Instrukcja do ćwiczenia nr 1 z MBI
Asemblacja *de novo* DNA

Wiktor Kuśmirek Robert Nowak

2024Z

1 Wstęp

Nośnikiem informacji genetycznej każdego organizmu żywego jest kwas deoksyrybonukleinowy [1], w skrócie DNA. Jest to wielocząsteczkowy, organiczny związek chemiczny o budowie liniowej (bez rozgałęzień). DNA składa się z dwóch nici - nici kodującej i nici komplementarnej, które biegną w przeciwnych kierunkach i owijają się wokół siebie tworząc tzw. podwójną helisę.

DNA jest polimerem, składa się z monomerów czyli deoksyrybonukleotydów. Monomerami występującymi w kwasie deoksyrybonukleinowym są:

- **a**denina;
- **c**ytozyna;
- **g**uanina;
- **t**ymina;

Do symbolicznego zapisu fragmentu DNA zwyczajowo używane są pierwsze litery monomerów, czyli A, G, C i T. Dodatkowo deoksyrybonukleodyty są do siebie parami komplementarne (wzajemnie uzupełniające):

- **a**denina komplementarna z **t**yminą;
- **g**uanina komplementarna z **c**ytozyną;

Komplementarność jest szczególną cechą DNA, mając odczyt z jednej nici, można w łatwy sposób otrzymać odpowiadający fragment drugiej nici.

Sekwencjonowanie DNA polega na odczycie kolejności deoksyrybonukleodytów w cząsteczce DNA. Istnieje wiele metod sekwencjonowania, jedną z nich jest metoda Sangera, opracowana w 1977 roku i nagrodzona nagrodą Nobla w 1980 roku. Obecnie metoda Sangera nie jest już używana, jednak dała początek kilkunastu innym metodom, które są znacznie szybsze i tańsze. Współczesne metody sekwencjonowania pozwalają na uzyskanie krótkich odczytów pochodzących z losowych miejsc fragmentu DNA. Ponadto dzisiejsze technologie,

[1] ang. deoxyribonucleic acid;

w przeciwieństwie do historycznych, pozwalają na uzyskiwanie sparowanych końców[2] - obiektów składających się z dwóch odczytanych sekwencji i fragmentu, którego sekwencja nie jest znana, ale jego długość należy do przedziału zależnego od użytej metody sekwencjonowania.

Wynikiem sekwencjonowania DNA jest zestaw odczytów o pojedyńczych lub sparowanych końcach. Odczyty te nie są niestety bezbłędne, do najczęściej występujących rodzajów błędów należą:

- Błędy InDel [3], polegające na wstawieniu lub usunięciu losowego nukleotydu z odczytu.
- Błędy podstawienia, czyli zmiana losowego nukleotydu z odczytu na inny monomer. Ilość błędów podstawiania może zostać oszacowana dzięki poziomowi jakości odczytu każdego nukleotydu - oczywiście jeśli zostanie użyta technologia sekwencjonowania dostarczająca takich informacji.

Asemblacja DNA polega na odtworzeniu z uzyskanego podczas sekwencjonowania zestawu odczytów możliwie najpełniejszej i najpoprawniejszej sekwencji nukleotydów. Proces ten nie jest łatwy, przebiega w kilku etapach, spośród których można wyróżnić dwa najważniejsze:

- Zbudowanie wynikowego zbioru kontigów z odczytów. Kontig jest spójnym fragmentem DNA uzyskanym przy użyciu zestawu odczytów i pewnej ich nadmiarowości - odczyty częściowow pokrywają się.
- Zbudowanie skafolda z zestawu kontigów. Skafold składa się z uzyskanych w poprzednim etapie, uporządkowanych kontigów i dziur oddzielających kontigi. Dziury są efektem występowania powtarzających się sekwencji DNA, po wypełnieniu dziur można uzyskać całkowity genom sekwencjonowanego organizmu.

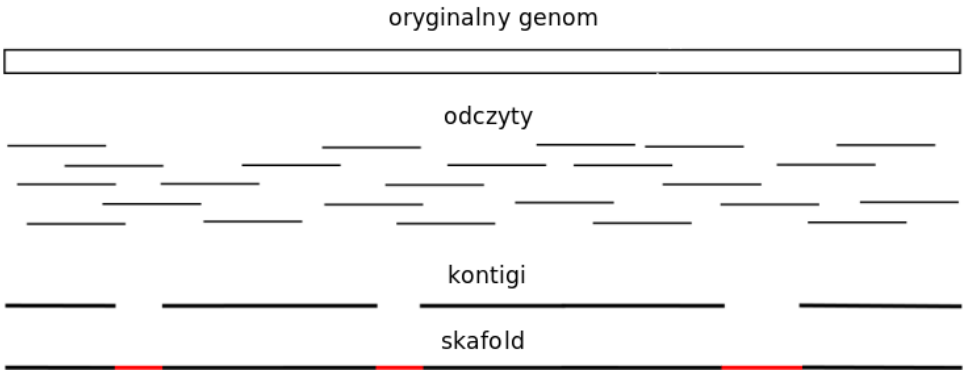
Główną strukturą danych wykorzystywaną do asemblacji *de novo* krótkich odczytów jest graf de Bruijn'a [4], w wierzchołkach przechowuje sekwencje o długości $k-1$, gdzie k to wymiar grafu. Sekwencje te pochodzą z wejściowego zestawu odczytów - dla każdego odczytu tworzone jest k-spektrum, czyli zbiór sekwencji o długości k. Następnie każda sekwencja z k-spektrum dodawana jest do grafu w trzech etapach:

1. $k-1$ pierwszych symboli sekwencji reprezentuje wierzchołek początkowy, który jest dodawany do grafu.
2. $k-1$ ostatnich symboli sekwencji reprezentuje wierzchołek końcowy, który jest dodawany do grafu.
3. Do grafu dodawana jest krawędź łącząca wierzchołek początkowy ($k-1$ pierwszych symboli) i końcowy ($k-1$ ostatnich symboli).

Zarówno dodawanie do grafu de Bruijn'a wierzchołków, jak i krawędzi nie powoduje zwielokrotnienia wierzchołka/krawędzi podczas dodawania identycznego elementu do grafu. Oznacza to, że jeżeli do grafu powinien zostać dodany nowy wierzchołek, który wcześniej istniał w grafie, wówczas wierzchołek nie jest dodawany. Podobnie postępuje się dla krawędzi, jednak wówczas waga krawędzi już występującej w grafie jest inkrementowana. Po zbudowaniu

[2] ang. Paired-end Tags;
[3] ang. **in**sertion **del**etion;
[4] ang. DBG - De Bruijn Graph;

grafu de Bruijn'a z wejściowego zbioru odczytów graf jest korygowany, po czym następuje etap wygenrowania wynikowych kontigów.



Niniejsze ćwiczenie polega na pobraniu genomu referencyjnego bakterii. Następnie z pobranego genomu zostanie wygenerowany zestaw krótkich odczytów aplikacją pIRS (symulacja sekwenatora DNA). Później, zestaw wygenerowanych krótkich odczytów zostanie złożony w kontigi za pomocą aplikacji dnaasm. Finalnie, wyniki asemblacji *de novo* zostaną porównane z wejściowym genomem referencyjnym.

2 Asemblacja de-novo DNA

Instrukcja pokazuje poszczególne kroki na Ubuntu 20.04 w wersji 64 bitowej. Wymagania wstępne to instalacja aplikacji docker.

```
sudo apt-get install docker
```

2.1 Genom referencyjny i aplikacje

Pobierz genom referencyjny badanego w ćwiczeniu organizmu oraz potrzebne do ćwiczenia aplikacje

```
docker pull wkusmirek/pirs
docker pull wkusmirek/dnaasm
docker pull wkusmirek/quast
wget path_to_genome
```

Odnośniki do bakteryjnych genomów referencyjnych są dostępne na stronie przedmiotu MBI. Linie w pliku indeksujemy od zera. Proszę wybrać genom o indeksie numer_indeksu mod 150. Jeżeli w numerze indeksu występują jakieś litery - pomijamy je. Rozważamy numer indeksu jednej osoby z zespołu. Przykładowo dla numeru 01027972 wybieramy genom z linii o indeksie 22 (1027972 mod 150 = 22). Proszę zamieścić w sprawozdaniu informacje o numerze indeksu oraz wybranym pliku z genomem referencyjnym.

2.2 Generowanie odczytów

Wygeneruj zestaw odczytów aplikacją pIRS.

```
docker run --rm -v /tmp:/tmp -w /tmp wkusmirek/pirs pirs simulate
```

W ramach ćwiczenia ustaw następujące parametry aplikacji pIRS:

- pokrycie genomu odczytami: 50x
- średnią odległość pomiędzy sparowanymi końcami: 400 bp
- odchylenie standardowe od odległości pomiędzy sparowanymi końcami: 20
- długość odczytów: 100 bp
- współczynnik błędów podstawienia: 1%

Pozostałe parametry aplikacji pozostaw jako wartości domyślne.
Zapoznaj się z wygenerowanymi plikami oraz z logami aplikacji z konsoli.
Wklej do dokumentacji część logów aplikacji (fragment od liczby opisującej długość badanej sekwencji referencyjnej do ilości nukleotydów zamaskowanych przez algorytm EAMSS):

```
[pIRS] Bases in reference sequences: 1575399
[pIRS] Read pairs simulated: 393849
[pIRS] Bases in reads: 78769800
[pIRS] Coverage: 50.00
[pIRS] Substitution error count: 1037869
[pIRS] Average substitution error rate: 1.318%
[pIRS] Insertion count: 346
[pIRS] Deletion count: 861
[pIRS] Average insertion rate: 0.00044%
[pIRS] Average deletion rate: 0.00109%
[pIRS] Average insertion length: 1.05
[pIRS] Average deletion length: 1.03
[pIRS] Fragments affected by GC bias: 12.96%
[pIRS] Bases masked by EAMSS algorithm: 0
```

Odpowiedz w sprawozdaniu na następujące pytania:

- **Ile zostało wygenerowanych odczytów? Jakiej długości?**
- **Oblicz wygenerowaną głębokość pokrycia genomu odczytami. Czy wynik jest przybliżony do zakładanego poziomu 50x?**
- **W jaki sposób można znaleźć odczyty, które zawierają błędy?**
- **Odczytaj z wygenerowanych plików odległośći pomiędzy sparowanymi odczytami. Czy wartości zgadzają się z ustawianymi parametrami aplikacji?**

2.3 Asemblacja de novo

Przeprowadź asemblacje *de novo* aplikacją dnaasm

```
docker run --rm -v /tmp:/tmp -w /tmp wkusmirek/dnaasm dnaasm -assembly
```

W ramach ćwiczenia ustaw następujące parametry aplikacji dnaasm:

- wymiar grafu de Bruijn'a: 55
- długość genomu, liczba nukleotydów w badanej sekwencji DNA bakterii
- średnia odległość pomiędzy sparowanymi odczytami: 400 bp
- odchylenie standardowe od odległości pomiędzy sparowanymi odczytami: 20
- próg wagi krawędzi w grafie de Bruijn'a: 5 (parametr -single_edge_counter_threshold)

Pozostałe parametry aplikacji pozostaw jako wartości domyślne. Dodatkowo, w aplikacji będziesz musiał ustawić parametry i1_1, i1_2, output_file_name

Zapoznaj się z wygenerowanymi plikami wynikowymi oraz z logami aplikacji znajdującymi się w pliku /tmp/dnaasm/dnaasm_calc_0.log. Wklej do dokumentacji część logów aplikacji (fragment od liczby wynikowych sekwencji DNA do parametru N50):

```
[2018-Feb-15 18:17:36.943130] [info] - num of sequences: 274
[2018-Feb-15 18:17:36.943166] [info] - sum: 3122178
[2018-Feb-15 18:17:36.943175] [info] - max: 214903
[2018-Feb-15 18:17:36.943185] [info] - average: 11394.810547
[2018-Feb-15 18:17:36.943193] [info] - median: 216.000000
[2018-Feb-15 18:17:36.943201] [info] - N50: 53050
```

Odpowiedz w sprawozdaniu na następujące pytania:

- **Czy suma długości wygenerowanych sekwencji jest w przybliżeniu równa długości badanego genomu? Dlaczego?**
- **Czy plik z sekwencjami wynikowymi jest w formacie FASTA czy FASTQ?**
- **Czy możliwa jest konwersja pliku w formacie FASTA na FASTQ? Jaką informacje należy wówczas sztucznie wygenerować?**
- **Czy możliwa jest konwersja pliku w formacie FASTQ na FASTA? Jaka informacja przy takiej konwersji zostaje utracona?**

2.4 Sprawdzenie wyników

Porównaj otrzymane wyniki z genomem referencyjnym, z którego były generowane odczyty.

```
docker run --rm -v /tmp:/tmp -w /tmp wkusmirek/quast quast.py -R ref.fa contigs.fa
```

Zapoznaj się z logami i wynikami aplikacji QUAST. Wklej do dokumentacji zawartość pliku quast_results/latest/report.txt

Odpowiedz w sprawozdaniu na następujące pytania:

- **Czy asemblacja *de novo* genomu pozwoliła uzyskać satysfakcjonujące wyniki? Dlaczego?**
- **Czym są translokacje w genomie? Czy aplikacja QUAST dostarcza informacji o liczbie translokacji w wynikach asemblacji *de novo* względem genomu referencyjnego? Jeśli tak, to skąd mogą być takie informacje odczytane?**

3 Zadanie implementacyjne

Proszę zapoznać się z tematyką par GC(*GC-content*). Proszę odczytać z aplikacji QUAST zawartość par GC w badanym podczas ćwiczenia genomie referencyjnym. Następnie proszę zaimplementować prosty skrypt umożliwiający odczytać zawartość par GC w pliku w formacie FASTA. Proszę wykorzystać biblioteki dedykowane do przetwarzania danych genomowych, np.:

- SeqAn;
- Biopython;
- BioJava.

Proszę porównać wyniki dostarczane przez aplikacje QUAST oraz zaimplementowany sprypt.