Politechnika Wrocławska

Programowanie efektywnych algorytmów

Problem komiwojażera (TSP)

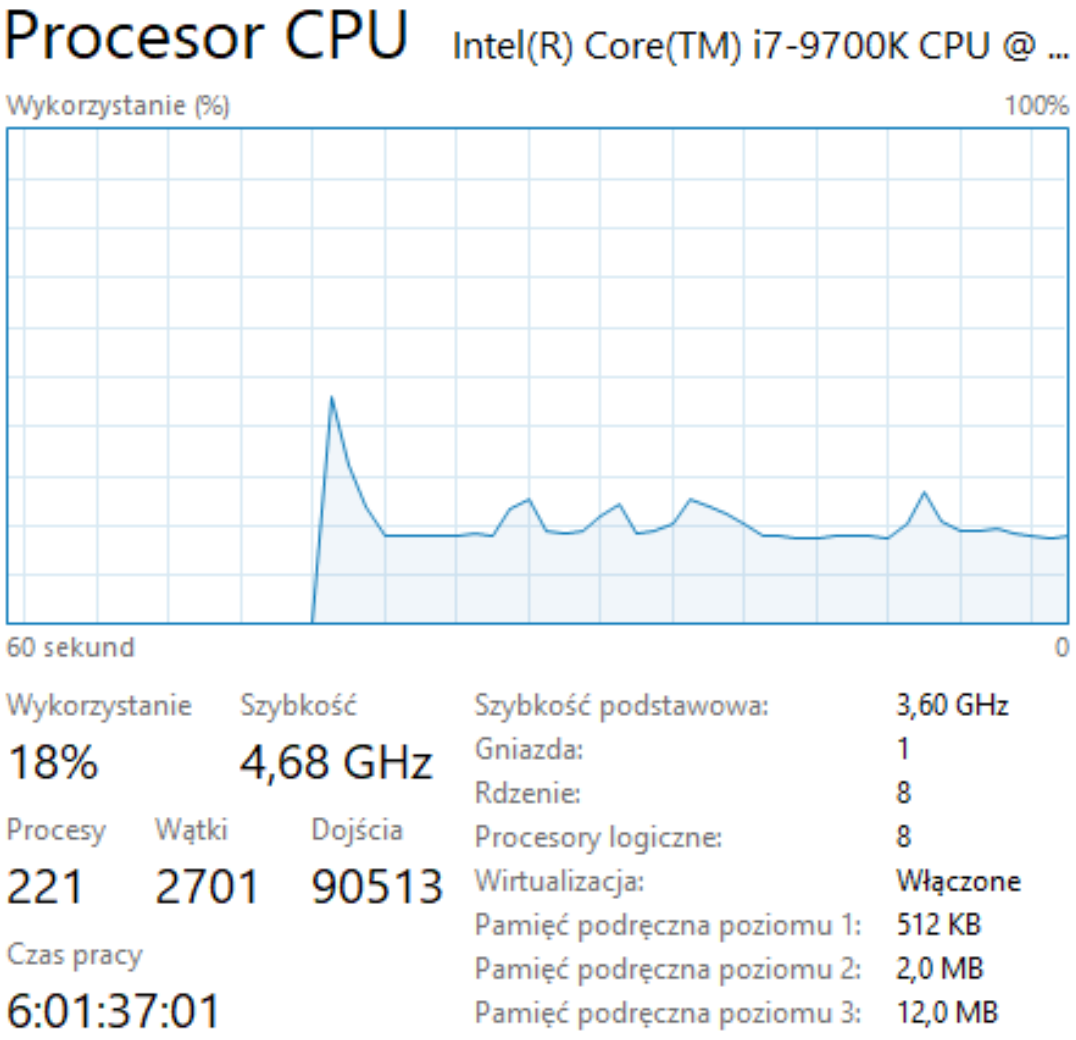
Krzysztof Zalewa

25.1.2025

Spis treści

1 Specyfikacja sprzętu użytego do badań

Badania zostały wykonanie na komputerze stacjonarnym o specyfikacji:
Procesor: Intel(R) Core(TM) i7-9700K CPU
Zegar: 3.60GHz
Wielkość pamięci RAM: 32,0 GB (dostępne: 31,8 GB)



Rysunek 1: Zurzycie procesora w trakcie wykonywania algorytmu Tabu search

2 Instancje użyte w badaniach

Do badań użyłem macierzy z TspLib[4.]
Macierze symetryczne - rat99.tsp,pr152.tsp,ts225.tsp oraz pr264.tsp
Macierze asymetryczne - ftv33.atsp,ftv64.atsp,kro124p.atsp* oraz ftv170.atsp
Gdzie najleprze znane rozwiązania to :

	Symetryczne				Asymetryczne			
Instancje	rat99	pr152	ts225	pr264	ftv33	ftv64	kro124p*	ftv170
	1211	73682	126643	49135	1286	1839	36230	2755

Table 1: Optymalne wyniki z TspLib

* Mimo tego że powinna być to instancja o 124 wierzchołkach po konwersji otrzymuję tylko 100 wierzchołków. Pozostałe instancje konwertują się poprawnie.

3 Tabu search

3.1 Opis Algorytmu

Algorytm tabu search jest algorytmem metahurystycznym służącym do rozwiązywania problemów optymalizacyjnych. Algorytm ten przechowuje część otrzymanych wcześniej rozwiązań w tablicy tabu. Takie podejście sprawia że szansa na ponowne wybranie tego samego rozwiązania znacznie maleje. Sprawia to też że jest mniejsza szansa na utknięcie w pętli. Algorytm ten można jeszcze usprawnić poprzez dodanie warunku krytycznego i strategii dywersyfikacji.
Warunek krytyczny: Jeżeli otrzymana wartość jest mniejsza od najlepszej do tej pory znalezionej wartości ale ścieżka znajduje się w tabu to i tak przypisujemy obecną wartość do najlepszej.
Strategia dywersyfikacji: Jeżeli przez określoną liczbę iteracji algorytmu wartość się nie poprawiła zmieniamy wybraną ścieżkę. W mojej implementacji zmiana ścieżki polega na wylosowaniu nowej przy użyciu funkcji std::shuffle.

3.2 Badanie wpływu metody przeszukiwania sąsiedztwa

W moim algorytmie zaimplementowałem dwie różne metody przeszukiwania sąsiedztwa.

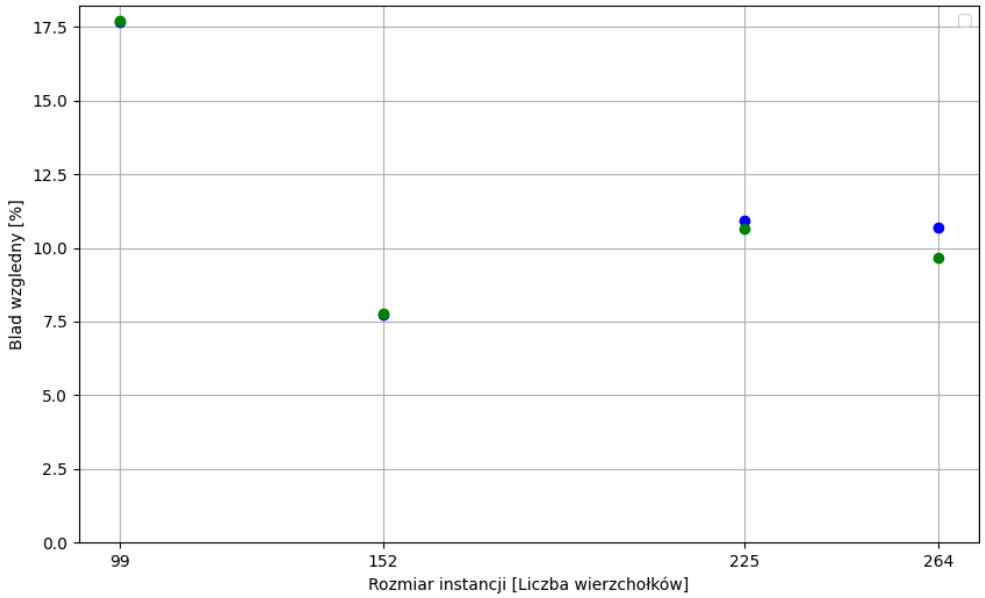
1. Swap - Wybieram dwa wierzchołki i zamienia je ze sobą. W mojej implementacji sprawdzam wszystkie unikalne możliwości więc generuję n*(n-1)/n sąsiadów.

2. Insert - wybieram wierzchołek i miejsce w tablicy. Wierzchołek kopiuje a następnie usuwam z tablicy. Na koniec kopię wierzchołka wstawiam w wybrane miejsce. W mojej implementacji obie wartości losuję. Wykonuję n*(n-1)/n losowań.

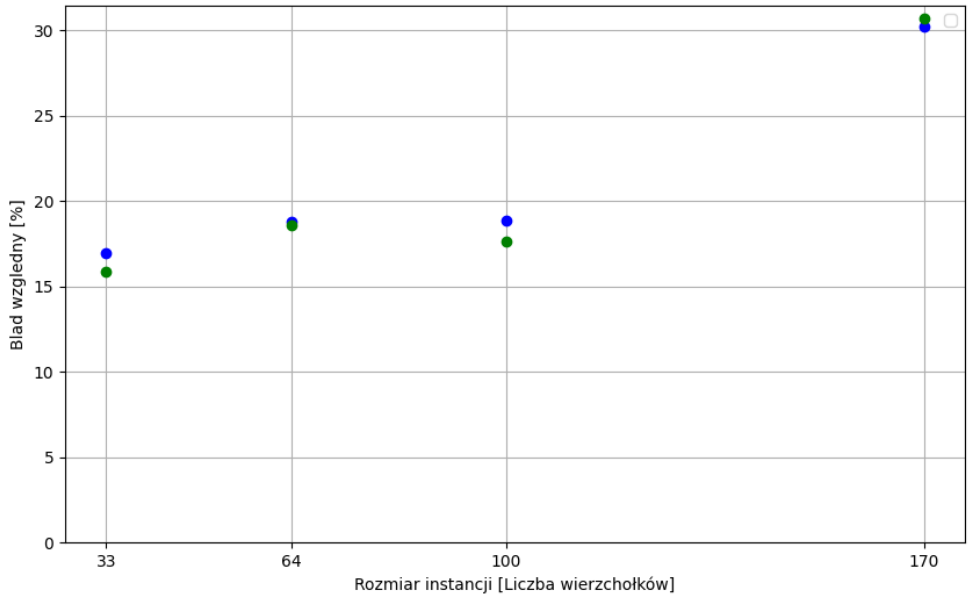
Hipoteza: W tabu search metoda przeszukiwania insert powinna być znacznie lepsza niż swap.
Badania: Wykonuję badania dla każdej z 8 instancji. Żeby otrzymać dobrą próbkę badanie powtarzam cztero krotnie dla każdej instancji. Więc wykonuje (4*ASYM + 4*ASM)*4 * 2 = 64 badań.

	Symetryczne				Asymetryczne			
Rozmiar[Liczba wierzchołków]	99	152	225	264	33	64	100	170
Swap	17.67	7.75	10.93	10.71	16.95	18.76	18.86	30.20
Insert	17.73	7.79	10.66	9.66	15.88	18.61	17.62	30.71

Table 2: Błędy w wynikach algorytmu dla macierzy symetrycznych i niesymetrycznych



Rysunek 2: Wyniki badań dla macierzy symetrycznych[%]



Rysunek 3: Wyniki badań dla macierzy asymetrycznych[%]

* Niebieskie punkty to wyniki dla Swap a zielone dla Insert **Wnioski:** Wbrew założeniom wyniki są bardzo zbliżone. (Tabela 2) Różnica nie przekracza 2 punktów procentowych.

3.3 Badanie wpływu metody tworzenia pierwszego rozwiązania

W moim algorytmie zaimplementowałem dwie różne metody tworzenia pierwszego rozwiązania.

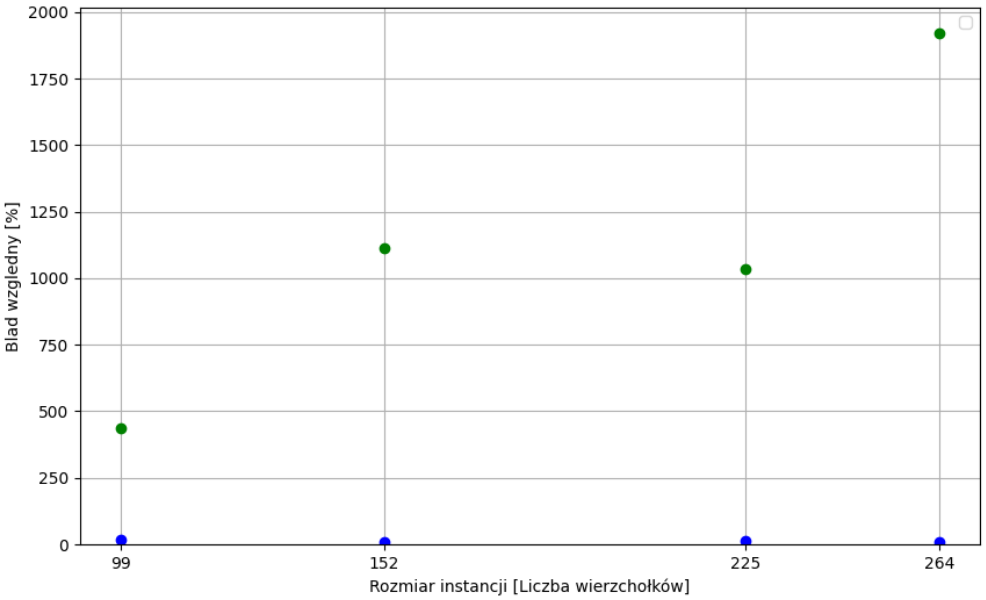
1. Losowa - Pierwsze rozwiązanie jest zupełnie losowe. Najpierw tworzę wektor z liczbami od 0 do n - 1. Następnie na tym wektorze używam funkcji shuffle.

2. Nearest neighbour - Korzystam z wcześniej zaimplementowanego algorytmu NN.

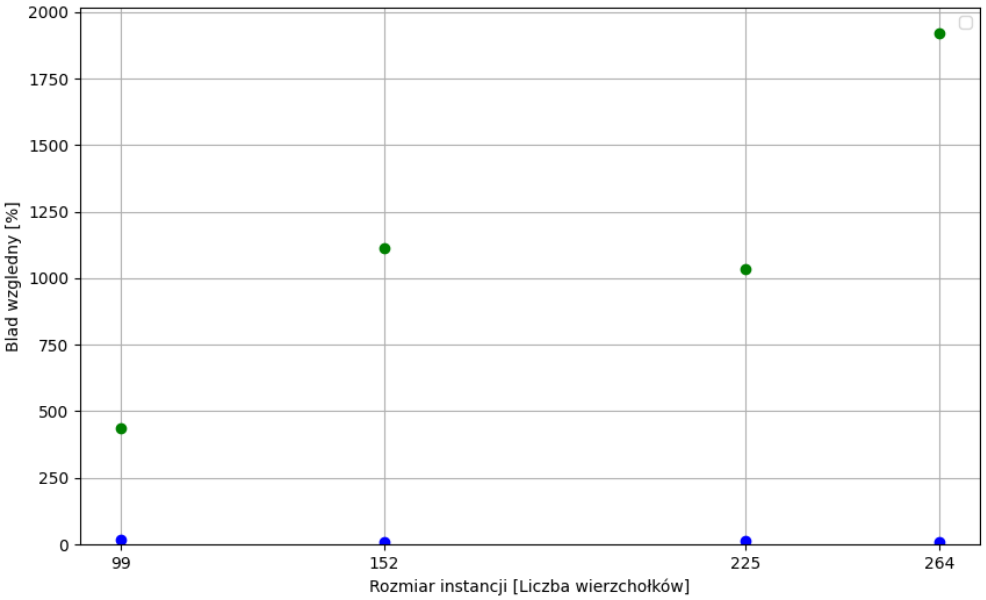
Hipoteza: Wyniki dla metody losowej będą znacznie gorsze (Większy błąd wzgledny) od NN. Ale jest też niewielka szansa na wyniki będące bliżej optymalnego rozwiązania. Jednakże wykonanie wielu pomiarów powinno wykluczyć znaczne odstępstwa.
Badania: Podobnie jak w poprzednim przypadku wykonuje badania dla każdej z 8 instancji po 4 powtórzenia. Więc znowu wykonuje (4*ASYM + 4*ASM)*4 * 2 = 64

	Symetryczne				Asymetryczne			
Rozmiar[Liczba wierzchołków]	99	152	225	264	33	64	100	170
NN	17.67	7.75	10.93	10.71	16.95	18.76	18.86	30.20
Random	436.23	1113.25	1034.01	1921.32	82.41	256.28	318.51	730.16

Table 3: Błędy w wynikach algorytmu dla macierzy symetrycznych i niesymetrycznych



Rysunek 4: Wyniki badań dla macierzy symetrycznych



Rysunek 5: Wyniki badań dla macierzy asymetrycznych

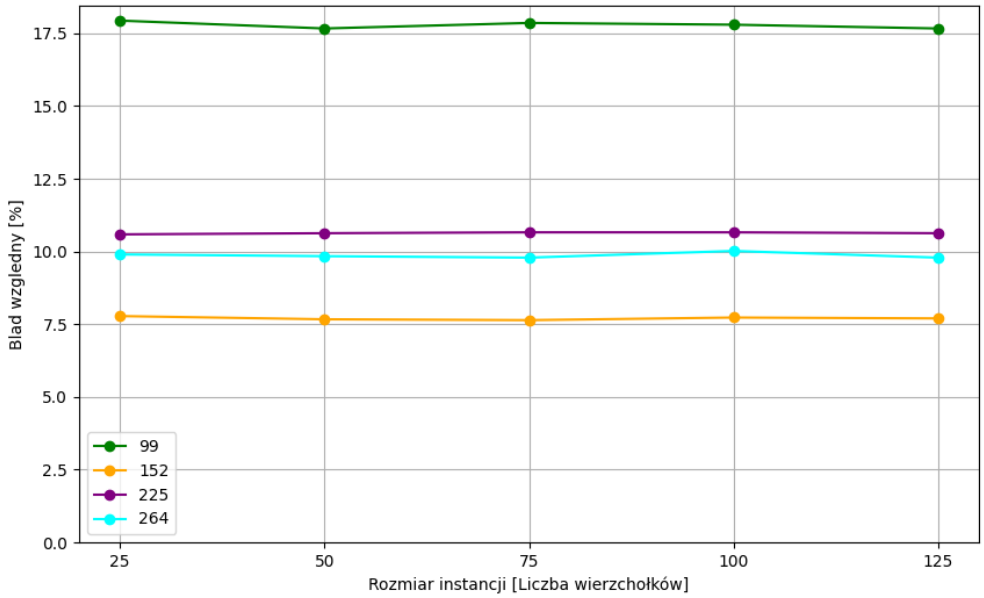
* Niebieskie punkty to wyniki dla NN a zielone dla random. **Wnioski:** Wyniki zgadzają się z moją hipotezą. Było to do przewidzenia Jako że random zaczyna w zupełnie losowym punkcie jego rozwiązania będą gorsze.

3.4 Badanie wpływu ilości iteracji bez zmian

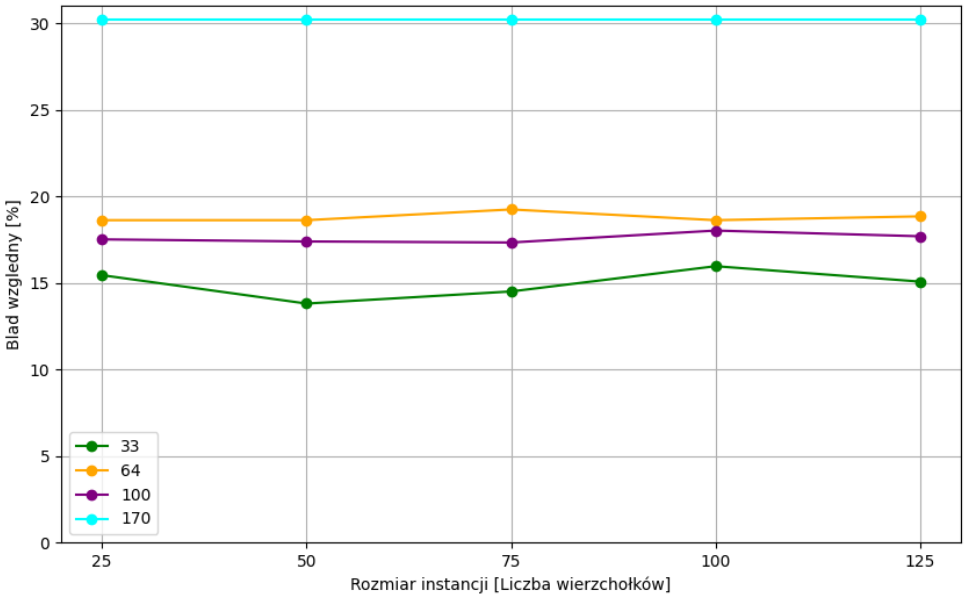
W mojej implementacji przekroczenie pewnej ilości bez zminay wyniku służy do dywersyfikacji. Do testów wybrałem wartości 25,50,75,100,125.
Hipoteza: Wraz ze wzrostem ilości iteracji bez zmian jakość rozwiązania też będzie rosła.
Badania: Dla każdej instancji testuje wszystkie 5 wartości. Żeby otrzymać dobrą próbkę badanie powtarzam cztero krotnie dla każdej instancji. Dla tego wykonuję (4*ASYM + 4*ASM)*5 * 4 = 160 badań.

	Symetryczne				Asymetryczne			
Rozmiar[Liczba wierzchołków]	99	152	225	264	33	64	100	170
25	17.94	7.78	10.59	9.90	15.44	18.62	17.51	30.20
50	17.67	7.67	10.63	9.84	13.80	18.62	17.39	30.20
75	17.86	7.64	10.66	9.79	14.50	19.24	17.33	30.20
100	17.80	7.73	10.66	10.02	15.96	18.62	18.02	30.20
125	17.67	7.70	10.63	9.79	15.07	18.84	17.69	30.20

Table 4: Błędy w wynikach algorytmu dla macierzy symetrycznych i niesymetrycznych



Rysunek 6: Wyniki badań dla macierzy symetrycznych



Rysunek 7: Wyniki badań dla macierzy asymetrycznych

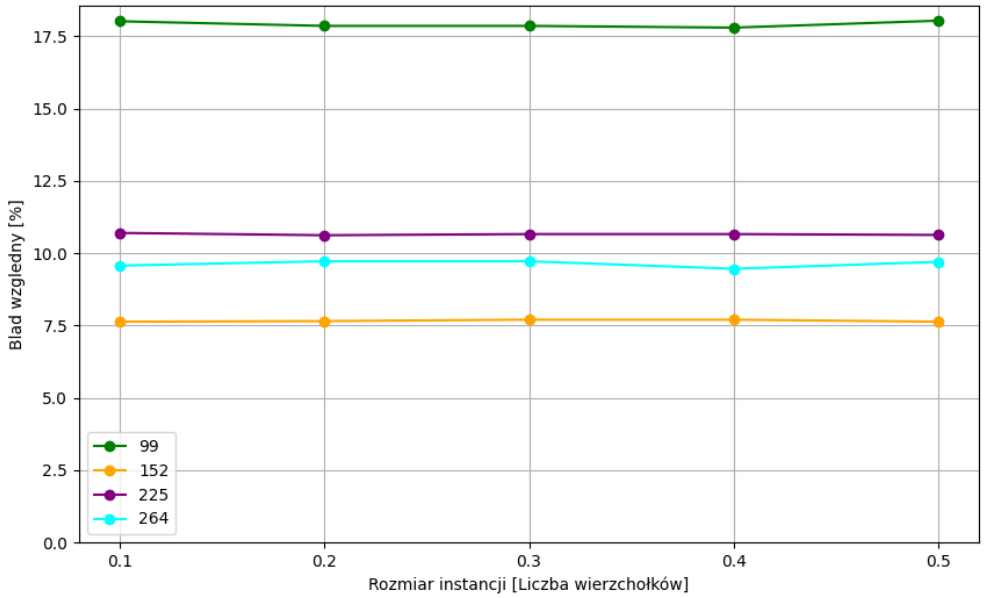
Wnioski: Wyniki są bardzo zbliżone. Różnica dla instancji symetrycznych nie jest większa niż 1 punkt procentowy. Dla asymetrycznych widać nieco większą różnorodność (ok 2 punkty procentowe). W obu przypadkach największy błąd jest dla dużych macierzy

3.5 Badanie wpływu długości tablicy tabu

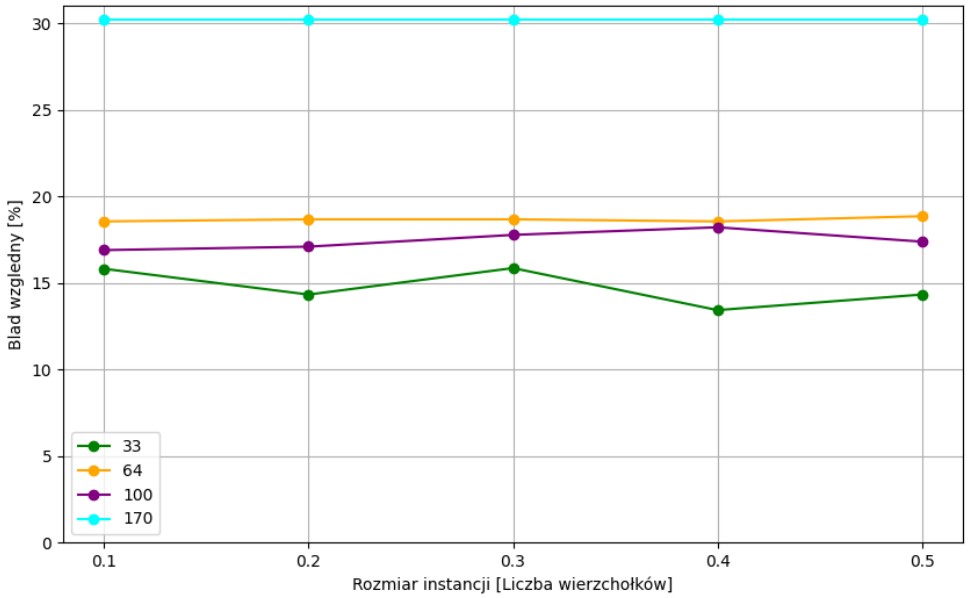
W mojej implementacji elementy pozostają w tablicy tak długo aż ilość elementów w tej tablicy nie przekroczy pewnej liczby. Długość tablicy zależna jest od ilości wierzchołków (n * mnożnik).Do testów wybrałem wartości 0.5,0.4,0.3,0.2,0.1.
Hipoteza: Jakość rozwiązania powinna rosnąć kiedy długość tabeli tabu maleje. Jednakże zbyt mała wartość spowodować cykliczne powracanie do już wygenerowanych rozwiązań.
Badania: Podobnie jak w ostatnim badaniu dla każdej instancji testuje wszystkie 5 wartości. Żeby otrzymać dobrą próbkę badanie powtarzam cztero krotnie dla każdej instancji. Dla tego wykonuję (4*ASYM + 4*ASM)*5 * 4 = 160 badań.

	Symetryczne				Asymetryczne			
Rozmiar[Liczba wierzchołków]	99	152	225	264	33	64	100	170
0.50	18.04	7.63	10.63	9.70	14.33	18.86	17.39	30.20
0.40	17.80	7.70	10.66	9.46	13.43	18.56	18.22	30.20
0.30	17.86	7.70	10.66	9.72	15.86	18.68	17.78	30.20
0.20	17.86	7.65	10.62	9.72	14.33	18.68	17.10	30.20
0.10	18.02	7.63	10.70	9.57	15.82	18.56	16.90	30.20

Table 5: Błędy w wynikach algorytmu dla macierzy symetrycznych i niesymetrycznych



Rysunek 8: Wyniki badań dla macierzy symetrycznych



Rysunek 9: Wyniki badań dla macierzy asymetrycznych

Wnioski: Wyniki są bardzo zbliżone zarówno w tym badaniu jak i w poprzednim. Tak samo jak w poprzednim badaniu wyniki dla macierzy asymetrycznych wykazują znacznie większe zróżnicowanie

3.6 Podsumowanie

W badaniach 3.4 i 3.5 używałem NN jako metodę tworzenia pierwszego rozwiązania. Możliwe że wyniki dla tych instancji były na tyle dobre że algorytm utykał w optimum lokalnym.

4 Simulated anealing

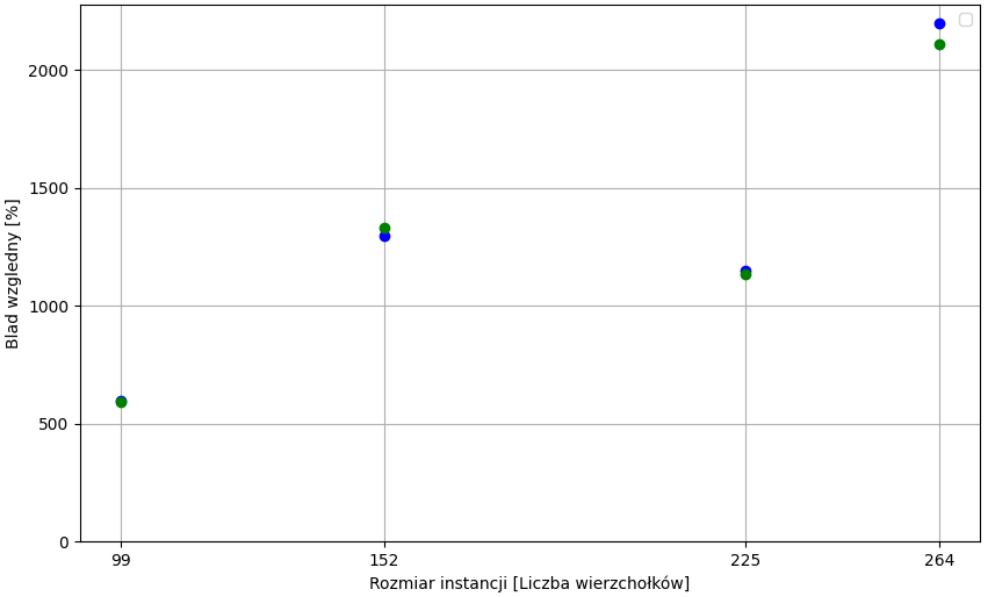
4.1 Badanie wpływu metody przeszukiwania sąsiedztwa

W moim algorytmie zaimplementowałem dwie różne metody przeszukiwania sąsiedztwa.

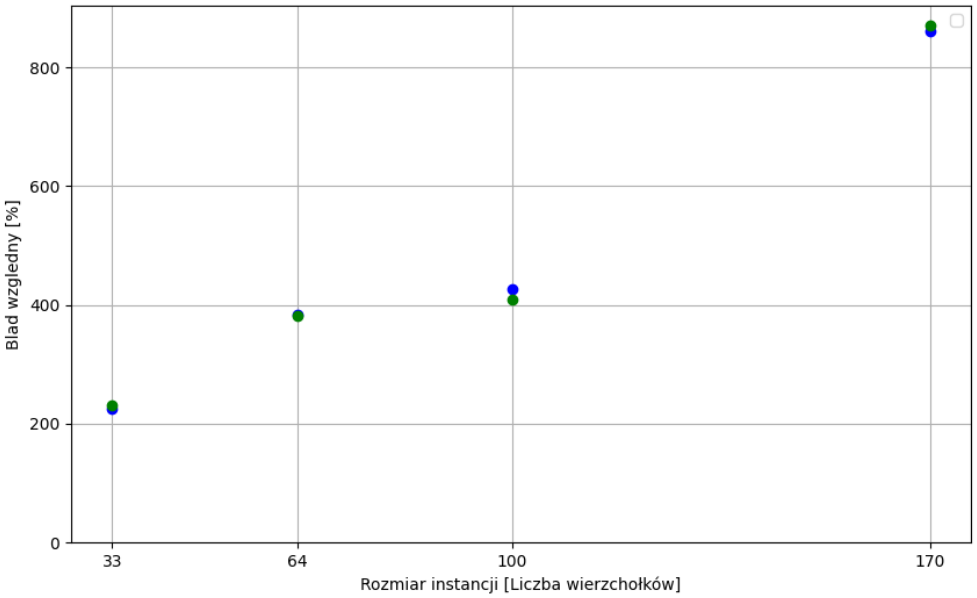
1. Swap - Wybieram dwa wierzchołki i zamienia je ze sobą. W mojej implementacji sprawdzam wszystkie unikalne możliwości więc generuję n*(n-1)/n sąsiadów.

2. Insert - wybieram wierzchołek i miejsce w tablicy. Wierzchołek kopiuje a następnie usuwam z tablicy. Na koniec kopię wierzchołka wstawiam w wybrane miejsce. W mojej implementacji obie wartości losuję. Wykonuję n*(n-1)/n losowań.

Hipoteza: W tabu search metoda przeszukiwania insert powinna być znacznie lepsza niż swap.
Badania: Wykonuję badania dla każdej z 8 instancji. Żeby otrzymać dobrą próbkę badanie powtarzam cztero krotnie dla każdej instancji. Więc wykonuje (4*ASYM + 4*ASM)*4 * 2 = 64 badań.



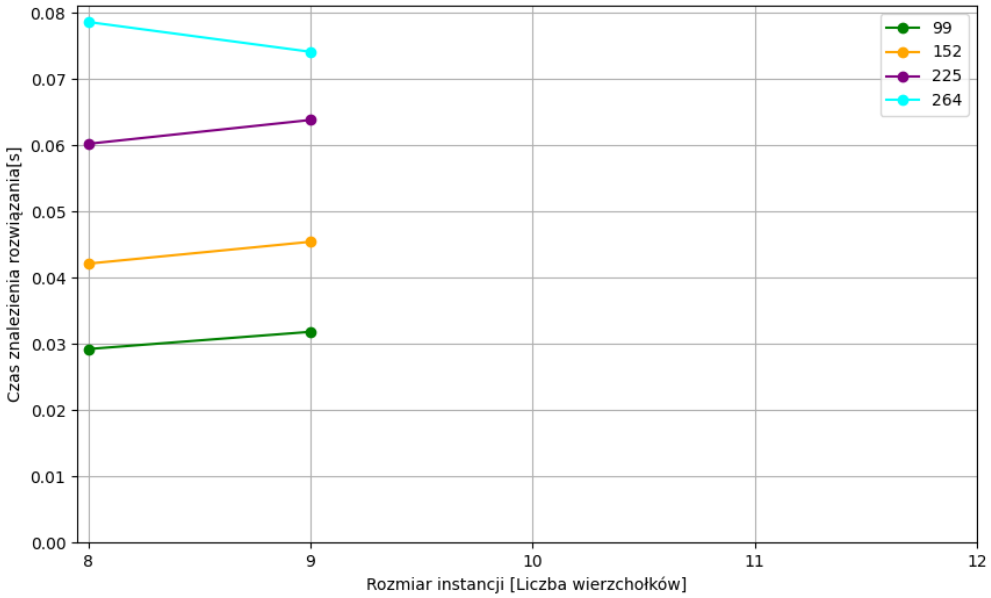
Rysunek 10: Wyniki badań dla macierzy symetrycznych



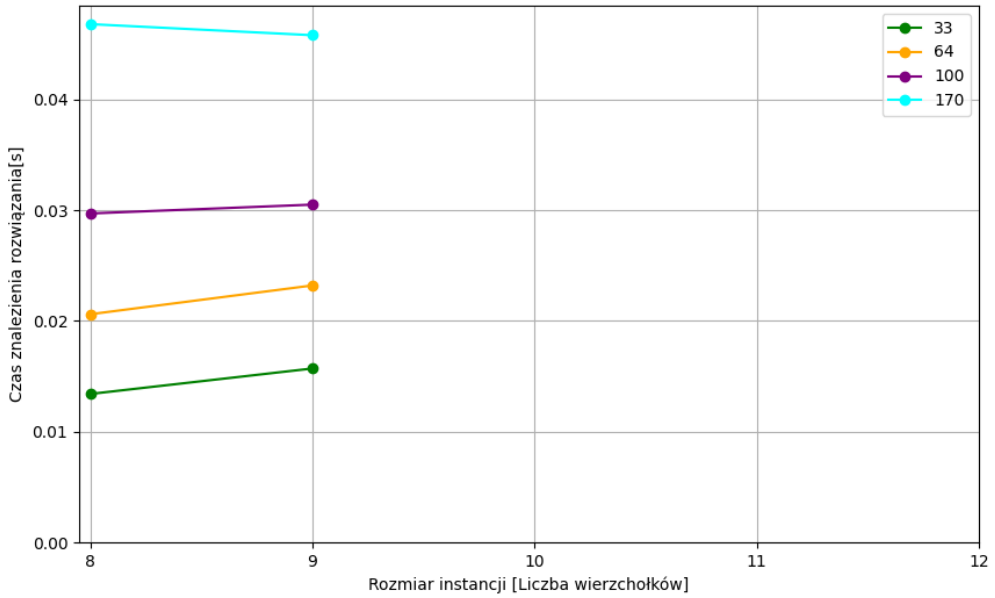
Rysunek 11: Wyniki badań dla macierzy asymetrycznych

	Symetryczne				Asymetryczne			
Rozmiar	99	152	225	264	33	64	100	170
8	0.029	0.042	0.060	0.079	0.013	0.021	0.030	0.047
9	0.032	0.045	0.064	0.074	0.016	0.023	0.030	0.046
10	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
11	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
12	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Table 6: Czas znalezienia wyniku algorytmu dla macierzy symetrycznych i niesymetrycznych[s]



Rysunek 12: Wyniki badań czasu dla macierzy symetrycznych



Rysunek 13: Wyniki badań czasu dla macierzy asymetrycznych

* Niebieskie punkty to wyniki dla Swap a zielone dla Insert
Wnioski: W tym badaniu błędy dla Random i NN były bardzo zbliżone. Czas znalezienia wyniku dla obu metod jest bardzo zbliżony (różnica ok 0.01 sekundy).

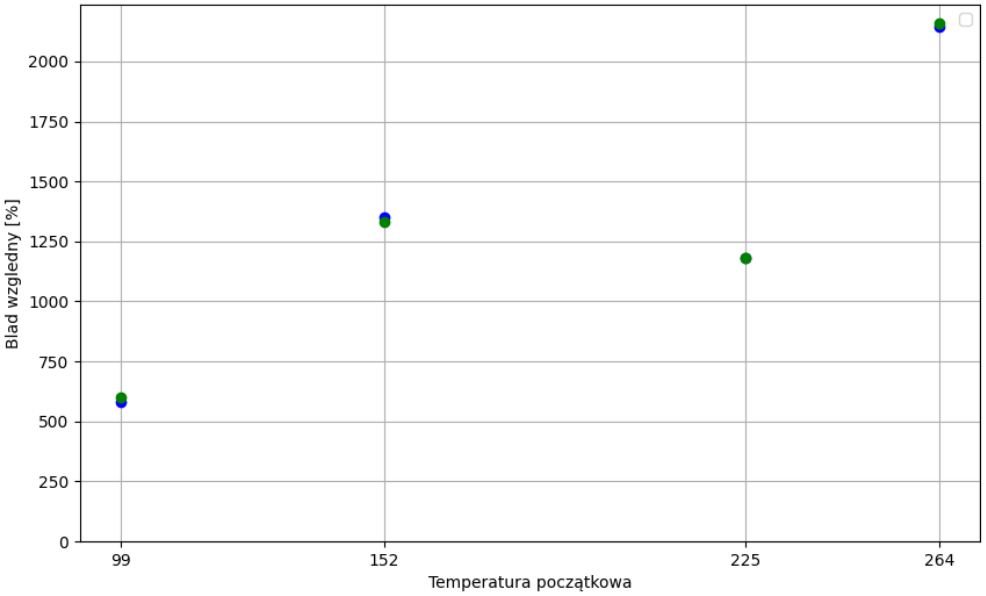
4.2 Badanie wpływu metody tworzenia pierwszego rozwiązania

W moim algorytmie zaimplementowałem dwie różne metody tworzenia pierwszego rozwiązania.

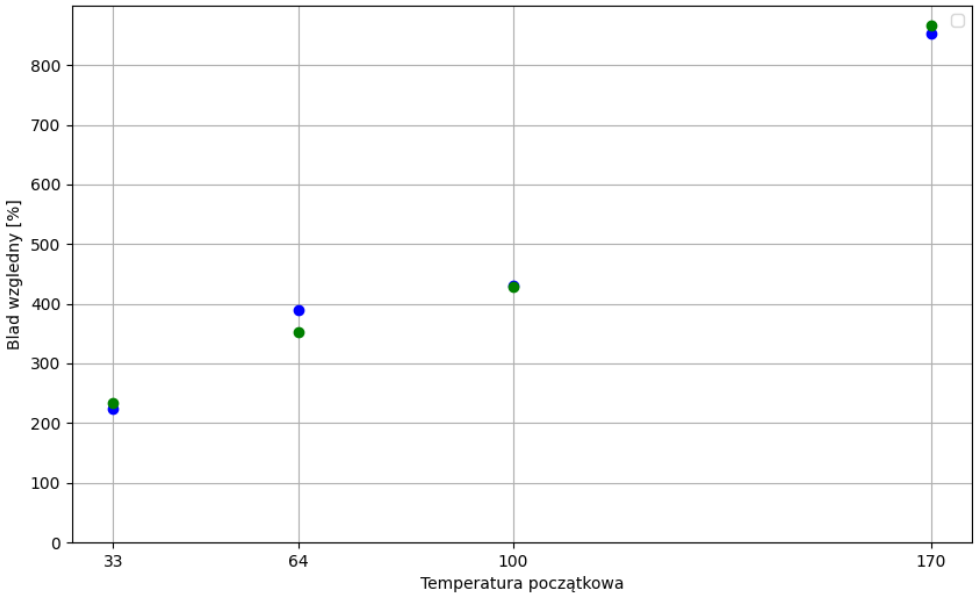
1. Losowa - Pierwsze rozwiązanie jest zupełnie losowe. Najpierw tworzę wektor z liczbami od 0 do n - 1. Następnie na tym wektorze używam funkcji shuffle.

2. Nearest neighbour - Korzystam z wcześniej zaimplementowanego algorytmu NN.

Hipoteza: Wyniki dla metody losowej będą znacznie gorsze (Większy błąd wzgledny) od NN. Ale jest też niewielka szansa na wyniki będące bliżej optymalnego rozwiązania. Jednakże wykonanie wielu pomiarów powinno wykluczyć znaczne odstępstwa.
Badania: Podobnie jak w poprzednim przypadku wykonuje badania dla każdej z 8 instancji po 4 powtórzenia. Więc znowu wykonuje (4*ASYM + 4*ASM)*4 * 2 = 64



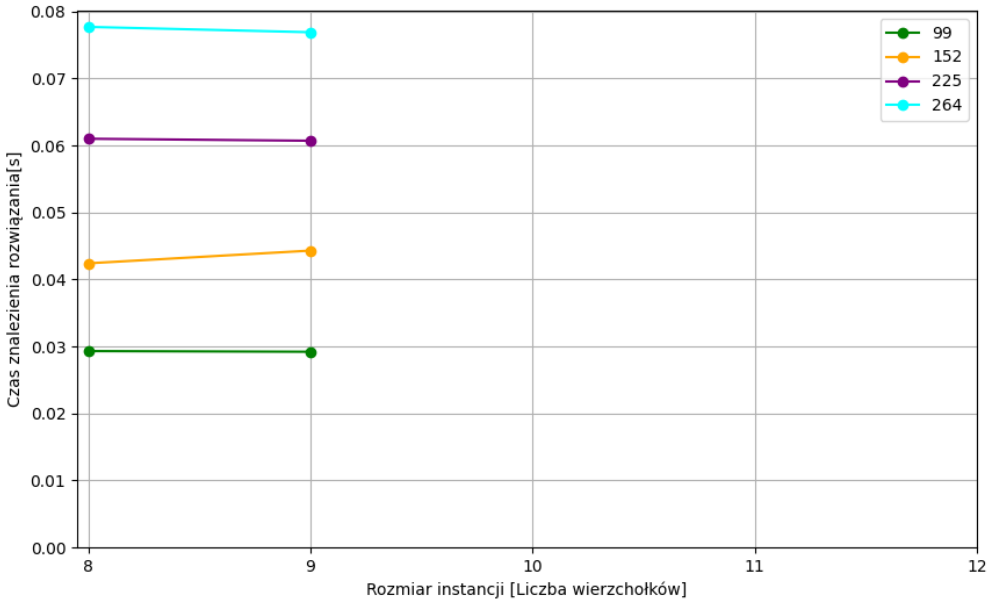
Rysunek 14: Wyniki badań dla macierzy symetrycznych



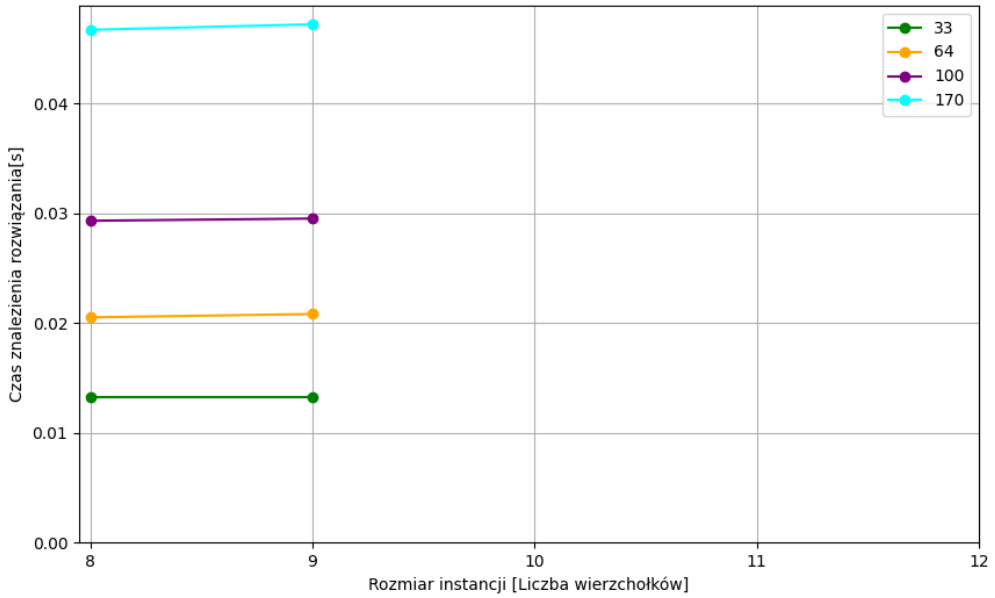
Rysunek 15: Wyniki badań dla macierzy asymetrycznych

	Symetryczne				Asymetryczne			
Rozmiar	99	152	225	264	33	64	100	170
8	0.029	0.042	0.061	0.078	0.013	0.021	0.029	0.047
9	0.029	0.044	0.061	0.077	0.013	0.021	0.029	0.047
10	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
11	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
12	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Table 7: Czas znalezienia wyniku algorytmu dla macierzy symetrycznych i niesymetrycznych[s]



Rysunek 16: Wyniki badań czasu dla macierzy symetrycznych

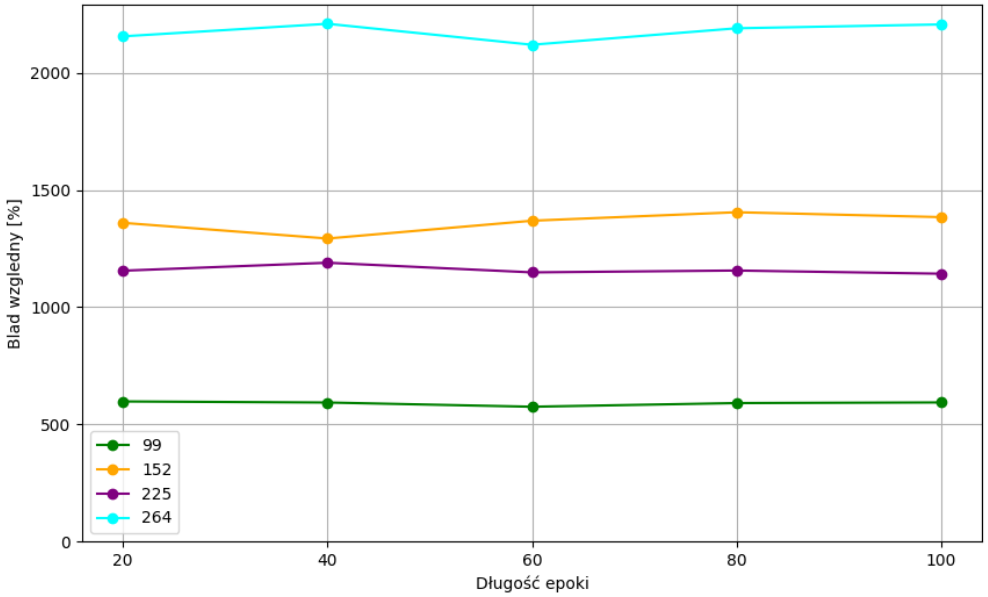


Rysunek 17: Wyniki badań czasu dla macierzy asymetrycznych

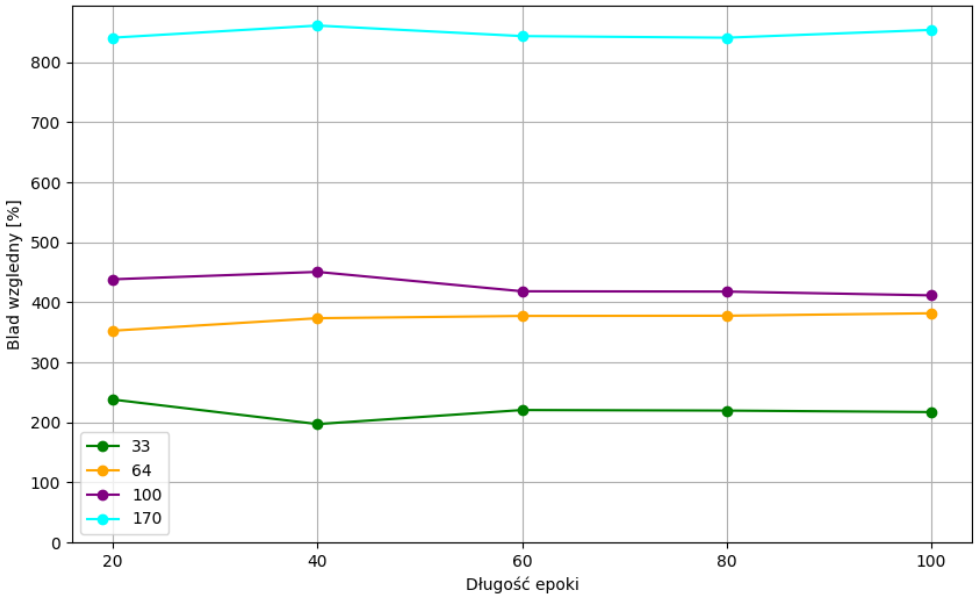
* Niebieskie punkty to wyniki dla Swap a zielone dla Insert
Wnioski: Podobnie jak w badaniu 3.2 wyniki dla obu metod były bardzo zbliżone.Czas znalezienia wyniku dla obu metod jest bardzo zbliżony (różnica ok 0.01 sekundy).

4.3 Wpływ długości epoki

W mojej implementacji długość epoki jest wczytywana z pliku config. Podczas implementacji wartości dla których dostawałem najlepsze wyniki były zbliżone do 50. Więc do badań używam wartości 20,40,60,80,100.
Hipoteza: Wraz ze wzrostem długości epoki jakość rozwiązania będzie rosła.
Badania: Dla każdej instancji testuje wszystkie 5 wartości. Żeby otrzymać dobrą próbkę badanie powtarzam cztero krotnie dla każdej instancji. Dla tego wykonuję (4*ASYM + 4*ASM)*5 * 4 = 160 badań.



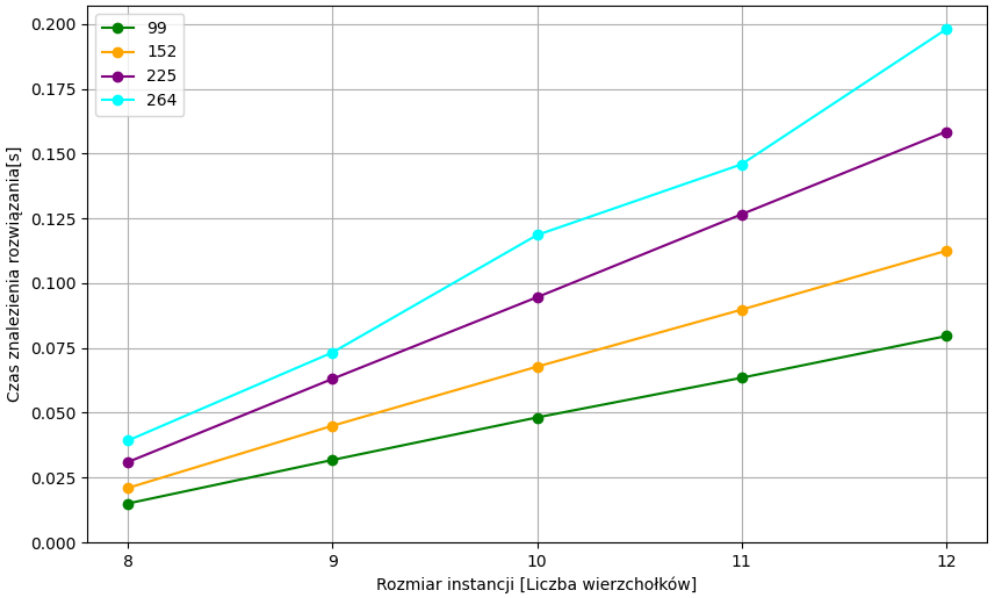
Rysunek 18: Wyniki badań dla macierzy symetrycznych



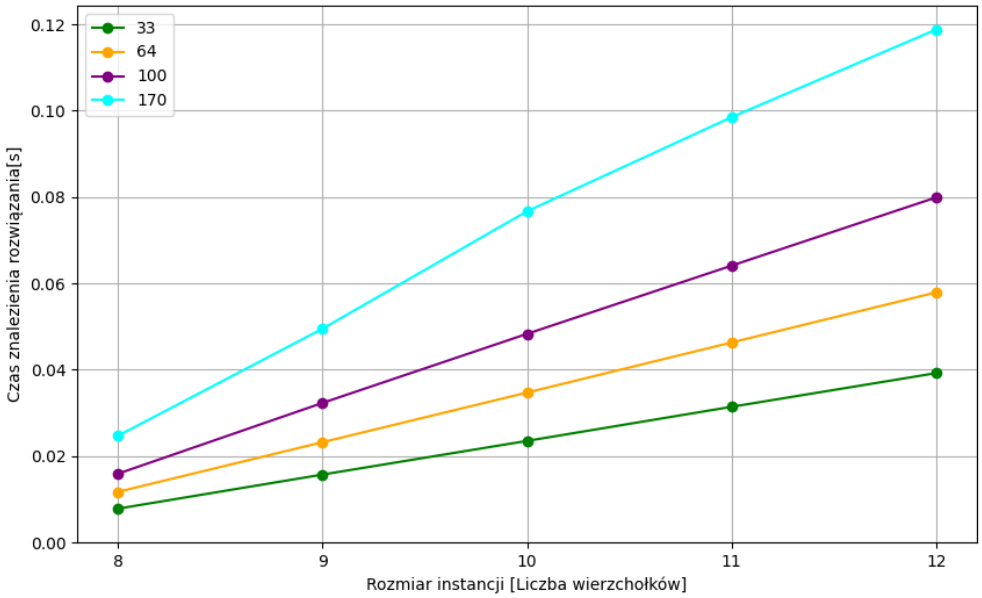
Rysunek 19: Wyniki badań dla macierzy asymetrycznych

	Symetryczne				Asymetryczne			
Rozmiar	99	152	225	264	33	64	100	170
8	0.015	0.021	0.031	0.039	0.008	0.012	0.016	0.025
9	0.032	0.045	0.063	0.073	0.016	0.023	0.032	0.050
10	0.048	0.068	0.095	0.119	0.024	0.035	0.048	0.077
11	0.064	0.090	0.127	0.146	0.031	0.046	0.064	0.099
12	0.080	0.113	0.159	0.198	0.039	0.058	0.080	0.119

Table 8: Czas znalezienia wyniku algorytmu dla macierzy symetrycznych i niesymetrycznych[s]



Rysunek 20: Wyniki badań czasu dla macierzy symetrycznych

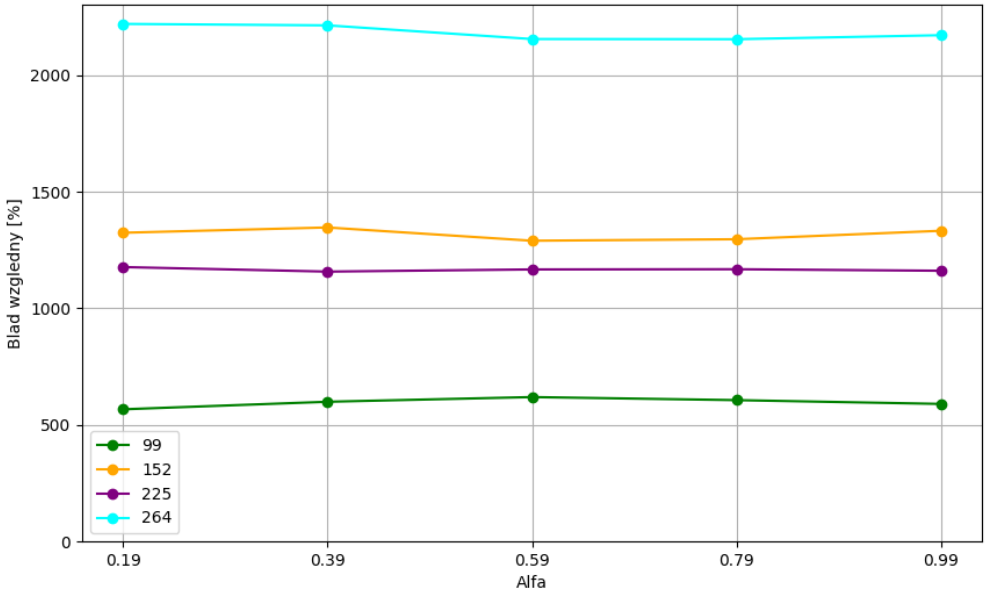


Rysunek 21: Wyniki badań czasu dla macierzy asymetrycznych

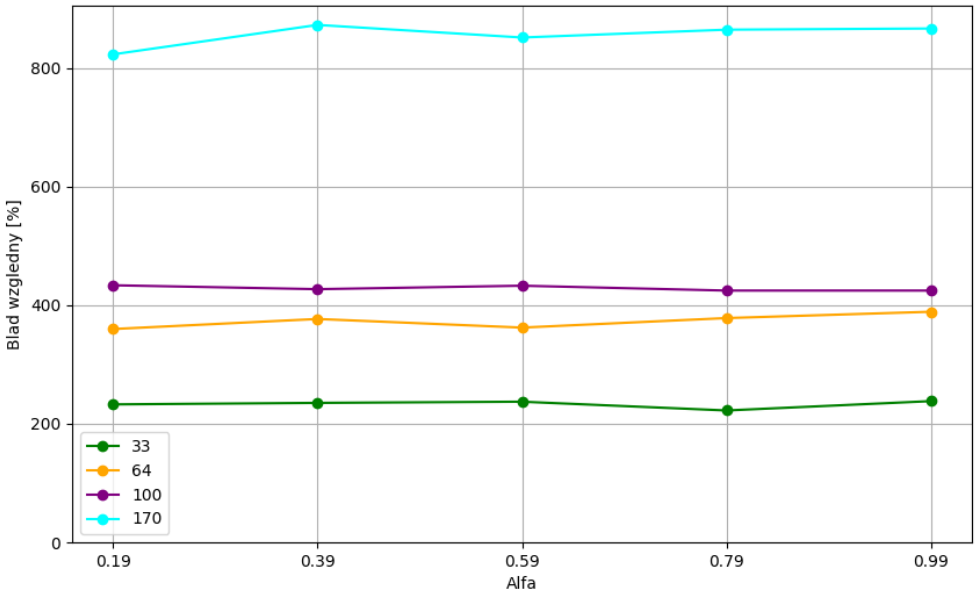
Wnioski: Z badania wynikia że czas rośnie liniowo w raz ze wzrostem długości epok. Dla wszyskich długości epok różnica w błędzie była niewielka. Jednakże otrzymany błąd był bardzo duży ok 800%

4.4 Wielkość alfa

W mojej implementacji wartość alfa jest wczytywana z pliku config. Alfa powinna być w zakresie 0 do 1. Więc do testów wybrałem wartości 0.19,0.39,0.59,0.79,0.99.
Hipoteza: Wraz ze wzrostem wartości alfy wyniki powinny się poprawiać. Alfa = 0.99 powinna dawać najlepsze wyniki.
Badania: Dla każdej instancji testuje wszystkie 5 wartości. Żeby otrzymać dobrą próbkę badanie powtarzam cztero krotnie dla każdej instancji. Dla tego wykonuję (4*ASYM + 4*ASM)*5 * 4 = 160 badań.



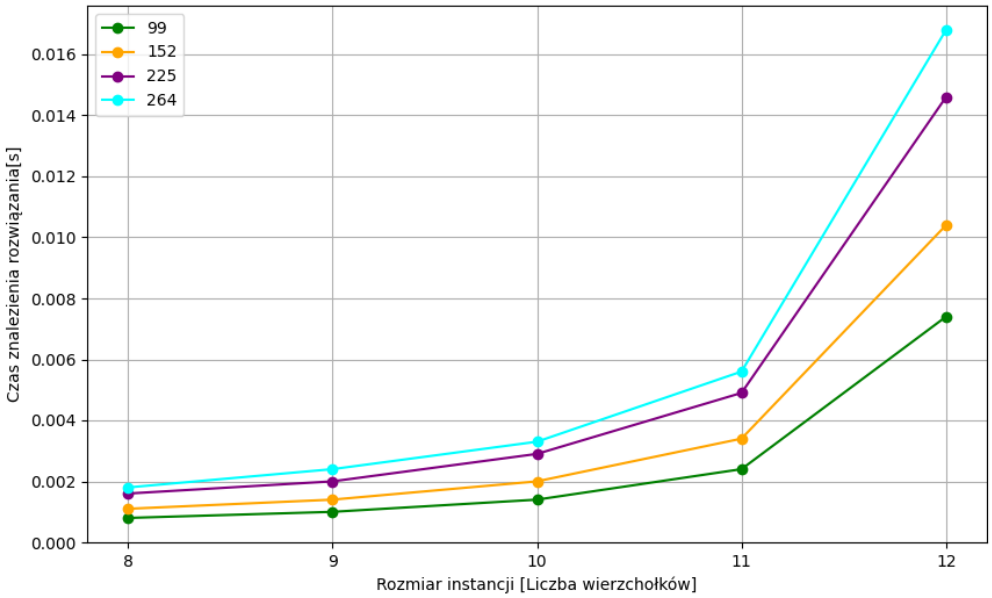
Rysunek 22: Wyniki badań dla macierzy symetrycznych



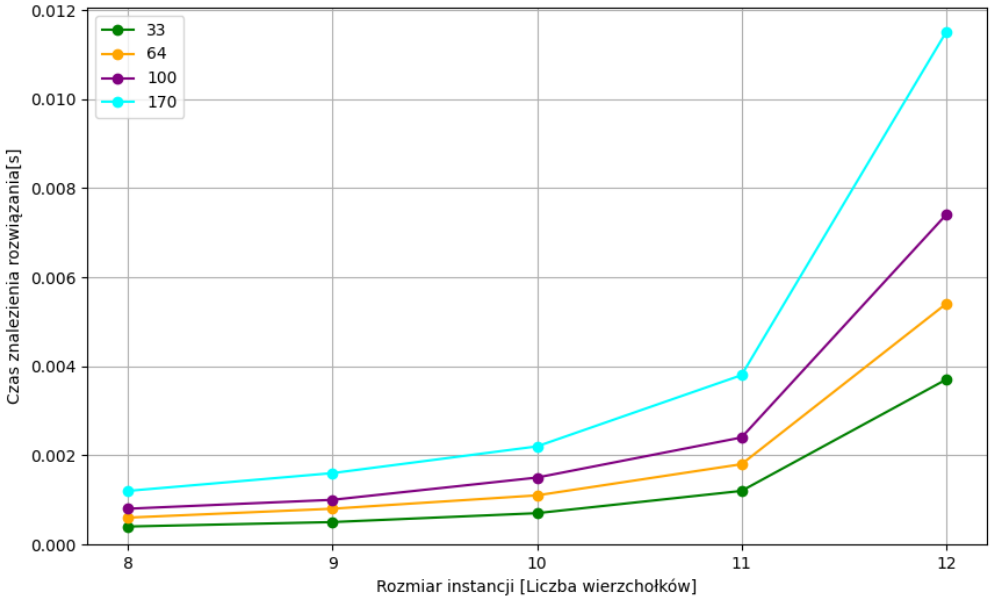
Rysunek 23: Wyniki badań dla macierzy asymetrycznych

	Symetryczne				Asymetryczne			
Rozmiar	99	152	225	264	33	64	100	170
8	0.001	0.001	0.002	0.002	0.000	0.001	0.001	0.001
9	0.001	0.001	0.002	0.002	0.001	0.001	0.001	0.002
10	0.001	0.002	0.003	0.003	0.001	0.001	0.002	0.002
11	0.002	0.003	0.005	0.006	0.001	0.002	0.002	0.004
12	0.007	0.010	0.015	0.017	0.004	0.005	0.007	0.011

Table 9: Czas znalezienia wyniku algorytmu dla macierzy symetrycznych i niesymetrycznych[s]



Rysunek 24: Wyniki badań dla macierzy symetrycznych

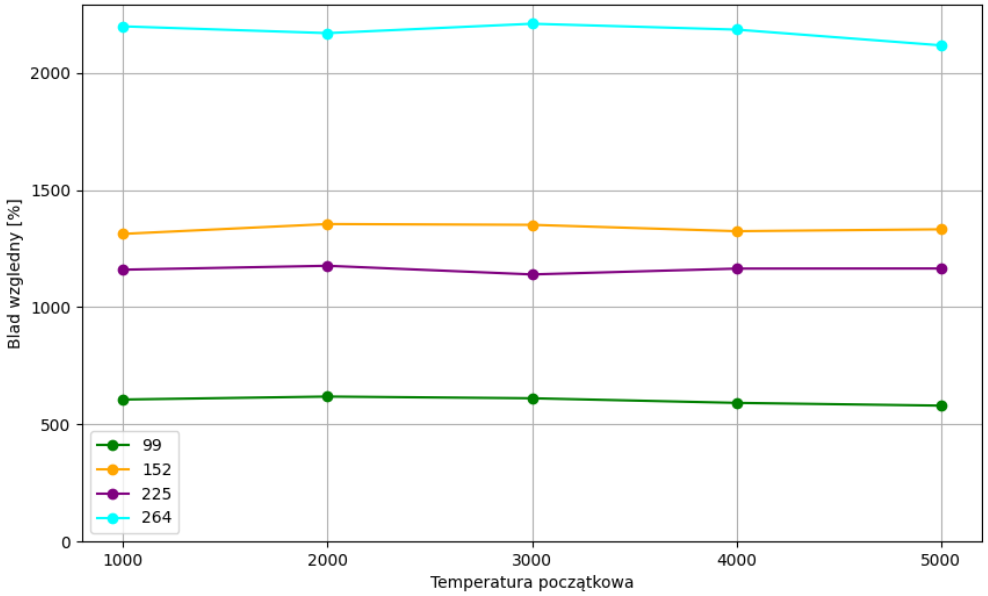


Rysunek 25: Wyniki badań dla macierzy asymetrycznych

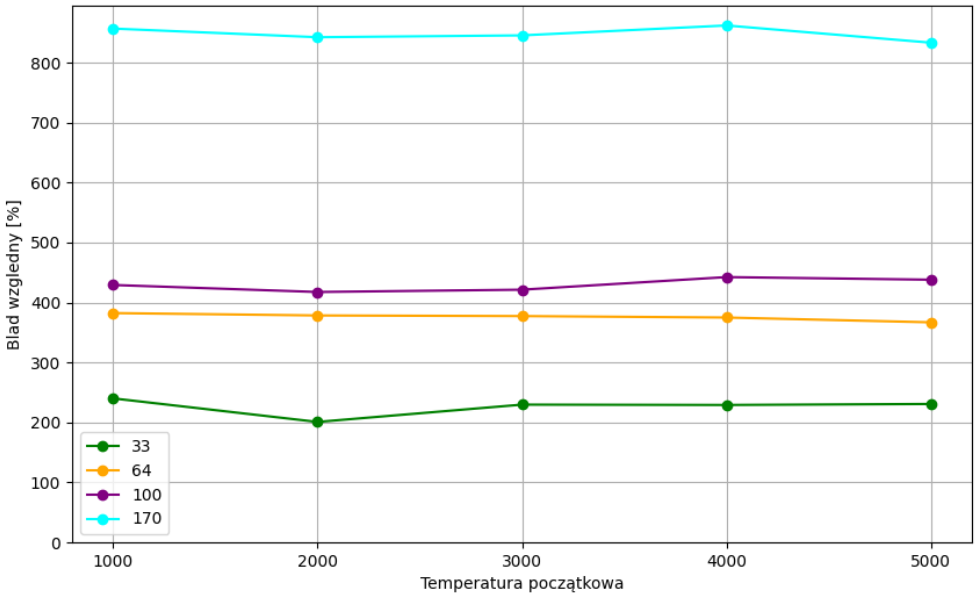
Wnioski: Różnica błędów dla wartości alfy była nieznacznia ok 10 punktów procentowych. Czas znalezienia rozwiązań znacznie poprawiał się dla małych wartości alfa. Czas potrzebny na znalezienie wyniku rośnie wykładniczo kiedy alfa zbliża się do 1.

4.5 Temperatura startowa

W mojej implementacji temperatura startowa jest wczytywana z pliku config. Podczas implementacji temperatura startowa zbliżona do 5000 dawała najlepsze wyniki. Więc do badań wybrałem wartości 1000,2000,3000,4000,5000.
Hipoteza: Temperatura startowa powinna być stosunkowo wysoka więc wraz ze wzrostem wyniki powinny się poprawiać.
Badania: Dla każdej instancji testuje wszystkie 5 wartości. Żeby otrzymać dobrą próbkę badanie powtarzam cztero krotnie dla każdej instancji. Dla tego wykonuję (4*ASYM + 4*ASM)*5 * 4 = 160 badań.



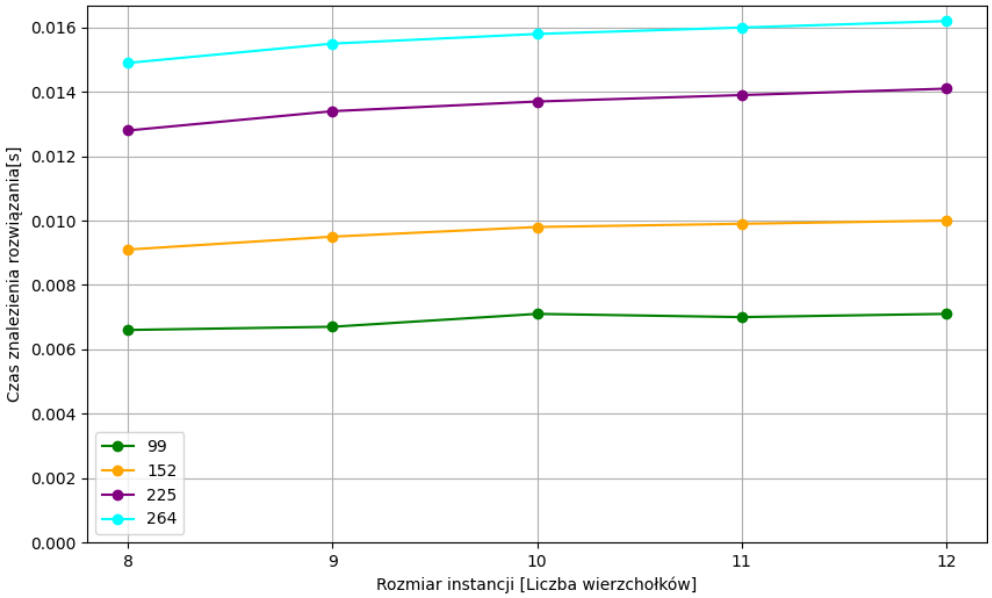
Rysunek 26: Wyniki badań dla macierzy symetrycznych



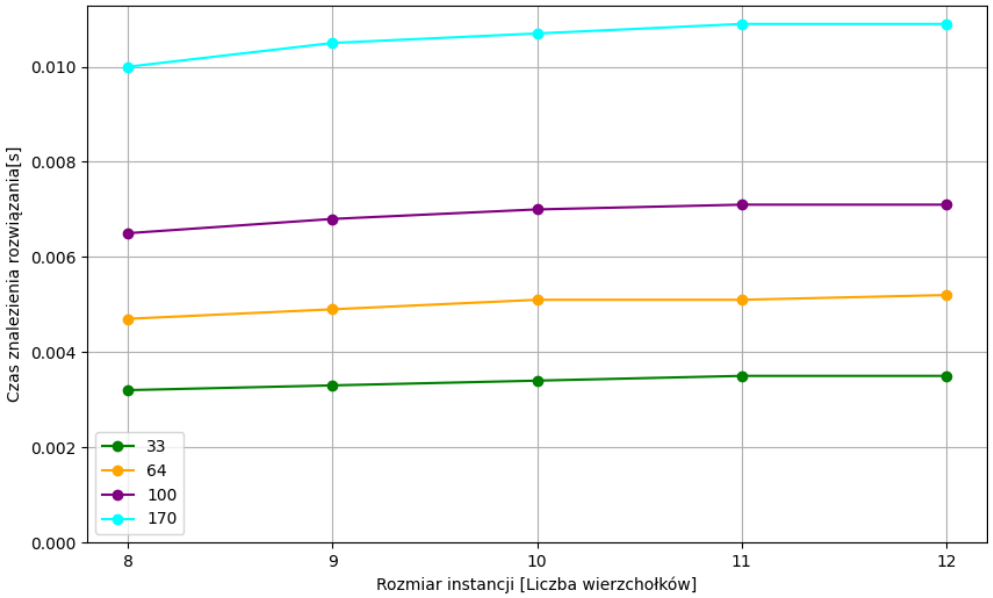
Rysunek 27: Wyniki badań dla macierzy asymetrycznych

	Symetryczne				Asymetryczne			
Rozmiar	99	152	225	264	33	64	100	170
8	0.007	0.009	0.013	0.015	0.003	0.005	0.006	0.010
9	0.007	0.009	0.013	0.015	0.003	0.005	0.007	0.011
10	0.007	0.010	0.014	0.016	0.003	0.005	0.007	0.011
11	0.007	0.010	0.014	0.016	0.004	0.005	0.007	0.011
12	0.007	0.010	0.014	0.016	0.004	0.005	0.007	0.011

Table 10: Czas znalezienia wyniku algorytmu dla macierzy symetrycznych i niesymetrycznych[s]



Rysunek 28: Wyniki badań dla macierzy symetrycznych



Rysunek 29: Wyniki badań dla macierzy asymetrycznych

Wnioski: Podobnie jak ostatnio różnica między błędami dla kolejnych wartości początkowych. Tym razem czas znalezienia rozwiązania też nie różnił się znacząco ok 0.01s.

4.6 Podsumowanie

Różnice które powstają przy stosowani różnych parametrów są znacznie lepiej widoczne kiedy badamy czas otrzymania najlepszego wyniku. Niestety błąd wzgledny dla tego algorytmu możliwe że w implementacji pojawiły się błędy.

5 Algorytm mrówkowy

5.1 Opis Algorytmu

Algorytm mrówkwy jest algorytmem metahurystycznym który modeluje zachowanie mrówek. Mrówki w wyruszają z mrowiska w poszukiwaniu jedzenia w losowych kierunkach. Po znalezieniu jedzenia mrówki wracają do mrowiska i pozostawiają za sobą feromony. Ilość zostawionych feromonów zależy od ilości znalezionego jedzenia. Następnie mrówki ponownie wyruszają z mrowiska ścieżki wybierają z pewnym prawdopodobieństwem które zależy od ilości feromonów. Jako że feromony parują to ścieżki nie uczęszczane zanikają a te prowadzące do dobrych rozwiązań są wzmacniane.
W mojej implementacji ścieżki wybierane są z pewnym prawdopodobieństwem wyliczonym ze wzoru:

$$p_{ij}^{k}(t) = \frac{a_{ij}(t)}{\sum l \in N_i a_{il}(t)}$$

Gdzie k to k-ta mrówka ;a i , j to ścieżka z i-tego wierzchołka do j-tego. Natomiast wartości początkowe feromonów szacowane są ze wzoru $\tau_0 = \frac{k}{C^{nn}}$ W tym wzorze C^{nn} to wynik algorytmu NN dla tej instancji.

5.2 Badanie wpływu typu rozkładu feromonów

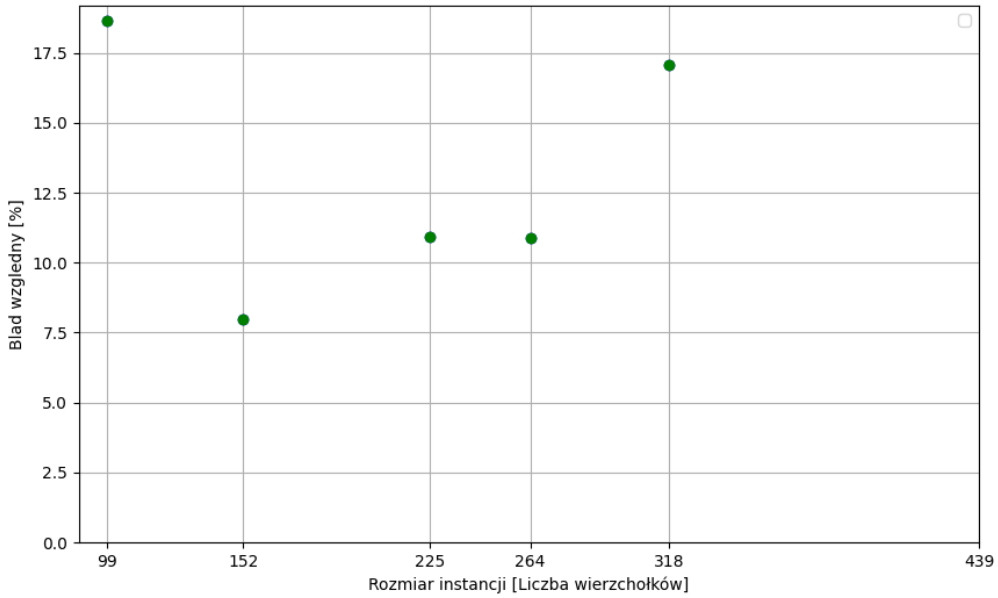
W mojej implementacji mam dwa typy rozkładu feromonów

1. CAS - Algorytm cykliczny aktualizuje wartości feromonów po wybraniu całej ścieżki. Aktualizacja przebiega według wzoru $\delta\tau_{ij}^{k}(t, t+n) = n/L^{k}$ gdzie L^{k} to długość otrzymanej trasy.
2. QAS - Algorytm ilościowy aktualizuje wartości feromonów po wybraniu krawędzi Aktualizacja przebiega według wzoru $\delta\tau_{ij}^{k}(t, t+n) = n/d_{ij}$ gdzie d_{ij} to koszt przejścia z i do j.

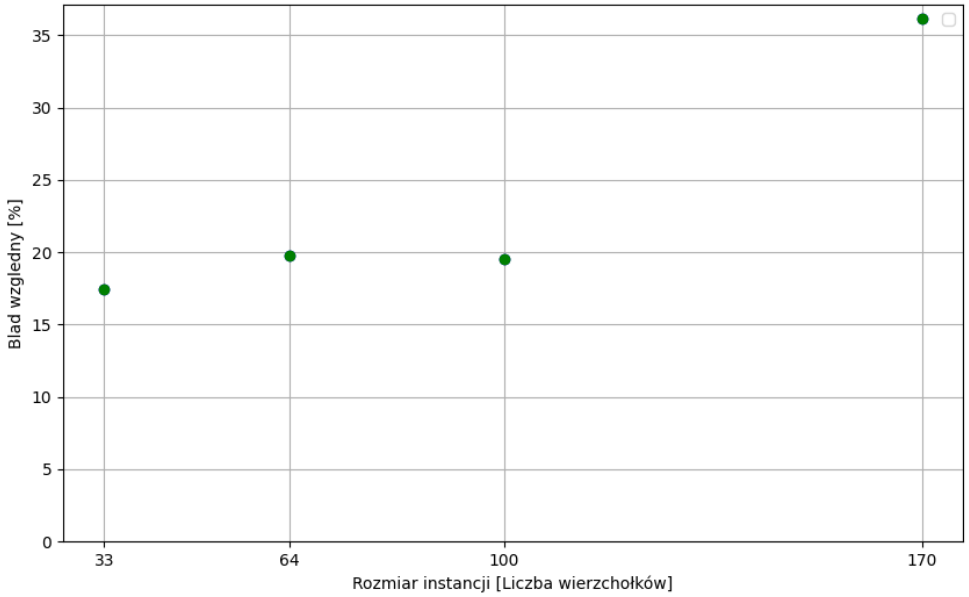
Hipoteza: Algorytm QAS powinien dawać nieznacznie lepsze wyniki.
Badania: Dla obu metod wykonuje badania dla każdej z 8 instancji, powtarzam je cztero krotnie by otrzymać dobrą próbkę. Więc wykonuje (4*ASYM + 4*ASM)*2 * 4 = 64 badań.

	Symetryczne						Asymetryczne			
Rozmiar	99	152	225	264	318	439	33	64	100	170
QAS	18.660	7.980	10.930	10.900	17.060	18.670	17.420	19.740	19.560	36.190
CAS	18.660	7.980	10.930	10.900	17.060	18.670	17.420	19.740	19.560	36.190

Table 11: Błędy w wynikach algorytmu dla macierzy symetrycznych i niesymetrycznych[%]



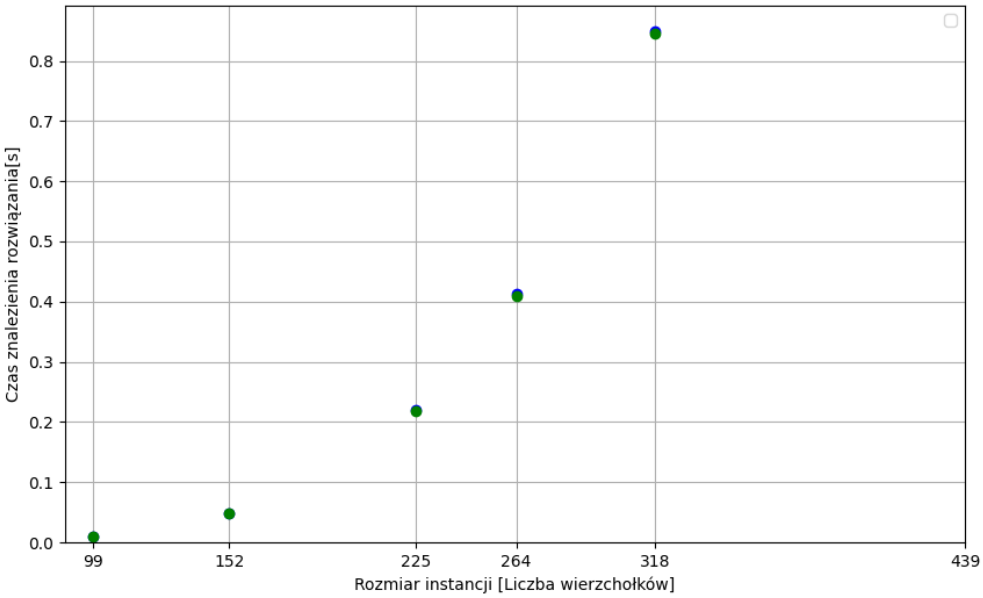
Rysunek 30: Wyniki badań dla macierzy symetrycznych



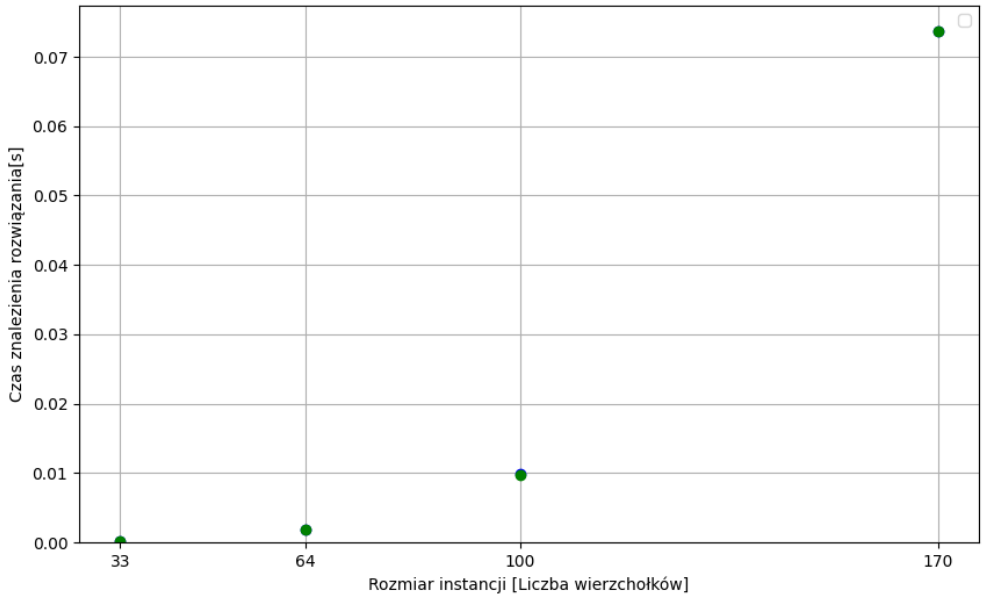
Rysunek 31: Wyniki badań dla macierzy asymetrycznych

	Symetryczne						Asymetryczne			
Rozmiar	99	152	225	264	318	439	33	64	100	170
QAS	0.009	0.049	0.220	0.413	0.850	3.038	0.000	0.002	0.010	0.074
CAS	0.009	0.048	0.218	0.408	0.846	3.026	0.000	0.002	0.010	0.074

Table 12: Czas znalezienia wyniku algorytmu dla macierzy symetrycznych i niesymetrycznych[s]



Rysunek 32: Wyniki badań dla macierzy symetrycznych



Rysunek 33: Wyniki badań dla macierzy asymetrycznych

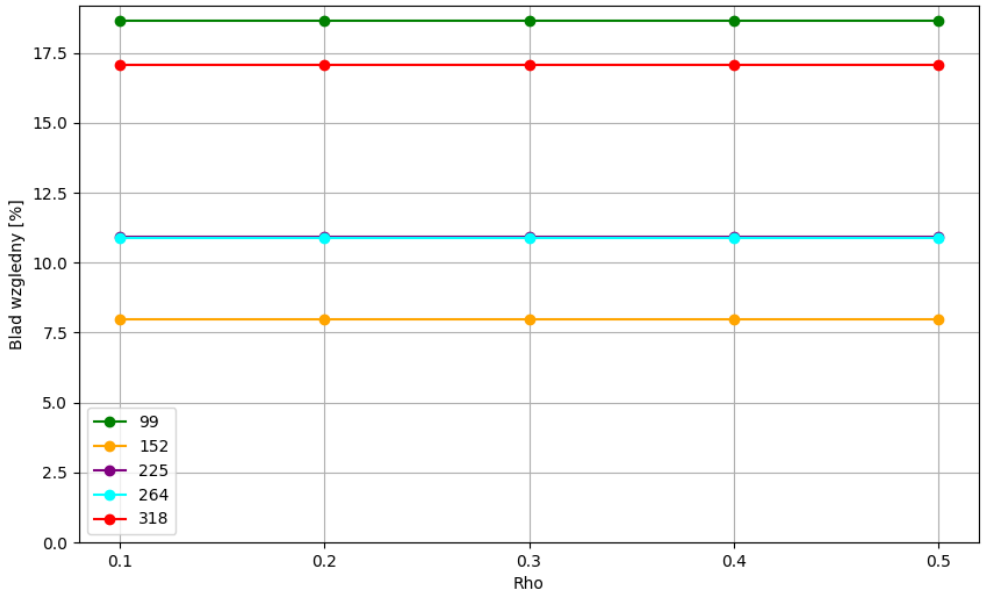
* Niebieskie punkty to wyniki dla Swap a zielone dla Insert
Wnioski: Dla obu zaimplementowanych algorytmów (CAS i QAS) błąd względny jest bardzo podobny (w większości przypadków taki sam). Czas potrzebny na znalezienie rozwiązania rośnie wykładniczo.

5.3 Badanie wpływu wartości rho

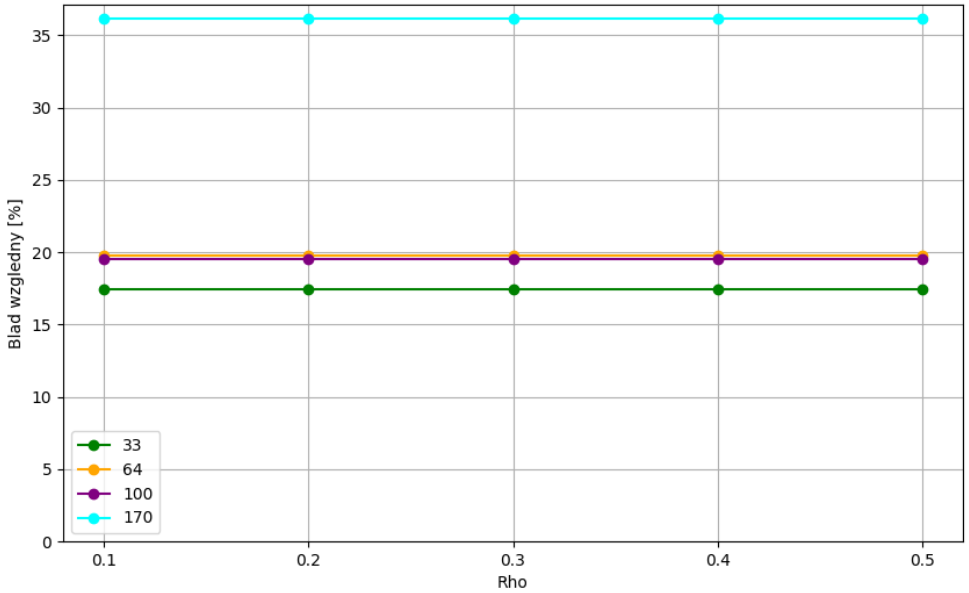
Wartość ρ to współczynnik parowania feromonów. Po tym jak wszystkie mrówki wybrały swoją ścieżkę wartości feromonów są przemnażane przez ρ. Dlatego też wartości ρ powinny być w zakresie 0 do 1. Dlatego wybrałem wartości 0.8,0.6,0.4,0.2
Hipoteza: Wysokie wartości ρ powinny zachęcać mrówki do wybierania różnych nie koniecznie optymalnych ścieżek. Natomiast niskie wartości powinny sprawiać że mrówki będą bardziej zachłannne
Badania: Dla każdej instancji testuje wszystkie 5 wartości. Żeby otrzymać dobrą próbkę badanie powtarzam cztero krotnie dla każdej instancji. Dla tego wykonuję (4*ASYM + 4*ASM)*5 * 4 = 160 badań.

	Symetryczne						Asymetryczne			
Rozmiar	99	152	225	264	318	439	33	64	100	170
0.100	18.660	7.980	10.930	10.900	17.060	18.670	17.420	19.740	19.560	36.190
0.200	18.660	7.980	10.930	10.900	17.060	18.670	17.420	19.740	19.560	36.190
0.300	18.660	7.980	10.930	10.900	17.060	18.670	17.420	19.740	19.560	36.190
0.400	18.660	7.980	10.930	10.900	17.060	18.670	17.420	19.740	19.560	36.190
0.500	18.660	7.980	10.930	10.900	17.060	18.670	17.420	19.740	19.560	36.190

Table 13: Błędy w wynikach algorytmu dla macierzy symetrycznych i niesymetrycznych[%]



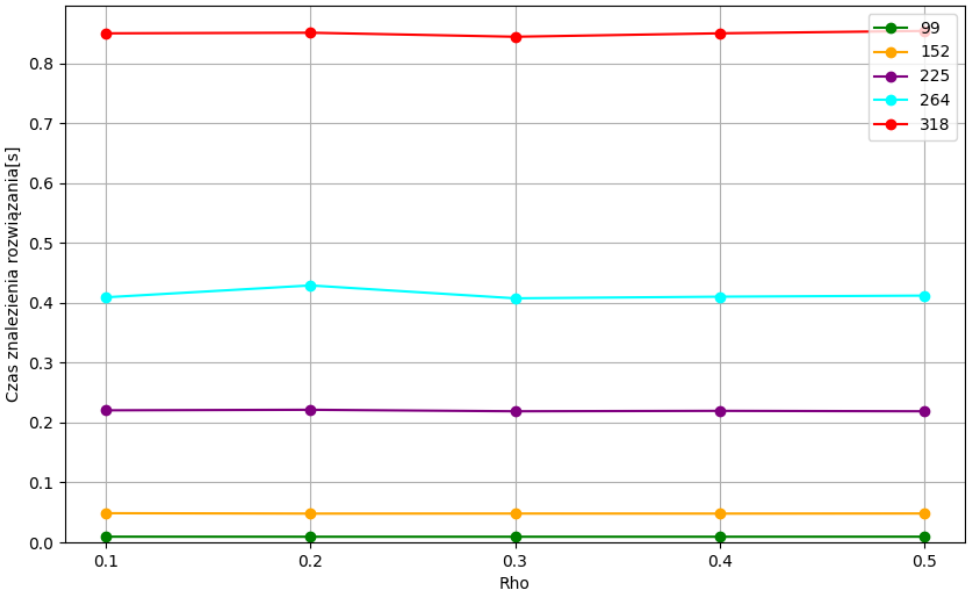
Rysunek 34: Wyniki badań dla macierzy symetrycznych



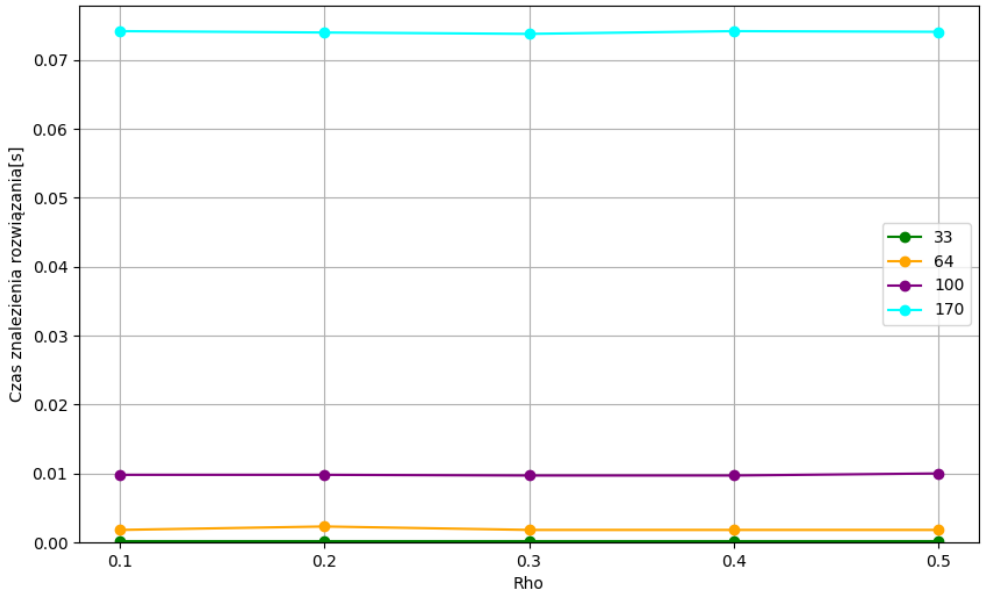
Rysunek 35: Wyniki badań dla macierzy asymetrycznych

	Symetryczne						Asymetryczne			
Rozmiar	99	152	225	264	318	439	33	64	100	170
0.100	0.009	0.049	0.221	0.409	0.851	3.019	0.000	0.002	0.010	0.074
0.200	0.009	0.048	0.222	0.429	0.852	3.031	0.000	0.002	0.010	0.074
0.300	0.009	0.048	0.219	0.408	0.845	3.026	0.000	0.002	0.010	0.074
0.400	0.009	0.048	0.220	0.410	0.851	3.026	0.000	0.002	0.010	0.074
0.500	0.009	0.048	0.219	0.412	0.855	3.012	0.000	0.002	0.010	0.074

Table 14: Czas znalezienia wyniku algorytmu dla macierzy symetrycznych i niesymetrycznych[s]



Rysunek 36: Wyniki badań dla macierzy symetrycznych



Rysunek 37: Wyniki badań dla macierzy asymetrycznych

Wnioski: W otrzymanych wynikach nie ma większych różnic dla różnych wartości rho. Może to być spowodowane miejscem w którym feromony parują (Po wyliczeniu ścieżek dla wszyskich mrówek).

5.4 Badanie wpływu stosunku alfy do bety

W mojej implementacji używam tablic decyzyjnych gdzie

$$A_i = [a_{ij}(t)]_{Ni}$$

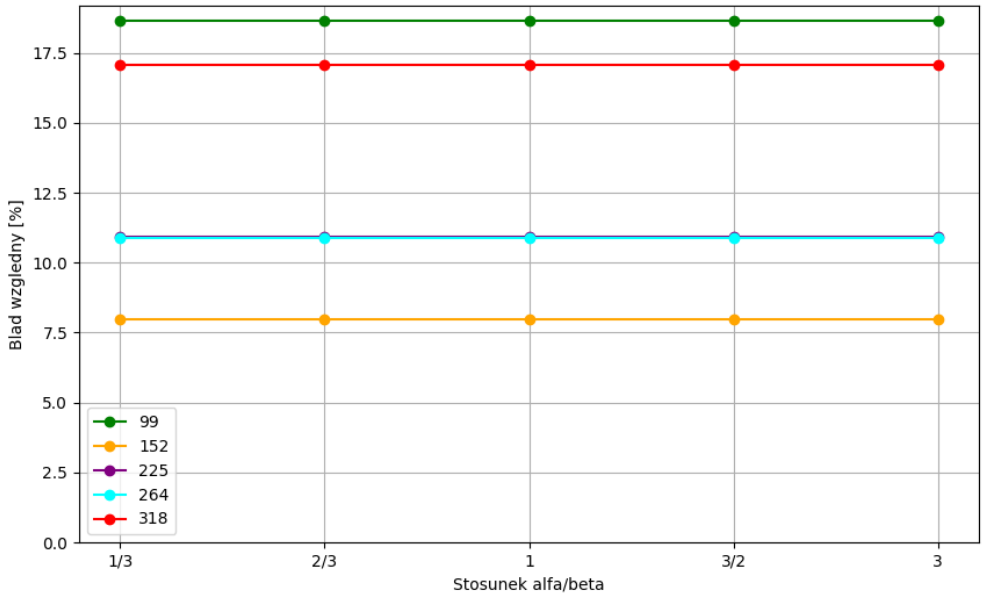
Wartości A_i wyliczam na początku nowego pokolenia. Natomiast elementy a_{ij} wyliczam ze wzoru:

$$a_{ij}t = \frac{[\tau_{ij}(t)]^\alpha * [\tau_{ij}(t)]^\beta}{\sum l \in N_i [\tau_{ij}(t)]^\alpha * [\tau_{ij}(t)]^\beta}$$

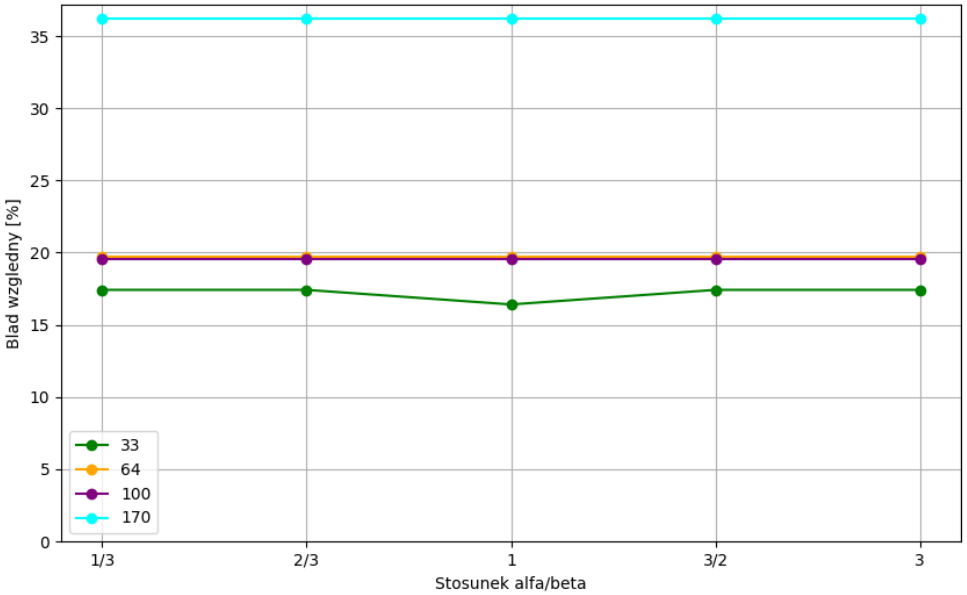
Do testów wybrałem 5 wartości $\frac{\alpha}{\beta}$ $\frac{1}{3}, \frac{2}{3}, 1, \frac{3}{2}, 3$
Hipoteza: Zwiększając element α zwiększa się szansa na wybranie ścieżki z dużą ilością feromonów. Natomiast Zwiększając β zwiększa się szansa na wybranie ścieżki z niewielką ilością feromonów. Kiedy $\alpha = \beta$ wynik będzie najgorszy. Ale dla małego α i dużego β powinien być najlepszy.
Badania: Dla każdej instancji testuje wszystkie 5 wartości. Żeby otrzymać dobrą próbkę badanie powtarzam cztero krotnie dla każdej instancji. Dla tego wykonuję (4*ASYM + 4*ASM)*5 * 4 = 160 badań.

	Symetryczne						Asymetryczne			
Rozmiar	99	152	225	264	318	439	33	64	100	170
1/3	18.660	7.980	10.930	10.900	17.060	18.670	17.420	19.740	19.560	36.190
2/3	18.660	7.980	10.930	10.900	17.060	18.670	17.420	19.740	19.560	36.190
1	18.660	7.980	10.930	10.900	17.060	18.670	16.410	19.740	19.560	36.190
3/2	18.660	7.980	10.930	10.900	17.060	18.670	17.420	19.740	19.560	36.190
3	18.660	7.980	10.930	10.900	17.060	18.670	17.420	19.740	19.560	36.190

Table 15: Błędy w wynikach algorytmu dla macierzy symetrycznych i niesymetrycznych[%]



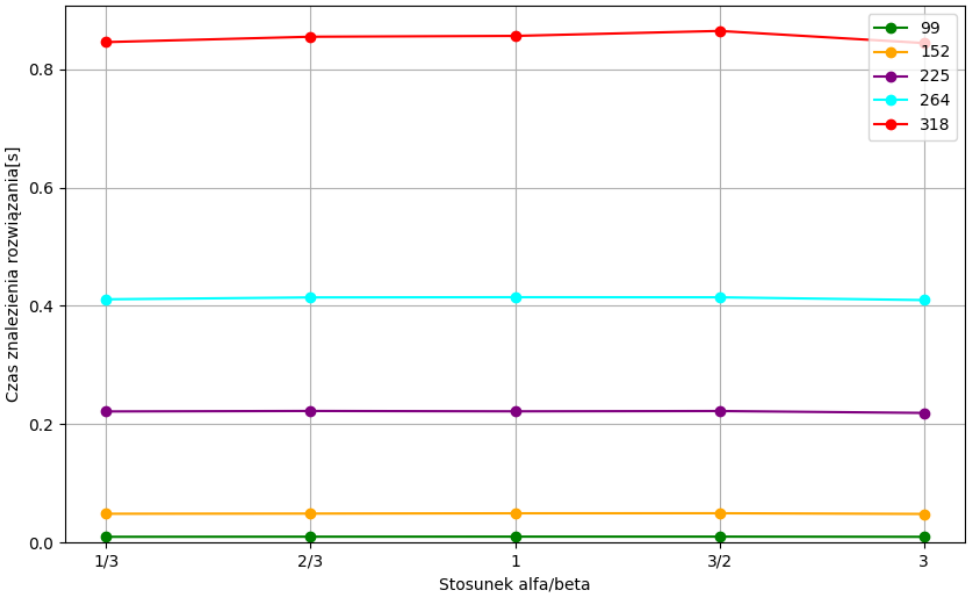
Rysunek 38: Wyniki badań dla macierzy symetrycznych



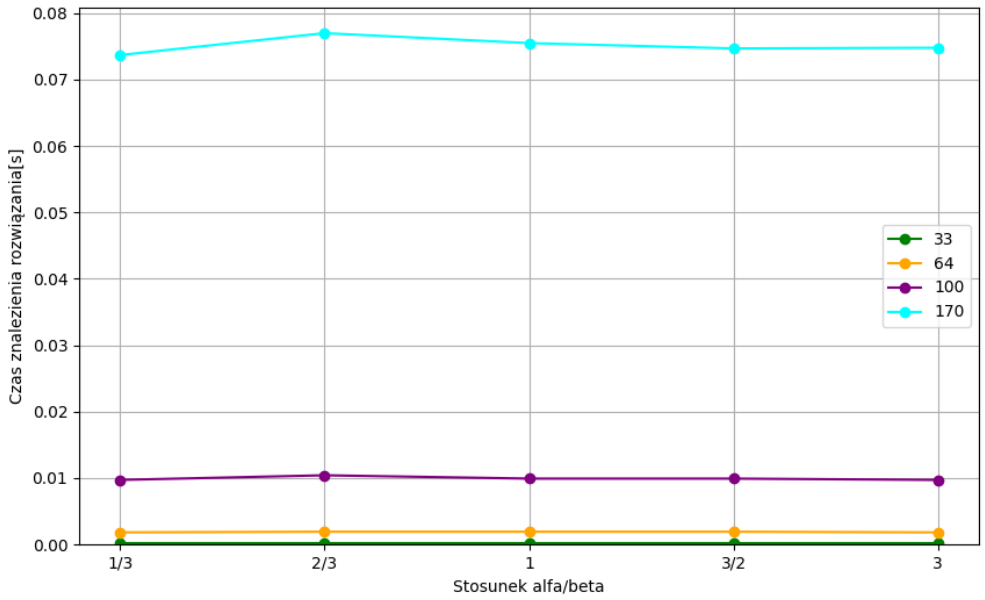
Rysunek 39: Wyniki badań dla macierzy asymetrycznych

	Symetryczne						Asymetryczne			
Rozmiar	99	152	225	264	318	439	33	64	100	170
1/3	0.009	0.048	0.221	0.411	0.846	3.006	0.000	0.002	0.010	0.074
2/3	0.009	0.049	0.222	0.414	0.855	3.037	0.000	0.002	0.010	0.077
1	0.010	0.049	0.222	0.414	0.856	3.038	0.000	0.002	0.010	0.075
3/2	0.010	0.049	0.222	0.414	0.865	3.024	0.000	0.002	0.010	0.075
3	0.009	0.048	0.219	0.410	0.844	3.019	0.000	0.002	0.010	0.075

Table 16: Czas znalezienia wyniku algorytmu dla macierzy symetrycznych i niesymetrycznych[s]



Rysunek 40: Wyniki badań dla macierzy symetrycznych



Rysunek 41: Wyniki badań dla macierzy asymetrycznych

Wnioski: Podobnie jak w przypadku badania 5.4 nie ma większych różnic dla różnych stosunków alfy do bety. Możliwe że podczas implementacji wzoru popełniłem błąd. Możliwe też że prawdopodobieństwo wyboru nie działa poprawnie. Podczas testów miałem z tym wiele problemów, obene rozwiązanie działa ale może mieć problemy.

5.5 Podsumowanie

Zaimplementowany algorytm wydaje się działać poprawnie ale występuje sporo problemów.

6 Wnioski z całego projektu

Poniżej przeprowadzono badanie dla małych tablic wykożystanych w poprzednim etapie. Dla algorytmu mrówkowego podano jedynie wartości przybliżone ponieważ algorytm wykonywał się zbyt szybko.

	Symetryczne					Asymetryczne			
Rozmiar	8	9	10	11	12	8	9	10	11
	0.500	0.001	0.001	0.500	0.500	0.000	0.000	0.001	0.500

Table 17: Czas znalezienia wyniku algorytmu dla macierzy symetrycznych i niesymetrycznych[s]

	Symetryczne					Asymetryczne			
Rozmiar	8	9	10	11	12	8	9	10	11
	0.002	0.013	0.014	0.014	0.014	0.001	0.013	0.007	0.014

Table 18: Czas znalezienia wyniku algorytmu dla macierzy symetrycznych i niesymetrycznych[s]

	Symetryczne					Asymetryczne			
Rozmiar	8	9	10	11	12	8	9	10	11
	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.001

Table 19: Czas znalezienia wyniku algorytmu dla macierzy symetrycznych i niesymetrycznych[s]

7 Źródła

1. https://www.javatpoint.com/what-is-a-tabu-search
2. https://www.geeksforgeeks.org/what-is-tabu-search/
3. https://www.baeldung.com/cs/tabu-search
4. http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/
5. https://eportal.pwr.edu.pl/pluginfile.php/209250/mod_resource/content/1/w8.pdf
6. https://www.geeksforgeeks.org/introduction-to-ant-colony-optimization/