Charles-Antoine Lanoix remis le 6 octobre 2019  
Félix Adam

TP1 Bio-informatique

Chevauchement de séquences

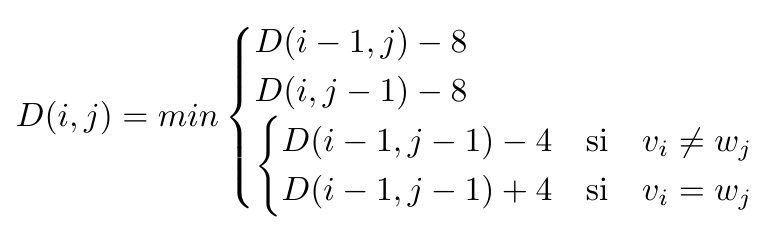
1. **Quelle est la différence entre un tel alignement et l’alignement global ?**

Le chevauchement représente l’alignement avec le meilleur score entre le suffixe de xi et le préfixe de xj. L’alignement global recherche un meilleur score entre deux séquences entières.

1. **Quelles doivent être les valeurs de la première ligne ( V (0, j)∀j ) ? et celles de la première colonne ( V (i, 0)∀i ) de la table de programmation dynamique V ? Justifiez votre réponse.**

Les valeurs de la première ligne et de la première colonne sont initialisées à zéro afin que ce qui se trouve hors du chevauchement ne pénalise pas le score de l’alignement.

1. **Quelles sont les équations de récurrence à utiliser pour remplir la table de programmation dynamique ?**



1. **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.**

On remplit la table de programmation dynamique selon l’algorithme vu en classe d’alignement global avec la distance d’édition. Une fois la table remplie, on s’intéresse à la plus grande valeur de la dernière colonne. On suit le chemin inverse emprunté pour se rendre à cette valeur afin d’écrire l’alignement avec le meilleur chevauchement.

1. **Déduisez-en un algorithme 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 de séquences : le score, l’alignement correspondant et la longueur du chevauchement. Votre programme doit prendre comme argument un fichier contenant les deux séquences.**

Voir algo\_chevauchement.py

Assemblage de fragments

1. **Pour chaque paire de reads Ri, Rj, calculer le score de l’alignement correspondant au chevauchement maximal entre Ri et Rj. On vous demande une matrice 20x20. Notez que la matrice n’est pas symétrique et que sa diagonale est nulle.**

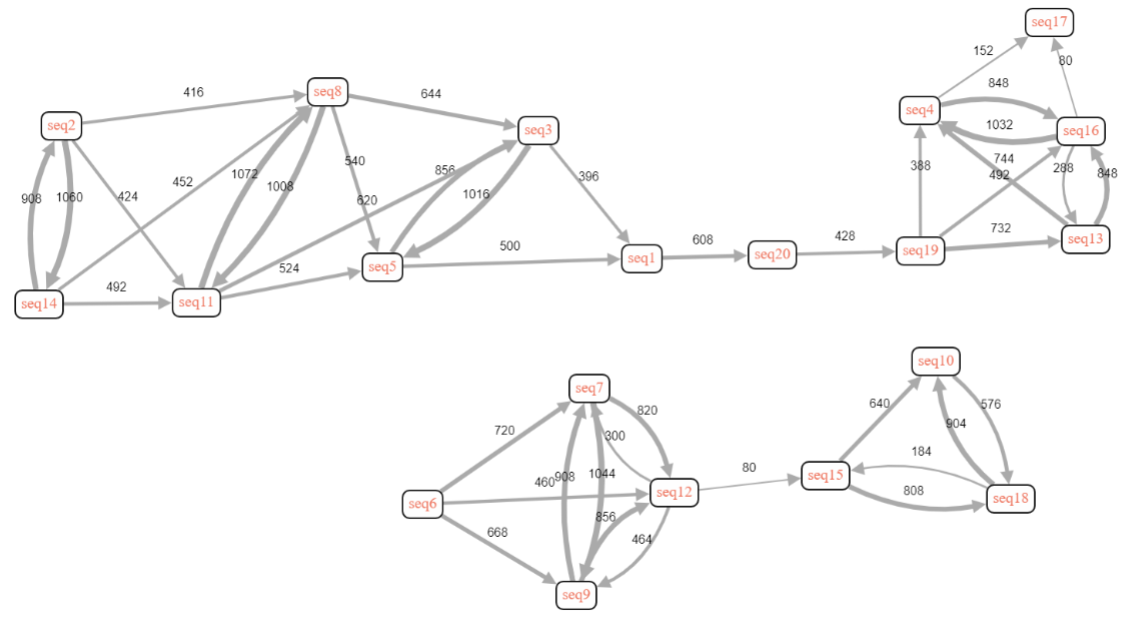
Voir fichier matrice\_score.ods

1. **En déduire le graphe orienté de chevauchement G = (V, E) où V désigne l’ensemble des reads. Il existe une arête eij = (Ri , Rj ) ∈ E entre Ri et Rj s’il existe un chevauchement entre un suffixe de Ri et un préfixe de Rj . Pour vous aider à répondre aux questions suivantes, vous pouvez utiliser cet outil de visualisation : https://mrnoutahi.com/dagViz/ ou un autre outil de visualisation de graphes, comme graphviz.**
2. **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 ?**

Avant d’appliquer un seuil, le graphe était connexe donc impossible de différencier les

reads forward et reverse. En appliquant un seuil de 80, on remarque que le graphe

devient non-connexe.



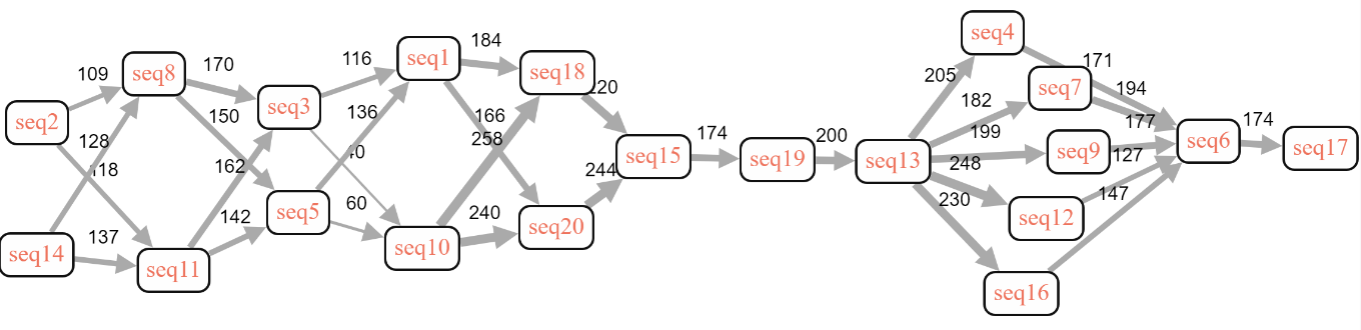
1. **Sachant que le read @READS\_2 est définitivement sur le brin forward, déduisez-en l’ensemble des reads reverse.**

Tous les sommets qui ne sont pas connexe au sommet seq2, soit l’ensemble {seq6, seq7, seq9, seq10, seq12, seq15, seq18}.

1. **Puisque vous venez d’identifier les reads reverse, vous pouvez remplacer leurs séquences par les séquences complémentaires inverses correspondantes. À titre d’exemple, la séquence complémentaire de X = ACTGCAA est Xc = TGACGTT et la séquence complémentaire inverse est = TTGCAGT.**
2. **Construire à nouveau le graphe de chevauchement avec les nouvelles séquences en applicant le seuil précédent. Que remarquez vous?**

Avec les séquences complémentaires et inversés des brins reverse, on obtient un graphe connexe. Le fait que tous les sommets soient connexes nous indique que nous avons bien identifié les brins reverse.

1. **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.**

****

Voir graphe\_chevauchement\_reduc\_transitive.png pour une meilleure résolution et longueur\_chevaucement\_reduc\_transitive.txt pour les longueurs de chevauchement entre les reads.

1. **Déduire la séquence du fragment génomique séquencé et sa longueur.**

Recherche d’intros et Blast

1. **Identifiez 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). 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 :**
2. **Dans quel cadre de lecture se trouve le codon start de la séquence protéique ?**

Le codon start de la séquence protéique se trouve dans le cadre de lecture 2.

1. **Décrivez un algorithme de programmation dynamique qui vous permet de retrouver les différents fragments de la protéine X au sein de la séquence nucléotidique. Vous ne devez considérer que les plus longs fragments qui ne se chevauchent pas. Comme indice, vous pouvez supposer qu’il ne devrait pas avoir de mismatch entre les fragments de la séquence protéique et la traduction des régions nucléotidiques correspondantes.**
2. **En déduire les intervalles de positions contenant les exons (au sein de la séquence génomique) ainsi que la séquence de l’ARNm mature (après épissage).**
3. **En vous servant de l’outil Blastp et/ou de uniprot , identifiez le nom de la protéine X ainsi que sa fonction**