Jakub Cieplucha 230331

Wstęp do bioinformatyki

Zadanie 1

1.Odnośnik do repozytorium z kodem źródłowym.

https://github.com/JakubCieplucha/Bioinformatyka/tree/zadanie1

2. Analiza złożoności obliczeniowej czasowej i pamięciowej.

2.1 Funkcja dotPlot

```
1  function [r] = dotPlot(genX,genY)
2  n= length(genX);
3  m= length (genY);
4  for i =1 : n
5      for j =1 : m
6            if genX(i) == genY(j);
7                r(i,j)=1;
8            else
9                r(i,j)=0;
10           end
11     end
12 end
```

a)Czasowa złożoność obliczeniowa

-2 podstawienia (linie 2-3)

- 1 podstawienie licznika pętli (linia 4)

-n inkrementacji licznika pętli (linia 4)

-n+1 sprawdzeń warunku pętli (linia 4)

-n powtórzeń pętli wewnętrznej (linie 5-12)

- 1 podstawienie licznika pętli (linia 5)
- m inkrementacji licznika pętli (linia 5)
- m+1 sprawdzeń warunku pętli (linia 5)
- m powtórzeń ciała pętli (linie 6 – 12)
    - 1 sprawdzenie zgodności liter (linia 6)
    - 1 podstawienie wartości macierzy kropkowej (linia 7 lub 9)

2 + 1 + n + (n+1) + n * (1 + m + m + 1 + m * (1 + 1)) =

= 4 (nm + n + 1)

b)Przestrzenna złożoność obliczeniowa

nm + n + m + 4

2.2 Funckcja checkSequence

```
function [check]= checkSequence(s1,s2)
n= numel(s1);
m= numel (s2);
for i=1 : n
    switch s1(i)
        case 'A'
            check=1;
        case 'T'
            check=1;
        case 'G'
             check=1;
        case 'C'
             check=1;
        otherwise
            check=0;
            return
    end
end
for l=1:m
       switch s2(l)
        case 'A'
            check=1;
        case 'T'
            check=1;
        case 'G'
             check=1;
        case 'C'
             check=1;
        otherwise
            check=0;
            return
        end
end
```

a) Czasowa złożoność obliczeniowa

- 2 podstawienia (linie 2-3)
- 1 podstawienie licznika pętli (linia 4)
- n inkrementacji licznika pętli (linia 4)
- n + 1 sprawdzeń warunku pętli (linia 4)
- n powtórzeń pętli wewnętrznej
    - 1 sprawdzenie warunku (linia 5)
    - 1 podstawienie
- 2 podstawienia (linie 2-3)
- 1 podstawienie licznika pętli (linia 4)
- m inkrementacji licznika pętli (linia 4)
- m + 1 sprawdzeń warunku pętli (linia 4)
- m powtórzeń pętli wewnętrznej
    - 1 sprawdzenie warunku (linia 5)
    - 1 podstawienie

$$2 + 1 + n + (n + 1) + n * (1 + 1) + 2 + 1 + m + (m + 1) + m * (1 + 1) = 4(n + m + 2)$$

b)Przestrzenna złożoność obliczeniowa

- 2 zmienne określające sekwencje wejściowe
- 2 zmienne przechowujące rozmiar sekwencji wejściowych
- 2 zmienne przechowujące liczniki pętli
- 1 zmienna określająca wynik

7

2.3 Funkcja myFilter

```
function [filteredMatrix,row,column] =
myFilter(matrix,window,error)
filteredMatrix = zeros(size(matrix));
[row,column]=size(matrix);
c=window-1;
d=row- window + 1 ;
e=column - window + 1;
for i = 1:d
    for j = 1:e
            a=i+window-1;
            b=j+window-1;
           check= diag(matrix(i:a,j:b));
           if sum(check)>=(window-error)
           for r=0:c
               filteredMatrix(i+r,j+r)=matrix(i+r,j+r);
           end
           end
    end
end
end
```

a) Czasowa złożoność obliczeniowa
   - Utworzenie macierzy o rozmiarach a*b (linia 2)
   - 5 podstawienia (linia 3-6)
   - 1 podstawienie licznika pętli (linia 7)
   - d inkrementacji licznika pętli (linia 7)
   - d + 1 sprawdzeń warunku pętli (linia 7)
   - d powtórzeń pętli wewnętrznej (linie 8-18)
     - ✓ 1 podstawienie licznika pętli (linia 9)
     - ✓ e inkrementacji licznika pętli (linia 9)
     - ✓ e + 1 sprawdzeń warunku pętli (linia 9)
     - ✓ e powtórzeń pętli wewnętrznej (linie 9-17)
       - 3 podstawienia (linie 9-11)
       - 1 sprawdzenie warunku (linia 12)
       - 1 podstawienie licznika pętli (linia 13)
       - c inkrementacji licznika pętli (linia 13)
       - c + 1 sprawdzeń warunku pętli (linia 13)
       - c powtórzeń pętli wewnętrznej (linie 14-15)

- 1 podstawienie (linia 14)

a*b + 5 + 1 + d + (d+1) + d * [1 + e + (e+1) + e (3 + 1 + c + (c+1) + c (1))] =

3cde + 7de + 4d + ab + 7

b)Przestrzenna złożoność obliczeniowa

- row*colum – rozmiar macierzy wejściowej oraz przefiltrowanej

-3 zmienne przechowujące liczniki pętli

- 10 dodatkowych zmiennych

2*row*column + 13

2.4 Funkcja parseFasta

```
function [s]= parseFasta (text)
a=strsplit(text,'>');
a(cellfun('isempty', a)) = [];
[m,n]=size(a);
number=0;
s= struct ('identifier',[],'sequence',[]);
for i=1:n
number=number+1;
k=char(a(i));
c=strsplit(k,'\n');
s(number).identifier = c(1);
sequence = char(c(2:end));
[row,column]=size(sequence);
element=0;
for i=1:row
for j=1:column
element=element +1;
sekwencja(element)=sequence(i,j);
end
end
sekwencja=strtrim(sekwencja);
s(number).sequence=sekwencja;
end
end
```

a) Czasowa złożoność obliczeniowa

- 1 podstawienie (linia 2)
- 1 operacja (linia 3)
- 2 podstawienia (linia 4)
- 1 podstawienie (linia 5)

- 1 inicjalizacja struktury (linia 6)
- 1 podstawienie licznika pętli (linia7)
- n inkrementacji licznika pętli (linia 7)
- n + 1 sprawdzeń warunku pętli (linia 7)
- n powtórzeń pętli wewnętrznej (linie 8-23)
    - 7 podstawienia (linie 8-14)
    - 1 podstawienie licznika pętli (linia 15)
    - row inkrementacji licznika pętli (linia 15)
    - row+ 1 sprawdzeń warunku pętli (linia 15)
    - row powtórzeń pętli wewnętrznej (linie 16-20)
        - 1 podstawienie licznika pętli (linia 16)
        - column inkrementacji licznika pętli (linia 16)
        - column + 1 sprawdzeń warunku pętli (linia 16)
        - column powtórzeń pętli wewnętrznej (linie 17-18)
            - ✓ 2 podstawienia (linie 17-18)
- 2 podstawienia (linie 21-22)

1 + 1 + 2 + 1 + 1 + 1 + n + (n + 1) + n * [7 + 1 + row + (row + 1) + row * (1 + column + (column + 1) + column * (2))] + 2 = 4 ((n* row) + (column * n * row)) + 11 * n + 10

b)Przestrzenna złożoność obliczeniowa

-c * d = rozmiar wejściowy text

-2* number = rozmiar struktury

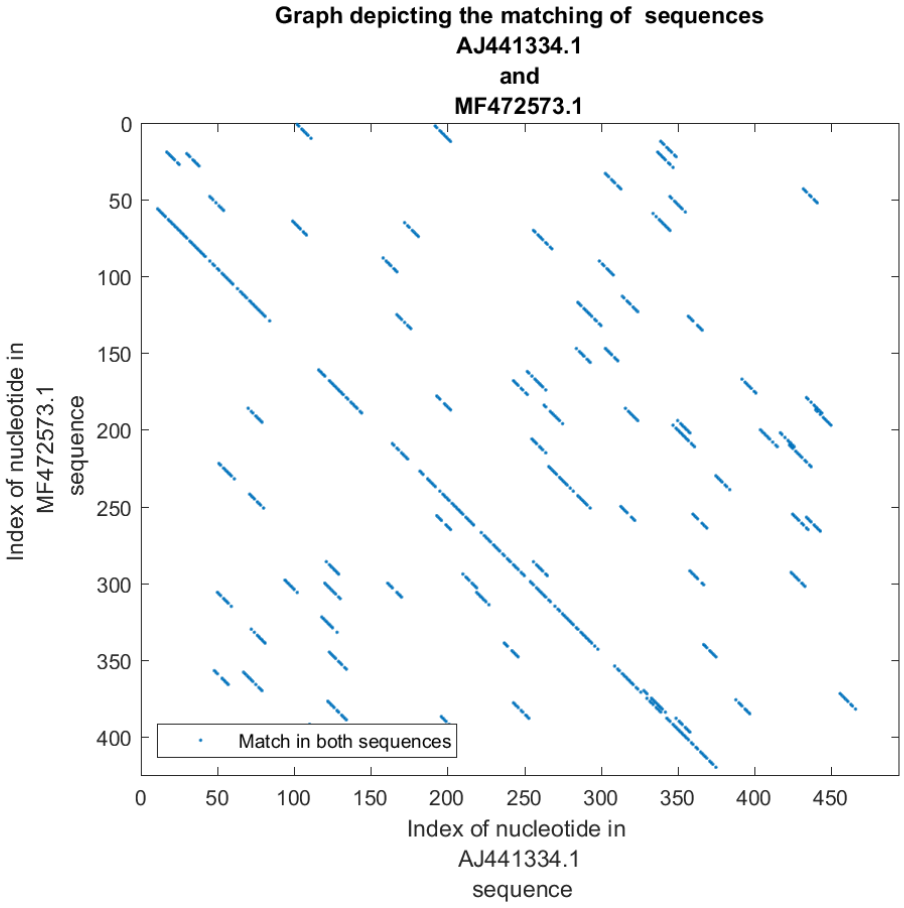
-3 zmienne przechowujące liczniki pętli

-11 dodatkowych zmiennych

c * d + 2 * number +14

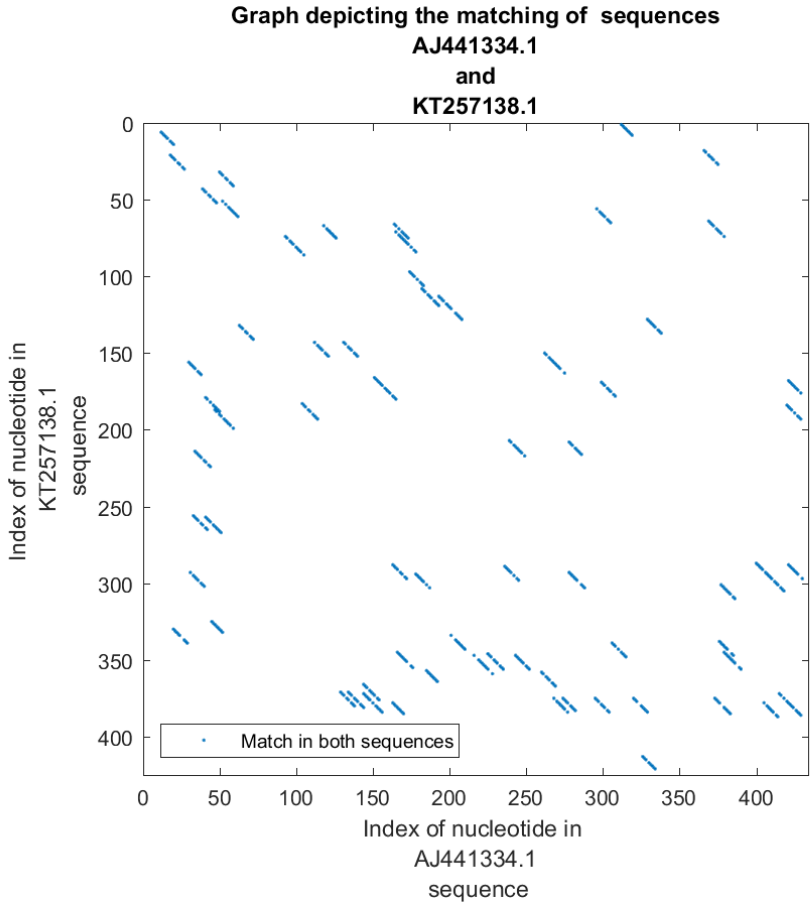
3.Porównanie przykładowych par sekwencji ewolucyjnie

3.1 Powiązanych



Wykres kropkowy, odfiltrowany dla parametrów okna = 10 oraz błędu = 2. Można zauważyć silne podobieństwo pomiędzy sekwencjami, sugerujące występowanie wspólnego przodka. Widać jedną przekątną, która zawiera bardzo dużo dopasowanych fragmentów. Jednakże przerwy wskazują na wystąpienie szeregu substytucji. Istnieją fragmenty, które uzupełniałyby ową przekątną lecz są w stosunku do niej przesunięte, co wskazuje na wystąpienie insercji względnie delecji. Owa przekątna jest również przesunięta względem głównej przekątnej wykresu, co świadczy o wystąpieniu insercji.

3.2 Niepowiązanych



Wykres kropkowy, odfiltrowany dla parametrów okna = 10 oraz błędu = 2. Pomimo restrykcyjnych parametrów dopasowania występuje szereg podobieństw w obu sekwencjach. Istnieje możliwość występowania wspólnego przodka. Zaszło wiele substytucji, gdyż istnieją rozległe przerwy pomiędzy dopasowanymi fragmentami znajdującymi się na jednej przekątnej. Duże rozprzestrzenienie w jakim znajdują się dopasowane sekwencje wskazuje na wielokrotne insercje bądź delecje.