Computational Biology

Krzysztof Giaro
Gdańsk University of Technology
ETI Faculty

Computer phylogenetics

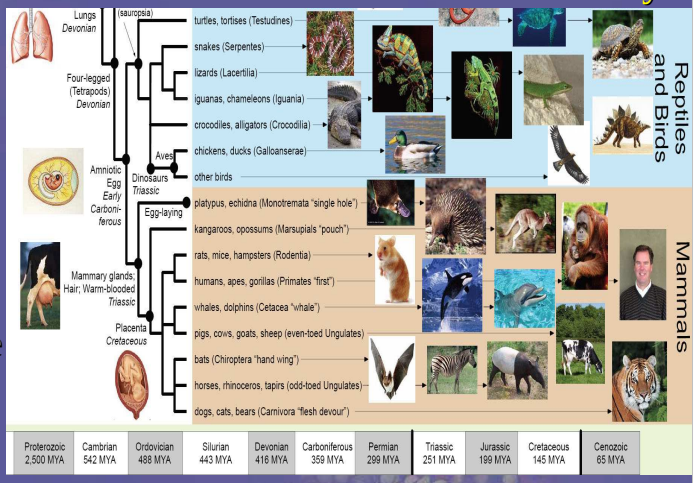
reconstruction of evolutionary history

Basic phylogenetics

Nothing in Biology Makes Sense Except in the Light of Evolution.
Dobzhansky

Phylogenetic tree – family-like tree of an examined set of contemporary species that shows history of their evolution from a common antecessor.

Speciation – the emergence of new species from one antecessor.

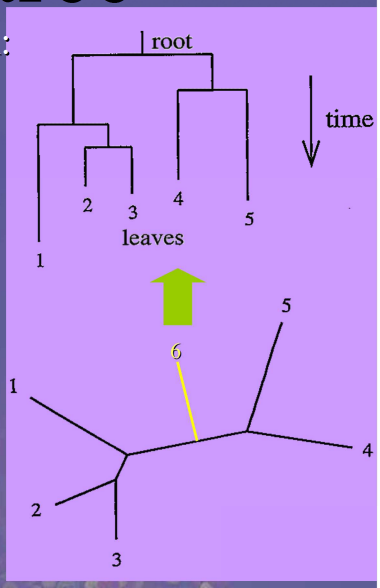


Aim: to reconstruct a phylogenetic tree relying on information about contemporary living species only (e.g. genomic).

Phylogenetic tree

A ***rooted phylogenetic tree*** $T(V,E)$ – a tree graph:

- ***leaves*** set (of degree 1) $L \subseteq V$ – contemporary species under examination,
- ***internal vertices*** ($V \setminus L$ set) – extinct ancestors of those from L,
- ***root*** – internal vertex, common ancestor of all vertices in T,
- no vertex with possible exception of root has degree 2.



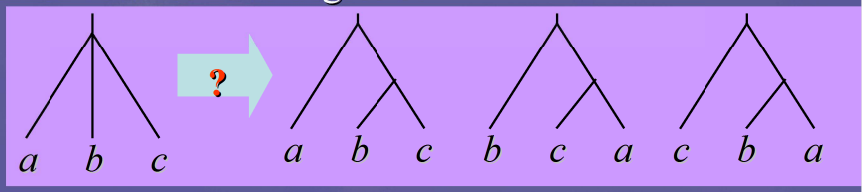
An ***unrooted phylogenetic tree*** $T(V,E)$ – analogous to rooted, but no root vertex is present,

- shows evolutionary relationships, but without time flow direction,
- rooting (on the edge or internal vertex) is often the next step of phylogenetic analysis (e.g. ***outgroup method***).

Phylogenetic tree

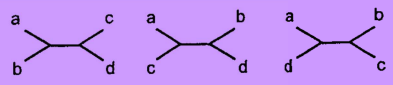
Binary phylogenetic tree – the root is of degree 2 (if the tree is rooted) and all other internal vertices have degree 3.

- the most informative structure.

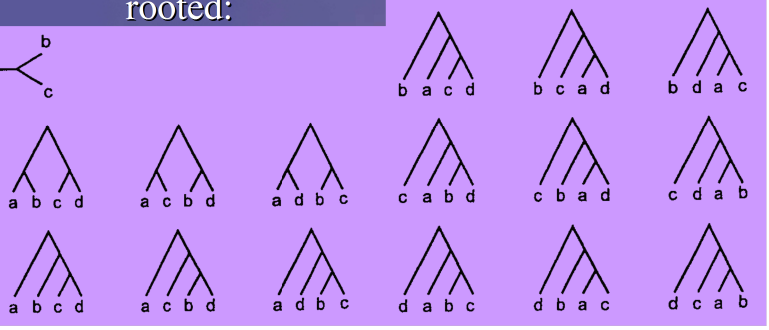


Example. All binary phylogenetic topologies on 4 leaves.

unrooted:



rooted:



Number of phylogenies

$T(V,E)$ – a binary unrooted phylogenetic tree with n leaves (set L).
$|V\backslash L|=w$, $|E|=m$.

$m=n+w-1$
$2m=n+3w$

$\Rightarrow$

Binary unrooted case: $2n-3$ edges, $n-2$ internal vertices.

Binary rooted case: $2n-2$ edges, $n-1$ internal vertices.

U_n, R_n – numbers of (unrooted/rooted) n-leaf binary phylogenetic trees.

$U_3=1$, $U_{n+1}=R_n$, $R_n=(2n-3)U_n$

$U_n = 3\cdot5\cdot\ldots\cdot(2n-5) = \Theta((2/e)^n n^{n-2})$

$R_n = 3\cdot5\cdot\ldots\cdot(2n-3)$

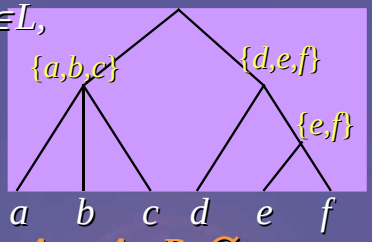
– superexponential growth rate!

	$U_n=$
$n=5$	15
$n=10$	$\sim 2\times10^6$
$n=15$	$\sim 7.9\times10^{12}$
$n=20$	$\sim 2.2\times10^{20}$

Non-graph rooted tree description

$T(V,E)$ – a rooted phylogenetic tree with leaves L. The ***cluster*** of vertex $v \in V$ is a set $C_v \subseteq L$ of leaves that are v's descendants.

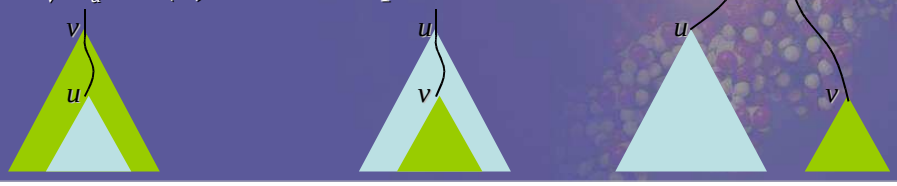
- ***trivial clusters*** $C_{root}=L$ and $C_u=\{u\}$ for $u \in L$,
- **$\sigma(T)$** – a set of all clusters in T.



Theorem. The set σ of L subsets which contains all trivial clusters from L is $\sigma(T)$ for some phylogenetic tree T with leaves L if and only if for each $A,B \in \sigma$: ***$A \subseteq B$ or $B \subseteq A$ or $A \cap B = \varnothing$.***

Proof. (***compatibility conditions***)

($\Rightarrow$) For $C_v, C_u \in \sigma(T)$ we have 3 posibilities:



Non-graph rooted tree description

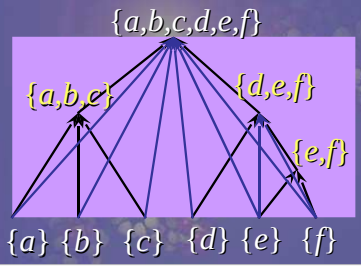
Theorem. The set σ of L subsets which contains all trivial clusters from L, is $\sigma(T)$ for some phylogenetic tree T with leaves L if and only if for each $A,B \in \sigma$: ***$A \subseteq B$ or $B \subseteq A$ or $A \cap B = \varnothing$.***

Proof.

($\Leftarrow$) Reconstruction algorithm for tree T from set σ:

1. Make a digraph of inclusion relation $\subset$ in σ,
2. Remove all transitive arches (result: *Hasse diagram*),
3. Put the root in vertex with L, make edges undirected.

Example. $L=\{a,b,c,d,e,f\}$, $\sigma=\{\{a\},\{b\},\{c\},\{d\},\{e\},\{f\},\{a,b,c\},\{d,e,f\},\{e,f\},\{a,b,c,d,e,f\}\}$.



Non-graph rooted tree description

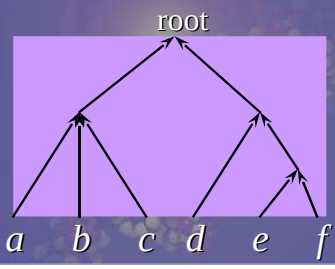
Theorem. The set σ of L subsets which contains all trivial clusters from L, is σ(T) for some phylogenetic tree T with leaves L if and only if for each $A,B \in \sigma$: $A \subseteq B$ **or** $B \subseteq A$ **or** $A \cap B = \varnothing$.

Proof.

(⇐) Reconstruction algorithm for tree T from set σ:

1. Make a digraph of inclusion relation $\subset$ in σ,
2. Remove all transitive arches (result: *Hasse diagram*),
3. Put the root in vertex with L, make edges undirected.

Example. $L=\{a,b,c,d,e,f\}$, $\sigma=\{\{a\},\{b\},\{c\},\{d\},\{e\},\{f\},\{a,b,c\},\{d,e,f\},\{e,f\},\{a,b,c,d,e,f\}\}$.

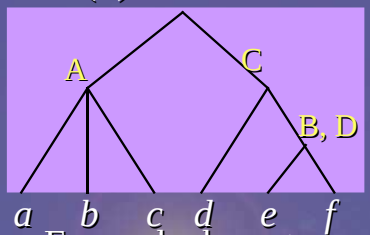


Perfect phylogeny problem

Problem. Given the set of species L and their binary (i.e. Yes/No) characters X. Is there a rooted phylogenetic tree T with leaves L such that for each character $x \in X$ there is a vertex $v \in V(T)$ that the set of all leaves having x is a v cluster?

Example.

	A	B	C	D
a	1	0	0	0
b	1	0	0	0
c	1	0	0	0
d	0	0	1	0
e	0	1	1	1
f	0	1	1	1



Solution. For each character $x \in X$ define set $L_x \subseteq L$ of species having x, check compatibility, if so – add necessary trivial clusters and reconstruct.

- Known algorithm with $O(|L||C|)$-time complexity.

Perfect phylogeny problem

- irreversible and unique (appearing once) species characters,
- internal vertices represent species which gained new characteristics

and passed them down to their descendants.

The model was used in classic cladistics, but „morphological" characters often violate the above conditions:
(e.g. viviparousness, ability to fly):

- once acquired may atrophy,
- similar adaptations often appear independently.

Genomic phylogenetic markers behave better:

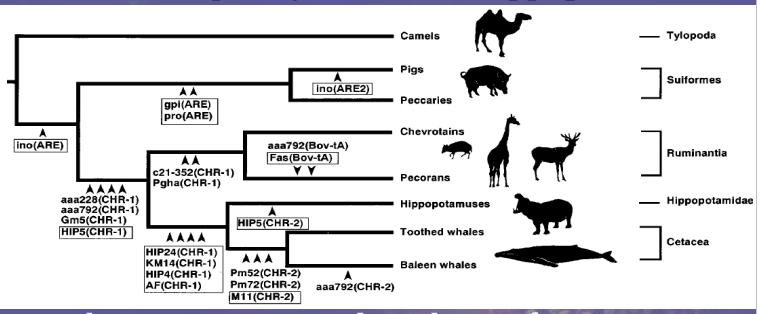
- the parasitic gene (transposon, retrovirus) emergence in the genome,
- vast rearrangement,
- appearance of a new intron in a gene

…

Perfect phylogeny problem

Example. Evolution of whales – examination of the presence/lack of 20 different repetitive sequences in genomes of a group of present-day species shows that the closest contemporary relative is hippopotamus.

	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Camel	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 ? 0 0 0
Pig	0 0 0 ? 0 0 0 0 ? 0 0 0 ? ? 0 0 ? 1 1 1
Peccary	? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? ? 1 1
Chevrotain	? 0 ? ? ? ? ? ? ? 1 0 ? ? ? 1 1 0 ? 0 0
Deer	0 0 0 0 0 0 0 1 ? 1 1 1 1 1 1 ? 1 1 0 0
Graffe	? 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0
Sheep	0 0 0 0 0 ? 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0
Cow	0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0
Hippo	0 ? 0 1 1 1 1 0 1 1 0 1 1 0 0 0 ? 1 0 0
Humpback	1 1 1 1 1 1 1 0 1 1 0 1 1 0 0 0 ? ? 0 0
Beaked	1 1 1 1 1 1 1 0 ? 1 0 1 1 0 0 0 ? 1 0 0



Problem. Incompatibility – characters may violate the perfect phylogeny assumptions or be read with errors.
Theorem. Finding in a given set σ of L subset the largest compatible $\sigma' \subseteq \sigma$ is NP-hard.

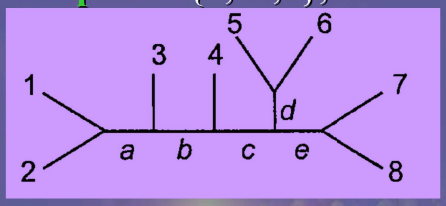
Non-graph unrooted tree description

$T(V,E)$ – an unrooted phylogenetic tree with leaves L. The ***split*** of edge $e \in E$ is a 2-partition $A|B$ of L (i.e. $A \cup B = L$, $A \cap B = \varnothing$, an unordered pair $A|B = B|A$), where A and B are leaves in both connected components obtained by removing e from T.

- ***trivial*** splits: of the form $\{v\}|L\setminus\{v\}$ for $v \in L$,
- **β(*T*)** – set of all splits in T.

Perfect phylogeny-like point of view: characters are still 2-valued, irreversible and unique, but we don't know which value was original. The edge is confirmed by change of character's states (split).

Example. $L=\{1,\ldots,8\}$, T:



β(T) contains
a: {1,2}|{3,4,5,6,7,8},
b: {1,2,3}|{4,5,6,7,8},
c: {1,2,3,4}|{5,6,7,8},
d: {1,2,3,4,7,8}|{5,6},
e: {1,2,3,4,5,6}|{7,8}
and 8 trivial splits.

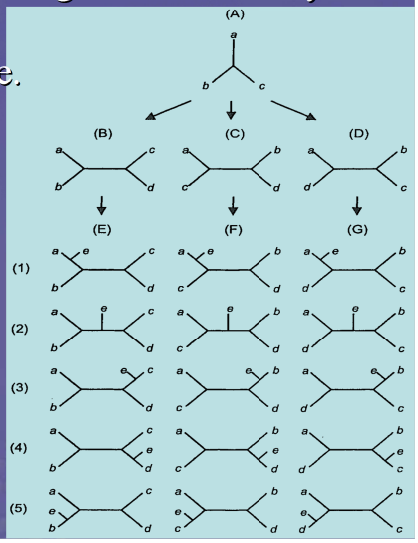
Non-graph unrooted tree description

Theorem. The set β of 2-splits of L, which contains all L trivial splits, is β(T) for some phylogenetic tree T with leaves L if and only if for every two $A|B, C|D \in$ β exactly one set of **$A\cap C$, $A\cap D$, $B\cap C$, $B\cap D$** is an empty set (***split's compatibility condition***).

Reconstruction algorithm for tree T from compatible splits set β:
1. Choose any leaf v,
2. For each $A|B \in$ β take a partition not containing v – they are compatible clusters over $L\setminus\{v\}$,
3. Their use creates a rooted phylogenetic tree with leaves $L\setminus\{v\}$,
4. Attach a leaf v to the root.

Phylogenetic methods

Typically, algorithms do not provide a tree directly. From certain assumptions we define a function f of a tree T (fast evaluable) „reliability" $f(T)$ and the algorithm searches among unrooted binary trees (over leaf set L) for the maximum of f .
Impossible one by one because of trees set size.



If f decreases after attaching a new leaf, the ***branch and bound*** method (by successive adding leaves in all ways) is applicable:

- remember the best already visited T" with all leaves L,
- if the current tree $f(T) \leq f(T")$ – skip the rest of this branch.

Practically, global optimum can by found for some $|L|>10$.

Phylogenetic methods

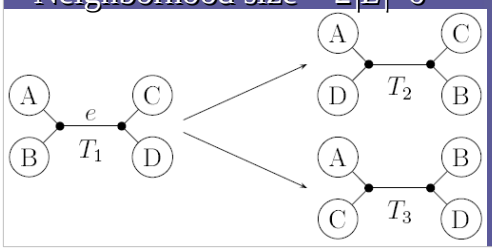
For larger L any artificial intelligence heuristic search for the function's (of large domain) optimum may be used. Metaheuristics cannot guarantee results according to the risk of being stuck in the local optimum.

simple hill climbing, genetic algorithm, simulated annealing, Monte Carlo Markov chain …

Heuristics usually explores the searched space point by point, jumping between appropriately defined *neighbours*. ***Tree rearangements***.

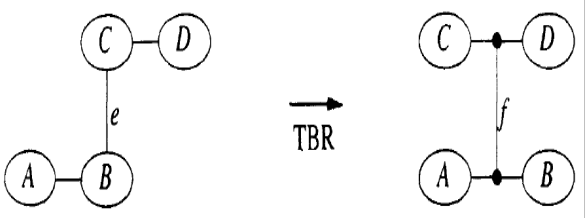
Nearest Neighbour Interchange (NNI)

Neighborhood size = $2|L|-6$



Tree Bisection–Reconnection (TBR)

Neighborhood size $\leq (2|L|-3)(|L|-3)^2$



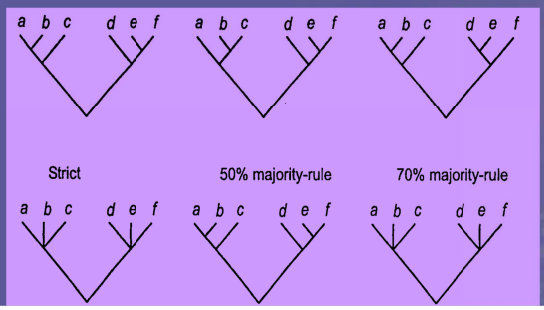
Consensus tree

Problem. Several analyses of species set L result in non-identical rooted trees $T_1, T_2, \ldots, T_k$. How to define phylogenetic information, reliable because confirmed in more than x% of these results?

Solution. Take σ as a set of clusters that appear in more than x% of $\sigma(T_1), \ldots, \sigma(T_k)$. If $x \geq 50$ then σ is compatible.

- ***consensus x% tree*** T fulfils $\sigma(T)=\sigma$,
- ***strict consensus tree*** – $x=100$,
- ***majority consensus tree*** – $x=50$.

Example.



For unrooted trees $T_1, T_2, \ldots, T_k$ just take splits sets $\beta(T_i)$ instead of clusters.

Distance methods

Phylogenetic analysis of species set L:

1. Find gene/protein whose homologues are present in all species of L,
2. Make multialignment,
3. Calculate evolutionary distance (in some evolutionary model) between every pair - ***distance matrix*** d (metric on L),
4. Distances in d are the only source of information for the rest of work.

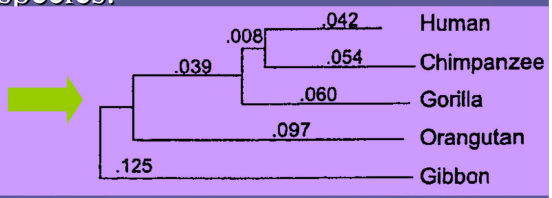
Properties:

- elegant polynomial time algorithms (e.g. UPGMA, NJ) directly reconstructing an unrooted tree (without tree search),
- edges of the returned tree are labelled with their approximate evolutionary distances (along edges),
- strong assumption, often violated in practice (high accuracy of estimated distances in d),
- reliable for small $|L|$ only or as the promising start point for heuristic search.

Additive tree model

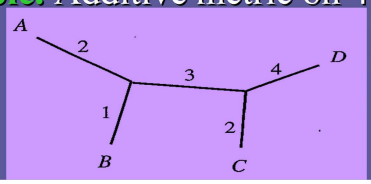
Example. Kimura Distance between common segments of mitochondrial DNA for 5 ...monkey species.

	Human	Chimpanzee	Gorilla	Orangutan
Chimpanzee	0.095 ± 0.011			
Gorilla	0.113 ± 0.012	0.118 ± 0.013		
Orangutan	0.183 ± 0.016	0.201 ± 0.018	0.195 ± 0.017	
Gibbon	0.212 ± 0.018	0.225 ± 0.019	0.225 ± 0.019	0.222 ± 0.018



- ***Additive tree*** for L – an unrooted phylogenetic tree $T(V,E)$ with leaves L and weights on edges $w:E\to R^{\geq 0}$. In the associated distance matrix $d(u,v)$ is the weights sum of edges on the path connecting u and v in T.
- Metric d on L is ***additive*** if it is associated with some additive tree.

Example. Additive metric on 4 species:



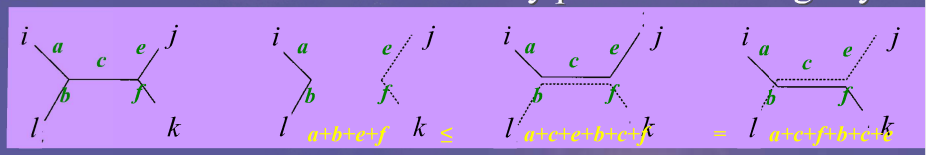
	A	B	C	D
A	0	3	7	9
B	3	0	6	8
C	7	6	0	6
D	9	8	6	0

Additive tree model

Additivity is a rare property among metrices.

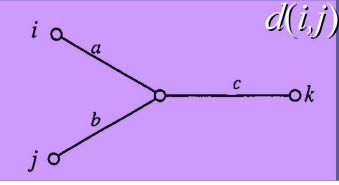
Theorem. Metric d on L is additive if and only if for every four different elements $i,j,k,l\in L$ among numbers $d(i,j)+d(k,l)$, $d(i,k)+d(j,l)$, $d(i,l)+d(j,k)$ the largest value is attained by at least two of them (**4 *point condition***).

Half of proof. $\Rightarrow$ Consider a subtree induced by paths connecting any four leaves.



The main **problem.** How to reconstruct an additive tree from additive d?

Example. For 3 species the problem is easy.



$d(i,j)=w(a)+w(b)$, $d(i,k)=w(a)+w(c)$, $d(j,k)=w(b)+w(c)$

$\Rightarrow$ $w(a)=[d(i,j)+d(i,k)-d(j,k)]/2$,
$w(b)=[d(i,j)+d(j,k)-d(i,k)]/2$,
$w(c)=[d(i,k)+d(j,k)-d(i,j)]/2$.

Neighbour–Joining algorithm

Algorithm (***NJ***). Input: additive metric d on L.

while $|L|>3$ **do begin**

find a pair $i,j \in L$ which must have a common neighbour $k \in V(T)\setminus L$

??? 1 $L:=L\cup\{k\}\setminus\{i,j\}$ // do not consider leaves i,j in the rest

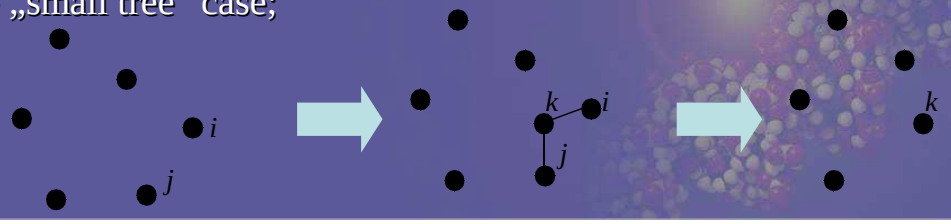
??? 2 update distances d for a new vertex k: find $d(k,l)$ for $l \in L\setminus\{k\}$

??? 3 introduce edges: i–k, j–k to the constructed tree

find their lenghts $w(\{k,i\})$, $w(\{k,j\})$

end;

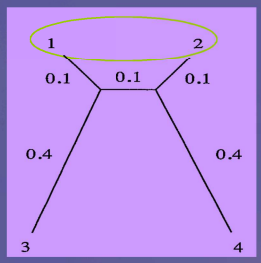
solve „small tree” case;



Neighbour–Joining algorithm

Question 1. *... find a pair $i,j \in L$ which must have a common neighbour $k \in V(T)\setminus L$*

Example. Joining the closest two leaves doesn't work.



Example. Simple solution: choose any $a \in L$ and return the pair of other $i,j \in L$ that maximizes $(d(a,i)+d(a,j)-d(i,j))/2$.

Theorem (Studier & Keppler). For all $a \in L$ let:

$$r_a=[\Sigma_{b \in L}\, d(a,b)]/(|L|-2).$$

Then the pair $i,j \in L$ that minimizes $d(i,j)-(r_i+r_j)$ must have a common neighbour.

Neighbour–Joining algorithm

Question 2. … *find d(k,l) for l∈L\{k}*

The tree is additive, so:

$$d(k,l)=[d(i,l)+d(j,l)-d(i,j)]/2$$

Question 3. … *find lenghts w({k,i}), w({k,j})*

Considering any other leaf l:

$$d(i,k)=[d(i,j)+d(i,l)-d(j,l)]/2$$

But often „mean value” is used:

$$d(i,k)=[d(i,j)+r_i-r_j]/2$$

Finally: $d(j,k)=d(i,j)-d(i,k)$.

⇒ An additive tree may be reconstructed from its additive matrix in polynomial time.

But … NJ always returns a binary tree

⇒ last step: shrink all non-adjacent to L edges e with $w(e)=0$.

Neighbour–Joining algorithm

d(i, j)	A	B	C	D	E
A	0				
B	3	0			
C	9	10	0		
D	9	10	2	0	
E	10	11	9	9	0

$r_a=[\Sigma_{b\in L}\, d(a,b)]/(|L|-2)$

a	r_a
A	31/3
B	34/3
C	30/3
D	30/3
E	39/3

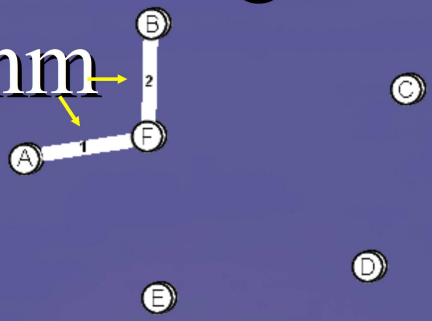
min

d(i, j)-(r_i+r_j)	A	B	C	D	E
A	-				
B	-56/3	-			
C	-34/3	-34/3	-		
D	-34/3	-34/3	-54/3	-	
E	-40/3	-40/3	-42/3	-42/3	-

Neighbour–Joining algorithm

$d(k,l)=[d(i,l)+d(j,l)-d(i,j)]/2$

$d(i,j)$	C	D	E	F
C	0			
D	2	0		
E	9	9	0	
F	8	8	9	0

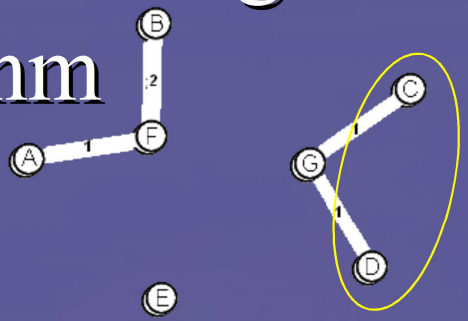


$d(i,k)=[d(i,j)+r_i-r_j]/2$

$d(j,k)=d(i,j)-d(i,k)$

Neighbour–Joining algorithm

$d(i,j)$	C	D	E	F
C	0			
D	2	0		
E	9	9	0	
F	8	8	9	0



a	r_a
C	19/2
D	19/2
E	27/2
F	25/2

min

$d(i,j)-(r_i+r_j)$	C	D	E	F
C	-			
D	-17	-		
E	-14	-14	-	
F	-14	-14	-17	-

Neighbour–Joining algorithm

d(*i*, *j*)	E	F	G
E	0		
F	9	0	
G	8	7	0

$w(a)=[d(i,j)+d(i,k)-d(j,k)]/2,$
$w(b)=[d(i,j)+d(j,k)-d(i,k)]/2,$
$w(c)=[d(i,k)+d(j,k)-d(i,j)]/2.$

d(*i*, *j*)	A	B	C	D	E
A	0				
B	3	0			
C	9	10	0		
D	9	10	2	0	
E	10	11	9	9	0