Atelier assemblage

Introduction

Machine virtuelle

Lancez votre machine virtuelle. Il s'agit d'une distribution Linux Ubuntu. Nous utiliserons principalement la console, accessible en un clic sur le bandeau gauche. Toutes les données se trouvent dans le répertoire "`~/Documents/DATA`".

Les temps donnés pour ce TP sont indicatifs, et peuvent varier en fonction de la puissance de ordianteur et de la quantité de mémoire.

Ce TP abordera plusieurs problématiques de l'assemblage : contrôle qualité, assemblage, mapping et scaffolding. L'assemblage pourra se faire avec Velvet ou Minia.

Bonnes pratiques

Créez des répertoires pour ranger les sorties de chaque outil et donnez des noms de fichier de sortie intelligible. Par exemple, *Sickle/paired_reads1.fastq* plutot que *~/toto.out* … Notez dans le même répertoire la commande utilisée pour générer les fichiers (l'historique des commandes se consulte avec la commande *history*).

Vérifiez régulièrement l'espace disque de votre machine virtuelle (par exemple avec la commande *df*). La machine virtuelle a été créée avec suffisamment d'espace (~150Go), mais il ne faut pas hésiter à faire régulièrement du nettoyage en supprimant les fichiers de résultats intermédiaires ou inutiles.

Contrôle qualité

Objectif

Lorsque vous récupérez les séquences du prestataire, vous vous posez certaines questions :

- Les séquences obtenues sont-elles conformes au niveau de prestation attendu ?
- Les séquences ont-elles la taille attendue ?
- Le nombre de séquences obtenu est-il en adéquation avec celui attendu ?
- Les séquences sont-elles de bonne qualité ?

- Reste-t-il des adaptateurs, des tags ?

Une question qu'on se pose moins souvent est : les séquences conviennent-elles pour répondre aux questions biologiques posées, et ce dans quelles limites ?

Pour répondre à ces deux questions, il faut effectuer un contrôle qualité.

Il est possible d'effectuer différentes étapes en vue de nettoyer les séquences. En effet, on peut éliminer les séquences contenant des N, les séquences d'adaptateurs[1], effectuer un élagage sur la qualité, filtrer sur la taille et rechercher des séquences contaminantes issues d'autres organismes biologiques.

Le format FASTQ

Le format FASTQ est un format textuel permettant de stocker à la fois la séquence biologique et les scores de qualité associés. Les nucléotides et les qualités sont encodés chacun par un caractère ASCII. Historiquement, le format FASTQ a été créé par le Sanger Institute, mais ce format est largement utilisé par les séquenceurs Illumina ou Nanopore. Le problème est qu'il existe plusieurs variantes du format FASTQ, notamment pour la plateforme Illumina. Les outils prennent généralement en entrée des séquences au format FASTQ-Sanger. Les dernières versions du FASTQ-Illumina (Casava à partir de 1.8) sont équivalentes au FASTQ-Sanger et ne nécessitent plus de conversion.

Pour aller plus loin:

- la page Wikipédia du format FASTQ,

- l'article de P. J. A. Cock *et al.* : The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. (Nucleic Acids Res, 2009).

Outils

Nous utiliserons plusieurs programmes pour évaluer la qualité du séquençage, écourter les reads, compresser/décompresser les fichiers volumineux, assembler, évaluer les assemblages, mapper et visualiser le mapping. Les programmes utilisés, accompagnés d'une brève description de leur fonction suit :

- Quip: Outil en ligne de commande permettant la compression optimisée de fichiers FASTQ, SAM et BAM. Il nous aide à diminuer la taille des fichiers FASTQ.

- FastQC: Outil graphique ou en ligne de commande produisant un rapport synthétique sur la qualité de données FASTQ.

- Sickle: Outil en ligne de commande de filtrage et nettoyage des bases mal séquencées, repérées selon des critères de qualité, dans les reads.

1 Généralement les adaptateurs sont filtrés par le prestataire de service proposant le séquençage.

- Minia, assembleur de données de séquence haut-débit utilisant très peu de mémoire
- Velvet, assembleur de données de séquence haut débit.
- QUAST: Calcule un ensemble de mesures sur l'assemblage.
- SSPACE: Raccorde les contigs en scaffolds plus longs.
- Bowtie2: Aligne des reads sur un génome de référence.
- Tablet: Visualisateur de génome spécialisé dans la représentation des données de mapping.

Exercice

Le jeu de données

Dans cet exercice, nous assemblerons en contigs des séquences d'une souche d'*E. coli* obtenues à partir d'un séquenceur de paillasse MiSeq d'Illumina. Ces données de très grande qualité sont mises à disposition par le fabricant. Tous les renseignements sur ce jeu de données peut être accessible sur: https://www.illumina.com/informatics/sequencing-data-analysis/data-examples.html.

En résumé, l'ADN génomique a été extrait de la souche K12 MG1655 d'*E. coli* et séquencé en utilisant un MiSeq, produisant des données paired-end 2 x 150 nt. Le génome d'*E. coli* K12 compte environ 4,6 millions de nucléotides. Dans le cadre de ce TP, afin d'accélérer les traitements, nous travaillerons sur 25% seulement des reads paired-ends :

- `MiSeq_Ecoli_MG1655_110721_PF_R1_25.fastq.qp`
- `MiSeq_Ecoli_MG1655_110721_PF_R2_25.fastq.qp`

Objectif

Votre travail consiste dans un premier temps à effectuer un contrôle qualité sur le jeu de données fournis. Dans un second temps, il vous faudra nettoyer les séquences en vous référant aux différentes étapes de nettoyage énoncées ci-dessus.

Analyse de la qualité des données

Décompressez les fichiers `.qp` avec la commande (< 1 min) :

`unquip` *`fichier1`*`.qp [`*`fichier2`*`.qp ...]`

`Observez les première lignes du fichier :`

less fichier.fastq

(ou head fichier.fasq)

Comment est encodée la qualité ? Quelle signification ont les identifiants ?

Lancez FastQC en ligne de commande sur le jeu de données décompressé avec la syntaxe suivante (~2 min) :

`fastqc` *`fichier1`* [*`fichier2`* ...] `-o` *`répertoire_sortie`*

Pour la visualisation des résultats, ouvrez avec firefox la page html générée :

`firefox` *`xxx`*`_fastqc/fastqc_report.html &`

Remarque : vous pouvez également lancer fastqc en mode graphique (commande fastqc *seule), ce qui vous permettra d'accéder à l'aide qui décrit les différents modules d'analyse.*

Qu'observez_vous ?

Pouvez vous calculer la couverture de séquençage ?

Nettoyage des reads

Le nettoyage des reads se fera avec *sickle*. Cet outil fait passer une fenêtre glissante de l'extrémité 3' à l'extrémité 5', et supprime tout ce qui est en-dessous d'une qualité donnée. Il peut supprimer une lecture, qui à la suite d'un nettoyage, compte un nombre de nucléotides inférieur à un seuil donné. Il utilise la syntaxe suivante (> 30 secondes) :

`sickle pe -f` *`fichier1.fastq`* `-r` *`fichier2.fastq`* `-t` *`encodage_FASTQ`* `-o` *`fichier1_propre.fastq`* `-p` *`fichier2_propre.fastq`* `-s` *`fichier_reads_seules.fastq`* `-q` *`qualité_min`* `-l` *`taille_min`*

Pour nous, l'encodage est au format `sanger`. Nous pouvons choisir une qualité minimale de 30, avec au moins des reads de taille 100, et sans N.

Relancez ensuite de nouveau FastQC, que remarquez-vous?

Assemblage

Deux assembleurs ont été installés sur la machine virtuelle. Vous utiliserez deux assembleurs (Minia et Velvet) et suivez les instructions pour l'assemblage. Nous comparerons les résultats en fin de TP sur une fiche récapitulative (~/Documents/Fiche recapitulative - Resultats assemblage.xlsx).

Minia

Introduction

Minia est un programme d'assemblage de génome utilisant très peu de mémoire. Il lit des données de séquençage haut-débit (typiquement venant d'un séquenceur Illumina) et construit un ensemble de contigs (c'est-à-dire des séquences assemblées) se rapprochant du génome attendu. Minia est basé sur une représentation compacte de l'arbre de de Bruijn. Minia a besoin de moins de ressources de calcul que d'autres assembleurs. Par exemple, il est capable d'assembler un génome humain avec seulement 5,7 Go de RAM.

Il est à noter que Minia n'assemble qu'en contigs. D'autres logiciels doivent être utilisés pour assembler ces contigs en scaffolds.

Minia a été développé pour assembler des reads produites par des séquenceurs Illumina. Minia peut également assembler des reads d'autres séquenceurs s'ils sont donnés au format Fasta ou FASTQ, toutefois cela n'est pas recommandé.

Minia n'utilise pas l'information de qualité des fichiers FASTQ durant l'assemblage. À la place, Minia quantifie l'abondance des k-mers et n'utilise que les k-mers présents au-dessus d'un seuil donné pour l'assemblage. Cette technique élimine la majorité des k-mers comportant des erreurs de séquençage (l'étape préalable de nettoyage n'est ainsi pas obligatoire).

Liens

Minia est déjà installé dans le VM,

Un court manuel de Minia : http://minia.genouest.org/files/manual.pdf

Sur la machine virtuelle, le manuel de minia est dans : */usr/local/bioinfo/minia/manual/*

Installation

Sur la machine virtuelle, Minia est déjà installé. Minia peut être installé sur n'importe quel ordinateur Linux ou Mac. Vous pouvez le télécharger sur http://minia.genouest.org/.

Un court manuel de Minia est sur http://minia.genouest.org/files/manual.pdf ou directement sur la machine virtuelle dans le répertoire */usr/local/bioinfo/minia/manual/*

Si vous l'installer (ou réinstaller), il faudra télécharger le ou les fichiers source, décompresser dans un répertoire préféré et tapez: make (c'est une compilation afin d'obtenir l'exécutable)

Il est aussi possible de compiler Minia pour utiliser des kmers plus longs. Pour ce faire, il faut le spécifier à la compilation. Par exemple, pour utiliser des kmers de taille 100, vous devrez taper : `make k=100`

Paramètres

Une utilisation classique de Minia suit la structure suivante :

`minia` *`fichier_entrée taille_kmer abondance_min taille_génome préfixe_sortie`*

Sur un exemple, cela donne :

minia reads.fastq 31 3 4500000 préfixe_sortie

où `reads.fastq` est le fichier d'entrée (les reads), `31` est la taille des *k*-mers, `3` est le nombre d'occurrence minimum, `4500000` est une estimation de la taille du génome, et `préfixe_sortie` indique l'emplacement des fichiers de sortie. Les détails des paramètres seront donnés ci-dessous.

`fichier_entrée`: le fichier d'entrée au format *FASTA ou FASTQ:* Minia n'utilise pas l'information de qualité présente dans le fichier FASTQ. Il accepte des reads paired-end ou mate-pairs, mais n'utilise pas l'information de paire. Les types des fichiers d'entrée peuvent être mélangés (FASTA ou FASTQ). Les fichiers peuvent être compressés avec gzip et doivent se terminer par *.gz*.

Plusieurs fichiers: Minia peut accepter plusieurs fichiers d'entrée. Un fichier contenant tous les fichiers d'entrée (une ligne par fichier) doit alors être créé et donné à Minia. C'est cette solution que nous utiliserons pour des fichiers paired-ends.

Sortie

Minia produit un ensemble de contigs au format FASTA dans le fichier `contigs.fa`.

Exercice

Assemblage de contigs de novo *avec Minia*

Durant l'assemblage des reads, il est intéressant de mesurer l'effet de l'abondance des reads, de la taille de *k*, et de l'abondance minimale sur l'assemblage. Mais, faute de de temps, nous ne travaillerons ici que sur un quart des données et nous évaluerons seulement l'effet de *k* sur l'assemblage.

Nous ferons cette évaluation avec QUAST, qui est un logiciel qui produit un grand nombre de mesures sur la distribution de contigs donnés en entrée.

Pour mémoire, la syntaxe générale de Minia est :

`minia` *`fichier_entrée taille_kmer abondance_min taille_génome préfixe_sortie`*

Il est difficile d'estimer la meilleure taille de k-mer à choisir, elle varie à chaque scénario d'assemblage. Une limite basse absolue pour k est la taille à partir de laquelle un mot aléatoire n'est vu qu'une fois dans le génome (10-12 pour les génomes courants). La limite haute est la longueur des reads – 1. Entre les deux, tout est une question de couverture de séquençage et de taux d'erreur. De manière à explorer l'impact des k-mers sur l'assemblage, nous allons vous demander de lancer Minia sur les données nettoyées, avec un k de votre choix, pris entre 31 et 71, `abondance_min=7` et `taille_génome=4500000`. Donnez un nom parlant pour `préfixe_sortie`, de manière à pouvoir le retrouver facilement.

L'assemblage peut prendre plus de 30 minutes. Vous pouvez en profiter pour lire le manuel de QUAST (*/usr/local/bioinfo/quast2.1/manual.html* sur la machine virtuelle).

Velvet

Introduction

Velvet est un outil d'assemblage de reads courtes développé à l'EBI. Velvet utilise les reads nettoyées au format FASTQSanger, FASTA selon la méthode et les paramètres présentées dans la première section.

Velvet fonctionne en 2 étapes: indexation des reads (commande `velveth`), puis assemblage (commande `velvetg`). Typiquement, la phase d'indexation n'est effectuée qu'une seule fois par taille de kmers, puis la phase d'assemblage peut être relancée avec des paramètres différents pour optimiser l'assemblage. On peut passer les reads à `velveth` au format FASTQ, FASTQ gzippé, FASTA ou FASTA gzippé. Velvet peut prendre en entrée des fichiers single end ou pairés de taille d'inserts différentes. Si on lui donne des reads pairés, Velvet peut effectuer un scaffolding. Cependant, le scaffolder de velvet n'est pas connu pour être le plus efficace. Nous le désactiverons donc dans le TP et utiliserons un outil dédié.

Indexation

Pour effectuer l'indexation :

`velveth [`*`nouveau_repertoire`*`] [`*`taille_kmer`*`] -fastq -shortPaired -separate [`*`reads1.fastq reads2.fastq`*`]`

De la même manière que pour Minia, il est difficile d'estimer à priori la meilleure taille de k-mer à choisir. Cependant, un outil, Velvet Adviser permet de donner une première indication à partir des caractéristiques du séquençage. Un autre outil, VelvetOptimiser, permet d'explorer de façon systématique différentes tailles de k-mers. Pour cet exercice, nous allons nous répartir différentes tailles de k-mer entre les participants au TP, afin de trouver la taille de k-mer optimale. Vous pouvez choisir un k allant de 31 à 71.

Assemblage

Pour effectuer l'assemblage:

```
velvetg [repertoire] -scaffolding no -cov_cutoff auto -exp_cov auto -amos_file yes
```

La phase d'assemblage demande de nombreux paramètres, notamment des seuils d'abondance (`cov_cutoff` et `exp_cov`). Ces seuils permettent à Velvet de décider quelles séquences sont répétées ou non, et de traverser avec des heuristiques plus agressives les séquences non répétées. Velvet sait assigner ces seuils par défaut, mais il faut lui demander en spécifiant :

```
-cov_cutoff auto -exp_cov auto
```

Veillez à utiliser l'option "`-amos_file yes`" si vous souhaitez sortir l'assemblage au format afg afin de le visualiser dans *tablet* (installé sur la VM) ou un outil dédie comme *hawkeye* (pas installé sur la VM).

Évaluation du contigage avec QUAST

Lancez QUAST sur votre fichier de sortie. La syntaxe est (~ 7 min) :

quast.py -o répertoire_sortie fichier_contig

Utilisez un nom parlant pour votre répertoire de sortie, où QUAST écrira un grand nombre de fichiers. Allez dans ce répertoire et lisez le fichier `report.html`. Les métriques sont décrites dans le manuel (*/usr/local/bioinfo/quast2.1/manual.html* sur la machine virtuelle).

Veuillez renseigner vos résultats sur la fiche récapitulative (~/Documents/Fiche recapitulative - Resultats assemblage.xlsx).

Scaffolding d'un assemblage avec SSPACE

Étant donné un assemblage réalisé par Minia ou Velvet (sans scaffolding), il est possible d'utiliser un autre outil pour combiner les contigs et l'information pairedend. Un tel outil est appelé un *scaffolder.* Les contigs sont reliés entre eux, lors que cela est possible, de manière à former des scaffolds (contigs séparés par des trous, représentés par des séquences de nucléotides `N`).

Nous allons utiliser SSPACE, un scaffoldeur répandu. L'outil est commercial mais gratuit pour utilisation académique. Il est installé sur la machine virtuelle.

Les données en entrée de SSPACE sont les suivantes:

- Contigs produits par Minia ou Velvet
- Reads pairés
- fichier de configuration de SSPACE

Le fichier de configuation de SSPACE contient le chemin des reads et quelques informations sur la distribution des inserts (taille moyenne, déviation, orientation). Placez la ligne suivante dans un fichier nommé `miseq-libs.txt:`

```
lib1 [reads1.fa] [reads2.fa] 300 0.75 FR
```

Vérifiez que `reads1.fa` et `reads2.fa` sont bien les données pairées MiSeq trimmées avec Sickle. Les données d'origines (non trimmées) ont trop d'erreurs de séquençage pour être exploitées par SSPACE. Les paramètres `300 0.75 FR` sont respectivement: taille d'insert moyenne 300 bp, incertitude de 75% i.e. les tailles d'inserts sont de [300*(10.75); 300*(1+0.75)] bp, orientation des reads (FR = forward-reverse = reads paired-end; RF = reverse-forward = reads mate-pairs).

SSPACE s'exécute avec la ligne de commande suivante:

```
SSPACE_Basic_v2.0.pl -l miseq-libs.txt -s [contigs.fa]
```

Par défaut, le résultat (les scaffolds) se trouve dans le fichier :

standard_output.final.scaffolds.fasta

Le fichier `standard_output.summaryfile.txt` contient des statistiques sur l'assemblage produit.

Relancez QUAST, en spécifiant un génome de référence, pour regarder la qualité des scaffolds:

```
quast.py -R ~/Documents/DATA/Reference/NC_000913.fna
standard_output.final.scaffolds.fasta
```

Optionnel: Vous pouvez aussi comparer le résultat de SSPACE avec le résultat d'un scaffolding produit par Velvet. Pour cela, il suffit juste de relancer `velvetg` en supprimant le paramètre:

`-scaffolding no` ce qui donne:

```
velvetg [repertoire] -cov_cutoff auto -exp_cov auto
```

Optionnel: avec QUAST et le génome de référence (NC_000913.fna) comparer la qualité des scaffolds produits par Velvet et SSPACE (en utilisant les contigs de Minia ou de Velvet). Que remarqueton ? (regarder le NGA50 = [NG50 corrigé], plutot que le NG50).

Recherche d'un gène d'intérêt (optionnel)

Avec notre nouvel assemblage, nous pouvons y chercher un gène d'intérêt (ou toute autre séquence).

Voici un exemple de gène à rechercher:

```
>b1068
GTGAAAAAATTACGTATCGGCGTAGTGGGATTAGGTGGCATTGCGCAAAA
```

```
AGCGTGGTTACCGGTGCTGGCGGCAGCGTCTGACTGGACGTTACAAGGAG
CCTGGTCGCCTACGCGCGCGAAAGCCCTGCCAATTTGTGAAAGCTGGCGC
ATTCCTTATGCCGATTCGTTATCCAGCCTTGCCGCCAGTTGCGATGCGGT
TTTTGTGCATTCCAGCACCGCCAGCCACTTTGACGTGGTCAGTACGTTAC
TCAATGCGGGGGTACATGTCTGTGTCGATAAACCGCTGGCAGAAAATCTG
CGCGATGCTGAACGGCTGGTGGAACTGGCGGCGCGGAAAAAACTGACGTT
GATGGTCGGTTTTAACCGTCGTTTCGCACCACTCTACGGTGAGTTAAAAA
CGCAACTCGCCACCGCAGCCTCGCTAAGAATGGATAAACATCGTAGCAAT
AGCGTCGGGCCACACGATCTTTATTTCACGTTGCTGGATGATTATCTGCA
TGTGGTGGATACCGCGCTGTGGTTGTCGGGCGGCAAAGCCTCTCTGGATG
GCGGTACGCTACTGACTAACGACGCTGGCGAAATGCTGTTTGCCGAGCAC
CATTTTTCGGCTGGTCCTTTGCAGATCACCACCTGTATGCATCGCCGTGC
CGGAAGTCAGCGTGAAACCGTGCAGGCCGTGACTGACGGTGCGCTCATCG
ACATTACGGATATGCGCGAATGGCGTGAGGAGCGCGGGCAGGGCGTAGTG
CATAAACCGATTCCTGGTTGGCAGAGTACGCTTGAGCAACGTGGGTTTGT
CGGCTGTGCGCGGCACTTCATTGAATGTGTGCAAAACCAGACAGTTCCGC
AAACCGCCGGCGAACAGGCCGTGCTGGCGCAACGTATCGTTGACAAGATC
TGGCGCGATGCGATGAGTGAATAA
```

Il s'agit d'un gène peu caractérisé contenant un éventuel facteur de virulence.

Vous pouvez le rechercher dans votre assemblage avec Gassst, qui s'utilise un peu comme Blast:

`Gassst -d` *`genome.fasta`* `-i` *`gene.fasta`* `-o` *`résultat.sam`* `-p` *`identité`* *`[-m format]`*

Vous pouvez prendre une identité de 90%, par exemple. Par défaut, le format du fichier de sortie ressemble au format SAM. Vous pouvez utiliser l'option `-m 0` pour avoir un format un peu plus lisible.

Trouvez-vous ce gène ? Complètement ? Sur un locus seulement ?

Alignement des reads sur le génome de référence (optionnel)

En pratique, si vous séquencez une nouvelle souche de *E. coli* ou tout autre organisme avec un génome de référence, il n'est pas forcément nécessaire de faire un assemblage *de novo*. Vous pouvez aligner vos reads sur le génome de référence. A la condition que le génome soit de référence soit très proche de votre souche et que cette référence soit de bonne qualité !

Pour cette étape, nos allons mapper les reads sur un génome de *E. coli* de référence, de manière à vous montrer le niveau de couverture que l'on obtient avec une référence.

Le génome de référence de *E. coli* est situé dans le répertoire *DATA/Reference*. Il est nommé `NC_000913.fna`. Nous utiliserons bowtie2 pour aligner les reads au format FASTQ sur la référence.

Avant d'aligner les reads, vous devrez créer un fichier d'index pour la référence. Utilisez `bowtie2-build` pour cela. La commande demande le fichier à indexer (dans ce cas, le fichier FASTA), et un nom pour les fichiers de sortie. Nous utiliserons le même nom pour les fichiers d'index et le génome de référence (~ 15 s).

```
bowtie2-build ~/Documents/DATA/Reference/NC_000913.fna NC_000913
```

Ceci devrait prendre quelques secondes et vous devriez voir 6 fichiers apparaître, qui portent tous l'extension `.bt2`. Ces fichiers constituent l'intégralité de l'index pour la séquence de référence et seront utilisés par *`bowtie2`* durant l'étape d'alignement.

Alignez vos reads décompressées avec cet index en utilisant *bowtie2*. Pour cela, vous aurez besoin (au minimum) de donner un fichier d'index (indiqué avec le paramètre `-x`) et l'endroit du fichier de reads à aligner. Dans notre cas, nous utiliserons le paramètres 1 et 2 , pour indiquer que nous avons des reads pairés et donner les fichiers d'entrée, et le paramètre `-S` indiquant que nous voulons un sortie au format SAM, que nous regardons dans la prochaine étape (~ 2 min 30 s).

```
bowtie2 -x NC_000913 -1 [paired_reads1.fastq] -2 [paired_reads2.fastq]
-S MiSeq_Ecoli_MG1655_vs_NC_000913.sam
```

Ceci prendra plusieurs minutes. Lorsque l'alignement sera complet, vous verrez quelques statistiques s'afficher sur votre terminal, et le fichier de sortie apparaître.

Vous pourrez voir le fichier SAM dans l'outil de visualisation Tablet. Vous pouvez lancer cet outil en tapant `tablet` dans votre console. Sélectionnez "Open a New Assembly", chargez le fichier SAM et le fichier FASTA du génome de référence. Le programme met plusieurs minutes à charger l'assemblage et les séquences. Vous devrez sélectionner un des contigs dans le menu, à gauche, afin de visualiser l'alignement. Tablet vous permet de naviguer et de zoomer sur l'alignement. Vous verrez ainsi que la couverture du génome en lecture est relativement profonde. Notre autre logiciel de visualisation de génome : IGB (non installé sur cette VM)

Aller plus loin : Faire de l' « orfing » et annotation, comparaison de génomes via « Mauve », etc.

Par exemple télécharger et installer l'application Mauve (https://darlinglab.org/mauve/mauve.html). Faire un alignement entre le génome de référence et votre génome assemblé. Ou un alignement entre vos 2 assemblages réalisés avec les 2 assembleurs. Visualiser les différences et similarités. Ensuite pour faire de l'orfing (détection des gènes dans un génome), il existe plusieurs logiciels. Vous pouvez par exemple installer et utiliser « prodigal » (https://github.com/hyattpd/Prodigal)