Metody bioinformatyki (MBI)
Dokumentacja wstępna projektu

Michał Andrzejczak Karol Piczak Andrzej Smyk

23 marca 2015

1 Temat projektu

Dopasowywanie sekwencji za pomocą algorytmów Gotoha oraz Altschula - Ericksona. Porównanie z inną dostępną implementacją (np. z pakietem Biostrings w języku R).

2 Cel projektu

Podstawowym zadaniem jest implementacja dwóch algorytmów służących do badania podobieństwa dwóch sekwencji np. nukleotydów: algorytmu Gotoha oraz Altschula - Ericksona. Oba korzystają z afinicznej funkcji kary za przerwę.

Następnie, za pomocą zaimplementowanych algorytmów przeprowadzonych zostanie szereg prób uliniowienia dla sekwencji o różnych długościach. Dla każdej próby zostanie zmierzony czas w jakim dopasowanie zostało ukończone. Na tej podstawie będziemy w stanie empirycznie zweryfikować zależność złożoności obliczeniowej algorytmów względem długości dopasowywanych sekwencji.

Pomiary czasów wykonania zostaną powtórzone dla jednego z istniejących pakietów umożliwiających dopasowywanie sekwencji parami, np. pakietu Biopython [1] lub Biostrings [2]. Pozwoli to na porównanie względnej złożoności czasowej naszej implementacji z rozwiązaniami gotowymi.

3 Funkcjonalność rozwiązania

Projekt zostanie zrealizowany jako aplikacja konsolowa lub okienkowa z prostym GUI. Podstawowa funkcjonalność jaka powinna się w nim znaleźć to:

- Wczytanie oraz sparsowanie danych z pliku tekstowego lub podanych bezpośrednio przez użytkownika: program pobierze dwie sekwencje nukleotydów oraz opcjonalnie wartości do systemu punktacji dla identycznych pozycji, różnych pozycji oraz kar za rozpoczęcie i kontynuację przerwy.
- Uliniowienie obu sekwencji za pomocą wybranego przez użytkownika algorytmu: wyznaczenie macierzy dopasowania oraz na ich podstawie najbardziej optymalnego dopasowania.

- Wydrukowanie na standardowe wyjście lub zapis do pliku wyniku dopasowania, z oznaczeniem przerw oraz zgodnych i niezgodnych par nukleotydów z obu sekwencji.

4 Dopasowanie par sekwencji

Dopasowanie pary sekwencji wykonuje się przy założeniu, że są one homologiczne, tj. wyewoluowały od jednego przodka. Różnice na poszczególnych pozycjach w obu sekwencjach mogą być wynikiem mutacji. Przerwy w sekwencjach sugerują zajście delecji lub insercji w jednej lub w obu sekwencjach.

Afiniczna funkcja kary za przerwę jest najczęściej wykorzystywanym sposobem punktacji przerw. Za je stosowaniem przemawia fakt, iż insercje i delecje mają charakter blokowy, tj. istnieje większe prawdopodobieństwo wystąpienia jednej dużej przerwy, niż kilku mniejszych, znajdujących się nieodległym sąsiedztwie.Kara za utworzenie nowej przerwy jest równa g_{open}, zaś kara za jednostkowe wydłużenie istniejącej przerwy wynosi g_{ext}. Całkowita kara za przerwę o długości l wynosi $W(l) = g_{open} + g_{ext}(l-1)$

Algorytm Gotoha [3] pozwala na znalezienie jednego, optymalnego dopasowania pomiędzy dwiema sekwencjami. Algorytm Altschula - Ericksona[4], który pozwala na znalezienie wszystkich optymalnych sekwencji, jest w rzeczywistości modyfikacją algorytmu Gotoha. Właściwa różnica pomiędzy nimi dwoma, polega na różnych sposobach zapamiętywania "ścieżek", pozwalających później na wyznaczenie najbardziej optymalnego dopasowania (lub wszystkich optymalnych dopasowań, jak w algorytmie A–E).

W obu algorytmach konieczne jest utworzenie w sumie 3 macierzy kosztu:

M - elementy $M_{i,j}$ tej macierzy, stanowią ocenę najlepszego dopasowania kończącego się na pozycjach i i j odpowiednio pierwszej sekwencji a i drugiej sekwencji b,

I - macierz najlepszego dopasowania zakończonego delecją (dopasowanie kończy się na a_i w pierwszej sekwencji oraz znakiem przerwy w drugiej),

J - macierz najlepszego dopasowania zakończonego insercją (dopasowanie kończy się znakiem przerwy w pierwszej sekwencji oraz resztą b_j w drugiej).

Zależności rekurencyjne dla wymienionych powyżej macierzy można przedstawić następująco:

$$M_{i,j} = max \begin{cases} M_{i-1,j-1} + S(a_i, b_j) \\ I_{i,} \\ J_{i,} \end{cases}$$

$$I_{i,j} = max \begin{cases} M_{i-1,j} - g_{open} \\ I_{i-1,j} - g_{ext} \end{cases}$$

$$J_{i,j} = max \begin{cases} M_{i,j-1} - g_{open} \\ J_{i,j-1} - g_{ext} \end{cases}$$

Oda algorytmy, zaimplementowane przy wykorzystaniu programowania dynamicznego, mają złożoność obliczeniową $O(mn)$, czyli taką samą jak ich odpowiedniki liniową funkcją kary za przerwę. Ponieważ wykorzystują one dodatkowo dwie macierze I oraz J, złożoność pamięciowa jest trzykrotnie wyższa w porównaniu do algorytmów stosujących liniową funkcję kary.

5 Testowanie

Testowanie gotowego rozwiązania będzie składało się z dwóch części:

1. Testów wydajnościowych: ich celem będzie sprawdzenie czasów wykonania obu algorytmów dla sekwencji o różnej długości. Uzyskane czasy pozwolą na empiryczne oszacowanie złożoności czasowej algorytmów oraz porównanie ich z czasami wykonania dostępnych implementacji.

2. Testów poprawnościowych: ich celem będzie weryfikacja, czy oba algorytmy prawidłowo dopasowują obie sekwencje tj. wykrywają zarówno mutacje, jak i insercje oraz delecje. W tym celu przetestujemy ich działanie na parach sekwencji, w których druga będzie zmodyfikowaną o mutacje oraz delecje i insercje wersją pierwszej.

6 Odnośniki

Literatura

[1] http://biopython.org

[2] http://master.bioconductor.org/packages/release/bioc/html/Biostrings.html

[3] http://www.cs.unibo.it/ dilena/LabBII/Papers/AffineGaps.pdf

[4] http://link.springer.com/article/10.1007%2FBF02462326