Projektowanie Efektywnych Algorytmów

Projekt

25/01/2023

numer indeksu	imię i nazwisko
259190	Jędrzej Czykier

(4) Algorytm mrówkowy

Spis treści strona

1. Sformułowanie zadania

Zadanie polega na opracowaniu, implementacji i zbadaniu efektywności algorytmów rozwiązujących problem komiwojażera, nazywanego również TSP *(eng. Traveling Salesman Problem)* w wersji optymalizującej. Problem ten, polega na znalezieniu minimalnego cyklu Hamiltona czyli znając zbiór wierzchołków grafu oraz wagi połączeń między nimi, należy znaleźć drogę w której każdy wierzchołek jest odwiedzany tylko raz, z wyjątkiem pierwszego, do którego algorytm powinien wrócić na końcu działania.

Problem ten jest NP-trudny, co oznacza że jego rozwiązanie jest co najmniej tak trudne, jak rozwiązanie innych problemów należących do klasy NP. Przykładami innych problemów należących do tej klasy są między innymi: problem plecakowy i problem zbioru niezależnego.

Jednym ze sposobów rozwiązania TSP jest metoda Brute-force. Algorytm ten sprawdza po kolei wszystkie możliwości, co czyni go relatywnie prostym do implementacji, lecz powolnym w działaniu. Kolejnym jest algorytm Helda-Karpa, nazywany też algorytmem Bellmana-Helda-Karpa. Wykorzystuje on programowanie dynamiczne aby rozwiązać problem komiwojażera. Oba te rozwiązania są algorytmami dokładnymi, co oznacza że zawsze znajdą optymalne rozwiązanie.

Do rozwiązywania problemów NP-trudnych często stosuje się algorytmy heurestyczne, czyli takie, które nie zapewniają optymalnych rozwiązań. Ze względu na wysoką złożoność obliczeniową algorytmów dokładnych, czasami godzi się na rozwiązanie wystarczająco dobre, które można otrzymać w krótszym czasie. Do takich algorytmów zalicza się symulowane wyżarzanie, które można wykorzystać do rozwiązania problemu komiwojażera.

Z tego samego powodu używa się również algorytmów metaheurystycznych, które również nie gwarantują optymalnego rozwiązania, oraz zwykle nie da się określić czasu ich działania. Skuteczność takich metod często zależy od dobranych parametrów. Przykładem takiego algorytmu jest algorytm mrówkowy.

2. Metoda

Algorytm mrówkowy jest algorytmem metaheurystycznym stosowanym do rozwiązywania wielu różnych problemów, z których jednym jest TSP. W przypadku problemu komiwojażera, podstawowa idea algorytmu, polega na zmodelowanie grafu jako kolonii mrówek, w której każdy z wierzchołków jest kopcem. Mrówki poruszają się z kopca do kopca, wybierając swoją destynacje z prawdopodobieństwem zależnym od atrakcyjności danego kopca oraz siły feromonowego śladu pozostawionego przez inne mrówki. Wartość atrakcyjności kopca może oznaczać np. odległość.

Działanie algorytmu polega na wylosowaniu wierzchołka startowego dla każdej mrówki, po czym iteracyjnie, każda mrówka przechodzi do kolejnego wierzchołka wybieranego na podstawie lokalnej ilości feromonu i pewnej heurystki. Następnie ilość feromonu jest aktualizowana według pewnych zasad. Na początku wykonywania się algorytmu, ilość feromonu na każdej krawędzi powinna być stosunkowo mała i wszędzie jednakowa.

Algorytm wykonuje się określoną ilość iteracji albo do czasu, kiedy wszystkie mrówki wybiorą tą samą ścieżkę.

Stężenie feromonu na krawędziach, aktualizowane jest w każdym kroku według wzoru:

$$\pi_{ij}(t+1) = \rho \cdot \tau_{ij}(t) + \Delta\tau_{ij}(t, t+1)$$

Gdzie $\tau_{ij}(t)$ – zawartość feromonu na krawędzi *E(i,j)* w chwili *t,* ρ – współczynnik z przedziału *<0,1>* określający ilość wyparowanego feromonu w jednostce czasu.

$$\Delta\tau_{ij}(t, t+1) = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}\,(t, t+1)$$

Gdzie $\Delta\tau_{ij}^{k}$ *(t,t+1)* jest ilością feromonu odkładaną na jednostkę długości krawędzi *(i,j)* rozkładanego przez *k*-tą mrówkę

Aby uniknąć odwiedzania tego samego wierzchołka więcej niż raz w jednej ścieżce, każda mrówka ma swoją listę tabu, w której zapisywane są odwiedzane przez nią wcześniej wierzchołki. Lista jest czyszczona po dopełnieniu cyklu.

Prawdopodobieństwo wyboru miasta *j* przez mrówkę znajdującą się w mieście *i* obliczane jest ze wzoru:

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

gdzie:

c – kolejne możliwe miasto (nie znajdujące się na liście tabu danej mrówki)

Ω – zbiór nieodwiedzonych przez daną mrówkę miast

η_{ij} – wartość lokalnej funkcji kryterium

α – parametr regulujący wpływ na τ_{ij} (wpływ doświadczeń poprzednich pokoleń)

β - parametr regulujący wpływ η_{ij} (określa siłę wyboru zachłannego drogi na podstawie odległości między wierzchołkami)

Wyróżnia się trzy schematy rozkładu feromonu (sposobu aktualizacji wartości jego ilości). Rozróżnia je sposób wyliczania $\Delta\tau_{ij}^{k}$ *(t,t+1)* oraz moment uaktualniania τ_{ij}.

- Algorytm Cykliczny (CAS) - stała rozkładanej ilości feromonu jest dzielona przez długość trasy L znalezioną przez daną mrówkę, wartość feromonu aktualizowana jest po *n* krokach.

- Algorytm Gęstościowy (DAS) – rozkładana ilość feromonu jest stała dla wszystkich krawędzi, wartość feromonu jest aktualizowana po każdym kroku.

- Algorytm Ilościowy (QAS) – stała rozkładanej ilości feromonu jest dzielona przez długość krawędzi d_{ij}, wartość feromonu jest aktualizowana po każdym kroku.

Twórca algorytmu zaleca używanie przyjęcie następujących wartości parametrów:

- α = 1

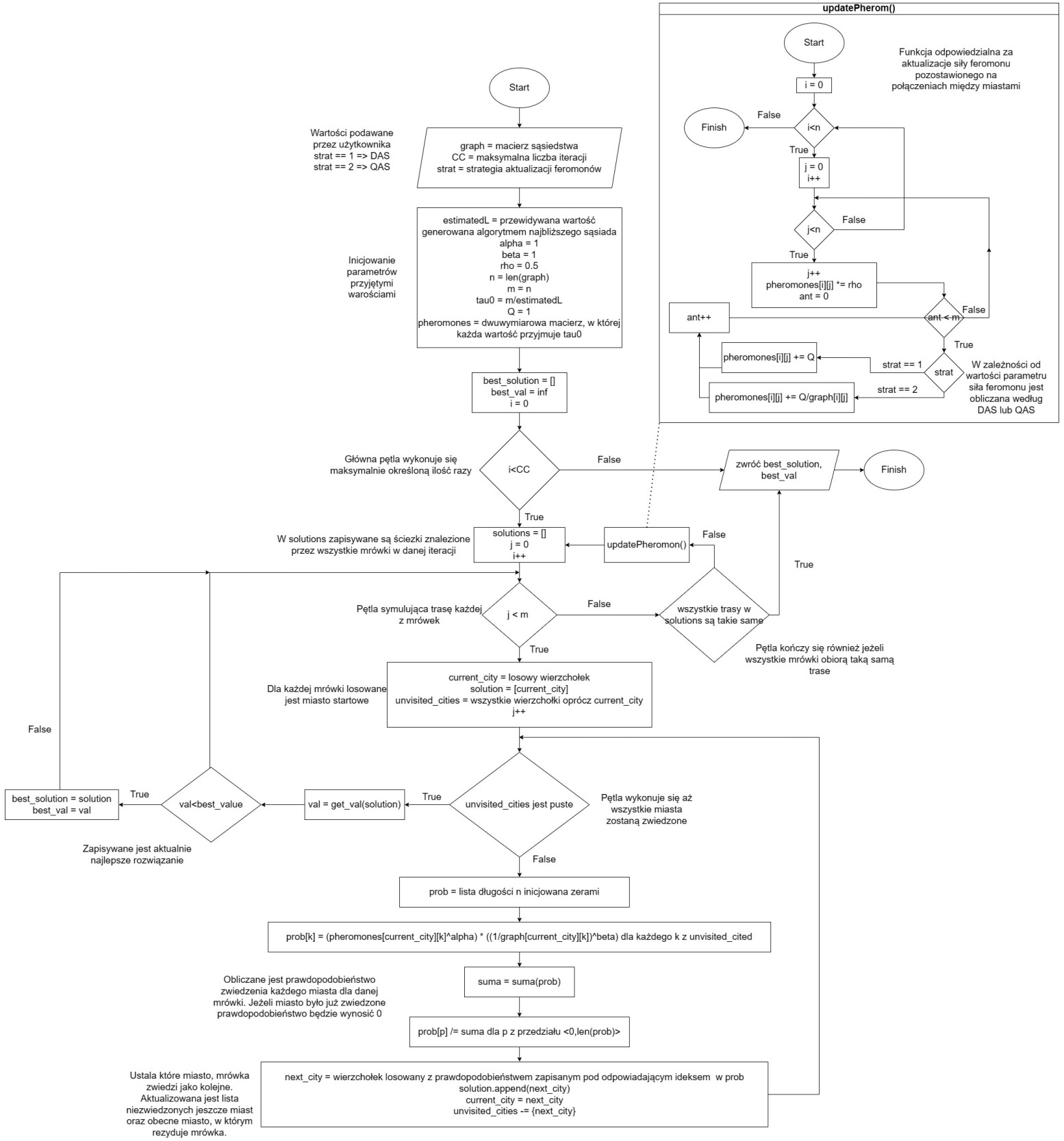
- β od 2 do 5

- ρ = 0.5

- m = n (m - liczba mrówek, n - liczba miast)

- τ0 = $\frac{m}{C^{nn}}$ gdzie C^{nn} jest szacowaną długością trasy

3. Opis algorytmu



Rysunek 1 Schemat blokowy algorytmu

4. Dane testowe

Dane wykorzystane do sprawdzenia poprawności algorytmu oraz poprawne wyniki:

Nazwa pliku, optymalna ścieżka, waga optymalnej ścieżki

1. tsp_6_1.txt, [0, 1, 2, 3, 4, 5], 132,

2. tsp_6_2.txt , [0, 5, 1, 2, 3, 4], 80,

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

Powyższe pliki można znaleźć po adresem:

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

Dane wykorzystane do badań:

Nazwa pliku, waga najlepszej znanej ścieżki

1. gr17.tsp, 2085

2. gr24.tsp, 1272

3. ftv38.atsp, 1530

4. ftv47.astp, 1776

5. ft53.atsp, 6905

6. ftv64.atsp, 1839

7. gr96.tsp, 55209

8. pr124.tsp, 59030

9. ch150.tsp 6528

Powyższe pliki można znaleźć pod adresami:

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

http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/

Na potrzebę testów, pliki zostały edytowane aby dane w nich zawarte przyjęły postać macierzy sąsiedztwa.

5. Procedura badawcza

Należało zbadać czas wykonywania się algorytmu i wartość błędy w zależności od rodzaju użytego schematu rozkładu feromonu oraz udowodnić prawdziwość tezy twierdzącej, że parametr α określa wpływ doświadczeń poprzednich pokoleń oraz że parametr β określa siłę zachłannego wyboru drogi na podstawie odległości między miastami.

Maksymalna liczba iteracji dla każdej instancji była taka sama i wynosiła 1000. Algorytm zbadano dla dwóch schematów rozkładu feromonu: DAS oraz QAS. Do badań użyto następujących parametrów:

$$\alpha = 1$$

$$\beta = 4$$

$$\rho = 0.5$$

m = liczba wierzchołków grafu

$\tau 0$ = m/przewidywana droga generowana algorytmem NN

$$Q = 1$$

Treść pliku .ini:

#file times optimal CC start

[section_a]

file1 = gr17.tsp 10 2085 1000 1

file2 = gr24.tsp 10 1272 1000 1

file3 = ftv38.atsp 10 1530 1000 1

file4 = ftv47.atsp 5 1776 1000 1

file5 = ft53.atsp 5 6905 1000 1

file6 = ftv64.atsp 5 1839 1000 1

file7 = gr96.tsp 2 55209 1000 1

file8 = pr124.tsp 1 59030 1000 1

file6 = ch150.tsp 1 6528 1000 1

[section_b]

outputFile = outputACO.csv

Każda z instancji przetestowana została, podaną obok nazwy pliku ilość razy, np. gr17.tsp wykonana została 5 razy. Kolejne liczby oznaczają po kolei: wartość najlepszej znanej ścieżki, maksymalną liczbę iteracji oraz przyjęty schemat rozkładu feromonu: 1 – DAS, 2 – QAS. Do pliku wyjściowego outputACO.csv zapisywany był czas wykonania się algorytmu, otrzymane rozwiązanie (koszt ścieżki), ścieżka (numery kolejnych węzłów, otrzymany błąd względny oraz zastosowane w algorytmie parametry. Plik wyjściowy zapisywany był w formacie csv.

Błąd obliczany był ze wzoru:

$$err = \frac{wartość\ otrzymana\ -\ wartość\ optymalna}{wartość\ optymalna} * 100\%$$

Czas wykonywania się algorytmu był mierzony za pomocą funkcji z biblioteki *time*. Biblioteka jest wbudowana w język Python.

Poniżej przedstawiono fragment pliku wyjściowego.

gr17.tsp [10, 2, 14, 13, 5, 7, 6, 16, 12, 3, 0, 15, 11, 8, 4, 1, 9] 2096 1000 1 0,527577938 6,786848783

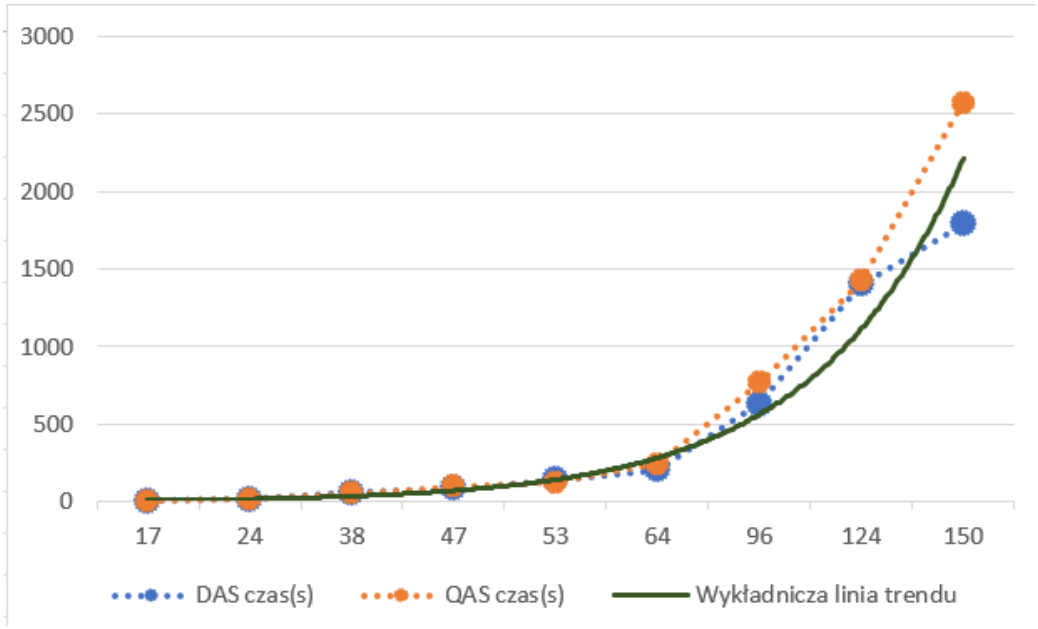
gr17.tsp [9, 10, 2, 14, 13, 16, 5, 7, 6, 12, 3, 0, 15, 11, 8, 4, 1] 2085 1000 1 0 7,302490473

…

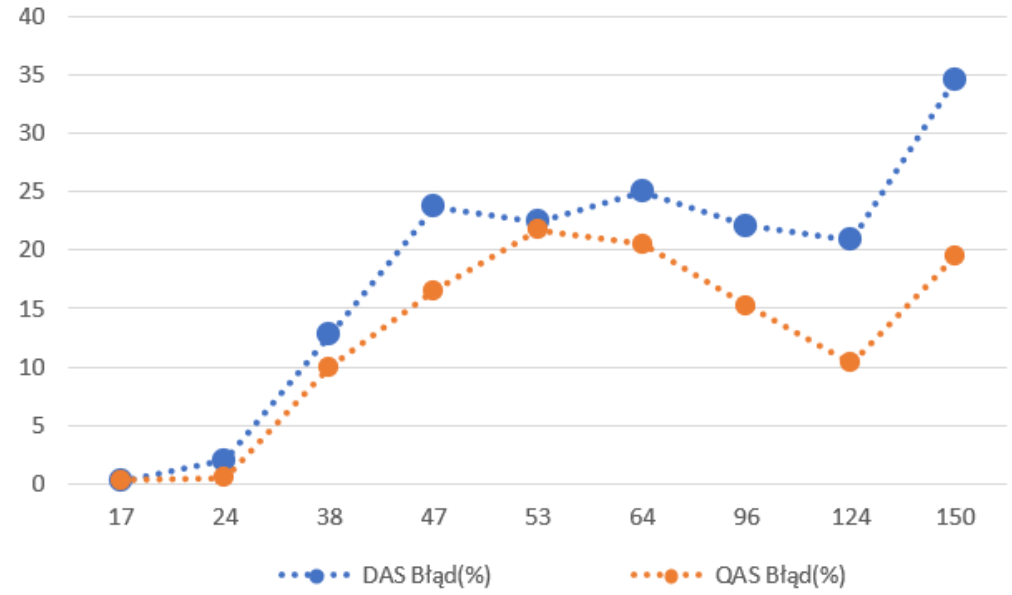
ch150.tsp [0, 97, 102, 86, 75, 72, 47, 62, 29, 83, 7, 88, 95, 9, 112, 93, 87, 120, 78, 58, 15, 110, 104, 51, 92, 34, 123, 125, 32, 53, 91, 45, 89, 1, 5, 8, 27, 41, 36, 98, 113, 101, 136, 49, 134, 69, 107, 85, 28, 57, 54, 64, 131, 84, 141, 17, 74, 25, 145, 55, 82, 140, 19, 24, 80, 109, 119, 46, 138, 39, 52, 11, 23, 117, 126, 68, 60, 35, 10, 147, 129, 16, 65, 59, 139, 116, 38, 56, 40, 100, 37, 22, 31, 130, 66, 42, 108, 50, 133, 137, 132, 76, 121, 13, 79, 14, 77, 43, 70, 44, 127, 12, 63, 111, 135, 144, 143, 146, 48, 71, 20, 149, 114, 3, 103, 118, 67, 90, 105, 73, 122, 30, 26, 128, 115, 18, 96, 99, 142, 4, 106, 94, 81, 21, 124, 148, 61, 2, 6, 33] 8781 1000 1 34,51286765 1789,331542

6. Wyniki

Wyniki zgromadzone w pliku acoPomiary.xlsx. Wszystkie pliki wynikowe zostały przesłane na Eportal.



Rysunek 2 Zależność czasu wykonywania się algorytmu dla wielkości instancji



Rysunek 3 Zależność otrzymanego błędu od wielkości instancji

7. Analiza wyników i wnioski

Z wykresu [Rysunek 1] można odczytać, że dobór schematu rozkładu feromonu nie ma większego wpływu na czas wykonywania się algorytmu. Dla niektórych, większych instancji, algorytm wykonywał się dłużej, gdy użyto schematu QAS, może być to jednak kwestia przypadku, ponieważ badania dla dużych instancji nie były powtarzane wiele razy, ze względu na to ile czasu zajmowało wykonanie się algorytmu.

Na czas działania algorytmu, z pewnością wpłynęła wielkość instancji. Linia trendu na wykresie zależności czasu od ilości wierzchołków [Rysunek 1] ma charakter wykładniczy.

Na wykresie [Rysunek 2] widać, że dla większych instancji, otrzymany błąd był większy, jednakże nie jest to wzrost ciągły, szczególnie gdy zastosowano schemat QAS. Może być to jednak kwestia przypadku lub specyfika danej instancji. Można za to na pewno stwierdzić, że użycie schematu QAS skutkowało uzyskaniem mniejszego błędu niż używanie schematu CAS.