Implementacja i analiza efektywności algorytmu Brute Force oraz Branch and Bound dla asymetrycznego problemu komiwojażera.

Jakub Grzegocki (264009)

listopad, 2023

Sprawozdanie z pierwszego zadania projektowego kursu "Projektowanie efektywnych algorytmów"

Spis treści

1 Wstęp

Zadaniem projektowym było zaimplementowanie oraz analiza efektywności dwóch algorytmów rozwiązujących asymetryczny problem komiwojażera (ATSP): algorytmu opartego na przeglądzie zupełnym (Brute Force) oraz algorytmu opartego na metodzie podziału i ograniczeń (Branch and Bound).

Asymetryczny problem komiwojażera polega na znalezieniu minimalnego cyklu Hamiltona w pełnym grafie skierowanym o ważonych krawędziach. Minimalny cykl Hamiltona to taki cykl, który przechodzi przez każdy wierzchołek dokładnie raz, a suma wag jego krawędzi jest minimalna. Ten problem jest szczególnie trudny, ponieważ liczba możliwych kombinacji krawędzi w grafie o n wierzchołkach rośnie bardzo szybko i dla naiwnych algorytmów opartych na przeglądzie zupełnym, złożoność obliczeniowa wynosi O(n!).

Głównym celem było przetestowanie algorytmów i porównanie ich wydajności. Algorytm oparty na przeglądzie zupełnym generuje wszystkie możliwe cykle i wybiera ten o najmniejszej wadze, co jest nieefektywne dla dużych instancji problemu. Algorytm podziału i ograniczeń (Branch and Bound) stosuje strategię eliminacji pewnych gałęzi drzewa przeszukiwań, co pozwala na ograniczenie liczby przeszukiwanych przypadków.

Niniejsza analiza algorytmów była konieczna ze względu na wydajność obliczeniową oraz skalowalność, zwłaszcza w przypadku dużych instancji problemu komiwojażera. Wnioski z takiej analizy są kluczowe dla wyboru odpowiedniego algorytmu w praktycznych zastosowaniach, gdzie efektywność obliczeniowa ma duże znaczenie.

2 Metoda podziału i ograniczeń - Branch and Bound

Metoda podziału i ograniczeń, analogicznie do metody przeglądu zupełnego, wykorzystuje przeszukiwanie drzewa reprezentującego przestrzeń rozwiązań problemu. Kluczową cechą tej metody, odróżniającą ją od podejścia naiwnego, jest zdolność do ograniczenia liczby analizowanych rozwiązań, co potencjalnie prowadzi do skrócenia czasu potrzebnego do znalezienia optymalnego rozwiązania globalnego. Zastosowanie tej metody opiera się na wprowadzeniu pewnych procedur w trakcie przeszukiwania drzewa rozwiązań. Podział (rozgałęzianie), polega na podziale zbioru rozwiązań reprezentowanego przez węzeł drzewa na rozłączne podzbiory, które są reprezentowane przez następników tego węzła. Ograniczanie (bounding) - obejmuje pomijanie w przeszukiwaniu tych gałęzi drzewa, w których wiadomo, że nie zawierają optymalnego rozwiązania w swoich liściach.

W ramach danego poddrzewa rozwiązań poszukiwane są rozwiązania znajdujące się pomiędzy dolnym a górnym ograniczeniem. Górne ograniczenie to wartość najlepszego dotychczas znalezionego rozwiązania, natomiast dolne ograniczenie to heurystyczna ocena najlepszego rozwiązania, które może zostać znalezione w podzbiorze reprezentowanym przez następnika bieżącego węzła. Dolne ograniczenie musi zostać obliczone dla każdego z następników.

Podczas przeszukiwania drzewa odrzucane są węzły, dla których dolne ograniczenie przekracza wartość górnego ograniczenia. Wartość dolnego ograniczenia musi być obliczona tak, aby żadne

możliwe do uzyskania rozwiązanie w danym poddrzewie nie przewyższało wartości tego ograniczenia. Mimo że może to prowadzić do pewnego oddalenia się od rzeczywistej wartości optymalnej, powinna być jednocześnie możliwie jak najbliższa rzeczywistej wartości, co umożliwia efektywne pomijanie niepotrzebnych analiz.

3 Opis implementacji algorytmu Branch and Bound

Ten algorytm stosuje przegląd drzewa rozwiązań w głąb, a dla następników każdego węzła oblicza dolne ograniczenia. Proces przeglądu skupia się na wyborze węzła, dla którego dolne ograniczenie jest najniższe spośród dostępnych i jednocześnie niższe od ograniczenia górnego. Wybrany węzeł jest omijany podczas dalszego przeglądu. Eksploracja kolejnych węzłów odbywa się rekurencyjnie, przy jednoczesnej modyfikacji aktualnej ścieżki oraz ograniczenia górnego.

Do implementacji opisanego algorytmu posłużyły nam różne struktury danych, takie jak dwuwymiarowa tablica, przechowująca reprezentację grafu, oraz dwie jednowymiarowe tablice, z których jedna informuje o odwiedzonych wierzchołkach, a druga służy do obliczania dolnych ograniczeń. Dodatkowo wykorzystano stos, który przechowuje najlepsze znalezione rozwiązanie, oraz stos tymczasowy dla danego etapu działania algorytmu. Kolejka priorytetowa zawiera węzły, których priorytet determinuje obliczone dla każdego z nich dolne ograniczenie. Wierzchołek o najniższym dolnym ograniczeniu ma najwyższy priorytet.

Obliczanie dolnego ograniczenia polega na wyborze z wierszy macierzy krawędzi krawędzi o możliwie najniższej wadze. Te wybrane krawędzie są zapamiętywane w tablicy, gdzie indeks odpowiada wierzchołkowi, z którego wychodzi dana krawędź. Suma wag tych krawędzi stanowi dolne ograniczenie dla wierzchołka początkowego. Ograniczenie to jest następnie aktualizowane i przypisywane do każdego z następników bieżącego węzła. Ten proces jest realizowany zgodnie z określoną regułą, co ilustruje poniższy przykład.

4 Przykład praktyczny z użyciem Branch and Bound

W programie został wygenerowany losowy graf o 4 wierzchołkach. Poniżej znajduje się macierz sąsiedztwa wraz z kolumną zwierającą minimalne wagi przejść wychodzących z wierzchołków każdego wiersza. Podczas rozpoczęcia algorytmu suma tych elementów zarazem jest dolnym ograniczeniem wybranego początkowego wierzchołka.

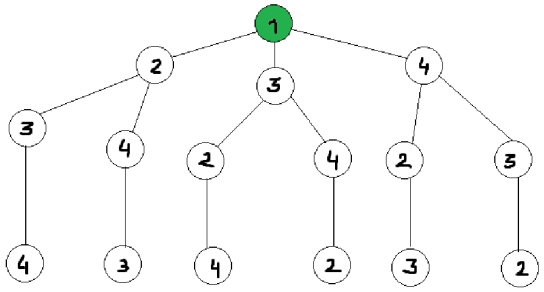
Tabela 1:

	1	2	3	4	redukcja
1	INF	27	37	41	27
2	55	INF	37	25	25
3	13	20	INF	48	13
4	24	29	36	INF	24

Dla uproszczenia przeprowadzania metody krok po kroku, kolorowane będą wierzchołki, dla których wyznaczamy koszt przejścia do nich. Górna granica jest wartością nieskończoną do momentu wyznaczenia pierwszego cyklu hamiltona.

Krok 1.
Zgodnie z powyższą informacją wyznaczamy dolną granicę.

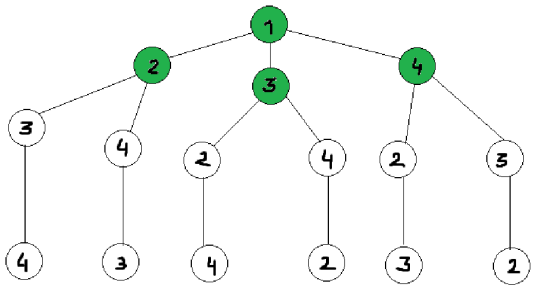
$27 + 25 + 13 + 24 = 89$
Tak więc $C(1) = 89$



Rysunek 1:

Krok 2.
Wyznaczamy dolne granice dla wierzchołków w kolejnym poziomie drzewa.

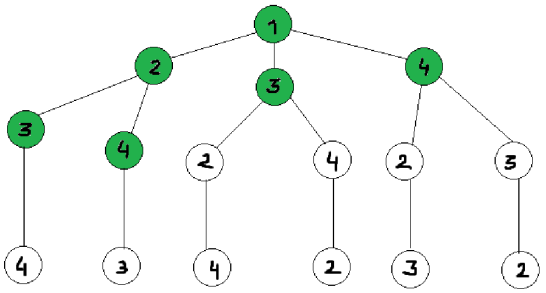
$LB(1,2) = 89 + 27 - 27 = 89$
$LB(1,3) = 89 + 37 - 27 = 99$
$LB(1,4) = 89 + 41 - 27 = 101$



Rysunek 2:

Krok 3.
Najmniejszą wartość ograniczenia posiada w tym momencie węzeł drugi tak więc przeszukiwanie będziemy kontynuować dla niego.

$LB(1,2,3) = 89 + 37 - 25 = 101$
$LB(1,2,4) = 89 + 25 - 25 = 89$

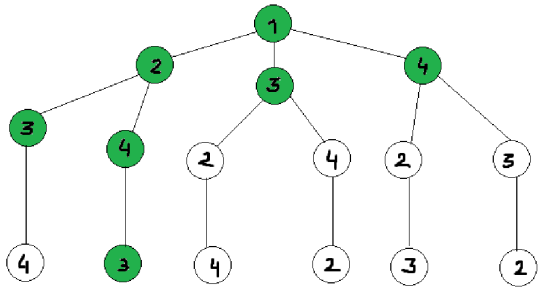


Rysunek 3:

Krok 4.
Najniższą wartość ograniczenia mamy w wierzchołku 4. Wybieramy ten wierzchołek i obliczmy kolejną dolną granicę.

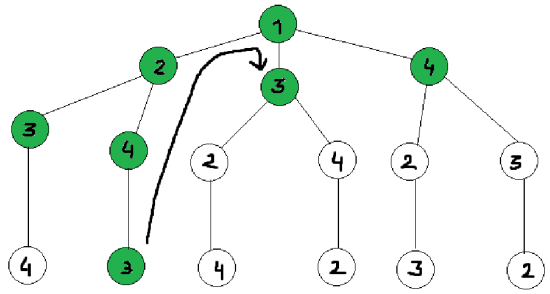
$$LB(1,2,4,3) = 89 + 36 - 24 = 101$$

Doszliśmy do liścia drzewa tak więć dolne ograniczenie dla tego wierzchołka ustawiamy jako górną granicę.



Rysunek 4:

Krok 5.
Wracamy się w wierzchołkach do poziomu, na którym znajdziemy niższą granice dolną niż górna. Taka istnieje jeszcze dla wierzchołka 3 na 2 poziomie drzewa licząc od korzenia.

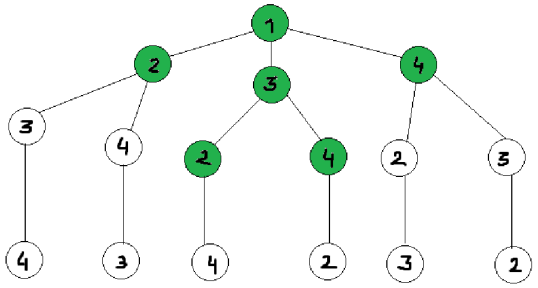


Rysunek 5:

Krok 6.
LB(1,3) wynosi 99 co jest niższe niż 101, więc dla tej ścieżki sprawdzamy dolne granice następników.

$LB(1,3,2) = 99 + 20 - 13 = 106$
$LB(1,3,4) = 99 + 48 - 13 = 134$

Obie nowo wyliczone granice dolne są wyższe niż ograniczenie górne co kończy działanie algorytmu z zapamiętanym przejściem (1,2,4,3).



Rysunek 6:

Potwierdzenie poprawności wyznaczonej ścieżki widzimy poniżej na zrzucie ekranu z programu:

```
Twoj wybor: 5
Podaj ilosc wierzcholkow: 4
999  27   37   41
55   999  37   25
13   20   999  48
24   29   36   999
=========================
1) Wczytaj graf
2) Brute Force
3) Branch and Bound
4) Testuj wybrany graf
5) Generuj graf losowy
6) Testowanie czasow
7) Wyswietl graf
8) Zakoncz program
=========================
Twoj wybor: 3
Branch and Bound ...
Droga: 0 1 3 2 0
Koszt: 101 Czas wykonywania: 2.06e-05 sekund
```

Rysunek 7:

5 Plan eksperymentu

Eksperymenty zostały przeprowadzone dla obu algorytmów, osobno generując różne losowe instancje grafów. Dla branch and bound zostały wygenerowane również grafy o większej ilości wierzchołków ze względu na jego specyfikę i większą szybkość działania w porównaniu do Brute Force. Dla każdej wartości n wierzchołków algorytm został przetestowany 100 razy a jego czasy uśrednione. Zakres liczby wierzchołków dla Brute Force to 1 do 13. Dla Branch and Bound zakres jest od 6 do 22 wierzchołków.

Wykres przedstawiający oba algorytmy został przygotowany generując instancje problemu i wykonując na tej samej instancji kolejno oba algorytmy. Dla każdej n ilości wierzchołków eksperyment został powtórzony 100 razy, jego czasy uśrednione. W tym przypadku n należy do przedziału od 6 do 13 włącznie.

6 Wyniki eksperymentów

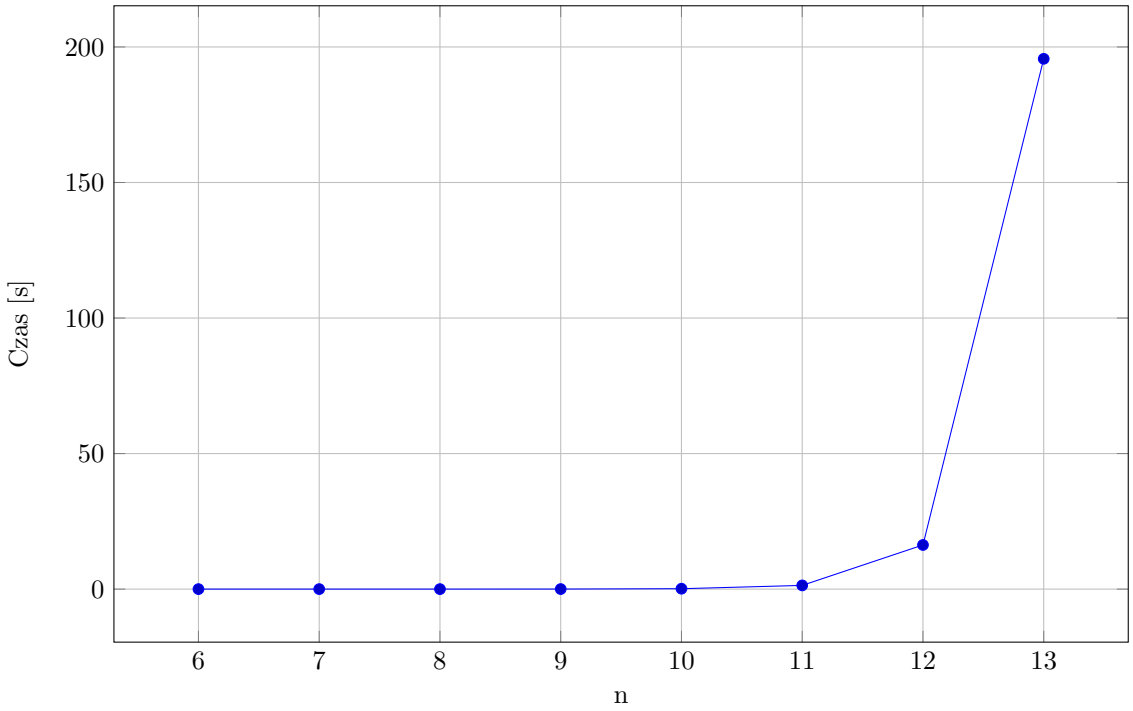
Tabela 2: czasy dla n wierzchołków Brute Force

n	**czas**[**s**]
6	0.00004462
7	0.000251
8	0.00175193
9	0.0148816
10	0.14836
11	1.36486
12	16.2835
13	195.628

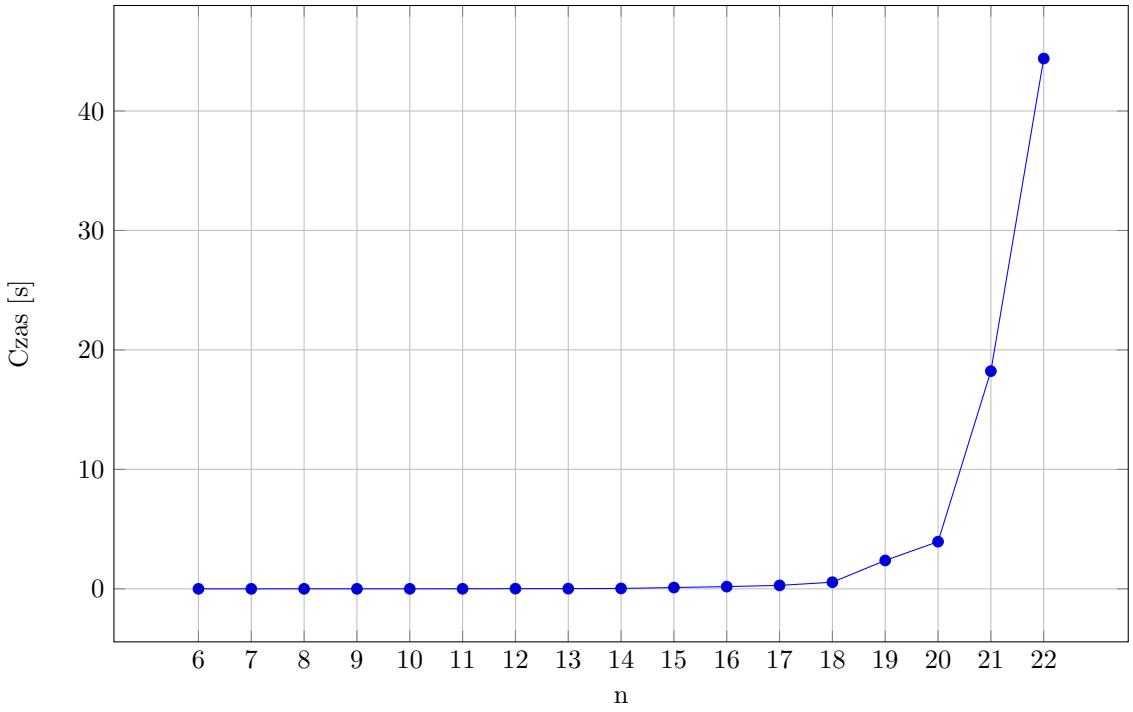
Tabela 3: czasy dla n wierzchołków Branch and Bound

n	**czas**[s]
6	0.0000837
7	0.0001823
8	0.00023442
9	0.00024734
10	0.00027831
11	0.00384747
12	0.0145302
13	0.0174674
14	0.0319109
15	0.108399
16	0.1854227
17	0.2872632
18	0.553328
19	2.375678
20	3.94642
21	18.223
22	44.3926

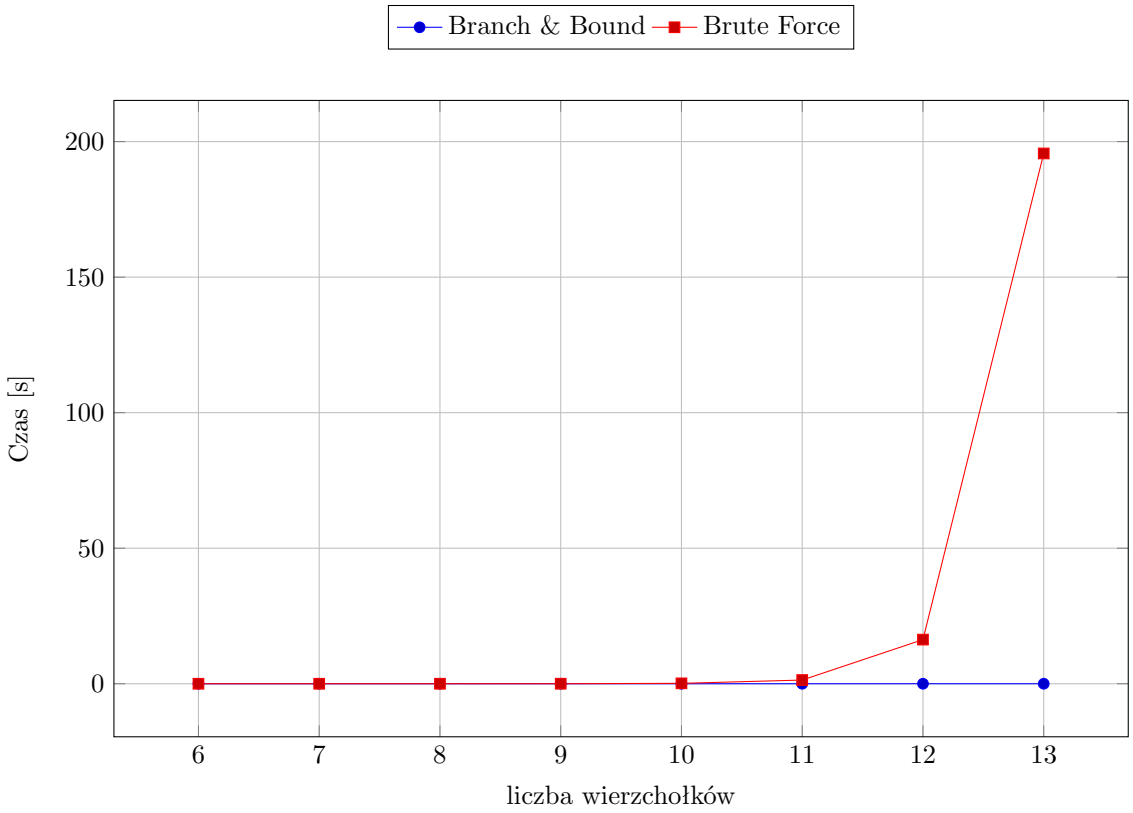
6.1 Wykres dla Brute Force



6.2 Wykres dla Branch and Bound



6.3 Wykres obu algorytmów



7 Wnioski

7.1 Brute Force

Analiza danych uzyskanych z przeprowadzonych pomiarów potwierdza zgodność wynikających z nich relacji z oszacowanymi wstępnie klasami złożoności obliczeniowej dla każdej z użytych metod. Algorytm implementujący przegląd zupełny wykazuje się teoretyczną klasą obliczeniową O(n!), co potwierdza złożoność wykładniczą tego rozwiązania.

7.2 Branch and Bound

Podobnie jak w przypadku przeglądu zupełnego, jego złożoność obliczeniowa jest ograniczona przez funkcję wykładniczą n!. Jednakże, dzięki zastosowaniu procedur rozgałęziania i ograniczania przeglądu możliwych rozwiązań, oraz dokładnemu doborowi funkcji obliczającej ograniczenie dolne, czas potrzebny do znalezienia rozwiązania dla każdej z testowanych instancji problemu został znacząco zredukowany w porównaniu do metody naiwnej.

8 źródła

https://www.youtube.com/watch?v=1FEP_sNb62k&ab_channel=AbdulBari
https://cs.pwr.edu.pl/zielinski/lectures/om/mow10.pdf
https://www.youtube.com/watch?v=nN4K8xA8ShM&t=927s&ab_channel=nptelhrd