IFT3295 - TP1

Assemblage de séquences

25 septembre 2025

Évaluation

- Ce TP est à faire en équipe de deux (pas d'équipe de trois).
- Vous devez le rendre au plus tard le jeudi 16 octobre 2025 avant la démo ($\leq 13h29$).
- Deux fichiers à remettre : le code Python pour le programme demandé (un fichier zip si plus d'un fichier) et un pdf pour les autres questions.
- Utiliser Studium pour la remise du TP. Une version papier des questions écrites est aussi correcte.
- Des fonctions de base sont fournies pour lire les fichiers FASTQ et FASTA en Python. Vous pouvez les utiliser en les important depuis le fichier *utils.py* fournit avec le TP. **Il est formellement interdit d'utiliser des librairies externes pour faire l'alignement de séquences ,l'assemblage de séquences ou autres** Vous devez implémenter vous-même les algorithmes. Pour le graphe, vous pouvez utiliser des librairies NetworkX afin de construire votre graphe et le convertir en format dot.
- Votre code doit être bien lisible et compréhensible (bonne indentation, noms de variables et de fonctions représentatifs, commentaires, etc.).
- Langage de programmation : **Python (recommandé)** ou le langage de votre choix (C, C++, Java, etc.). Si vous utilisez un autre langage que Python, vous devez fournir des instructions claires pour compiler et exécuter votre code.
- Note pour l'usage de Librairies : Si vous utilisez des librairies externes, vous devez les ajouter dans le fichier requirements.txt que je vous fournis avec le TP. Plus d'infos sur les fichiers requirements.txt

Mise en situation

Vous commencez un stage dans le laboratoire du Dr Osborn. Ce dernier étudie activement un nouveau virus et a récemment identifié une protéine humaine X impliquée dans la synthèse de protéines virales. Il décide donc d'étudier cette nouvelle protéine pour mieux comprendre son rôle dans les infections virales. Pour ce faire, Dr Osborn séquence la région génomique contenant le gène de la protéine X . La technologie qu'il utilise produit des petits fragments chevauchant appelés *reads* qui recouvrent la région d'intérêt, dans un seul sens de lecture (voir figure suivante).

Le Dr Osborn requiert votre assistance pour assembler tous les reads afin d'obtenir la séquence nucléotidique de la région génomique.

Chevauchement de séquences (25pts)

Veuillez considérer les pondérations suivantes :

— "match" : +4

— "mismatch" : -4

— "indel" (insertion ou suppression) : -8

Pour assembler les reads, vous devez considérer pour chaque paire (ordonnée) de séquence X_i, X_j , le meilleur alignement tel que le chevauchement (tel illustré sur la figure suivante) entre X_i et X_j soit de score maximal.

Afin de développer un algorithme pour le problème du chevauchement maximal entre deux séquences, répondez aux questions suivantes :

1. Quelle est la différence entre un tel alignement et l'alignement global ?
2. Quelles doivent être les valeurs de la première ligne ($V(0, j) \forall j$) ? et celles de la première colonne ($V(i, 0) \forall i$) de la table de programmation dynamique V ? Justifiez votre réponse.

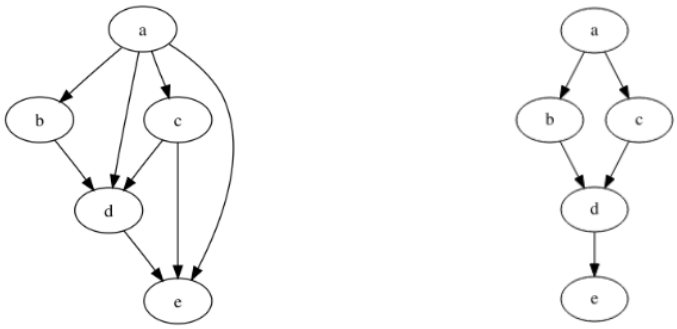
3. Quelles sont les équations de récurrence à utiliser pour remplir la table de programmation dynamique ?
4. Comment peut-on retrouver l'alignement avec le meilleur chevauchement à partir de la table de programmation dynamique ? Vous devez décrire la procédure entière pour retrouver l'alignement.
5. Implémenter l' algorithme de programmation dynamique de tel sorte qu'il fonctionne en temps $O(|X_i||X_j|)$ pour trouver l'alignement maximisant le chevauchement entre deux séquences et son score en fonction des coûts donnés plus haut. Vous devez remettre un code qui retourne pour une paire (ordonnée) de séquences : le score, l'alignement correspondant et la longueur du chevauchement. Votre programme doit prendre comme argument un fichier en format FASTQ contenant les deux séquences. Une description du format FASTQ est disponible sur wikipédia.

Assemblage de fragments (40pts)

En vous servant de votre algorithme précédent, vous pouvez procéder à l'assemblage des séquences. À cet effet, vous disposez d'un fichier *reads.fq* qui contient les séquences et identifiants de 20 reads en format FASTQ.

1. Pour chaque paire (ordonnée) de reads R_i, R_j, calculer le score de l'alignement correspondant au chevauchement maximal entre R_i et R_j. On vous demande une matrice 20x20. Notez que le score de chevauchement de R_i, R_j est différent de celui de R_j, R_i et donc que la matrice n'est pas symétrique. La diagonale de la matrice est nulle.
2. En déduire le graphe orienté de chevauchement $G = (V, E)$ où V désigne l'ensemble des reads. Il existe une arête $e_{ij} = (R_i, R_j) \in E$ entre R_i et R_j s'il existe un chevauchement entre un suffixe de R_i et un préfixe de R_j. Pour vous aider à répondre aux questions suivantes vous pouvez utiliser graphviz dot ou n'importe quel autre outil de visualisation de graphes.
   (a) Afin de ne considérer que les chevauchements pertinents, filtrez les alignements par leurs scores, en considérant un seuil minimum de 80 en dessous duquel les chevauchements sont ignorés. Quel effet a ce seuil sur le graphe résultant ?

(b) Afin d'assembler les reads, vous devez vous servir du graphe de chevauchement. Cependant, ce dernier contient plusieurs arêtes "inutiles" et peut donc être simplifié. Vous utiliserez le principe de la réduction transitive (illustrée sur la figure suivante) qui enlève les arêtes entre deux noeuds, tant qu'il existe encore un chemin simple reliant ces deux noeuds. On vous demande l'ordre des reads dans le graphe réduit, ainsi que la longueur du chevauchement entre deux reads consécutifs.
Note : Lorsque vous réduisez le graphe, si vous avez des cycles contenant seulement deux noeuds, conservez uniquement l'arête de plus grand score et supprimez l'autre.



(c) Déduire la séquence du fragment génomique séquencé et sa longueur.

Recherche d'introns et Blast (20pts)

Grâce à votre aide, le Dr Osborn a finalement obtenu la séquence génomique désirée (*sequence.fasta*). À partir d'une autre expérience, il a réussi à identifier la séquence protéique du gène X (*geneX.fasta*). Il désire maintenant identifier la séquence de l'ARNm du gène ainsi que ses introns.

1. Identifier la position de la protéine X au sein de la région génomique. Pour ce faire traduisez d'abord la séquence nucléotidique de (*sequence.fasta*) dans les trois différents cadres de lecture, en considérant qu'il s'agit du brin codant. Utilisez le code génétique standard (disponible ici ==> table génétique). Veuillez noter que les nucléotides et les acides aminés sont toujours représentés par un seul caractère dans les séquences et que le signal de terminaison de la traduction (stop) peut être représenté par une étoile (*). Répondez aux questions suivantes :
   (a) Dans quel cadre de lecture se trouve le codon start de la séquence protéique ?
   (b) Décrivez un algorithme de programmation dynamique (conditions initiales, relations des récurrence, chemin à suivre dans la table pour retrouver l'alignement optimal) permettant de retrouver le gène de la protéine en minimisant les introns. En raison d'erreur de séquençage, on autorisera des mismatchs dans les exons, mais pas de indels. Vous pouvez supposer que tous les exons sont dans le même cadre de lecture.
2. En vous servant de l'outil Blastp et/ou de uniprot , identifiez le nom de la protéine X ainsi que sa fonction.

Repliement d'ARNs (15pts)

Pour les questions suivantes, étant donné une séquence d'ARN S, on veut retrouver le repliement en une ou plusieurs tiges-boucles de S qui maximise le nombre d'appariement de bases. Notez qu'une tige-boucle peut contenir des nucléotides non-appariés et que seuls les appariements (A-U et G-C) sont considérés.

7. Considérez la séquence d'ARN ci-dessous avec ses paires de bases inférées. Dessinez un repliement en 2D qui illustre les sous-structures principales de la molécule.

8. **Bonus :** Suposons que l'on veuille maximiser le nombre d'empilements, c'est-à-dire le nombre d'appariements (i,j) tels que (i+1,j-1) sont aussi appariés. Décrivez un algorithme de programmation dynamique (conditions initiales, relations des récurrence, chemin à suivre dans la table pour retrouver l'alignement optimal) fonctionnant en temps temps $O(n^3)$ qui permet de retrouver un repliement en une ou plusieurs tige-boucles maximisant le nombre d'empilements.