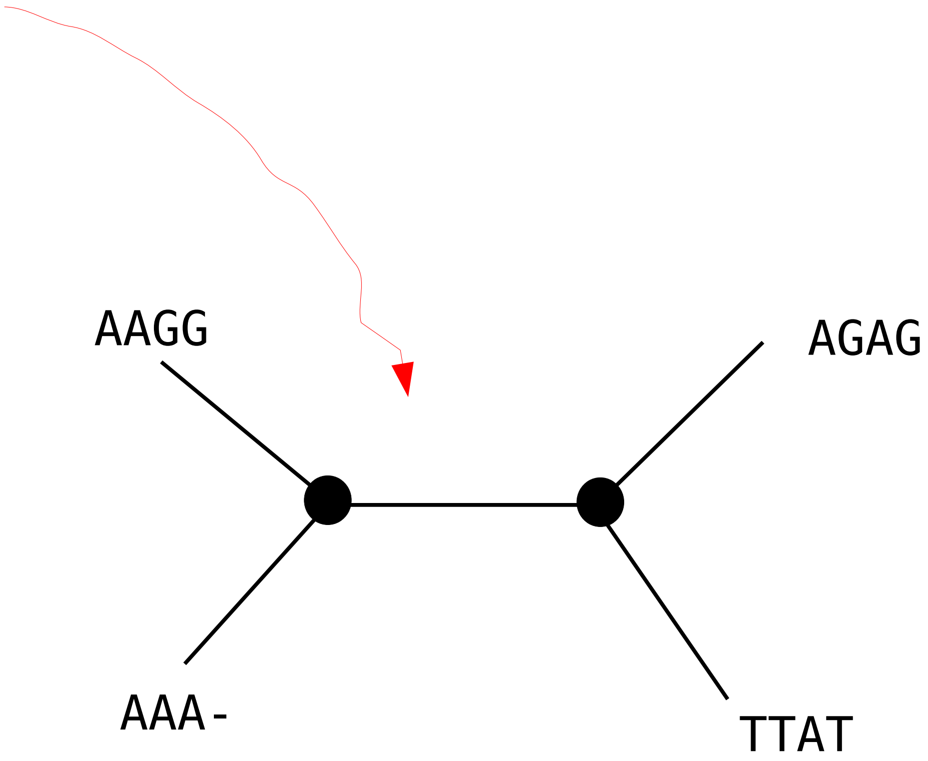
```
S1: AAGG
S2: AAA-
S3: AGAG
S4: TTAT
```



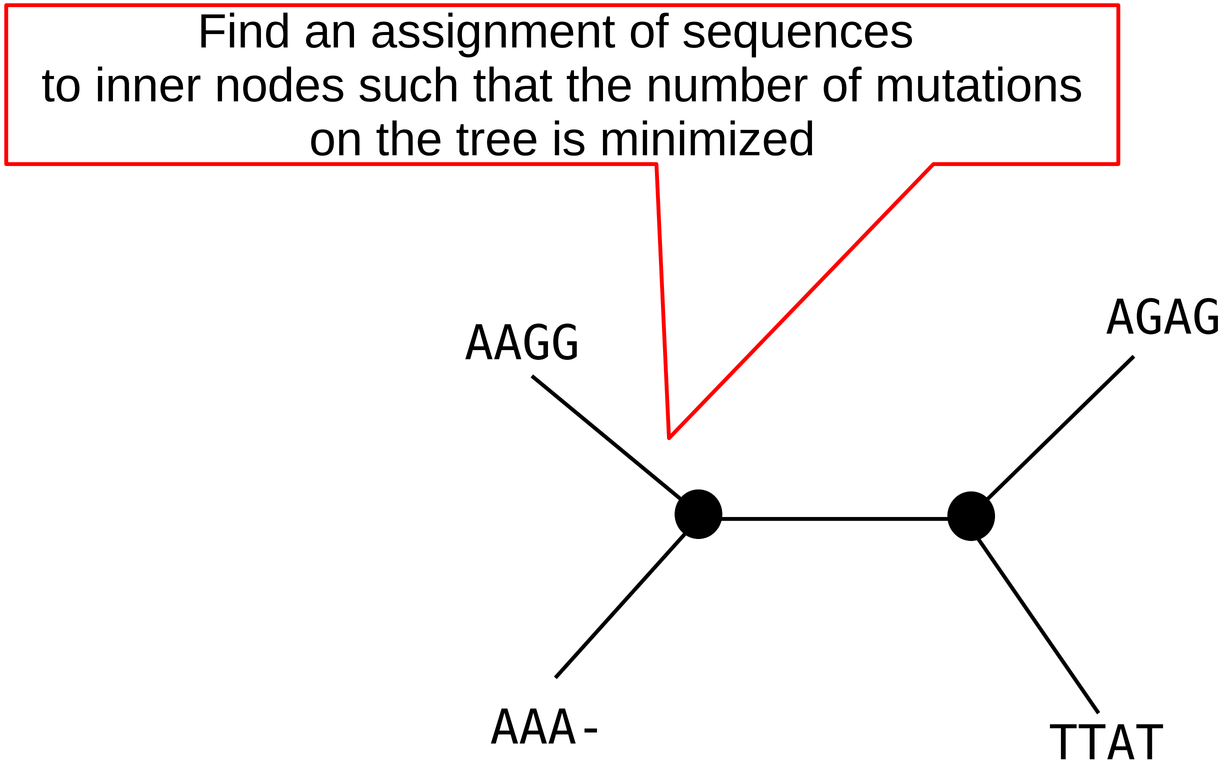
Parsimony

MSA

```
S1: AAGG
S2: AAA-
S3: AGAG
S4: TTAT
```



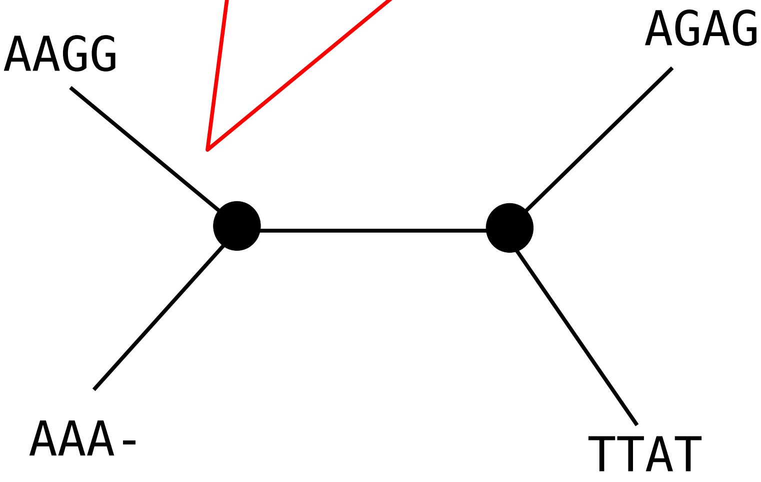
Parsimony



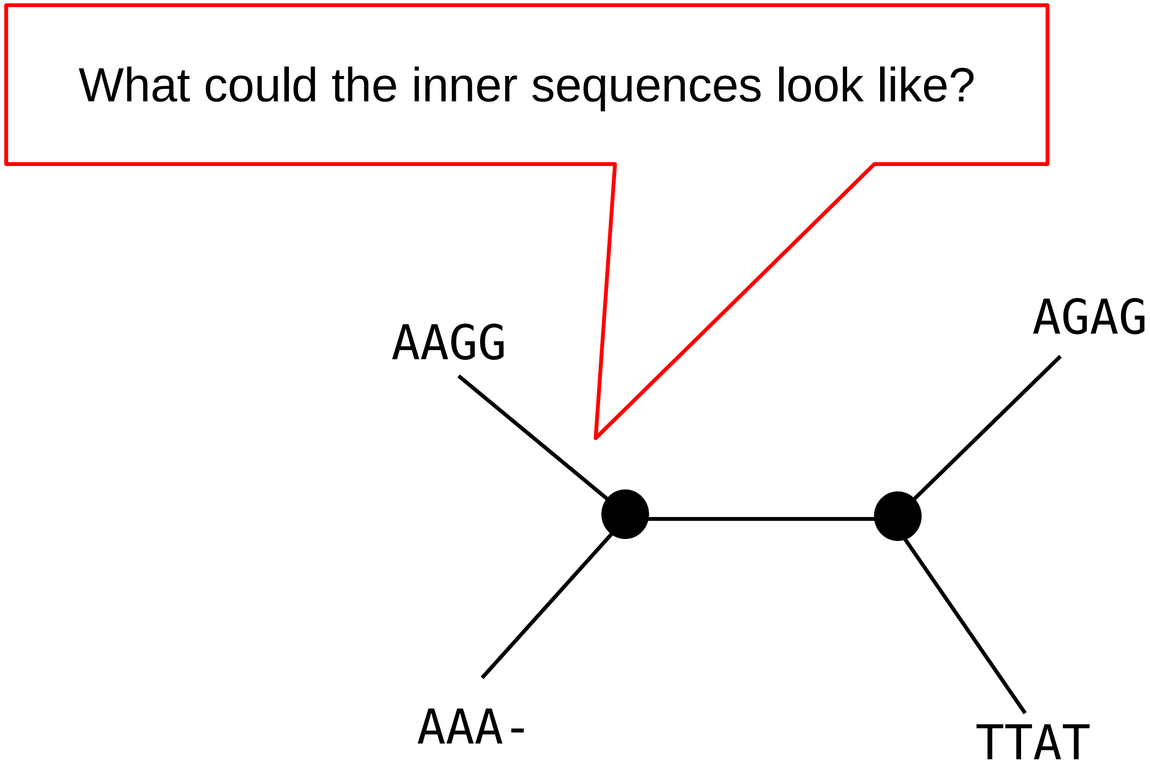
Find an assignment of sequences
to inner nodes such that the number of mutations
on the tree is minimized

Parsimony

This is somewhat similar to the tree alignment problem, but here, we are given an alignment!



Parsimony



What could the inner sequences look like?

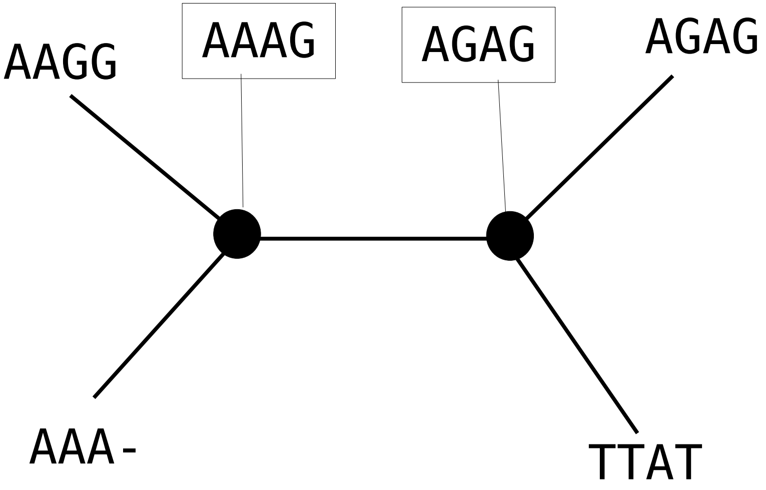
AAGG

AGAG

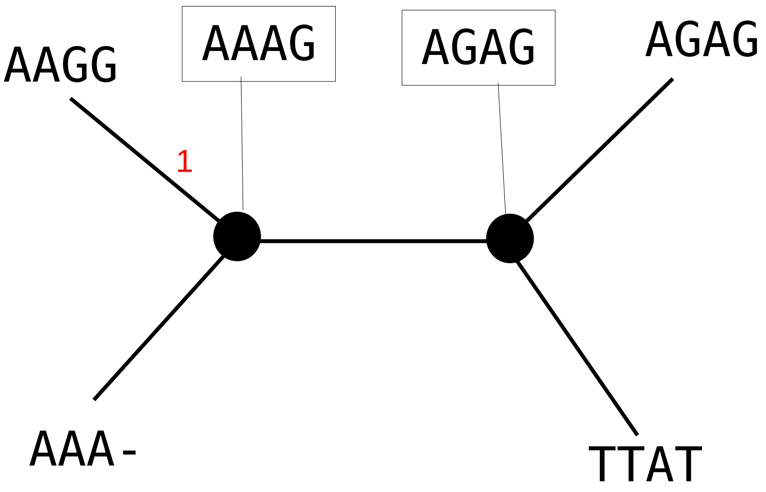
AAA-

TTAT

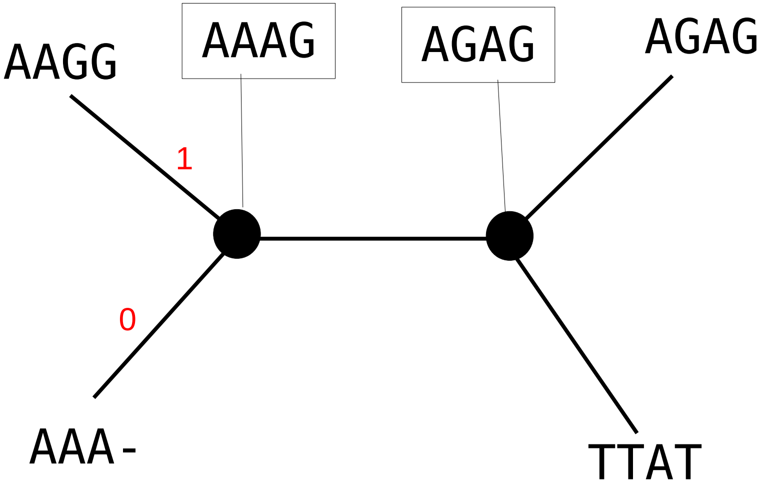
Parsimony



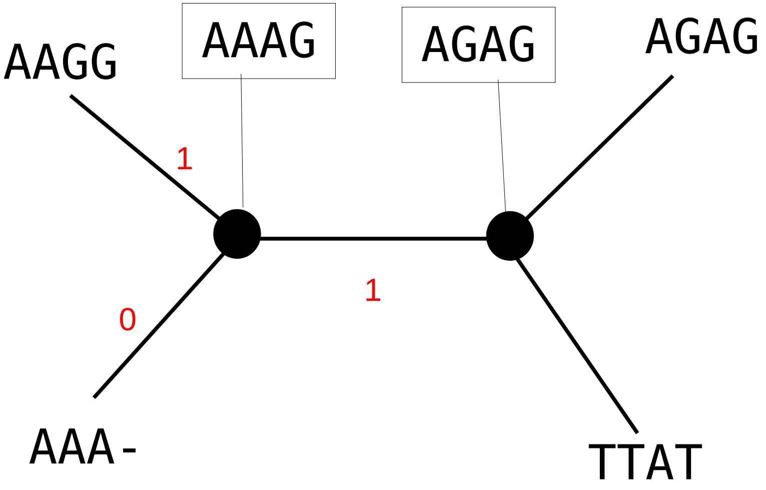
Parsimony



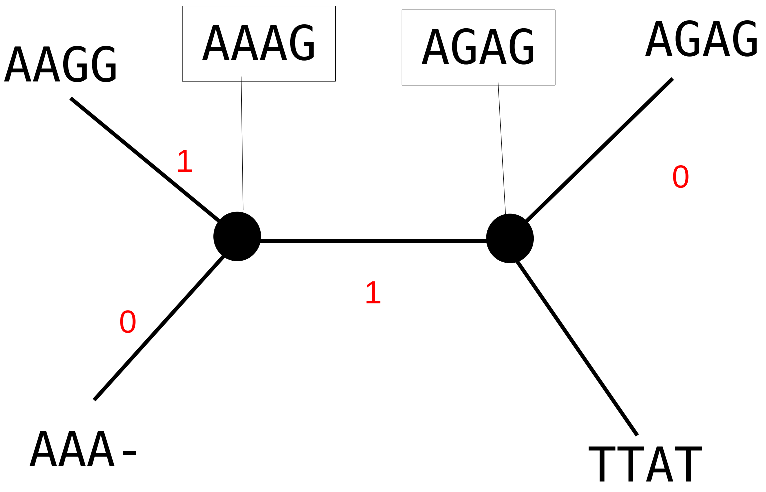
Parsimony



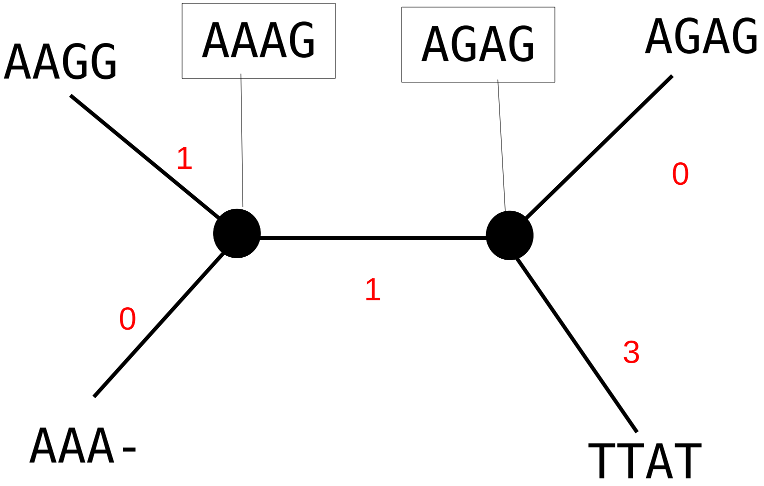
Parsimony



Parsimony

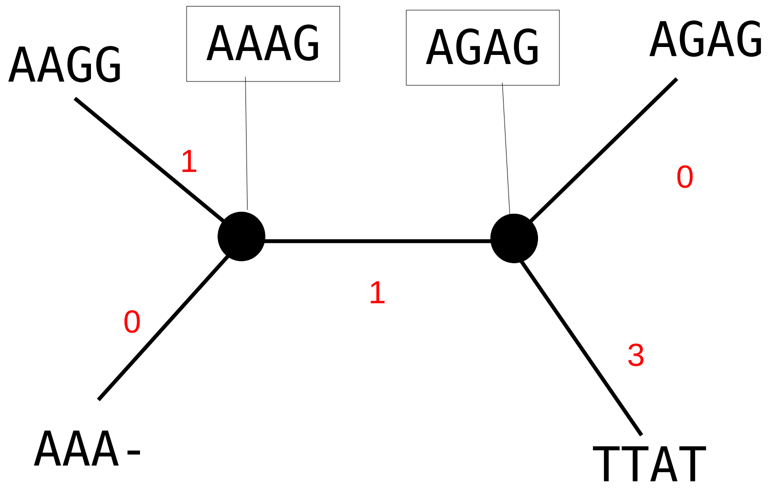


Parsimony



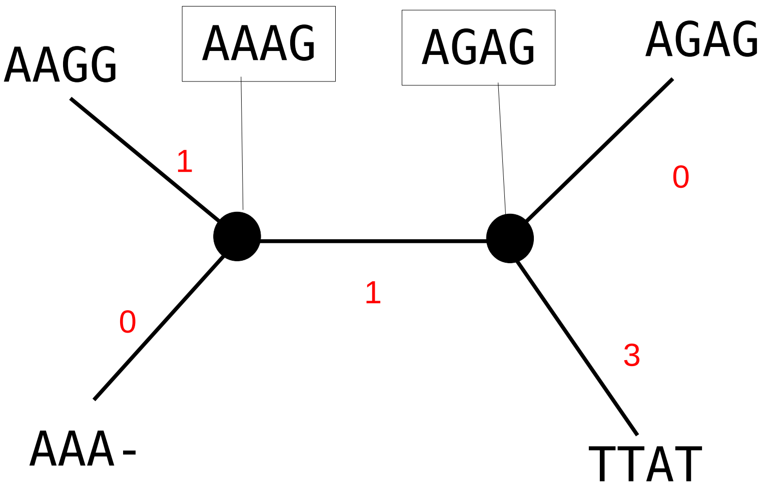
Parsimony

Parsimony Score of this tree = 5



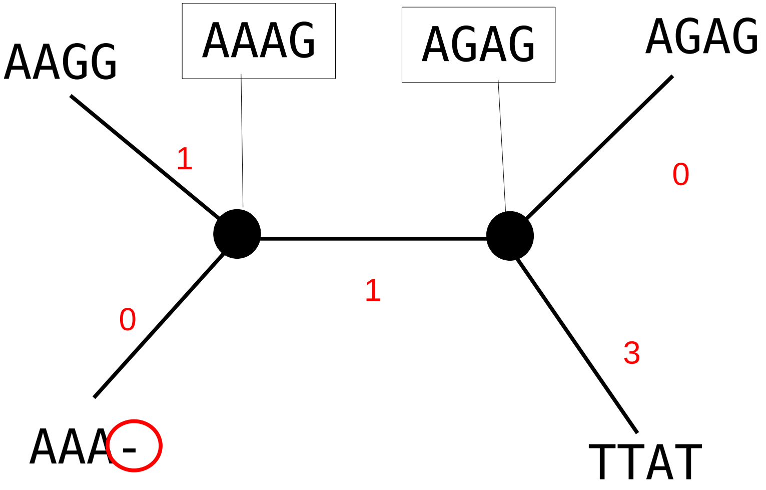
Parsimony

Parsimony Score of this tree = 5
This is also the minimum score for this tree.



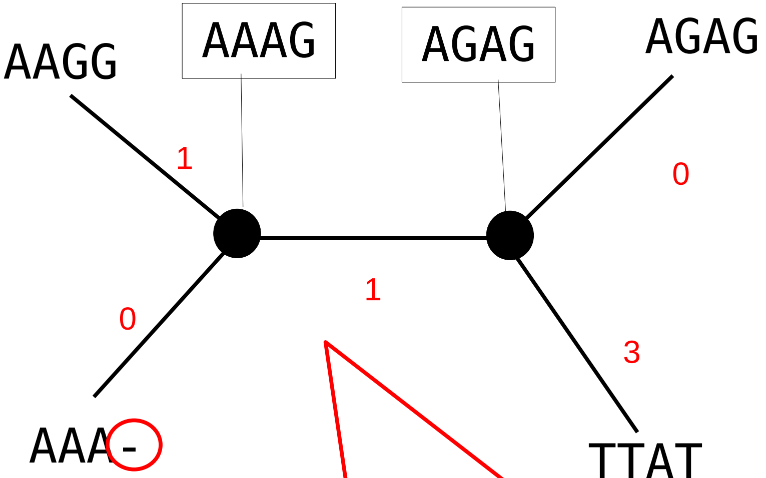
Parsimony

Parsimony Score of this tree = 5



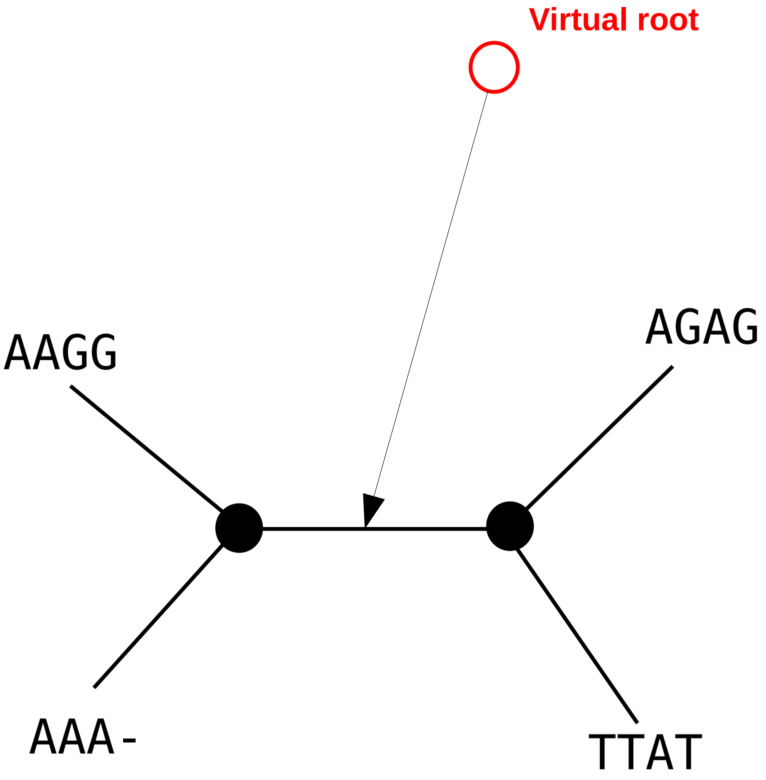
Gaps (also called Indels → Insertions or Deletions) are treated as so-called undetermined characters, also frequently denoted as ***N*****.**
The interpretation is that *N* could be either *A, C, G*, or *T*.

Parsimony

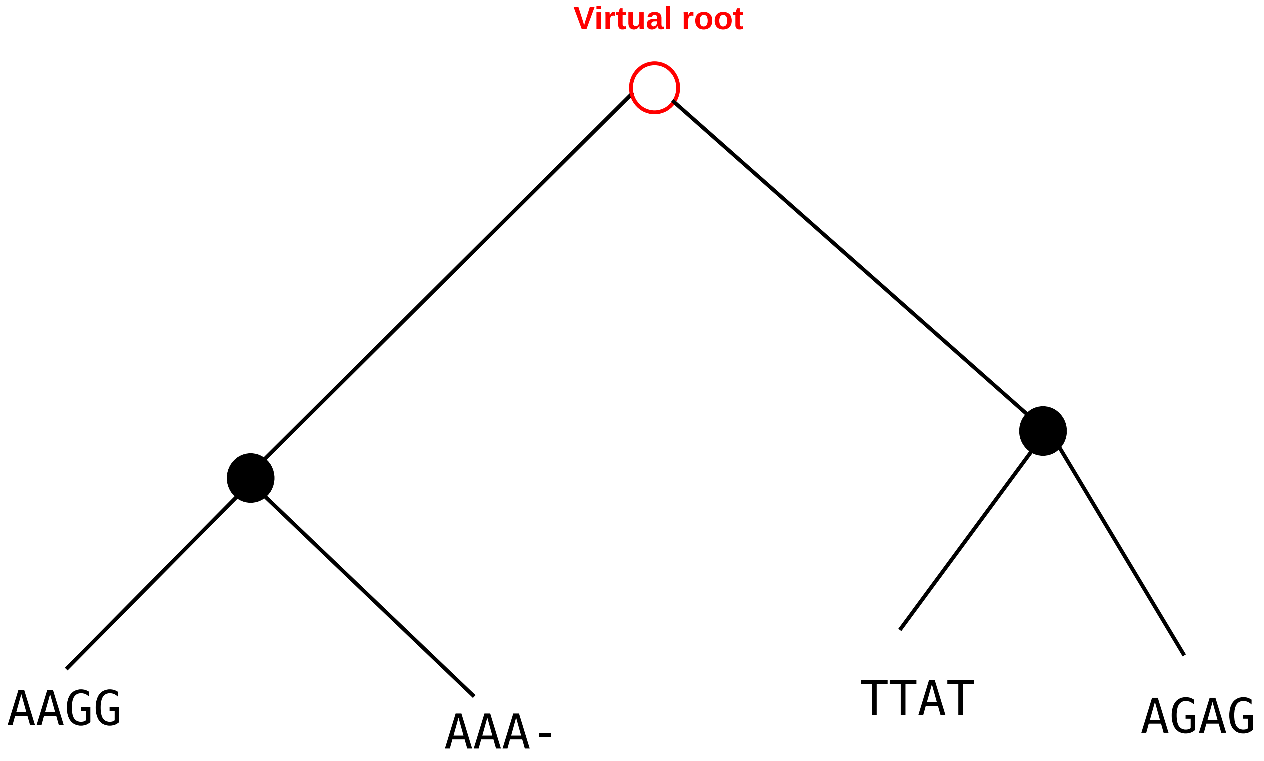


So, how do we compute the score?

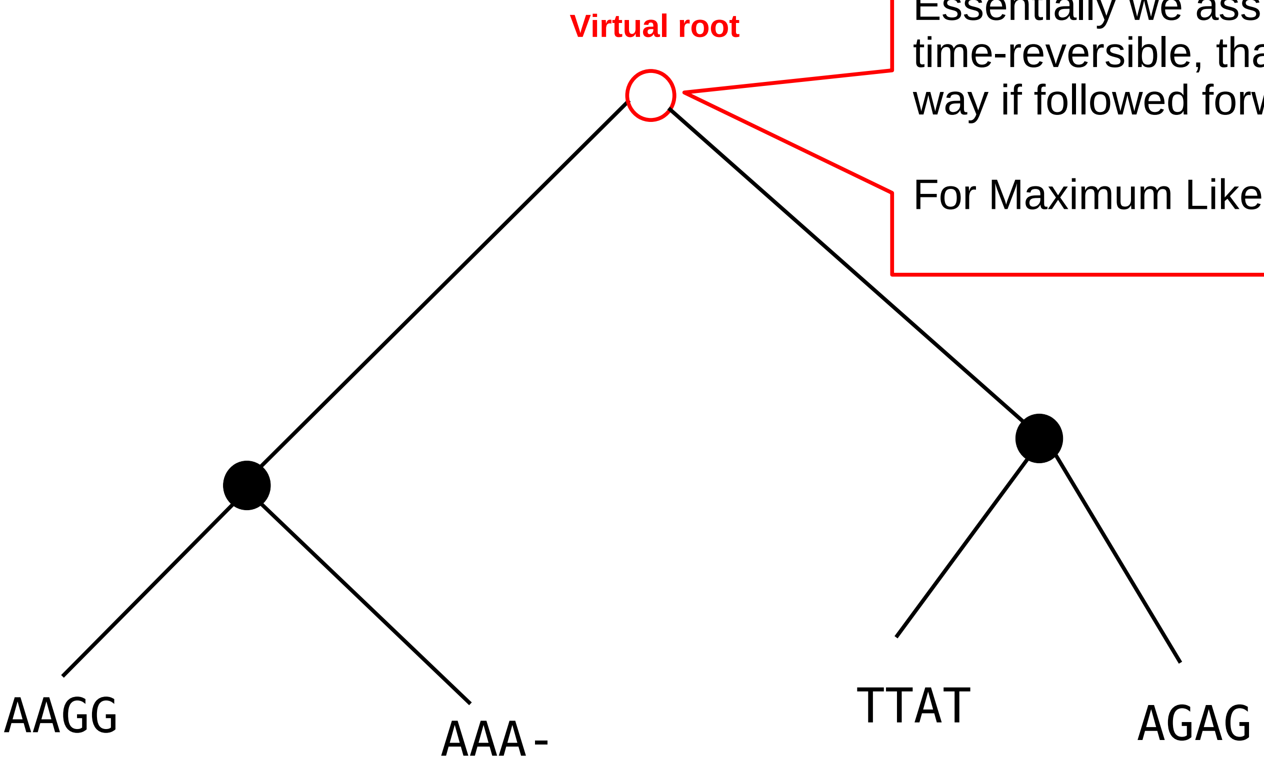
Parsimony



Parsimony



Parsimony



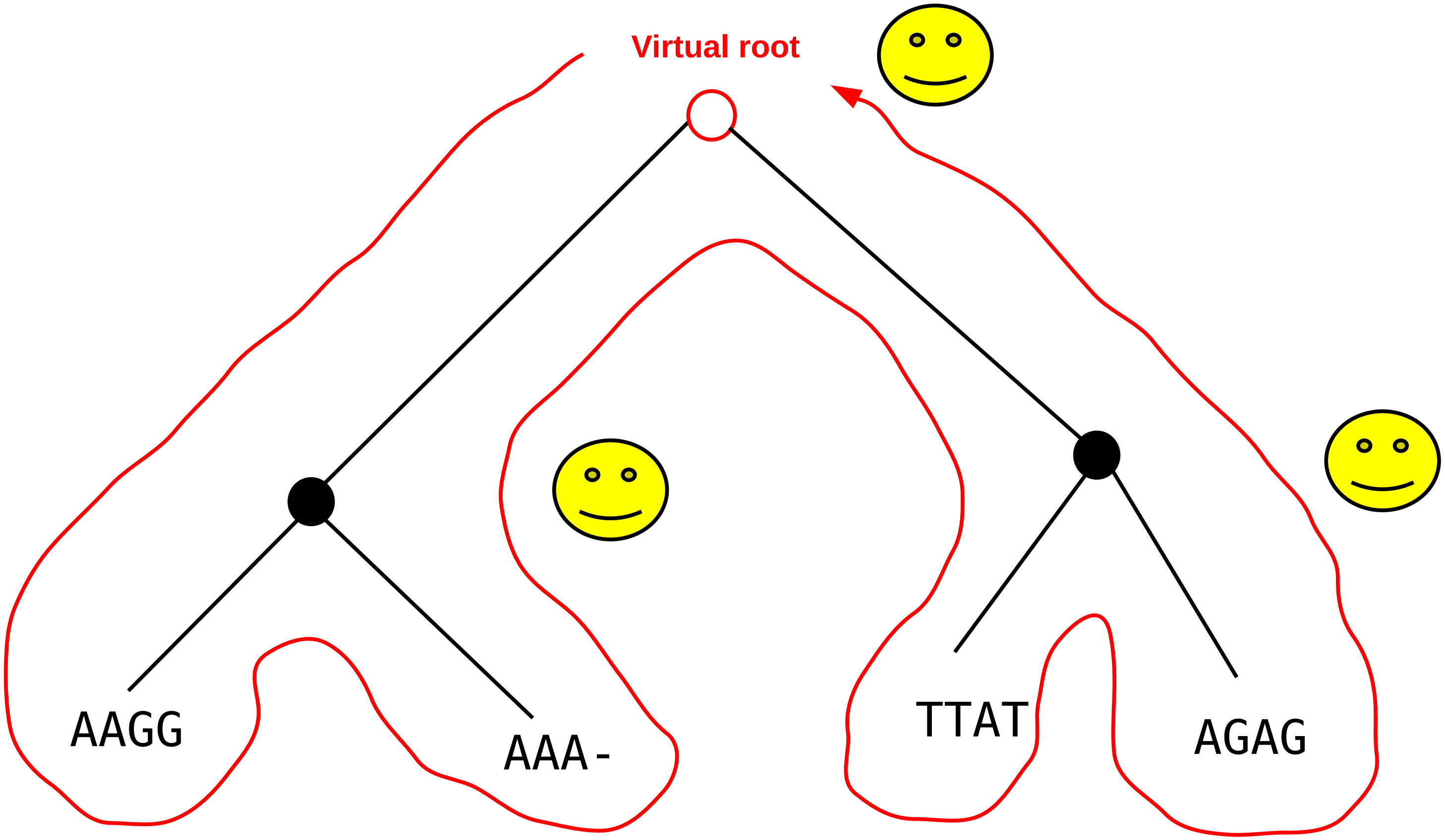
The score will be the same, regardless of where we place the root
→ makes computations much easier

Essentially we assume that evolution is time-reversible, that is, it occurred in the same way if followed forward or backward in time

For Maximum Likelihood we use a similar concept!

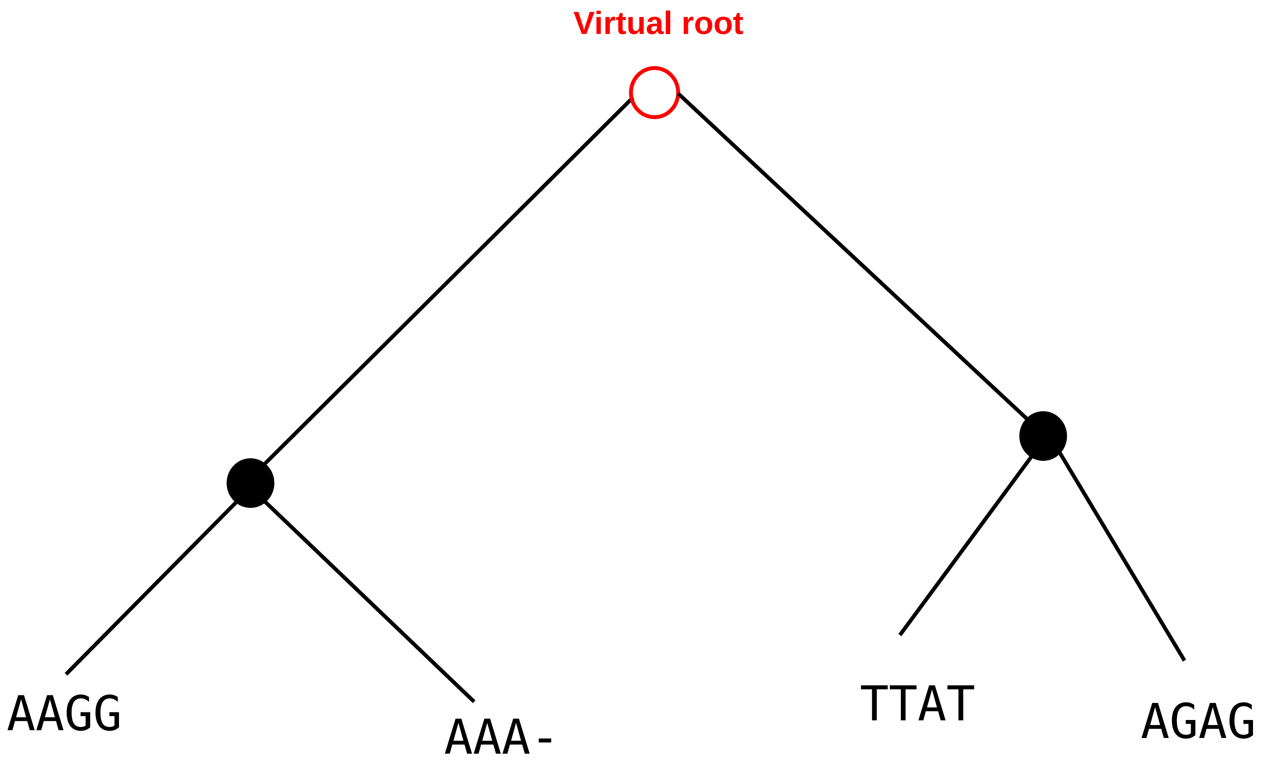
Parsimony

Post-order traversal to compute inner states

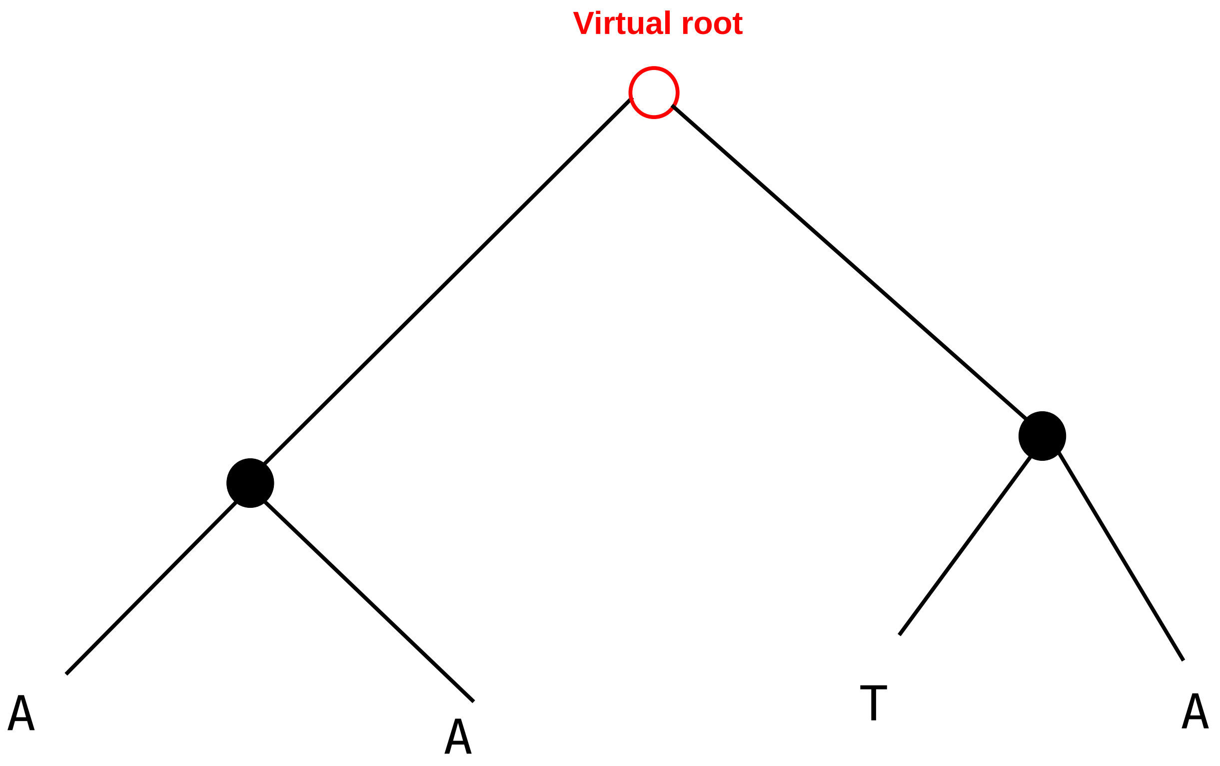


Parsimony

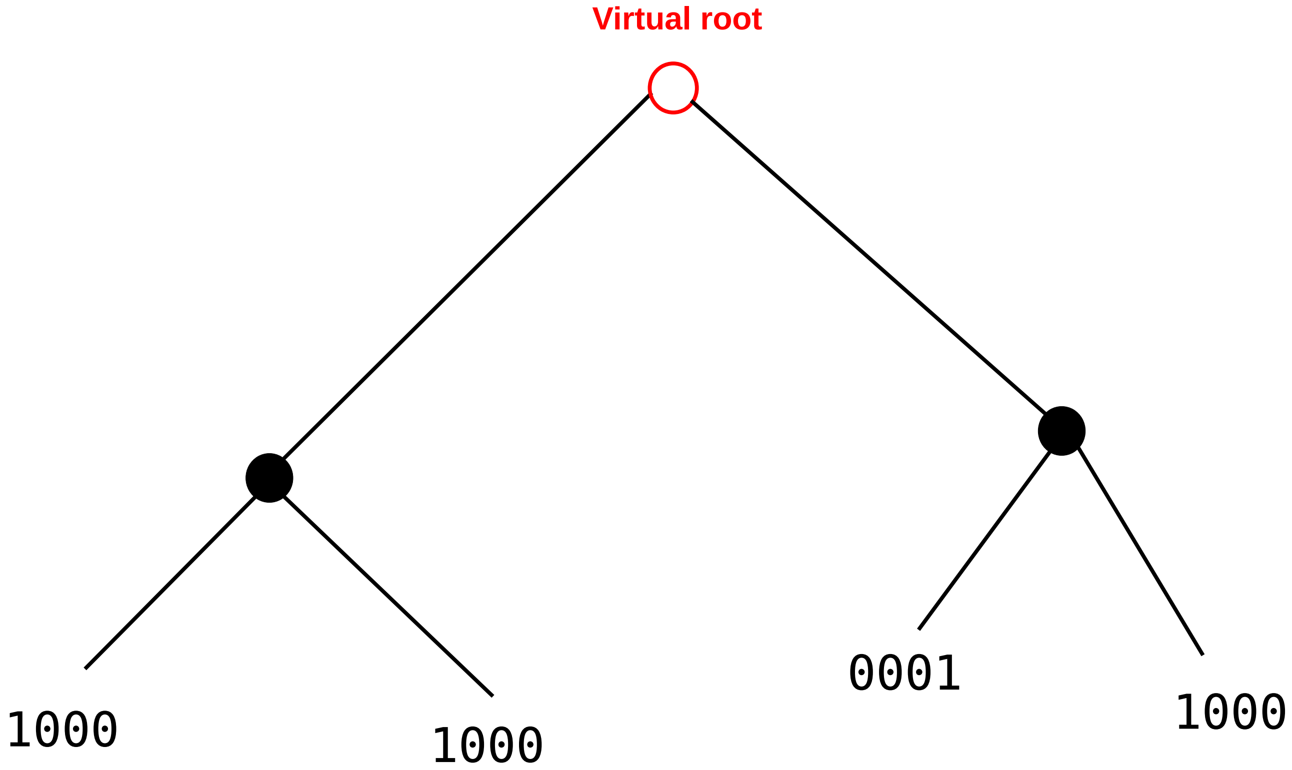
Compute scores on a site-per-site basis
→ we assume that sites evolve independently!



Parsimony



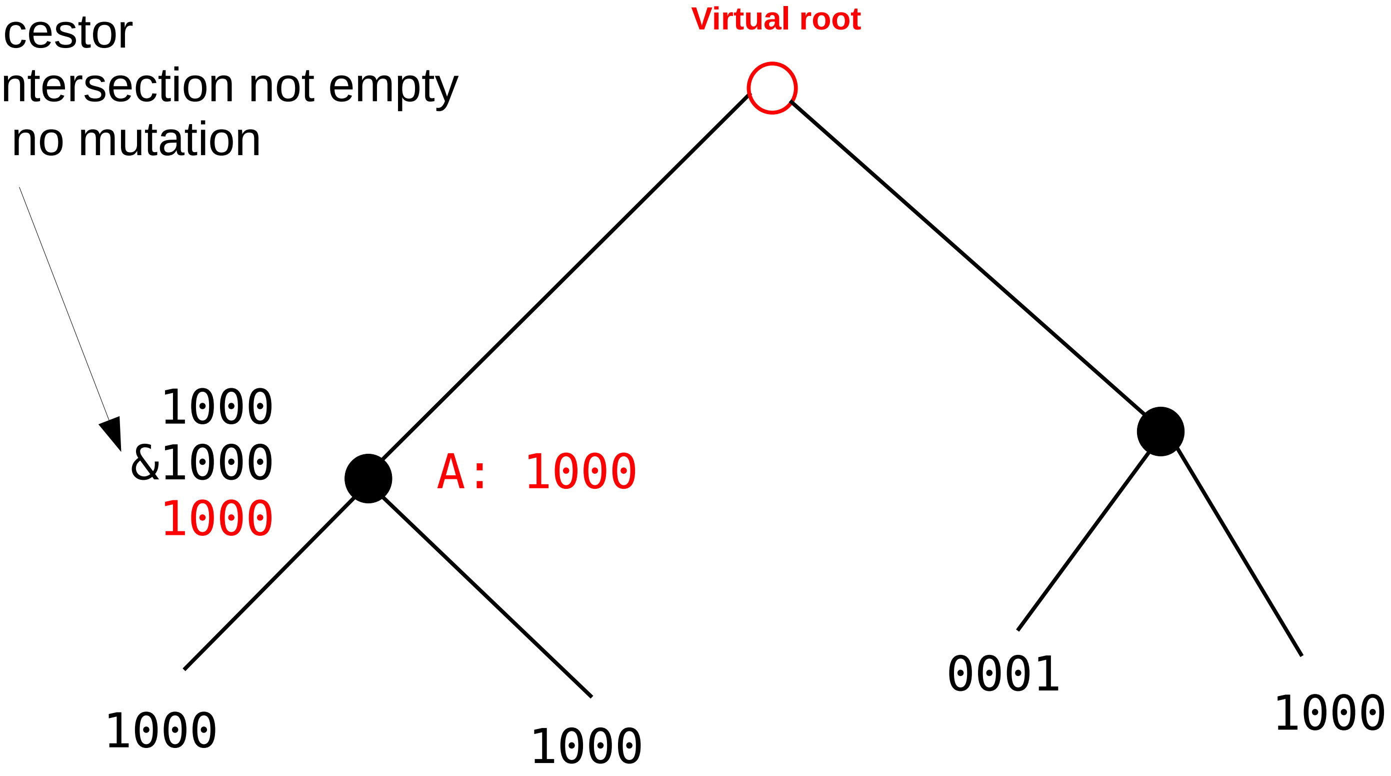
Parsimony



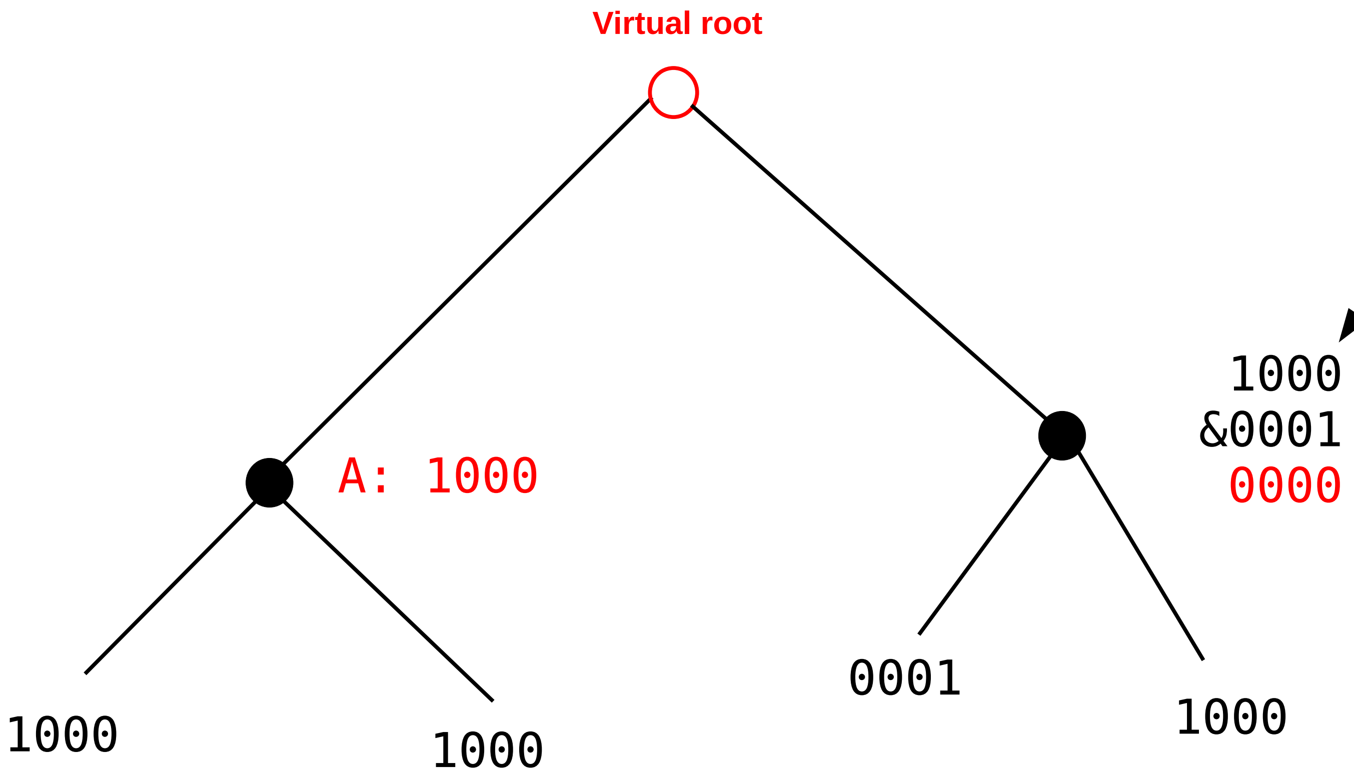
Parsimony

Intersection of sets of possible states at ancestor

If intersection not empty → no mutation



Parsimony

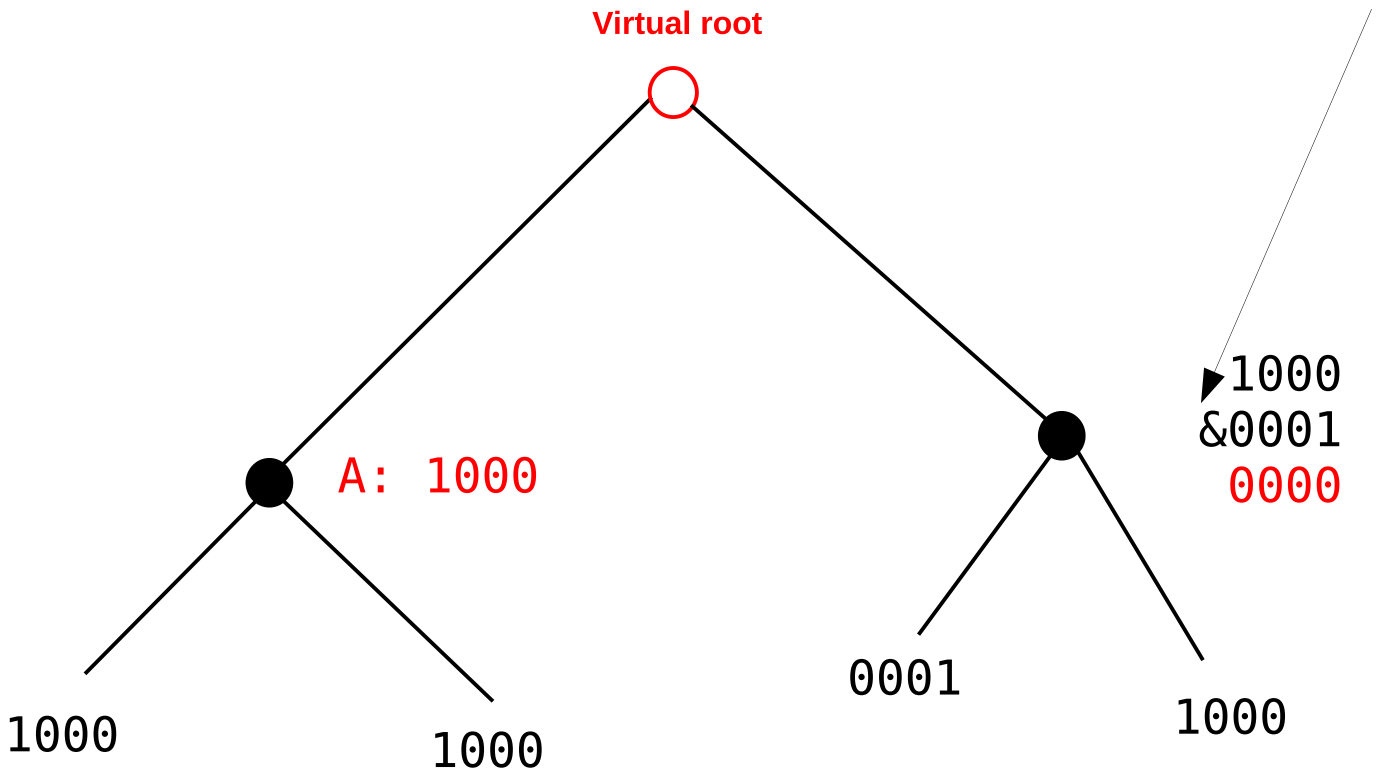


Intersection empty
→ count one mutation

Parsimony

+1

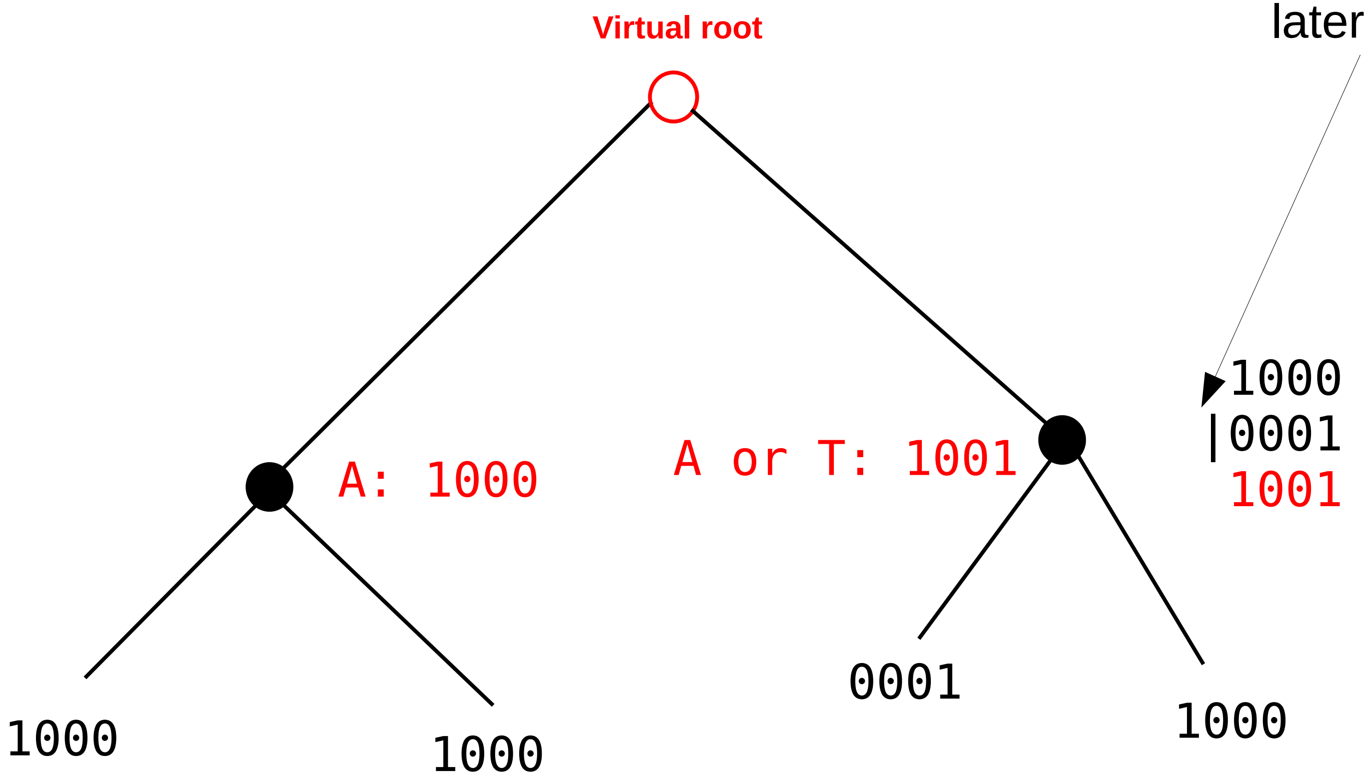
Intersection empty
→ count one mutation



Parsimony

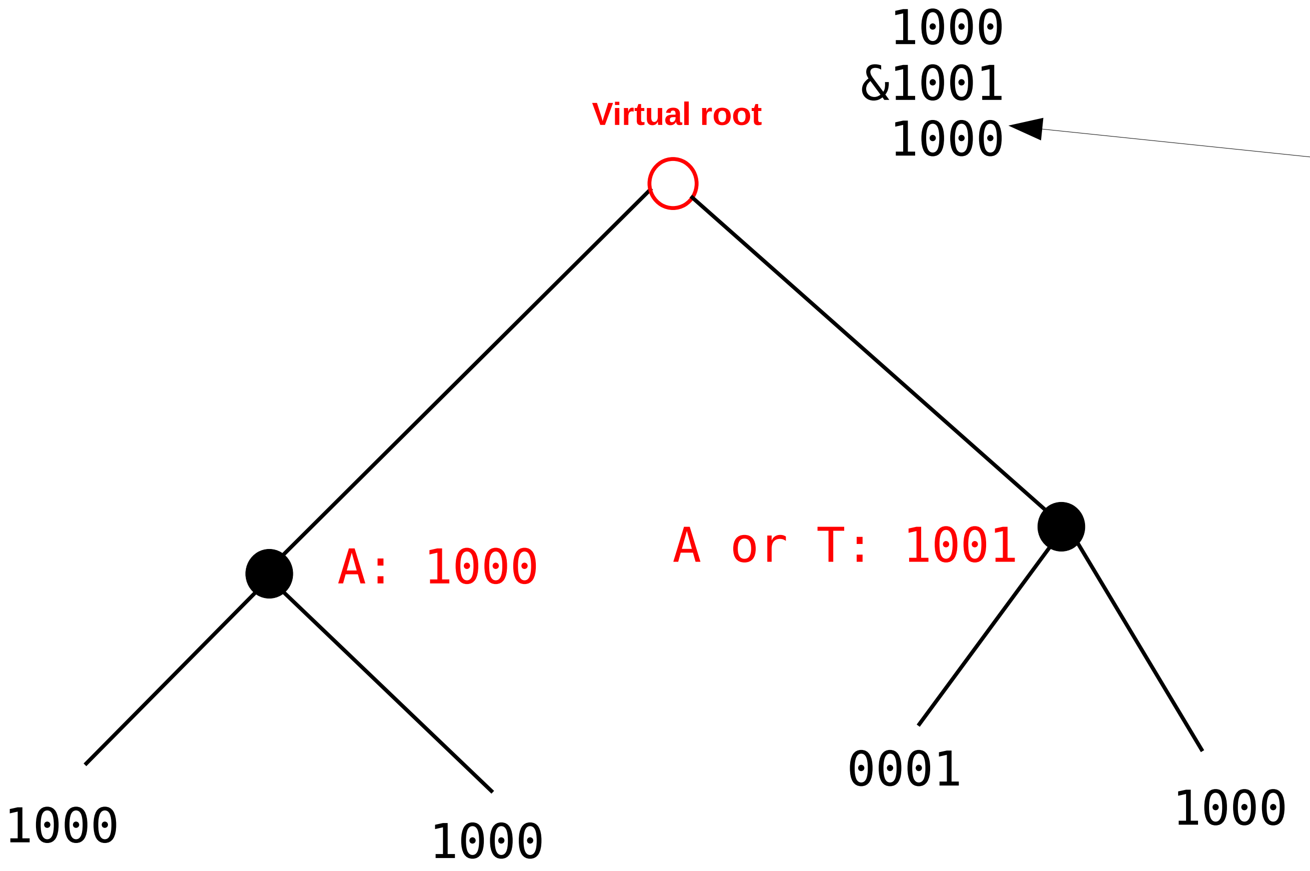
1

Ancestral state: **union!**
→ key trick, defer decision
Whether this is a T or A to a
later point!



Parsimony

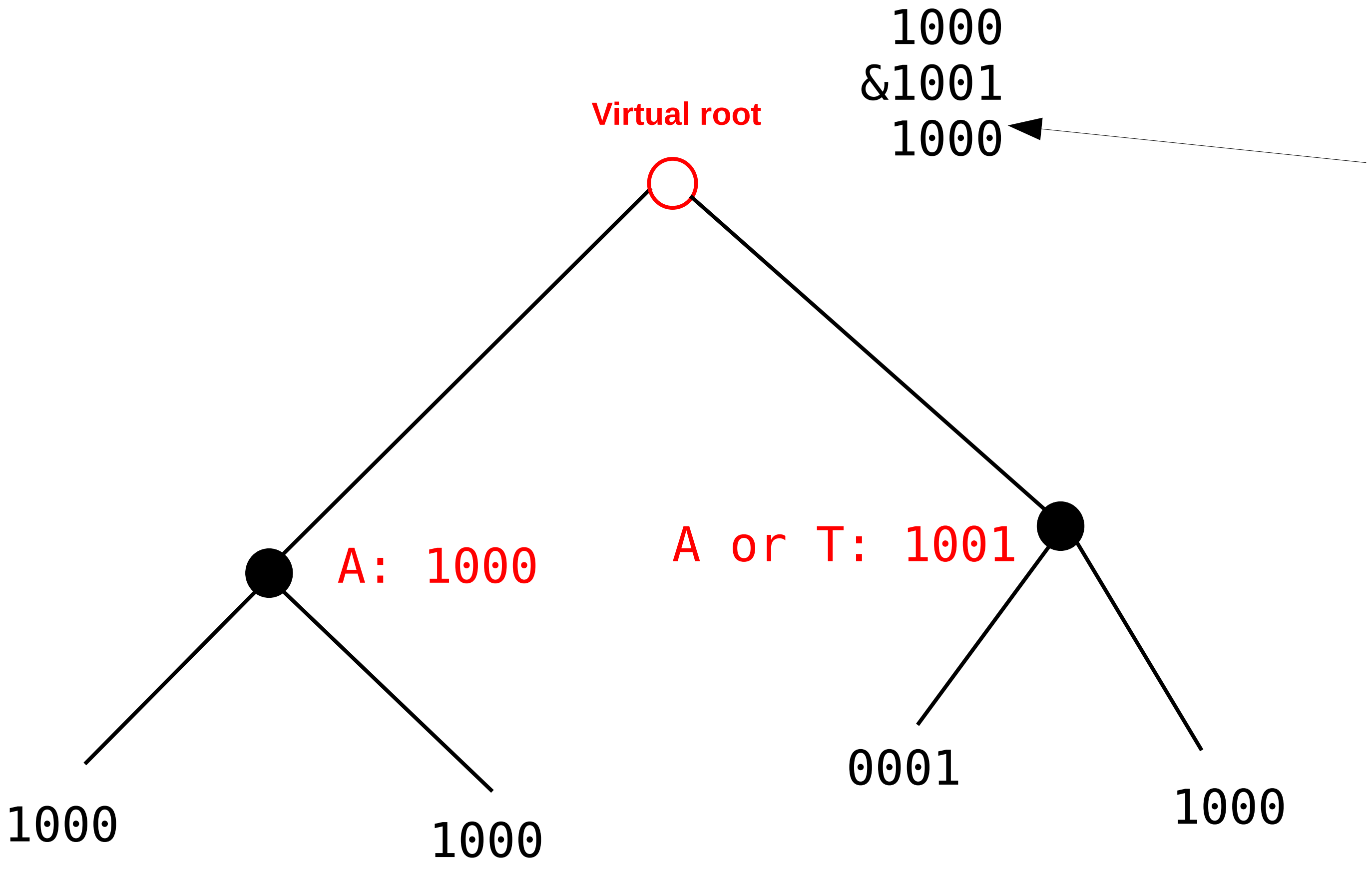
1



Intersection non-empty,
Overall score for this
Site: 1

Parsimony

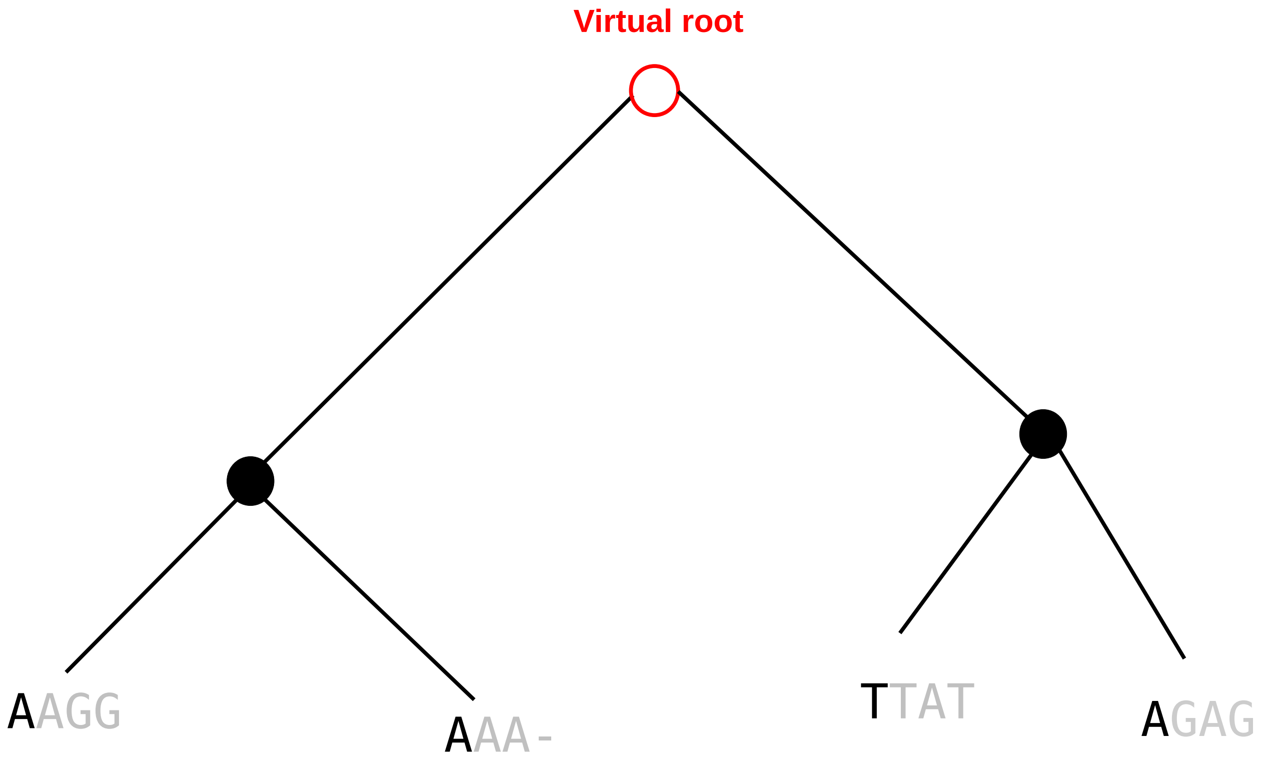
1



Intersection non-empty,
Overall score for this
Site: 1

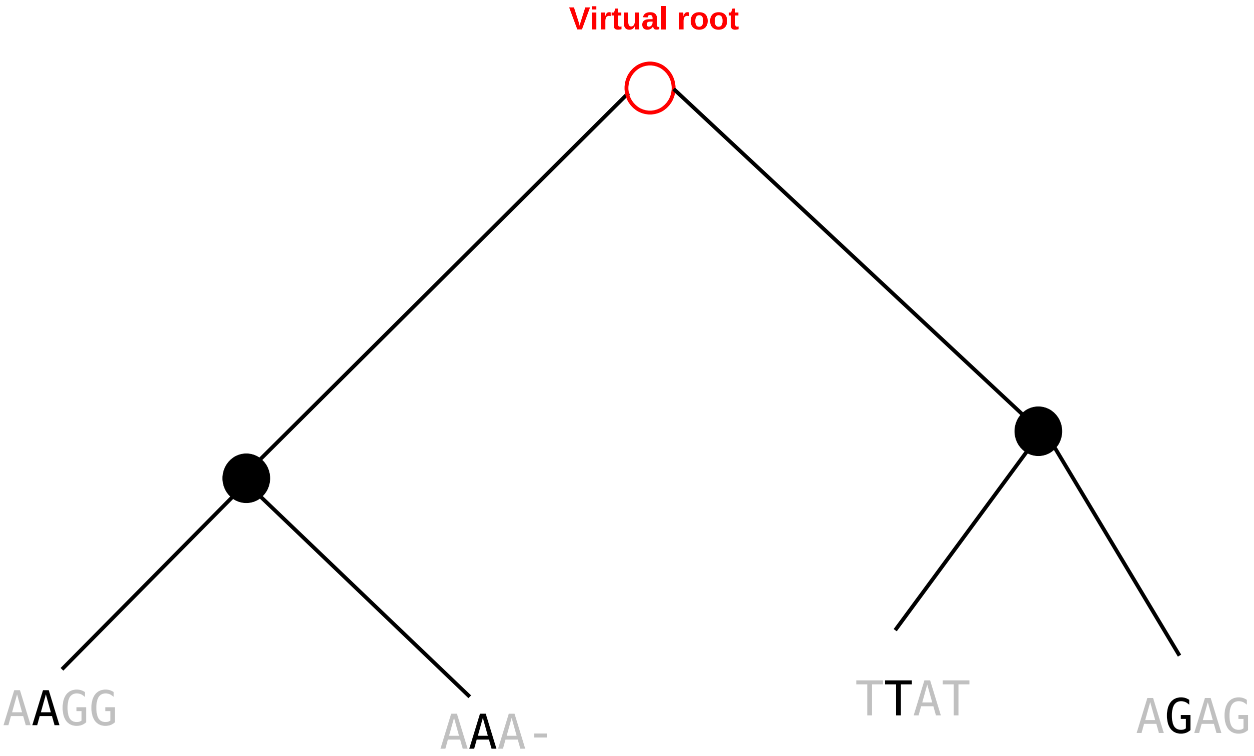
Parsimony

1



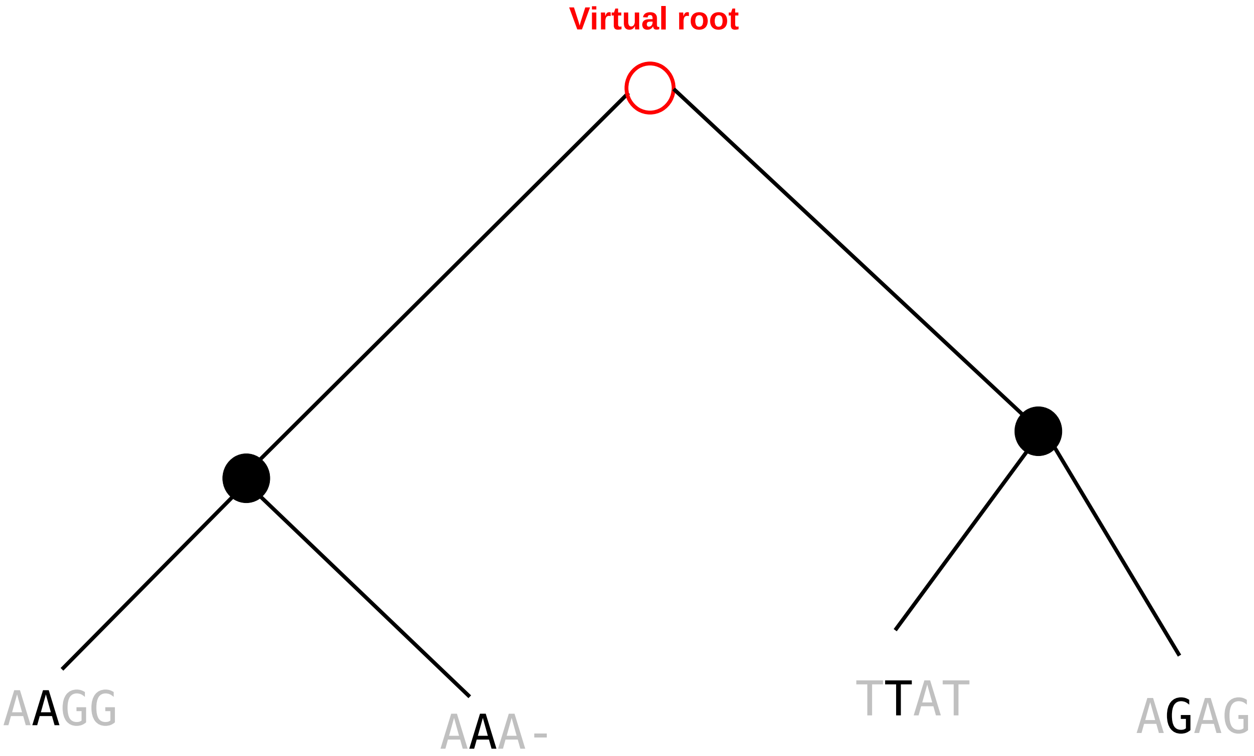
Parsimony

1+?



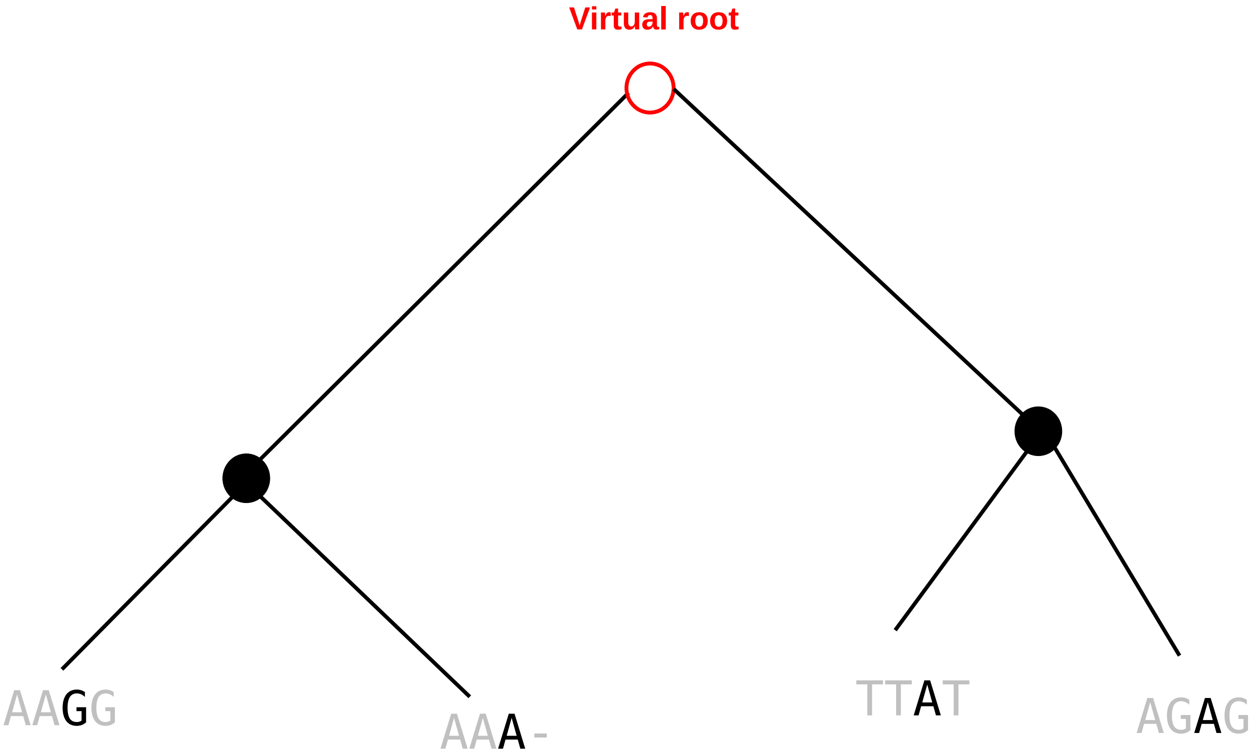
Parsimony

1+2



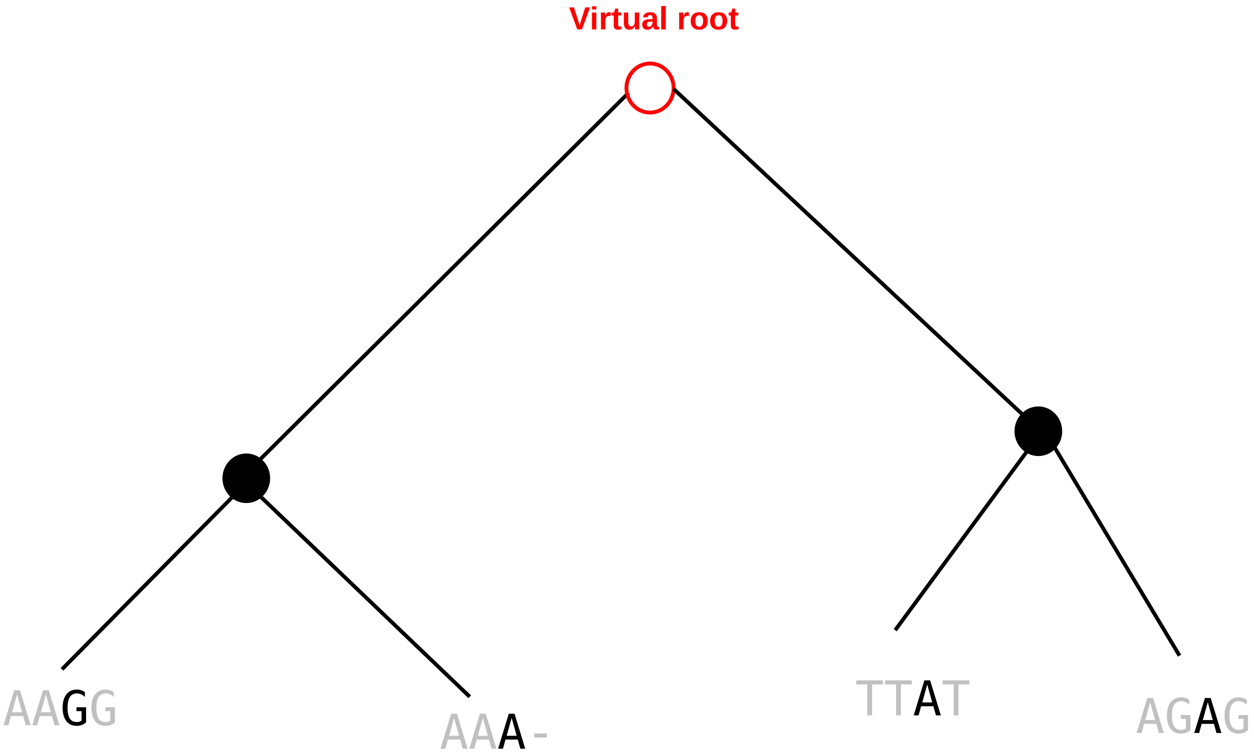
Parsimony

1+2+?



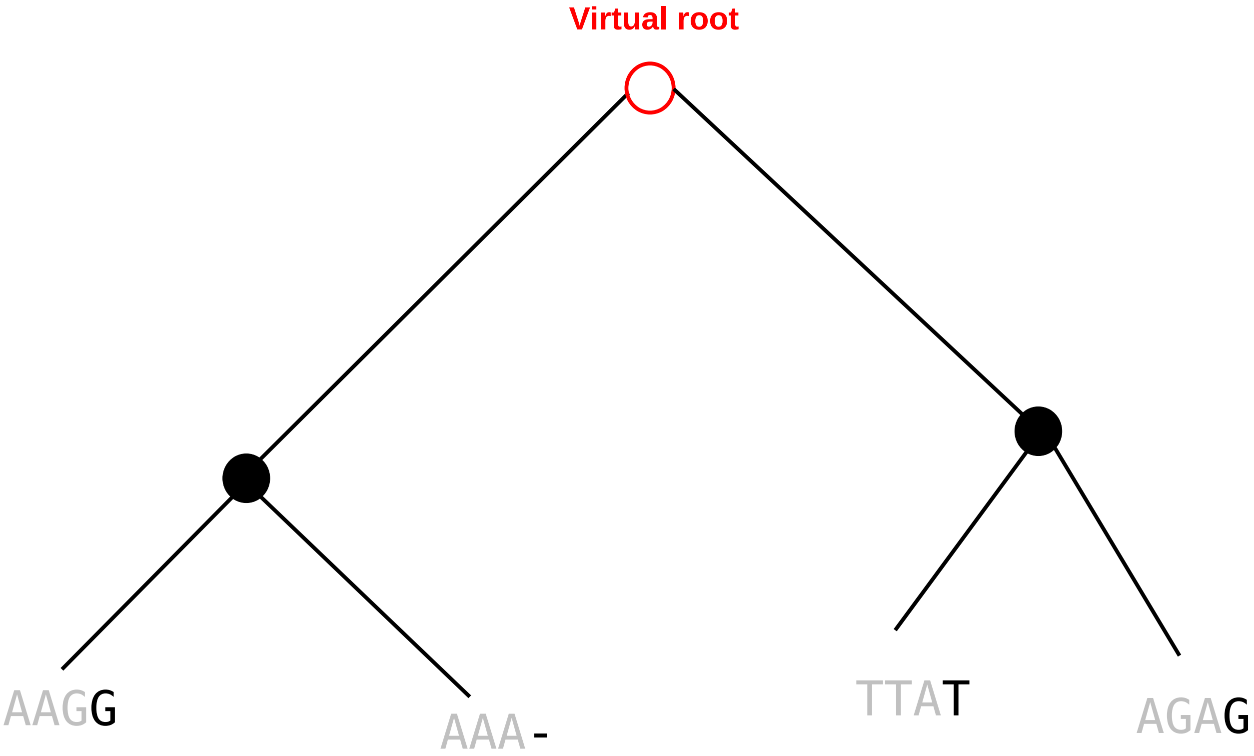
Parsimony

1+2+1



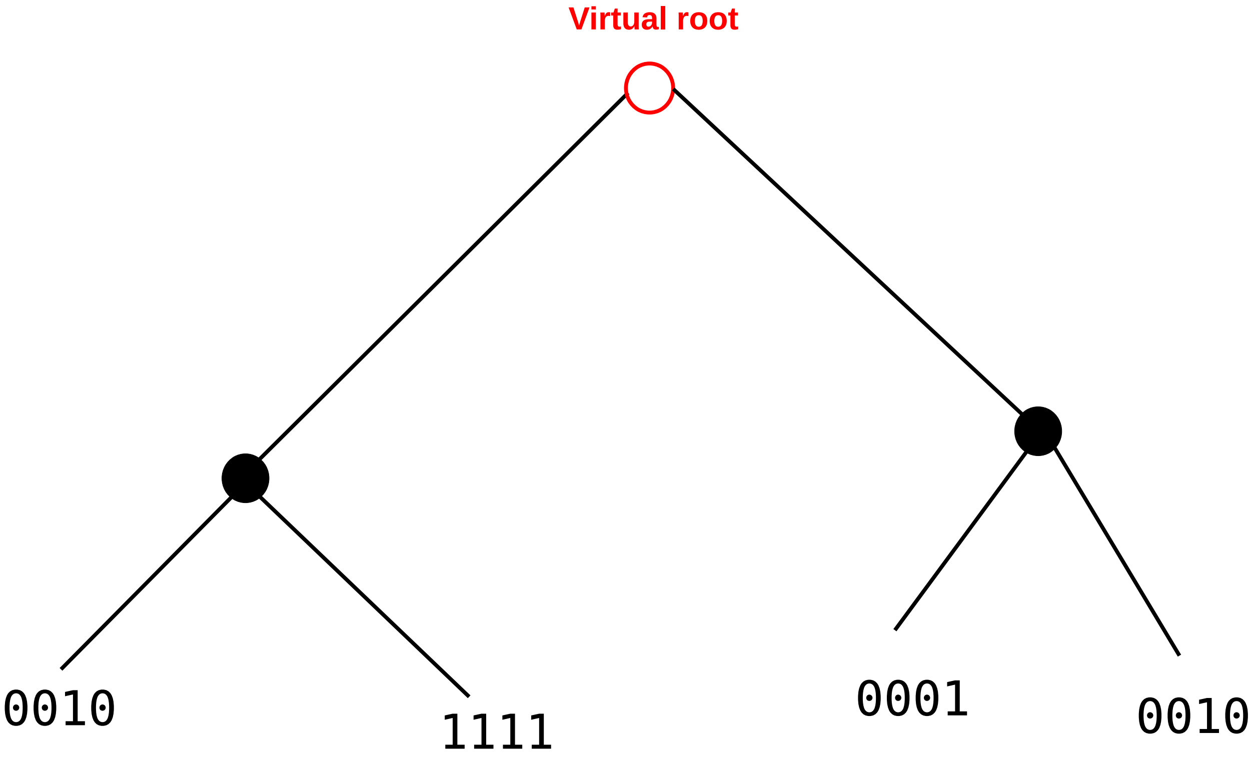
Parsimony

1+2+1



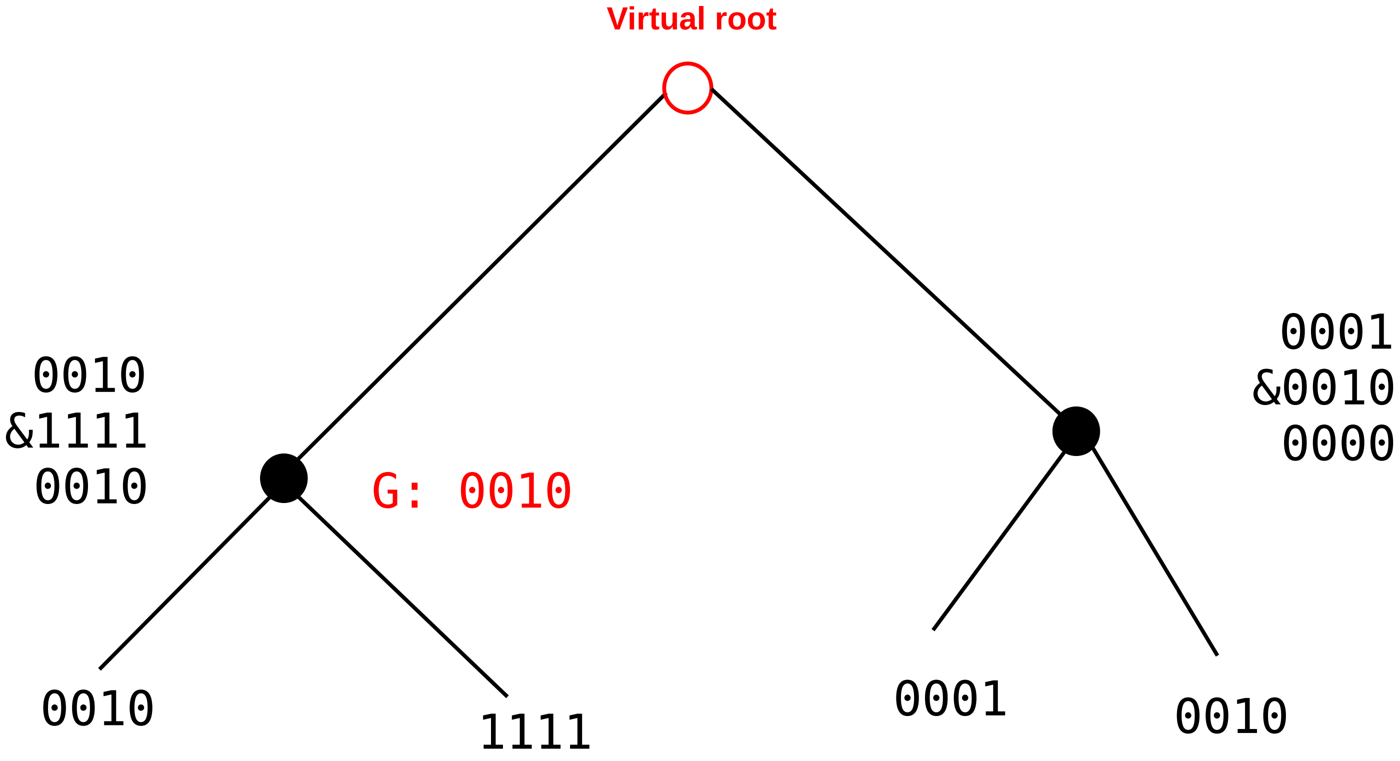
Parsimony

1+2+1



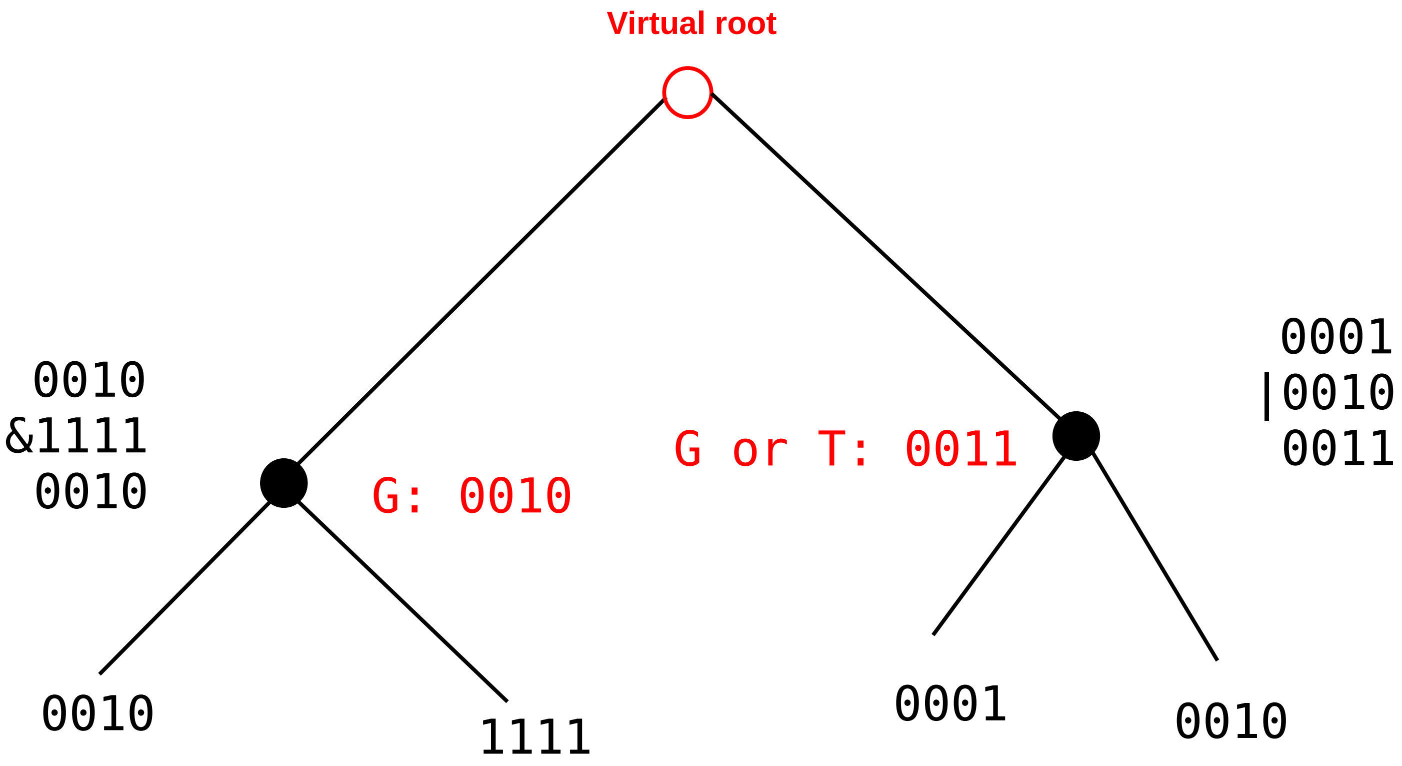
Parsimony

1+2+1



Parsimony

1+2+1+?



Virtual root

0010
&1111
0010

G: 0010

G or T: 0011

0001
|0010
0011

0010

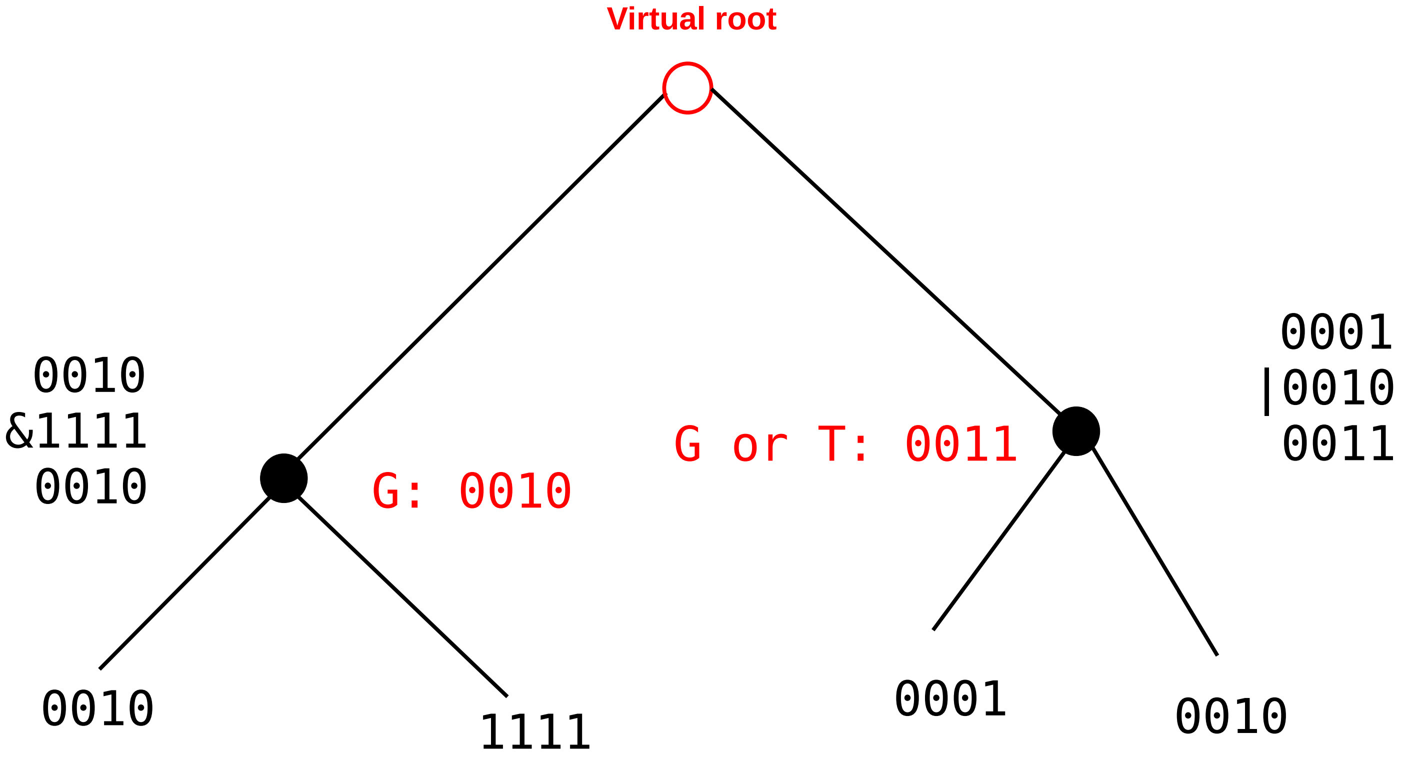
1111

0001

0010

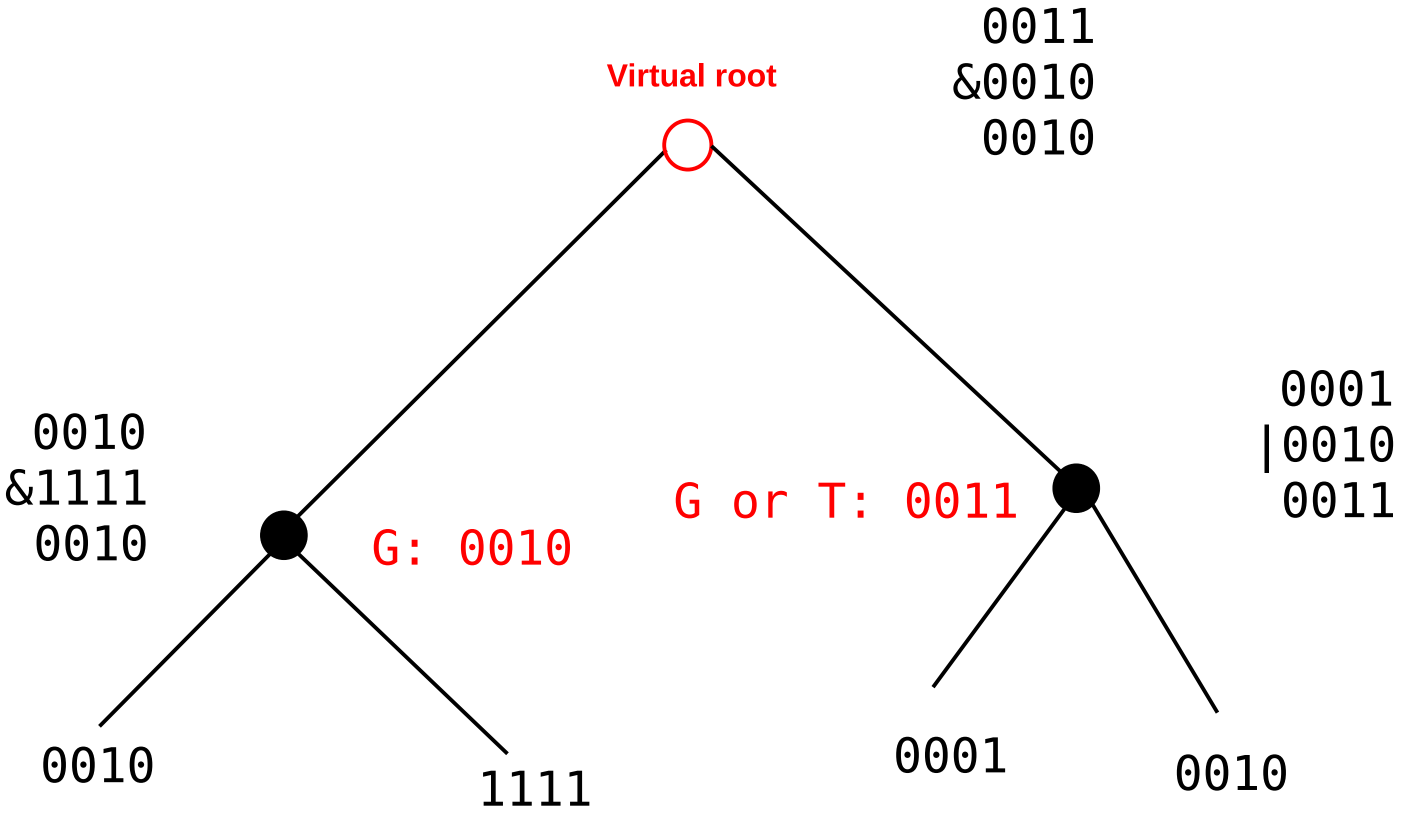
Parsimony

1+2+1+1



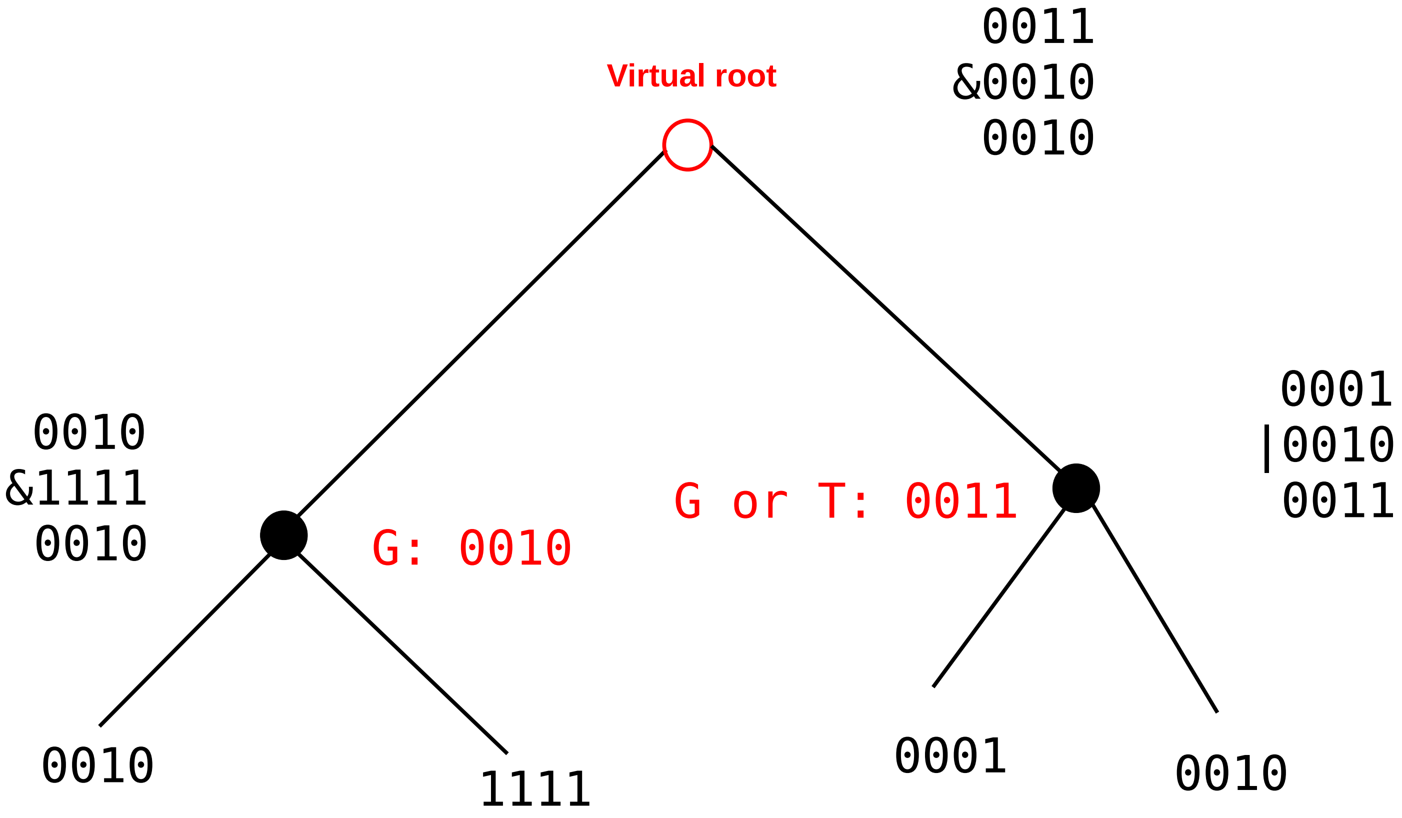
Parsimony

1+2+1+1



Parsimony

1+2+1+1= 5



Virtual root

0011
&0010
0010

0010
&1111
0010

G: 0010

G or T: 0011

0001
|0010
0011

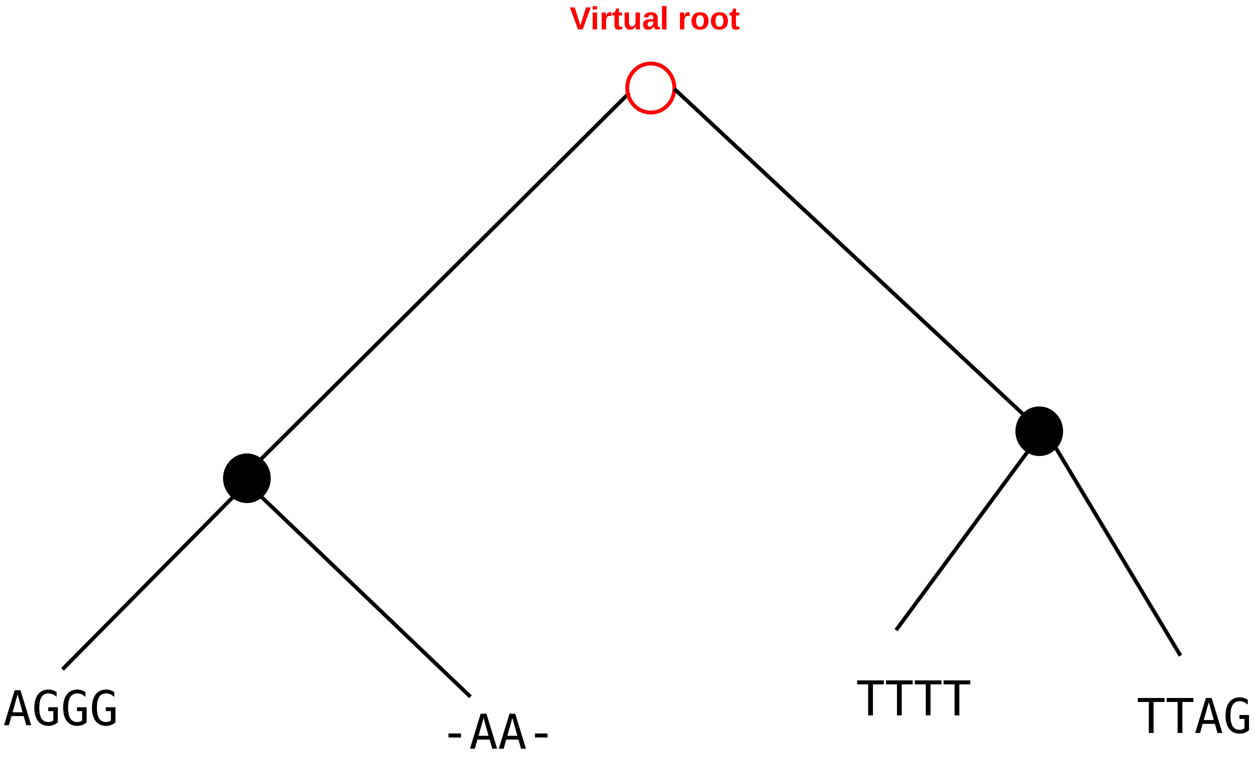
0010

1111

0001

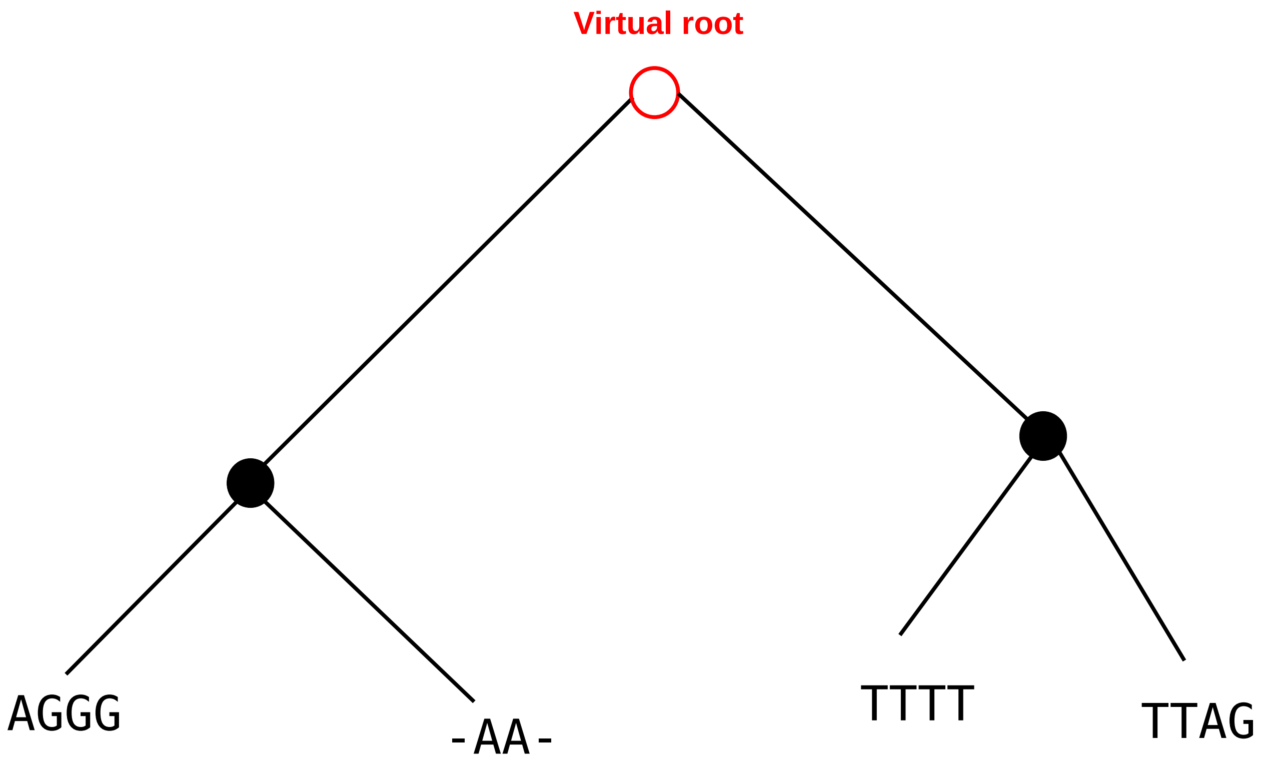
0010

Exercise: What's the parsimony score of this tree?



Exercise: What's the parsimony score of this tree?

1+2+2+1=6



Parsimony

- Time complexity to score one tree

  MSA with *n* taxa and *m* sites

  → *(n-2) * m* calculations; *n-2* is the number of inner nodes of a tree with *n* taxa

  → *O(nm)*, but the constant hidden in *O()* is very small

- Space complexity *DNA* data

  → alignment: *n * m* * 4 bits

  → ancestral nodes: *(n-2) * m* * 4 bits

  → score counter: *(n-2)* * 32 bits

  → space complexity *O(nm)*, but the constant hidden in *O()* is very small

- **Maximum Likelihood**: same time & space complexity, but constants much, much larger!

Parsimony Implementation Notes

- Intersections and Unions can be implemented efficiently at the bit-level
- 4 bits for one DNA character (remember, ambiguous character encoding)
- Plain implementation: 32 bits
- `SSE3` vector intrinsics: 128 bits
- `AVX` vector intrinsics: 256 bits
- `Parsimonator` program (www.exelixis-lab.org/software.html)
  - → uses SSE3 and AVX intrinsics
  - → I will show a demo now
  - → Implements simple search algorithm
  - → probably fastest available open-source parsimony implementation

Parsimony Implementation Notes

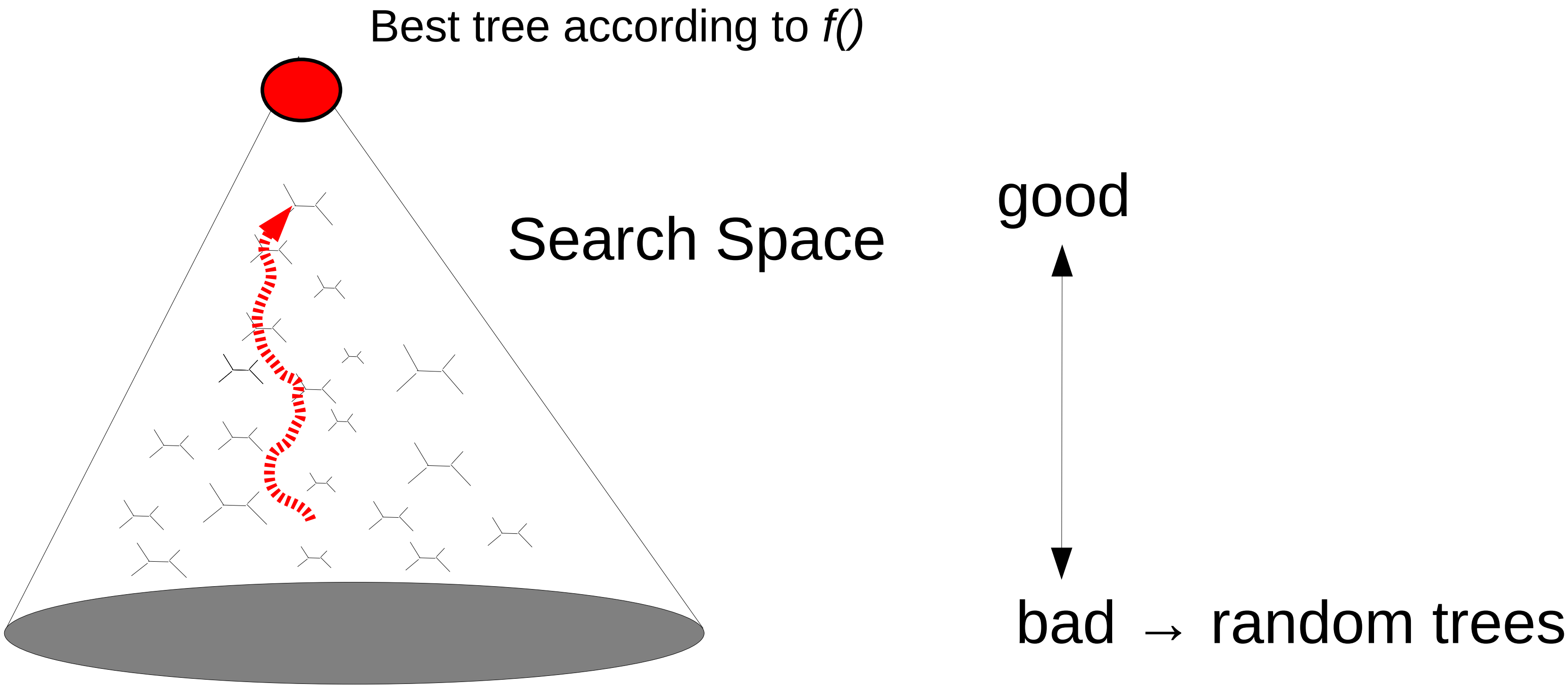
- Without going into the details:
- In the `parsimonator` implementation we need to compute a so-called population count (`popcount`) that computes the number of bits (# mutations) that are set to `1` in a 32-, 128-, or 256-bit word
- `popcount` is a very important operation
- There are various fancy bit-twisting implementations for fast `popcounts`
- In fact, this operation is so important that modern `x86` architectures have a dedicated HW-based `popcount`
- You can use it in C code via `__builtin_popcount(x)`

Parsimony Implementation Notes

- Why did we write `parsimonator`?
- A paper was published that claimed to have achieved a FPGA-based acceleration of the parsimony function of up to factor 10,000
- Remember: the speedup is defined as *T(1)/T(N)*, where *T(1)* is the fastest available sequential implementation/algorithm!
- Compared to Parsimonator (`AVX` version), the corresponding FPGA design achieved a speedup of up to 10, only!
- N. Alachiotis, A. Stamatakis: "FPGA Acceleration of the Phylogenetic Parsimony Kernel**?**", *FPL 2011*.

How do we search for “good” trees under any criterion?

Search Space



Best tree according to *f()*

Search Space

good

bad → random trees

Tree Search Algorithms

- How do we obtain an initial starting tree with *n* taxa → comprehensive tree

  → NJ or UPGMA tree

  → random tree

  → stepwise addition algorithm

- How do we change such a comprehensive tree to improve its score?

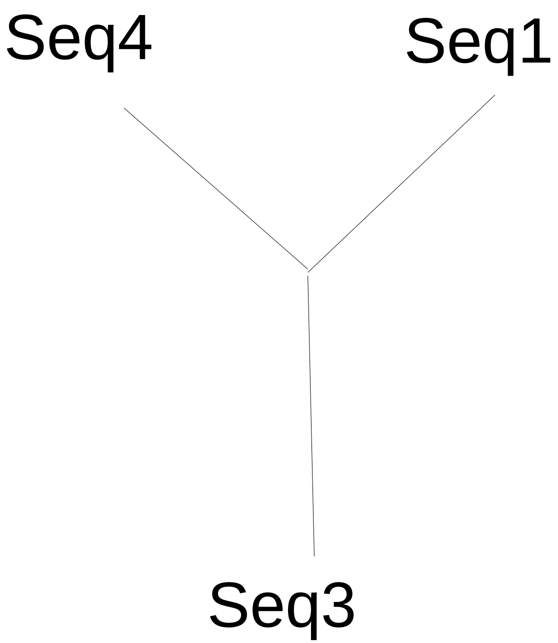
  Scores can be improved with optimality criteria: least squares, minimum evolution, parsimony, maximum likelihood

Building a Random Tree

Seq0
Seq1
Seq2
Seq3
Seq4

Building a Random Tree

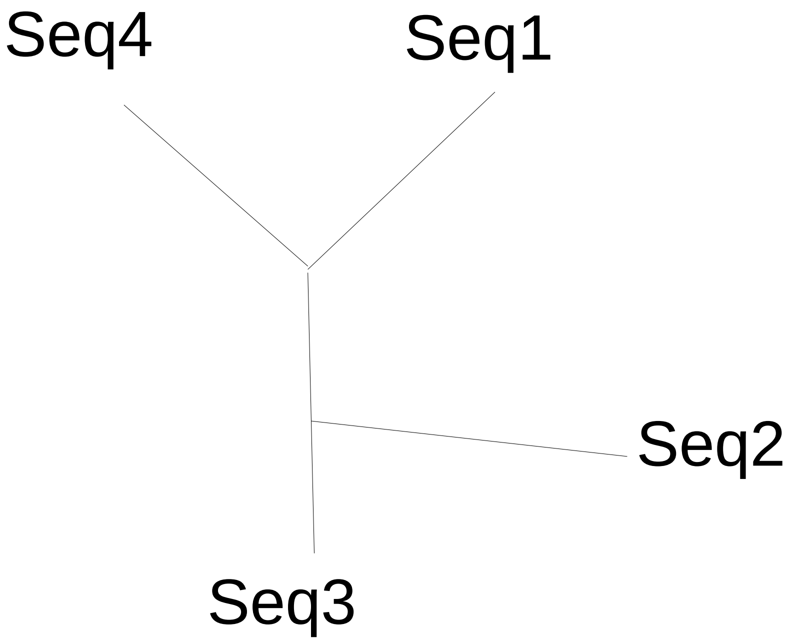
Seq0
~~Seq1~~
Seq2
~~Seq3~~
~~Seq4~~



Seq4
Seq1
Seq3

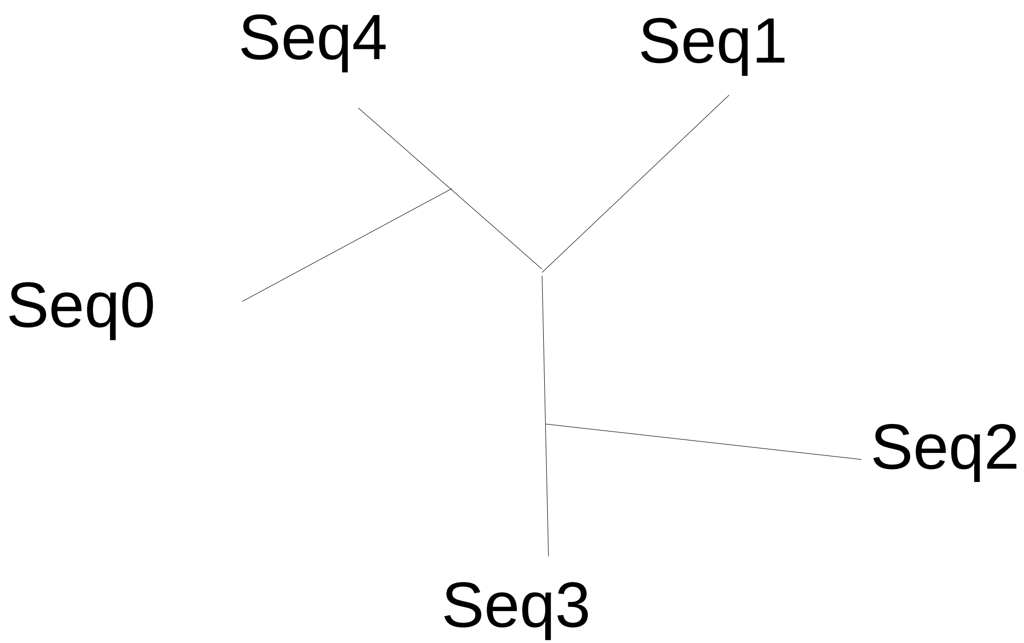
Building a Random Tree

Seq0
~~Seq1~~
~~Seq2~~
~~Seq3~~
~~Seq4~~



Building a Random Tree

~~Seq0~~
~~Seq1~~
~~Seq2~~
~~Seq3~~
~~Seq4~~

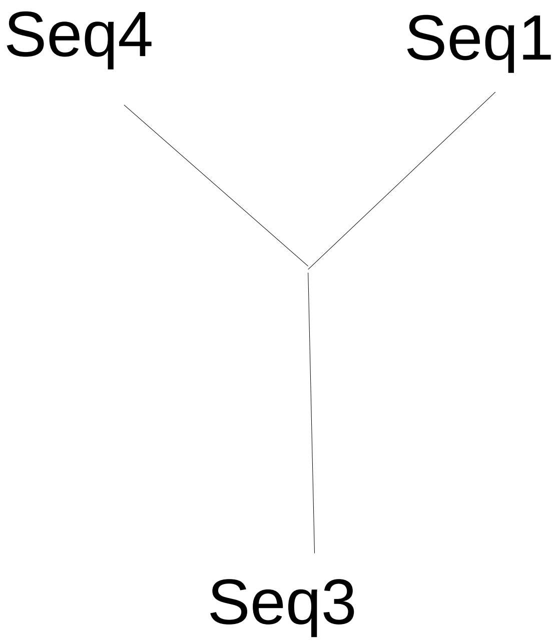


Randomized Stepwise Addition Order Algorithm

Seq0
Seq1
Seq2
Seq3
Seq4

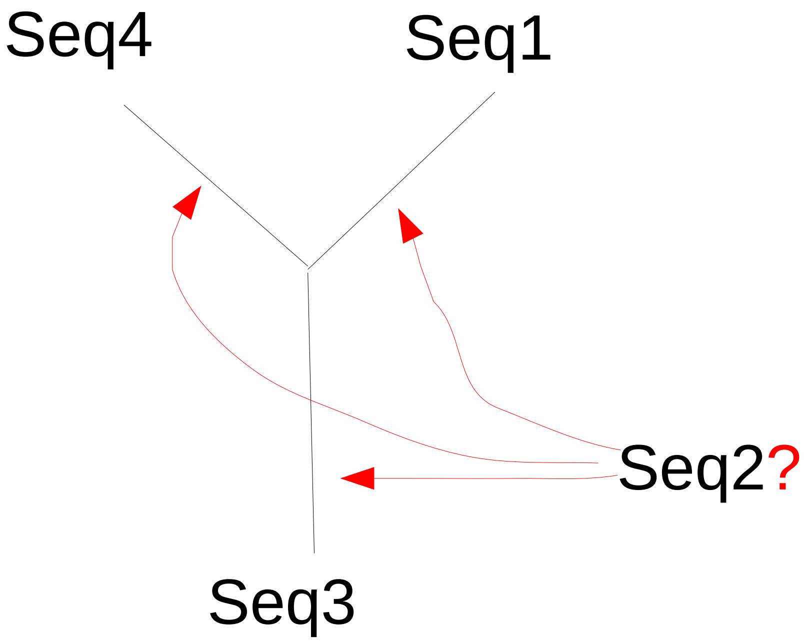
Randmoized Stepwise Addition Order Algorithm

Seq0
~~Seq1~~
Seq2
~~Seq3~~
~~Seq4~~



Randomized Stepwise Addition Order Algorithm

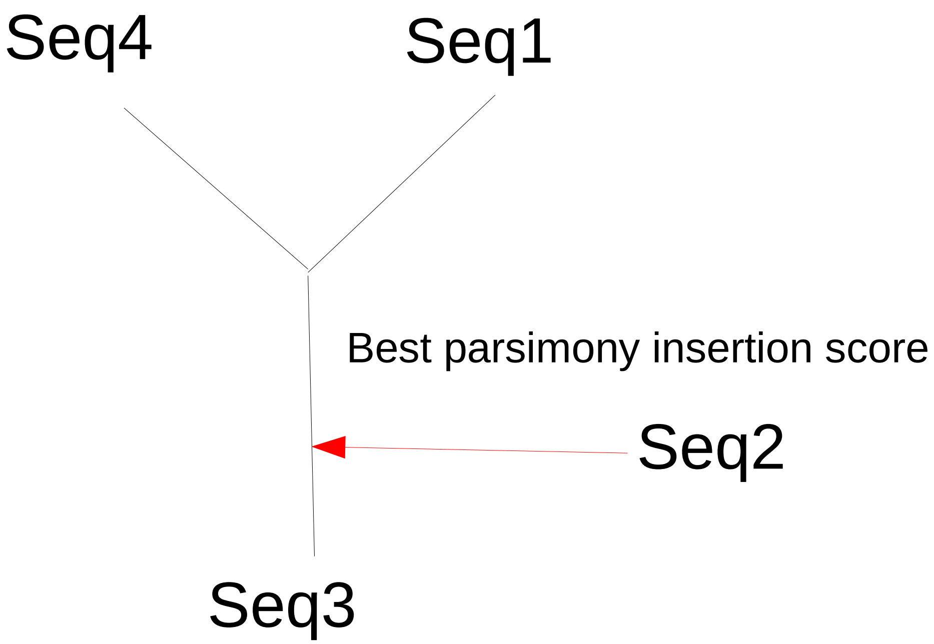
Seq0
~~Seq1~~
~~Seq2~~
~~Seq3~~
~~Seq4~~



Seq4
Seq1
Seq2?
Seq3

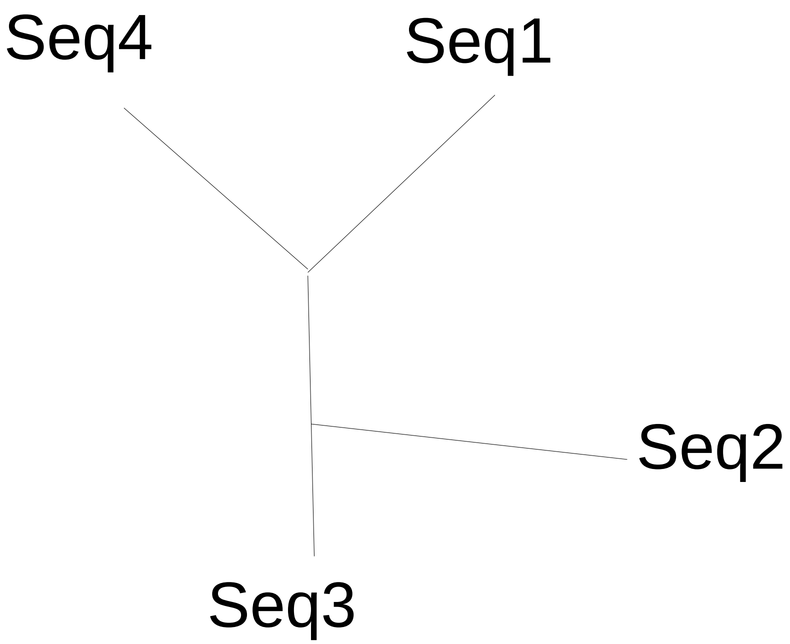
Randomized Stepwise Addition Order Algorithm

Seq0
~~Seq1~~
~~Seq2~~
~~Seq3~~
~~Seq4~~


Seq4
Seq1
Best parsimony insertion score
Seq2
Seq3

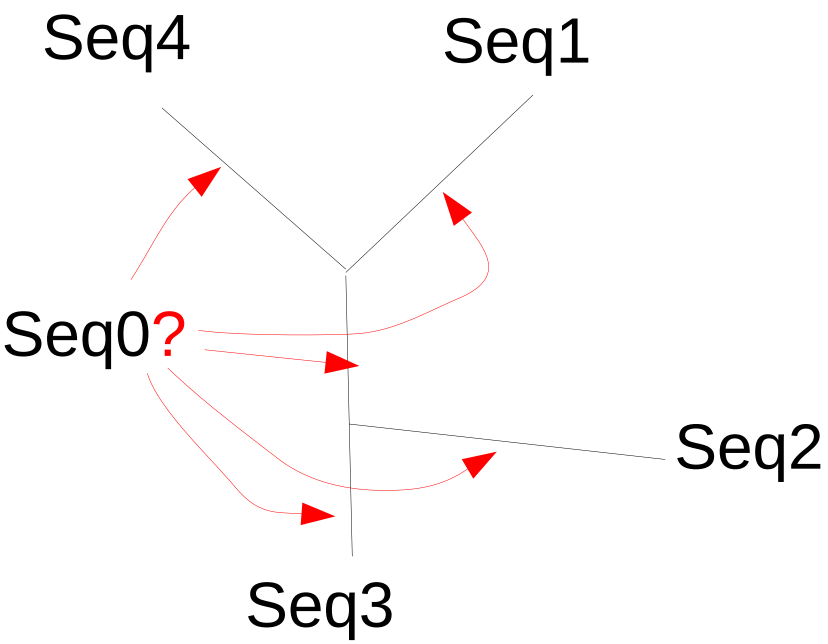
Randomized Stepwise Addition Order Algorithm

Seq0
~~Seq1~~
~~Seq2~~
~~Seq3~~
~~Seq4~~



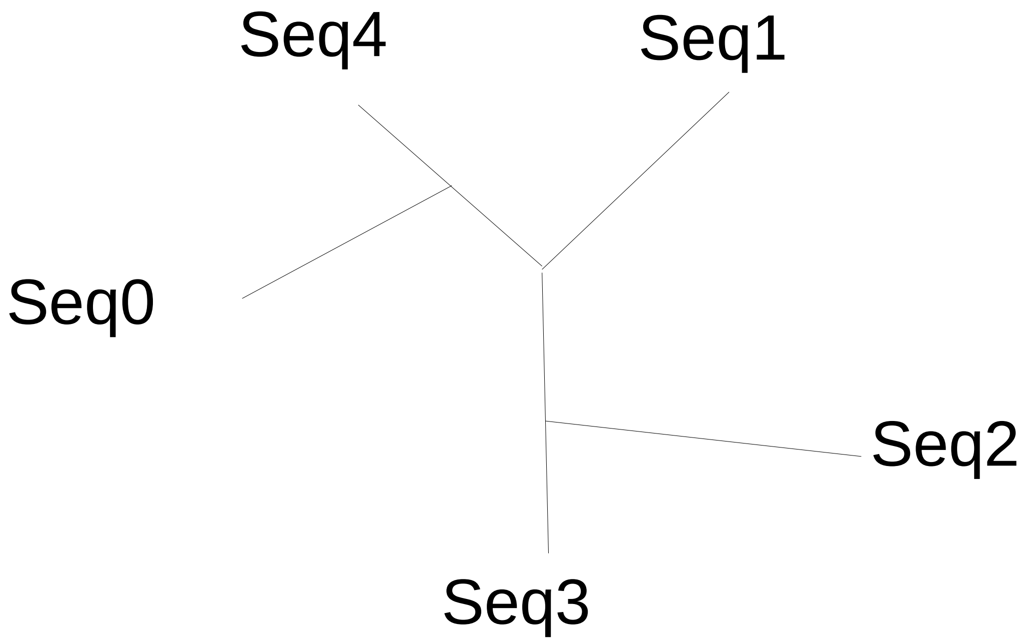
Randomized Stepwise Addition Order Algorithm

~~Seq0~~
~~Seq1~~
~~Seq2~~
~~Seq3~~
~~Seq4~~



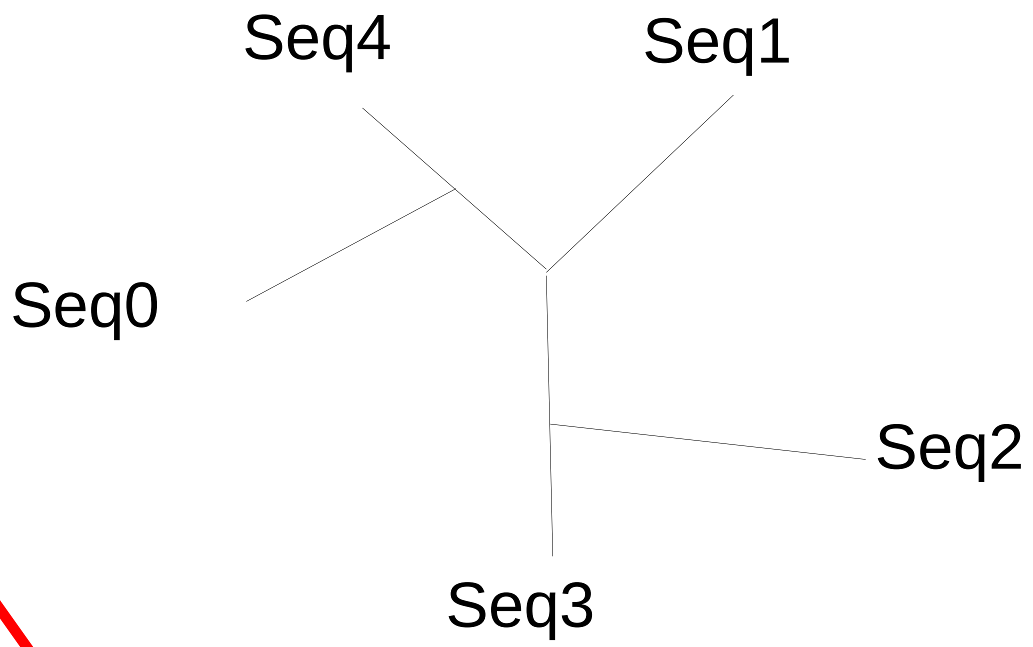
Randomized Stepwise Addition Order Algorithm

~~Seq0~~
~~Seq1~~
~~Seq2~~
~~Seq3~~
~~Seq4~~



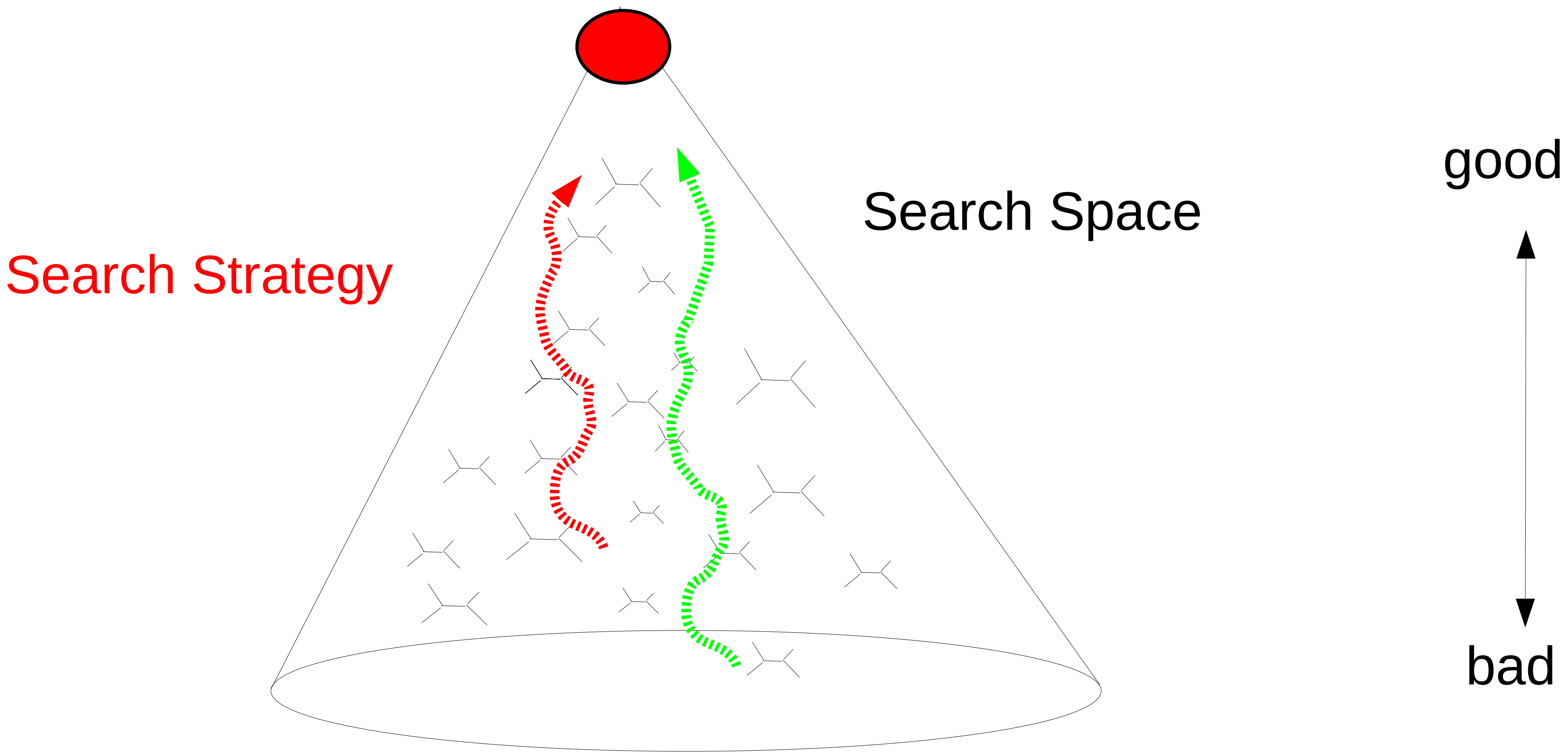
Randomized Stepwise Addition Order Algorithm

~~Seq0~~
~~Seq1~~
~~Seq2~~
~~Seq3~~
~~Seq4~~

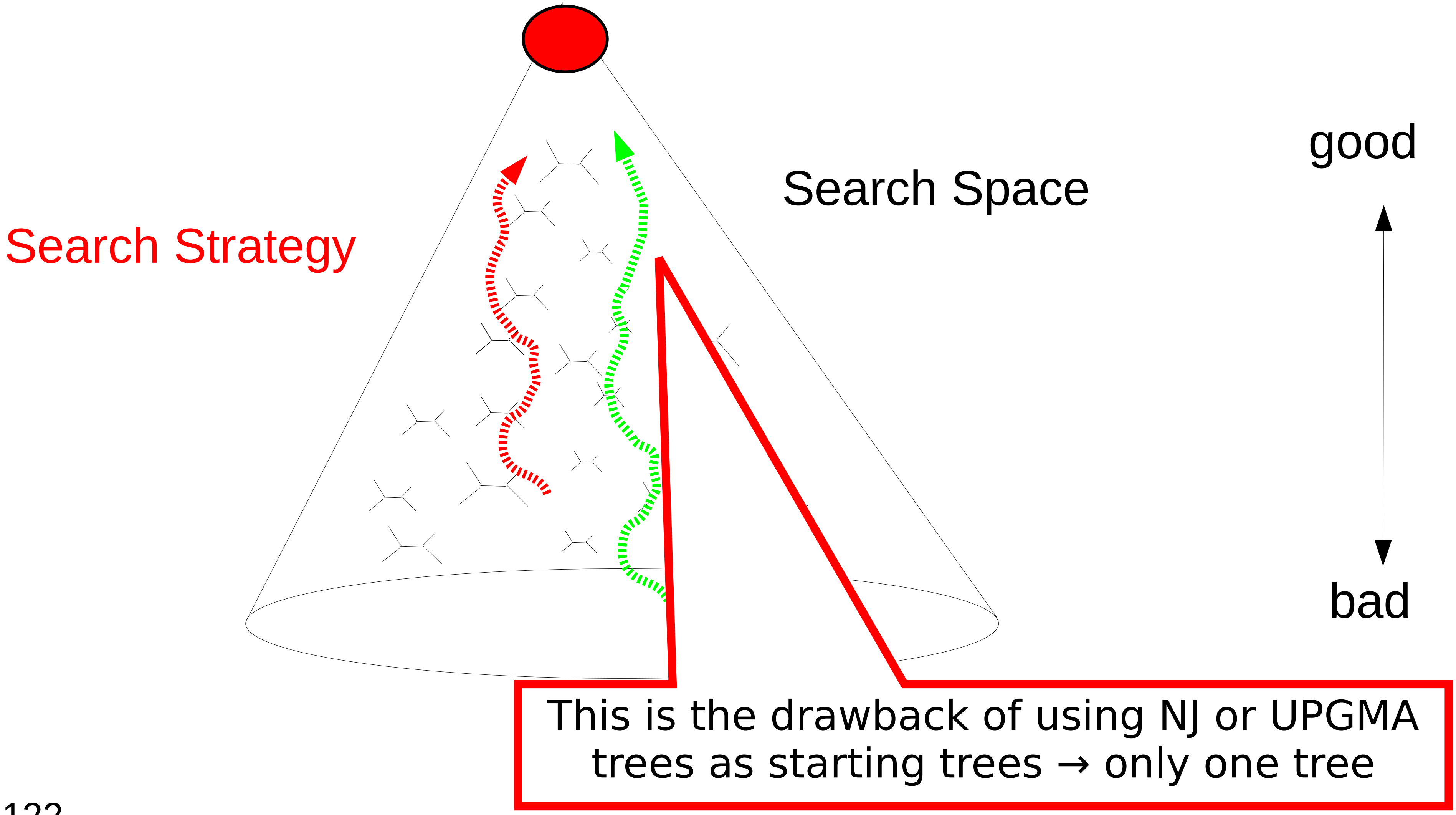


Distinct addition order, e.g.,
Seq0→Seq1→Seq2→Seq3→Seq4
can yield a different tree!

Why are distinct Starting Trees useful?

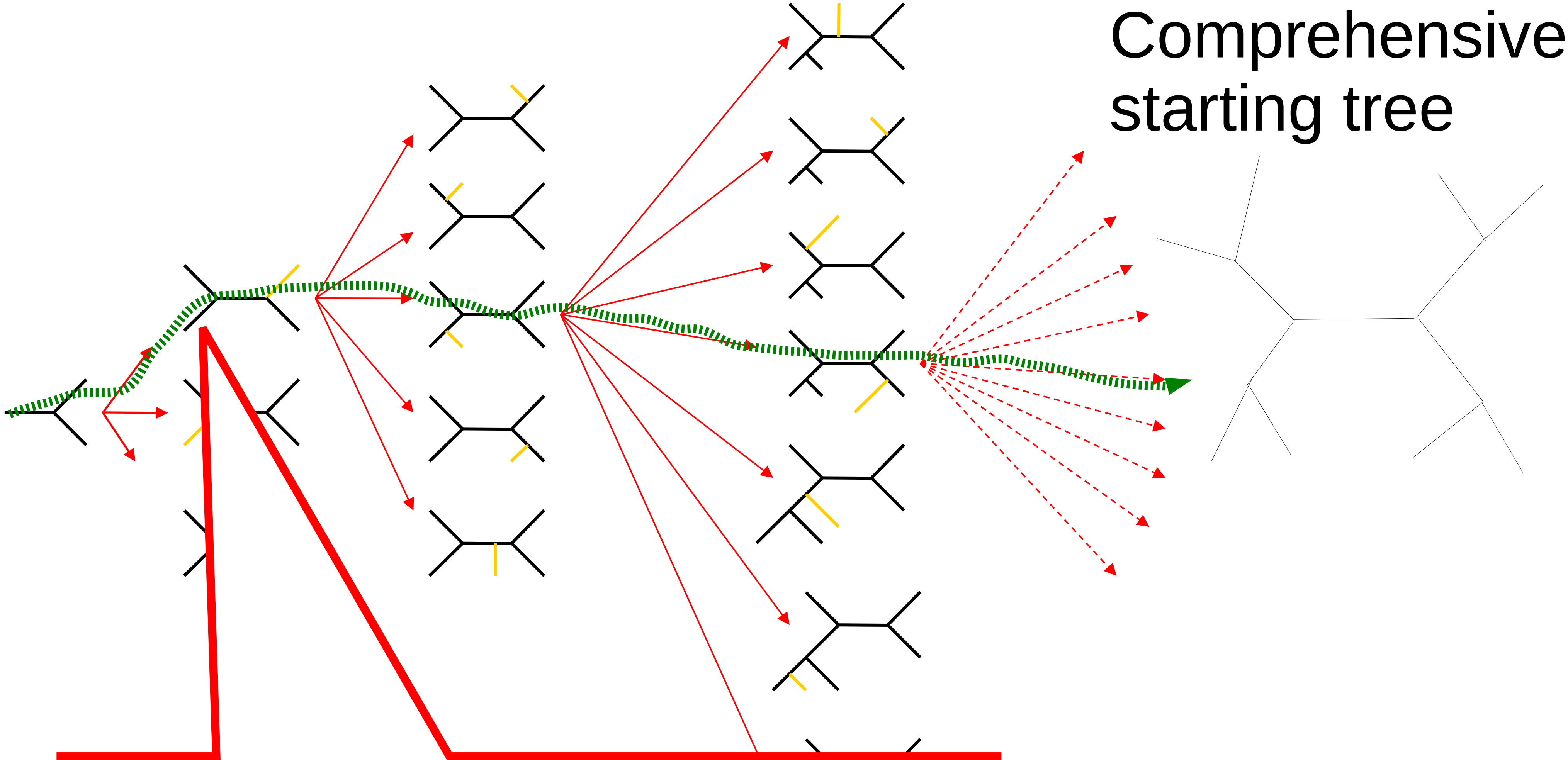


Why are distinct Starting Trees useful?



This is the drawback of using NJ or UPGMA trees as starting trees → only one tree

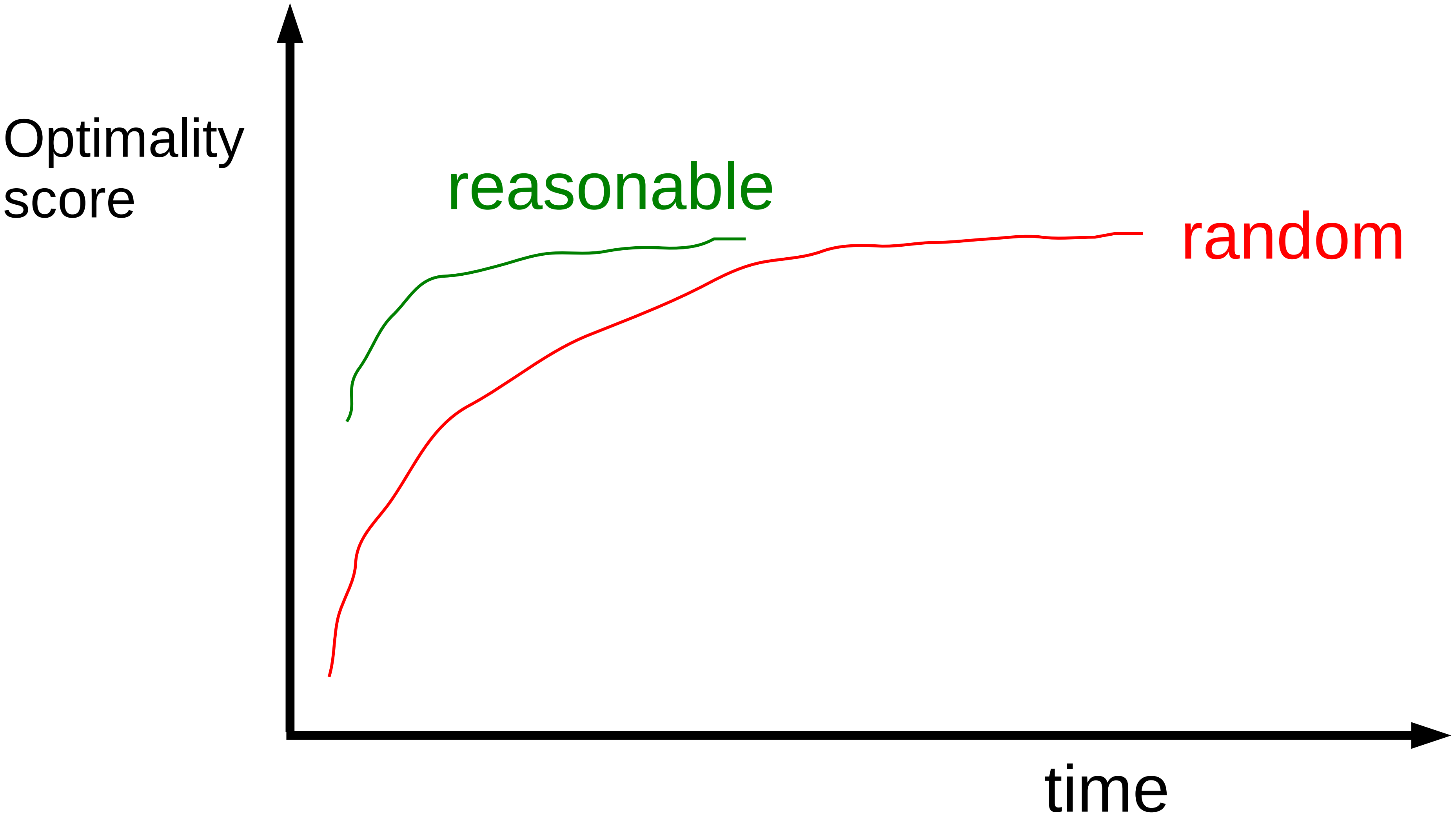
The number of trees



Comprehensive starting tree

Stepwise addition is like following a single path through this!

Random versus Reasonable Starting trees

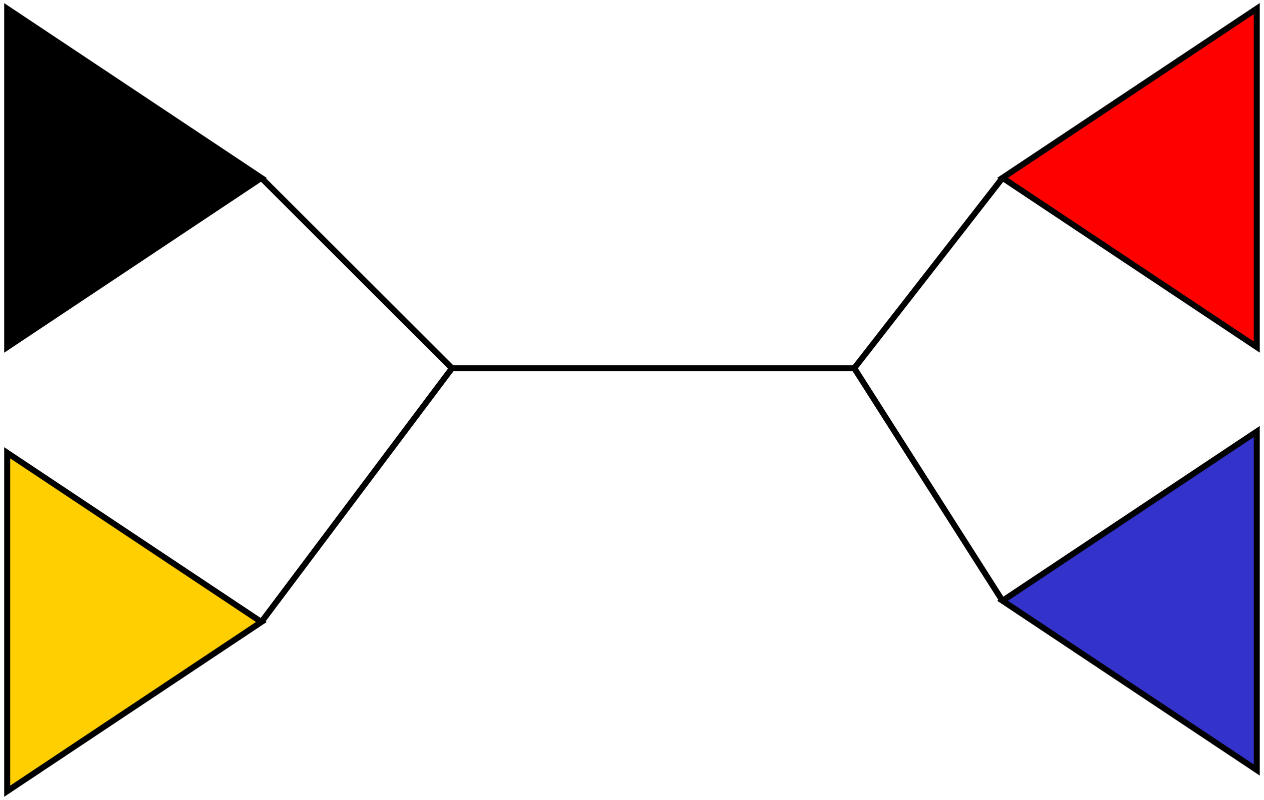


Search Strategies

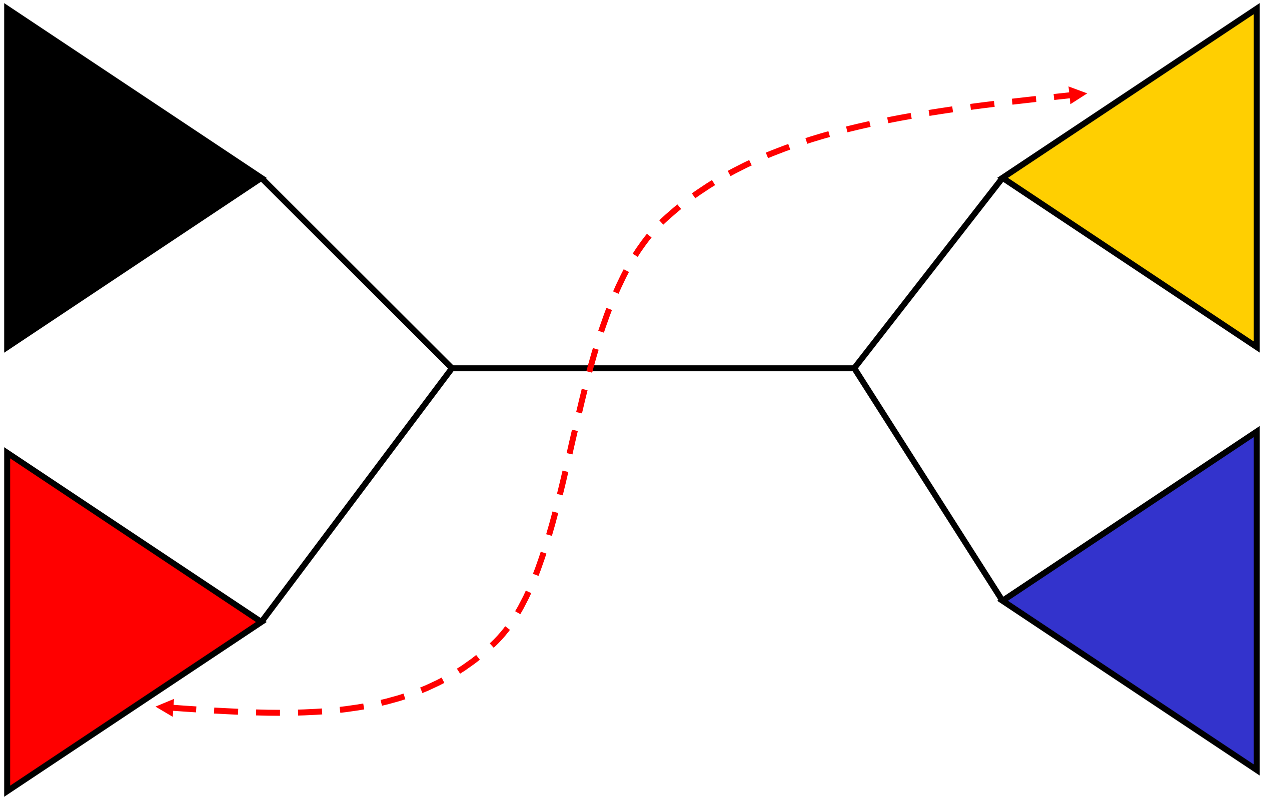
- Given a comprehensive tree
- Apply topological alteration mechanisms in some order to improve the score, for instance, via
  - Hill-climbing
  - Simulated annealing
  - Some other technique

    → design of ad hoc heuristics
- The three basic moves are:
  - **NNI**: Nearest Neighbor Interchange
  - **SPR**: Subtree Pruning and Re-Grafting
  - **TBR**: Tree Bisection and Reconnection

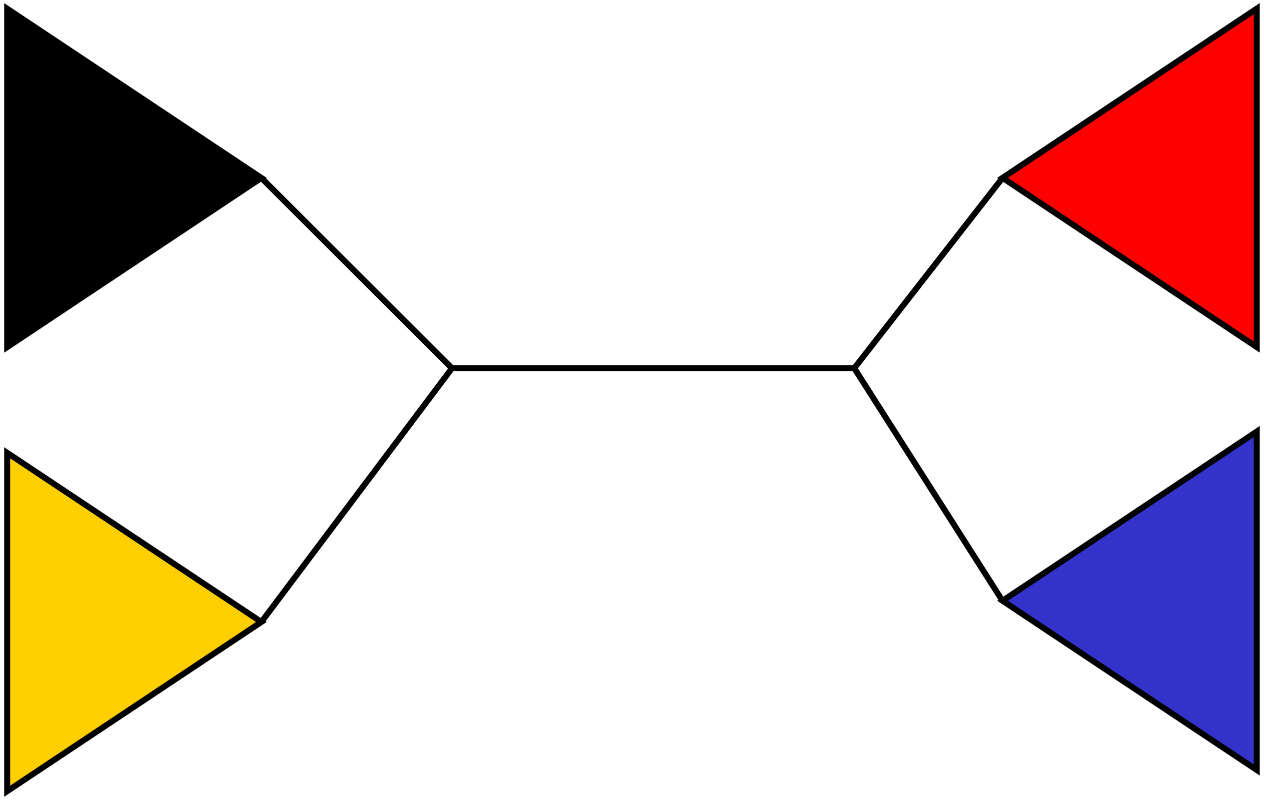
NNI



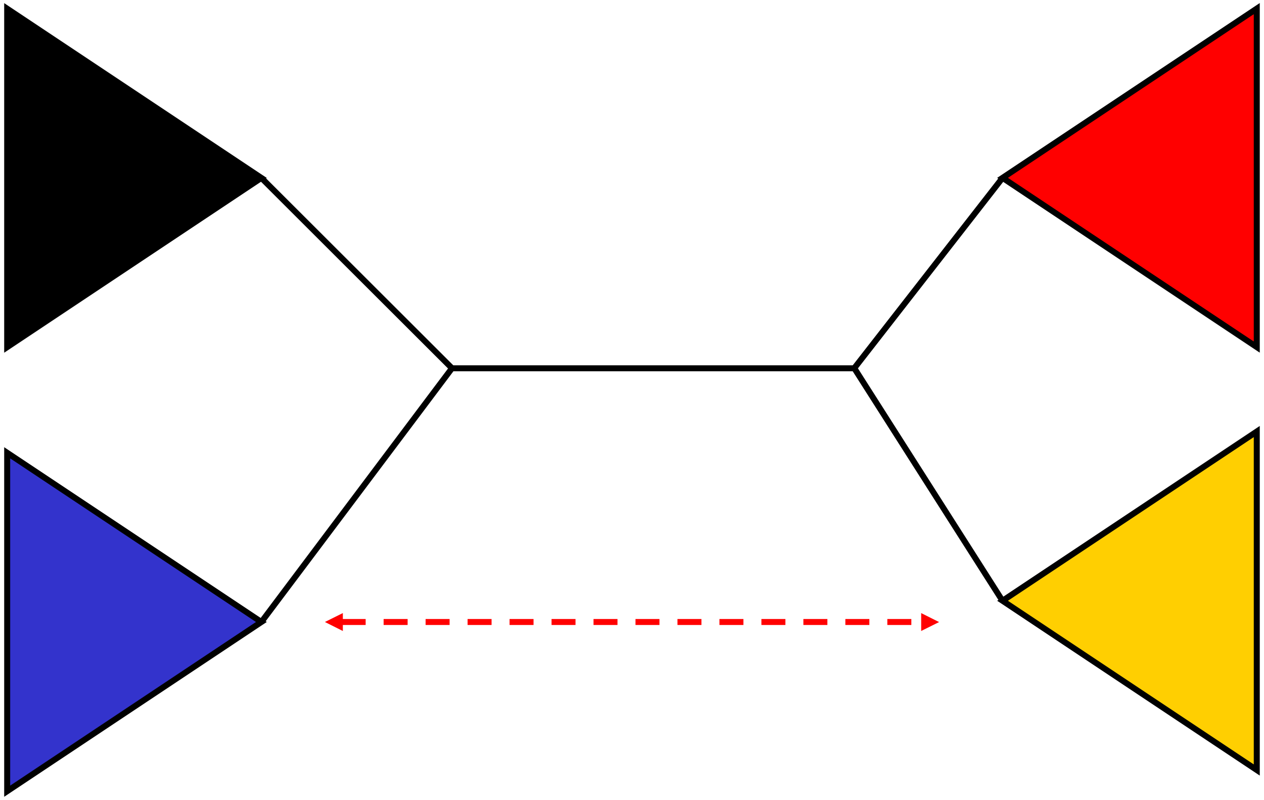
NNI



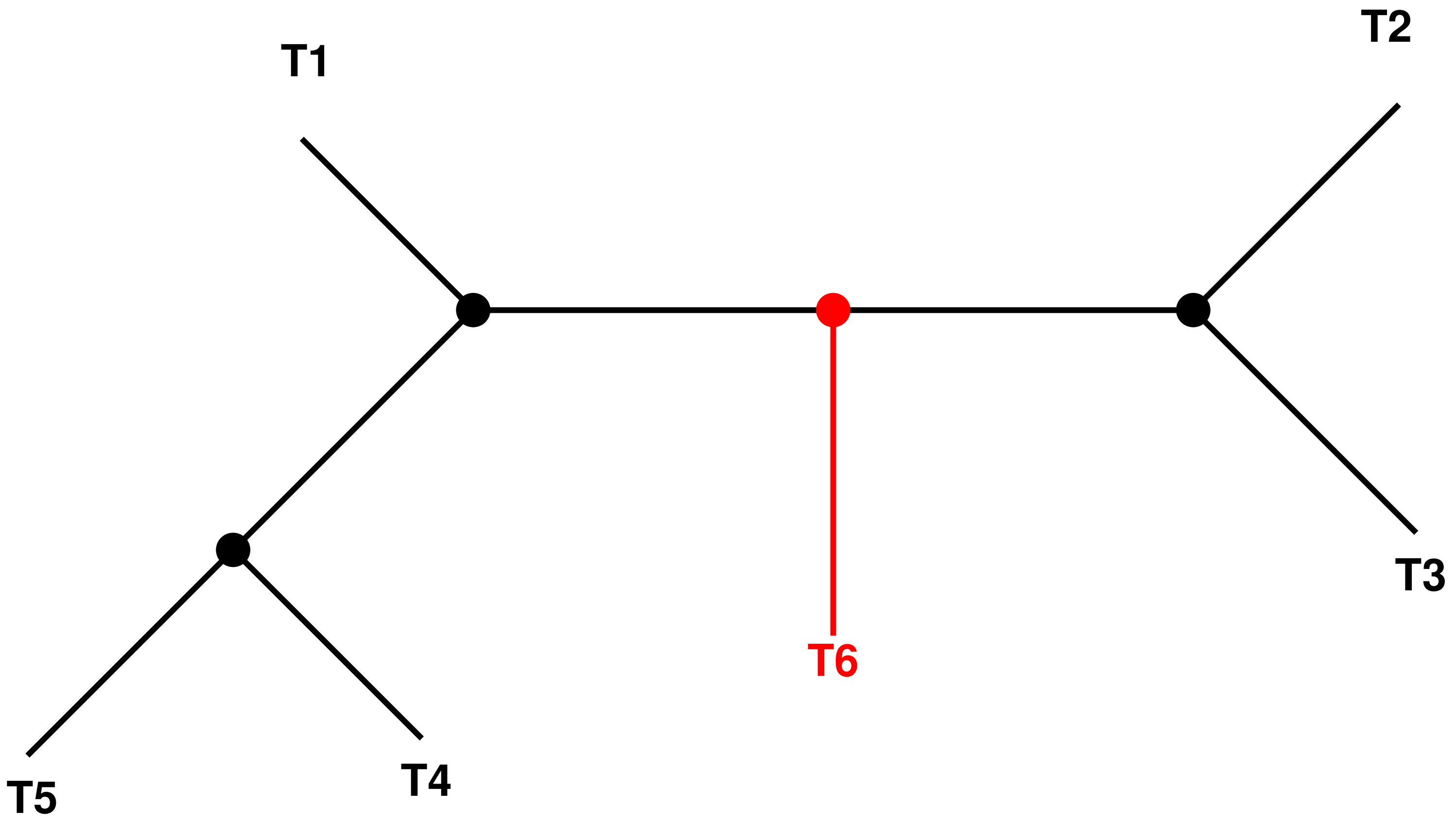
NNI



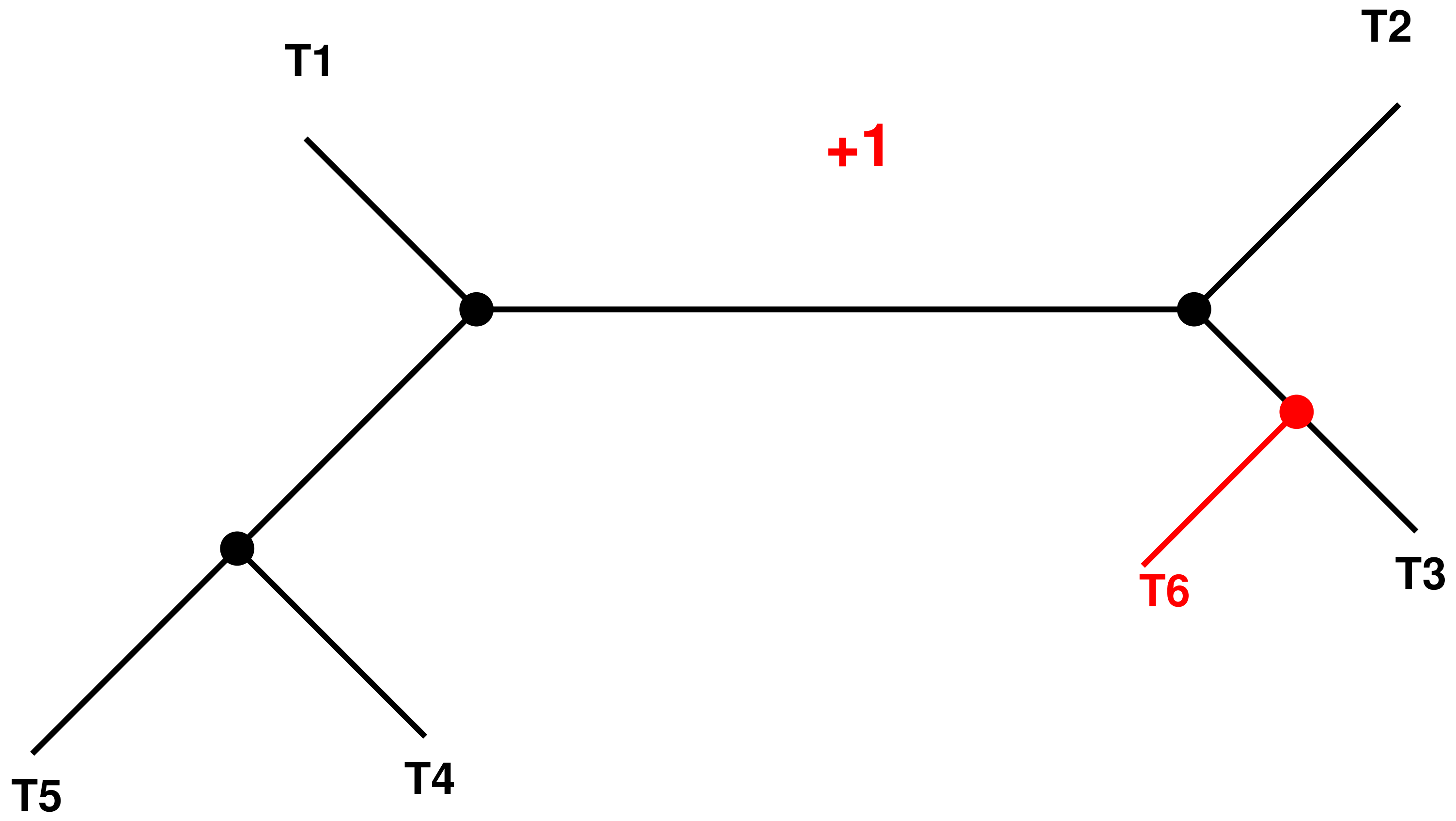
NNI



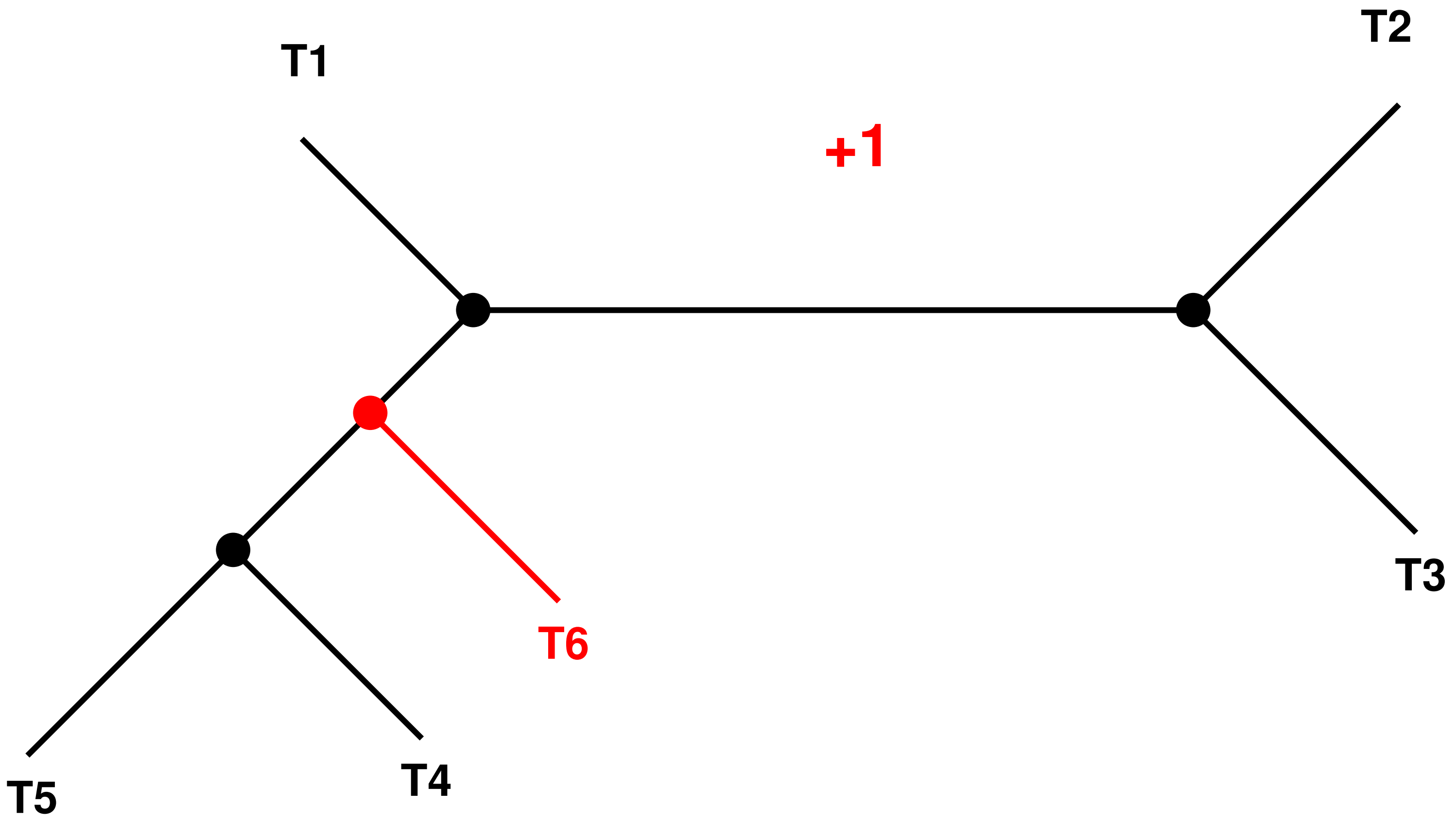
SPR



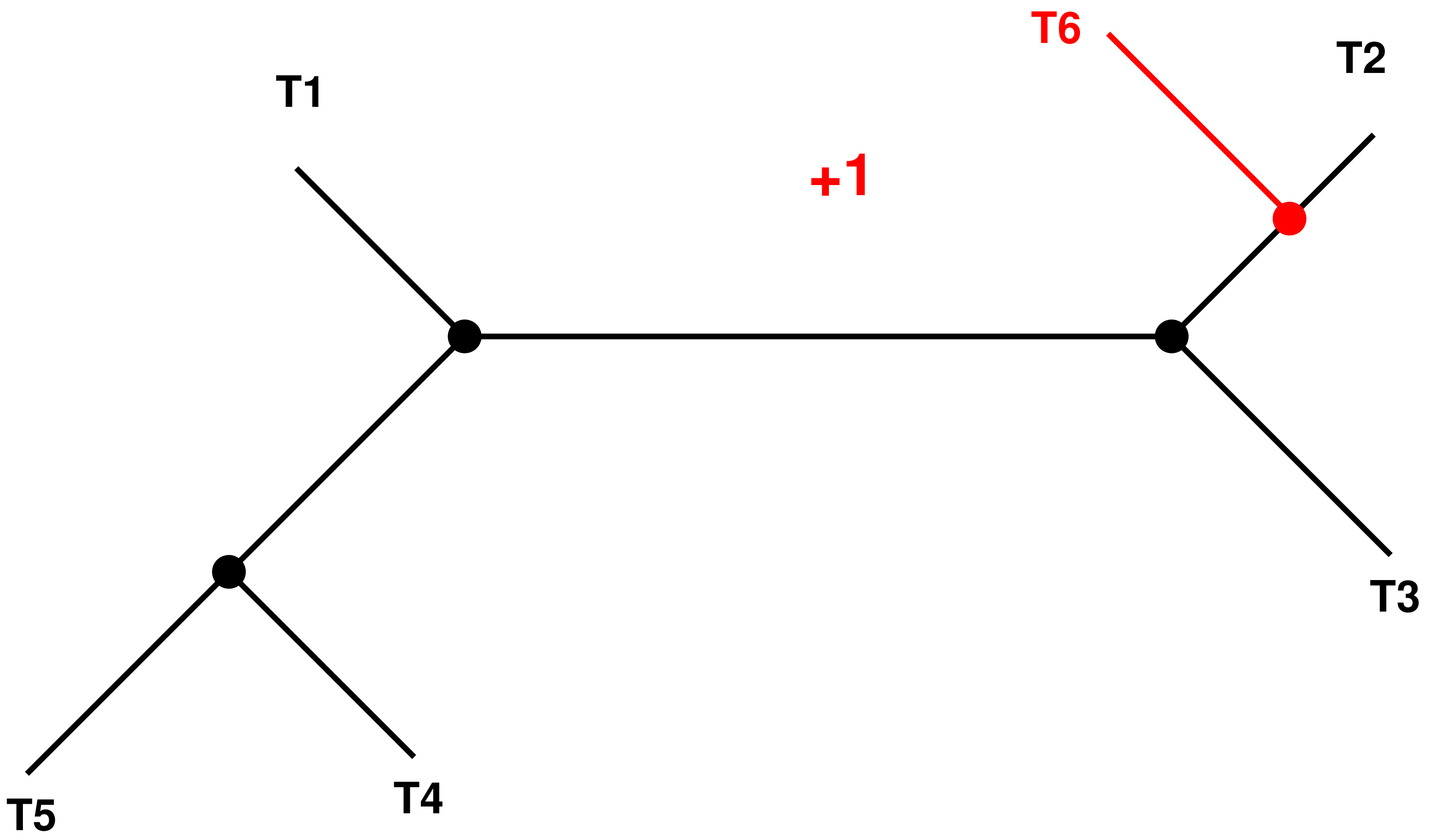
SPR



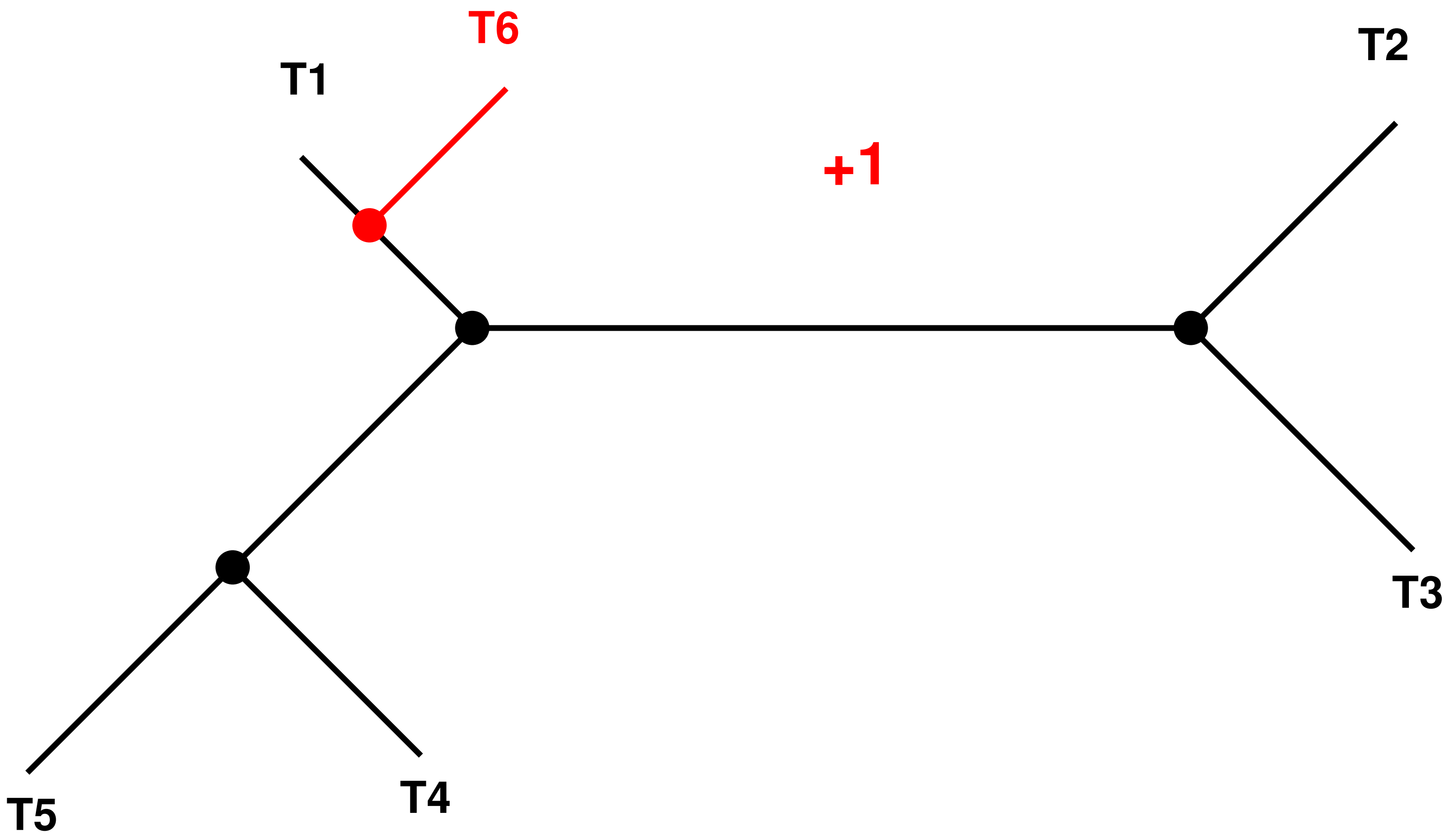
SPR



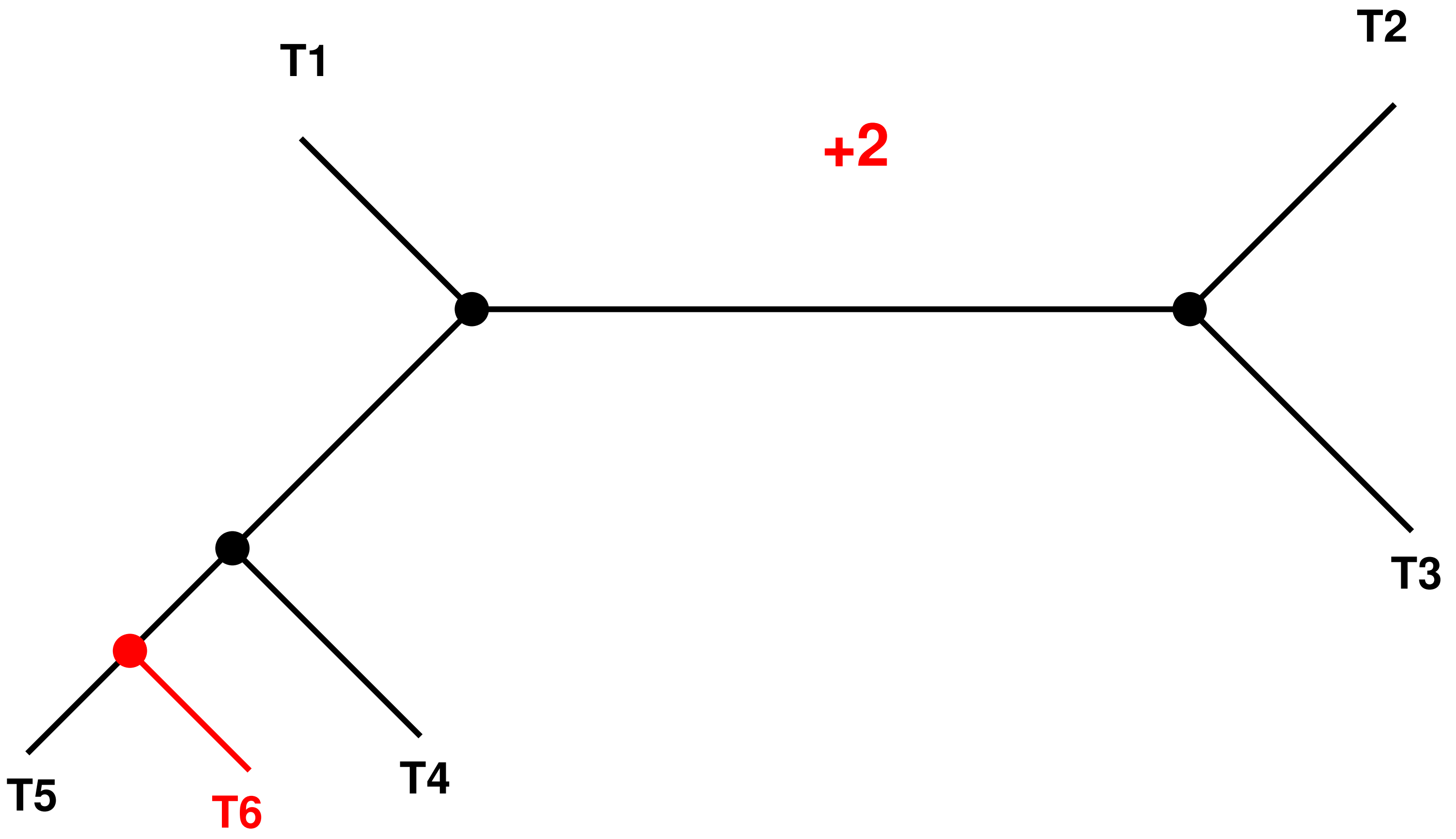
SPR



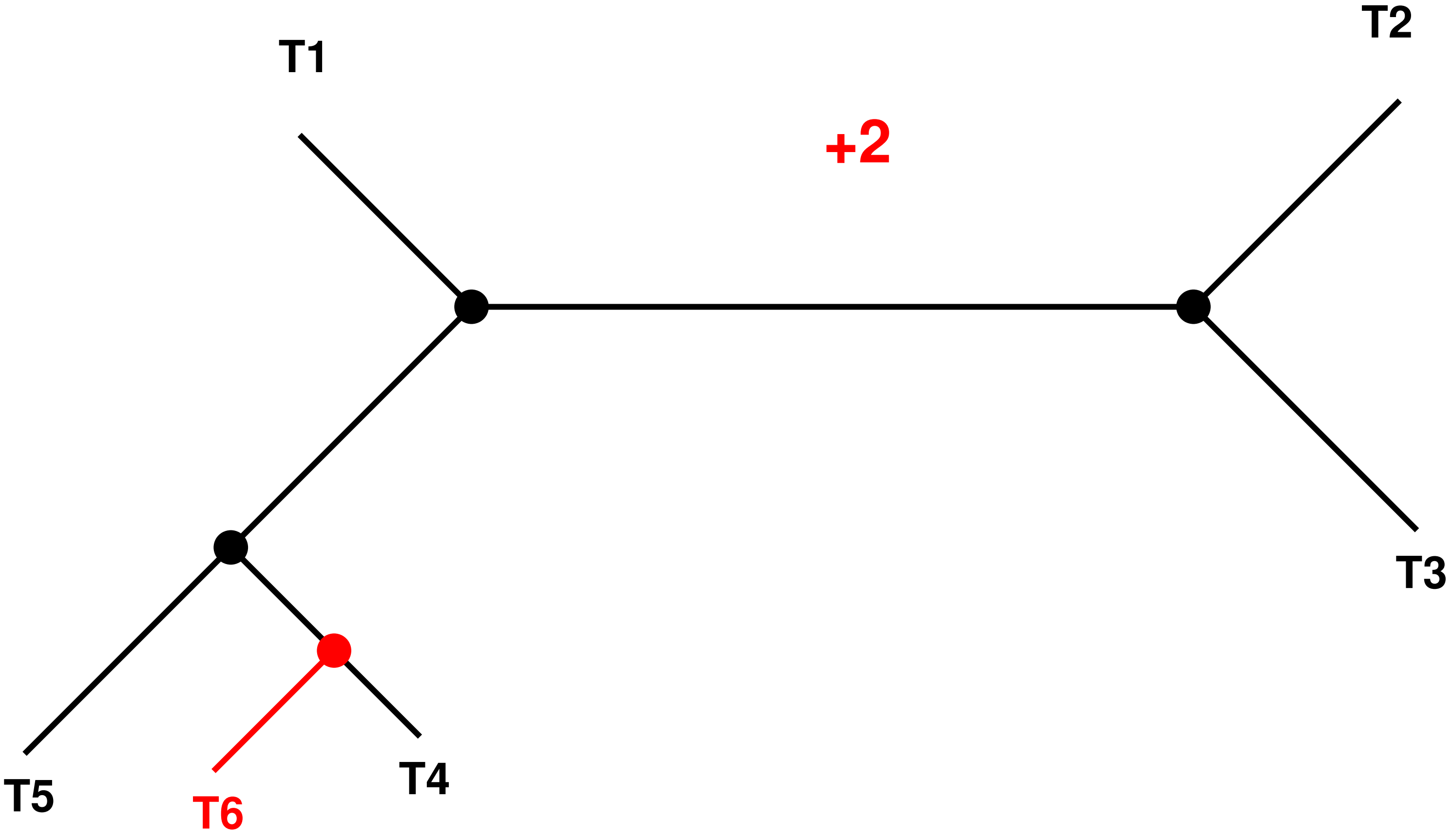
SPR



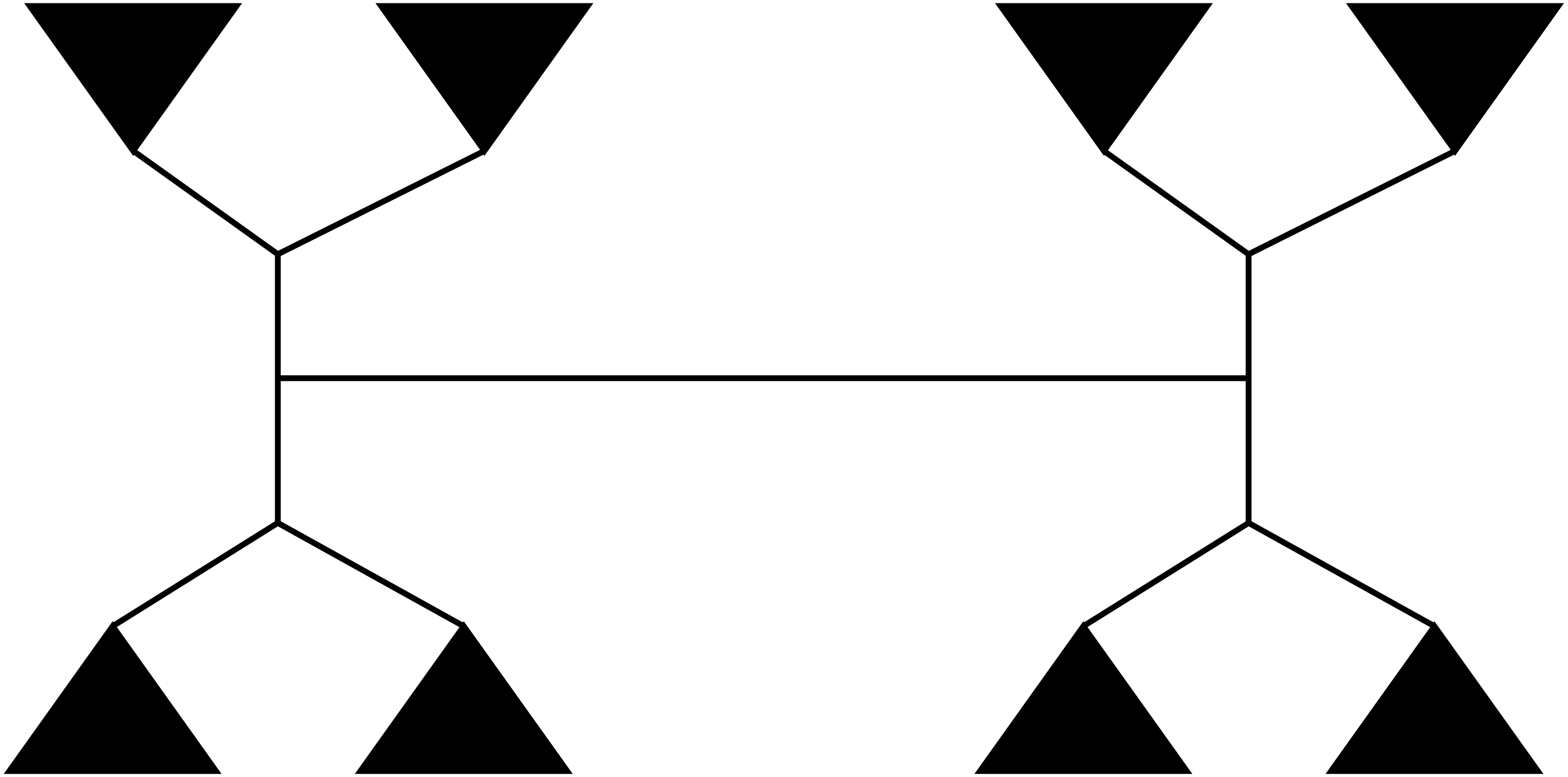
SPR



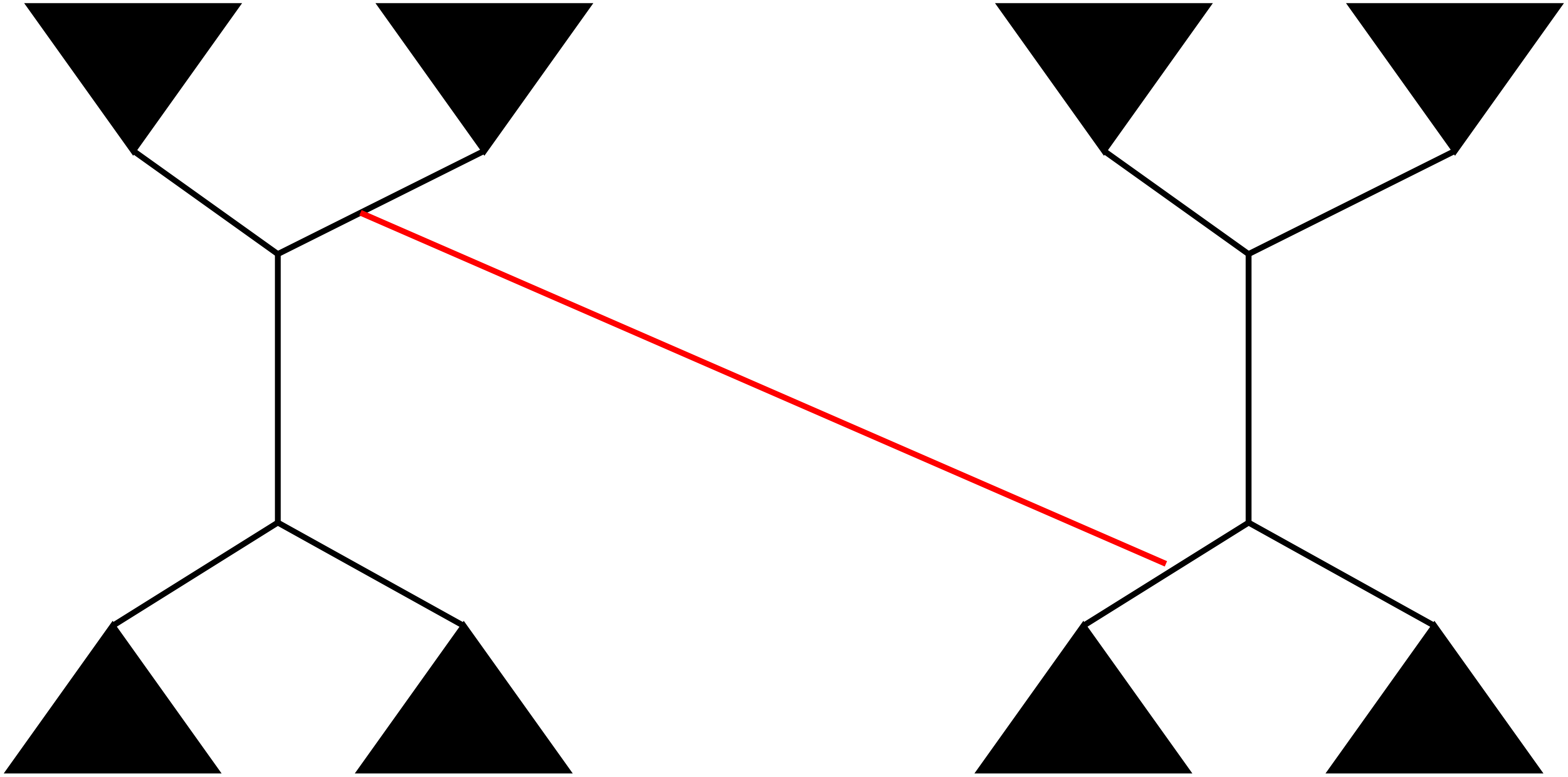
SPR



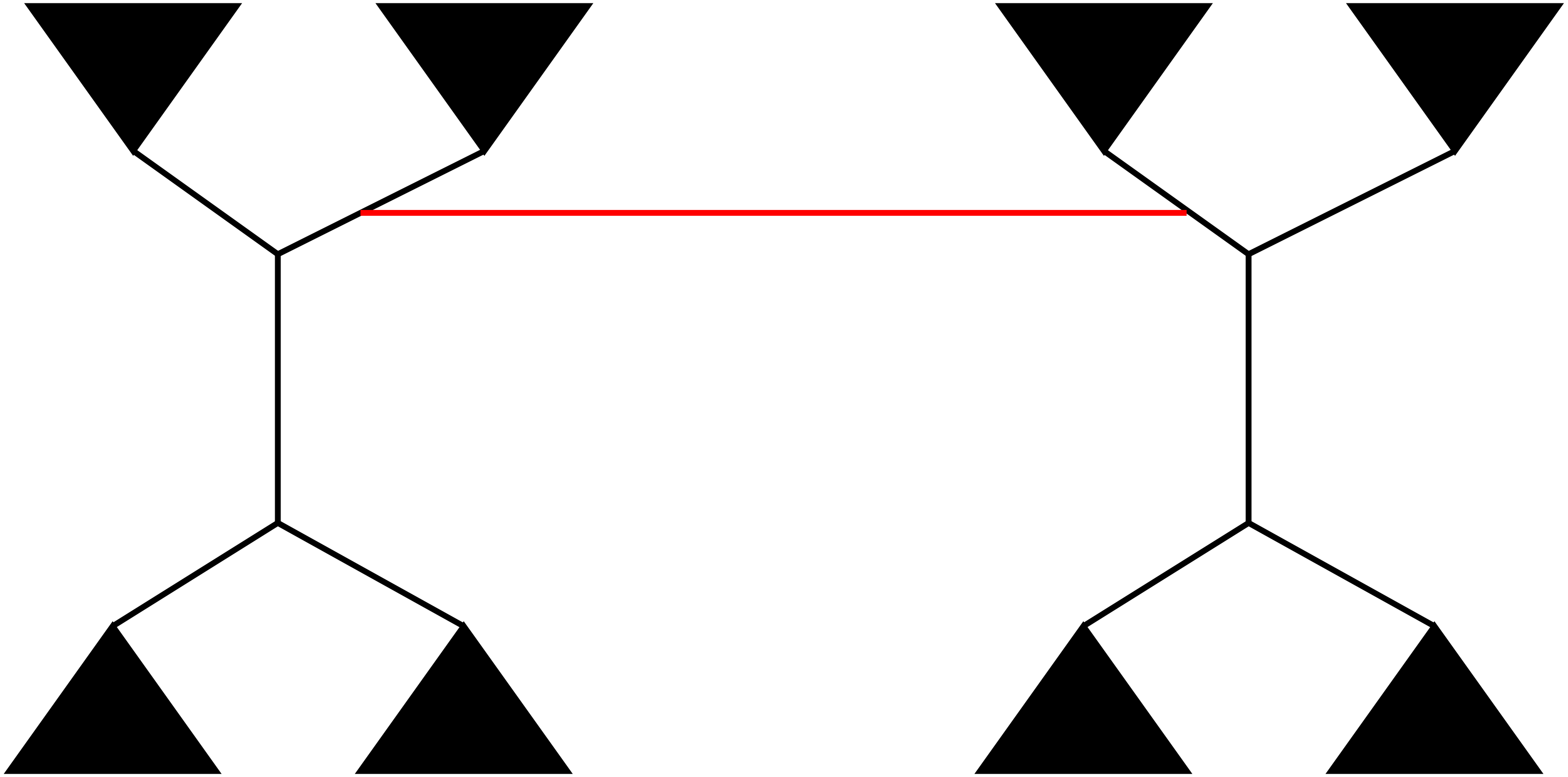
TBR



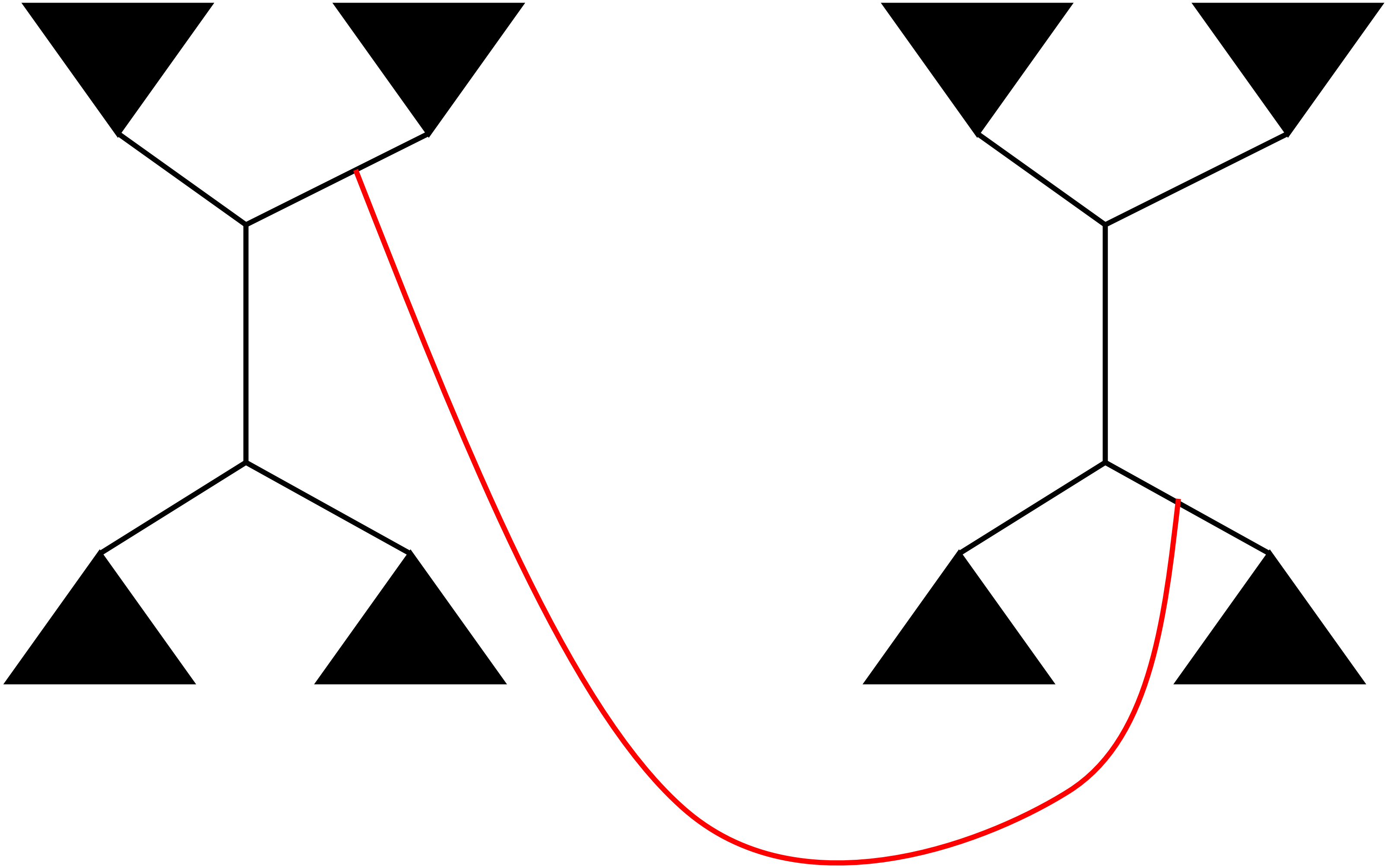
TBR



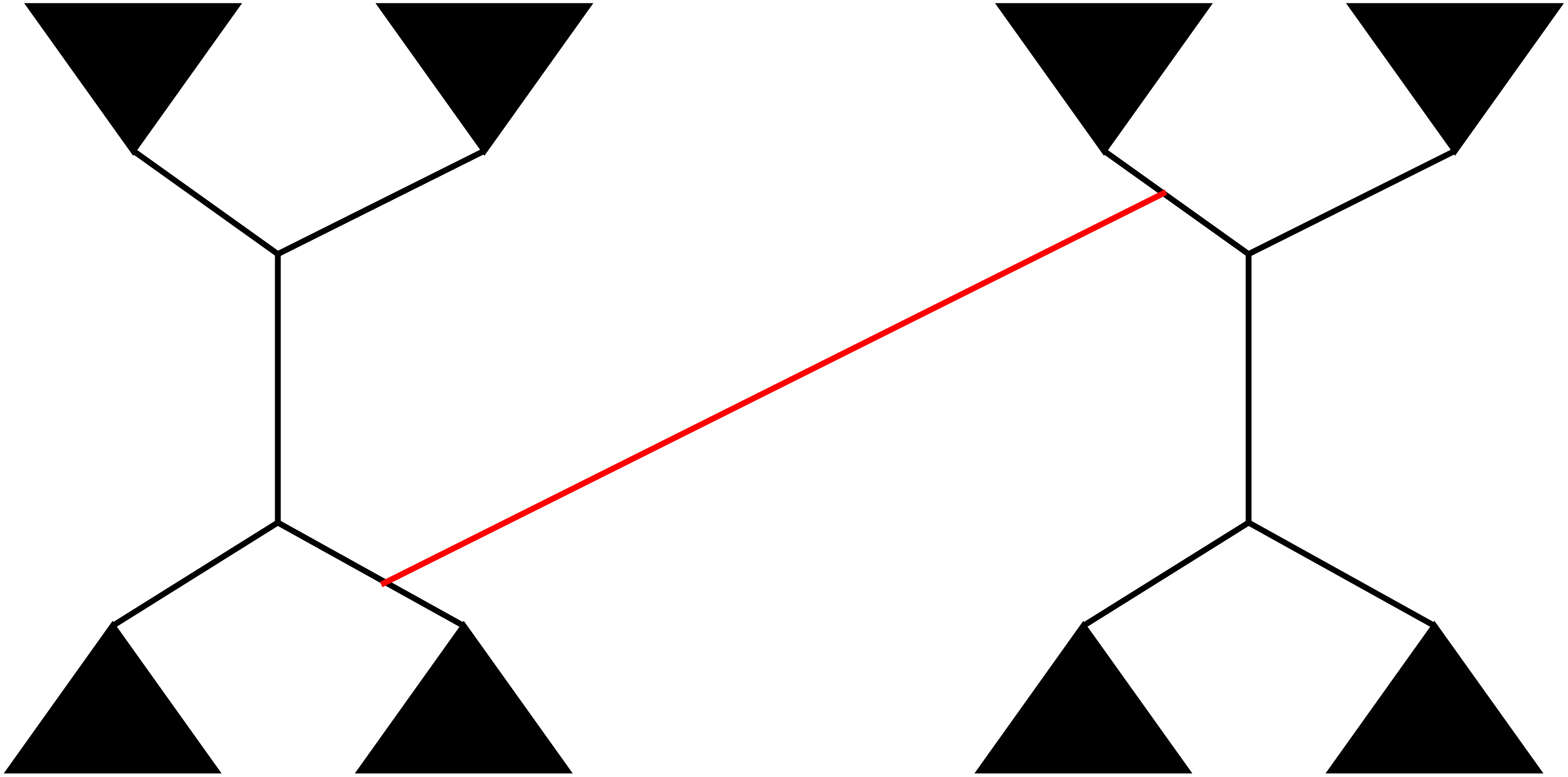
TBR



TBR



TBR



Question

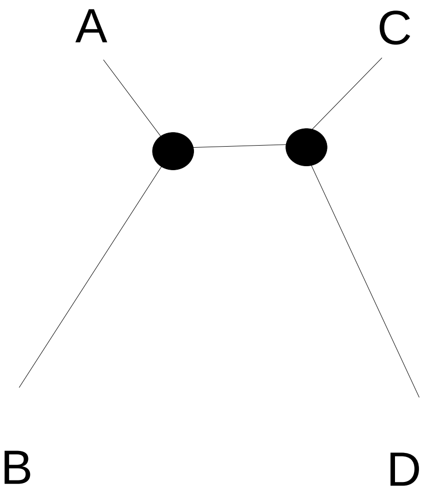
- How could one design a search algorithm for the least squares criterion given a function *f()* and a distance matrix *D* to compute the least squares score on a given tree?

The Parsimonator Algorithm

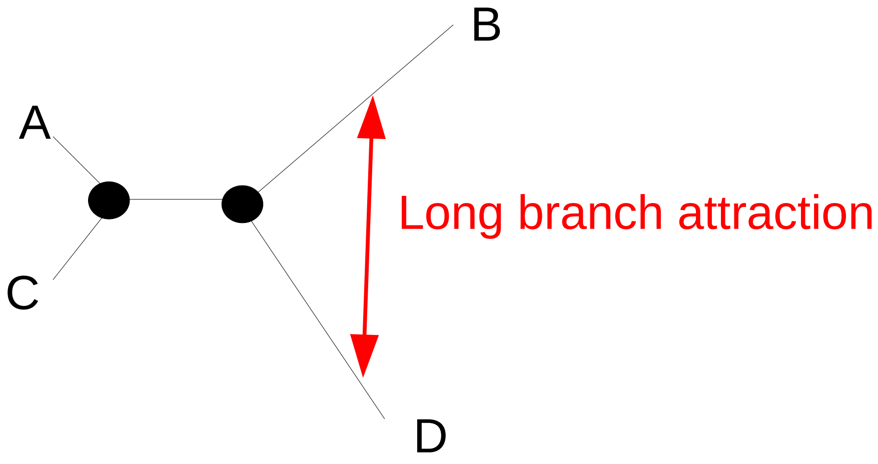
- Build a randomized stepwise addition order parsimony tree
- Apply SPR moves to all subtrees of the current (comprehensive) tree with a rearrangement radius of 20
- If the rearrangement of a subtree yields an improved parsimony score, keep it immediately
  - → this is somewhat greedy as opposed to a steepest ascent hill climbing algorithm
- Continue applying SPR moves with a radius of 20 to all subtrees until no tree with a better parsimony score can be found
- There are much more sophisticated algorithms available
  - → TNT tool by Pablo Goloboff
- Keep in mind that parsimony returns discrete scores, that is, there may be many equally parsimonious trees among which we can not distinguish!

Parsimony & Long Branch Attraction

- Because parsimony tries to minimize the number of mutations it faces some problems on trees with long branches



Correct tree



Wrong tree inferred by parsimony

Parsimony & Long Branch Attraction

- Settings under which parsimony recovers the wrong tree are also called “the Felsenstein Zone” after *Joe Felsenstein* who has made numerous very important contributions to the field, e.g.
  - The Maximum Likelihood model
  - The Bootstrapping procedure
- If you are interested in statistics, there are some on-line courses by Joe at http://evolution.gs.washington.edu/courses.html

On-line Resources

- https://cme.h-its.org/exelixis//web/teaching/slides.html
- At the above page you will find all slides of my regular lecture on “Introduction to Bioinformtics for Computer Scientists” including many more advanced topics in the area of phylogenetic inference
- You will also find links to youtube recordings of these lectures there !