Méthodes de *mapping* de *reads* avec indexation des *reads*

Pierre Morisse
Master IGIS spécialité ITA
2° année
2015 - 2016

Introduction

Depuis le milieu des années 2000 et le développement des séquenceurs à très haut débit (*Next Generation Sequencing*), la biologie doit faire face au traitement d'énormes quantités de données, formées par des millions de très courtes séquences appelées *reads*. Ces *reads* sont utilisés pour le traitement de problèmes de *mapping* ou d'assemblage, et doivent souvent subir une procédure de correction avant utilisation. Philippe et al. ont souligné, dans [2], l'importance de l'indexation des *reads* afin de résoudre ces problèmes, et ont développé un index supportant les 7 requêtes suivantes, pour une séquence f de longueur k donnée :

— Dans quels *reads* f apparaît ?
— Dans combien de *reads* f apparaît ?
— Quelles sont les occurrences de f ?
— Quel est le nombre d'occurrences de f ?
— Dans quels *reads* f n'apparaît qu'une fois ?
— Dans combien de *reads* f n'apparaît qu'une fois ?
— Quelles sont les occurrences de f dans les *reads* où f n'apparaît qu'une fois ?

Nous présentons ici l'état de l'art concernant les technologies de séquençage et les problèmes susmentionnés, ainsi que le problème sur lequel nous nous sommes principalement penchés.

État de l'art

Les tableaux présentés ci-dessous résument brièvement l'état de l'art concernant les technologies de séquençage et les outils existants, utilisant une structure d'index sur les *reads*, permettant de résoudre des problèmes de correction, de $mapping$, ou de traitement des 7 requêtes précédentes.

Technologie	Technique de séquençage	Plateforme	Nombre de $reads$	Longueur	Précision (en %)	Temps	Débit (en Gb)	Coût (en $)	Erreurs
Illumina	Synthèse, basé sur ADN polymérases	HiSeq 2500/1500 MiSeq	3 milliards 17 millions	36 - 100 25 - 250	99 >99	2 - 11 jours 4 - 27 heures	600 8,5	740 000 125 000	Substitutions
Roche	Pyroséquençage	454 GS FLX+ 454 GS Junior	1 million 1 million	700 400	99.997 >99	23 heures 10 heures	0,7 0,4	450 000 108 000	Indels.
ABI Life Technologies	Ligature Détection de protons	5500xl SOLiD Ion Proton Chip I/II	2,8 millions 60 - 80 millions	75 jusqu'à 200	99,99 >99	7 jours 2 heures	180 10 - 100	595 000 243 000	Indels.
Pacific Biosciences	Simple molécule en temps réel	PacBio RS	50 000	3 000 en moyenne	85	2 heures	13	750 000	Indels.
Oxford Nanopore	Exonucléase par Nanopore	GridION MinION	4 - 10 millions 70 000	dizaines de milliers dizaines de milliers	96 70	variable 48 heures	quelques dizaines 0,132	variable 1 000	Indels.

TABLE 1 : Récapitulatif des différentes technologies de séquençage. Attention, Gb signifie ici Gigabases, et non Gigabits.

Structure de données	Erreurs corrigées	Nombre de $reads$ (longueur)	Espace mémoire (en Mo)	Temps (en min)	*reads* corrigés (en %)
Arbre des suffixes	subs.	1 090 946 (70)	1 500	183	88,56
Arbre des suffixes	subs. + indels	977 971 (178)	15 000	28	98,39
Table des suffixes	subs.	1 090 946 (70) 4 639 675 (70)	757 3 210	28 125	94,43
Table des suffixes partielle	subs. + indels	977 971 (178) 2 464 690 (142)	2 000 3 000	15 32	66,76
Table de hachage	subs. + indels	977 971 (178)	8 000	5	92,88
Table de hachage	subs.	2 119 404 (75) 101 548 652 (457 595)	1 437 41 700	23 104	76,65 42,95
Filtres de Bloom	subs. + indels	1 096 140 (101)	11	6	84,38
Graphe de De Bruijn	subs. + indels	33 360 $reads$ longs (2 938) et 2 313 613 $reads$ courts (100)	960	10	85,78

TABLE 2 : Récapitulatif des différentes méthodes de correction des *reads*. La valeur indiquée pour le nombre de *reads* corrigés est une moyenne obtenue à partir de différents ensembles de données.

Structure de données	Erreurs prises en compte	Nombre de *reads* (longueur)	Espace mémoire (en Mo)	Temps (en min)	*reads* mappés (en %)
Table de hachage	subs. + indels	1 000 000 (44)	1 200	331	92,53
Table de hachage	subs.	1 000 000 (100)	20 000	169	90,70

TABLE 3 : Récapitulatif des différentes méthodes de *mapping* de *reads*. La valeur indiquée pour le nombre de *reads mappés* est une moyenne obtenue à partir de différents ensembles de données.

Peu de méthodes sont présentés ici, mais de nombreuses alternatives, n'utilisant pas de structure d'index sur les *reads*, existent et produisent de très bons résultats, aussi bien en espace et en temps, qu'en qualité de *mapping*.

Structure de données	Nombre de *reads* (longueur)	Espace mémoire (en Go)	Temps R1 (en ms)	Temps R2 (en ms)	Temps R3 (en ms)	Temps R4 (en ms)
Table des suffixes modifiée + Table des suffixes modifiée inverse + Table associant k-mer - nombre d'occurrences	42 400 000 (75)	20	16	25	25	0,1
Table de suffixes échantillonnée + 3 vecteurs de bits	42 400 000 (75)	3 - 7	1203	28	1278	28
Table des suffixes échantillonnée + Table auxiliaire d'information sur les *reads* et k-mers	42 400 000 (75)	1 - 4	70	58	70	58

TABLE 4 : Récapitulatif des différentes méthodes permettant de traiter les 7 requêtes. Les requêtes 5-7 sont exclues du comparatif, car non implémentées dans toutes les méthodes de traitement.

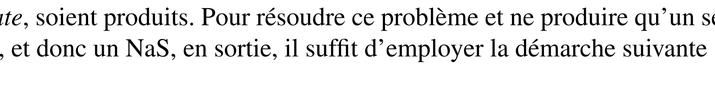
Correction de *reads* longs : Les *reads* NaS

Les nouvelles technologies de séquençage permettent de séquencer des *reads* de plus en plus longs, très utiles pour résoudre des problèmes d'assemblage. posent cependant d'un important taux d'erreur, avoisinant notamment les 30% pour les *reads* séquencés par la plateforme MinION. Comme le montre la T les méthodes de correction classiques ne sont pas adaptées à de tels *reads*, et sont donc très peu efficaces. Une solution alternative pour résoudre ce problèm génération de *reads* dits synthétiques. Ces derniers sont générés via une approche hybride, utilisant des *reads* longs comme *templates* et des *reads* courts dis d'un plus faible taux d'erreur. Les *reads* ainsi synthétisés, appelés *reads* NaS (Nanopore *Synthetic-long*), peuvent atteindre une longueur de 60 000, et s' intégralement et sans erreurs. Nous présentons ici une première méthode permettant de synthétiser de tels *reads*, et la méthode que nous avons développé.

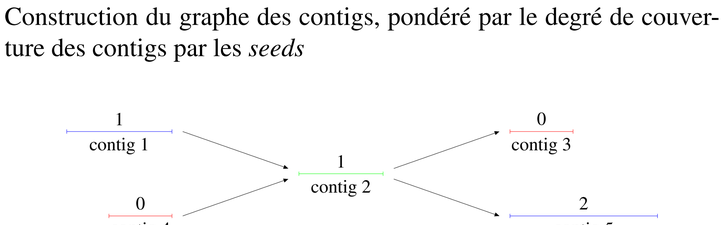
Première méthode

La première méthode de synthèse des NaS [1] aligne les *reads* courts entre eux, et sur les *reads* longs *templates*. Elle repose sur les étapes suivantes :

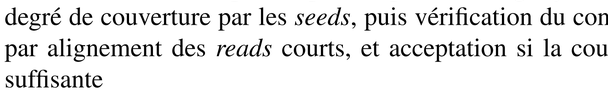
1. Alignement des *reads* courts sur le *template*, pour trouver les *seeds*



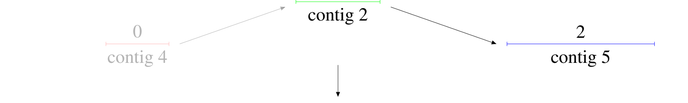
2. Recrutement de nouveaux *reads* similaires aux *seeds*, en alignant les *reads* courts entre eux. Un *read* court est similaire à un *seed* s'ils partagent au moins t k-mers ne se chevauchant pas.



3. Micro-assemblage de l'ensemble de *reads* obtenu

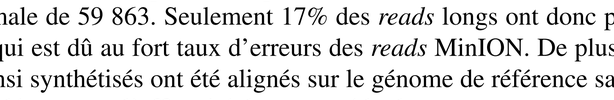


4. Obtention d'un contig

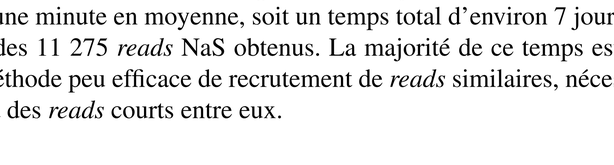


En général, un seul contig est produit , mais il est possible que de mauvais *reads* soient recrutés, et que des contigs erronés, ne devant pas être associés au *template*, soient produits. Pour résoudre ce problème et ne produire qu'un seul contig, et donc un NaS, en sortie, il suffit d'employer la démarche suivante :

1. Obtention de plusieurs contigs



2. Construction du graphe des contigs, pondéré par le degré de couverture des contigs par les *seeds*



3. Sélection du chemin optimal, passant par les contigs ayant le plus haut degré de couverture par les *seeds*, puis vérification du contig produit par alignement des *reads* courts, et acceptation si la couverture est suffisante

Sur un ensemble de 66 492 *reads* longs MinION, et à l'aide de *reads* courts Illumina, cette méthode a permis de produire 11 275 *reads* NaS, d'une longueur maximale de 59 863. Seulement 17% des *reads* longs ont donc produit un NaS, ce qui est dû au fort taux d'erreurs des *reads* MinION. De plus, 97% des *reads* ainsi synthétisés ont été alignés sur le génome de référence sans aucune erreur, démontrant l'efficacité de cette méthode.

Le temps de traitement d'un *read* long, et donc la synthèse d'un *read* NaS, est de moins d'une minute en moyenne, soit un temps total d'environ 7 jours pour la synthèse des 11 275 *reads* NaS obtenus. La majorité de ce temps est utilisée par la méthode peu efficace de recrutement de *reads* similaires, nécessitant l'alignement des *reads* courts entre eux.

Notre méthode

Notre méthode repose sur le même principe que la méthode précédent ne déduit des informations qu'à partir de l'alignement des *reads* courts *reads* longs. Les *reads* courts sont donc alignés sur les *reads* longs *tem* en se fixant un seuil $lmin$, pour récupérer les *reads* :

— Totalement alignés, et servant de *seeds*
— Avec un préfixe de longueur $\geq lmin$ aligné
— Avec un suffixe de longueur $\geq lmin$ aligné

Une fois ces trois ensembles calculés, notre méthode se décompose e étapes :

1. Recrutement de *reads* partiellement alignés, similaires aux *see* d'étendre ces derniers. Un *read* est considéré comme similai *seed* si son préfixe (respectivement son suffixe) correctement alig vauche ce *seed* sur une longueur supérieure ou égale au seuil lm

2. Recrutement de *reads* partiellement alignés afin d'étendre à n les contigs obtenus à l'étape précédente. Un *read* partiellement est ici recruté si son préfixe (respectivement son suffixe) cheva contig sur une longueur inférieure ou égale à un seuil fixé $lmax$

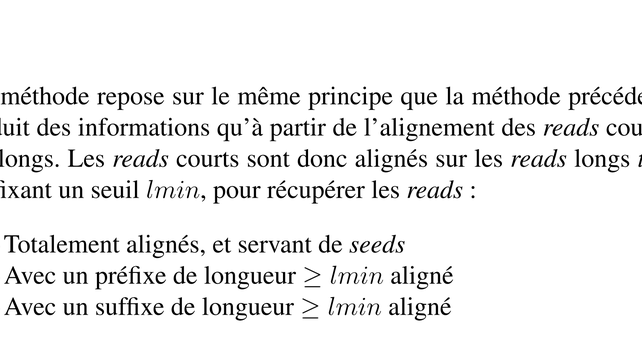


FIGURE 1 : Illustration des deux étapes de notre méthode. Les par en gras des *reads* courts correspondent aux parties correctement gnées sur le *template*. Les parties hachurées du *template* correspond aux zones fortement bruitées de celui-ci.

Nombre de *reads*	Longueur moyenne	Contigs produits par *read*	Longueur moyenne	Précision moyenne	Plus long contig
463	4 078	2,42	623	89,90	3 728 (95
322	9 619	3,17	1 495	87,78	17 797 (9
1 713	1 549	2,89	742	87,94	15 223 (9
2 871	11 583	2,41	4 349	88,83	33 317 (9

TABLE 5 : Résultats obtenus par notre méthode sur différents ensem de données.

Notre méthode permet donc de produire des contigs assez longs et pr semble constituer un prétraitement efficace pour la production de *reads* serait nécessaire d'étudier les contigs produits, afin de voir s'ils se cheva et peuvent ainsi être fusionnés, afin de produire, pour chaque *read* lo unique contig de longueur plus importante, et donc un éventuel *read* N

Références

[1] M.-A. Madoui, S. Engelen, C. Cruaud, C. Belser, L. Bertrand, berti, A. Lemainque, P. Wincker, and J.-M. Aury. Genome assembl Nanopore-guided long and error-free DNA reads. *BMC Genomics*, 1 2015.

[2] N. Philippe, M. Salson, T. Lecroq, M. Leonard, T. Commes, and E. Querying large read collections in main memory : a versatile data str *BMC bioinformatics*, 12(1) :242, 2011.