Projektowanie Efektywnych Algorytmów

Projekt

14/12/2022

4. Symulowane wyżarzanie

1. Sformułowanie zadania

Zadane polega na opracowaniu, implementacji w języku C++ i zbadaniu efektywności algorytmu **symulowanego wyżarzania** rozwiązującego problem komiwojażera w wersji optymalizacyjnej.

Problem komiwojażera polega na znalezieniu minimalnego cyklu Hamiltona w pełnym grafie ważonym: dane jest *n* miast, które komiwojażer ma odwiedzić, oraz odległość / cena podróży / czas podróży pomiędzy każdą parą miast. Celem jest znalezienie najkrótszej / najtańszej / najszybszej drogi łączącej wszystkie miasta, zaczynającej się i kończącej się w określonym punkcie.

W terminologii grafów miasta są wierzchołkami grafu, a trasy pomiędzy nimi to krawędzie z wagami. Waga krawędzi może odpowiadać odległości pomiędzy miastami połączonymi tą krawędzią, czasowi podróży lub kosztom przejazdu - zależy, co chcemy w podróżny komiwojażera zminimalizować. Trasa komiwojażera jest cyklem przechodzącym przez każdy wierzchołek grafu dokładnie jeden raz - jest to zatem cykl Hamiltona.

Algorytm symulowanego jest algorytmem heurystycznym, co znaczy że niekoniecznie znajduje rozwiązanie optymalne, ale w rozsądnym czasie znajduje rozwiązanie dostatecznie dobre, tj. takie, które nie odbiega znacząco od optymalnego. Metoda ta powinna sprawdzić się dla większych instancji problemu, w przypadku których użycie poprzednich metod **przeglądu zupełnego** oraz **programowania dynamicznego** nie zostałoby rozwiązane w sensownym czasie oraz z użyciem rozsądnej ilości zasobów komputera. Złożoność obliczeniowa tego algorytmu zależy głównie od doboru schematu schładzania, długośći epoki oraz temperatury początkowej.

2. Metoda

Symulowane wyżarzanie jest techniką projektowania algorytmów heurystycznych. Cechą charakterystyczną tej metody jest parametr sterujący zway temperaturą, który wraz z przebiegiem algorytmu maleje aż do osiągnięcia równowagi. Podejście to jest zainspirowne metalurgią, w której to wyżarzanie jest operacją nadającą metalom odpowiednią, zrównoważoną mikrostrukturę która przekłada się na lepszą jakośc surowca.

Podobnie jak w metalurgii, algorytm zostaje „nargzany" poprzez nadanie mu jakiejś (najczęściej wysokiej) temperatury początkowej. Od temperatury oraz schematu schładzania zależy prawdopodobieństwo wybrania gorszego niż optymalne rozwiązanie – im większa temperatura, tym większe prawdopodobieństwo. Wybór gorszych rozwiązań ma na celu opuszczenie minimów lokalnych w przestrzeni rozwiązań.

Algorytm zaczyna rozwiązywanie problemu z zadaną temperaturą, z losowego stanu. Wyliczany jest koszt początkowy, a następnie zaczyna przeszukiwanie rozwiązań w przestrzeni lokalnej, tj. w sąsiedztwie. Każde nowe rozwiązanie lepsze od aktualnego najlepszego rozwiązania jest akceptowane jako nowe najlepzse rozwiązanie, jednak aby uniknąć utknięcia w minimum lokalnym istnieje szansa, że zaakceptowane może zostać rozwiązanie gorsze od aktualnego najlepzsego.

Po zakończeniu się epoki temperatura ulega obniżeniu zgodnie ze schematem, a następnie poprzednie kroki są powtarzane aż do osiągnięcia temperatury końcowej.

Pseudokod programu (*E* koszt danego rozwiązania, *L* – długośc epoki):

```
1: Zainicjalizowanie zmiennych, parametrów wejściowych i losowe
dobranie stanu początkowego oraz wyliczenie jego kosztu E
2: POWTARZAJ L razy
3:      Wybór następnego stanu zgodnie z metodą wyboru sąsiedniego
        rozwiązania
4:      JEŻELI nowe E < E minimalne TO
5:           E minimalne = nowe E
6:      W P.P. zaakceptuj nowe, gorsze rozwiązanie z
        prawdopodobieństwem exp(-ΔE/T)
7: Obniż temperaturę zgodnie ze schematem schładzania
8: Powtarzaj kroki od 2 do 7 aż do osiągnięcia temperatury
minimalnej
```

Na początku działania algorytmu podanych zostaje 5 parametrów wejściowych:

- długość epoki ***L,***

- metoda schładzania: schemat ***Boltzmanna*** lub ***geometryczny***,

- metoda wyboru sąsiedniego rozwiązania: ***2-zmiana*** lub ***zamiana łuków,***

- temperatura początkowa T_0,

- parametr schładzania α.

Długość epoki:

Od długości epoki zależy czas rozwiązywania instancji, a dokładniej okres pomiędzy ochłodzeniami temperatury.

Sposób wyboru rozwiązania początkowego

Rozwiązanie początkowe jest wybierane losowo – droga uporządkowana w kolejności leksykograficznej zostaje całkowicie przelosowana przez algorytm, a następnie liczony jest koszt tej nowo powstałej drogi. I koszt i droga zostają zapisane jako kolejno najlepsze znalezione rozwiązanie oraz najlepsza znaleziona droga. W ten sposób z każdym uruchomieniem algorytmu istnieje prawie stuprocentowa szansa, że zaczniemy z kompletnie innego rozwiązania początkowego.

Innym sposobem który znam byłoby wylosowanie jednego wierzchołka startowego, a następnie wybranie wszystkich następnych wierzchołków metodą zachłanną. Mogłoby to usprawnić działanie algorytmu i poprawić osiągane wyniki, jednak ja zdecydowałem się nie implementować takiego podejścia.

Metody schładzania:

1. **Schemat Boltzmanna**

Inaczej schemat logarytmiczny, opiera się na wzorze:

$$T_t = \frac{T_{t-1}}{1 + \log(1 + L)}$$

gdzie T_t to wyliczana temperatura, T_{t-1} to poprzednia temperatura a L to długość epoki.

Dla odpowiednich parametrów teoretycznie możliwe jest wyliczanie za każdym razem wartości optymalnej, jednak zajęłoby to zdecydowanie więcej czasu niż możemy preznaczyć na rozwiązanie problemu.

2. Schemat geometryczny

Opiera się na wzorze:

$$\boldsymbol{T_t = T_{t-1} * a^L}$$

gdzie $\boldsymbol{T_t}$ to wyliczana temperatura, $\boldsymbol{T_{t-1}}$ to poprzednia temperatura,, $\boldsymbol{a}$ to parametr schładzania podawany na początku działania algorytmu, a $\boldsymbol{L}$ to długośc epoki.

Strategia ta nie gwarantuje znalezienia globalnego optimum, ale w rozsądnym czasie jest w stanie znaleźć rozwiązanie bardzo dobre, tj. zbliżone do wartości optymalnej.

Metody wyboru sąsiedniego rozwiązania:

1. **2-zmiana**

Program losuje 2 różne od siebie miasta z obecnej drogi, po czym zamienia je miejscami i wylicza koszt powstałej w ten sposób drogi, np. dla drogi

1-2-3-4-5-6-7-8

wylosowano 2 miasta – 2 i 5, po czym zamieniono je miejscami, w wyniku czego powstała droga:

1-**5**-3-4-**2**-6-7-8

2. **Zamiana łuków**

Program losuje 2 różne od siebie miasta z obecnej drogi, po czym odwraca przebieg między nimi i wylicza koszt powstałej w ten sposób drogi, np. dla drogi

1-2-3-4-5-6-7-8

wylosowano miasta – 3 i 7, po czym odwrócono przebieg, w wyniku czego powstała droga:

1-2-**7**-**6**-**5**-**4**-**3**-8

Wyliczanie prawdopodobieństwa zaakceptowania gorszego rozwiązania:

Jeśli nowe rozwiązanie jest gorsze niż aktualne i nie zostało zaakceptowane, istnieje prawdopodobieństwo *P*, że zostanie zamienione z aktualnym, wyliczane na podstawie wzoru

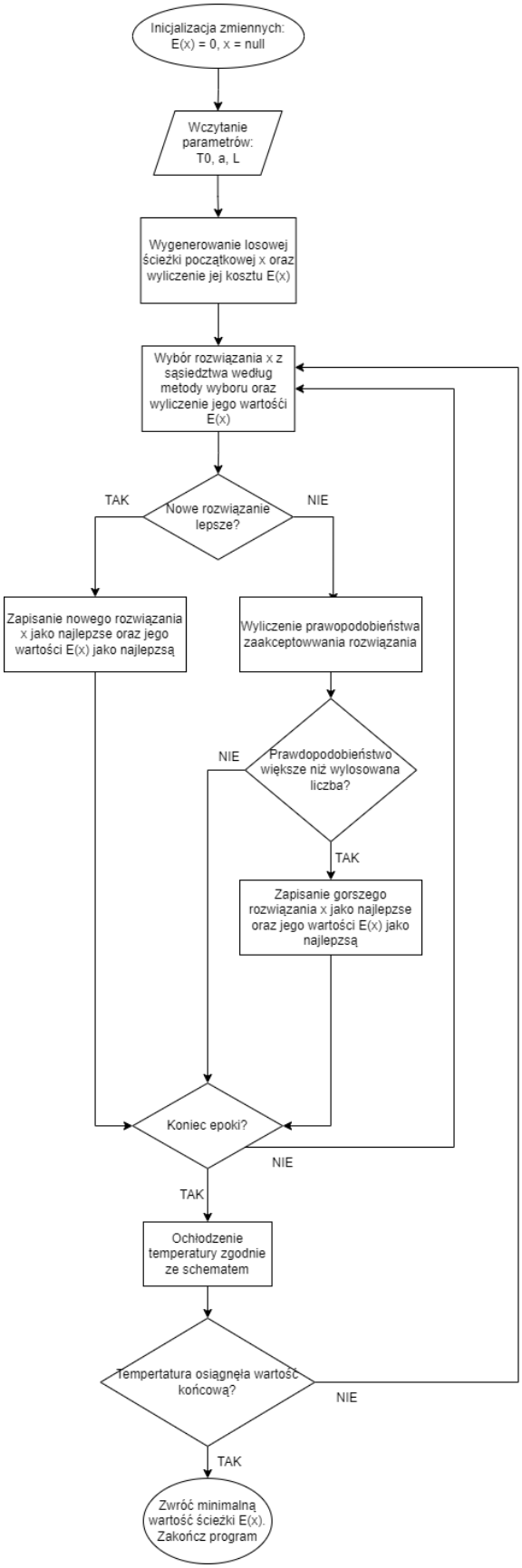
$$\boldsymbol{P = \exp\left(-\frac{\Delta E}{T}\right)}$$

gdzie ***exp*** oznacza funkcję $\mathbf{e^x}$, $\boldsymbol{\Delta E}$ to różnica między nowym kosztem a kosztem minimalnym, a $\boldsymbol{T}$ to aktualna temperatura.

Prawdopodobieństwo to jest porównywanem z losowo wygenerowaną liczbą z zakresu [0,1], jeśli jest większe to algorytm akceptuje zmianę, jeżeli nie to nie akceptuje.

3. Algorytm

Poniżej znajduje się schemat działania algorytmu



4. Dane testowe

Do sprawdzenia poprawności działania algorytmu oraz do dostrojenia wybrano następujący zestaw instancji:

1. tsp_10.txt, http://jaroslaw.mierzwa.staff.iiar.pwr.wroc.pl/pea-stud/tsp/;
2. gr431.tsp,
3. lin318.tsp, http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsp/;

Do wykonania badań wybrano następujący zestaw instancji:

1. tsp_10.txt, http://jaroslaw.mierzwa.staff.iiar.pwr.wroc.pl/pea-stud/tsp/;
2. gr17.txt,
3. gr24.tsp,
4. danzig42.tsp,
5. kroA100.tsp,
6. gr137.tsp,
7. gr202.tsp,
8. lin318.tsp,
9. gr431.tsp, http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsp/;
10. ft53.atsp,
11. ft65.atsp,
12. ft71.atsp, http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/atsp/

Pomiary czasu zostały wykonane na komputerze o specyfikacji:

- procesor Ryzen 5 2600X, częstotliwość taktowania dynaminczna 3.6 – 4.2 GHz

- pamięć RAM G.Skill Ripjaws V 16GB 3066MHz

- środowisko programowania C.Lion 2022.1

- system Windows 11

Czas wykonywania algorytmu został zmierzony w mikrosekundach za pomocą funkcji bilbioteki chrono:

```
std::chrono::high_resolution_clock::time_point start = std::chrono::high_resolution_clock::now();
```

Rysunek 1. Start funkcji mierzącej czas wykonywania algorytmu

```
std::chrono::high_resolution_clock::time_point stop = std::chrono::high_resolution_clock::now();
```

Rysunek 2. Koniec funkcji mierzącej czas wykonywania algorytmu

Podczas testowania działania algorytmu komputer nie był w ogóle używany, w związku z czym wszelkie anomalie powinny zostać ograniczone do minimum.

5. Procedura badawcza

W przypadku algorytmu realizującego symulowane wyżarzanie, występowały istotne parametry początkowe algorytmu. Podawane były one na początku pliku *.ini*. Procedura badawcza polegała na uruchomieniu programu sterowanego plikiem inicjującym *.ini* (format pliku:

schemat chłodzenia;
metoda wyboru rozwiązania;
temperatura początkowa;
parametr schładzania;
długość epoki;
nazwa_pliku_wyjściowego;
nazwa_instancji liczba_wykonań rozwiązanie_optymalne;)

Treść pliku *.inil*:

2

1

1000

0.999999

1000

output.csv

10.txt 5 212

17.tsp 5 2085

24.tsp 5 1272

42.tsp 5 699

53.txt 5 6905

65.txt 5 1839

71.txt 5 1950

100A.txt 5 21282

137.txt 5 69853

202.txt 5 40160

318.txt 5 42029

431.txt 5 171414

Każda instancja rozwiązana została 5 razy dla każdej z możliwych kombinacji parametrów wejściowych:

- schemat chłodzenia (Boltzmann lub geometryczna)
- metoda wyszukiwania rozwiązania w sąsiedztwie (2-zmiana oraz zamiana łuków)
- długość epoki (*1*, *100*, *10000*, *1000000* dla schematu Boltzmanna, *1*, *10*, *100*, *1000* dla schematu geometrycznego)
- temperatura początkowa (*2, 5, 10* dla schematu Boltzmanna, *1.5, 10, 100, 1000* dla schematu geometrycznego)
- parametr schładzania (*0.8, 0.99, 0.9999, 0.999999* dla schematu geometrycznego)

Temperatury początkowe oraz parametry schładzania dla odpowiadających im metod ochładzania zostały wybrane metodą empiryczną, poprzez obserwację długości wykonywania programu oraz jakości otrzymanego rozwiązania dla wielu różnych wartości tych parametrów.

Do pliku wyjściowego *output.csv* zapisywane były kolejno:

- Ilość wierzchołków problemu (nazwa pliku bez rozszerzenia); rozwiązanie optymalne

- schemat chłodzenia(1. Boltzmann, 2. Geometryczna);

- metoda wyszukiwania rozwiązań (1. 2-Zamiana, 2. Zamiana łuków);

- temperatura początkowa;

- parametr schładzania;

- długosc epoki;

- odnaleziona minimalna ścieżka; czas rozwiązywania problemu;

Plik wyjściowy zapisywany był w formacie csv. Poniżej przedstawiono fragment zawartości pliku wyjściowego

```
FILE/NUMBER OF VERTICES:10
Cooling method:
1                    Best 212
Swap method: 1
Start temperature: 2
Era length: 1
        425        12
        337         7
        331         7
        359         7
        397         7
```

Wyniki opracowane zostały w programie MS Excel.

6. Wyniki i analiza wyników

Wyniki zgromadzone zostały w pliku *output.csv*.

1. Wpływ sposobu wyboru temperatury początkowej

Przetestowałem algorytm z różnymi temperaturami dla obu schematów schładzania. Temperatury maksymalne zostały wybrane metodą prób i błędów, przy czym najważniejszym kryterium był czas działania algorytmu.

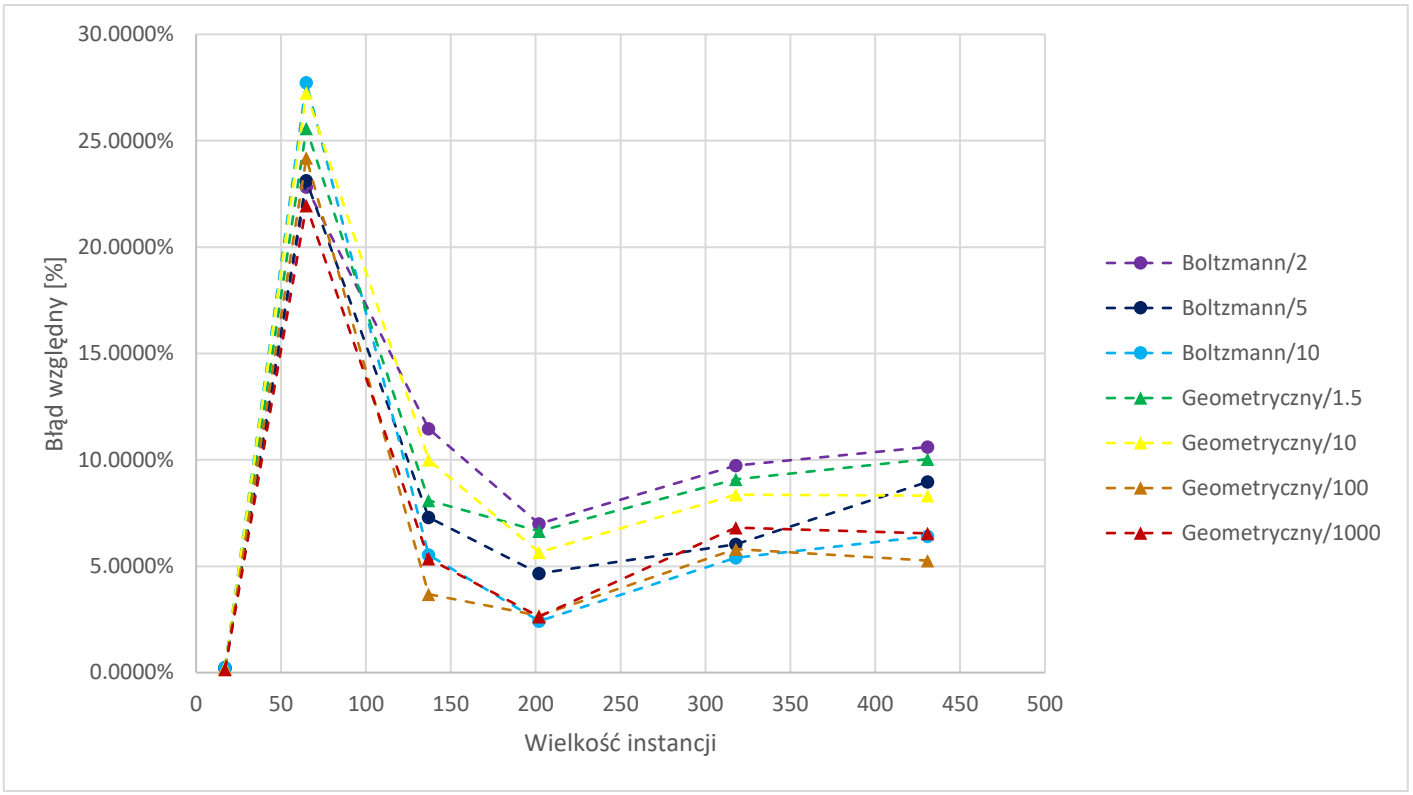
Z publikacji na temat symulowanego wyżarzania w internecie (strona 498, https://www.scielo.org.mx/pdf/cys/v21n3/1405-5546-cys-21-03-00493.pdf) wyczytałem, że w przypadku algorytmu Boltzmanna najlepszym rozwiązaniem jest ustawianie jak najmniejszej temperatury ponieważ czas działania algorytmu jest bardzo zależny od temperatury początkowej. Z tego powodu w testach użyłem wartości temperatur *2, 5* i *10*

Do porównania wpływu wyboru temperatury początkowyej wybrane zostały największe z pozostałych parametrów testowych, tj. ***L*** *= 1000000* dla schematu Boltzmanna, oraz ***L*** *= 1000,* ***α*** *= 0.999999* dla schematu geometrycznego. Porównywane instancje to gr17.tsp, ft65.atsp, gr137.tsp, gr202.tsp, lin318.tsp oraz gr431.tsp. Metoda wyszukiwania rozwiązania dla większości plików to zamiana łuków, plik ft65.atsp został zawarty w porównaniu z użyciem metody 2-zamiany z powodu niewystarczająco dobrych wyników przy użyciu drugiej metody.

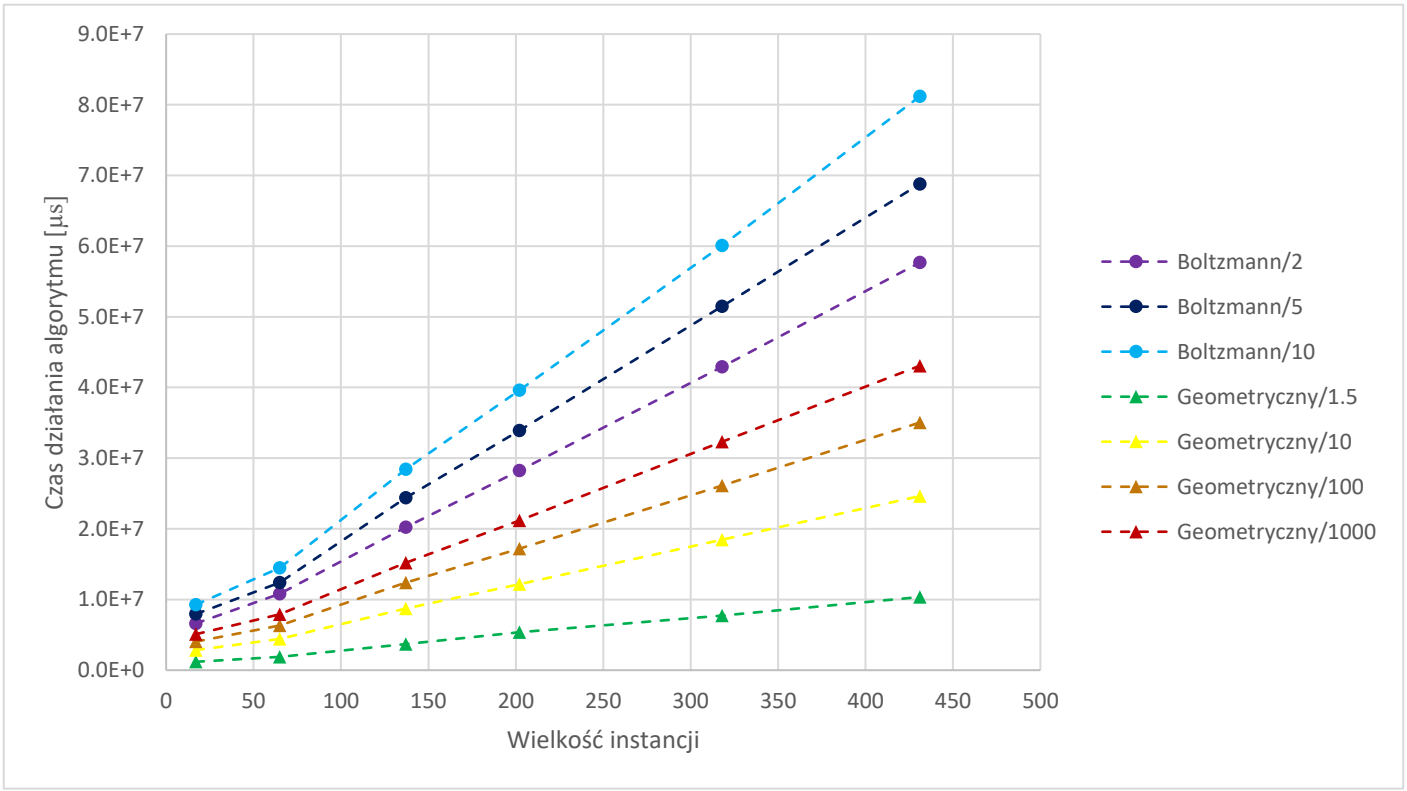
Jak widać na wykresach 1. i 2. (strona 12), najlepsza jakość rozwiązania dla symetrycznych problemów jest wyliczana dla kombinacji Boltzmann/5 oraz geometryczny/100 i geometryczny/1000. Z tych trzech kombinacji, zarówno najkrótszy czas działania oraz jakość wyniku daje nam schemat geometryczny z temperaturą początkową 100. Jest to najbardziej optymalna kombinacja ponieważ algorytm skutecznie wychodzi z minimów lokalnych, nie przyjmując przy tym rozwiązań dużo gorszych niż aktualne. Schemat Boltzmanna z drugiej strony przyjmuje zbyt niskie temperatury początkowe, w związku z czym trudniej jest mu opuścić minima lokalne i skutecznie przeglądać przestrzeń rozwiązań w sąsiedztwie. Dodatkowo, złożoność czasowa algorytmu z zastosowaniem schematu Boltzmanna jest dużo większa niż z zastosowaniem schematu geometrycznego, i w przypadku zwiększenia temperatury rosłaby w szybszym tempie, co tylko przemawia na jego niekorzyść.

Tabela 1. Wartości zmierzone

Instancja	Schemat/temperatura początkowa	Średnia znalezionych rozwiązań	Średni czas wykonania algorytmu	% błędu
17	Boltzmann/2	2089.0	6611212.2	0.1918%
	Boltzmann/5	2090.0	7935509.4	0.2398%
	Boltzmann/10	2090.0	9270107.2	0.2398%
	Geometryczny/1.5	2090.0	1189282.0	0.2398%
	Geometryczny/10	2090.0	2841204.6	0.2398%
	Geometryczny/100	2089.0	4041128.6	0.1918%
	Geometryczny/1000	2088.0	5091681.4	0.1439%
65	Boltzmann/2	2258.8	10783087.2	22.8276%
	Boltzmann/5	2264.2	12387445.2	23.1213%
	Boltzmann/10	2349.0	14478202.6	27.7325%
	Geometryczny/1.5	2309.3	1861459.3	25.5755%
	Geometryczny/10	2340.2	4427828.8	27.2539%
	Geometryczny/100	2284.0	6316557.0	24.1979%
	Geometryczny/1000	2242.6	7901621.0	21.9467%
137	Boltzmann/2	77864.6	20208586.8	11.4692%
	Boltzmann/5	74949.6	24388811.4	7.2962%
	Boltzmann/10	73716.6	28443490.6	5.5310%
	Geometryczny/1.5	75510.8	3702014.8	8.0996%
	Geometryczny/10	76845.6	8719992.2	10.0105%
	Geometryczny/100	72421.8	12375248.8	3.6774%
	Geometryczny/1000	73594.4	15197253.8	5.3561%
202	Boltzmann/2	42970.0	28224311.8	6.9970%
	Boltzmann/5	42029.4	33912263.6	4.6549%
	Boltzmann/10	41134.2	39597726.4	2.4258%
	Geometryczny/1.5	42831.8	5362697.2	6.6529%
	Geometryczny/10	42428.0	12154426.4	5.6474%
	Geometryczny/100	41232.2	17187351.2	2.6698%
	Geometryczny/1000	41219.2	21183023.6	2.6375%
318	Boltzmann/2	46119.2	42910336.6	9.7319%
	Boltzmann/5	44563.0	51494266.2	6.0292%
	Boltzmann/10	44295.8	60072440.8	5.3934%
	Geometryczny/1.5	45845.2	7719756.2	9.0799%
	Geometryczny/10	45544.4	18460723.6	8.3642%
	Geometryczny/100	44472.0	26102053.2	5.8127%
	Geometryczny/1000	44895.8	32312781.6	6.8210%
431	Boltzmann/2	189597.2	57673310.4	10.6078%
	Boltzmann/5	186789.6	68780645.2	8.9699%
	Boltzmann/10	182396.0	81171686.0	6.4067%
	Geometryczny/1.5	188604.4	10349625.0	10.0286%
	Geometryczny/10	185669.6	24624871.0	8.3165%
	Geometryczny/100	180445.2	35033978.4	5.2686%
	Geometryczny/1000	182641.0	43066008.0	6.5496%



Wykres 1. Wykres zależności błędu od temperatury początkowej



Wykres 2. Wykres zależności czasu działania algorytmu od temperatury początkowej

2. Porównanie schematów chłodzenia

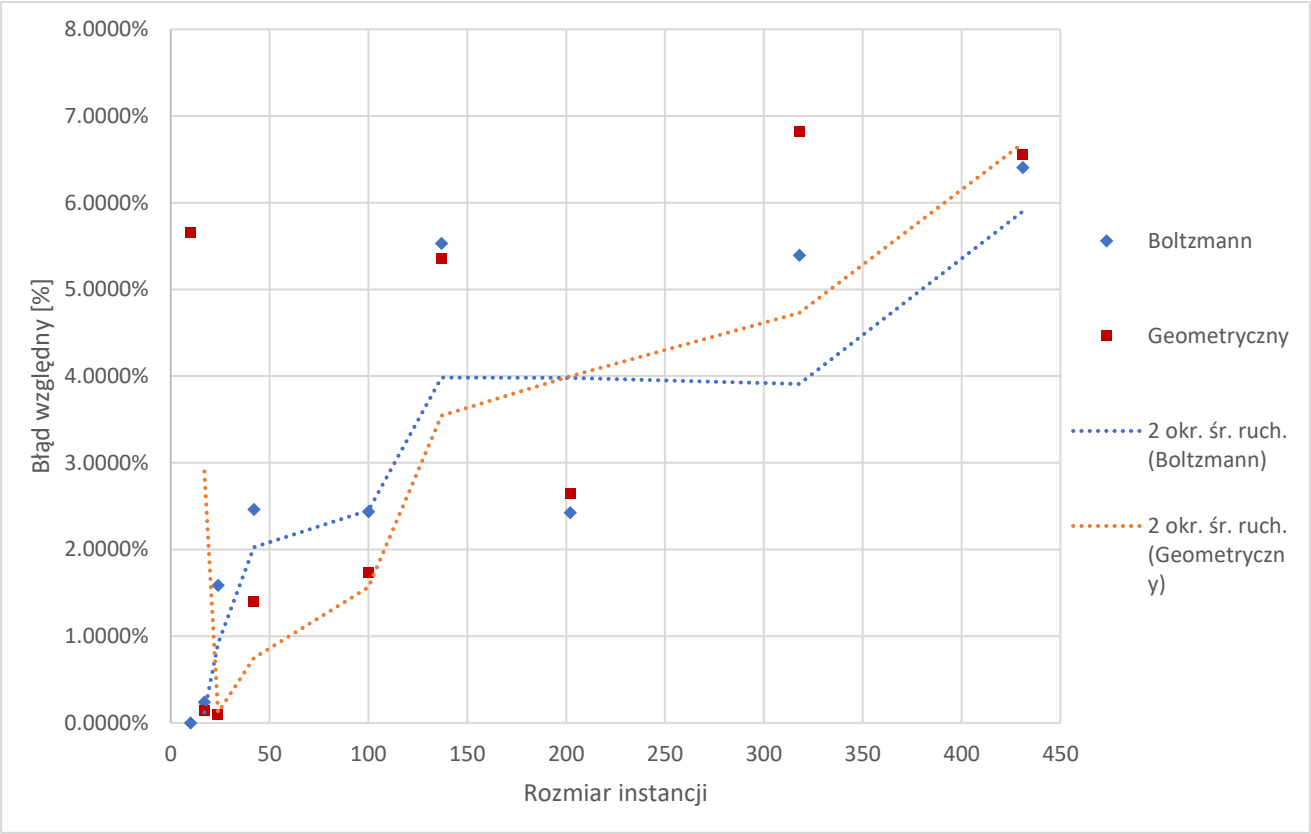
Do porównania schematów chłodzenia wybrane zostały największe z testowanych parametrów, tj. $L = 1000000$, $T_0 = 10$ dla schematu Boltzmanna, oraz $L = 1000$, $T_0 = 1000$, $\alpha = 0.999999$ dla schematu geometrycznego. Metoda zamiany to zamiana łuków dla większości instancji, natomiast dla problemów 53, 65 i 71 metoda zamiany to 2-zamiana.

Tabela 2. Wartości zmierzone (schemat Boltzmanna):

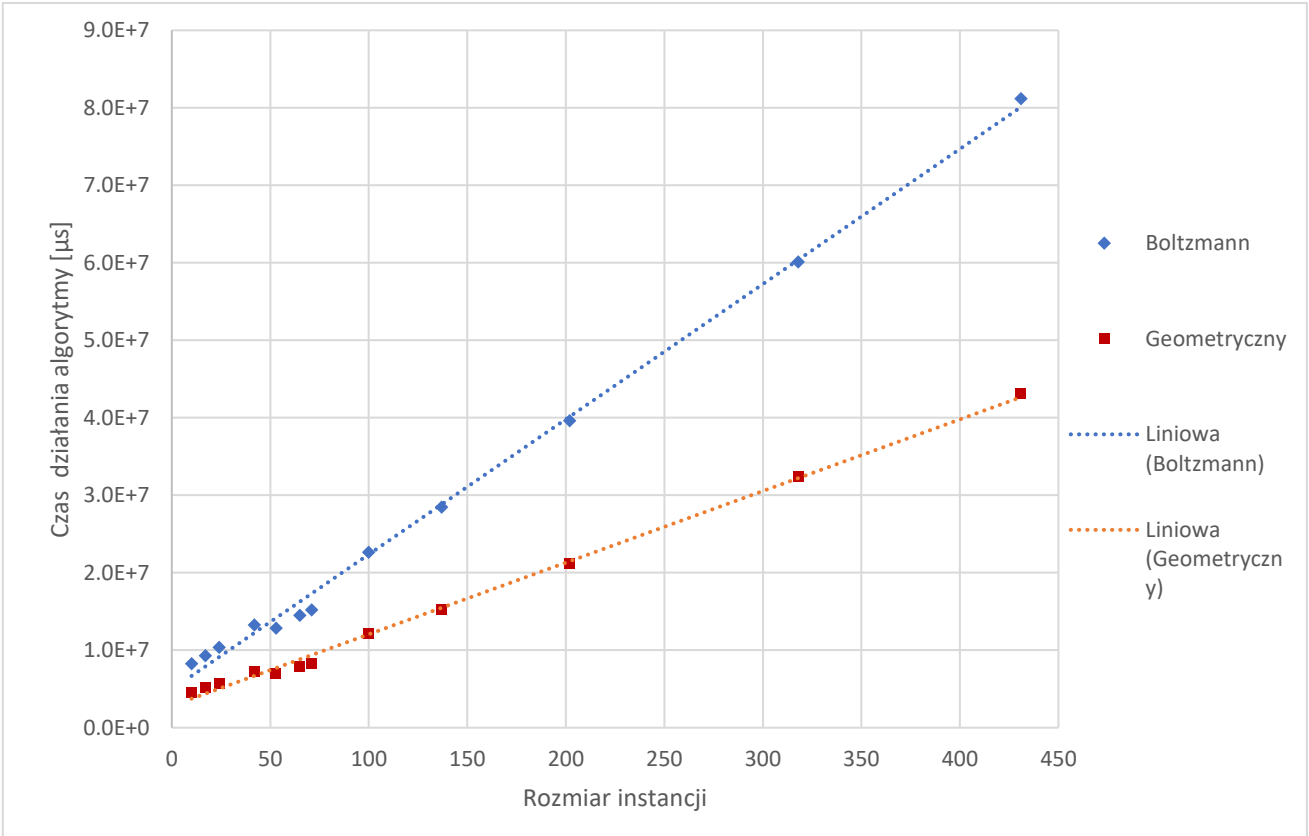
Instancja	Średnia znalezionych rozwiązań	Średni czas działania algorytmu [µs]	% błędu
10	212.0	8227274.2	0.0000%
17	2090.0	9270107.2	0.2398%
24	1292.2	10348559.4	1.5881%
42	716.2	13239864.8	2.4607%
53	8749.6	12842017.8	26.7140%
65	2349.0	14478202.6	27.7325%
71	2476.0	15196505.6	50.0606%
100	21800.6	22629294.2	2.4368%
137	73716.6	28443490.6	5.5310%
202	41134.2	39597726.4	2.4258%
318	44295.8	60072440.8	5.3934%
431	182396.0	81171686.0	6.4067%

Tabela 3. Wartości zmierzone (schemat geometryczny):

Instancja	Średnia znalezionych rozwiązań	Średni czas działania algorytmu [µs]	% błędu
10	224.0	4533341.8	5.6604%
17	2088.0	5091681.4	0.1439%
24	1273.2	5703073.4	0.0943%
42	708.8	7269087.4	1.4020%
53	8638.0	6996009.4	25.0978%
65	2242.6	7901621.0	21.9467%
71	2452.4	8298287.0	48.6303%
100	21650.2	12087248.4	1.7301%
137	73594.4	15197253.8	5.3561%
202	41219.2	21183023.6	2.6375%
318	44895.8	32312781.6	6.8210%
431	182641.0	43066008.0	6.5496%



Wykres 3. Zależność błędu procecntowego od rozmiaru isntancji dla różnych schematów schładzania



Wykres 4. Zależność czasu wykonywania algorytmu od rozmiaru isntancji dla różnych schematów ochładzania

W badaniu zależności błędu od rozmiaru instancji zostały pominięte instancje *53, 65* oraz *71*, ponieważ dawały one wyniki tak duże, że gdyby zestawić je na tym samym wykresie, to reszta wyników byłaby nieczytelna.

Na wykres zależności błędu od rozmiaru instancji naniesiona została linia trendu ruchomej średniej z dwóch okresów pomiarów, natomiast na wykres zależności czasu od rozmiaru instancji naniesiona zostala zwykła linia trendu.

Co do bardzo dużego wyniku instancji o wielkości *10* przy użyciu metody geometrycznej, musiała zajść sytuacja, w której algorytm już w początkowej fazie działania osiągnął optymalny wynik, lecz później jakiś wynik został zaakceptowany przez funkcję przyjmowania gorszych rozwiązań. Oznacza to, że stosowanie tak dużych parametrów początkowych nie jest konieczne dla małych instancji, a wynik bliski optymalnemu można byłoby znaleźć w dużo krótszym czasie.

Z wykresów 3. i 4. oraz z tabel 2. i 3. możemy zaobserwować, że metoda geometryczna znajduje średnio lepsze rozwiązania od metody Boltzmanna w przypadku instancji mniejszych od *137*. Dla większych isntancji to metoda Boltzmanna spisuje się lepiej. Nie jest to jednak wystarczająco dobre usprawiedliwienie tego, że dla schematu Boltzmanna algorytm średnio szukał rozwiązania prawie 2 razy dłużej.

W związku z powyższymi obserwacjami twierdzę, że użycie schematu schładzania Boltzmanna nie jest opłacalne z uwagi na prawie dwukrotnie dłuższy czas otrzymania rozwiązania, przy marginalnie lepszej jakości tego rozwiązania.

3. Porównanie sposobów wyboru rozwiązania w sąsiedztwie

Porównanie metod zamiany rozbiłem na 2 podpunkty: dla symetrycznych i asymetrycznych instancji problemu.

Asymetryczne instancje problemu to ft53.atsp, ft65.atsp i ft71.atsp. Wszyzstkie pozostałe to instancje symetryczne.

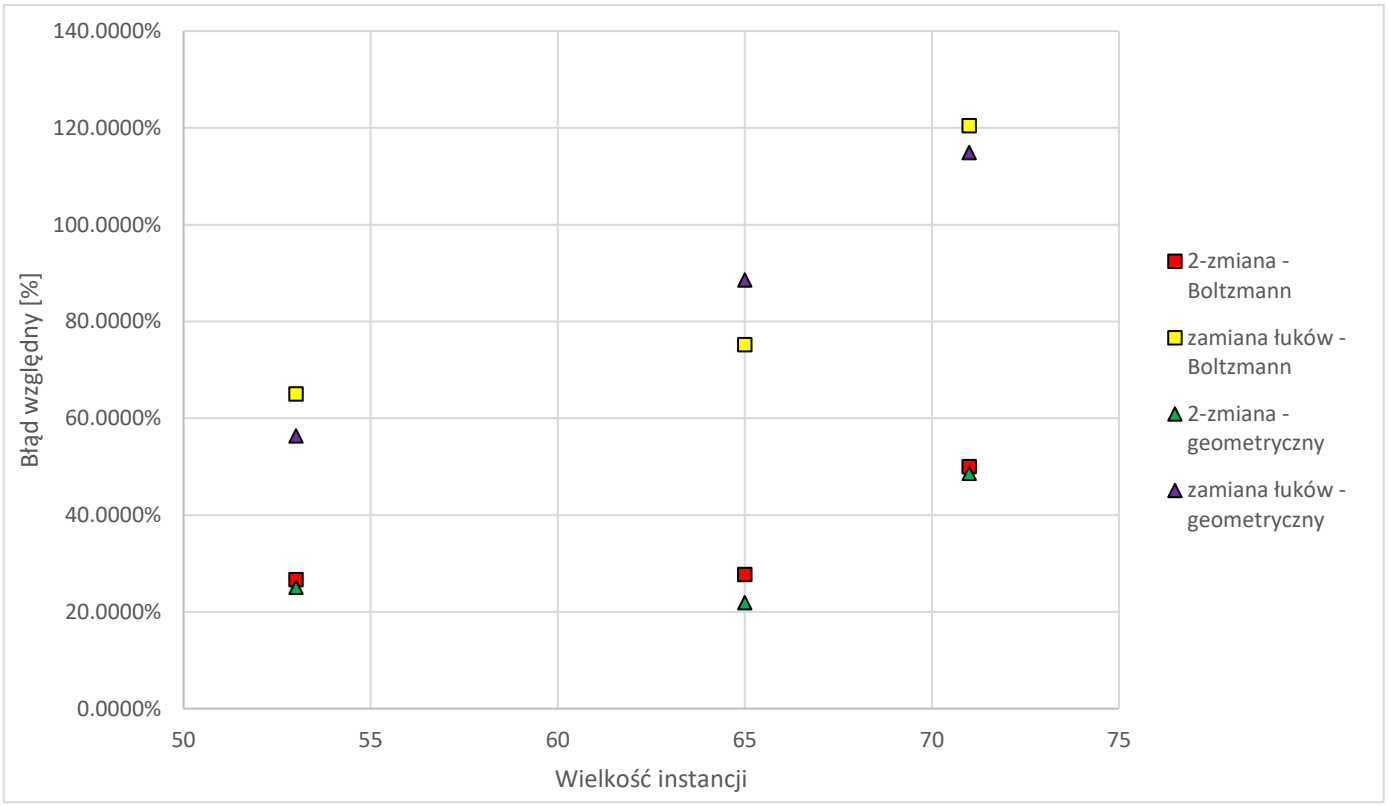
Porównanie zostało wykonane dla następujących parametrów: schemat Boltzmanna – ***L** = 1000000, T_0 = 10*; schemat geometryczny – ***L** = 1000, T_0 = 1000, **α** = 0.999999.*

Tabela 4. Wartości zmierzone (schemat Boltzmanna – obie metody zamiany - atsp)

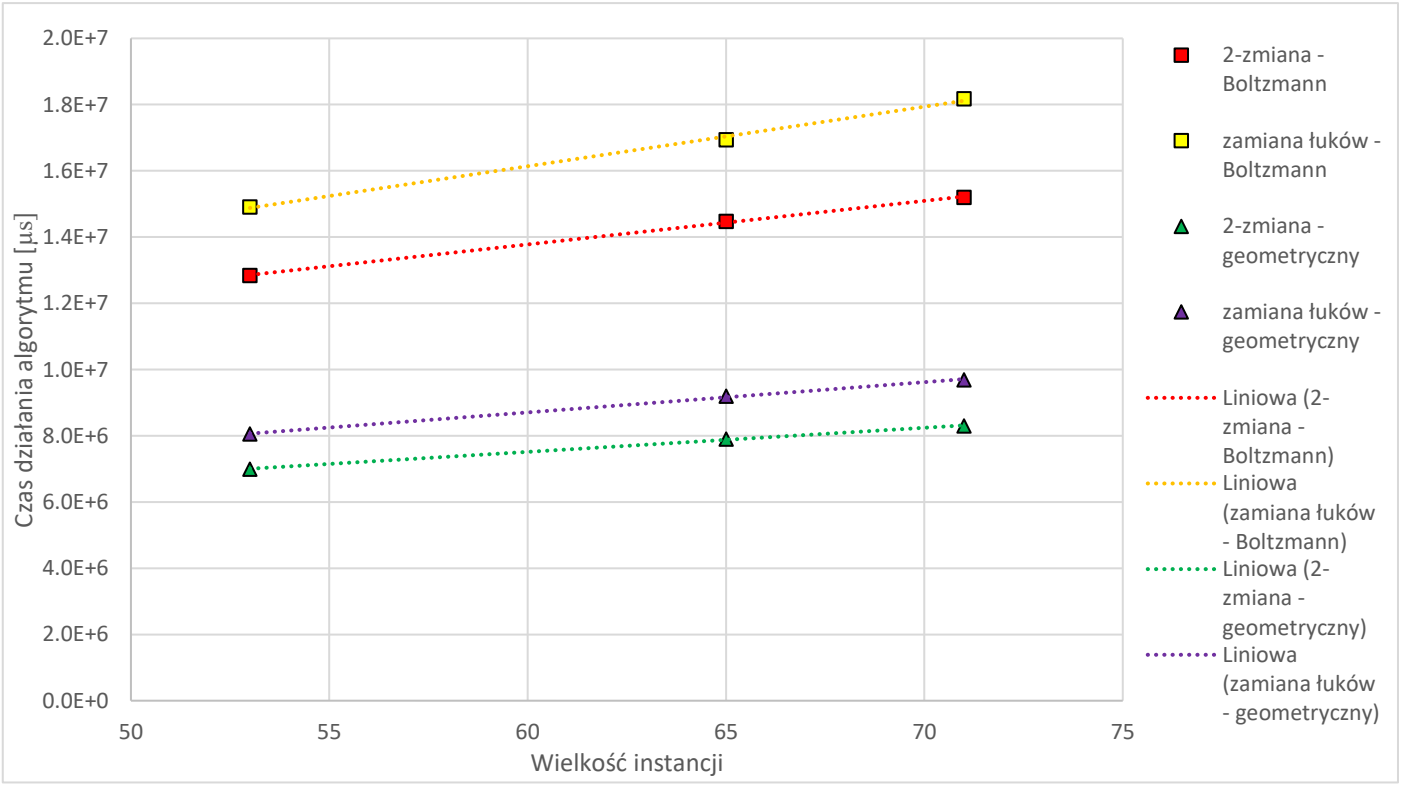
Metoda	Instancja	Średnia znalezionych rozwiązań	Średni czas działania algorytmu [µs]	% błędu
2zmiana	53	8749.6	12842017.8	26.7140%
	65	2349.0	14478202.6	27.7325%
	71	2476.0	15196505.6	50.0606%
zamiana łuków	53	11397.4	14911772.4	65.0601%
	65	3222.4	16944185.8	75.2257%
	71	3638.4	18179993.4	120.5091%

Tabela 5. Wartości zmierzone (schemat geometryczny – obie metody zamiany – atsp)

Metoda	Instancja	Średnia znalezionych rozwiązań	Średni czas działania algorytmu [µs]	% błędu
2zmiana	53	8638.0	6996009.4	25.0978%
	65	2242.6	7901621.0	21.9467%
	71	2452.4	8298287.0	48.6303%
zamiana łuków	53	10800.2	8053562.2	56.4113%
	65	3468.0	9197367.2	88.5808%
	71	3546.0	9686543.0	114.9091%



Wykres 5. Wykres zależności błędu procentowego od wielkości instancji dla różnych metod zamiany – problemy asymetryczne



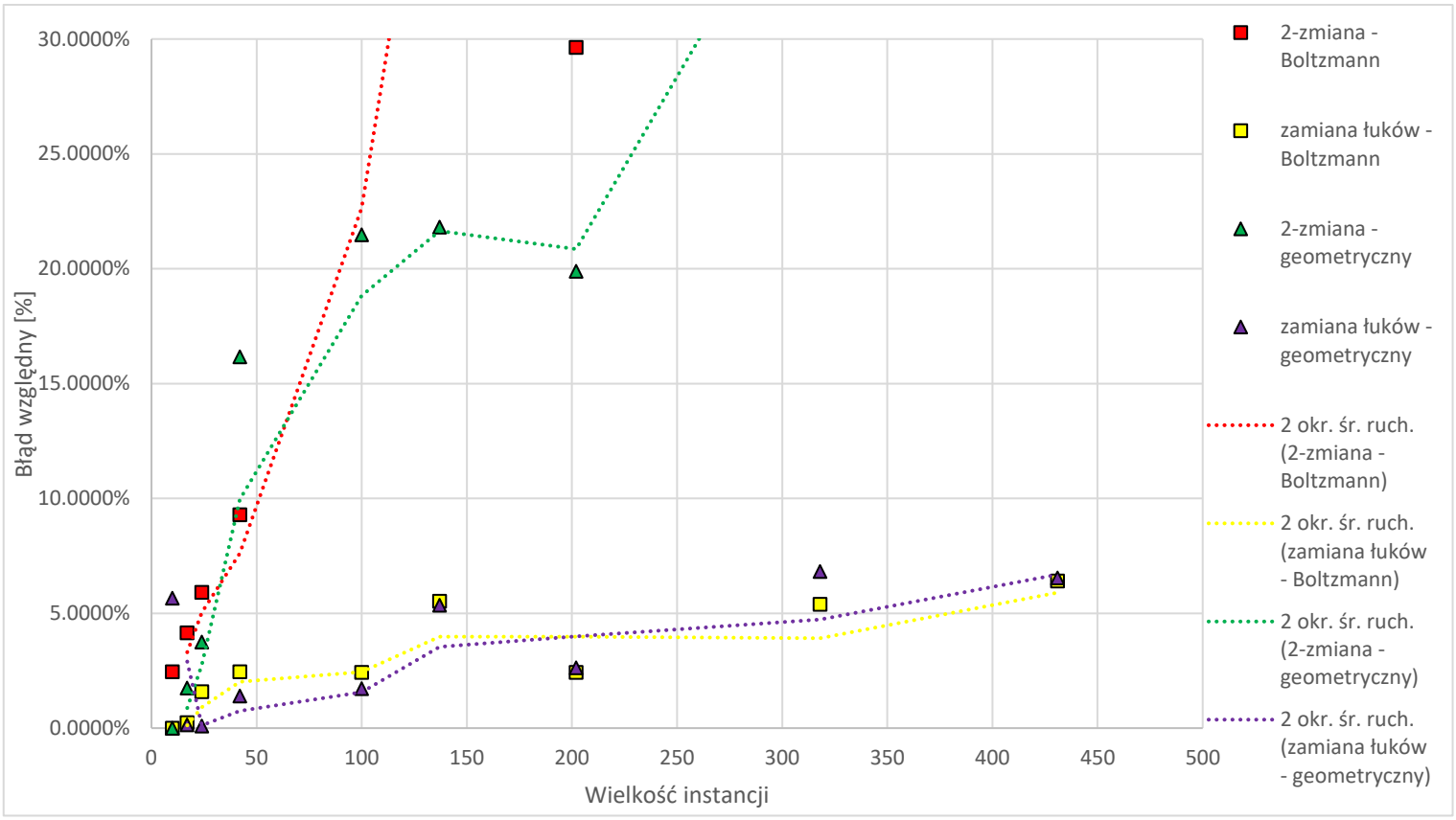
Wykres 6. Wykres zależności czasu wykonywania algorytmu od wielkości instancji dla różnych metod zamiany – problemy asymetryczne

Tabela 6. Wartości zmierzone (schemat Boltzmanna – obie metody zamiany - tsp)

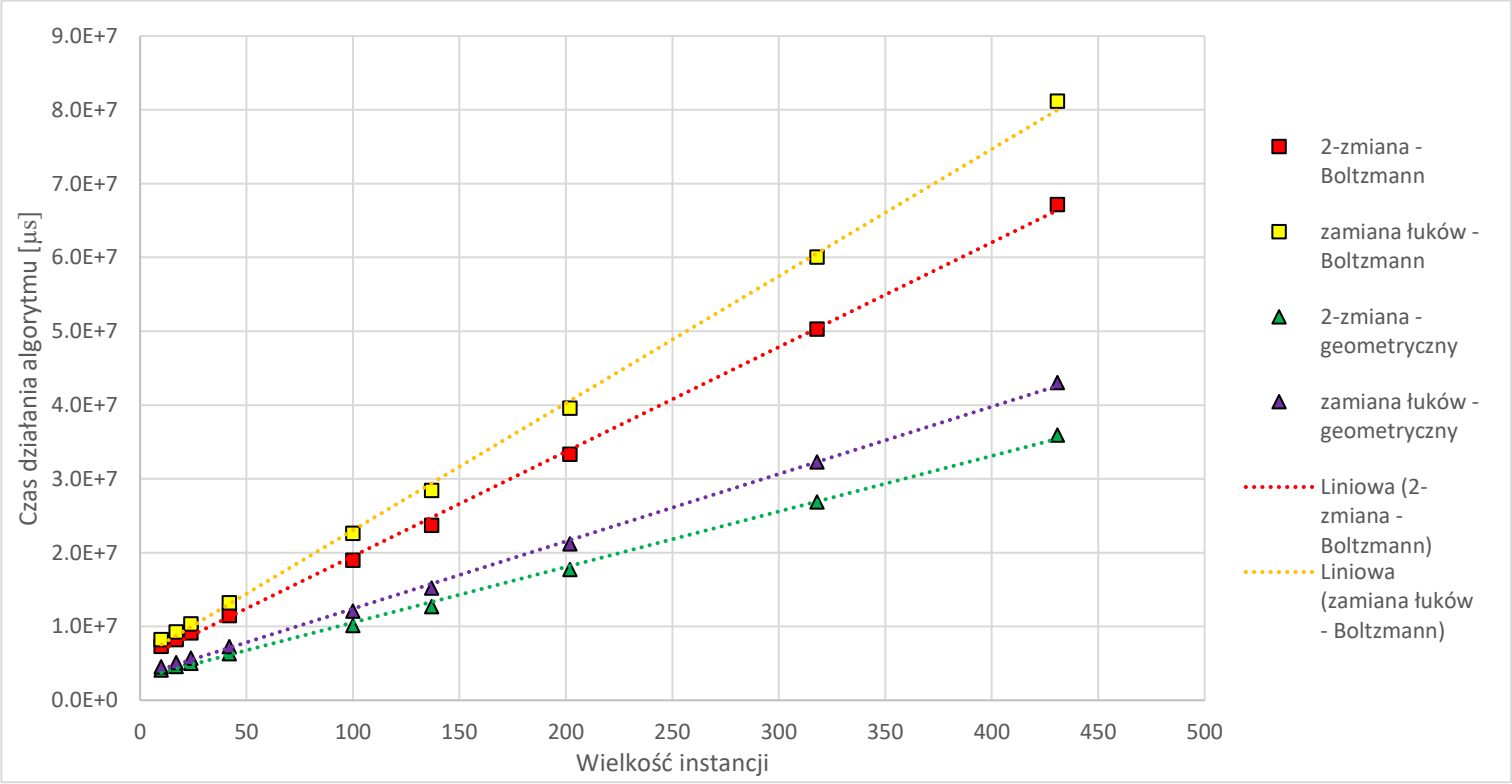
Metoda	**Instancja**	**Średnia znalezionych rozwiązań**	**Średni czas działania algorytmu [µs]**	**% błędu**
	10	217.2	7313702.2	2.4528%
	17	2171.6	8229546.0	4.1535%
	24	1347.2	9140566.4	5.9119%
	42	764.0	11493142.2	9.2990%
2zmiana	100	28941.8	18989463.4	35.9919%
	137	105653.8	23736643.6	51.2516%
	202	52063.8	33340027.6	29.6409%
	318	66883.8	50311091.6	59.1373%
	431	254227.4	67174192.8	48.3119%
	10	212.0	8227274.2	0.0000%
	17	2090.0	9270107.2	0.2398%
	24	1292.2	10348559.4	1.5881%
	42	716.2	13239864.8	2.4607%
zamiana łuków	100	21800.6	22629294.2	2.4368%
	137	73716.6	28443490.6	5.5310%
	202	41134.2	39597726.4	2.4258%
	318	44295.8	60072440.8	5.3934%
	431	182396.0	81171686.0	6.4067%

Tabela 7. Wartości zmierzone (schemat geometryczny – obie metody zamiany - tsp)

Metoda	Instancja	Średnia znalezionych rozwiązań	Średni czas działania algorytmu [µs]	% błędu
	10	212.0	7313702.2	0.0000%
	17	2121.4	8229546.0	1.7458%
	24	1319.8	9140566.4	3.7579%
	42	812.0	11493142.2	16.1660%
2zmiana	100	25853.0	18989463.4	21.4782%
	137	85093.0	23736643.6	21.8172%
	202	48151.4	33340027.6	19.8989%
	318	66451.4	50311091.6	58.1084%
	431	232121.0	67174192.8	35.4154%
	10	224.0	8227274.2	5.6604%
	17	2088.0	9270107.2	0.1439%
	24	1273.2	10348559.4	0.0943%
	42	708.8	13239864.8	1.4020%
zamiana łuków	100	21650.2	22629294.2	1.7301%
	137	73594.4	28443490.6	5.3561%
	202	41219.2	39597726.4	2.6375%
	318	44895.8	60072440.8	6.8210%
	431	182641.0	81171686.0	6.5496%



Wykres 7. Wykres zależności błędu procentowego od wielkości instancji dla różnych metod zamiany – problemy symetryczne



Wykres 8. Wykres zależności czasu wykonywania algorytmu od wielkości instancji dla różnych metod zamiany – problemy asymetryczne

Zdecydowałem się na osobne przeanalizowanie wyników dla problemów symetrycznych i asymetrycznych z powodu znacząco odbiegających od reszty wyników osiągniętych przy rozwiązywaniu instancji atsp.

Dla problemów asymetrycznych można zauważyć, że osiągane wyniki są dalekie od optymalnego, co pokazują tabele 4 i 5 oraz wykres 5. Metoda zamiany po łuku kompletnie nie sprawdza się w rozwiązywaniu tego problemu, o czym świadczą błędy rzędu 60-120%. Dla tych problemów, w przeciwieństwie do symetrycznych, dużo lepsza okazuje się 2-zmiana. Co prawda błąd rozwiązania instancji ft71 przekracza 30%, lecz dla mniejszych instancji ta metoda zamiany spisuje się całkiem dobrze.

Co do problemów symetrycznych, to różnica w rozwiązaniach jest już na korzyść metody zamiany łuków. Nawet dla największych testowanych instancji metoda ta osiąga błąd tylko o około 7% odbiegający od wartości optymalnej, co jest wynikiem bardzo dobrym. Porównując to z błędami metody 2-zamiany w okolicach 20% dla nawet mniejszych instancji, zdecydowanie widać przewagę metody zamiany łuków.

Jeśli chodzi o złożoność czasową obu tych metod, to możemy dostrzec różnicę rzędu 15-20% na korzyść 2-zamiany. Nie jest to jednak różnica na tyle drastyczna żeby przeważyć o zdecydowanej dominacji metody zamiany łuków.

Podsumowując, metoda zamiany łuków idealnie sprawdza się do problemów symetrycznych, ponieważ nawet dla dużych instancji daje nam bardzo dobre rozwiązania. Z kolei do rozwiązywania problemów asymetrycznych lepsza okazuje się metoda 2-zamiany – nie daje najlepszych wyników, ale mieści się w granicach 30% błędu, lecz tylko dla mniejszych instancji.

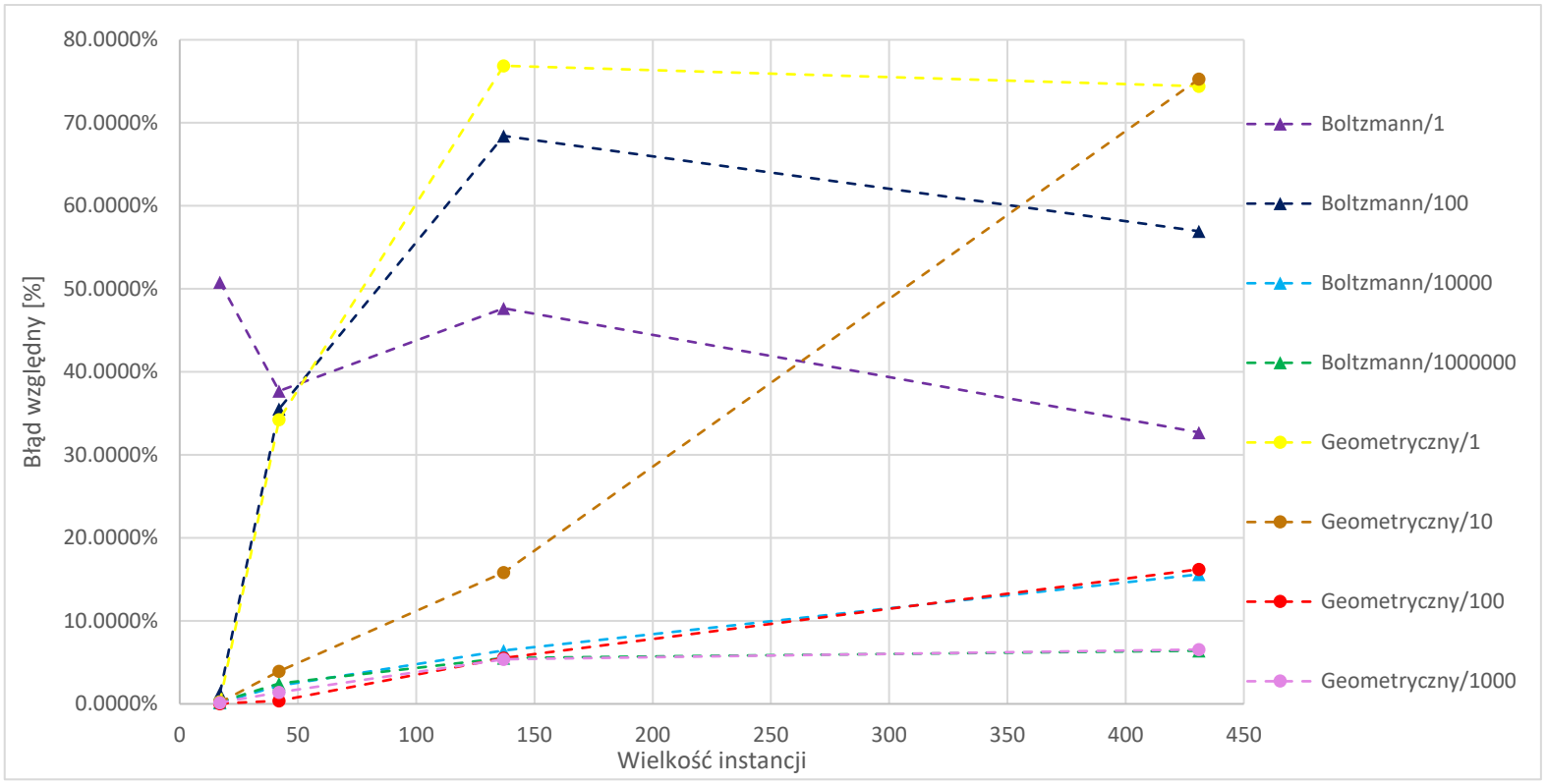
4. Wpływ długości epoki

Długosc epoki określa ile razy algorytm wybierze potencjalne nowe rozwiązanie z przestrzeni lokalnej przed ochłodzeniem temperatury.

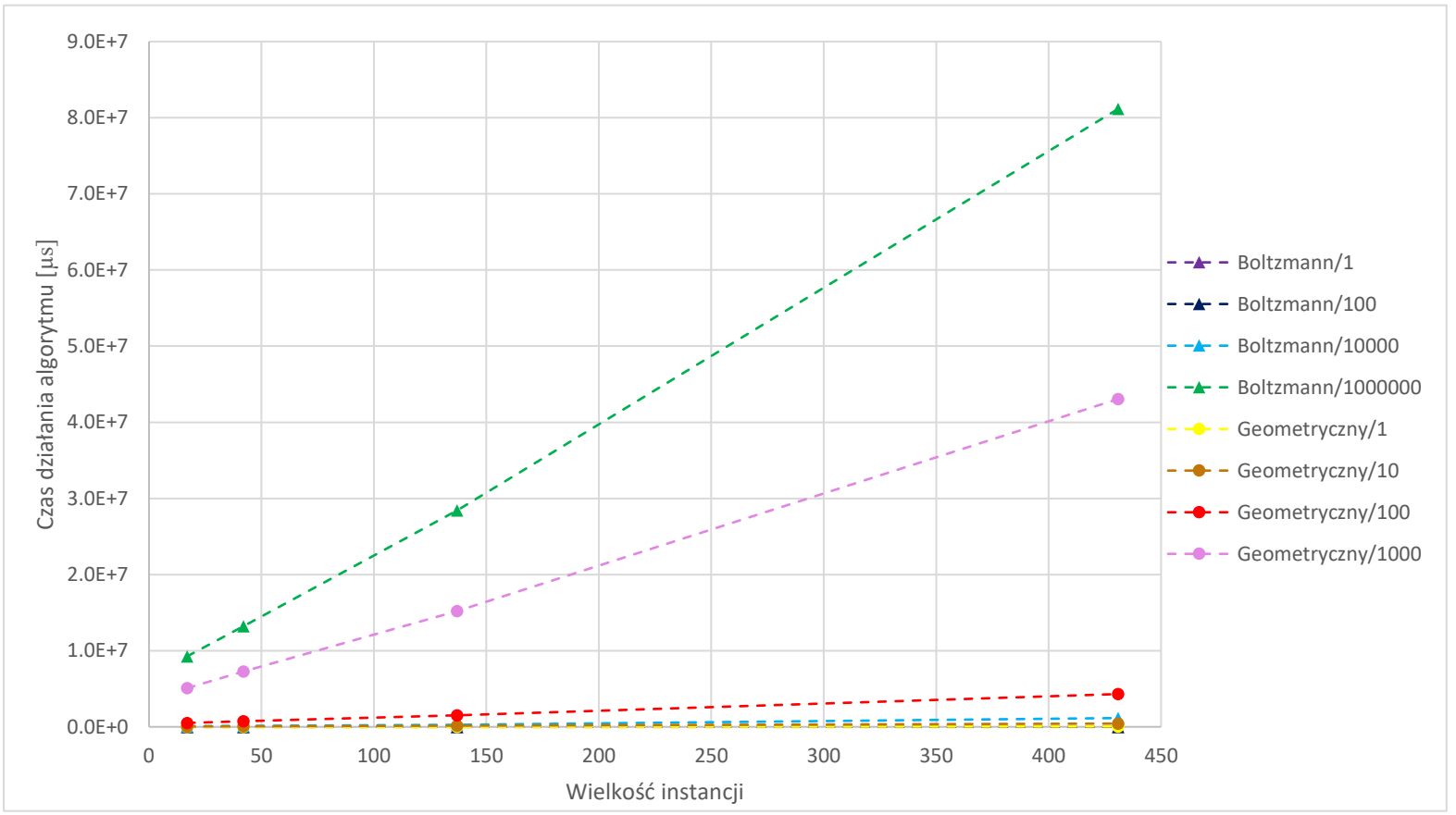
Do zbadania wpływu długości epoki użyłem następujących parametrów: schemat Boltzmanna – $T_0 = 10$, dla schematu geometrycznego – $T_0 = 1000$, $\alpha = 0.999999$. Dla obu schematów metoda zamiany to zamiana łuków. Zbadane instancje to gr17.tsp, danzig42.tsp, gr137.tsp i gr431.tsp

Tabela 8. Wartości zmierzone

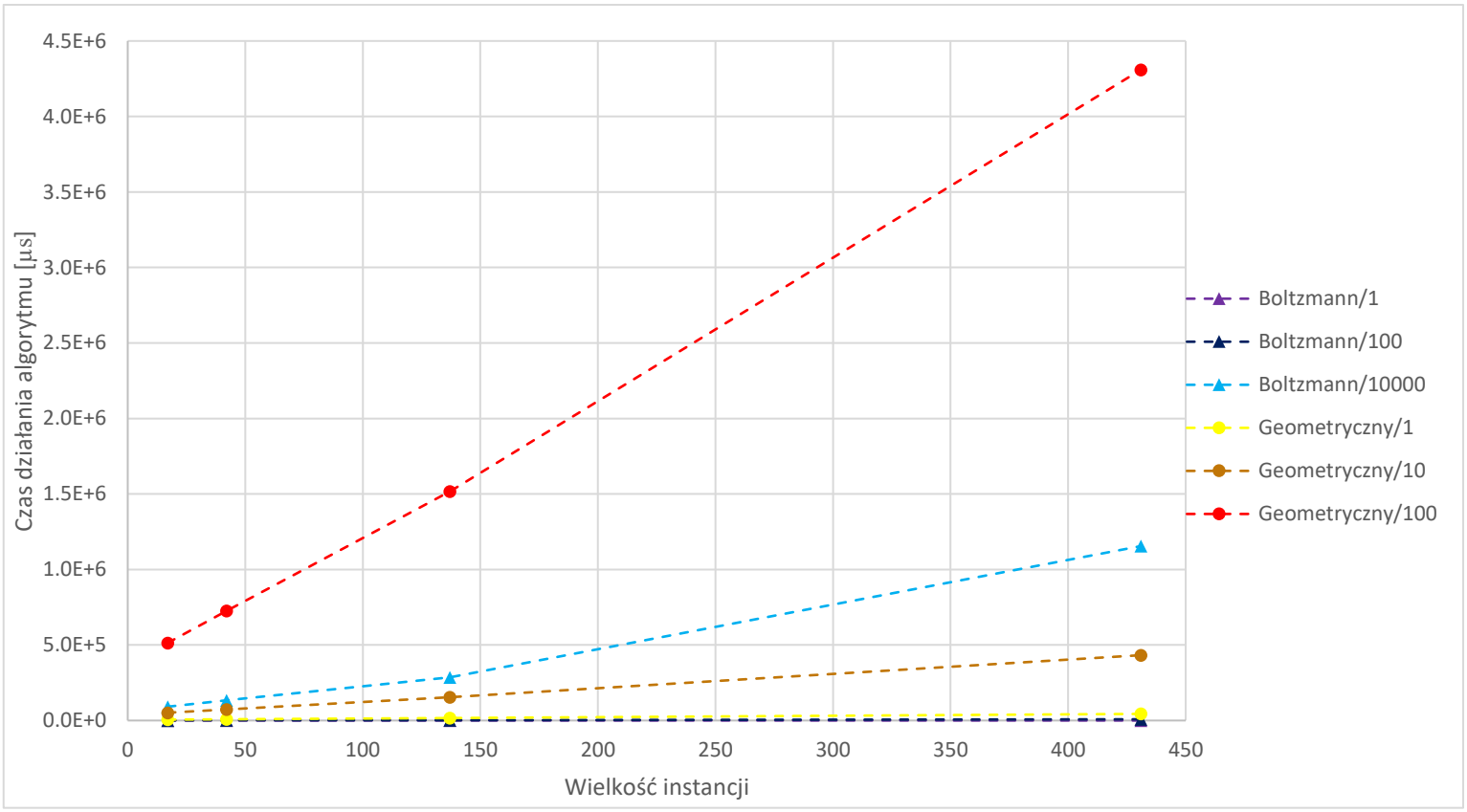
Instancja	Schemat/długość epoki	Średnia znalezionych rozwiązań	Średni czas wykonania algorytmu	% błędu
17	Boltzmann/1	3144.4	19.6	50.8106%
	Boltzmann/100	2111.8	915.8	1.2854%
	Boltzmann/10000	2088.0	91406.8	0.1439%
	Boltzmann/1000000	2090.0	9270107.2	0.2398%
	Geometryczny/1	2090.2	5569.8	0.2494%
	Geometryczny/10	2088.6	51629.8	0.1727%
	Geometryczny/100	2086.0	513878.6	0.0480%
	Geometryczny/1000	2088.0	5091681.4	0.1439%
42	Boltzmann/1	962.6	25.2	37.7110%
	Boltzmann/100	947.4	1352.6	35.5365%
	Boltzmann/10000	714.4	133669.0	2.2031%
	Boltzmann/1000000	716.2	13239864.8	2.4607%
	Geometryczny/1	938.4	7905.8	34.2489%
	Geometryczny/10	726.6	74076.2	3.9485%
	Geometryczny/100	701.6	725742.0	0.3720%
	Geometryczny/1000	708.8	7269087.4	1.4020%
137	Boltzmann/1	103155.2	47.4	47.6747%
	Boltzmann/100	117647.4	2896.4	68.4214%
	Boltzmann/10000	74354.8	286120.0	6.4447%
	Boltzmann/1000000	73716.6	28443490.6	5.5310%
	Geometryczny/1	123550.2	15998.4	76.8717%
	Geometryczny/10	80913.8	153820.0	15.8344%
	Geometryczny/100	73721.4	1517114.4	5.5379%
	Geometryczny/1000	73594.4	15197253.8	5.3561%
431	Boltzmann/1	227482.4	114.4	32.7093%
	Boltzmann/100	268990.8	8270.8	56.9246%
	Boltzmann/10000	198160.4	1155159.4	15.6034%
	Boltzmann/1000000	182396.0	81171686.0	6.4067%
	Geometryczny/1	298946.0	43802.6	74.4000%
	Geometryczny/10	300445.6	432661.0	75.2748%
	Geometryczny/100	199212.8	4309472.4	16.2173%
	Geometryczny/1000	182641.0	43066008.0	6.5496%



Wykres 9. Wykres zależności błędu procentowego od wielkości instancji dla różnych długości epoki – wybrane instancje



Wykres 10. Wykres zależności czasu wykonynwania algorytmu od wielkości instancji dla różnych długości epoki – wybrane instancje



Wykres 11. Wykres zależności czasu wykonywania algorytmu od wielkości instancji dla różnych długości epoki – wybrane instancje – widok przybliżony na dolną część wykresu 10.

Na wykresie 9. widać, że najlepsze wyniki osiągają metody z największą długością epoki, tj. Boltzmann/1000000 oraz geometryczny/1000. Szczególnie widoczne jest to dla większych instancji, gdzie te dwie metody dają rozwiązanie o co najmniej 10 punktów procentowych lepsze od innych.

Te algorytmy które uruchomione zostały z małą długością epoki z rozwiązaniem problemu poradziły sobie fatalnie. Dla instancji 137 tylko schemat geometryczny/10 dal wynik z błędem mniejszym niż 30%.

Przy podanych parametrach wejściowych, najbardziej sensownymi długościami epok są 1000000 dla schematu Boltzmanna i 1000 dla schematu geometrycznego, ponieważ produkują one najlepsze wyniki. Jeśli jednak wziąć pod uwagę wykresy zależności czasowej, mogę śmiało stwierdzić, że jeżeli czas działania algorytmu jest w jakiś sposób ograniczony to można również używać mniejszych wartości, odpowiednio 10000 i 100. Te długości epok również sprawdziły się dobrze, a co najważniejsze są znacznie szybsze, zwłaszcza przy użyciu schematu Boltzmanna.

7. Wnioski

Po przeprowadzeniu analizy wyników jestem w stanie wskazać, że obiektywnie najlepszymi dwoma konfiguracjami dla algorytmu symulowanego wyżarzania są, w zależności od schematu schładzania (po lewej parametry do geometrycznego, po lewej do Boltzmanna):

- schemat ochładzania: geometryczny/Boltzmann
- metoda zamiany: „zamiana łuków”
- temperatura początkowa: 1000/10
- współczynnik ochładzania: 0.999999/nie dotyczy
- długośc epoki: 1000/1000000

Dla tych metod średni błąd procentowy nie przekraczał zazwyczaj nawet 10%. Warto zwrócić uwagę, że dla mniejszych instancji nie ma potrzeby ustawiania tak wysokich parametrów początkowych.

Ważne jest również, że algorytm kompletnie inaczej zachowuje się dla problemów symetrycznych i asymetrycznych. Program napisany przeze mnie zdecydownie bardziej polecałbym do stosowania w przypadku problemów symetrycznych.

W algorytmie symulowanego wyżarzania głównym ograniczeniem nie jest pamięć, a czas. Ilość pamięci używanej przez algorytm była bardzo mała (około 40MB) i praktycznie nie zmieniała się na przestrzeni działania algorytmu. Najważniejszymi parametrami wpływającymi na długość działania program były temperatura początkowa, a przede wszystkim długość epoki. W przypadku schematu geometrycznego, jeśli długośc epoki zwiększyliśmy 10-ciokrotnie, to czas też rósł też 10-krotnie. Dla schematu logarytmicznego zależnośc ta wyglądała podobnie, z tym że widoczne to było na wykresach jako bardziej drastyczne zmiany, ponieważ testowane długości epoki nie były 10-krotnościami, tylko 100-krotnościami poprzednich.

Trudno jest oszacować złożoność czasową takiego algorytmu, jednak śmiało mogę stwierdzić że nie jest ona aż tak zależna od wielkości instancji tylko od właśnie – długości epoki i temperatury początkowej.