Algorytmy Kombinatoryczne w Bioinformatyce – Projekt IV

Michał Konon
Nr Albumu: 152499
Semestr III, Stopień I
Bioinformatyka
Wydział Informatyki i Telekomunikacji

Opis Algorytmu

Program wczytuje dane z pliku i tworzy klasę zawierającą: tablice z wprowadzonymi danymi, tablice z wartościami prawda/fałsz informującą czy wartość we wprowadzonej sekwencji została sprawdzona oraz tablicę z wynikiem. Po odczytaniu danych, zostaje sprawdzona wielkość tablicy według wzoru:

$$|A| = \binom{k+2}{2}$$

Jeżeli k jest ułamkiem, to wprowadzona instancja jest błędna, program się kończy. W przeciwnym rozpoczyna się pomiar czasu i oblicza długość pierwszego fragmentu sekwencji poprzez odjęcie przed ostatniego elementu od ostatniego elementu wprowadzonych danych. Wynik zostaje wprowadzony do rezultatu oraz pierwsze jego wystąpienie zostaje oznaczone jako sprawdzone. Następnie wywołuje się rekurencyjna funkcja do budowania „mapy" rozwiązania. Każde wywołanie tej funkcji rozpoczyna od sprawdzenia, czy wszystkie wartości zostały sprawdzone, jeżeli tak zwraca prawdę. W przeciwnym wypadku zostaje utworzona tablica zawierająca pozycje dodanych elementów. Funkcja wchodzi w pętle o wielkości tablicy z danymi, sprawdza czy dany element został już sprawdzony, jeżeli tak to przechodzi do kolejnego, jeżeli nie dodaje go do wyniku. Zostają utworzone zmienne zawierające wynik sumy (początkowo 0) oraz informację czy wynik jest poprawny (początkowo false). Program rozpoczyna iterację po elementach „mapy" rozpoczynając od końca. Każdy element jest dodawany do sumy. Program sprawdza czy jest nie sprawdzony element we wprowadzonym zbiorze równy sumie, jeżeli tak jego pozycja zostaje zapisana a program wychodzi z pętli. Jeżeli pętla się zakończy, a element nie spełnił tego warunku program przestaje sprawdzać elementy wyniku. Wartość zostaje usunięta z tablicy wyników, a każda wprowadzona wartość której pozycja znajduje się w zbiorze pozycji, oznaczona jako nie sprawdzona. Jeżeli jednak warunek został spełniony dla każdej wartości wyniku, zostaje uruchomiona kolejna iteracja funkcji. Gdy wynik funkcji będzie prawdą zwróci prawdę. Jeżeli jednak funkcja przejdzie przez wszystkie elementy wprowadzonych danych, nie zwracając prawdy, zwróci fałsz.

Przeprowadzone Testy

Testy zostały przeprowadzone na dostarczonych oraz wygenerowanych instancjach.

Dostarczone Instancje

- Ins-PDP-11a-asc.txt
  Czas: 15267 ms = 15.27 s
  Mapa: (4, 6, 5, 8, 3, 9, 5, 2, 4, 7, 8, 6)
- Ins-PDP-12a-asc.txt
  Czas: 163750 ms = 163.75 s
  Mapa: (4, 6, 5, 8, 3, 9, 5, 2, 4, 7, 8, 6, 6)

- Ins-PDP-13a-asc.txt
  Czas: 752720 ms = 752.72 s
  Mapa: (3, 6, 6, 8, 7, 4, 2, 5, 9, 3, 8, 5, 6, 4)
- Ins-PDP-14a-asc.txt
  Czas powyżej 1 godziny.
- Ins-PDP-11b-asc.txt
  Czas: 15 ms
  Mapa: (38, 74, 27, 66, 42, 15, 89, 47, 35, 13, 12, 54)
- Ins-PDP-11b-desc.txt
  Czas: 15 ms
  Mapa: (38, 74, 27, 66, 42, 15, 89, 47, 35, 13, 12, 54)
- Ins-PDP-14b-asc.txt
  Czas: 0 ms -> poniżej 1 ms
  Mapa: (54, 12, 13, 35, 47, 89, 15, 42, 66, 27, 74, 38, 25, 57, 79)
- Ins-PDP-14b-desc.txt
  Czas: 0 ms -> poniżej 1 ms
  Mapa: (54, 12, 13, 35, 47, 89, 15, 42, 66, 27, 74, 38, 25, 57, 79)

Wygenerowane Instancje

- instance_5_asc
  Czas: 0 ms -> poniżej 1 ms
  Mapa: (1, 3, 6, 3, 4)
- instance_5_desc
  Czas: 0 ms -> poniżej 1 ms
  Mapa: (1, 3, 6, 3, 4)
- instance_8_asc
  Czas: 16 ms
  Mapa: (3, 6, 6, 1, 6, 3, 4, 7)
- instance_8_desc
  Czas: 17 ms
  Mapa: (3, 6, 6, 1, 6, 3, 4, 7)
- instance_11_asc
  Czas: 8827 ms
  Mapa: (4, 6, 5, 8, 5, 1, 4, 14, 3, 4, 11)
- instance_11_desc
  Czas: 9135 ms
  Mapa: (4, 6, 5, 8, 5, 1, 4, 14, 3, 4, 11)
- instance_14_asc
  Czas: 26351 ms = 26.35 s
  Mapa: (13, 1, 2, 11, 5, 14, 7, 1, 8, 10, 1, 2, 9, 13)
- instance_14_desc
  Czas: 26773 ms = 26.77 s
  Mapa: (13, 1, 2, 11, 5, 14, 7, 1, 8, 10, 1, 2, 9, 13)

Wnioski

Przeprowadzone testy wykazały, że nawet nieznaczne zwiększenie liczebności zbioru skutkuje gwałtownym wzrostem czasu działania algorytmu. Zła ilość cieć zostanie wskazana wykryta, gdy k nie będzie całkowite. Gdy jednak k okażę się być całkowite lub numeracja będzie błędna, program może nigdy nie zakończyć swojego działania.