FT2125 : Introduction à l'algorithmique Devoir 2, session hiver 2014

À remettre le mardi 8 avril, avant 19h30 (au cours)

1. (10 points) Soit G = (V, E) un graphe non-orienté, connexe, où chaque arête a un poids non négatif. Voici une stratégie diviser-pour-régner pour le problème de trouver un arbre couvrant minimal pour ce graphe :

On partitionne l'ensemble des sommets V en deux ensembles V_1 et V_2 tels que leur cardinalité diffère d'au plus 1. Soit E_1 l'ensemble des arêtes qui ne sont incidentes qu'à des sommets de V_1 et soit E_2 l'ensemble des arêtes qui ne sont incidentes qu'à des sommets de V_2 . On résout alors récursivement le problème de recherche d'un arbre couvrant minimal pour chacun des sous-graphes $G_1 = (V_1, E_1)$ et $G_2 = (V_2, E_2)$. Finalement, on sélectionne l'arête de poids minimal de E entre un sommet de V_1 et un sommet de V_2 et l'on utilise cette arête pour obtenir l'arbre couvrant minimal pour G.

Prouvez que cette algorithme calcule bien un arbre couvrant minimal pour G ou donnez un contre-exemple.

- 2. (30 points) Étant donnée une séquence $X = x_1 \dots x_n$, une séquence $Z = z_1 \dots z_k$ est une sous-séquence de X s'il existe une séquence strictement croissante d'indices i_1, \dots, i_k de X telle que $x_{i_j} = z_j, \forall j, 1 \leq j \leq k$. Par exemple, Z = BCBD est une sous-séquence de X = ABCABADB correspondant à la séquence d'indices (2,3,5,7).
 - Étant données deux séquences $X = x_1 \dots x_n$ et $Y = y_1 \dots y_m$, on dit qu'une séquence Z est une sous-séquence commune de X et Y si Z est une sous-séquence de X et une sous-séquence de Y. Par exemple, si X = ACGTTGTA et Y = ACCCCTTTTA, Z = ACTTTA est un sous-séquence commune à X et Y et en fait, Z = ACTTTA est la plus longue sous-séquence commune de X et Y.
 - a) (10pts) Donnez un algorithme de programmation dynamique qui calcule la plus longue sous-séquence commune entre deux séquences X et Y. Quelle est la complexité en temps de votre algorithme?
 - b) (5pts) Construisez la table de programmation dynamique de votre algorithme en a) pour les séquences X = ACGTTGTA et Y = ACCCCTTTTA.
 - c) (10pts) Un palindrome est une chaîne de caractères non vide qui se lit de la même façon de gauche à droite et de droite à gauche (par exemple, eibohphobie (phobie des palindromes), radar, ressasser, etc.). Donnez un algorithme de programmation dynamique efficace qui retourne le plus long palindrome qui est une sous-séquence d'un mot X donné en entrée. Quelle est la complexité en temps de votre algorithme?
 - d) (5pts) Construisez la table de programmation dynamique de votre algorithme en c) pour la séquence X = ACTCCTCGA.

- 3. (10 points) Soient S et T, deux séquences quelconques sur un alphabet fini Σ de taille |S| = n et |T| = m. Combien d'alignements possibles existe-il entre S et T si les opérations considérées sont seulement les identités (match), remplacements (mismatch) et suppressions (i.e. on ne considère pas les insertions). (Indice : L'alignement résultant aura des "-" seulement dans l'une des deux séquences.)
- 4. (10 points) Soient P (de longueur m) un motif et T (de longueur n), m << n un génome circulaire (voir Figure 1). Décrivez un algorithme en $\mathcal{O}(n^2m)$ permettant d'obtenir toutes les positions des occurrences approximatives (ayant au plus k erreurs) de P dans T.

Par exemple, si P = AAGT, T = GATAAGTAA et k = 1, il y a une occurrence exacte de P commençant à la position 4 et se terminant à la position 8 du texte et une occurrence ayant une erreur commençant à la position 8 du texte et se terminant à la position 3 (occurrence circulaire).

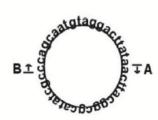


Figure 1 – Deux séquences (A et B) apparemment différentes mais qui proviennent du même génome et débutent à des positions différentes. La région foncée montre que forcément, le préfixe d'une séquence devient le suffixe de l'autre (et vice versa).

A. acttacggcgcatatcgcccagcaatgtaggacttata

 $B.\ ccag caatg taggact tata {\color{blue}acttacg} {\color{blue}c} {\color{bulue}c} {\color{blue}c} {\color{blue$

- 5. (20 points) Soit A[1..n] un tableau de n entiers distincts. Si i < j et A[i] > A[j], on dit que le couple (i, j) est une **inversion** de A.
 - a) (5pts) Quel est le maximum d'inversions que peut avoir un tableau A de taille n? Justifiez votre réponse.
 - b) (5pts) Quelle est la relation entre le temps d'exécution du tri par insertion et le nombre d'inversions du tableau en entrée? Justifiez votre réponse.
 - c) (10pts) Donnez un algorithme, dont la complexité en temps dans le pire cas est de $\Theta(n \log n)$, qui détermine le nombre d'inversions présentes dans une permutation quelconque de n éléments. (Indice : modifier tri-fusion)
- 6. (20 points) Soit L une liste de n éléments rangés dans un tableau d'indices 1 à n. On suppose que la seule opération qu'on sache effectuer sur les éléments est de vérifier si deux éléments sont égaux ou non. On dit qu'un élément $x \in L$ est dominant si l'ensemble $Egaux_x = \{y \in L \mid x = y\}$ a strictement plus que $\frac{n}{2}$ éléments. On suppose ici que n est une puissance de 2.

- a) (10pts) Donnez un algorithme naïf calculant le nombre d'éléments de l'ensemble $Egaux_x$ pour un x donné. En déduire un algorithme pour vérifier si L possède un élément dominant. Quelle est la complexité en temps dans le pire cas de votre algorithme?
- b) (10pts) Donnez un autre algorithme diviser-pour-régner pour ce problème. Quelle est la complexité en temps dans le pire cas de votre algorithme?