AAGB - TME 3 - Algorithme de Sankoff

December 4, 2020

1 Problème : Phylogénie Pokémon

On voudrait étudier une branche particulière de l'arbre phylogénétique des Pokémon. Pour chaque espèce étudiée, on récupère une position bien conservée dans un gène de leur ADN. Voici les différentes espèces considérées, ainsi que le nucléotide associé :

Probopass : A Aggron : T Bastiodon : T Regirock : G Registeel : G Regice : G Klingklang : G Metagross : C Genesect : A Porygon−Z : C Magnezone : C Forretress : T Eletrode : A Ferrothorn : G

Deux groupes de scientifiques proposent des arbres phylogénétiques différents pour expliquer l'évolution des pokémon. On a récupéré la structure des deux arbres sous le format Newick :

N1 = "((((Electrode , Magnezone) ,Porygon−Z) , ((((Aggron , Bastiodon) , Forretress) , Ferrothorn) , (((((Regirock , Regice) , Registeel) , Metagross) , Klingklang) , Genesect))) , Probopass);" N2 = "(((((Regirock , Regice) , Registeel) , ((Metagross , Klingklang) , Genesect)) , (((Aggron , Bastiodon) ,(Forretress , Ferrothorn)) , Probopass)) ,(Porygon−Z,(Magnezone , Electrode)));"

L'objectif est de déterminer l'arbre le plus probable, en utilisant l'algorithme de Sankoff pour trouver lequel a un score de parcimonie minimal.

2 Algorithme de Sankoff

Une méthode efficace serait d'utiliser une structure d'arbre, disponible par exemple avec le package ete3, et de parcourir le graphe à partir des feuilles pour calculer les scores de parcimonie, et à partir de la racine pour le traceback. On voudrait ici ne pas utiliser de package particulier, donc on propose un guide pour le développement de l'algorithme de Sankoff sans y avoir recours.

2.1 Visualisation des arbres

Visualiser les deux arbres proposés, par exemple grâce au site suivant : etetoolkit treeview.

2.2 Calcul des scores pour les noeuds internes

Une stratégie peut être, dans un premier temps, de créer un dictionnaire faisant correspondre le nom des noeuds à leur étiquetage. On peut par exemple créer une fonction d'initialisation d'un tel dictionnaire par toutes les feuilles, que l'on mettre à jour à chaque nouveau noeud interne. Coder ensuite une fonction permettant, étant donnés deux noeuds avec leur vecteur de score associé, de retourner leur ancêtre commun et son score de parcimonie.

On peut ensuite parser la chaîne de caractères correspondant au format Newick pour progressivement clusteriser chaque couple de feuilles/noeuds interne.

2.3 Traceback

On peut construire le traceback en parallèle, en stockant au fur et à mesure dans une liste tous les couples de noeuds que l'on a clusterisés, ainsi que les argmin qui ont permis d'obtenir les différents scores de parcimonie. Il suffit alors de parcourir la liste dans le sens inverse pour partir de la racine et redescendre progressivement dans l'arbre. L'idée est d'avoir en sortie du programme une chaîne de caractère avec l'étiquetage des noeuds internes, ainsi que le score de parcimonie de l'arbre.

2.4 Verdict

Quelle est l'arbre donnant le score de parcimonie minimal ? (Voir l'image pour l'arbre phylogénétique complet des Pokémon...)