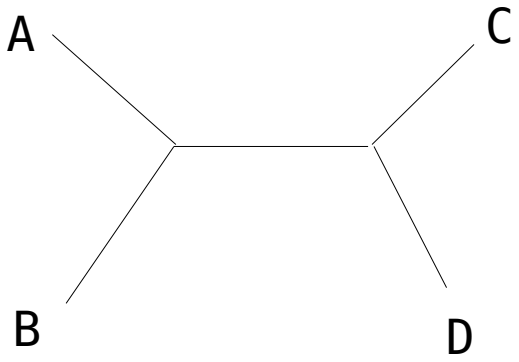
Introduction to Bioinformatics for Computer Scientists

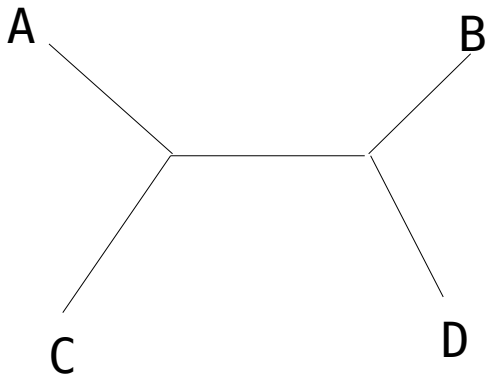
Lecture 6

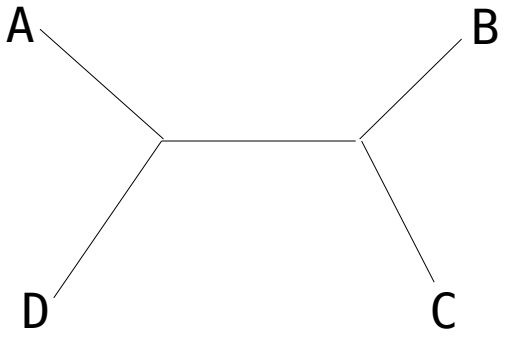
Plan for Today

- Last time:
  - Multiple Sequence Alignment
  - Introduction to phylogenetics
- Today:
  - Introduction to phylogenetics (continued)
  - Phylogenetic search algorithms

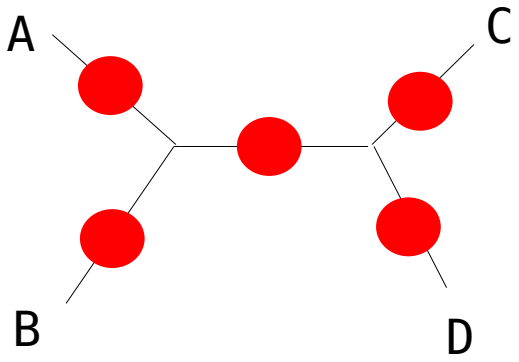
How many unrooted 4-taxon trees exist?

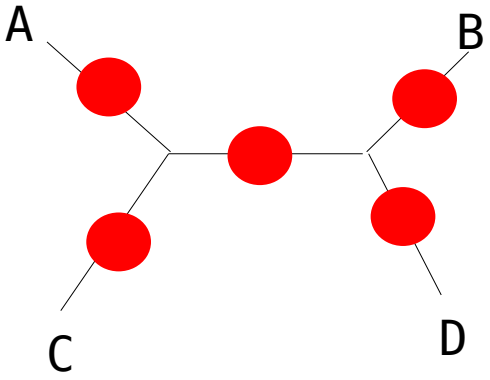


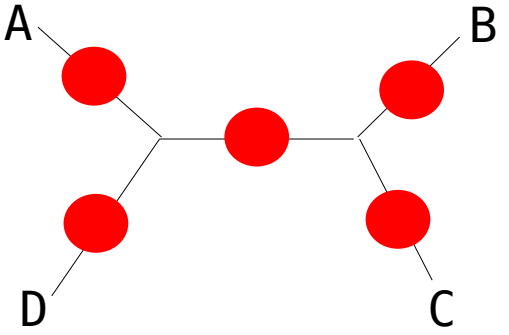




How many rooted 4-taxon trees exist?



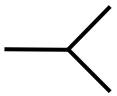




Tree Counts

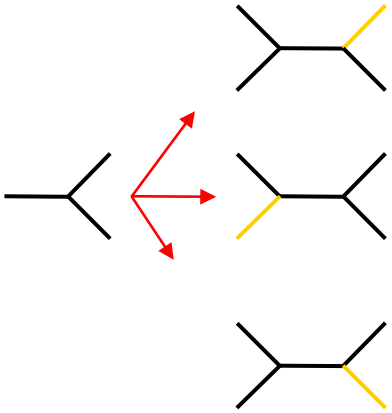
- Unrooted binary trees
  - *4* taxa → *3* distinct trees
  - A tree with *n* taxa has *n-2* inner nodes
  - And *2n-3* branches
- Rooted binary trees
  - *4* taxa → *3* unrooted trees * *5* branches each (rooting points) = *15* trees
  - *n-1* inner nodes
  - *2n-2* branches

The number of trees



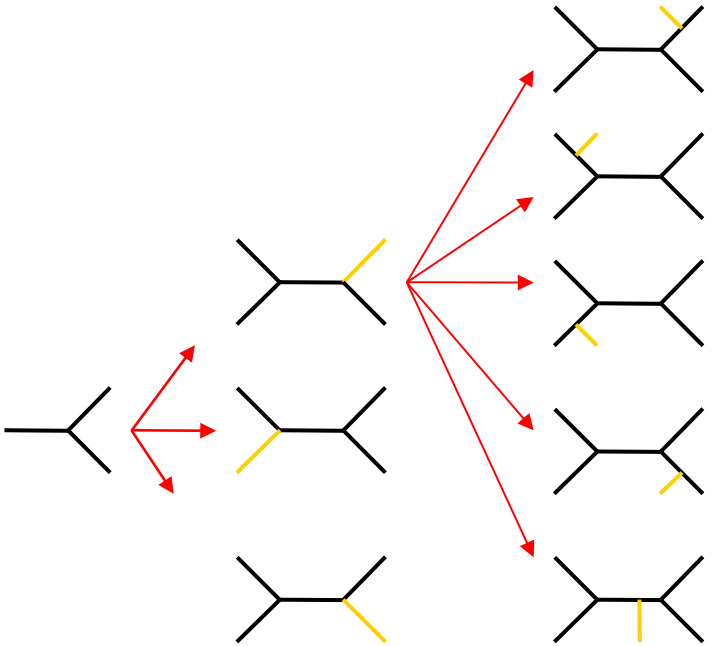
3 taxa = 1 tree

The number of trees



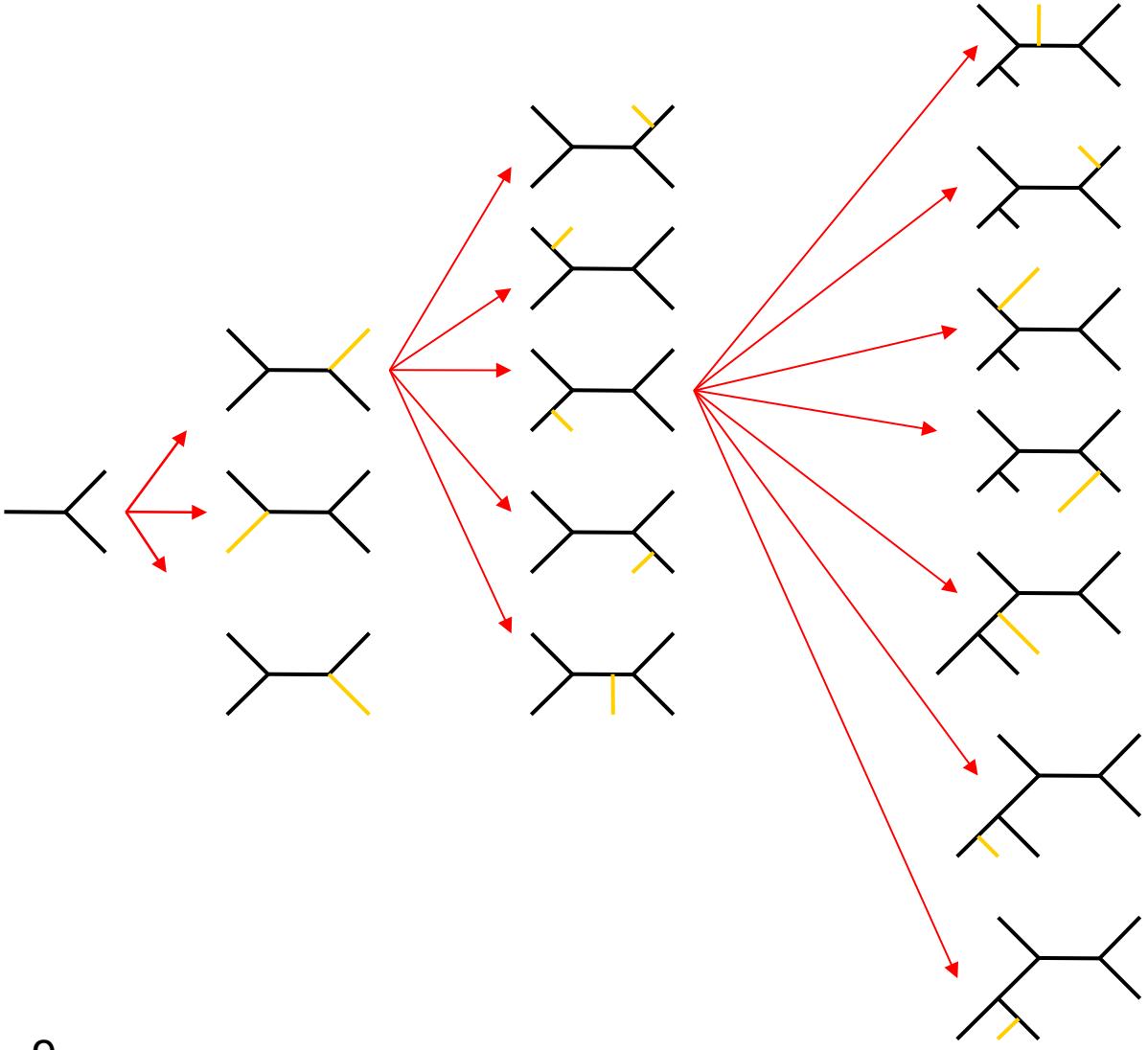
4 taxa: 3 trees
u: # trees of size 4-1 := 1
v: # branches in a tree of size 4-1 := 3
Number of unrooted binary trees with 4 taxa: u * v = 3

The number of trees



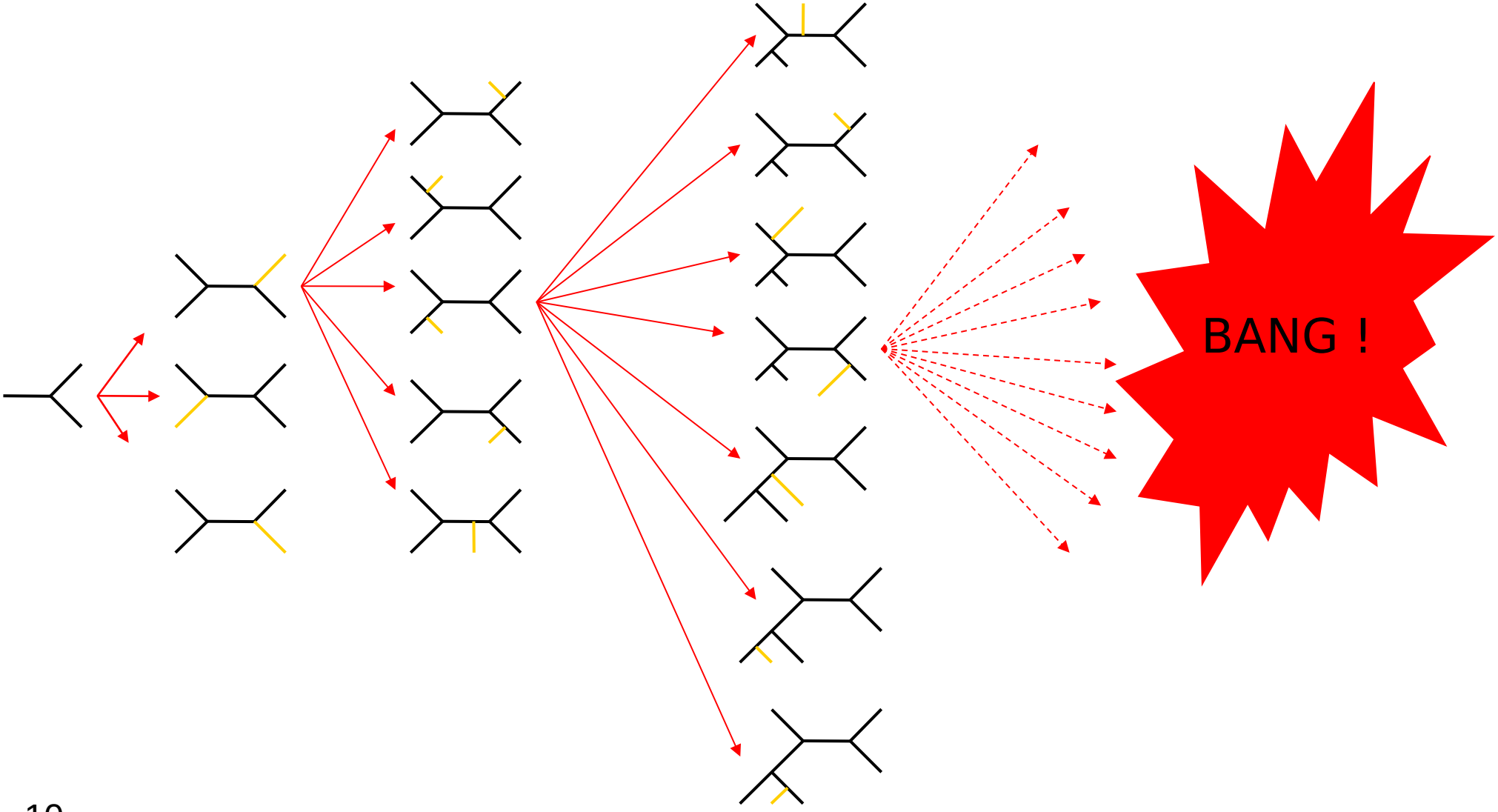
5 taxa: 15 trees
u = 3
v = 5
Number of unrooted trees with 5 taxa: 3 * 5 = 15

The number of trees



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

The number of trees explodes!



Some Numbers

Number of Organisms	Number of alternative Trees
3	1
4	3
5	15
6	105
7	945
10	2.027.025
15	7.905.853.580.625
20	$2.21 * 10^{20}$
50	$2.84 * 10^{70}$

Table 2.1: Number of possible trees for phylogenies with 3–50 organisms

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: 3004963817421165615163291006568181498137723207423701308950495404301263652525830821082768599668824700046435273521426563428829589150234460006314939691306329704360561848618774654822779912235368092334555631999108345976931267565250128998674331877528114019609916315223670306091217357097623798477054676677795324797182614385273338226727784250737252849916669687584403510579587020686505817687044663181237429010214385064324713609344916670211359697569403006662526464792691245510314942366195542824118277625114848758254581227914289801132648902674033761294712745767036267579086843169660718609847941818865957214557044744572288661729053583520744253688123124010661315694886196094119564673620034257524133527757508582916109642257572769976799140828334321016132740165283099380390459232769069003597291970994073934956348620389901074268728229759746553771022576726768428580118772249501062181173405232082653973429622273525365905158656313832720311198419874675997386463182903203832523085979979922161012272157808052481458312068440167606239306009711616729715504728487799634337531348994230372437347879131989085953764070134849446113877572576952408702461720107874297380462275052545706689372319418206440706891884003870590289772197516454495975821662130620506461776109948566373416818358498932907699338206780105243728461492403422961155182609778228619192672071295189589360099591309742330723163825184281103305710174411568843051318658775443763085003114511107238370397074651822320404061547082730786299575493310312752086167006607912980142622300565123522718063819509335872651728623589020520016144361756075654286471422126613004434807084067501589247673166341539540575074474994909831496473031080411401891849735912811228378774049884834056210242056642446386009389965085742961947269054301528123752651096581528469979703679217112903556809818079169587951614159281049528179855847292534447864424435998085315372047968149694659917686145337010519859285771574824559433772423695825762426630169463204824951822559392874031776234338810486046309751915569238711675130952134150988167154643078623526062377864068386804246902527491139319276802611515990582603886733172930713673903403618637463980605764836474670274446727880885337074254421922726677747003329403201038288035112689026255183096791948358678929370163768175304820633894387149793115235369822962511116307148294599211620803302684762013335690441089668145436150905155877581167977001256391215111623744417049737170460402948110411482228646613191882199757138336835207252605520276982397461321849524926489705079039836025625560628985228883956135787417565766488999260873286612630642543260248979229113560071640573984516375245243376943755857384725545564397599604255914640112221144755235573176239973057747183956531217416532295986675901294116123924072209325036967312488449155375921065065601541672077415923624086866767534828651296488873905970757880247339346347084815901163977279774748041731626870091672873561216422684681606831989598012603764856153127816116895872151231233087600634733810972531184233396403909373783950668355787353078863586464005632994949063118742402902779272693300322445377595797224873456891511458557078385054168166766742581130195806362190750079029503108820909727174813643698947397107993277770067630173061756653873972603777717300844134394051236690555449324861650825399577950363267049478442934988531727973481777971465671751511788763964340693324580763461107342143281950499096808740273976889147045174720555436948966874260147724189469321290245331733418828677319465354411330210086657508171324034264758048921862366346160795594372051639515409694980648624230979694721116966596158004178839922322646284989423522769263911243607675089976717896834593378907423463545571935306656153799002779162656363618619740485909382340622354597697331372136593437175855906644396461328300113672601934068706442339489199215304385281541659630118549423634863524585746642834609062916279564926584723003608555598989761916293248140094592489899468468862322586017055146890564983727600392748706955046378881974169942904910285318041077651600726338472163890378227001384359959973026572719880432466431235975855521719696051392101022659632478878309774053331315517629781528807186526031763272648280944993704562580999380530584976995700802893798080149029010052938984722799471678048216894241591182842576964647865057315325178302336307298251692210346584265894447464916123854689718507968172909139032182834111184821384767728316548653212317382004131990510518967022201887049585687180509590730360693040293721603896891760558767695538231809370582625708389838740909846865663427139750001329183510594332172987982524370750827208795985943715766766015578269966034319775262330889962587800628009560944416932377944955441033696586261556256010669390303203878970983673786087056641433585106111658314520424513208508589994932364831689671194951671619567622707090697388958885579562466641536561723549301807394004760529801721771391686788000277851966173070061284517307582503735643102065112443730825229625040453160590741343881872563477913830660590931880252231008534017684026140153961698919207514710803375770884974014183459975397205987868206487911606496985817760115397205849822698907181349432691801821173318806365391089368981171489135745668054280748517017585826663963357018935444983266976283509265792220174637219027311964175148994401007963687601782674710701994547321888783274260889667243715747134206000937042513098936305374597842799804031329894172664922904257309583685344162156405572902820662240038632375263809102332698978388604237596256015679752626950798639868104294832333160267216555178120899264677804935741326387137408423885546533836158643451305439624281397279559725995110706314305992615495622958320232708057681156690489586610522030057372529847211874782713671366605866927109487556397485848947591081972703387828443986448674345620095816193031472734596190049931842433797524366248936332124485059719925236685292493053462527641378534132089431289015237380925560459870909127666623296787033288820591349495800740744731433888007245323217473096597419671144414531271327902051010047671014350638857953478447255389801541923317027519896180635152682543173193832925891931530164130548972311128666465492971930479296432829556719092881692091042334122007454242049900872585046208051104875883059495990311188736668509414882172573457635523396403848131821316740835900691640005326225818478376506780445117771732865818989921535830944776535034179687 5

Approximately 3.00 times 10^6328
```

A side-note
The `treeCounter` tool

- Evidently, the tree count can not be computed using normal integers

  → we need an arbitrary precision library

  → I used the GNU GMP (Multiple Precision Arithmetic) library

  → `treeCounter` available as open-source code at

  https://github.com/stamatak

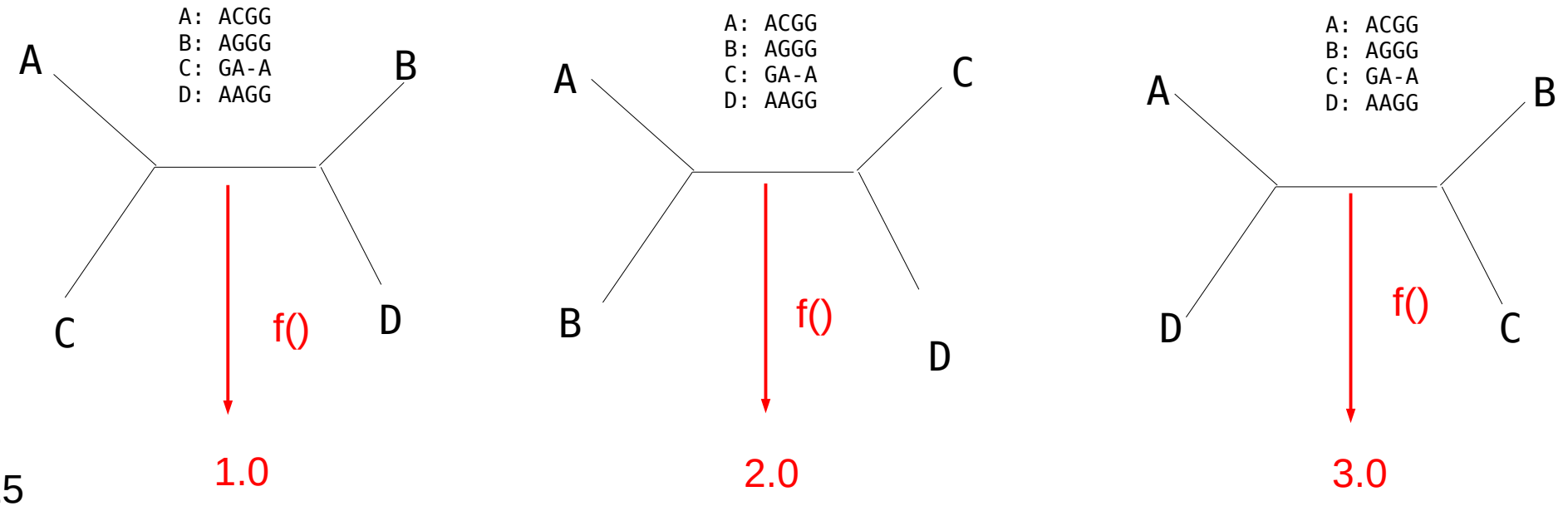
  → Has anybody already used GNU GMP?

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

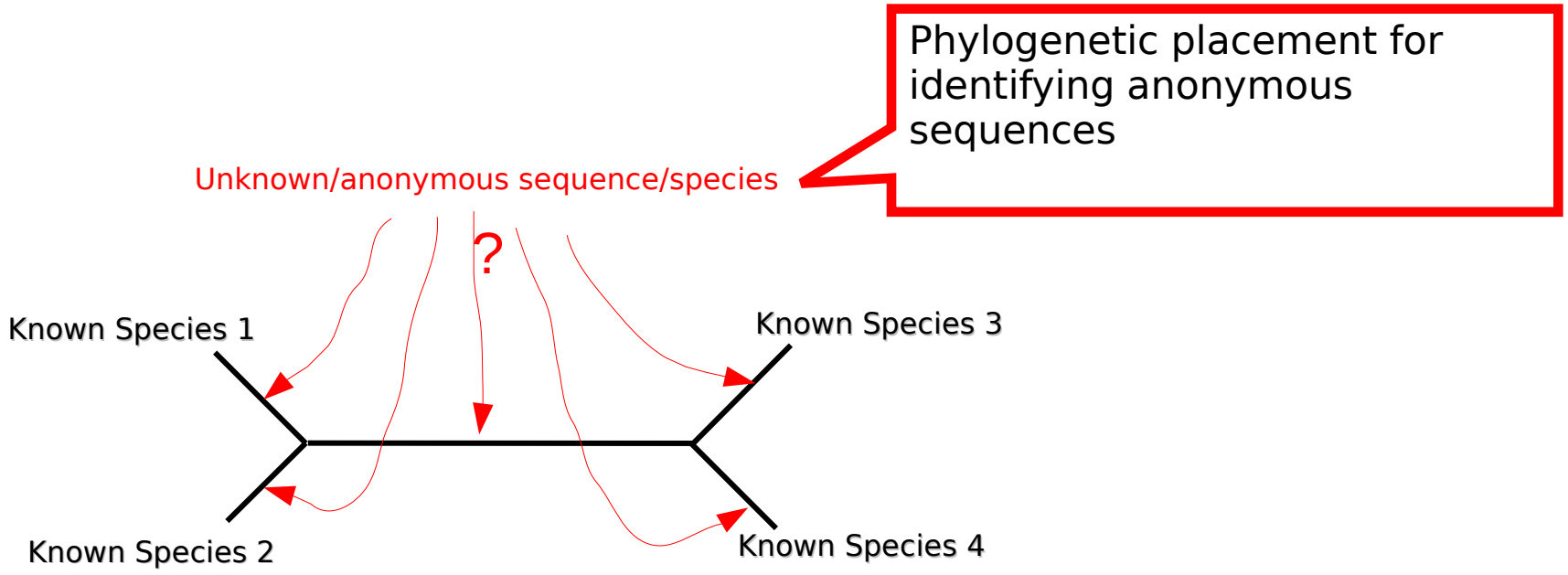
  → finding the optimal tree under most of these criteria is NP-Hard



Before we continue with Criteria & Algorithms:

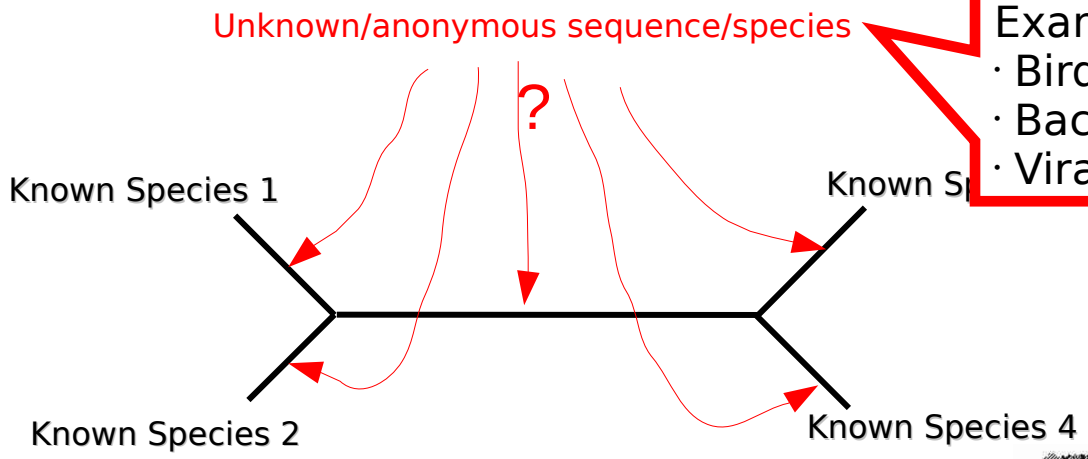
- What are phylogenetic trees good for?

What can we do with Phylogenies?



reference phylogeny

What can we do with Phylogenies?



Phylogenetic placement for identifying anonymous sequences

Examples:

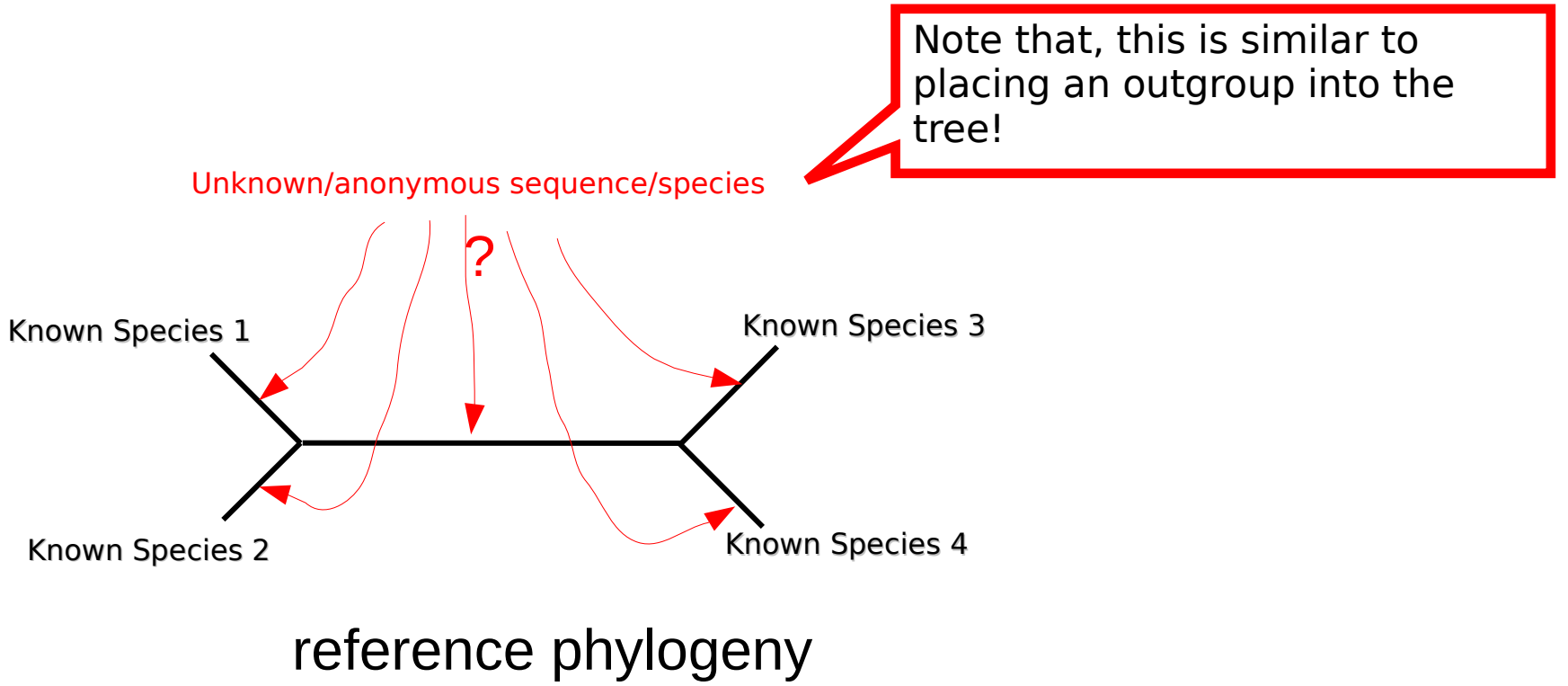
· Bird strike

· Bacteria

· Viral strains

reference phylogeny

What can we do with Phylogenies?



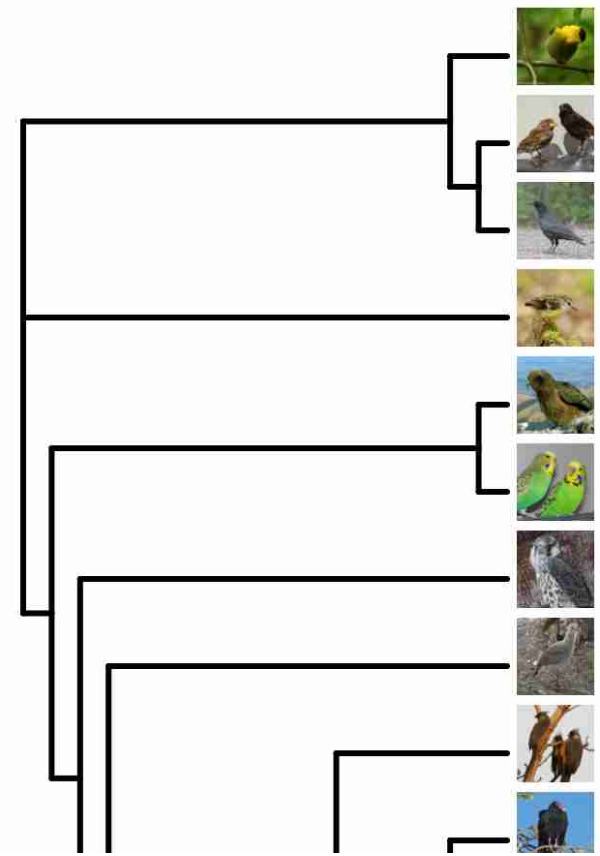
reference phylogeny

The Birdstrike Web-Game

- https://cme.h-its.org/exelixis/eseb/public/en/core/title.html

Aerial Collisions

A research team found out, how birds around the world are related to each other.



Do you see, which birds are closely realted to each other? That is fascinating, right? Click on the images to find out more about the bird on the image.

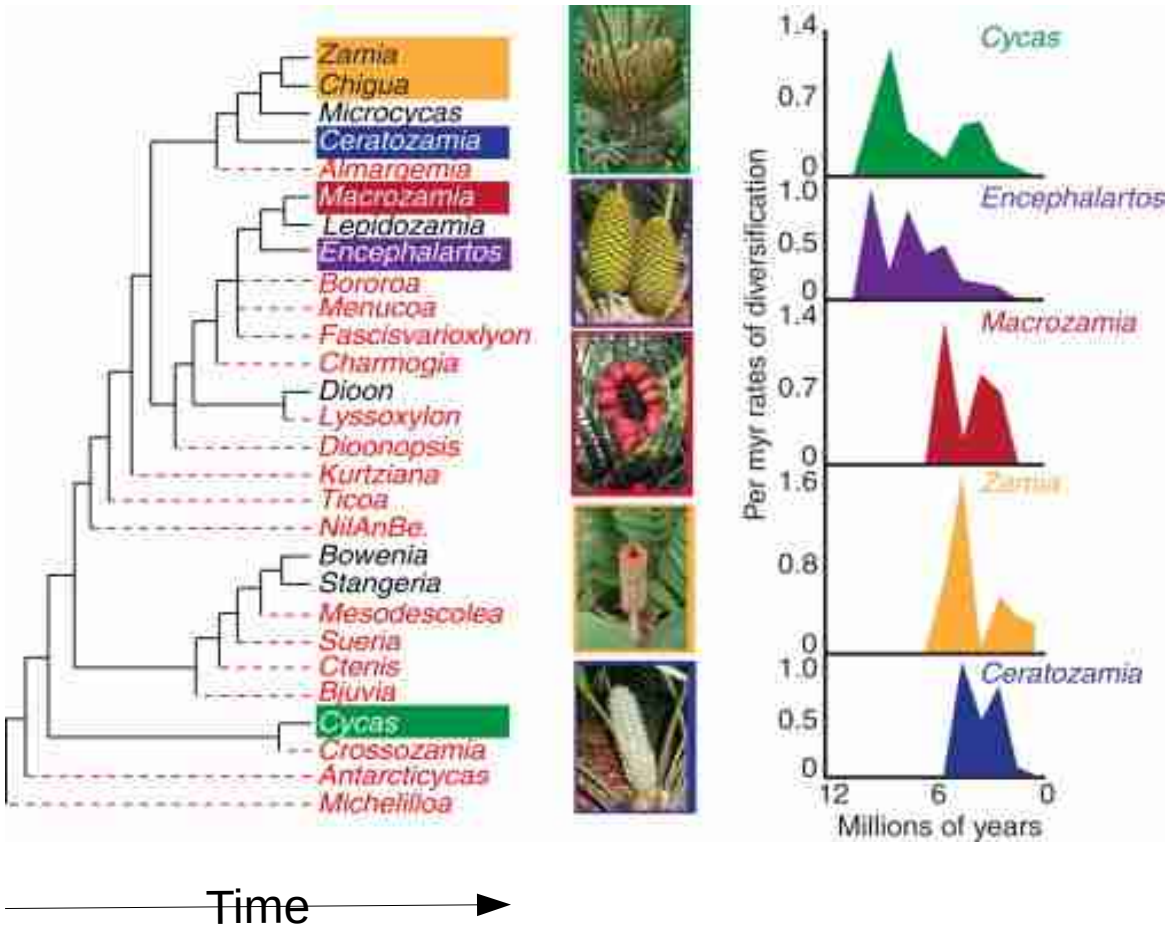
What is a phylogenetic tree?

The molecular laboratory received a bird sample from a plane that should be identified. Can you help them?

Become a real bird researcher

Where do the DNA samples come from? Watch this short movie!

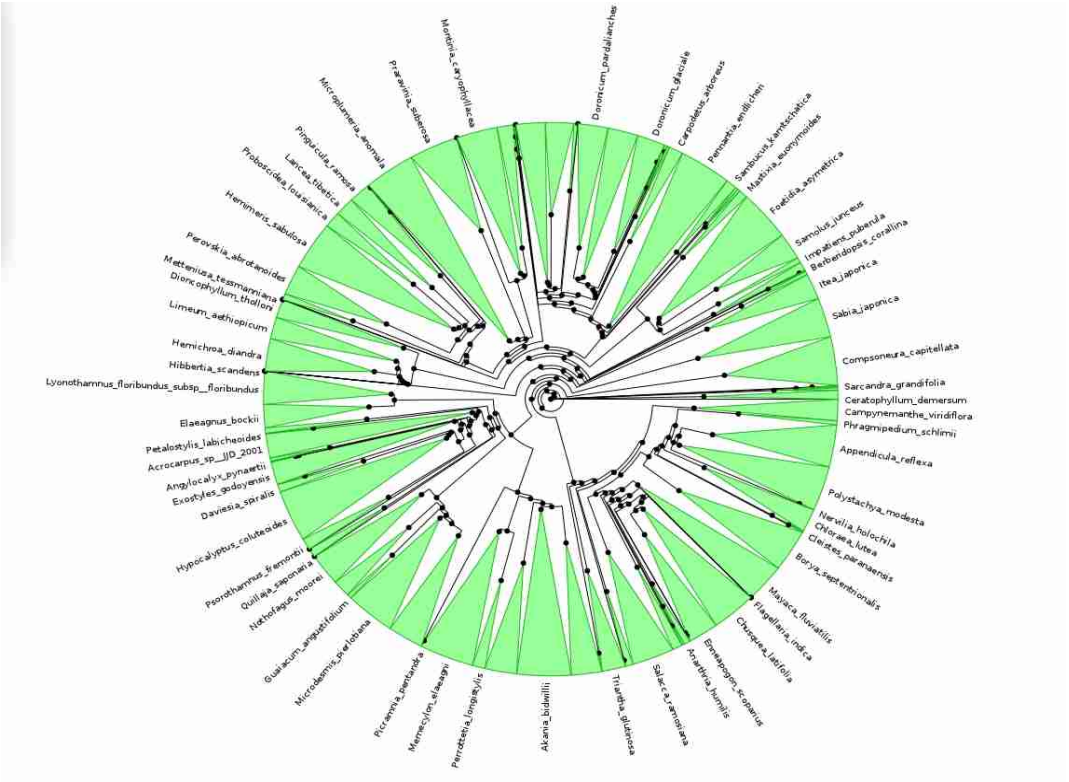
Diversification Rates



From: Charles C. Davis, Hanno Schaefer: “Plant Evolution: Pulses of Extinction and Speciation in Gymnosperm Diversity”, *Current Biology*, 2011.

Diversification Rates

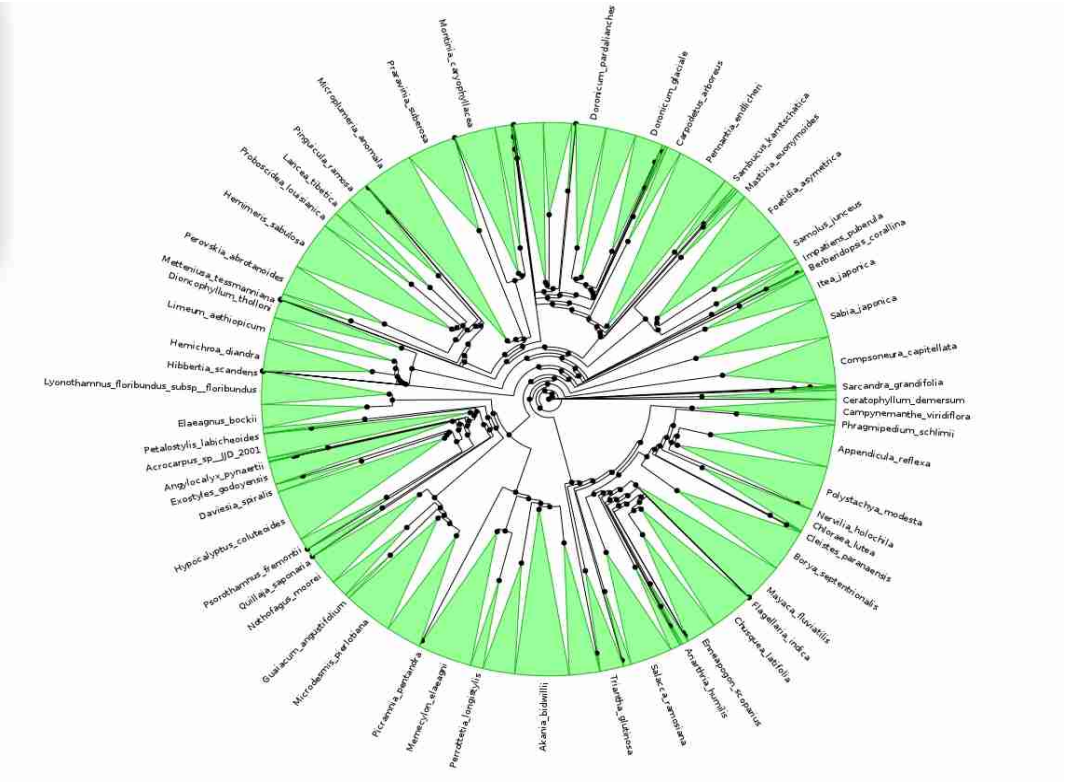
- With former PostDoc Stephen Smith: “Understanding angiosperm diversification using small and large phylogenetic trees”, *American Journal of Botany* 98 (3), 404-414, 2011.
- Largest tree of angiosperms computed to date
- 55,000 taxa



Diversification Rates

- With former PostDoc Stephen Smith: “Understanding angiosperm diversification using small and large phylogenetic trees”, *American Journal of Botany* 98 (3), 404-414, 2011.
- Largest tree of angiosperms computed to date
- 55,000 taxa

Visualizing big trees also represents a challenge → graph drawing & layout algorithms.

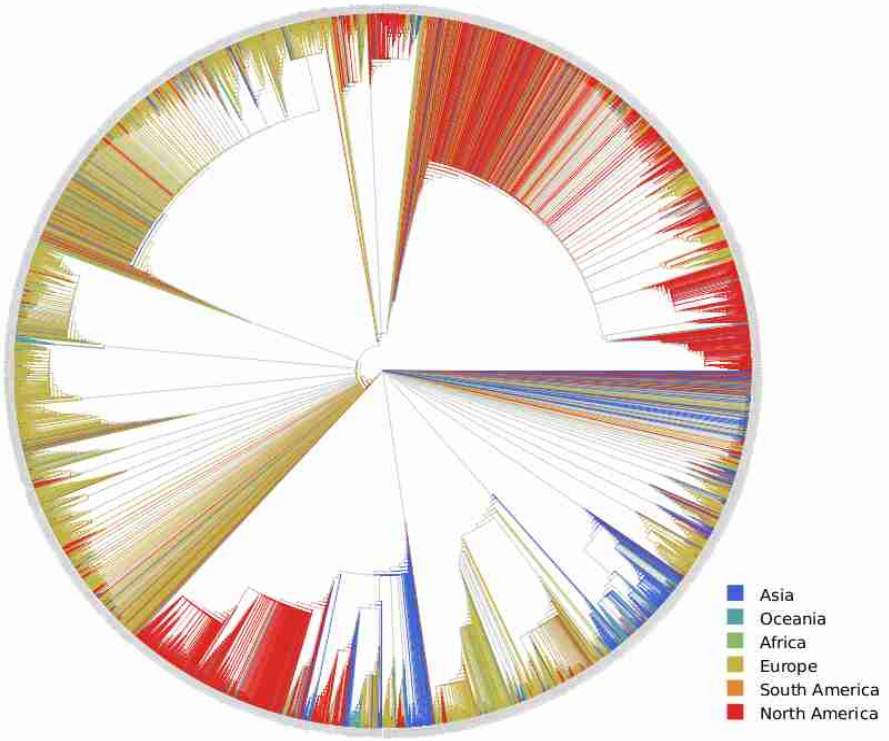


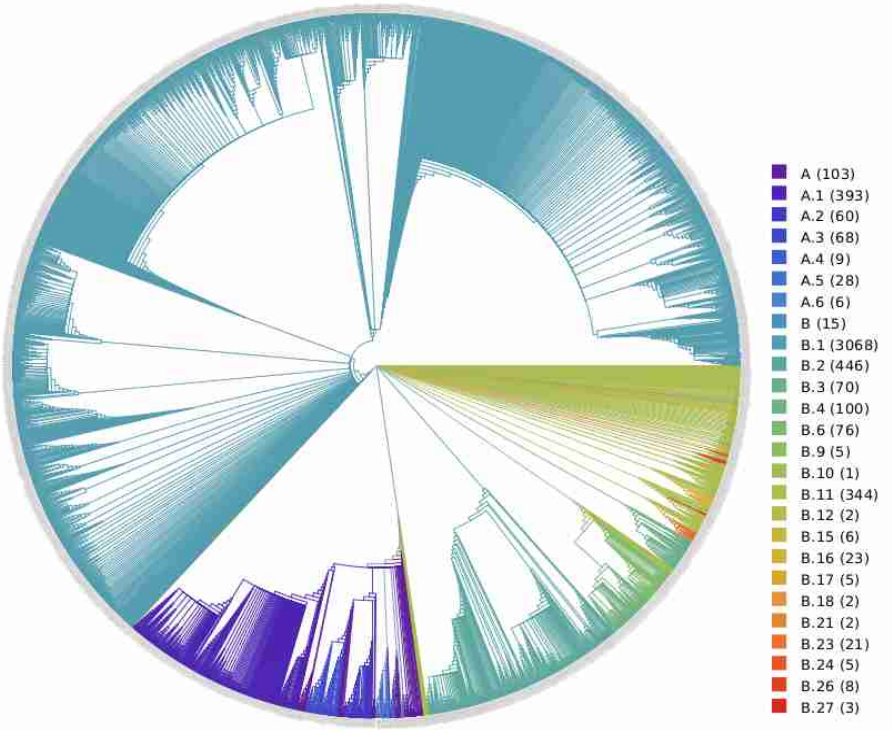
Influenza Outbreaks

And of course ... SARS-CoV-2

Phylogenetic analysis of SARS-CoV-2 data is difficult

Benoit Morel[*,1], Pierre Barbera[*,1], Lucas Czech[1], Ben Bettisworth[1], Lukas Hübner[1,2], Sarah Lutteropp[1], Dora Serdari[1], Evangelia-Georgia Kostaki[3], Ioannis Mamais[3], Alexey M Kozlov[1], Pavlos Pavlidis[4], Dimitrios Paraskevis[3], and Alexandros Stamatakis[1,2]





Snakebites

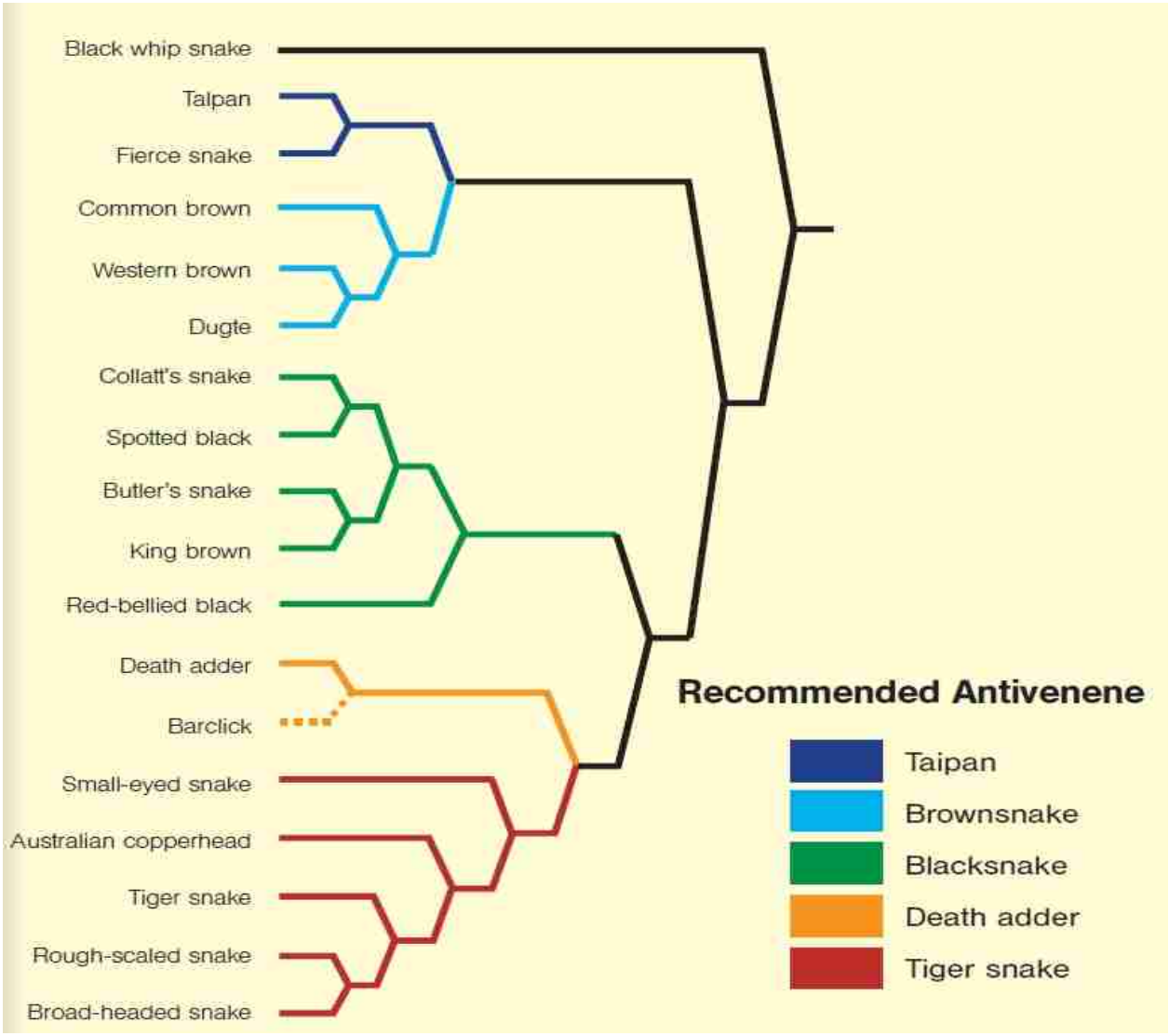
Australia has more poisonous snakes than any other continent, and many people die from snakebites each year. Developing effective antivenins is thus a high priority, but little is known about the venins of most species.

Phylogenetic analysis is helping with this task because venin properties correlate strongly with evolutionary relationships.

Although the red-bellied black snake looks very different from the king brown, it is actually closely related and can be treated with the same antivenin.

Conversely, the western brown looks very similar to the king brown, but it is only distantly related and thus responds best to different antivenin.

The phylogeny is also predictive: the recent demonstration that the poorly-known barclick is closely related to the death adder (orange lineage) predicts that the former is also highly dangerous and might respond to widely-available death adder antivenin.



Snakebites

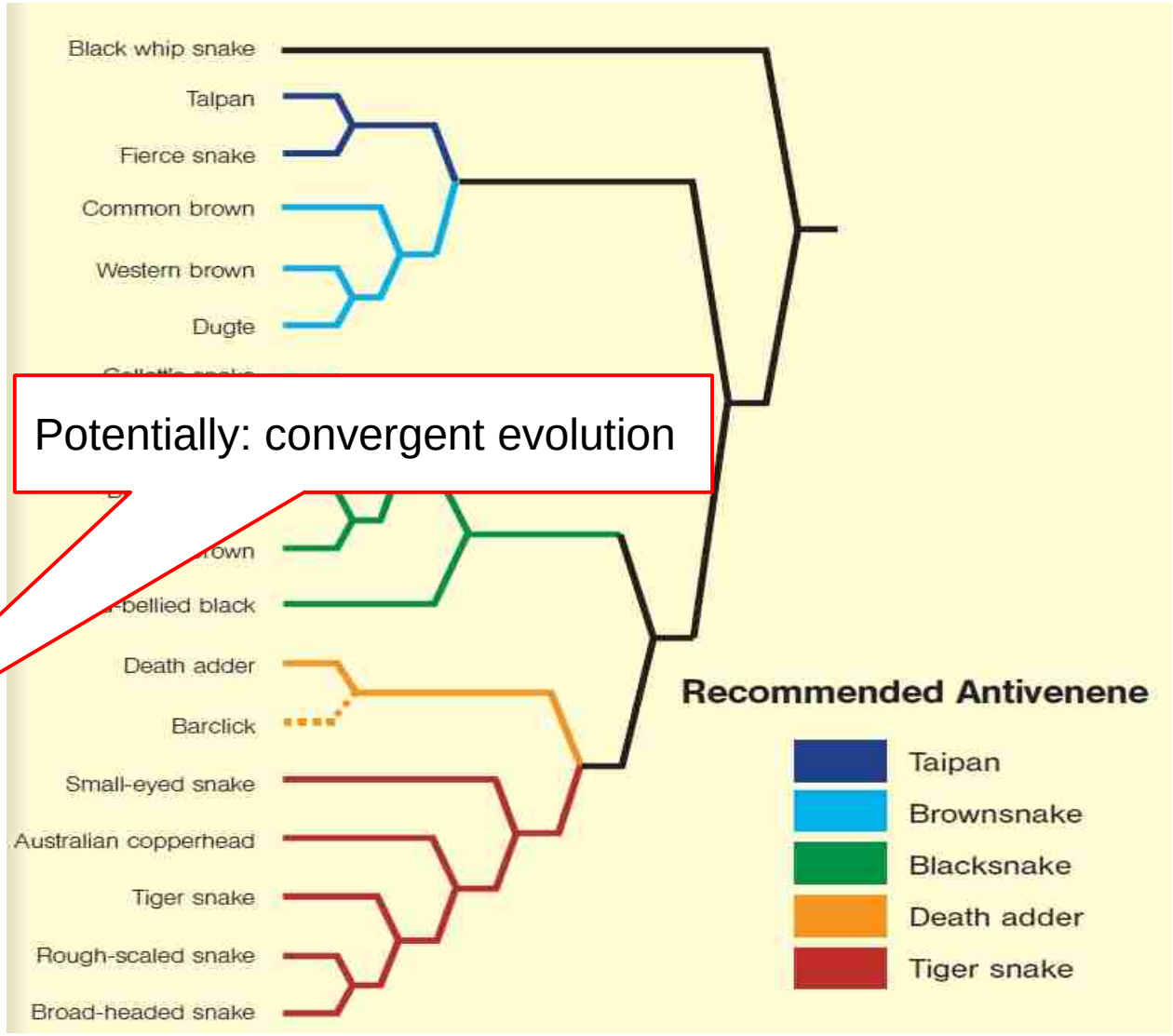
Australia has more poisonous snakes than any other continent, and many people die from snakebites each year. Developing effective antivenins is thus a high priority, but little is known about the venins of most species.

Phylogenetic analysis is helping with this task because venin properties correlate strongly with evolutionary relationships.

Although the red-bellied black snake looks very different from the king brown, it is actually closely related and can be treated with the same antivenin.

Conversely, the western brown looks very similar to the king brown, but it is only distantly related and thus responds best to different antivenin.

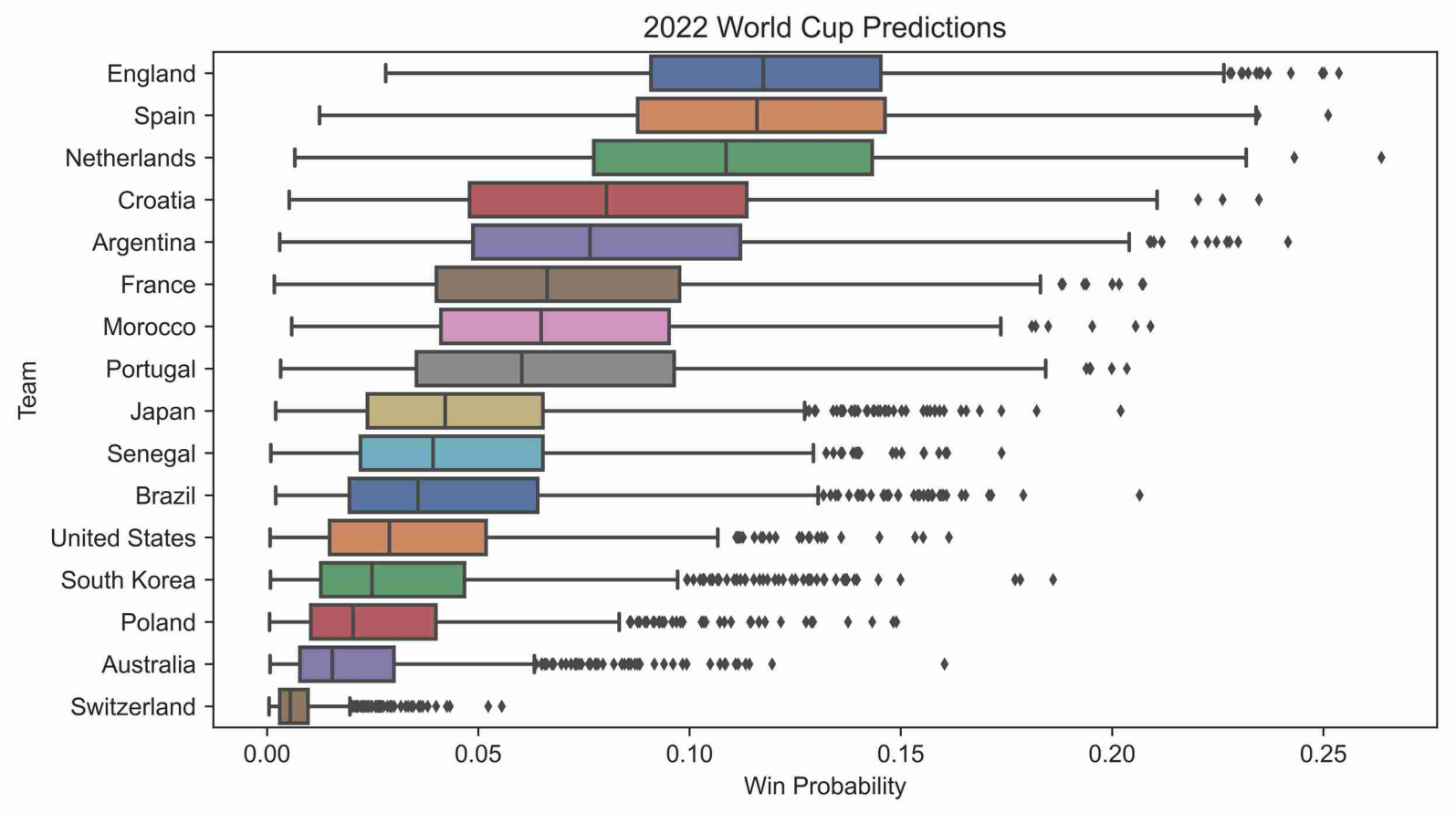
The phylogeny is also predictive: the recent demonstration that the poorly-known barclick is closely related to the death adder (orange lineage) predicts that the former is also highly dangerous and might respond to widely-available death adder antivenin.



What can we do with phylogenetic trees?

- identify unknown species
- estimate divergence times
- diversification rates
- viral outbreaks
- forensics → M.L. Metzker, D.P. Mindell, X.M. Liu, R.G. Ptak, R.A. Gibbs, D.M. Hillis: “Molecular evidence of HIV-1 transmission in a criminal case” PNAS: 99(22):14292-7, 2002.

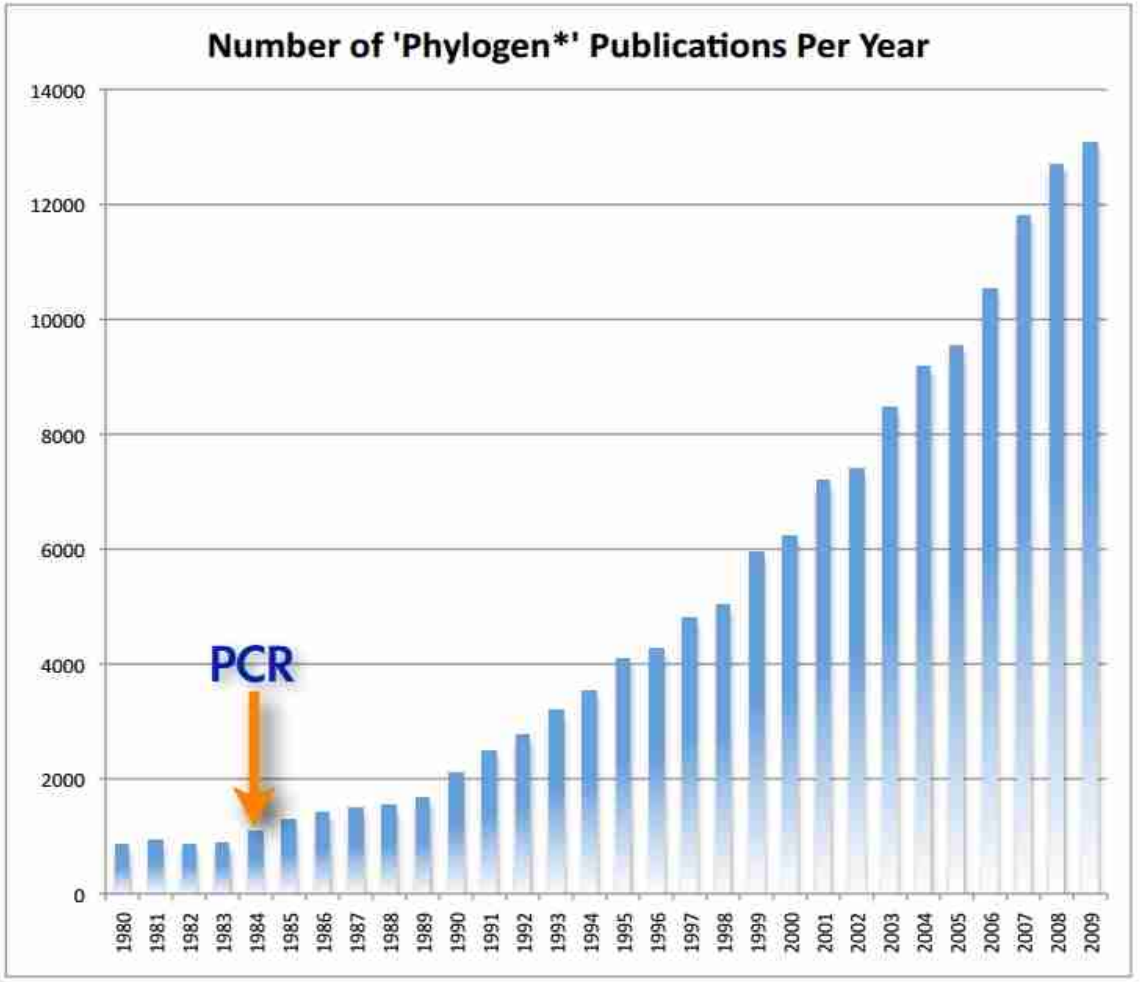
Phylogenetic Methods for Tournament Prediction



"Nothing in Biology makes sense, except in the light of evolution"

Why this increase in Phylogenetics papers?

→ Advances in:
- Sequencing technology
- Hardware
- Methods & Tools



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
  - *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

Back to Criteria and 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
  - *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

Less accurate, but faster

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
  - *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!

Less accurate, but faster

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 c
  - *Character-based methods*
    - → optimality criteria *f()* operate directly
      - → 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

Less accurate, but faster

What could be the computational limitation here?

Memory-intensive!

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 clus ing methods)
      - → *Neighbor Joining:* NJ
      - → *Unweighted Pair Group Method with Arithmetic M* GMA
    - → least-squares method: explicit optimality criteri
  - *Character-based methods*
    - → optimality criteria *f()* operate directly on t
      - → 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

Less accurate, but faster

Storing this matrix can become problematic memory-wise
→ out-of-core/external memory algorithms
→ e.g.: NINJA tool for Neighbor joining
"Large-scale neighbor-joining with ninja"
T Wheeler,
Algorithms in Bioinformatics, 2009

Out-of-core Algorithms

- Definition from Wikipedia:

  Out-of-core or External memory algorithms are algorithms that are designed to process data that is too large to fit into a computer's main memory at one time. Such algorithms must be optimized to efficiently fetch and access data stored in slow bulk memory such as hard drive or tape drives.

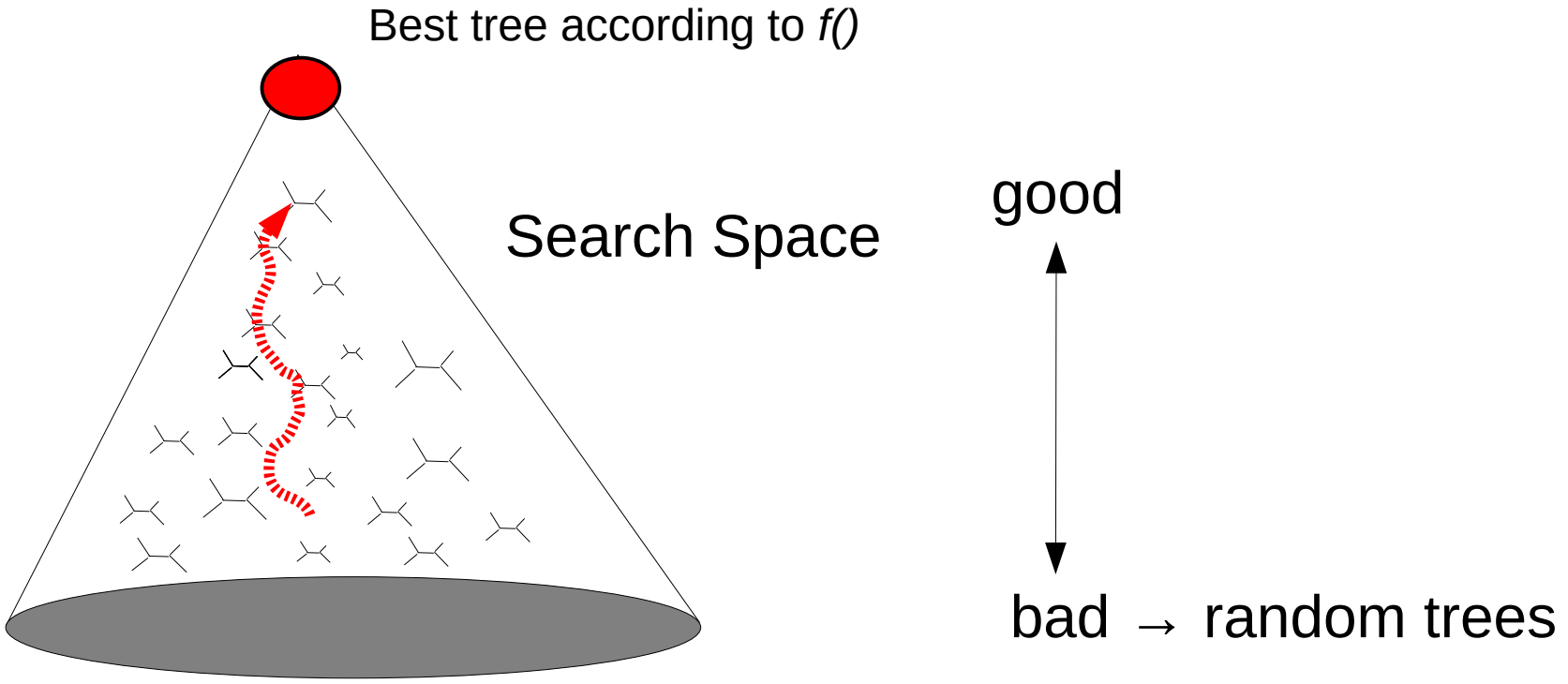
- We do the data transfer RAM ↔ disk explicitly from within the application code by using application-specific knowledge (e.g., about the data access patterns)
- This is to circumvent the paging procedure that would normally be initiated by the OS
- Out-of-core algorithms are typically much faster than the *application-agnostic* paging procedure carried out by the OS
- For an example from phylogenetics see:

  Fernando Izquierdo-Carrasco, Alexandros Stamatakis: "Computing the Phylogenetic Likelihood Function Out-of-Core", *IEEE HICOMB 2011 workshop*, Anchorage, USA, May 2011.

NP-Hardness

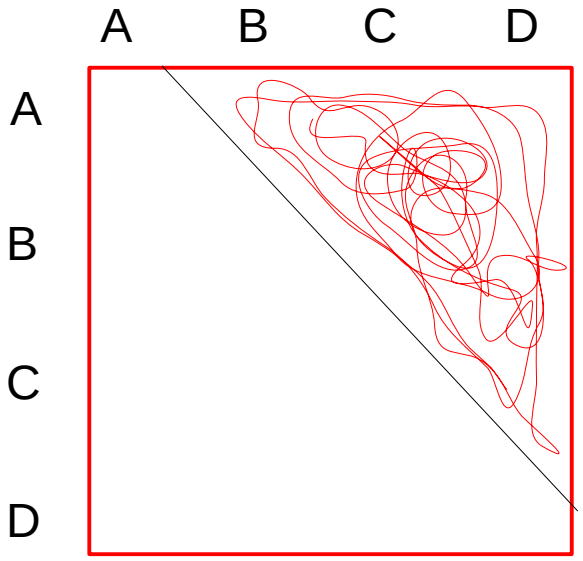
- Because of the **super-exponential increase** in the number of possible trees for *n* taxa ...
- all interesting optimality criteria on trees are NP-hard:
  - Least squares
  - Parsimony → discrete criterion
  - Likelihood → statistical criterion
  - Bayesian → integrate likelihood over entire tree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

Search Space



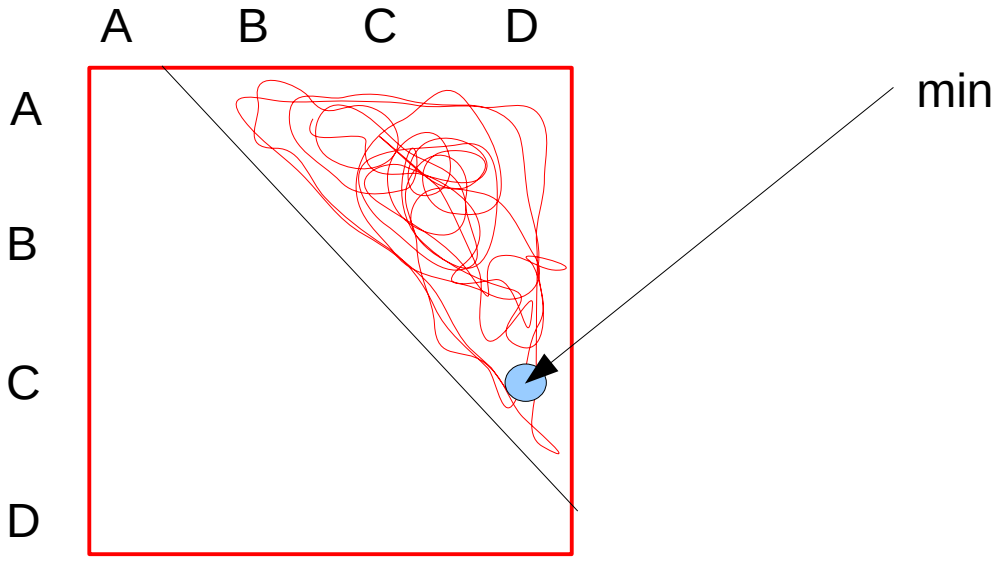
Let’s start with distance based methods/heuristics

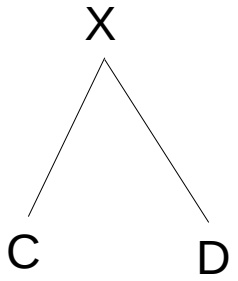
Neighbor Joining → Principle



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

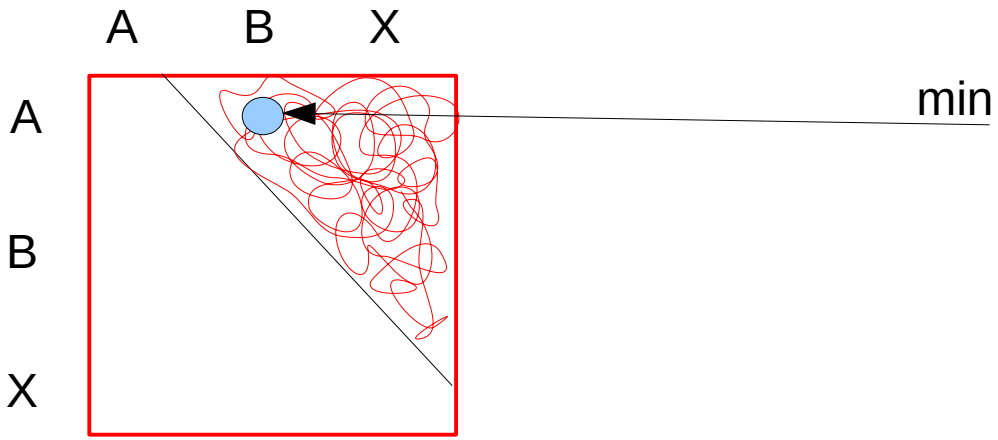
Neighbor Joining → Principle

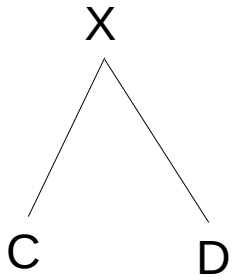




Given a kind of distance matrix $D_{i,j}$ where $i,j=1...4$
Find minimum and merge taxa

Neighbor Joining → Principle





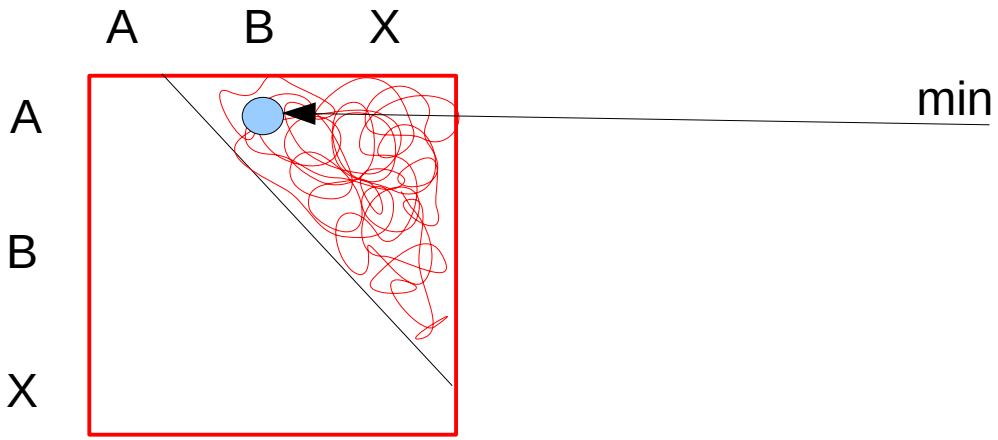
Given a kind of distance matrix $D_{i,j}$ where $i,j=1...4$

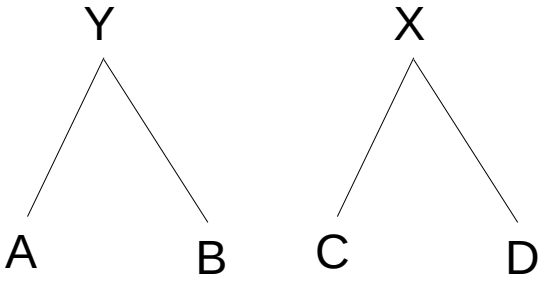
Find minimum and merge taxa

Compute a new distance matrix of size $n-1 = 3$

Find minimum

Neighbor Joining → Principle





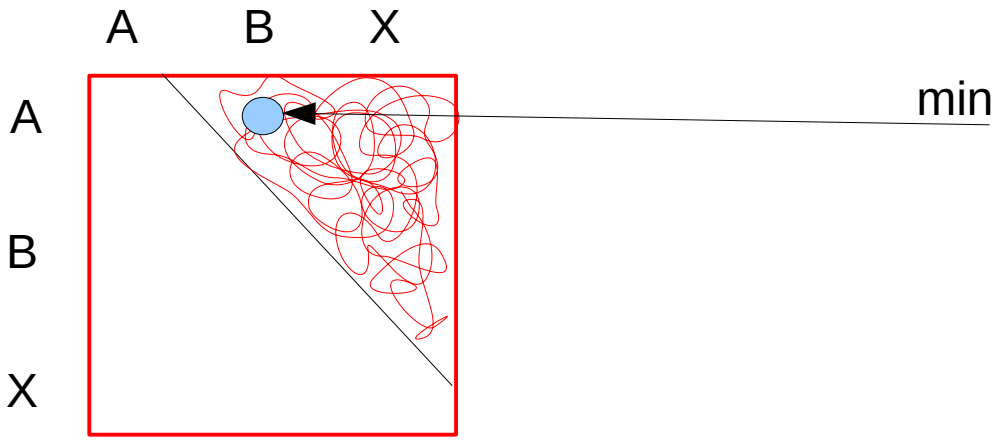
Given a kind of distance matrix $D_{i,j}$ where *i,j=1...4*

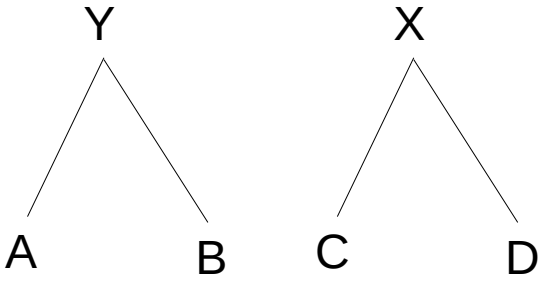
Find minimum and merge taxa

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

Neighbor Joining Algorithm

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

Neighbor Joining Algorithm

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

Distance matrix, usually denoted as *D*

i	u_i
A	(17+21+27)/2=32.5
B	(17+12+18)/2=23.5
C	(21+12+14)/2=23.5
D	(27+18+14)/2=29.5

Average distance

Neighbor Joining Algorithm

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

i	u_i
A	(17+21+27)/2=32.5
B	(17+12+18)/2=23.5
C	(21+12+14)/2=23.5
D	(27+18+14)/2=29.5

	A	B	C	D
A	-	-39	-35	-35
B		-	-35	-35
C			-	-39
D				-

$D_{ij}-u_i-u_j$

Usually denoted as *Q* matrix

Neighbor Joining Algorithm

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

i	u_i
A	(17+21+27)/2=32.5
B	(17+12+18)/2=23.5
C	(21+12+14)/2=23.5
D	(27+18+14)/2=29.5

	A	B	C	D
A	-	-39	-35	-35
B		-	-35	-35
C			-	-39
D				-

$D_{ij}-u_i-u_j$

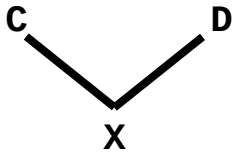
Neighbor Joining Algorithm

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

i	u_i
A	(17+21+27)/2=32.5
B	(17+12+18)/2=23.5
C	(21+12+14)/2=23.5
D	(27+18+14)/2=29.5

	A	B	C	D
A	-	-39	-35	-35
B		-	-35	-35
C			-	-39
D				-

$D_{ij}-u_i-u_j$



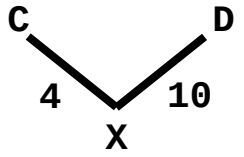
Neighbor Joining Algorithm

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

i	u_i
A	(17+21+27)/2=32.5
B	(17+12+18)/2=23.5
C	(21+12+14)/2=23.5
D	(27+18+14)/2=29.5

	A	B	C	D
A	-	-39	-35	-35
B		-	-35	-35
C			-	-39
D				-

$D_{ij}-u_i-u_j$

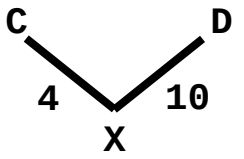


$b_C = 0.5 \times 14 + 0.5 \times (23.5-29.5) = 4$

$b_D = 0.5 \times 14 + 0.5 \times (29.5-23.5) = 10$

Neighbor Joining Algorithm

	A	B	C	D	X
A	-	17	21	27	
B		-	12	18	
C			-	14	
D				-	
X					-

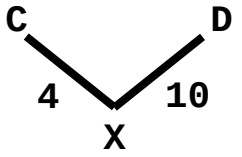


Neighbor Joining Algorithm

	A	B	C	D	X
A	-	17	21	27	
B		-	12	18	
C			-	14	
D				-	
X					-

$$D_{XA} = (D_{CA} + D_{DA} - D_{CD})/2$$
$$= (21 + 27 - 14)/2$$
$$= 17$$

$$D_{XB} = (D_{CB} + D_{DB} - D_{CD})/2$$
$$= (12 + 18 - 14)/2$$
$$= 8$$

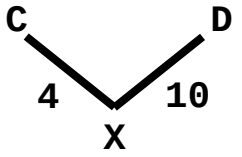


Neighbor Joining Algorithm

	A	B	C	D	X
A	-	17	21	27	17
B		-	12	18	8
C			-	14	
D				-	
X					-

$$D_{XA} = (D_{CA} + D_{DA} - D_{CD})/2$$
$$= (21 + 27 - 14)/2$$
$$= 17$$

$$D_{XB} = (D_{CB} + D_{DB} - D_{CD})/2$$
$$= (12 + 18 - 14)/2$$
$$= 8$$

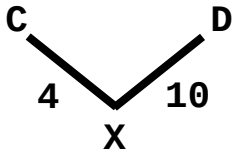


Neighbor Joining Algorithm

	A	B	X
A	-	17	17
B		-	8
X			-

$$D_{XA} = (D_{CA} + D_{DA} - D_{CD})/2$$
$$= (21 + 27 - 14)/2$$
$$= 17$$

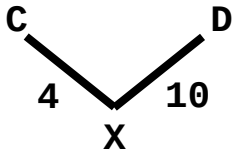
$$D_{XB} = (D_{CB} + D_{DB} - D_{CD})/2$$
$$= (12 + 18 - 14)/2$$
$$= 8$$



Neighbor Joining Algorithm

	A	B	X
A	-	17	17
B		-	8
X			-

i	u_i
A	(17+17)/1 = 34
B	(17+8)/1 = 25
X	(17+8)/1 = 25



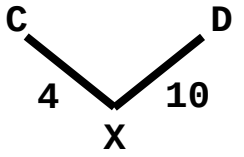
Neighbor Joining Algorithm

	A	B	X
A	-	17	17
B		-	8
X			-

i	u_i
A	(17+17)/1 = 34
B	(17+8)/1 = 25
X	(17+8)/1 = 25

	A	B	X
A	-	-42	-28
B		-	-28
X			-

$D_{ij}-u_i-u_j$



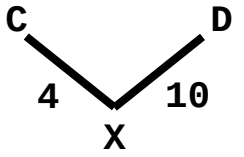
Neighbor Joining Algorithm

	A	B	X
A	-	17	17
B		-	8
X			-

i	u_i
A	(17+17)/1 = 34
B	(17+8)/1 = 25
X	(17+8)/1 = 25

	A	B	X
A	-	-42	-28
B		-	-28
X			-

$D_{ij}-u_i-u_j$



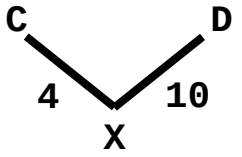
Neighbor Joining Algorithm

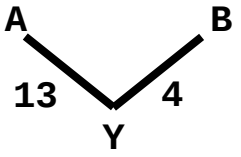
	A	B	X
A	-	17	17
B		-	8
X			-

i	u_i
A	(17+17)/1 = 34
B	(17+8)/1 = 25
X	(17+8)/1 = 25

	A	B	X
A	-	-42	-28
B		-	-28
X			-

$D_{ij}-u_i-u_j$



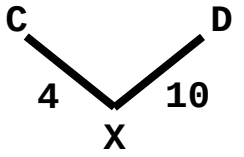


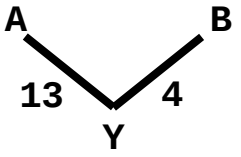
$b_A = 0.5 \times 17 + 0.5 \times (34-25) = 13$

$b_D = 0.5 \times 17 + 0.5 \times (25-34) = 4$

Neighbor Joining Algorithm

	A	B	X	Y
A	-	17	17	
B		-	8	
X			-	
Y				

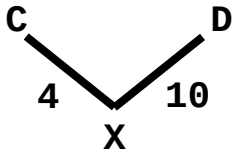


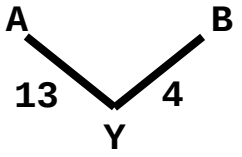


Neighbor Joining Algorithm

	A	B	X	Y
A	-	17	17	
B		-	8	
X			-	4
Y				

$$D_{YX} = (D_{AX} + D_{BX} - D_{AB})/2$$
$$= (17 + 8 - 17)/2$$
$$= 4$$

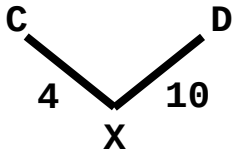


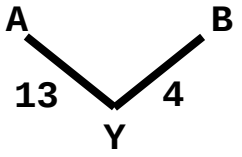


Neighbor Joining Algorithm

	X	Y
X	-	4
Y		-

$$D_{YX} = (D_{AX} + D_{BX} - D_{AB})/2$$
$$= (17 + 8 - 17)/2$$
$$= 4$$

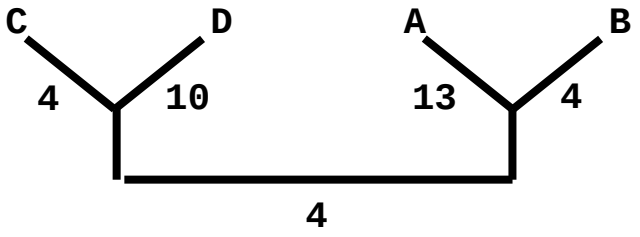




Neighbor Joining Algorithm

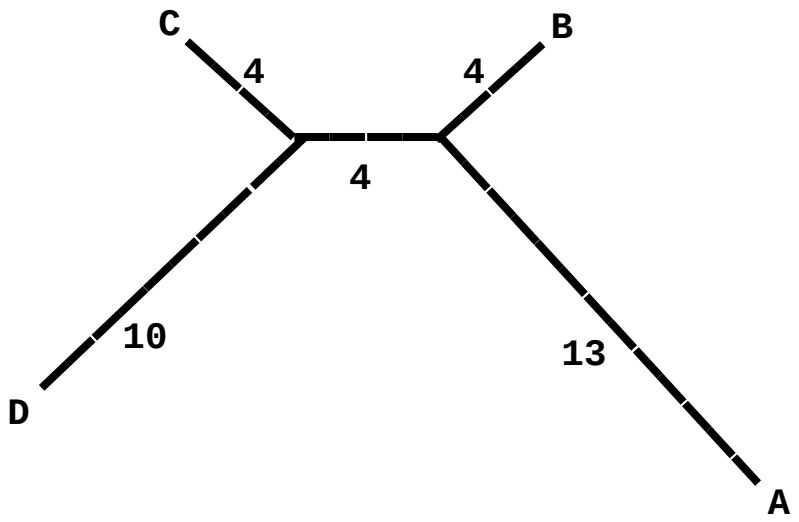
	X	Y
X	-	4
Y		-

$$D_{YX} = (D_{AX} + D_{BX} - D_{AB})/2$$
$$= (17 + 8 - 17)/2$$
$$= 4$$



Neighbor Joining Algorithm

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



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				

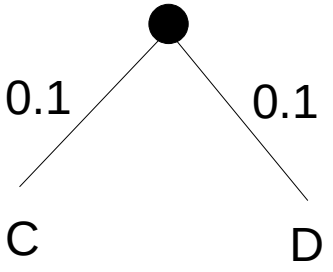
We will first walk through the algorithm and then look at the formal description!

UPGMA example

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

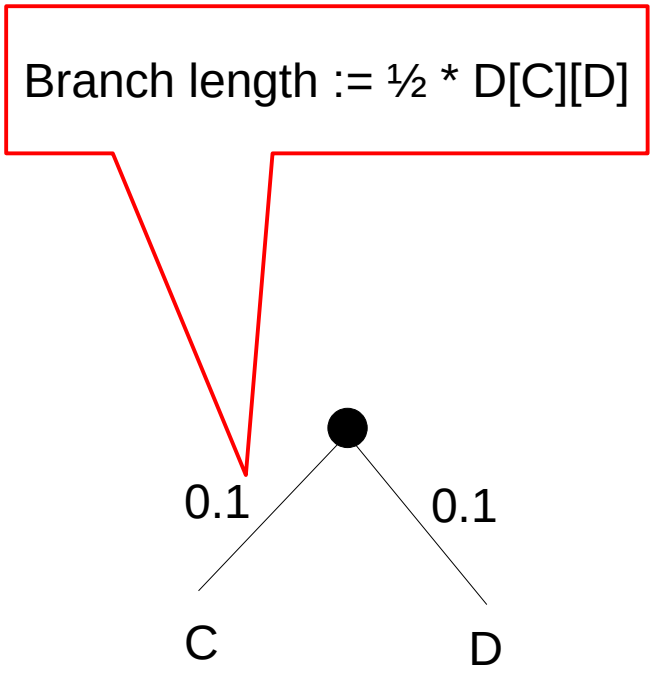
UPGMA example

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



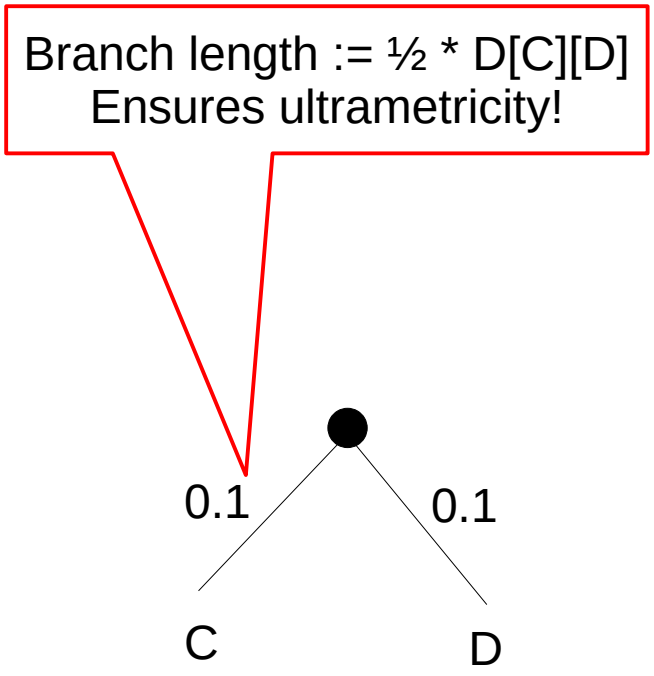
UPGMA example

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



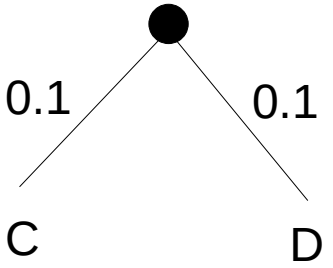
UPGMA example

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



UPGMA example

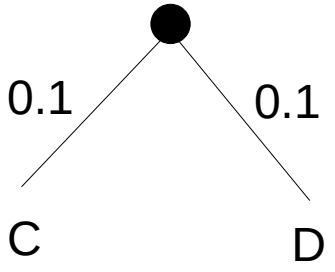
	A	B	C	D
A		0.4	0.6	0.6
B			0.6	0.6
C				0.2
D				



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

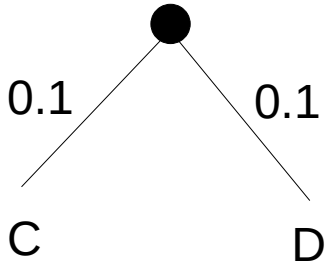
UPGMA example

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



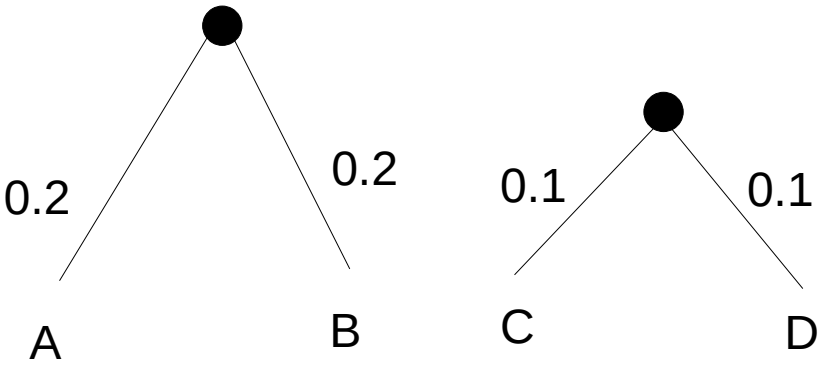
UPGMA example

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



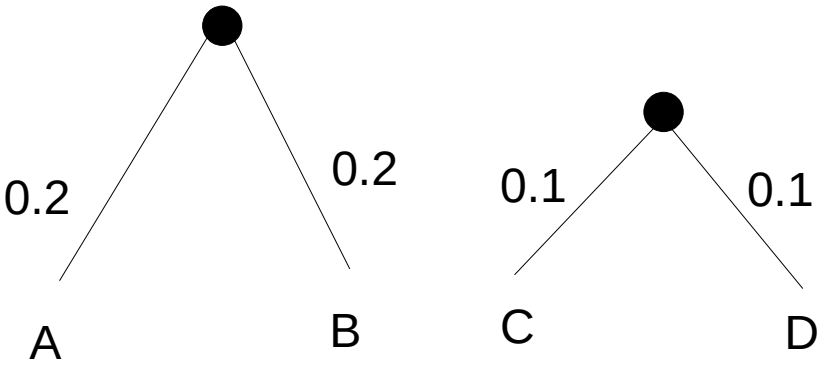
UPGMA example

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



UPGMA example

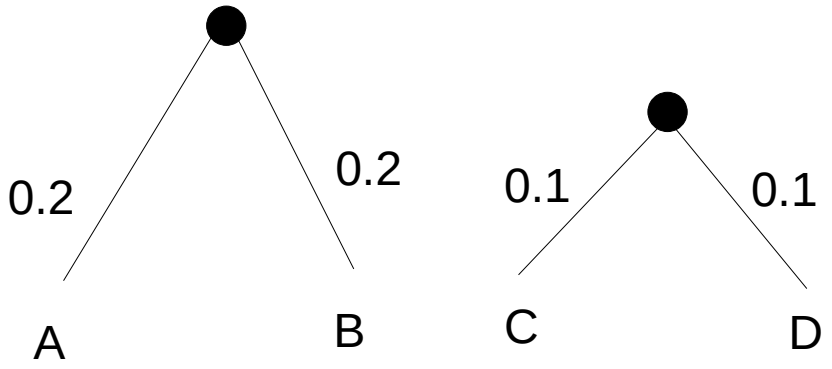
	A	B	(C,D)
A		0.4	0.6
B			0.6
(C, D)			



D[A,B][C,D] = ½ * 0.6 + ½ * 0.6

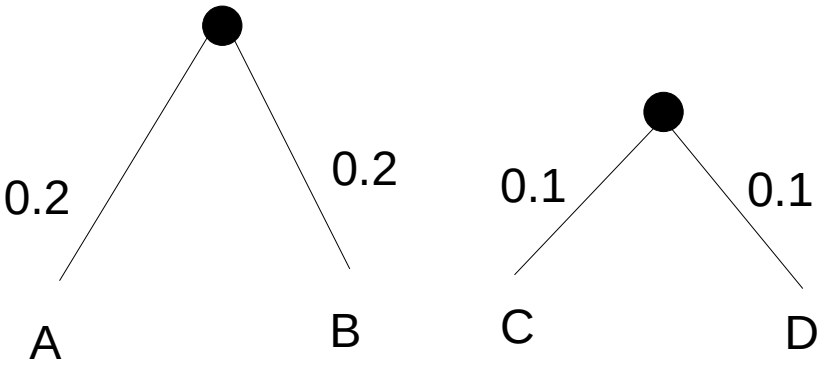
UPGMA example

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



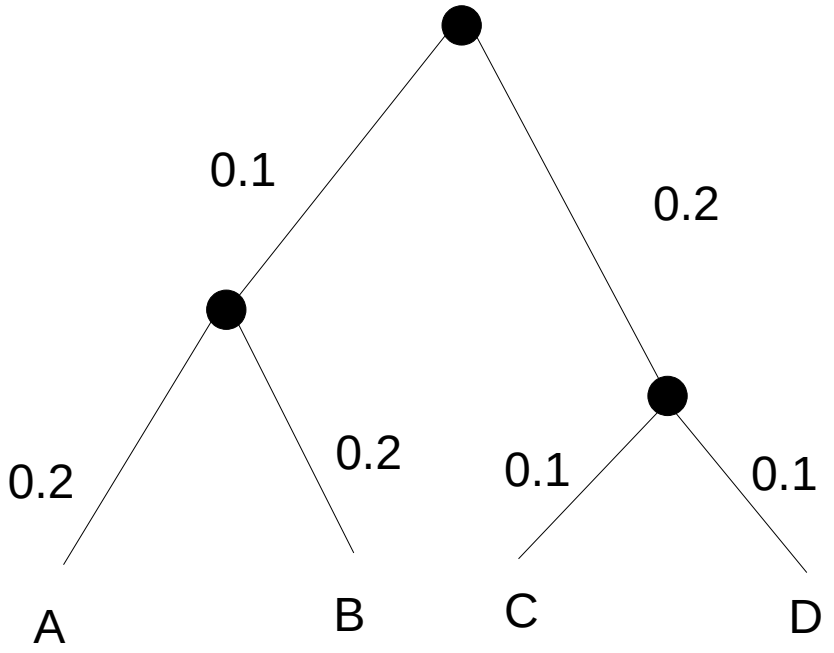
UPGMA example

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



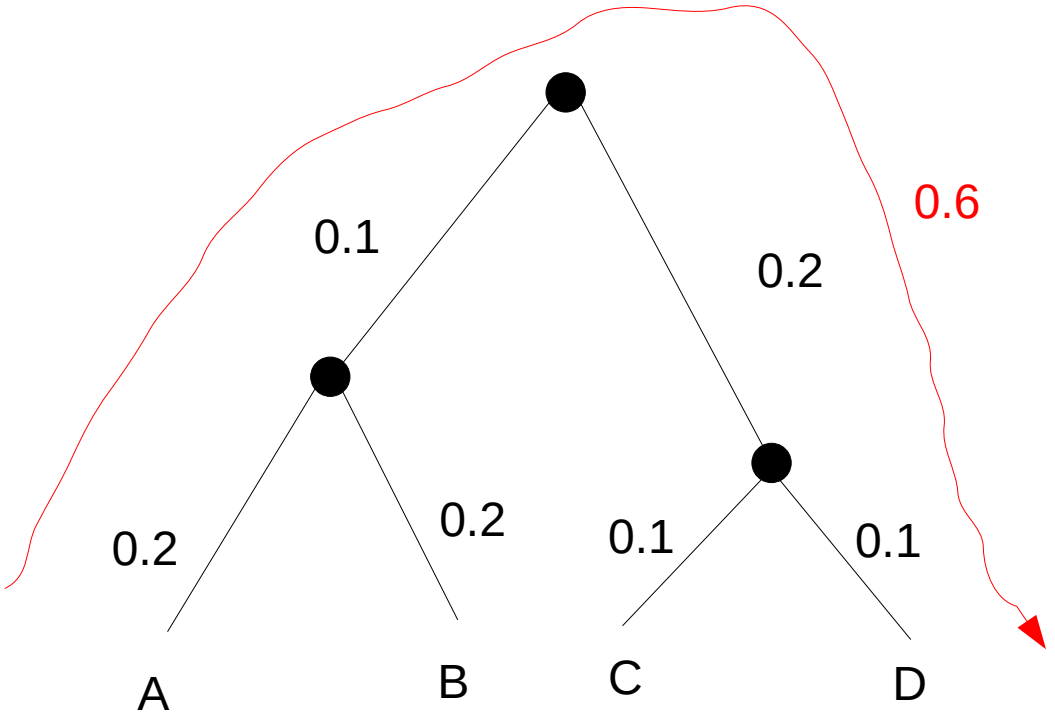
UPGMA example

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



UPGMA example

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

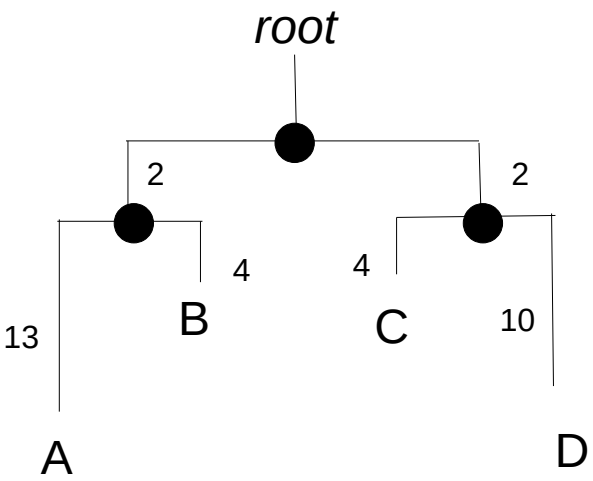


UPGMA Formal description

- Find the minimum *D[i][j]*
- Merge *i* and *j* → *(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* → *(i,j)* and *j* → *(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

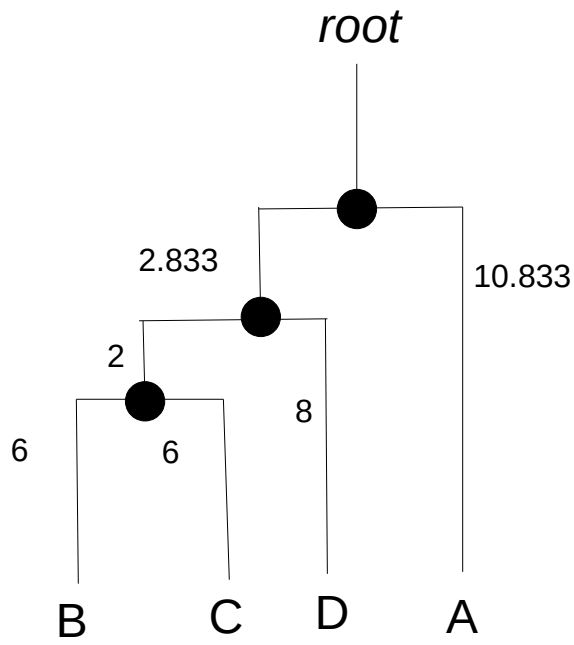
- Can yield misleading results
- Most trees are not ultrametric → do not have equal evolutionary rates among all lineages



True tree

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

Patristic distance matrix

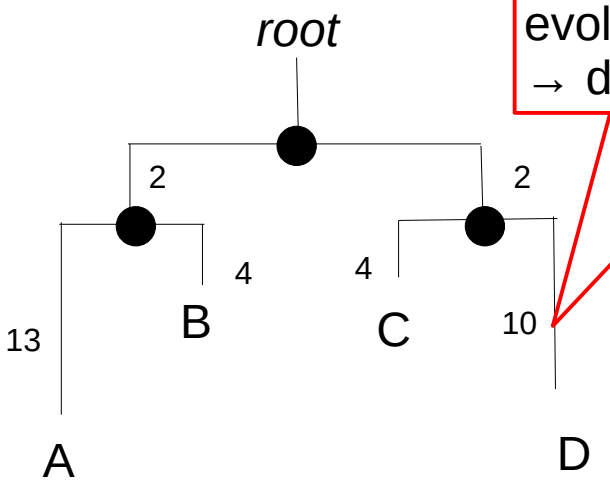


UPGMA tree

UPGMA on non-ultrametric trees

- Can yield misleading results
- Most trees are not ultrametric → do not have equal evolutionary r ages

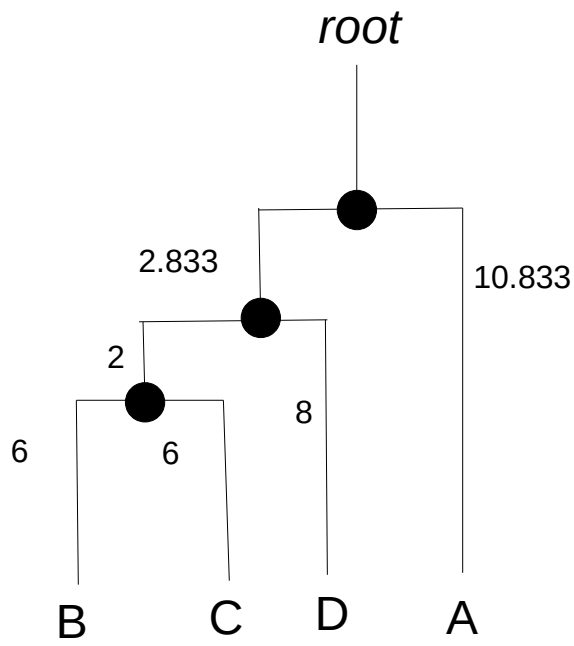
Imagine a higher evolutionary pressure!
→ difficult life conditions!



True tree

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

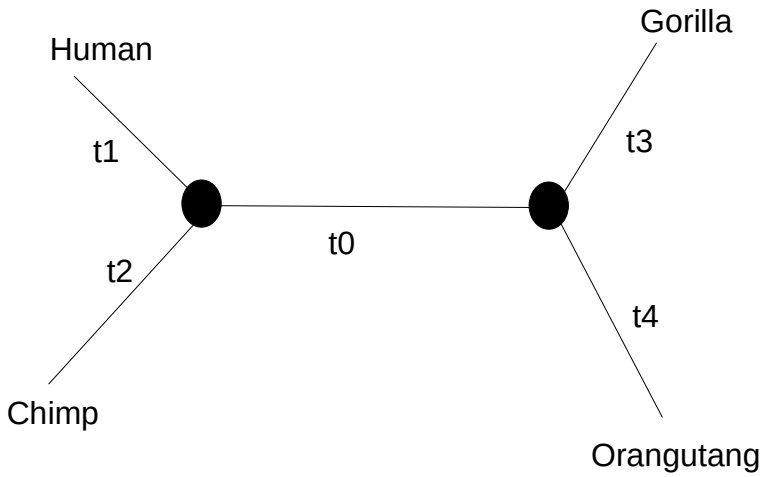
Patristic distance matrix



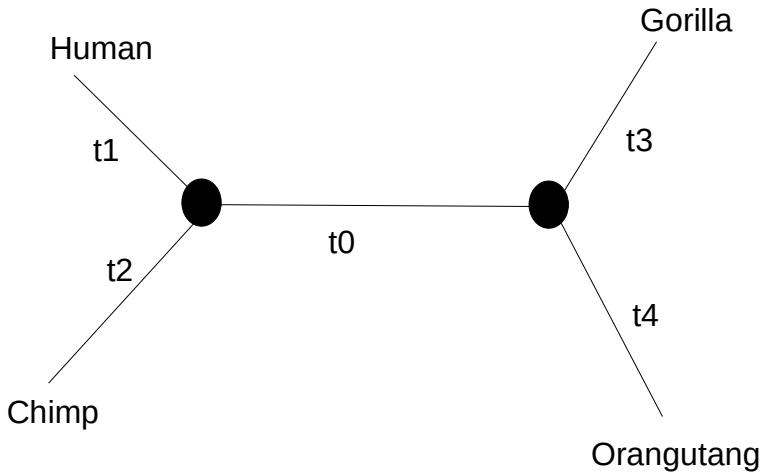
UPGMA tree

Now let’s look at a distance-based criterion

Least Squares



Least Squares



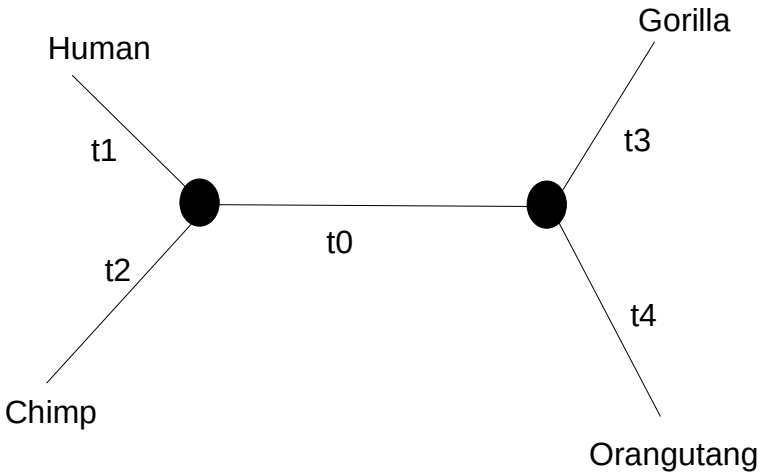
Patristic distances

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				

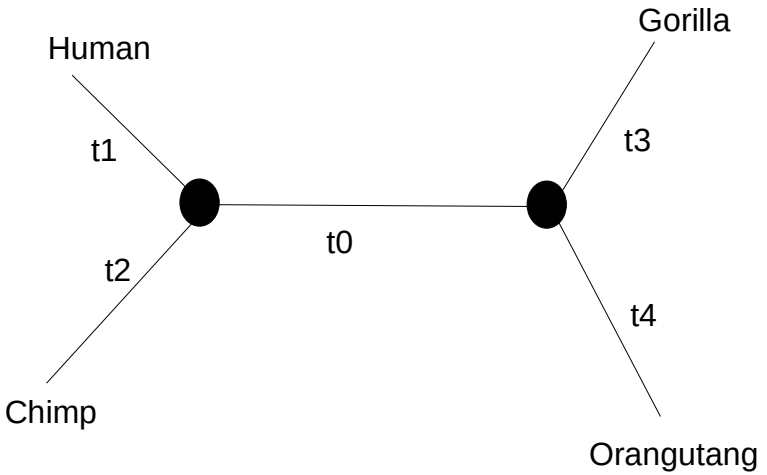


d[H][C] = t1 + t2
d[H][G] = t1 + t0 + t3
d[H][O] = t1 + t0 + t4
d[C][G] = t2 + t0 + t3
d[C][O] = t2 + t0 + t4
d[G][O] = t3 + t4

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, ..., t4* such that deviation of *d[i][j]* from *D[i][j]* is minimized!

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

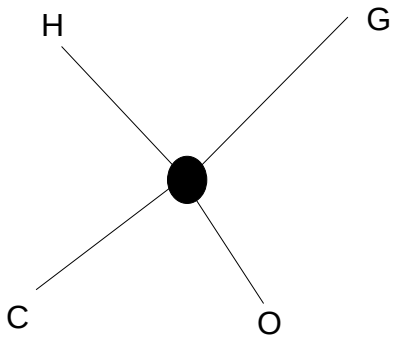
$d[H][C] = t1 + t2$
$d[H][G] = t1 + t0 + t3$
$d[H][O] = t1 + t0 + t4$
$d[C][G] = t2 + t0 + t3$
$d[C][O] = t2 + t0 + t4$
$d[G][O] = t3 + t4$

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

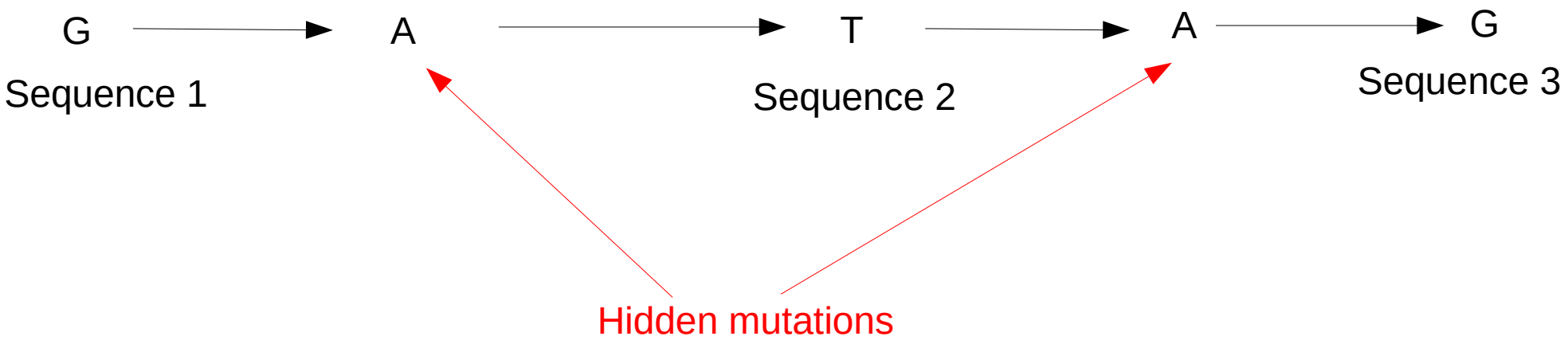
- *NP-hard* because of tree search problem
- Scoring a single tree takes time between $O(n)$ to $O(n^3)$
- There also exist weighted versions:

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

  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

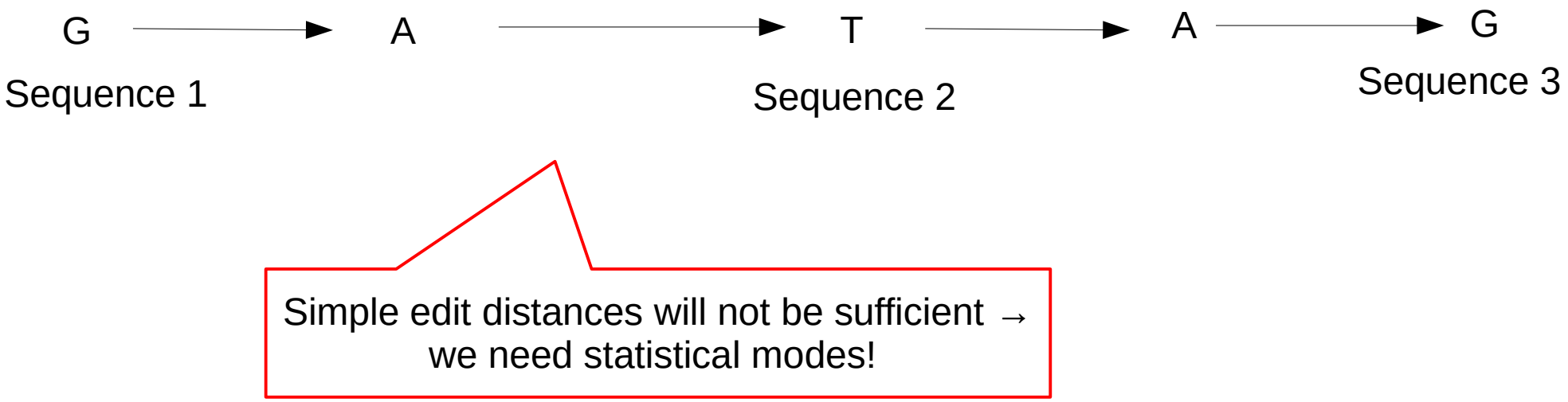
Distances

- 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



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

Time for a break

- Thus far, we have seen distance-based methods and distance-based criteria
- Let’s take a break before we start talking about character-based methods

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!

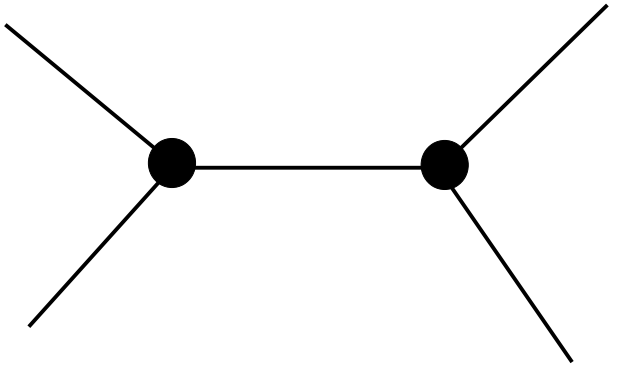
    → remember the number of trees!

    → this is also NP-hard!

Parsimony

MSA

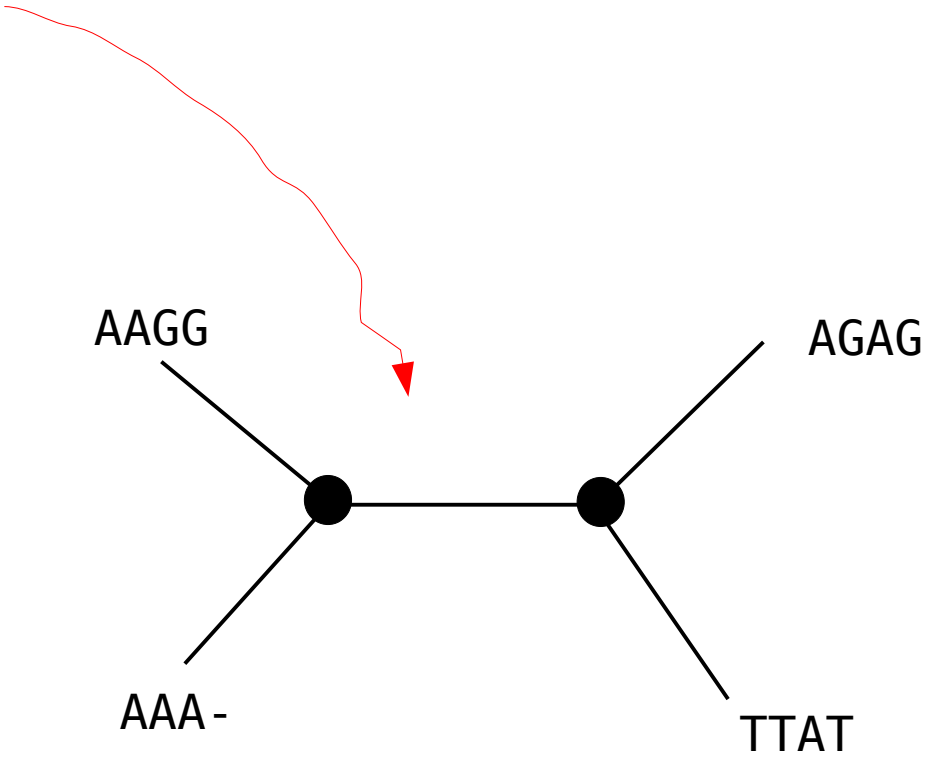
```
S1: AAGG
S2: AAA-
S3: AGAG
S4: TTAT
```



Parsimony

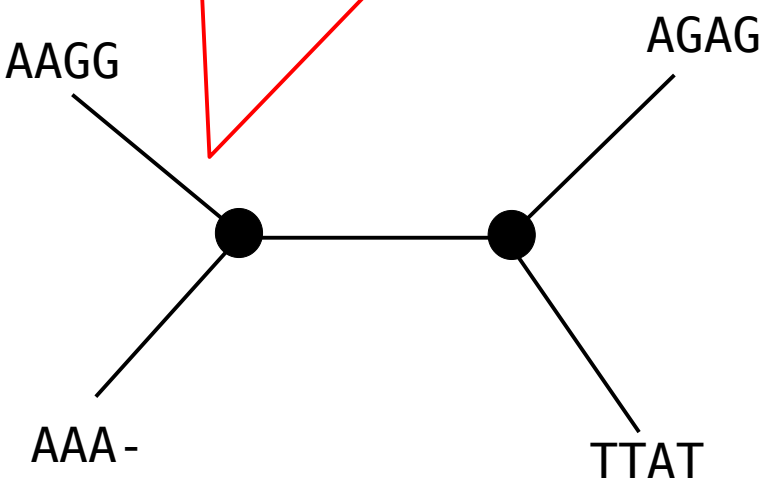
MSA

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



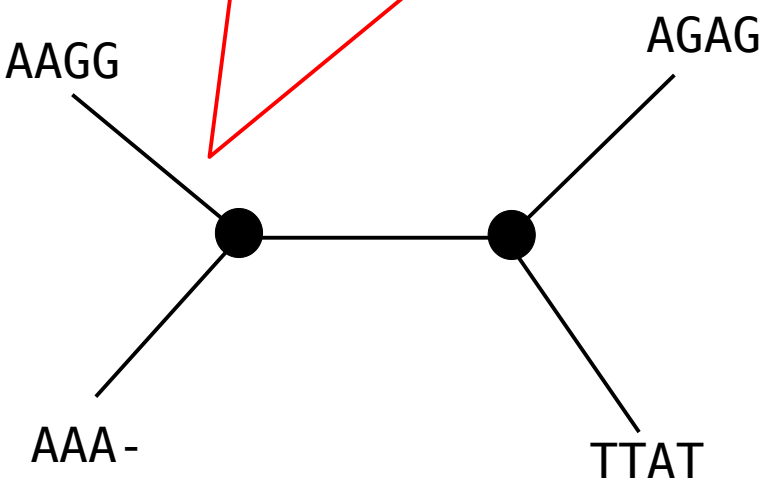
Parsimony

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



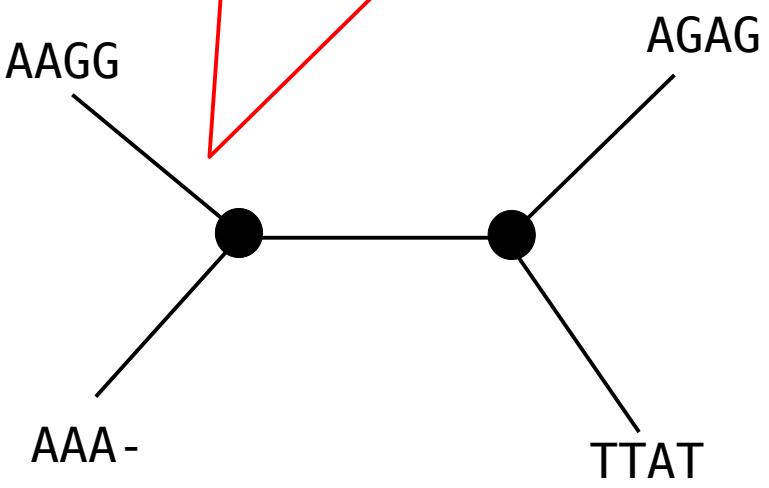
Parsimony

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

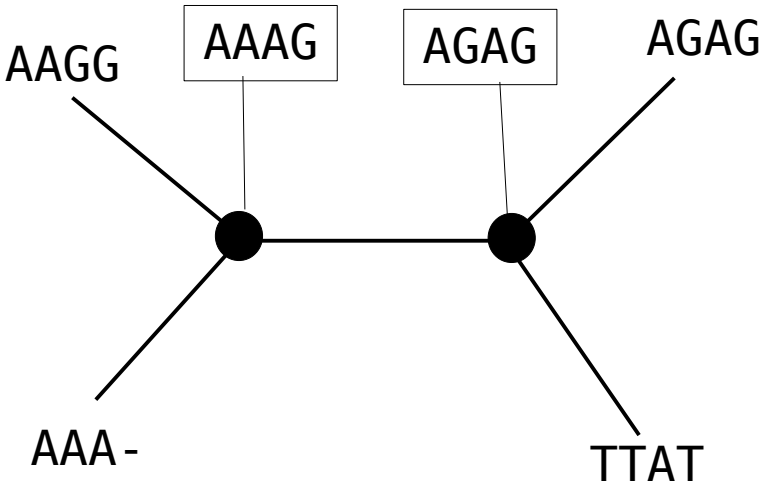


Parsimony

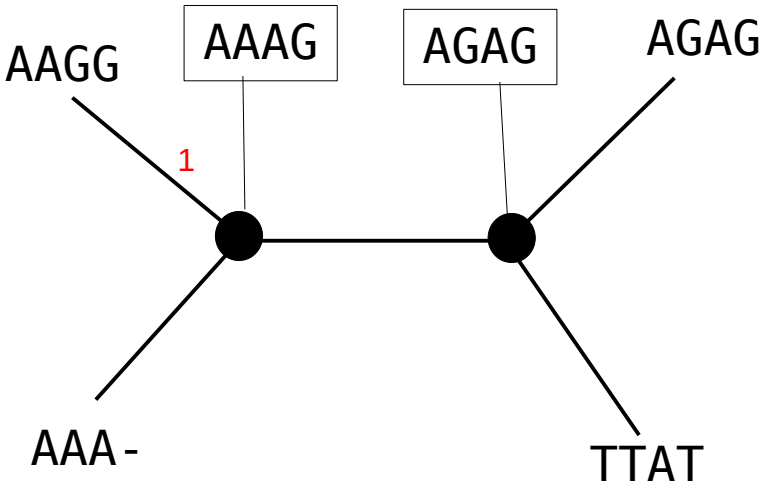
What could the inner sequences look like?



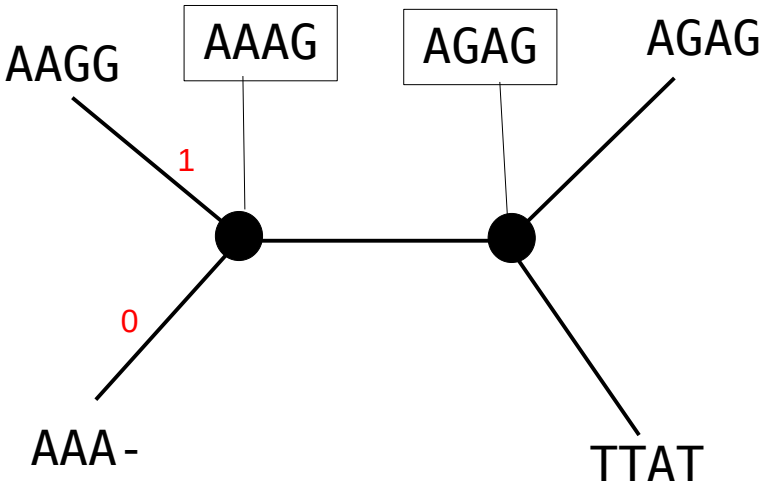
Parsimony



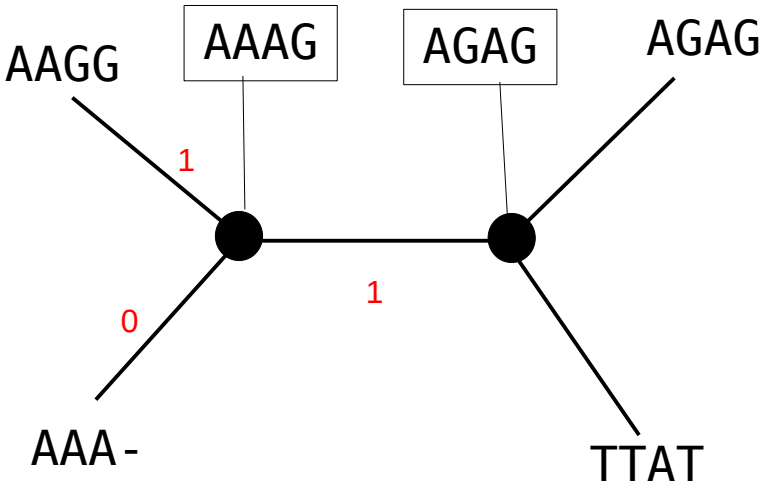
Parsimony



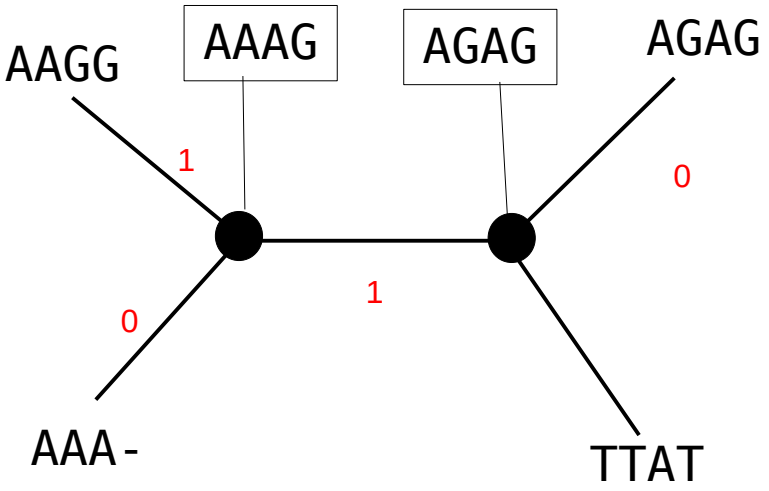
Parsimony



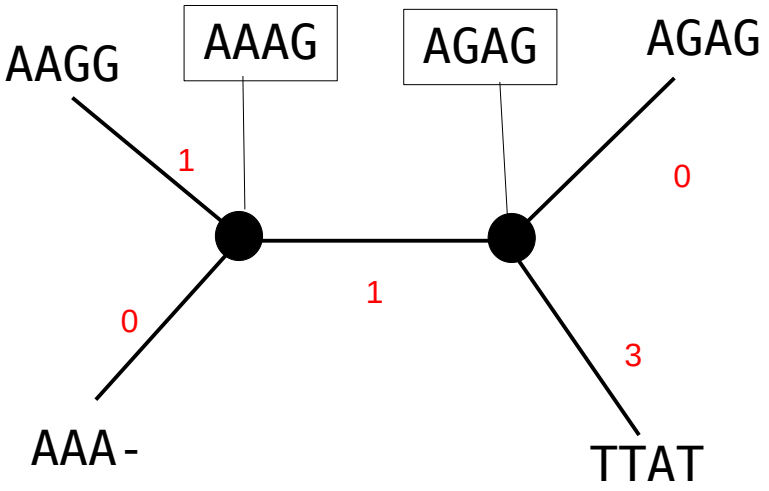
Parsimony



Parsimony

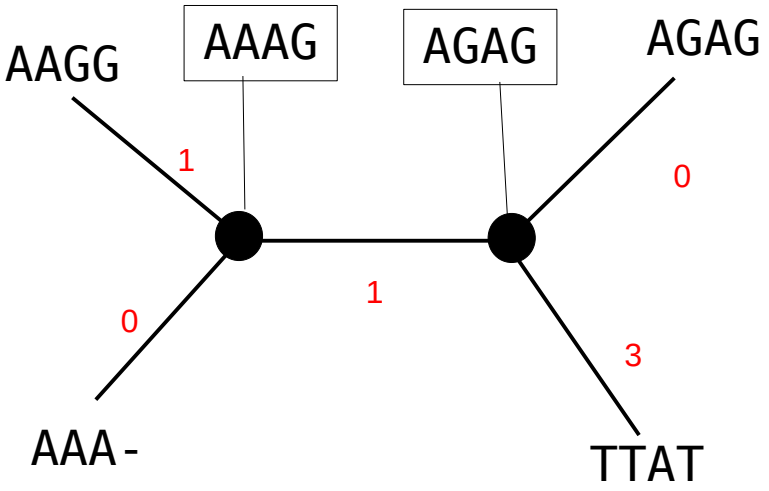


Parsimony



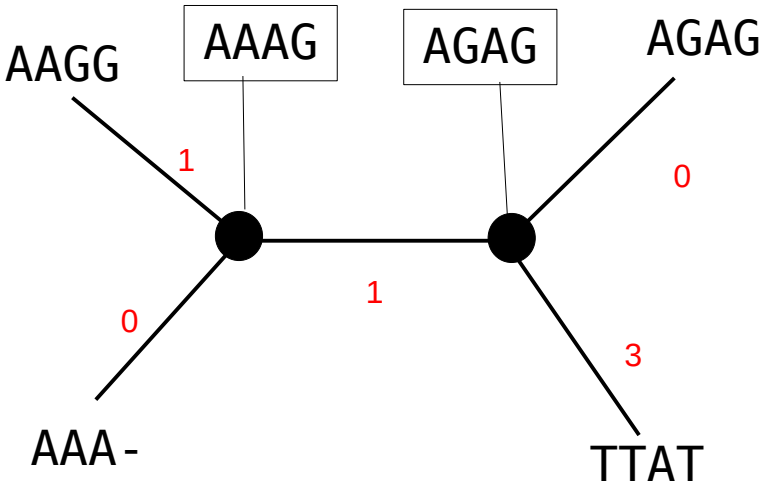
Parsimony

Parsimony Score of this tree = 5



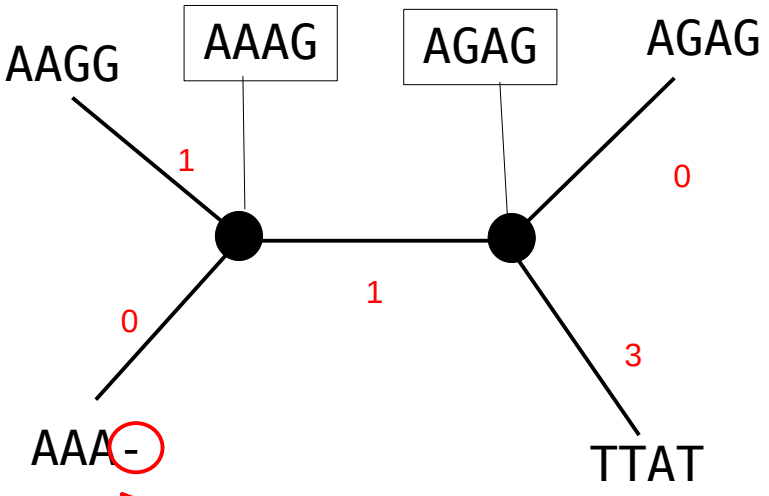
Parsimony

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



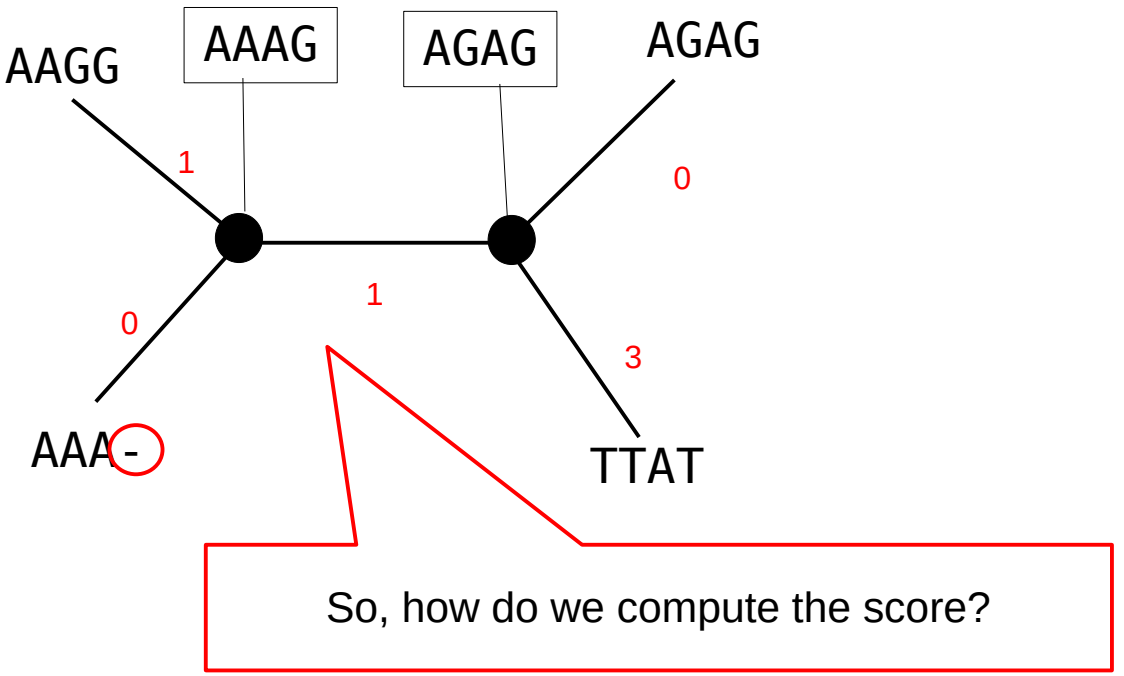
Parsimony

Parsimony Score of this tree = 5

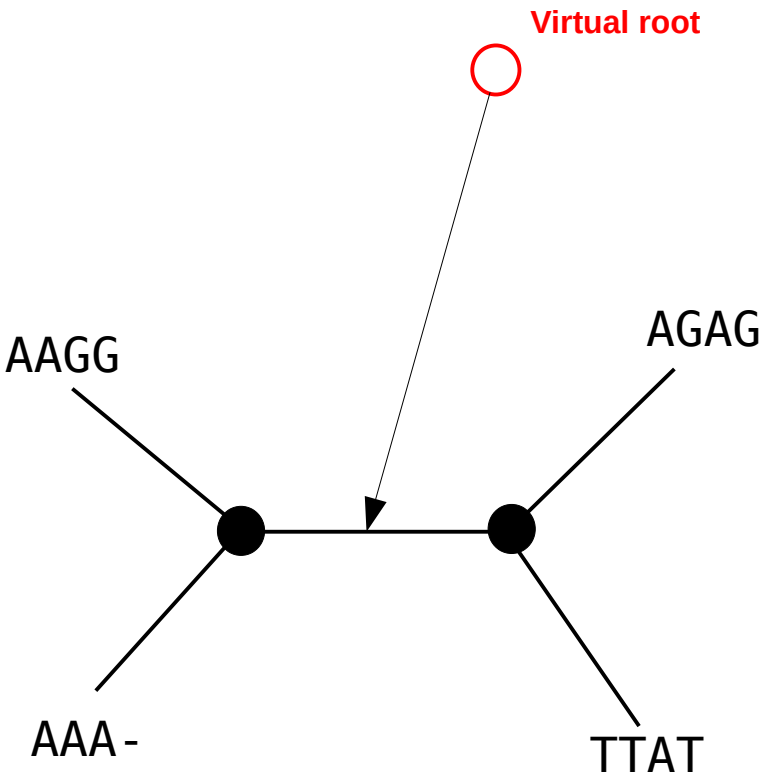


Gaps (also called Indels → Insertions or Deletions) are treated as so-called undetermined characters, also frequently denoted as ***N*.**
The interpretation is that *N* could be either *A, C, G*, or *T*.

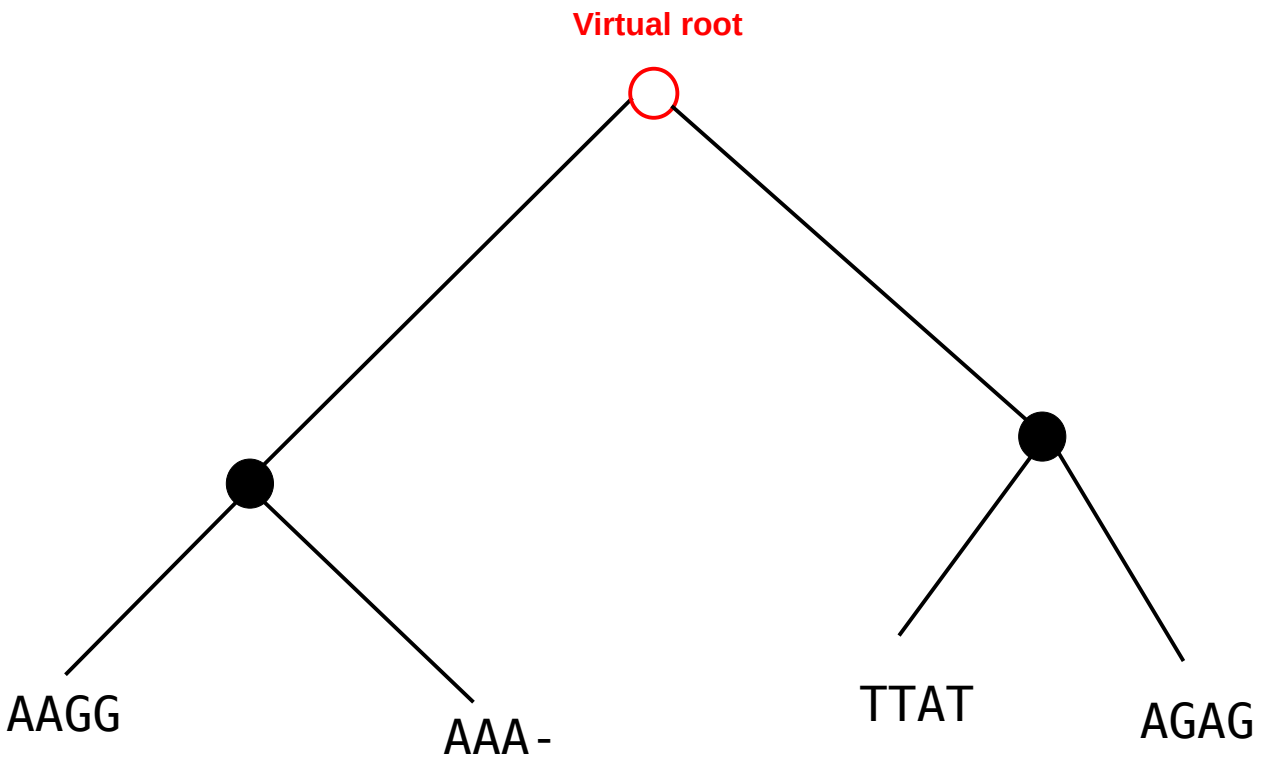
Parsimony


AAGG
AAAG
AGAG
AGAG
1
0
1
0
3
AAA-
TTAT
So, how do we compute the score?

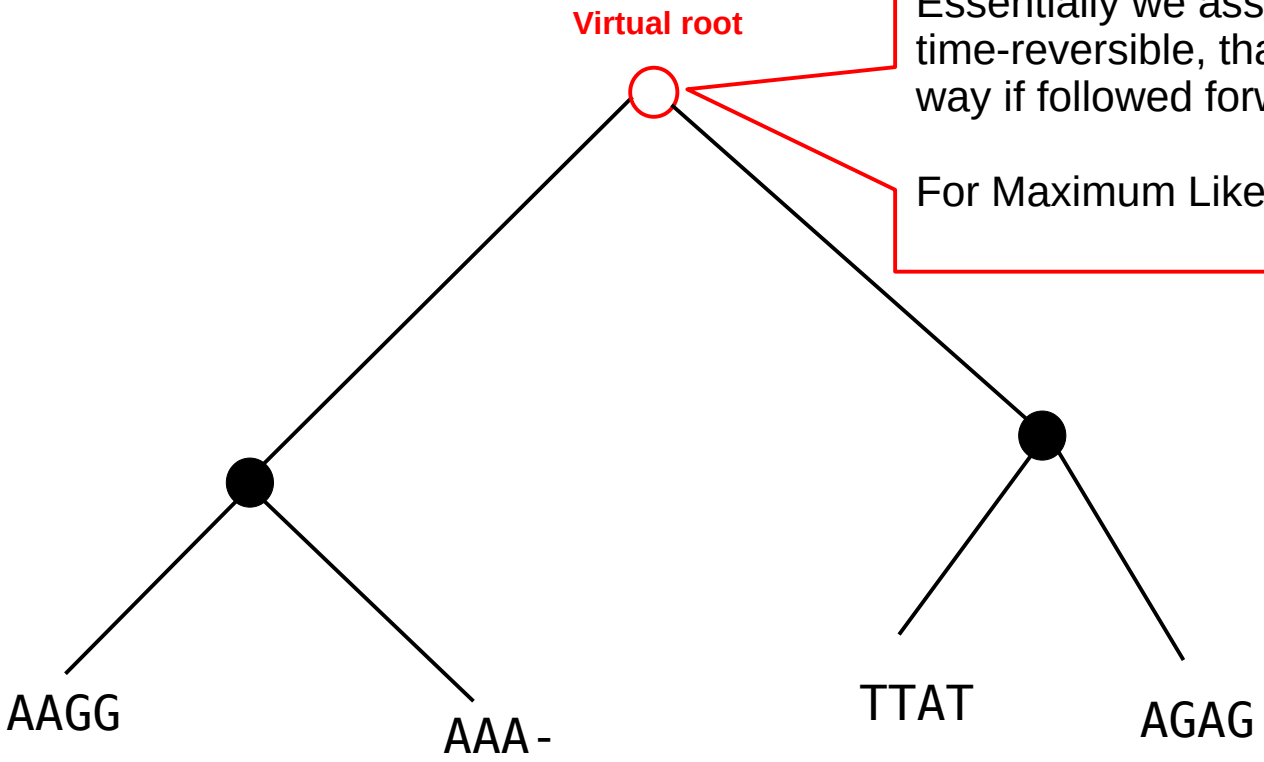
Parsimony



Parsimony



Parsimony



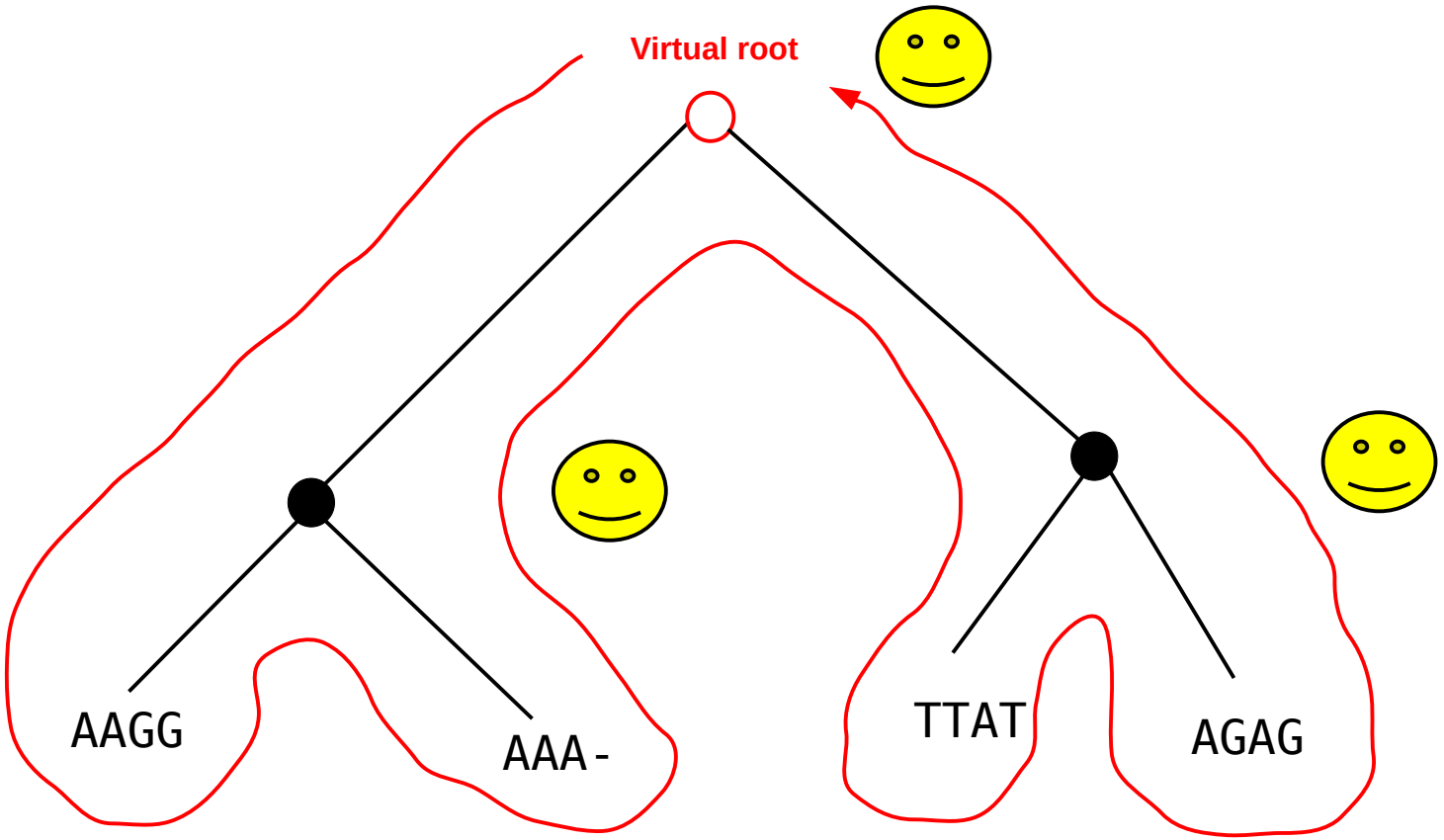
The score will be the same, regardless of where we place the root
→ makes computations much easier

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!

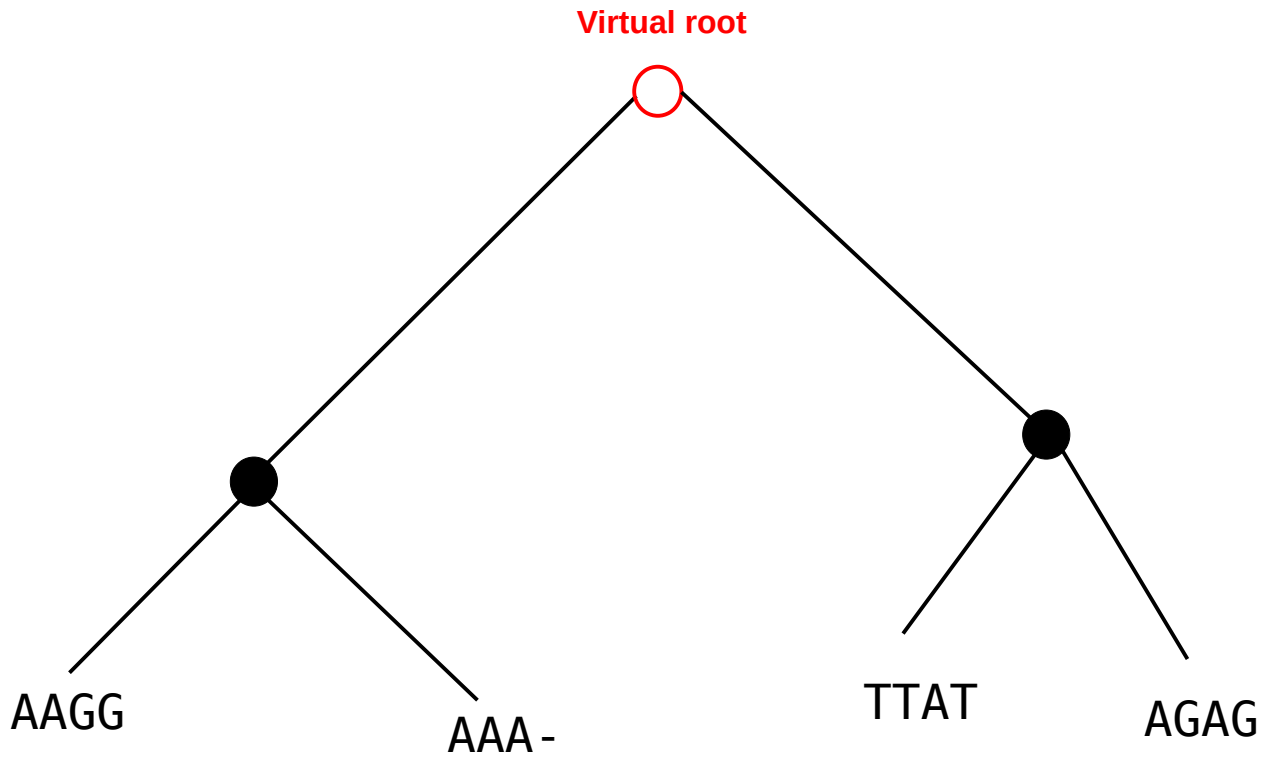
Parsimony

Post-order traversal to compute inner states

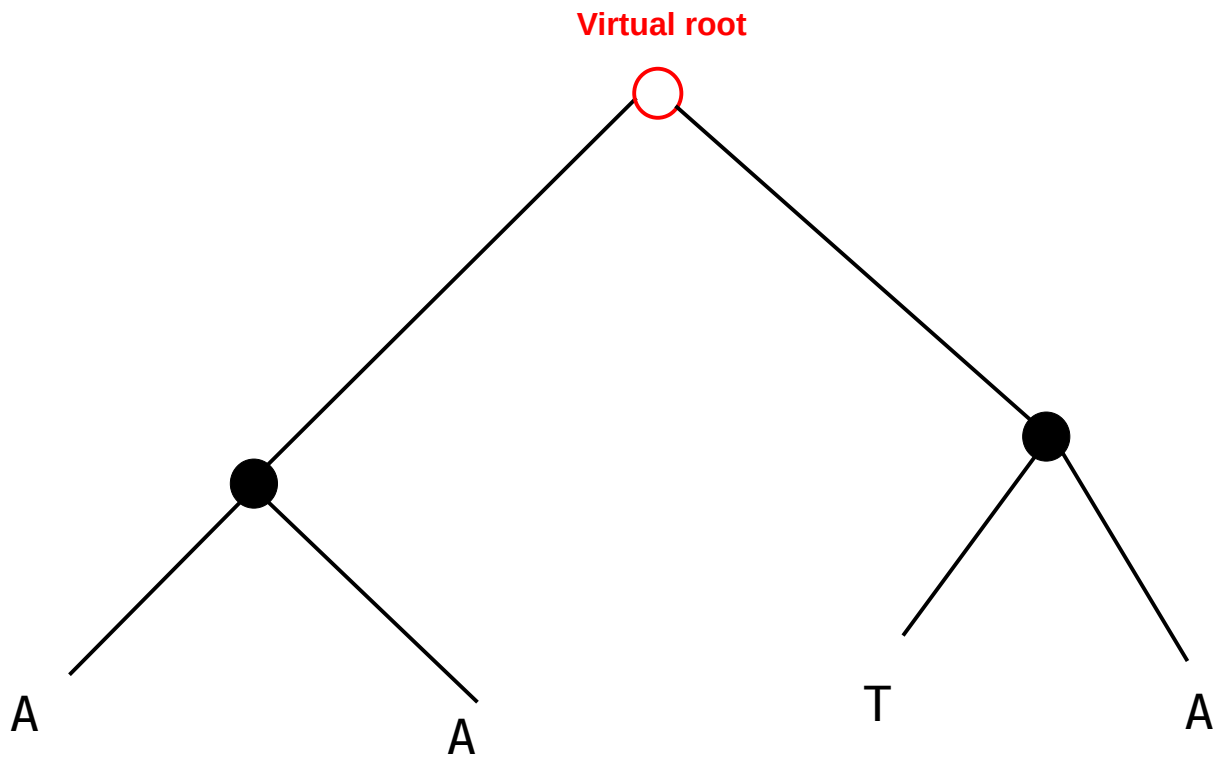


Parsimony

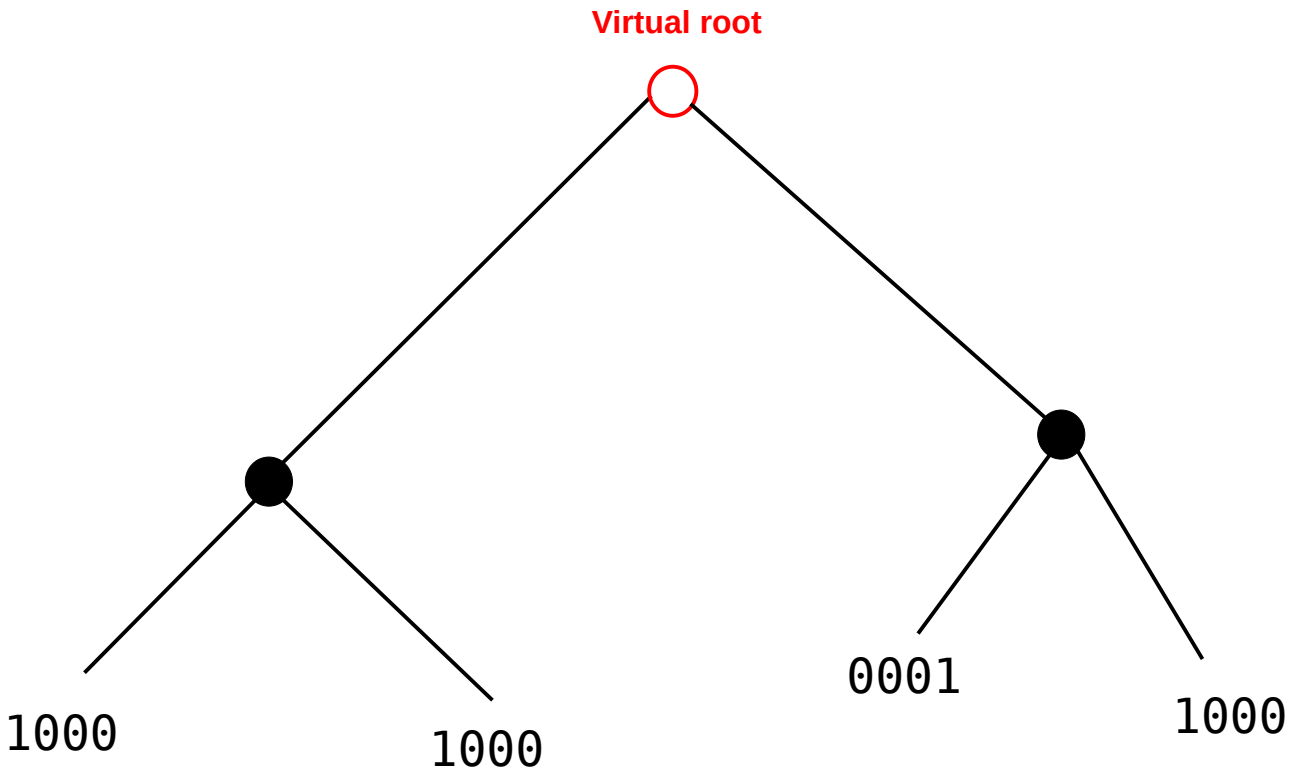
Compute scores on a site-per-site basis
→ we assume that sites evolve independently!



Parsimony

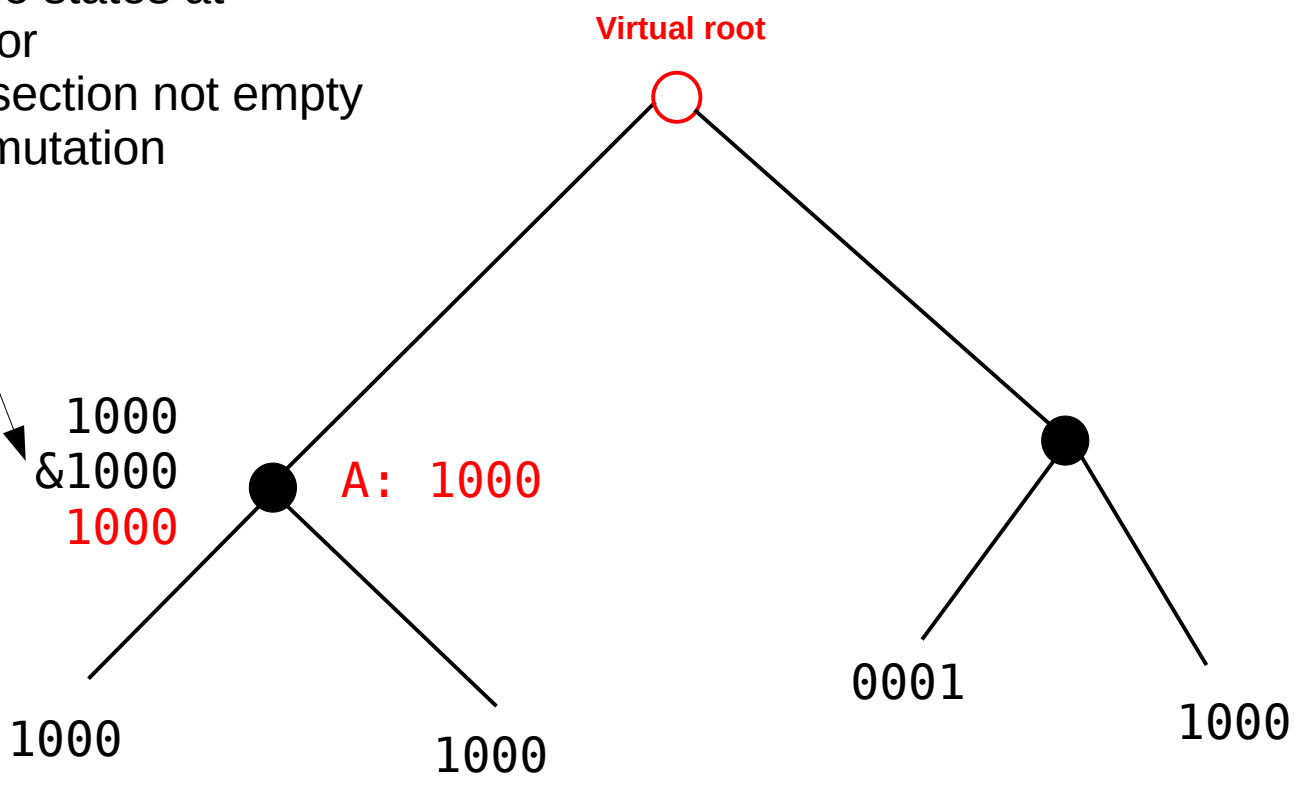


Parsimony

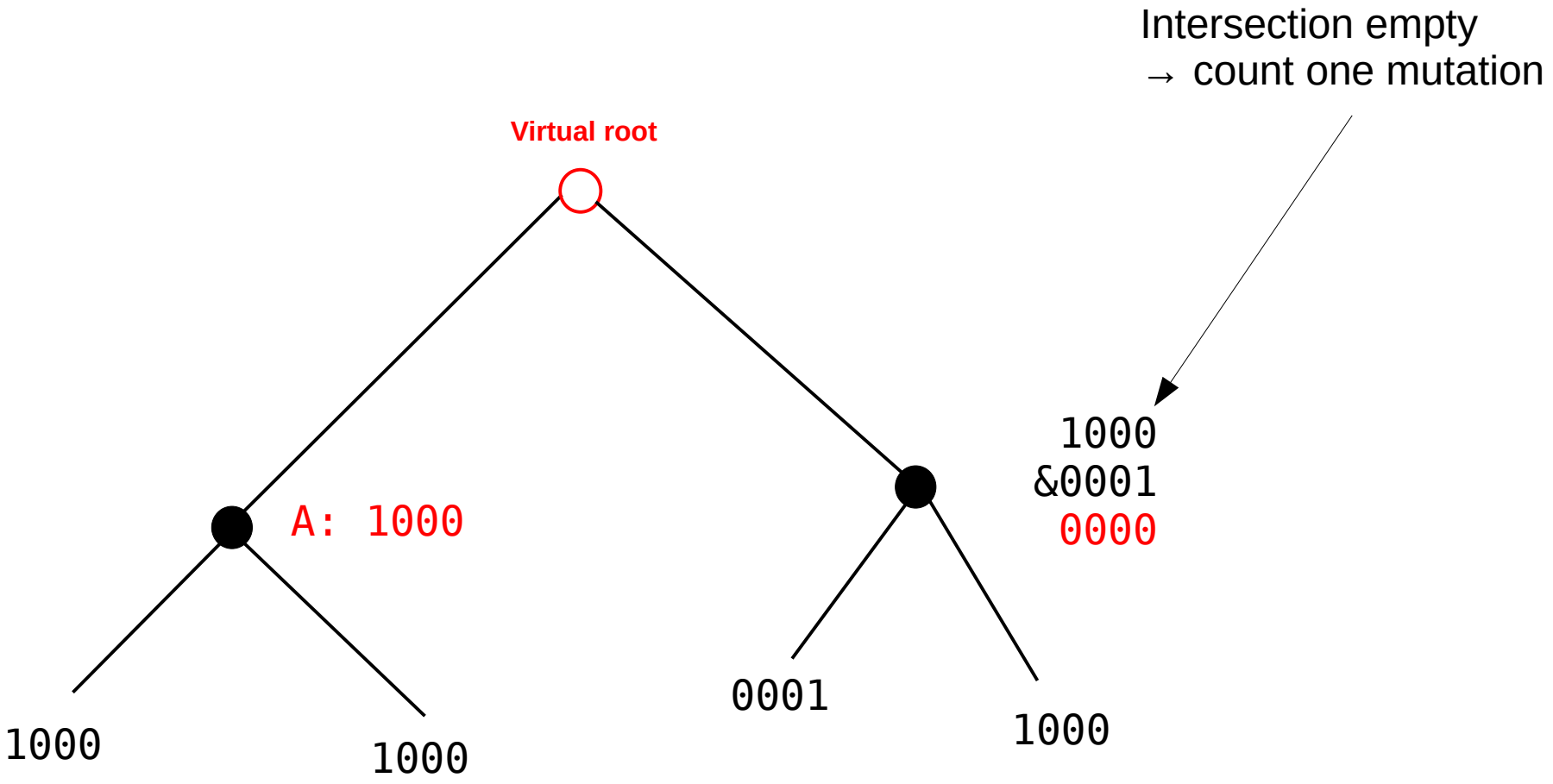


Parsimony

Intersection of sets of possible states at ancestor
If intersection not empty
→ no mutation

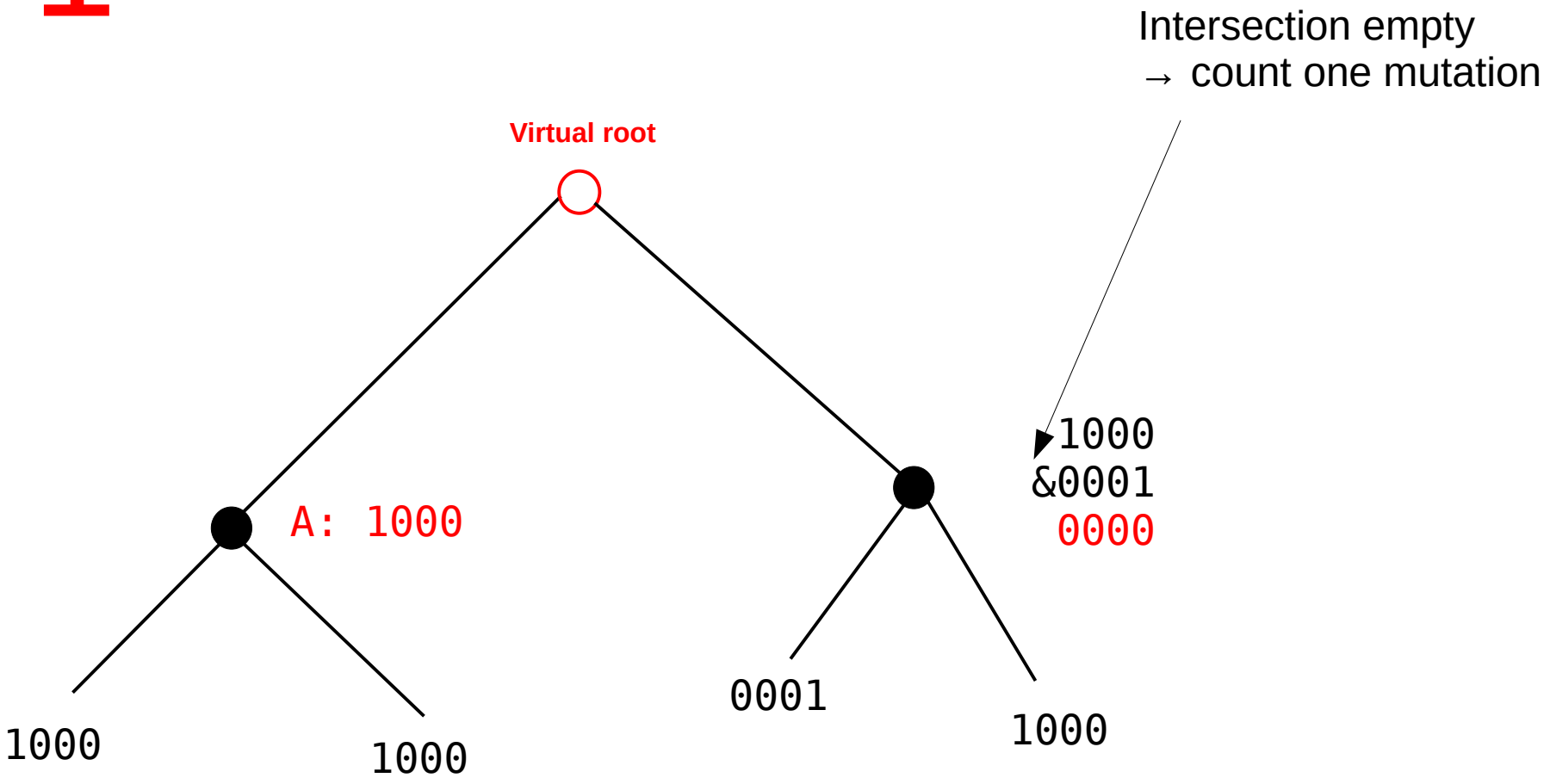


Parsimony


Intersection empty
→ count one mutation
Virtual root
A: 1000
1000
1000
0001
1000
1000
&0001
0000

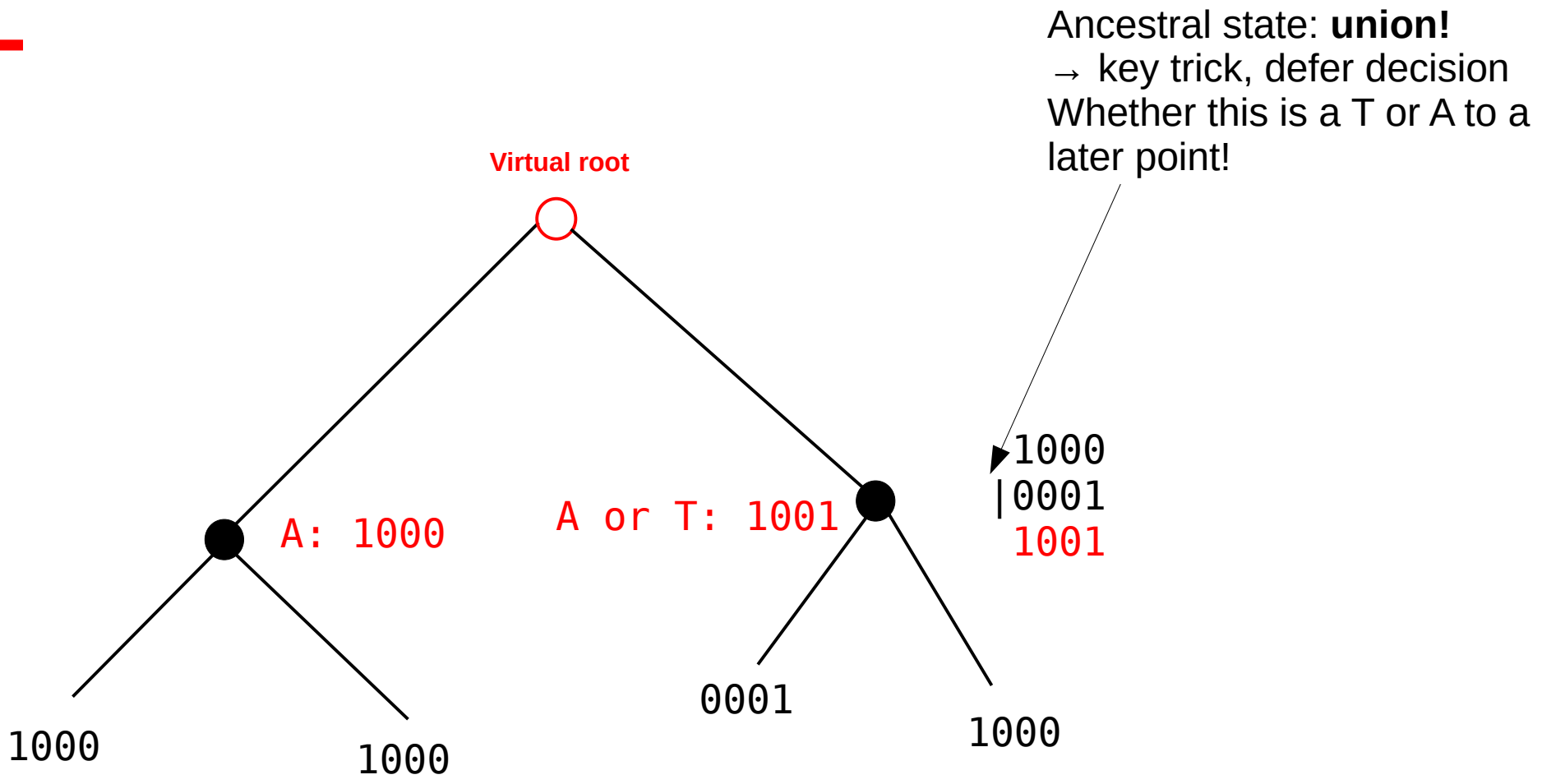
Parsimony

+1


Intersection empty
→ count one mutation
Virtual root
A: 1000
1000
&0001
0000
1000
1000
0001
1000

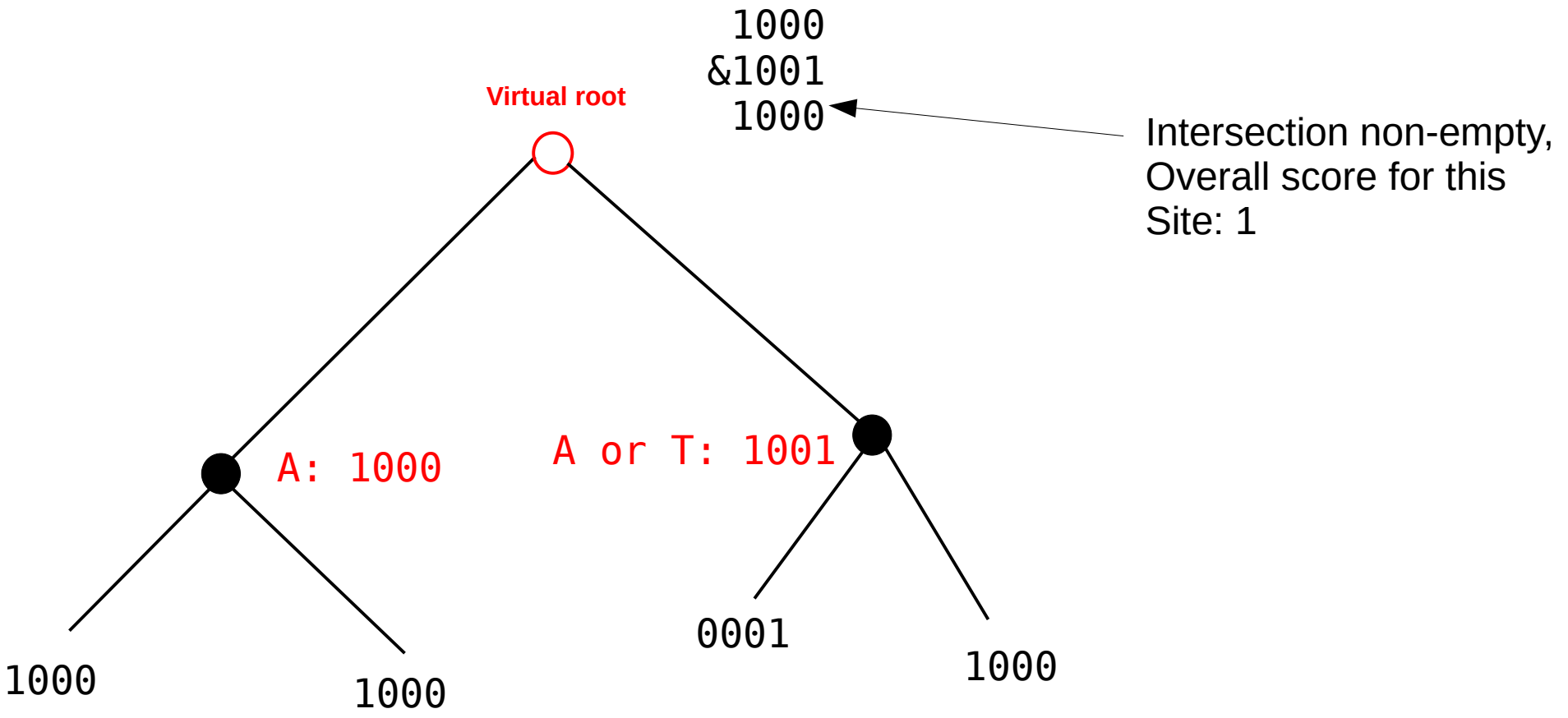
Parsimony

1


Ancestral state: **union!**
→ key trick, defer decision
Whether this is a T or A to a later point!
Virtual root
A: 1000
A or T: 1001
1000
|0001
1001
1000
1000
0001
1000

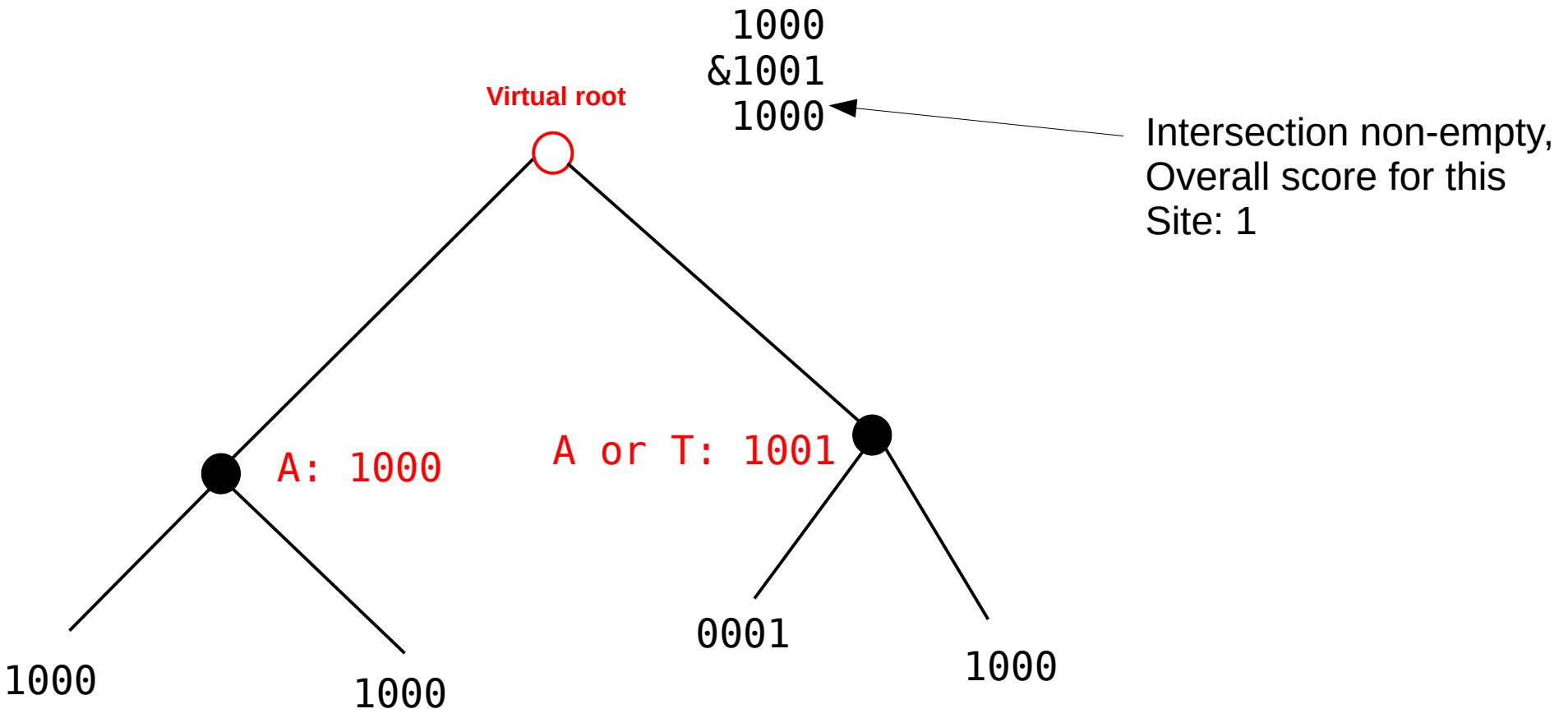
Parsimony

1


1000
&1001
1000
Virtual root
Intersection non-empty,
Overall score for this
Site: 1
A: 1000
A or T: 1001
1000
1000
0001
1000

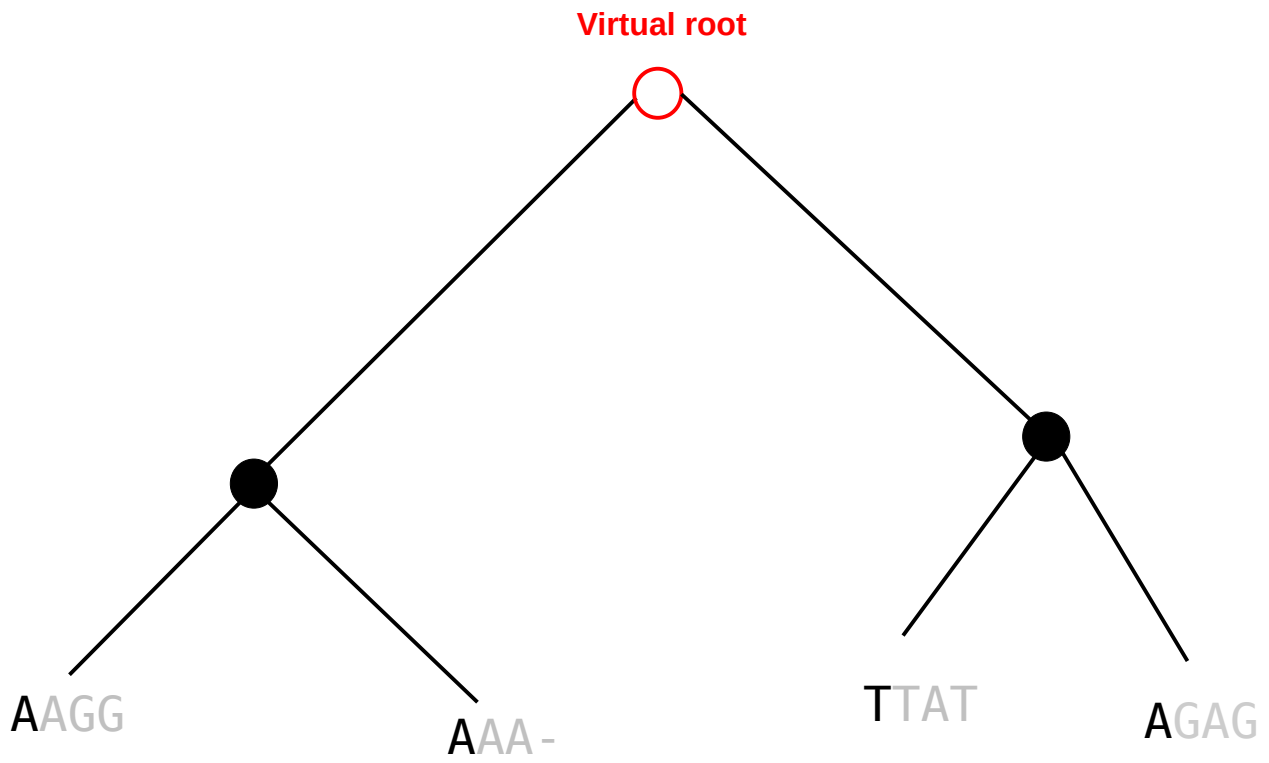
Parsimony

1


1000
&1001
1000
Virtual root
Intersection non-empty,
Overall score for this
Site: 1
A: 1000
A or T: 1001
1000
1000
0001
1000

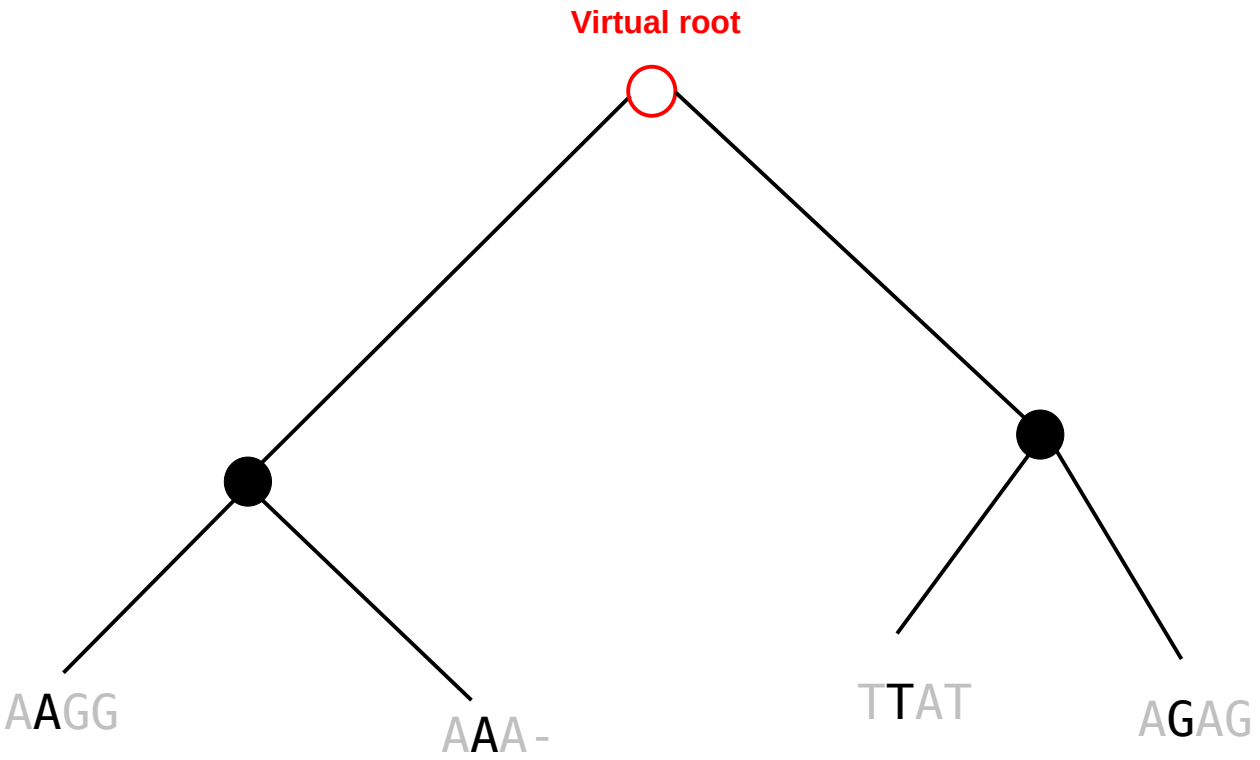
Parsimony

1



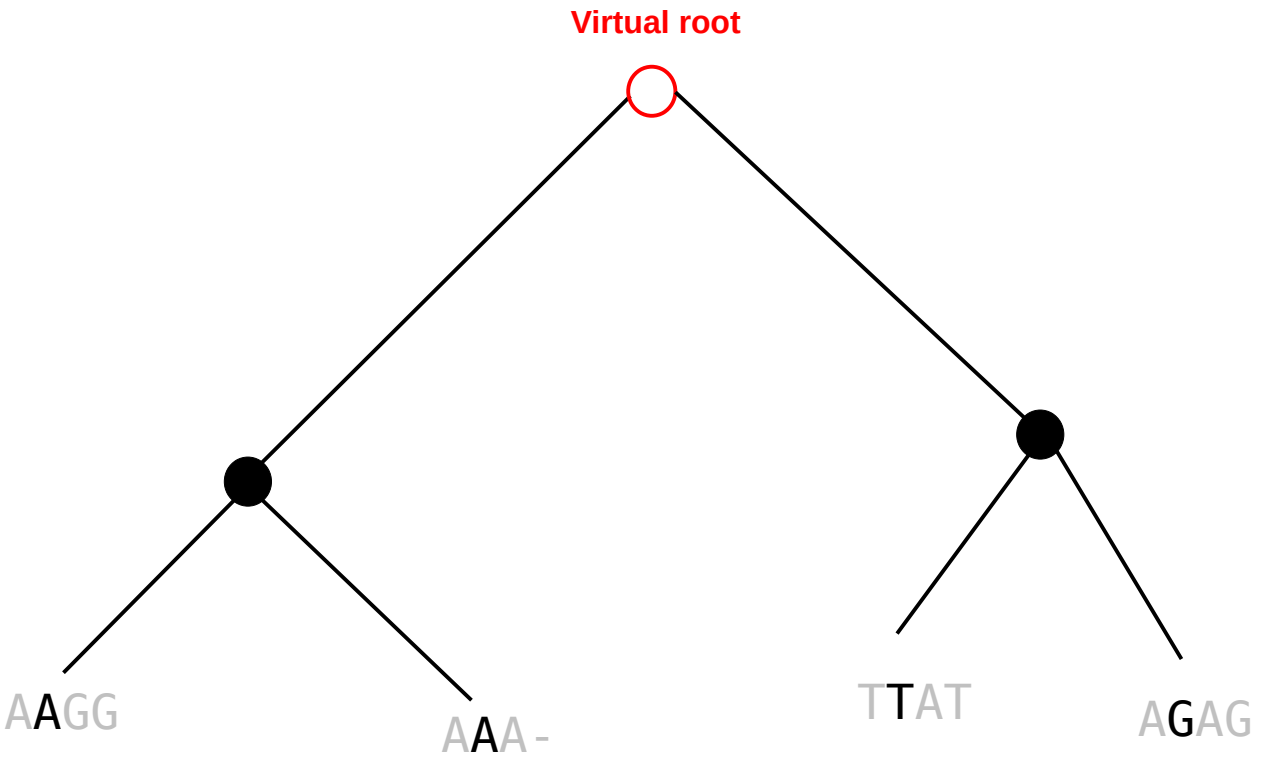
Parsimony

1+?



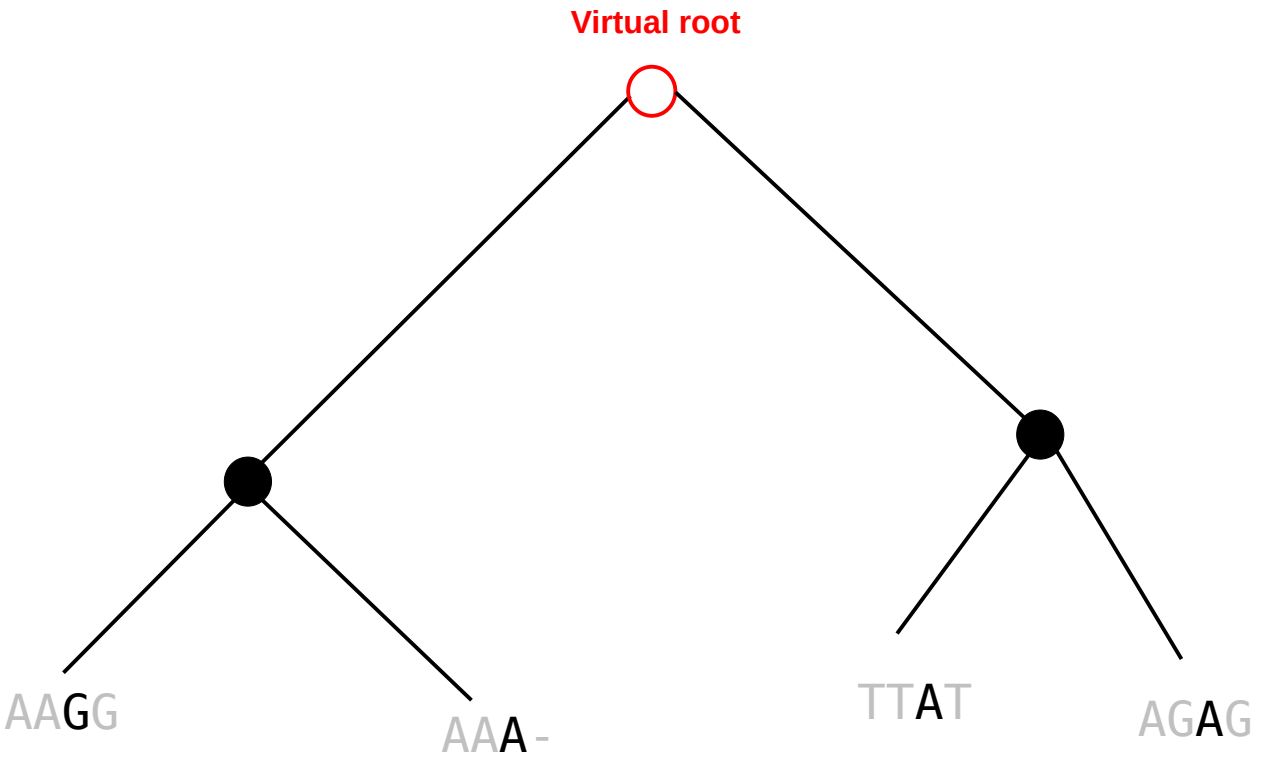
Parsimony

1+2



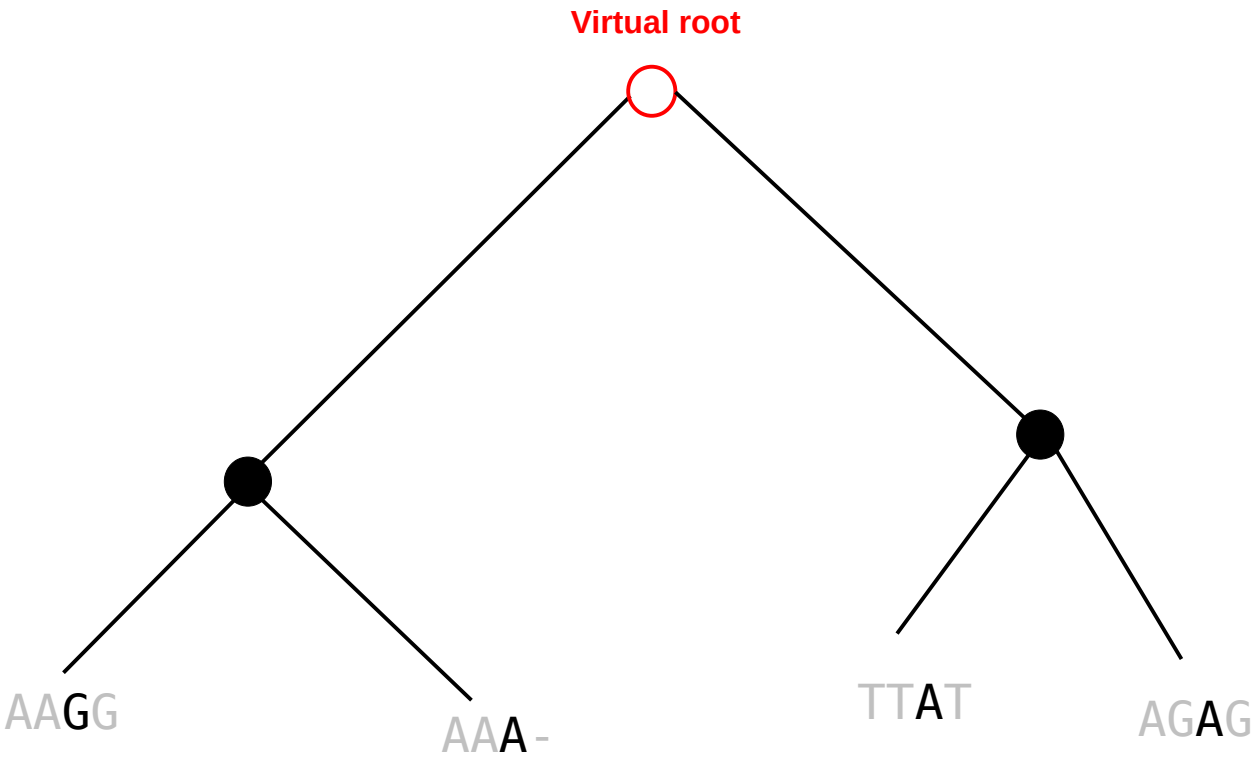
Parsimony

1+2+?



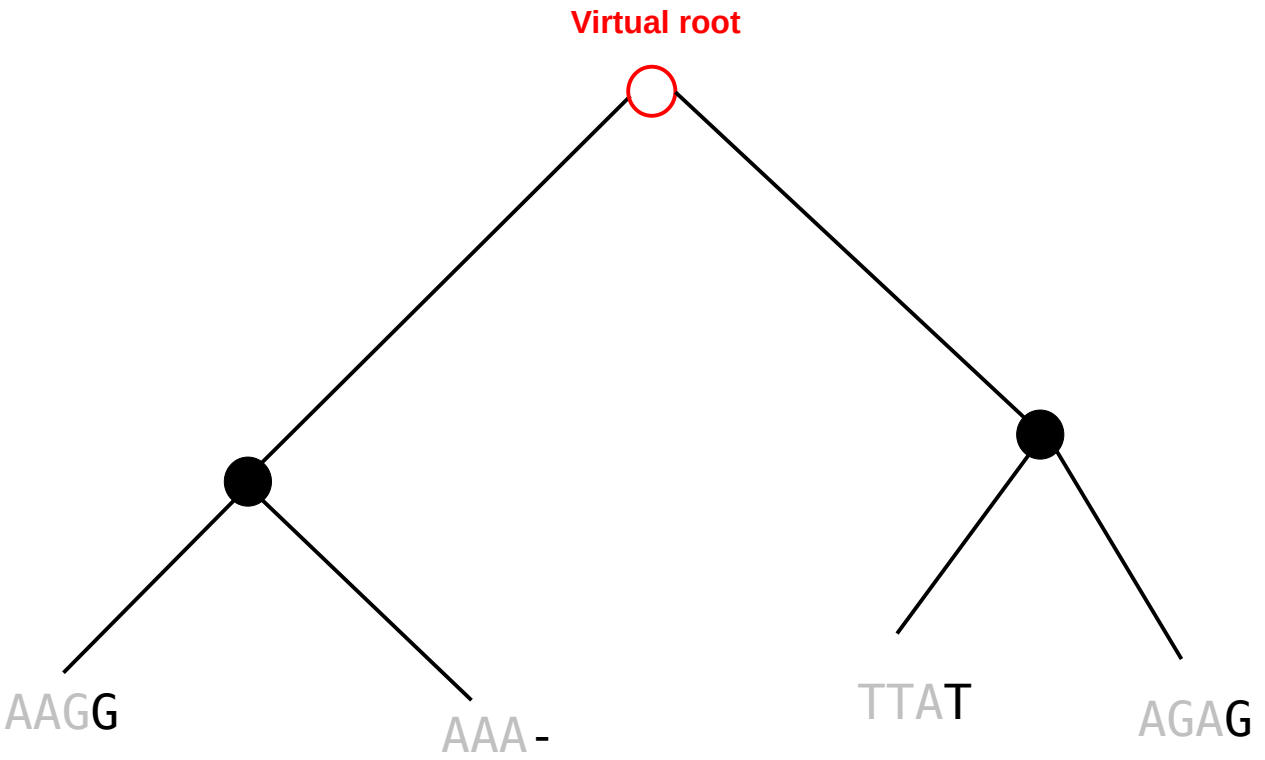
Parsimony

1+2+1



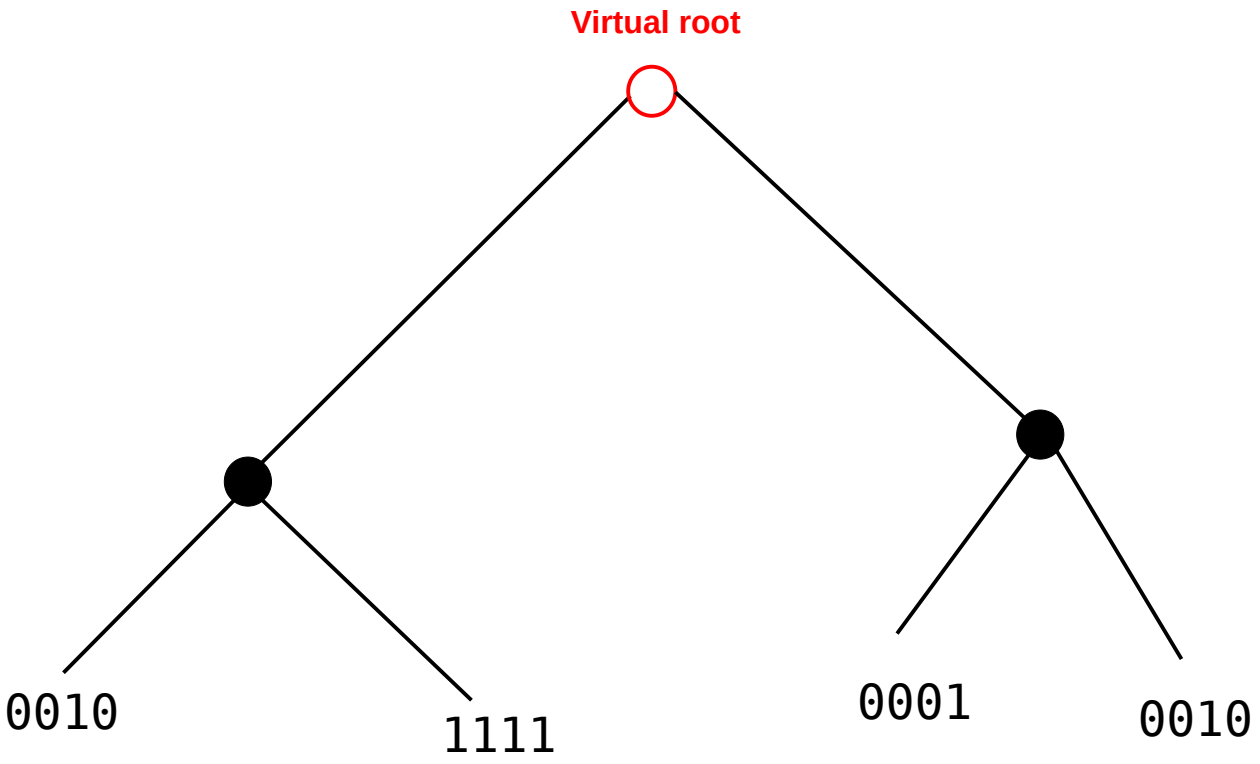
Parsimony

1+2+1



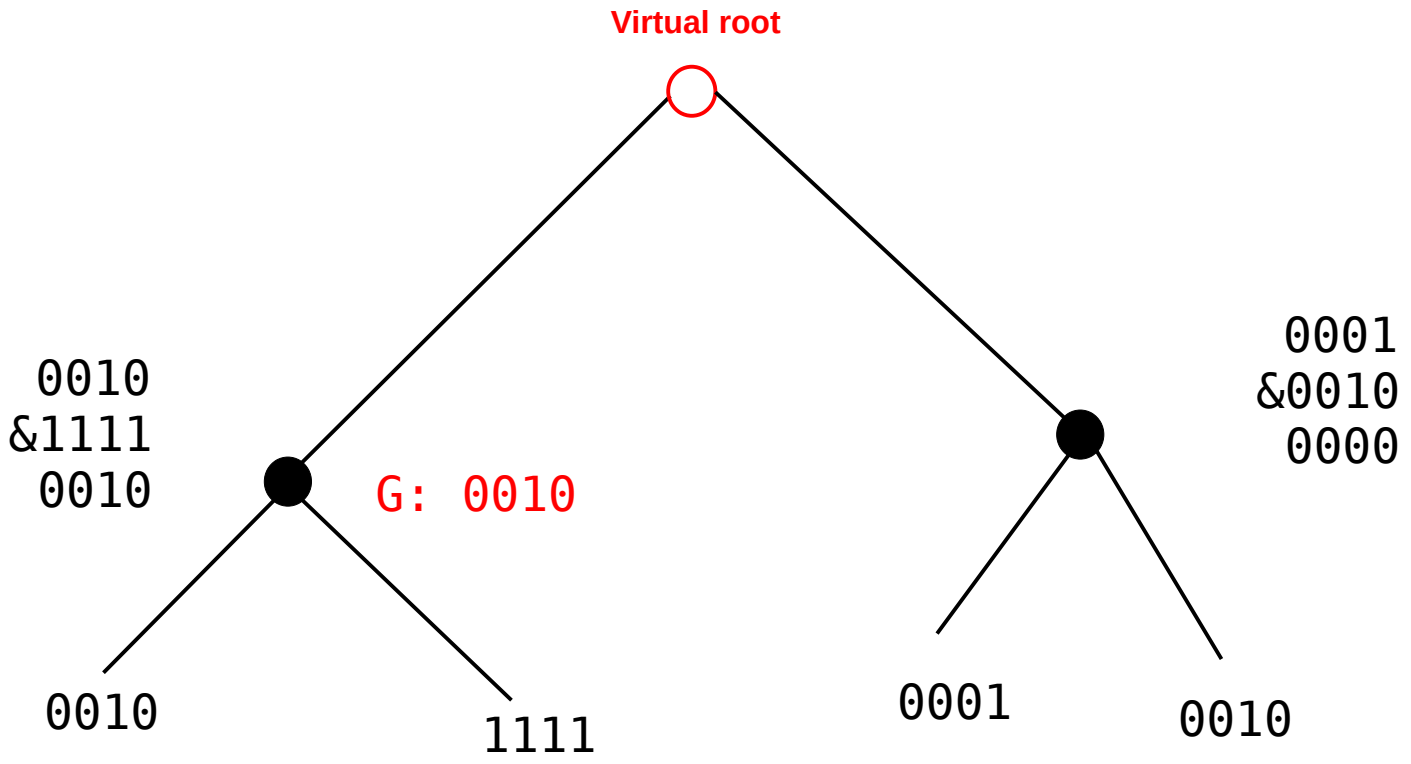
Parsimony

1+2+1



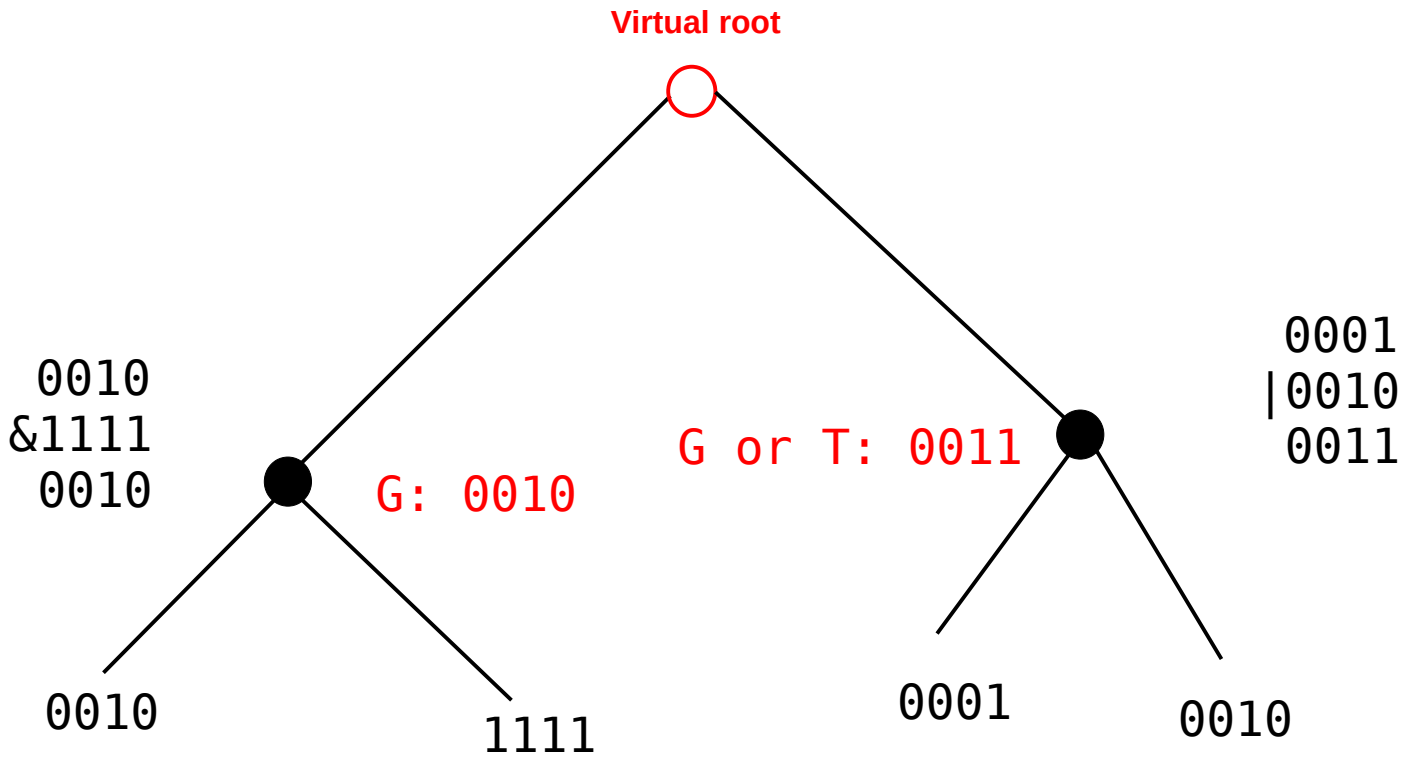
Parsimony

1+2+1



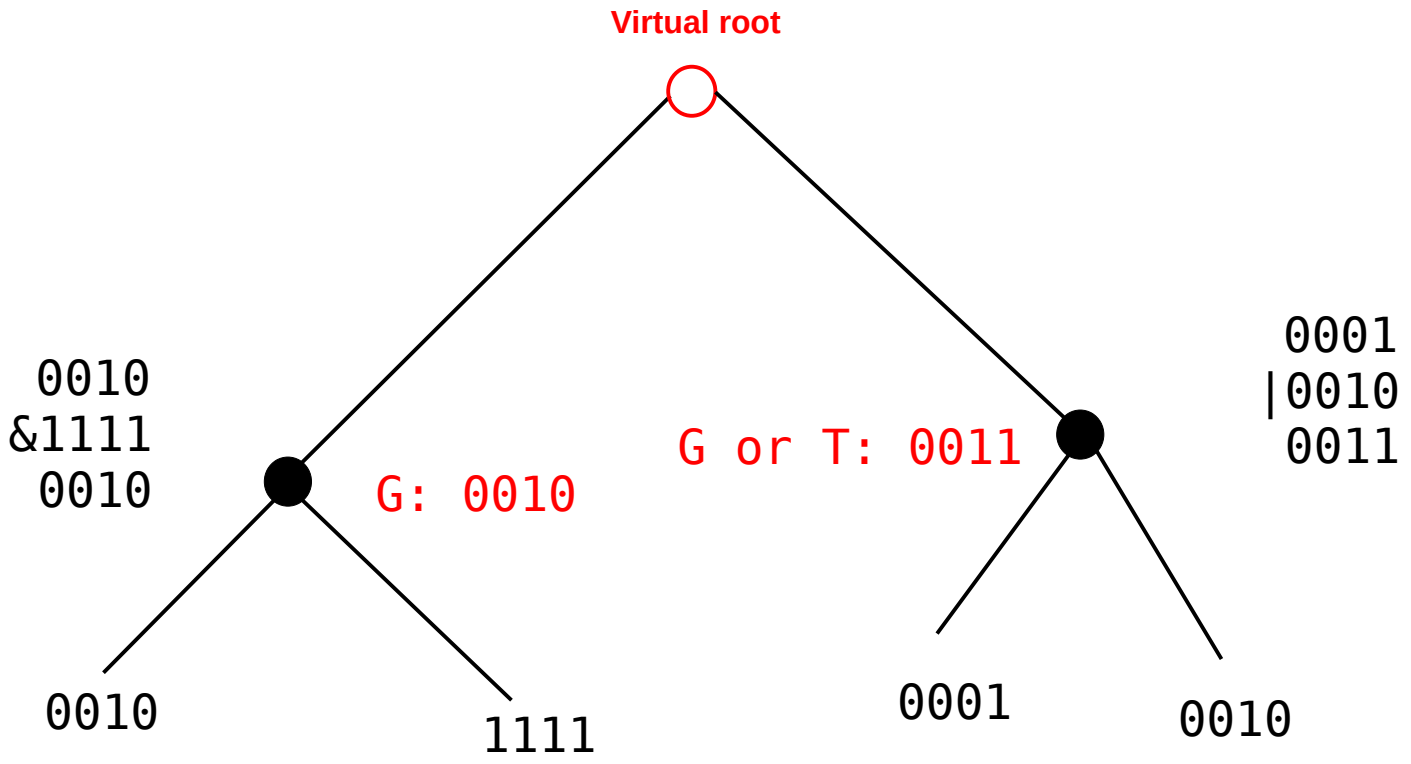
Parsimony

1+2+1+?



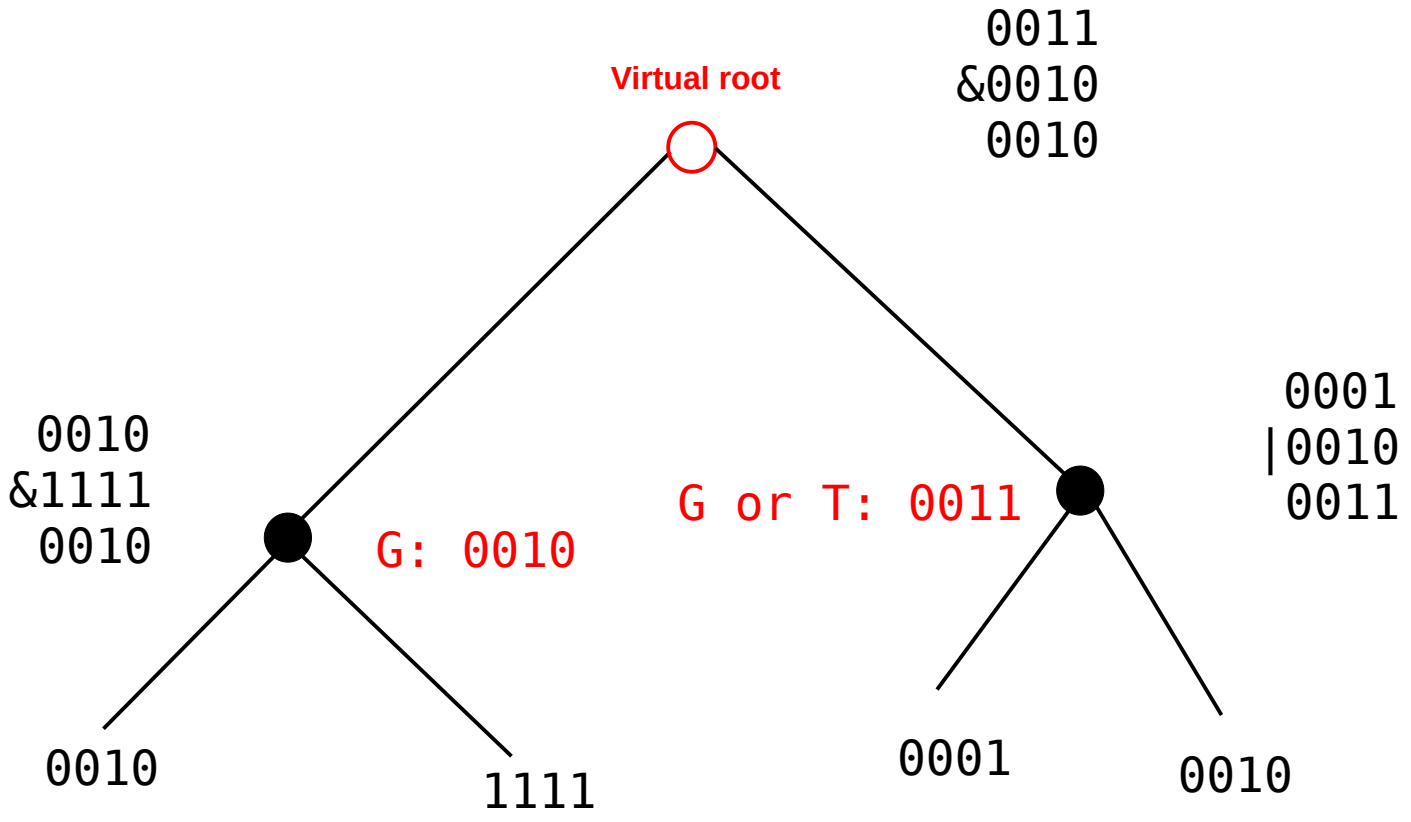
Parsimony

1+2+1+1



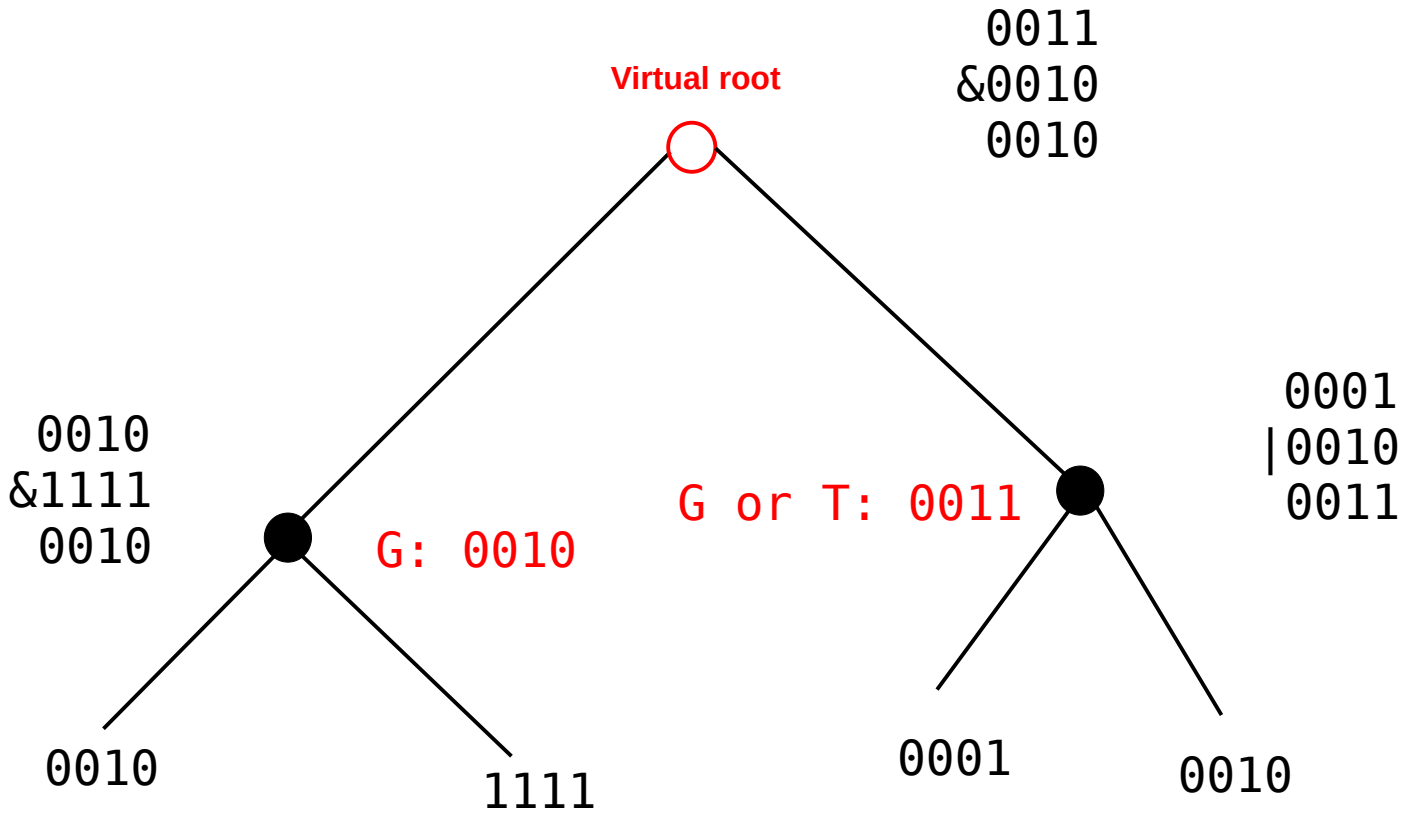
Parsimony

1+2+1+1

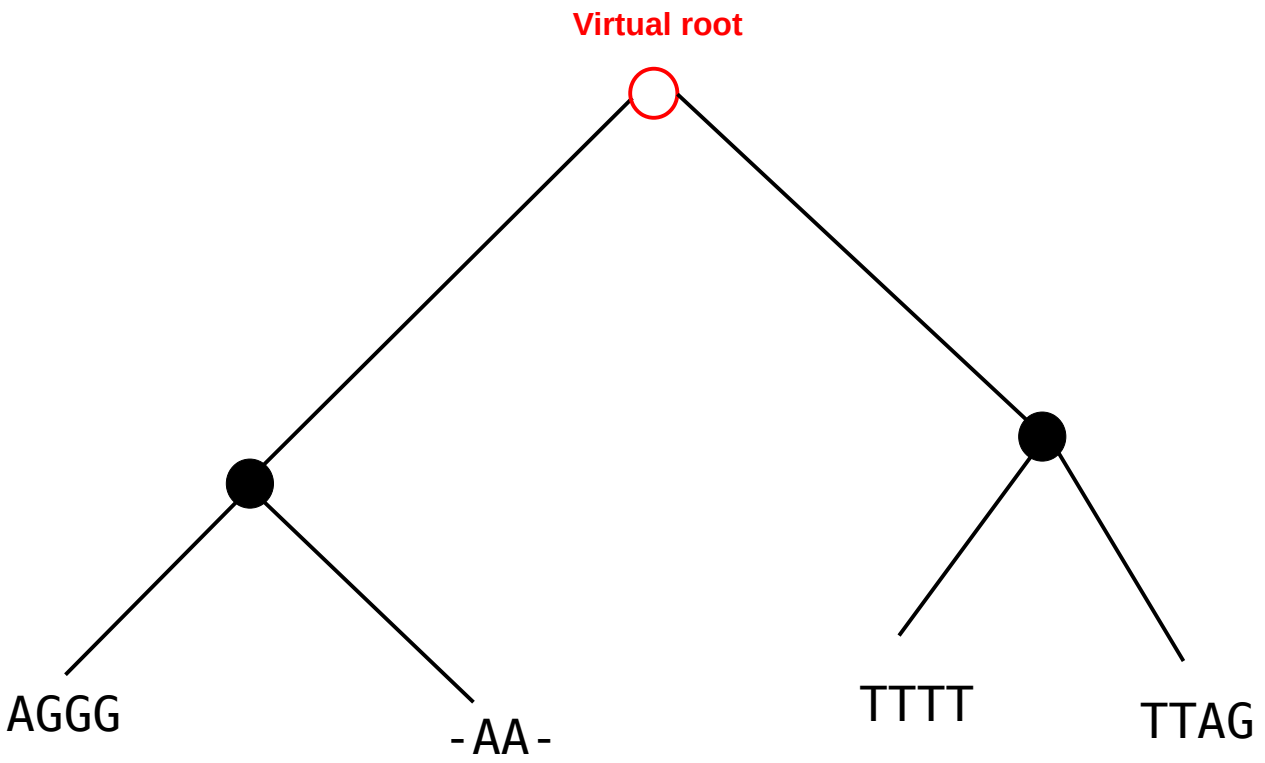


Parsimony

1+2+1+1= 5

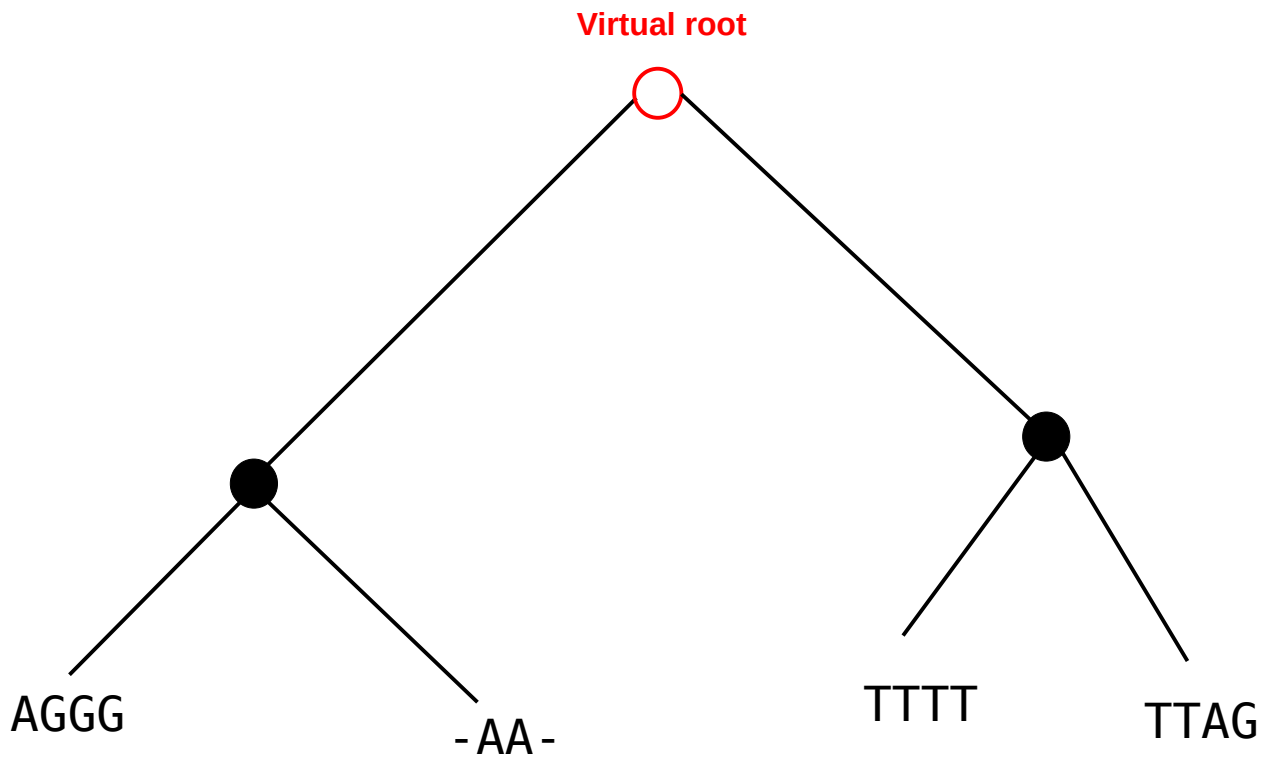


Exercise: What's the parsimony score of this tree?



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

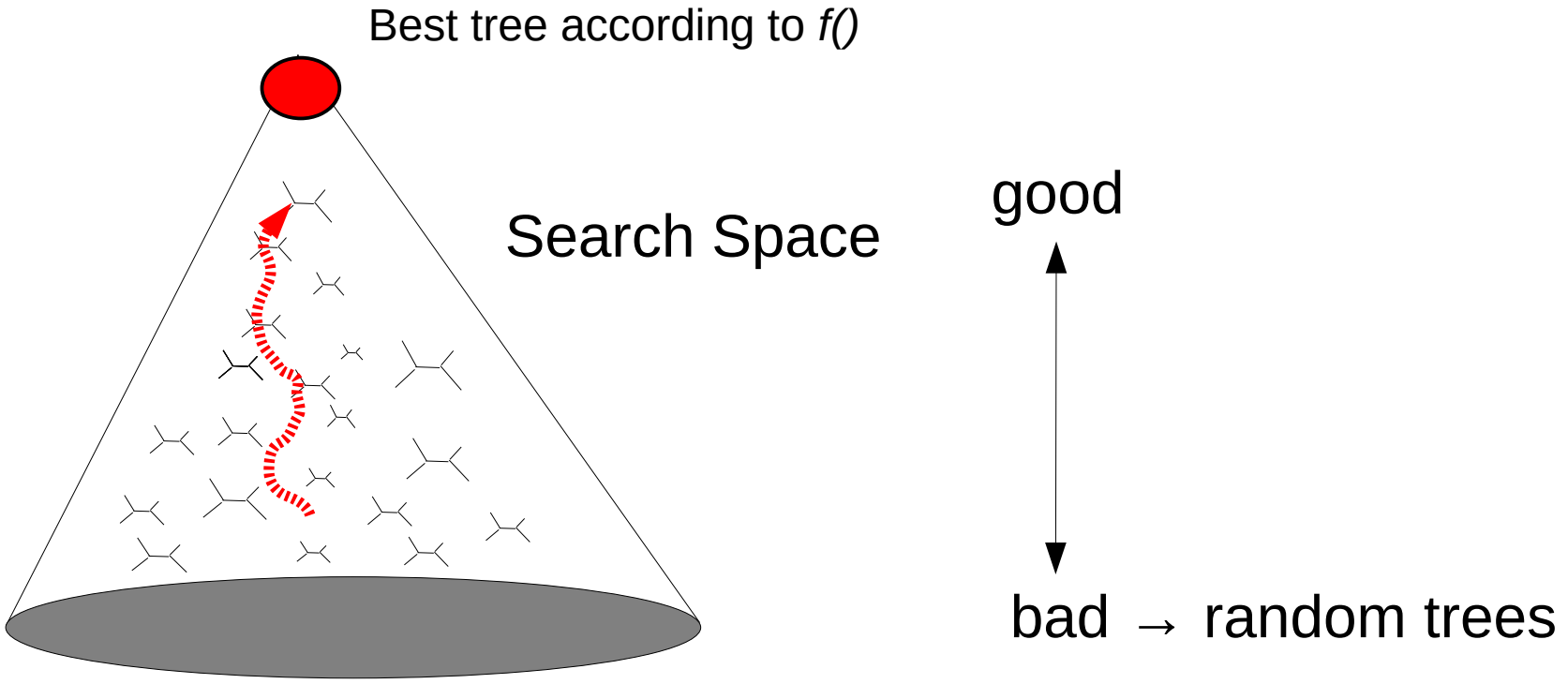
- Without going into the details:
- In the `parsimonator` implementation we need to compute a so-called population count (`popcount`) that computes the number of bits (# mutations) that are set to `1` in a 32-, 128-, or 256-bit word
- `popcount` is a very important operation
- There are various fancy 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



Tree Search Algorithms

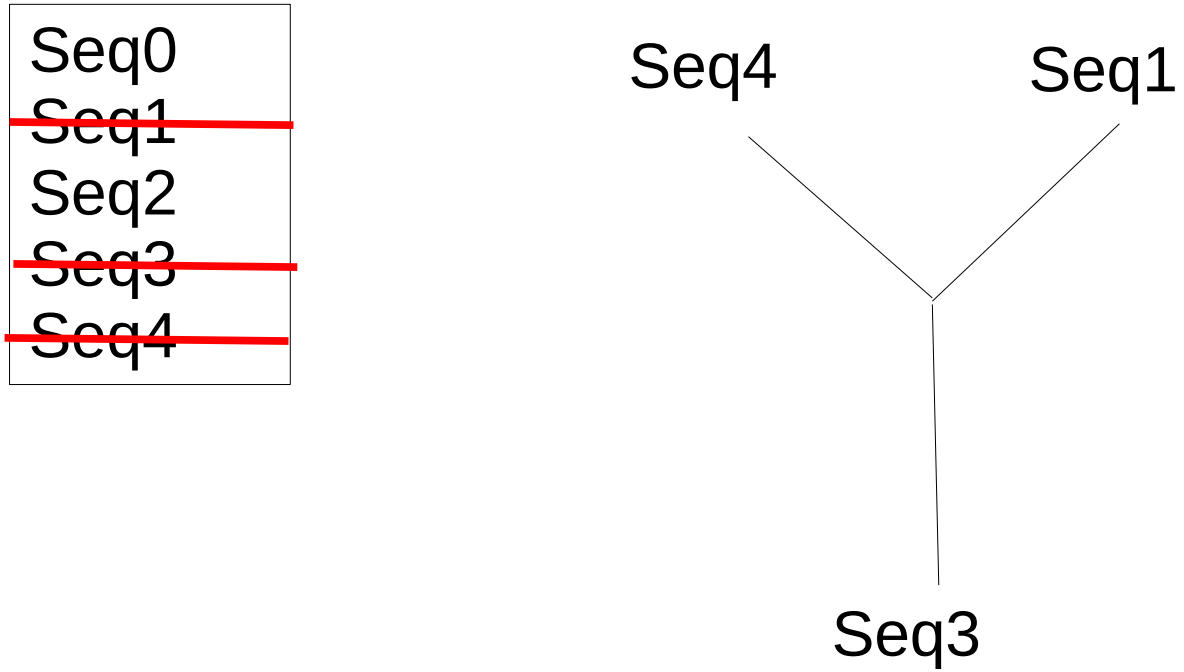
- How do we obtain an initial starting tree with *n* taxa → comprehensive tree
  - → NJ or UPGMA tree
  - → random tree
  - → stepwise addition algorithm
- How do we change such a comprehensive tree to improve its score?

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

Building a Random Tree

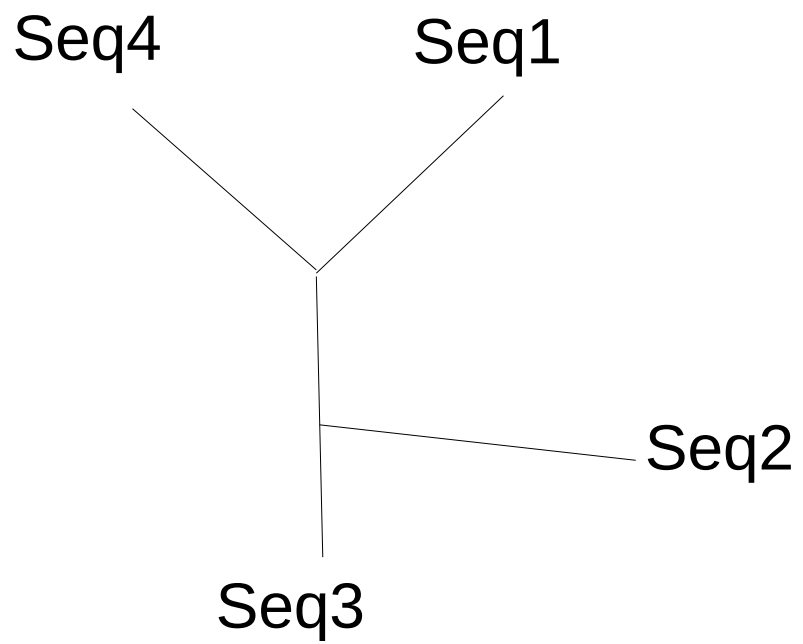
Seq0
Seq1
Seq2
Seq3
Seq4

Building a Random Tree

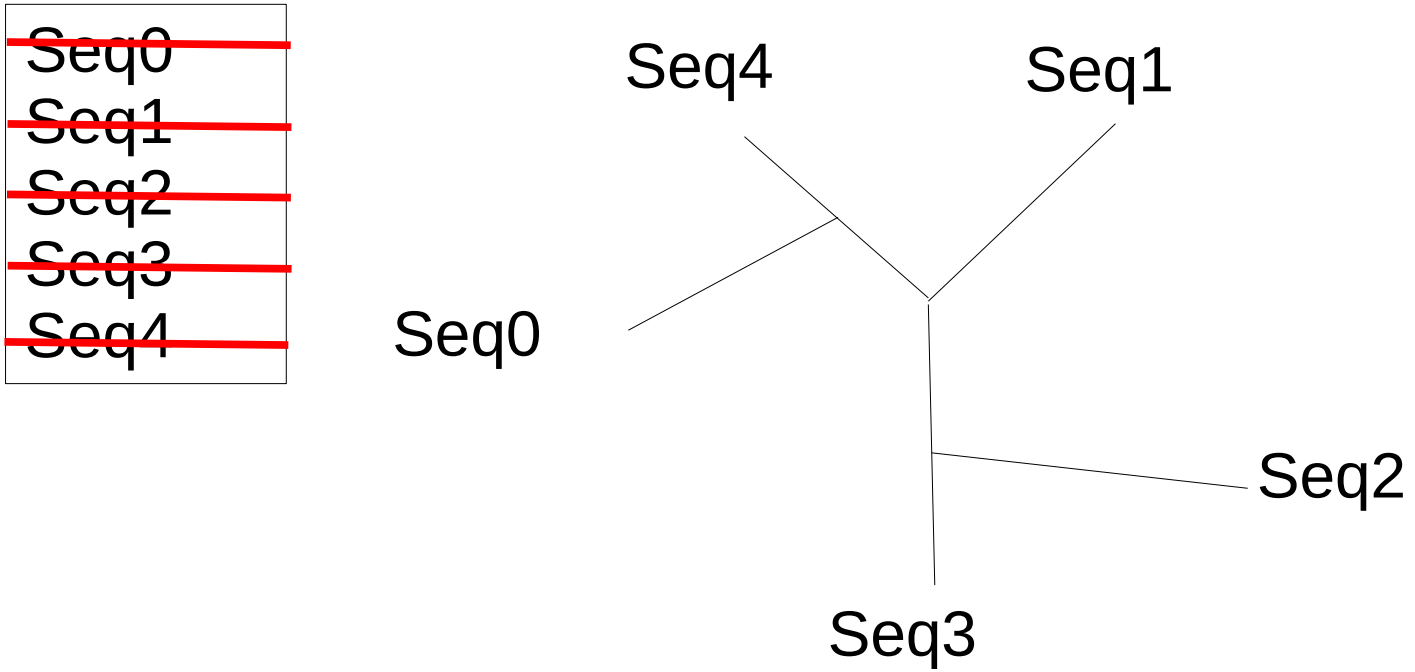


Building a Random Tree

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



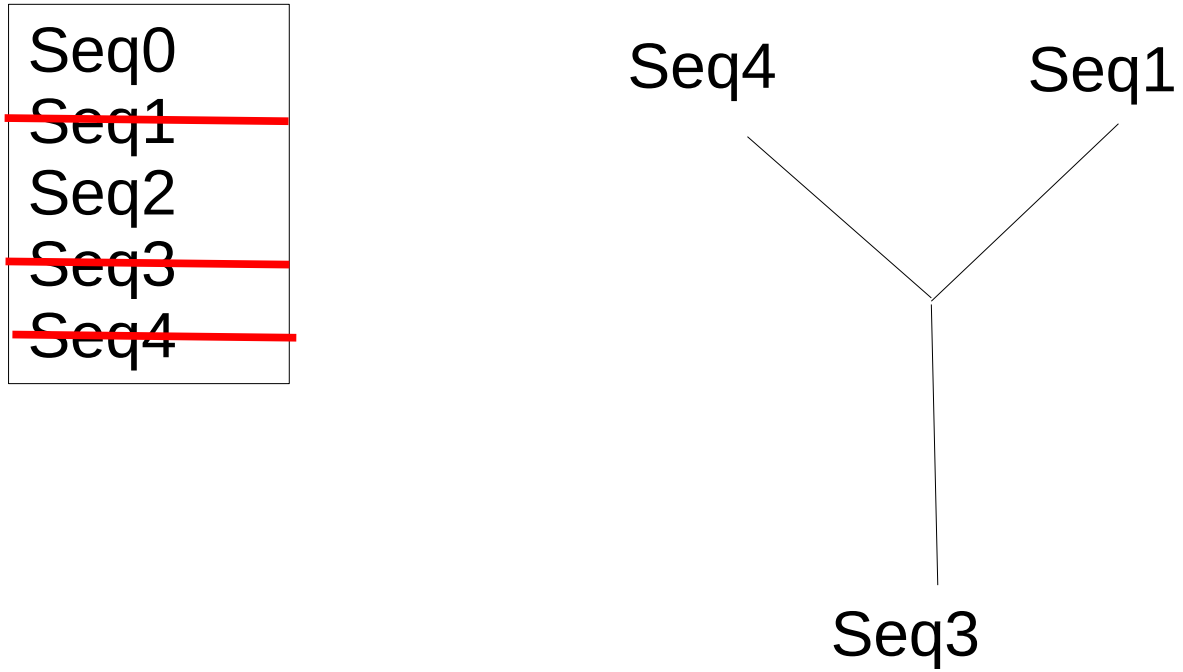
Building a Random Tree



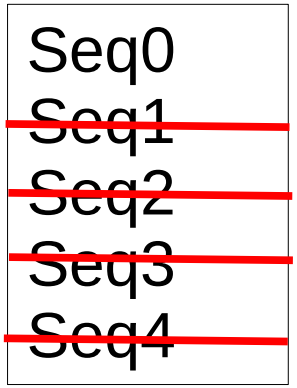
Randomized Stepwise Addition Order Algorithm

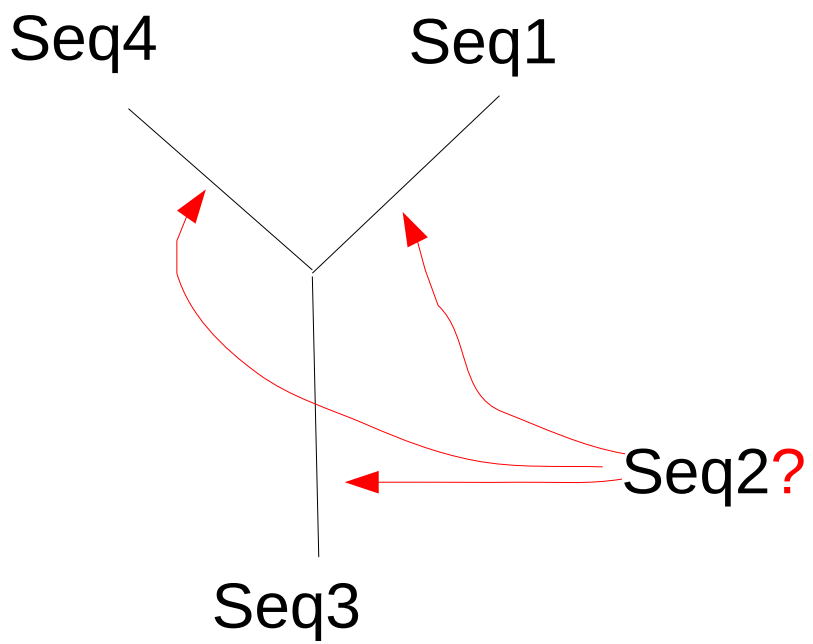
Seq0
Seq1
Seq2
Seq3
Seq4

Randmoized Stepwise Addition Order Algorithm

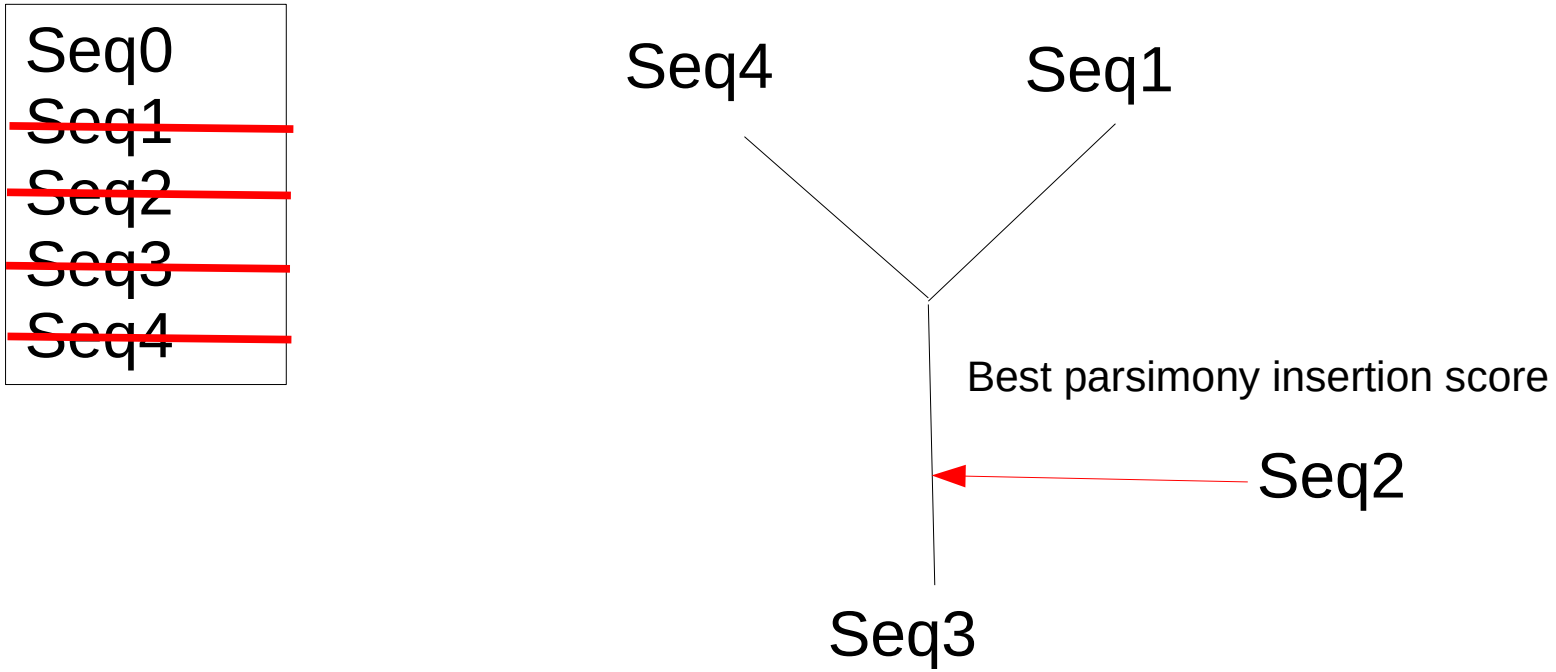


Randomized Stepwise Addition Order Algorithm



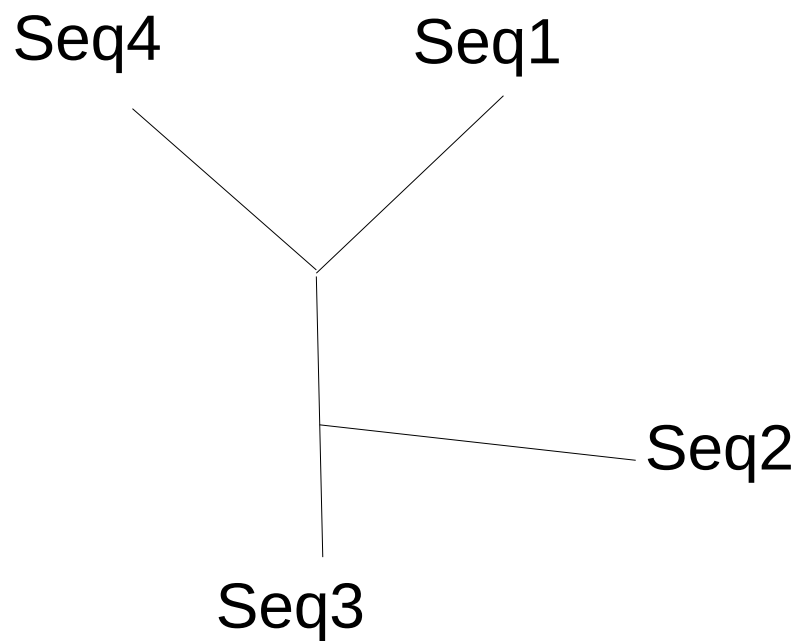


Randomized Stepwise Addition Order Algorithm

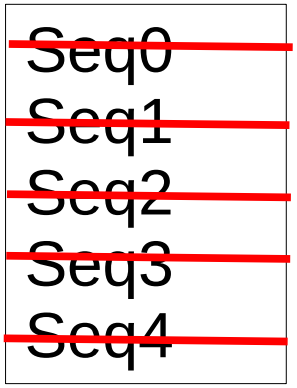


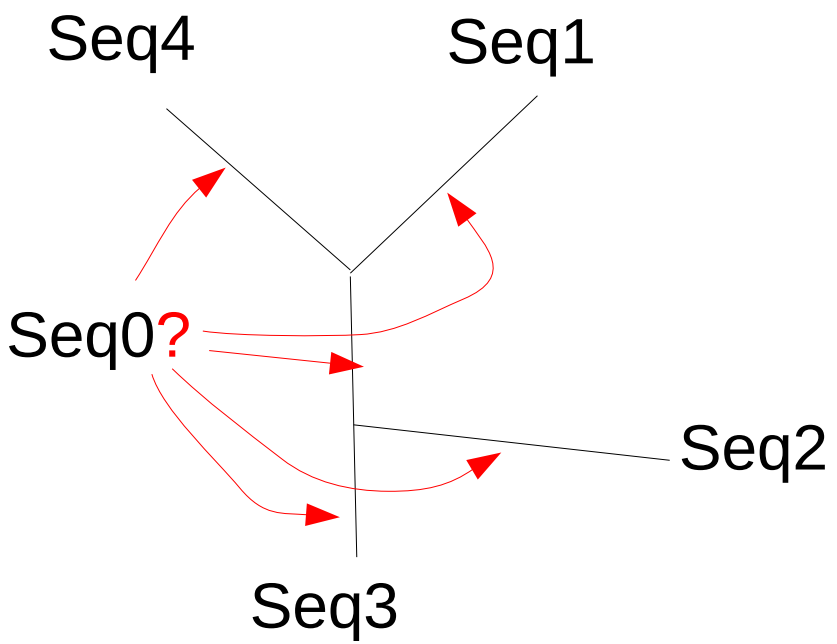
Randomized Stepwise Addition Order Algorithm

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

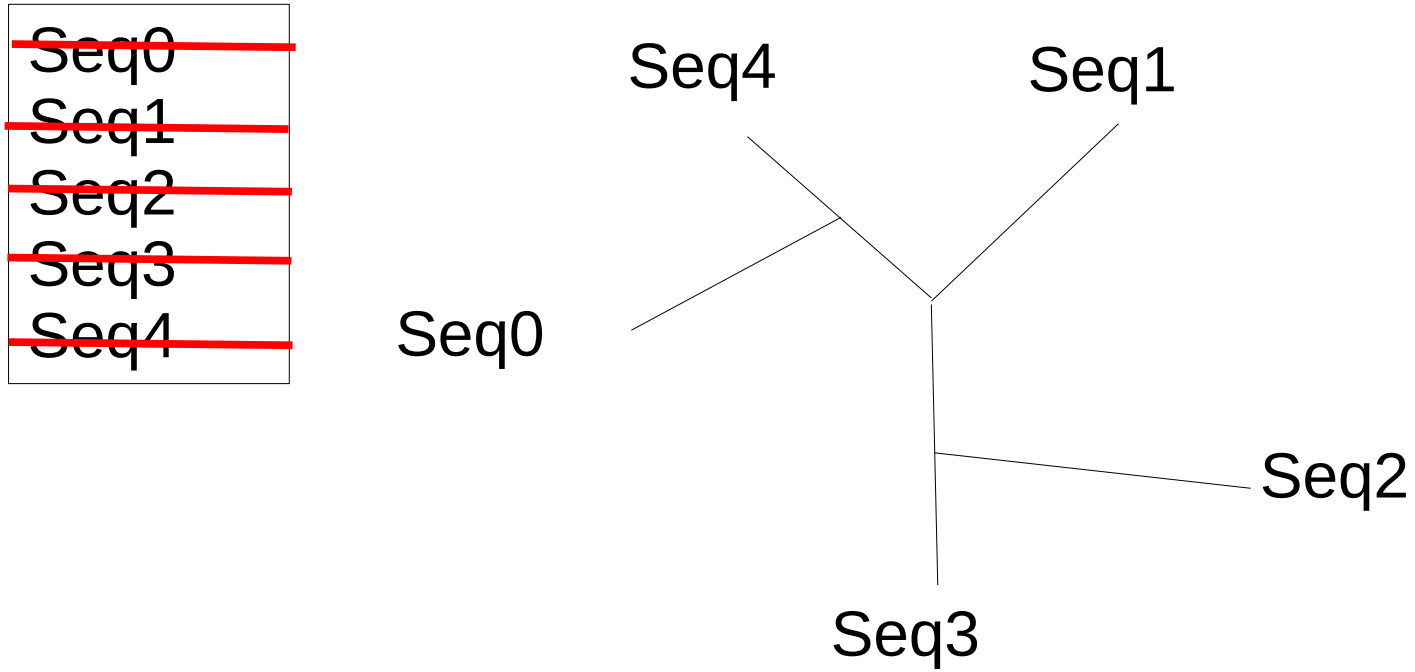


Randomized Stepwise Addition Order Algorithm

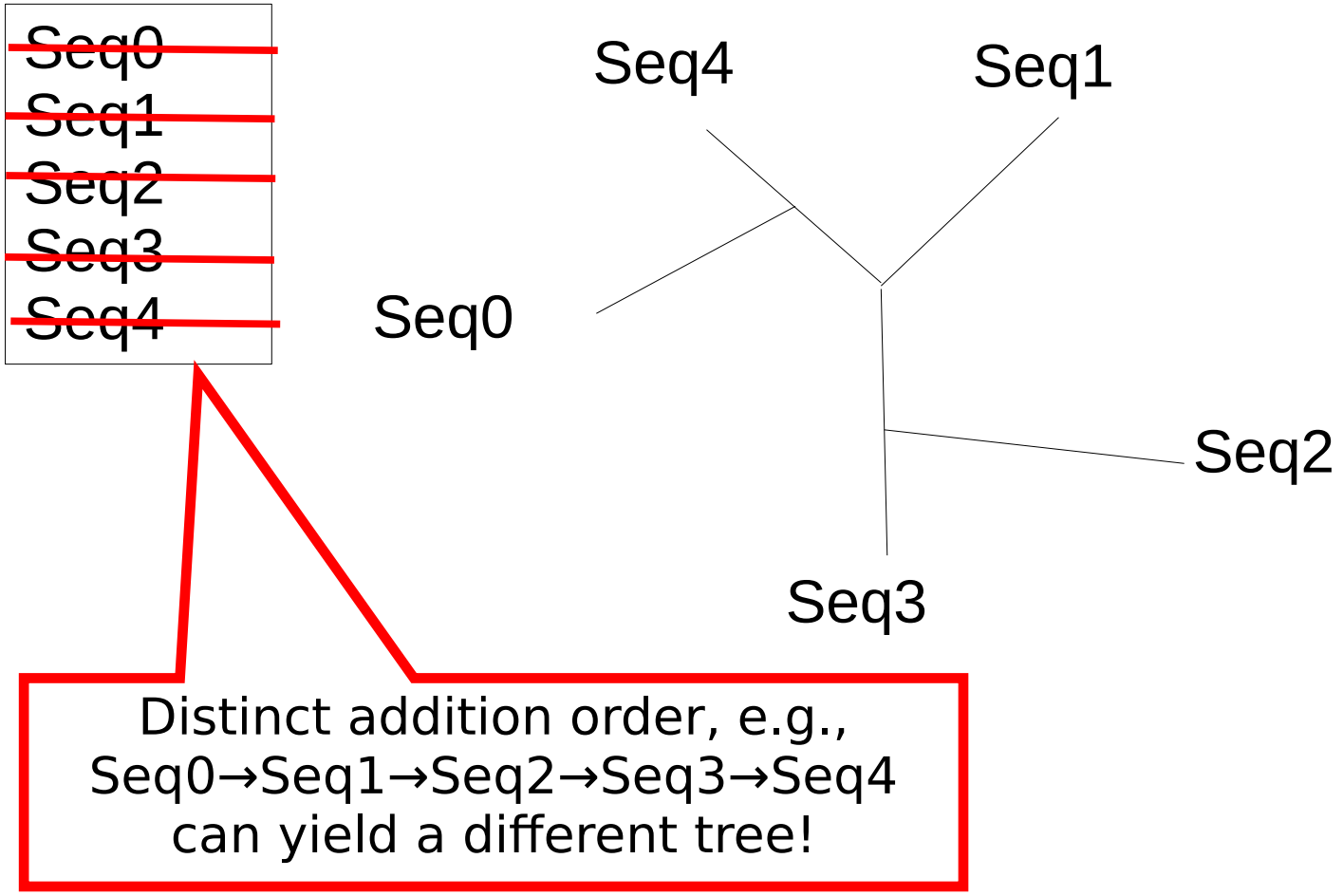




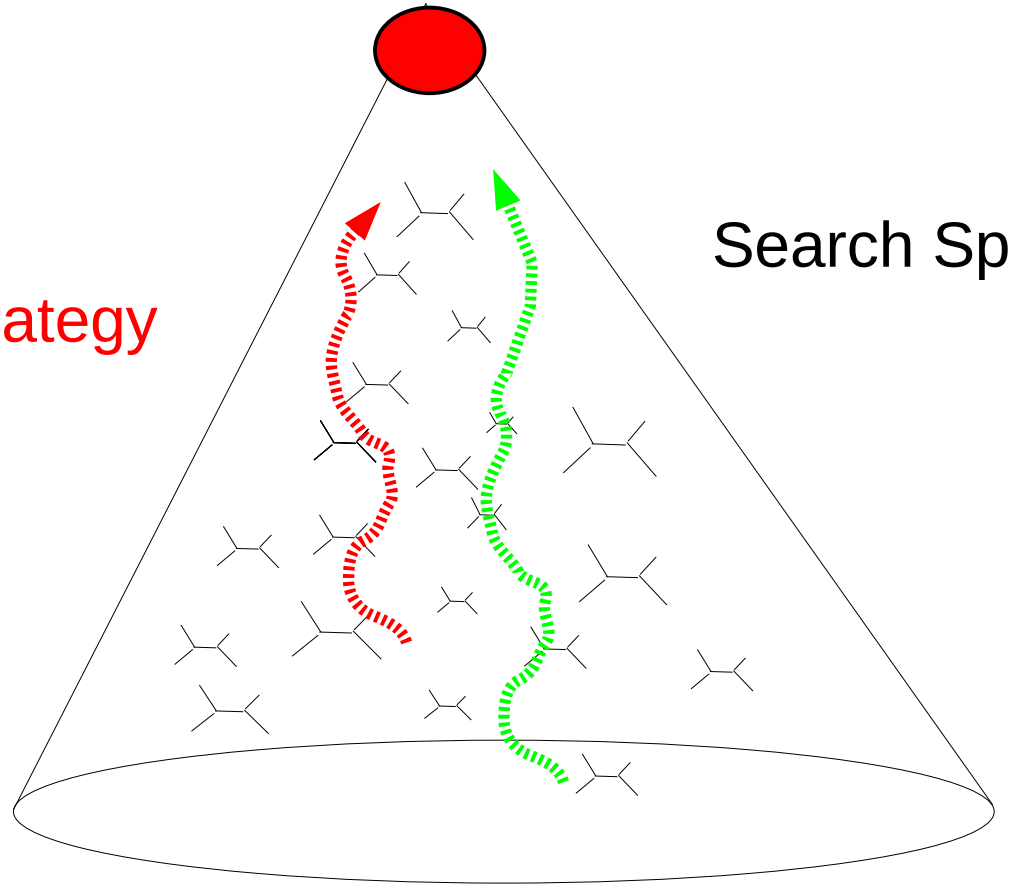
Randomized Stepwise Addition Order Algorithm



Randomized Stepwise Addition Order Algorithm



Why are distinct Starting Trees useful?



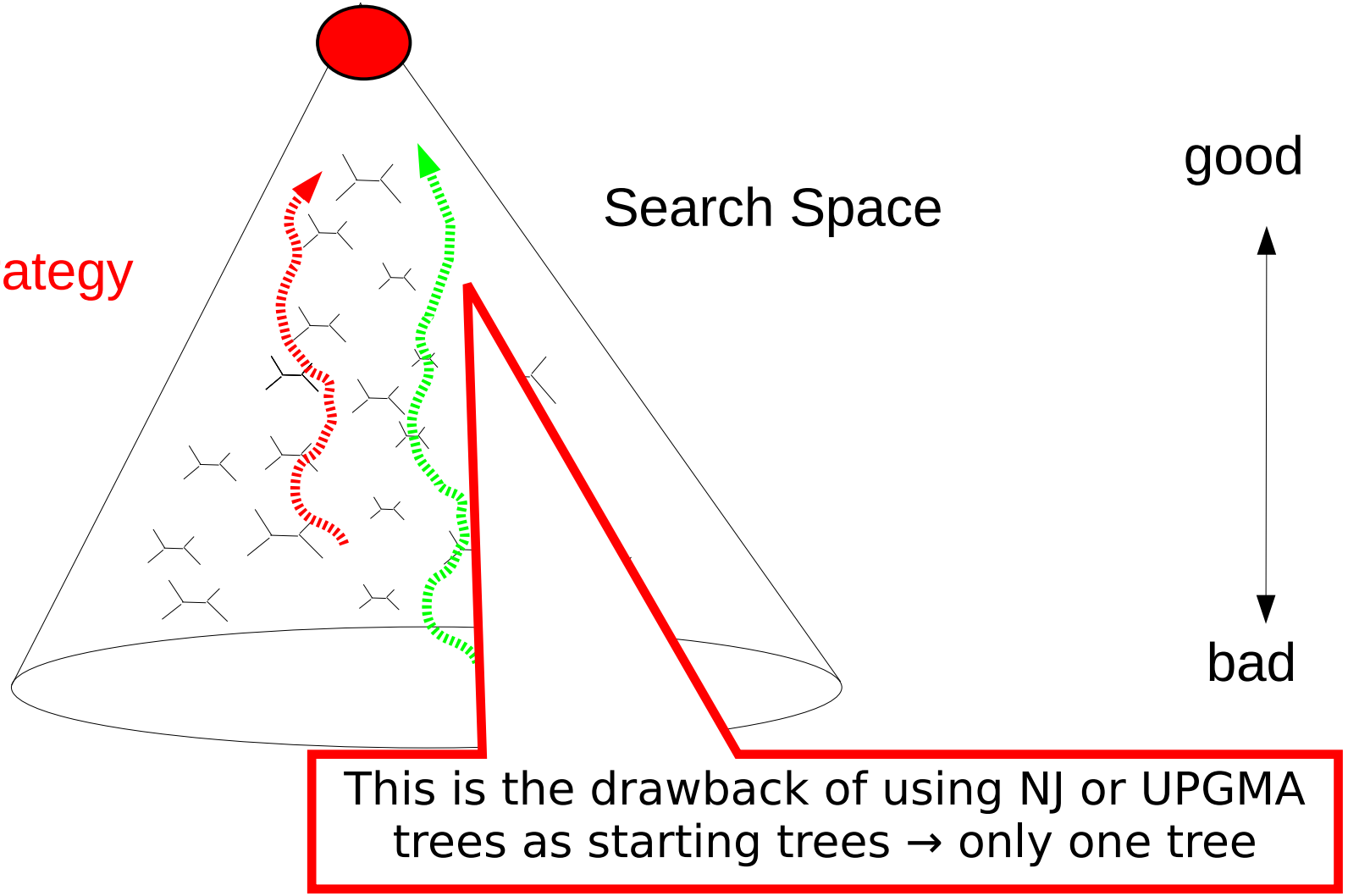
Search Strategy

Search Space

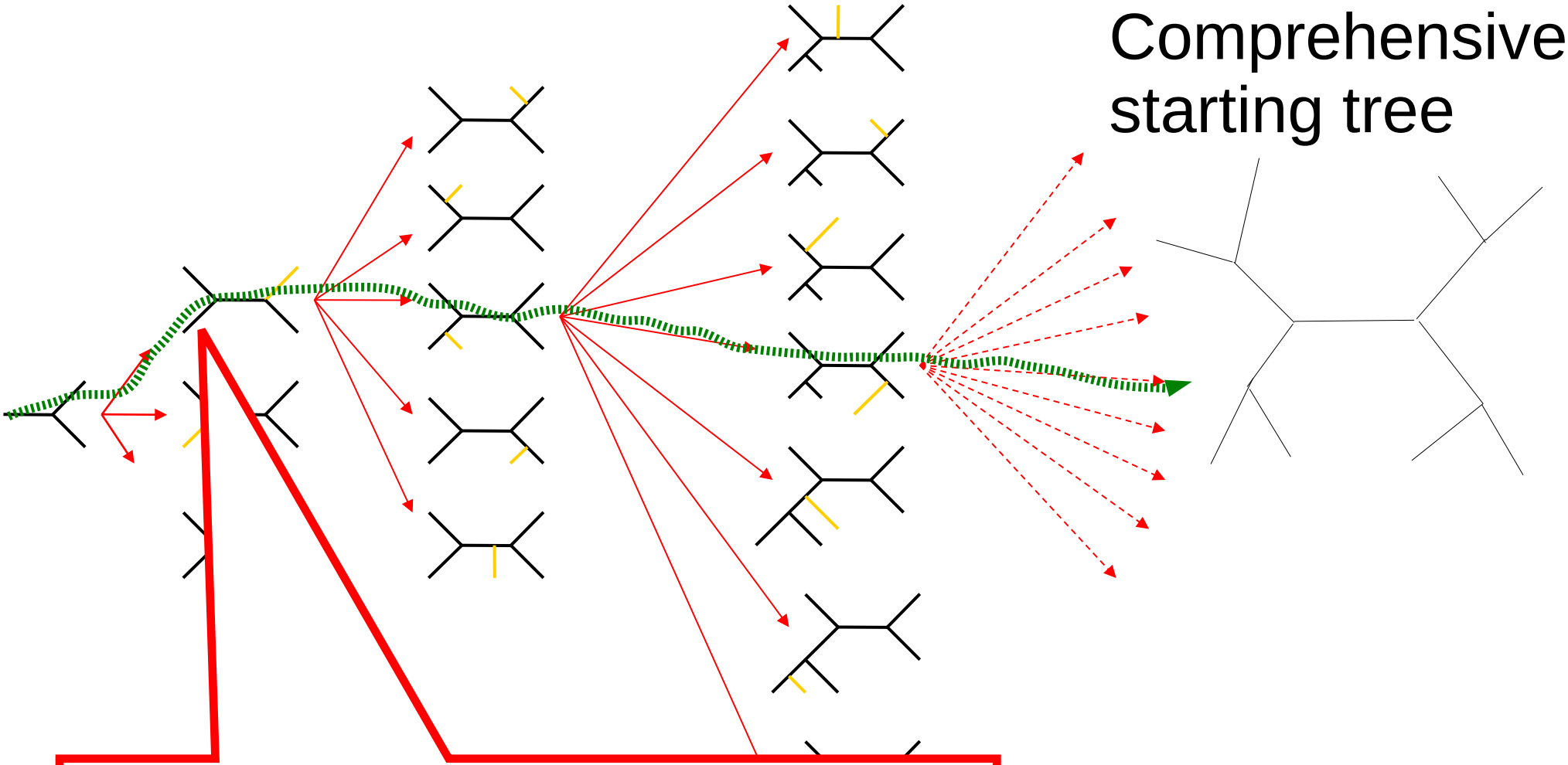
good

bad

Why are distinct Starting Trees useful?

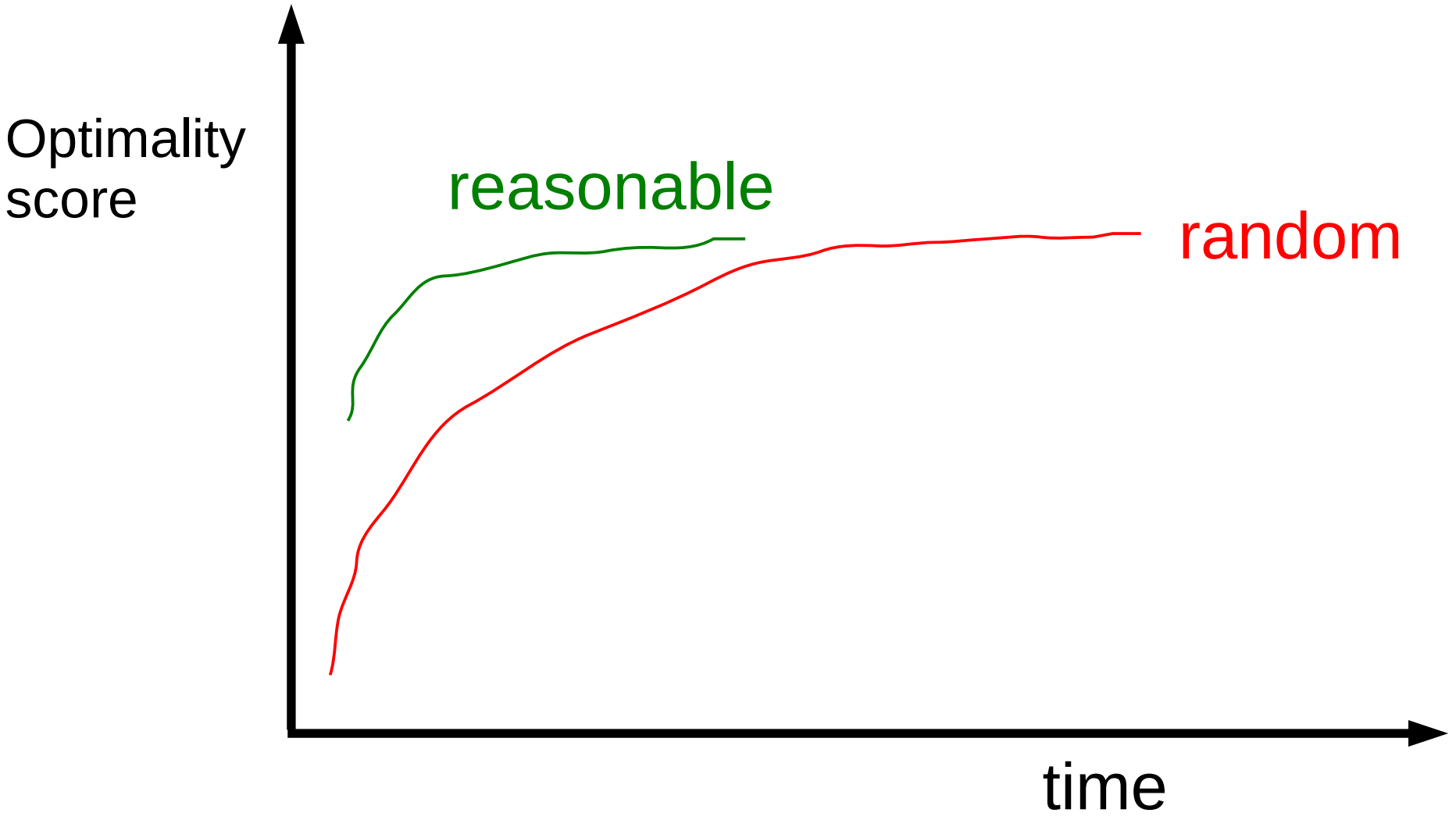


The number of trees



Stepwise addition is like following a single path through this!

Random versus Reasonable Starting trees

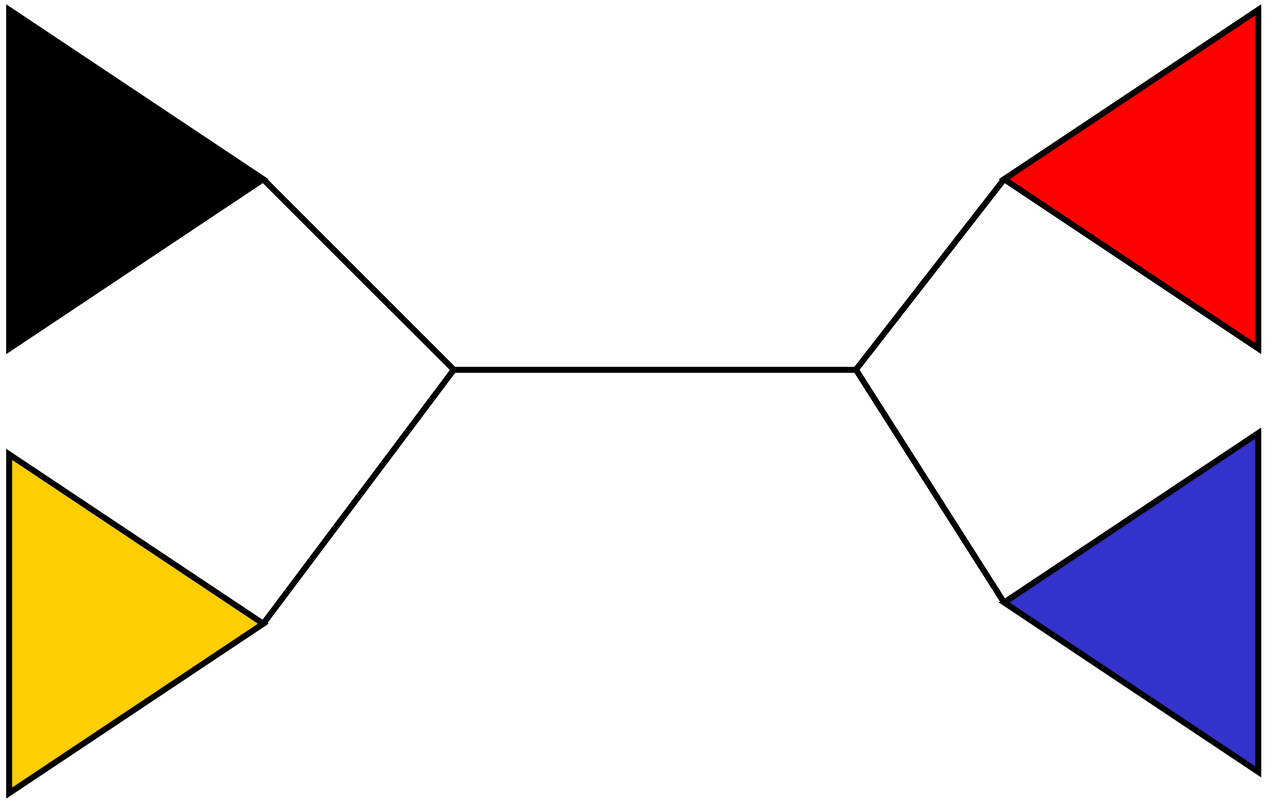


Search Strategies

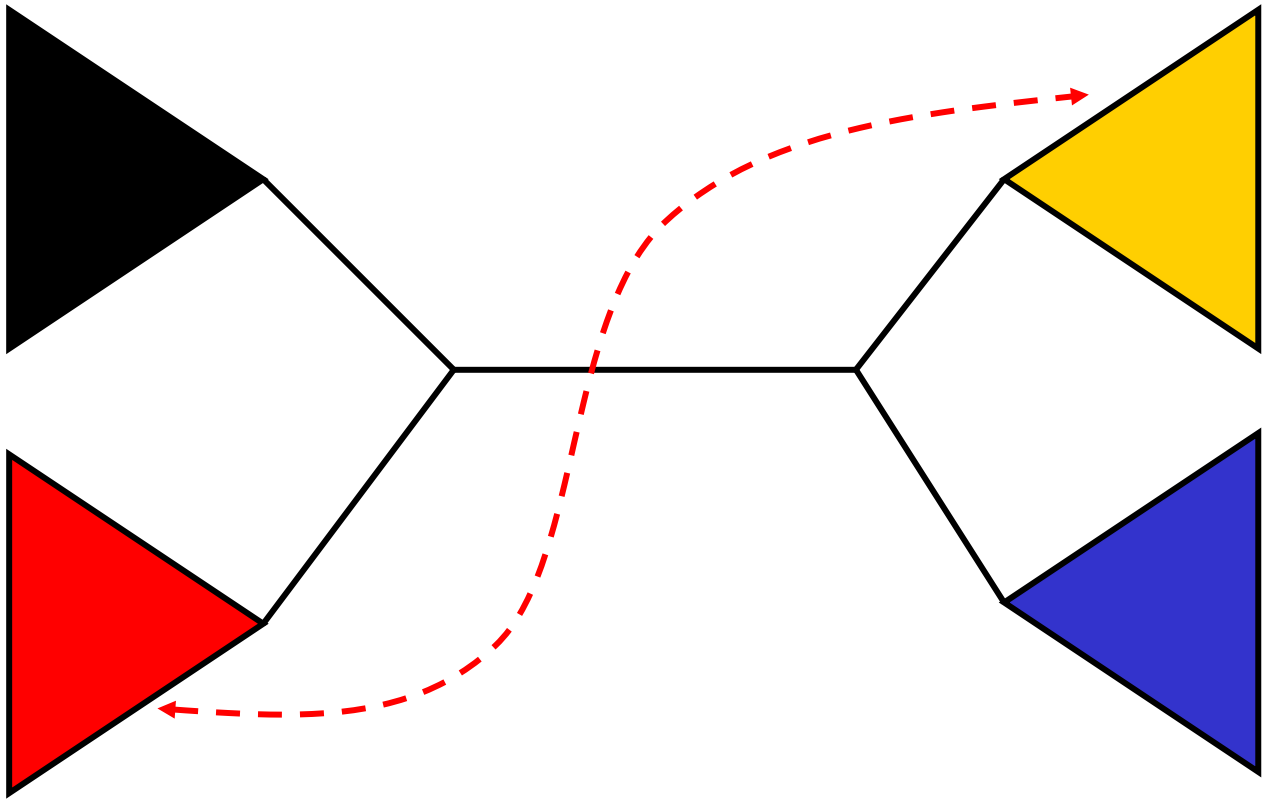
- Given a comprehensive tree
- Apply topological alteration mechanisms in some order to improve the score, for instance, via
  - Hill-climbing
  - Simulated annealing
  - Some other technique

    → 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

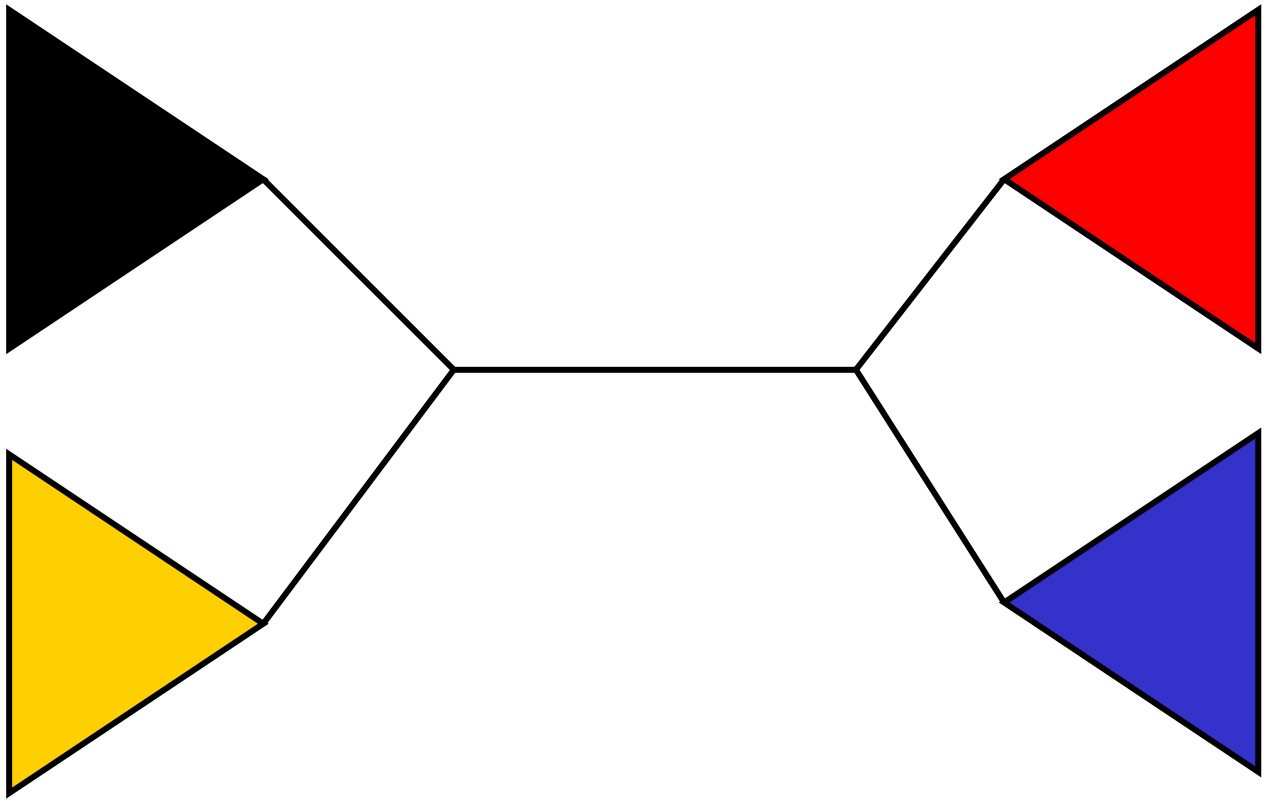
NNI



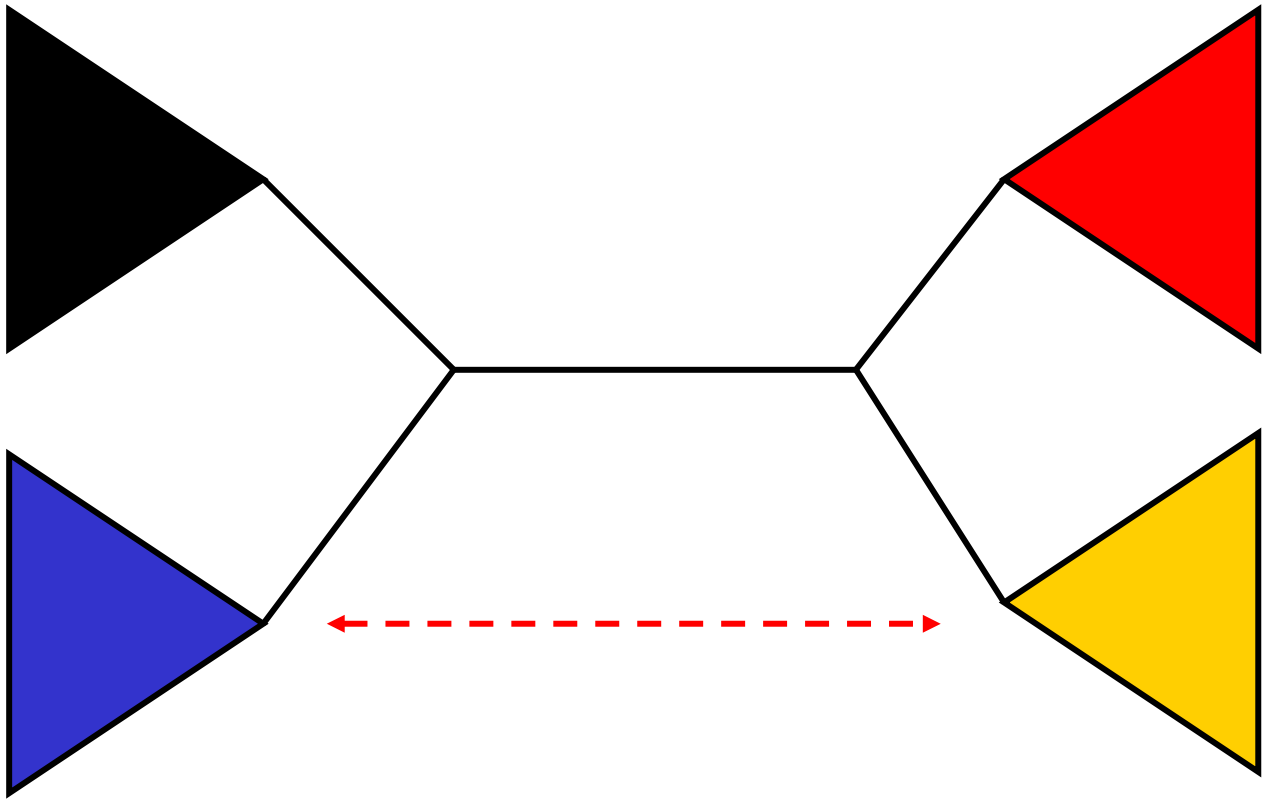
NNI



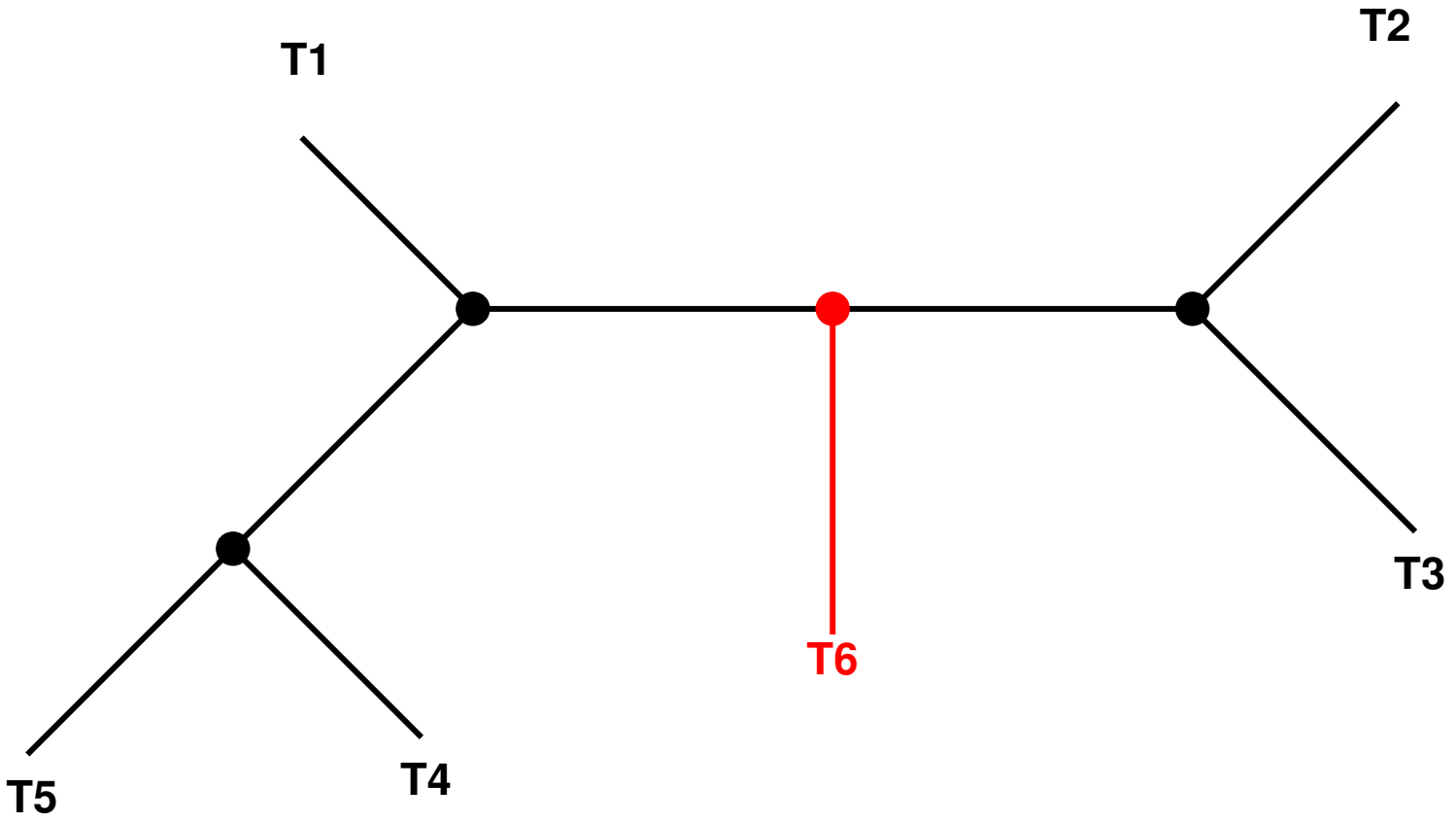
NNI



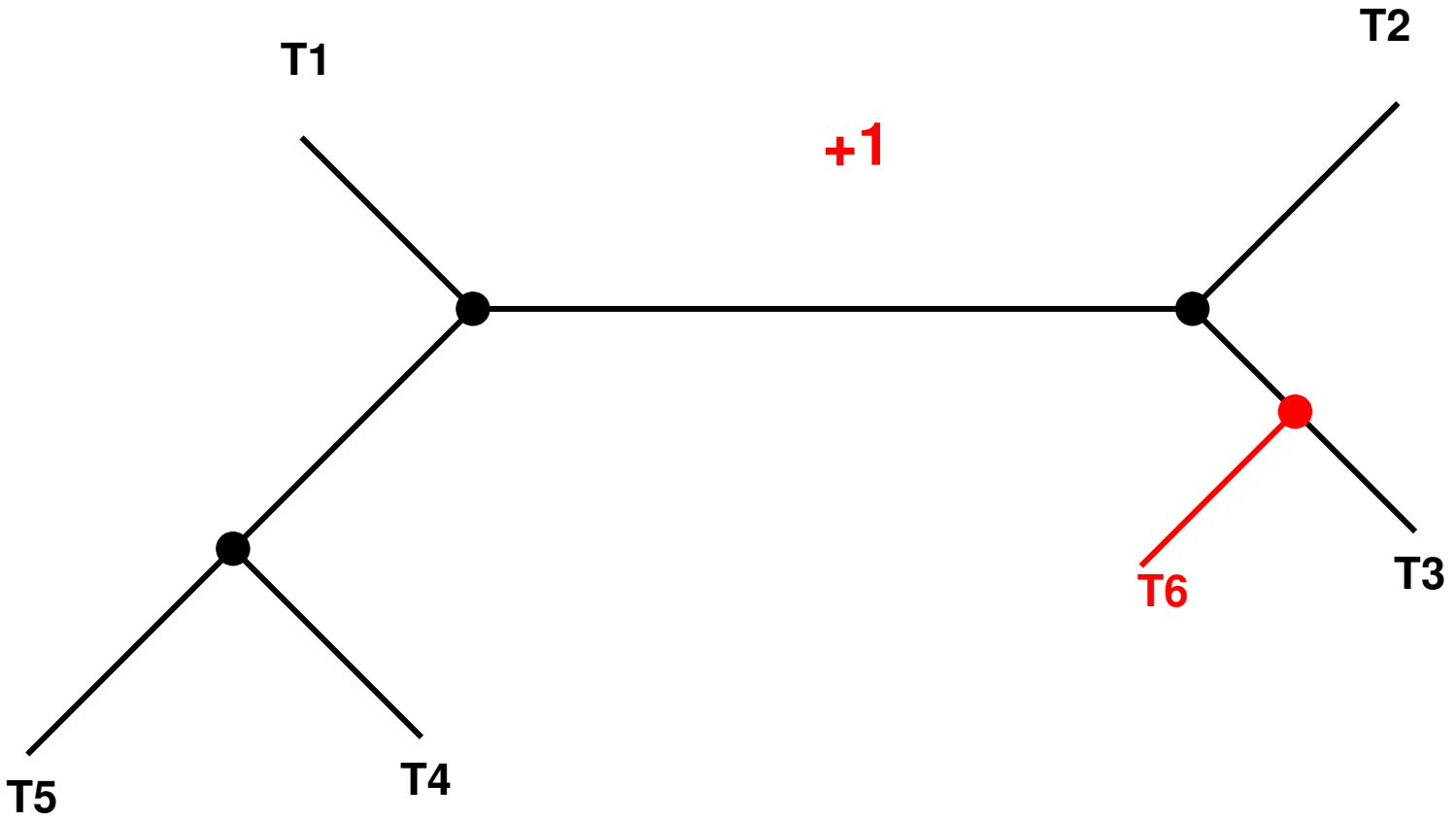
NNI



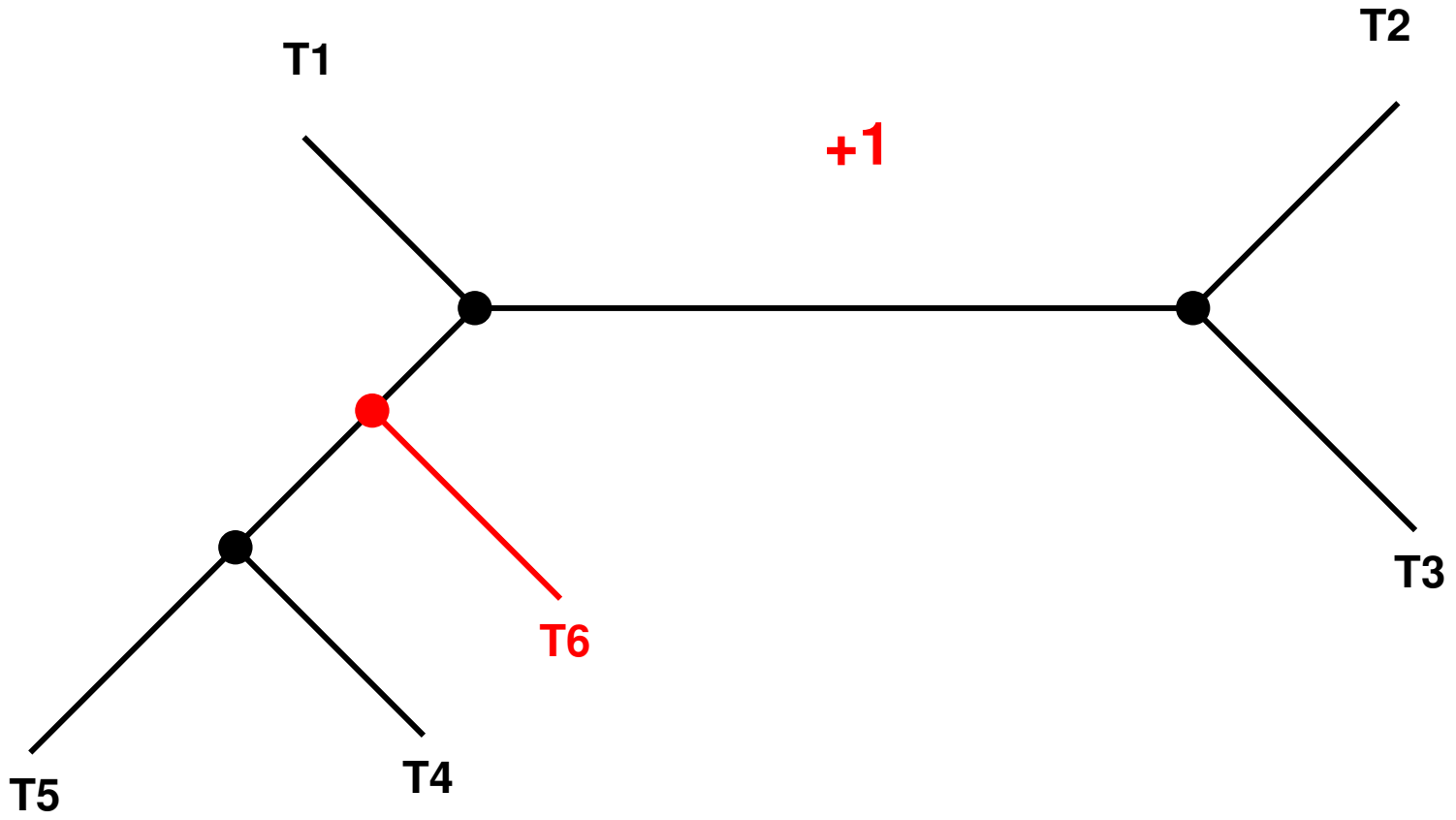
SPR



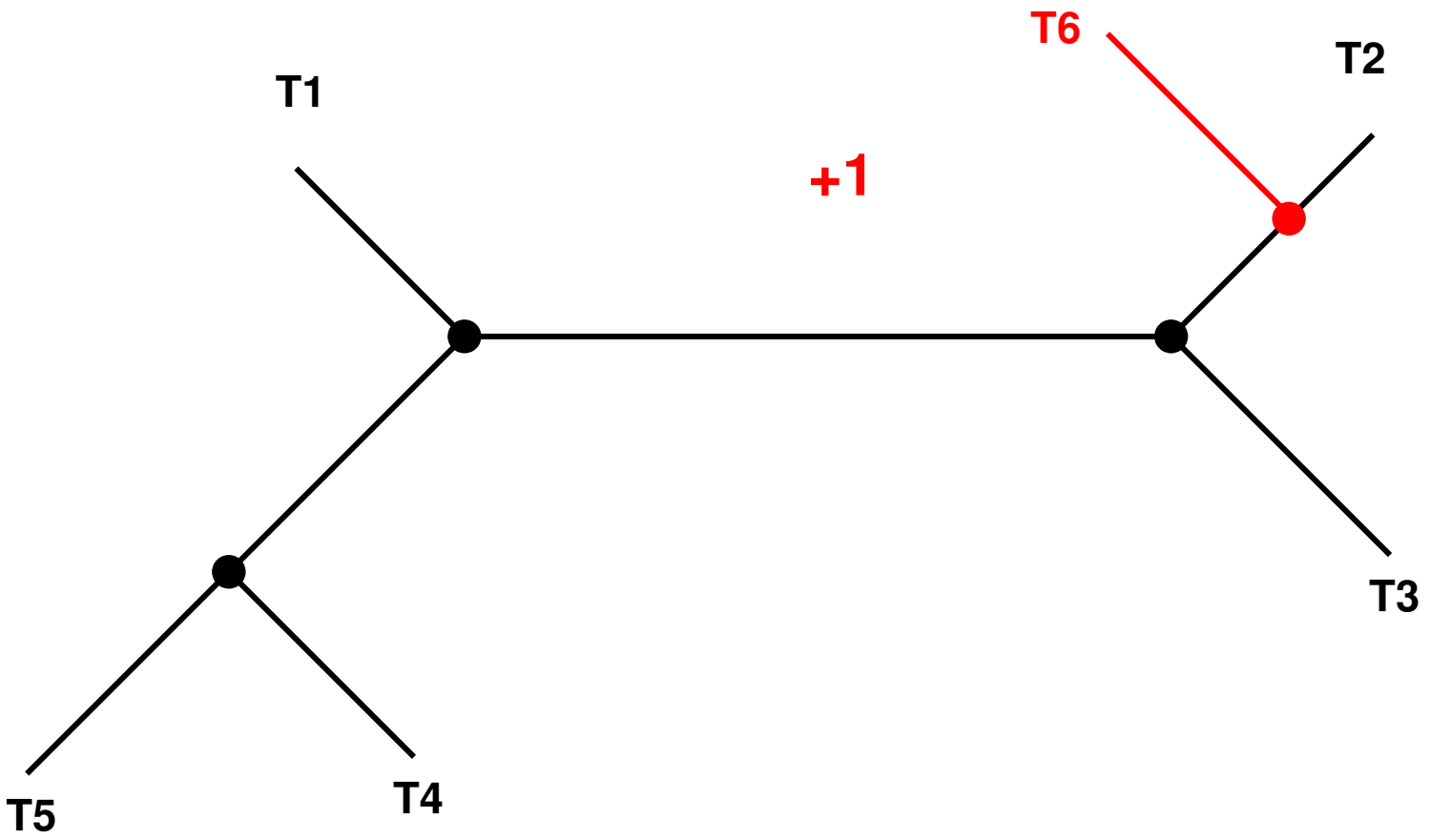
SPR



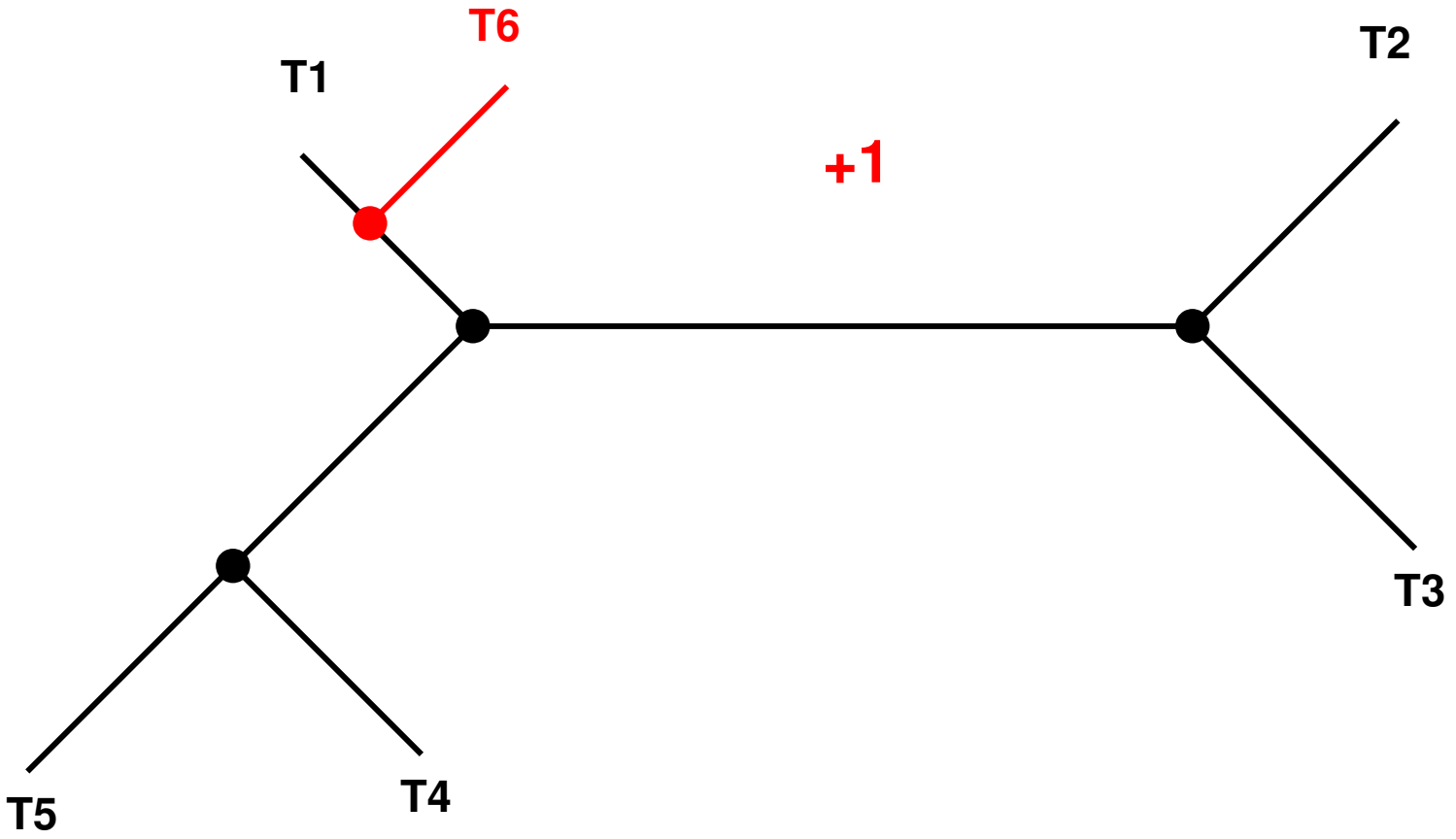
SPR



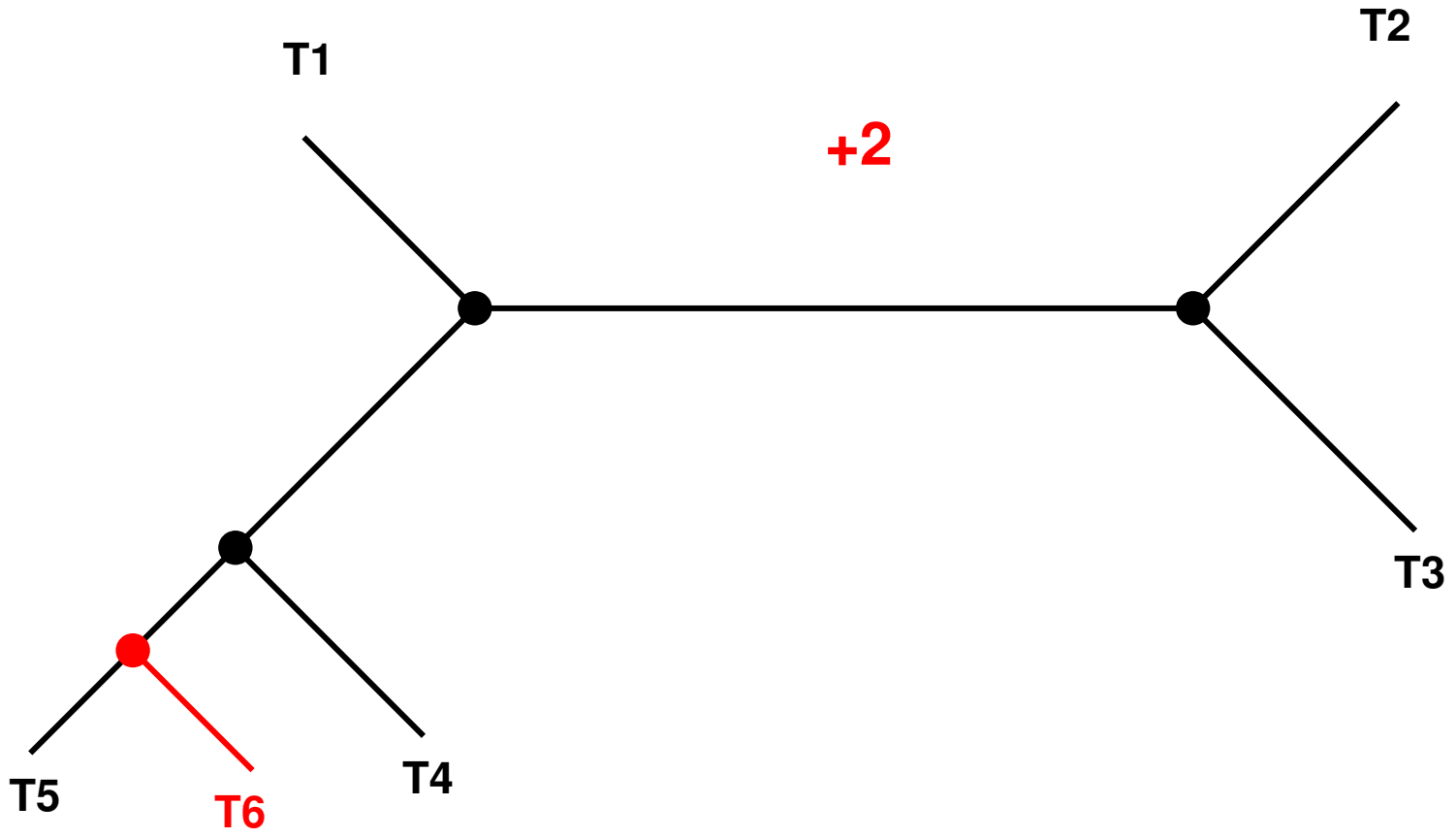
SPR



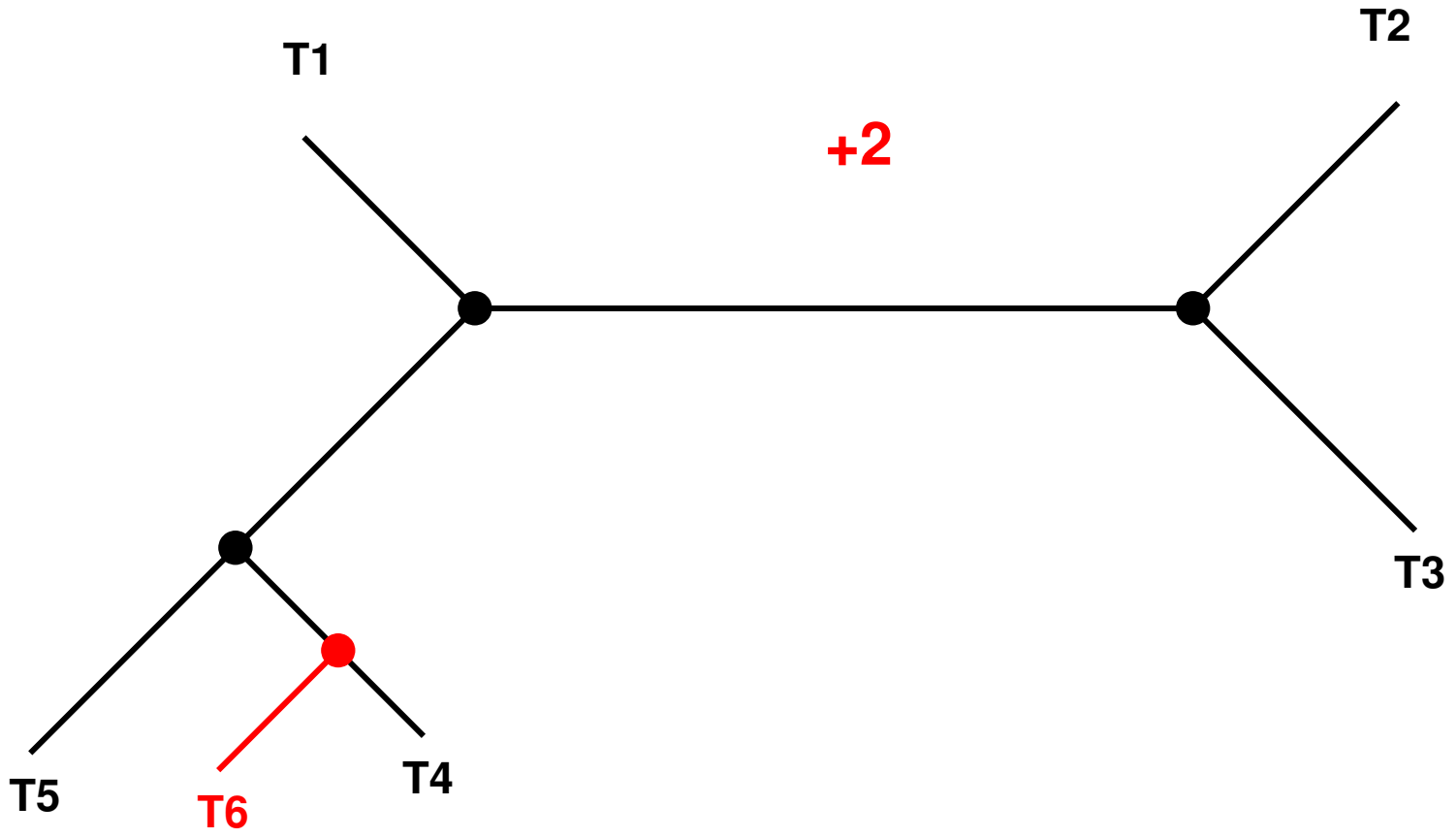
SPR



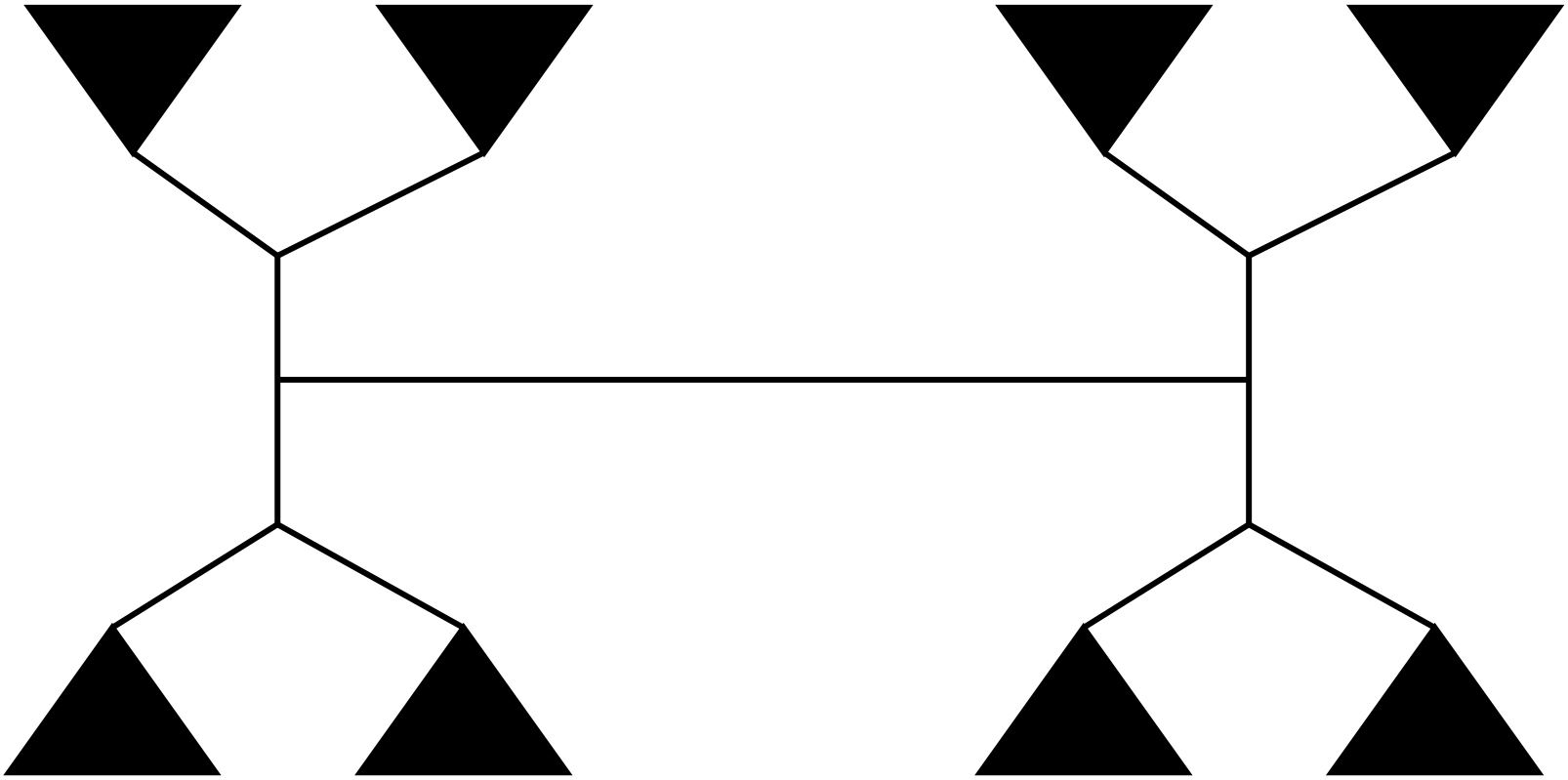
SPR



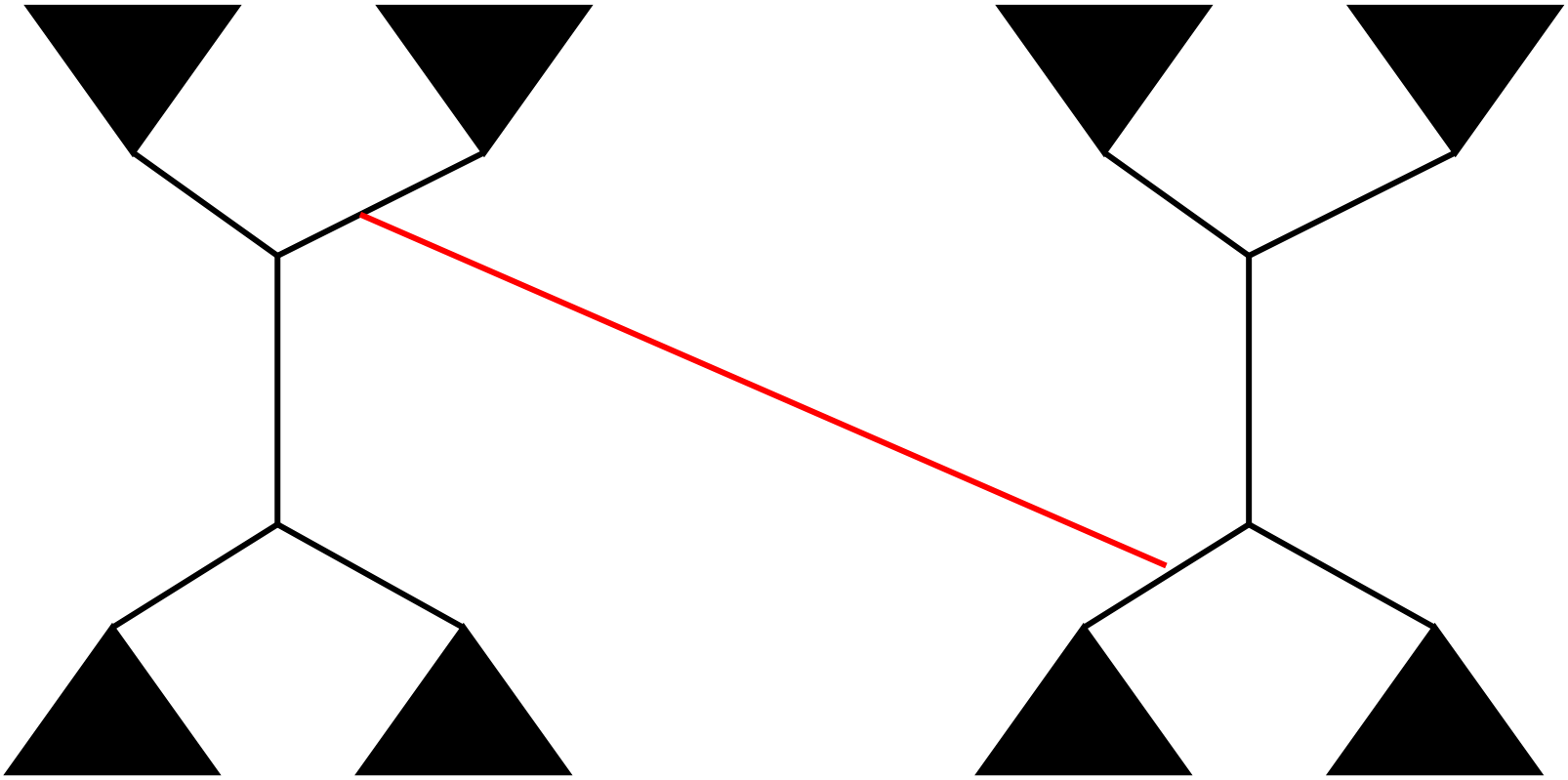
SPR



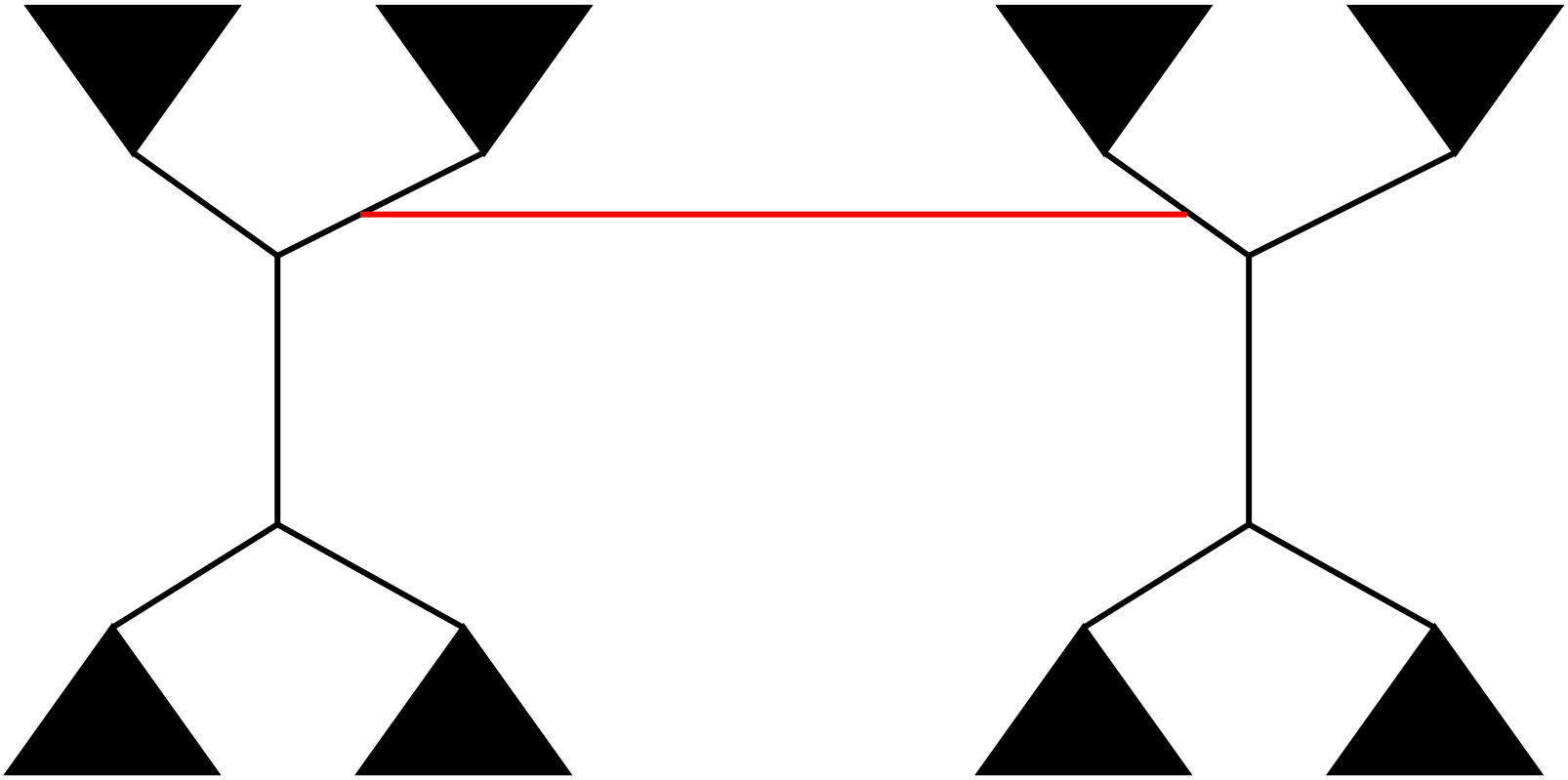
TBR



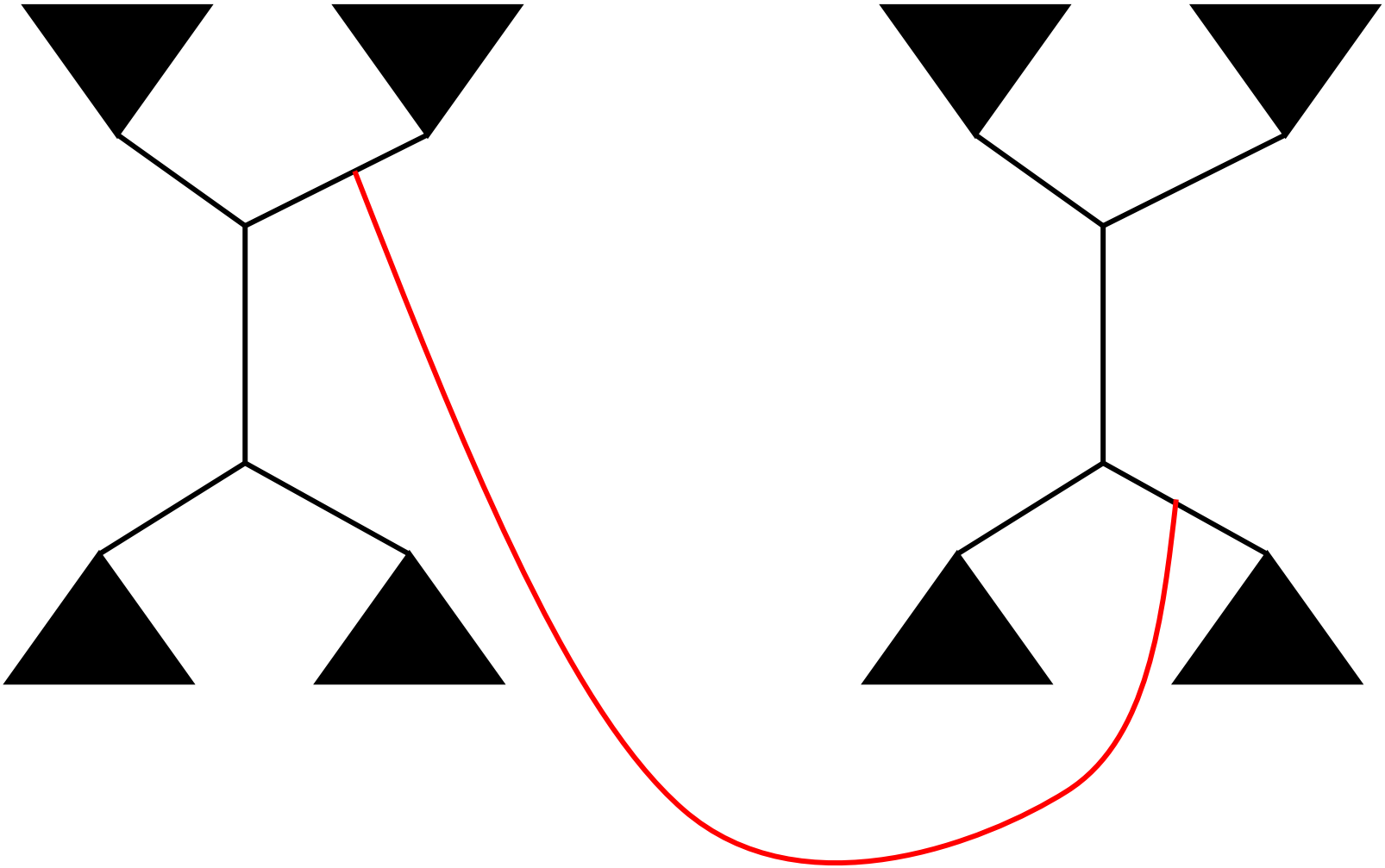
TBR



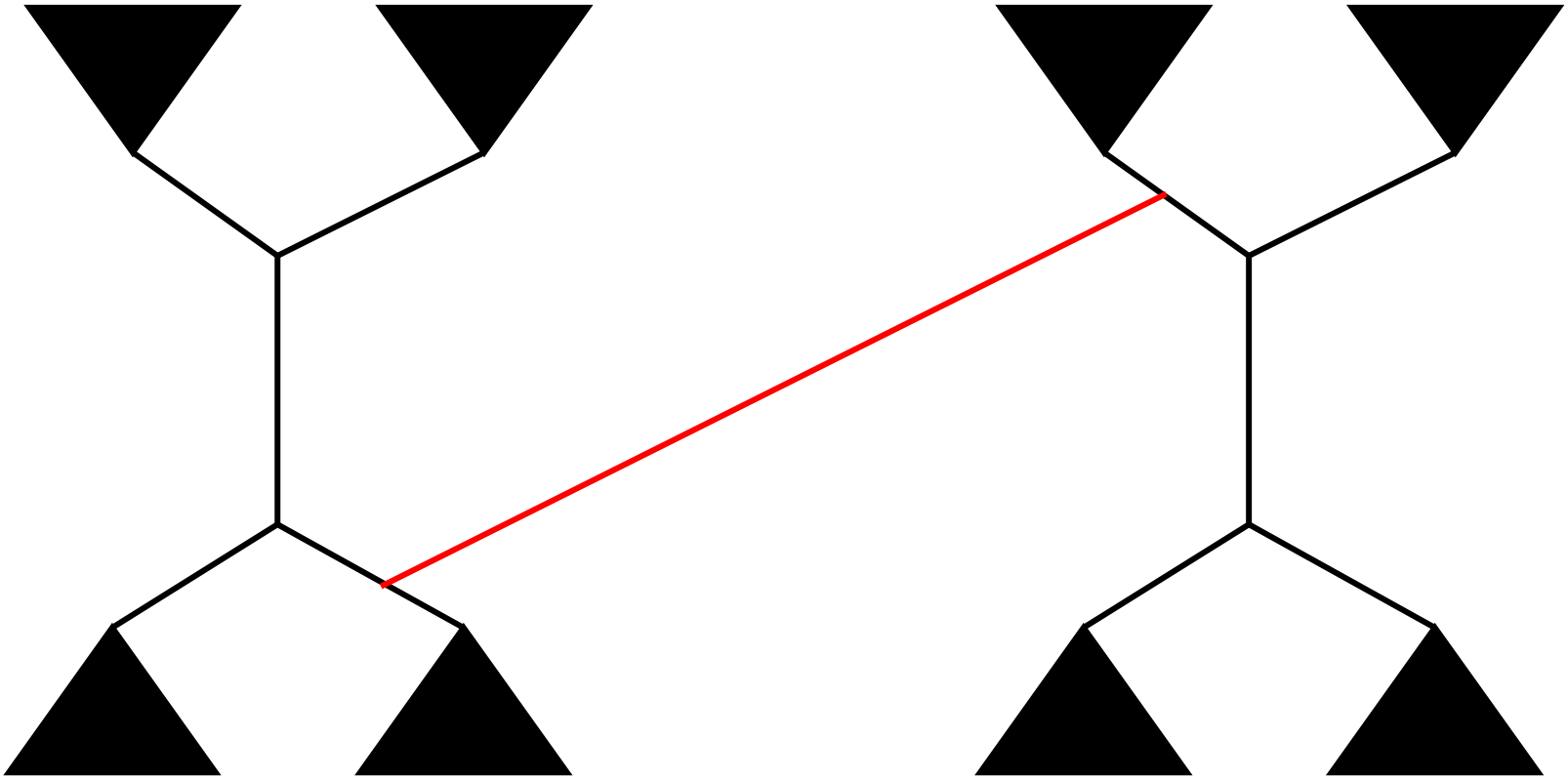
TBR



TBR



TBR



Question

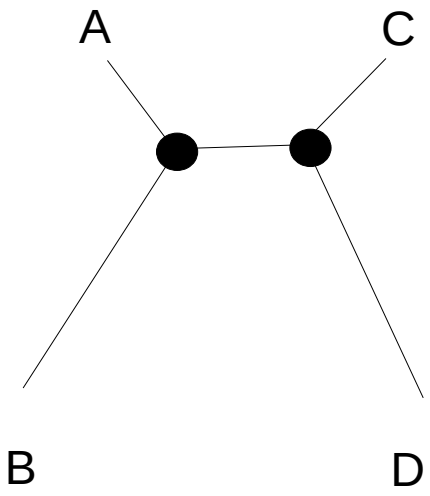
- How could one design a search algorithm for the least squares criterion given a function *f()* and a distance matrix *D* to compute the least squares score on a given tree?

The Parsimonator Algorithm

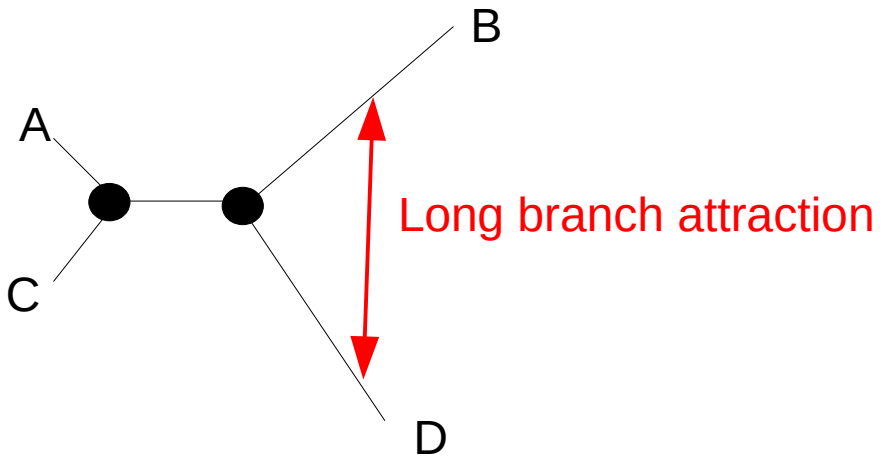
- Build a randomized stepwise addition order parsimony tree
- Apply SPR moves to all subtrees of the current (comprehensive) tree with a rearrangement radius of 20
- If the rearrangement of a subtree yields an improved parsimony score, keep it immediately
  - → this is somewhat greedy as opposed to a 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

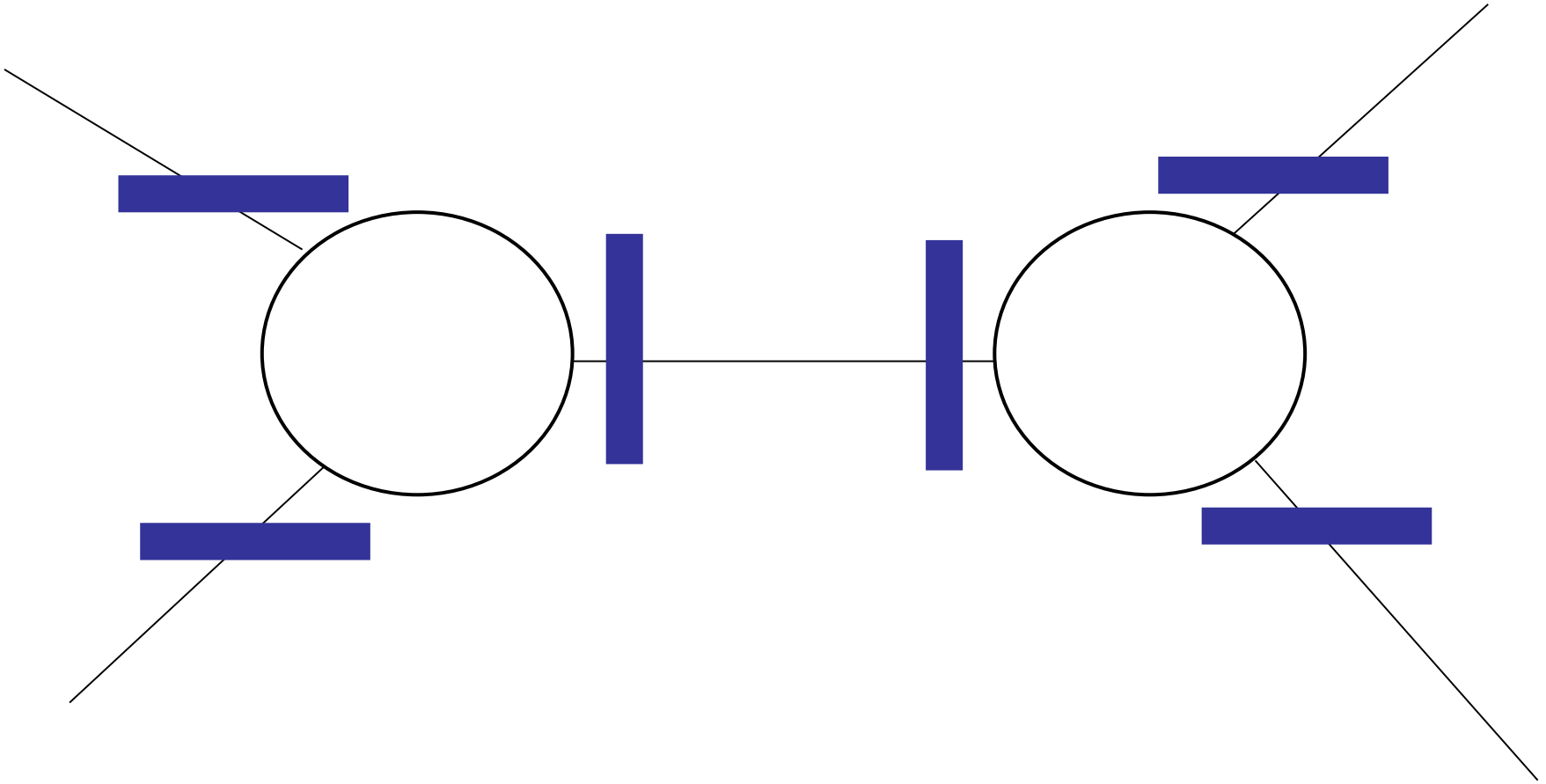
Parsimony & Long Branch Attraction

- 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

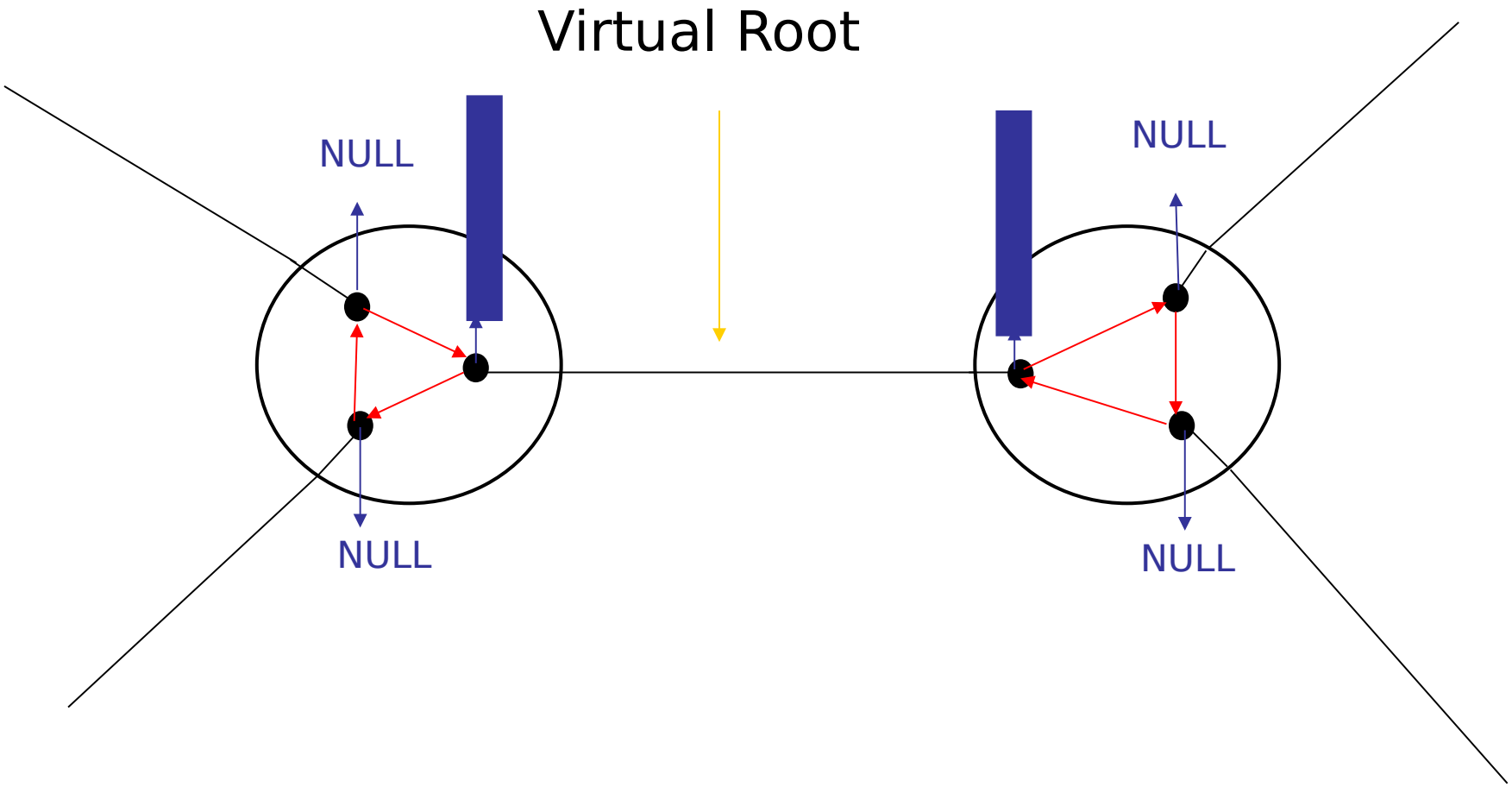
Data Structures for unrooted Trees

- Unrooted trees with dynamically changing virtual roots need a dedicated tree data structure

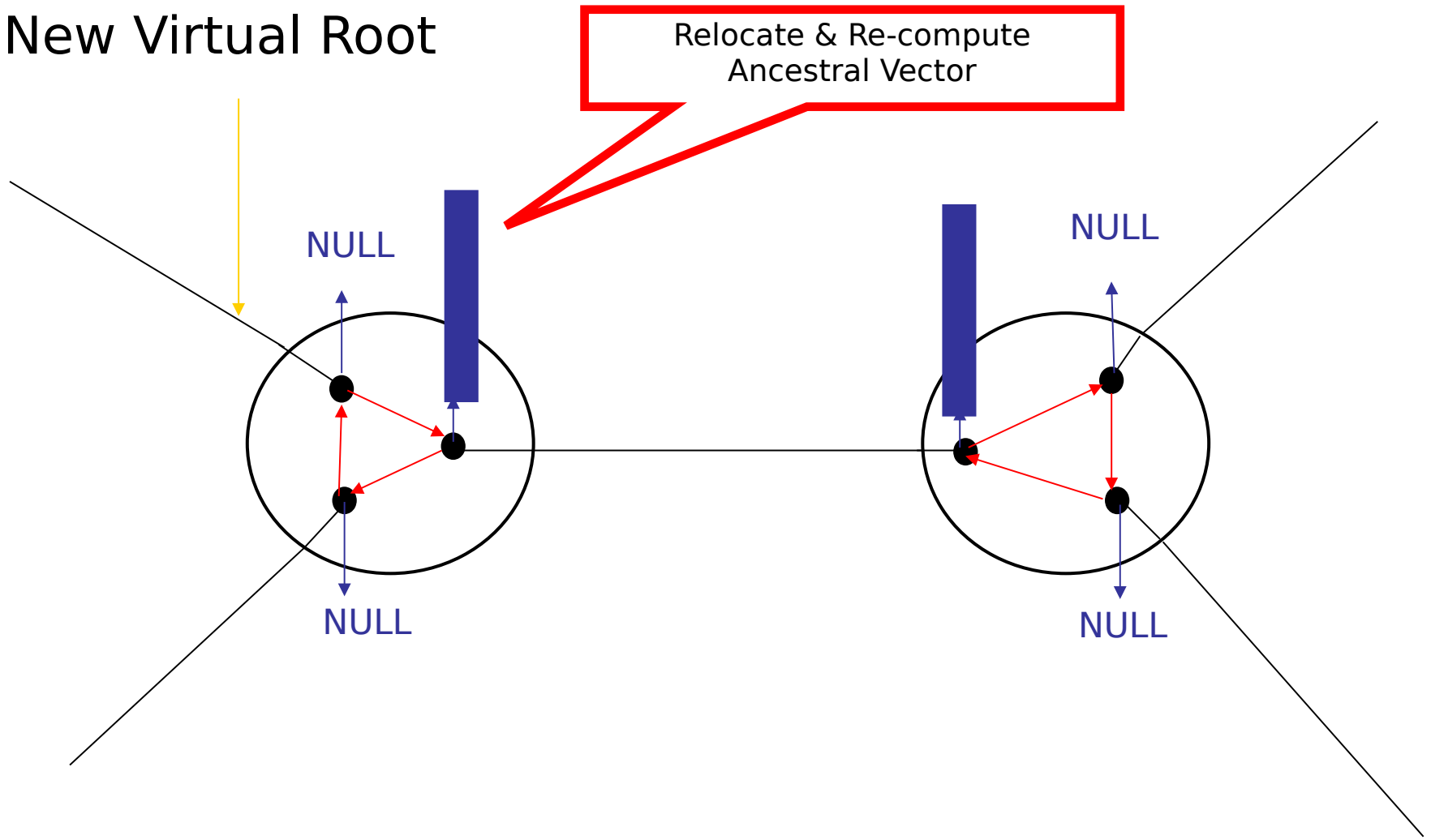
Memory Organization: Ancestral Vectors with an Unrooted View



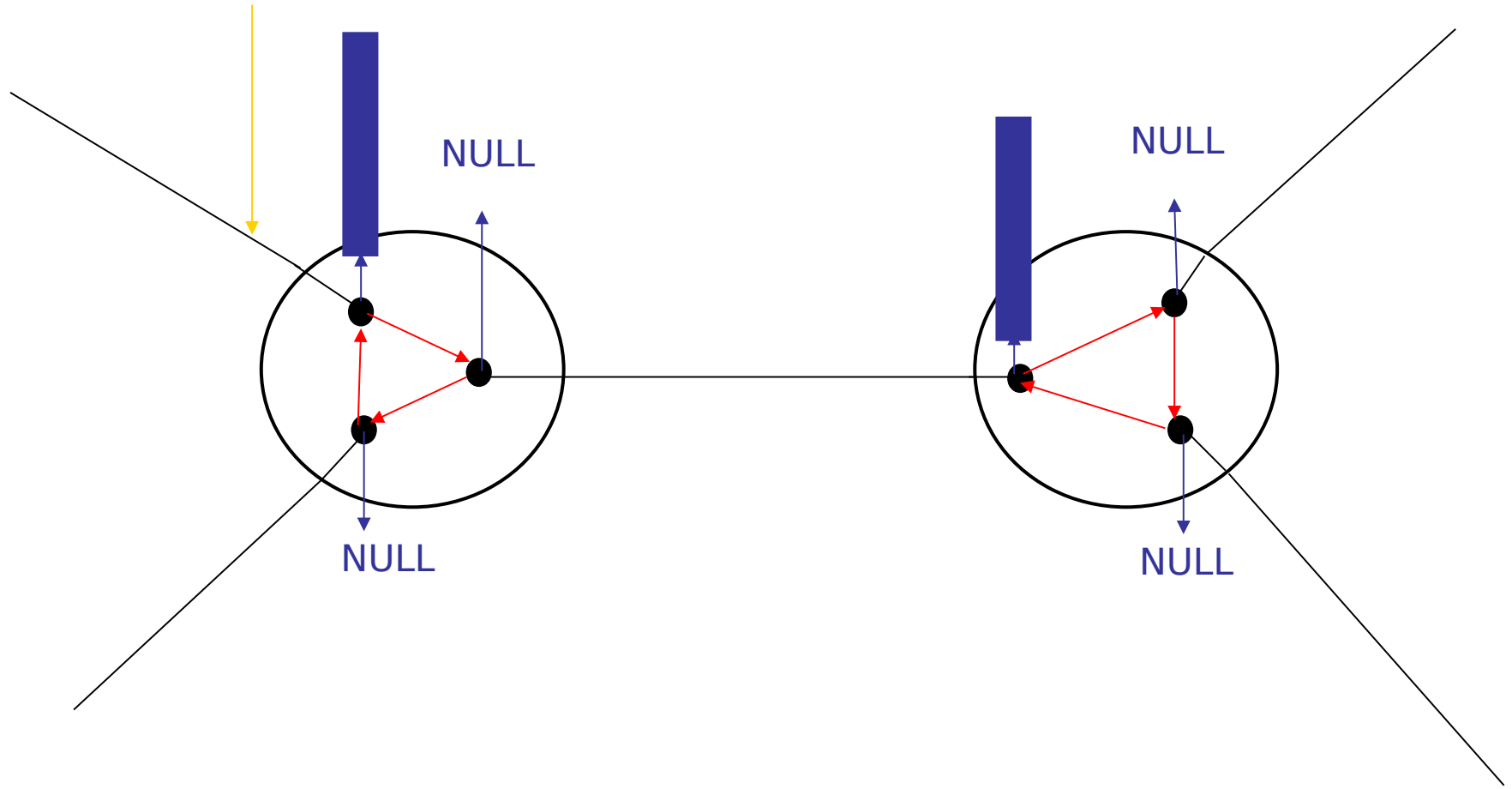
Memory Organization: Ancestral Vectors with a Rooted View



Memory Organization: Ancestral Vectors with a Rooted View



Memory Organization: Ancestral Vectors with a Rooted View



Memory Organization: Tip Vectors