Projektowanie Efektywnych Algorytmow

Projekt

15/11/2022

259091 Jakub Mordalski

(3) Branch & Bound

1. Sformułowanie zadania

Celem zadania projektowego jest implementacja algorytmu Branch & Bound rozwiązującego problem komiwojażera w wersji optymalizacyjnej, a następnie zbadaniu jego efektywności poprzez pomiar czasu wykonania algorytmu. Algorytm Branch & Bound powinien być efektywniejszy od wcześniej implementowanego Brute Force, Wynika to z ograniczenia ilości przeszukiwanych ścieżek.
Złożoność obliczeniowa algorytmu BnB wynosi O(n^2*2^n).
Problem komiwojażera (TSP - Traveling Salesman Problem) to zagadnienie polegające na odnalezieniu minimalnego cyklu Hamiltona w grafie pełnym ważonym.
Graf pełny ważony – między każdą parą wierzchołków istnieje połączenie, a każda z krawędzi ma swoją wagę
Cykl Hamiltona – Cykl przechodzący prze każdy wierzchołek dokładnie raz, z wyłączeniem wierzchołka początkowego, do niego wracamy po przejściu wszystkich wierzchołków.
Problem Komiwojażera polega zatem na odnalezieniu ścieżki zamkniętej, która przechodzi przez wszystkie wierzchołki, kończąc w miejscu początkowym. Koszt ścieżki powinien być jak najmniejszy.
TSP występuje w dwóch wersjach:
Symetryczny - opiera się na grafie nieskierowanym.
Asymetryczny – opiera się na grafie skierowanym.

2. Metoda

Zastosowana metoda to Branch & Bound zwana też metodą Podziału i ograniczeń.

Metoda polega na wyszukiwaniu coraz to lepszych ścieżek na podstawie przeglądania kolejnych gałęzi drzewa utworzonego z permutacji elementów instancji.

Pierwszym krokiem jest wybranie punktu odniesienia, czyli ścieżki która będzie nasza początkową, domniemaną najkrótszą ścieżką. Mamy na to wiele sposobów:

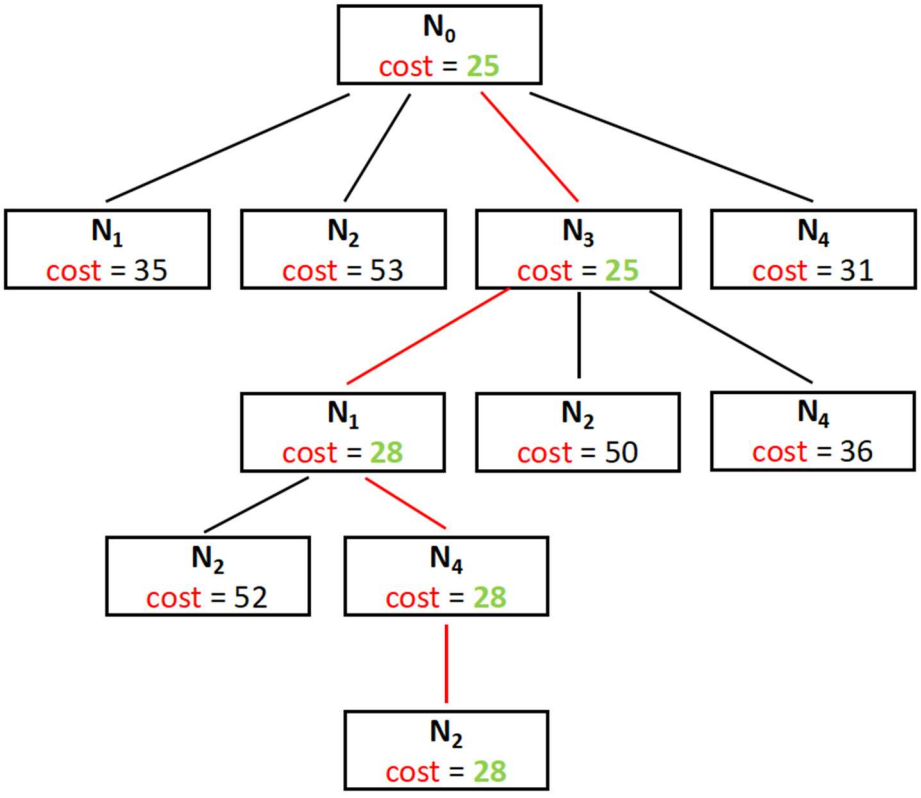
- Wybranie ścieżki zgodnie z kolejnością (1 -> 2 -> 3 …)
- Wybranie losowej ścieżki
- Wybranie ścieżki za pomocą algorytmu

W moim przypadku zastosowany został wybór zgodnie z kolejnością.

Następnie przy każdym przejściu w głąb drzewa, porównujemy aktualną domniemaną najkrótszą ścieżkę z aktualną ścieżką.

Gdy trafiamy na ścieżkę która jest dłuższa od aktualnie najkrótszej, przerywamy przeszukiwanie danej gałęzi. Ponieważ już na tym etapie można stwierdzić że optymalnego rozwiązania w tej gałęzi nie znajdziemy.

Gdy wyczerpiemy wszystkie możliwości, zwracamy domniemaną najkrótszą ścieżkę jako optymalne rozwiązanie problemu.



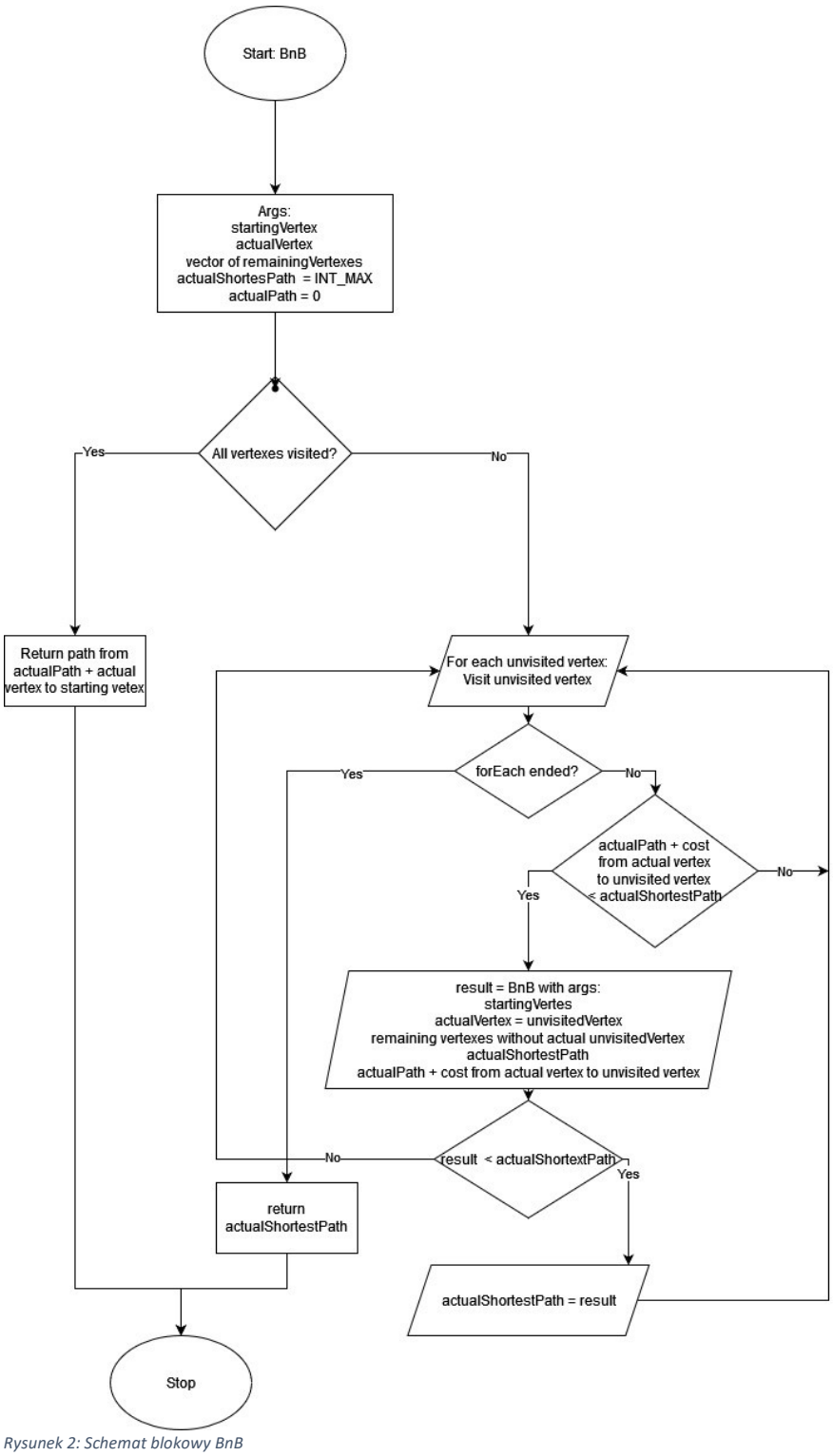
Rysunek 1: Wizualizacja BnB

3. Porównanie algorytmów

Przewagą tego algorytmu nad przeglądem zupełnym jest znacznie mniejsza ilość ścieżek do sprawdzenia, niestety zależne jest to od zróżnicowania instancji.
Różnica wynika z odrzucania ścieżek których dalsze sprawdzanie nie ma sensu, dzięki czemu w optymistycznym przypadku, czyli trafieniu w optymalne rozwiązane w momencie wyboru pierwszej ścieżki, oraz przy znacznie dłuższych pozostałych ścieżkach. Algorytm wykona się po przejściu niewielkiej części wszystkich możliwych permutacji.
Jednak w przypadku pesymistycznym, gdy będziemy trafiać na ścieżki w kolejności malejącej, algorytm wykona się w czasie podobnym do przeglądu zupełnego.

4. Algorytm

Zastosowany algorytm polega na rekurencyjnym wywoływaniu funkcji BnB która ma za zadanie wyszukiwanie coraz to krótszych ścieżek.



Rysunek 2: Schemat blokowy BnB

5. Dane testowe

Do sprawdzenia poprawności działania algorytmu wybrano następujący zestaw instancji:

- tsp_6_1.txt
- tsp_6_2.txt
- tsp_10.txt
- tsp_12.txt
- tsp_13.txt
- tsp_14.txt
- tsp_15.txt
- tsp_17.txt

Pochodzące ze strony : http://jaroslaw.mierzwa.staff.iiar.pwr.wroc.pl/pea-stud/tsp/

6. Procedura badawcza

Procedura badawcza polegała na uruchomieniu algorytmu dla instancji o różnych wielkościach, oraz zmierzeniu czasu działania algorytmu dla tych instancji. Czas mierzony był z wykorzystaniem biblioteki std::chrono poprzez odczyt różnicy czasu systemowego przed oraz po każdym uruchomieniu algorytmu.
Dla algorytmu realizującego metodę Branch and Bound nie występowały parametry mające wpływ na wynik działania algorytmu.
Przy uruchomieniu używany jest plik sterujący conf.ini, zawierający następujące dane
<nazwa pliku> <ilość testów> <wielkość instancji> <optymalna długość ścieżki> <optymalna ścieżka>.
Zawartość pliku użytego podczas testów :

tsp_6_1.txt 100 6 132 0 1 2 3 4 5

tsp_6_2.txt 100 6 80 0 5 1 2 3 4

tsp_10.txt 100 10 212 0 3 4 2 8 7 6 9 1 5

tsp_12.txt 50 12 264 0 1 8 4 6 2 11 9 7 5 3 10

tsp_13.txt 10 13 269 0 10 3 5 7 9 11 2 6 4 8 1 12

tsp_14.txt 10 14 282 0 10 3 5 7 9 13 11 2 6 4 8 1 12

tsp_15.txt 5 15 291 0 10 3 5 7 9 13 11 2 6 4 8 14 1 12

tsp_17.txt 1 17 39 0 11 13 2 9 10 1 12 15 14 5 6 3 4 7 8 16
Dla większych instancji czas wykonania algorytmu wynosił powyżej 30 min dlatego nie zostały one zbadane.

Plik wyjściowy

Wyniki Powyższego testu zostały zapisane w pliku result.csv, który zawierał następujące dane :
<ścieżka><długość ścieżki> <czas wykonania>,

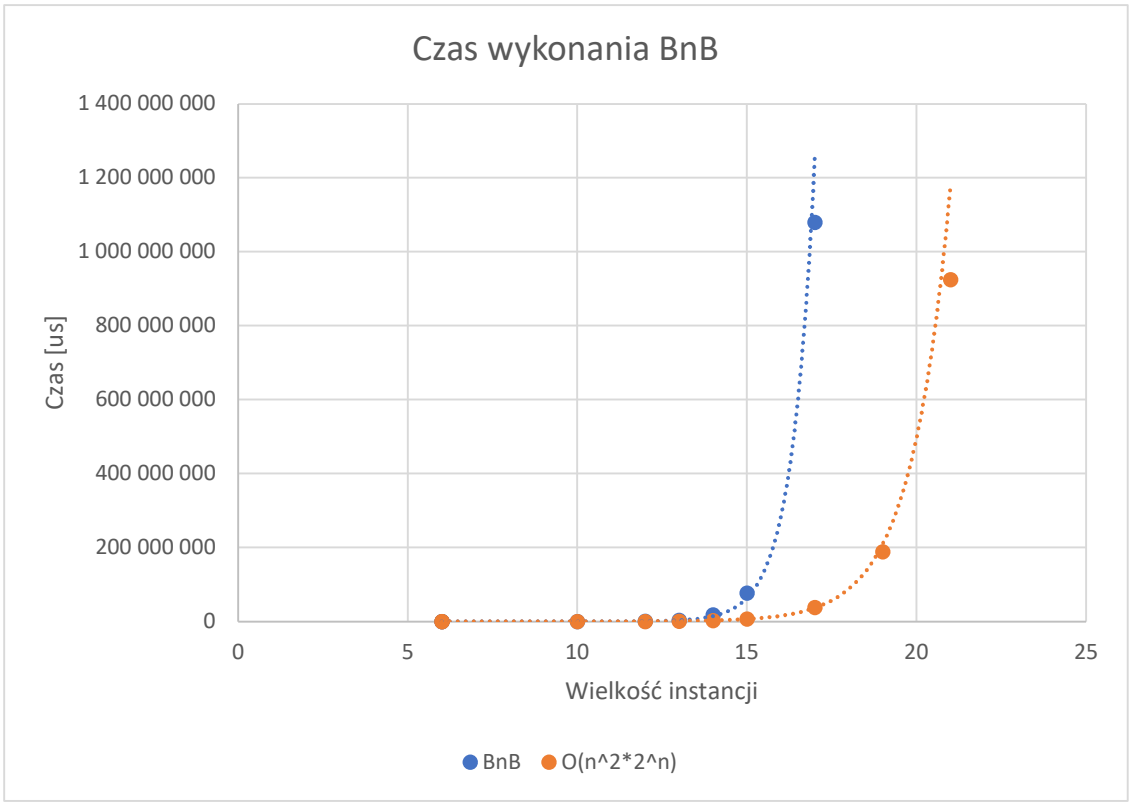
Testy dla poszczególnych plików zostały oddzielone linią zawierającą:
<nazwa pliku> <ilość testów> <wielkość instancji> <optymalna długość ścieżki> <optymalna ścieżka>

Test był przeprowadzany na platformie

- AMD Ryzen 5 3500X 4.1 GHz
- 16GB RAM DDR4 3400MHz CL16
- Samsung 500GB M.2 PCIe NVMe 970 EVO Plus

7. Wyniki

Wyniki działania programu zostały zapisane w pliku result.csv oraz zamieszczone na e portalu wraz z kodem źródłowym:



Rysunek 3: Wykres czasu działania algorytmu BnB względem ilości instancji

8. Analiza wyników i wnioski

Wykres czasu działania algorytmu Branch and Bound w zależności od instancji jest wykładniczy. Czas działania algorytmu jest mocno związany z rozkładem instancji oraz z doborem pierwszej najkrótszej ścieżki. Algorytm będzie bardziej efektywny dla instancji o mocno zróżnicowanych wartościach ścieżek, natomiast trafne dobranie pierwszej najkrótszej ścieżki może znacznie zmniejszyć ilość operacji potrzebnych do odnalezienia optymalnego rozwiązania.
Na wykresie widać zbliżenie czasu działania BnB do złożoności O(n^2*2^n).

9. Źródła

https://pl.wikipedia.org/wiki/Problem_komiwoja%C5%BCera
https://pl.wikipedia.org/wiki/Cykl_Hamiltona
http://jaroslaw.mierzwa.staff.iiar.pwr.wroc.pl/pea-stud/tsp/
https://www.youtube.com/watch?v=XaXsJJh-Q5Y&t=402s