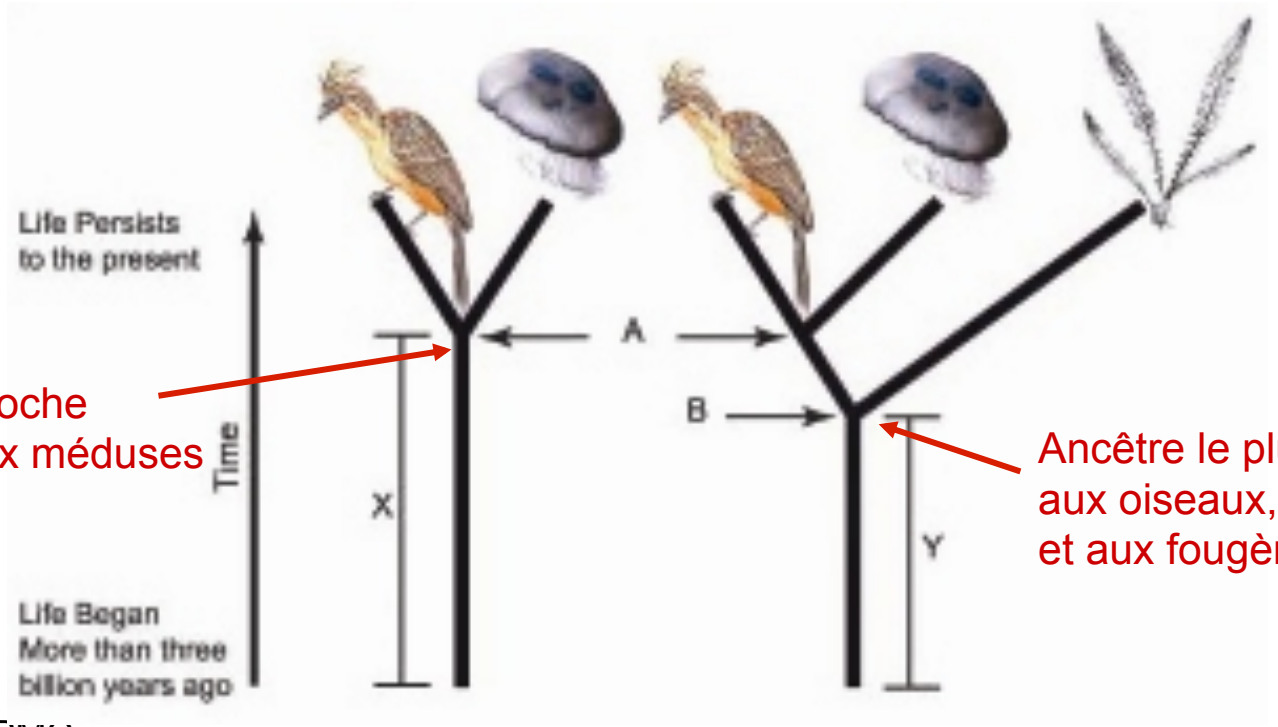
M1 - AAGB / M2 - PHYL

Algorithmes de reconstruction des arbres phylogénétiques

Alessandra Carbone
Université Pierre et Marie Curie

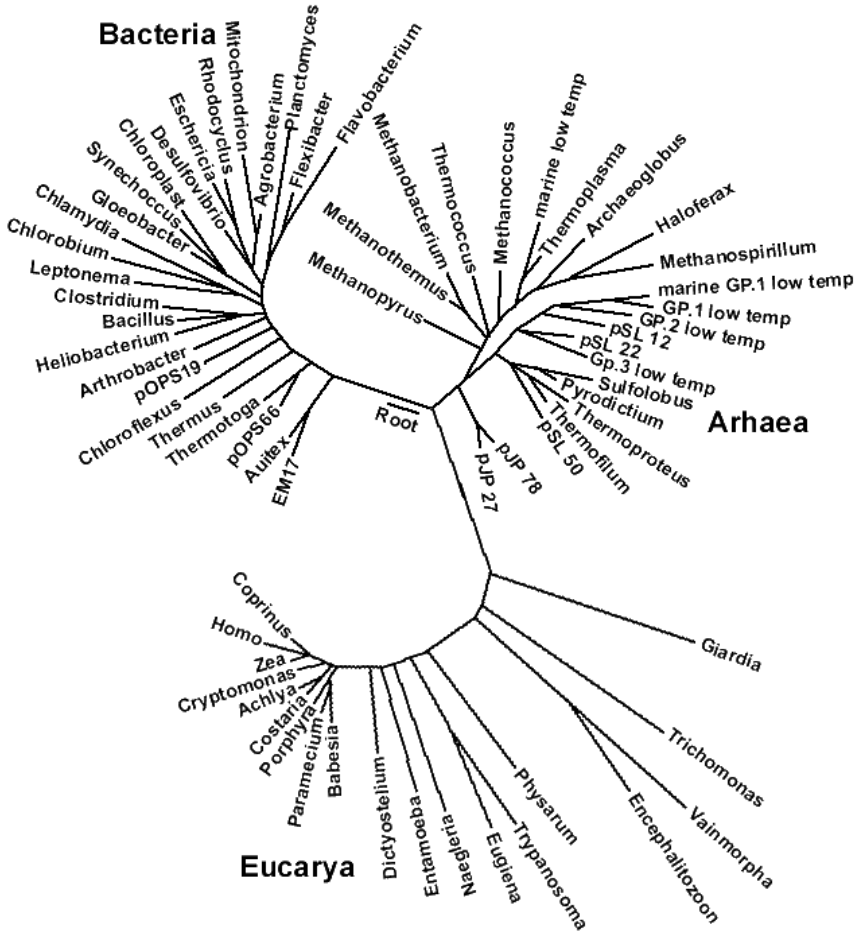
Reconstruction des arbres phylogénétiques

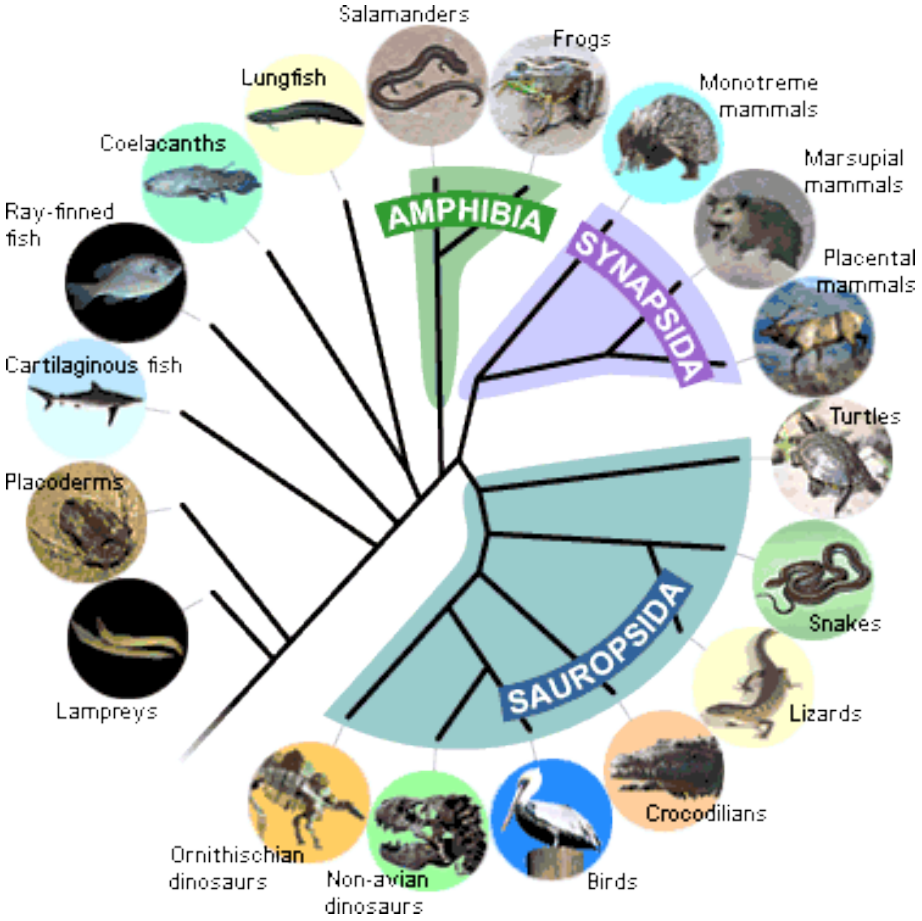
Quelles sont les relations génétiques entre espèces ?
Idée : comparer des caractères spécifiques des espèces, sous l'hypothèse que des espèces similaires soient morphologiquement/génétiquement proches.



Ancêtre le plus proche
aux oiseaux et aux méduses

Ancêtre le plus proche
aux oiseaux, aux méduses
et aux fougères





Phylogénie classique: caractères physiques comme taille, couleur, nombre de pattes …
de Darwin jusqu'au début des années '60…

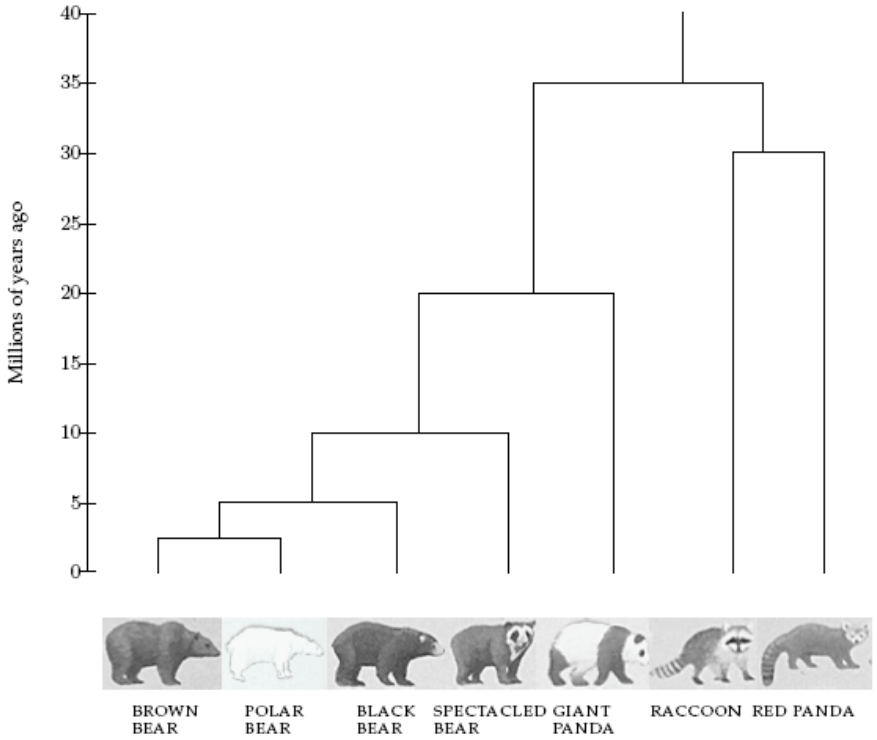
Phylogénie moderne : utilise l'information génétique, séquences d'ADN et de protéines.
Les relations entres espèces sont déduites de bloques très conservés dans l'alignement de plusieurs séquences, une pour chaque espèce considérée.

Basée sur des séquences de 16S Ribosomal RNA

La phylogénie ne s'occupe pas de reconstruire des génomes ancestraux

Evolution et analyse de l'ADN: la devinette du Panda Géant

- Pour presque 100 ans les scientifiques ont été incapable de comprendre à quelle famille les panda géants appartiennent.
- Panda géants ressemblent les ours mais ils ont des caractéristiques assez différent et typique des ratons laveurs, il n'hibernent pas par exemple.
- En 1985, Steven O'Brien et al. ont résolu ce problème de classification en utilisant les séquences d'ADN et algorithmes.



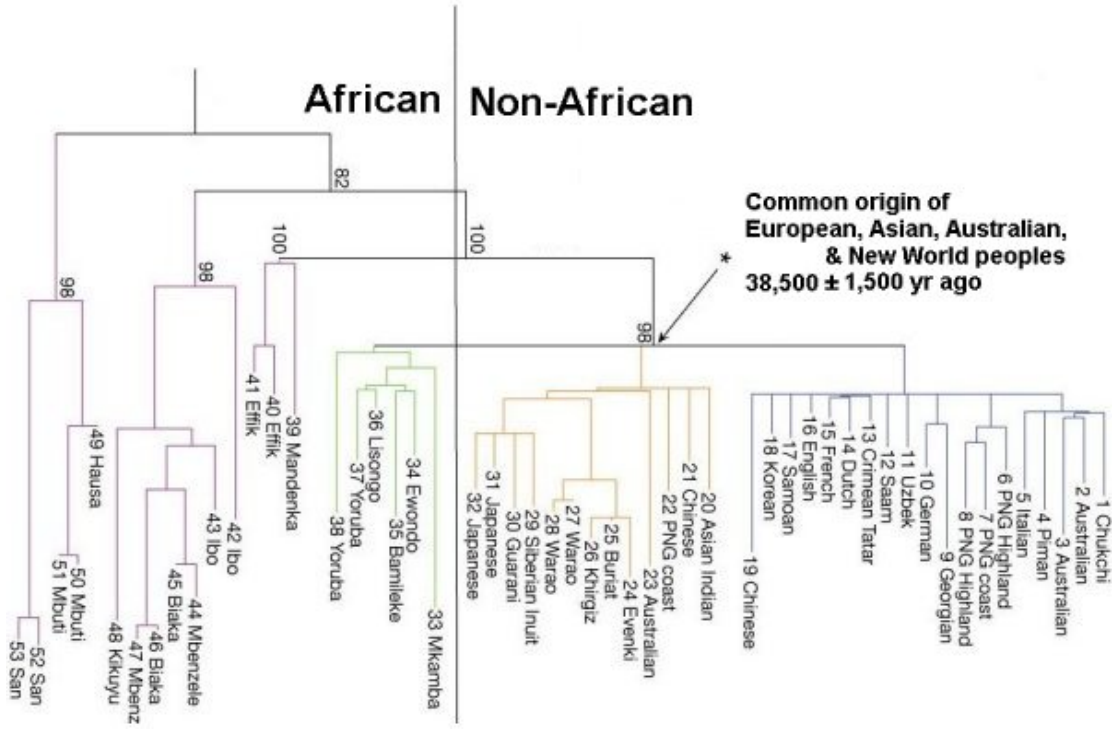
•Il y a 50 ans: Emile Zuckerkandl et Linus Pauling ont proposé la reconstruction phylogénétique basée sur l'ADN.
•Au départ, la possibilité de reconstruire les arbres phylogénétiques par l'analyse d'ADN a été discutée beaucoup. A nos jours, elle constitue l'approche dominant pour comprendre l'évolution.

Out of Africa Hypothesis

L'origine africaine de tous les hommes modernes est indiquée dans les évidences génétiques:

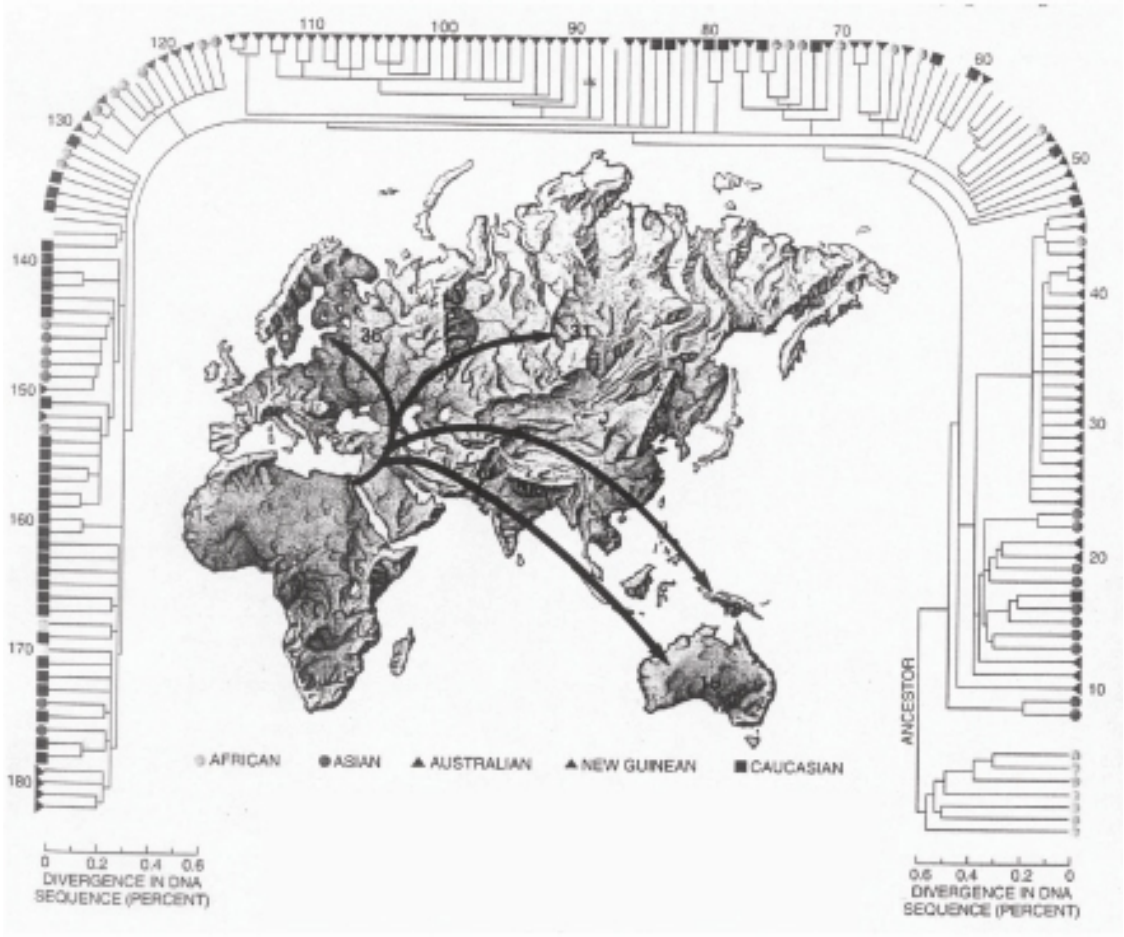
- Dans les mêmes années, quand la classification du panda géant a été résolue, une reconstruction basée sur l'ADN de l'arbre phylogénétique de l'homme a amené à l'hypothèse (**Out of Africa Hypothesis)** que l'ancêtre plus ancien de l'homme a vécu en Afrique à peu près il y a 200,000 ans.

L'arbre phylogénétique a séparé un groupe africain d'un groupe contenant toutes les 5 populations restantes.



http://www.mun.ca/biology/scarr/Out_of_Africa2.htm

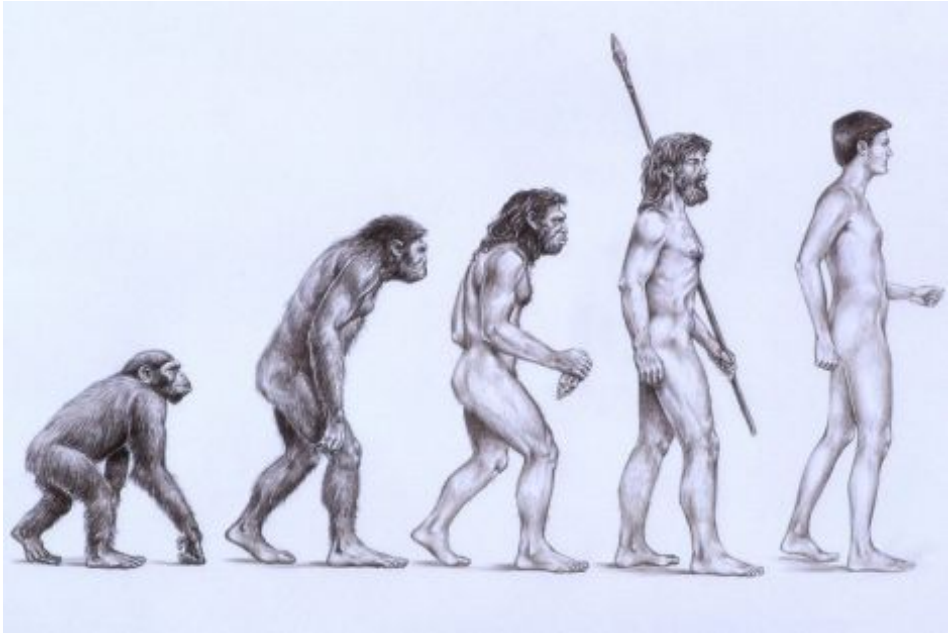
L'approche d'analyse a été basé sur l'ADN mitochondriale de 182 personnes. L'ADN mitochondriale est particulièrement important parce que il est complètement copié de mère à enfant, sans recombinaison avec l'ADN mitochondriale du père.

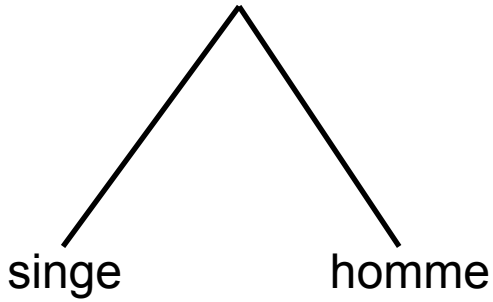


Arbres Phylogénétiques

Comment sont-ils reconstruit ces arbres à partir des séquences d'ADN?

- Les feuilles représentent les espèces existantes
- Les noeuds internes représentent les ancêtres
- La racine représente l'ancêtre le plus ancien du point de vu de l'évolution.



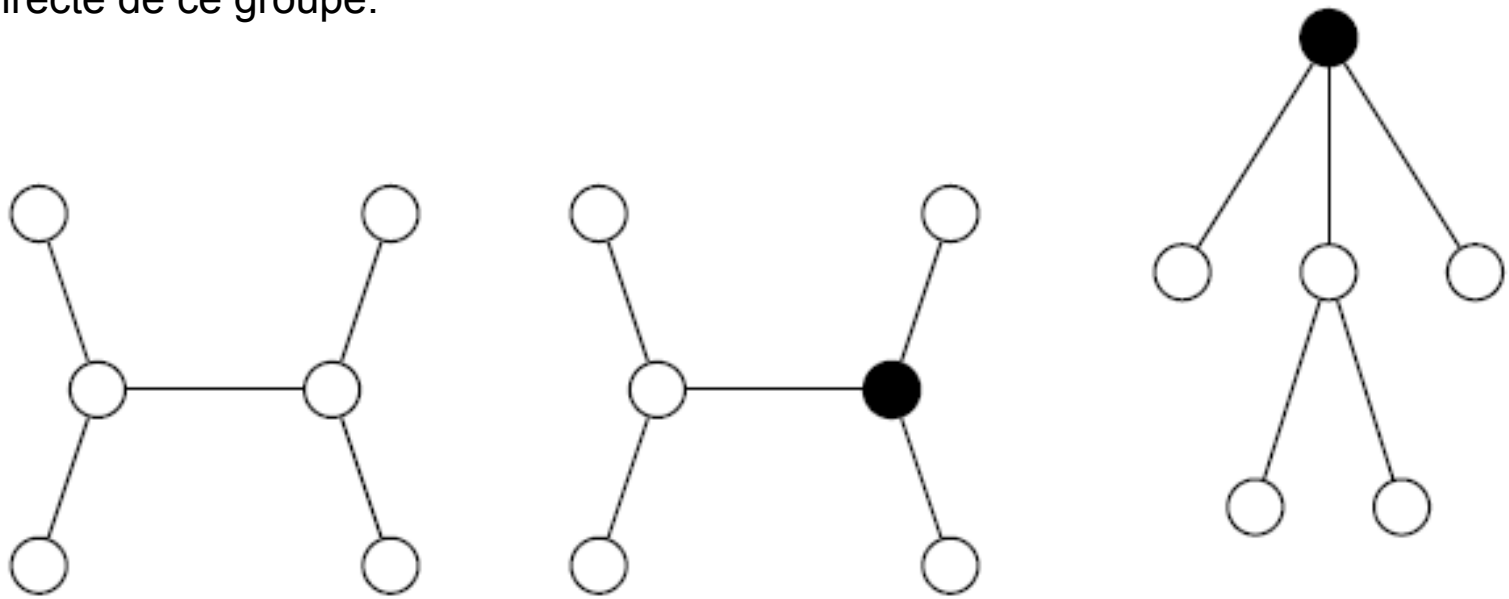


Arbres racinés et non racinés

Un arbre montre la proximité entre espèces. Il n'a pas une racine pre-déterminée et il n'induit pas de hiérarchie. Donc la distance entre nœuds est symétrique.

Dans un arbre non-raciné, la position de la racine ("l'ancêtre le plus vieux") n'est pas connue.

On peut choisir un nœud déjà existant comme racine, ou bien on ajoute un nouveau nœud qui jouera le rôle de la racine. On peut par exemple introduire une espèce qui est distante des autres espèces considérées et la racine proposée sera le prédécesseur directe de ce groupe.



(a) Unrooted tree (b) Rooted tree (c) The same rooted tree

Distances dans l'arbre

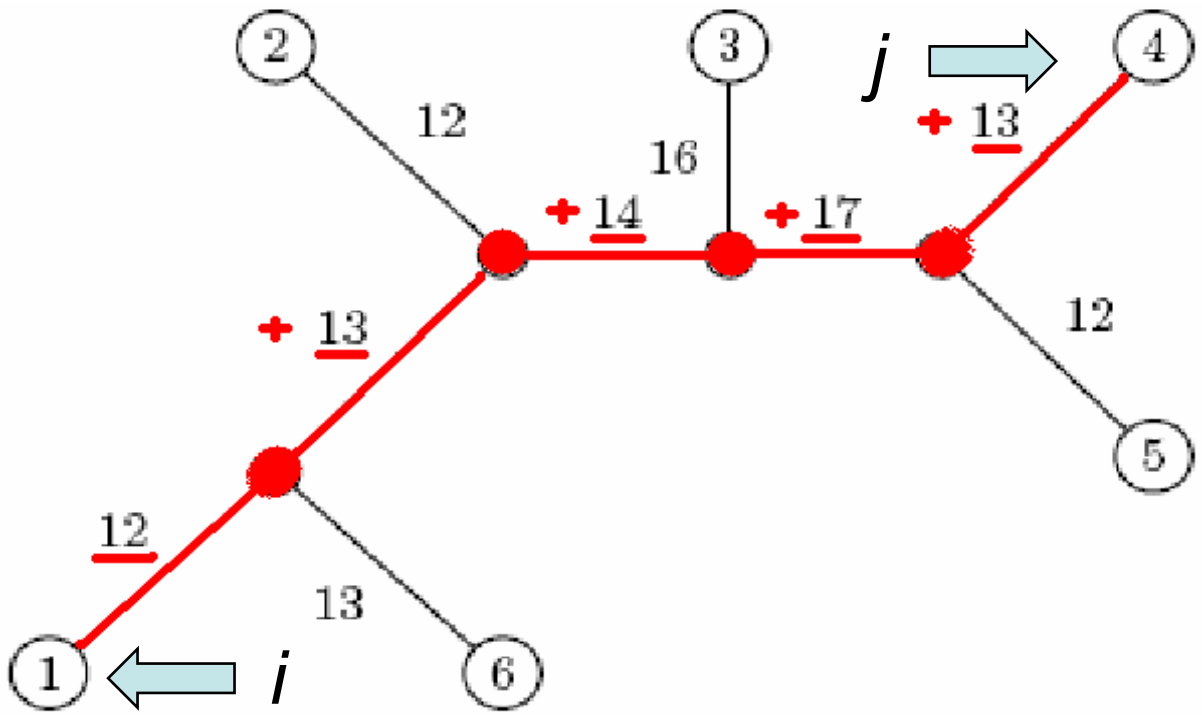
- Les arcs peuvent avoir associés des poids qui reflèctent:
  - Le nombre de mutations dans le chemin d'évolution d'une espèce vers l'autre
  - L'estimation du temps d'évolution d'une espèce dans l'autre

- Pour un arbre *T*, on calcule souvent

$d_{ij}(T)$ – la longueur du chemin entre les feuilles *i* et *j*

$d_{ij}(T)$ – ***distance dans l'arbre entre i et j***

Distance dans l'arbre: exemple

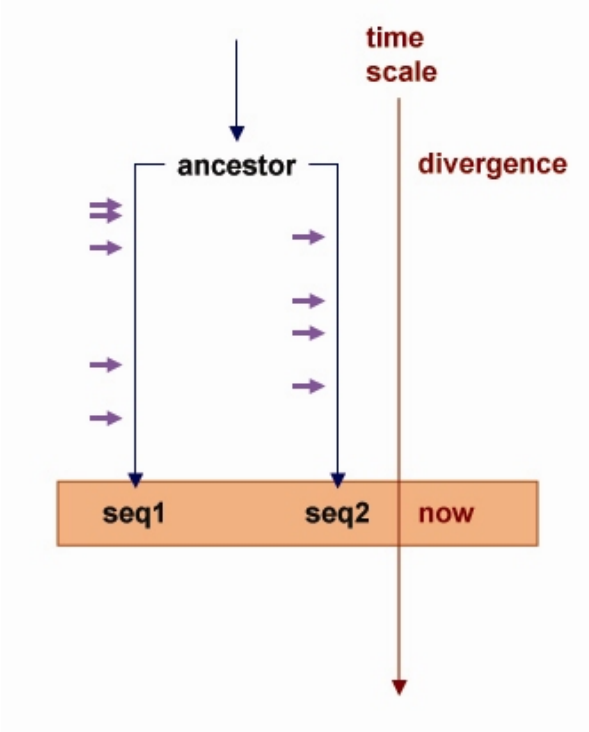


$$d_{1,4} = 12 + 13 + 14 + 17 + 12 = 68$$

Pour reconstruire un arbre, on peut faire l'hypothèse, qu'il existe un **horloge moléculaire** qui gouverne l'évolution des espèces et qui représente un rythme constant du processus évolutionnaire.

Si cela était le cas, on aurait pu produire théoriquement un arbre phylogénétique qui préserve la distance et qui peut être représenté à l'aide d'un axe de temps, qui assigne a chaque nœud le moment quand il est paru dans l'histoire de l'évolution.

Dans cet arbre parfait, la longueur de chaque arc serait calculée comme la différence de temps entre le nœud père et le nœud fils.



Il y a deux types de données utilisées pour reconstruire les arbres phylogénétiques:

Reconstruction basée sur les caractères: examiner chaque caractère séparément

Reconstruction basée sur la distance: l'entrée est une matrice de distances entre espèces

Matrice des distances

D_{ij} –distance observée entre i et j

Etant données *n* espèces, nous pouvons calculer une matrice *n* x *n* des distances D_{ij}. D_{ij} peut être définie, par exemple, comme la distance d'édition entre un gène dans l'espèce i et l'espèce j, où le gène d'intérêt appartient aux n espèces.

Noter la différence avec

$d_{ij}(T)$ – distance dans l'arbre entre i et j

Matrices de distance adéquate

- Etant données *n* espèces, on calcule la matrice *n* x *n de* ***distance*** D_{ij}

- Pour visualiser les relations évolutives entre gènes, on utilisera un arbre que, a priori, ***on ne connaît pas****,* mais qui l'on veut reconstruire.

- On a besoin d'un algorithme pour construire un arbre qui correspond au mieux à la matrice de distance D_{ij}

Adéquation de la matrice de distance

Idéalement:

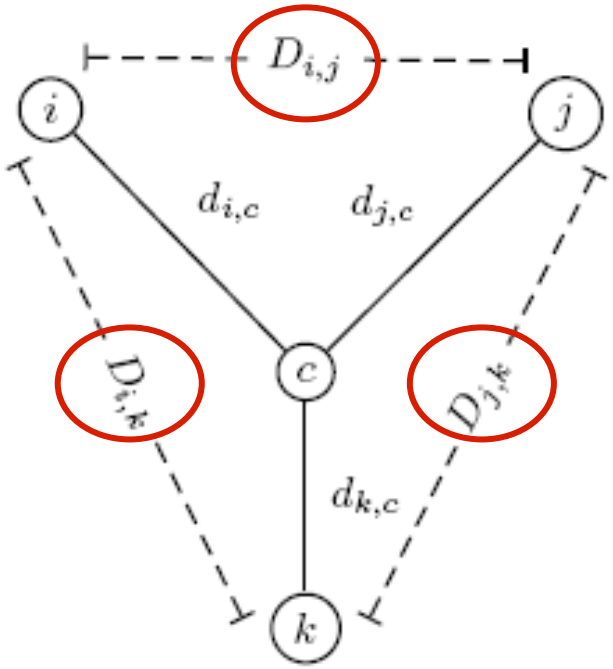
Longueur du chemin dans un arbre (***inconnu***) T

adequate signifie $D_{ij} = d_{ij}(T)$

Distance d'édition entre espèces (***connue, il s'agit de la distance observée***)

Reconstruction de l'arbre à 3 feuilles

- La reconstruction de l'arbre pour toute matrice 3x3 est évidente.
- Nous avons 3 feuilles *i, j, k* et un noeud centrale *c :*



Observe:

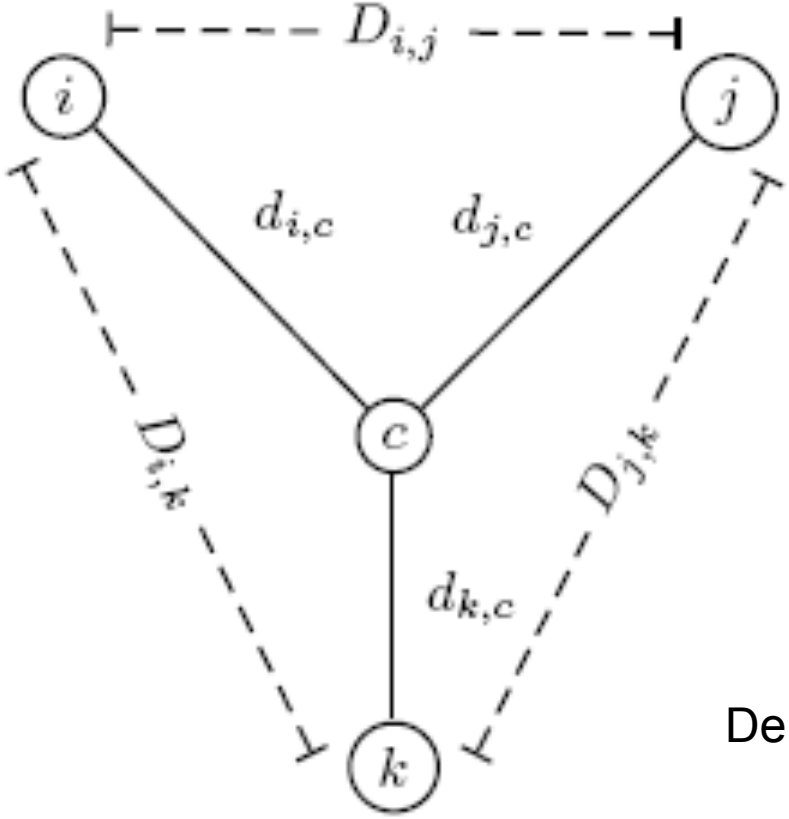
Valeurs connues

$$d_{ic} + d_{jc} = D_{ij}$$

$$d_{ic} + d_{kc} = D_{ik}$$

$$d_{jc} + d_{kc} = D_{jk}$$

Reconstruction de l'arbre à 3 feuilles



$$d_{ic} + d_{jc} = D_{ij}$$

$$+\ d_{ic} + d_{kc} = D_{ik}$$

$$2d_{ic} + d_{jc} + d_{kc} = D_{ij} + D_{ik}$$

$$2d_{ic} + D_{jk} = D_{ij} + D_{ik}$$

$$\boldsymbol{d_{ic} = (D_{ij} + D_{ik} - D_{jk})/2}$$

De façon similaire,

$$\boldsymbol{d_{jc} = (D_{ij} + D_{jk} - D_{ik})/2}$$

$$\boldsymbol{d_{kc} = (D_{ki} + D_{kj} - D_{ij})/2}$$

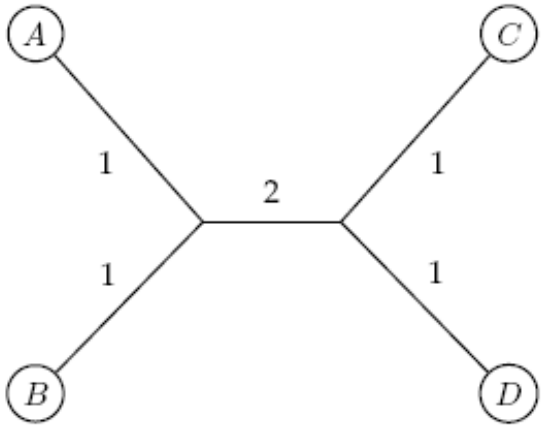
Arbre ayant > 3 feuilles

- Un arbre avec *n* feuilles a >*2n-3* arcs
- Cela signifie que rendre adéquat un arbre à une matrice de distance *D* demande la résolution d'un système de C^n_2 équations avec *2n-3* variables
- Cela n'est pas toujours possible pour *n* > 3

Matrices de distance

La matrice *D* est ADDITIVE s'il existe un arbre *T* avec $d_{ij}(T) = D_{ij}$

	A	B	C	D
A	0	2	4	4
B	2	0	4	4
C	4	4	0	2
D	4	4	2	0



NON-ADDITIVE sinon

	A	B	C	D
A	0	2	2	2
B	2	0	3	2
C	2	3	0	2
D	2	2	2	0

?

Problème de la reconstruction phylogénétique basée sur les distances

Problème: Reconstruire un arbre phylogénétique à partir de la matrice des distances

Entrée: matrice n x n des distances D_{ij}

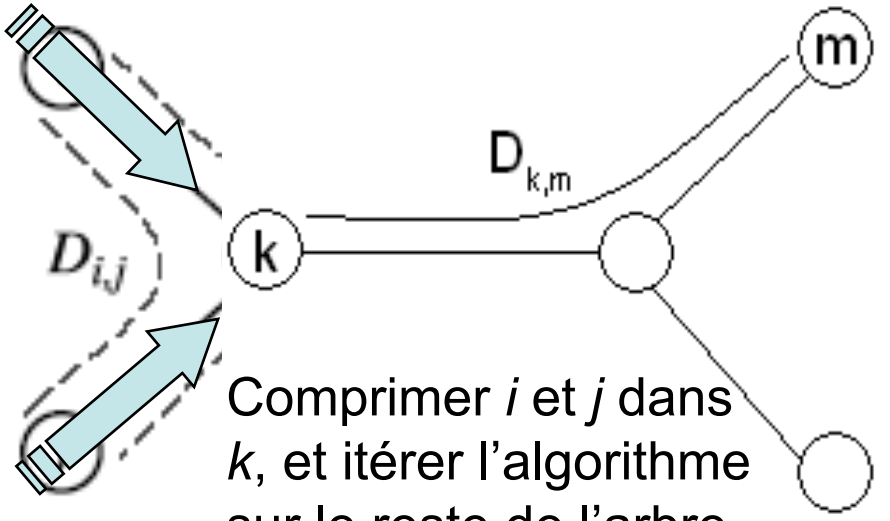
Sortie: un arbre pondéré T avec n feuilles représentant D

Si D est additive, alors le problème a une solution et il existe un algorithme simple pour le résoudre.

Usage des feuilles “voisines” (cad ayant un nombre minimale d’arcs entre elles) pour la construction de l’arbre

- Trouver des feuilles « voisines » *i* et *j* avec père *k*
- Effacer lignes et colonnes de *i* et j de la matrice
- Ajouter une nouvelle ligne et colonne correspondant à k, où la distance entre k et toute autre feuille m peut être calculée comme suit :

$$D_{km} = (D_{im} + D_{jm} - D_{ij})/2$$



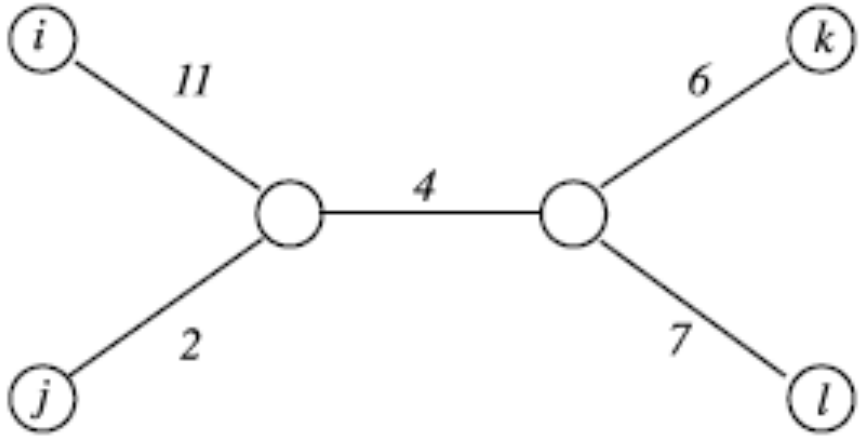
Trouver les feuilles voisines

Pour trouver les feuilles voisines on sélectionne simplement une paire de feuilles proches (cad les feuilles *i* et *j* ayant D_{ij} minimale) .

erroné

Trouver les feuilles voisines

- Les feuilles proches (cad les feuilles *i* et *j* ayant D_{ij} minimale) ne sont pas forcement des feuilles voisines :
- *i* et *j* sont voisines, mais (d_{ij} = 13) > (d_{jk} = 12)



Trouver une paire de feuilles voisines en utilisant seulement la matrice de distance est un problème non triviale!

Reconstruction des arbres à partir d'une matrice additive : une première approche

Triplets dégénérés

- Un triplet dégénérés est un ensemble de trois éléments distincts $1 \le i,j,k \le n$ où $D_{ij} + D_{jk} = D_{ik}$ (d'habitude on a $D_{ij} + D_{jk} \ge D_{ik}$)

- L' élément *j* dans un triplet dégénéré *i,j,k* fait partie d'un chemin évolutionnaire de *i* à *k* (ou bien il est attaché à ce chemin par un **arc de longueur 0**).

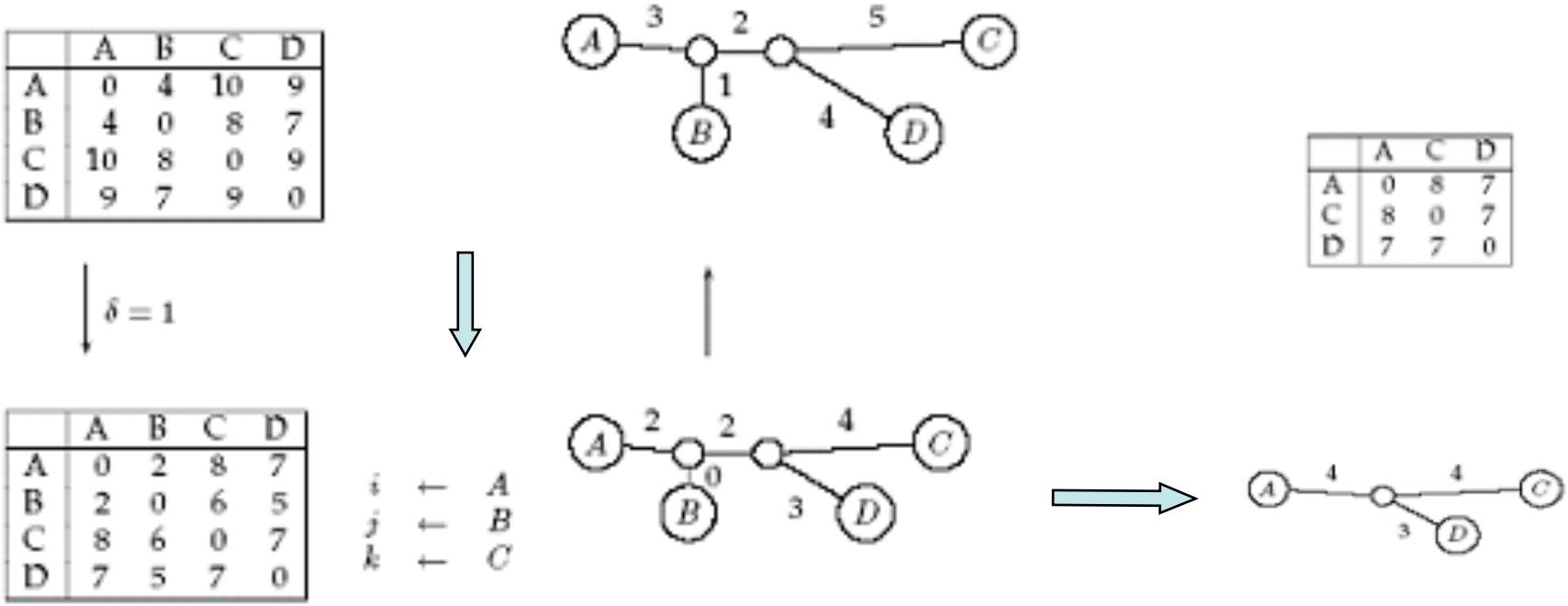
Recherche de triplets dégénérés

- Si la matrice des distances *D* **a un triplet dégénéré** *i,j,k* alors *j* peut être "éliminé" de *D* et cela réduit la taille du problème.

- Si la matrice de distance *D* **n'a pas un triplet dégénéré** *i,j,k,* on peut "créer" un triplet dégénéré dans *D* en réduisant la taille de tous les arcs pendants/suspendus (dans l'arbre).

Réduction des arcs “pendants” pour produire un triplet dégénéré

Raccourcir tous les arcs “pendants” (arcs qui connectent aux feuilles) jusqu’au moment ou un triplet dégénéré est trouvé



Trouver triplets dégénérés

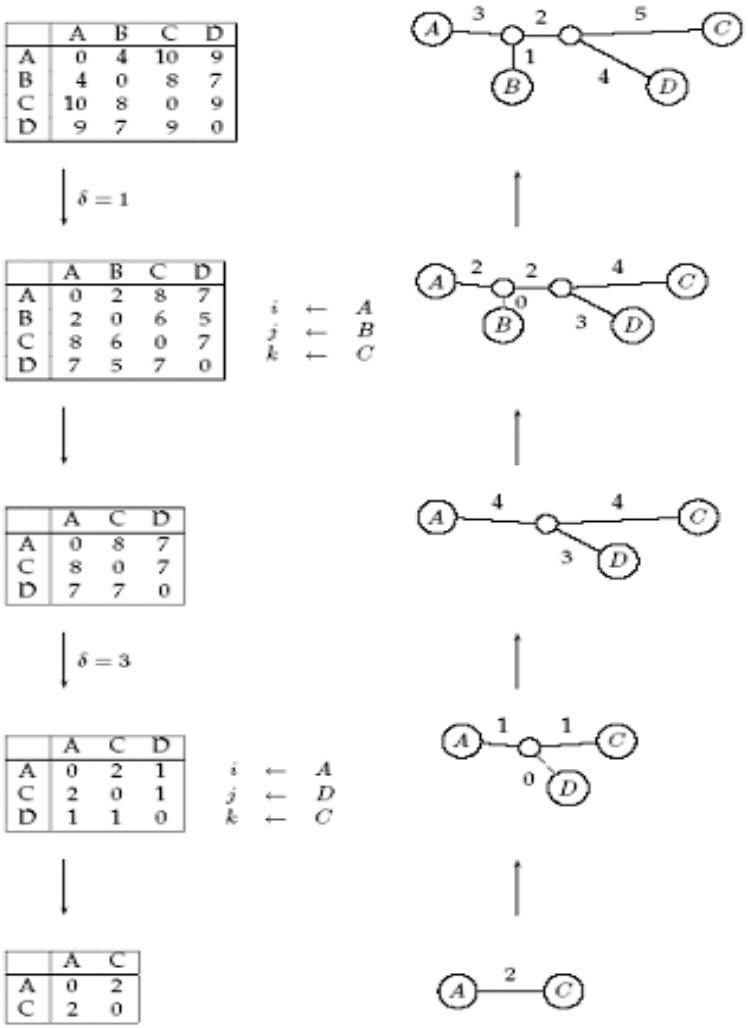
- S'il n'y a pas de triplet dégénéré, tous arcs pendants sont réduits par la même quantité δ, de telle façon que toutes les distances entre paires dans la matrice sont réduites par 2δ.

- A la fin, ce processus collapse une des feuilles (quand δ = longueur de l'arc « pendant » le plus court), et forme un triplet dégénéré *i,j,k* et réduit la taille de la matrice de distance *D.*

- Le point d'attachement pour *j* peut être récupéré dans les transformations renversées en sauvant D_{ij} pour chaque feuille collapsée.

Reconstruction des arbres pour des matrices de distance additives

$D_{ij} + D_{jk} = D_{ik}$



Algorithme AdditivePhylogeny

1. **AdditivePhylogeny**(D)

2. **si** D est une matrice 2 x 2
3. T = arbre constitué par un seul arc de longueur $D_{1,2}$
4. **returner** T
5. **si** D est non-dégénéré
6. δ = paramètre de coupure de la matrice D
7. **pour** tout $1 \leq i \neq j \leq n$
8. $D_{ij} = D_{ij} - 2\delta$
9. **sinon**
10. $\delta = 0$

AdditivePhylogeny

1. trouver un triplet i, j, k dans D tel que $D_{ij} + D_{jk} = D_{ik}$
2. $x = D_{ij}$
3. effacer la ligne j^{th} et la colonne j^{th} de D
4. T = AdditivePhylogeny(D)
5. Ajouter un nouveau noeud v à T à la distance x de i à k
6. ajouter j a T en creant un arc (v,j) de longueur 0
7. **pour** chaque feuille l dans T
8. **si** la distance de l à v dans l'arbre $\neq D_{l,j}$
9. output "la matrice n'est pas additive"
10. **returner**
11. Etendre tous les arcs "pendant" d'une longueur δ
12. **returner** T

La condition des quatre points

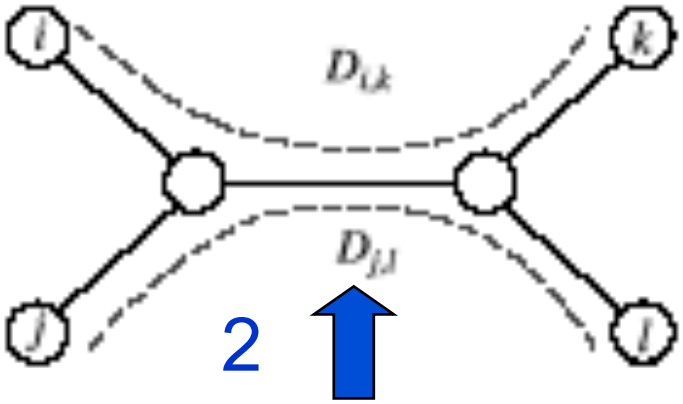
- AdditivePhylogeny fourni une façon pour tester si la matrice de distance *D* est additive

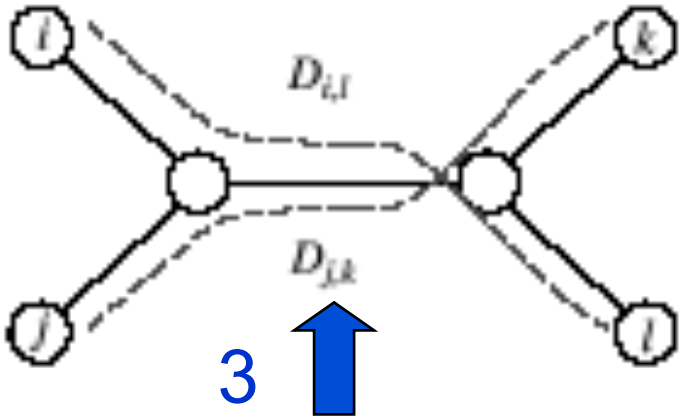
- **Un test d'additivité encore plus efficace est la "condition des 4 points"**

- Soit $1 \leq i,j,k,l \leq n$ quatre feuilles distinctes dans un arbre.

La condition des quatre points

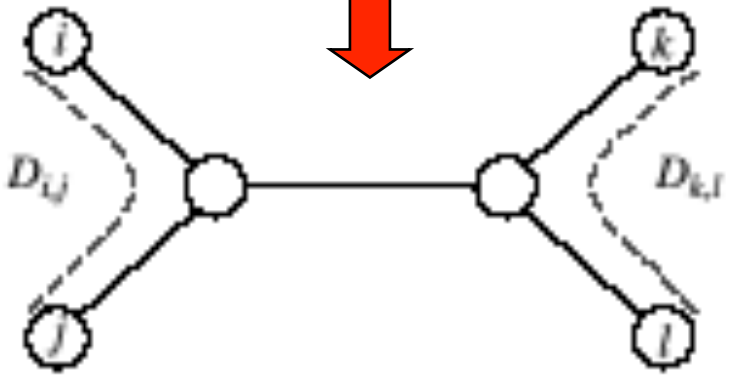
Calcule : 1. $D_{ij} + D_{kl}$, 2. $D_{ik} + D_{jl}$, 3. $D_{il} + D_{jk}$





2 et **3** représentent le **même nombre**: la longueur de tous les arcs + l'arc du milieu (il est compté deux fois)

1



1 représente un **nombre plus petit**: **la longueur de tous les arcs – 2 x l'arc au milieu**

La condition des quatre points : théorème

- La condition des 4 points pour le quartet *i,j,k,l* est satisfaite si deux des sommes sont les mêmes, avec la troisième somme plus petite que les premières deux.

- **Théorème** : Une matrice *n* x *n D* est additive ssi la condition des 4 points est vraie pour **tous** les quartets *i,j,k,l* où $1 \leq i,j,k,l \leq n$

Least Squares Distance Phylogeny Problem

- Si la matrice de distance *D* n'est pas additive, alors on cherche un arbre *T* que approxime *D* pour le mieux:

  Squared Error : $\sum_{i,j} (d_{ij}(T) - D_{ij})^2$

- Squared Error est une mesure de la qualité de l'adéquation entre matrice de distance et arbre: on veut la minimiser.

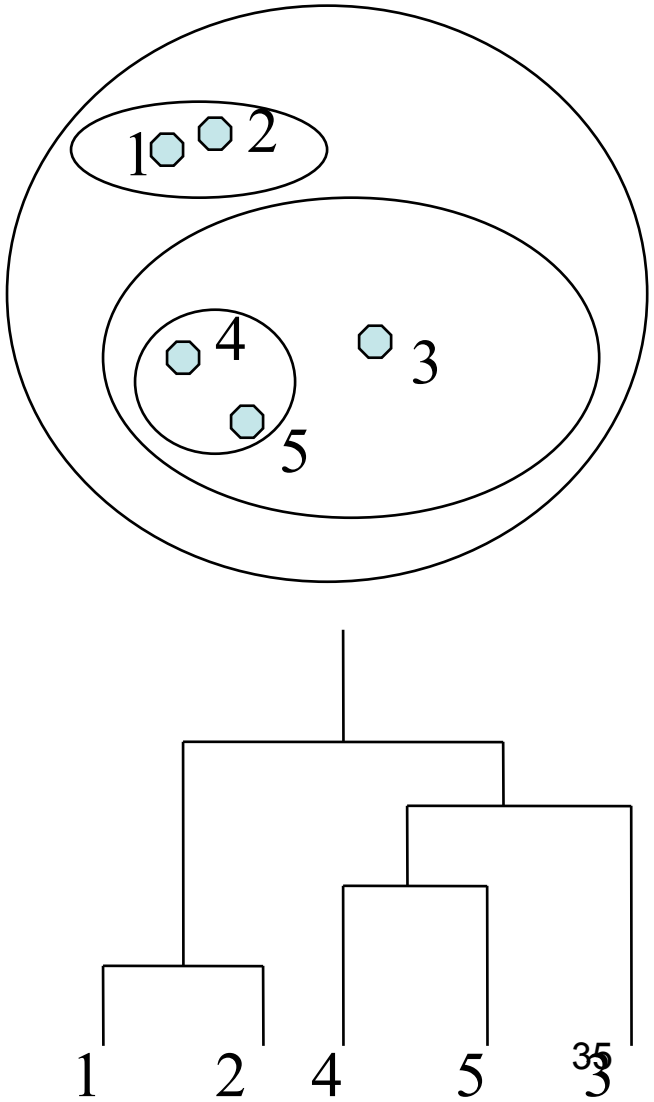
- **Least Squares Distance Phylogeny Problem**: trouver le meilleur arbre approximé *T* pour une matrice non-additive *D (*NP-hard).

On présentera dans la suite deux algorithmes euristiques polynomiaux, UPGMA et Neighbor-Joining.

UPGMA: Unweighted Pair Group Method with Arithmetic Mean

UPGMA est un algorithme de clustering:

- Il calcule la distance entre clusters en utilisant la distance moyenne entre paires
- Il assigne une *hauteur* à chaque noeud dans l'arbre, en supposant l'existence d'un horloge moléculaire



Clusterisation dans UPGMA

Etant données deux clusters disjoints C_i, C_j d'espèces,

$$d_{ij} = \frac{1}{|C_i| \times |C_j|} \Sigma_{\{p \in Ci,\ q \in Cj\}} d_{pq}$$

distance moyenne de Ci, Cj calculée sur toutes les paires

Noter que si $C_k = C_i \cup C_j$, alors la distance entre C_k et un autre cluster C_l est:

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

Moyenne pondérée des distances entre les composantes

Algorithme UPGMA

Initialisation:

Assigner chaque x_i à son propre cluster C_i

Définir une feuille par espèce, chacune a hauteur 0

Iteration:

Trouver deux clusters C_i et C_j tels que d_{ij} est minimale

Soit $C_k = C_i \cup C_j$

Ajouter un noeud qui connecte C_i, C_j *et le* placer à une hauteur $d_{ij}/2$

Effacer C_i *et* C_j

Calculer la distance entre le nouveau cluster C_k et tous les autres comme une moyenne pondérée des distances entre les composantes

Terminaison:

Quand on obtient un seul cluster

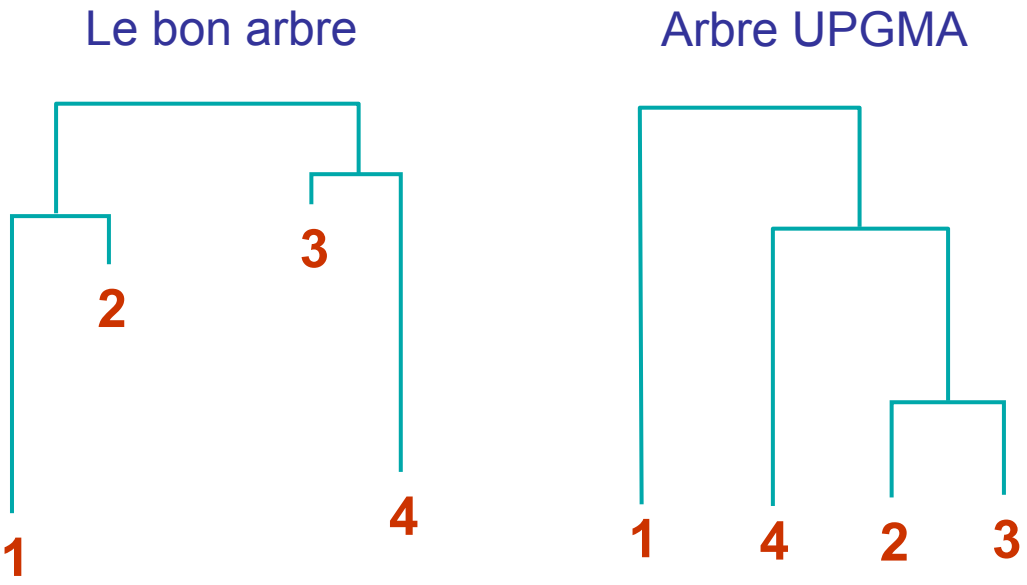
La complexité de UPGMA est $O(n^2)$: il y a n-1 itérations, avec O(n) opérations effectuées dans chacune.

Faiblesses de UPGMA

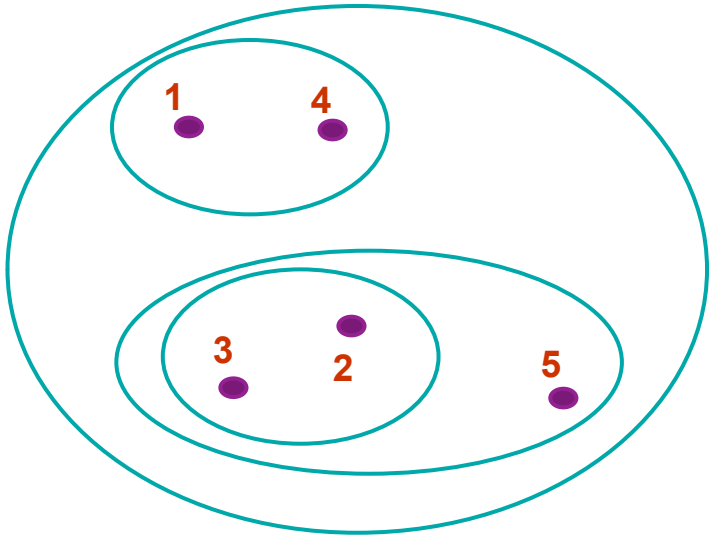
- L'algorithme produit un arbre **ultramétrique** : la distance de la racine à chaque feuille est la même. Cela signifie qu'il existe un **horloge moléculaire** ayant rythme constant: toutes les espèces représentées par les feuilles dans l'arbre sont pensées comme ayant accumulé des mutations (et donc comme ayant évoluées) au même taux de mutation.

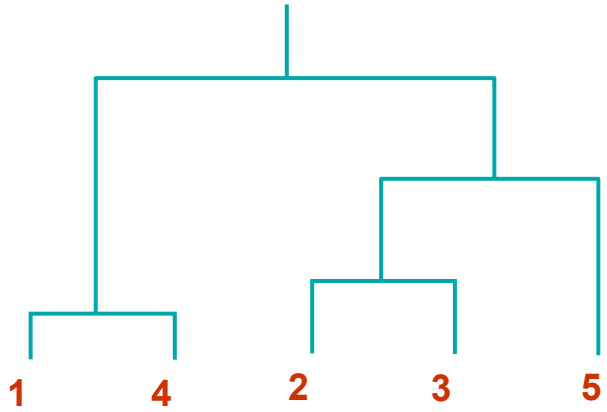
  Ceci est le problème majeur de UPGMA.

  Par exemple:



- **Il présuppose que les clusters fusionnés soient proches l'un des autres, sans pourtant demander qu'ils soient aussi loin du reste.**





Algorithme Neighbor Joining

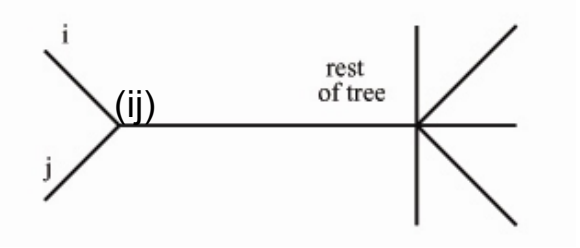
- En 1987, Naruya Saitou and Masatoshi Nei ont développé un algorithme de « neighbor joining » pour la reconstruction des arbres phylogénétiques

- **Trouver une paire de feuilles qui sont proches entre elles mais loin des autres feuilles:** implicitement il faut trouver une paire de feuilles dans un « voisinage ». A chaque itération, l'algorithme essaye de trouver l'ancêtre directe de deux espèces dans l'arbre. Pour chaque nœud i,sa distance u_i du reste de l'arbre est estimée en utilisant la formule

$$u_i = \sum_{k \neq i} \frac{D_{i,k}}{n-2}$$

Pour minimiser la somme de toutes les longueurs de branchement, les nœuds i et j qui sont clustérisés dans la suite sont ceux avec

$$D_{i,j} - u_i - u_j \quad \text{minimisée}$$



- Les longueurs $d_{k(ij)}$ des nouvelles branches sont calculée en résolvant un système d'équations linéaires qui donne les distances entre paires (voir algorithme étape 6).

Avantages: il marche bien pour des matrices additives et certaines matrices non-additive; il ne considère pas l'hypothèse de horloge moléculaire.

Algorithme NJ

Initialisation:

1. Initialiser n clusters avec les différentes espèces, une espèce par cluster
2. Fixer la taille des clusters à 1, pour chaque cluster
3. dans l'arbre de sortie T, affecter une feuille à chaque espèce, ayant une hauteur 0

Itération:

4. pour chaque espèce, calculer $u_i = \sum_{k \neq i} \frac{D_{i,k}}{n-2}$

5. choisir le i et j tels que $D_{i,j} - u_i - u_j$ est minimale

6. rejoindre les clusters i et j dans un nouveau cluster (ij), avec un noeud correspondant dans T. Calculer les longueurs des branches de i et j au nouveau noeud comme suit:

$$d_{i(ij)} = \tfrac{1}{2} D_{ij} + \tfrac{1}{2} (u_i - u_j) \qquad d_{j(ij)} = \tfrac{1}{2} D_{ij} + \tfrac{1}{2} (u_j - u_i)$$

7. Calculer les distances entre le nouveau cluster et tous les autres clusters

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

8. Effacer clusters i et j des tables, et remplacer les par (ij)
9. Si plus que 2 nœuds (clusters) restent, retourner à 4. Si non,connecter les deux noeuds restants par une branche de longueur D_{ij}.

La complexité de l'algorithme NJ est $O(n^3)$

Reconstruction de l'arbre basée sur les caractères

Problème (arbre phylogénétique optimale)

Entrée:
- Un ensemble de n espèces
- Un ensemble de m caractères concernant toutes les espèces
- Pour chaque espèces, la valeur de chaque caractère
- Paramètres optimaux qui sont dépendant du problème

Sortie: un arbre phylogénétique complètement étiquetté qui explique pour le mieux les données, cad qu'il maximise une fonction cible donnée.

Note : Les caractères peuvent être des nucléotides, où A, G, C, T sont les états de ce caractère. Autres caractères peuvent être le # des yeux ou le # des pattes ou les formes de ...

Hypothèses:

1 Les caractères sont mutuellement indépendant, càd que un changement dans un caractère n'a pas d'effet sur la distribution d'un autre caractère.

2 Après que deux espèces ont divergé dans l'arbre, elles continuent à évoluer indépendamment.

Probablement ces hypothèses ne sont pas correctes, mais elles simplifient l'analyse.

Une **solution triviale** du problème de la phylogénie est celui d'énumérer tous les possibles arbres et calculer la valeur d'une « fonction cible » (target function) pour chacun d'entre eux.

Sous l'hypothèse que les arbres phylogénétiques soient binaires, le nombre d'arbres non-isomorphes, étiquetés, binaires, racinés et contenant n feuilles est:

$$1 * 3 * 5 * \ldots * (2n-3) = \prod_{i=2}^{n} (2i-3) = (2n-3)!! = \Omega((2n/3)^{n-1})$$

càd super-exponentiel. Pour n=20, on a à peu près 10^{21} tels arbres.

La même chose est vraie pour le nombre d'arbres non-racinés

$$= (2n-5)!! = \Omega((2n/3)^{n-2})$$

(Cavalli-Sforza et Edwards, 1967)

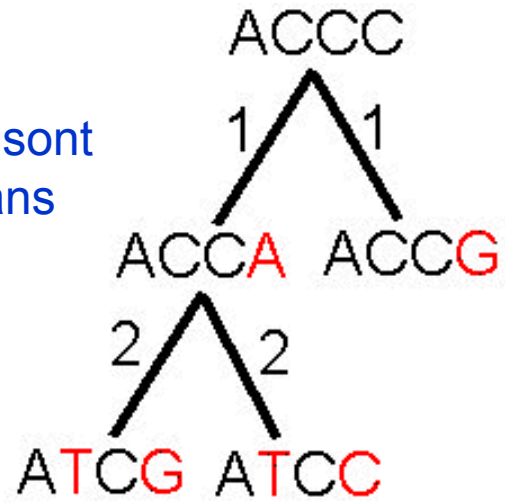
Approche de parcimonie à la reconstruction d'un arbre phylogénétique

- Appliquer le principe du rasoir de Occam – càd identifier la plus simple explication pour les données
- Faire l'hypothèse que les différences entre caractères observés sont le résultat d'un nombre minimale de mutations
- Chercher l'arbre que corresponde au **score de parcimonie** le plus petit – somme des coûts de toutes les mutations trouvées dans l'arbre.

Parcimonie et reconstruction de l'arbre

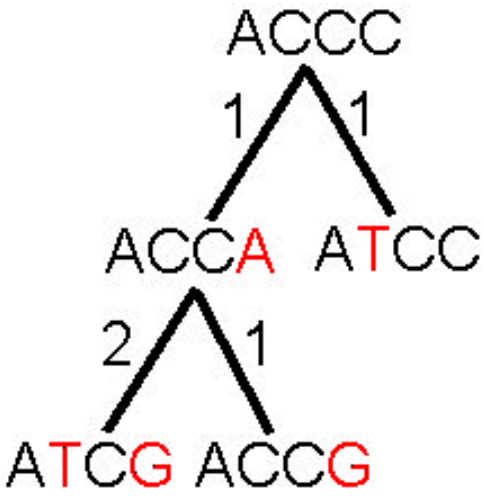
En fixant la longueur d'un arc dans l'arbre à l'aide de la distance de Hamming, on peut définir le score de parcimonie d'un arbre de séquences comme la somme des longueurs (poids) des arcs.

Les caractères sont les positions dans une séquence



Moins parcimonieux

Score: 6



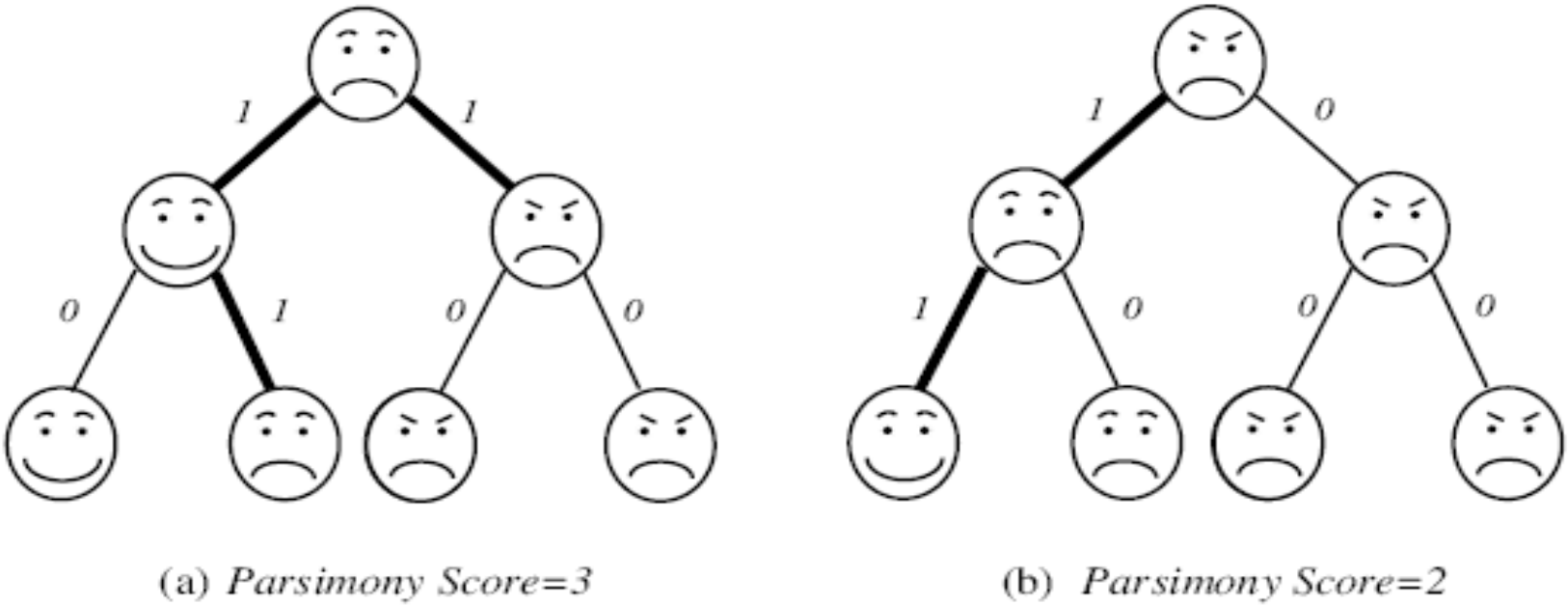
Plus parcimonieux

Score: 5

L'approche de minimiser le score est appelé parcimonie

Un deuxième exemple de reconstruction de l'arbre : deux caractères

(sourcils bouche) : on regarde des mots de deux lettres



(a) *Parsimony Score=3* (b) *Parsimony Score=2*

Figure 10.16 If we label a tree's leaves with characters (in this case, eyebrows and mouth, each with two states), and choose labels for each internal vertex, we implicitly create a *parsimony* score for the tree. By changing the labels in (a) we are able to create a tree with a better parsimony score in (b).

Problème 1 : Le «petit problème de parcimonie»

Entrée : arbre *T* avec toutes feuilles étiquetées par un mot de m caractères

Sortie : étiquetage des nœuds internes de l'arbre *T* minimisant le score de parcimonie

On peut faire l'hypothèse que chaque feuille est étiquetée par un seul caractère, parce que les caractères dans les mots sont indépendants.

Problème 2 : problème de parcimonie pondérée

Il s'agit d'une version plus générale du «petit problème de parcimonie».

- L'entrée considère une matrice de score *k* * *k* qui décrit le coût de transformation de chacun des *k* états dans un autre.

- On rappelle que pour le petit problème de parcimonie, la matrice de score est définie par la distance de Hamming

$d_H(v, w) = 0$ si $v=w$

$d_H(v, w) = 1$ sinon

valeurs d'un caractère

Matrices de scores

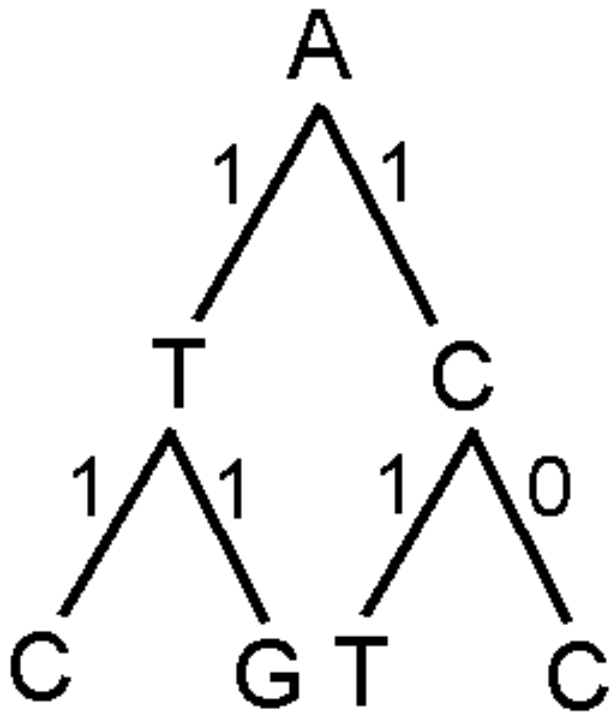
Small Parsimony Problem

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

Weighted Parsimony Problem

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

Non pondéré vs. pondéré

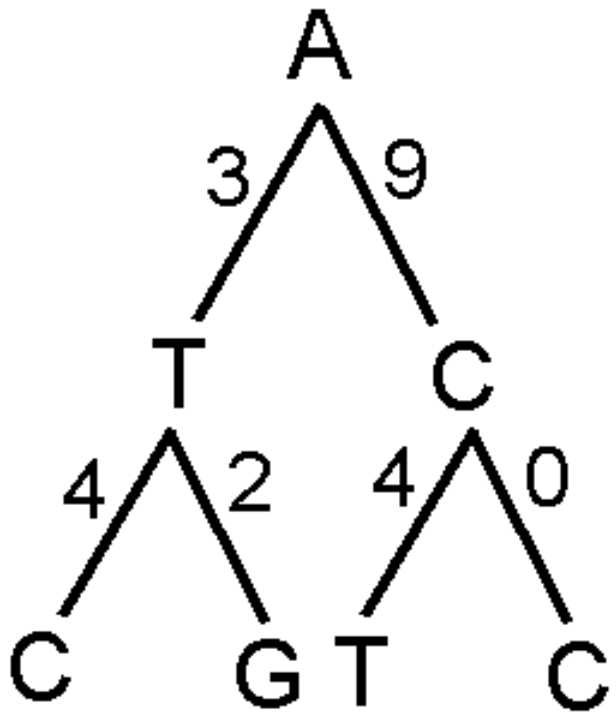


Matrice de score de petite parcimonie:

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

Score de petite parcimonie : 5

Non pondéré vs. pondéré



Matrice de scores de parcimonie pondérée :

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

Score de parcimonie pondérée : 22

Problème de parcimonie **pondérée** : formulation

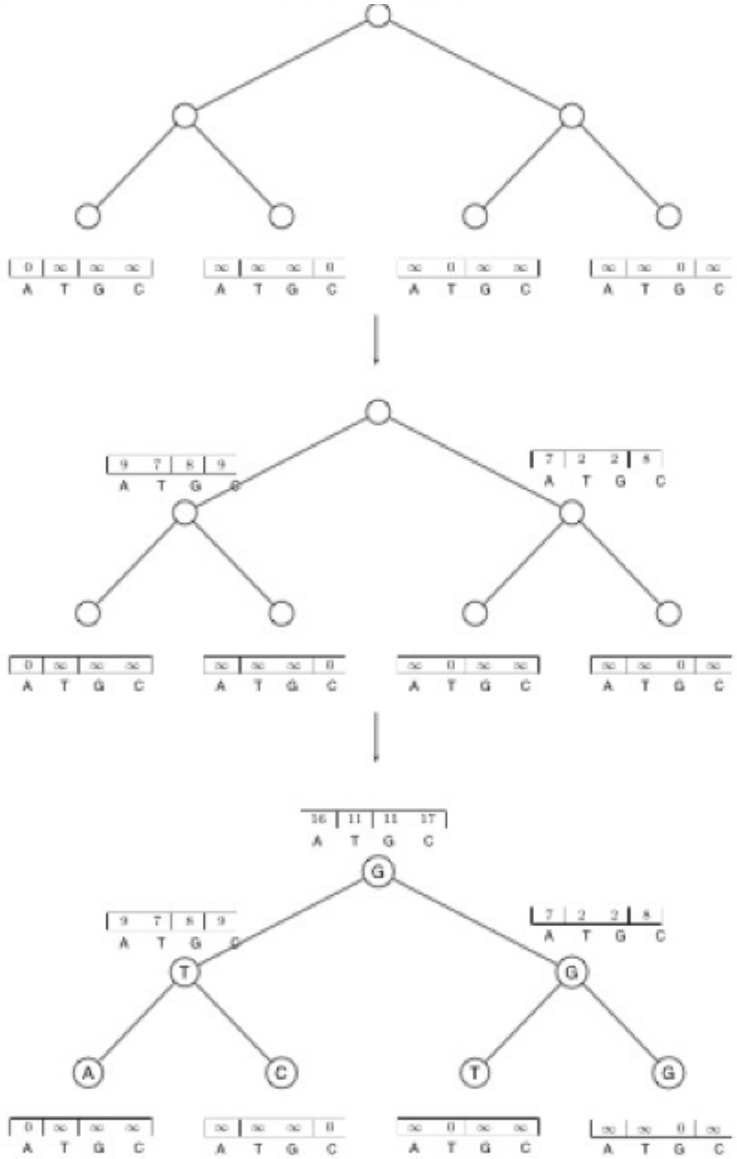
Entrée: Un arbre *T* (défini par sa topologie et sa racine) avec toutes les feuilles étiquetées par des éléments d'un alphabet à k lettres et une matrice de score *k* x *k* (δ_{ij})

Sortie: étiquetage des noeuds internes d'un arbre T qui minimise le score de parcimonie pondéré.

Algorithme de Sankoff

- Tester tous les fils de chaque noeud v et déterminer le coût minimum du sous-arbre ayant v comme racine
- Un exemple:

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



Algorithme de Sankoff: programmation dynamique

Etape 1:

- Pour chaque noeud de l'arbre, calculer et garder le score associé à chaque possible étiquette
  - $s_t(v)$ = score de parcimonie minimale du sous-arbre racine au noeud *v*, si *v* a un caractère de valeur *t*

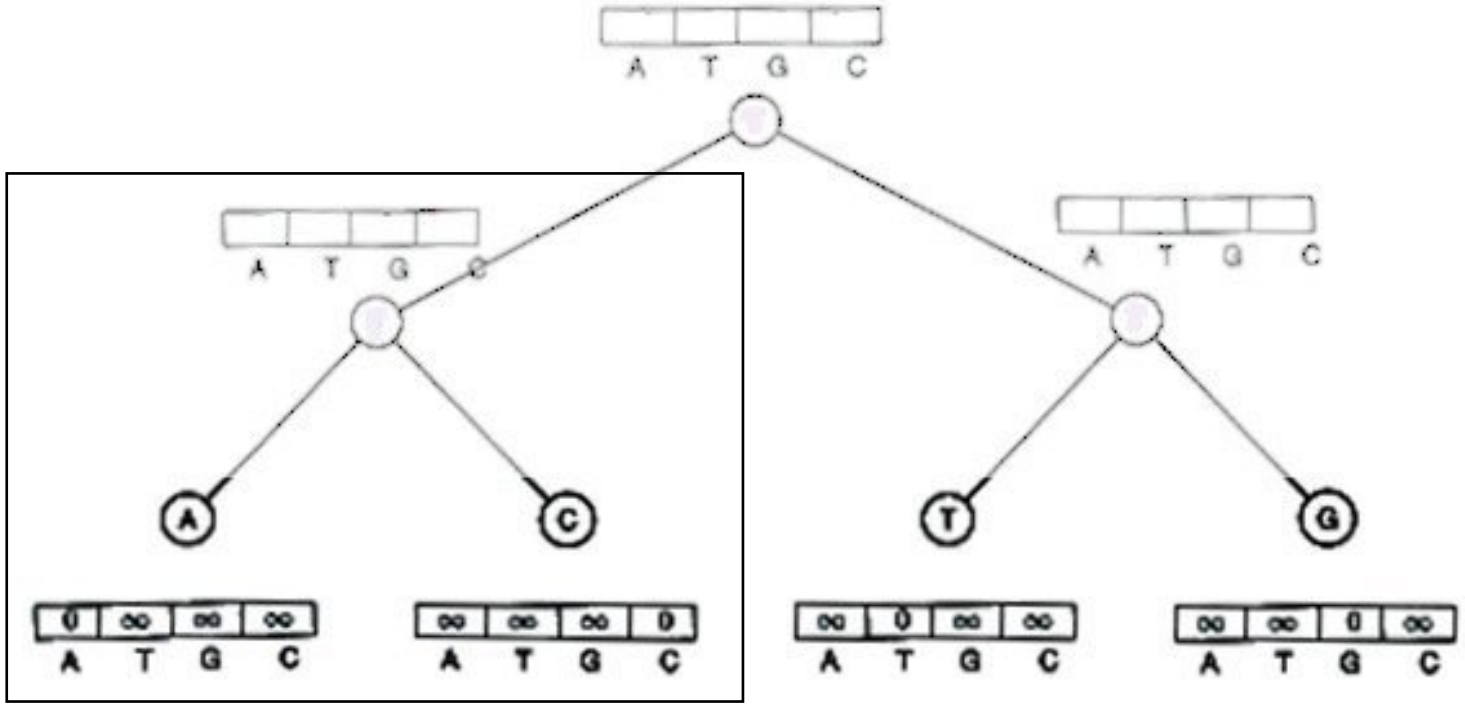
- Le score à chaque noeud est basé sur le score de ses fils:
  - $s_t(\textbf{père}) = \min_i \{s_i(\textbf{fils gauche}) + \delta_{i,t}\} + \min_j \{s_j(\textbf{fils droit}) + \delta_{j,t}\}$

où *t,i,j* sont des indices qui varient sur les différentes valeurs d'un caractère et $\delta_{i,t}$ représente le coût de la transformation d'une valeur i associée au fils dans une valeur t associée au père

Algorithme de Sankoff (2)

On commence avec les feuilles:

- Si une feuille a un caractère alors le score est 0
- Sinon le score est ∞



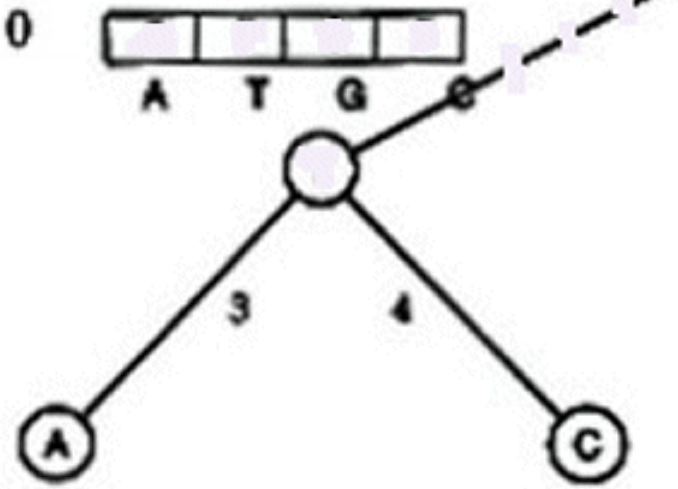
A

Algorithme de Sankoff (3)

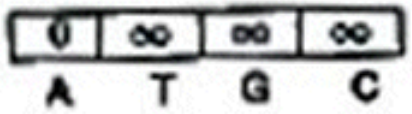
δ	A	T	G	C
A	0	3	4	9
T	3	0	2	4
G	4	2	0	4
C	9	4	4	0

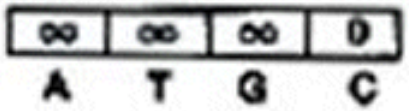
$$s_t(v) = \min_i \{s_i(u) + \delta_{i,\,t}\} + \min_j \{s_j(w) + \delta_{j,\,t}\}$$

$$s_A(v) = 0 + \min_j \{s_j(w) + \delta_{j,\,A}\}$$



	$s_i(u)$	$\delta_{i,\,A}$	sum
A	0	0	**0**
T	∞	3	∞
G	∞	4	∞
C	∞	9	∞



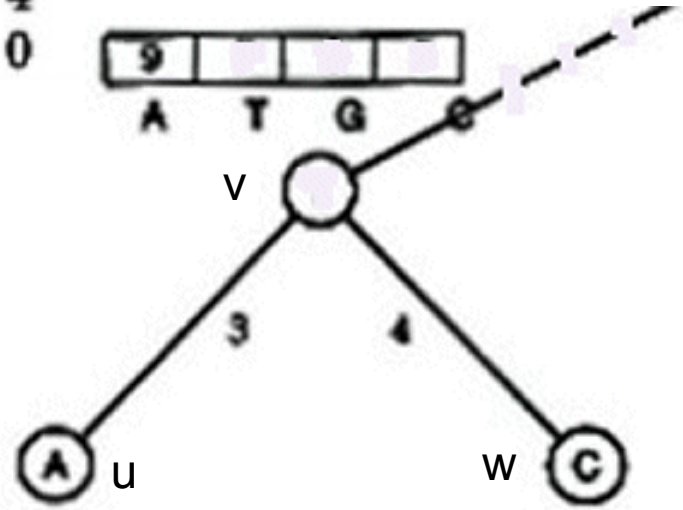


Algorithme de Sankoff (4)

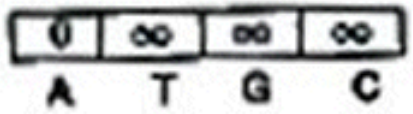
δ	A	T	G	C
A	0	3	4	9
T	3	0	2	4
G	4	2	0	4
C	9	4	4	0

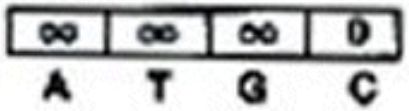
$$s_t(v) = \min_i \{s_i(u) + \delta_{i,t}\} + \min_j \{s_j(w) + \delta_{j,t}\}$$

$s_A(v) = 0 + 9 = \mathbf{9}$



	$s_j(u)$	$\delta_{j,A}$	sum
A	∞	0	∞
T	∞	3	∞
G	∞	4	∞
C	0	9	**9**



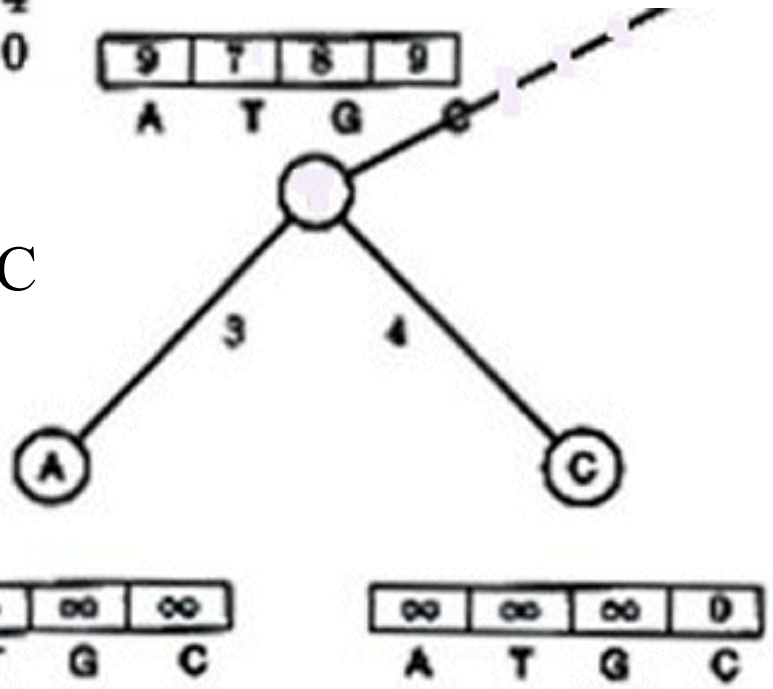


Algorithme de Sankoff (5)

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

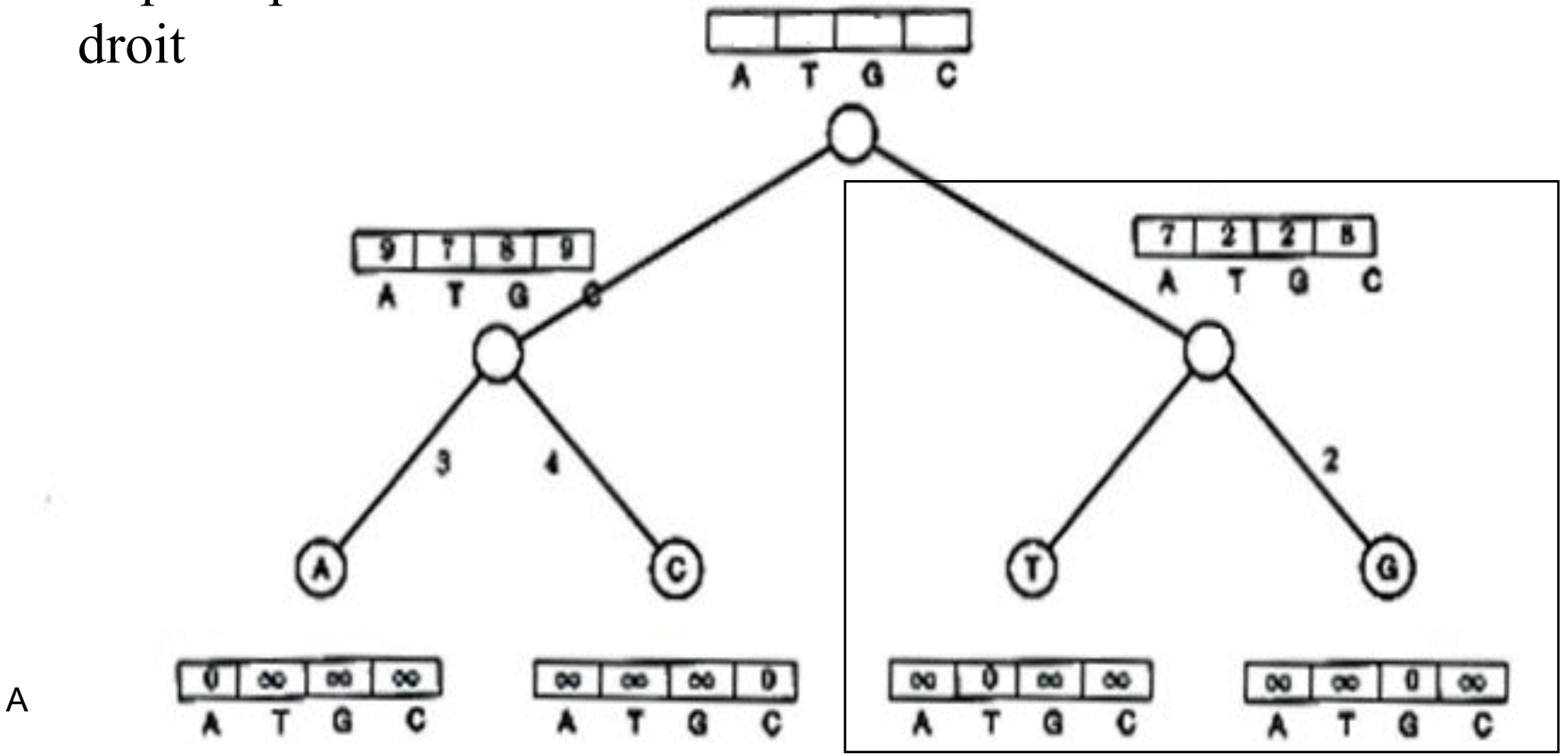
$$s_t(v) = \min_i \{s_i(u) + \delta_{i,\,t}\} + \min_j \{s_j(w) + \delta_{j,\,t}\}$$

Répéter pour T, G, C



Algorithme de Sankoff (6)

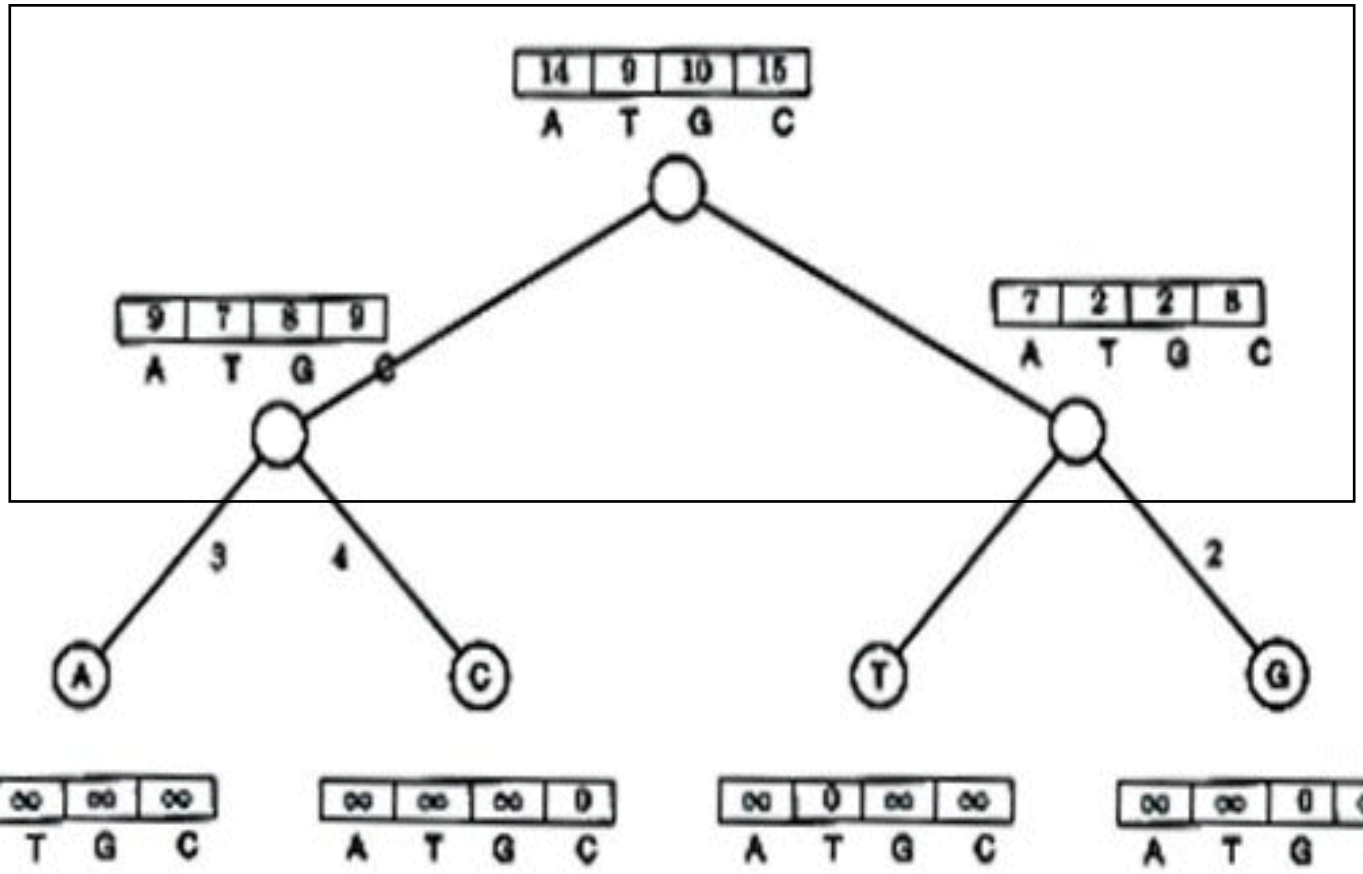
Répéter pour le sous-arbre droit



A

Algorithme de Sankoff (7)

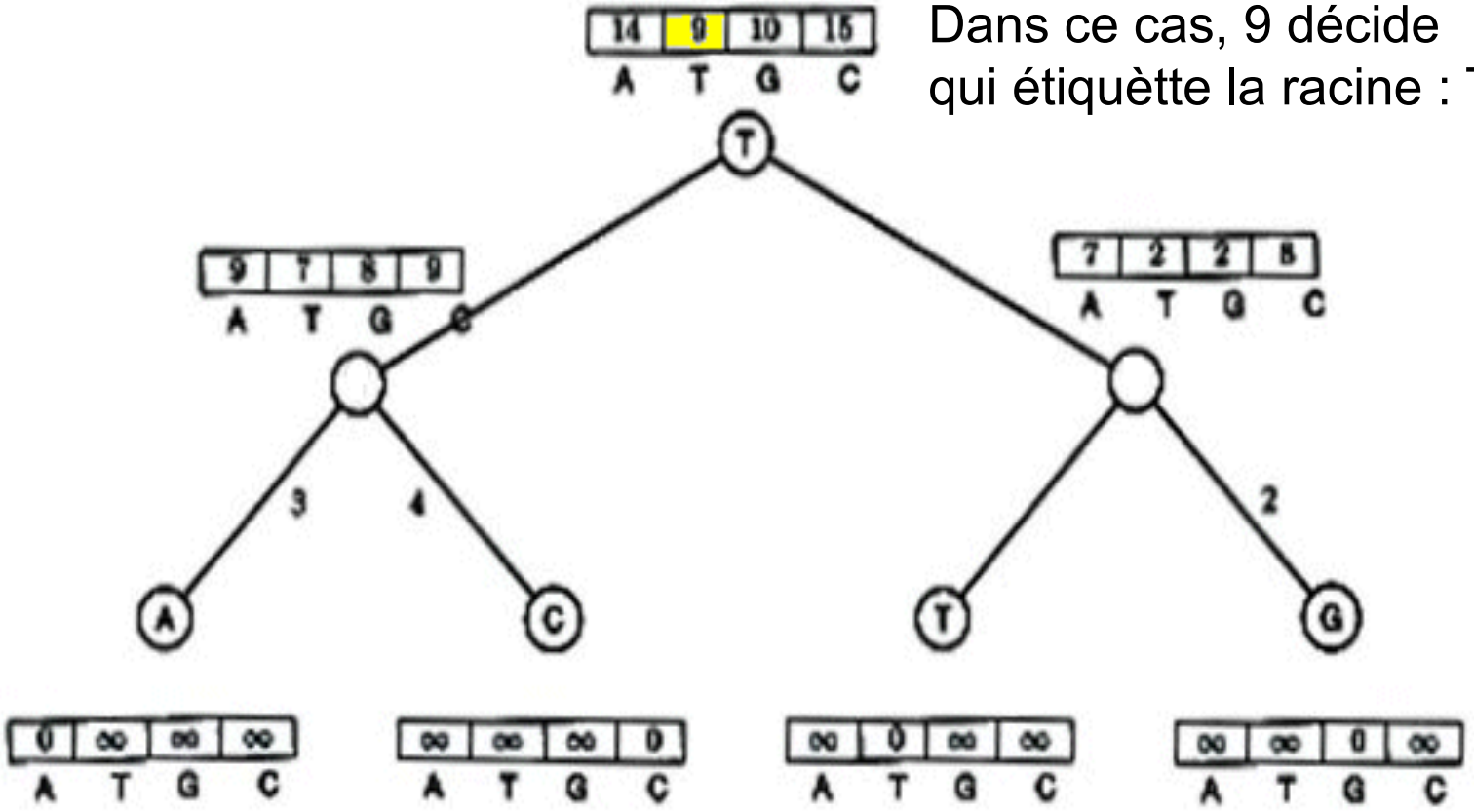
Répéter pour la racine



A

Algorithme de Sankoff (8)

Le plus petit score sur la racine est le score de parcimonie pondérée minimum



Dans ce cas, 9 décide qui étiquètte la racine : T

A

Algorithme de Sankoff: voyager de la racine vers les feuilles

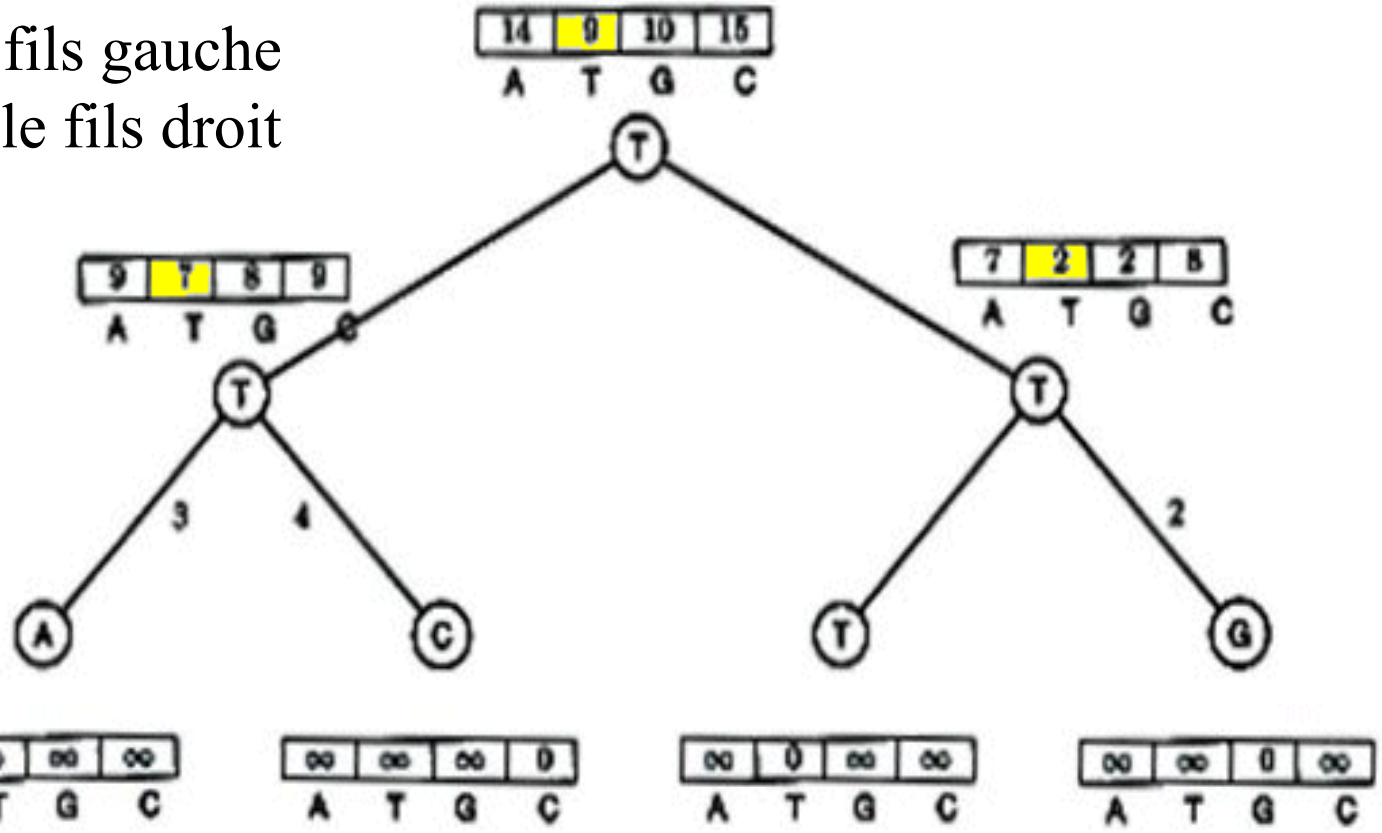
Etape 2:

- Les scores à la racine ont été calculés en remontant dans l'arbre.
- Après que le score à la racine est calculé, l'algorithme descend dans l'arbre et affecte chaque noeud avec un caractère optimale

Algorithme de Sankoff (9)

9 est derivé de 7 + 2

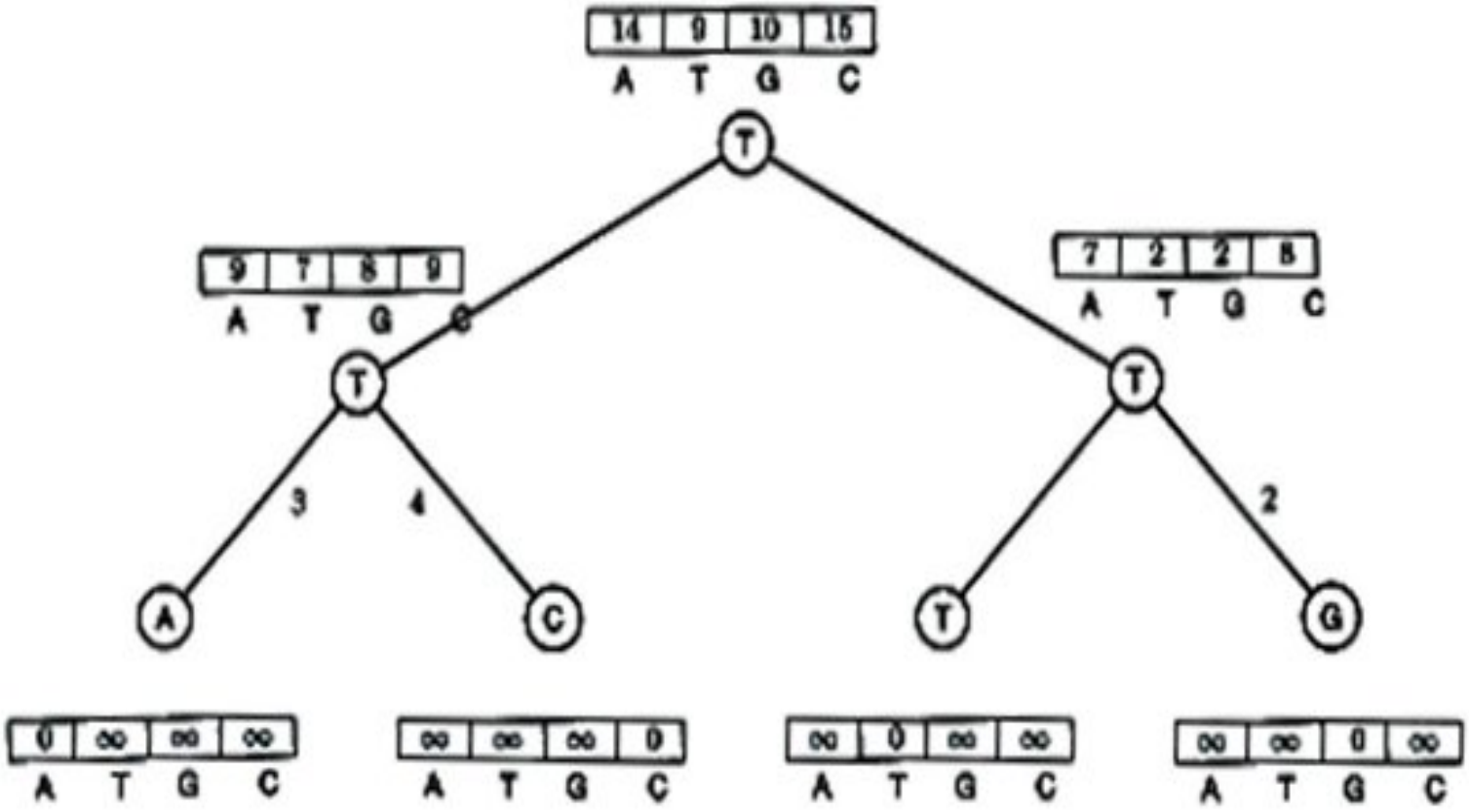
Donc le fils gauche est T, et le fils droit est T



A

Algorithme de Sankoff (10)

Et l'arbre est alors étiqueté…



A.

Algorithme de Fitch

Il s'agit d'un algorithme de programmation dynamique qui résout le petit problème de parcimonie :

Etape 1:

Assigner un ensemble de lettres à chaque nœud de l'arbre:

- Si les deux ensembles de caractères des fils chevauchent, alors on considère l'intersection des ensembles.
- Sinon, on fait l'union des ensembles. (Par exemple, si le noeud que l'on examine a un fils gauche étiqueté {A, C} et un fils droit étiqueté {T}, le noeud sera étiqueté par {A, C, T})

Algorithme de Fitch

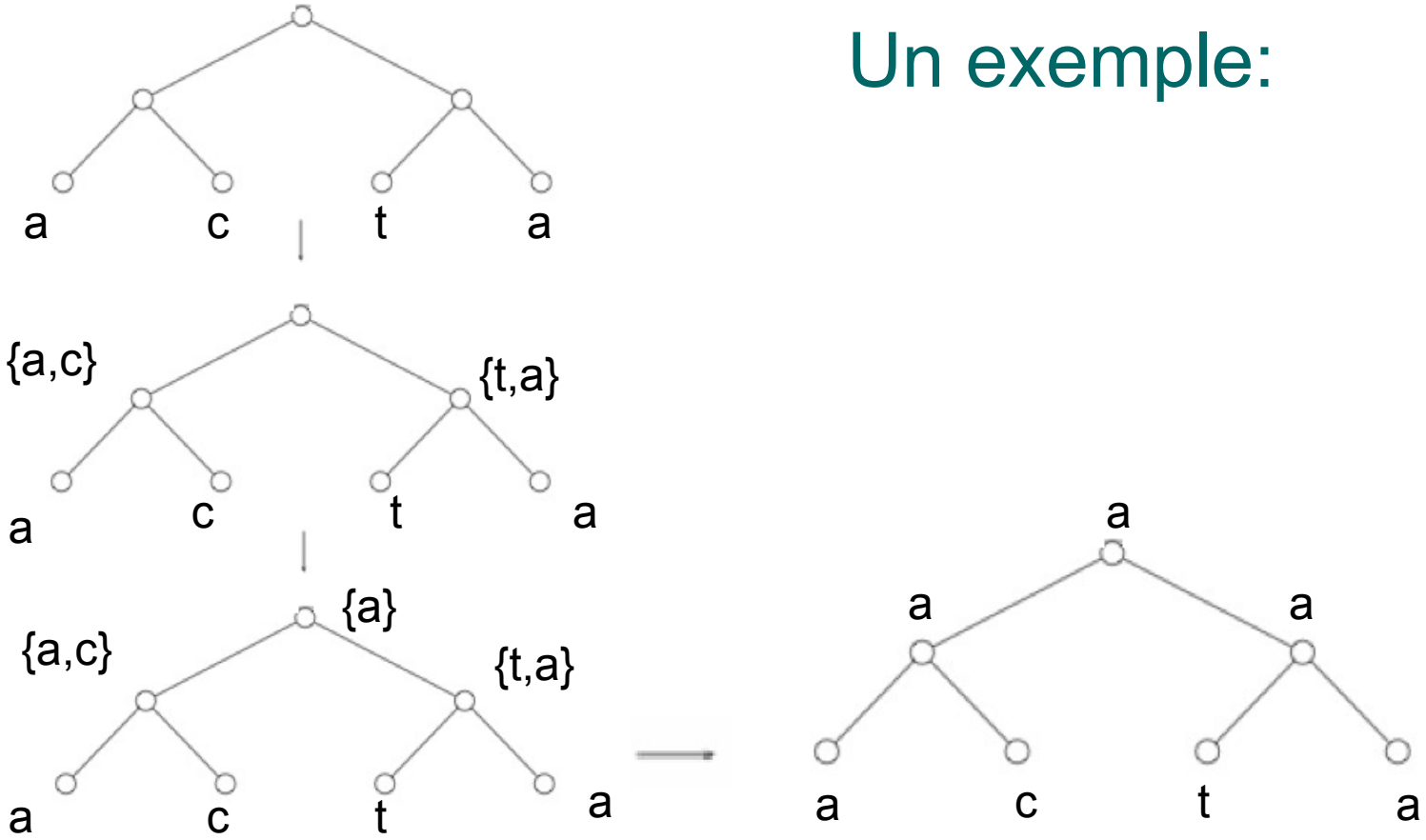
Etape 2:

Assigner des étiquettes à chaque noeud, en traversant l'arbre de la racine vers les feuilles

- Assigner la racine de façon arbitraire en choisissant dans l'ensemble des lettres déterminées à l'étape 1.
- Pour tout autre noeud, si l'étiquette du père est dans son ensemble de lettres, assigner lui l'étiquette du père.
- Sinon, choisir une lettre arbitraire dans l'ensemble de ses étiquettes possibles.

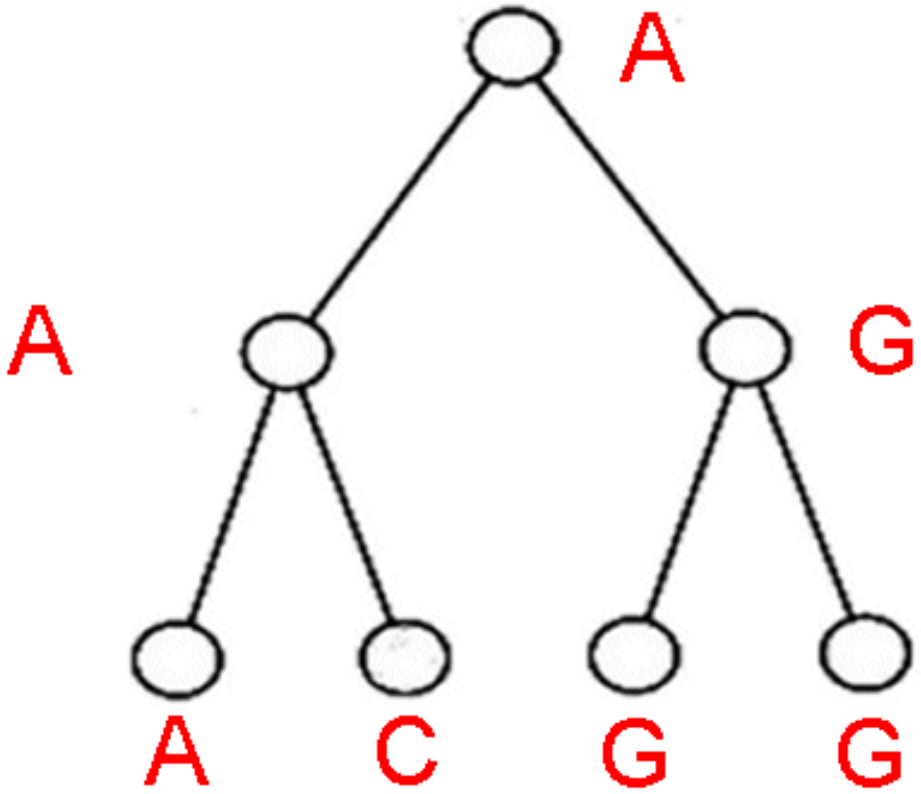
Algorithme de Fitch (1)

Un exemple:



Algorithme de Fitch (2)

Un autre exemple :

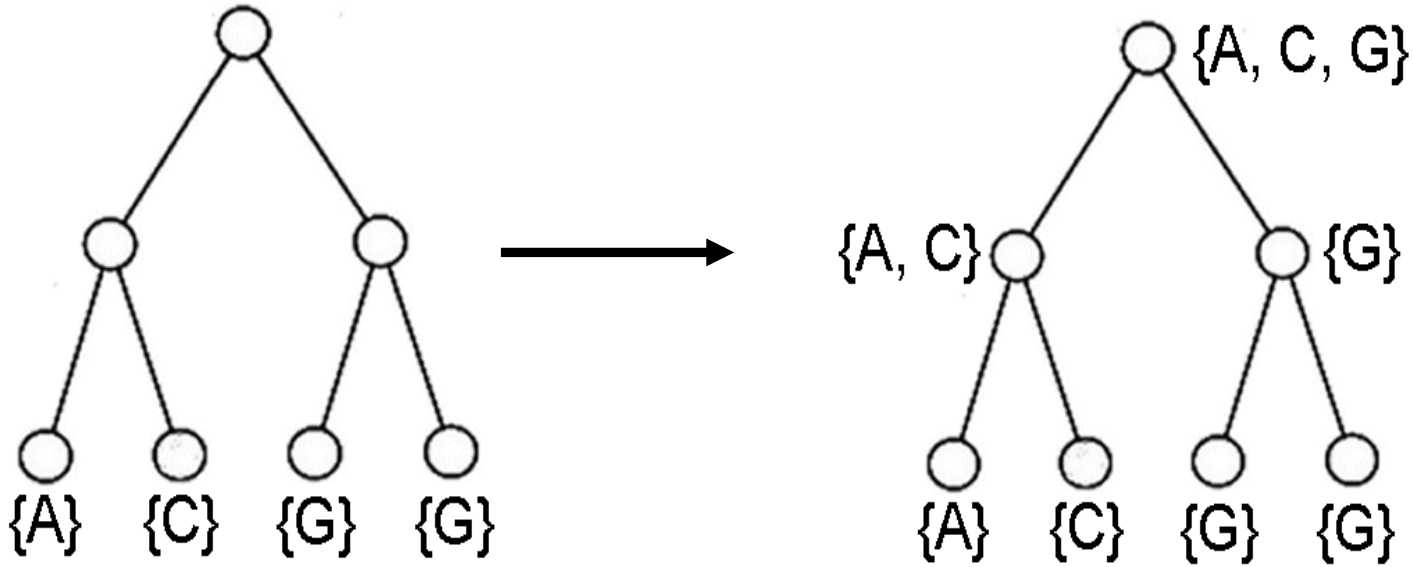


Fitch vs. Sankoff

- Les deux algorithmes ont un temps O(*nk*)
- Sont-ils effectivement différents ?
- On les compare …

Fitch

Comme on a vu avant:



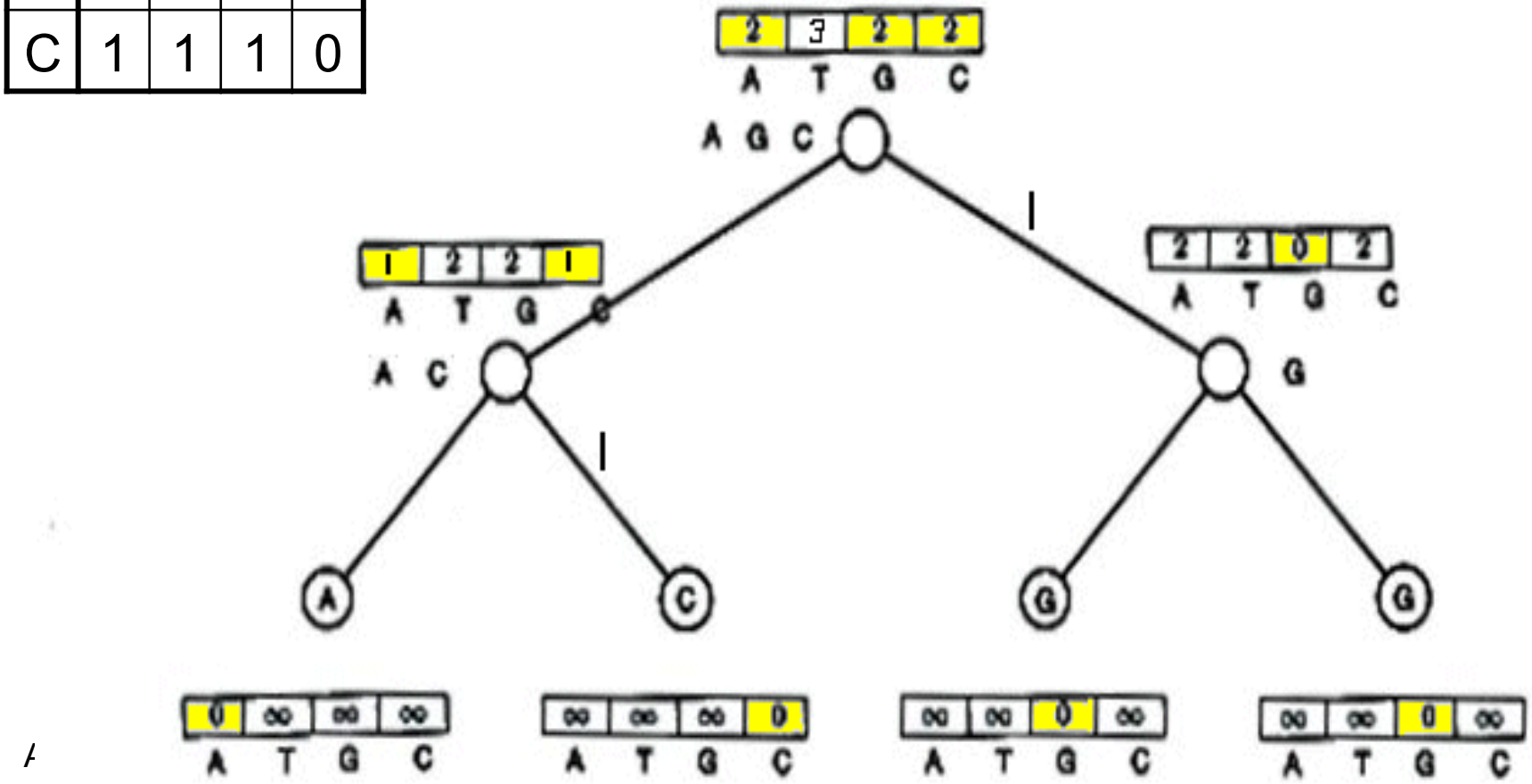
avec la matrice de score (pour l'algorithme de Fitch) qui est simplement:

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

On va résoudre maintenant le même problème avec l'algorithme de Sankoff et la même matrice de score.

Sankoff

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



Sankoff vs. Fitch

- L'algorithme de Sankoff donne le **même** ensemble d'étiquettes **optimales** que l'algorithme de Fitch.

- Pour l'algorithme de Sankoff, un caractère *t* est *optimale* pour un noeud *v* si $s_t(v) = \min_{1 \leq i \leq k} s_i(v)$

  - Dénoter l'ensemble des lettres optimale au noeud *v* comme *S*(*v*)
    - si *S*(*fils gauche*) et *S*(*fils droit*) chevauchent, *S*(*père*) est l'intersection
    - sinon, faire l'union de *S*(*fils gauche*) et *S*(*fils droit*)
    - Mais ceci est aussi la récurrence de Fitch

- Les deux algorithmes sont **identiques** (sur des matrices de score 0/1)

Problème de parcimonie large

Problème de parcimonie large

Entrée: Une matrice *n* x *m* *M* décrivant *n* espèces, chacune représentée par un mot de *m* caractères

Sortie: Un arbre *T* avec *n* feuilles étiquetés par les *n* lignes de la matrice *M*, et un étiquetage des noeuds internes tel que le score de parcimonie est minimisé sur tous les possibles arbres et
tous les possibles étiquetages des noeuds internes.

- L'espace de recherche possible est énorme, et particulièrement quand n croit
  - $(2n - 3)!!$ arbres racinés possibles
  - $(2n - 5)!!$ arbres non-raciné possibles

- Le problème est NP-complet
  - Une recherche exhaustive est possible pour des *n* petits (< 10)

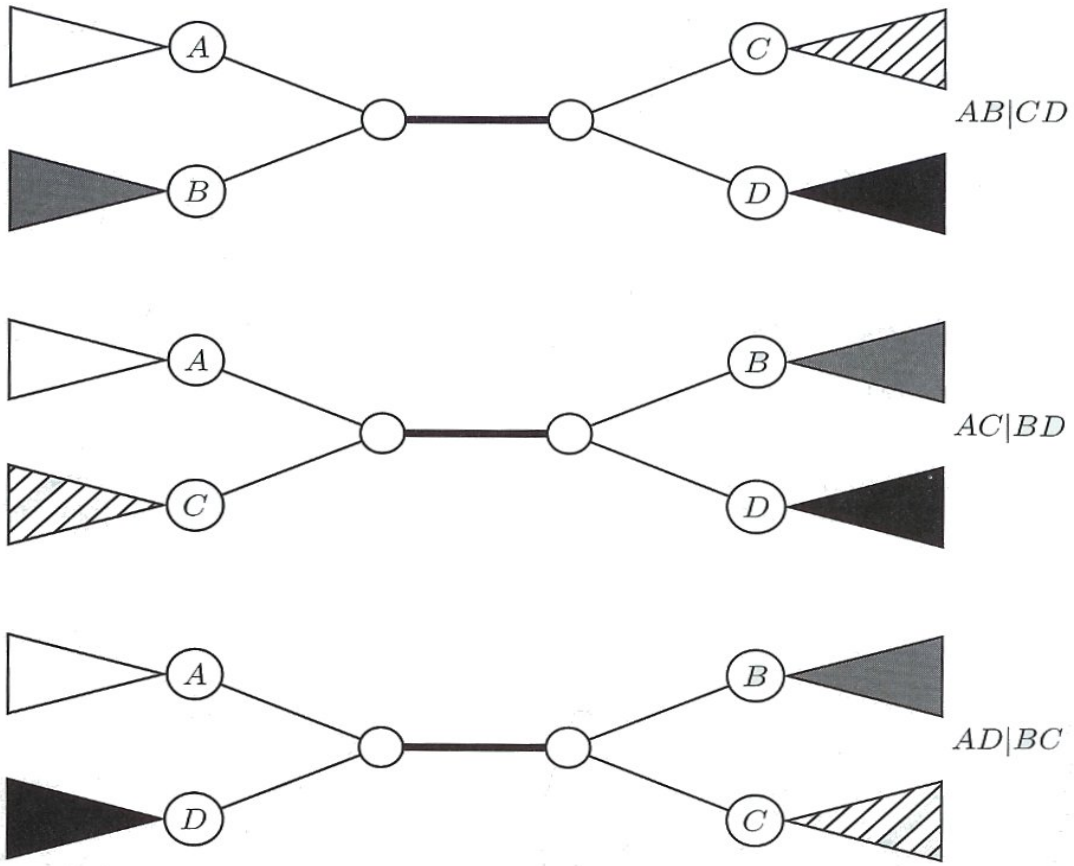
- Donc, branch and bound ou des algorithmes heuristiques sont employés

Nearest Neighbor Interchange : un algorithme gourmand

Il s'agit d'un algorithme de réarrangement des branches

- Il évalue seulement un sous-ensemble des arbres possibles
- Il défini un *voisinage* d'un arbre comme celui joignable par un *échange du voisin le plus proche :*
  - Il s'agit du réarrangement (à définir) des quatre sous-arbres déterminés par un arc interne.
  - Etant donnée une règle de manipulation d'un sous-arbre, ils existent seulement 3 arrangements différents par arc.

Nearest Neighbor Interchange



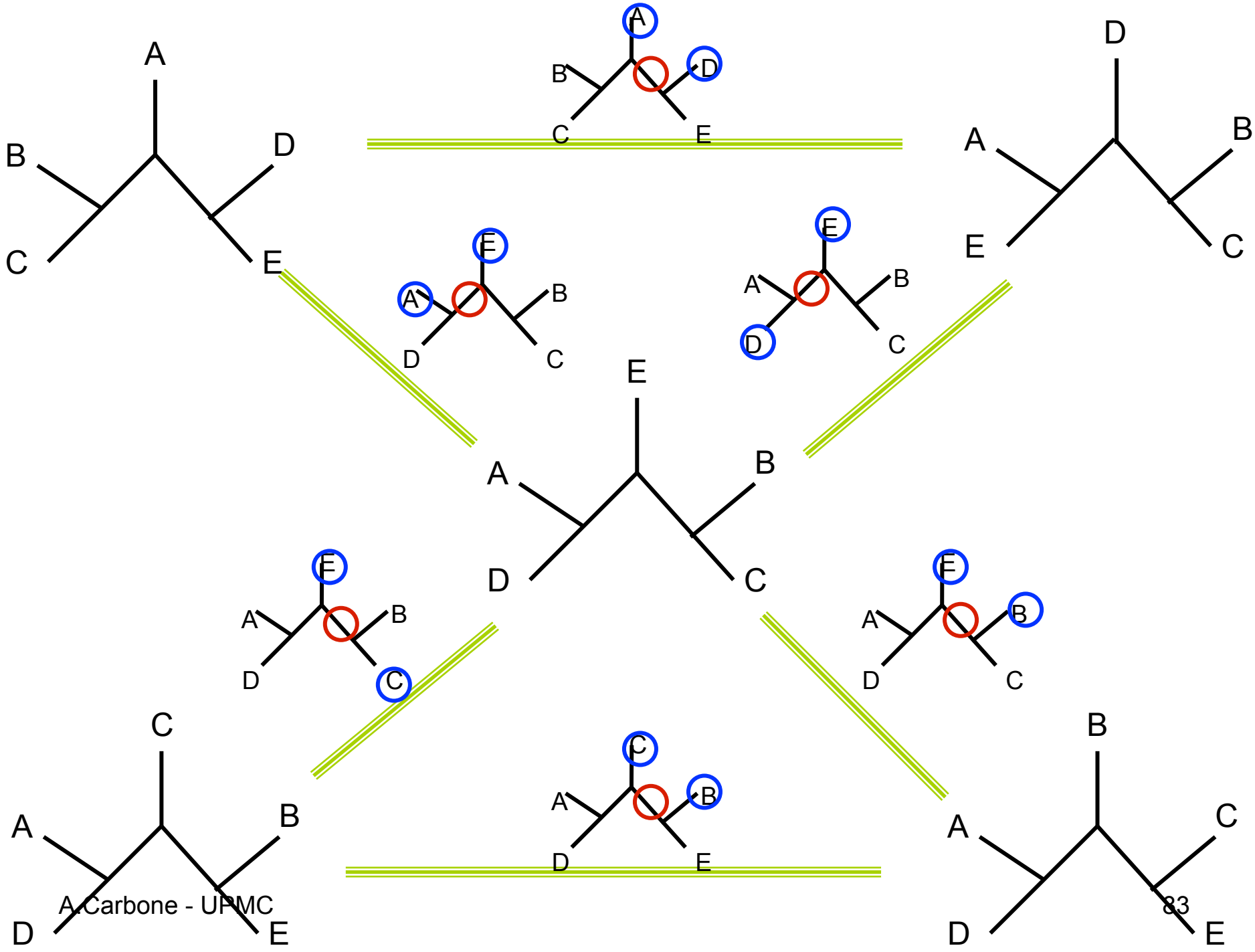
Algorithme : Nearest Neighbor Interchange

- Commencer par un arbre choisi arbitrairement et tester les voisins
- Bouger à un voisin s'il donne un score de parcimonie amélioré
- Il n'y a pas de moyens pour savoir si le résultat obtenu est l'arbre **le plus parcimonieux**
- Il peut s'arrêter à un optimum locale

Relation de voisinage possible:

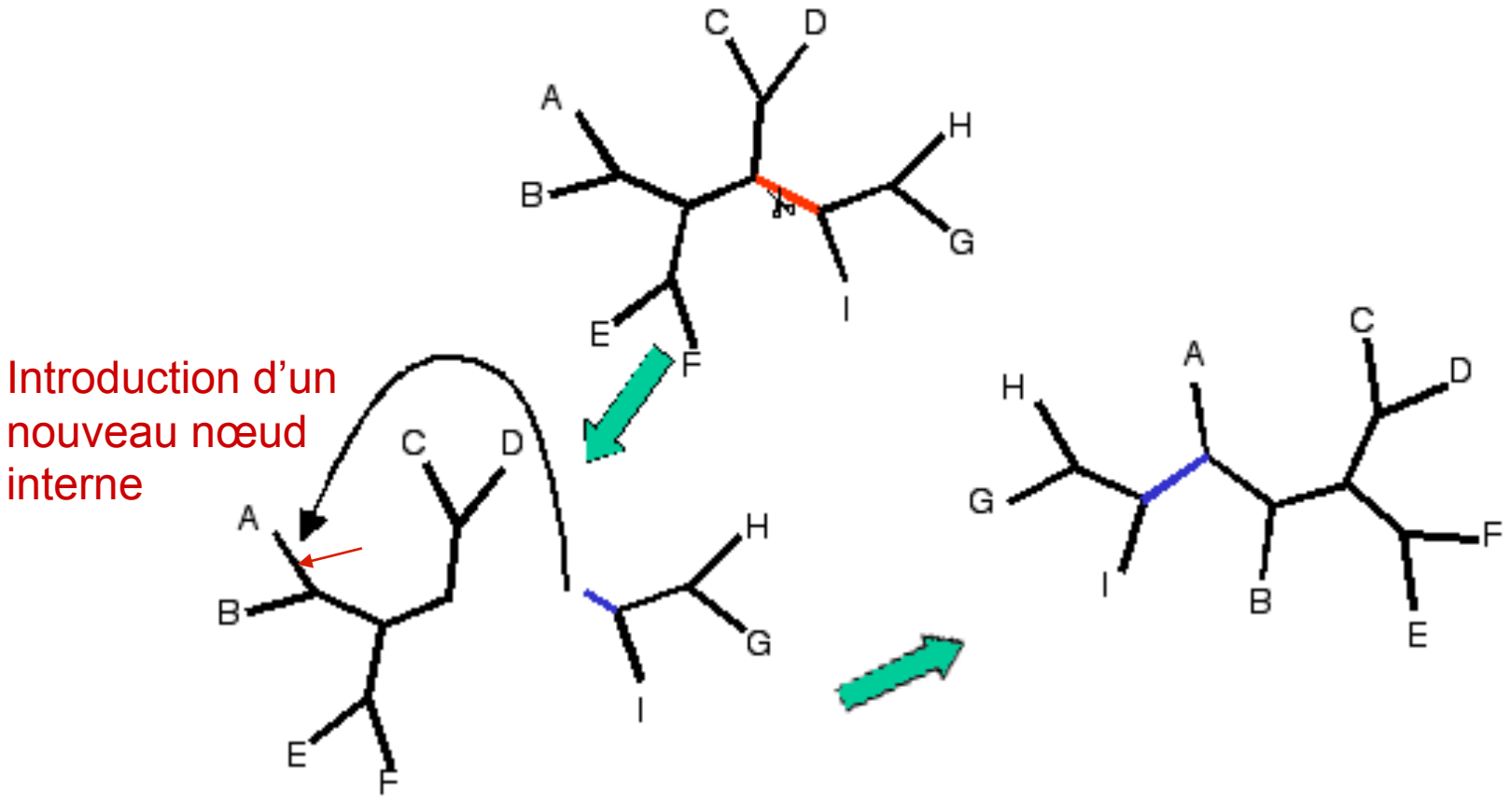
deux arbres sont voisins s'ils peuvent être présentés comme des quartets joignant les mêmes 4 sous-arbres.

Par exemple:


A
B
C
D
E

Tailler les sous-arbres et les regreffer

Un algorithme de rearrangement des branches

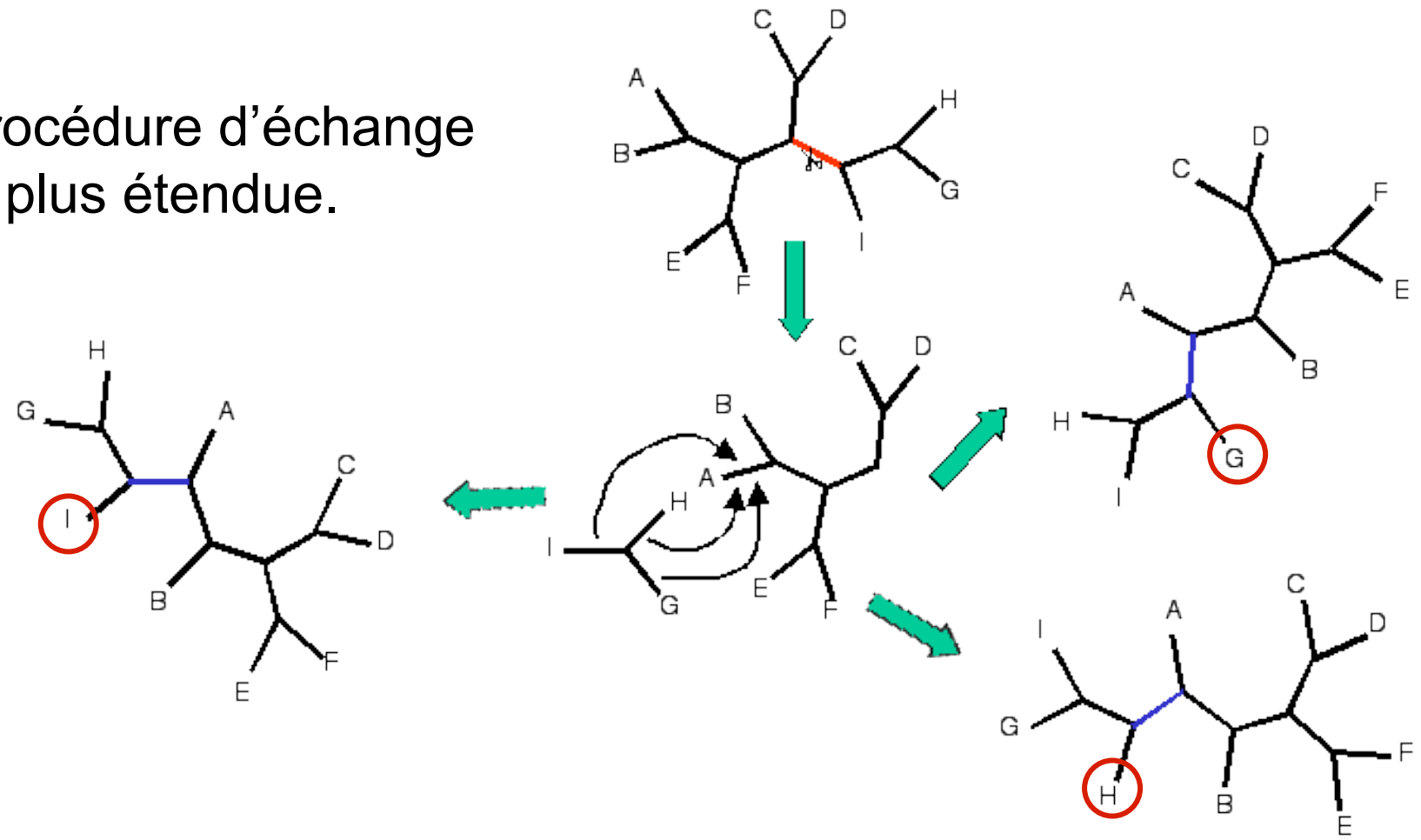


http://artedi.ebc.uu.se/course/BioInfo-10p-2001/Phylogeny/Phylogeny-TreeSearch/SPR.gif

Bisection d'un arbre et Reconnection

Un autre algorithme de rearrangement des branches

Procédure d'échange
la plus étendue.



Problèmes avec la reconstruction phylogénétique

- Important: une seule reconstruction phylogénétique fourni très souvent une image incomplète de la réalité.

- Quand plusieurs méthodes (parcimonie, basés sur la distance, probabilistes) donnent le même résultat, alors il est plus probable d'avoir obtenu une réponse correcte.

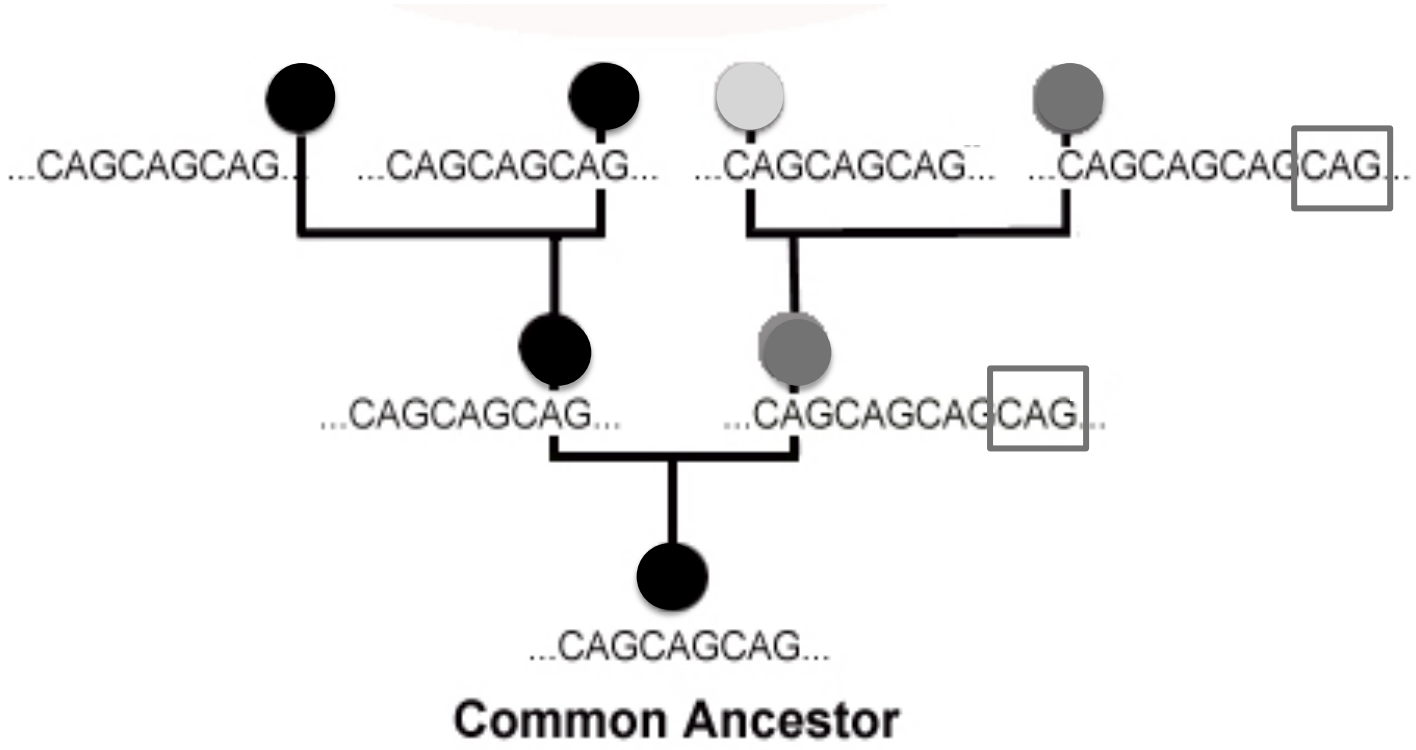
1. Homoplasie

- Etant données:
  - 1: ...CAGCAGCAG...
  - 2: ...CAGCAGCAG...
  - 3: ...CAGCAGCAGCAG...
  - 4: ...CAGCAGCAG...
  - 5: ...CAGCAGCAG...
  - 6: ...CAGCAGCAG...
  - 7: ...CAGCAGCAGCAG...

Les séquences 1, 2, 4, 5 et 6 semblent avoir évolues à partir d'un ancêtre commun, avec une insertion qui amène à la présence de 3 et 7

Homoplasie

- Mais si l'arbre vrai était celui-ci ?



Homoplasie

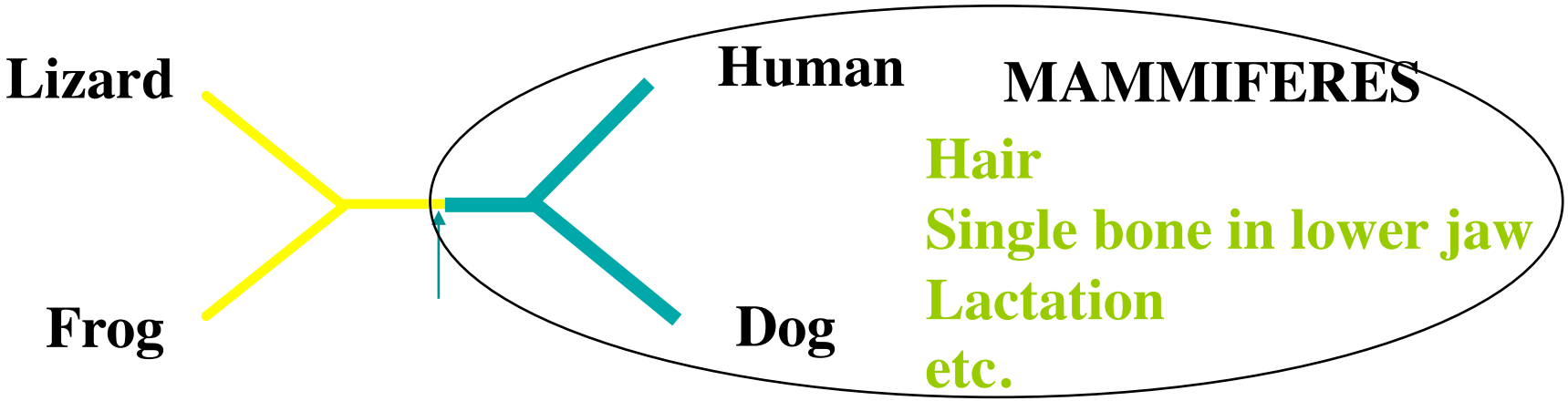
- ● a évolué séparément par rapport aux ● mais la parcimonie grouperait les ● ensemble comme ayant évolués d'un ancêtre commun.

- Homoplasie: évolution indépendante (ou parallèle) d'un même/similaire caractère.

- Les résultats de parcimonie **minimisent** l'homoplasie, de telle façon que si l'homoplasie est fréquente, la parcimonie peut donner des résultats très erronés.

1. Caractères contradictoires: les queues

Un arbre phylogénétique a une plus forte probabilité d'être correcte quand il est soutenu par plusieurs caractères, comme vu dans l'exemple:



Note: les queues sont homoplasiques

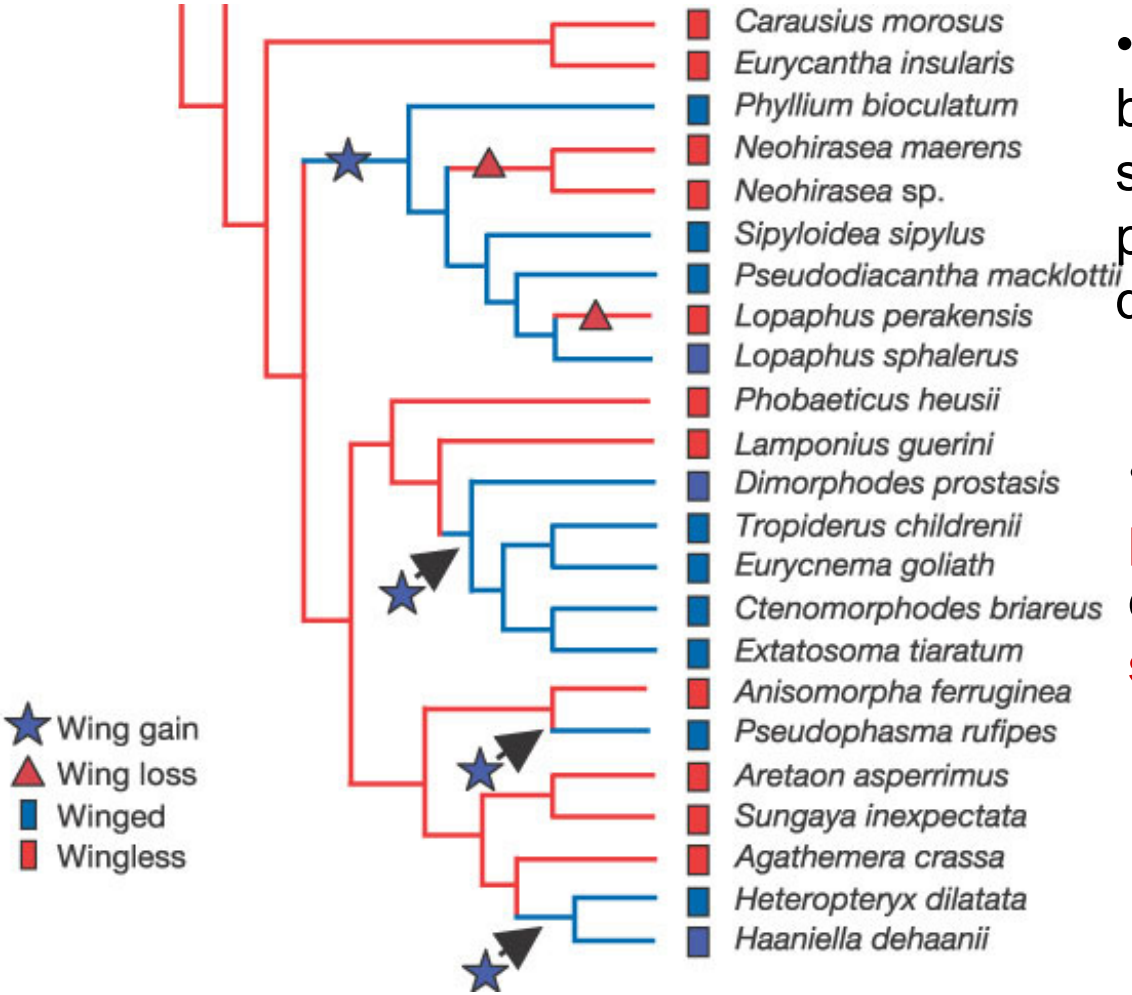
2. Combien de fois l'évolution a re-inventé les ailes?

- Whiting, et. al. (2003) a regardé aux insectes avec et sans ailes.

Réinventer les ailes

- Etudes précédents ont montré des transitions “avec → sans” ailes
- Les transitions “sans → avec” ailes sont beaucoup plus compliquées (elles demandent le développement de plusieurs chemins biochimiques nouveaux)
- La reconstruction de plusieurs arbres phylogénétiques a amené à établir toujours une re-évolution des ailes.

Arbre phylogénétique le plus parcimonieux des insectes avec et sans ailes



- arbre phylogénétique basé sur les **deux :** séquences d'ADN et présence/absence d'ailes

- la reconstruction la plus parcimonieuse donne un **ancêtre sans ailes**

Pourquoi les insectes sans ailes volent encore?

Car les reconstructions phylogénétiques les plus parcimonieuses demandent une re-invention des ailes, il est possible que les chemins de développement des ailes soient conservés dans les insectes sans ailes.

Références

[1] L.Billera, S.Holmes, K.Vogtmann, Geometry of the space of phylogenetic trees, *Advances in Applied Mathematics*, 27:733-767, 2001.

[2] H. L. Bodlaender, M. R. Fellows, and T. J. Warnow. Two strikes against perfect phylogeny. In *Proc. 19th*. Springer, 1992.

[3] C. Bron and J. Kerbosch. Finding all cliques of an undirected graph. *Communications of the Association for Computing Machinery*, 16:575–577, 1973. Algorithm 457.

[4] W. H. E. Day. Computational complexity of inferring phylogenies from dissimilarity matrices. *Bulletin of Mathematical Biology*, 49:461–467, 1986.

[5] J. Felsenstein. Maximum likelihood and minimum-steps methods for estimating evolutionary trees from data on discrete characters. *Systematic Zoology*, 22:240–249, 1973.

[6] J. Felsenstein. Phylogenies from molecular sequences: inference and reliability. *Annuals Rev. Genetics*, 22:521–565, 1988.

[7] J. Felsenstein. *Inferring Phylogenies*. ASUW Publishing, Seattle, WA, 1998.

[8] W. M. Fitch. Toward defining course of evolution: minimum change for a specified tree topology. *Systematic Zoology*, 20:406–416, 1971.

[9] L. R. Foulds and R. L. Graham. The Steiner problem in phylogeny is NP-complete. *Advances in Applied Mathematics*, 3:43–49, 1982.

[10] Jr. G.F. Estabrook, C.S. Johnson and F.R. McMorris. An algebraic analysis of cladistic characters. *Discrete Mathematics*, 16:141–147, 1976.

[11] D. Gusfield. The steiner tree problem in phylogeny. Technical Report 334, Yale University, Computer Science Dep., 1984.

[12] D. Gusfield. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, 1997.

[13] M. D. Hendy and D. Penny. Branch and bound algorithms to determine minimal evolutionary trees. *Mathematical Biosciences*, 60:133–142, 1982.

[14] T. H. Jukes and C. Cantor. *Mammalian Protein Metabolism*, pages 21–132. Academic Press, New York, 1969.

[15] S. Kannan and T. Warnow. Inferring evolutionary history from dna sequences. *SIAM J. Comput.*, 23:713–737, 1994.

[16] M. Kimura. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *Molecular Evolution*, 16:111–120, 1980.

[17] W. H. Li. *Molecular Evolution*, chapter 5, pages 105–112. Sinauer Associates, Inc., Publishers, Sunderland, Massachusetts, 1997.

[18] T. Yano M. Hasegawa, H. Kishino. Dating of the human-ape splitting by a molecular clock of mitochondrial dna. *Molecular Evolution*, 22:160–174, 1985.

[19] C. D. Michener and R. R. Sokal. A quantitative approach to a problem in classification. *Evolution*, 11:130–162, 1957.

[20] A. Krogh G. Mitchison R. Durbin, S. Eddy. *Biological Sequence Analysis*, chapter 7, pages 160–191. Cambridge University Press, Cambridge, United Kingdom, 1998.

[21] N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4:406–425, 1987.

[22] D. Sankoff. Minimal mutation trees of sequences. *SIAM Journal of Applied Mathematics*, 28:35–42, 1975.

[23] M. Steel. The complexity of reconstructing trees from qualitative characters and subtrees. *J. of Classification*, 9:91–116, 1992.

[24] A. Wilson and R. Cann. The recent african genesis of humans. *Scientific American*, April, 1992.

[25] E. O. Wilson. A consistency test for phylogenies based on contemporaneous species. *Systematic Zoology*, 14:214–220, 1965.

Approches probabilistes

Etant donné un arbre, on veut souvent avoir une mesure statistique de sa bonne représentation des données.

On peut utiliser le score de vraisemblance pour estimer notre hypothèse, dans ce cas constituée par l'arbre : pour un ensemble d'espèces avec valeur(s) observée(s) M, on choisira un arbre maximisant P(M|T).

Dans la suite on travaillera sous l'hypothèse que la topologie de l'arbre soit connue, et on montrera comment trouver la longueur des branches optimale.

Pour cela, on montrera comment calculer la vraisemblance d'un arbre efficacement.

Vraisemblance d'un arbre

Définition : les étiquettes, ou états, sont des vecteurs de m caractères associés a chaque espèce, ou bien à chaque nœud dans l'arbre. Une reconstruction est un étiquetage complet des nœuds internes de l'arbre. La longueur d'une branche t_{vu} est la longueur de l'arc entre les nœuds v et u, et elle mesure le temps biologique, ou bien la distance génétique entre les espèces associées avec ces nœuds.

On fait l'hypothèse que les caractères sont indépendants par paires, et que **le branchement est un processus de Markov**, cad que la probabilité d'un nœud d'avoir une étiquette donnée est une fonction seulement de l'état du père et de la longueur t de l'arc entre père et fils.

On considère une fonction de distance qui calcule la probabilité que l'état x se transformera dans l'état y dans un temps t_{vu} : $P_{x \to y}(t_{vu})$.

La fréquence des caractères est fixée dans l'histoire évolutionnaire, et elle est décrite par P(x).

Problème (calculer la vraisemblance d'un arbre)

Entrée:

- une matrice M qui décrit un ensemble de m caractères pour chacune des n espèces
- un arbre T avec les espèces qui étiquettent les feuilles et avec une longueur des branches connues t_{vu}.

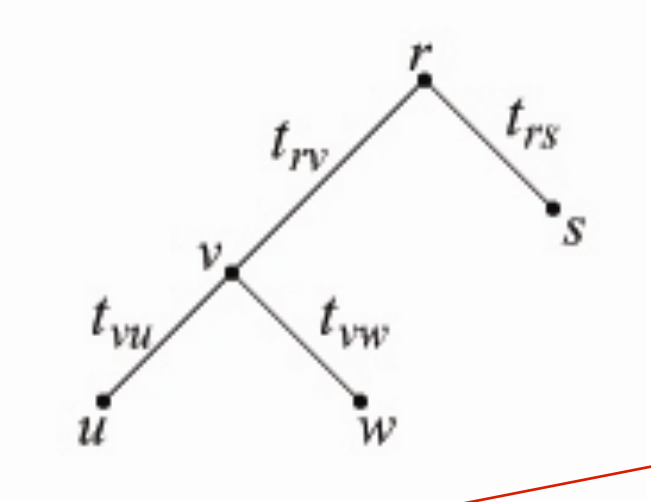
Sortie: maximisation de la probabilité P(M|T) obtenue en cherchant une reconstruction optimale T, les étiquettes des noeuds internes et les longueurs des branches.

Hypothèses:

- Les caractères sont indépendants
- Modèle de Markov: la probabilité d'une étiquette dépend seulement du père.

Cas simple : il existe seulement un caractère qui identifie chaque espèce.

Les étiquettes des nœuds internes sont inconnues, et donc on a besoin de faire la somme sur toute possible reconstruction.

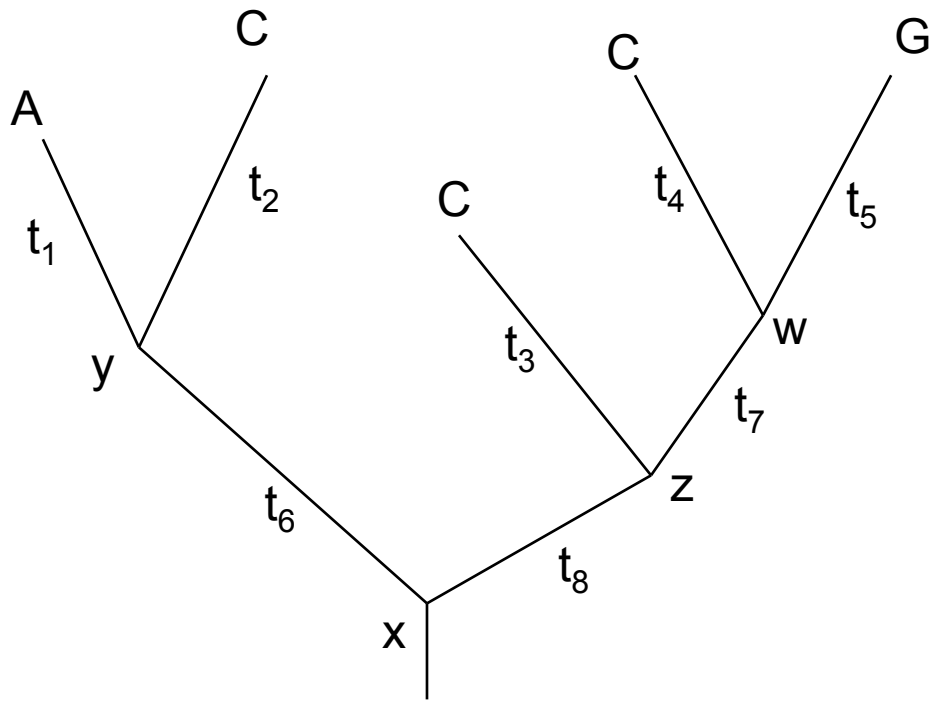


Vraisemblance - Likelihood

$$L = P(M|T) = \sum_r \sum_v P(r) \cdot P_{r \to s}(t_{rs}) \cdot P_{r \to v}(t_{rv}) \cdot P_{v \to u}(t_{vu}) \cdot P_{v \to w}(t_{vw})$$

où r et v sont des possibles étiquettes/caractères pour les nœuds.

Exemple:



$$P(A,C,C,C,G,x,y,z,w|T) = P(x)\,P(y|x,t_6)\,P(A|y,t_1)\,P(C|y,t_2)\,P(z|x,t_8)\,P(C|z,t_3)\,P(w|z,t_7)\,P(C|w,t_4)\,P(G|w,t_5)$$

L'expression est pleine de termes: $4^4=256$ (4 nœuds internes qui peuvent avoir associés 4 possibles lettres)

n=10 , 262.144 termes

n=20, 274.877.906.944 termes

Cas pour des caractères multiples : on répétera le calcul précèdent pour chacun des caractères séparément, et puis on multiplie le résultats (rappel que les caractères sont indépendant par paires). L'équation est:

$$
\begin{aligned}
L &= P(M|T) = \prod_{\text{character j}} P(M_j|T) \\
&= \prod_{\text{character j}} \left\{ \sum_{\text{reconstruction R}} P(M_j, R|T) \right\} \\
&= \prod_{\text{character j}} \left\{ \sum_{\text{reconstruction R}} \left[P(root) \cdot \prod_{\text{edge u}\rightarrow\text{v}} P_{u\rightarrow v}(t_{uv}) \right] \right\}
\end{aligned}
$$

Remarque: Par cette description, les arbres apparaissent comme racinés. Par contre, si le modèle est réversible, cad si $P(x)P_{x \rightarrow y}(t) = P(y)P_{y \rightarrow x}(t)$, alors l'arbre est sans racine, et une racine peut être choisie arbitrairement, sans changer la vraisemblance de l'arbre.

Algorithme :

On dénote $C_j(x,v) = P(\text{sous-arbre ayant racine } v \mid v_j = x)$

ou $C_i(x,v)$ est la vraisemblance conditionnelle du sous-arbre de racine v, cad la probabilité d'observer un sous-arbre donné au noeud v jusqu'à les feuilles, étant donné que v a l'étiquette x à la position du caractère j.

Initialisation:

Pour chaque feuille v et état x: $C_j(x,v) = \begin{cases} 1 & \text{si } v_j=x \\ 0 & \text{sinon} \end{cases}$

Recursion:

Traverser l'arbre en ordre postfixe; pour un nœud interne v avec fils u et w, calculer pour chaque état possible x (= valeur du caractère j):

$$C_j(x,v) = [\sum_y C_j(y,u) \cdot P_{x \to y}(t_{vu})] \cdot [\sum_y C_j(y,w) \cdot P_{x \to y}(t_{vw})]$$

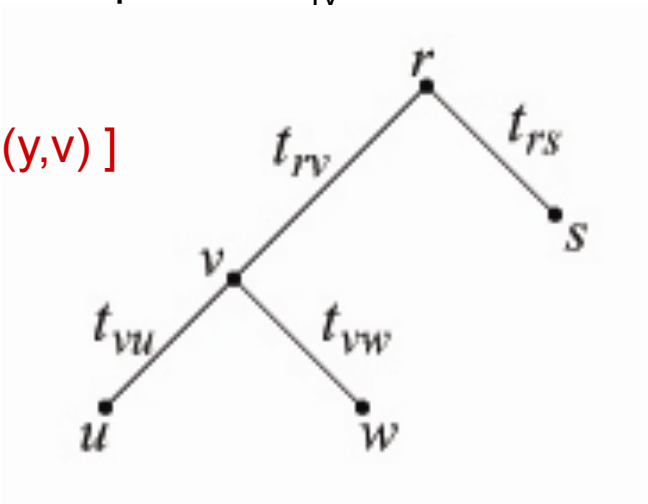
La solution finale est $L = \prod_{j=1}^{m} [\sum_x C_j(x,racine) \cdot P(x)]$

Complexité: pour n espèces, m caractères, et k états possibles pour chaque caractère, l'algorithme est $O(m \cdot k^2)$ sur O(n) nœuds. Donc le temps de l'algorithme est $O(n \cdot m \cdot k^2)$.

Recherche de la longueur des branches optimale pour une topologie d'arbre donnée

Supposons que l'on connaît toutes les longueurs a l'exception de t_{rv}.
Si r est la racine, alors

$$\log L = \sum_{j=1}^{m} \log [\sum_{x,y} P(x) \cdot C_j^v(x,r) \cdot P_{x \to y}(t_{rv}) \cdot C_j^r(y,v)]$$

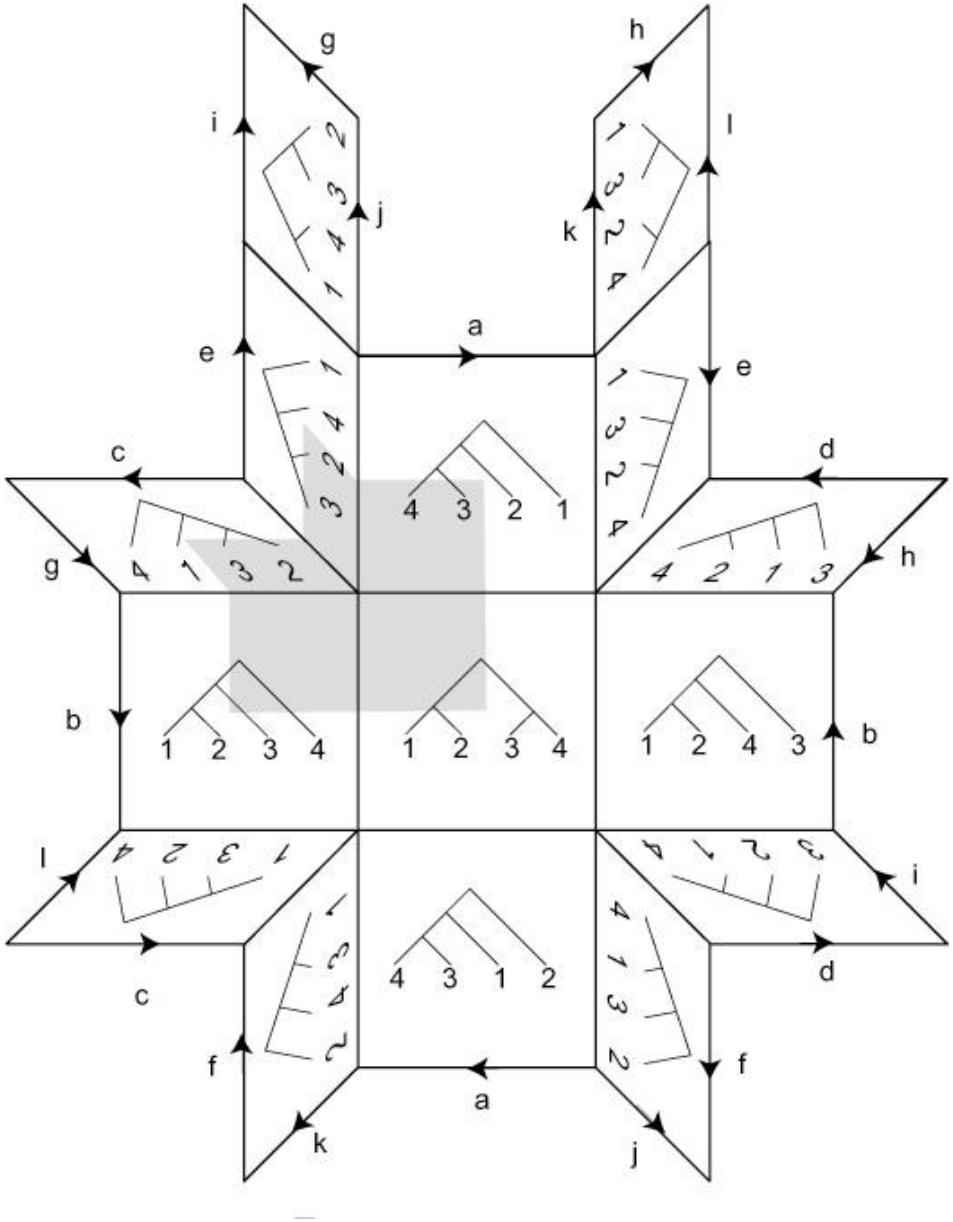


$C_j^u(x,y)$ dans un arbre ou u est la racine

On a besoin de maximiser logL par rapport a t_{rv}. Cela peut être réalisé a travers plusieurs méthodes (Newton-Raphson, EM).

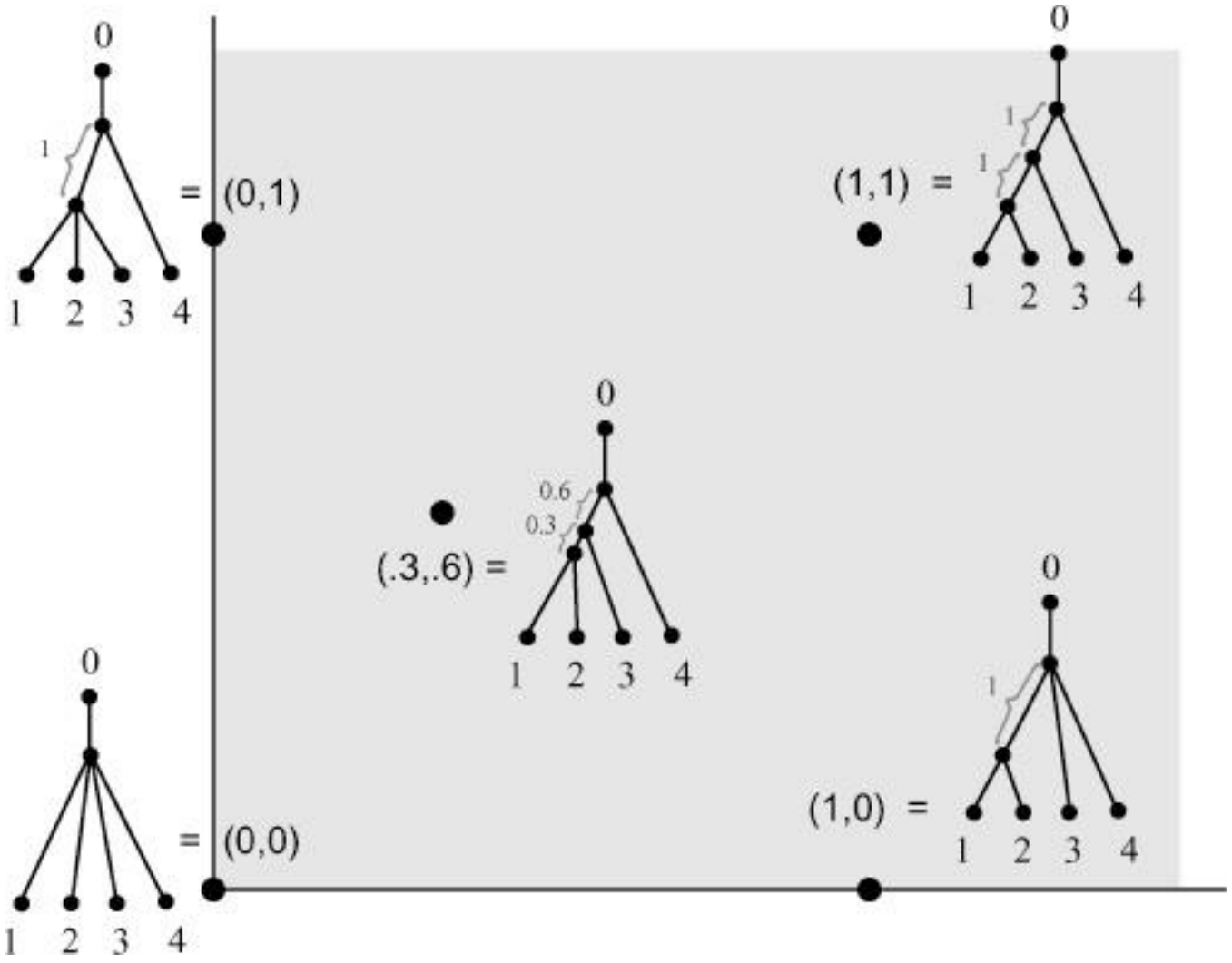
On peut essayer en suite d'optimiser la longueur des branches, en optimisant une branche a la fois. Ca marche assez bien. Apres quelques étapes dans l'arbre, calculant la longueur optimale de chaque arc, la vraisemblance converge, et le résultat est un arbre phylogénétique proche a l'optimale.

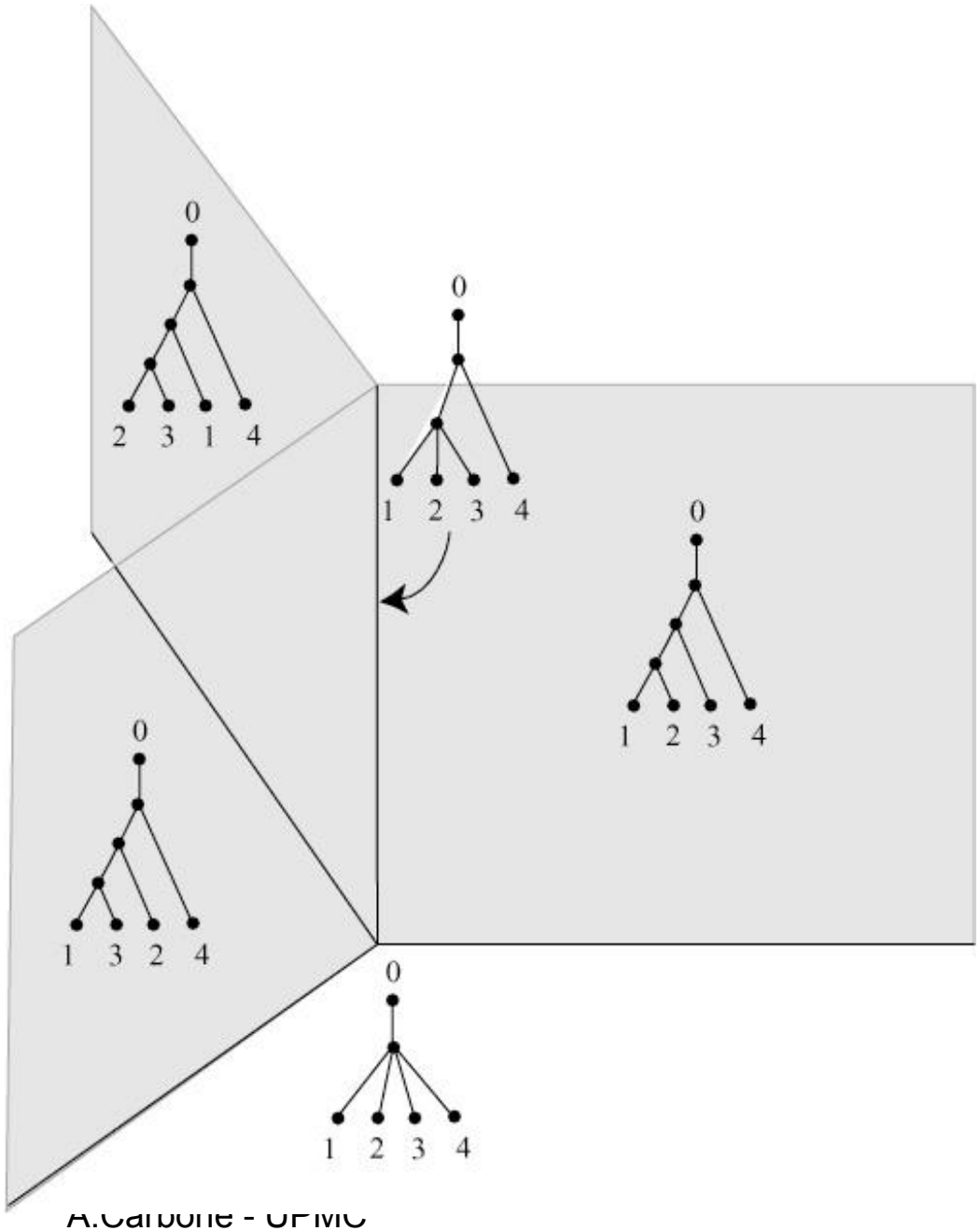
Géométrie de l'espace des arbres phylogéniques

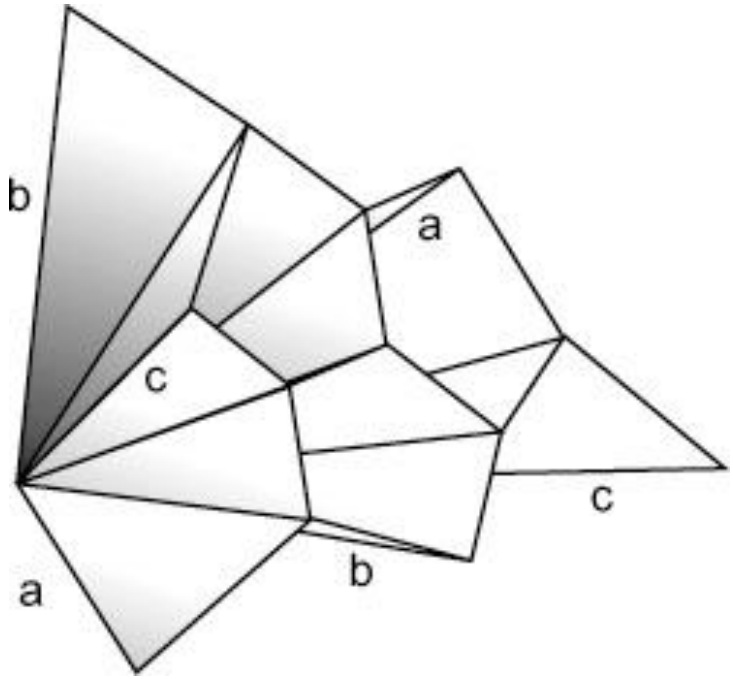


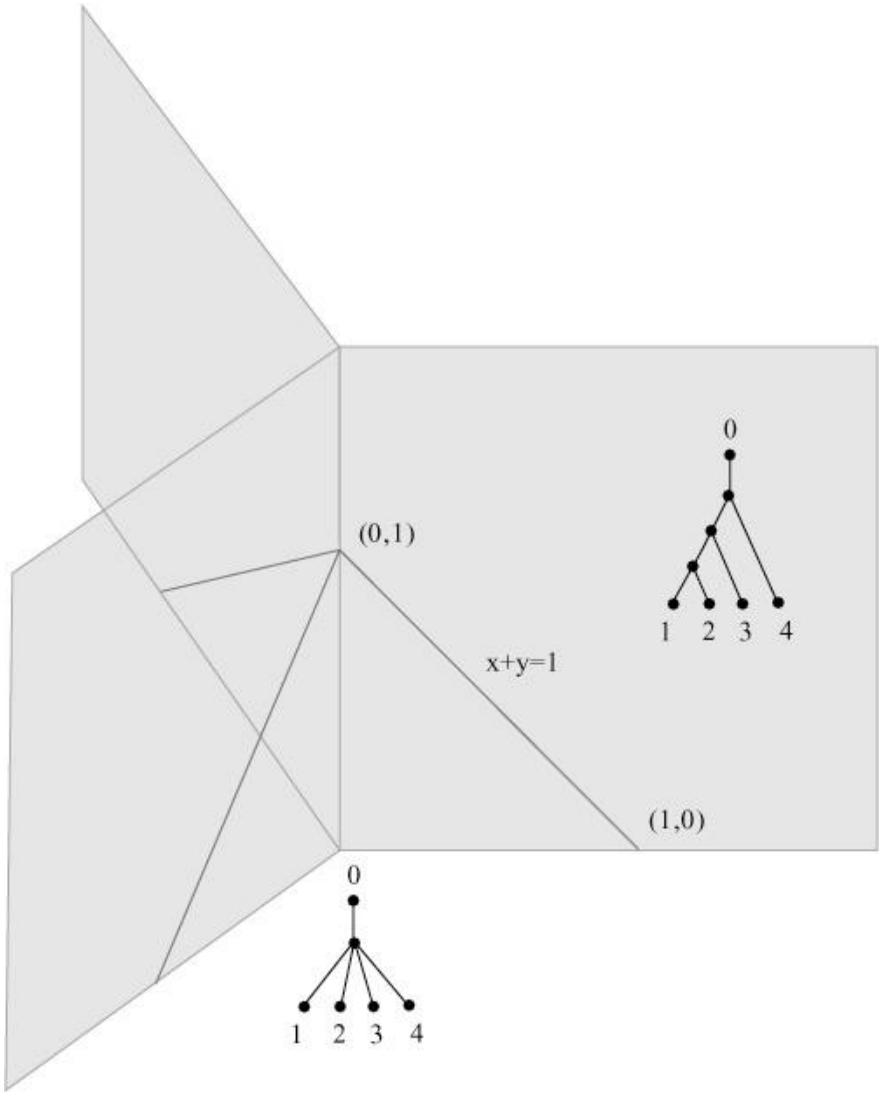
Arbres de 4 feuilles
et leur transformation

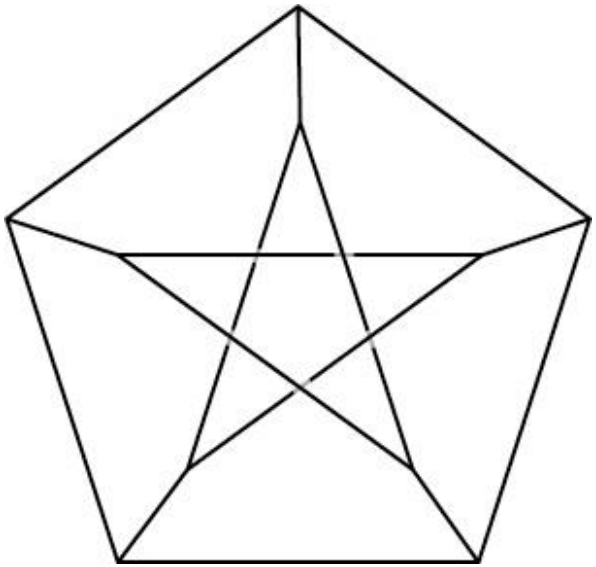
Arbres métriques: avec noeuds étiquettés




0
2 3 1 4
0
1 2 3 4
0
1 2 3 4
0
1 3 2 4
0
1 2 3 4


b
a
c
c
a
b

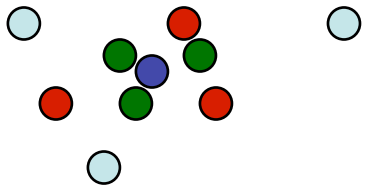




Graphe de Peterson

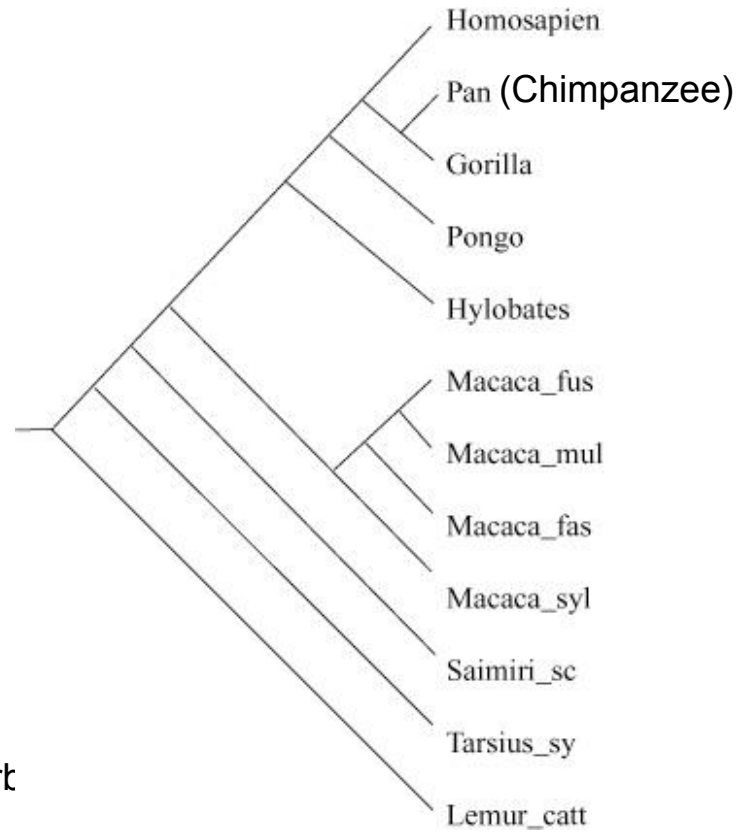
Propriétés de cet espace

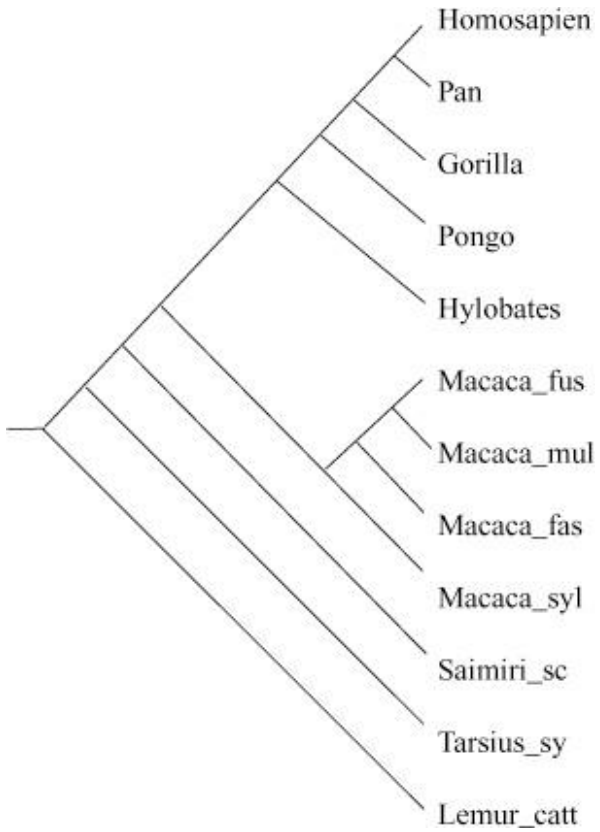
- Il s'agit d'un espace CAT(0), à curvature non-positive.
- Il y a toujours une géodesique entre chaque paire de points.
- Dans un espace CAT(0) les **centroides** existent pour chaque ensemble fini de points, et la fonction centroide est convexe.



```
'Lemur_catta'      AAGCTTCATAGGAGCAACCATTCTAATAATCGCACATGGCCTTACATCATCCA...
'Tarsius_syrichta'AAGTTTCATTGGAGCCACCACTCTTATAATTGCCCATGGCCTCACCTCCTCCC...
'Saimiri_sciureus'AAGCTTCACCGGCGCAATGATCCTAATAATCGCTCACGGGTTTACTTCGTCTA...
'Macaca_sylvanus'  AAGCTTCTCCGGTGCAACTATCCTTATAGTTGCCCATGGACTCACCTCTTCCA...
'Macaca_fascicul.'AAGCTTCTCCGGCGCAACCACCCTTATAATCGCCCACGGGCTCACCTCTTCCA...
'Macaca_mulatta'   AAGCTTTTCTGGCGCAACCATCCTCATGATTGCTCACGGACTCACCTCTTCCA...
'Macaca_fuscata'   AAGCTTTTCCGGCGCAACCATCCTTATGATCGCTCACGGACTCACCTCTTCCA...
'Hylobates'        AAGCTTTACAGGTGCAACCGTCCTCATAATCGCCCACGGACTAACCTCTTCCC...
'Pongo'            AAGCTTCACCGGCGCAACCACCCTCATGATTGCCCATGGACTCACATCCTCCC...
'Gorilla'          AAGCTTCACCGGCGCAGTTGTTCTTATAATTGCCCACGGACTTACATCATCAT...
'Pan'              AAGCTTCACCGGCGCAATTATCCTCATAATCGCCCACGGACTTACATCCTCAT...
'Homo_sapiens'     AAGCTTCACCGGCGCAGTCATTCTCATAATCGCCCACGGGCTTACATCCTCAT...
```

Exemple





Centroide de deux arbres phylogénétiques