– TD5 OCAML –

Durant la dernière décennie les progrès de la numérisation des génomes des organismes vivant ouvre de nouvelles voies algorithmiques sur leur analyse. Un génome peut être considérée comme un assemblage de *séquences génomiques*, où chaque séquence correspond au code génétique d'une protéine. Une séquence est elle-même un assemblage de 4 *bases* d'ADN: Adénine, Thyrosine, Guanine, Cytosine. Une séquence sera codée en informatique par une liste de symboles des bases A,T,G,C.

Pour faire face au challenge du très grand nombre de génomes séquencés, il est nécessaire de trouver un encodage performant permettant un accès rapide aux informations relatives aux séquences. Le dictionnaire est un moyen efficace de conserver un grand ensemble de séquences en ayant des temps d'accès courts aux informations.

Un dictionnaire est un arbre où chaque nœud possède 4 fils correspondant respectivement aux bases. Chaque feuille contient une chaine de caractères symbolisant des informations relatives à la séquence qui est toujours distincte de la chaine vide "", sauf pour le dictionnaire vide.

Une séquence correspondra à un chemin dans cet arbre jusqu'à une feuille tel que chaque nœud décrit une base. Pour chaque nœud, dans l'ordre, le premier fils correspond à A, le second à T, le troisième à G et le dernier à C. Ainsi, pour coder une séquence débutant par A, on choisit le premier fils et le codage du reste de la séquence se poursuivra dans son sous-arbre.

La figure suivante décrit l'intégration de 7 séquences dans un dictionnaire. On ne représente ici que les sous-arbres utiles pour coder les séquences, les autres sont des sous-arbres/dictionnaires vides. Les noms des séquences décrivent ici l'information associée à chacune d'elles.

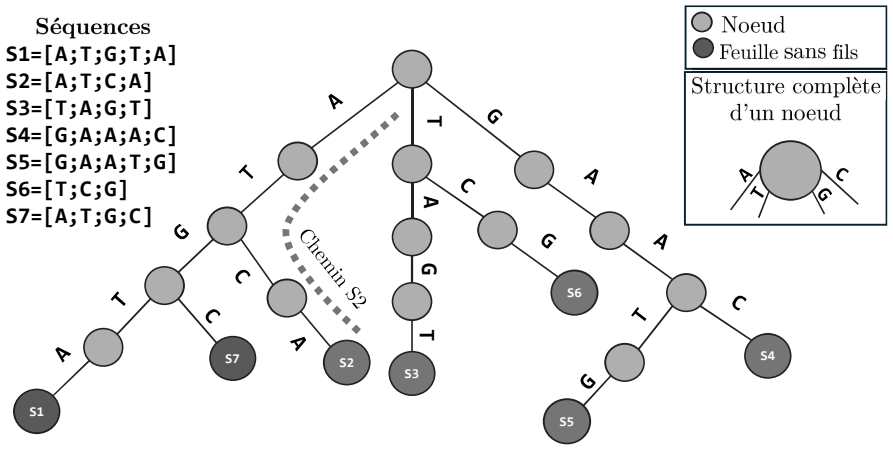

Figure 1: Encodage de 7 séquences génomiques

Enfin, aucune séquence n'est le préfixe d'une autre. Ainsi, les informations relatives aux séquences sont dans les feuilles uniquement.

1. Déclarer le type `adn_t` correspondant aux bases d'ADN. Décrire la séquence $AATGCCT$ sous forme d'une liste.

2. Déclarer le type `adndico_t` codant le dictionnaire. Comment représente-t-on le dictionnaire vide ?

3. Faites la fonction `height` donnant la hauteur de l'arbre codant le dictionnaire. Pour l'exemple cette hauteur est de 5 (cf. s_1).

4. Faites une fonction `dicreate seq data` qui crée un dictionnaire ne contenant que la séquence `seq` et qui insère l'information contenue dans `data` en sa feuille distincte du dcitionnaire vide.

5. Faites la fonction `insert seq dico data` qui insère une séquence dans un dictionnaire en respectant les règles du dictionnaire. `data` représente l'information qui est relative à la séquence à insérer dans la feuille concluant l'insertion. Par exemple, `insert [T;C;G] dico "s6"` insérera la séquence s_6 dans le dictionnaire de la figure.

6. Exécuter le code fourni sous E-campus (`td5.ml`) et tester.

7. Faites une fonction `countseq dico` comptant le nombre de séquences encodées dans un dictionnaire (`dico`). Pour la figure le nombre retournée sera 7.

8. Faites la fonction `search seq dico` qui recherche l'information associée à une séquence. Si la séquence ne se trouve pas dans le dictionnaire, la chaine vide `""` sera retournée. Par exemple, avec le dictionnaire de la figure, `search [A;T;C;A] dico` retourne `"s2"` et `search [A;T;C] dico` rend `""`.

9. Quelle est la complexité de l'algorithme de recherche. La complexité de la recherche d'une information sur une séquence rend ce codage très performant.

10. Faites la fonction `sameprefix` qui retorune les informations sur les séquences ayant le même préfixe donné en paramètre. Pour l'exemple de la figure `sameprefix d [A;T]` retourne la liste `["s1";"s2";"s7"]`.

11. Faites la fonction `dico2seqs` qui convertit un dictionnaire en liste de séquences.

12. Faites la fonction `delete dico seq` qui retire une séquence du dictionnaire.

13. Quelle solution à adopter pour admettre qu'une séquence soit le préfixe d'une autre.