Projet 2 : à la recherche de la petite bête

Quelle est cette pathologie mystérieuse ? Nous sommes le 1er janvier 2020, et une équipe vient de séquencer des échantillons pulmonaires de personnes présentant une maladie respiratoire atypique. Cette question hante le laboratoire de recherche dans lequel vous vous trouvez.

Les données sont accessibles à cette adresse (et les données brutes de séquençage peuvent directement être récupérées ici).

Votre but sera d'identifier à quel virus ressemble le pathogène présent dans cet échantillon pulmonaire. Mais attention, l'échantillon récupéré contient principalement des cellules humaines, on s'attend à ce que seule une petite fraction corresponde à des séquences du pathogène.

Parmi les pathogènes envisagés, nous allons chercher si on trouve des séquences semblables à ces virus : grippe, rhinovirus, HIV et coronavirus.

Voici les adresses auxquelles vous pouvez récupérer les séquences des génomes correspondants :

- Grippe A
- HIV-1
- Rhinovirus
- Alpha-coronavirus

Dans tous les cas, téléchargez le fichier dont l'extension est `.fna.gz` (c'est un fichier FASTA compressé) et décompressez-le (par exemple avec la commande `gunzip`).

Le but est donc d'aligner les reads sur chacun de ces génomes et de voir sur quel génome, le plus de reads sont alignés. L'alignement que vous développerez consiste en une stratégie *seed and extend*.

seed On extrait des courtes séquences (des k-mers) du read et on essaie de les trouver, sans tolérer d'erreur, dans le génome d'intérêt.

extend Une fois une graine trouvée, on essaie d'aligner le read en entier sur cette région du génome.

Comment aligner ?

Il existe de nombreuses façons de réaliser cet alignement. Néanmoins, un préalable est de pouvoir rechercher des k-mers dans un génome et de connaître leur position dans ce génome. Pour cela, le génome devra être indexé, nous allons choisir une table des suffixes.

Ensuite, pour la phase d'extension, vous pourrez utiliser des bibliothèques existantes qui permettent de réaliser de l'alignement semi-global de séquences. Par exemple la bibliothèque `parasail` est efficace et disponible aussi bien en Python qu'en C/C++, Rust ou Java.

Passage à l'échelle

Dans un premier temps vous alignerez vos reads sur de petits génomes (des génomes de virus), néanmoins votre programme doit idéalement pouvoir passer à l'échelle sur de plus grands génomes…par exemple de chauve-souris. Attention donc à la mémoire dont vous aurez besoin.

D'autre part, vous avez un grand nombre de reads à traiter, le traitement ne doit donc pas être trop long (en Python, il ne faut pas s'attendre à traiter beaucoup plus de 1 000 reads par seconde). Pour optimiser le traitement, penser à ce qui vous permettra de gagner du temps :

- il n'y a pas forcément besoin d'essayer tous les k-mers d'un read : entre un k-mer et le suivant dans un read, $k-1$ nucléotides sont communs. Il y a donc peu de nouvelle information en passant d'un k-mer au suivant. On pourrait donc envisager de sauter des k-mers sans perte d'information et ce qui permettrait de gagner du temps
- à l'inverse, si on souhaite gagner de la place, il n'est peut-être pas indispensable d'indexer tous les suffixes mais on pourrait n'en indexer qu'une fraction (à l'inverse, il faudrait dans ce cas probablement interroger tous les k-mers pour compenser la perte d'information dans l'index).

Concernant la bibliothèque `parasail` mentionnée précédemment, celle-ci permet d'exécuter plusieurs instructions en parallèle (il s'agit des fonctions vectorisées), et donc d'aller plus vite.

De manière générale, avant de foncer vers une solution, il faut toujours se poser la question de son passage à l'échelle sur de grands génomes (en milliards de nucléotides) ou sur de grands jeux de données, en centaines de millions de reads.

Entrée et sortie

Votre programme prendra en entrée un génome au format FASTA, ou multi-FASTA ainsi qu'un jeu de reads au format FASTQ. En sortie, votre programme écrira un fichier contenant uniquement les reads alignés sur le génome dans un format tabulé, détaillé après.

Votre programme acceptera donc les paramètres suivants **obligatoires** :

- `-genome` : chemin vers un fichier FASTA contenant le génome de référence
- `-reads` : chemin vers un ou plusieurs fichiers FASTQ.gz contenant des reads à aligner sur le génome de référence
- `-out` : fichier de sortie

Il acceptera également des paramètres optionnels, dont :

- `-k` : la taille des k-mers utilisés

Format de sortie

Le fichier de sortie produira une ligne par read aligné sur le génome de référence. Chaque ligne sera séparée en différentes colonnes, séparées par des tabulations.

Voici la liste des colonnes :

- Nom du read dans le fichier FASTQ
- Position (approximative) du début de l'alignement sur le génome de référence
- Score de l'alignement
- Séquence du read
- (optionnel) CIGAR de l'alignement (fourni par `parasail`)

Nouveau jeu de données

Après la réussite de la première étape, et de l'identification du virus, on vous donne accès à un nouveau jeu de données qui permettrait de remonter à l'origine de cette pandémie. Des échantillons prélevés sur des chauves-souris dans un endroit encore tenu secret ont été expédiés à quelques laboratoires autour du monde pour analyse.

Votre laboratoire, reconnu pour la qualité de ses analyses, figure parmi les heureux élus : vous avez réceptionné un échantillon il y a quelques jours. L'équipe du *wet lab* s'est immédiatement mise au travail pour mettre en place le séquençage de ces échantillons provenant de chauve-souris.

Le séquenceur vient de terminer son travail et vous récupérez un fichier FASTQ (ou un extrait comportant un million de reads) contenant les reads produits.

À l'aide de votre outil, retirez tous les reads correspondant au SARS-CoV-2 et à la chauve-souris il doit rester un certain nombre de reads qui, pour diverses raisons, n'ont pas été filtrés.

Il est normal que des reads ne soient pas filtrés. Cela peut être à cause de reads comportant trop d'erreurs de séquençage, à cause de contamination du séquençage par un autre séquençage, des morceaux d'adaptateurs qui ont été séquencés, …ou une autre espèce qui était réellement présente dans l'échantillon.

Pour savoir ce qu'il en est, déterminez quels sont les k-mers les plus fréquents parmi les reads non filtrés [1] et déterminez s'ils appartiennent à une espèce donnée. Si c'est le cas vous filtrerez également les reads appartenant à cette espèce.

1. Pour cela, vous pouvez utiliser le programme `kmc` qui permet de compter les k-mers, une fois lancé sur votre jeu de données, vous utiliserez `kmc_dump` pour visualiser les k-mers et leurs nombre d'occurrences

Rendu

Le rendu de votre travail est attendu pour le **lundi 4 avril 12h**.
Ce rendu devra comporter les éléments suivants :

- Votre code source permettant de réaliser les différentes tâches demandées
- Un fichier README expliquant la manière d'utiliser (voire compiler s'il y a lieu) vos programmes
- Des tests (par exemple des doctests en Python) permettant de vérifier le bon fonctionnement de vos fonctions, mais aussi des tests fonctionnels qui vérifient, sur de tous petits exemples, que votre programme produit le fichier de sortie attendue,
- Un petit rapport (au format Markdown, par exemple) expliquant votre travail
  - les structures de données utilisées,
  - l'implantation choisie pour ces structures de données,
  - la raison de ces choix,
  - les manières de favoriser le passage à l'échelle de vos programmes,
  - la façon dont vous avez choisi les paramètres pour vos programmes, ainsi que les évaluations réalisées afin de déterminer ces paramètres,
  - les résultats obtenus (nombre de reads alignés pour chaque espèce sur les différents jeux de données, temps écoulé et mémoire consommée).