

Examen de Bio-informatique

Vous devez rendre chaque exercice sur une feuille différente. Le barème est donné à titre indicatif, pour vous permettre de juger de la difficulté des questions.

1 Utilisation de BLAST

Question 1 (2 points). On considère les deux séquences d'ADN ATTCATTCATTCATTCATTCATTCATTCATTC et ATTGATTGATTGATTGATTGATTGATTGATTG. Quel est, à première vue, leur pourcentage d'identité ? Quand on fait un alignement avec l'algorithme de BLAST, aucune similarité n'est trouvée. Pourquoi ?

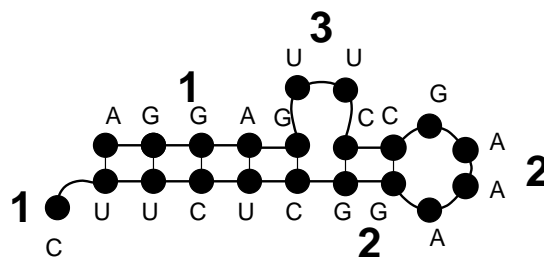
Question 2 (2 points). Pourquoi Blast n'est-il pas le bon outil pour fournir une liste exhaustive des occurrences potentielles d'un site de fixation de facteur de transcription de plus de 15 nucléotides ? Pour obtenir quelques occurrences, quels paramètres faut-il modifier ? Vers quelles valeurs ? Pourquoi ?

2 Comparaison de structures d'ARN

Une structure d'ARN est constituée par un ensemble de tiges (régions appariées) et de boucles (régions non appariées, libres), qui s'organisent en un arbre ordonné. Pour simplifier la notation, on attribue à chaque tige et à chaque boucle un numéro. L'ordre n'a pas d'importance. Une structure d'ARN peut ainsi être décrite à partir de trois constructeurs

- `boucle` qui prend un argument un numéro de boucle,
- `tige` qui prend deux arguments i et s : i est un numéro de tige et s est la sous-structure issue de i
- un opérateur binaire de concaténation `o` qui prend en premier argument une boucle ou une tige, et en second argument une sous-structure. L'ordre de concaténation correspond à l'ordre dans l'arbre (ou la structure).

Par exemple, la structure ci-dessous est représentée par `boucle(1)otige(1,tige(2,boucle(2))oboucle(3))`.



On souhaiterait pouvoir comparer des structures d'ARN, en calculant leur similarité. Les opérations d'édition autorisées sont

- substituer une tige en une autre tige
- substituer une feuille en une autre feuille
- supprimer une tige
- insérer une tige

- supprimer une feuille
- insérer une feuille

Dans ce modèle, il est impossible de substituer une tige en une boucle, et vice-versa. Comme pour l'alignement de séquences, la suite des opération d'édition **ne doit pas définir de croisements au niveau de la séquence**.

Question 3 (1 points). On considère les deux structures suivantes :

$$\begin{aligned} A &= \text{tige}(1, \text{tige}(2, \text{boucle}(1)) \circ \text{tige}(3, \text{boucle}(2))) \\ B &= \text{boucle}(1) \circ \text{tige}(1, \text{boucle}(4)) \circ \text{boucle}(2) \circ \text{tige}(2, \text{boucle}(5)) \circ \text{boucle}(3) \end{aligned}$$

Représenter les graphiquement.

Question 4 (2 points). Dans l'exemple précédent, si la tige 2 de A est substituée avec la tige 1 de B , la boucle 1 de A peut-elle être substituée avec la boucle 5 de B ? Pourquoi ?

Question 5 (5 points). Donnez les formules de récurrence pour calculer la similarité entre deux structures. Veillez à bien préciser les cas de base et à tenir compte de l'incompatibilité entre une boucle et une tige. On note Del , Ins et sub les coûts associés aux différentes opérations d'édition.

Question 6 (2 points). Avez-vous des suggestions pour définir le score pour une substitution de deux boucles, le score pour une substitution de deux tiges ?

3 Réarrangements

Nous nous intéressons dans cet exercice aux transpositions sur des permutations (les permutations seront *non signées* ou plutôt tous les éléments seront signés positivement). On ne connaît pas aujourd'hui d'algorithme exact pour le tri par transpositions. Le but de l'exercice est d'écrire un algorithme approximatif avec un ratio de 2.

Une *permutation* est une application de $\{1, \dots, n\}$ dans $\{1, \dots, n\}$. Une *transposition* $\rho(i, j, k)$ est une application qui transforme une permutation

$$l_1, l_2, \dots, l_{i-1}, l_i, \dots, l_{j-1}, l_j, \dots, l_{k-1}, l_k, \dots, n$$

en

$$l_1, l_2, \dots, l_{i-1}, l_j, \dots, l_{k-1}, l_i, \dots, l_{j-1}, l_k, \dots, n$$

(avec l_k appartenant à $\{1, \dots, n\}$ et $l_p \neq l_q$ pour tout $p \neq q$).

Le but d'un tri par transpositions d'une permutation est de trouver une suite la plus courte possible de transpositions permettant de transformer une permutation $\alpha = l_1, \dots, l_n$ en la permutation $\beta = 1, \dots, n$. Le nombre de ces transpositions est noté $d(\alpha)$.

On considérera dans la suite les permutations étendues : permutations auxquelles on ajoute les deux labels 0 et $n + 1$ comme dans le cas des inversions.

Question 7 (1 points). Trouvez un scénario de longueur minimale de tri par transposition pour la permutation 1 4 5 3 2.

Comme dans le cas des permutations signées, on introduit le concept de *point de cassure* (ou *breakpoint*). Un point de cassure est une paire de labels adjacents dans α mais pas dans β . Le nombre de points de cassure de α est noté $b(\alpha)$.

Question 8 (3 points). Combien au maximum peut-on enlever de point de cassures ? Expliquez. Donnez une borne inférieure pour $d(\alpha)$ en fonction de $b(\alpha)$.

Comme dans le cas des inversions, la permutation identité est celle qui maximise le nombre de cycles et on va donc chercher à maximiser ce nombre de cycles pour trier.

Question 9 (1 point). Représentez sur le diagramme RD suivant les arcs désirs et les arcs réalité (ceux que l'on peut dessiner !). Les \circ représentent les bornes gauche et droite d'un élément.

$$0 \circ \dots \circ i-1 \circ \circ i \circ \dots \circ j-1 \circ \circ j \circ \dots \circ k-1 \circ \circ k \circ \dots \circ n+1$$

Question 10 (1 point). Dessinez le diagramme et les arcs désir et réalité après application de la transposition $\rho(i, j, k)$.

Question 11 (3 points). En déduire une borne inférieure pour $d(\alpha)$ en fonction du nombre de cycles $c(\alpha)$ de la permutation.

On définit la longueur d'un cycle comme étant le nombre d'arcs réalités parcourus. On numérote les arcs réalité de 1 à n en partant de la gauche. On redéfinit un cycle comme une suite de numéros d'arc réalité parcourus en suivant le chemin des arcs (réalité et désir) à partir de l'arc réalité le plus à droite dans le cycle. On définit un cycle non orienté : à partir de l'arc réalité le plus à droite, on se déplace de manière monotone vers la gauche (la suite de numéros d'arc réalité du cycle est décroissante). Les autres cycles sont orientés.

Question 12 (2 points). Dessinez la permutation 4 5 1 6 3 2 sous forme de diagramme RD. Numérotez les arcs réalité. Indiquez quel est le cycle orienté et quel est celui non orienté.

On peut montrer que pour tout cycle orienté (i_1, \dots, i_k) de longueur supérieure ou égale à 3, il existe $3 \leq t \leq k$, $i_t > i_{t-1}$ et une transposition $\rho(i_{t-1}, i_t, i_1)$ qui ajoute deux cycles à la permutation.

Question 13 (1 point). Donnez la transposition sur le cycle orienté de l'exemple ci-dessus qui permet de créer deux cycles.

On peut montrer de plus que s'il n'existe plus de cycle orienté, un cycle non orienté peut être transformé en cycle orienté grâce à une seule transposition.

Question 14 (3 points). Grâce aux deux propriétés énoncées avant, décrivez un algorithme qui permet de trier une permutation par transpositions.

