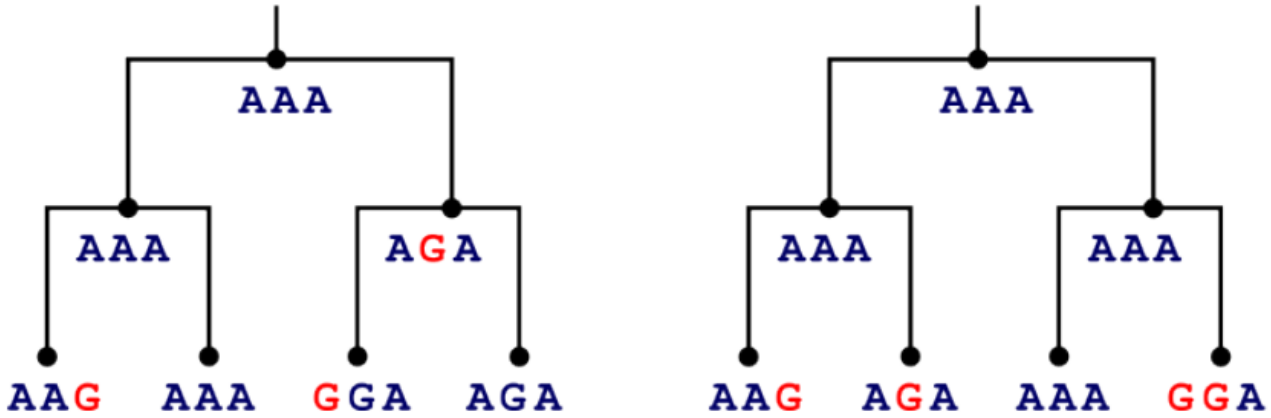
PARCIMONIE

Question 1 :
La méthode de parcimonie est basée sur l'analyse des données morphologiques.
Le postulat de base est que l'évolution est parcimonieuse, c'est a dire que , pour un groupe d'espèces, la phylogénie la plus vraisemblable est celle qui nécessite le plus petit nombre de changement évolutifs. L'arbre phylogénétique est conçu de manière à impliquer le minimum d'événements évolutifs.
Donc Ces méthodes recherchent parmi tous les arbres possibles, et toutes les séquences possibles aux nœuds internes, la combinaison qui minimise le nombre total de mutations requis pour expliquer les données observées.

Exemple :
Etant donnée la séquence **AAG, AAA, GGA, AGA,** on peut construire à partir de cette séquence plusieurs arbres pouvant expliquer leur phylogénie :



La méthode de parcimonie préfère l'arbre de gauche qui présente le moins d'événements de substitution.

Question 2 : Problème de la :

- **Petite parcimonie :** Trouver l'étiquetage le plus parcimonieux des nœuds internes d'un arbre enraciné
- **Parcimonie large :** Trouver le score de parcimonie le plus minimal sur tous les arbres possibles avec n feuilles

La parcimonie large est considérée comme plus compliquée à mesure que le nombre de feuilles augmente(problème NP-complet)

Question 3 :

SANKOFF

Etant donné la matrice de pénalité suivante :

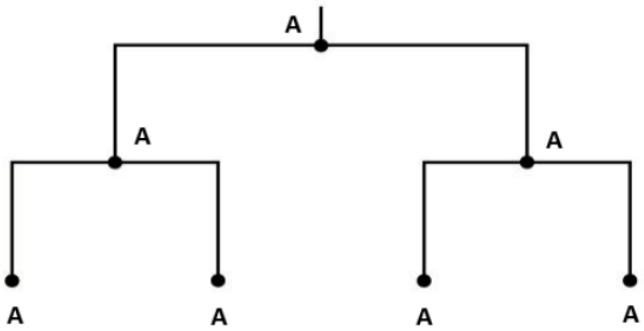
Sankoff algorithm

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

On parcourt l'arbre des feuilles à la racine pour trouver le cout minimum en utilisant cette matrice.

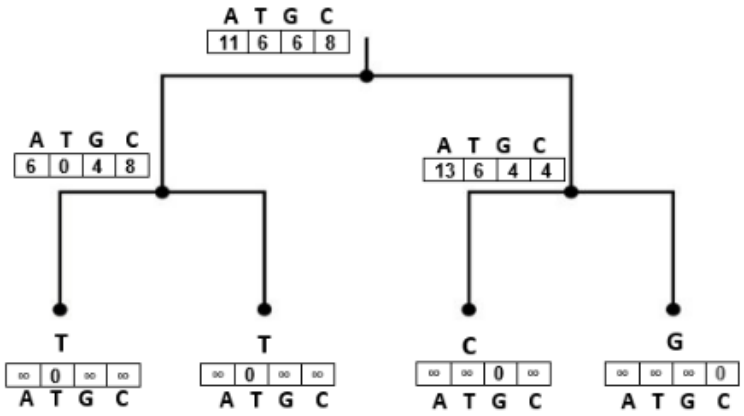
A=ATCCTG B=ATCCGG C=ACGGCC D=AGGGCA

Position 1: AAAA



A=ATCCTG B=ATCCGG C=ACGGCC D=AGGGCA

Position 2: TTCG



TT→ A: 3+3=6
TT→ T: 0+0=0
TT→ C: 2+2=4
TT→ G: 4+4=8

CG→ A: 4+9=13
CG→ T: 2+4=6
CG→ G: 4+0=4
CG→ C: 0+4=4

A→A: 0+6=6 A→A: 0+13=13

On additionne les deux minimums, on trouve **11**

T→A: 3+0=3 + T→A: 3+6=9

G→A: 4+4=8 **G→A: 4+4=8**

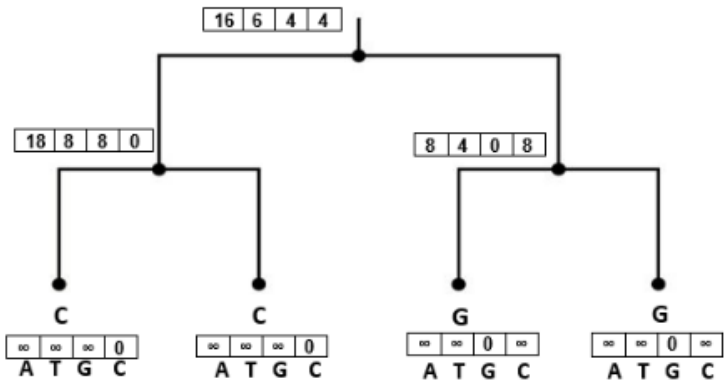
C→A: 9+8=17 C→A: 9+4=13

On fait la même chose pour les autres positions et on se retrouve avec ces valeurs **11 6 6 8**

On repete le meme principe pour obtenir les arbres suivants :

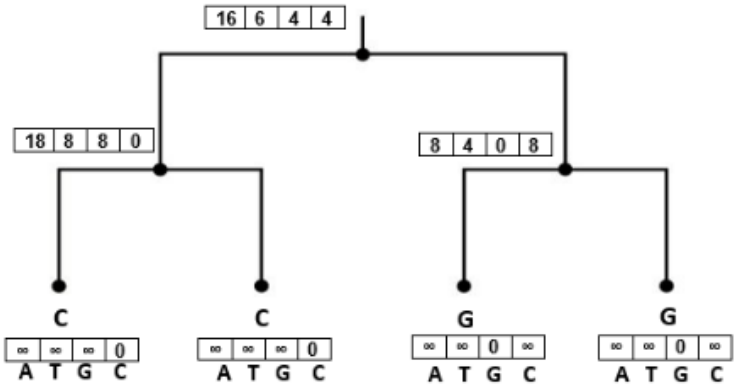
A=ATCCTG B=ATCCGG C=ACGGCC D=AGGGCA

Position 3: CCGG



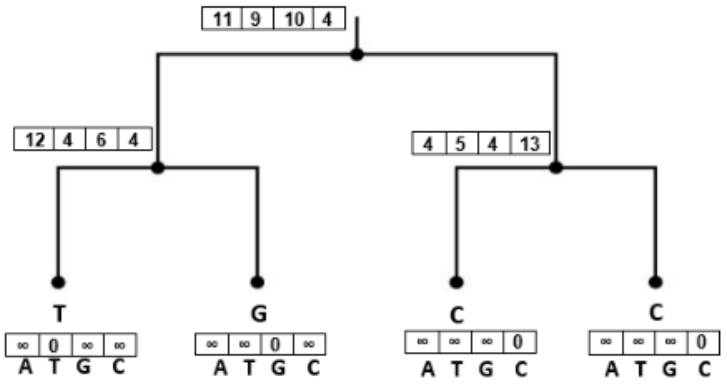
A=ATCCTG B=ATCCGG C=ACGGCC D=AGGGCA

Position 4 : CCGG



A=ATCCTG B=ATCCGG C=ACGGCC D=AGGGCA

Position 5 : TGCC



FITCH ;

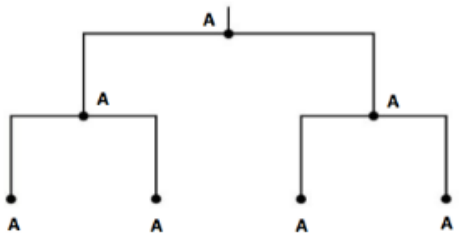
L'algo de Fitch se base sur le même principe que pour Sankoff. Il faut chercher le moins d'étapes possibles et le cout minimum pour obtenir l'arbre. En utilisant la matrice de pénalité suivante :

Fitch algorithm

	A	T	G	C
A	0	1	2	3
T	1	0	2	4
G	2	2	0	1
C	3	4	1	0

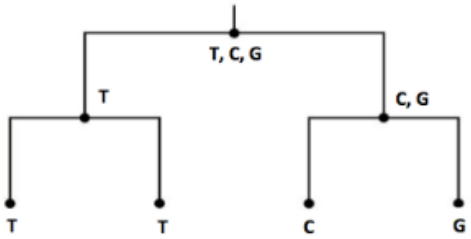
A=ATCCTG B=ATCCGG C=ACGGCC D=AGGGCA

Position 1: A, A, A, A



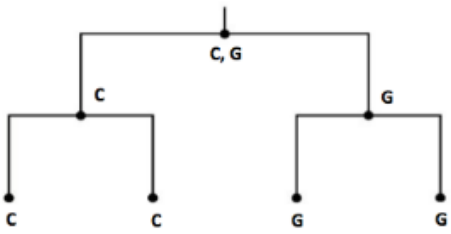
A=ATCCTG B=ATCCGG C=ACGGCC D=AGGGCA

Position 2: T, T, C, G



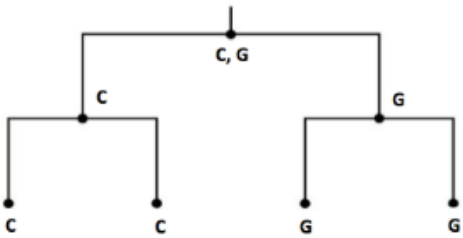
A=ATCCTG B=ATCCGG C=ACGGCC D=AGGGCA

Position 3: C, C, G, G



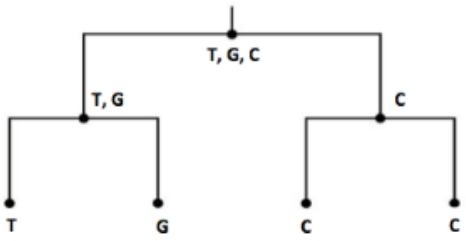
A=ATCCTG B=ATCCGG C=ACGGCC D=AGGGCA

Position 4: C, C, G, G



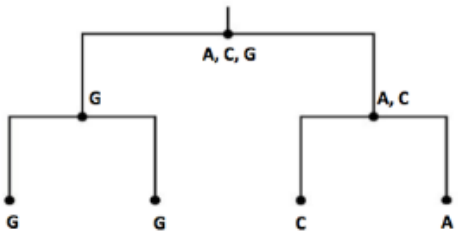
A=ATCCTG B=ATCCGG C=ACGGCC D=AGGGCA

Position 5: C, C, G, G

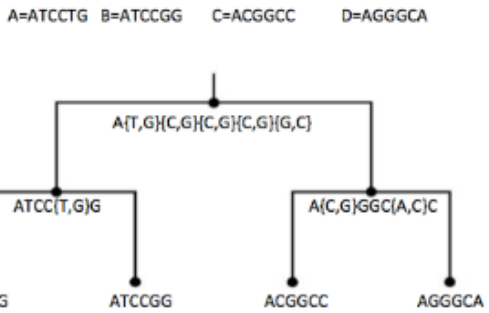


A=ATCCTG B=ATCCGG C=ACGGCC D=AGGGCA

Position 6: G, G, C, A



Et l'arbre final :



Question 4 :

l'agorithme **nearest neighbor interchange** est un algorithme de réarrangement des branches où on évalue seulement un sous-ensemble des arbres possibles et où on définit un voisinage dun arbre comme celui joignable par un échange du voisin le plus proche.

Alors, l'algorithme commence par un arbre choisi arbitrairement et teste les voisins. Si un voisin donne un score de parcimonie amélioré, alors on bouge à ce dernier. L'algorithme est dit heuristique parce que il fournit un résultat rapide mais y a pas de moyen pour s'assurer que ce résultat est optimal.