

PROJEKTOWANIE EFEKTYWNYCH ALGORYMTÓW

Sprawozdanie z zadania 1 i 2

Krzysztof Czerniachowicz (233182)

Data złożenia: 21.11.2024

Spis treści

Opis problemu.....	2
Lista instancji, specyfikacja sprzętu.	3
Opis procedury badawczej.....	5
Wyniki.	6
Analiza wyników.	10

Opis problemu.

Problem komiwojażera (TSP - traveling salesman problem) polega na znalezieniu najkrótszej możliwej trasy pomiędzy zadanymi miastami, przebiegająca przez nie dokładnie raz. Poszukujemy więc najkrótszego możliwego cyklu Hamiltona w grafie ważonym.

Jest to problem NP-trudny, co oznacza, że nie ma znanego algorytmu, który mógłby rozwiązać go w czasie wielomianowym dla dowolnej liczby wierzchołków w grafie.

Praca ta będzie badać efektywność czasową sześciu wybranych metod rozwiązujących ten problem. W każdym przypadku poszukiwany jest czas potrzebny na znalezienie rozwiązania optymalnego. Jedynym wyjątkiem od tego jest metoda Najbliższego Sąsiada, która tego nie gwarantuje.

Badane instancje zostały wygenerowane przeze mnie osobiście wygenerowane. Przygotowany program zapewnia, że utworzony graf zawiera cykl Hamiltona, a następnie wypełnia go do zadanej wartości w pełni losowymi krawędziami. Wagi znajdują się w zakresie od 1 do 100.

Przygotowałem instancje zarówno symetryczne, jak i asymetryczne, pełne oraz wypełnione w 70%. W każdej grupie znajduje się 10 grafów o rozmiarach od 6 do 15. Za najmniejszy znany wynik uznałem ten, który uzyskałem w trakcie testowania za pomocą metody przeglądu zupełnego.

Badane metody:

- Losowa – w tej metodzie generujemy losowe cykle Hamiltona, do czasu aż wylosujemy najkrótszy możliwy. Moja implementacja tworzy cykl, losując każdy kolejny wierzchołek spośród sąsiadów ostatniego, do czasu aż będą do niego wszystkie dodane. Następnie obliczany jest jego koszt, jeśli jest on optymalny algorytm kończy działanie.
- Najbliższego sąsiada (Nearest Neighbour) – dla każdego wierzchołka dobieramy do ścieżki jego sąsiada o najmniejszym koszcie. Ta metoda nie daje gwarancji odnalezienia rozwiązania optymalnego.
- Przeglądu zupełnego, brute-force – pierwsza metoda, która gwarantuje odnalezienie optymalnego rozwiązania. Przygotowany algorytm bazuje na przeszukiwaniu grafu w głąb – w ten sposób odnajduje każdy możliwy cykl Hamiltona znajdujący się w instancji.
- Podziału i ograniczeń - metoda analizująca drzewo przestrzeni stanów, w przypadku TSP wszystkie możliwe ścieżki. W każdym węźle oblicza koszty dojścia do jego sąsiadów i odrzuca te węzły których koszt przekracza ograniczenie. Znacząco ogranicza to ilość możliwych ścieżek do sprawdzenia i pozwala szybciej znaleźć rozwiązanie. Utworzone przeze mnie algorytmy bazują na trzech różnych metodach przeszukiwania drzewa – przeszukiwania w głąb, w wszerz oraz metodę najmniejszego kosztu.

Lista instancji, specyfikacja sprzętu.

Z uwagi na to, że przygotowane instancje są w całości losowe, nie jest wyznaczona optymalna ścieżka. W trakcie testów ustaliłem minimalne koszty za pomocą metody brute-force lub branch and bound. Łącznie zostało wykorzystane 20 różnych grafów. Grafy pełne kończą swoją nazwę _100.txt lub p.txt. Pozostałe są zapełnione w 70%.

- Instancje asymetryczne:

- a6_100.txt 145
- a7_100.txt 142
- a8_100.txt 110
- a9_100.txt 129
- a10_100.txt 138
- a11_100.txt 200
- a12_100.txt 131
- a13_100.txt 148
- a14_100.txt 105
- a15_100.txt 179
- a6.txt 214
- a7.txt 333
- a8.txt 229
- a9.txt 240
- a10.txt 237
- a11.txt 285
- a12.txt 269
- a13.txt 268
- a14.txt 266
- a15.txt 289

- Instancje symetryczne

- s6p.txt 120
- s7p.txt 150
- s8p.txt 203
- s9p.txt 236
- s10p.txt 159
- s11p.txt 168
- s12p.txt 232
- s13p.txt 139
- s14p.txt 271
- s15p.txt 192
- s6.txt 134
- s7.txt 206
- s8.txt 262
- s9.txt 278

- s10.txt 297
- s11.txt 215
- s12.txt 444
- s13.txt 269
- s14.txt 275
- s15.txt 275

Badanie odbywało się na komputerze stacjonarnym z procesorem Intel Core i7-10700F o częstotliwości 2,9GHz, posiadającym 32 GB pamięci RAM. W trakcie badań nie działały żadne inne programy.

Opis procedury badawczej.

Badanie polegało na zbadaniu czasów rozwiązania problemu dla każdej metody z osobna. Dla każdej z nich badane były instancje asymetryczne i symetryczne. Rozmiar największej został tak dostosowany, aby czas rozwiązywania nie przekraczał 15 min.

Metoda losowa została zbadana 10 krotnie dla każdego pliku wejściowego. Miało to na celu obliczenie średniej, która w sposób miarodajny wyznaczy czas potrzebny na rozwiązanie problemu.

Pozostałe metody zostały zbadane jednokrotnie. Nie ma w nich żadnego elementu losowości, a uzyskiwanie wyniki w trakcie prób nie różniły się o więcej niż kilka milisekund dla danej instancji.

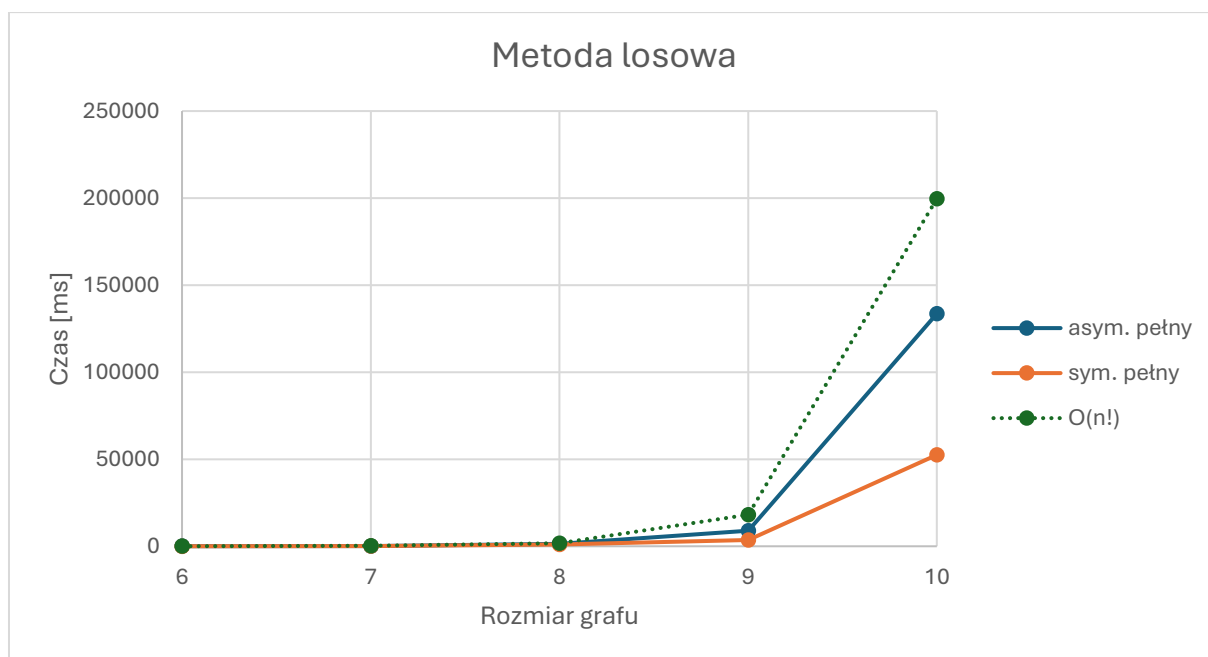
Rozmiar największej instancji dla metody losowej to 10. Algorytm bazujący na metodzie brute-force został zbadany dla instancji o maksymalnym rozmiarze 14 dla grafów pełnych i 13 dla wypełnionych w 70%. Metody NN oraz podziału i ograniczeń otrzymały największe instancje o 15 wierzchołkach.

Instancja s14p.txt spowodowała wystąpienie anomalii w trakcie badania metod podziału i ograniczeń przy przeszukiwaniu DFS i best first search. Występowała ona stale przy kolejnych próbach, dlatego zakładam że jest ona w pełni związana z tą instancją. Jej wyniki zostały pominięte przy analizie krzywej na wykresach.

Wyniki.

Tabela 1. Wyniki metody losowej

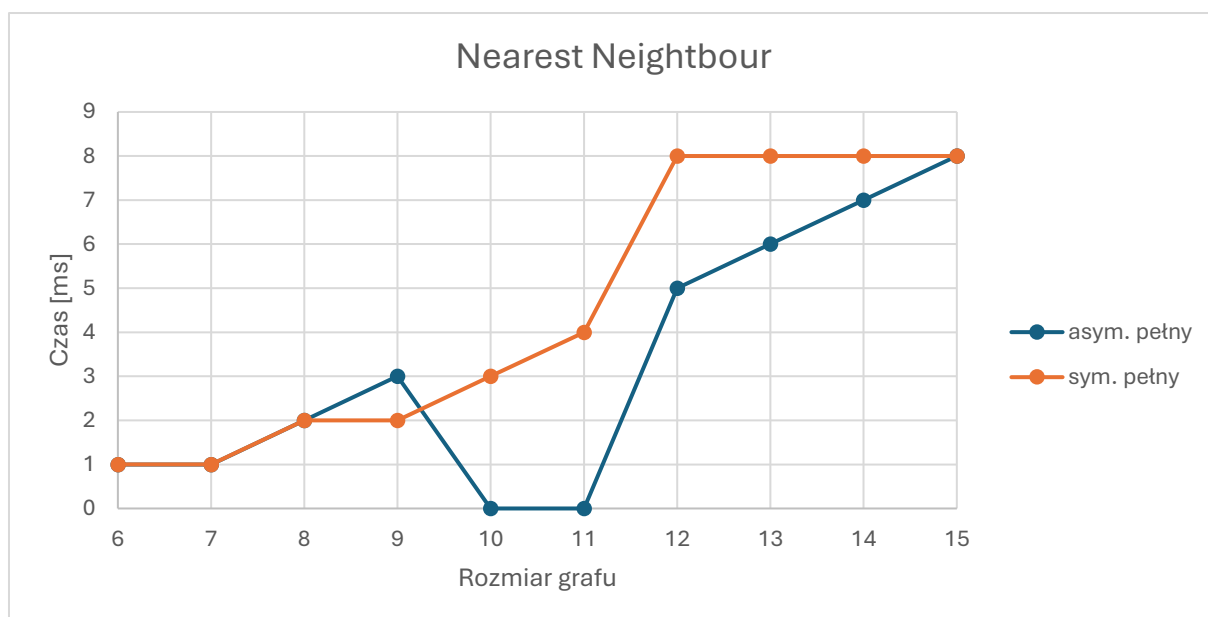
rodzaj \ rozmiar	6	7	8	9	10
asym. pełny	30	156	1430	8850	133581
sym. pełny	8	70	1097	3580	52484



Rysunek 1. Wykres porównujący typy instancji - metoda losowa

Tabela 2. Wyniki metody Nearest Neighbour

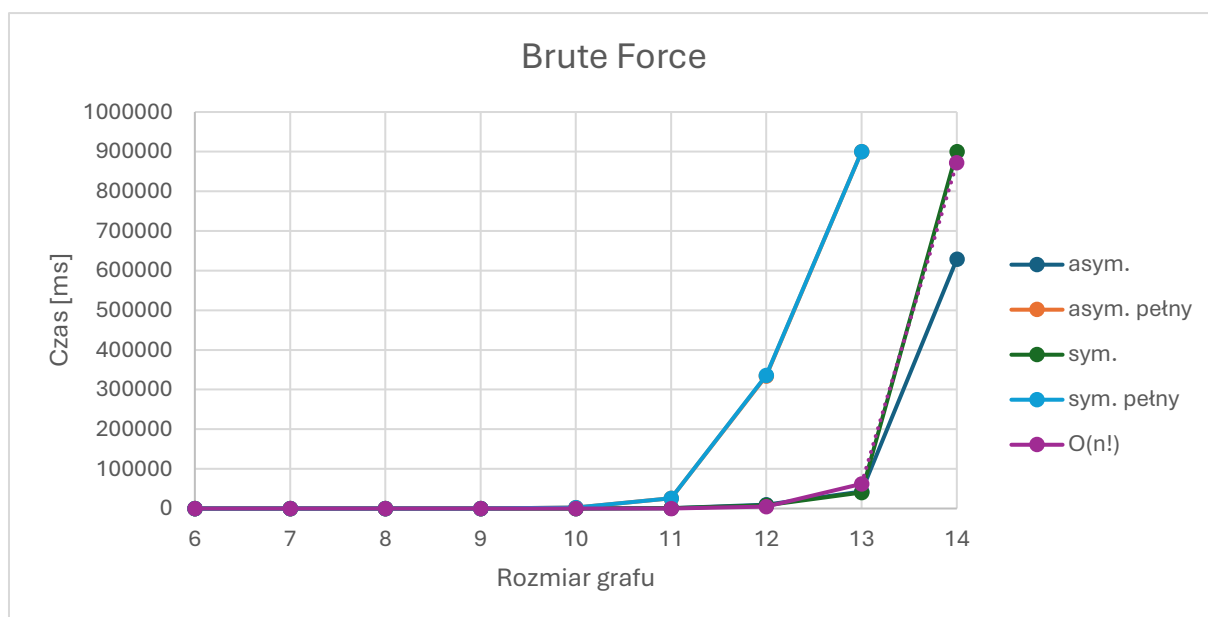
rodzaj \ rozmiar	6	7	8	9	10	11	12	13	14	15
asym. pełny	1	1	2	3	0	0	5	6	7	8
sym. pełny	1	1	2	2	3	4	8	8	8	8



Rysunek 2. Wykres porównujący typy instancji - metoda najbliższego sąsiada

Tabela 3. Wyniki metody brut-force

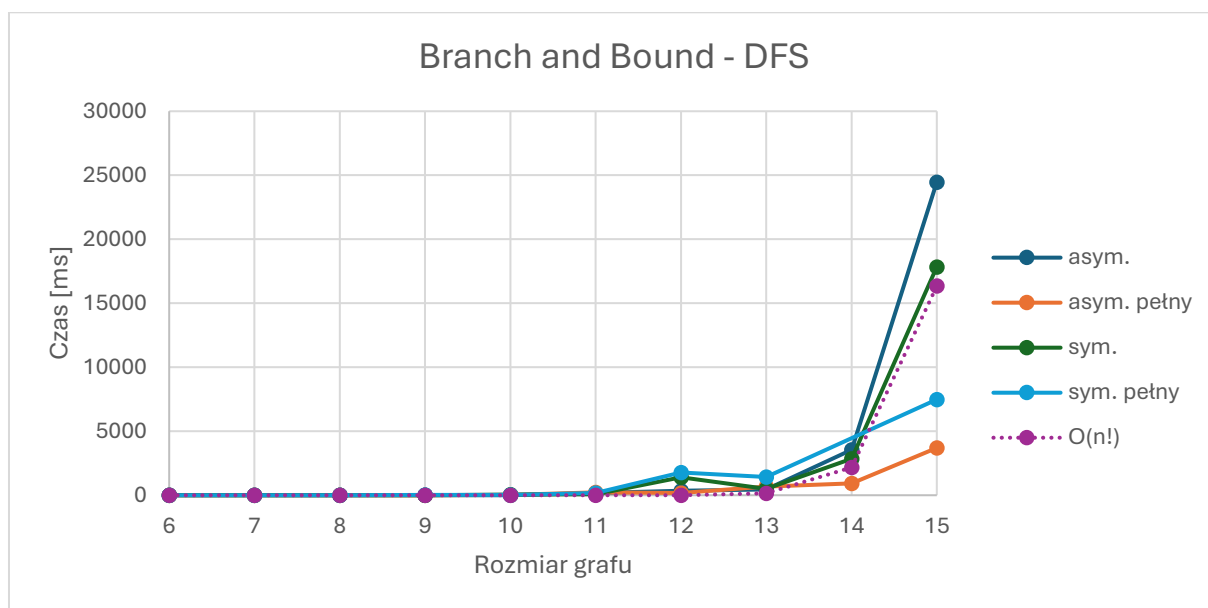
rodzaj \ rozmiar	6	7	8	9	10	11	12	13	14
asym.	0	0	2	21	141	1045	10061	42985	628703
asym. pełny	0	4	28	313	2798	25663	334308	900000	-
sym.	0	0	3	18	168	1153	7850	40586	900000
sym. pełny	3	5	28	311	2772	25591	335579	900000	-



Rysunek 3. Wykres porównujący typy instancji - metoda brute-force

Tabela 4. Wyniki metody BxB DFS

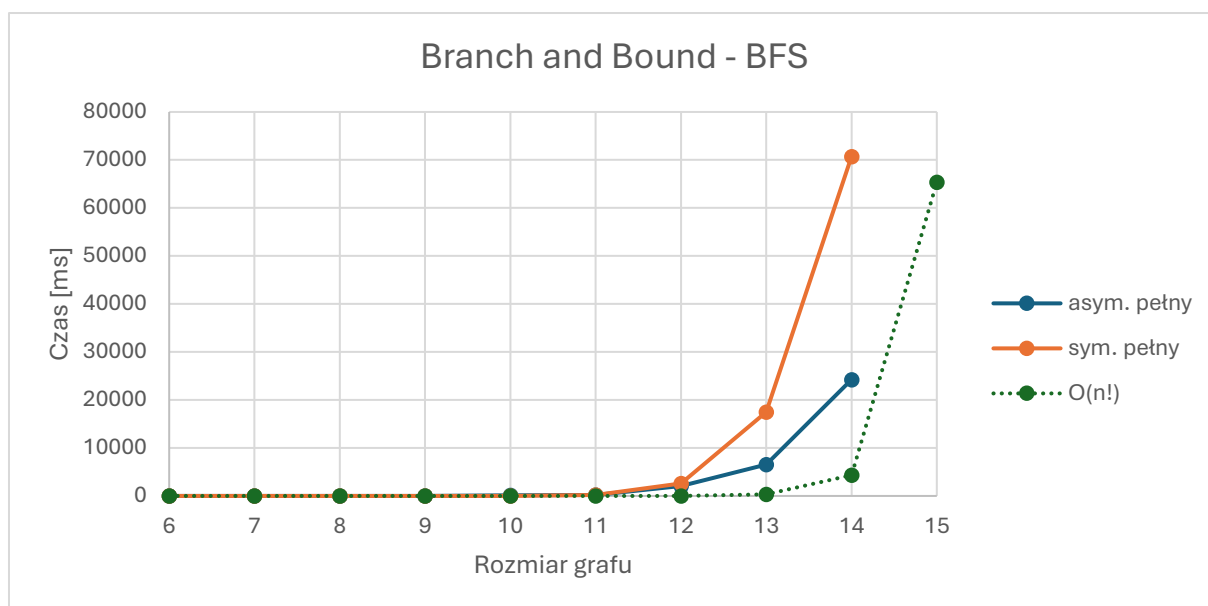
rodzaj \ rozmiar	6	7	8	9	10	11	12	13	14	15
asym.	0	0	1	7	8	166	353	446	3543	24465
asym. pełny	0	0	1	7	25	219	179	687	929	3697
sym.	0	3	1	5	38	63	1399	498	2840	17841
sym. pełny	0	1	5	20	22	166	1786	1433	16717	7482



Rysunek 4. Wykres porównujący typy instancji - metoda branch and bound DFS

Tabela 5. Wyniki metody BxB BFS

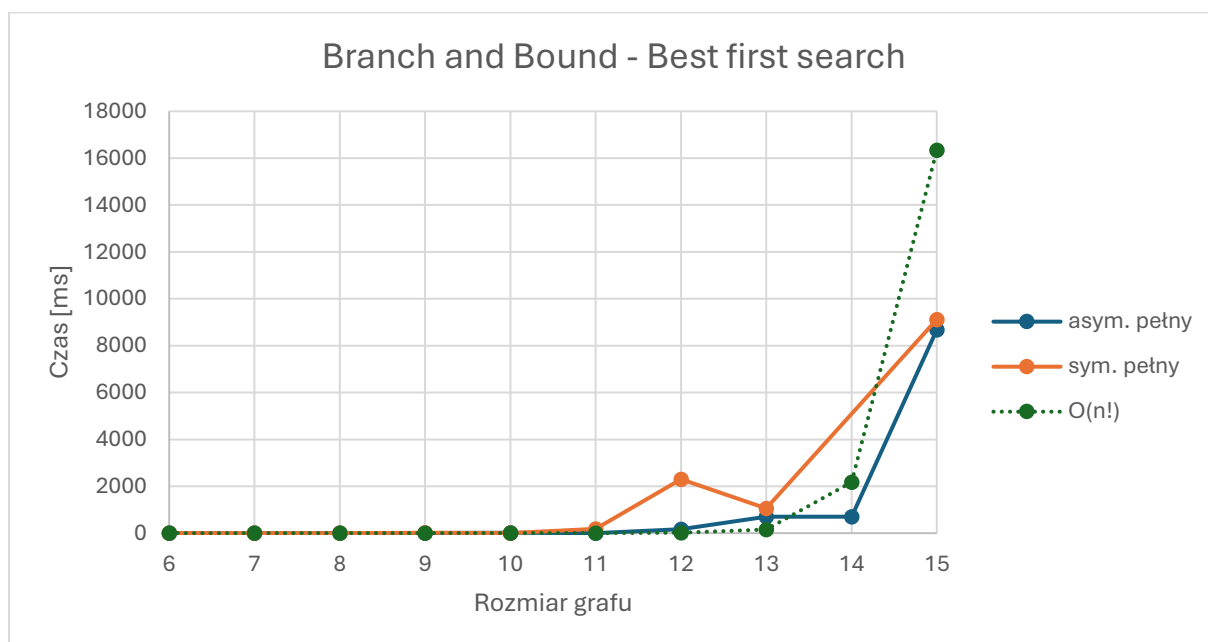
rodzaj \ rozmiar	6	7	8	9	10	11	12	13	14	15
asym. pełny	0	0	0	24	103	188	2048	6527	24236	900000
sym. pełny	0	0	0	22	8	215	2630	17482	70711	900000



Rysunek 5. Wykres porównujący typy instancji - metoda branch and bound BFS

Tabela 6. Wyniki metody BxB best search

rodzaj \ rozmiar	6	7	8	9	10	11	12	13	14	15
asym. pełny	0	0	1	7	15	0	167	708	696	8675
sym. pełny	1	0	0	22	10	176	2302	1055	29787	9118



Rysunek 6. Wykres porównujący typy instancji - branch and bound Best First

Analiza wyników.

Metoda losowa jest zdecydowanie najwolniejsza ze wszystkich, jej wyniki będące średnimi uzyskanych czasów już dla instancji o rozmiarze 10 przekraczają 2 minuty.

Rozwiązanie za pomocą metody najbliższych sąsiadów najbardziej różni się od pozostałych wyników. Jej złożoność obliczeniową można by określić jako zbliżoną do klasy $O(n)$ dla tak małych instancji. Trzeba jednak podkreślić że rzadko kiedy znajduje ona optymalne rozwiązanie problemu TSP. Ciężko z uzyskanych wyników określić czy rodzaj instancji miał wpływ na uzyskane wyniki.

W przypadku metody brute-force widać, że stopień wypełnienia grafu ma znaczenie na uzyskane wyniki. Rozwiązanie dla instancji pełnych jest rozwiązywane w wolniejszym czasie niż dla tych wypełnionych w 70%. Co widać na tabeli nr 3 to, że niepełne instancje mogą być większe – udało się uzyskać czas poniżej 15 min dla grafu asymetrycznego o rozmiarze 14. Dla grafów pełnych największe możliwe instancje miały po 12 wierzchołków.

Analizując tabele 4,5 i 6 zawierające wyniki dla metody podziału i ograniczeń, od razu rzuca się w oczy fakt, że zgodnie z założeniem uzyskane czasy są dużo niższe od poprzednich metody brute-force, a badane instancje większe. Jedynie ta bazująca na przeszukiwaniu wszerz nie podołała 15 wierzchołkowym grafom. Wyniki metody z przeszukiwaniem w głąb i najniższego kosztu są do siebie zbliżone z niewielką przewagą tej pierwszej.

Wpływ symetryczności instancji lub jej braku nie jest taki oczywisty. W przypadku brute-force nie robi znaczącej różnicy dla tych pełnych. Asymetryczne grafy wypadają gorzej w przypadku metody losowej i podziału i ograniczeń DFS, a jedyny przypadek gdzie dają niższe czasy rozwiązania to metoda branch and bound przy przeszukiwaniu wszerz.

Na każdym rysunku przedstawiającym wykresy, z wyłączeniem dotyczącego metody NN, widać zgodnie z założeniem że kształty krzywych przedstawiających wyniki są zbliżone do krzywej przedstawiającej złożoność $O(n!)$.

Najlepszymi z badanych metod jest metoda podziału i ograniczeń przy zastosowaniu przeszukiwania w głąb i przy minimum kosztów. W porównaniu do innych można określić je miarą efektywnych. Być może dały by radę rozwiązać problem TSP dla instancji o 16 wierzchołkach.