

Mini-Projet : Arbres Phylogénétiques

25 novembre 2024

Plan

- Administration
 - Informations générales/Point de départ
 - Groupes et soumissions
 - Portail de soumission
- Projet
 - But et algorithme à implémenter
 - Code fourni
 - Etapes du projet
 - Les structures de données
 - Étape 1 : fonctions utilitaires
 - Étape 2 : gestion des structures de données
 - Étape 3 : arbres phylogénétiques

Échéances

- Publication du matériel : Lundi 25 Novembre
- Séance TP du 26.11 entièrement dédiée au lancement du mini-projet
- Dernier mardi du semestre (17.12) dédié à la finalisation
- Ouverture du portail de soumission : **Lundi 2.12**
- **Délai de soumission : Mercredi, 18.12, 13h.**

Ressources pour le projet

- Toutes les informations et ressources sont sur Moodle
 - Archive Zip avec le matériel fourni et le fichier à compléter
 - Description (document .pdf) expliquant le matériel fourni et les tâches à réaliser
 - Annexe du .pdf : rappel de l'algorithme à implémenter
- Avant de commencer lisez la documentation fournie complètement et avec soin !



▼ Mini-projet (Programmation) ✎



GROUP CHOICE

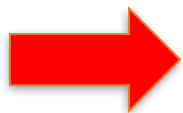
Inscription et choix du groupe ✎

À réaliser **entre le 26.11.2024 et le 18.12.2024**

Ce travail à réaliser en binôme compte pour 10% de la note du semestre.

Soumission

- **Deadline: Mercredi 18.12, 13:00**
- Groupes d'**au plus 2 étudiants**
- Les instructions pour la soumission seront sur Moodle



[Programmation] Etat des lie...

Ed Discussion remplace Plaz...

▼ **Mini-projet (Programmation)**

Inscription et choix du groupe

À réaliser entre le 26.11.202...

Précisions et correctifs (dern...

Transparents (présentation d...

Rediffusion de la présentatio...

▼ Mini-projet (Programmation) ✎



GROUP CHOICE

Inscription et choix du groupe ✎

À réaliser **entre le 26.11.2024 et le 18.12.2024**

Ce travail à réaliser en binôme compte pour 10% de la note du semestre.

Contenu de la soumission

- UN SEUL FICHIER
 - phylogenetics.cpp



Veillez à ne pas faire dépendre votre code de nouveaux ajouts dans les autres fichiers fournis !

Portail de soumission

- Sera ouvert le **2.12** su Moodle
 - **Pas de garantie de réponse rapide en cas de problèmes sur des soumissions faites le week-end**
- Fermeture le **18.12 (délai strict)**
- Vous pouvez soumettre autant de fois que vous voulez
- **Tâche 0:** Soumettez dès l'ouverture du portail même avec une version très incomplète. Le but est de vous familiariser avec le protocole de rendu et de **vérifier que votre code compile bien sur les machines de correction**

Soumission – Plagiat

- Le projet est noté
- Les discussions et échanges d'idée entre groupes est permis et recommandé
- L'échange de code est **strictement interdit!**
- Le plagiat sera **contrôlé** et considéré comme **tentative de tricherie.**
- En guise de sanction la note "NA" sera attribuée:
Art. 18 "**Fraude de l'ordonnance sur la discipline**"
<https://www.admin.ch/opc/fr/classified-compilation/20041650/index.html>
- Vous devez en tout temps être capable d'expliquer le code que vous avez produit.

Notation

Le projet comporte 3 parties obligatoire:

- Étape 1 : Fonctions utilitaires (BFS) : **30 points**
 - Étape 2 : Gestion des structures de données : **35 points**
 - Étape 3 : Arbres phylogénétiques : **35 points**
-
- Fixez vous des **objectifs raisonnables** en fonction de votre niveau!

Aide pendant les TPs/ en semaine

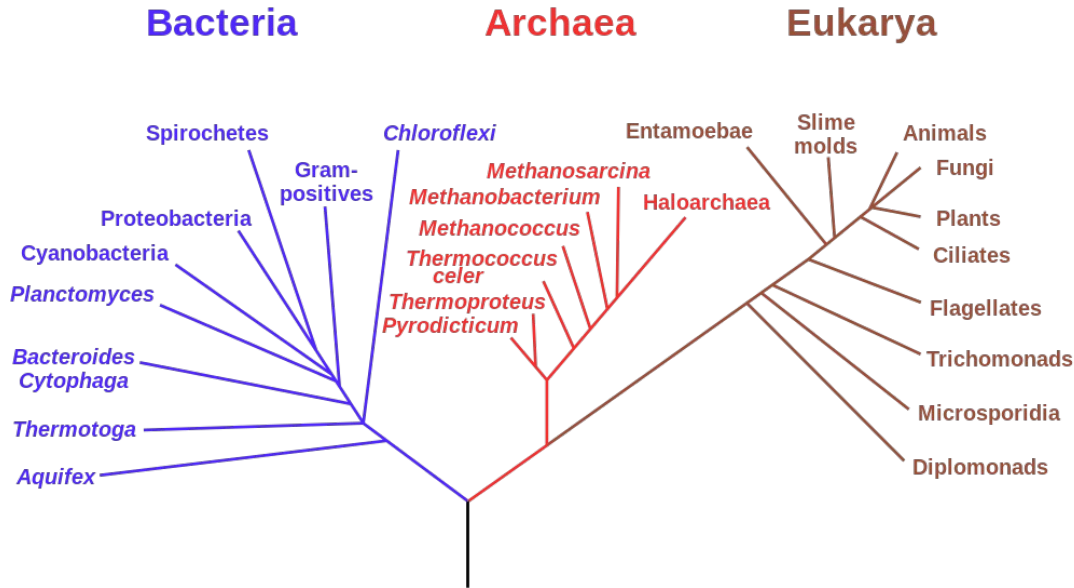
- Les assistants n'ont pas pour tâche de résoudre vos erreurs d'exécution
- Ils peuvent vous aider sur les fautes de compilation
- Et vous aider à développer des stratégies pour débuser les sources des erreurs par vous même
- Il y a une **équipe très limitée en appui: privilégiez le forum Ed pendant la semaine**

Plan

- Administration
 - Informations générales/Point de départ
 - Groupes et soumissions
 - Portail de soumission
- **Projet**
 - But et algorithme à implémenter
 - Code fourni
 - Etapes du projet
 - Les structures de données
 - Étape 1 : Fonctions utilitaires
 - Étape 2 : Gestion des structures de données
 - Étape 3 : Arbres phylogénétiques

But du projet

- Un arbre phylogénétique montre la proximité évolutive entre différentes entités



- Deux algorithmes connus:
 - **WPGMA** (Weighted Pair Group method with arithmetic mean)
 - **UPGMA** (Unweighted Pair Group method with arithmetic mean)

WPGMA : exemple introductif

Supposons que nous souhaitons analyser la proximité génétique de quatre entités au travers d'un fragment d'ADN particulier:

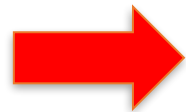
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
T1	C	A	T	A	G	A	C	C	T	G	A	C	G	C	C	A	G	C	T	C
T2	C	A	T	A	G	A	C	C	C	G	C	C	A	T	G	A	G	C	T	C
T3	C	G	T	A	G	A	C	T	G	G	G	C	G	C	C	A	G	C	T	C
T4	C	C	T	A	G	A	C	G	T	C	G	C	G	G	C	A	G	T	C	C

Les fragments d'ADN qui servent de base à la comparaison sont nommés **«taxons»** : T1, T2, T3 et T4

WPGMA : matrice de distance

On commence par quantifier la différence entre les taxons, par exemple en comptant les différences:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
T1	C	A	T	A	G	A	C	C	T	G	A	C	G	C	C	A	G	C	T	C
T2	C	A	T	A	G	A	C	C	C	G	C	C	A	T	G	A	G	C	T	C
T3	C	G	T	A	G	A	C	T	G	G	G	C	G	C	C	A	G	C	T	C
T4	C	C	T	A	G	A	C	G	T	C	G	C	G	G	C	A	G	T	C	C



	T1	T2	T3	T4
T1	0	5	4	7
T2	5	0	7	10
T3	4	7	0	7
T4	7	10	7	0

Matrice de distance

WPGMA : algorithme

- Au départ chaque taxon est dans son propre «cluster»/aggrégat
- À chaque étape, deux clusters ayant une **distance minimale** entre eux sont fusionnés
- La matrice de distance reflète cette fusion : la distance entre le cluster $T_i T_j$ et T_k est la moyenne arithmétique de la distance entre T_i et T_k et de la distance entre T_j et T_k
- L'arbre phylogénétique est construit en **regroupant itérativement les groupes les plus proches** en terme de distance et en répartissant uniformément la distance entre les deux branches

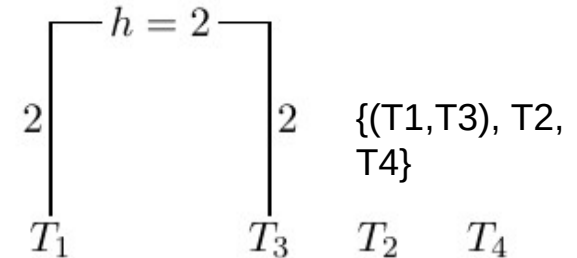
{T1, T2, T3, T4}



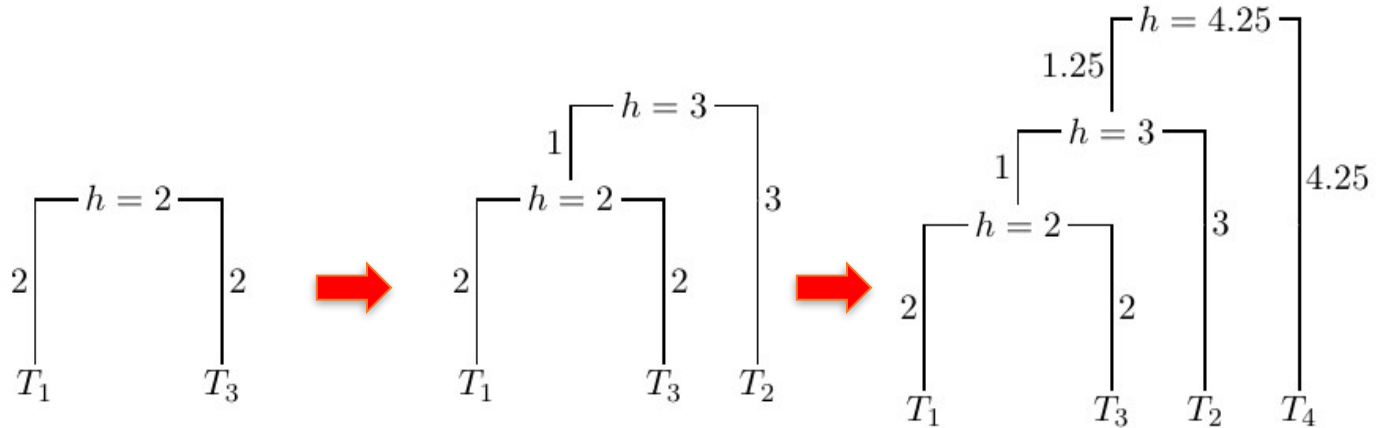
	T1	T2	T3	T4
T1	0	5	4	7
T2	5	0	7	10
T3	4	7	0	7
T4	7	10	7	0




	T1T3	T2	T4
T1T3	0	6	7
T2	6	0	10
T4	7	10	0



WPGMA : construction itérative de l'arbre



Note: UPGMA est simplement une variante de l'algorithme qui répartit la distance de façon pondérée sur les branches de l'arbre.

Plan

- Administration
 - Informations générales/Point de départ
 - Groupes et soumissions
 - Portail de soumission
- **Projet**
 - But et algorithme à implémenter
 - Code fourni
 - Etapes du projet
 - Les structures de données
 - Étape 1 : Fonctions utilitaires
 - Étape 2 : Gestion des structures de données
 - Étape 3 : Arbres phylogénétiques

Fichiers fournis

- phylogenetic.zip contient :
 - **Cmakelists.txt**: fichier pour compiler avec QtCreator
 - **Makefile.geany**: fichier pour compiler en ligne de commande ou avec Geany
 - Fichiers .hpp or .cpp (voir transparents suivants)
- Vos contributions seront dans un **seul fichier** mais vous travaillerez dans un environnement “Projet”
 - Plusieurs fichiers sont impliqués lors de la compilation
 - Conformez vous aux directives d’installation décrites dans le descriptif général du projet.



Code fourni (1/2)



Ne pas modifier !

- **test_utils.hpp, test_utils.cpp** : utilitaires de comparaison des données
- **utils.hpp, utils.cpp** :
 - **double double_to_string (double)** : convertit en **double** en **string** en limitant le nombre de décimales
 - **vector<Taxon>readSequencesFromFile(const std:string& filename)** : lit ensemble de taxons à partir d'un fichier

Code fourni (2/2)

- **phylogenetics.hpp** :
 - Contient les **types** et les **prototypes** des fonctions à coder
Ne pas modifier !
- **phylogenetics.cpp** : **C'est ici que vous codez ! Ce fichier est noté !**
 - contient les corps vides des fonctions à coder
 - si nécessaire vous pouvez ajouter vos propres types et prototypes
- **main.cpp** : **Fichier non noté !**
 - tests fournis pour commencer à tester votre code.
 - **Ces tests ne sont pas exhaustifs, vous devez les compléter.**

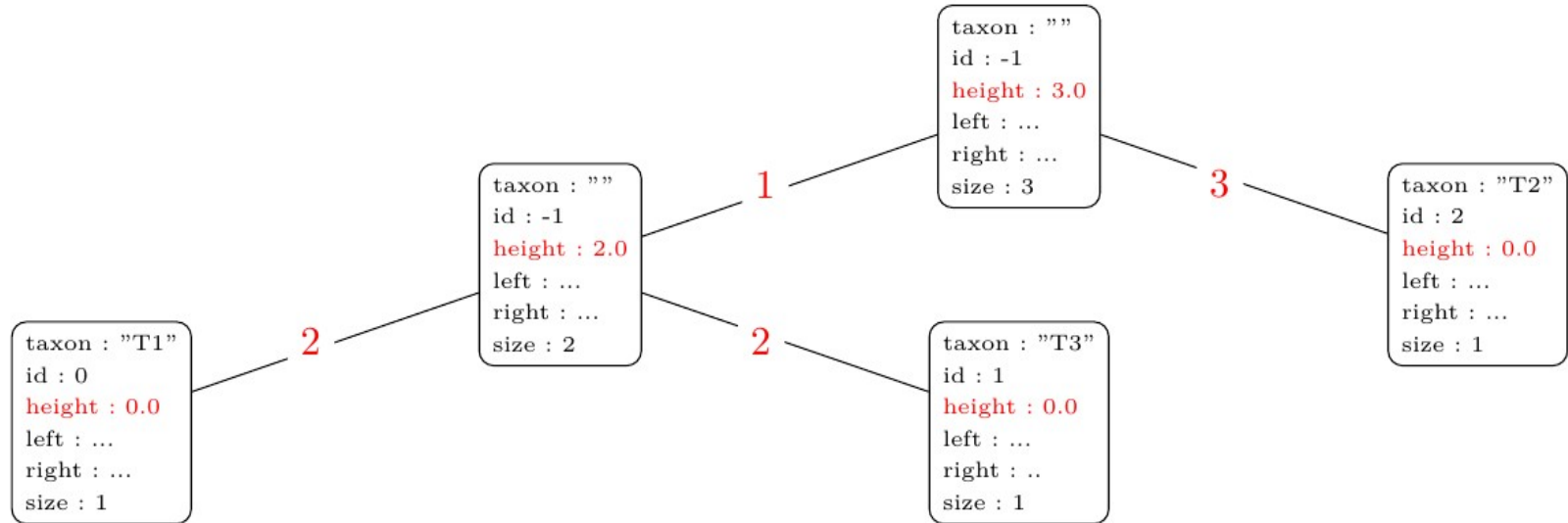
Plan

- Administration
 - Informations générales/Point de départ
 - Groupes et soumissions
 - Portail de soumission
- Projet
 - But et algorithme à implémenter
 - Code fourni
 - Etapes du projet
 - **Les structures de données**
 - Étape 1 : Fonctions utilitaires
 - Étape 2 : Gestion des structures de données
 - Étape 3 : Arbres phylogénétiques

Structures de données (1/2)

- **Taxon** : intitulé d'un taxon, par exemple "ACGT".
- **DistanceMatrix** : matrice de distances
- **Cluster**: agrégat de taxons (noeud d'un arbre phylogénétique); chaque cluster a un identifiant entier
- **ClusterIdPair** : paire d'identificateurs de cluster
- **Tree** : vector de clusters
- **AlgoType** : type énuméré pour désigner l'algorithme à exécuter

Structures de données (2/2)



Étape 1 : fonctions utilitaires

L'objectif de cette tâche est de coder différentes méthodes permettant de produire la **représentation sous forme de chaîne de caractères** du contenu des structures de données



Utile pour l'affichage et le debugging

Méthodes **toString** (ou équivalent) à implémenter pour:

- **DistanceMatrix**
- **Cluster**
- **ClusterIdPair**
- **Tree**

Étape 2 : gestion des structures de données

Initialisation et manipulation des structures de données **Tree** et **DistanceMatrix**

- Méthodes à implémenter:
 - **calculateDistance(. .)**: permet de calculer la distance entre une paire de taxons
 - **initTree(. .)**: initialise un arbre phylogénétique à partir d'un ensemble de taxons
 - **DistanceMatrix (..)**: initialise matrice de distance à partir d'un arbre phylogénétique constitués uniquement de feuilles (avant les regroupements)
 - **eraseColumn(...)/eraseRow(...)**: supprime une colonne ou ligne dans une matrice de distances
 - **void deleteTree(. .)**: libère la mémoire associée à un arbre

Étape 3 : arbres phylogénétiques

- But : construire des arbres phylogénétiques au moyen des algorithmes **WPGMA** et **UPGMA** et les afficher.
- Méthodes à implémenter:
 - **ClusterIdPair minimumDistance(...)** : trouve la paire de clusters séparés par la plus petite distance (deux clusters les plus proches)
 - **void mergeClusters(...)**: permet de fusionner deux clusters dans un arbre donné, ce qui implique de mettre à jour aussi la matrice de distance
 - **void buildPhlyogeneticTree(...)**: construit un arbre phylogénétique selon un algorithme donné
 - **string phylogeneticTreeToString(...)**: produit une représentation sous forme de chaîne de caractères d'un arbre phylogénétique (ce qui permet de l'afficher)

Conseils

- Lisez et comprenez le but de chaque tâche complètement avant de vous lancer dans le codage
- Lisez et comprenez les tests de **main.cpp** pour bien comprendre ce qui est attendu
- N'hésitez pas à commenter les appels à certaines fonctions dans les tests fournis pour tester les étapes encore incomplètes
- Soumettez régulièrement sur le portail de soumission dès son ouverture
- Utilisez le debugger (à combiner avec des instructions d'affichage)
- Sauvegardez régulièrement votre travail
- N'hésitez pas à enrichir les tests fournis dans **main.cpp**

Bon Codage!

Questions?