TP MEA

Contexte

Un des traitements principaux de la génomique est l'assemblage des génomes. A partir des textes des fragments d'ADN produits par les séquenceurs, il s'agit de reconstituer le texte complet du génome d'un organisme vivant.

Il est quelque fois difficile d'extraire uniquement l'ADN de l'organisme étudié parce qu'il vit en symbiose étroite avec d'autres organismes. Chez de nombreuses plantes, par exemple, il existe toute une communauté microbienne qui co-existe. Ce peut être la même chose chez quelques insectes qui ne peuvent vivre sans abriter quelques bactéries bien spécifiques.

L'ADN de l'organisme étudié et de ses symbiotes est alors difficile à séparer. On séquence donc l'ensemble. Par conséquent, les données brutes contiennent à la fois des fragments d'ADN qui proviennent du génome de l'organisme et des fragments d'ADN qui proviennent de ses symbiotes.

Une stratégie classique pour séparer les fragments se base sur l'observation que les organismes symbiotiques sont beaucoup plus nombreux et que leurs génomes sont beaucoup plus petits. Le puceron du poix dont la taille du génome est d'environ 400Mbp, par exemple, abrite des millions de petites bactéries de l'espèce *buchnera* (taille du génome = 0.65Mbp).

La séparation des textes de l'ADN se fait en comptant des k-mers. Si on fait l'hypothèse que l'ADN prélevé contient en proportion beaucoup plus d'ADN symbiotique, alors le nombre de k-mers appartenant aux symbiotes est beaucoup plus élevé.

Problème

On a le résultat du séquençage d'un organisme de 80 Mpb et de son symbiote (unique) de 1 Mbp. La couverture globale est de 20X. On sait également que le génome du symbiote est 20 fois plus présent. La taille des fragments d'ADN produits par le séquenceur (reads) est de 100bp. Globalement, la taille cumulée des génomes est de 80 Mbp + 20 x 1Mbp = 100Mbp. Une couverture de 50 donne un séquençage de 50x100Mbp = 5Gpb. Sachant que les fragments (reads) ont une taille de 100, les données brutes représentent donc 5.10^9 / 100 = 50.10^6 reads. Le nombre de k-mers à prendre en considération est donc égale à :

nombre de reads x nombre de k-mers par read = 50.10^6 x (100-31+1) = **3.5×10^9**

L'objectif du TP est de découper tous les reads en k-mer de taille **31**, de les compter et de les séparer en 2 groupes : ceux qui sont vus un petit nombre de fois et ceux qui sont vus un grand nombre de fois. A partir de ces 2 ensembles de k-mers, on peut lancer un assemblage sur l'un ou l'autre de ces ensembles. Celui qui contient les k-mers de forte occurrence produira l'assemblage du symbiote. Inversement, l'ensemble qui contient des k-mers de faible occurrence produira l'assemblage de l'organisme.

Objectif du TP

Elaborer un programme qui produit seulement l'ensemble des k-mers à forte occurrence (i.e. les k-mers qui appartiennent au symbiote). Les données d'entrée sont les données de séquençage. Les données de sortie seront au format Fasta, avec un k-mer par ligne.

Réalisation du TP

Sur le serveur CeSGO :
https://data-access.cesgo.org/index.php/s/nqTrqbhyGhCr2wl (mot de passe : esir2018)
allez dans le répertoire TP

Les fichiers suivants sont disponibles :

- **xtrkmer31_template.c** : programme C pour initier le TP.
- **generator.h** : fichier qui contient une fonction permettant de simuler la génération des données de séquençage.
- **generator.txt** : fichier de données utilisé pour la génération de données de séquençage
- **bactmyst.fasta** : fichier fasta qui contient le texte du symbiote. A utiliser pour vérification que la liste des k-mers produits est correcte
- **kmer31.txt** : exemple de fichier de sortie au format fasta

Travail à rendre

1. Un rapport (format pdf) décrivant l'architecture du programme ainsi que ces performances : temps d'exécution, empreinte mémoire, nombre de faux positif (k-mers trouvés mais qui n'appartiennent pas au symbiote), nombre de faux négatifs (k-mers non trouvés), …

2. Un fichier contenant le programme C (avec des commentaires). Le programme sera testé avec le compilateur gcc.

Divers

Pour optimiser les performances et faciliter le debug, compiler de la manière suivante :

```
gcc –O3 –Wall –o xtrkmer31 xtrkmer31.c
```