Projektowanie Efektywnych Algorytmów

Projekt

21/10/2020

241141 Wojciech Ziębicki

(1) Brute Force

Spis treści

1. Sformułowanie zadania

Zadane polega na opracowaniu, implementacji i zbadaniu efektywności algorytmu przeglądu zupełnego rozwiązującego problem komiwojażera w wersji optymalizacyjnej. Dane potrzebne do poprawnego uruchomienia programu zostały umieszczone w pliku inicjalizującym, natomiast wyniki pomiarów można odczytać bezpośrednio z wyświetlanych napisów na konsoli, lub w formie raportu w pliku o podanej nazwie w pliku inicjalizującym.

Założenia pracy programu:

- Celem metody jest zwrócenie najkrótszej znalezionej ścieżki pomiędzy wszystkimi punktami kończąc w punkcie początkowym, przy założeniu, że każde z miast odwiedziło się tylko raz
- Program powinien być uruchamiany z wykorzystaniem pliku inicjującego zawierającego nazwę pliku do zapisu obliczonych danych oraz wszystkie instancje problemu o schemacie: nazwa pliku z macierzą odległości miast, optymalna wartość ścieżki, ilość powtórzeń algorytmu
- Po wykonaniu obliczeń wyniki wygenerowane będą w postaci raportu w formacie csv

2.Metoda

Metoda przeglądu zupełnego, tzw. przeszukiwanie wyczerpujące (eng. exhaustive search) bądź metoda siłowa (eng. brute force), polega na znalezieniu i sprawdzeniu wszystkich rozwiązań dopuszczalnych problemu, wyliczeniu dla nich wartości funkcji celu i wyborze rozwiązania o ekstremalnej wartości funkcji celu – najniższej (problem minimalizacyjny) bądź najwyższej (problem maksymalizacyjny).Wadą takiej metody jest duża złożoność czasowa, która wynosi O (n!), gdzie n jest ilością miast. Z tego powodu dla większych problemów sposób ten może być nieefektywny. Mocną stroną tego metody jest natomiast prosta implementacja oraz pewność znalezienia poprawnego wyniku (w przypadku poprawnej implementacji).

Zrealizowana metoda wykorzystywała tablicę do zapisu danych i została napisany w oparciu o poniższy pseudokod:

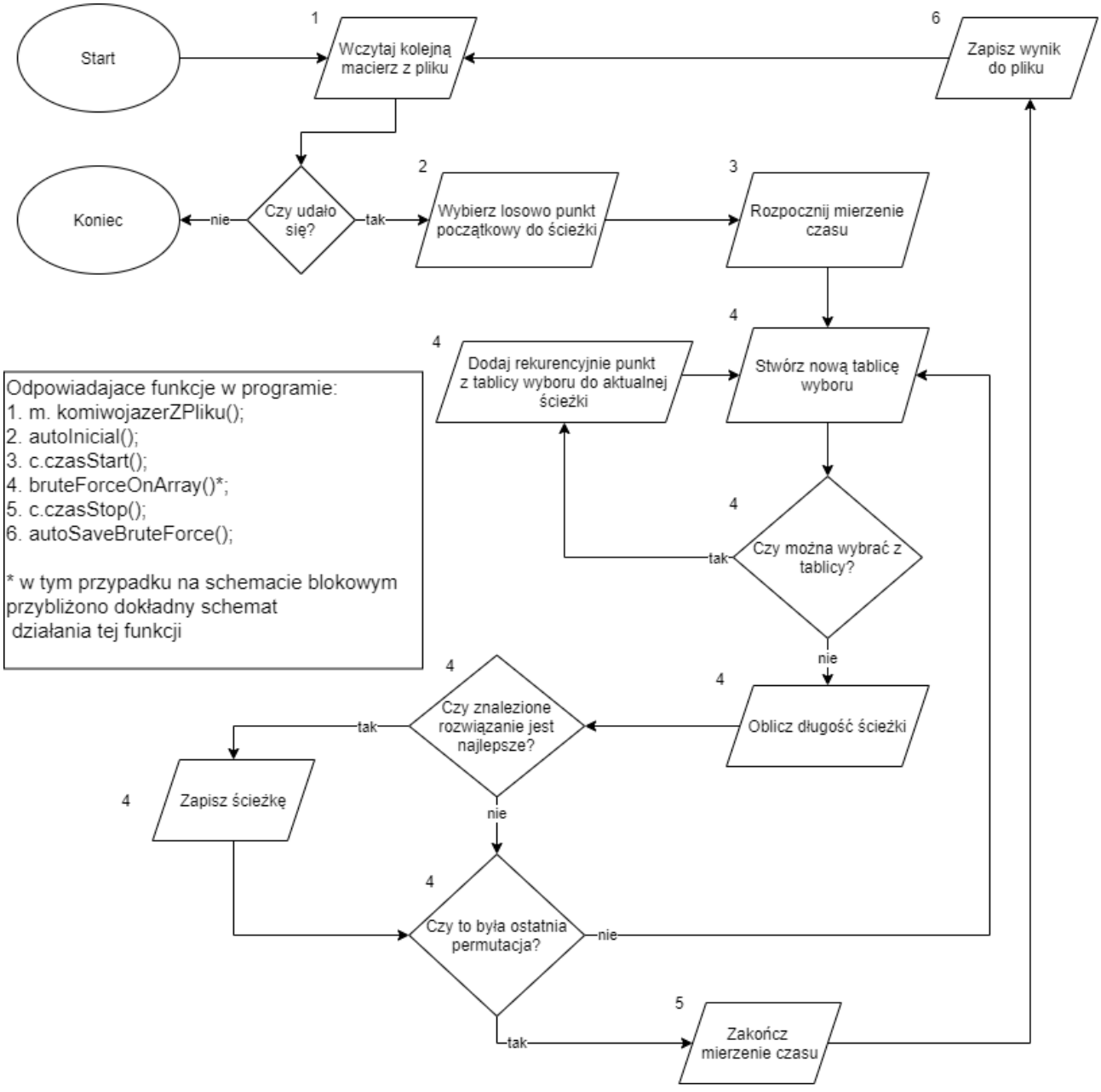
```
FUNCKJA BRUTEFORCE(rozmiar,permutacja){
//Petla wykonuje sie zawsze tyle razy ile permutacji poszukuje na danym poziomie drzewa

  for (int i = 0; i < rozmiar; i++) {
    /*1.*/ Stwórz tablicę wyboru, czyli zbiór wszystkich możliwych do odwiedzenia
           wierzchołków danym poziomie drzewa.
    /*2.*/ Przypisz odpowiedni węzeł z tablicy wyboru do tablicy odwiedzonych:
           tablicaOdwiedzonych[permutacja] = tablicaWyboru[i]
    /*3.*/ jeśli rozmiar == 1 -> policz długość ciągu i sprawdź czy jest najmniejszy
    /*4.*/ jeśli długość jest mniejsza niż aktualnie najmniejsza przypisz nową długość wraz
           z najmniejszą ścieżką
    /*5.*/ wywołuj rekurencyjnie dopóki nie dojdziesz do ostatniego:

    if(rozmiar > 1) FUNKCJA BRUTEFORCE(rozmiar-1, permutacja + 1)
    }
}
```

W swoim programie wykorzystałem implementację na tablicy. Tworzenie permutacji wykonywane jest rekurencyjnie.

3.Algorytm



Odpowiadajace funkcje w programie:
1. m. komiwojazerZPliku();
2. autoInicial();
3. c.czasStart();
4. bruteForceOnArray()*;
5. c.czasStop();
6. autoSaveBruteForce();

* w tym przypadku na schemacie blokowym przybliżono dokładny schemat działania tej funkcji

4. Dane testowe

Do sprawdzenia poprawności działania wybrano następujący zestaw instancji:

1. m8.atsp http://jaroslaw.rudy.staff.iiar.pwr.wroc.pl/pea.php#instances
2. m9.atsp http://jaroslaw.rudy.staff.iiar.pwr.wroc.pl/pea.php#instances
3. tsp_10.txt; http://jaroslaw.mierzwa.staff.iiar.pwr.wroc.pl/pea-stud/tsp/
4. m11_atsp; http://jaroslaw.rudy.staff.iiar.pwr.wroc.pl/pea.php#instances
5. tsp_12.txt; http://jaroslaw.mierzwa.staff.iiar.pwr.wroc.pl/pea-stud/tsp/
6. tsp_13.txt; http://jaroslaw.mierzwa.staff.iiar.pwr.wroc.pl/pea-stud/tsp/

5.Procedura badawcza

Należało zbadać zależność czasu rozwiązania problemu od wielkości instancji. W przypadku algorytmu realizującego przegląd zupełny w przestrzeni rozwiązań dopuszczalnych nie występowały parametry programu, które mogły mieć wpływ na czas i jakość uzyskanego wyniku. W związku z tym procedura badawcza polegała na uruchomieniu programu sterowanego plikiem inicjującym: ***test.ini.***

(format pliku: nazwa_instancji liczba_wykonań rozwiązanie_optymalne [ścieżka optymalna]; nazwa_pliku_wyjściowego). Zapis ścieżki optymalnej nie ma znaczenia. Program konwertuje ją do jednego systemu.

m8.atsp 20 202 0-3-2-4-1-6-7-5-0
m9.atsp 20 215 0-3-4-2-8-7-6-1-5-0
tsp_10.txt 20 212 0 - 3 - 4 - 2 - 8 - 7 - 6 - 9 - 1 - 5 - 0
m11.atsp 20 251 [1-7-5-3-10-0-1-8-4-6-2-9-7]
tsp_12.txt 5 264 0 - 1 - 8 - 4 - 6 - 2 - 11 - 9 - 7 - 5 - 3 - 10 - 0
tsp_13.txt 1 269 0 - 10 - 3 - 5 - 7 - 9 - 11 - 2 - 6 - 4 - 8 - 1 - 12 - 0
bruteForce.csv

Ilość powtórzeń danej instancji zależał indywidualnie od wielkości jej macierzy. Do pliku wyjściowego bruteForce.csv zapisywana była nazwa pliku, czas obliczeń, najlepsza znaleziona ścieżka, wartość tej ścieżki, wartość optimum, ścieżka optimum oraz obliczony procent błędu znalezionego rozwiązania względem optymalnego. Plik wyjściowy zapisywany był w formacie csv. Poniżej przedstawiono fragment zawartości pliku wyjściowego.

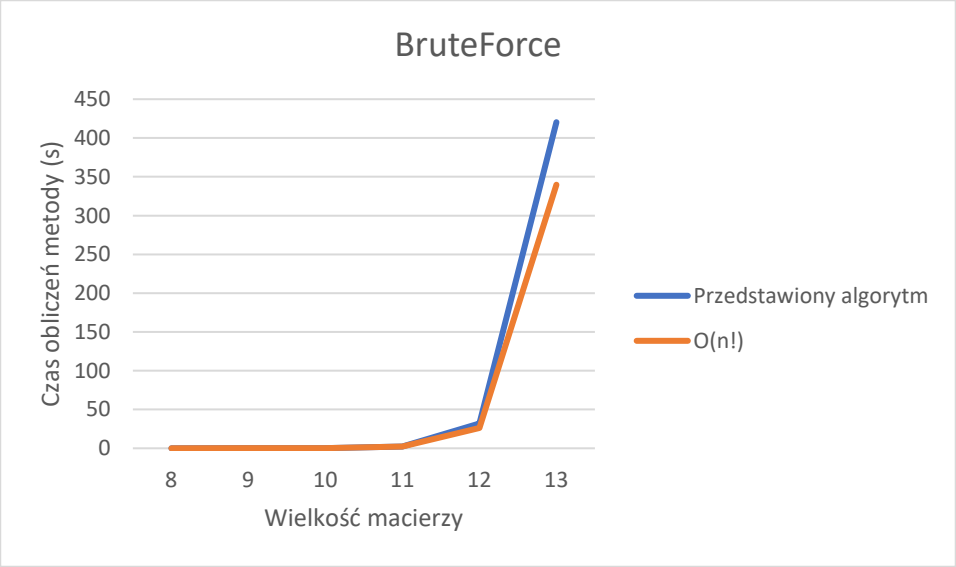
File name, Time(s), Best path, Path lenght, Optimum, % of mistake
m8.atsp, 0.015000, 0-5-7-6-1-4-2-3-0, 202, 202, 0-3-2-4-1-6-7-5-0, 0
m8.atsp, 0.009000, 0-5-7-6-1-4-2-3-0, 202, 202, 0-3-2-4-1-6-7-5-0, 0
...
File name, Time(s), Best path, Path lenght, Optimum, % of mistake
m9.atsp, 0.002000, 0-3-4-2-8-7-6-1-5-0, 215, 215, 0-3-4-2-8-7-6-1-5-0, 0
m9.atsp, 0.004000, 0-3-4-2-8-7-6-1-5-0, 215, 215, 0-3-4-2-8-7-6-1-5-0, 0
...
Wyniki opracowane zostały w programie MS Excel.

6. Wyniki

Pomiary efektywności algorytmów zostały przeprowadzone indywidualnie dla każdej instancji. Ilość powtórzeń badania zależało od wielkości instancji. Wynik każdej z nich został uśredniony. Jako macierz odległości podawane były pliki udostępnione na stronie projektu dr. Mierzwy oraz dr. Rudego. Pomiary były przeprowadzone w możliwie zbliżonym do siebie obciążeniu procesora.

Wyniki pomiarów dla wybranych plików:

Nazwa pliku	Ilość pomiarów	Średni czas (s)
m8.atsp	20	0,0022
m9.atsp	20	0,0209
tsp_10.txt	20	0,2119
m11.atsp	10	2,3463
tsp_12.txt	5	32,0530
tsp_13.txt	1	420,3610



Rys. 1 Wykres zależności wielkości macierzy od czasu obliczeń

Wynik uzyskany w badaniu w porównaniu do teoretycznego wyniku który powinienem uzyskać jest nieznacznie dłuższy. Na wynik mógł wpłynąć niedokładny pomiar czasu dla najmniejszej macierzy, który wynosił średnio tylko 0,0022 (s). Nie mniej badanie pokazało, że zależność czasu od kolejnych macierzy jest wykładnicza.

Wyniki zostały przedstawione w pliku wynikiBruteForce.csv, natomiast wszystkie ww. pliki zostały dołączone do raportu i znajdują się na dysku Google pod aresem: https://drive.google.com/drive/folders/1B_GRZD-fZ8nLayQy3TZB1A9hTUo4G9PB?usp=sharing

7. Analiza wyników i wnioski

Przegląd zupełny ze względu na swoją złożoność obliczeniową staje się przydatny tylko dla bardzo małych instancji. Największymi zaletami tej metody są: otrzymanie wyniku dokładnego oraz prosta implementacja. Niestety, krzywa wzrostu czasu względem wielkości instancji ma charakter wykładniczy [O(n!)]. Z tego powodu dla większych instancji tą metodą nie da się uzyskać wyniku w rozsądnym czasie.