Inférence phylogénétique

Inférence d'arbres phylogénétiques

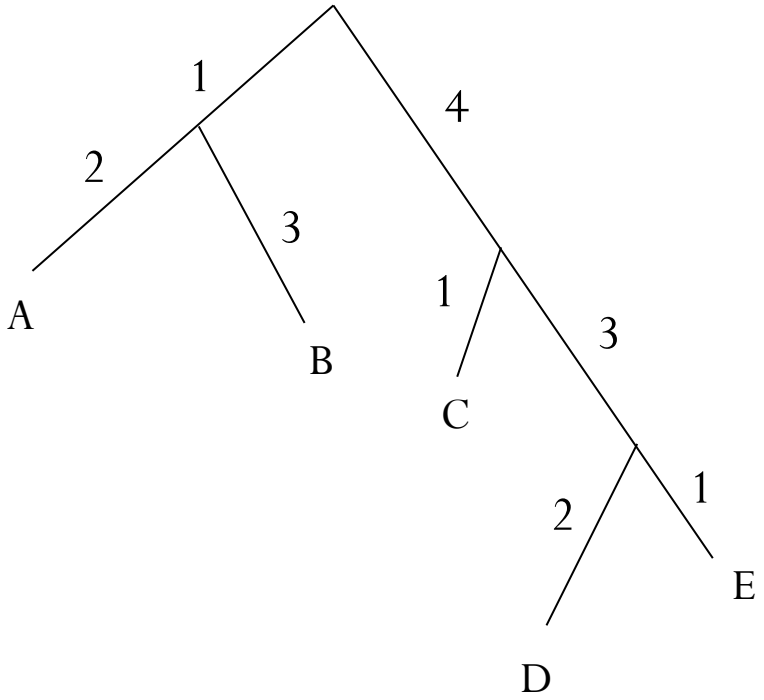
- Méthodes de distance
  - Input: Matrice de distances D
  - Construire un arbre qui « réalise » cette matrice: chaque paire (x,y) de feuilles est reliée par un chemin dont le score est égal à la distance $D(x,y)$ entre x et y.
- Méthodes de parcimonie: Arbre qui explique l'évolution des espèces par un nombre minimum de mutations. Deux composantes principales:
  - Calcul d'un score d'un arbre donné.
  - Recherche, parmi tous les arbres, l'arbre de score minimal.
- Méthodes probabilistes
  - Maximisation de la vraisemblance d'un arbre
  - Inférence Bayésienne, basée sur la probabilité postérieure des hypothèses en fonction des données.

I. Méthodes de distance

Distance additive

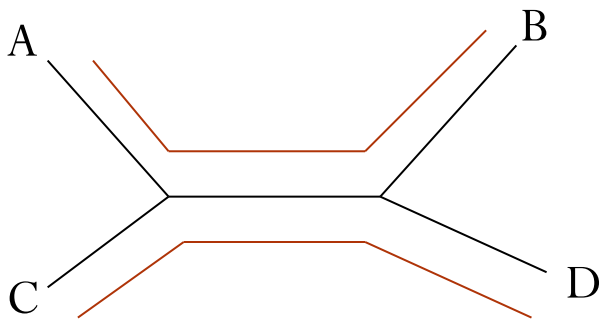
- Étant donnée une matrice de distance *D*, existe-t-il un **arbre binaire avec un étiquetage des branches qui « réalise » la matrice**, i.e. telque pour chaque paire de feuilles i, j, la somme des étiquettes des branches du chemin qui relie i et j est égal à *D(i,j)*.
- Si oui, la matrice ***D* est additive.**

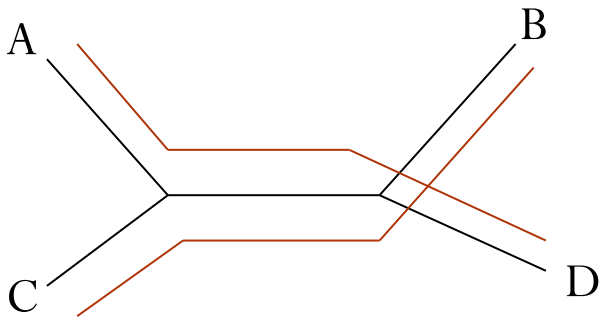
	A	B	C	D	E
A	0	5	8	12	11
B	5	0	9	13	12
C	8	9	0	6	5
D	12	13	6	0	3
E	11	12	5	3	0

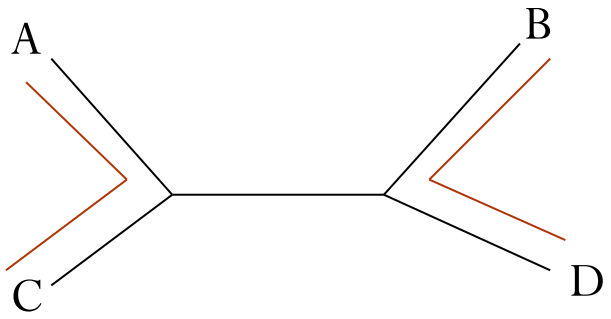


Condition des 4 points

- ***D* est additive si et seulement *D* satisfait la condition des 4 points:** Pour tout choix de 4 feuilles A, B, C, D, **deux des sommes suivantes sont égales et supérieures à la 3ème:** ***D(A,B) + D(C,D), D(A,D) + D(B,C) et D(A,C)+D(B,D)***





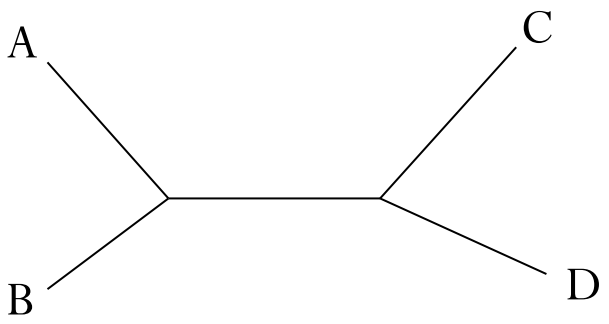


Distances additives

- Une matrice de distance qui satisfait la **condition des 4 points** est une matrice de **distance additive**.

D	A	B	C	D
A	0	3	3	5
B		0	4	6
C			0	4
D				0

$$\boldsymbol{D}(A,B) + \boldsymbol{D}(C,D) = 3 + 4 = 7$$
$$\boldsymbol{D}(A,C) + \boldsymbol{D}(B,D) = 3 + 6 = 9$$
$$\boldsymbol{D}(A,D) + \boldsymbol{D}(B,C) = 5 + 4 = 9$$

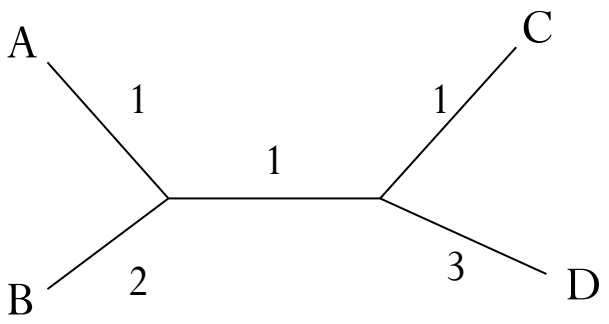


Distances additives

- Une matrice de distance qui satisfait la **condition des 4 points** est une matrice de **distance additive**.

D	A	B	C	D
A	0	3	3	5
B		0	4	6
C			0	4
D				0

$$\boldsymbol{D}(A,B) + \boldsymbol{D}(C,D) = 3 + 4 = 7$$
$$\boldsymbol{D}(A,C) + \boldsymbol{D}(B,D) = 3 + 6 = 9$$
$$\boldsymbol{D}(A,D) + \boldsymbol{D}(B,C) = 5 + 4 = 9$$

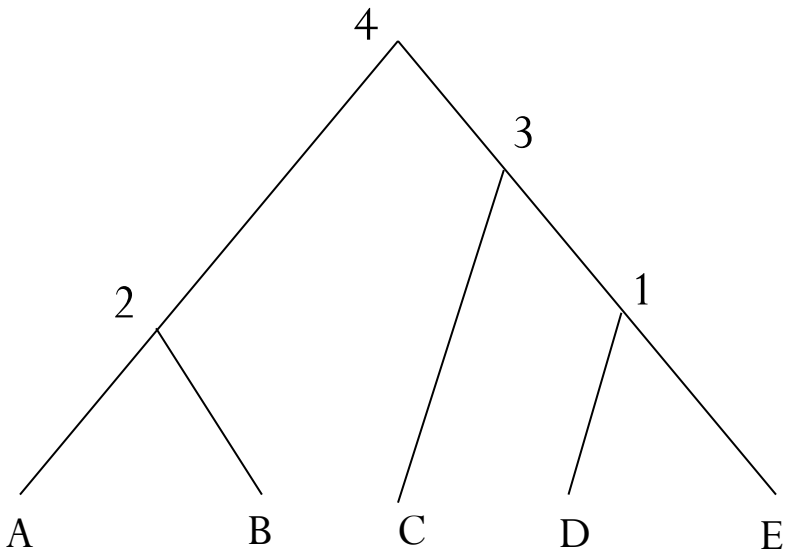


Distance ultramétrique

***D* est ultramétrique** ssi il existe un **arbre binaire raciné avec étiquetage des nœuds** tq

- Le long d'un chemin de la racine à une feuille les valeurs des étiquettes des nœuds décroissent strictement;
- **l'arbre « réalise » la matrice**, i.e. pour chaque paire de feuilles i, j, le LCA de i et j est étiqueté *D(i,j)*.

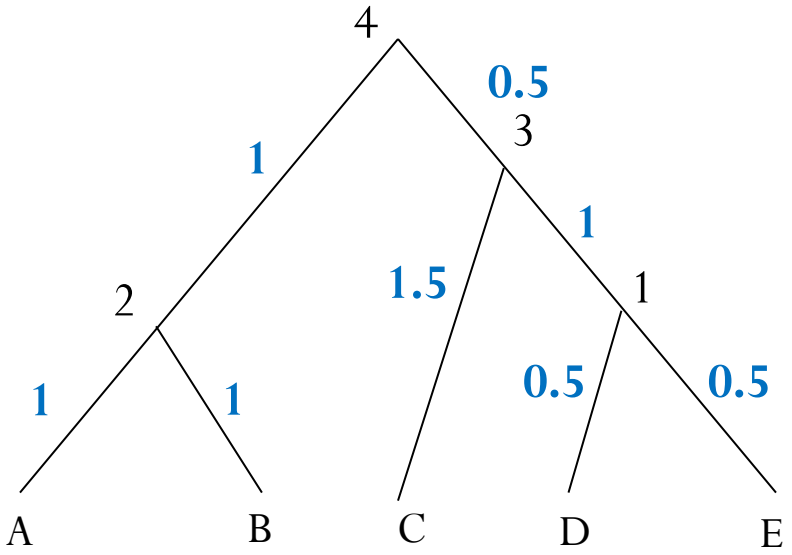
D	A	B	C	D	E
A	0	2	4	4	4
B	2	0	4	4	4
C	4	4	0	3	3
D	4	4	3	0	1
E	4	4	3	1	0



Distance ultramétrique

- Contrainte plus forte qu'une distance additive:
  - Toute distance ultramétrique est additive

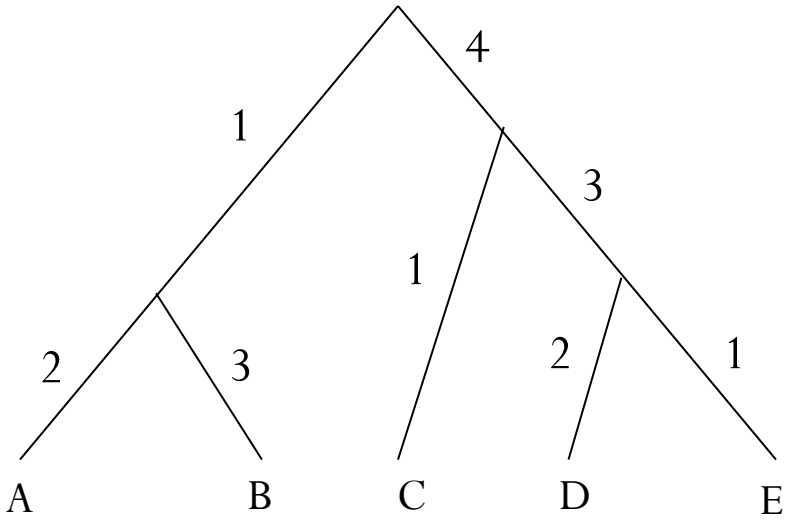
	A	B	C	D	E
A	0	2	4	4	4
B	2	0	4	4	4
C	4	4	0	3	3
D	4	4	3	0	1
E	4	4	3	1	0



Distance ultramétrique

- Contrainte plus forte qu'une distance additive:
  - Toute distance ultramétrique est additive
  - Mais toute distance additive n'est pas ultramétrique

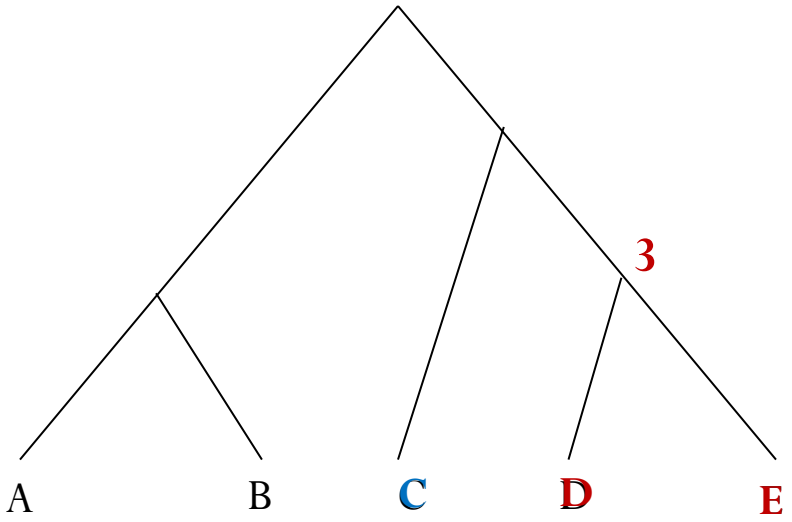
D	A	B	C	D	E
A	0	5	8	12	11
B	5	0	9	13	12
C	8	9	0	6	5
D	12	13	6	0	3
E	11	12	5	3	0



Distance ultramétrique

- Contrainte plus forte qu'une distance additive:
  - Toute distance ultramétrique est additive
  - Mais toute distance additive n'est pas ultramétrique

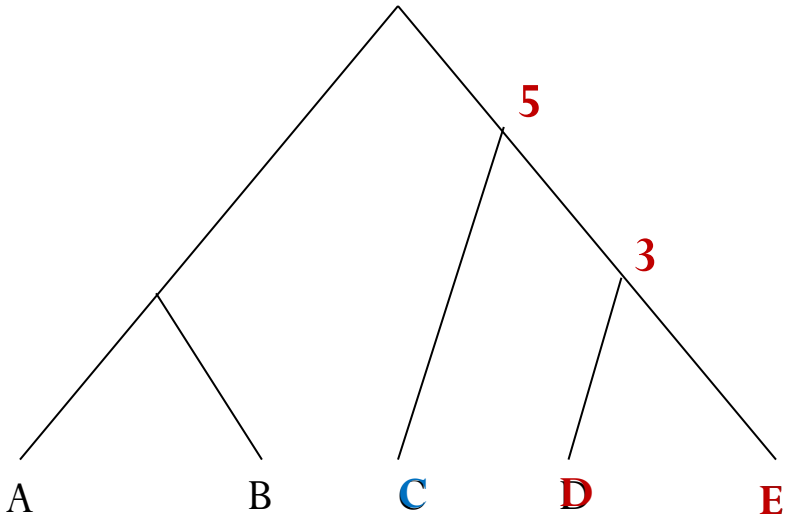
D	A	B	C	D	E
A	0	5	8	12	11
B	5	0	9	13	12
C	8	9	0	6	5
D	12	13	6	0	3
E	11	12	5	3	0



Distance ultramétrique

- Contrainte plus forte qu'une distance additive:
  - Toute distance ultramétrique est additive
  - Mais toute distance additive n'est pas ultramétrique

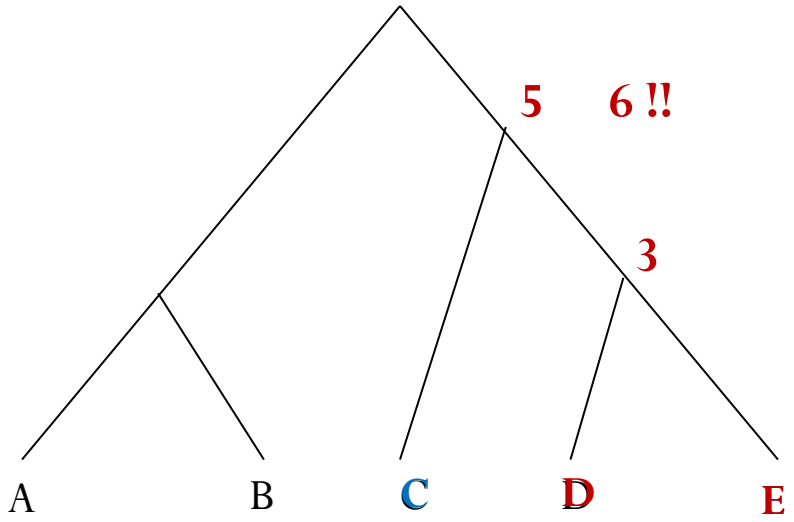
	A	B	C	D	E
A	0	5	8	12	11
B	5	0	9	13	12
C	8	9	0	6	5
D	12	13	6	0	3
E	11	12	5	3	0



Distance ultramétrique

- Contrainte plus forte qu'une distance additive:
  - Toute distance ultramétrique est additive
  - Mais toute distance additive n'est pas ultramétrique

	A	B	C	D	E
A	0	5	8	12	11
B	5	0	9	13	12
C	8	9	0	6	5
D	12	13	6	0	3
E	11	12	5	3	0



Distance ultramétrique

T est un arbre ultramétrique associé à la distance ultramétrique *D* ssi:

- *T* contient n feuilles, chacune étiquetée par une ligne de *D*;
- Chaque nœud interne est étiqueté par une case de *D* et a au moins deux fils;
- **Le long d'un chemin de la racine à une feuille les valeurs des étiquettes des nœuds décroissent strictement;**
- **Pour deux feuilles quelconques *i, j, D(i,j)* est l'étiquette du dernier ancêtre commun de *i* et *j* dans *T*.**

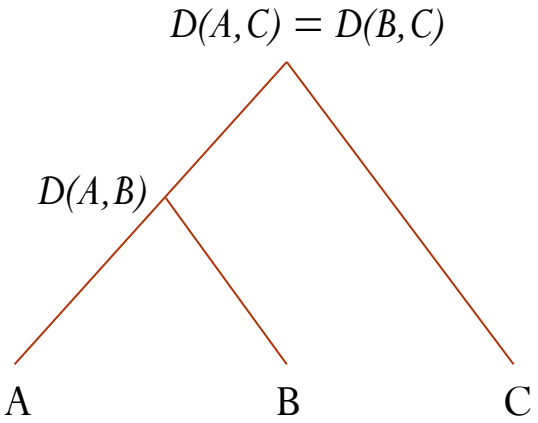
T, s'il existe, est une représentation compacte de *D*.

Remarque: *T* a au plus *n-1* nœuds internes. Donc, si *D* a plus de *n-1* valeurs, il n'existe pas d'arbre ultramétrique pour *D*.

Distance ultramétrique

- *D* est ultramétrique si et seulement si *D* satisfait la condition des 3 points: Pour tout choix de 3 feuilles A, B,C, **parmi les trois distances *D(A,B), D(A,C) et D(B,C),* deux sont égales et supérieures à la troisième.**

Preuve: => **Évident**



Distance ultramétrique

_D_ est ultramétrique si et seulement si _D_ satisfait la condition des 3 points:

Preuve: <= Méthode constructive.

Procéder pour chaque ligne i de *D*.

- Pour chaque ligne, former les clades en fonction de leur distance avec i.
- Ordonner les clades dans l'arbre par ordre croissant de distance.
- Itérer sur chaque clade non-résolu.

Distance ultramétrique

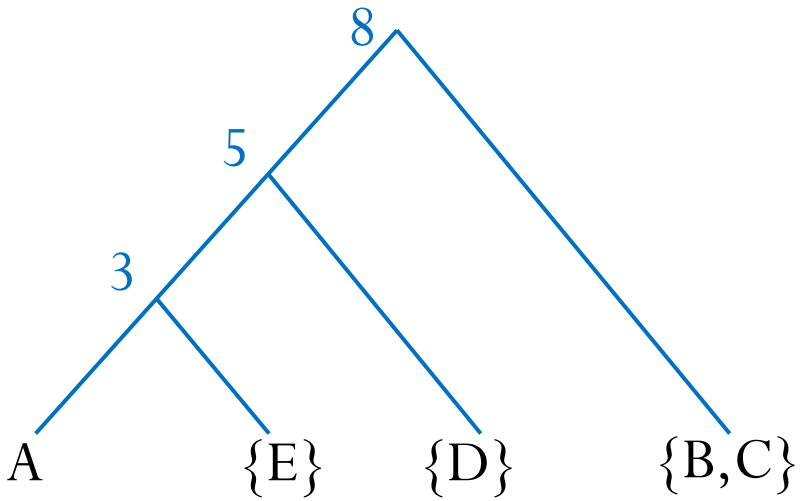
Exemple:

(a)

	A	B	C	D	E
A	0	8	8	5	3
B		0	3	8	8
C			0	8	8
D				0	5
E					0

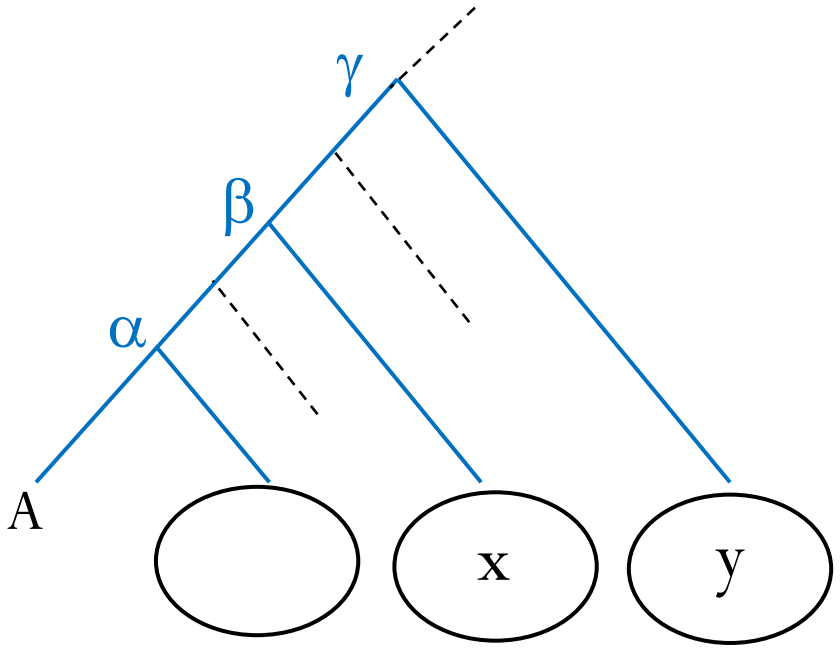
Figure 1: (a) Matrice symétrique D.
à D.

Clades: {E}, {D}, {B,C}



Distance ultramétrique

Cas général:



D(x,y) = γ ?

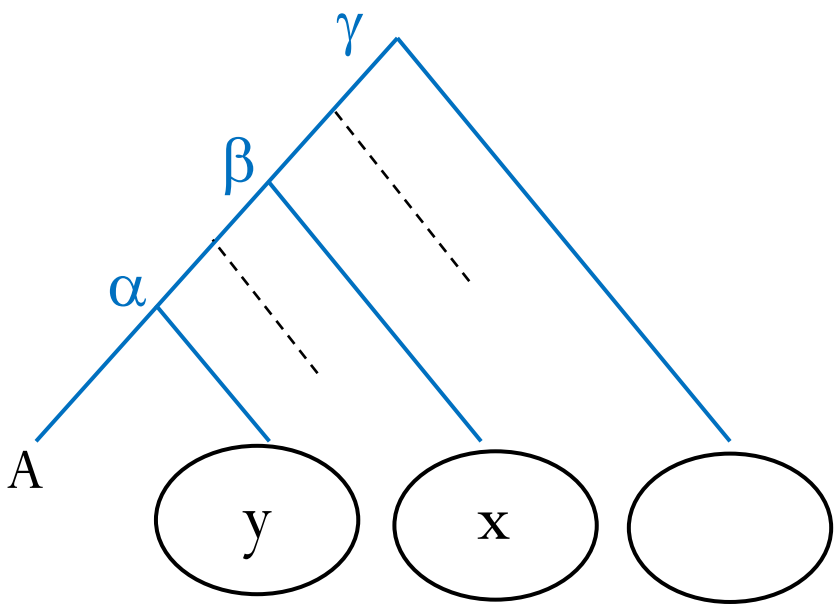
Ce qu'on sait par construction:

$D(A,x) = \beta,\ D(A,y) = \gamma$ et $D(A,x) < D(A,y)$

La condition des 3 points => D(x,y) = D(A,y) = γ

Distance ultramétrique

Cas général:



D(x,y) = β ?

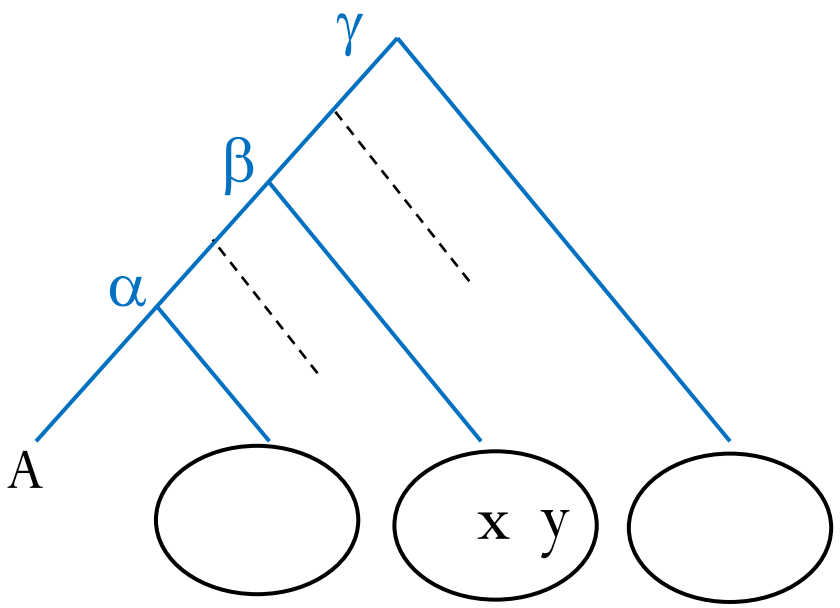
Ce qu'on sait par construction:

$D(A,x) = \beta$, $D(A,y) = \alpha$ et $D(A,x) > D(A,y)$

La condition des 3 points => D(x,y) = D(A,x) = β

Distance ultramétrique

Cas général:



$D(x,y) < \beta$?

Ce qu'on sait par construction:

$D(A,x) = D(A,y) = \beta,$

La condition des 3 points => $D(x,y) < \beta$

Distance/arbre ultramétrique

Théorème: Si D est une matrice ultramétrique, alors l'arbre ultramétrique de D est unique.

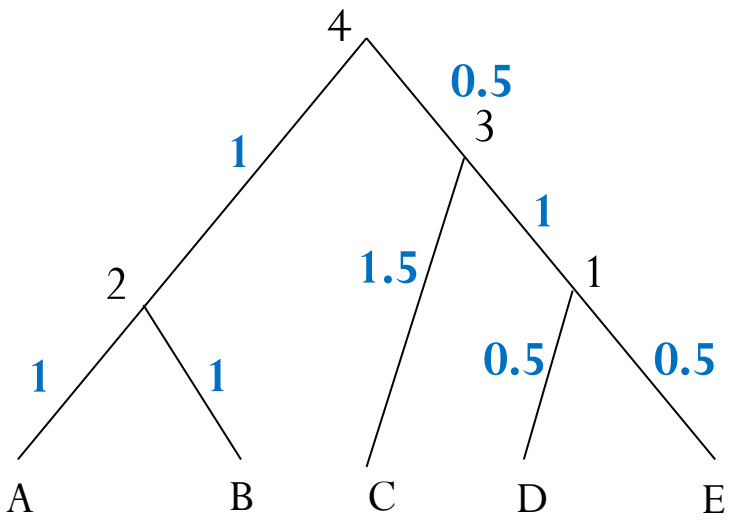
En effet, dans la preuve précédente, les classes sont « forcées ».

Conséquence: Si D reflète effectivement la distance d'évolution entre les espèces, alors l'arbre obtenu est nécessairement le vrai arbre.

Théorème: Si D est ultramétrique, alors l'arbre ultramétrique peut-être construit en temps $O(n^2)$. De plus, on peut déterminer en $O(n^2)$ si une distance est ultramétrique ou non.

Distance ultramétrique

- Si *T* est un arbre ultramétrique pour *D*, **il existe un étiquetage des branches de *T* qui réalise *D*** tel que tous les chemins de la racine à n'importe quelle feuille sont de même longueur.
- Un tel arbre satisfait la théorie de **l'horloge moléculaire**: taux de mutation constant sur toutes les branches.



Que signifient des données ultramétriques?

- Distances étiquetant les arbres ultramétriques supposées refléter le temps qui s'est écoulé depuis la séparation des deux espèces.
- *Théorie de l'horloge moléculaire* (1960): Pour une protéine donnée, le taux de mutations acceptées par intervalle de temps est constant.
- Donc, si k mutations acceptées entre les protéines A et B, on estime à $k/2$ le nombre de mutations survenues sur chaque branche depuis l'ancêtre commun de A et B.

Algorithme UPGMA

- UPGMA: Algorithme de classification ascendante hiérarchique.
- Procède par regroupement des séquences les plus proches. À chaque étape, les deux regroupements les plus « proches » sont fusionnés.
- Si *D* est une distance ultramétrique, alors UPGMA construit l'arbre ultramétrique associé.

Algorithme UPGMA

- n séquences; $D_{i,j}$: Distance entre les séquences i et j.
- D_{ij} *d*istance entre deux regroupements C_i et C_j : Moyenne des distances des paires de séquences entre les deux regroupements.

$$d_{i,j} = \frac{1}{|C_i||C_j|} \Sigma_{p \in C_i, q \in C_j} D_{p,q}$$

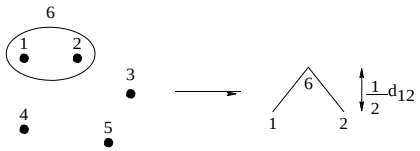
- Si $C_k = C_i \cup C_j$ et C_l est un autre regroupement, alors:

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

1: A T G T T C G
2: A T C T T T G
3: G G C T A C G
4: G C C T T G C
5: G C A T T C G

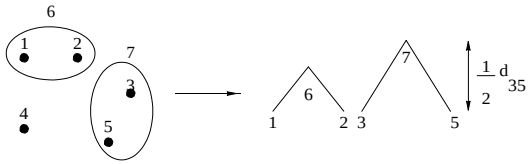
(I)

	1	2	3	4	5
1		2	4	5	3
2			4	4	4
3				4	3
4					3
5					



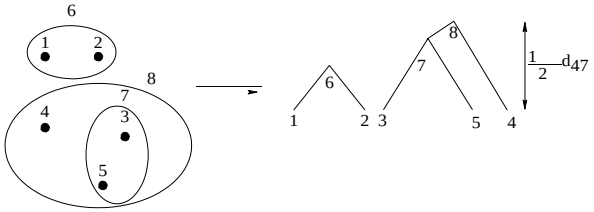
(II)

	3	4	5	6
3		4	3	4
4			3	4.5
5				3.5
6				



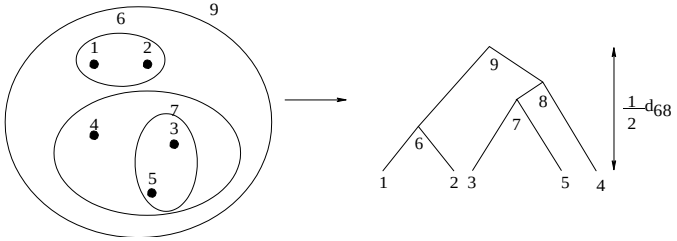
(III)

	4	6	7
4		4.5	3.5
6			3.75
7			



(IV)

	6	8
6		4
8		



Algorithme UPGMA :

Initialisation:

Définir n regroupements, chaque regroupement C_i contenant la seule séquence numéro i;

Définir un arbre T restreint à un ensemble n de feuilles, une feuille pour chaque regroupement. Toutes les feuilles sont de hauteur 0;

Itération

Considérer deux regroupements C_i, C_j tels que d_{ij} soit minimal;

Définir un nouveau regroupement $C_k = C_i$ ***union*** C_j, et définir les d_{kl} pour out l;

Définir un nouveau nœud k de fils i, j et placer le à une hauteur $d_{ij}/2$;

Rajouter C_k dans l'ensemble des regroupements et éliminer C_i et C_j;

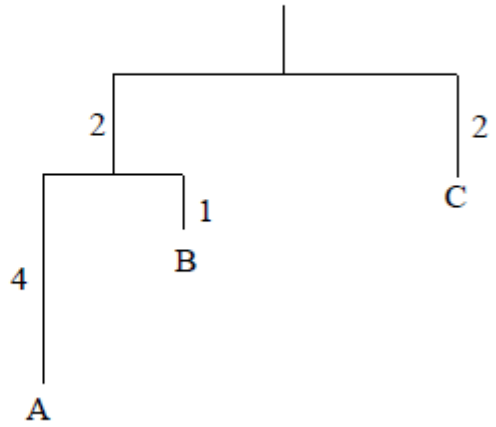
Fin

Lorsqu'il ne reste plus que deux regroupements C_i et C_j, placer la racine à une hauteur $d_{ij}/2$;

Distance/arbre additif

- *T* arbre additif pour *D* si pour toute paire de nœuds *(i,j)*, le poids total du chemin de *i* à *j* est D(i,j).

	A	B	C
A	0	5	8
B		0	5
C			0



Distance/arbre additif

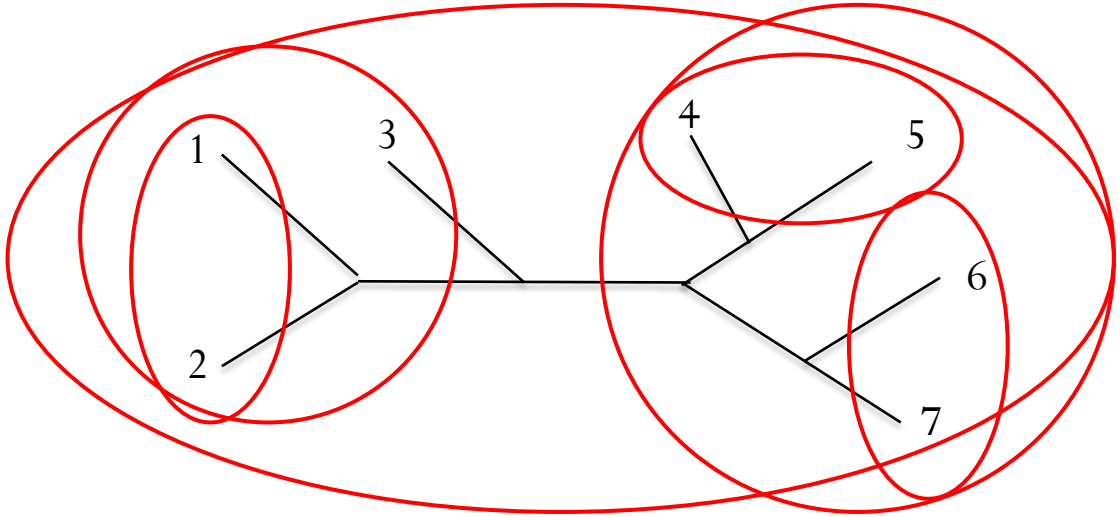
- Problème: Trouver un arbre additif pour D ou déterminer qu'un tel arbre n'existe pas.

- Théorème: Il existe un arbre additif pour D ssi D est une distance additive (i.e. vérifie la condition des 4 points).

- Distance additive: Contrainte moins forte que la contrainte ultramétrique. Une distance ultramétrique est additive. Le contraire n'est pas vrai.
- Cependant, les données réelles sont rarement additives

Neighbor-Joining (Saitou et Nei en 1986)

- Algorithme glouton qui choisit à chaque étape une paire de feuilles voisines.
- Paire de feuilles voisines: Deux feuilles de T ayant le même père.

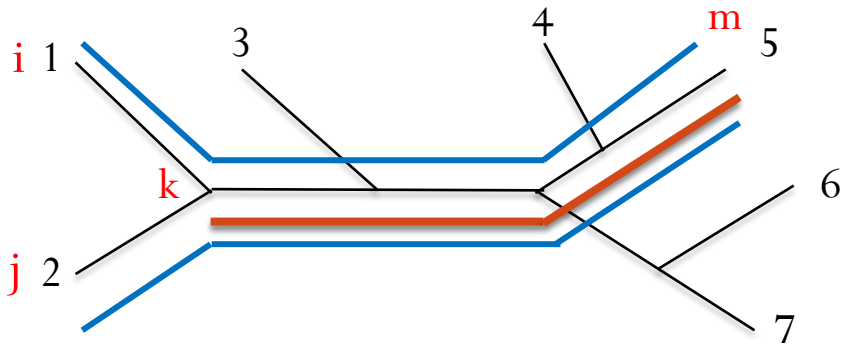


(1,2), (6,7), (4,5), ((1,2),3), ((4,5),(6,7)), (((4,5),(6,7)), (1,2),3))

- Un arbre est déterminé par l'ensemble des (n-2) paires de voisins qu'il contient.

Neighbor-Joining

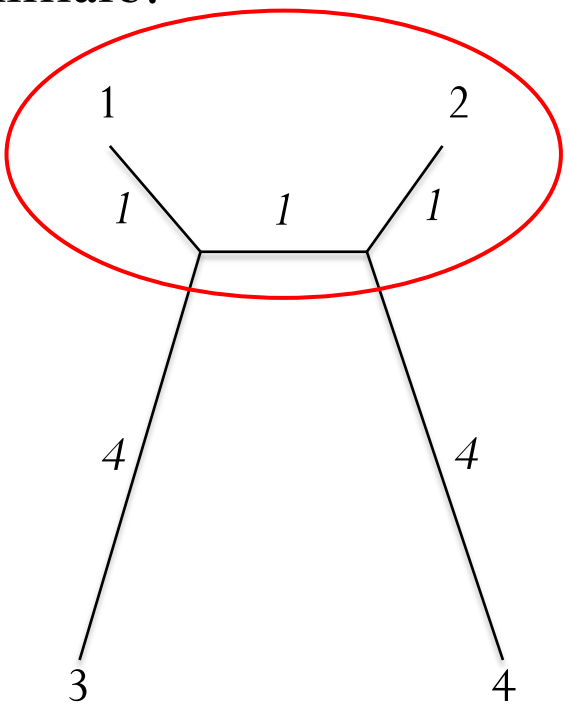
- Choisir deux objets *i, j* garantis d'être voisins dans un arbre additif.
- Supprimer *i, j* de la liste des objets et rajouter le nœud créé *k* correspondant au père commun de *i* et *j*.
- Distance de *k* à une feuille *m* quelconque:



$$D(k,m) = 1/2\ (D(i,m) + D(j,m) - D(i,j))$$

Neighbor-Joining

- Comment déterminer, à partir de *D*, deux feuilles qui sont nécessairement voisines dans un arbre additif de *D*?
- Il ne suffit pas de choisir une paire d'objets dont la distance est minimale:



(1, 2) de distance minimale, mais pas voisines dans l'arbre.

Neighbor-Joining

- L: Ensemble des feuilles d'un arbre additif.
- Pour tout (i,j), $\mathcal{D}(i,j)$: valeur obtenue en soustrayant de $D(i,j)$ la distance moyenne de i et j à toutes les autres feuilles.

$$\mathcal{D}(i,j) = D(i,j) - (r_i + r_j)$$

$$r_i = \frac{1}{|L| - 2} \sum_{k \in L} D(i,k)$$

Théorème: Si T est un arbre additif pour la distance additive D, si (i,j) est une paire de feuille telle que $\mathcal{D}(i,j)$ est minimal parmi toutes les paires de feuilles, alors i et j sont voisines dans T.

Algorithme Neighbour-joining

Initialisation:

T ensemble de toutes les feuilles, une pour chaque objet, et $L = T$.

Itération:

Considérer une paire (i, j) de L telle que $\mathcal{D}(i, j)$ est minimal.

Définir un nouveau nœud k et poser:

$D(k, m) = \frac{1}{2}(D(i, m) + D(j, m) - D(i, j))$ pour tout $m \in L$.

Rajouter k dans T, créer deux arêtes (i, k) et (j, k) de poids

$D(i, k) = \frac{1}{2}(D(i, j) + r_i - r_j)$ et $D(j, k) = D(i, j) - D(i, k)$.

Supprimer i et j de L et rajouter k.

Fin:

L contient exactement deux feuilles i, j.

Créer une arête (i, j) de poids $D(i, j)$.

$\frac{1}{2}(D(i, j) + r_i - r_j)$: moyenne de $\frac{1}{2}(D(i, j) + D(i, m) - D(j, m))$ pour toutes les feuilles m. Chacune de ces expressions représente exactement $D(i, k)$

Méthode de Fitch-Margoliash

- Méthode des moindres carrés : Déterminer en même temps l'arbre et la valeur de longueur des branches δ_{ij} de telle sorte à minimiser

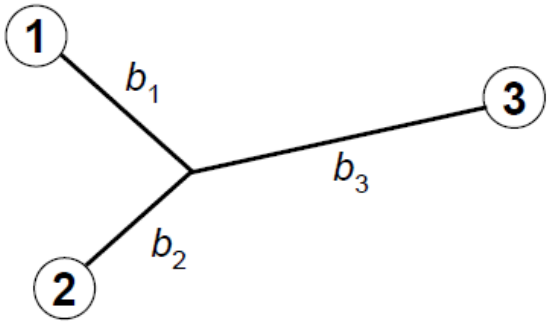
$$Q = \sum_{i<j} w_{ij}(d_{ij} - \delta_{ij})^2$$

Où w_{ij} est la valeur de pondération associée à chaque paire (i,j). On peut tout simplement considérer $w_{ij} = 1$ pour tout (i,j).

- Algorithme de Fitch-Margoliash permet d'approximer la valeur de Q.

Méthode de Fitch-Margoliash

- L'idée est de ramener à un arbre à trois groupes :



$$d_{12} = b_1 + b_2$$
$$d_{13} = b_1 + b_3$$
$$d_{23} = b_2 + b_3$$

$$b_1 = (d_{12} + d_{13} - d_{23})/2$$
$$b_2 = (d_{12} + d_{23} - d_{13})/2$$
$$b_3 = (d_{13} + d_{23} - d_{12})/2$$

Dans le cas d'un arbre à plus de trois taxons, l'algorithme complet se décompose en six étapes :

1. Identifier les deux taxons (notés A et B) pour lesquels la valeur de d_{ij} est minimale.
2. Regrouper tous les autres taxons dans un groupe noté C. La distance entre A et C est donnée par la moyenne des distances séparant A de tous les éléments de C, et la distance entre B et C par la moyenne des distances séparant B de tous les éléments de C.
3. Grâce à cette opération le système est réduit à un ensemble à trois groupes pour lequel il est possible de calculer les valeurs des longueurs de branches au moyen de (4.2). Comme dans le cas de la classification ascendante hiérarchique, il faut soustraire à ces valeurs les longueurs des branches déjà calculées.
4. Regrouper A et B dans un même ensemble $\{A, B\}$, puis calculer pour chaque taxon restant (ceux qui étaient regroupés en C à l'étape 2) la moyenne des distances le séparant de $\{A, B\}$, ceci afin de construire une nouvelle matrice D.
5. S'il reste trois éléments ou plus dans D, retourner en 1, sinon passer à l'étape suivante.
6. Calcul de la valeur de Q.

« Concepts et méthodes en phylogénie moléculaire »,
Guy Perrière et Céline Brochier-Armanet, Springer, 2010

Exemple –Fitch-Margoliash

	Homme	Chimpanzé	Gorille	Orang-outan
Chimpanzé	0.092			
Gorille	0.106	0.111		
Orang-outan	0.177	0.193	0.188	
Gibbon	0.207	0.218	0.218	0.219

Table 4.1 – Distances calculées avec le modèle K80 entre les séquences d'ADN mitochondrial de cinq espèces d'Hominoïdes.

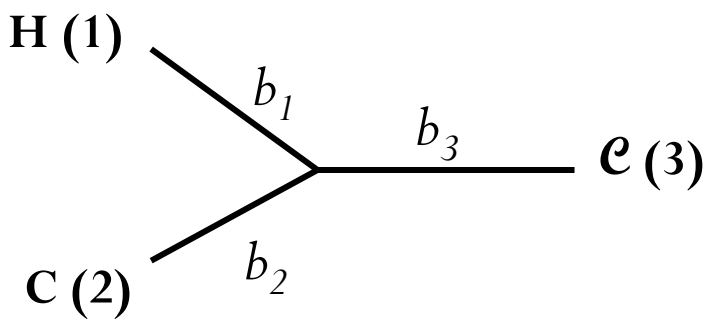
1. Prendre les deux taxons les plus proches

   → (Chimpanzé, Homme) ou (C,H)

2. Mettre tous les autres dans un seul regroupement

   → $\mathcal{C}$ = {Go,O,Gi}

3.



$$b_1 = (d_{12} + d_{13} - d_{23})/2$$
$$b_2 = (d_{12} + d_{23} - d_{13})/2$$
$$b_3 = (d_{13} + d_{23} - d_{12})/2$$

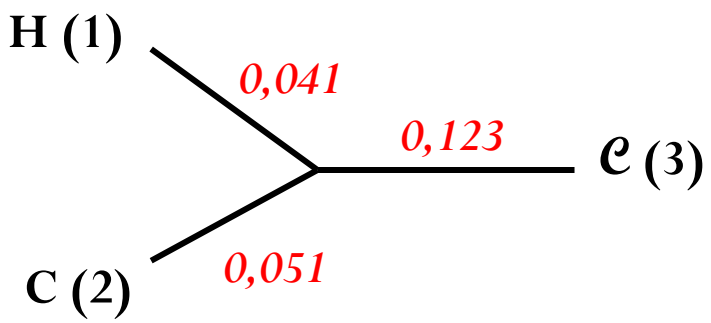
→ $d_{12} = 0.092$

$d_{13} = (0.106+0.177+0.207)/3 = 0.163$

$d_{23} = (0.111+0.193+0.218)/3 = 0,174$

→ $b_1 = 0,041$; $b_2 = 0,051$; $b_3 = 0,123$

3.



$$b_1 = (d_{12} + d_{13} - d_{23})/2$$
$$b_2 = (d_{12} + d_{23} - d_{13})/2$$
$$b_3 = (d_{13} + d_{23} - d_{12})/2$$

→ $d_{12} = 0.092$

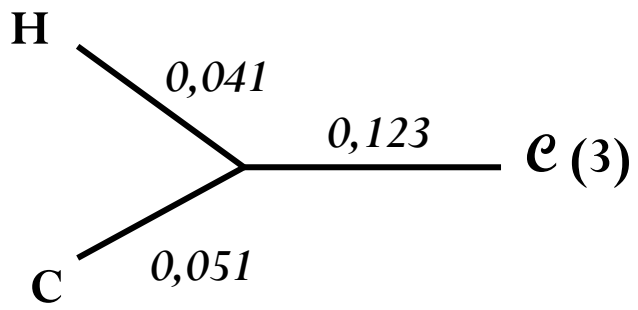
$d_{13} = (0.106+0.177+0.207)/3 = 0.163$

$d_{23} = (0.111+0.193+0.218)/3 = 0,174$

→ $b_1 = 0,041$; $b_2 = 0,051$; $b_3 = 0,123$

4.

	(C,H)	Go	O	Gi
(C,H)				
Go				
O		0.188		
Gi		0.218	0.219	

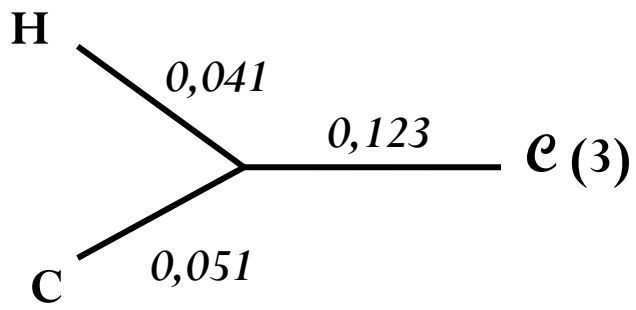


4.

	(C,H)	Go	O	Gi
(C,H)				
Go	0.109			
O	0.185	0.188		
Gi	0.213	0.218	0.219	

1. Prendre les deux taxons les plus proches

→ ((C,H), Go)



4.

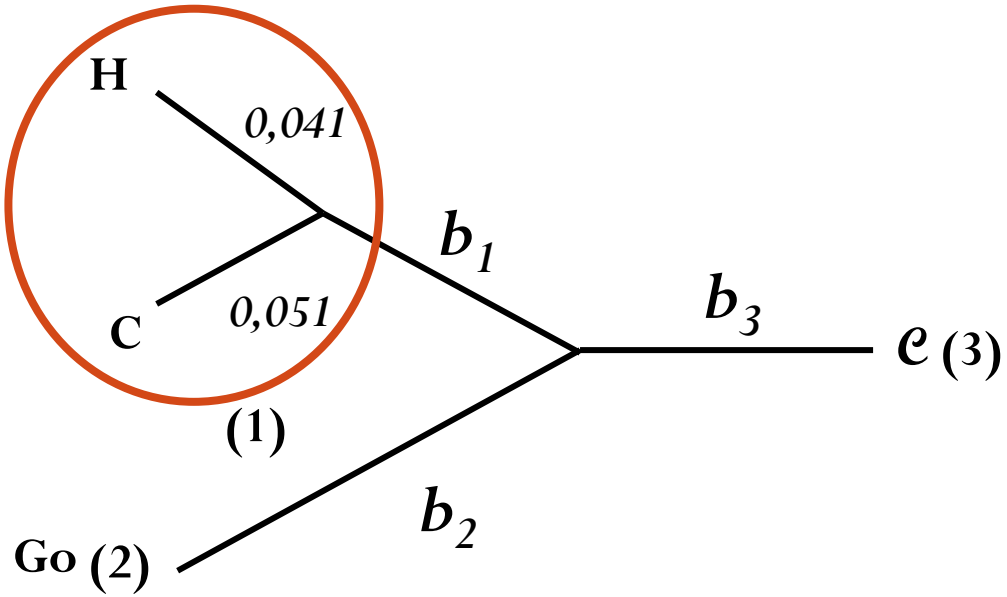
	(C,H)	Go	O	Gi
(C,H)				
Go	0.109			
O	0.185	0.188		
Gi	0.213	0.218	0.219	

1. Prendre les deux taxons les plus proches

→ ((C,H), Go)

2. Mettre tous les autres dans un seul regroupement

→ $\mathcal{e}$ = {O,Gi}



4.

	(C,H)	Go	O	Gi
(C,H)				
Go	0.109			
O	0.185	0.188		
Gi	0.213	0.218	0.219	

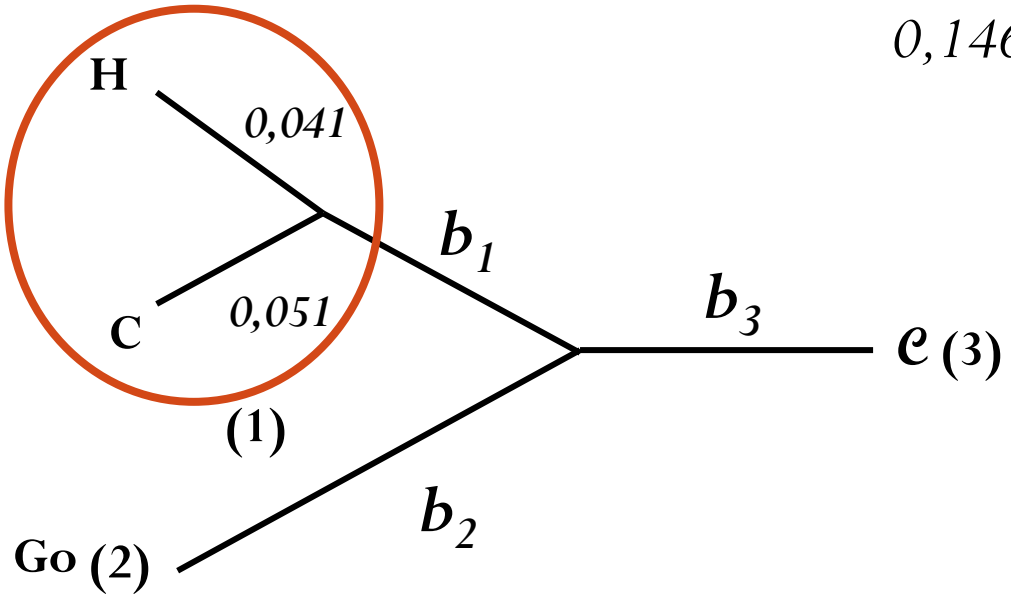
$$b_1 = (d_{12} + d_{13} - d_{23})/2$$
$$b_2 = (d_{12} + d_{23} - d_{13})/2$$
$$b_3 = (d_{13} + d_{23} - d_{12})/2$$

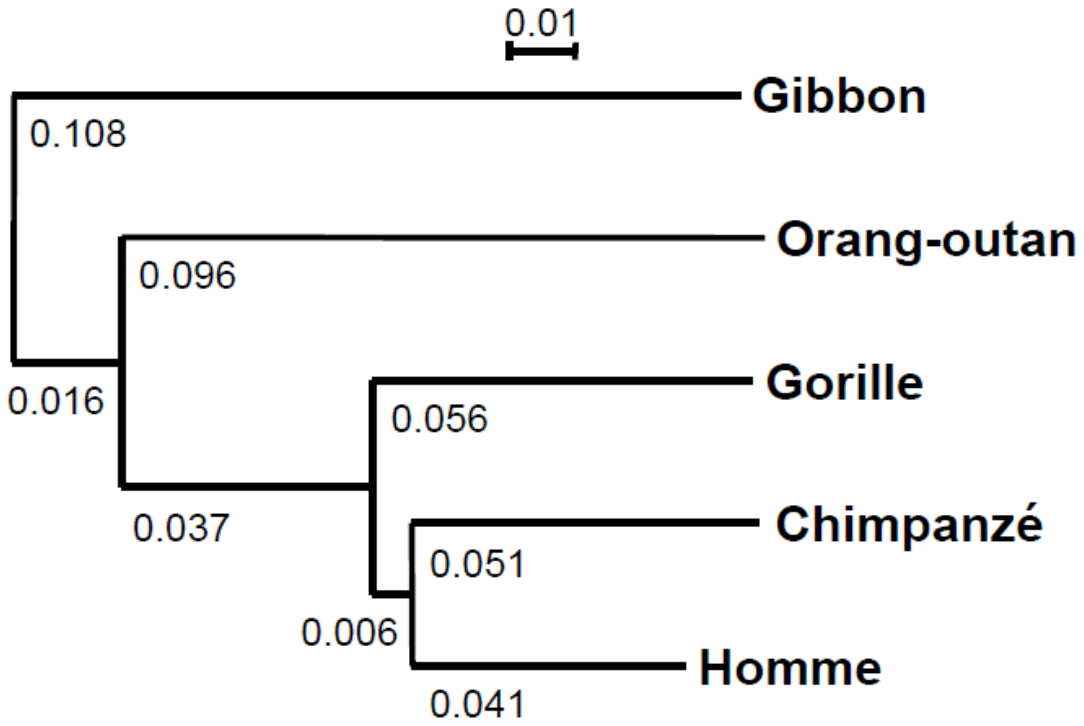
→ $d_{12} = 0.109$

$d_{13} = (0.185+0.213)/2 = 0.199$

$d_{23} = (0.188+0.218)/2 = 0,203$

→ $b_1 = 0,053$; $b_2 = 0,056$; $b_3 = 0,146$





Arbre final obtenu par Fitch-Margoliash

Fitch-Margoliash : Optimisation

- Une fois un premier arbre est construit, le processus est répété en l'initialisant avec toutes les paires d'autres taxons possibles
- Si un seul arbre est considéré, la complexité de Fitch-Margoliash est en $O(n^3)$, comme pour UPGMA.
- Sinon, comme il y a n(n-1)/2 paires de départ possible, complexité en $O(n^5)$.

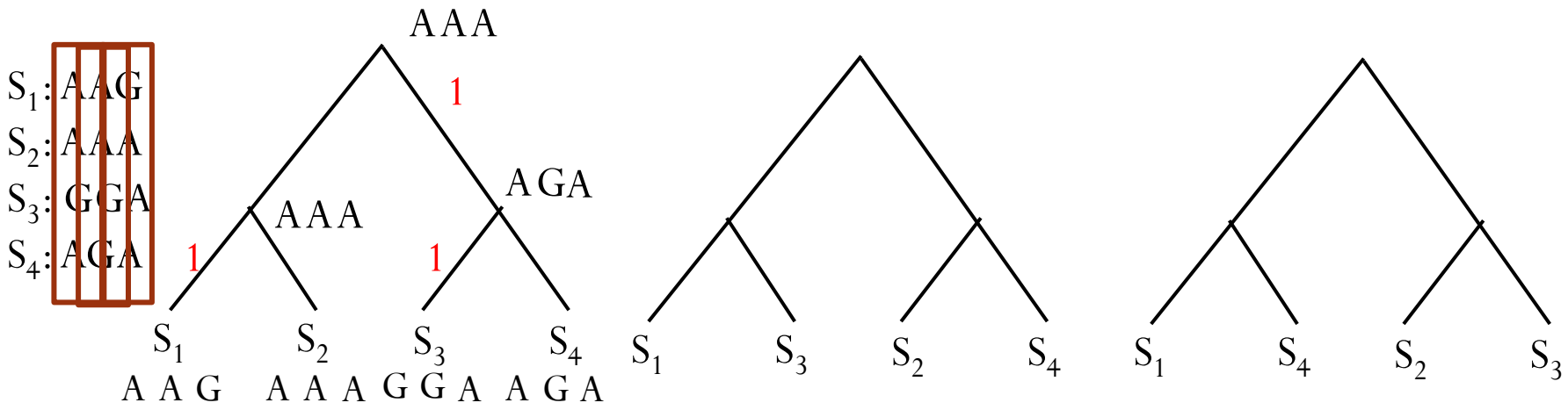
II. Méthode de parcimonie

- Basée sur le principe de maximum de parcimonie. Hypothèse la plus simple: expliquer un processus par le plus petit nombre d'événements.
- À la différence des méthodes de distances, considère chaque site d'un alignement multiple individuellement. Sous-entend l'hypothèse d'indépendance des sites.
- Méthode générale: Pour un ensemble de feuilles:
  - Considérer toutes les topologies d'arbre possibles;
  - Calculer un poids pour chaque arbre;
  - Sélectionner un arbre de poids minimal.

Méthode de parcimonie

- Pondération d'un arbre: Affecter des séquences aux nœuds internes de telle sorte à minimiser le poids total de l'arbre (somme des poids des branches).

  Exemple:

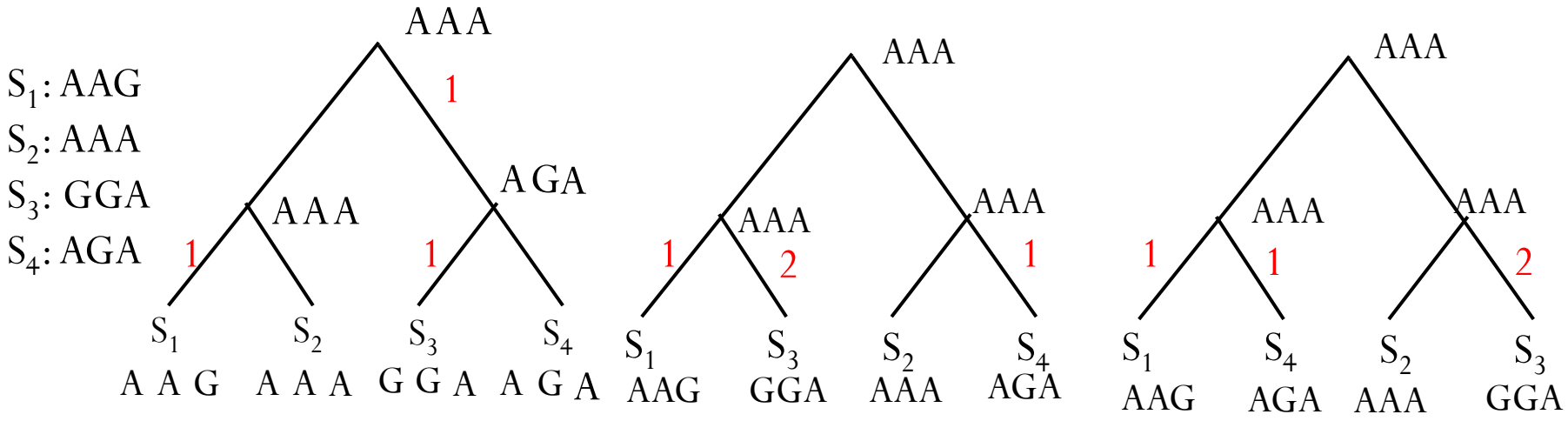


Poids de l'arbre: 3

Méthode de parcimonie

- Pondération d'un arbre: Affecter des séquences aux nœuds internes de telle sorte à minimiser le poids total de l'arbre (somme des distances des branches).

Exemple:



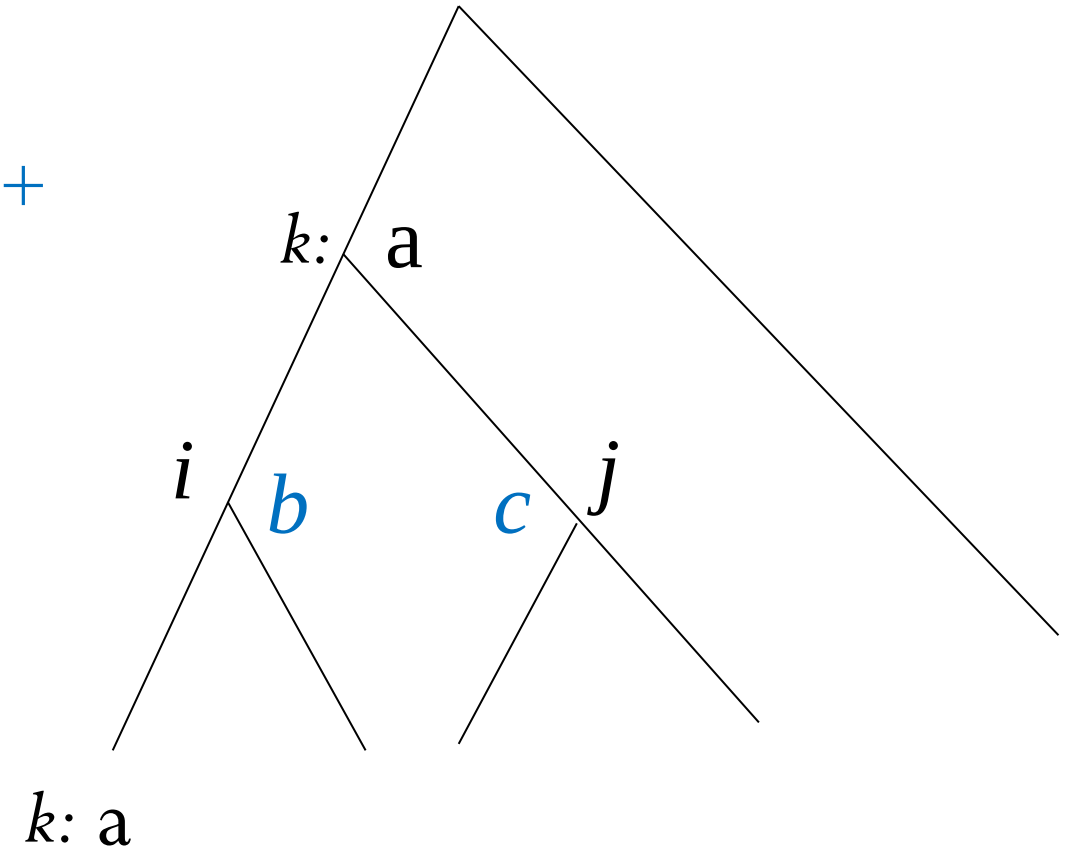
Poids de l'arbre: 3 4 4

Parcimonie pondérée (Algorithme de Sankoff)

- On ne compte pas juste le nombre de substitutions, mais un poids $S(a, b)$ pour la substitution de a en b.
- Étiqueter les nœuds internes de telle sorte a minimiser le poids total de l'arbre.
- Par récurrence: étiquette d'un nœud déduite des étiquettes des nœuds fils.
- $S_k(a)$: poids du sous-arbre de racine k, sous la condition que k est étiqueté par a.

Parcimonie pondérée (Algorithme de Sankoff)

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



$S_k(a) = 0$
$S_k(b) = \infty$

k: a

Parcimonie pondérée (Algorithme de Sankoff)

Algorithme parsimonie pondérée:

Initialisation:

Poser $k = 2n - 1$, le numéro de la racine;

Récurrence - Calculer $S_k(a)$ pour tous les a:

Si k est une feuille

Poser $S_k(a) = 0$ pour a étiquette de k, $S_k(a) = \infty$ si non;

Si k n'est pas une feuille

Calculer $S_i(a), S_j(a)$ pour tous les a, où i, j sont les fils de k;

Poser $S_k(a) = \min_b(S_i(b) + S(a,b)) + \min_b(S_j(b) + S(a,b))$;

Fin:

Poids minimal de l'arbre $= \min_a S_{2n-1}(a)$;

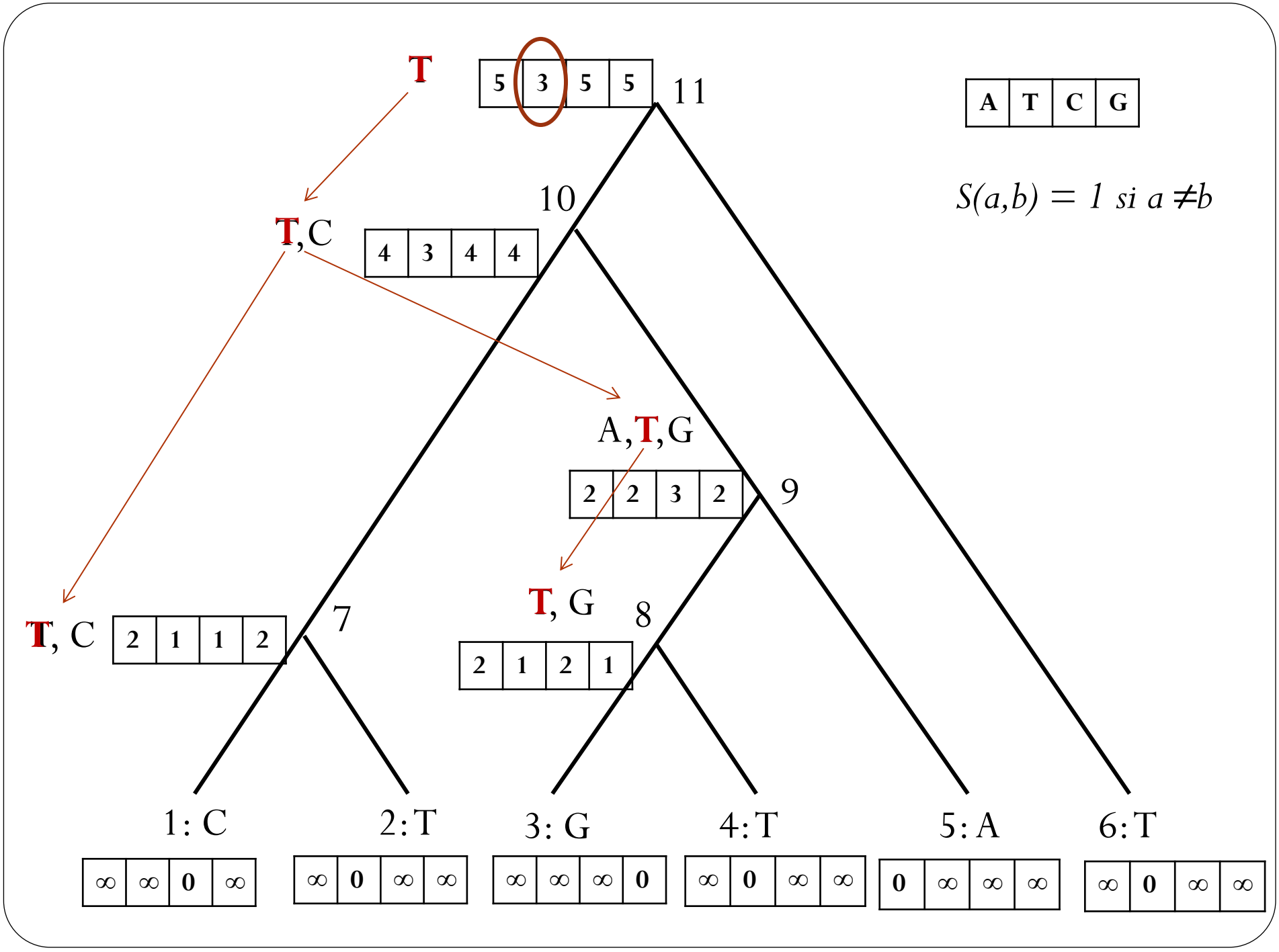
Parcimonie pondérée (Algorithme de Sankoff)

- Pour retrouver les nucléotides aux nœuds internes, garder des pointeurs $l_k(a)$ et $r_k(a)$ pour chaque a et chaque nœud k, et rajouter les deux instructions suivantes dans le bloc de récurrence:

  Poser $l_k(a) = argmin_b(S_i(b) + S(a,b))$

  Poser $r_k(a) = argmin_b(S_j(b) + S(a,b))$

- Pour retrouver une assignation correcte pour les nœuds internes, choisir un nucléotide à la racine qui donne lieu à un poids $S_{2n-1}(a)$ minimal, et suivre les pointeurs.
- Complexité: Pour un nœud donné, il faut calculer $2|\Sigma|^2$ minimas. D'où, complexité de l'algorithme en $O(n|\Sigma|^2)$ où n est la taille de l'arbre (nombre de nœuds).


T
5 3 5 5
11
A T C G
S(a,b) = 1 si a ≠b
10
T,C
4 3 4 4
A,T,G
2 2 3 2
9
T, C
2 1 1 2
7
T, G
8
2 1 2 1
1: C
2:T
3: G
4:T
5: A
6:T
∞ ∞ 0 ∞
∞ 0 ∞ ∞
∞ ∞ ∞ 0
∞ 0 ∞ ∞
0 ∞ ∞ ∞
∞ 0 ∞ ∞

Parcimonie traditionnelle
Algorithme de Fitch

- Minimiser le nombre de substitutions de caractères. Garder à chaque nœud une liste de nucléotides « valides ».

C: poids courant de l'arbre.

Initialisation:

Poser $k = 2n - 1$, le numéro de la racine, et $C = 0$;

Récurrence:

Si k est une feuille

Poser $R_k = \{\text{étiquette de } k\}$;

Si k n'est pas une feuille

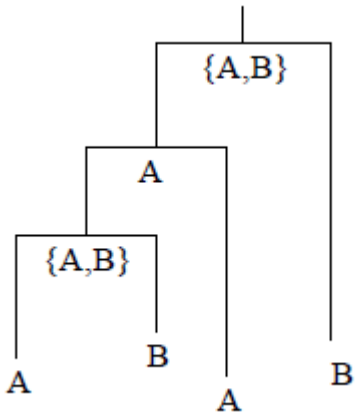
Calculer R_i, R_j, où i, j sont les fils de k;

Si $R_i \cap R_j \neq \emptyset$, poser $R_k = R_i \cap R_j$;

Si non, $R_k = R_i \cup R_j$ et incrémenter C;

Fin:

Poids minimal de l'arbre $= C$;

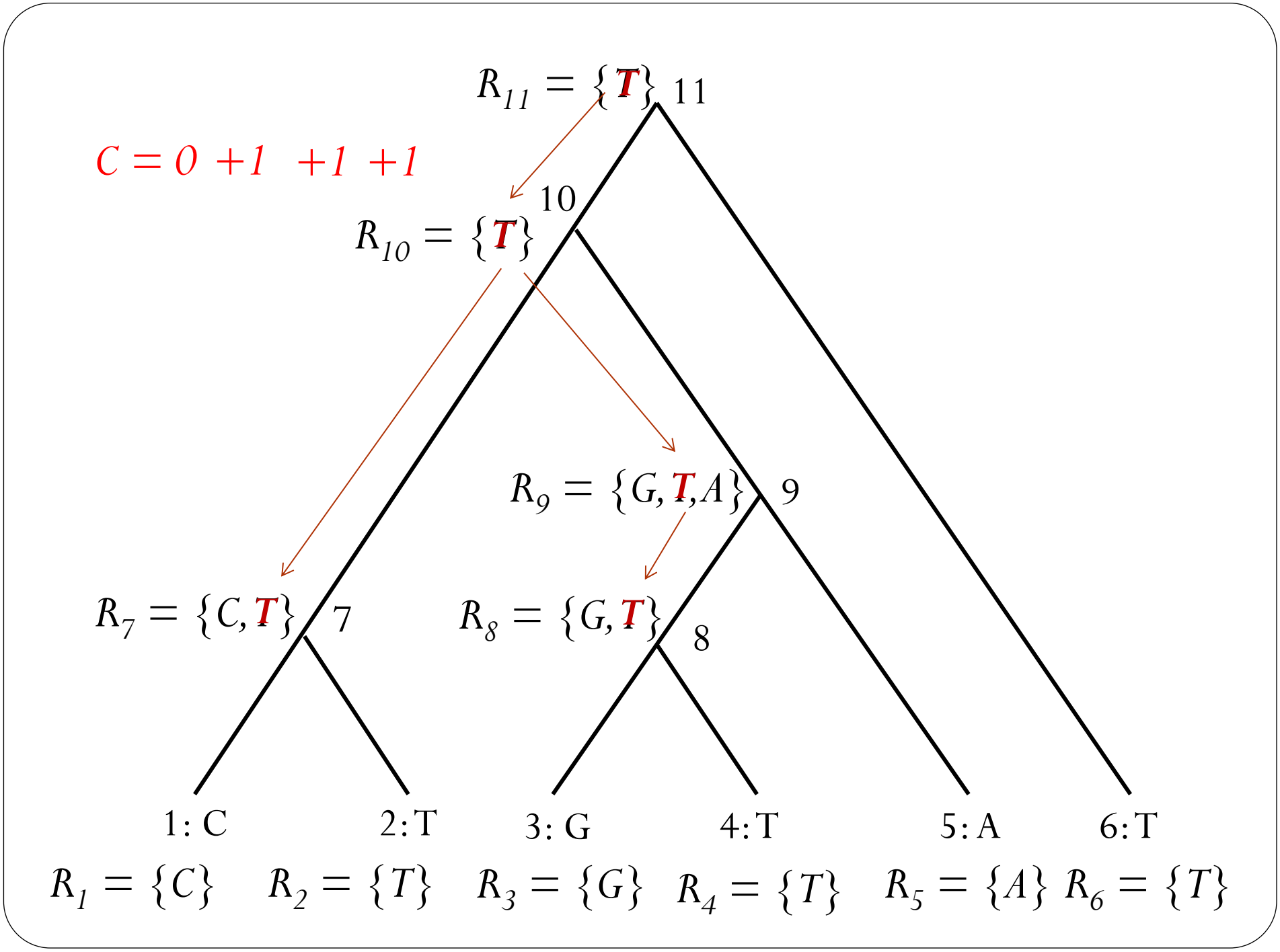


Parcimonie traditionnelle
Algorithme de Fitch

- Pour retrouver les nucléotides des nœuds internes: Choisir un nucléotide dans R_{2n-1}(racine) puis descendre dans l'arbre. Si on a choisit a pour k, alors, pour le fils i de k, choisir a si possible, si non choisir un nucléotide au hasard dans R_i.

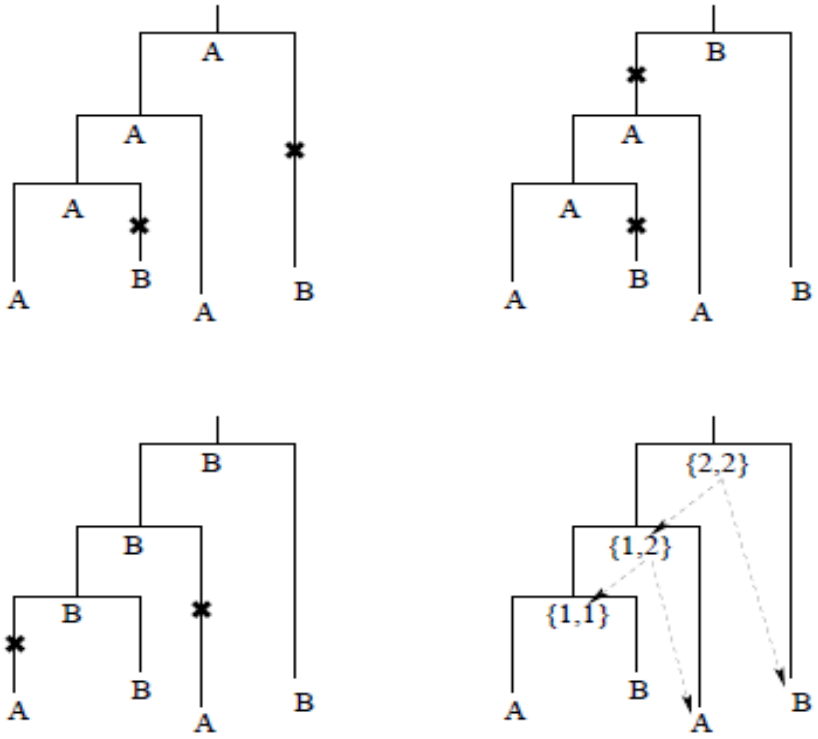
- *Complexité*: $O(n|\Sigma|)$

- Observation: Le poids minimal d'un arbre calculé par l'algorithme de Fitch est indépendant du choix de la racine. Conséquence: on n'a pas besoin de tester tous les arbres racinés possibles.


$R_{11} = \{T\}$ 11
$C = 0 \ +1 \ \ +1 \ +1$
10
$R_{10} = \{T\}$
$R_9 = \{G, T, A\}$ 9
$R_7 = \{C, T\}$ 7
$R_8 = \{G, T\}$
8
1: C
2: T
3: G
4: T
5: A
6: T
$R_1 = \{C\}$ $R_2 = \{T\}$ $R_3 = \{G\}$ $R_4 = \{T\}$ $R_5 = \{A\}$ $R_6 = \{T\}$

Parcimonie traditionnelle
Algorithme de Fitch

- Problème de la parcimonie traditionnelle: Certaines assignations possibles des nœuds internes ne sont jamais considérées.



Énumération de tous les arbres possibles

Arbres binaires enracinés de n feuilles:

n feuilles $\Longrightarrow$ $n-1$ nœuds internes

$\Longrightarrow$ Nombre total de nœuds et feuilles $= 2n-1$

$\Longrightarrow$ $2n-2$ arêtes.

Arbre sans racine: $2n-2$ nœuds et $2n-3$ arêtes.

Pour former un arbre enraciné, on rajoute une racine au milieu d'un des $2n-3$ arêtes $\Longrightarrow$ $2n-3$ arbres enracinés

3 façons de former un arbre sans racine de 4 feuilles à partir d'un arbre sans racine de 3 feuilles.

5 façons de former un arbre sans racine de 5 feuilles à partir d'un arbre sans racine de 4 feuilles$\cdots$

Énumération de tous les arbres possibles

Par récurrence: $(3).(5).(7).\cdots(2n-5)$ arbres sans racine de n feuilles.

$\Longrightarrow$ $(3).(5).(7).\cdots(2n-5)(2n-3) = (2n-3)!!$ arbres enracinés de n feuilles

Le nombre d'arbres croît très rapidement en fonction de n.

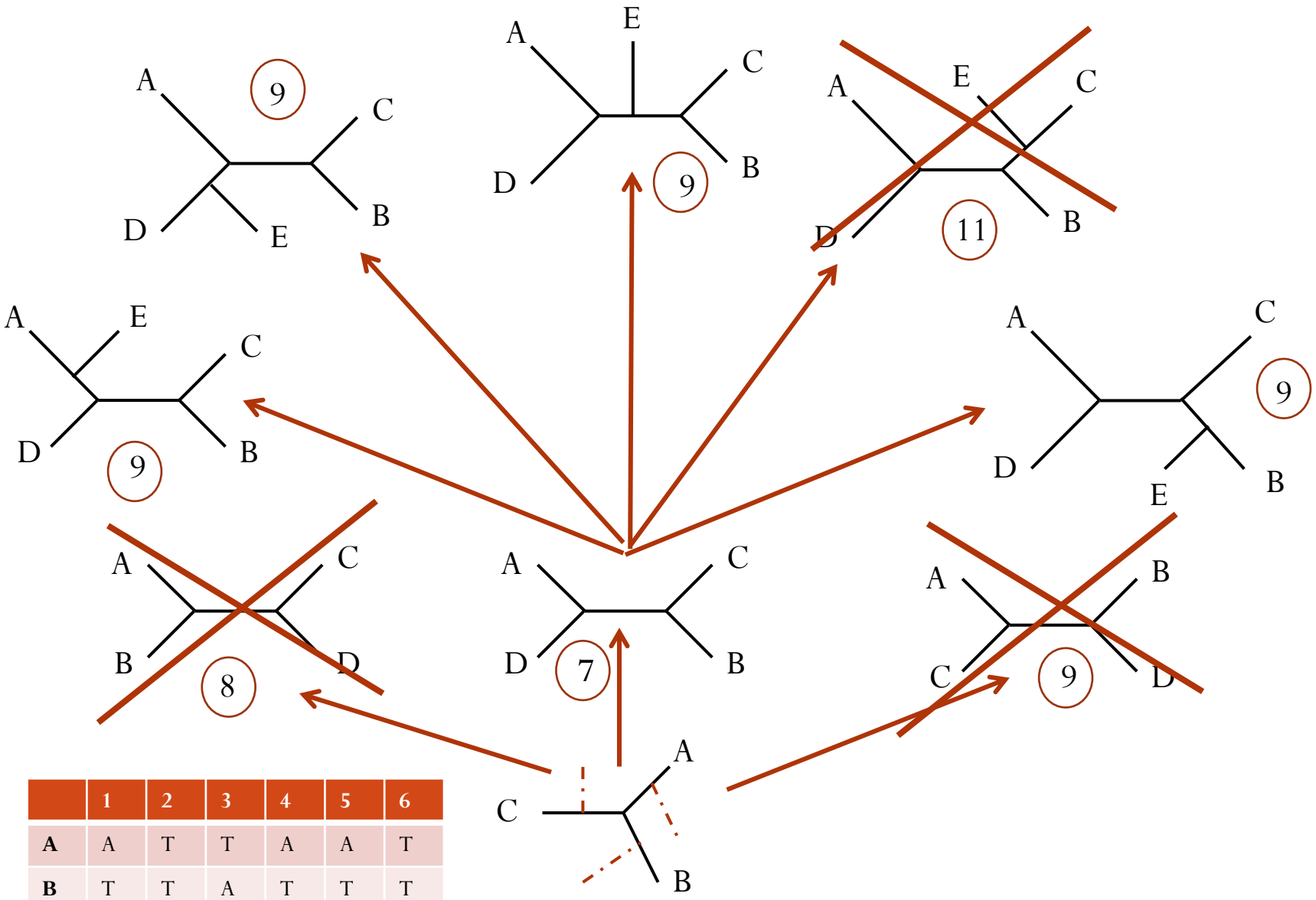
Exemple: Pour $n = 10$, ~ 2000000 arbres. Pour $n = 20$, $\sim 2.2 \times 10^{20}$ arbres.

Exploration des topologies

- Plutôt que de considérer toutes les topologies possibles, on a recours à des heuristiques.
- Méthode générale:
  - Considérer une topologie initiale T,
  - Explorer un voisinage de T: Tous les arbres qui sont à une distance donnée de T. Conserver l'arbre (ou les arbres) qui a le meilleur score.
- Différentes distances utilisées:
  - Nearest Neighbor Interchange (NNI)
  - Subtree Pruning and Regrafting (SPR)
  - …

Choix de l'arbre de départ

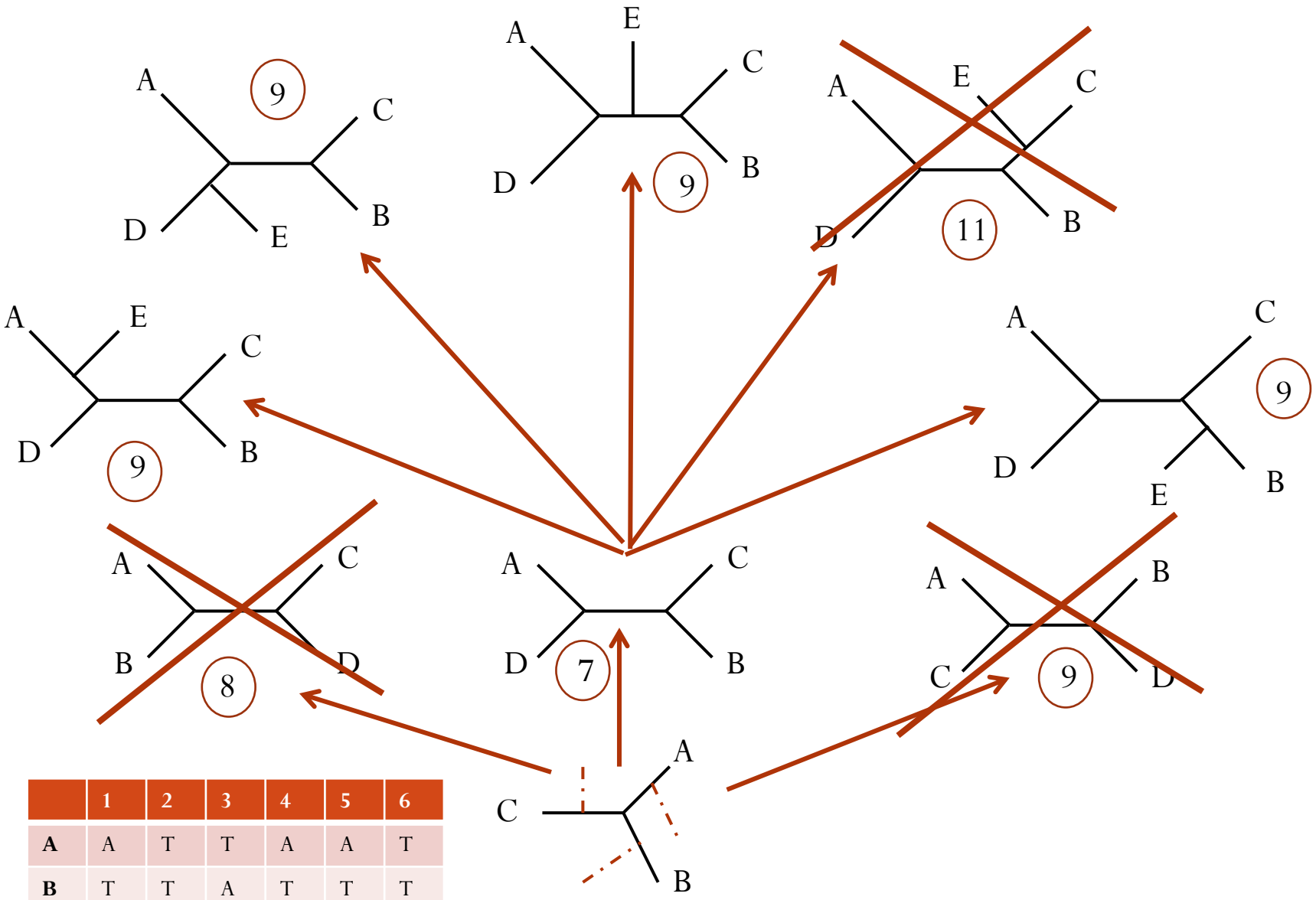
- Recherche séquentielle: Algorithme glouton; construit l'arbre en rajoutant une arête à chaque étape.
  - Construire un arbre *T* non raciné à partir de trois objets de départ (2 objets dans le cas d'un arbre raciné).
  - Pour *T* contenant *r* feuilles, choisir un *r+1*ème objet et le placer de façon optimale dans *T*.
- Choix des taxons à rajouter
  - Aléatoire; pas idéal
  - Approche du *maximum du minimum*: Pour chaque taxon, essayer les 3 positionnements possibles sur l'arbre initial de 3 feuilles. Garder les valeurs minimales. Ordonner les taxon par cette procédure: valeurs décroissantes. Procédure qui augmente la vitesse d'obtension de l'arbre le plus parcimonieux (N. Nei and S. Kumar, 2000)



	1	2	3	4	5	6
A	A	T	T	A	A	T
B	T	T	A	T	T	T
C	A	A	T	T	T	T
D	A	A	T	A	A	A
E	T	T	A	A	A	T

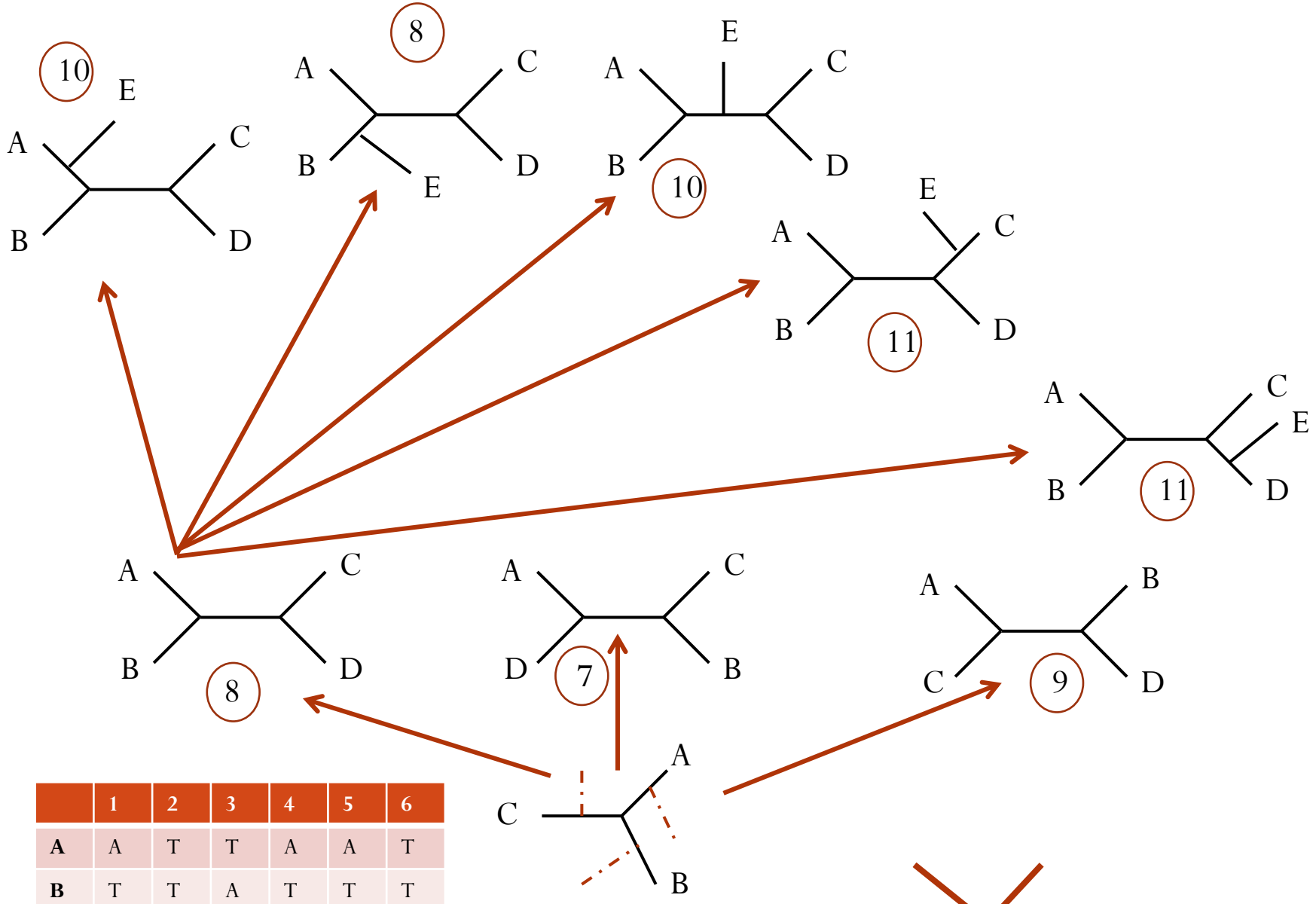
Exploration des topologies par Branch-and-Bound

- Branch-and-bound: Explorer toutes les topologies possibles. Lorsque le poids de l'arbre courant dépasse une certaine borne L_S, arrêter l'insertion de feuilles pour cet arbre.
- L_S: Poids du meilleur arbre complet obtenu au cours de la recherche séquentielle.



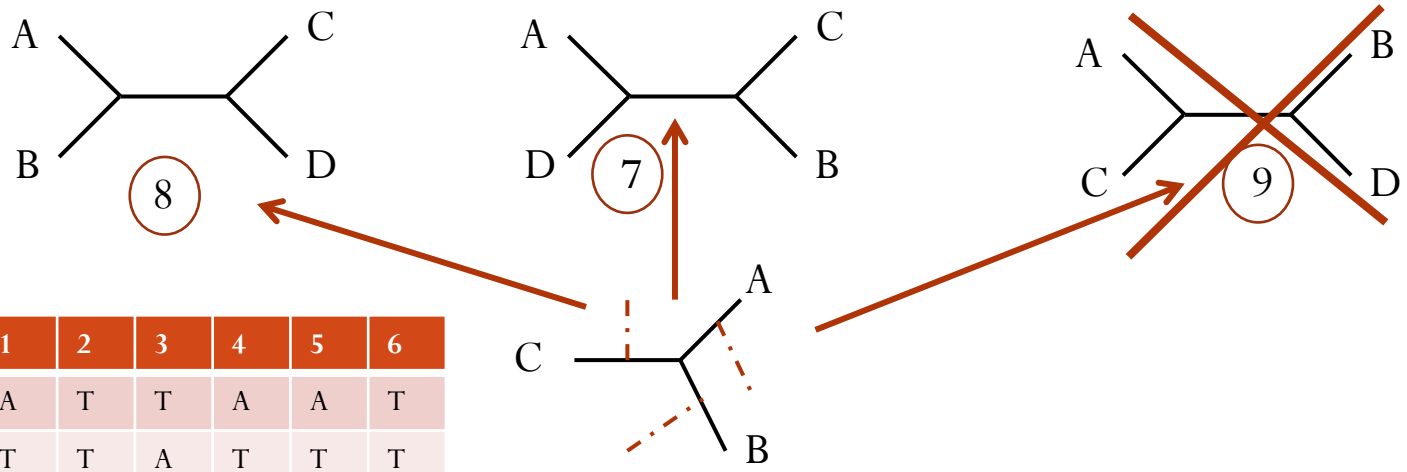
	1	2	3	4	5	6
A	A	T	T	A	A	T
B	T	T	A	T	T	T
C	A	A	T	T	T	T
D	A	A	T	A	A	A
E	T	T	A	A	A	T

$L_S = 9$



	1	2	3	4	5	6
A	A	T	T	A	A	T
B	T	T	A	T	T	T
C	A	A	T	T	T	T
D	A	A	T	A	A	A
E	T	T	A	A	A	T

$L_S = 9$ $L_S = 8$



	1	2	3	4	5	6
A	A	T	T	A	A	T
B	T	T	A	T	T	T
C	A	A	T	T	T	T
D	A	A	T	A	A	A
E	T	T	A	A	A	T

$L_S = 8$

III. Qualité d'un arbre

- Un arbre construit par une méthode phylogénétique n'est qu'une estimation de la véritable histoire évolutive. Comment évaluer la pertinence de cette estimation?
- L'arbre généré peut-être instable: des séquences plus ou moins longues peuvent affecter la topologie obtenue. Comment estimer cette instabilité?
- Bootstrap et jacckknife: Deux méthodes statistiques utilisant une technique de rééchantillonnage. Méthodes empiriques permettant d'inférer la variabilité des paramètres dans le cas de modèles trop complexes pour qu'il soit possible d'en calculer la variance. Particulièrement bien adaptées aux données de la phylogénie moléculaire.

Bootstrap

- Méthode la plus couramment utilisée pour mesurer les incertitudes sur les arbres phylogénétiques. Pour que les résultats soient valides, il faut que la méthode soit consistante.
- Données représentées par des alignements de séquences. Les caractères sont les sites de l'alignement. C'est sur eux que porte le rééchantillonnage.
- *n* séquences de longueur *l*.
  - Effectuer *B* tirages avec remise de *l* sites.

    → *B* matrices de la même taille que l'alignement initial.

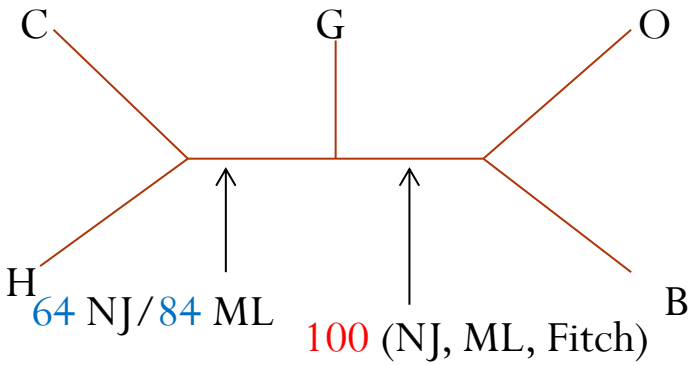
    B: Nombre de réplicats de bootstrap. Doit être suffisemment grand (> 500)
  - Utiliser le même méthode de reconstruction phylogénétique pour chaque matrice (alignement)
  - Tester individuellement la validité de chaque branche de l'arbre de départ, en comptant le nombre de fois que cette branche est présente dans les *B* arbres.

Exemple

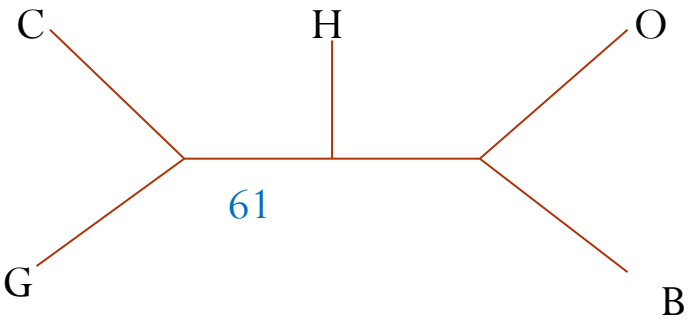
- Brown et al (J.Mol. Evol., vol. 18,1982) ont produit un alignement de courtes séquences d'ADN mitochondrial (895 pb) provenant de 5 espèces d'Hominoïdes: l'Homme (H), le Chimpanzé (C), le Gorille (G), l'Orang-outan (O) et le Gibbon (B). Le Gibbon est le groupe externe (Hylobatidé versus Hominidés).
- Les méthodes de distance et de ML regroupent toutes l'Homme avec le Chimpanzé. Mais pas si on utilise la parcimonie de Fitch. Est-ce que cette différence est due à la méthode ou à la faiblesse du signal phylogénétique?

Exemple

Arbre inféré par toutes les méthodes sauf Fitch



Arbre inféré par Fitch



- Bootstrap 500 réplicats

→ Le regroupement OB est retrouvé dans 100% des réplicats.

→ La quantité de signal phylogénétique en faveur des deux bipartitions concurrentes {BOG|CH} et {BOH|CG} est faible. Données pas suffisantes pour résoudre complètement les relations de parentés entre l'Homme, le Chimpanzé et le Gorille.