ALGORYTM EWOLUCYJNY

SPRAWOZDANIE

AGNIESZKA MIECHOWICZ
242533

UWAGI WSTĘPNE:

- W pliku projektu pojawiła się dodatkowa klasa statyczna „EvolutionaryAlgorithmExperiments", która gromadzi metody ułatwiające testowanie poszczególnych parametrów i zapisywanie ich do pliku.

- Większość badań przeprowadzana jest metodą selekcji turniejowej, ponieważ daje ona znacząco lepsze rezultaty niż metoda ruletki. (Wyjątkiem jest, oczywiście, porównanie operatorów selekcji)

- Badania przeprowadzone zostały dla 5 problemów o różnych stopniach trudności: Berlin52, KroA100, KroA150, KroA200 i fl417. (O tym, że można badanie parametrów przeprowadzać tylko dla 3 problemów dowiedziałam się już po badaniach). W wielu przypadkach dodatkowe wykresy (np. czasu w zależności od parametru) pojawiają się właśnie dla tych 3 średnich problemów.

- Dla większości przypadków wzięłam pod uwagę 10 przebiegów algorytmu. (jeśli nie jest napisane inaczej)

- Domyślne wartości prawdopodobieństwa mutacji i krzyżowania to – odpowiednio- 0,2 oraz 0,8. Jeśli nie jest napisane inaczej, te wartości były wykorzystywane.

INFORMACJE O WYKORZYSTANYM ALGORYTMIE:

- Metoda krzyżowania: OX
- Metoda mutacji: Inwersja
- Metoda generowania pierwszej populacji: Wszystkie osobniki są całkowicie losowe.
- Wagi dla ruletki: $w(i) = (\max(F(j)) - F(i) + \varepsilon)$ * **$\text{avg}(F(j)) / F(i)$**

BADANIE ROZMIARU POPULACJI, LICZBY GENERACJI I ROZMIARU TURNIEJU

Cel: Zbadanie wpływu działania parametrów rozmiaru populacji, liczby pokoleń oraz rozmiaru turnieju na wynik działania algorytmu ewolucyjnego. Znalezienie optymalnych wartości dla każdego problemu.

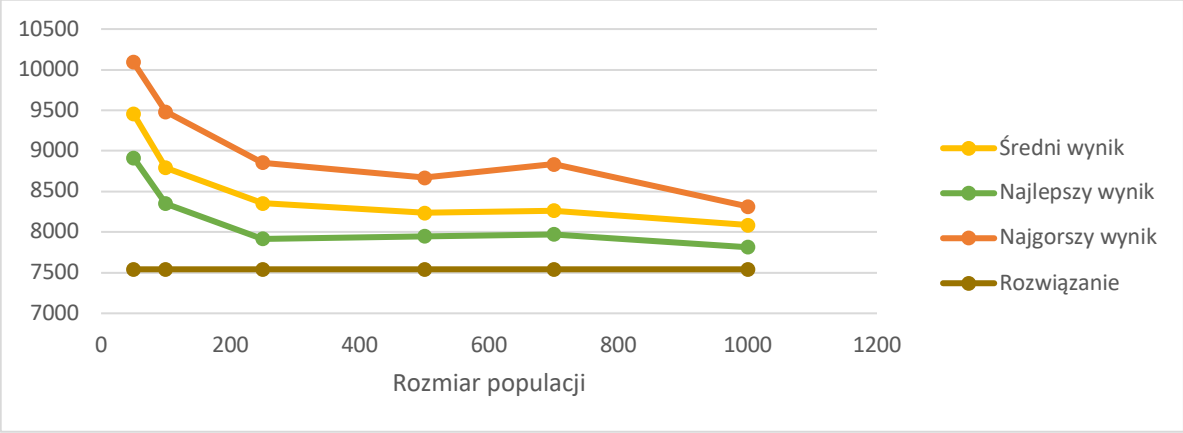
BADANIE WPŁYWU ROZMIARU POPULACJI

DLA PROBLEMU BERLIN52:

Liczba generacji: 500 Rozmiar turnieju: 25

Tabela 1 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku działania algorytmu dla 10 uruchomień w zależności od rozmiaru populacji dla problemu Berlin52

Rozmiar populacji	Średnia	Minimum	Maximum	Odchylenie standardowe
50	9456,462	8916,313	10091,024	361,1681798
100	8793,9704	8351,846	9482,889	378,261031
250	8355,2524	7916,3706	8856,592	320,1763645
500	8238,0199	7948,388	8671,217	259,5940904
700	8262,8102	7972,286	8836,932	276,297454
1000	8087,9426	7813,605	8316,317	171,8286016



Wykres 1 Zależność wyniku działania algorytmu od rozmiaru populacji dla problemu Berlin 52 i liczby generacji 250

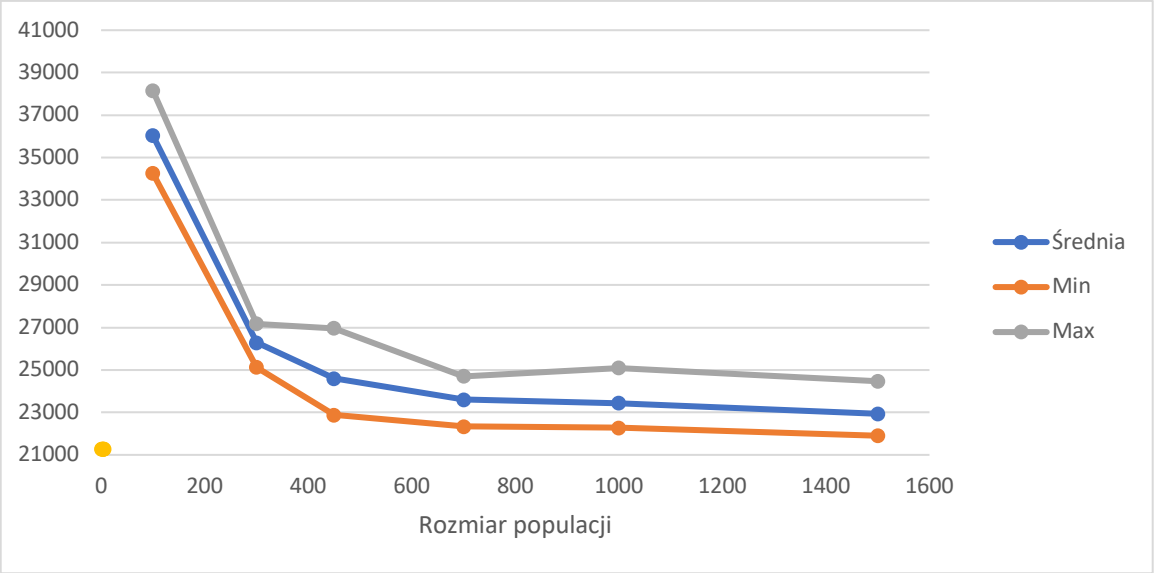
Wnioski:
Dla tego problemu z podanymi parametrami optymalna liczność populacji wynosi 250-500 osobników (dla większej populacji korzyść z jej rozmiaru może nie być warta zwiększenia czasu obliczeń). Warto zauważyć, że najlepszy uzyskany wynik różni się od optymalnego o około 300.

DLA PROBLEMU KROA100:

Liczba generacji: 450 Rozmiar turnieju: 25

Tabela 2 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku działania algorytmu dla 10 uruchomień w zależności od rozmiaru populacji dla problemu Kroa100

Liczność populacji	Średnia	Minimum	Maximum	Odchylenie stand.
100	36040,3138	34260,137	38153,88	1573,81944
300	26288,0503	25131,102	27168,758	827,344746
450	24590,2479	22879,332	26962,275	1186,18291
700	23602,7015	22334,559	24708,553	854,808094
1000	23434,0661	22278,578	25100,379	881,871272
1500	22935,2934	21901,686	24467,598	954,919733



Wykres 2 Zależność wyniku od rozmiaru populacji dla problemu KroA100 i liczby generacji 450

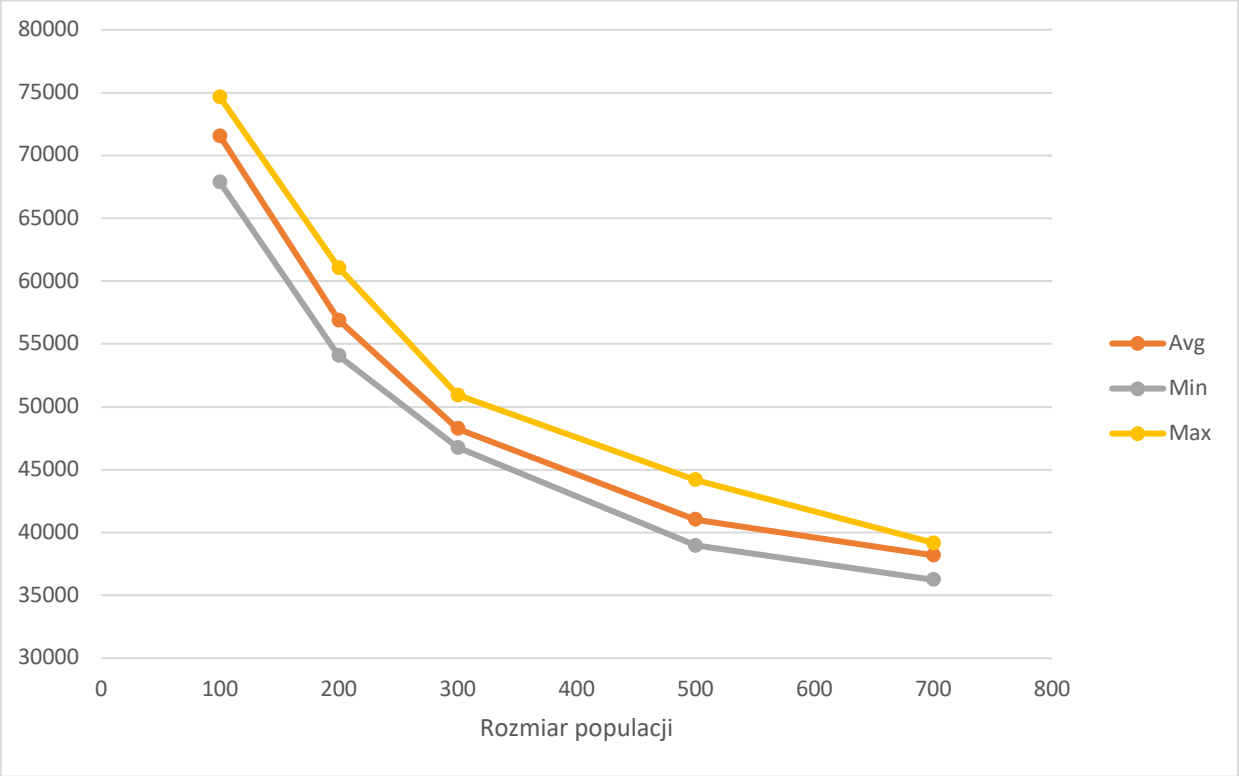
Wniosek:
100 osobników nie jest wystarczającym rozmiarem populacji i daje wyniki dalekie od pożądanych. Spłaszczenie wykresu, które w prostszym problemie obserwowaliśmy już dla wartości około 250, tutaj pojawia się dopiero w okolicach 500-700 osobników w populacji.

DLA PROBLEMU KROA150:

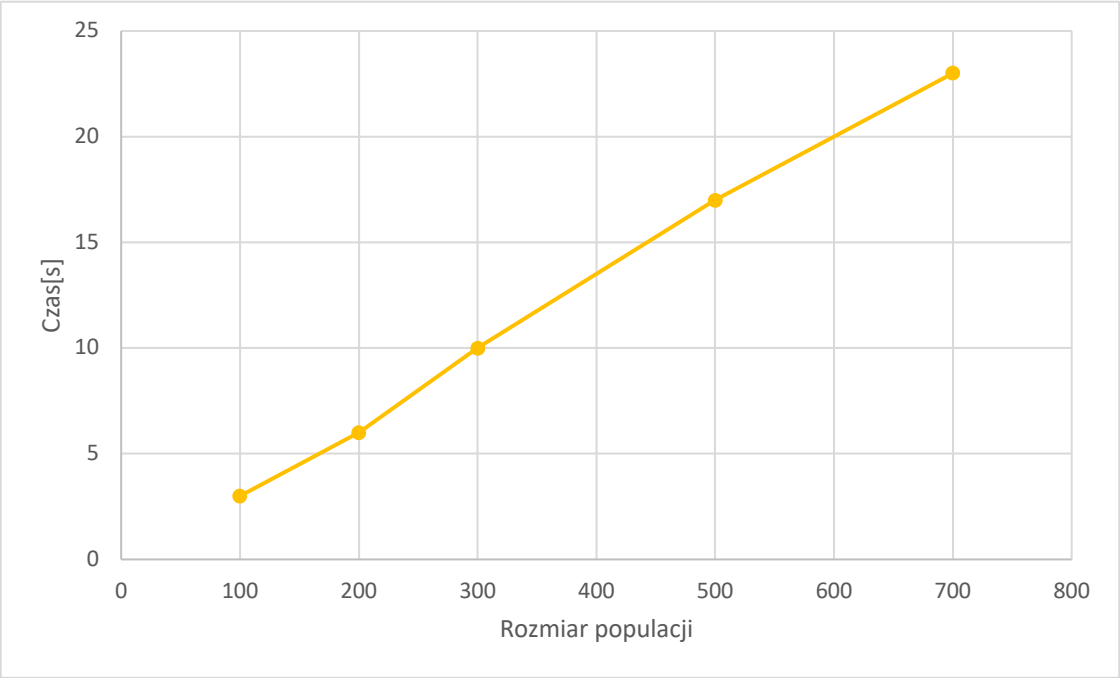
Liczba generacji: 300 Rozmiar turnieju:20

Tabela 3 Średni, minimalny, maksymalny wynik oraz czas działania algorytmu dla 10 uruchomień w zależności od rozmiaru populacji dla problemu Kroa150

POPSIZE	AVG	MIN	MAX	AVGTIME
100	71538,91	67859,164	74628,18	3
200	56898,562	54069,977	61078,14	6
300	48272,18	46757,246	50928,586	10
500	41036,336	38979,133	44191,35	17
700	38191,617	36241,027	39184,574	23



Wykres 3 Zależność wyniku od rozmiaru populacji dla problemu Kroa150 oraz liczby generacji 300.



Wykres 4 Zależność czasu od rozmiaru populacji dla problemu KroA150 i liczby generacji 300.

Wnioski:
Wykres pierwszy ma kształt podobny do pozostałych, choć tym razem ze względu na niskie wartości badane i małą liczbę generacji (w stosunku do problemu), wykres kończy się „wcześniej” – zanim jeszcze nastąpi jego pełne spłaszczenie. Pokazuje to kolejne przesunięcie wykresu dla większego problemu

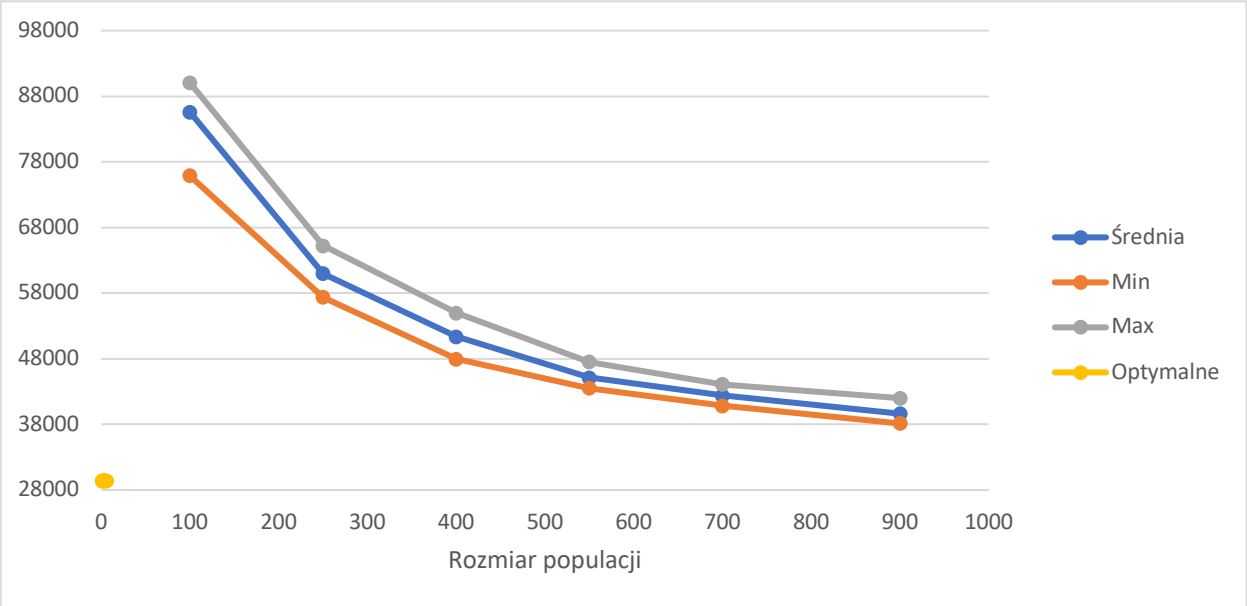
Na drugim wykresie możemy zauważyć liniową zależność między rozmiarem populacji a czasem jednokrotnego wykonania algorytmu.

DLA PROBLEMU KROA200:

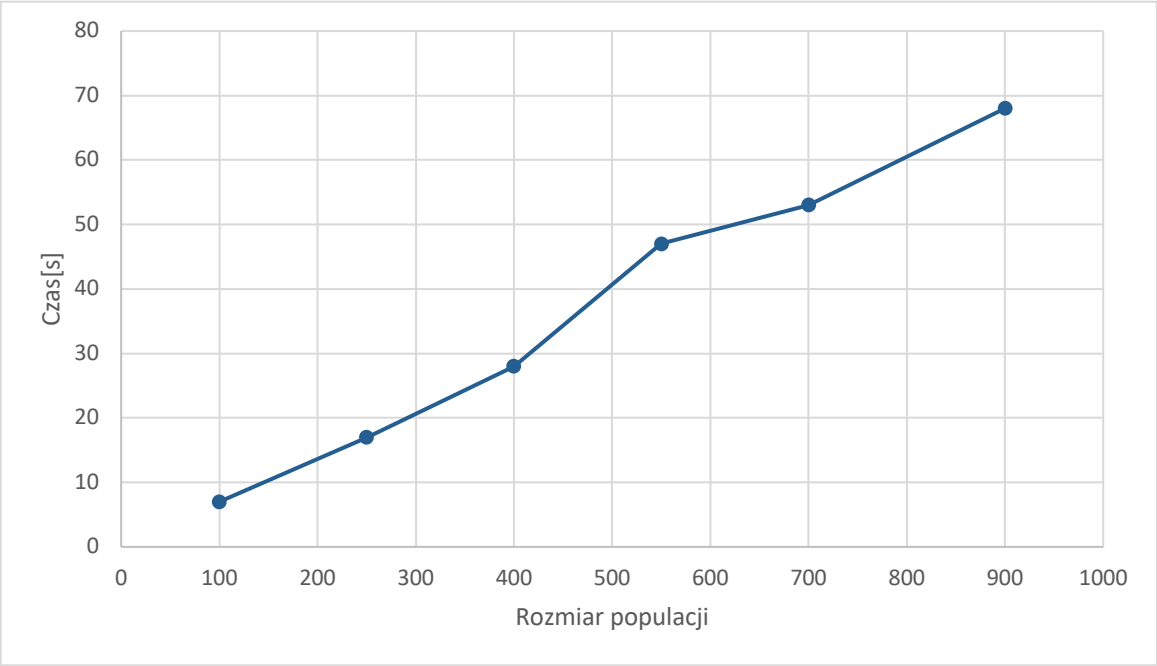
Liczba generacji: 500 Rozmiar turnieju: 20

Tabela 4 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od rozmiaru populacji dla problemu Kroa200

Rozmiar populacji	Avg	Min	Max	Odch. St.	Czas[s]
100	85597,75	75872,04	90065,336	4115,0337	7
250	60999,457	57370,76	65209,52	2433,382	17
400	51380,934	47979,52	54979,664	2060,3262	28
550	45121,945	43512,094	47517,406	1145,8037	47
700	42399,19	40832,797	44109,23	977,2646	53
900	39636,293	38136,473	42001,117	1334,1858	68



Wykres 5 Zależność wyniku od rozmiaru populacji dla problemu Kroa200 i liczby generacji 500.



Wykres 6 Zależność czasu od rozmiaru populacji dla problemu KroA200 i liczby generacji 500.

Wnioski:
Wykres pierwszy podobny jest kształtem do tych dla innych problemów – na początku wraz ze wzrostem liczby osobników w populacji szybko maleje, by następnie maleć coraz wolniej.
Trzeba wziąć pod uwagę również czas jednokrotnego wykonania algorytmu – znacząco rośnie on wraz ze wzrostem rozmiaru populacji.

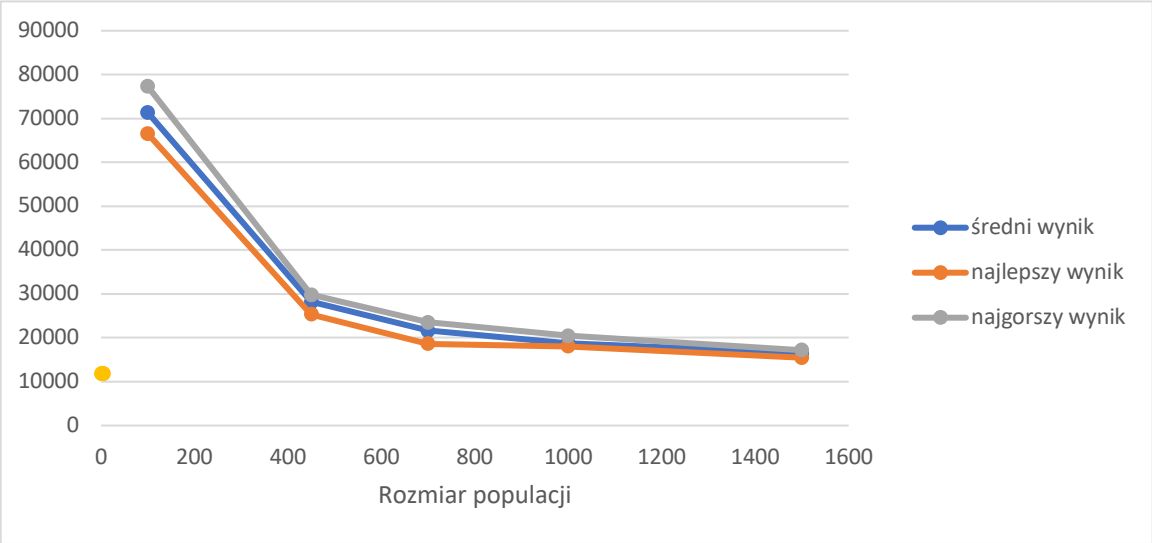
Wykres czasu od rozmiaru populacji dla tego problemu również przypomina zależności liniową, chociaż dla jednej wartości widać lekkie zakłócenie – prawdopodobnie związane z wykorzystaniem zasobów komputera przez inne programy podczas liczenia.

DLA PROBLEMU FL417:

Liczba generacji: 1000 Wielkość turnieju: 40

Tabela 5 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od rozmiaru populacji dla problemu Fl417

Liczność populacji	Średnia	Minimum	Maximum	Odchylenie s
100	36040,3138	34260,137	38153,88	1573,81944
300	26288,0503	25131,102	27168,758	827,344746
450	24590,2479	22879,332	26962,275	1186,18291
700	23602,7015	22334,559	24708,553	854,808094
1000	23434,0661	22278,578	25100,379	881,871272
1500	22935,2934	21901,686	24467,598	954,919733



Wykres 7 Zależność wyniku od rozmiaru populacji dla problemu fl417 i liczby generacji 1000.

Wnioski:
Po raz kolejny dla większego problemu obserwujemy zwiększenie się wymaganej liczby osobników. W tym problemie optymalny rozmiar populacji to około 700-1000 dla liczby generacji 1000.

Wniosek do badania liczności populacji:

Można zauważyć, że wraz **ze wzrostem poziomu trudności problemu, zwiększa się liczba osobników potrzebna**, aby go efektywnie rozwiązać.

Ponadto, **wykresy mają bardzo podobny kształt** – początkowo różnice pomiędzy kolejnymi rozmiarami populacji są bardzo duże, jednak w pewnym momencie wykres spłaszcza się i nie obserwuje się tak ogromnej poprawy wyników.

Nie obserwuje się również pogorszenia wyniku dla „zbyt dużych" rozmiarów populacji, natomiast wraz ze wzrostem rozmiaru populacji znacznie zwiększa się czas wykonania algorytmu, stąd konieczne jest dobranie najmniejszego rozmiaru populacji, który daje zadawalające wyniki.

Czas zwiększa się proporcjonalnie do rozmiaru populacji.

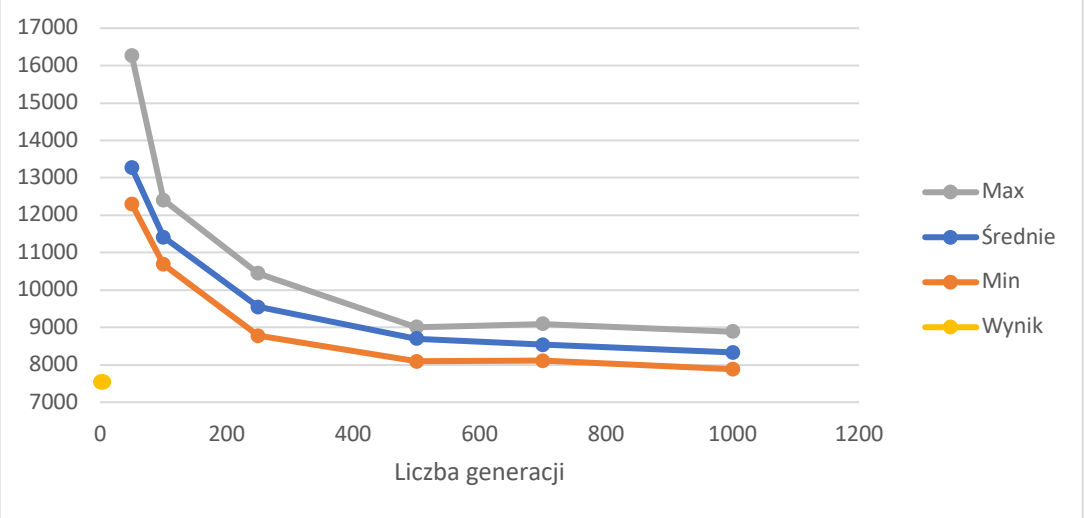
BADANIE WPŁYWU LICZBY GENERACJI

DLA PROBLEMU BERLIN52

Rozmiar populacji: 250 Rozmiar turnieju: 20
Prawdopodobieństwo krzyżowania: 0,7 Prawdopodobieństwo mutacji: 0,1

Tabela 6 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku działania algorytmu dla 10 uruchomień w zależności od liczby generacji dla problemu Berlin52

Liczba generacji	Średnia	Minimum	Maximum	Odchylenie standardowe
50	13273,0159	12301,952	16273,855	1231,876416
100	11407,8574	10680,889	12407,367	542,7687408
250	9545,3482	8780,054	10443,504	447,0107311
500	8700,03866	8093,6006	9007,809	321,4823745
700	8535,2016	8107,8345	9089,568	279,8668796
1000	8331,05024	7882,354	8883,221	337,8292853



Wykres 8 Zależność wyniku od liczby generacji dla problemu Berlin52 i rozmiaru populacji 250

Wnioski:

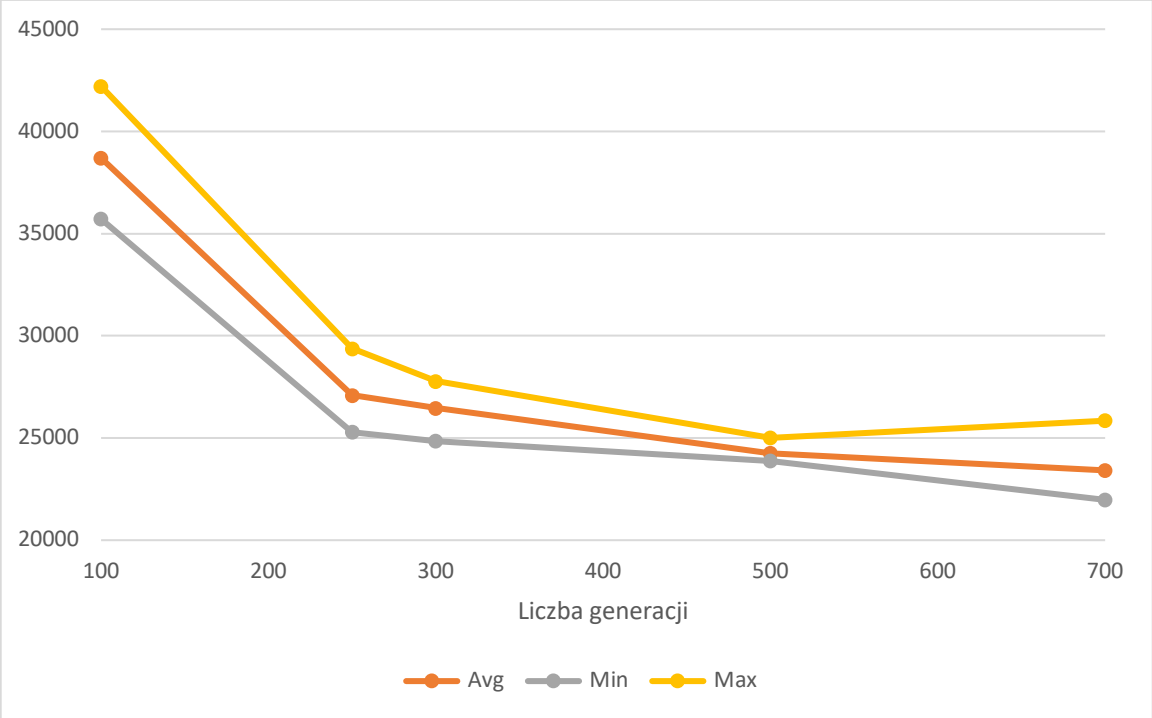
Tak samo jak w przypadku rozmiaru populacji, wynik maleje wraz ze wzrostem liczby generacji. Dla tego problemu oraz rozmiaru populacji spłaszcza się dla wartości około 500 – od tego momentu wyniki różnią się od siebie nieznacznie.

DLA PROBLEMU KROA100:

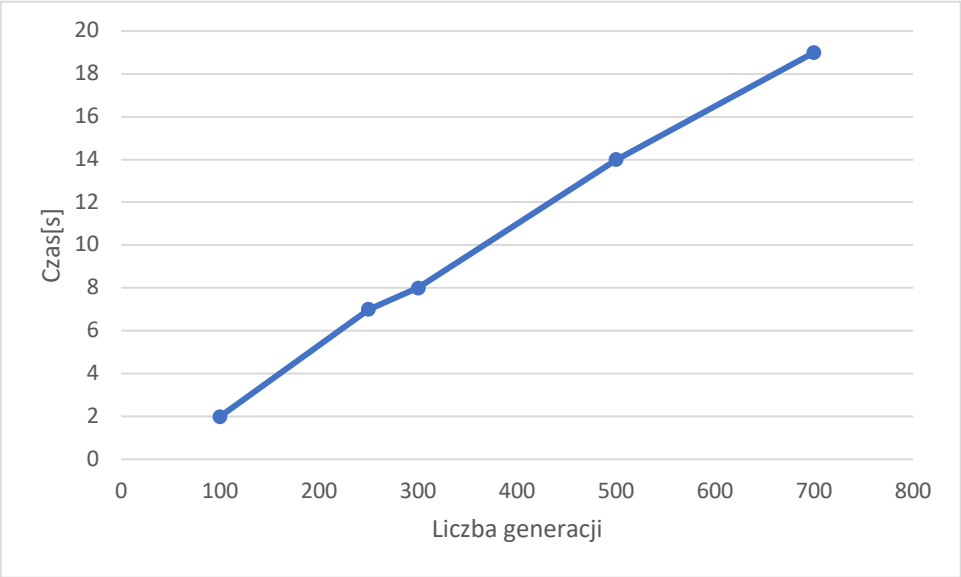
Rozmiar populacji: 500 Rozmiar turnieju: 25
Prawdopodobieństwo krzyżowania: 0,7 Prawdopodobieństwo mutacji: 0,1

Tabela 7 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od liczby generacji dla problemu Kroa100

LICZBA GENERACJI	AVG	MIN	MAX	STAND.DEV	AVGTIME
100	38697,33	35710,344	42208,55	2081,2476	2
250	27084,266	25297,195	29372,988	1347,7937	7
300	26459,832	24851,111	27777,467	977,81934	8
500	24249,598	23888,598	25006,762	414,8178	14
700	23415,934	21972,984	25851,162	1032,9315	19



Wykres 9 Zależność wyniku od liczby generacji dla problemu KroA100 i rozmiaru populacji 500



Wykres 10 Zależność czasu jednokrotnego wykonania algorytmu od liczby generacji dla problemu KroA100 i rozmiaru populacji 500

Wnioski:

Pierwszy z wykresów pokazuje, że wraz ze wzrostem liczby generacji, maleje wynik działania algorytmu. Dla mniejszych wartości maleje on szybciej, a dla większych wolniej.

Drugi wykres pokazuje ważne ograniczenie – przy zwiększaniu liczby generacji czas wykonania problemu rośnie liniowo.

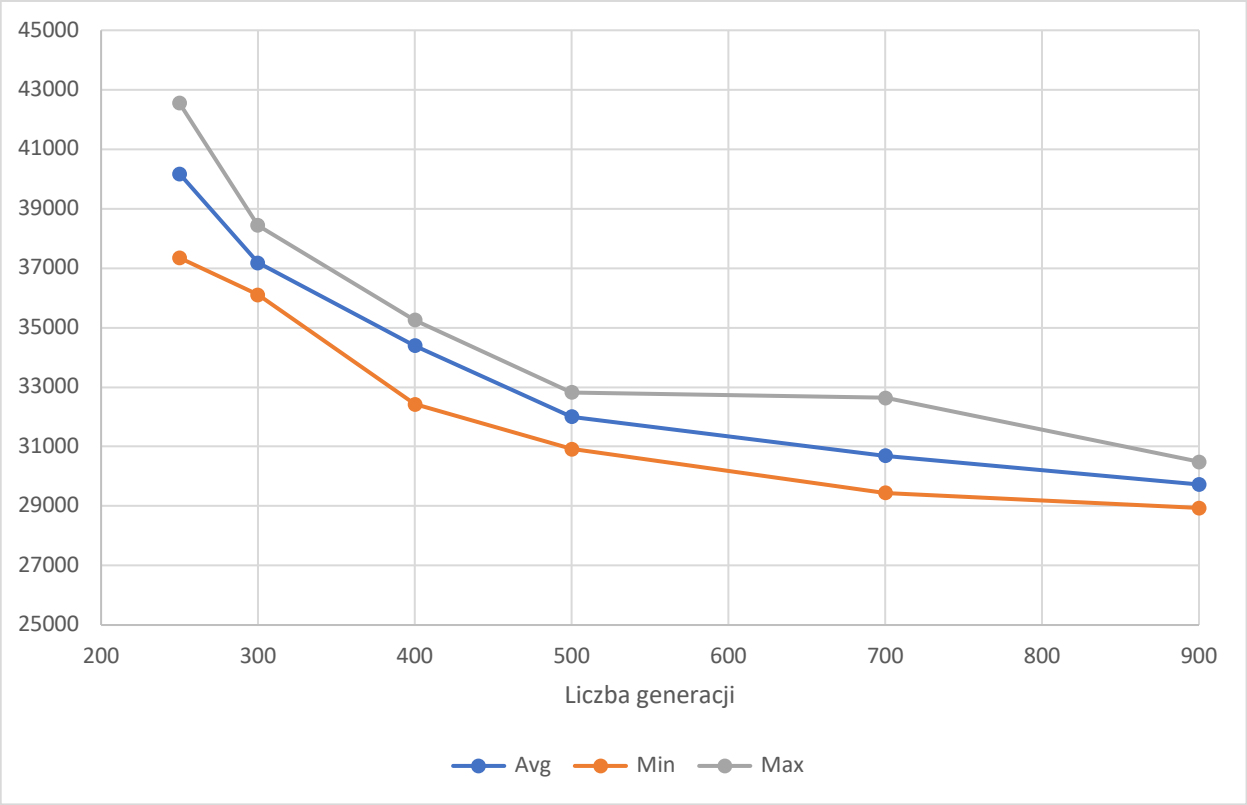
Wybranie odpowiedniej liczby generacji jest więc kompromisem pomiędzy uzyskaniem lepszego wyniku a czasem działania algorytmu.

DLA PROBLEMU KROA150

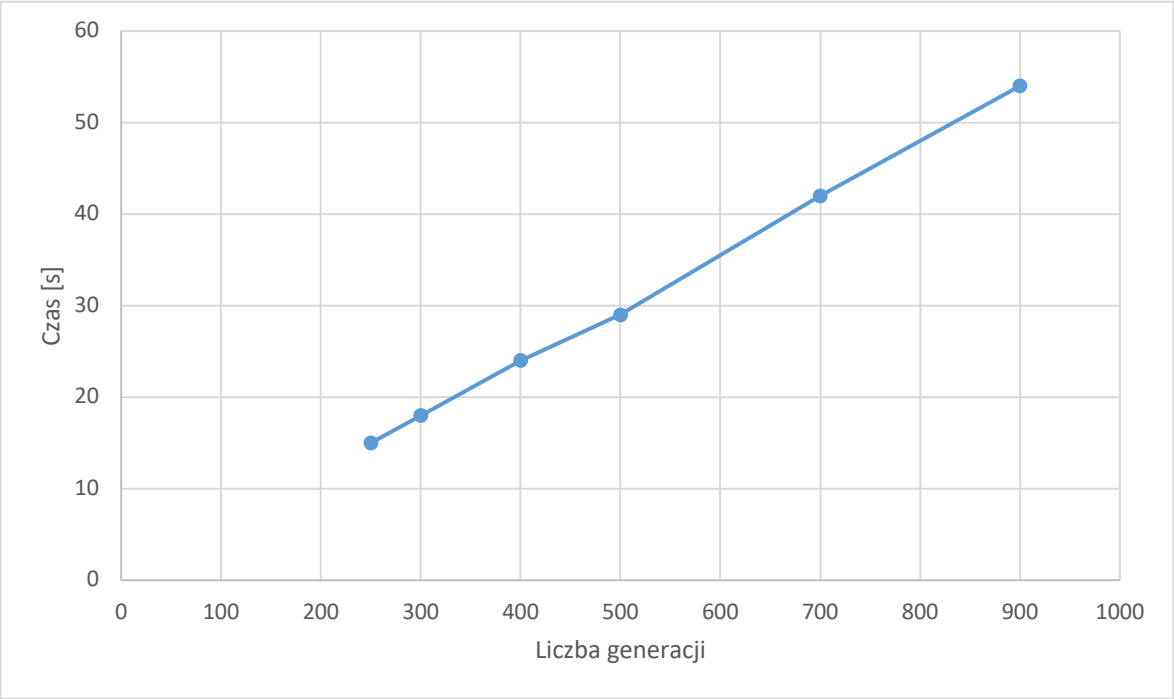
Rozmiar populacji: 700 Rozmiar turnieju: 25
Prawdopodobieństwo krzyżowania: 0,7 Prawdopodobieństwo mutacji: 0,1

Tabela 8 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od liczby generacji dla problemu Kroa150

LICZBA GENERACJI	AVG	MIN	MAX	STAND.DEV	AVGTIME
250	40176,9	37346,906	42546,91	1800,335	15
300	37179,383	36102,598	38445,734	719,0368	18
400	34392,594	32421,639	35260,273	808,2376	24
500	32000,152	30923,51	32824,59	659,8673	29
700	30693,105	29444,396	32644,4	993,92865	42
900	29725,793	28936,643	30492,379	455,40405	54



Wykres 11 Zależność wyniku od liczby generacji dla problemu KroA150 i rozmiaru populacji 700



Wykres 12 Zależność czasu jednokrotnego wykonania algorytmu od liczby generacji dla problemu KroA150 i rozmiaru populacji 700

Wnioski:
Wykres pierwszy pokazuje, że tak jak w pozostałych problemach, im większa liczba generacji, tym lepszy wynik. Początkowo poprawia się on szybciej w stosunku do liczby generacji, by następnie zmieniać się coraz mniej.

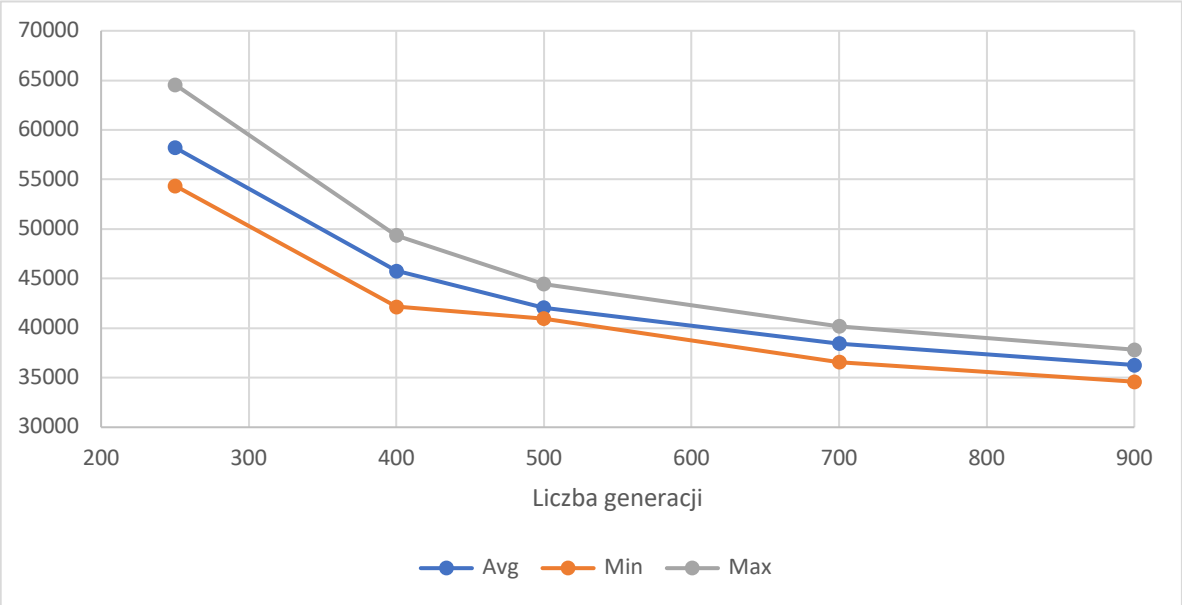
Tak samo jak w poprzednim problemie, przyrost czasu wykonania algorytmu jest proporcjonalny do liczby generacji.

DLA PROBLEMU KROA 200

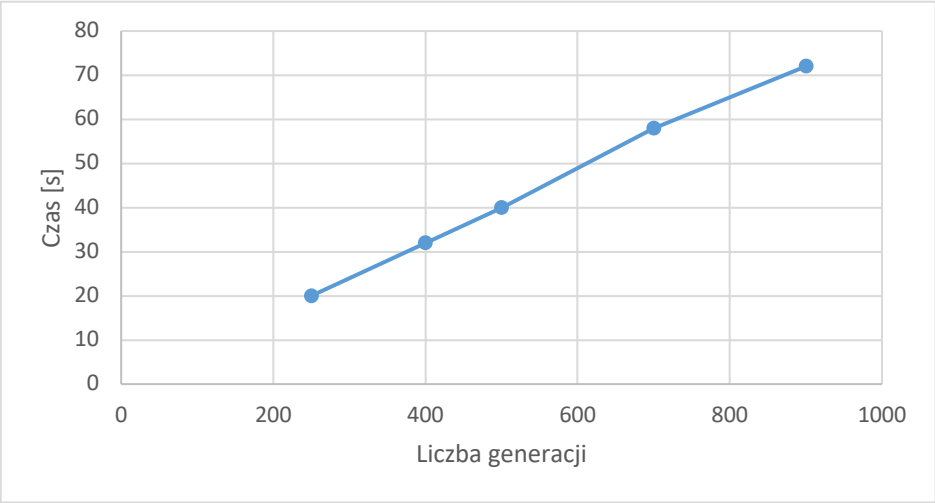
Rozmiar populacji: 700 Rozmiar turnieju: 25
Prawdopodobieństwo krzyżowania: 0,7 Prawdopodobieństwo mutacji: 0,1

Tabela 9 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od liczby generacji dla problemu Kroa200

LICZBA GENERACJI	AVG	MIN	MAX	STAND.DEV	AVGTIME
250	58222,086	54380,164	64585,95	2954,9429	20
400	45772,395	42148,82	49364,008	2049,1233	32
500	42069,992	40956,453	44454,73	1140,122	40
700	38465,13	36590,16	40186,066	970,61224	58
900	36280,49	34597,754	37825,258	993,3092	72



Wykres 13 Zależność wyniku od liczby generacji dla problemu KroA200 i rozmiaru populacji 700



Wykres 14 Zależność czasu jednokrotnego wykonania algorytmu od liczby generacji dla problemu KroA200 i rozmiaru populacji 700

Wnioski:

Tak jak przy poprzednich problemach, zwiększająca się liczba generacji powoduje poprawę wyniku, początkowo gwałtowną, a następnie wolniejszą.

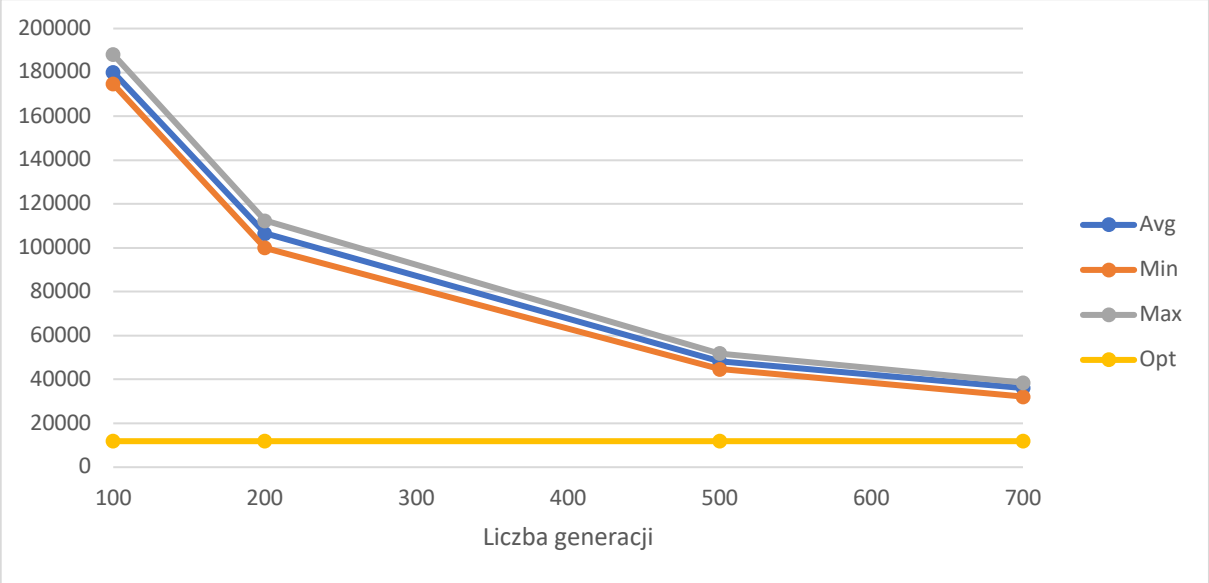
Ograniczeniem, tak jak w poprzednich problemach, jest czas – przyrasta proporcjonalnie do liczby pokoleń.

DLA PROBLEMU FL417:

Rozmiar populacji: 400 Rozmiar turnieju: 20
Prawdopodobieństwo krzyżowania: 0,7 Prawdopodobieństwo mutacji: 0,1

Tabela 10 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od liczby generacji dla problemu Fl417

LICZBA GENERACJI	AVG	MIN	MAX	STAND.DEV	AVGTIME
100	180071,6	174663,05	188170,98	3972,9448	10
200	106540,2	100000,5	112479,266	3463,0085	20
500	48270,023	44735,816	51832,79	2285,9866	50
700	36089,746	32194,95	38594,02	1702,057	68



Wykres 15 Zależność wyniku od liczby generacji dla problemu Fl417 i rozmiaru populacji 1000

Wnioski:
Tak jak przy poprzednich problemach możemy zaobserwować zarówno poprawę wyniku w zależności od liczby generacji, jak i proporcjonalnie przyrastający czas wykonania algorytmu.

Wnioski do badania liczby generacji:

Im większa liczba generacji, tym lepsze rozwiązanie problemu.
Nie obserwuje się gorszego działania dla „zbyt dużych" wartości liczby generacji.

Największym problemem tego parametru jest optymalizacja czasu – czas przyrasta proporcjonalnie do liczby generacji.

Znalezienie optymalnej liczby generacji jest szukaniem kompromisu pomiędzy wynikiem działania algorytmu a czasem jego wykonania.

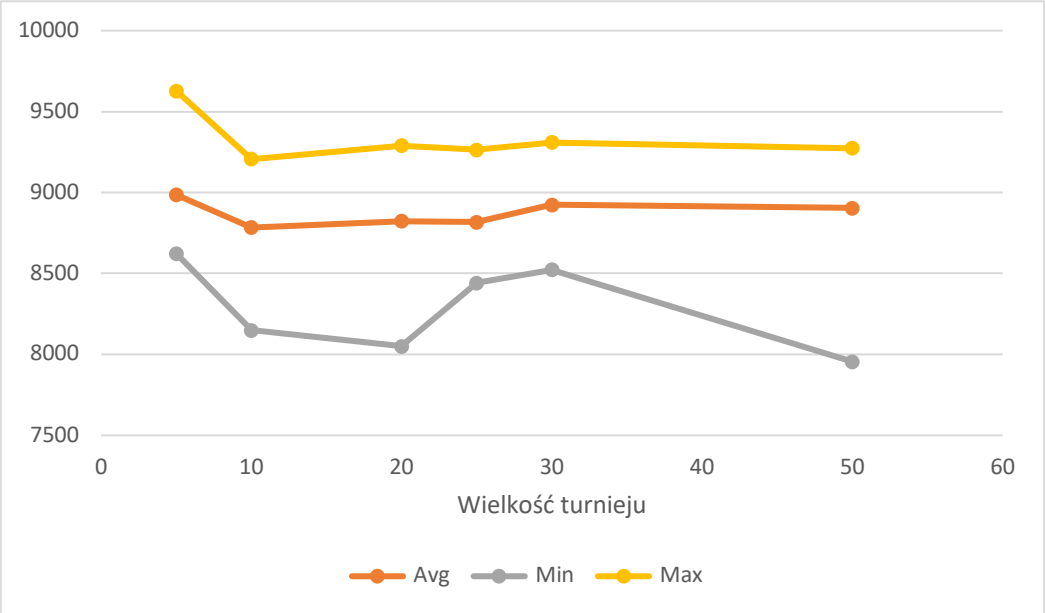
BADANIE WPŁYWU WIELKOŚCI TURNIEJU

DLA PROBLEMU BERLIN52

Rozmiar populacji: 250 Liczba generacji: 250

Tabela 11 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od rozmiaru turnieju dla problemu Berlin52

TNM SIZE	AVG	MIN	MAX	STAND.DEV	TIME[S]
5	8985,584	8623,181	9628,279	320,42178	0
10	8783,448	8149,602	9205,685	291,37463	0
20	8822,811	8049,676	9288,961	362,5496	1
25	8817,521	8440,915	9263,41	299,3895	1
30	8923,399	8522,344	9309,305	277,52768	1
50	8903,49	7955,2783	9273,675	352,52542	2



Wykres 16 Zależność wyniku od wielkości turnieju dla problemu Berlin52, rozmiaru populacji 250 oraz liczby generacji 250

Wnioski:
Średni oraz maksymalny(najgorszy) wynik nie zmieniają się zanadto w zależności od wielkości turnieju, jedynie dla zbyt niskich wielkości są one wysokie.

Wielkość turnieju osiąga w przypadku tego problemu minimum lokalne w granicach 20.

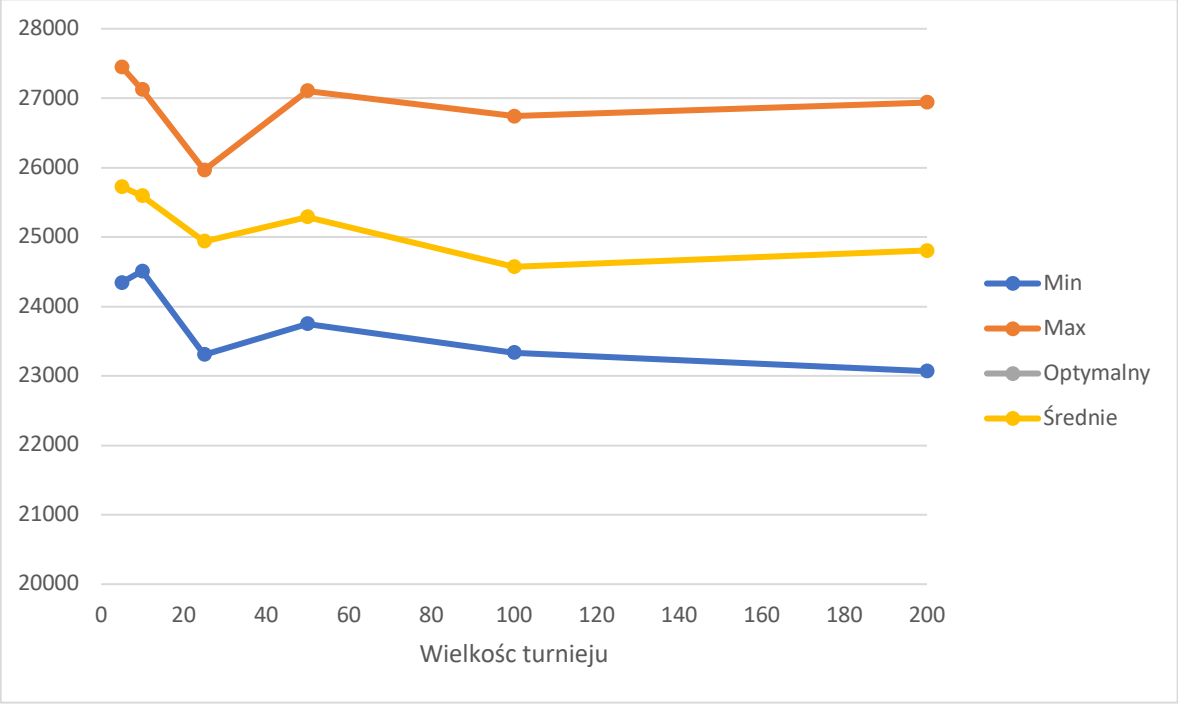
Badanie wykazało też zależność czasu wykonania od rozmiaru turnieju.

DLA PROBLEMU KROA100

Rozmiar populacji: 400 Liczba generacji: 400

Tabela 12 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku algorytmu dla 10 uruchomień w zależności od rozmiaru turnieju dla problemu Kroa100

ROZMIAR TURNIEJU	ŚREDNIA	MINIMUM	MAXIMUM	ODCH.STAND.
2	50677,0697	31398,746	84437,41	21133,8103
5	25727,8083	24342,893	27450,578	911,85384
10	25594,4629	24510,418	27124,693	800,243856
25	24939,8655	23307,523	25964,338	908,236578
50	25291,0438	23750,715	27105,273	980,639136
100	24571,6197	23335,578	26739,078	970,790872
200	24807,2596	23067,838	26939,334	1154,30084



Wykres 17 Zależność wyniku od wielkości turnieju dla problemu KroA100, rozmiaru populacji 400 oraz liczby generacji 400.

Wnioski:

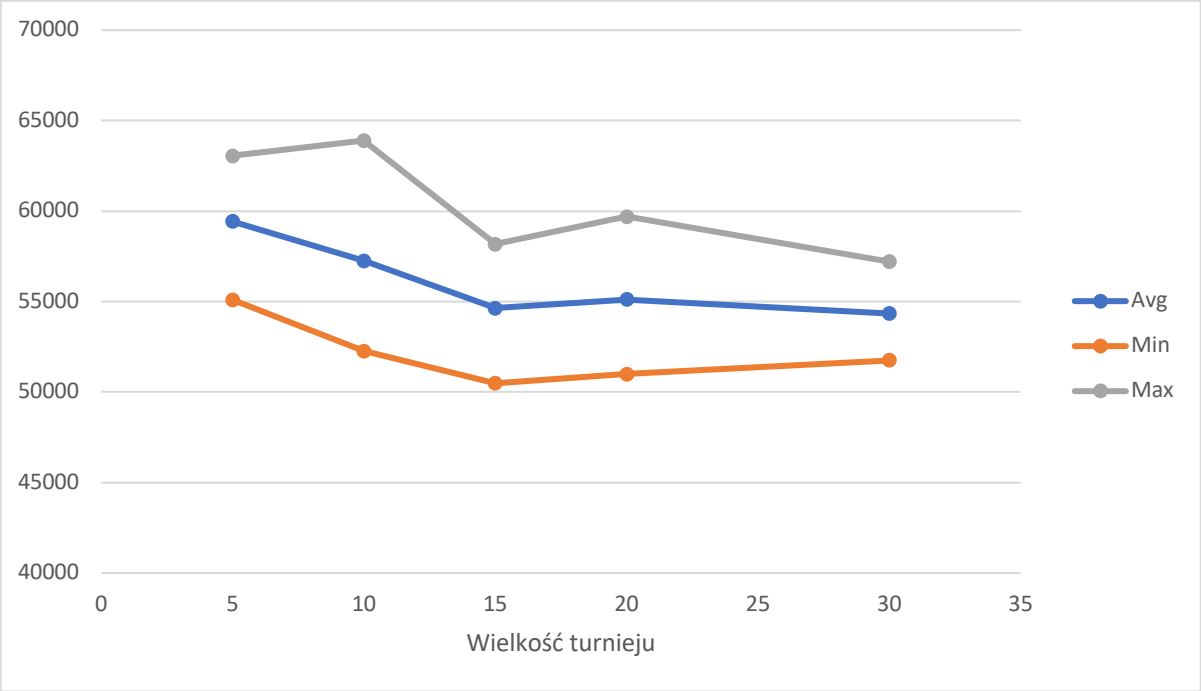
Turniej osiąga swoje minimum lokalne dla 25, a następnie wynik zwiększa się, by wraz ze wzrostem wielkości turnieju powoli opadać.
Ze względu na zależność czasu od wielkości turnieju, przyjmuję 25 za optymalną wartość dla tego problemu.

DLA PROBLEMU KROA150

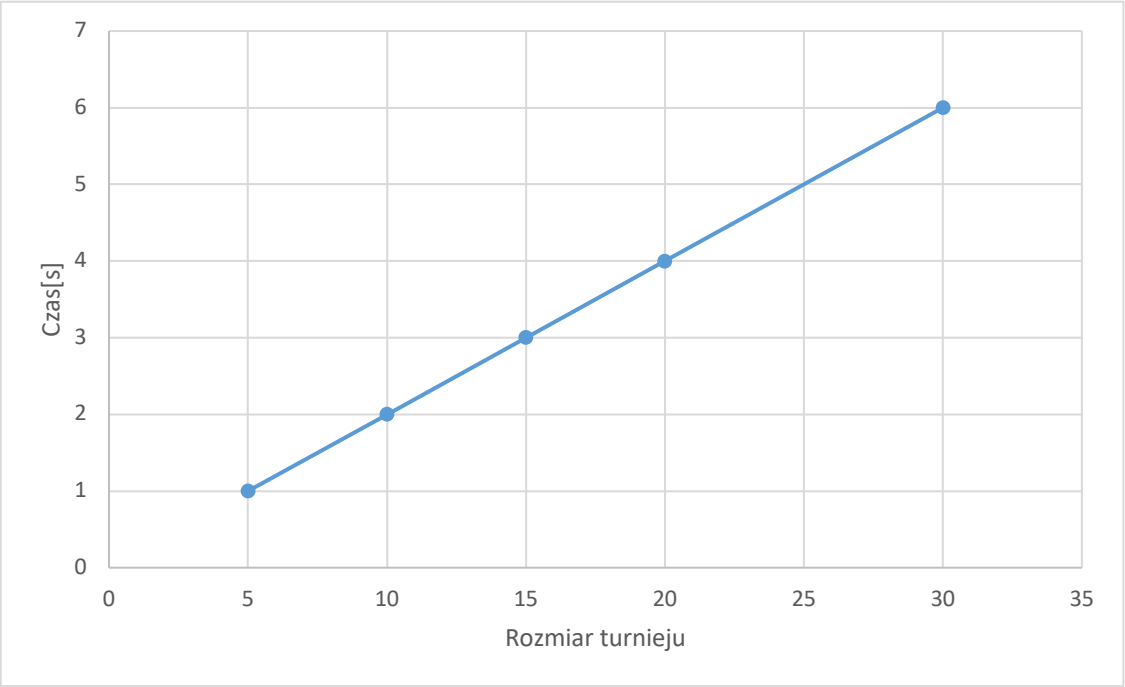
Rozmiar populacji: 300 Liczba generacji: 300

Tabela 13 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od rozmiaru turnieju dla problemu KroA150

ROZMIAR TURNIEJU	AVG	MIN	MAX	STAND.DEV	AVGTIME
5	59425,22	55101,508	63050,316	2464,4653	1
10	57259,387	52261,48	63899,62	3021,7598	2
15	54633,633	50485,67	58178,156	2058,53	3
20	55110,75	51002,97	59696,074	2407,468	4
30	54340,72	51750,32	57207,21	1746,5751	6



Wykres 18 Zależność wyniku od wielkości turnieju dla problemu KroA150, rozmiaru populacji 400 oraz liczby generacji 400



Wykres 19 Zależność czasu od wielkości turnieju dla problemu KroA150 oraz rozmiaru populacji 300 i liczności generacji 300.

Wnioski:
Po raz kolejny można zauważyć, że wynik początkowo maleje wraz ze wzrostem wielkości turnieju, by następnie osiągnąć pewne minimum lokalne (tutaj około 15-20).

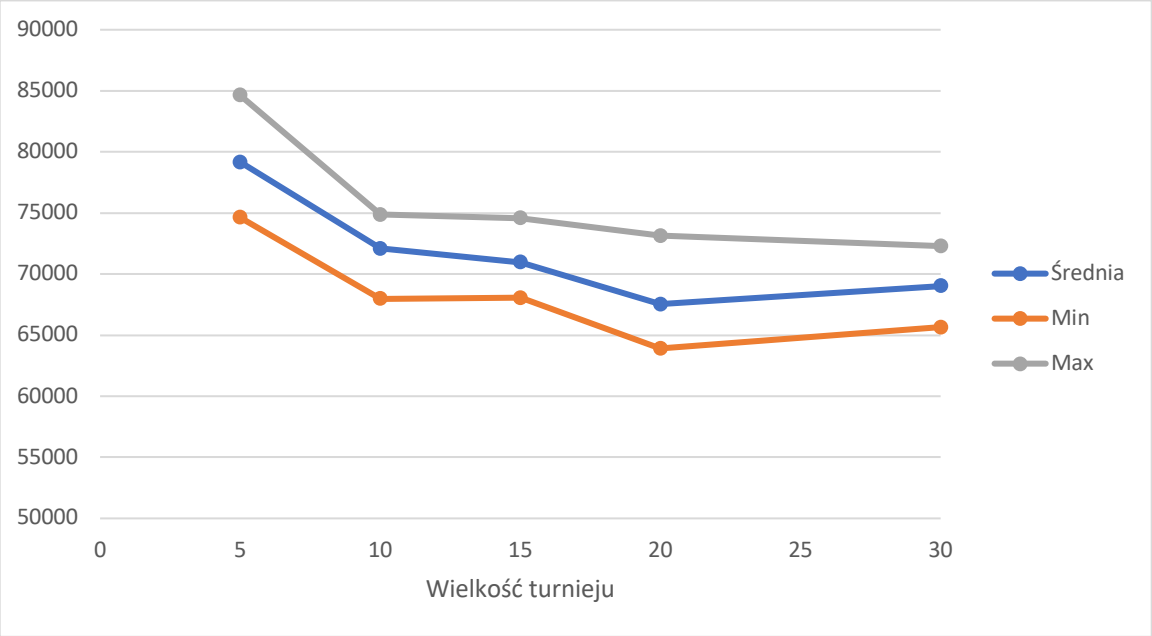
Widzimy też liniową zależność czasu od wielkości turnieju.

DLA PROBLEMU KROA 200:

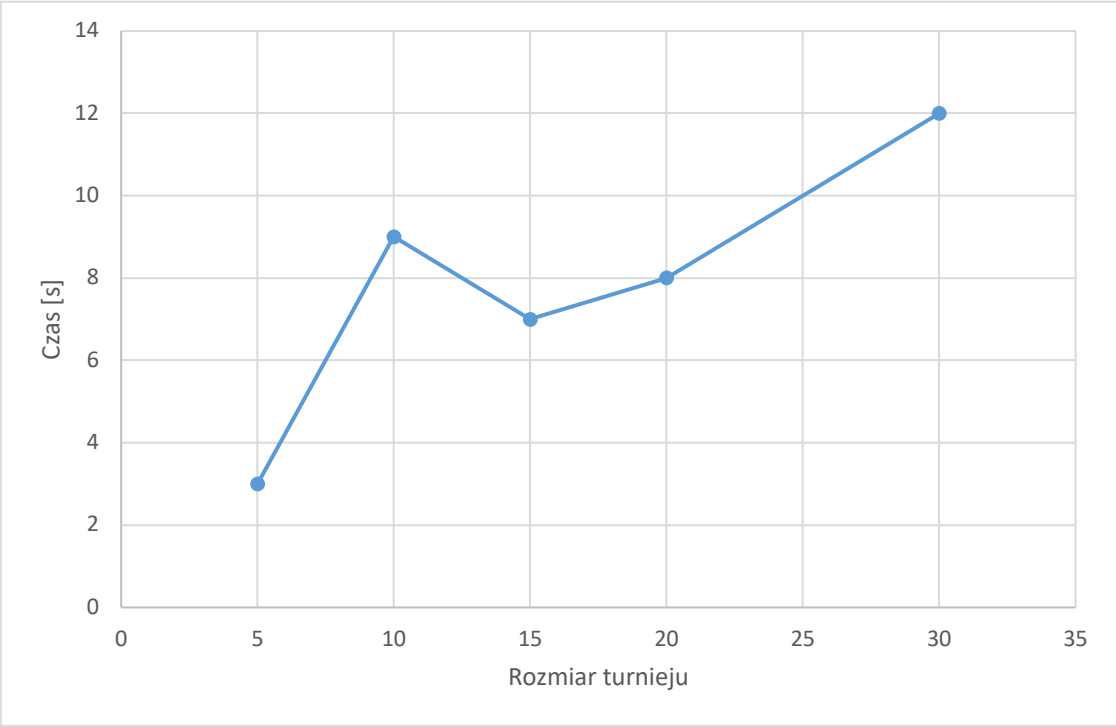
Rozmiar populacji: 550 Liczba generacji: 550

Tabela 14 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od rozmiaru turnieju dla problemu KroA200

ROZMIAR TURNIEJU	AVG	MIN	MAX	STAND.DEV	AVGTIME
5	79170,1	74627,54	84629,35	3405,2446	3
10	72088,484	67967,8	74864,54	1957,7925	9
15	70962,06	68046,83	74568,91	2085,2766	7
20	67543,95	63913,332	73134,13	2824,6838	8
30	69032,52	65656,46	72291,21	2240,4587	12



Wykres 20 Zależność wyniku od wielkości turnieju dla problemu KroA200, rozmiaru populacji 550 oraz liczby generacji 550



Wykres 21 Zależność czasu od wielkości turnieju dla problemu KroA200 oraz rozmiaru populacji 500 i liczności generacji 500.

Wnioski:

Dla zbyt małych (poniżej 15) rozmiarów turnieju obserwuje się wyraźnie gorsze wyniki. W dalszej części funkcji, zmienia się ona bardzo powoli. W okolicach 20 znajduje się minimum lokalne, dalej funkcja bardzo nieznacznie rośnie.

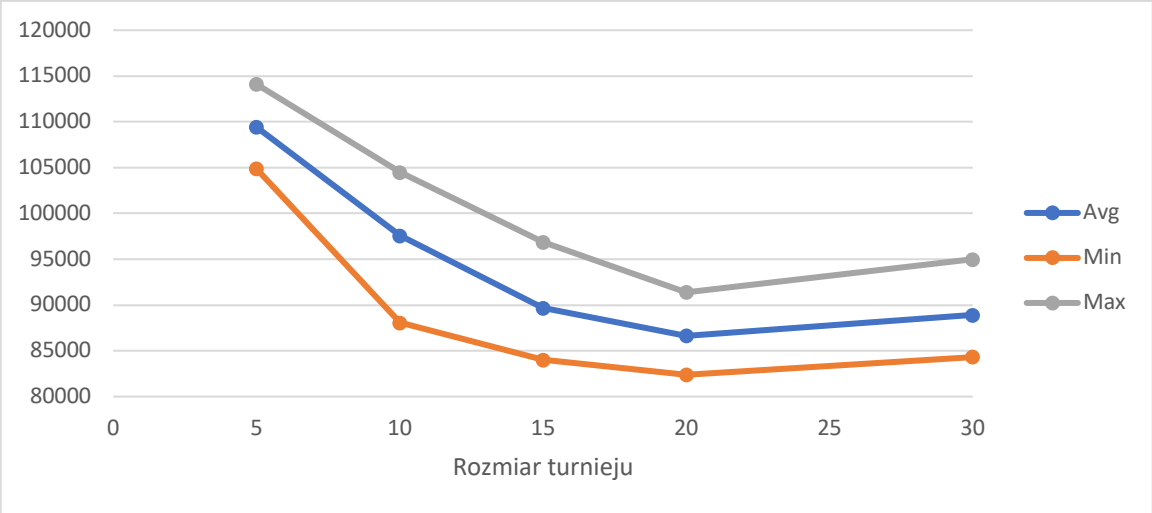
Obserwuje się też po raz kolejny znaczny wzrost czasu wykonania wraz ze zwiększaniem rozmiaru turnieju.

DLA PROBLEMU FL417:

Rozmiar populacji: 300 Liczba generacji: 300

Tabela 15 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od rozmiaru turnieju dla problemu Fl417

ROZMIAR TURNIEJU	AVG	MIN	MAX	STAND.DEV	AVGTIME
5	109420,3	104859,766	114102,234	3260,4036	8
10	97550,75	88048,08	104480,8	4660,0796	11
15	89685,69	84012,52	96863,79	3476,6929	14
20	86628,46	82383,19	91382,3	2606,4783	18
30	88900,3	84310,47	94995,78	2869,9778	26



Wykres 22 Zależność wyniku od wielkości turnieju dla problemu Fl417, rozmiaru populacji 300 oraz liczby generacji 300

Wnioski do badania rozmiaru turnieju:

Na większości wykresów można zauważyć pewne minimum występujące w okolicach wartości 15-25. W dalszych częściach zazwyczaj wartość funkcji nie zmienia się gwałtownie, czasami jest nieco lepsza, jednak równocześnie większy rozmiar turnieju to ogromne koszty czasowe. Czas wykonania algorytmu zmienia się proporcjonalnie do zmiany rozmiaru turnieju.

Warto zwrócić uwagę, że dla każdego problemu widać było, że rozmiar turnieju poniżej 15 daje znacząco gorsze wyniki niż większe rozmiary turnieju – dzieje się tak dlatego, że w takim ustawieniu algorytmu metoda selekcji zbliża się do losowej – np. dla wartości 1 jest całkowicie losowa.

Podsumowanie badania wpływu rozmiaru populacji, liczby generacji oraz wielkości turnieju

Każdy z tych parametrów w znaczący sposób **wpływa na czas** wykonania algorytmu- czas wykonania algorytmu jest proporcjonalny do każdego z tych trzech parametrów.

Rozmiar populacji i liczba generacji potrzebne do efektywnego rozwiązania problemu **rosną wraz ze stopniem jego trudności** – dla najprostszego wystarczą po około 250, a dla najtrudniejszego nawet ponad 1000.

W przypadku rozmiaru populacji oraz liczby generacji nie obserwuje się pogorszenia wyniku dla „zbyt dużej" wartości – przeciwnie, wynik jest coraz lepszy przy coraz większych wartościach rozmiaru populacji oraz liczby generacji. Głównym ograniczeniem jest więc czas obliczeń.
Dobranie optymalnych parametrów to znalezienie kompromisu pomiędzy wynikiem a czasem wykonania algorytmu.

Niełatwo jest zbadać wpływ poszczególnych parametrów (szczególnie w przypadku pierwszych dwóch), ponieważ są one bardzo zależne od siebie. Dodatkowo, nawet przy dużym rozmiarze populacji, jeśli liczba generacji jest zbyt mała, algorytm nie osiągnie dobrych wyników i odwrotnie.

Zwiększenie rozmiaru turnieju nie powoduje natomiast zawsze poprawy wyniku. Ważne jest, by rozmiar ten nie był zbyt mały (poniżej 10), gdyż wtedy selekcja staje się zbyt losowa.

Nie zaobserwowałam też wpływu trudności problemu na rozmiar turnieju. Być może rozmiar populacji ma w nim znaczenie.

Dla zbadanych problemów, rozmiarów populacji i liczności generacji jego optymalna wartość była najczęściej w okolicy 15-25 – większe wartości albo dawały podobnie/nieznacznie gorsze/lepsze wyniki, lecz wymagały zdecydowanie więcej czasu.

PORÓWNANIE OPERATORÓW SELEKCJI

Cel: Zbadanie wpływu zastosowanego operatora selekcji na działanie algorytmu ewolucyjnego. Pod uwagę brane są dwa operatory: ruletka oraz turniej.

DLA PROBLEMU BERLIN52:

Tabela 16 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od metody selekcji dla problemu Berlin52 i parametrów podanych poniżej.

METODA	AVG	MIN	MAX	STAND.DEV	AVGTIME
TOURNAMENT	8247,322	7804,5566	8504,146	233,27281	2
ROULETTE	9225,724	8545,067	10171,48	561,667	0
TOURNAMENT	8362,94	7782,7114	8739,249	340,6745	0
ROULETTE	10043,5	9046,511	10562,22	434,2275	0
ROULETTE	8642,226	7935,2324	9234,794	381,19824	3

1) Rozmiar populacji: 250 Liczba generacji: 250
2) Rozmiar populacji: 150 Liczba generacji: 150
3) Rozmiar populacji: 500 Liczba generacji: 500 (Tylko ruletka)

Wnioski:
Dla najmniejszego problemu ruletka daje gorsze rezultaty przy tej samej liczbie generacji i rozmiarze populacji.

Przy zmniejszonej liczbie osobników, wynik dla turnieju pogarsza się nieznacznie (różnica pomiędzy średnimi około 100), a ten dla ruletki – bardzo (różnica pomiędzy średnimi - 800).

Równocześnie warto zauważyć, że z uwagi na jej niższą złożoność obliczeniową, ruletka ma krótszy czas wykonania. Spróbowałam porównać działanie ruletki dla czasu porównywalnego z działaniem turnieju, lecz i w tym wypadku dała ona gorsze rezultaty (tym razem jednak tylko o średnio 400).

DLA PROBLEMU KROA100:

Tabela 17 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od metody selekcji dla problemu Kroa100 i parametrów podanych poniżej.

METODA	AVG	MIN	MAX	STAND.DEV	AVGTIME
TOURNAMENT	23437,393	22476,518	24658,7	692,2393	15
ROULETTE	32617,215	25926,91	35443,06	2693,599	3
TOURNAMENT	23923,922	22807,557	24731,69	657,6052	5
ROULETTE	37579,336	32884,207	41434,83	2741,662	1
ROULETTE	29902,506	27692,988	32278,09	1256,664	8

1) Rozmiar populacji: 500 Liczba generacji: 500
2) Rozmiar populacji: 300 Liczba generacji: 300
3) Rozmiar populacji: 750 Liczba generacji: 750 (tylko ruletka)

Wnioski:
Po raz kolejny ruletka okazuje się znacznie mniej skuteczna od turnieju, ale na jej korzyść przemawia szybkość wykonania (jest pięciokrotnie szybsza).

Po raz kolejny obserwujemy też znaczne pogorszenie się wyniku ruletki przy zmniejszeniu rozmiaru populacji i liczby generacji, które tylko nieznacznie pogarszają wynik przy metodzie turniejowej.

Można też zauważyć, że ruletka o czasie wykonania 8 s daje nadal gorsze wyniki niż turniej o czasie 5 s, więc nawet szybkość wykonania nie przemawia na jej korzyść.

DLA PROBLEMU KROA150:

Tabela 18 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od metody selekcji dla problemu Kroa150 i parametrów podanych poniżej.

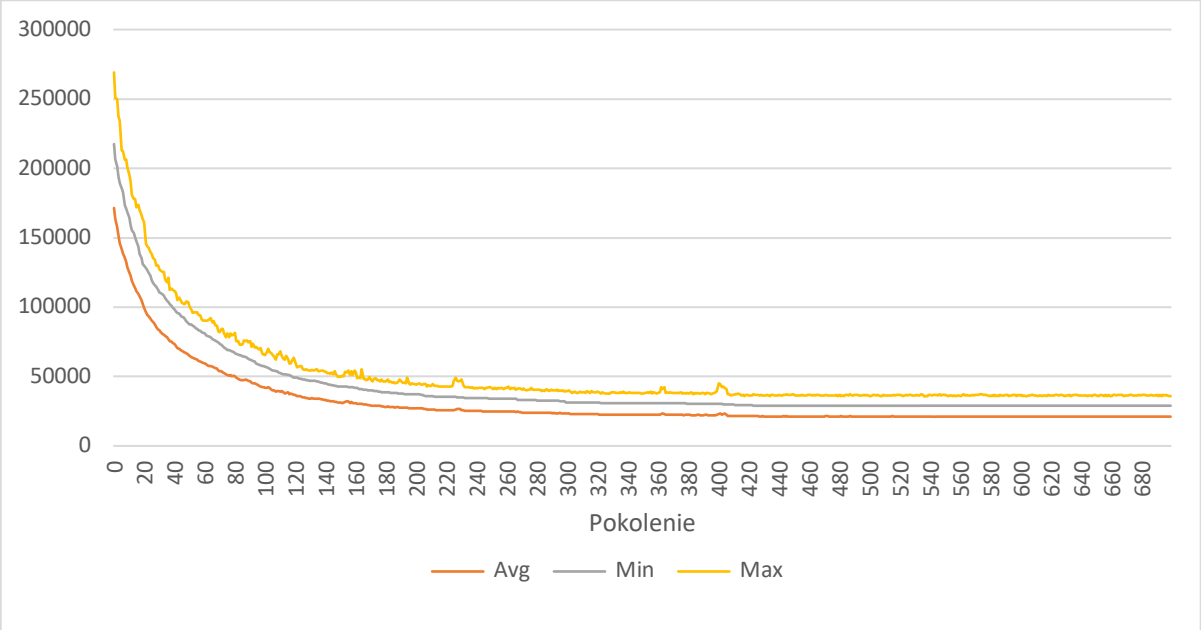
METODA	AVG	MIN	MAX	STAND.DEV	AVGTIME
TOURNAMENT	28871,395	27946,28	29847,844	500,3845	44
ROULETTE	46326,395	43056,523	50872,316	2579,4172	10
TOURNAMENT	29716,781	28395,676	31075,443	644,2161	24
ROULETTE	52038,492	43375,15	55700,88	3326,9102	5

1) Rozmiar populacji: 700 Liczba generacji: 700
2) Rozmiar populacji: 500 Liczba generacji: 500

Wnioski:
Ruletka daje znacznie gorsze wyniki od turnieju, ale także ponad czterokrotnie krótszy czas.

PORÓWNANIE PRZEBIEGU ALGORYTMU DLA PROBLEMU KROA150:

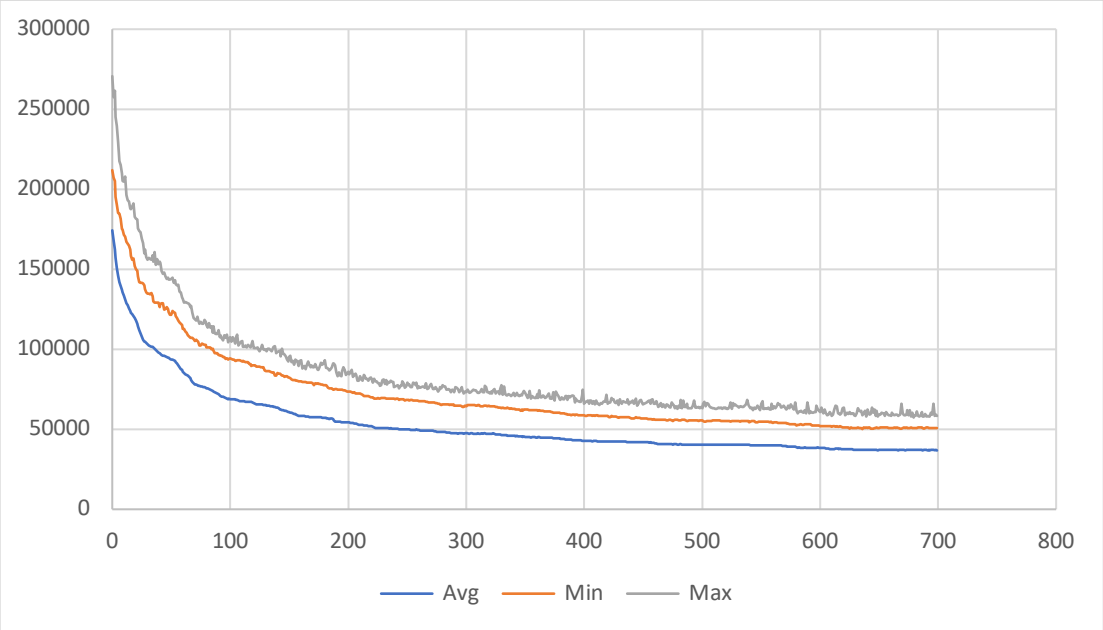
Turniej:
Rozmiar populacji: 500 Liczba generacji: 700 Rozmiar turnieju: 25
Prawdopodobieństwo krzyżowania: 0,9 Prawdopodobieństwo mutacji: 0,4



Wykres 23 Przebieg algorytmu dla metody selekcji turniejowej dla problemu KroA150

Ruletka:

Rozmiar populacji: 500 Liczba generacji: 700 Prawdopodobieństwo krzyżowania: 0,9 Prawdopodobieństwo mutacji: 0,4



Wykres 24 Przebieg algorytmu dla metody selekcji ruletki dla problemu KroA150

Wnioski:

W przypadku metody ruletki opadanie wykresu jest nieco bardziej rozłożone na przestrzeni pokoleń, w metodzie turniejowej wynik obniża się dość gwałtownie i osiąga lepsze rezultaty.

Da się też zauważyć, że wynik ruletki obniża się gwałtownie nieznacznie szybciej

DLA PROBLEMU KROA200:

Tabela 19 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od metody selekcji dla problemu Kroa200 i parametrów podanych poniżej.

METODA	AVG	MIN	MAX	STAND.DEV	AVGTIME
TOURNAMENT	33372,82	32832,38	34606,66	511,98325	59
ROULETTE	61422,71	55187,09	64859,99	2938,9482	12
TOURNAMENT	34965,25	32984,06	36596,08	1015,63043	41
ROULETTE	69527,28	65636,93	76492,24	3223,8665	7

1) Rozmiar populacji: 700 Liczba generacji: 700
2) Rozmiar populacji: 500 Liczba generacji: 500

Wnioski:

Ruletka dała dla tego problemu niemal dwukrotnie gorsze rozwiązanie w pięciokrotnie krótszym czasie. Niestety

DLA PROBLEMU FPL 417:

Tabela 20 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od metody selekcji dla problemu Fl417 i parametrów podanych poniżej.

METODA	AVG	MIN	MAX	STAND.DEV	AVGTIME
TOURNAMENT	15586,355	14699,094	16575,168	610,1215	207
ROULETTE	49362,344	46609,28	56331,11	3485,4426	36
TOURNAMENT	19533,791	18236,387	20596,602	681,15564	140
ROULETTE	58978,75	53113,266	63685,934	2865,2456	24

1) Rozmiar populacji: 900 Liczba generacji: 900
2) Rozmiar populacji: 700 Liczba generacji: 700

Wnioski:

Dla większego problemu wyraźniej jeszcze widać przepaść, jaka dzieli wyniki działania obydwu metod.

Wnioski do porównania operatorów selekcji:

Operator turniejowy zdecydowanie wygrywa z ruletką pod względem wyniku dla danej liczności populacji i liczby generacji.

Operator ruletki daje jednak rezultaty w o wiele krótszym czasie.

Niestety po zbadaniu wyników ruletki i turnieju dla podobnych czasów dla niektórych problemów, okazuje się, że mimo to ruletka nie daje lepszych wyników.

Można też zauważyć, że w ruletce przesunięta jest optymalna liczba generacji i rozmiar populacji – występują o wiele większe różnice w wyniku dla różnych ich wartości niż przy metodzie turniejowej, potrzeba też więcej generacji i większego rozmiaru populacji, aby osiągnąć „akceptowalne” wyniki.

BADANIE WPŁYWU PRAWDOPODOBIEŃSTWA MUTACJI I KRZYŻOWANIA

Cel: Zbadanie wpływu zastosowanego prawdopodobieństwa mutacji i krzyżowania na działanie algorytmu ewolucyjnego w różnych problemach.

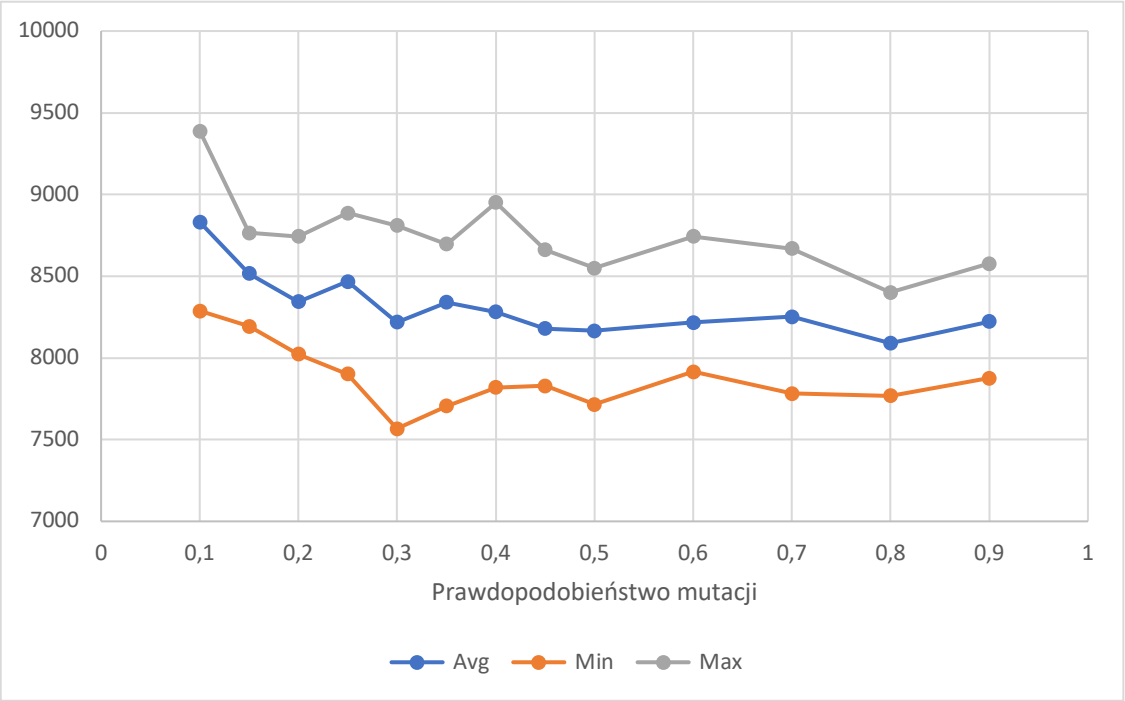
BADANIE WPŁYWU PRAWDOPODOBIEŃSTWA MUTACJI

DLA PROBLEMU BERLIN52:

Rozmiar populacji: 200 Liczba generacji: 200 Rozmiar turnieju: 20
Prawdopodobieństwo krzyżowania: 0,85

Tabela 21 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od prawdopodobieństwa mutacji dla problemu Berlin52.

PRAWDOPODOB. MUTACJI	AVG	MIN	MAX	STAND.DEV	AVGTIME
0,1	8831,089	8287,859	9388,205	339,00006	1
0,15	8516,823	8193,439	8765,067	207,46085	1
0,2	8343,324	8022,449	8744,117	219,00067	1
0,25	8467,598	7900,7134	8885,365	264,07935	1
0,3	8219,001	7567,328	8809,314	377,80173	1
0,35	8339,893	7705,6914	8696,771	266,09793	1
0,4	8280,464	7819,5483	8951,763	344,78976	1
0,45	8179,2554	7829,951	8662,0625	230,75899	1
0,5	8166,5522	7715,797	8549,469	276,536	1
0,6	8216,299	7914,3735	8743,848	229,16132	1
0,7	8252,78	7782,4053	8668,936	269,7283	1
0,8	8090,1157	7769,289	8401,056	229,31879	1
0,9	8222,118	7876,015	8577,294	220,3504	1



Wykres 25 Zależność wyniku od prawdopodobieństwa mutacji dla problemu Berlin52, rozmiaru populacji 200 i liczby generacji 200.

Wnioski:
Prawdopodobieństwo w przedziale 0-0,2 jest zdecydowanie zbyt niskie– daje wysokie wyniki.

Zależność zdaje się osiągać swoje minimum lokalne dla dwóch wartości prawdopodobieństwa mutacji – 0,3 oraz 0,8.

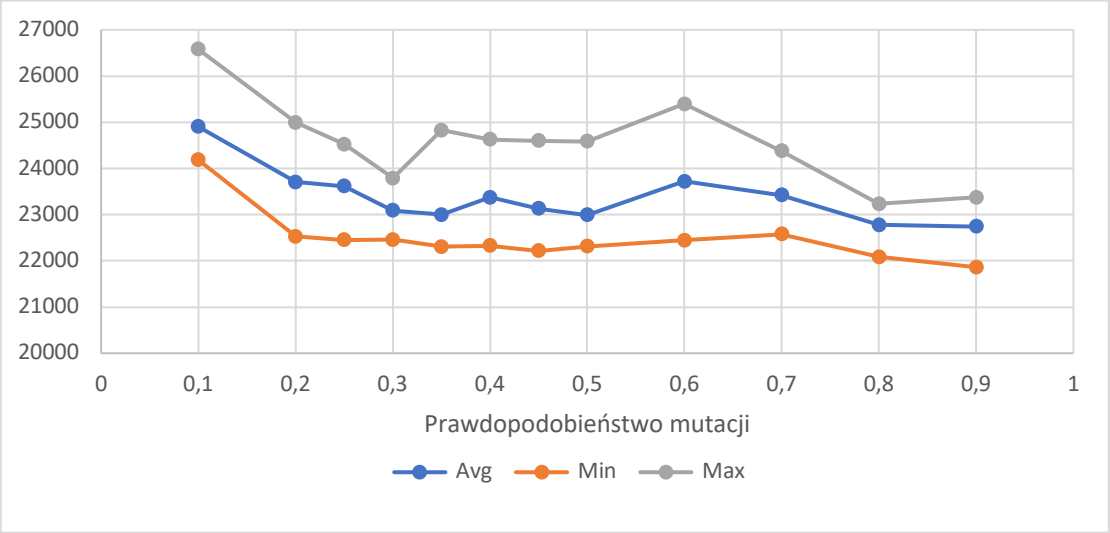
Prawdopodobieństwo mutacji nie ma wpływu na czas wykonania algorytmu.

DLA PROBLEMU KROA100:

Rozmiar populacji: 450 Liczba generacji: 450 Rozmiar turnieju: 20
Prawdopodobieństwo krzyżowania: 0,85

Tabela 22 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od prawdopodobieństwa mutacji dla problemu KroA100.

PRAWDOPODOB. MUTACJI	AVG	MIN	MAX	STAND.DEV	AVGTIME
0,1	24903,473	24184,535	26580,05	645,71075	10
0,2	23705,727	22529,615	24997,568	746,42303	10
0,25	23616,402	22452,49	24523,898	731,3615	10
0,3	23086,24	22458,688	23789,621	417,88855	10
0,35	23002,23	22308,438	24827,81	748,59674	10
0,4	23372,285	22324,215	24623,77	818,02	10
0,45	23128,309	22214,639	24602,045	846,60626	10
0,5	22995,39	22316,166	24587,95	656,69617	10
0,6	23717,645	22447,803	25397,566	833,18555	10
0,7	23419,002	22576,588	24374,854	495,2954	10
0,8	22777,951	22085,338	23235,58	419,02365	10
0,9	22740,6	21862,562	23372,29	570,19507	10



Wykres 26 Zależność wyniku od prawdopodobieństwa mutacji dla problemu KroA100, rozmiaru populacji 450 i liczby generacji 450.

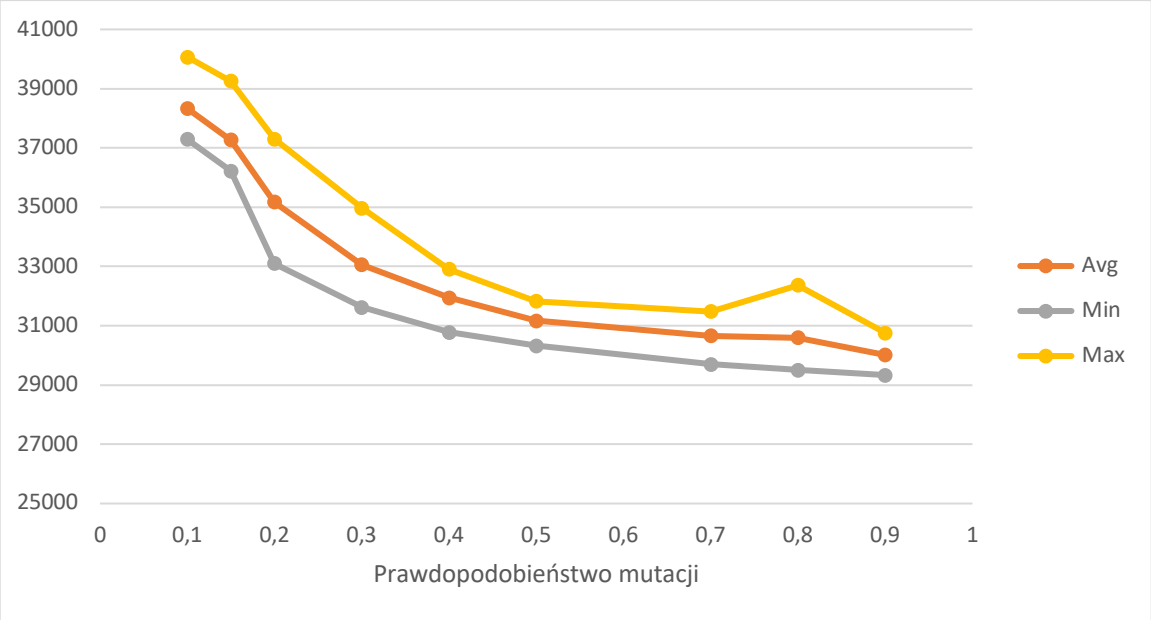
Wnioski: Po raz kolejny wartości są niestabilne – na pewno można stwierdzić, że prawdopodobieństwo poniżej 0,2 daje zauważalnie gorsze wyniki, natomiast optymalne znajduje się około 0,4. Nie wpływa ono na czas działania algorytmu.

DLA PROBLEMU KROA150:

Rozmiar populacji: 450 Liczba generacji: 450 Rozmiar turnieju: 20
Prawdopodobieństwo krzyżowania: 0,85

Tabela 23 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od prawdopodobieństwa mutacji dla problemu KroA150.

MUTPROB	AVG	MIN	MAX	STAND.DEV	AVGTIME
0,1	38329,73	37287,055	40071,79	993,11664	11
0,15	37275,418	36226,414	39258,684	881,0066	11
0,2	35172,82	33103,082	37298,29	1154,3646	11
0,3	33054,676	31622,293	34969,633	896,86847	11
0,4	31942,691	30779,428	32909,348	526,4669	11
0,5	31166,525	30329,438	31826,203	521,7813	11
0,7	30658,443	29703,377	31483,135	531,30316	11
0,8	30596,723	29509,934	32362,227	834,5533	11
0,9	30014,584	29336,057	30760,713	444,38275	11



Wykres 27 Zależność wyniku od prawdopodobieństwa mutacji dla problemu KroA150, rozmiaru populacji 450 i liczby generacji 450.

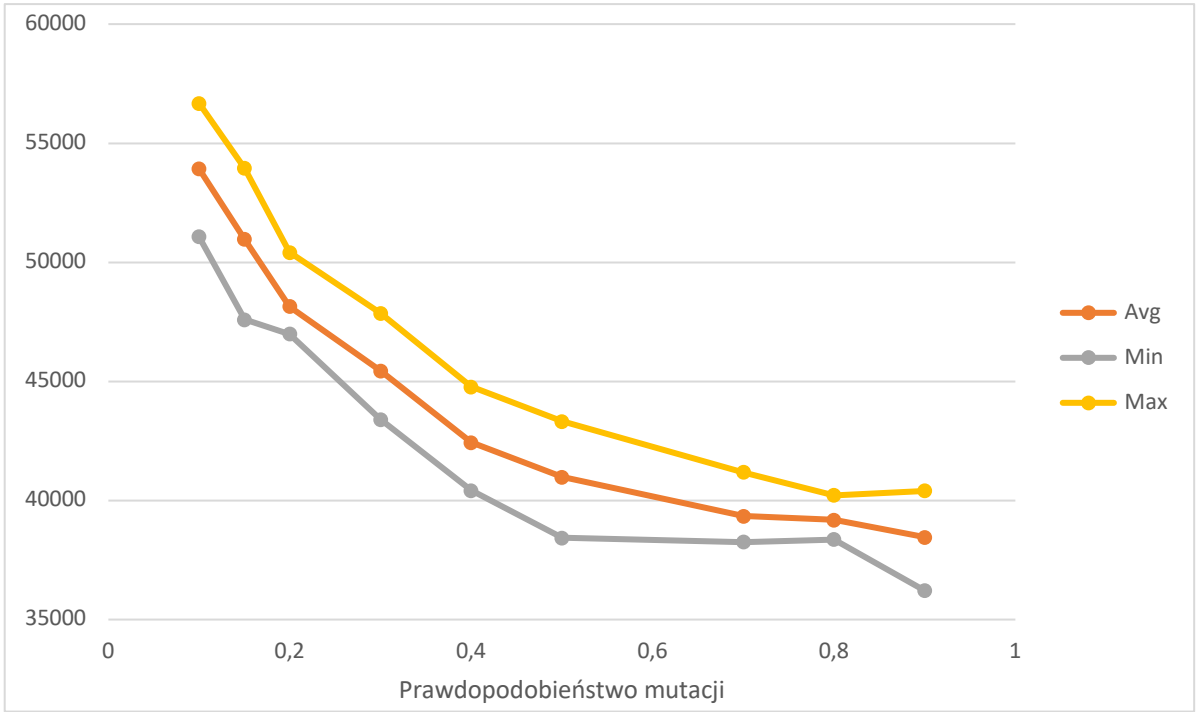
Wnioski: Dla tego problemu wzrost prawdopodobieństwa powoduje poprawę wyniku. Najgorsze wyniki uzyskuje się dla prawdopodobieństwa mutacji<0,4; dla większych wyników poprawa jest mniejsza.

DLA PROBLEMU KROA200:

Rozmiar populacji: 400 Liczba generacji: 400 Rozmiar turnieju: 20

Tabela 24 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od prawdopodobieństwa mutacji dla problemu KroA200.

MUTPROB	AVG	MIN	MAX	STAND.DEV	AVGTIME
0,1	53922,633	51067,613	56679,297	2072,482	19
0,15	50960,434	47594,17	53942,98	1868,0732	19
0,2	48144,887	46994,863	50412,8	1063,8796	19
0,3	45440,895	43400,684	47866,797	1407,2734	19
0,4	42439,293	40421,715	44772,14	1507,9509	19
0,5	40984,992	38434,727	43314,855	1385,3593	19
0,7	39345,22	38249,215	41181,613	795,6368	19
0,8	39181,375	38364,566	40212,57	632,65076	19
0,9	38462,668	36217,31	40408,363	1124,649	19



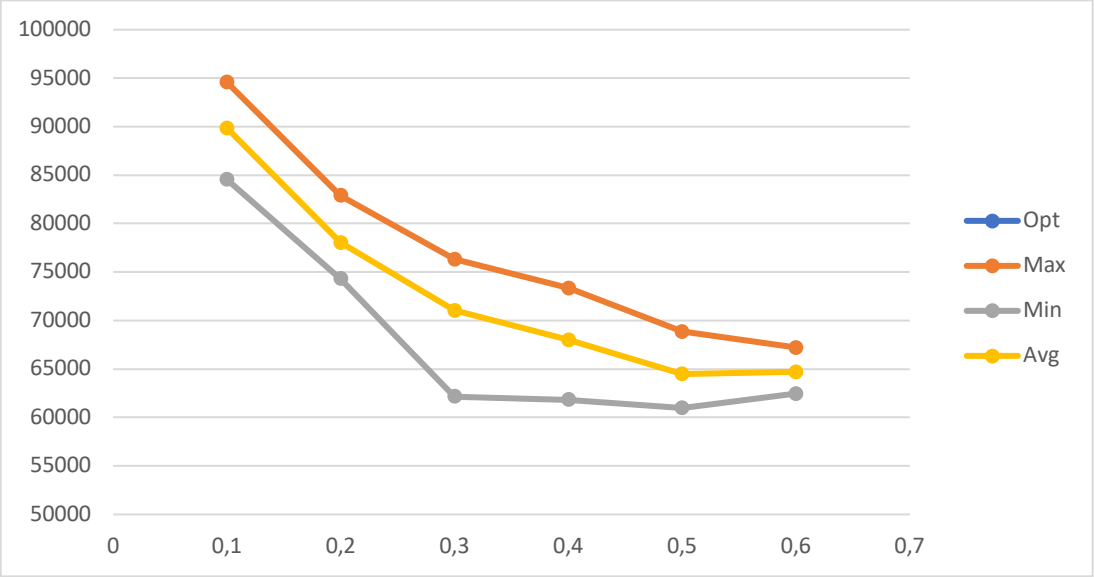
Wykres 28 Zależność wyniku od prawdopodobieństwa mutacji dla problemu KroA100, rozmiaru populacji 400 i liczby generacji 400.

Wnioski:
Wydaje się, że wyższe prawdopodobieństwo mutacji daje znacząco lepsze wyniki w przypadku, gdy rozmiar populacji i liczba generacji są niskie w stosunku do poziomu trudności problemu

DLA PROBLEMU FPL 417:

Rozmiar populacji: 300 Liczba generacji: 300 Rozmiar turnieju: 20

MUTPROB	AVG	MIN	MAX	STAND.DEV	AVGTIME
0,1	89838,92	84534,72	94565,44	3018,5298	22
0,2	78039,34	74290,125	82886	2403,8838	22
0,3	71051,36	62158,875	76327,695	4250,3613	22
0,4	68000,42	61815,164	73333	3181,203	22
0,5	64482,22	60982,707	68855,88	2651,136	22
0,6	64711,113	62469,6	67225,02	1544,433	22



Wnioski: Funkcja przyjmuje najlepsze średnie wartości dla prawdopodobieństwa mutacji ok. 0,5. Najlepsze wyniki (Min) nie zmieniają się znacznie dla prawdopodobieństwa w przedziale 0,3 – 0,5. Optymalnym prawdopodobieństwem mutacji jest więc około 0,5. Warto też po raz kolejny zauważyć, że parametr ten nie ma wpływu na czas wykonania algorytmu, co jest szczególnie ważne przy trudnych problemach

Wnioski do badania prawdopodobieństwa mutacji:

Badanie prawdopodobieństwa mutacji dało mniej „stabilne” wyniki niż wcześniejszych parametrów.
Przy badaniach dla „optymalnej” liczby generacji i rozmiaru populacji (problem 1 i 2), prawdopodobieństwo mutacji nie miało ogromnego wpływu na rezultat, o ile nie było zbyt niskie (poniżej 0,2).
Natomiast dla problemów, dla których liczba generacji i rozmiar populacji były mniejsze od optymalnych, im wyższe było prawdopodobieństwo mutacji, tym lepszy wynik. Być może większa różnorodność populacji, na którą wpływa prawdopodobieństwo mutacji, dla zbyt małych wartości tych parametrów, pomaga uzyskać dobry wynik.

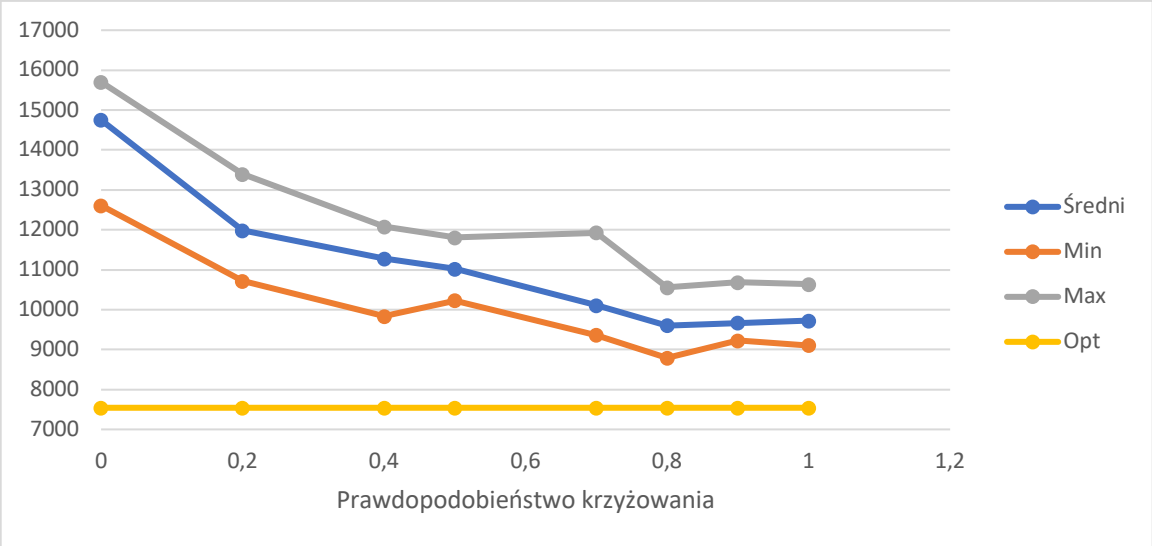
BADANIE PRAWDOPODOBIEŃSTWA KRZYŻOWANIA:

DLA PROBLEMU BERLIN52:

Rozmiar populacji:200 Liczność populacji:200 Rozmiar turnieju:20
Prawdopodobieństwo mutacji:0.2

Tabela 25 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od prawdopodobieństwa krzyżowania dla problemu Berlin52.

PRAWD KRZYŻ	AVG	MIN	MAX	STAND.DEV	AVGTIME
0	14749,891	12609,553	15702,442	890,79425	1
0,2	11981,494	10713,368	13391,27	739,48303	1
0,4	11278,044	9833,721	12076,675	700,37085	1
0,5	11017,856	10226,801	11807,6875	450,86475	1
0,7	10106,025	9361,067	11922,584	689,21906	1
0,8	9599,637	8791,1	10559,4375	593,3956	1
0,9	9664,188	9221,855	10684,923	386,19058	1
1	9722,888	9103,684	10638,142	418,2034	1



Wykres 29 Zależność wyniku od prawdopodobieństwa krzyżowania dla problemu Berlin52, rozmiaru populacji 200 i liczby generacji 200

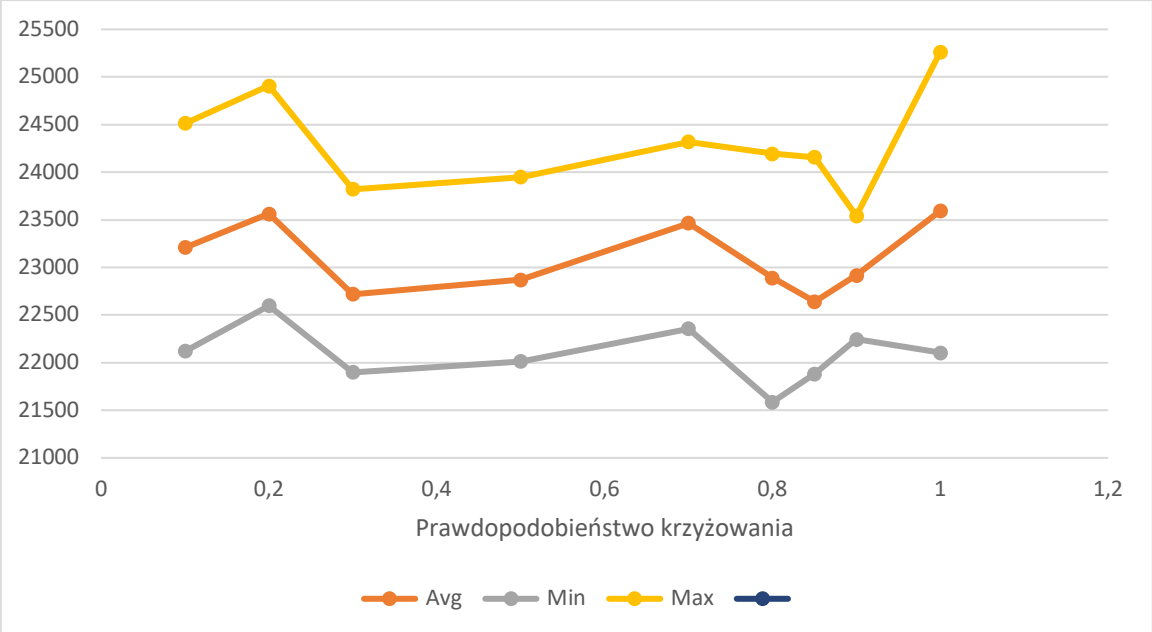
Wnioski:
Wynik maleje wraz ze wzrostem prawdopodobieństwa krzyżowania do wartości około 0,8- 0,9 a następnie nieznacznie wzrasta. Optymalna wartość znajduje się w tym przedziale. Parametr nie ma wpływu na czas wykonania algorytmu.

DLA PROBLEMU KROA100:

Rozmiar populacji: 500 Liczba generacji: 500 Rozmiar turnieju: 20
Prawdopodobieństwo mutacji: 0,4

Tabela 26 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od prawdopodobieństwa krzyżowania dla problemu Kroa100.

PRAWD. KRZYŻ.	AVG	MIN	MAX	STAND.DEV	AVGTIME
0,1	23209,152	22121,678	24513,557	797,1261	11
0,2	23561,775	22597,146	24906,557	799,966	11
0,3	22718,879	21900,646	23821,49	669,8297	11
0,5	22869,549	22014,195	23949,154	615,3893	11
0,7	23464,74	22357,676	24319,064	542,27814	11
0,8	22891,344	21586,426	24195,527	728,49835	11
0,85	22640,965	21881,219	24156,748	746,543	11
0,9	22919,045	22246,797	23542,379	438,97787	11
1	23594,602	22104,436	25264,918	1017,4561	11



Wykres 30 Zależność wyniku od prawdopodobieństwa krzyżowania dla problemu KroA100, rozmiaru populacji 500 i liczby generacji 500

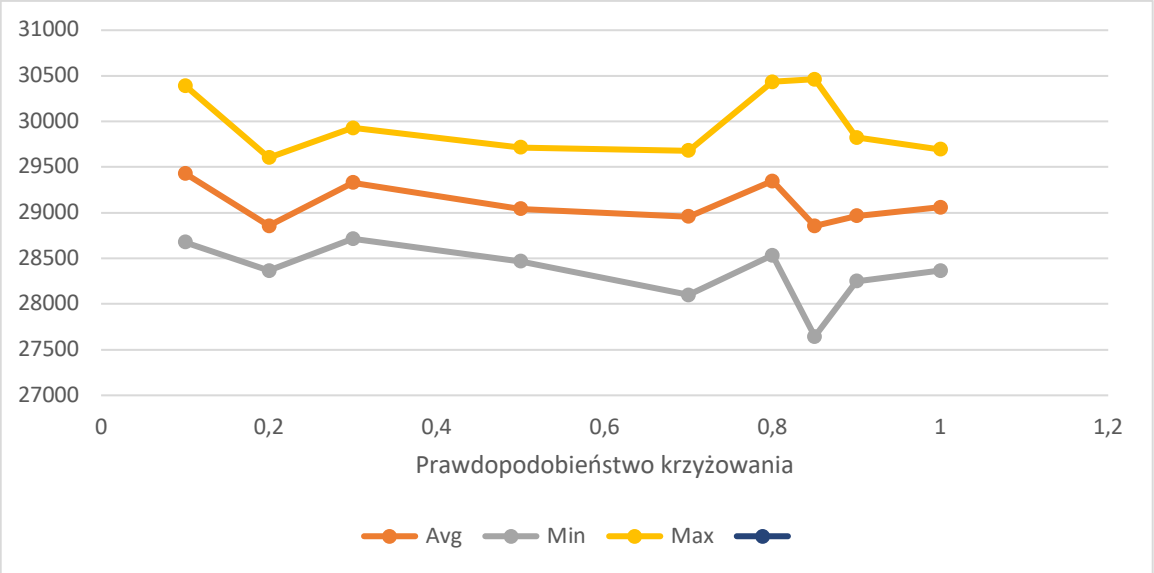
Wnioski:
Minimum funkcji średnio wypada pomiędzy 0,8 a 0,9, tak jak w poprzednim problemie, chociaż tutaj trudno zaobserwować stałą zależność wyniku od prawdopodobieństwa krzyżowania – jest ona bardzo niestabilna.
Zmiana prawdopodobieństwa krzyżowania nie wpływa na czas wykonania algorytmu.

DLA PROBLEMU KROA150:

Rozmiar populacji: 700 Liczba generacji: 700 Rozmiar turnieju: 20
Prawdopodobieństwo mutacji: 0,4

Tabela 27 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od prawdopodobieństwa krzyżowania dla problemu Kroa150

PRAWD. KRZYŻ.	AVG	MIN	MAX	STAND.DEV	AVGTIME
0,1	29431,578	28676,377	30393,213	514,3101	32
0,2	28857,156	28366,318	29603,678	326,6514	32
0,3	29331,34	28715,082	29930,365	412,5998	32
0,5	29042,75	28468,26	29714,268	374,98236	32
0,7	28957,137	28098,312	29681,125	440,0992	32
0,8	29346,146	28533,365	30434,7	582,3228	32
0,85	28855,676	27644,852	30463,512	746,06433	32
0,9	28965,947	28251,363	29824,879	530,40125	32
1	29059,875	28365,35	29693,934	435,53967	32



Wykres 31 Zależność wyniku od prawdopodobieństwa krzyżowania dla problemu KroA150, rozmiaru populacji 700 i liczby generacji 700

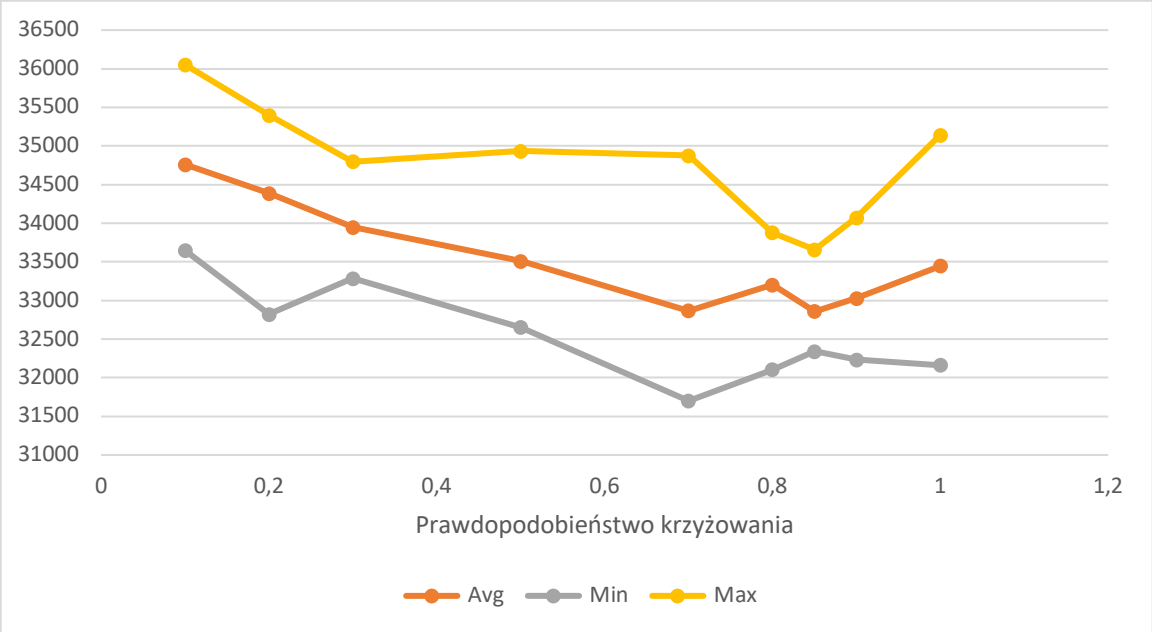
Wnioski:
Wykres jest podobny do tego dla poprzedniego problemu – niestabilny i trudno jednoznacznie wskazać najlepszą wartość, jednak w przedziale 0,1-0,7 można zauważyć opadanie wykresu, a także pewne minimum dla 0,85.

DLA PROBLEMU KROA200:

Rozmiar populacji: 700 Liczba generacji: 700 Rozmiar turnieju: 20
Prawdopodobieństwo mutacji: 0,4

Tabela 28 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od prawdopodobieństwa krzyżowania dla problemu Kroa200

PRAWD. KRZYŻ.	AVG	MIN	MAX	STAND.DEV	AVGTIME
0,1	34760,32	33647,49	36052,152	798,4899	44
0,2	34385,273	32819,41	35398,08	715,2907	44
0,3	33946,05	33283,973	34796,42	451,97464	44
0,5	33509,457	32652,611	34935,582	757,1729	44
0,7	32867,97	31699,76	34876,586	898,47845	44
0,8	33201,637	32106,025	33877,645	603,09875	44
0,85	32856,676	32339,463	33658,016	401,66296	44
0,9	33027,41	32234,47	34071,77	647,0634	44
1	33446,367	32161,68	35140,79	1010,26984	44



Wykres 32 Zależność wyniku od prawdopodobieństwa krzyżowania dla problemu KroA200, rozmiaru populacji 700 i liczby generacji 700

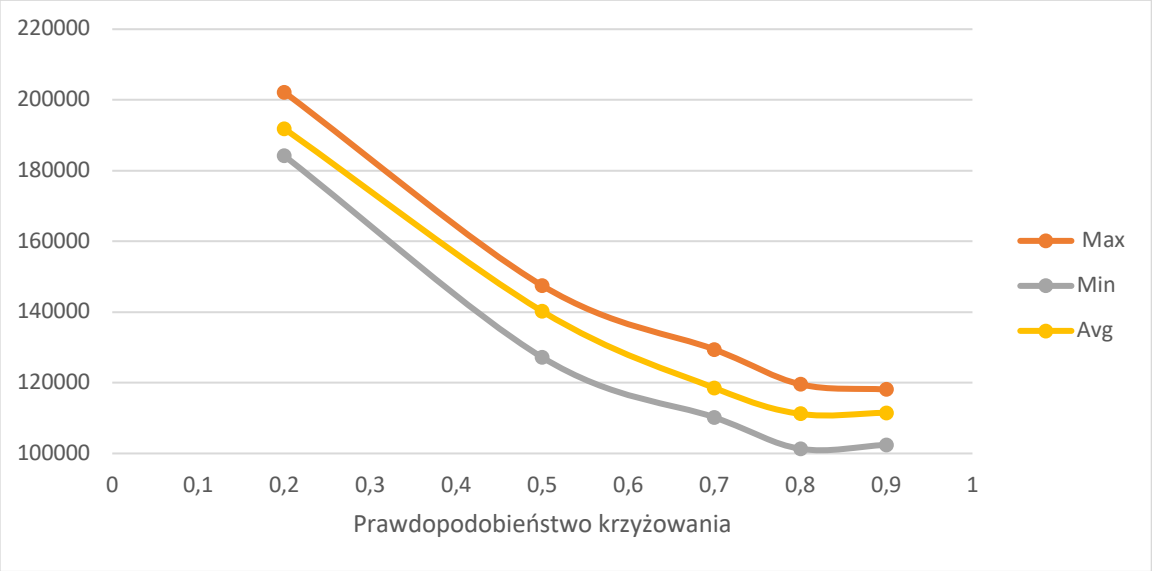
Wnioski:
Dla tego problemu znowu można zauważyć opadanie wyniku dla coraz mniejszych wartości prawdopodobieństwa i wzrost w przedziale 0,85-1.

DLA PROBLEMU FL 417:

Rozmiar populacji: 250 Liczba generacji: 250 Rozmiar turnieju: 20
Prawdopodobieństwo mutacji: 0,4

Tabela 29 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania algorytmu dla 10 uruchomień w zależności od prawdopodobieństwa krzyżowania dla problemu Fl417

PRAWD. KRZYŻ.	AVG	MIN	MAX	STAND.DEV	AVGTIME
0,2	191862,61	184182,38	202187,72	5432,3984	20
0,5	140181,28	127156,664	147456,36	6600,6904	20
0,7	118532,234	110198,52	129402,625	5724,0034	20
0,8	111195	101325,77	119567	5189,4434	20
0,9	111518,33	102438,3	118083,41	3996,0144	20



Wykres 33 Zależność wyniku od prawdopodobieństwa krzyżowania dla problemu Fl417, rozmiaru populacji 600 i liczby generacji 600

Wnioski:

Dla tego problemu obserwujemy gwałtowny spadek wyniku dla większych prawdopodobieństw. Jest tak być może ze względu na bardzo niską liczbę pokoleń i rozmiar populacji w stosunku do tego problemu.

Po raz kolejny nie obserwuje się wpływu prawdopodobieństwa krzyżowania na czas.

Wnioski do badania prawdopodobieństwa krzyżowania:
Można zaobserwować znaczące różnice wpływu prawdopodobieństwa krzyżowania na różne problemy.

Dla problemów o niemal wysokiej, optymalnej dla nich liczbie generacji i rozmiarze populacji prawdopodobieństwo nie ma dużego wpływu na wynik

Im bardziej liczba generacji i rozmiar populacji są oddalone od optymalnych (KroA200, Berlin52, Fl417), tym gwałtowniej wzrost prawdopodobieństwa krzyżowania wpływa na poprawę wyniku. Jest to prawdopodobnie związane z szybszym tempem zmian w kolejnych populacjach.

Prawdopodobieństwo krzyżowania nie wpływa w znaczący sposób na czas wykonania algorytmu.

PODSUMOWANIE BADANIA PRAWDOPODOBIEŃSTWA KRZYŻOWANIA I MUTACJI

Zależność wyniku od prawdopodobieństwa krzyżowania i mutacji jest znacznie bardziej subtelna oraz zależna od rozmiaru populacji i liczności generacji.

Dla większych wartości rozmiaru populacji i liczności generacji (w stosunku do problemu), **wpływ tych prawdopodobieństw jest niewielki**, o ile nie są zbyt małe (dla prawdopodobieństwa mutacji <0,2; a prawdopodobieństwa krzyżowania <0.4). Trudno też ocenić ich wpływ, ponieważ zależności wydają się niestabilne.

Dla niewielkich wartości rozmiaru populacji i liczności generacji w stosunku do trudności problemu, dobór prawdopodobieństwa krzyżowania i mutacji staje się bardzo ważny. Wyraźnie widać, że **im większe wartości mają te parametry tym lepszy wynik** (poza ostatnim odcinkiem 0,9 – 1, gdzie dla wielu wykresów widać niewielkie pogorszenie wyniku).

Taka zależność może wynikać z tego, że dla małych rozmiarów populacji i liczności generacji szybkość zmieniania się populacji jest kluczowym czynnikiem, a prawdopodobieństwa te przyspieszają te zmiany.

Prawdopodobieństwo mutacji oraz krzyżowania **nie ma znacznego wpływu na czas** wykonania algorytmu.

PORÓWNANIE ALGORYTMU EWOLUCYJNEGO Z METODAMI NAIWNYMI

Cel: Zbadanie i porównanie działania trzech różnych sposobów rozwiązania problemu – „algorytmu" losowego, algorytmu ewolucyjnego oraz zachłannego.

DLA NIEWIELKIEJ POPULACJI I LICZBY GEN – (100, 100) ORAZ WIELKOŚCI TURNIEJU 5

Tabela 30 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania „algorytmu" losowego dla 10000 uruchomień

		Algorytm losowy			
Problem		Best	Worst	Avg	Std
berlin52	7542	23437,402	35494,125	29896,701	1570,1876
kroA100	21282	136823,56	200411,8	168468,6	8024,339
kroA150	26524	217640,34	291640,28	256265,55	10195,568
kroA200	29368	289084,97	376773,6	337442,06	11712,534
: fl417	11861	438437,56	546221,3	494585,66	13605,067

Tabela 31 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku działania algorytmu ewolucyjnego dla 10 uruchomień, liczba pokoleń-100 i rozmiar populacji - 100

	Algorytm ewolucyjny			
Problem	Best	Worst	Avg	Std
berlin52	10138,392	12515,886	11104,572	530,04956
kroA100	54319,996	70671,71	62526,1	2935,016
kroA150	99609,52	122884,84	111138,28	4394,819
kroA200	148936,16	178573,45	161131,2	5755,209
: fl417	229924,34	285481,1	260974,27	7953,702

Tabela 32 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku działania algorytmu zachłannego dla N uruchomień (N –rozmiar problemu)

	Algorytm zachłanny			
Problem	Best	Worst	Avg	Std
berlin52	8182,1924	10299,372	9339,926	486,69257
kroA100	24694,336	29186,004	27404,055	884,43115
kroA150	30173,42	36187,867	32986,633	1213,8065
kroA200	34327,926	42205,285	38350,77	1706,3612
: fl417	14753,613	17085,678	15783,502	418,89737

Wnioski:
Pierwsze badanie dotyczyło niewielkiej populacji i liczby generacji, dlatego szczególnie dla trudniejszych problemów, ich liczba nie jest wystarczająca.

Badanie pokazało, że algorytm ewolucyjny działa znacznie lepiej od losowego, lecz równocześnie o wiele gorzej od zachłannego. Ponadto algorytm zachłanny działa znacznie szybciej (ma niewielką złożoność).

DLA ŚREDNIEJ LICZBY POKOLEŃ I WIELKOŚCI POPULACJI

Liczba pokoleń: 400 Wielkość populacji: 400 Wielkość turnieju: 20

Tabela 33 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku i czas działania „algorytmu" losowego dla 160000 uruchomień

	Algorytm losowy			
Problem	Best	Worst	Avg	Std
berlin52	21486,338	35754,926	29907,951	1582,0753
kroA100	134584,44	201073,53	168432,33	8096,3813
kroA150	209252,62	301579,5	256366,31	10127,672
kroA200	283544,3	382525,28	337546,75	11696,83
: fl417	425842,66	552593,5	494690,34	13477,167

Tabela 34 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku działania algorytmu zachłannego dla N uruchomień

	Algorytm zachłanny			
Problem	Best	Worst	Avg	Std
berlin52	8182,1924	10299,372	9339,926	486,69257
kroA100	24694,336	29186,004	27404,055	884,43115
kroA150	30173,42	36187,867	32986,633	1213,8065
kroA200	34327,926	42205,285	38350,77	1706,3612
: fl417	14753,613	17085,678	15783,502	418,89737

Tabela 35 Średni, minimalny, maksymalny wynik oraz odchylenie standardowe wyniku działania algorytmu ewolucyjnego dla 10 uruchomień, liczba pokoleń-400 i rozmiar populacji - 400

	Algorytm ewolucyjny			
Problem	Best	Worst	Avg	Std
berlin52	7661,013	9161,188	8327,955	279,54803
kroA100	25406,457	32503,594	29071,336	1230,0756
kroA150	42317,934	52912,516	47506,914	2050,3652
kroA200	60458,934	75756,76	68878,14	3019,465
: fl417	74803,664	100995,27	87142,67	4101,7583

Wnioski:

Po raz kolejny algorytm zachłanny okazał się szybszym i efektywniejszym sposobem rozwiązania problemów, chociaż dla najmniejszego z nich możemy zobaczyć jego słabości – jako algorytm deterministyczny nigdy nie zbliży się do wyniku bardziej.

Algorytm ewolucyjny przy odpowiednich ustawieniach jest w stanie dać lepszy wynik (jak widać, tutaj dla najmniejszego problemu jest to wynik oddalony zaledwie o około 120 od optymalnego). Trzeba jednak zauważyć, że jego złożoność obliczeniowa i czas działania są znacząco większe.

Wnioski do porównania algorytmów:

Algorytm ewolucyjny działa znacznie lepiej od **algorytmu losowego** dla wszystkich problemów.

Algorytm zachłanny daje w większości przypadków znacznie lepsze wyniki niż algorytm ewolucyjny, ale zawsze te same i nie może zbliżyć się do wyniku bardziej.

Być może warto rozważyć wykorzystanie algorytmu zachłannego przy tworzeniu części osobników z pierwszego pokolenia w algorytmie ewolucyjnym.

PODSUMOWANIE: WYKORZYSTANIE WYNIKÓW BADAŃ DO USPRAWNIENIA ALGORYTMU

Po poznaniu wyników badań zdecydowałam się uruchomić algorytm z następującymi zmianami:

1) **Zmiana sposobu generowania populacji – wykorzystanie algorytmu zachłannego do tworzenia 10% pierwszej populacji.**

2) Zmiana domyślnego prawdopodobieństwa krzyżowania na 0.8 oraz prawdopodobieństwa mutacji na 0.4

3) *Uruchomienie dla optymalnych (odpowiednich do problemu) wartości rozmiaru populacji i liczby generacji.

4) *Zmiana wielkości turnieju na 20.

*- Choć te wartości są dobrane już w czasie trwania badania, trudno byłoby porównać wynik „przed" i „po", stąd dla obu uruchomień wykorzystuję te same.

Wynik przed zmianami:

Tabela 36 Średni, najlepszy i najgorszy wynik przed wprowadzeniem w życie zmian w algorytmie dla podanych problemów i parametrów.

PROBLEM	POPSIZE	NUMOFGEN	TNMSIZE	CP	MP	BEST	WORST	AVG
BERLIN52	250	250	20	0,7	0,2	7910,603	8680,699	8286,261
KROA100	500	500	20	0,7	0,2	21783,06	24214,293	23077,5
KROA150	700	700	20	0,7	0,2	28267,26	30536,9	29749,422
KROA200	850	850	20	0,7	0,2	32214,725	35460,914	33489,426
: FL417	1000	1000	20	0,7	0,2	15351,517	18459,072	16928,078

Wynik po zmianach:

Tabela 37 Średni, najlepszy i najgorszy wynik po wprowadzeniu w życie zmian w algorytmie dla podanych problemów i parametrów.

PROBLEM NAME	POPSIZE	NUMOFGEN	TNMSIZE	CP	MP	BEST	WORST	AVG
BERLIN52	250	250	20	0,8	0,4	7544,3657	8172,924	7879,62
KROA100	500	500	20	0,8	0,4	21565,906	22079,672	21843,309
KROA150	700	700	20	0,8	0,4	27197,418	28174,662	27810,088
KROA200	850	850	20	0,8	0,4	30489,365	31343,64	30993,59
FL417	1000	1000	20	0,8	0,4	12802,101	13419,997	13042,014

Dzięki wiedzy zdobytej w trakcie badania, rezultaty działania algorytmu znacząco się poprawiły dla każdego problemu. (Szczególnie biorąc pod uwagę fakt, że wyniki „przed" wykorzystują już część badań nad parametrami)

W ogromnej mierze jest to jednak wynik nowego sposobu generowania pierwszej populacji – wykorzystania algorytmu zachłannego.