Badanie podobieństwa trzech sekwencji

Michał Bobowski, Andrzej Dudziec

2013-05-15

1 Opis algorytmu

Algorytm realizuje badanie podobieństwa trzech sekwencji przy stałym zużyciu pamięci. Jest uogólnioneniem algorytmu Needlmana-Wunscha, korzystającym z zasady dziel i zwyciężaj. Podobieństwo jest określane jedynie dla całych sekwencji, a wynikiem działania algorytmu jest ciąg współrzędnych, tworzących ścieżkę w macierzy przejść (od pierwszego do ostatniego punktu).

Przeprowadzenie pełnych obliczeń zgodnie z algorytmem Needlmana-Wunscha wymaga alokacji trójwymiarowej macierzy przejść, której wymiary odpowiadają długościom badanych sekwencji nukleotydów. Uniemożliwia to zbadanie podobieństwa zestawu sekwencji o długości kilkuset nukleotydów.

Niniejszy algorytm redukuje ilość potrzebnej pamięci, zwiększając przy tym złożoność obliczeniową. Macierz przejść jest obliczana „w locie" przy użyciu dwóch aktualnych plastrów. Celem obliczenia macierzy jest znalezienie komórki ze środkowego plastra przez którą przechodzi najlepsze rozwiązanie, a następnie rekurencyjne wywołanie funkcji dla podzbiorów sekwencji wyznaczonych przez tą komórkę. Warunkiem zatrzymania rekurencji jest uzyskanie przynajmniej jednej sekwencji o długości równej 1 - od tego momentu obliczana jest pełna tablica przejść. Tym samym złożoność pamięciowa algorytmu jest ograniczona do poziomu algorytmu obliczającego dwie sekwencje, gdyż w końcowych wywołaniach istnieje potrzeba zbudowania dwuwymiarowej macierzy.

2 Opis aplikacji

Aplikacja została zrealizowana w języku Java w oparciu o wzorzec projektowy MVC. Zawiera następujące klasy:

- *Model* – klasa zawierająca zmienne wpływające na działanie algorytmu, czyli macierz kar i nagród oraz wysokość kar za przerwę i wiszący nukleotyd.
- *View* – klasa odpowiedzialna za widok, interakcję z użytkownikiem i wyświetlanie wyników końcowych.
- *Controller* – klasa implementująca algorytm i przeprowadzająca obliczenia z nim związane.
- *Cell* – klasa reprezentująca pojedynczą komórkę macierzy.
- *Tuple* – krotka opakowujące współrzędne w macierzy.

Interfejs użytkownika składa się z czterech przycisków:

- *Matrix* - pozwala na edycję macierzy kar i nagród.
- *Open* - wczytuje plik z danymi wejściowymi.
- *Save* - zapisuje wynik do pliku tekstowego.
- *Run* - uruchamia algorytm

Dodatkowo wyświetlane są sekwencje wejściowe, dopasowanie końcowe, czas obliczeń oraz maksymalne zużycie pamięci. Formatem wejściowym dla programu są pliki tekstowe, zawierające w trzech pierwszych liniach kolejne sekwencje znaków A, C, G oraz T.

3 Testy wydajnościowe

Testy wydajnościowe zostały przeprowadzone na komputerze wyposażonym w procesor Intel Dual Core z taktowaniem 2,6 GHz. Polegały one na porównaniu trzech identycznych sekwencji o założonej długości (od 100 do 1000 nukleotydów). Dane testowe są zapisane w plikach test100.txt, ..., test1000.txt.

Warto zaznaczyć, że pomiar alokacji pamięci jest obarczony pewnym błędem. Po pierwsze część pamięci jest pochłaniana przez elementy niezwiązane z rzeczywistymi obliczeniami (np. interfejs użytkownika), a po drugie wirtualna maszyna Javy może nie raportować dokładnych wartości.

Wyniki testów wskazują, że dwukrotne zwiększenie długości sekwencji powoduje ok. czterokrotny wzrost czasu obliczeń. Poziom alokacji pamięci rośnie znacznie wolniej i nie powinien być ograniczeniem nawet dla bardzo długich sekwencji.

Tablica 1: Testy wydajnościowe

Długość sekwencji	Alokacja pamięci (MB)	Czas obliczeń (s)
100	86	0.461
200	184	1.262
300	184	4.393
500	208	22.849
1000	493	349.171

4 Pozostałe testy

Pozostałe testy pokazują poprawność działania programu pod kątem merytorycznym. Wykorzystane zostały w nich rzeczywiste sekwencje nukleotydów.

Plik *testA.txt* bazuje na sekwencji 368 nukleotydów występującej w mRNA człowieka. Pierwsza badana sekwencja jest kompletna, natomiast w drugiej i trzeciej usunięto losowo po 10 nukleotydów. Wynik odzwierciedla przerwy zgodnie z rzeczywistością.

Plik *testB.txt* wykorzystuje gen HSGLTH1 o długości 1021 nukleotydów. Pierwsza sekwencja zawiera kompletny gen. W drugiej wprowadzono punktowe mutacje na indeksach 10, 20, 30, 40 i 50 oraz kilka par przerw. W trzeciej wprowadzono punktowe mutacje na indeksach 15, 25, 35, 45 i 55 oraz kilka par przerw. Wynik jest zgodny z oczekiwaniami