

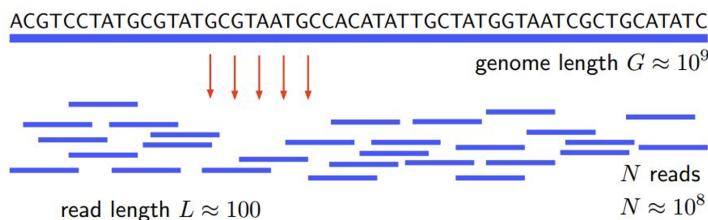
# Travaux pratiques - TP5

## Assembleur glouton

Emmanuelle Becker

### 1 Petite description du problème biologique

Les technologies de séquençage à haut débit sont couramment utilisées pour séquencer les génomes. Cependant, le génome n'est pas séquencé en un seul bloc, mais plutôt par petites sections qui se chevauchent, appelées **lectures**.



Pour trouver la séquence initiale, les lectures doivent être **assemblées**. Le processus d'assemblage est basé sur les chevauchements entre les lectures qui permettront d'effectuer une fusion de ces lectures. Parfois, il n'est pas possible de reconstruire toute la séquence initiale, mais il est possible de reconstruire une longue section qui implique de nombreuses lectures mais qui ne peut pas être étendue plus loin : cette section est appelée **contig**.

Dans ce travail, vous allez programmer un outil pour assembler des lectures :

1. dans la première section, nous allons travailler en supposant que les machines de séquençage sont parfaites et ne produisent pas d'erreur de séquençage ;
2. dans la deuxième section, nous allons considérer une hypothèse plus réaliste et prendre en compte les erreurs potentielles de séquençage.

### 2 Préparation des objets

Nous manipulerons 3 classes :

- une interface appelée **Sequence** ;
- une classe appelée **Read** qui implémente **Sequence** ;
- Une classe appelée **Contig** (rappel : un contig est une fusion de plusieurs reads) qui implémente également **Sequence**.

**Question 1 :** Dans un package appelé **assembly**, téléchargez les différents fichiers fournis avec le sujet.

**Question 2 :** Modifier la classe **Read** de façon à ce que (i) elle implémente correctement l'interface **Sequence** ; et (ii) les attributs de la classe soient correctement cachés (cela va avec l'encapsulation...)

**Question 3 :** Modifier la classe `Contig` pour que (i) elle implémente correctement l'interface `Sequence` ; et (ii) les attributs de la classe soient correctement cachés (cela va dans le sens de l'encapsulation...)

**Question 4 :** Pour vérifier que votre travail est correct, ajoutez le code suivant dans la méthode principale de `Contig`, et complétez-le pour créer un `Contig` avec la première séquence de la liste.

```
1 public static void main(String[] args) throws IOException {
2
3     System.out.println(System.getProperty("user.dir"));
4
5     String filename = "/src/assembly/my_reads.txt" ;
6     File monFichierTexte = new File(System.getProperty("user.dir") +
7         filename);
8
9     //Simple test to verify that the file exists.
10    if (monFichierTexte.exists()){
11        System.out.println("The file " + filename + " is present in the
12            given directory\n");
13    } else{
14        System.out.println("The file " + filename + " is NOT present in the
15            given directory");
16    }
17
18    List<Read> list_reads = new LinkedList<Read>();
19
20    BufferedReader br = new BufferedReader(new FileReader(monFichierTexte));
21    String line;
22    while ((line = br.readLine()) != null) {
23        Read r1 = new Read(line);
24        list_reads.add(r1);
25    }
26    br.close();
27
28    <complete here>
29}
```

**Question 5 :** Ajouter dans chaque classe une méthode `fastaFormat()` qui renvoie une chaîne avec la séquence écrite au format fasta (exactement 60 nucléotides par ligne).

### 3 Assemblage sans erreurs dans les lectures

**Question 6 :** Dans la classe `Contig`, ajoutez une méthode `bestOverlap(Read r)` qui retourne la longueur du meilleur recouvrement entre le contig et le read passé en argument. Nous ne considérerons que le cas où le recouvrement est entre la fin du contig et le début du read (voir exemple ci-dessous).

Contig : AZERTYUIOP      and      Read : YUIOPQS

AZERTYUIOP  
-----YUIOPQS -> best overlap of length 5

**Question 7** : Dans la classe `Contig`, ajoutez une méthode `nextRead(LinkedList<Read> l)` qui trouve la position de la lecture dans la liste avec le plus grand recouvrement avec la séquence du contig. Si la longueur du recouvrement est inférieure à 8, elle est considérée non significative et dans ce cas la méthode renverra -1.

Attention : la méthode doit retourner la **position** du meilleur read. Si le meilleur read se chevauchant est l'élément en position 6, elle doit retourner 6.

**Question 8 :** Dans la classe `Contig`, ajoutez une méthode `fusion(Read r)` qui retournera un nouveau contig qui est la fusion du contig et du read r.

**Question 9 :** Dans la méthode principale de la classe `Contig`, ajoutez des lignes qui effectueront l'assemblage selon le principe suivant :

Algorithme (appele algorithme glouton)

Au debut, le contig est la premiere lecture.

La premiere lecture est retiree de la liste des lectures.

Ensuite, tant que l'on trouve une lecture avec un chevauchement > 8 :

- determiner la position de cette lecture ;
- effectuer une fusion du contig actuel avec cette lecture ;
- supprimer cette lecture de la liste des lectures.

**Question 10 :** Ajoutez quelques lignes pour avoir des informations au fur et à mesure de l'assemblage. Plus précisément, votre processus d'assemblage doit produire le résultat suivant (c'est en fait la bonne réponse, ce qui vous permet de vérifier que votre programme est correct) :

```
Fusion with 37, still 42 reads to assemble... work in process
Fusion with 37, still 41 reads to assemble... work in process
Fusion with 40, still 40 reads to assemble... work in process
Fusion with 0, still 39 reads to assemble... work in process
Fusion with 0, still 38 reads to assemble... work in process
Fusion with 0, still 37 reads to assemble... work in process
Fusion with 0, still 36 reads to assemble... work in process
Fusion with 5, still 35 reads to assemble... work in process
Fusion with 29, still 34 reads to assemble... work in process
Fusion with 33, still 33 reads to assemble... work in process
Fusion with 1, still 32 reads to assemble... work in process
Fusion with 4, still 31 reads to assemble... work in process
Fusion with 4, still 30 reads to assemble... work in process
```

[...]

Contig obtained with 43 reads

```
CCACACCACACCCACACACCCCCACACACACACACACACACACACA
CATCCTAACACTACCCTAACACAGCCCTAACCTAACCTGGCCAACCTGTCTCAACTT
ACCCTCCATTACCCTGCCTCCACTCGTTACCCTGTCCATTCAACCATAACCACTCCGAAC
CACCATCCATCCCTACTTACTACCACTACCCACCGTTACCCTCCAATTACCCATATC
CAACCCACTGCCACTTACCCTACCATTACCCTACCATCCACCATGACCTACTCACCATA
TGTTCTTCTACCCACCATATTGAAACGCTAACAAATGATCGTAAATAACACACACGTGCT
TACCCCTACCACTTTATACCACCCACATGCCATACTCACCCCTACTTGTATACTGATT
TACGTACGCACACGGATGCTACAGTATACCATCTCAAACCTACCTACTCTCAGATTC
```

```
CACTTCACTCCATGGCCCATCTCACTGAATCAGTACCAAATGCACTCACATCATTATG  
CACGGCAGTGCTCAGCGGTCTACCCCTGCCATTACCCATAACGCCATCATTAT  
CCACATTGATATCTATCTCATCGCGGTCCAAATATTGTATAACTGCCCTTAAT  
ACATACGTTATACCACTTTGCACCATATACTTACCACTCCATTATACACTTATGTC  
AATATTACAGAAAAATCCCCACAAAAATCACCTAACATAAAATATTCTACTTTCAAC  
AATAATACATAAACATATTGGCTTGTGGTAGCAACACTATCATGGTATCACTAACGTAAA  
AGTCCTCAATATTGCAATTGCTGAACGGATGCTATTCAGAATATTCGTAAC
```

## 4 Assemblage avec quelques erreurs de séquençage

**Question 11 :** Dans la classe `Read`, ajouter une méthode statique `nearlyEquals(String s1, String s2)` qui retourne `true` si et seulement si `s1` et `s2` ont zéro ou un défaut de concordance.

**Question 12 :** Dans la classe `Read`, utilisez le polymorphisme de surcharge et surchargez la méthode statique `nearlyEquals(String s1, String s2, float perror)`, qui retourne `true` si et seulement si `s1` et `s2` ont moins de `perror` pourcent de mismatch.

**Question 13 :** Dans la classe `Contig`, ajoutez 2 nouvelles méthodes appelées `bestOverlapWithError(<complete here>)` et `nextReadWithErrors(<complete here>)` qui peuvent être appelées pour réaliser un assemblage prenant en compte les erreurs de séquençage.

**Question 14 :** Dans la fonction principale de la classe `Contig`, ajoutez quelques lignes qui vont réaliser un assemblage avec erreurs. Cette fois, le fichier d'entrée sera `my_reads_with_sequencing_errors.txt`.

Vous pouvez vous inspirer de l'assemblage précédent sans erreurs, mais ne modifiez pas le code précédent, ajoutez de nouvelles lignes (nous devons pouvoir vérifier que le premier assemblage fonctionne...).