Projektowanie Efektywnych Algorytmów

Projekt

17/01/2023

259126 Maciej Fras

(7) Algorytm mrówkowy

1. Sformułowanie zadania

Zadane polega na opracowaniu, implementacji i zbadaniu efektywności algorytmu mrówkowego rozwiązującego problem komiwojażera w wersji optymalizacyjnej.

Problem komiwojażera (ang. Travelling salesman problem, TSP) to zagadnienie optymalizacyjne, polegające na znalezieniu minimalnego cyklu Hamiltona w pełnym grafie ważonym, gdzie:

Cykl Hamiltona – to taki cykl w grafie, w którym każdy wierzchołek grafu odwiedzany jest dokładnie raz (oprócz pierwszego wierzchołka)

Graf Ważony Pełny – struktura składająca się ze zbioru wierzchołków oraz zbioru krawędzi. Każdej krawędzi przypisana jest pewna wartość liczbowa.

Algorytmy mrówkowe to stosunkowo nowy gatunek algorytmów optymalizacyjnych, służący do rozwiązywania problemów kombinatorycznych i optymalizacji problemów przedstawionych na grafach.

Algorytm mrówkowy został zaproponowany przez włoskiego badacza, Marco Dorigo w 1992 r. Jest to rodzaj algorytmu opartego o fenomen zbiorowej inteligencji mrówek objawiającej się w umiejętności poszukiwania jedzenia dla swoich kolonii z wykorzystaniem naturalnych zmysłów. Mrówki komunikują się ze sobą pośrednio z wykorzystaniem *śladu feromonowego* – wraz ze wzrostem jego stężenia na danej drodze, staje się ona chętniej uczęszczana, co prowadzi do odnalezienia optymalnej drogi warunkującej przetrwanie kolonii.

W implementacji algorytmu mrówki również pozostawiają na odwiedzanych wierzchołkach grafu ślad feromonowy (będący tak naprawdę pewną wartością liczbową), pozwalający na podjęcie decyzji o wyborze najczęściej uczęszczanej drogi. Dodatkowo, w algorytmie swoje odzwierciedlenie znalazło rzeczywiste zjawisko *parowania feromonów,* dzięki któremu możliwe jest wybieranie optymalnych ścieżek. Gdyby nie było parowania, mrówki stwierdziłyby, że najkrótszą ścieżką jest tylko ta pierwotna.

Przy wykorzystaniu algorytmu mrówkowego oczekujemy, że złożoność obliczeniowa wyniesie:

$$O(CC \cdot n^2 \cdot m)$$

Ze względu na to, że istnieje liniowa zależność pomiędzy liczbą mrówek a liczbą miast sposób przedstawia złożoności obliczeniowej możemy uprościć do postaci:

$$O(CC * n^3)$$

Gdzie:

CC – liczba iteracji (cykli)
n – wielkość instancji rozwiązywanego problemu

Algorytm mrówkowy powinien umożliwić rozwiązanie problemu komiwojażera w dużo szybszym czasie niż dla algorytmów: naiwnego i Helda-Karpa, jednakże kosztem dokładności rozwiązania, ponieważ bazuje on na niedokładnym przeszukiwaniu przestrzeni rozwiązań z wykorzystaniem parametrów losowych.

W stosunku do algorytmu wykorzystującego metodę *symulowanego wyżarzania*, również oczekiwanym jest, że efektywność rozwiązywania problemu wzrośnie, jednakże w tym przypadku

różnica nie powinna być bardzo duża. Prawdopodobnie wyniesie ona maksymalnie jeden rząd wielkości.

Oczekiwane maksymalne wartości błędów bezwzględnych w stosunku do optymalnej wartości rozwiązania w zależności od wielkości badanej instancji n:

dla $n < 25$, 0%

dla $24 < n < 350$, 50%

dla $350 < n < 2500$, 150%.

Dodatkowo, spodziewamy się, że złożoność pamięciowa nie będzie stanowić przeszkody w badaniu działania algorytmu.

2. Opis metody

W rzeczywistości, mrówki poruszają się w sposób losowy; gdy znajdują pożywienie, wracają do swojej kolonii pozostawiając ślad składający się z feromonów (zjawisko *stygmergii*, czyli *„pośredniego sposobu porozumiewania się poprzez zmiany środowiska*"). Gdy inna mrówka natknie się na ten ślad, przestaje poruszać się w sposób losowy i podąża za śladem w kierunku pożywienia. Feromony z czasem ulatują i zaciera się ślad, dlatego pozostałe mrówki go wzmacniają, aż do momentu, gdy inna mrówka nie znajdzie krótszej ścieżki do jedzenia. Parowanie ma bardzo ważną zaletę - pozwala dobrać optymalną ścieżkę. Gdyby nie było parowania, mrówki stwierdziłyby, że najkrótszą ścieżką jest tylko ta pierwotna.

Implementacja algorytmu oraz przebieg jego wykonywania znajduje odzwierciedlenie w rzeczywistym sposobie funkcjonowania kolonii mrówek. Każda z mrówek rozwiązuje zadanie w sposób unikalny, za pomocą losowych kroków, przy okazji zostawiając po sobie pewnego rodzaju ślad. Rozwiązanie zadania jest tutaj analogią drogi pomiędzy kolonią a jedzeniem. Następnie rozwiązanie to jest oceniane, a dalej nadawana jest mu waga, która jest odpowiednikiem intensywności zapachu (feromonów). Ocena bazuje na ocenie jakości, którą można przedstawić jako długość drogi — im krótsza droga, tym lepsza jakość rozwiązania zadania.

Mrówki w zaimplementowanym algorytmie możemy określić mianem „przeszukiwczy", którzy wspólnym wysiłkiem dążą do odnalezienia optymalnego rozwiązania problemu. Przyjmujemy, że mrówki posiadają kilka wspólnych cech:

- Mają możliwość tworzenia nowych zadań w sposób losowy
- Są prymitywne i indywidualnie zdolne do podejmowania decyzji
- Posiadają idealny sposób oceny jakości rozwiązania. W tym celu wykorzystują feromon
- Mogą posiadać pamięć węzłów odwiedzonych – *lista tabu*

W algorytmie stosowane są następujące parametry:

- α – określa wpływ doświadczeń poprzednich pokoleń; wybór dokonywany jest na podstawie wartości τ_{ij} – ilości informacji o liczbie wyborów określonej trasy przez inne osobniki.
- β – określa siłę wyboru zachłannego drogi na podstawie odległości pomiędzy miastami; wybór dokonywany jest na podstawie wartości η_{ij} – stopień widoczności miasta j z miasta i
- ρ – współczynnik z przedziału <0,1>, określający ilość wyparowującego feromonu w jednostce czasu (1 - wyparowuje cały, 0 – wyparowuje nic)
- m – liczba mrówek
- τ_0 – początkowa zawartość feromonu na krawędziach

Typowe rozwiązanie problemu algorytmem mrówkowm wygląda następująco:

1. Stwórz reprezentację zadania w postaci zadania do wykonania na grafie
2. Ustal początkową ilość feromonów na krawędziach

Wzór pozwalający na obliczenie początkowego stężenia feromonu na krawędziach:

$$\tau 0 = \frac{m}{C^{nn}}$$

Gdzie:

M – liczba mrówek
C^{nn} – Szacowana długość trasy

3. Ustal dodatkowe reguły za pomocą których agenci będą się poruszać wewnątrz grafu (zwykle jest to pamiętanie tablicy tabu już odwiedzonych wierzchołków, a także wybór pomocniczej heurystyki)
4. Zdecyduj w jaki sposób ruch agentów będzie wpływać na poziom feromonów na krawędziach grafu, a także jak oni sami będą się posiłkować już położonymi feromonami

Wzór określający aktualizację stężenia feromonu w każdym kroku:

$$\tau ij(t + 1) = \rho \cdot \tau ij(t) + \Delta\tau ij(t, t + 1)$$

5. Iteracyjnie wpuszczaj nowe grupy agentów (po usunięciu starej grupy) i pozwalaj im przemieszczać się pomiędzy wierzchołkami grafu zgodnie ze stworzonymi regułami

Prawdopodobieństwo wyboru miasta *j* przez *k*-tą mrówkę w mieście *i* dane jest wzorem:

$$p_{ij} = \begin{cases} \dfrac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{c_{i,l}\in\Omega}(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}} & \bigvee c_{i,l} \in \Omega \\ 0 & \bigvee c_{i,l} \notin \Omega \end{cases}$$

gdzie:

c - kolejne możliwe (nie znajdujące się na liście tabu miasto)
Ω - dopuszczalne rozwiązanie (nieodwiedzone miasta, nienależące do $tabu_k$)
η_{ij} -wartość lokalnej funkcji kryterium; np. $\eta = \frac{1}{d_{ij}}$ (visibility), czyli odwrotność odległości pomiędzy miastami
α - parametr regulujący wpływ τij
β - parametr regulujący wpływ ηij.

6. Po każdej iteracji pamiętaj najlepsze dotychczas znalezione rozwiązanie (w liście tabu)

Wyróżniamy trzy rodzaje algorytmów w systemie mrówkowym:

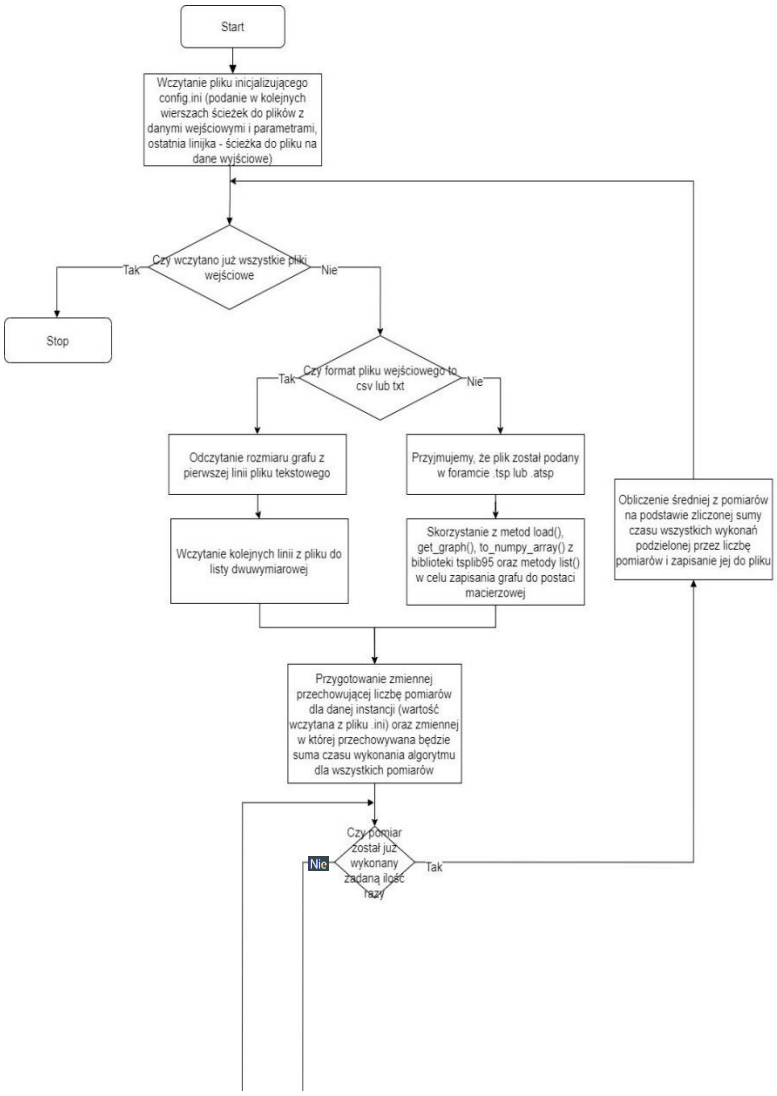
• Algorytm Gęstościowy (ang. ant-density DAS)
• Algorytm Ilościowy (ang. ant-quantity QAS)
• Algorytm Cykliczny (ang. ant-cycle CAS)

W sprawozdaniu zbadane zostaną algorytmy DAS oraz QAS

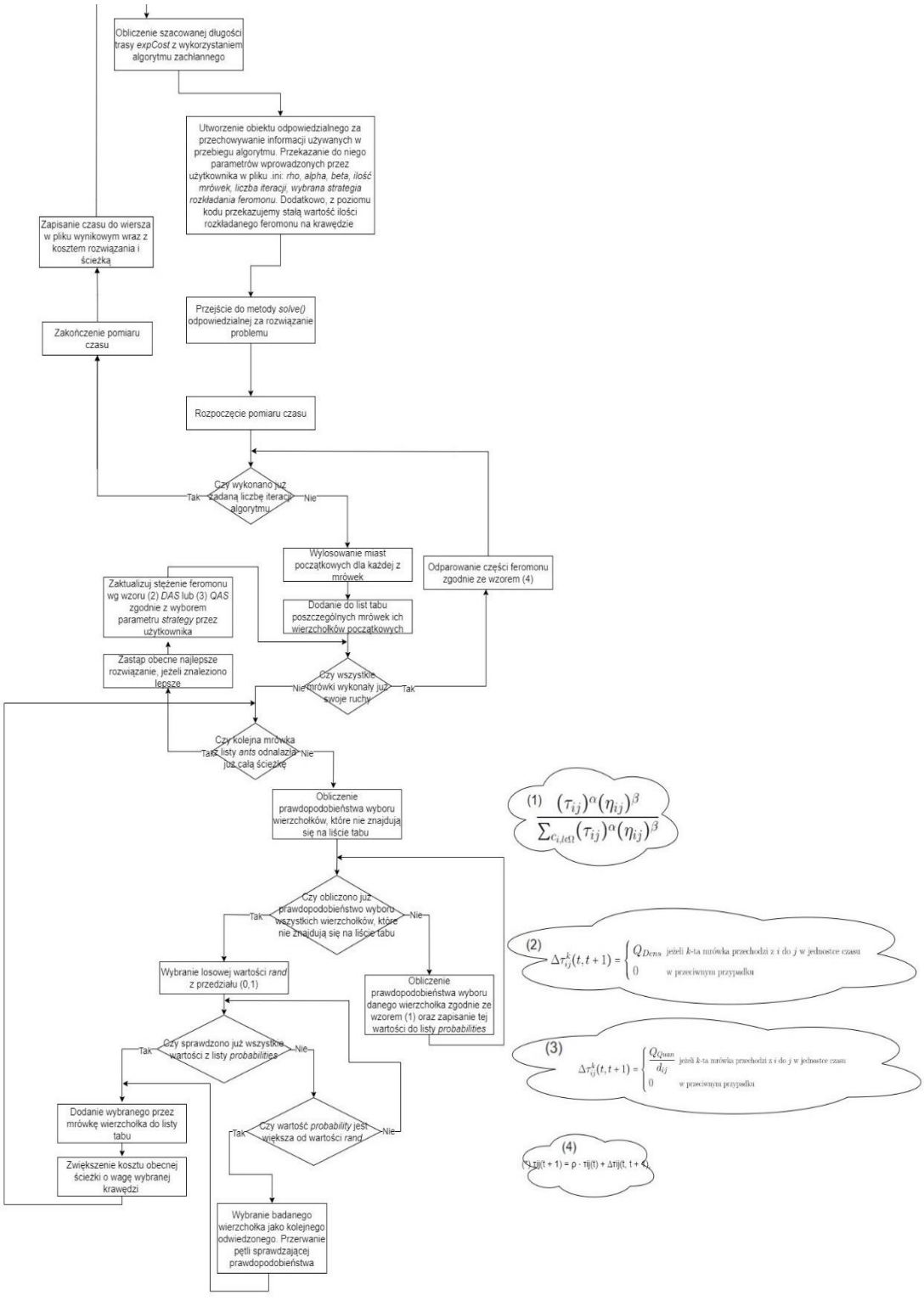
- W DAS przy przejściu po krawędzi *(i, j)* na każdą jednostkę jej długości rozkładana jest stała ilość feromonu, niezależnie od długości krawędzi
- W QAS przy przejściu po krawędzi *(i, j)*, stała ilość feromonu dzielona jest przez długości krawędzi d_{ij}.

Dokładny sposób działania zaimplementowanego algorytmu został zilustrowany na schemacie blokowym w *punkcie 3: Opis algorytmu* na *rysunku 1* oraz *rysunku 2*.

3. Opis algorytmu



Rysunek 1 – Pierwsza część schematu blokowego opisującego rozwiązanie problemu komiwojażera z wykorzystaniem algorytmu mrówkowego



Rysunek 2 – Druga część schematu blokowego opisującego rozwiązanie problemu komiwojażera z wykorzystaniem algorytmu mrówkowego

4. Dane testowe

Dane testowe są wczytywane zgodnie z procedurą przedstawioną na *rysunku* 1.

Do sprawdzenia poprawności działania algorytmu oraz do wykonania badań wybrano następujący zestaw instancji, dla których podano również najlepsze znane rozwiązanie:

Dane pochodzące ze strony Dr. Jarosława Mierzwy

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

tsp_10.txt - 212
tsp_12.txt - 264
tsp_13.txt -269
tsp_15.txt - 291

Dane z biblioteki tsp-lib

http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsp/index.html

gr17.tsp - 2085
gr21.tsp - 2707
gr24.tsp - 1272
gr48.tsp - 5046
gr96.tsp – 55209
kroA100.tsp - 21281
kro124p.atsp – 36230
kroB150.tsp - 26130
kroB200.tsp - 29437
lin318.tsp - 42029
fl417.tsp - 11861
pa561.tsp – 2763
pr1002.tsp – 259045

5. Procedura badawcza

Należało zbadać zależność czasu rozwiązania problemu od wielkości instancji. Zdefiniowane zostały następujące parametry mające zasadniczy wpływ na czas wykonania oraz dokładność znalezionego rozwiązania:

- Liczba mrówek m
- Liczba iteracji algorytmu
- Parametr α
- Parametr β
- Parametr ρ
- Strategia rozkładania feromonu na krawędziach

Parametry te podawane są przez użytkownika w pliku config.ini

Dodatkowo dla wszystkich pomiarów ustawiono z poziomu kodu wartość parametru Q = 10 (z wyjątkiem dla badań dla strategii QAS, gdzie Q = 1000)

Procedura badawcza polegała na uruchomieniu programu sterowanego plikiem konfiguracyjnym *config.ini*

Przykładowa zawartość gotowego pliku:

```
Nazwa pliku, Liczba testów, Optymalny koszt, Optymalna ścieżka,
m - Liczba mrówek, liczba iteracji, alba, beta, rho, strategia (1 -
gęstościowy , 2 - ilościowy, 3 - cykliczny)
gr24.tsp 2 1272 [] 24 25 1.0 3.0 0.5 1
gr48.tsp 1 5046 [] 48 10 1.0 3.0 0.5 1
wyniki.csv
```

Listing1: plik konfiguracyjny *config.ini*

Do pliku wyjściowego zapisywany był czas wykonania, otrzymane rozwiązanie (koszt ścieżki) oraz ścieżka (numery kolejnych odwiedzanych węzłów). Zapisywane są również: wykorzystane parametry dla badanej instancji, procentowa wartość określająca o ile obliczona wartość kosztu odbiega od podanej optymalnej, uśredniony wynik pomiaru, wartość średniego błędu pomiaru, największy znaleziony koszt i najmniejszy znaleziony koszt. Plik wyjściowy może być zapisany w wybranym formacie: *txt* lub *csv*. Poniżej przedstawiono zawartość pliku wyjściowego dla przykładowych pomiarów.

```
['gr24.tsp', '2', '1272', '[]']
m: 24 iteracje: 25 alpha: 1.0 beta: 3.0 rho: 0.5 strategia: DAS
---
0.1765279769897461 1307.0 2.75% [16, 9, 4, 23, 5, 15, 0, 11, 3, 22, 8, 12, 13,
19, 1, 14, 18, 21, 17, 2, 10, 7, 6, 20]
0.14584708213806152 1302.0 2.36% [16, 9, 4, 23, 5, 6, 7, 20, 2, 10, 15, 0, 11,
3, 22, 8, 12, 13, 19, 14, 1, 18, 21, 17]
---
Usredniony wynik pomiaru dla 2 instancji: 0.1611875295639038
Sredni blad: 2.55%
Najwiekszy znaleziony koszt: 1307.0, blad: 2.75%, sciezka: [16, 9, 4, 23, 5,
15, 0, 11, 3, 22, 8, 12, 13, 19, 1, 14, 18, 21, 17, 2, 10, 7, 6, 20]
```

```
Najmniejszy znaleziony koszt: 1302.0, blad: 2.36%, sciezka: [16, 9, 4, 23, 5,
6, 7, 20, 2, 10, 15, 0, 11, 3, 22, 8, 12, 13, 19, 14, 1, 18, 21, 17]
Najbardziej dokladny znaleziony koszt: 1302.0, blad: 2.36%, sciezka: [16, 9,
4, 23, 5, 6, 7, 20, 2, 10, 15, 0, 11, 3, 22, 8, 12, 13, 19, 14, 1, 18, 21,
17]%
--------------------------------------------------------------------------------
['gr48.tsp', '1', '5046', '[]']
m: 48 iteracje: 10 alpha: 1.0 beta: 3.0 rho: 0.5 strategia: DAS
---
0.39999914169311523 5646.0 11.89% [40, 43, 27, 6, 28, 12, 47, 10, 15, 35, 5,
25, 7, 21, 8, 13, 20, 31, 26, 16, 4, 11, 9, 30, 32, 14, 23, 36, 46, 17, 45,
33, 22, 24, 2, 18, 3, 29, 37, 19, 39, 38, 41, 34, 1, 44, 42, 0]
---
Usredniony wynik pomiaru dla 1 instancji: 0.39999914169311523
Sredni blad: 11.89%
Najwiekszy znaleziony koszt: 5646.0, blad: 11.89%, sciezka: [40, 43, 27, 6,
28, 12, 47, 10, 15, 35, 5, 25, 7, 21, 8, 13, 20, 31, 26, 16, 4, 11, 9, 30, 32,
14, 23, 36, 46, 17, 45, 33, 22, 24, 2, 18, 3, 29, 37, 19, 39, 38, 41, 34, 1,
44, 42, 0]
Najmniejszy znaleziony koszt: 5646.0, blad: 11.89%, sciezka: [40, 43, 27, 6,
28, 12, 47, 10, 15, 35, 5, 25, 7, 21, 8, 13, 20, 31, 26, 16, 4, 11, 9, 30, 32,
14, 23, 36, 46, 17, 45, 33, 22, 24, 2, 18, 3, 29, 37, 19, 39, 38, 41, 34, 1,
44, 42, 0]
Najbardziej dokladny znaleziony koszt: 5646.0, blad: 11.89%, sciezka: [40, 43,
27, 6, 28, 12, 47, 10, 15, 35, 5, 25, 7, 21, 8, 13, 20, 31, 26, 16, 4, 11, 9,
30, 32, 14, 23, 36, 46, 17, 45, 33, 22, 24, 2, 18, 3, 29, 37, 19, 39, 38, 41,
34, 1, 44, 42, 0]%
--------------------------------------------------------------------------------
```

Listing 2: zawartość przykładowego pliku z wynikami *wyniki.csv*

Pomiar czasu został wykonany za pomocą funkcji time() z modułu time języka Python. Funkcja ta zwraca czas POSIX, czyli liczbę sekund, które upłynęły od 1 stycznia 1970 r. Procedura pomiaru czasu składa się z dwukrotnego wywołania tejże funkcji – przed rozpoczęciem wykonywania algorytmu oraz zaraz po jego ukończeniu. Wynikiem pomiaru czasu jest różnica tych wywołań.

Pomiar czasu odbywa się dla każdego wykonania funkcji rozwiązującej problem komiwojażera. Każdy pojedynczy wynik pomiaru zapisywany jest do pliku wyjściowego. Czasy wszystkich wywołań są sumowane do jednej zmiennej, a gdy zakończy się pętla sprawdzająca jak wiele razy problem ma być rozwiązany suma czasów dzielona jest przez liczbę wywołań dzięki czemu uzyskujemy średnią z pomiarów, która zapisywana jest do wynikowego pliku .csv lub .txt.

Pomiarów dokonano z wykorzystaniem interpretera języka Python o nazwie *PyPy* w celu zwiększenia wydajności algorytmu. Interpreter ten okazał się być 5x – 10x szybszy w stosunku do domyślnego rozwiązania *cpython*.

Pomiarów dokonano na laptopie Lenovo Legion 15 o specyfikacji:

Pamięć Ram: 16GB DDR4
Procesor: Intel Core i5 8300H 4,0GHz (4 rdzenie, 8 wątków)
Dysk: SSD 512gb nvme
Karta graficzna: GeForce GTX 1050Ti

6. Wyniki

Wyniki zostały zgromadzone w osobnych katalogach z podziałem na cel badania. Wyniki opracowane zostały w programie MS Excel.

1) Wyniki zgromadzone w celu pomiaru **wydajności algorytmu**

Badanie to miało na celu osiągnięcie jak najlepszych wyników działania algorytmu przy zachowaniu możliwe niskiej wartości błędu względem najlepszego znanego optymalnego rozwiązania problemu.

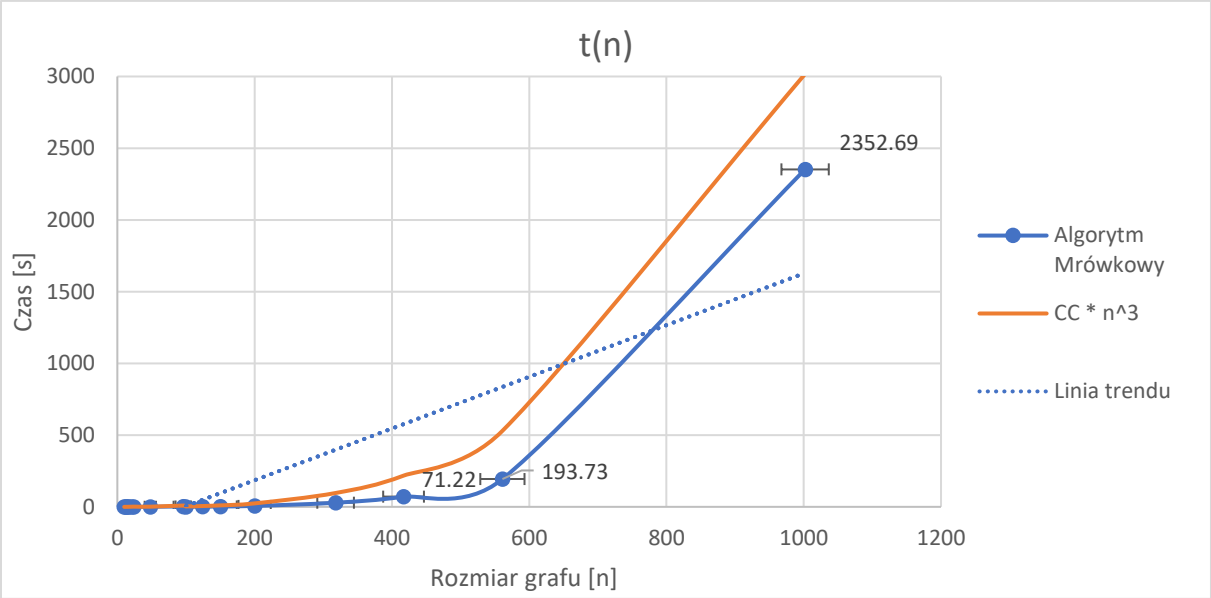
Parametry:

- Liczba mrówek m = wielkości danej instancji n
- Parametr α = 1.0
- Parametr β = 3.0
- Parametr ρ = 0.5
- Parametr Q = 10
- Strategia rozkładania feromonu na krawędziach: DAS

Tabela 1: Wyniki pomiaru czasu rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego

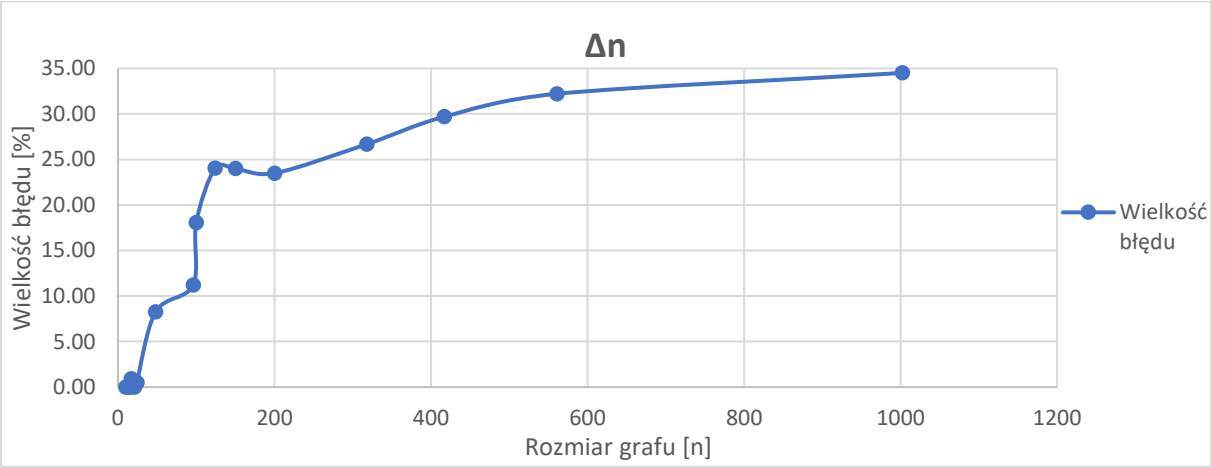
Instancja	Czas [s]	Wynik	Wynik Optymalny	Błąd [%]	Liczba iteracji
tsp10.txt	0.03	212	212	0.00	3
tsp12.txt	0.02	264	264	0.00	3
tsp13.txt	0.01	269	269	0.00	3
tsp15.txt	0.01	291	291	0.00	3
gr17.tsp	0.25	2104	2085	0.91	100
gr21.tsp	0.41	2707	2707	0.00	100
gr24.tsp	0.36	1278	1272	0.47	100
gr48.tsp	0.39	5463	5046	8.26	10
gr96.tsp	2.54	61395	55209	11.20	10
kroA100.tsp	0.92	25127	21281	18.07	3
kro124p.atsp	1.12	44931	36230	24.02	3
kroB150.tsp	2.73	32400	26130	24.00	3
kroB200.tsp	6.37	36353	29437	23.49	3
lin318.tsp	28.65	53244	42029	26.68	3
fl417.tsp	71.22	15383	11861	29.69	3
pa561.tsp	193.73	3653	2763	32.21	3
pr1002.tsp	2352.69	348404	259045	34.50	3

Wyniki przedstawione zostały w postaci wykresu zależności czasu uzyskania rozwiązania problemu od wielkości instancji na *rysunku 3* na podstawie danych z *tabeli 1*. W celu oceny poprawności wykonania algorytmu, obok wyników pomiaru naniesiona została również krzywa O($CC * n^3$) skorygowana o współczynnik równy 0.00001, tak aby móc czytelnie porównać tę zależność z otrzymanym rezultatem. Krzywa ta obrazuje teoretyczny, wzorcowy rozkład zależności czasu wykonania algorytmu od wielkości instancji.



Rysunek 3: Wpływ wielkości instancji na czas uzyskania rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego

Nałożenie krzywej $CC * n^3$ potwierdza, że badany algorytm wyznacza rozwiązania problemu komiwojażera dla badanych instancji w czasie $CC * n^3$ zależnym względem wielkości instancji (obie krzywe są zgodne co do kształtu). Złożoność czasowa opracowanego algorytmu wynosi $O(CC * n^3)$



Rysunek 4: Wpływ wielkości instancji na wielkość błędu względnego (wyrażonego względem najlepszego znanego rozwiązania danego problemu) w rozwiązywaniu problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego.

2) Wyniki zgromadzone w celu porównania **różnych schematów rozkładu feromonu** (DAS i QAS)

wyniki_qas.csv wyniki_das.csv

Celem tego badania było sprawdzenie różnic w wynikach działania algorytmu mrówkowego dla różnych schematów rozkładu feromonu, ze szczególnym zwróceniem uwagi na wielkość błędu pomiaru.

Parametry:

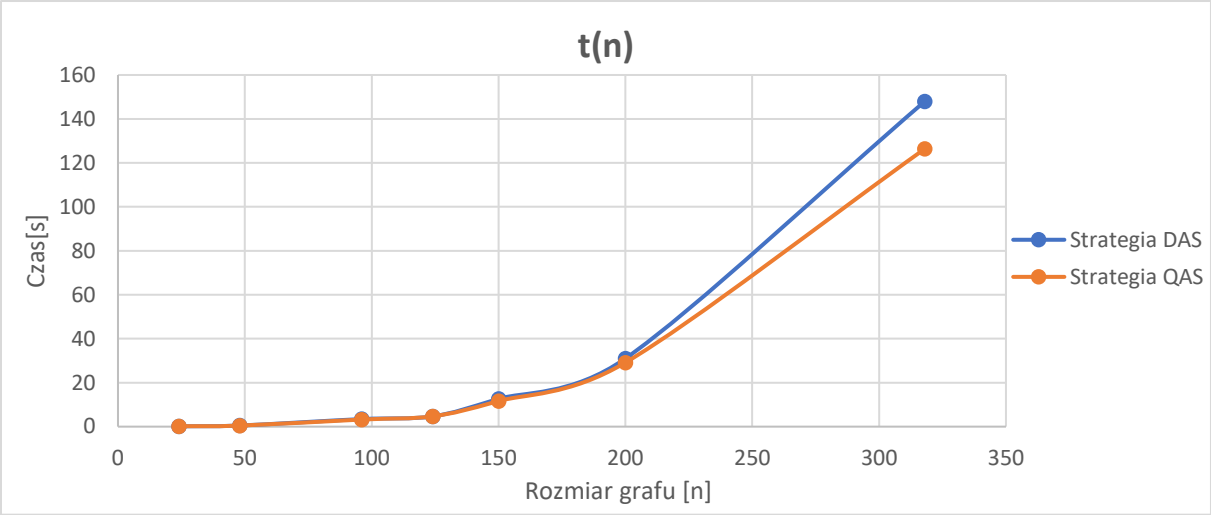
- Liczba mrówek m = wielkości danej instancji n
- Parametr α = 1.0
- Parametr β = 3.0
- Parametr ρ = 0.5
- Parametr Q = 10 (DAS), Q = 1000 (QAS)
- Liczba iteracji algorytmu = 10

Tabela 2: Wyniki pomiaru czasu rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego – badanie ze względu na schemat rozkładu feromonu (DAS)

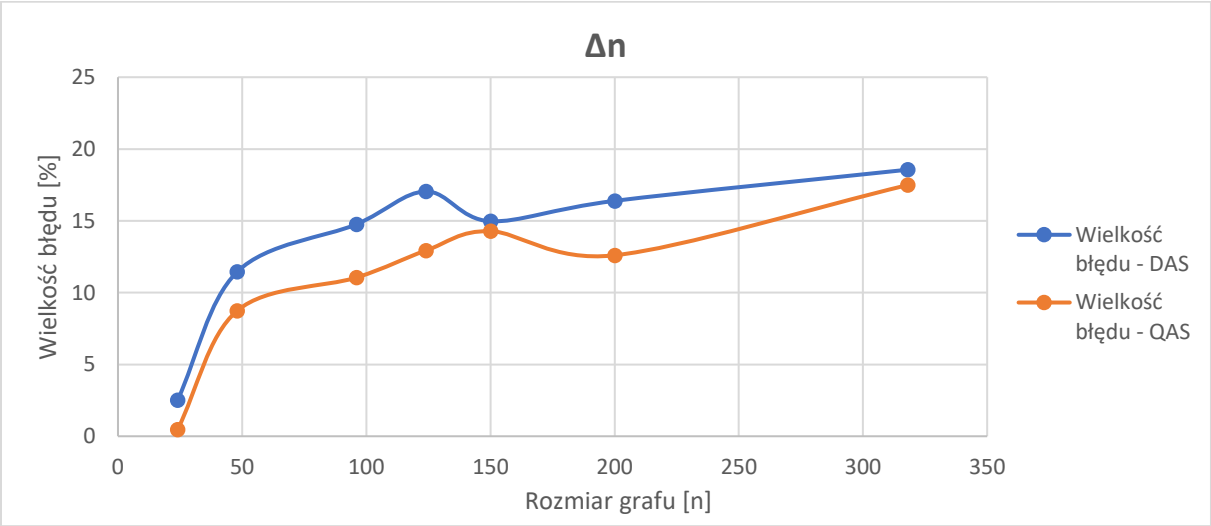
Instancja	Czas [s]	Wynik	Wynik Optymalny	Błąd [%]
gr24.tsp	0.15	1304	1272	2.52
gr48.tsp	0.52	5624	5046	11.45
gr96.tsp	3.45	63356	55209	14.76
kro124p.atsp	4.69	42404	36230	17.04
kroB150.tsp	12.67	30042	26130	14.97
kroB200.tsp	31.00	34262	29437	16.39
lin318.tsp	147.97	49831	42029	18.56

Tabela 3: Wyniki pomiaru czasu rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego – badanie ze względu na schemat rozkładu feromonu (QAS)

Instancja	Czas [s]	Wynik	Wynik Optymalny	Błąd [%]
gr24.tsp	0.18	1278	1272	0.47
gr48.tsp	0.48	5487	5046	8.74
gr96.tsp	3.23	61308	55209	11.05
kro124p.atsp	4.71	40914	36230	12.93
kroB150.tsp	11.66	29862	26130	14.28
kroB200.tsp	29.14	33146	29437	12.60
lin318.tsp	126.39	49379	42029	17.49



Rysunek 5: Wpływ strategii rozkładu feromonu przy rozwiązywaniu problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego na czas pomiaru



Rysunek 6: Wpływ strategii rozkładu feromonu przy rozwiązywaniu problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego na błąd pomiaru

Na podstawie *rysunku 6* można zauważyć, że wybranie strategii QAS z odpowiednio dobraną wartością współczynnika *Q* pozwoliło na uzyskanie dokładniejszych wyników pomiaru w stosunku do DAS. Wybór strategii nie miał istotnego wpływu na czas rozwiązania problemu.

3) Wyniki zgromadzone w celu porównania wpływu **parametru α** na wynik pomiaru

wyniki_alfa1,0.csv wyniki_alfa0.csv wyniki_alfa0,75.csv wyniki_alfa0,50.csv wyniki_alfa0,25.csv

Celem tego badania było sprawdzenie różnic w wynikach działania algorytmu mrówkowego dla różnych wartości parametru α określającego wpływ doświadczeń poprzednich pokoleń.

Parametry:

- Liczba mrówek m = wielkości danej instancji n
- Zmienny parametr α
- Parametr β = 3.0
- Parametr ρ = 0.5
- Parametr Q
- Wykorzystano strategię DAS
- Liczba iteracji algorytmu = 10

Tabela 4: Wyniki pomiaru czasu rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego – badanie ze względu na wartość parametru α (α = 0.0)

Instancja	Czas [s]	Wynik	Wynik Optymalny	Błąd [%]
gr24.tsp	0.16	1604	1272	26.10
gr48.tsp	0.45	6661	5046	32.01
gr96.tsp	2.27	85785	55209	55.38
kro124p.atsp	3.29	58964	36230	62.75
kroB150.tsp	8.82	43859	26130	67.85
kroB200.tsp	21.60	52875	29437	79.62
lin318.tsp	112.70	76997	42029	83.20

Tabela 5: Wyniki pomiaru czasu rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego – badanie ze względu na wartość parametru α (α = 0.25)

Instancja	Czas [s]	Wynik	Wynik Optymalny	Błąd [%]
gr24.tsp	0.15	1347	1272	5.90
gr48.tsp	0.52	6142	5046	21.72
gr96.tsp	3.45	72534	55209	31.38
kro124p.atsp	4.69	50675	36230	39.87
kroB150.tsp	12.67	35807	26130	37.03
kroB200.tsp	31.00	42428	29437	44.13
lin318.tsp	147.97	62068	42029	47.68

Tabela 6: Wyniki pomiaru czasu rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego – badanie ze względu na wartość parametru α (α = 0.50)

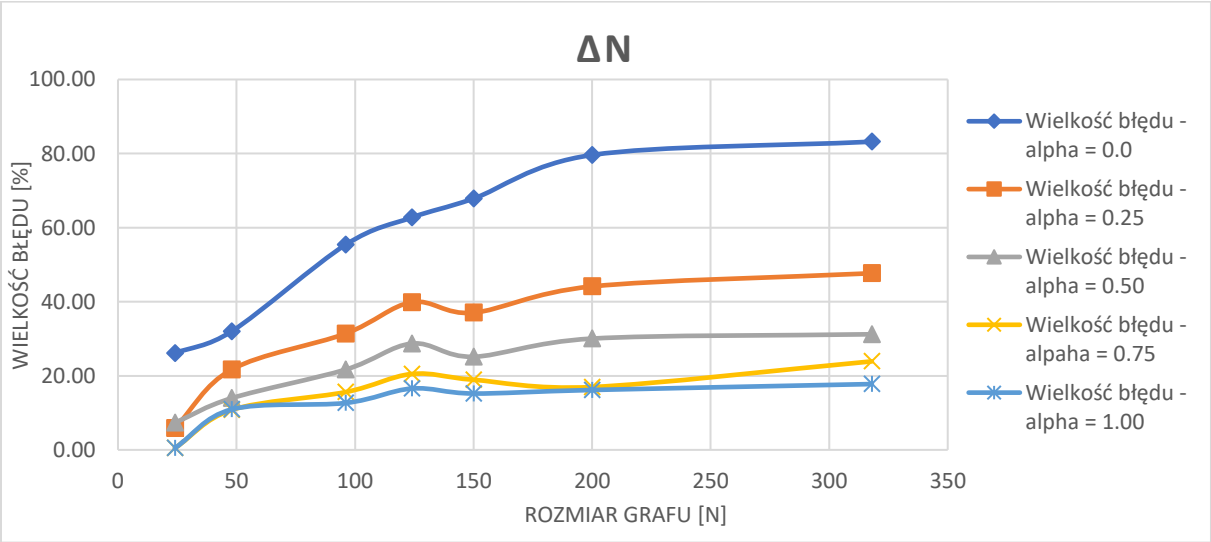
Instancja	Czas [s]	Wynik	Wynik Optymalny	Błąd [%]
gr24.tsp	0.17	1365	1272	7.31
gr48.tsp	0.62	5751	5046	13.97
gr96.tsp	3.74	55209	55209	21.67
kro124p.atsp	5.03	46628	36230	28.70
kroB150.tsp	15.39	32698	26130	25.14
kroB200.tsp	35.41	38284	29437	30.05
lin318.tsp	159.68	55147	42029	31.21

Tabela 7: Wyniki pomiaru czasu rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego – badanie ze względu na wartość parametru α (α = 0.75)

Instancja	Czas [s]	Wynik	Wynik Optymalny	Błąd [%]
gr24.tsp	0.19	1278	1272	0.47
gr48.tsp	0.61	5591	5046	10.80
gr96.tsp	3.82	63824	55209	15.60
kro124p.atsp	5.28	43653	36230	20.49
kroB150.tsp	13.45	31077	26130	18.93
kroB200.tsp	34.91	34433	29437	16.97
lin318.tsp	151.17	52076	42029	23.90

Tabela 8: Wyniki pomiaru czasu rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego – badanie ze względu na wartość parametru α (α = 1.00)

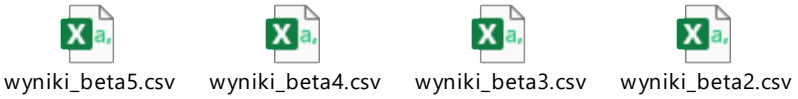
Instancja	Czas [s]	Wynik	Wynik Optymalny	Błąd [%]
gr24.tsp	0.17	1278	1272	0.47
gr48.tsp	0.52	5598	5046	10.94
gr96.tsp	3.50	62196	55209	12.66
kro124p.atsp	4.86	42229	36230	16.56
kroB150.tsp	11.98	30104	26130	15.21
kroB200.tsp	31.12	34198	29437	16.17
lin318.tsp	138.17	49497	42029	17.77



Rysunek 7: Wpływ wartości parametru α przy rozwiązywaniu problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego na błąd pomiaru

Na podstawie *Rysunku 7* można stwierdzić, że wraz ze wzrostem wartości parametru α od 0 aż do 1 (wartość optymalna według literatury) rośnie dokładność pomiaru, przy zachowaniu zbliżonych czasów wykonania algorytmu (zgodnie z wynikami zawartymi w *tabelach 4,5,6,7,8*. Dla α = 0 nie wykorzystywany jest wpływ doświadczeń poprzednich pokoleń (czyli de facto wcześniejszych iteracji algorytmu), zatem zwiększanie liczby iteracji w tym przypadku nie będzie prowadzić do poprawy wyniku pomiaru. Dla coraz wyższych parametrów α postępujemy ze zmniejszaniem wartości błędu pomiaru, gdyż coraz bardziej zwiększamy wpływ doświadczeń poprzednich pokoleń w kolejnych iteracjach algorytmu.

4) Wyniki zgromadzone w celu w celu porównania wpływu **parametru β** na wynik pomiaru



Celem tego badania było sprawdzenie różnic w wynikach działania algorytmu mrówkowego dla różnych wartości parametru β określającego siłę wyboru zachłannego drogi na podstawie odległości pomiędzy miastami.

Parametry:

- Liczba mrówek m = wielkości danej instancji n
- Parametr α = 1.0
- Zmienny Parametr β
- Parametr ρ = 0.5
- Parametr Q
- Wykorzystano strategię DAS
- Liczba iteracji algorytmu = 10

Tabela 9: Wyniki pomiaru czasu rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego – badanie ze względu na wartość parametru β (β = 2)

Instancja	Czas [s]	Wynik	Wynik Optymalny	Błąd [%]
gr24.tsp	0.09	1333	1272	4.80
gr48.tsp	0.32	5529	5046	9.57
gr96.tsp	1.85	63590	55209	15.18
kro124p.atsp	2.21	4696	36230	23.37
kroB150.tsp	6.11	31475	26130	20.46
kroB200.tsp	14.73	37885	29437	28.70
lin318.tsp	67.57	42029	42029	27.37

Tabela 10: Wyniki pomiaru czasu rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego – badanie ze względu na wartość parametru β (β = 3)

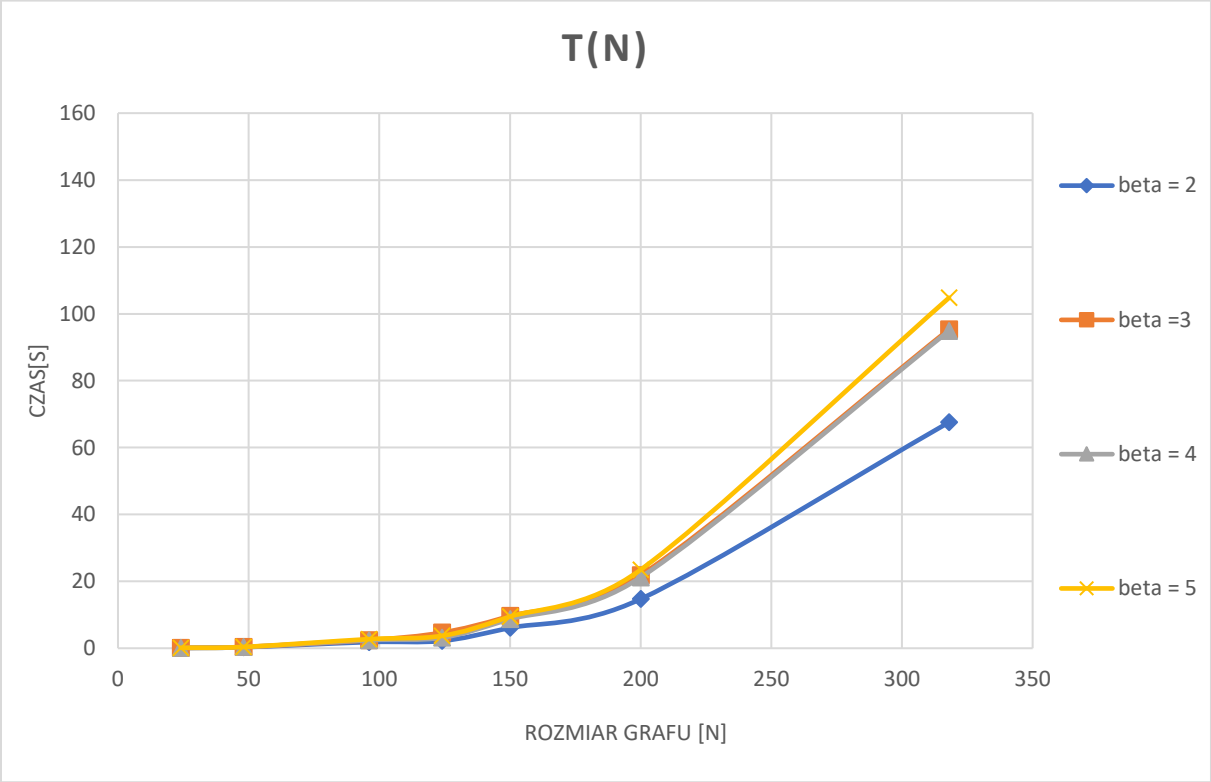
Instancja	Czas [s]	Wynik	Wynik Optymalny	Błąd [%]
gr24.tsp	0.13	1278	1272	0.47
gr48.tsp	0.39	5653	5046	12.03
gr96.tsp	2.43	61618	55209	11.61
kro124p.atsp	4.69	40981	36230	13.11
kroB150.tsp	9.65	30863	26130	18.11
kroB200.tsp	21.85	34590	29437	17.51
lin318.tsp	95.34	50313	42029	19.71

Tabela 11: Wyniki pomiaru czasu rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego – badanie ze względu na wartość parametru β (β = 4)

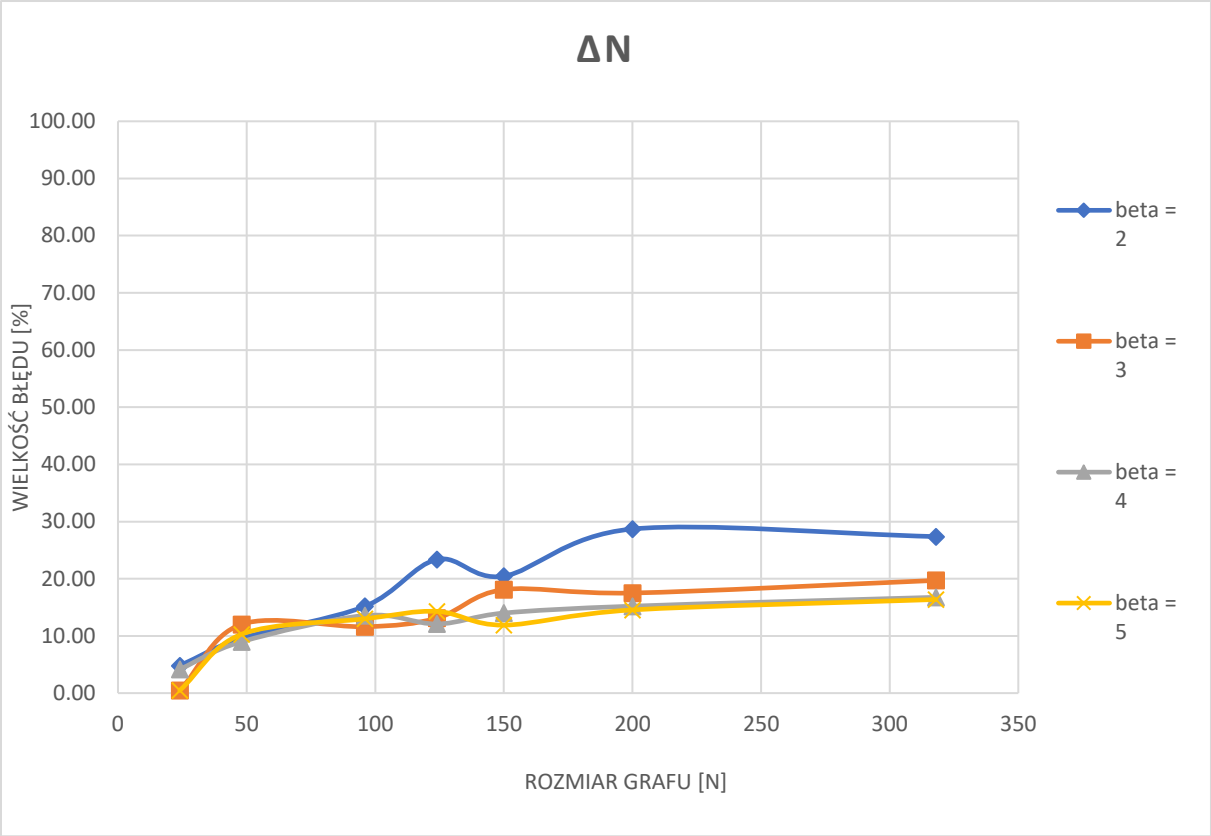
Instancja	Czas [s]	Wynik	Wynik Optymalny	Błąd [%]
gr24.tsp	0.11	1325	1272	4.17
gr48.tsp	0.37	5499	5046	8.98
gr96.tsp	2.42	62705	55209	13.58
kro124p.atsp	3.19	40614	36230	12.10
kroB150.tsp	8.73	29788	26130	14.00
kroB200.tsp	21.20	33914	29437	15.21
lin318.tsp	94.91	49071	42029	16.76

Tabela 12: Wyniki pomiaru czasu rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego – badanie ze względu na wartość parametru β (β = 5)

Instancja	Czas [s]	Wynik	Wynik Optymalny	Błąd [%]
gr24.tsp	0.14	1278	1272	0.47
gr48.tsp	0.39	5565	5046	10.29
gr96.tsp	2.73	62384	55209	13.00
kro124p.atsp	3.73	41409	36230	14.29
kroB150.tsp	9.47	29237	26130	11.89
kroB200.tsp	23.48	33721	29437	14.55
lin318.tsp	104.79	48911	42029	16.37



Rysunek 8: Wpływ wartości parametru β przy rozwiązywaniu problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego na czas rozwiązania problemu



Rysunek 9: Wpływ wartości parametru β przy rozwiązywaniu problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego na błąd pomiaru

Na podstawie *Rysunku 8* można zauważyć, że wraz ze wzrostem wartości para metru β wydłuża się czas rozwiązania problemu, natomiast na podstawie *Rysunku 9* stwierdzamy, że zwiększanie wartości β pozwala na odnalezienie dokładniejszych rozwiązań. Różnice są dobrze widoczne dla instancji o wielkości 200 i 318.

5) Wyniki zgromadzone w celu w celu porównania wpływu **parametru ρ** na wynik pomiaru

wyniki_rho1.csv wyniki_rho0,5.csv wyniki_rho0,1.csv

Celem tego badania było sprawdzenie różnic w wynikach działania algorytmu mrówkowego dla różnych wartości parametru ρ określającego ilość wyparowującego feromonu w jednostce czasu

Parametry:

- Liczba mrówek m = wielkości danej instancji n
- Parametr α = 1.0
- Parametr β = 3.0
- Parametr ρ = 0.5
- Parametr Q
- Wykorzystano strategię DAS
- Liczba iteracji algorytmu = 10

Tabela 13: Wyniki pomiaru czasu rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego – badanie ze względu na wartość parametru ρ (ρ = 0.1)

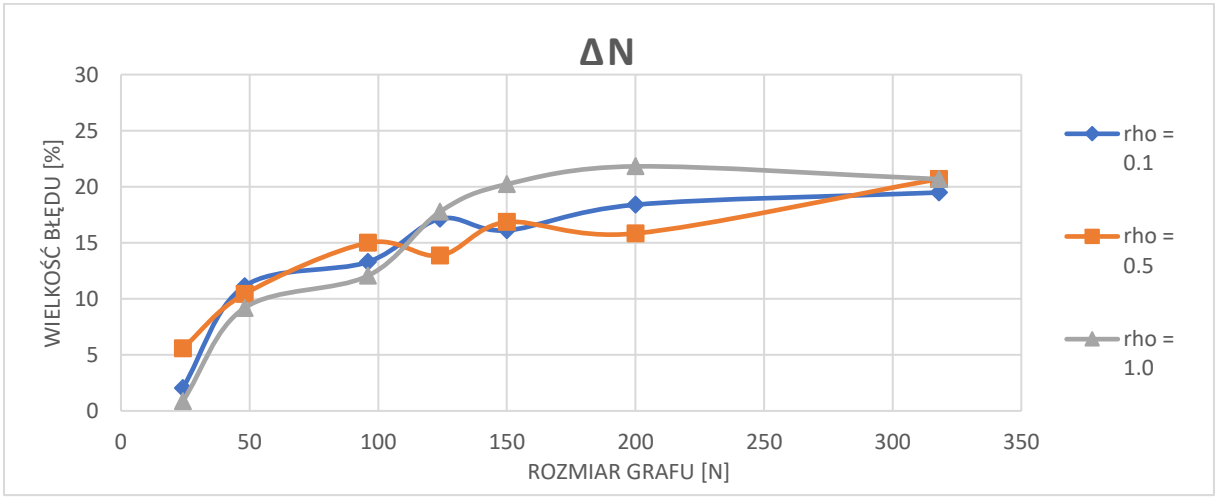
Instancja	Czas [s]	Wynik	Wynik Optymalny	Błąd [%]
gr24.tsp	0.15	1298	1272	2.04
gr48.tsp	0.52	5607	5046	11.12
gr96.tsp	2.58	62544	55209	13.29
kro124p.atsp	3.52	42435	36230	17.13
kroB150.tsp	8.97	30338	26130	16.10
kroB200.tsp	24.37	34849	29437	18.39
lin318.tsp	94.00	50215	42029	19.48

Tabela 14: Wyniki pomiaru czasu rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego – badanie ze względu na wartość parametru ρ (ρ = 0.5)

Instancja	Czas [s]	Wynik	Wynik Optymalny	Błąd [%]
gr24.tsp	0.11	1343	1272	5.58
gr48.tsp	0.39	5572	5046	10.42
gr96.tsp	2.36	63483	55209	14.99
kro124p.atsp	3.52	41252	36230	13.86
kroB150.tsp	9.04	30531	26130	16.84
kroB200.tsp	21.88	34098	29437	15.83
lin318.tsp	101.83	50725	42029	20.69

Tabela 15: Wyniki pomiaru czasu rozwiązania problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego – badanie ze względu na wartość parametru ρ (ρ = 1.0)

Instancja	Czas [s]	Wynik	Wynik Optymalny	Błąd [%]
gr24.tsp	0.11	1283	1272	0.86
gr48.tsp	0.50	5509	5046	9.18
gr96.tsp	3.57	61859	55209	12.05
kro124p.atsp	4.19	42665	36230	17.76
kroB150.tsp	9.23	31414	26130	20.22
kroB200.tsp	23.70	35856	29437	21.81
lin318.tsp	101.85	50716	42029	20.67



Rysunek 10: Wpływ wartości parametru ρ przy rozwiązywaniu problemu Komiwojażera z wykorzystaniem algorytmu mrówkowego na błąd pomiaru

Zgodnie z wynikami pomiarów przedstawionymi w *tabeli 13*, *tabeli 14* oraz *tabeli 15* a także graficznym przedstawieniem rezultatów na *rysunku 10* manipulowanie parametrem ρ nie miało znacznego wpływu na dokładność działania algorytmu oraz szybkość znalezienia rozwiązania.

7. Analiza wyników i wnioski

Najbardziej interesujące nas wyniki badań z perspektywy oceny wydajności algorytmu znajdują się w tabeli 1, natomiast zostały one zwizualizowane na wykresach przedstawionych na rysunku 3 i rysunku 4. Wykorzystanie algorytmu mrówkowego pozwoliło na rozwiązanie problemu dla instancji o maksymalnej wielkości 1002 z błędem 34,5%.

Algorytm mrówkowy pozwolił na zbadanie większych instancji w krótszym czasie niż algorytm naiwny (Brute Force), algorytm Helda-Karpa oraz Symulowanego Wyżarzania.

W opracowaniu wyników zdecydowanie pomogły optymalne wartości parametrów znane z literatury. Dodatkowo zostały przeprowadzone testy, na podstawie których można określić wpływ poszczególnych parametrów na wynik działania algorytmu.

Najbardziej istotnym parametrem z punktu widzenia użytkownika okazał się być współczynnik α, ponieważ odpowiednie jego dobranie może skutkować uzyskaniem nawet kilkukrotnie dokładniejszych wyników pomiaru.

Parametr β miał nieco bardziej ograniczony wpływ na wynik działania algorytmu, jednakże jego odpowiednie dobranie również jest w stanie zoptymalizować działanie programu.

Wybór odpowiedniej strategii rozkładania feromonu na krawędziach również może prowadzić do znacznego poprawienia wyników działania algorytmu. Na podstawie *rysunku 6* określono, że strategia QAS pozwala na osiągnięcie dokładniejszych wyników niż strategia DAS.

Dla sprawdzanych instancji tj. w przedziale od 24 do 318 manipulowanie parametrem ρ w przedziale od 0.1 do 1 nie miało istotnego wpływu na wyniki pomiarów.

Dodatkowo należy zwrócić uwagę na wybranie odpowiedniej liczby iteracji algorytmu w zależności od potrzeb. Jeżeli naszym celem jest uzyskanie wyniku w krótkim czasie, lecz przy zachowaniu dobrego poziomu dokładności warto jest stopniowo zwiększać liczbę iteracji dochodząc do pożądanej precyzji otrzymanego wyniku.

Przeprowadzone badania dowodzą, przy rozwiązywaniu problemu komiwojażera z wykorzystaniem algorytmu mrówkowego odpowiednie dobranie parametrów początkowych odgrywa kluczową rolę w uzyskaniu względnie dokładnego rozwiązania w umiarkowanym czasie.

Projektowanie Efektywnych Algorytmów

Projekt

17/01/2022

259126 Maciej Fras

Dodatek do sprawozdania 4 – Brute Force, algorytm Helda-Karpa, Symulowane wyżarzanie i Algorytm Mrówkowy

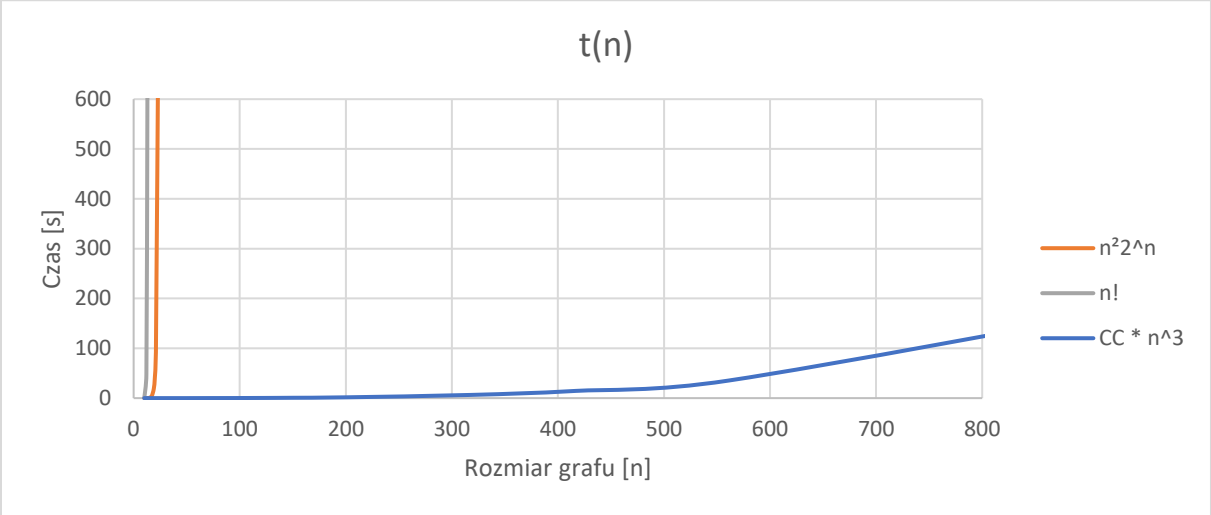
1. Oszacowanie teoretycznej różnicy czasu wykonania algorytmu napisanego z wykorzystaniem algorytmu Helda-karpa oraz metody brute force

Oszacowanie wielokrotności różnicy czasu rozwiązywania problemu dla instancji 13-elementowej:

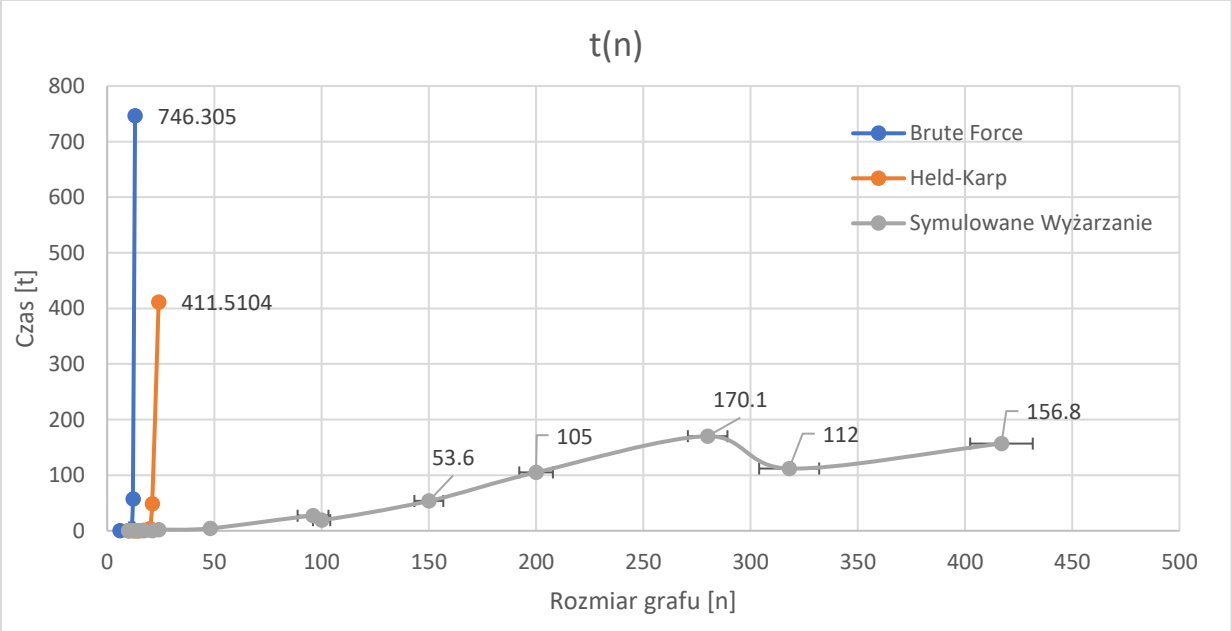
$$\Delta = \frac{n!}{2^n * n^2} = \frac{13!}{2^{13} * 13\text{^}2} = 4497.83653 \approx 4500$$

Zatem dla takiej instancji możemy spodziewać się rozwiązania problemu w czasie krótszym rzędu 4500 razy wykorzystując algorytm Helda Karpa w stosunku do algorytmu wykorzystującego metodę brute force. Jest to oczywiście oszacowanie w pełni teoretyczne, na które wpływ mogą mieć zakłócenia takie jak niedokładne zaimplementowanie algorytmu, czy też specyfika sprzętu, na którym rozwiązujemy problem, niemniej jednak może ono być przydatne w celu dodatkowego porównania obu sprawdzanych metod.

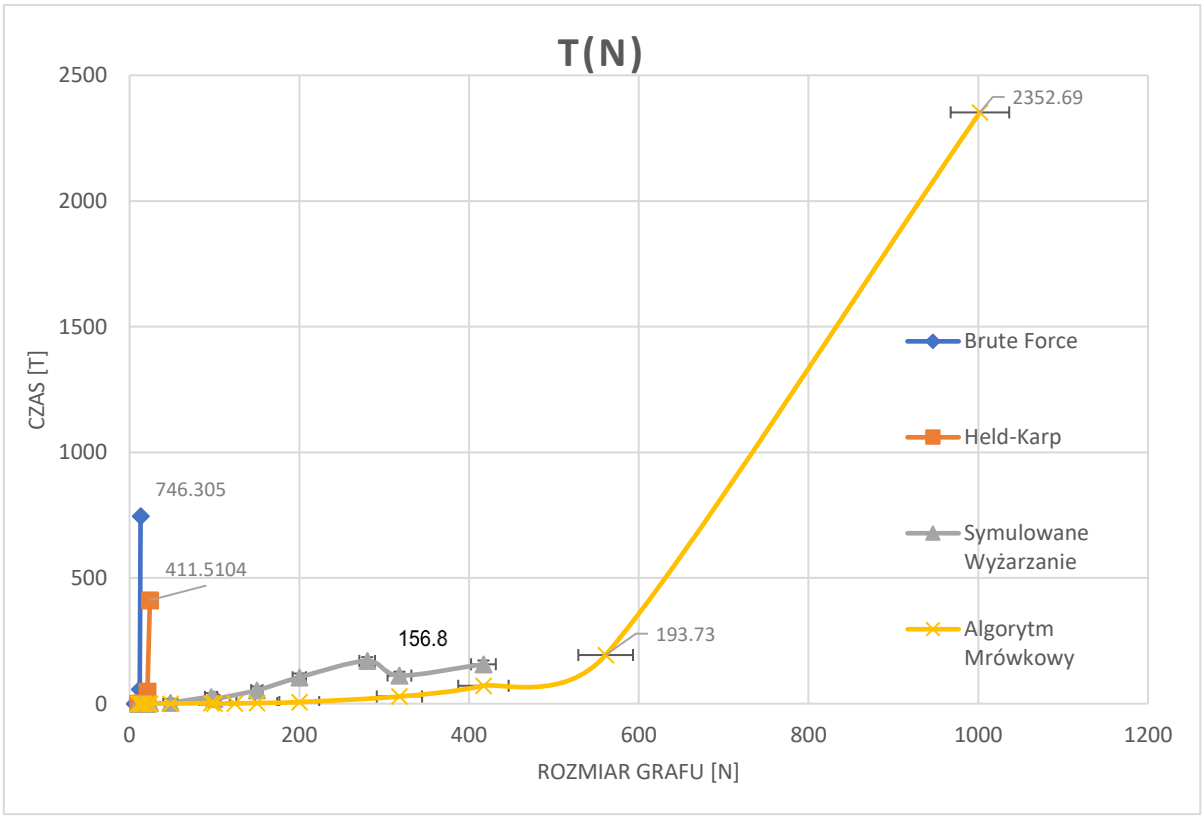
2. Porównanie wyników działania algorytmów



Rysunek 1 - porównanie złożoności obliczeniowych algorytmu Helda Karpa, Brute Force oraz Mrówkowego pomnożonych przez współczynnik równy 10^{-7}



Rysunek 2 - porównanie czasu wykonania algorytmów Brute Force, Helda Karpa i Symulowanego Wyżarzania dla różnych instancji problemu



Rysunek 3 - porównanie czasu wykonania algorytmów Brute Force, Helda Karpa, Symulowanego Wyżarzania, Mrówkowego dla różnych instancji problemu

Brute Force i algorytm Helda-Karpa

Na podstawie wykresów możemy zaobserwować wzrost wydajności algorytmu *Helda-Karpa* względem algorytmu wykorzystującego metodę *Brute Force*. Nagły wzrost czasu rozwiązania problemu w zależności od wielkości instancji w przypadku algorytmu Helda Karpa następuje później o 10 instacji w stosunku do *Brute Force*. Krzywa dla *Helda Karpa* jest łagodniej nachylona niż dla *Brute Force*.

Charakterystyka wykresu na *rysunku 2* przedstawiającym wyniki pomiaru jest zbliżona do wykresu porównującego teoretyczne złożoności obliczeniowe z *rysunku 1*. Ze względu na wyczerpanie dostępnej pamięci RAM w komputerze nie udało mi się wykonać pomyślnego pomiaru dla instancji 29 elementowej, który pozwoliłby jeszcze dokładniej porównać wyniki rzeczywiste z teoretycznymi.

Dla instancji 13 elementowej, która jest największą wspólną badaną instancją dla obu podejść wykonane zostanie porównanie względem teoretycznych złożoności obliczeniowych.

Obliczenie rzeczywistej różnicy wielokrotności czasu wykonania algorytmu dla badanej instancji 13 elementowej:

$$\frac{746s}{0.023s} = 32434.7826 \approx 32435.0$$

W porównaniu do wartości teoretycznej obliczonej w *punkcie 1* uzyskałem wynik od niego odbiegający, lecz zgodny z założeniami, że na wynik wpływ może mieć dokładność wykonania algorytmów, dokładność pomiarów a także sposób dysponowania zasobami przez system operacyjny.

Porównanie różnicy teoretycznej z różnicą rzeczywistą:

$$\frac{32435.0}{4500} = 7.20777 \approx 7.21$$

Różnica, mimo że jest znaczna, to wydaje się akceptowalna ze względu na wszystkie wzięte pod uwagę zakłócenia.

Symulowane Wyżarzanie

Dla algorytmu symulowanego wyżarzania ze względu na liczne parametry (temperatura początkowa, współczynnik schładzania, sposób wyboru rozwiązania w sąsiedztwie, liczba wykonywanych iteracji) oraz duże znaczenie czynnika losowego w przebiegu wykonywania algorytmu (ocena szansy na wybór nowego, gorszego rozwiązania) trudnym zadaniem jest określenie dokładnej złożoności obliczeniowej. Z tego powodu przyjrzymy się wynikom działania algorytmu w sposób poglądowy.

Symulowane wyżarzanie pozwala na rozwiązanie problemu komiwojażera dla instancji nawet kilkunastokrotnie większych niż w przypadku metody naiwnej oraz algorytmu Helda-Karpa. Naszym ograniczeniem nie jest już długi czas wykonania (podejście naiwne) a także złożoność pamięciowa algorytmu (Held-Karp). Wadą algorytmu opartego o metodę symulowanego wyżarzania jest niedokładność wyników pomiarów, którą można minimalizować poprzez wydłużenie czasu badania przestrzeni rozwiązań oraz odpowiedni wybór parametrów początkowych.

Dodatkowo, na *rysunku* 2 zostały opisane błędy względne opisujące różnicę w wyniku działania algorytmu w porównaniu do najlepszego znanego rozwiązania problemu.

Dokładne wyniki pomiarów oraz wartości błędów zwizualizowane na *rysunku 2* zostały zgromadzone w *tabeli* 1 głównej części sprawozdania: *4 (Symulowane Wyżarzanie).*

Sposób obliczenia błędu względnego pomiaru:

$$\Delta n = \frac{res * 100}{opt}$$

Gdzie:

res - koszt obliczony z wykorzystaniem algorytmu opartego o metodę symulowanego wyżarzania
opt – najlepsze znane rozwiązanie problemu (rozwiązanie o najmniejszym koszcie)

Algorytm Mrówkowy

Dla algorytmu mrówkowego, podobnie jak w przypadku algorytmu symulowanego wyżarzania występuje wiele parametrów mających zasadniczy wpływ na wynik jego pracy, takich jak: α – określający wpływ doświadczeń poprzednich pokoleń, β – określający siłę wyboru zachłannego drogi na podstawie odległości pomiędzy miastami, ρ – określający ilość wyparowującego feromonu w jednostce czasu, m – liczba mrówek. Poza tym występuje czynnik losowości, z którym mamy do czynienia podczas wyboru następnej odwiedzanej krawędzi przez mrówkę. Fakty te są powodem niedokładności wyników działania algorytmu dla dużych instancji problemu, tj. większych niż 25 miast.

Z *rysunku 3* łatwo wywnioskować, że algorytm mrówkowy pozwolił na rozwiązanie problemów o wielkości instancji w przedziale 100 – 400 w czasie krótszym niż dla algorytmu wykorzystującego metodę symulowanego wyżarzania, lecz odbywa się to kosztem dokładności algorytmu. Możliwym rozwiązaniem poprawienia dokładności działania algorytmu mrówkowego jest zwiększenie ilości iteracji programu, jednakże w ten sposób znacznie wydłuża się czas działania. Z tego wynika, że w celu uzyskania optymalnych wyników dla tych algorytmów należy poświęcić dużo uwagi na dobranie odpowiednich wartości parametrów sterujących przebiegiem wykonania algorytmu.

Dodatkowo, dzięki zastosowaniu algorytmu mrówkowego udało się zbadać instancje 1002 elementową przy błędzie na poziomie 34,5%. Jest to rozwiązanie problemu dla największej instancji większej o ok. 2.5 razy względem algorytmu wykorzystującego metodę symulowanego wyżarzania. Udało się przy tym jedynie nieznacznie przekroczyć błąd o wartości 30%, którego nie przekraczano dla pomiarów działania algorytmu symulowanego wyżarzania.

W porównaniu do algorytmów przeszukujących przestrzeń rozwiązań: brute force oraz Held-Karp zyskujemy możliwość rozwiązania problemów nawet kilkudziesięciokrotnie większych, mając wciąż na uwadze, że algorytmy te mają przewagę w postaci pewności odnalezienia rozwiązania dokładnego

Podsumowanie

W miejscu, gdzie zależy nam na dokładności powinniśmy sięgnąć po algorytmy: naiwny oraz Helda-Karpa, natomiast gdy najistotniejszym kryterium jest krótki czas rozwiązania problemu możemy zdecydować się na algorytm symulowanego wyżarzania lub mrówkowy (przy założeniu, że mamy czas na dostrojenie algorytmu poprzez dobranie takich parametrów, które zagwarantują nam wystarczająco dokładne wyniki przy krótkim czasie działania). Kluczowym wnioskiem z wykonanego projektu jest fakt, że nie istnieje algorytm idealny w celu rozwiązania danego problemu. Najważniejszym powinno być dobranie odpowiedniego algorytmu, metody czy też heurystyki w zależności od indywidualnych potrzeb. Dla problemu komiwojażera są to wielkość instancji, rodzaj problemu do rozwiązania (drzewo w formie *tsp* lub *atsp*) czy też wymagania względem złożoności czasowej lub pamięciowej oraz dokładności pożądanych wyników.