Building Phylogenetic Trees

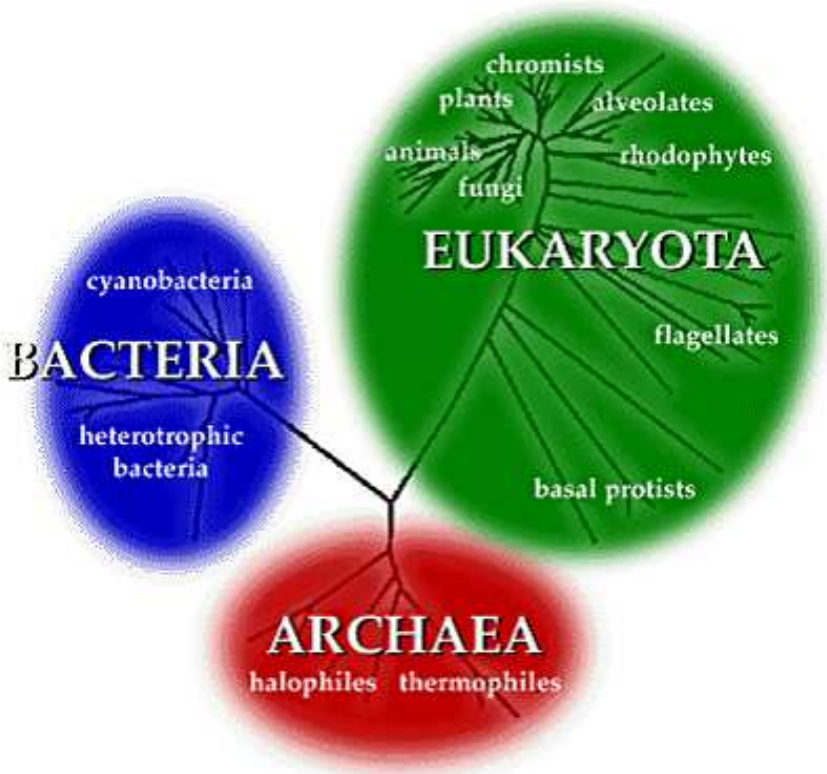
based on:

Biological Sequence Analysis, Ch. 7

by R. Durbin et al., 1998

Introduction to Biological Algorithms, Ch. 10

by N. Jones and P. Pevzner, 2004



[a tree of life]

Acknowledgements:

M.Sc. student Daniel Bolohan
M.Sc. student Diana Popovici

PLAN

1 Introduction to Phylogeny

2 Distance-based Phylogeny

- Average Linkage (UPGMA) algorithm
- Neighbour-Joining algorithm

3 Character-based Phylogeny

Small Parsimony

- traditional parsimony (Fitch) algorithm
- weighted parsimony (Sankoff) algorithm

Large Parsimony

- a greedy approach: Nearest Neighbour Interchange
- a branch and bound approach

4 Simultaneous Phylogeny and Multiple Sequence Alignment

- gap-substitution (Sankoff-Cedergren) algorithm
- affine gap (Hein) algorithm

1 Introduction to Phylogeny

"The field of phylogeny has the goal of working out the biological relationships among species, populations, individuals or genes..." (Arthur Lesk, *Introduction to Bioinformatics*, 2002) ...based on similarities of their characteristics.

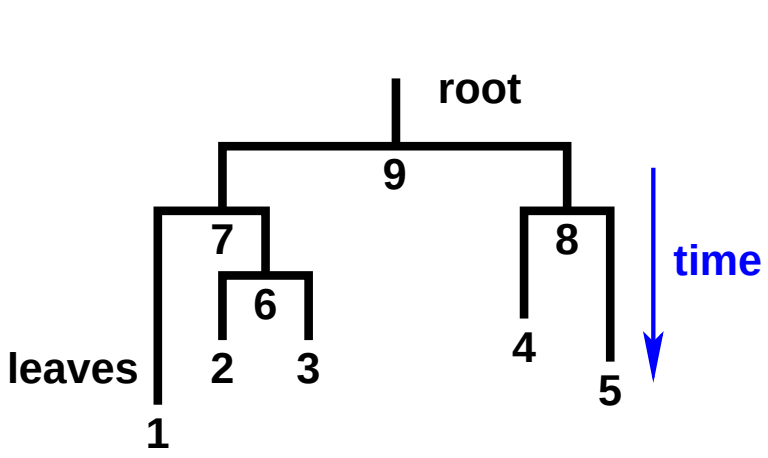
Basic principle in evolution theory: the origin of similarity is common ancestry.

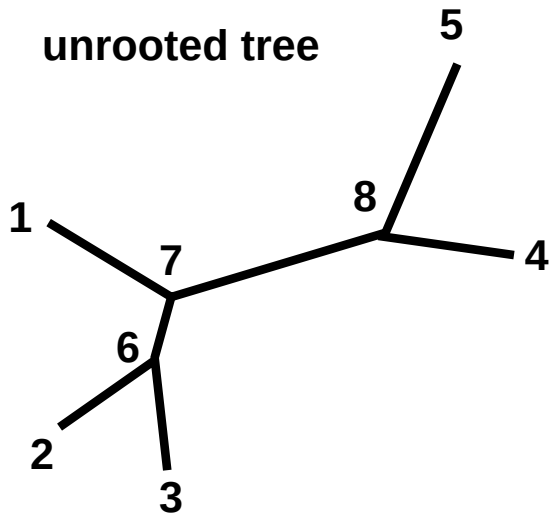
Relationships in phylogenetics are usually expressed as binary (rooted or unrooted) trees:
leaves represent species or sequences to be compared;
nodes are bifurcations (not necessarily ancestors).

Edge length signifies either
some measure of the similarity (distance) between two species, or
the length of time since their separation.

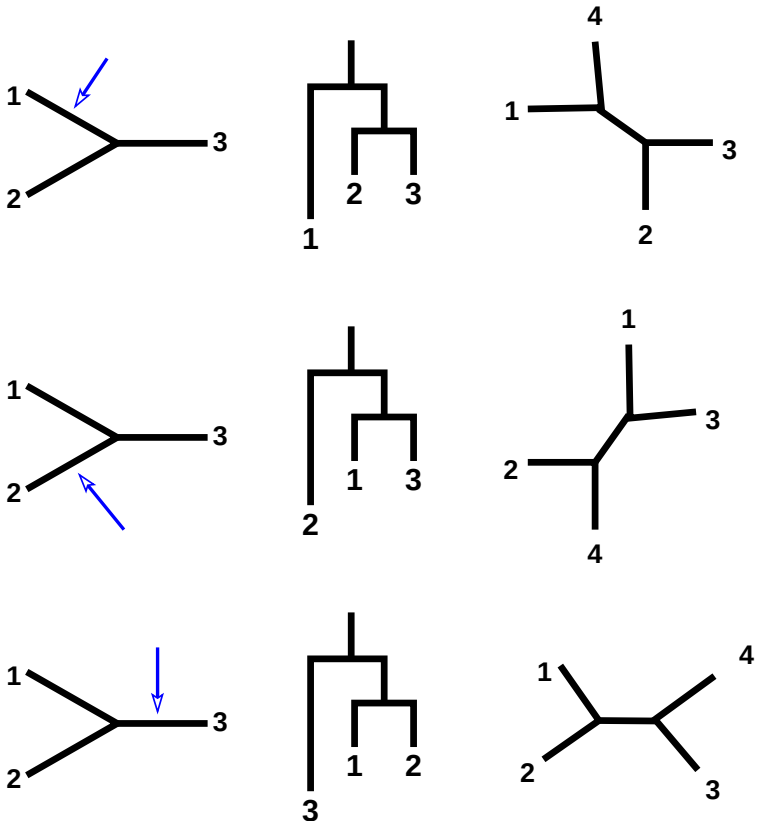
Today, DNA sequences provide the best measures of similarities among species for phylogenetic analysis.

Some terminology: Rooted vs. Unrooted Trees





An example of a binary tree showing the root and leaves, and the direction of evolutionary time. The corresponding unrooted tree is also shown; the direction of time here is undetermined.



The rooted trees (center column) and the unrooted trees (right column) obtained from an unrooted tree with 3 leaves.

Proposition

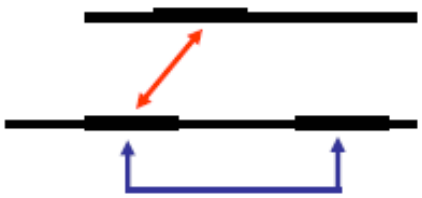
There are $(2n-3)!! = 1 \cdot 3 \cdot \ldots \cdot (2n-3)$ rooted trees with n leaves, and $(2n-5)!!$ unrooted trees with n leaves.

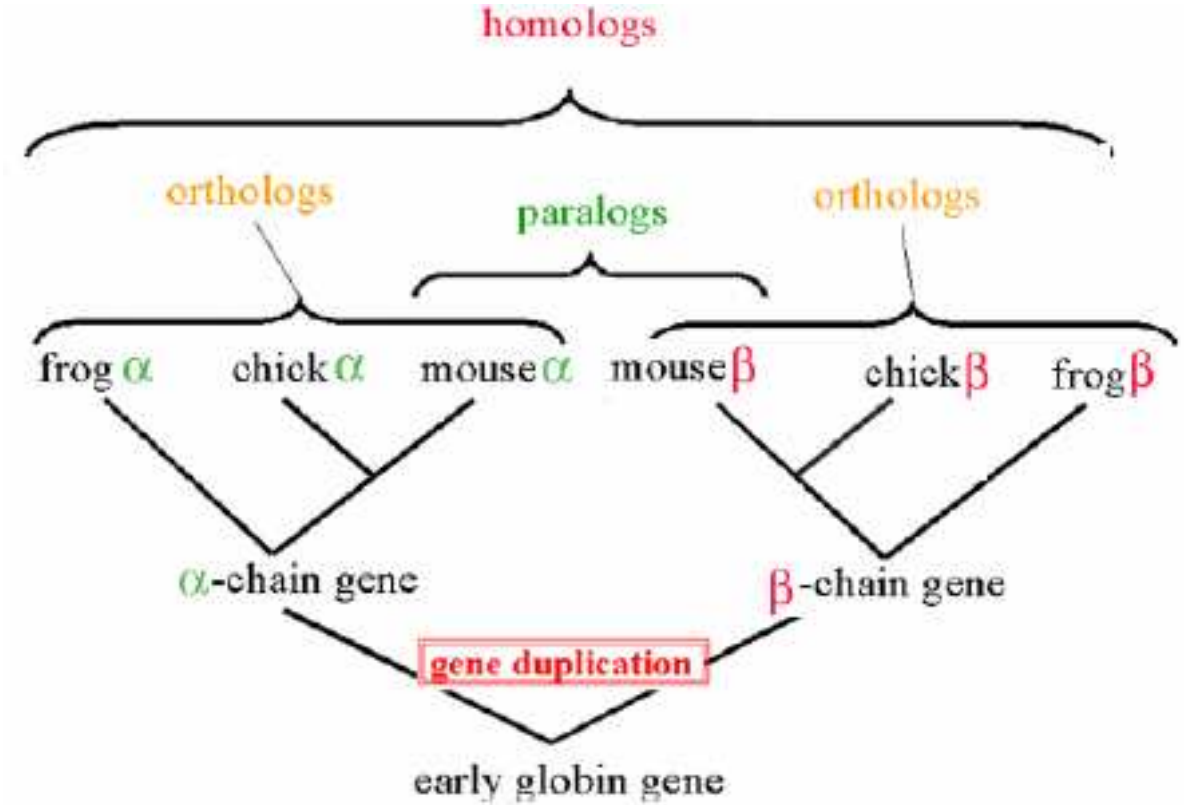
LC: We can also show (by induction) that any unrooted tree with n leaves has $(2n-3!!$ edges.

Some terminology: Homologous genes

Orthologous genes are homologous (corresponding) genes in different species.

Paralogous genes are homologous genes in the same species (genome).





Acknowledgement: this is a slide from the Sequence Analysis Master Course, Centre for Integrative Bioinformatics, Vrije Universiteit, Amsterdam

Xenologous genes are homologs resulting from the horizontal transfer (...) of a gene between two organisms.

The function of xenologs can be variable, depending on how significant the change was in the context of horizontally moving the gene. In general, though, the function tends to be similar, between and after the horizontal transfer.

Illustrating success stories in phylogenetics (I)

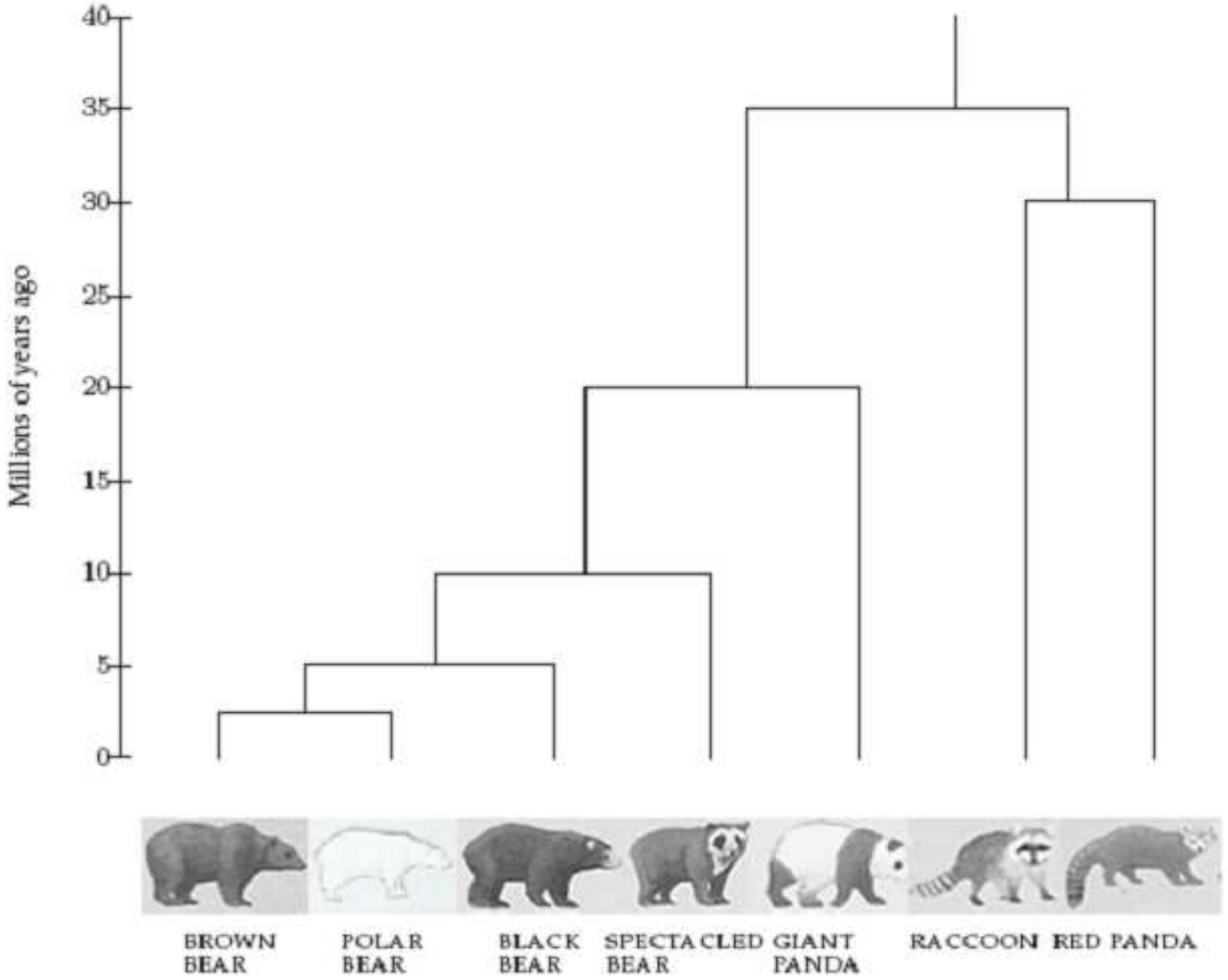
For roughly 100 years (more exactly, 1870-1985), scientists were unable to figure out which family the giant panda belongs to.

Giant pandas look like bears, but have features that are unusual for bears but typical to raccoons: they do not hibernate, they do not roar, their male genitalia are small and backward-pointing.

> Anatomical features were the dominant criteria used to derive evolutionary relationships between species since Darwin till early 1960s.
>
> The evolutionary relationships derived from these relatively subjective observations were often inconclusive. Some of them were later proved incorrect.

In 1985, Steven O'Brien and colleagues solved the giant panda classification problem using DNA sequences and phylogenetic algorithms.

8.


Millions of years ago
40
35
30
25
20
15
10
5
0
BROWN BEAR
POLAR BEAR
BLACK BEAR
SPECTACLED BEAR
GIANT PANDA
RACCOON
RED PANDA

Illustrating success stories in phylogenetics (II)

In 1994, a woman from Lafayette, Louisiana (USA), clamed that her ex-lover (who was a phisician) injected her with HIV+ blood.

Records showed that the physician had drawn blood from a HIV+ patient that day.

But how to prove that the blood from that HIV+ patient ended up in the woman?

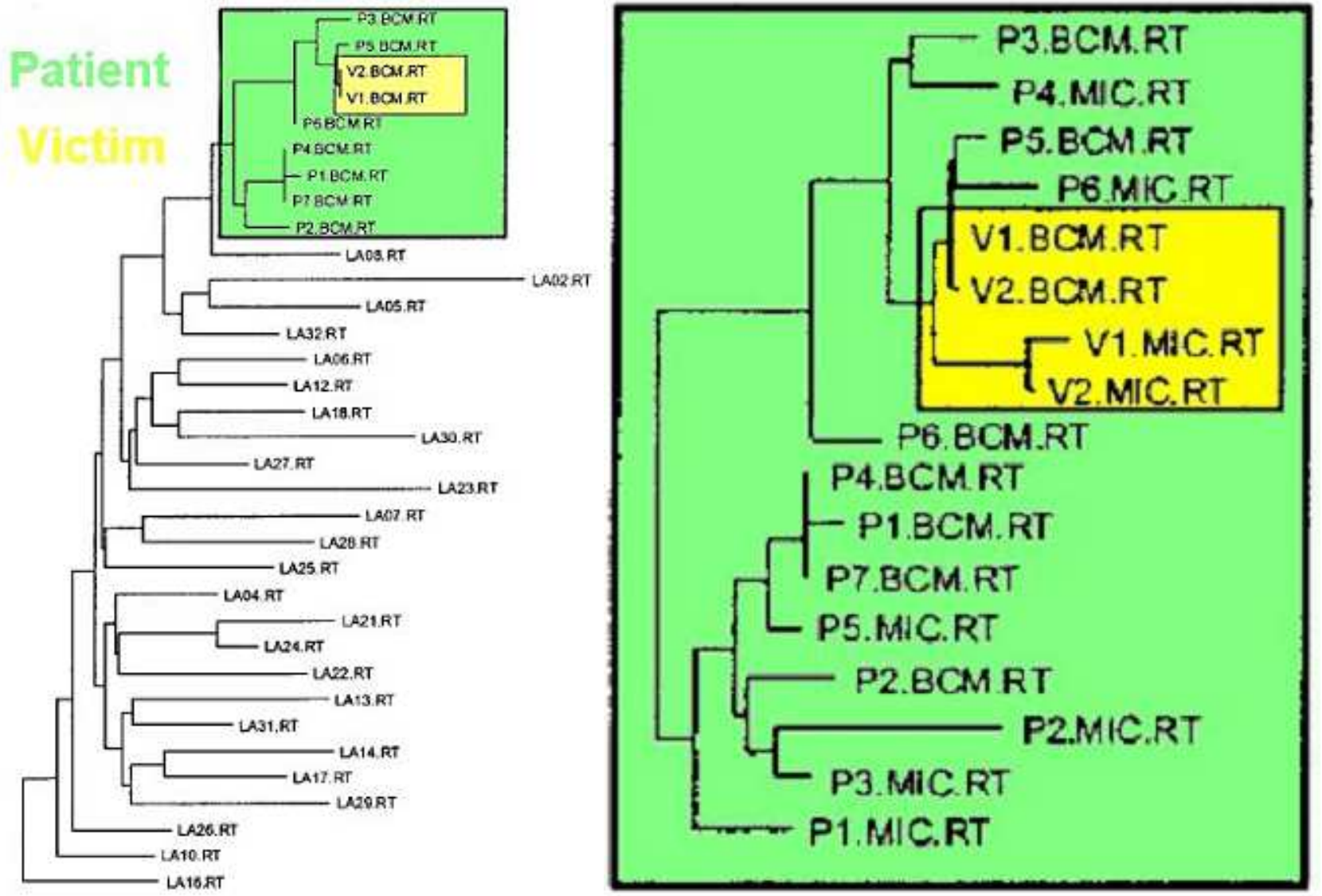
HIV has a high mutation rate, which can be used to trace paths of transmission.

Two people who got the virus from two different people will have very different HIV sequences.

Three different phylogenetic trees (including parsimony-based) were used to track changes in two genes in HIV (gp120 and RT). Multiple samples from the physician's patient, the woman and controls (non-related HIV+ people) were used.

In every reconstruction, the woman's sequences were found to be evolved from the patient's sequences.

This was the first time when phylogenetic analysis was used in court as evidence (cf. Metzker et al., 2002)


Patient
Victim
P3.BCM.RT
P5.BCM.RT
V2.BCM.RT
V1.BCM.RT
P6.BCM.RT
P4.BCM.RT
P1.BCM.RT
P7.BCM.RT
P2.BCM.RT
LA08.RT
LA02.RT
LA05.RT
LA32.RT
LA06.RT
LA12.RT
LA18.RT
LA30.RT
LA27.RT
LA23.RT
LA07.RT
LA28.RT
LA25.RT
LA04.RT
LA21.RT
LA24.RT
LA22.RT
LA13.RT
LA31.RT
LA14.RT
LA17.RT
LA29.RT
LA26.RT
LA10.RT
LA16.RT
P3.BCM.RT
P4.MIC.RT
P5.BCM.RT
P6.MIC.RT
V1.BCM.RT
V2.BCM.RT
V1.MIC.RT
V2.MIC.RT
P6.BCM.RT
P4.BCM.RT
P1.BCM.RT
P7.BCM.RT
P5.MIC.RT
P2.BCM.RT
P2.MIC.RT
P3.MIC.RT
P1.MIC.RT

Deriving Phylogenetic Trees

Aim:

Given a set of data (DNA, protein sequences, protein structure, etc.) that characterize different groups of organisms,

try to derive information about the relationships among the organisms in which they were observed.

The distance-based ("phenetic") approach:

Proceed by measuring a set of distances between (data provided for these) species, and generate the tree by a hierarchical clustering procedure.

Note: Hierarchical clustering is perfectly capable of producing a tree even in the absence of evolutionary relationships!

The character-based ("cladistic") approach:

Consider possible pathways of evolution, infer the features of the ancestor at each node, and choose an optimal tree according to some model of evolutionary change (maximum parsimony, maximum likelihood, or based on genealogy or homology).

2 Distance-based Phylogeny

These most intuitive methods of building phylogenetic trees begin with a set of distances d_{ij} between each pair (i, j) of sequences in the given dataset.

There are many ways of defining a distance.

For instance, given an analigment of two sequences i and j, the distance d_{ij} can be simply taken as the fraction f of sites u where residues x_u^i and x_u^j differ.

However, if one would like the distance to become very large as f tends to the fraction of differences expected by chance, the Jukes-Cantor distance can be used. For example:

$$d_{ij} = -\frac{3}{4}\, log(1 - f \times 4/3)$$

It tends to infinity as the equilibrium value of f (75% of residues different) is approached.

2.1 The Average Linkage (UPGMA) algorithm

[Sokal and Michener, 1958]

UPGMA = Unweighted Pair Group Method using arithmetic Averages

This is a hierarchical agglomerative (i.e. bottom-up) clustering algorithm: at each stage it amalgamates two clusters and creates a new node on the output tree.

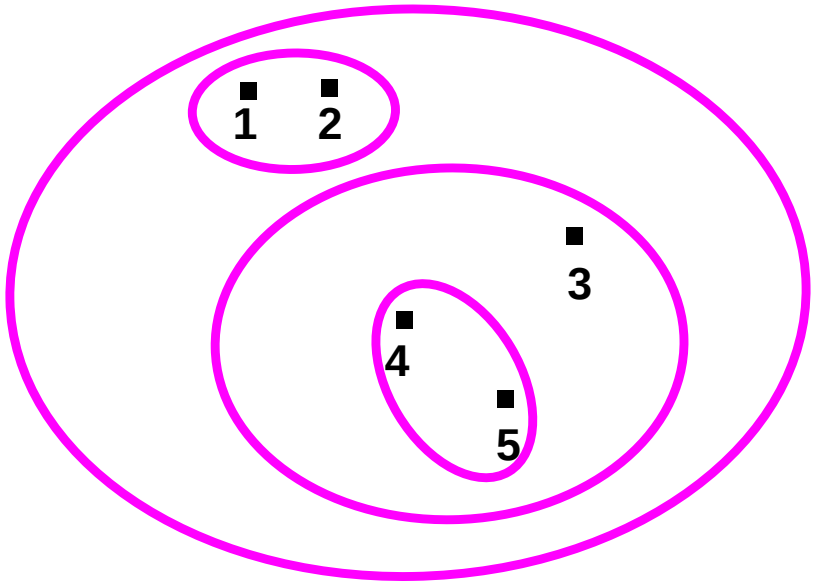
The distance between two clusters C_i and C_j is the average distance between pairs of sequences from each cluster:

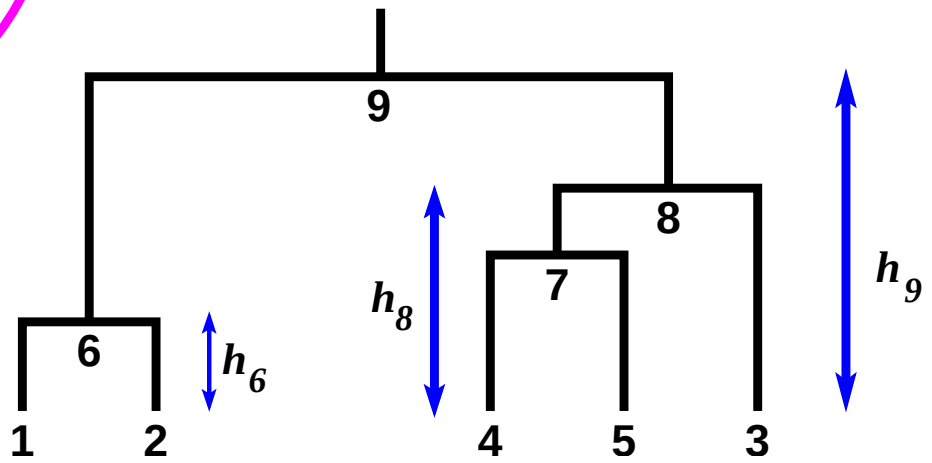
$$d_{ij} = \frac{1}{|C_i|\,|C_j|} \sum_{p \text{ in } C_i,\, q \text{ in } C_j} d_{pq}$$

Note: It can be shown that if C_k is the union of two clusters C_i and C_j, and if C_l is any other cluster, then:

$$d_{kl} = \frac{d_{il}\,|C_i| + d_{jl}\,|C_j|}{|C_i| + |C_j|}$$

UPGMA: Thw idea





$$h_6 = \frac{1}{2}d_{12},\ h_7 = \frac{1}{2}d_{45},\ h_8 = \frac{1}{2}d_{37},\ h_9 = \frac{1}{2}d_{68}$$

The UPGMA algorithm

Initialisation:

assign each sequence i to its own cluster C_i;
define one leaf of T for each sequence, and place it at height zero.

Iteration:

determine the two clusters i, j for which the mutual distance is minimal
(If there are several equidistant minimal pairs, pick one randomly.)
define a new cluster $C_k = C_i \cup C_j$, and compute d_{kl} for all l:

$$d_{kl} = \frac{d_{il}\,|C_i| + d_{jl}\,|C_j|}{|C_i| + |C_j|}$$

define a node k with daughter nodes i and j; place it at height $d_{ij}/2$
add C_k to the current clusters and remove C_i and C_j.

Termination:

when only two clusters C_i and C_j remain, place the root at height $d_{ij}/2$.

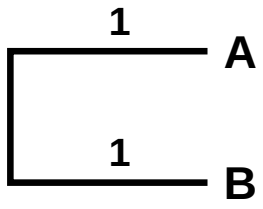
Complexity: space: $\mathcal{O}(n^2)$, time: $\mathcal{O}(n^3)$, where n is the number of sequences.

Note: The time complexity can be improved to $\mathcal{O}(n^2)$, by searching for the minimum (of distances) using ordered lists.

The UPGMA algorithm: Example

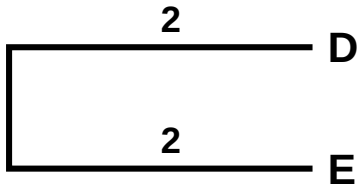
Xavier Declerc, Guy Henrard, UCL Belgium, INGI2368 course, 2005

	A	B	C	D	E
B	**2**				
C	4	4			
D	6	6	6		
E	6	6	6	4	
F	8	8	8	8	8



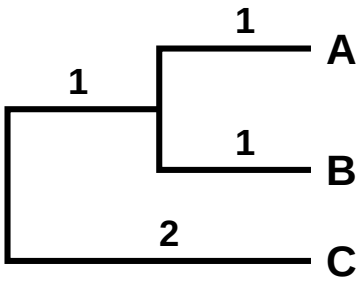
$d_{(AB),C} = \frac{1}{2}(d_{AC} + d_{BC}) = 4$
$d_{(AB),D} = 6$
$d_{(AB),E} = 6$
$d_{(AB),F} = 8$

	AB	C	D	E
C	4			
D	6	6		
E	6	6	**4**	
F	8	8	8	8



$d_{(DE),(AB)} = \frac{1}{2}(d_{D,(AB)} + d_{E,(AB)}) = 6$
$d_{(DE),C} = 6$
$d_{(DE),F} = 8$

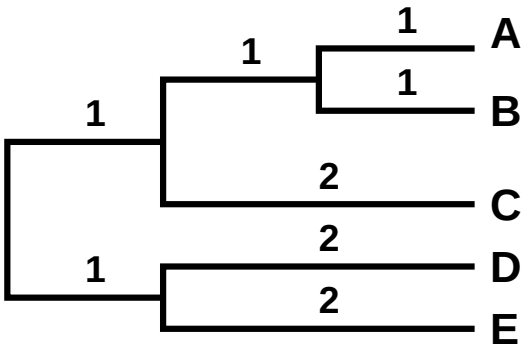
	AB	C	DE
C	**4**		
DE	6	6	
F	8	8	8



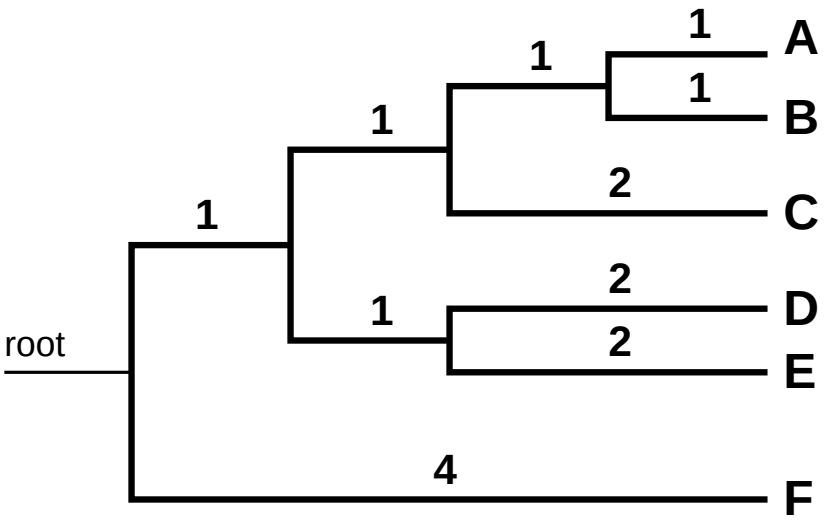
$d_{(ABC),(DE)} = \frac{1}{3}(2d_{(DE),(AB)} + d_{(DE),C}) = 6$
$d_{(ABC),F} = 8$

UPGMA example (cont'd)

	ABC	DE
DE	6	
F	8	8



	ABCDE
F	8



UPGMA specificity

as a hierarchical agglomerative clustering algorithm

UPGMA produces an ultrametric tree: the distance/height from each node in the tree to every one of its descendent leaves will be the same.

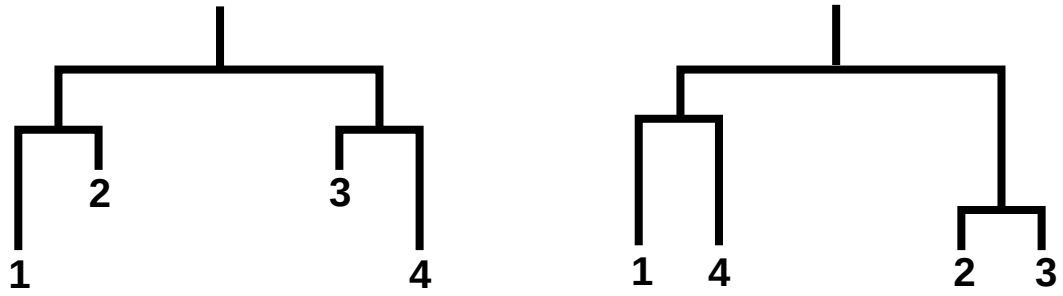
This corresponds to the so-called molecular clock assumption: mutations are generated with a constant rate along each path in the tree.

The ultrametric condition: The distances d_{ij} are ultrametric (i.e. they are generated by an ultrametric tree) if and only if for any triplet of sequences x_i, x_j , x_k, the distances d_{ij}, d_{jk}, d_{ik} are either all equal, or two are equal and the remaining one is smaller.

Note

If the input (distance data) submitted to the UPGMA algorithm are derived by additivity — i.e. summing the edge lengths/heights on connecting paths — in an ultrametric tree T, then UPGMA will reconstruct T correctly.

If the input data submitted to UPGMA is derived by additivity from a tree T which is not ultrametric, then UPGMA will produce a different tree (which is ultrametric).



2.2 The Neighbour-Joining algorithm

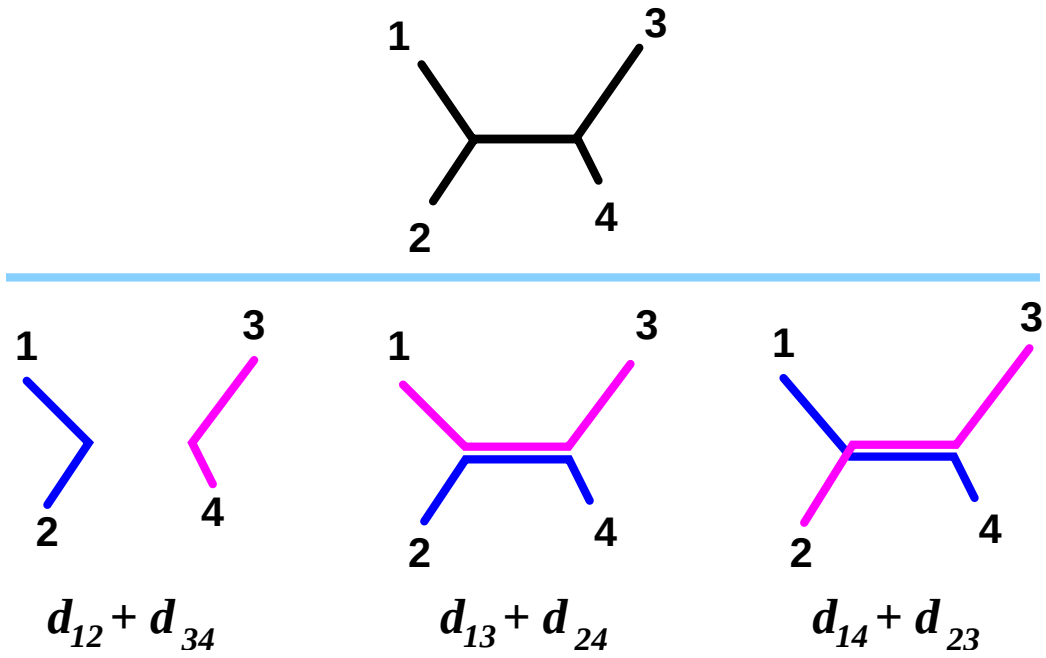
[Saitou and Nei, 1987] and [Studier and Keppler, 1988]

Neighbour-Joining, unlike UPGMA, produces unrooted trees. It is suitable for additive (or nearly additive) distance data.

The distances d_{ij} are additive if and only if the following condition holds:

Four-point condition

For every set of four leaves i, j, k and l, two of the distances $d_{ij} + d_{kl}$, $d_{ik} + d_{jl}$ and $d_{il} + d_{jk}$ must be equal and larger than the third.



Notes

1. The ultrametric property implies additivity. Obviously, there are additive trees for which the ultrametric property doesn't hold.

2. It is shown in Ch. 8 [Durbin et al. 1998] that a certain type of maximum likelihood distance measure on genomic data would be expected to give (approximate) additivity, in the limit of a large amount of data.

The Neighbour-Joining algorithm: main idea

The algorithm proceeds repetitively:

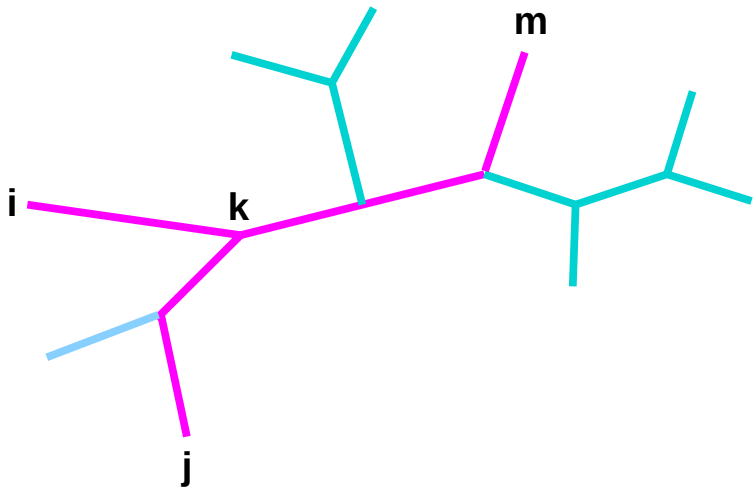
At each iteration it finds a pair of neighbouring leaves, i.e. leaves that have the same parent node k.

The distance from the node k to a leaf m is:

$$d_{km} = \frac{1}{2}(d_{im} + d_{jm} - d_{ij})$$

which is due to additivity: $d_{im} = d_{ik} + d_{km}$, $d_{jm} = d_{jk} + d_{km}$ and $d_{ij} = d_{ik} + d_{kj}$.

Then the algorithm discards i, and j from the set of leaf nodes and instead adds the node k.



The number of leaves decreases by one at each iteration, until we get down to a single pair of leaves.

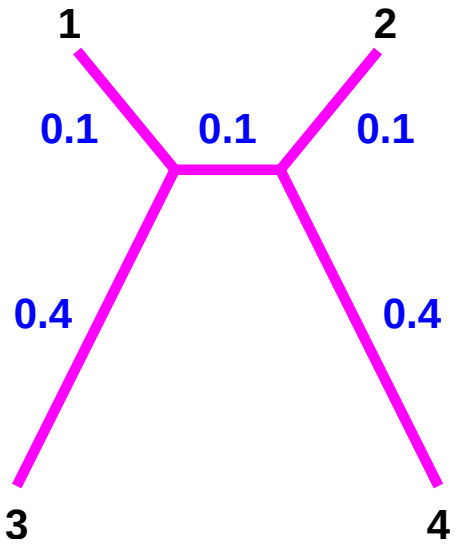
How to determine neighbouring leaves

Note that the closest pair of leaves are not necessarily neighbouring leaves (due to long edges).

To eliminate the effect of long edges, subtract the averages of distances to all other leaves, therefore define

$$D_{ij} = d_{ij} - (r_i + r_j) \text{ where } r_i = \frac{1}{|L| - 2} \sum_{k \in L} d_{ik}.$$

Minimizing on D_{ij} (instead of d_{ij}) is guaranteed to find neighbouring leaves. (See proof in the Appendix of Ch. 7, Durbin et al., 1998.)



The Neighbour-Joining algorithm

Initialisation:

define T to be the set of leaf nodes, one for each given sequence, and let $L = T$

Iteration:

pick a pair i, j in L so that D_{ij} defined by $D_{ij} = d_{ij} - (r_i + r_j)$, where $r_i = \frac{1}{|L|-2} \sum_{k \in L} d_{ik}$, is minimal

define a new node k and set $d_{km} = \frac{1}{2}(d_{im} + d_{jm} - d_{ij})$ for all m in L

add k to T with edges of lengths $d_{ik} = \frac{1}{2}(d_{ij} + r_i - r_j)$ and $d_{jk} = d_{ij} - d_{ik} = \frac{1}{2}(d_{ij} - r_i + r_j)$, joining k to i and j, respectively

remove i and j from L and add k

Termination:

When L consists of two leaves i and j, add the remaining edge between i and j, with length d_{ij}

Complexity: time: $\mathcal{O}(n^3)$, space: $\mathcal{O}(n^2)$, with n the number of leaf nodes.

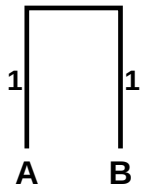
The Neighbour-Joining algorithm: Example

Xavier Declerc, Guy Henrard, UCL Belgium
INGI2368 course, 2005

d	A	B	C	D	E	F
A		2	4	6	6	8
B	2		4	6	6	8
C	4	4		6	6	8
D	6	6	6		4	8
E	6	6	6	4		8
F	8	8	8	8	8	

r	
A	6.5
B	6.5
C	7
D	7.5
E	7.5
F	10

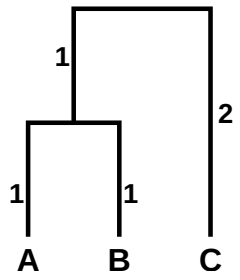
D	A	B	C	D	E	F
A						
B	$\mathbf{-11}$					
C	-9.5	-9.5				
D	-8	-8	-8.5			
E	-8	-8	-8.5	-11		
F	-8.5	-8.5	-9	-9.5	-9.5	



d	C	D	E	F	AB
C		6	6	8	3
D	6		4	8	5
E	6	4		8	5
F	8	8	8		7
AB	3	5	5	7	

r	
C	7.67
D	7.67
E	7.67
F	10.33
AB	6.67

D	C	D	E	F	AB
C					
D	-9.33				
E	-9.33	-11.33			
F	-10	-10	-10		
AB	$\mathbf{-11.33}$	-9.33	-9.33	-10	

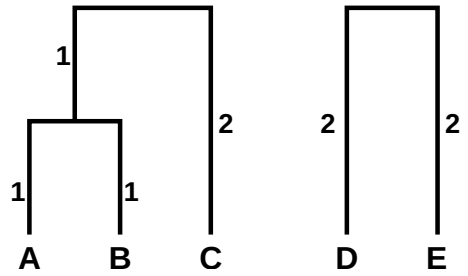


Neighbour-Joining example (cont'd)

d	D	E	F	ABC
D		4	8	4
E	4		8	4
F	8	8		6
ABC	4	4	6	

r	
D	8
E	8
F	11
ABC	7

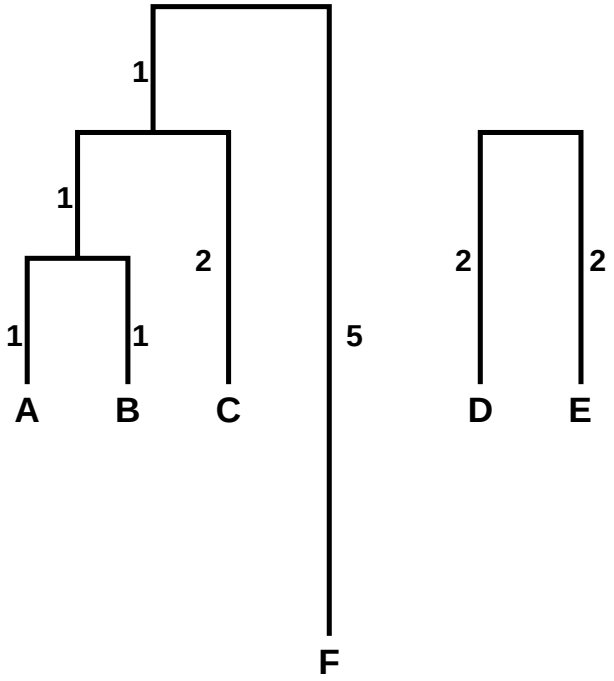
D	D	E	F	ABC
D				
E	$\mathbf{-12}$			
F	-11	-11		
ABC	-11	-11	-12	



d	F	ABC	DE
F		6	6
ABC	6		2
DE	6	2	

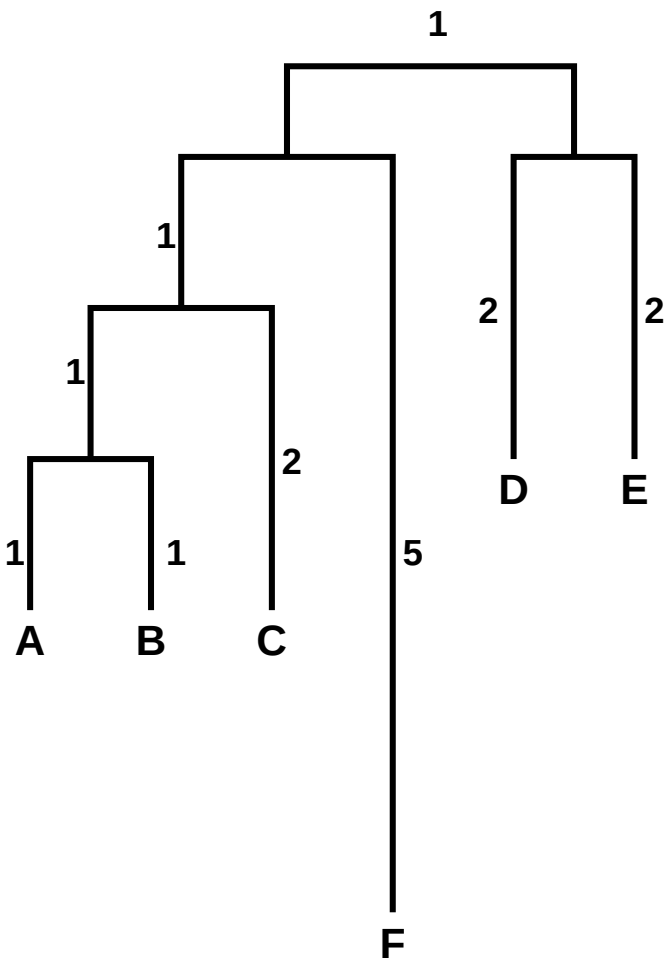
r	
F	12
ABC	8
DE	8

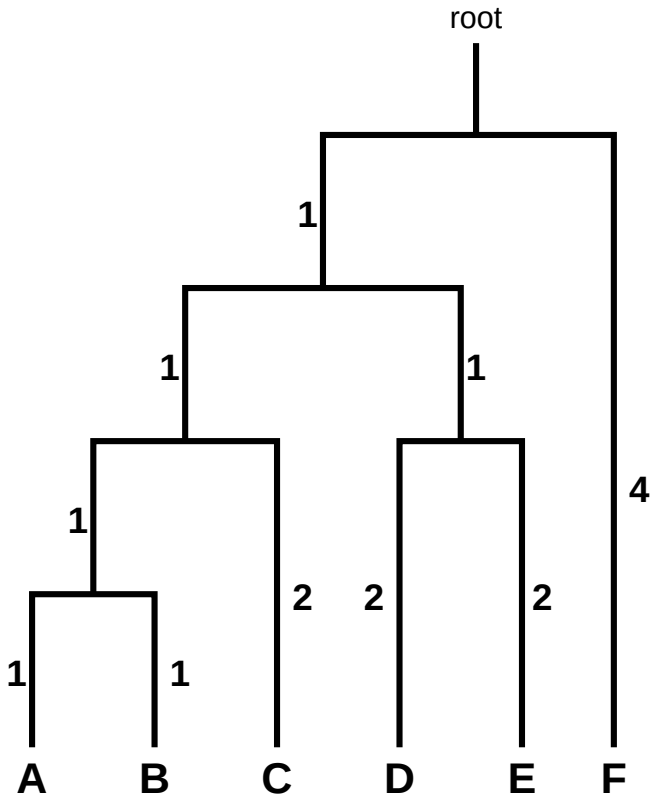
D	F	ABC	DE
F			
ABC	$\mathbf{-14}$		
DE	-14	-14	



Neighbour-Joining example (cont'd)

the final unrooted tree





the same tree rooted at the midpoint of the longest path between leaf nodes

Rooting (unrooted) trees

Finding the root of an unrooted tree can be done by adding an outgroup, a species that is known to be more distantly related to each of the remaining species than they are to each other. The point in the tree where the edge to the outgroup joins is therefore the best candidate for the root position.

Another strategy is to pick the midpoint of the longest chain of consecutive edges.

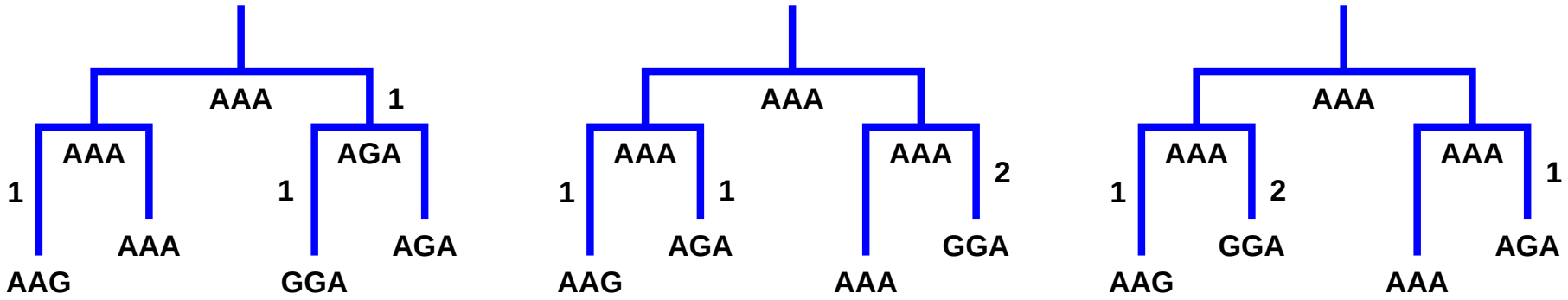
3 Character-based Phylogeny

Aim:

Given a set of sequences, build a (binary) tree
labeling its leaves by these input sequences, and
assigning its internal nodes similar sequences
so as to explain their generation using a minimal number of substitutions. (This number will be called the parsimony score.)

Note:

More generally, instead of sequences we can consider objects, each one of them being characterised by a string of characteristics.



3.1 Small Parsimony

Problem:

Given a tree T, each leaf of whom is labeled by an m-character string, label the internal nodes of T with m-character strings so as to minimize the cost (i.e. number of substitutions) needed to derive strings from their ancestors.

Note:

We can assume that every leaf is labeled by a single character, because the characters in the string are independent.

Traditional parsimony:

Use the Hamming distance to score substitutions: $d_H(v, w) = 0$ iff $v = w$, and $d_H(v, w) = 1$ otherwise.

Weighted parsimony:

Use a $l \times l$ scoring matrix (l is the size of the character alphabet).

3.1.1 Weighted Parsimony
Sankoff's Algorithm (1975)

Initialisation: **index the nodes of the tree in bottom-up manner;**
assuming that there are n leaves, the root node will have the index $2n-1$.

Recursion:

compute $S_k(a)$ for all a as follows:

if k is leaf node:
for $a = x_u^k$ set $S_k(a) = 0$; otherwise $S_k(a) = \infty$

if k is not a leaf node:
compute $S_i(a), S_j(a)$ for all a at the daughter nodes i, j, and define

$$S_k(a) = min_b(S(a,b) + S_i(b)) + min_c(S(a,c) + S_j(c))$$

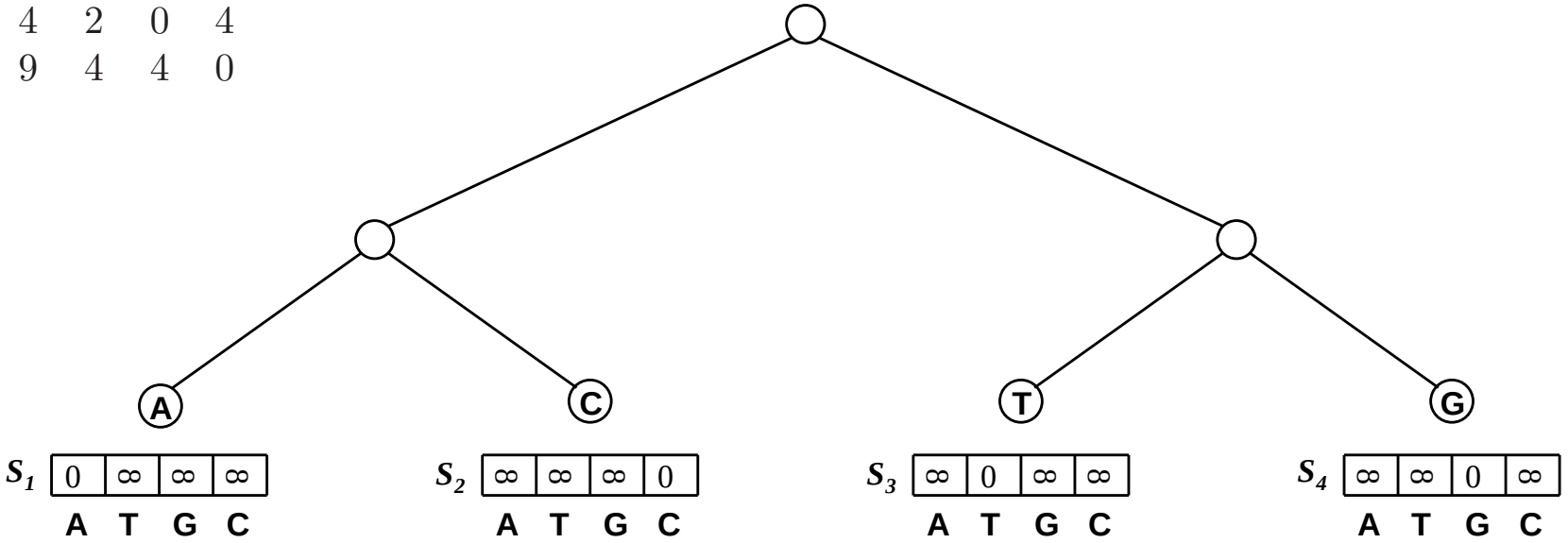
Termination: **the minimal cost of the tree is $min_a S_{2n-1}(a)$**

Traceback (one solution): **for the root node: $argmin_a(S_{2n-1}(a)$, and**
then for $k = 2n-1, \ldots, n+1$:
left$_k(a) = argmin_b(S(a,b) + S_i(b))$ **and** ***right***$_k(a) = argmin_c(S(a,c) + S_j(c))$.

Complexity: **space: $\mathcal{O}(nl)$, time: $\mathcal{O}(nl^2)$, with l the size of the character alphabet.**

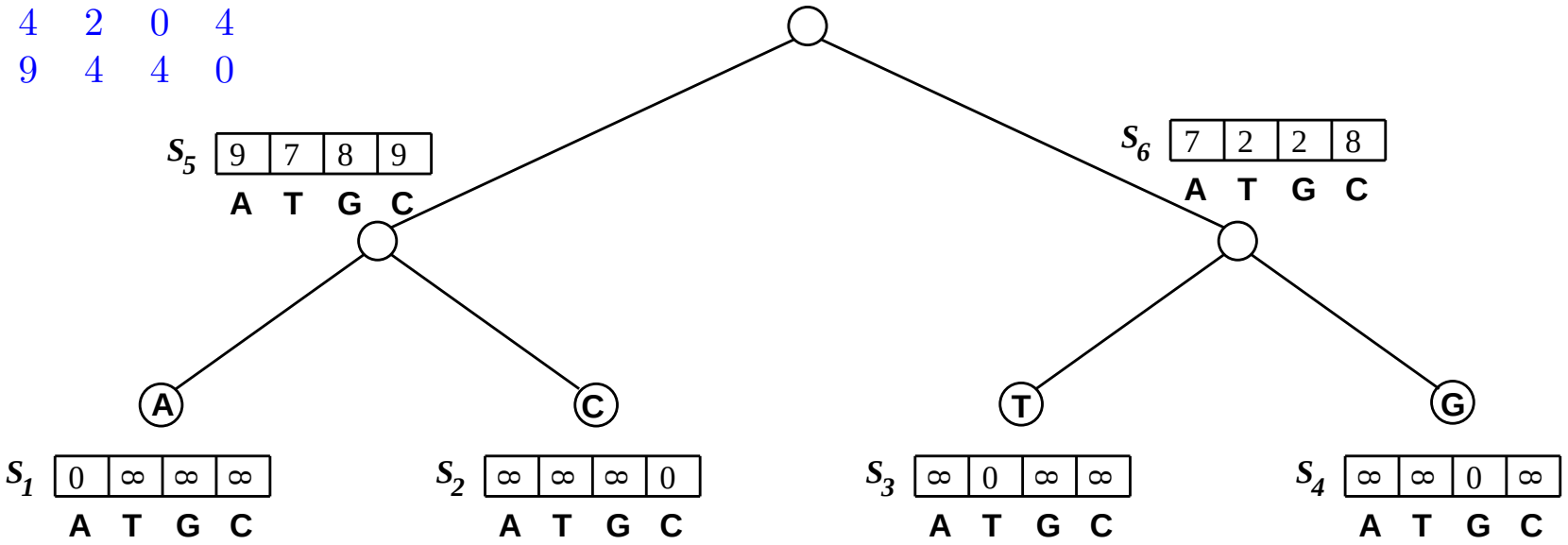
Sankoff's Algorithm: Example

S	A	T	G	C
A	0	3	4	9
T	3	0	2	4
G	4	2	0	4
C	9	4	4	0



Sankoff's Algorithm: Example (cont'd)

S	A	T	G	C
A	0	3	4	9
T	3	0	2	4
G	4	2	0	4
C	9	4	4	0



$S_5(\mathsf{A}) = S(\mathsf{A}, \mathsf{A}) + S(\mathsf{A}, \mathsf{C}) = 0 + 9 = 9$
$S_5(\mathsf{T}) = S(\mathsf{T}, \mathsf{A}) + S(\mathsf{T}, \mathsf{C}) = 3 + 4 = 7$
...

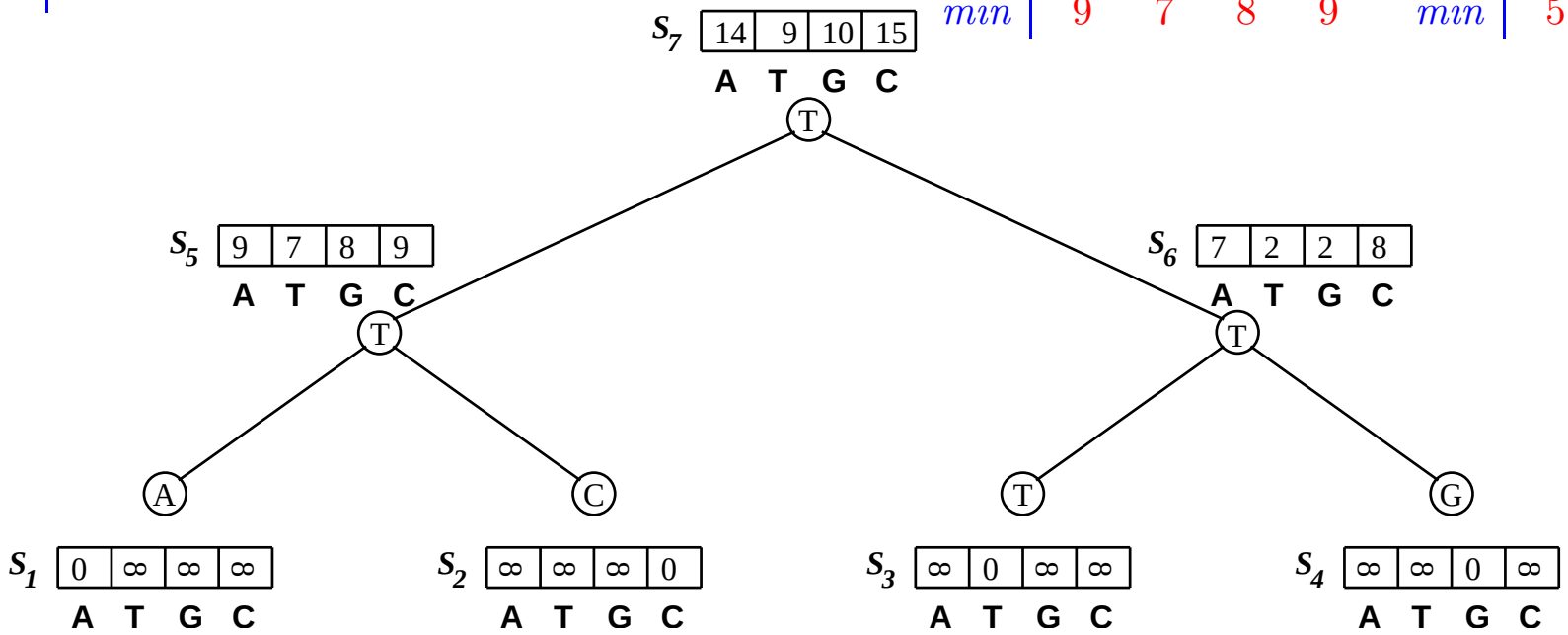
LC: $S_{7,5} = S + [\ S_5^T\ S_5^T\ S_5^T\ S_5^T\ ]$
$S_{7,6} = S + [\ S_6^T\ S_6^T\ S_6^T\ S_6^T\ ]$

Sankoff's Algorithm: Example (cont'd)

S	A	T	G	C
A	0	3	4	9
T	3	0	2	4
G	4	2	0	4
C	9	4	4	0

$S_{7,5}$	A	T	G	C
A	9	12	13	18
T	10	7	9	11
G	12	10	8	12
C	18	13	13	9
min	9	7	8	9

$S_{7,6}$	A	T	G	C
A	7	10	11	16
T	5	2	4	6
G	6	4	2	6
C	17	12	17	8
min	5	2	2	6



$$S_7(\mathsf{A}) = min_b(S(b,\mathsf{A}) + S_5(b)) + min_c(S(c,\mathsf{A}) + S_6(c)) = 9 + 5 = 14$$

...

3.1.2 Traditional Parsimony
Fitch's Algorithm (1971)

Initialisation: index the nodes of the trees in bootom up manner;
set $k = 2n - 1$ (the root node), and initialize the parcimony cost $C = 0$.

Recursion for tree adnotation:
to obtain the set R_k

- if k is leaf node:
  set $R_k = \{x_u^k\}$
- if k is not a leaf node:
  compute R_i, R_j for the daughter nodes i, j of k, and

  set $R_k = R_i \cap R_j$ if this intersection is not empty, or
  else set $R_k = R_i \cup R_j$ and increment C

Termination of tree adnotation:
the minimal cost of the tree is C

Complexity: $\mathcal{O}(nl)$, where l is the size of the character alphabet.

Fitch's Algorithm (cont'd)

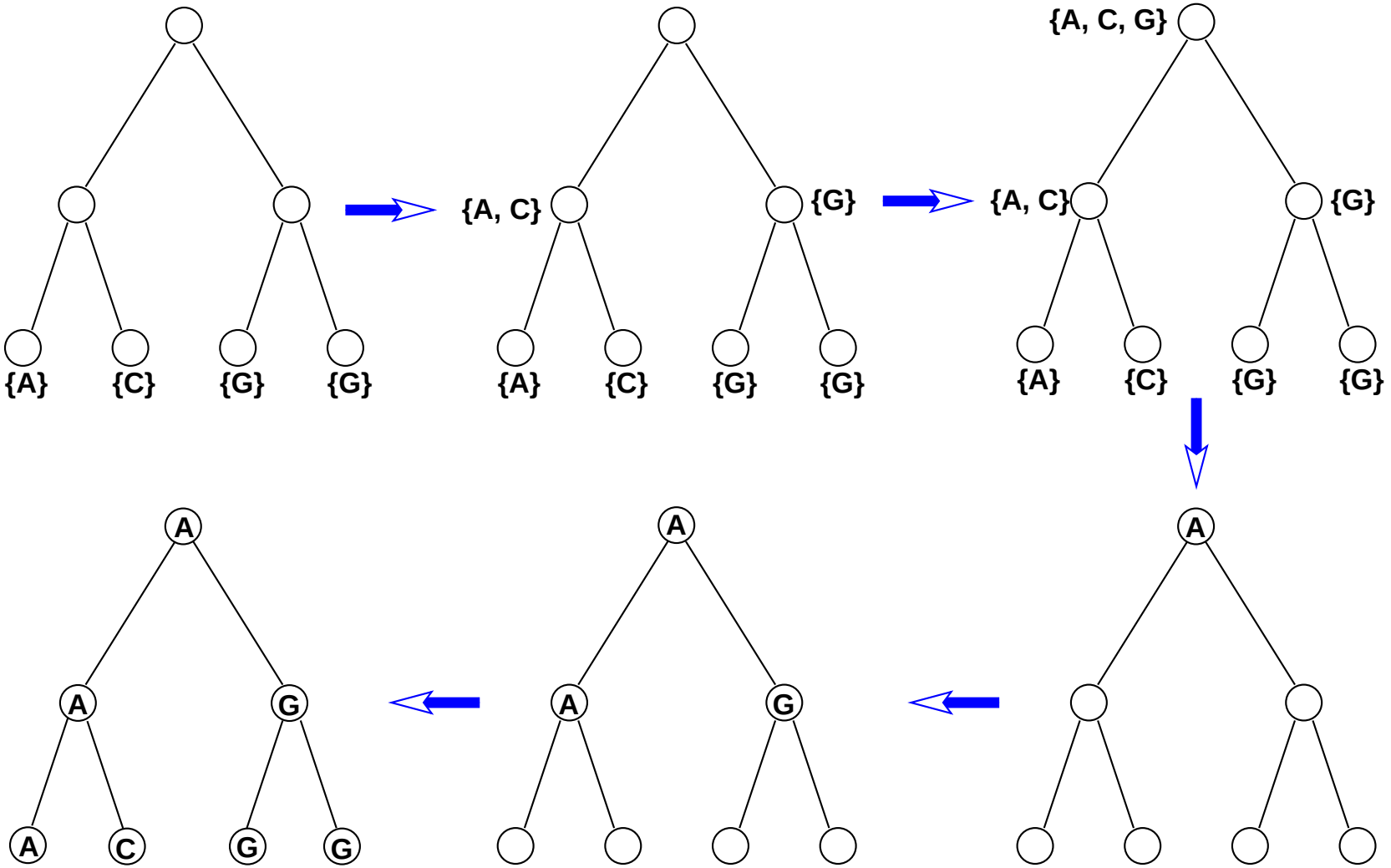
Traceback (one solution):

for the root node, choose arbitrarily one residue from R_{2n-1}, then proceed down the tree:

having chosen a residue from the set R_k,

- choose the same residue from the daughter set R_i if possible, otherwise choose at random a residue from R_i;
- similarly for the daughter set R_j.

Fitch's Algorithm: Example

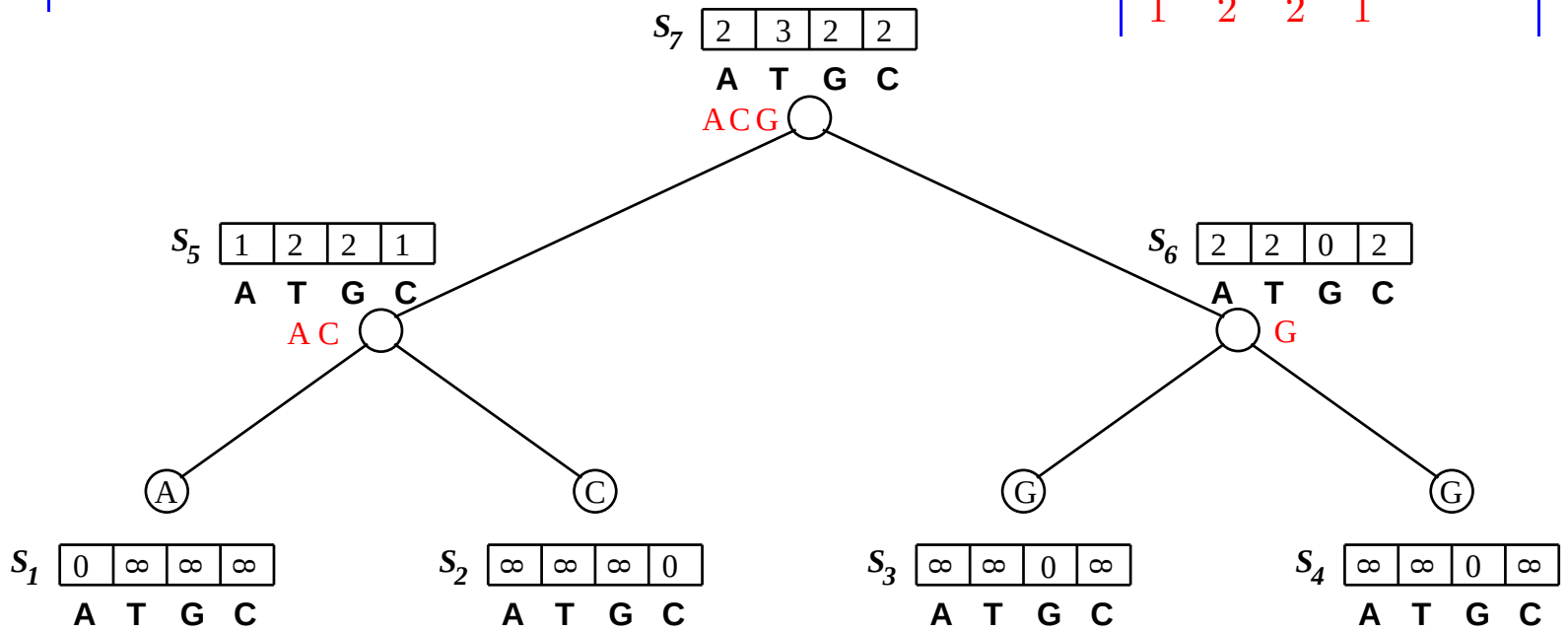


The same example, solved by Sankoff's algorithm

S	A	T	G	C
A	0	1	1	1
T	1	0	1	1
G	1	1	0	1
C	1	1	1	0

$S_{7,5}$	A	T	G	C
A	1	2	2	2
T	3	2	3	3
G	3	3	2	3
C	2	2	2	1
	1	2	2	1

$S_{7,6}$	A	T	G	C
A	2	3	3	3
T	3	2	3	3
G	1	1	0	1
C	3	3	3	2
	1	1	0	1



Note

It can be shown that, when using the Hamming distance,

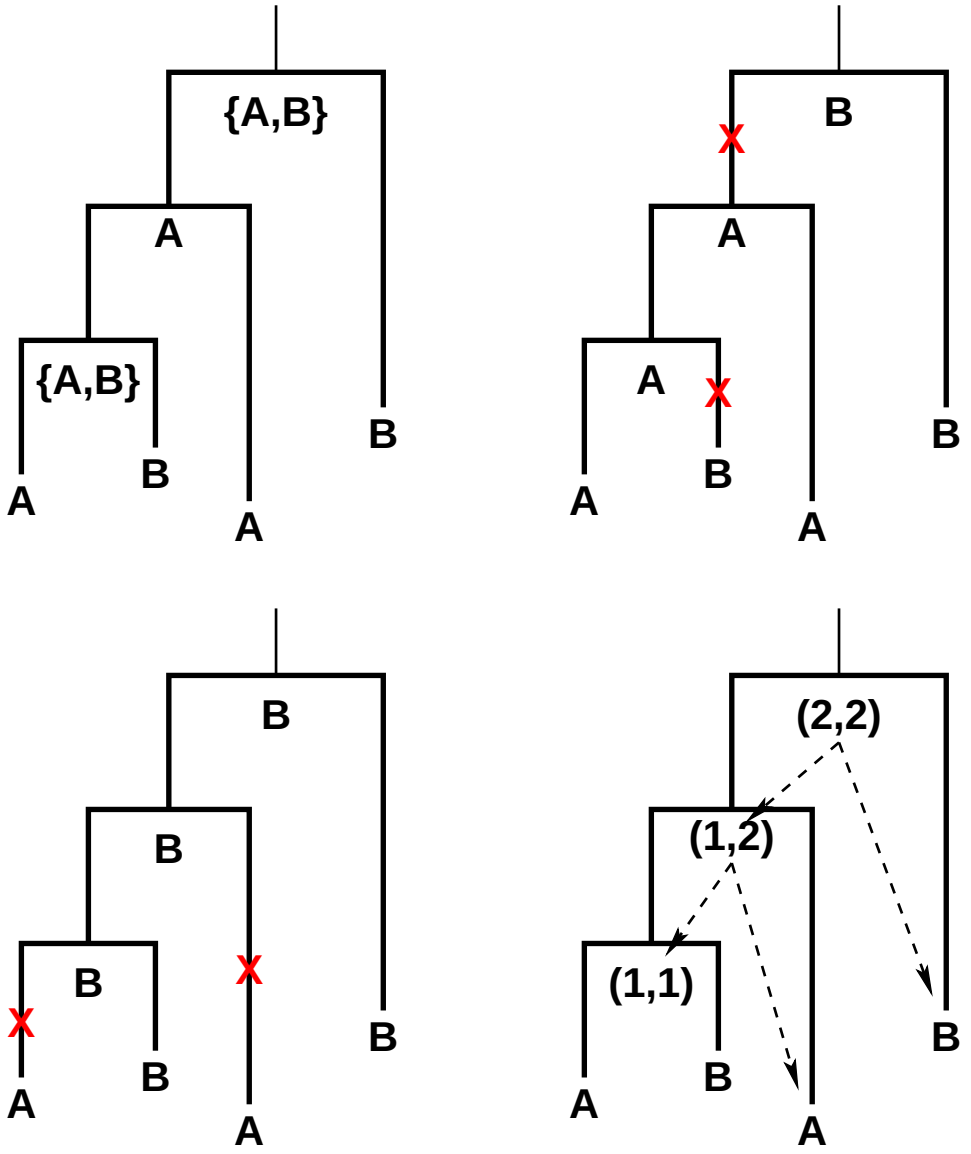
- **both algorithms (Sankoff and Fitch) compute the same parsimony score.**
- **unlike Sankoff's algorithm,**
  backtracing in Fitch's algorithm cannot produce all optimal trees; for an exemplification, see the next slide.
  (An improvement is suggested in Durbin et al., pag. 176.)

A problem with backtracing in Fitch's algorithm

The upper left tree cannot yield the bottom left one.

See right bottom for how that tree can be obtained by (using backtracing in) Sankoff's algorithm.

Note: The upper right tree and another onother, similar one, constitute the output of Fitch's algorithm.



3.2 Large Parsimony

Problem:

Given n strings, find a (binary) tree T

- labelling its leaves with these input strings, and
- assigning its internal nodes similar strings
- so as to minimize the parsimony score over all possible trees and all possible labelings of the internal nodes.

Note:

The Large Parsimony problem is $\mathcal{NP}$-complete.

If n is small, one can explore all tree topologies with n leaves, solve the Small Parsimony problem for each topology, and select the best solution.

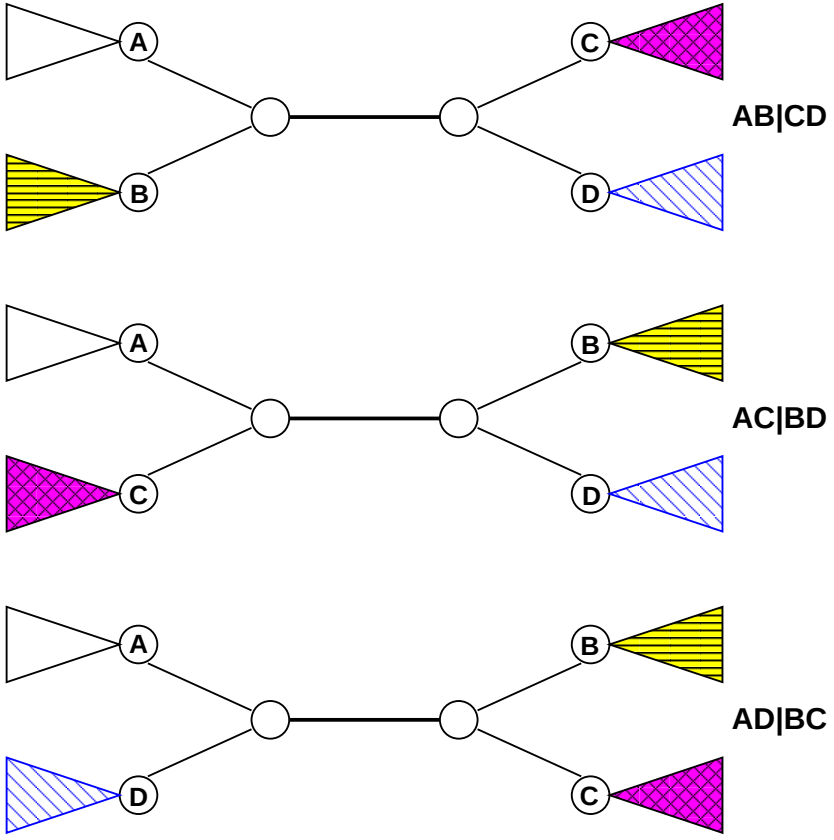
As the number of possible topologies grows very fast — $((2n-3)!!$ rooted trees, respectively $(2n-5)!!$ unrooted trees —, we must use local search heuristics.

3.2.1 The greedy approach to large parsimony

3.2.1.1 Nearest Neighbour Interchange algorithm

[David Robinson, 1971] [Jones & Pevzner, 2004]

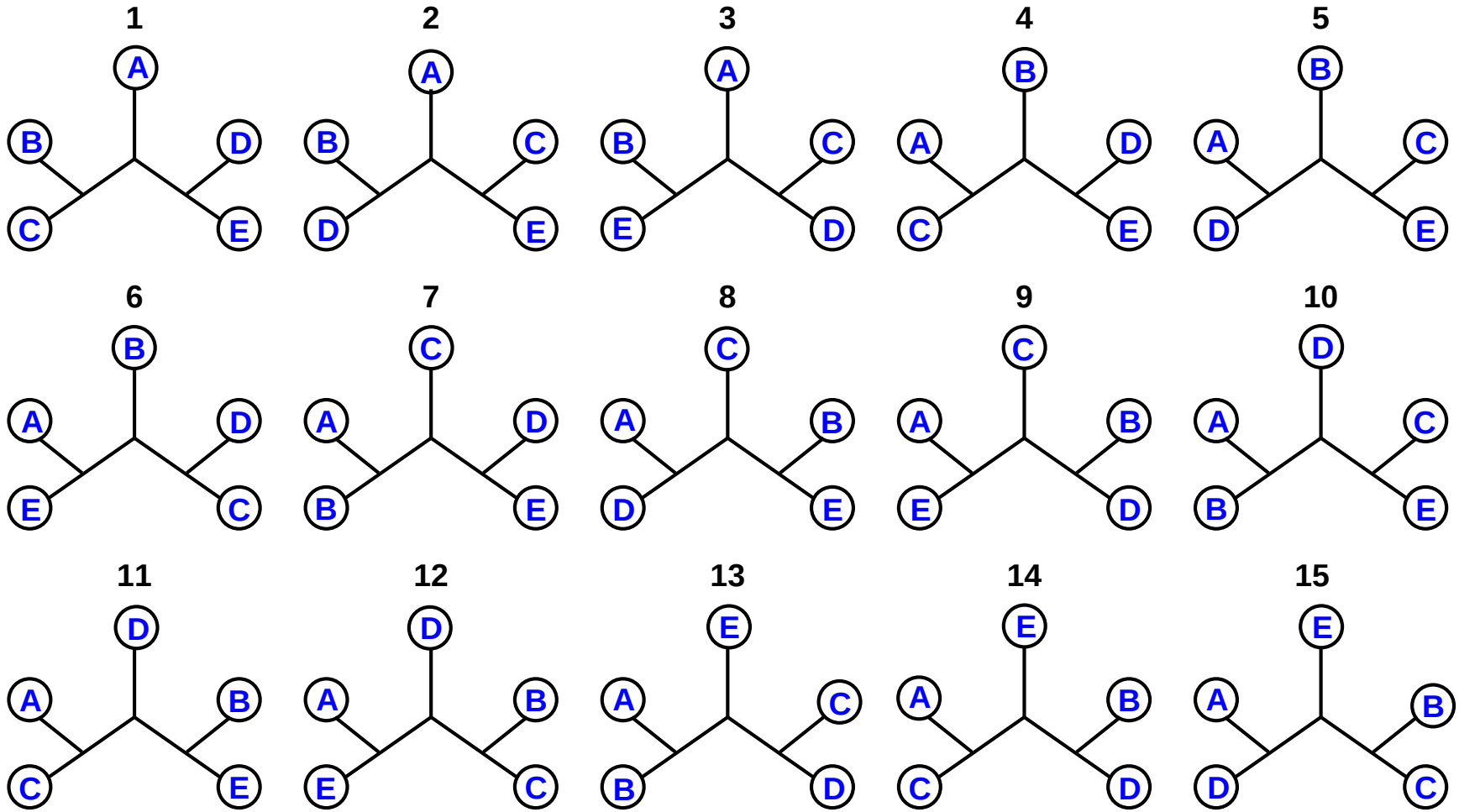
Three ways of combining the four subtrees connected to an internal edge of a binary tree.



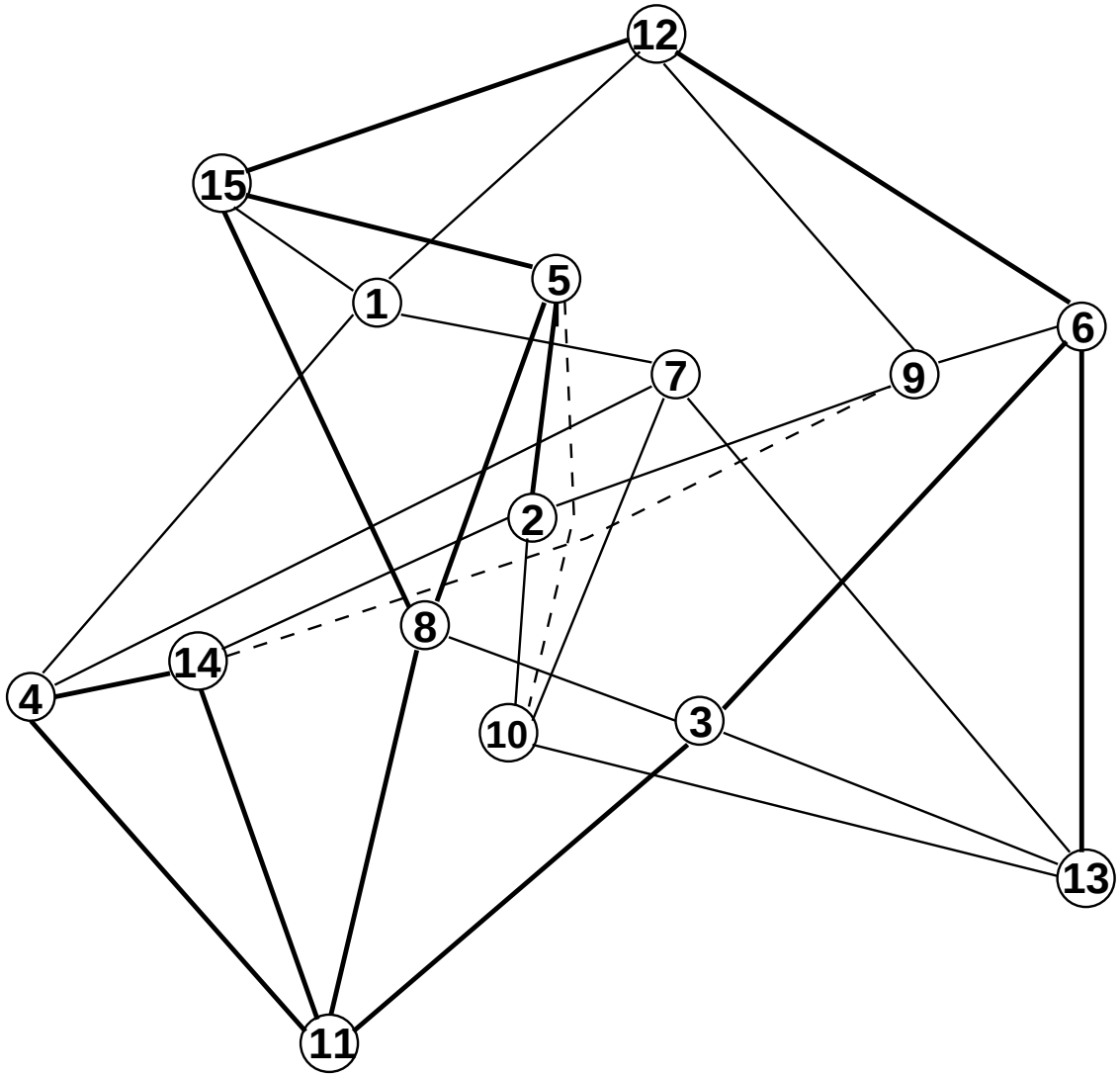
The Nearest Neighbour Interchange algorithm

- Start from an arbitrary tree T;
- Move from one tree to another by nearest neighbour interchange, as shown in the previous figure,
  if such a move provides the best improvement in the parsimony score — computed using for instance Sankoff's algorithm — among all nearest neighbours of the tree T.
- Note:
  This algorithm is not guaranteed to find the overall best tree.

All the 5-leaf binary trees...



...and the stereo representation of the graph of all 5-leaf trees, in which two trees (represented as vertices) are connected iff they are interchangeable by a single nearest neighbour interchange operation



3.2.1.2 Another greedy strategy for large parsimony:

Build up the tree by adding edges one at a time [Felsenstein, 1981]

- Three of the input strings are chosen randomly and placed on an unrooted tree which has 3 leaves (see slide #4).
- Another input string is then chosen and added to the edge that gives the best score for the tree of the four strings. Repeat this step until the tree is complete.
- This is not guaranteed to find the overall best tree, and indeed adding the strings in different orders can yield different final trees.

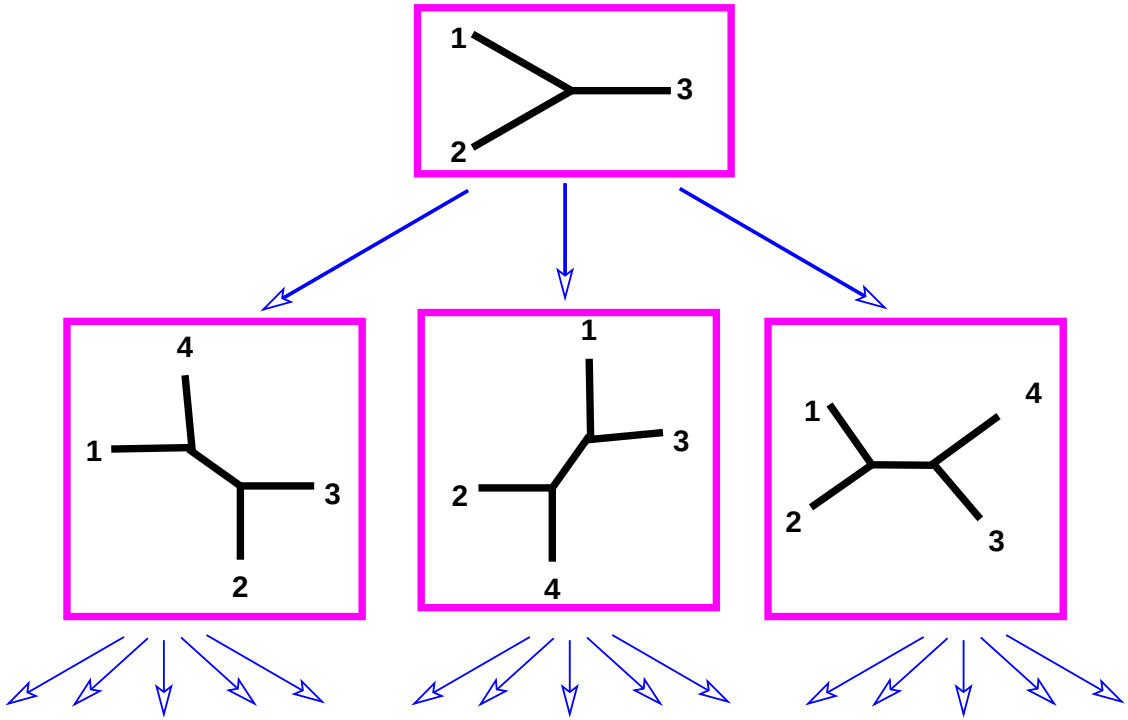
3.2.2 The branch-and-bound approach to large parsimony

- Systematically search through the space of unrooted trees having increasing numbers of leaves, but abandon a particular avenue of tree building whenever the current incomplete tree has a cost exceeding the smallest cost obtained so far for a complete tree.

  (For technical details see *Bilogical Sequence Analysis*, R. Durbin et al., 1998, p. 178–179.)

- This method can save a great deal of searching and is guaranteed to find the overall best tree.

Searching through the space of unrooted trees



When applying the branch-and-bound approach to solve the large parsimony problem, the bottom nodes of this search space are n-leaf trees.

The next level in this search space are 5-leaf unrooted trees (as shown on a previous slide).

How much should one trust the phylogenetic trees?

The bootstrap (assessing) method

[Felsenstein, 1985]

Given a dataset consisting of an alignment of sequences, an artificial dataset of the same size is generated by picking columns from the alignment at random with replacement. (A given column in the original dataset can therefore appear several times in the artificial dataset.) The tree building algorithm is then applied to this new dataset.

The whole selection and tree building procedure is repeated some number of times (typically of the order of 1000 times).

The frequency with which a chosen phylogenetic feature appears is taken to be a measure of the confidence we can have in this feature.

For certain probabilistic models (see Durbin et al., Ch. 8) the boostrap frequency of a phylogenetic feature F can be shown to approximate the posterior distribution $P(F \mid \text{data})$.

4 Simultaneous Phylogeny and Multiple Sequence Alignment

4.1 Sankoff & Cedergren gap-substitution algorithm (1983)

Sankoff & Cedergren's algorithm is guaranteed to find ancestral sequenes, and alignments of them and the leaf sequences, that together minimise a tree-based, parsimony-type cost.

The minimum cost $\alpha_{i_1,i_2,...,i_N}$ of an alignment ending with $x^1_{i_1}, x^2_{i_2}, ..., x^N_{i_N}$ is computed by multidimensional dynamic programming, using the recurrence relation

$$\alpha_{i_1,i_2,...,i_N} = min_{\Delta_1+...+\Delta_N>0}\{\alpha_{i_1-\Delta_1,i_2-\Delta_2,...,i_N-\Delta_N}+\sigma(\Delta_1 \cdot x^1_{i_1}, \Delta_2 \cdot x^2_{i_2}, \ldots, \Delta_N \cdot x^N_{i_N})\}$$

where $\Delta_i \cdot x = x$ if $\Delta_i = 1$, and $\Delta_i \cdot x = -$ if $\Delta_i = 0$.

σ is the weighted parsimony cost for aligning a set of symbols of the alphabet extended with the gap symbol '$-$'.

Note

The recurrence relation:

$$\alpha_{i_1,i_2,\ldots,i_N} = min_{\Delta_1+\ldots+\Delta_N>0}\{\alpha_{i_1-\Delta_1,i_2-\Delta_2,\ldots,i_N-\Delta_N} + \sigma(\Delta_1 \cdot x^1_{i_1}, \Delta_2 \cdot x^2_{i_2}, \ldots, \Delta_N \cdot x^N_{i_N})\}$$

where $\Delta_i \cdot x = x$ if $\Delta_i = 1$, and $\Delta_i \cdot x = -$ if $\Delta_i = 0$

is the condensed form of the (more intuitive) relation

$$\alpha_{i_1,\ldots,i_N} = min \begin{cases} \alpha_{i_1-1,i_2-1,\ldots,i_N-1} & + \sigma(x^1_{i_1}, x^2_{i_2}, \ldots, x^N_{i_N}), \\ \alpha_{i_1,i_2-1,\ldots,i_N-1} & + \sigma(-, x^2_{i_2}, \ldots, x^N_{i_N}), \\ \alpha_{i_1-1,i_2,\ldots,i_N-1} & + \sigma(x^1_{i_1}, -, \ldots, x^N_{i_N}), \\ & \vdots \\ \alpha_{i_1-1,i_2-1,\ldots,i_N} & + \sigma(x^1_{i_1}, x^2_{i_2}, \ldots, -), \\ \alpha_{i_1,i_2,\ldots,i_N-1} & + \sigma(-, -, \ldots, x^N_{i_N}), \\ & \vdots \\ \alpha_{i_1,i_2-1,\ldots,i_N} & + \sigma(-, x^2_{i_2}, \ldots, -), \\ & \vdots \end{cases}$$

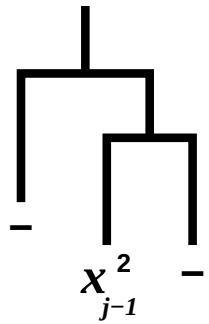
Computation of σ

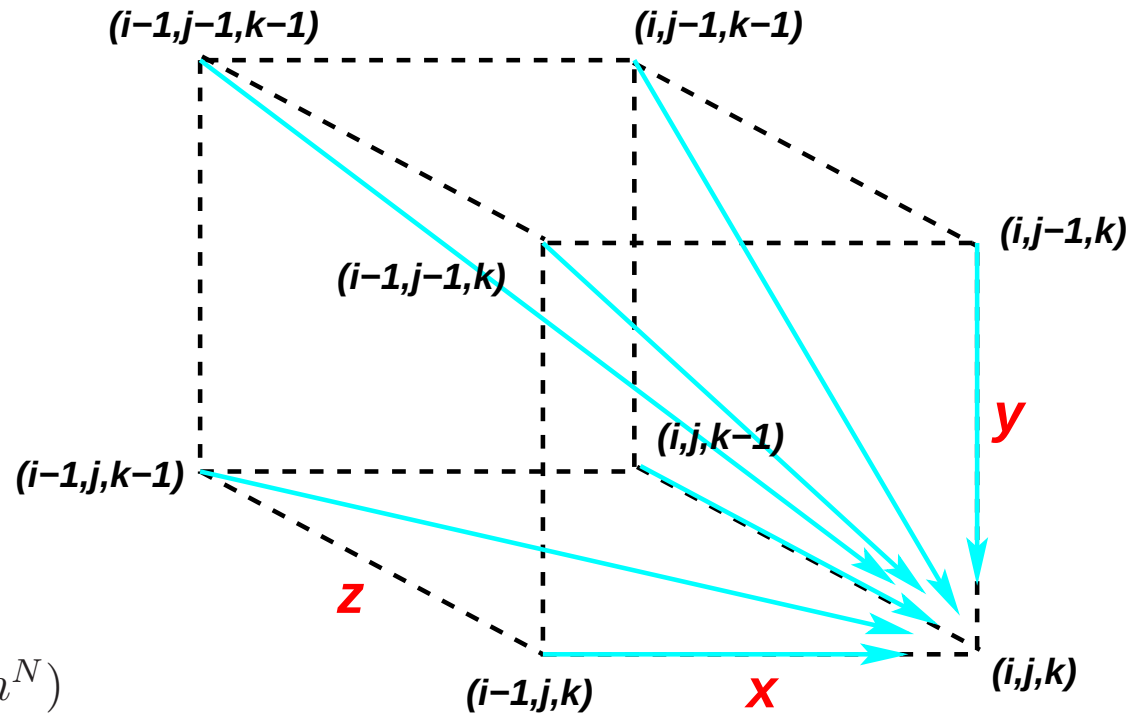
$\sigma(\Delta_1 \cdot x^1_{i_1}, \Delta_2 \cdot x^2_{i_2}, \ldots, \Delta_N \cdot x^N_{i_N})$ **can be calculated by an upward pass through the tree, using the weighted parsimony (Sankoff's) algorithm, where** $S(a, b)$ **is now defined also when one or both arguments are '–'.**

When applying Sankoff's algorithm, the (labels of the) leaves of the tree are assigned according to the DP transition, as follows:

- **if 1 is subtracted from a coordinate, the relevant leaf is assigned the preceding character in the input sequence;**
- **if the coordinate is unchanged, its leaf is assigned a '–'.**

For instance, the transition from $(i, j-1, k)$ **to** (i, j, k) **is assigned the following (labeling for the leaves of the) tree:**





Complexity

Space complexity: $\mathcal{O}(m^N)$

where N is the number of sequences, and m is the length of the sequences.

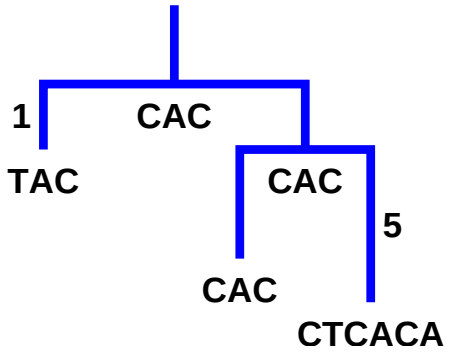
Time complexity: $\mathcal{O}(lN2^N m^N)$

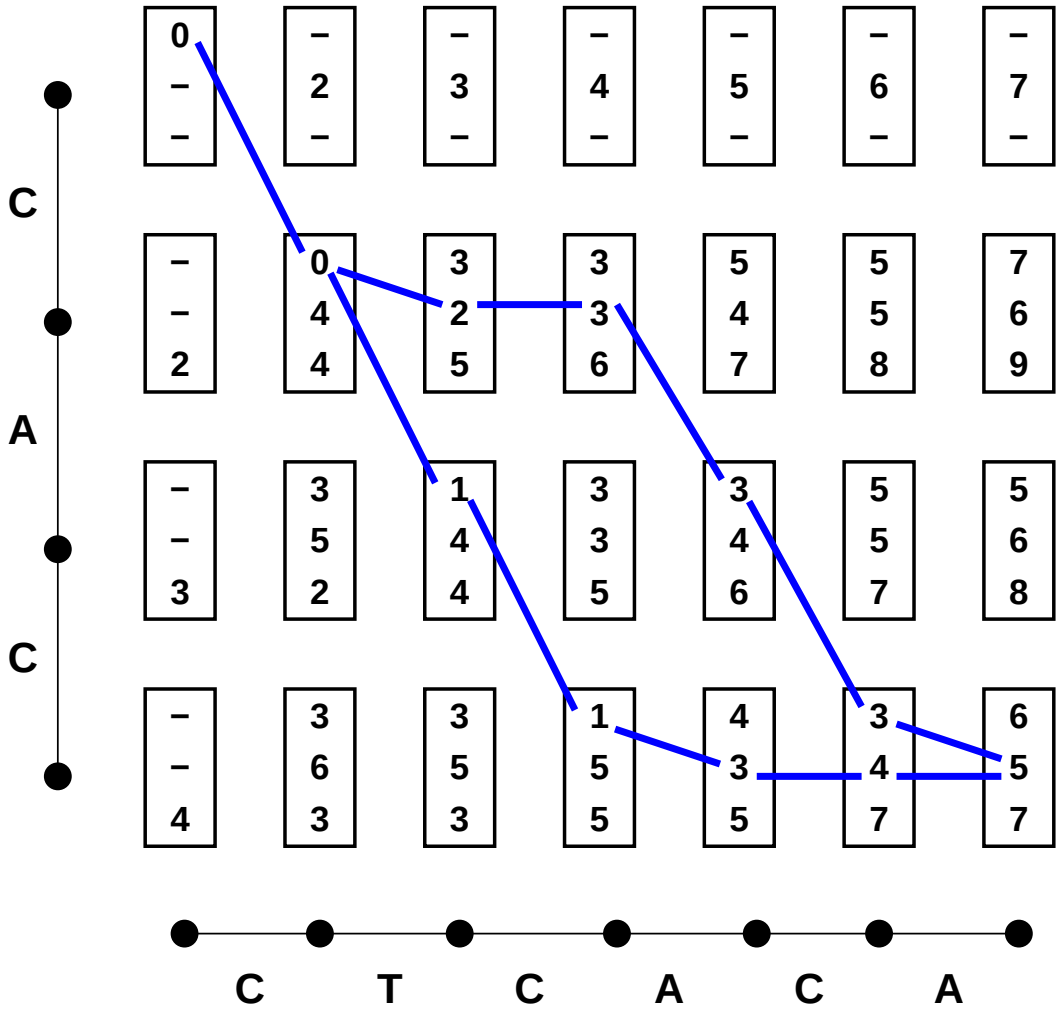
where l is the size of the character alphabet.

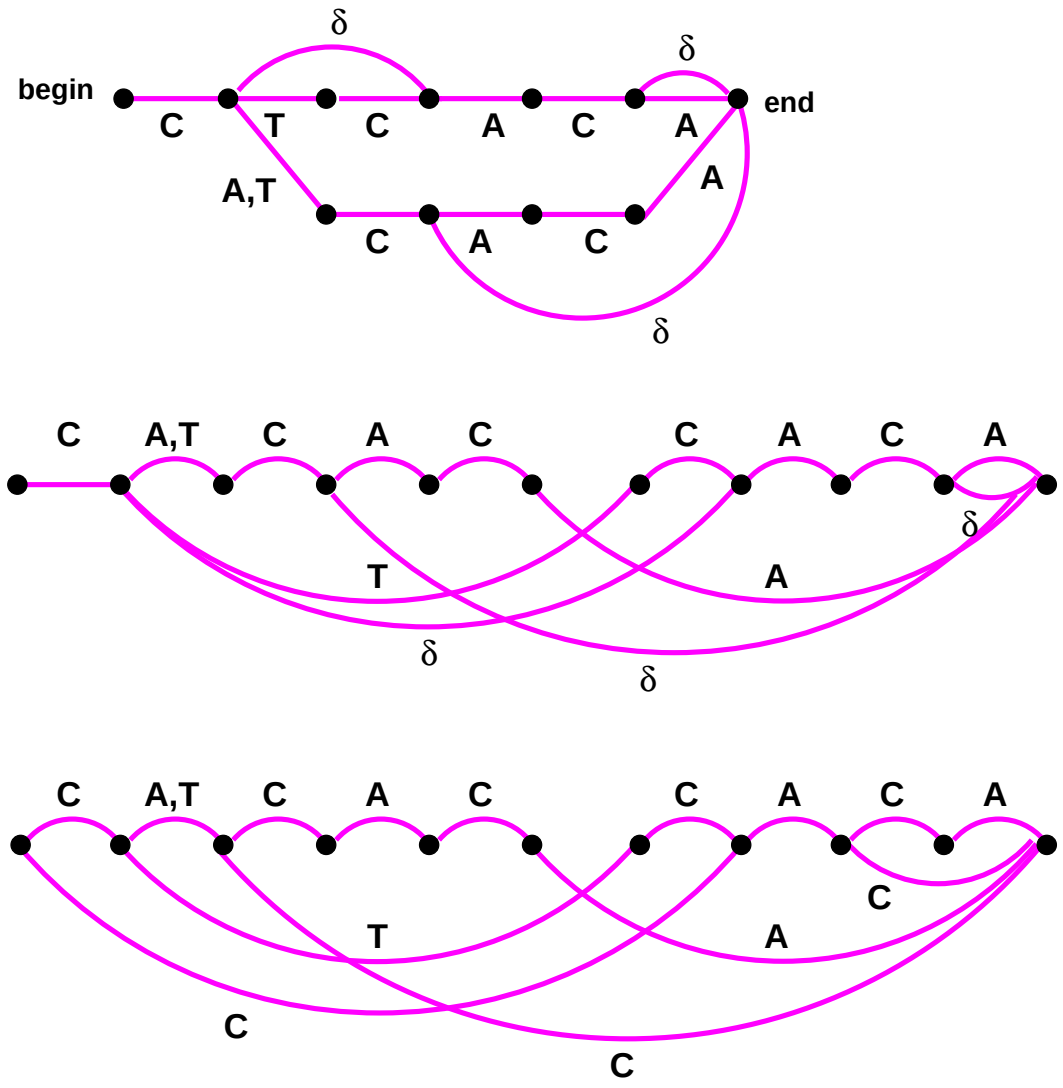
Unfortunately this is too large for more than a half a dozen or so of sequences of normal length (of the order of 100 residues).

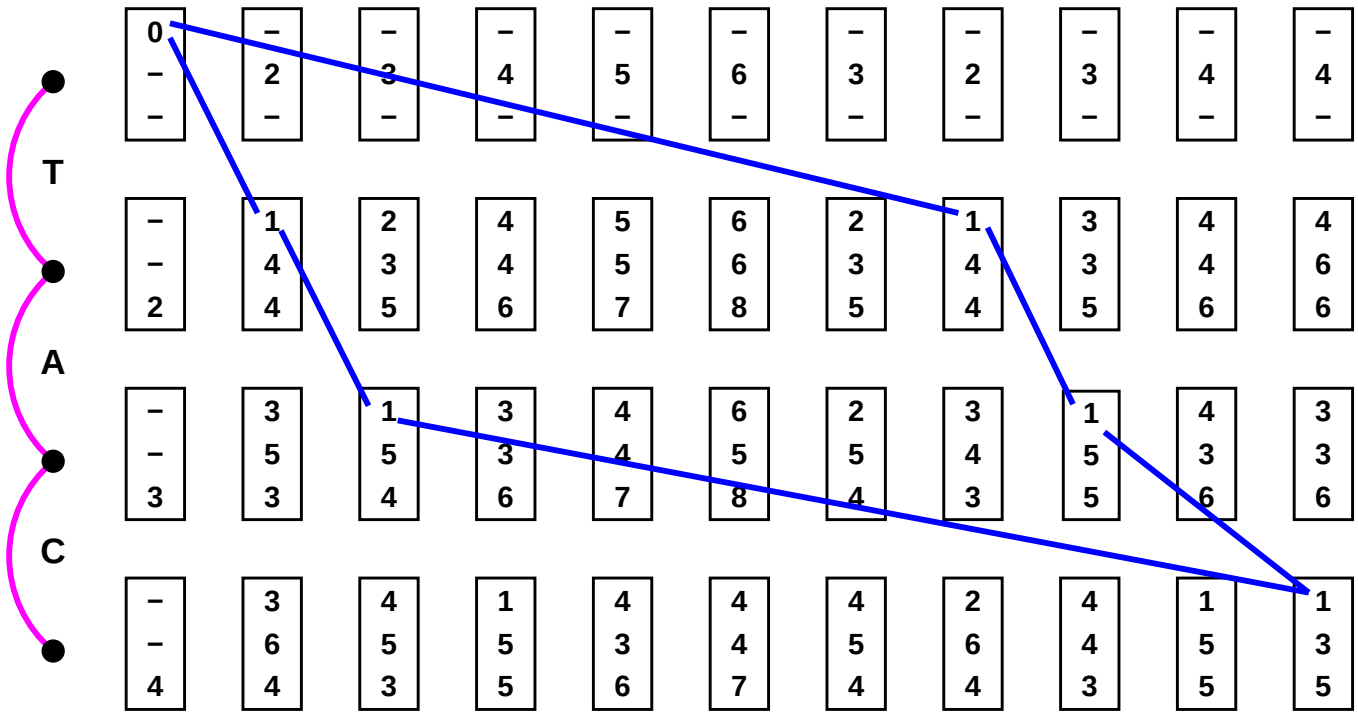
4.2 Hein's affine cost algorithm (1989)

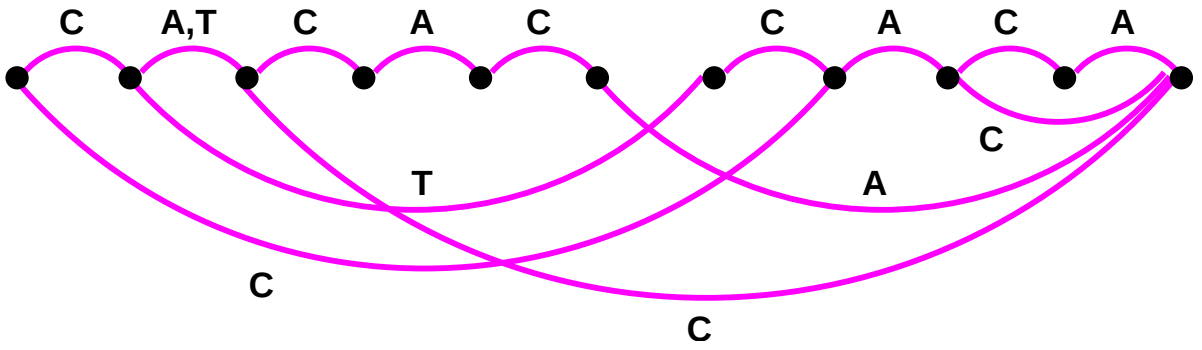
comments to follow sometime, hopefully soon...

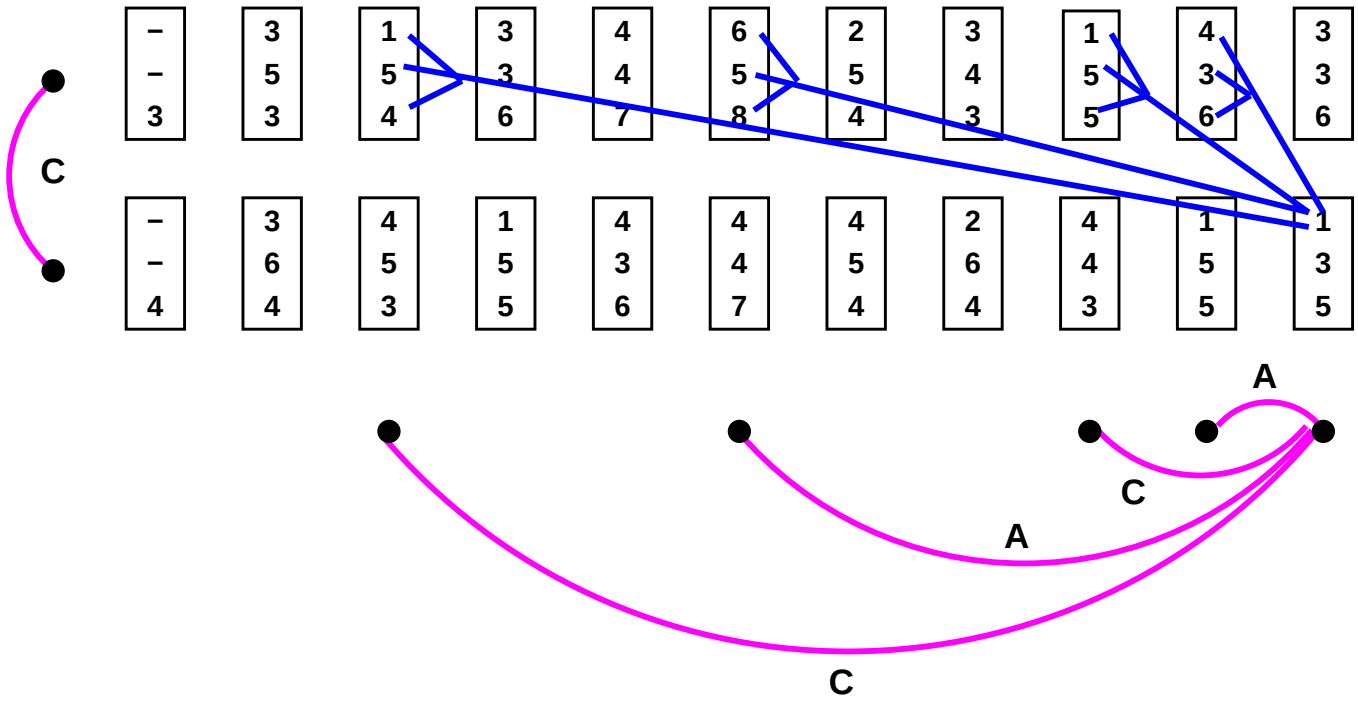


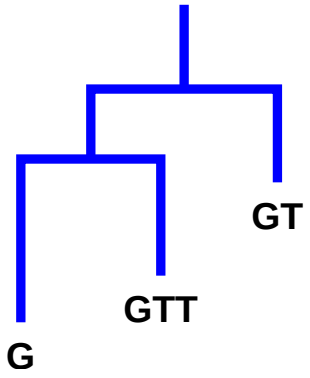

C
A
C
0 – –
– 2 –
– 3 –
– 4 –
– 5 –
– 6 –
– 7 –
– – 2
0 4 4
3 2 5
3 3 6
5 4 7
5 5 8
7 6 9
– – 3
3 5 2
1 4 4
3 3 5
3 4 6
5 5 7
5 6 8
– – 4
3 6 3
3 5 3
1 5 5
4 3 5
3 4 7
6 5 7
C T C A C A


δ
δ
begin
end
C
T
C
A
C
A
A,T
A
C
A
C
δ
C
A,T
C
A
C
C
A
C
A
δ
T
A
δ
δ
C
A,T
C
A
C
C
A
C
A
C
T
A
C
C


T
A
C
0 – –
– 2 –
– 3 –
– 4 –
– 5 –
– 6 –
– 3 –
– 2 –
– 3 –
– 4 –
– 4 –
– – 2
1 4 4
2 3 5
4 4 6
5 5 7
6 6 8
2 3 5
1 4 4
3 3 5
4 4 6
4 6 6
– – 3
3 5 3
1 5 4
3 3 6
4 4 7
6 5 8
2 5 4
3 4 3
1 5 5
4 3 6
3 3 6
– – 4
3 6 4
4 5 3
1 5 5
4 3 6
4 4 7
4 5 4
2 6 4
4 4 3
1 5 5
1 3 5


C
A,T
C
A
C
C
A
C
A
C
T
A
C
C


C
A
C
A
C


GT
GTT
G