Assemblage de novo avec Spark

Ronan Bocquillon, Stéphane Gazut, Lorène Allano

Assemblage de novo avec Spark

Ronan Bocquillon[1,2], Stéphane Gazut[1], Lorène Allano[1]

[1] CEA, LIST, Laboratoire d'Analyse de Données et Intelligence des Systèmes, 91191 Gif-sur-Yvette
[2] Université de Tours, LI (EA 6300), ROOT (ERL CNRS 6305), 37200 Tours

ronan.bocquillon@univ-tours.fr, {stephane.gazut,lorene.allano}@cea.fr

Mots-clés : *big data, graphes, algorithmes distribués, Spark, Hadoop, génomique, séquençage à haut débit (Next-Generation Sequencing).*

1 Introduction

Les récentes avancées en biologie moléculaire et l'avènement des méthodes de séquençage à haut débit ont rendu possible la lecture, l'analyse et la réutilisation d'une très grande partie de l'information présente dans le génome. Malheureusement, notre capacité à analyser l'immense masse de données générée par ces nouvelles technologies de séquençage est aujourd'hui limitée par nos moyens de calcul. Dans ces travaux, nous nous intéressons plus particulierèrement aux problématiques de l'*assemblage de novo*, dont l'objectif est de reconstruire une séquence ADN à partir d'un ensemble de fragments issus de cette même séquence. Nous mettons en évidence les avantages d'utiliser les outils (notamment méthodologiques) de la communauté "big data" pour résoudre ce problème sur des instances réelles de très grande taille.

Le papier s'organise comme suit. Dans la Section 2, nous décrivons le problème auquel nous nous sommes interessés et l'une des approches communément suivies pour le résoudre. Dans la Section 3, nous discutons des résultats numériques prometteurs que nous avons pu obtenir en implémentant cette approche à l'aide du framework Apache Spark.

2 Description du problème et approche de résolution

Le *génome* de tout individu peut être vu comme une séquence de caractères sur l'alphabet $\{\texttt{A}, \texttt{C}, \texttt{G}, \texttt{T}\}$. Chaque caractère est appelé *base*. Les séquenceurs à haut débit (*Next Generation Sequencing*) permettent d'obtenir ("séquencer") rapidement et à faible coût un grand nombre de fragments ADN appelés *lectures*. Chaque lecture est une sous-chaîne (d'environ 150 bases) du génome "cible" (plus de 3 milliards de bases chez l'être humain). Chaque base est couverte par plusieurs lectures (une quarantaine dans notre cas). L'assemblage consiste à "reconstruire" un génome (ou une région d'un génome) à partir d'un ensemble de lectures – en exploitant les recouvrements qui existent entre elles. Par exemple, on peut retrouver la séquence `AACGTCCGT` à partir des trois lectures `AACGT`, `CGTCC` et `TCCGT`.

L'une des principales approches pour résoudre ce problème s'appuie sur le graphe dit de de Bruijn. Les sommets représentent les chaînes de caractères de longueur fixe k, appelées k-mers, extraites des lectures. Les arcs représentent les recouvrements parfaits de longueur $k-1$ entre ces k-mers. Il nous faut alors rechercher, dans ce graphe, le chemin correspondant au génome cible [5]. Un exemple est donné dans la Figure 1. Notez que le choix de k est décisif. Une valeur trop petite mènera à un graphe trop dense, avec de nombreux points de choix. Une valeur trop grande empêchera la bonne détection des recouvrements.

D'un point de vue théorique, ce problème d'assemblage peut donc s'apparenter à un problème de postier chinois. En pratique, il est beaucoup plus dur que cela. Il faut en effet tenir compte des erreurs de lecture, des régions non couvertes, de la structure à double brins de l'ADN, *etc.* On fait alors rapidement face à des problème NP-difficiles qu'il faut résoudre sur des graphes

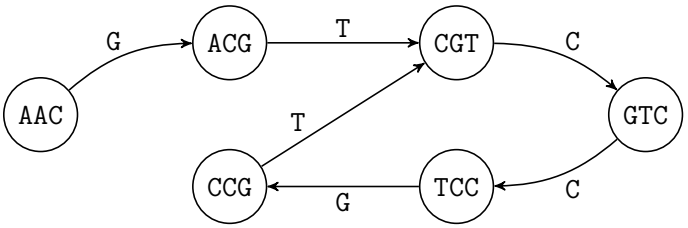

FIG. 1 – Graphe de de Bruijn correspondant aux lectures {AACGT, CGTCC, TCCGT} pour $k = 3$; solution possible : le chemin (AAC, ACG, CGT, GTC, TCC, CCG, CGT), c'est-à-dire la séquence AACGTCCGT.

de taille gigantesque (de 10^7 à 10^9 sommets/arcs !) [4]. La reconstruction est particulièrement délicate dans les régions du génome où des séquences de longueur supérieure à k se répètent de nombreuses fois. Plus de détails seront donnés durant la présentation.

3 Résultats

Nous avons implémenté différents algorithmes de compression et de nettoyage du graphe de de Bruijn à l'aide du framework de calcul distribué Apache Spark.

Ces algorithmes cherchent à supprimer les sommets n'ayant qu'un seul arc entrant et un seul arc sortant (voir les sommets ACG, GTC, TCC et CGG de la Figure 1). Les super-arcs ainsi obtenus (voir (AAC–[GT]–CGT) et (CGT–[CCGT]–CGT)) correspondent aux régions correctement assemblées du génome, nommées *contigs*. Il s'agit d'une étape importante de la reconstruction, réalisée par tous les "assembleurs" de l'état de l'art. Il est important aussi de noter que le graphe résultant est de taille plus raisonnable. La suite du processus peut donc être abordé avec des méthodes plus classiques, issues de la recherche opérationnelle [2].

Comme nous le montrerons au cours de la présentation, les résultats numériques que nous avons obtenus sont encourageants. Les contigs sont de bonne qualité et l'utilisation d'outil "big data" comme Spark permettent de passer à l'échelle [1]. Nous avons exclusivement travaillé sur des jeux de données réelles [3] (*cf.* Tableau 1). Nous essayons actuellement de traiter le génome humain complet (plus de 5 milliards sommets !).

Jeu de données	Staphy	Rhodob	Chr14
# de sommets initial	39 655 624	59 852 043	472 030 134
# d'arcs initial	39 566 297	59 966 143	475 525 075
# de sommets final	11 707	24 753	725 638
# d'arcs final	8 378	17 685	1 046 338

TAB. 1 – Aperçu des jeux de données utilisés.

Références

[1] Anas Abu-Doleh and Ümit V. Çatalyürek. Spaler : Spark and graphx based de novo genome assembler. In *IEEE International Conference on Big Data*, 2015.

[2] E. Bourreau, A. Chateau, C. Dallard, and R. Giroudeau. Une modèlisation par contraintes de graphe pour résoudre l'échafaudage de génome. In *ROADEF*, 2017.

[3] Genome Assembly Gold-Standard Evaluations. http://gage.cbcb.umd.edu/.

[4] P. Medvedev, K. Georgiou, G. Myers, and M. Brudno. Computability of models for sequence assembly. In *International Workshop on Algorithms in Bioinformatics*, 2007.

[5] J. R. Miller, S. Koren, and G. Sutton. Assembly algorithms for next-generation sequencing data. *Genomics*, 2010.