**Sprawozdanie z projektu z kursu Projektowanie Efektywnych Algorytmów**

Autor: Piotr Kołeczek

Numer albumu: 234940

Prowadzący projekt: Dr inż. Dariusz Banasiak

Grupa zajęciowa: środa 18:55 – 20:35

Termin oddania: 18.11.2019r.

Temat projektu: Implementacja i analiza algorytmu heurystycznego Symulowanego Wyżarzania rozwiązującego asymetryczny problem komiwojażera (ATSP).

1. **Wstęp.**

Rozważanym problemem projektowym jest programowa implementacja asymetrycznego problemu komiwojażera. Ów problem polega na wyznaczeniu takiego cyklu Hamiltona w grafie pełnym ważonym, którego koszt jest możliwie jak najmniejszy. Problem ilustrowany jest często jako problem podróżnika, który ma za zadanie przejść przez wszystkie miasta w krainie przebywając możliwie jak najkrótszą drogę (lub poświęcając na podróż jak najmniej czasu).

Problem projektowy możemy rozważyć w przypadku, gdy droga z miasta **A** do miasta **B** jest taka sama jak z miasta **B** do miasta **A** oraz kiedy w przeciwnym kierunku **może** mieć inną wartość. Ta różnica dzieli problem na symetryczny (STSP) oraz asymetryczny problem komiwojażera (ATSP).

Problem komiwojażera jest problemem, który zalicza się do tak zwanych problemów *NP-trudnych,* czyli takich, dla których nie wynaleziono algorytmów, które rozwiązują ów problem w sposób optymalny o wielomianowej złożoności obliczeniowej. Algorytmy, które wyszukują optymalny cykl Hamiltona w grafie dla większych instancji grafów są bardzo złożone obliczeniowo i są bardzo czasochłonne, dlatego też często wykorzystuje się algorytmy liczące przybliżoną wartość optymalnej drogi.

Niniejsze sprawozdanie dotyczy implementacji programowej algorytmu **Symulowanego Wyżarzania** dla asymetrycznego problemu komiwojażera.

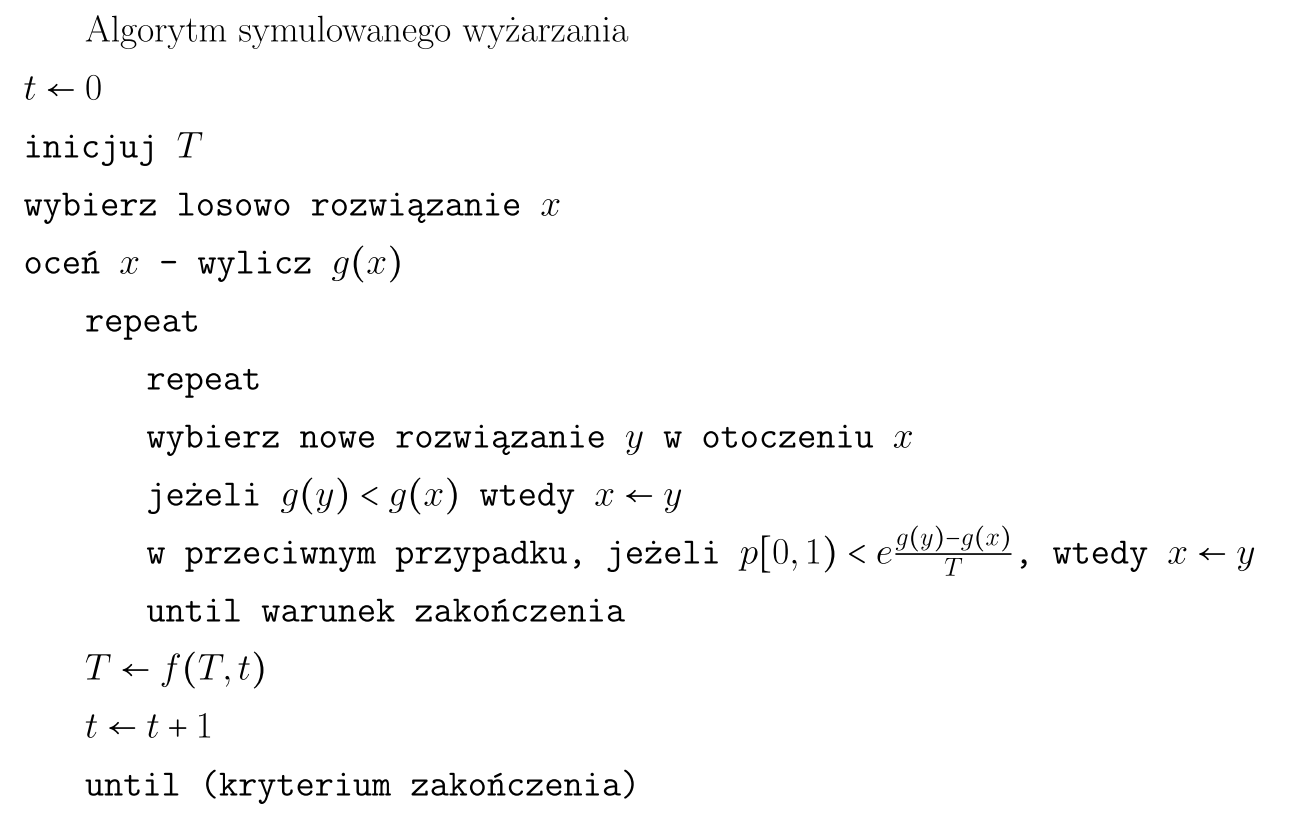
1. **Opis metody.**

Algorytm symulowanego wyżarzania jest przykładem algorytmu heurystycznego, czyli takiego, dla którego nie ma pewności znalezienia optymalnego rozwiązania. Wynikiem algorytmu jest w zdecydowanej większości wartość przybliżona rozwiązania optymalnego. Algorytm ma dążyć do znalezienia globalnego minimum dla funkcji wyniku rozwiązania problemu, w naszym przypadku problemu komiwojażera.

Algorytm ma swoje podłoże fizyczne i zostało zainspirowane procesem wyżarzania w metalurgii, gdzie metal jest stopniowo ochładzany, aby nie stracić pewnych zamierzonych właściwości fizycznych. Różni się od innych prostych metod iteracyjnych, które polegają na ciągłym ulepszaniu obecnego wyniku tym, iż czasem dla spełnionego warunku akceptacji może zostać przyjęte gorsze od obecnego wyniku rozwiązanie, aby wyjść z minimum lokalnego, które to z kolei może być bardzo oddalone od rzeczywistej wartości szukanego wyniku. Takie przyjęcie gorszego wyniku pozwoli algorytmowi wyjść z pewnym prawdopodobieństwem z minimum lokalnego w celu szukania bliższemu realnemu wynikowi. Oczywiście może mieć to czasem efekt odwrotny od oczekiwanego, czyli gorszy wynik będzie znajdował się w pobliżu jeszcze bardziej oddalonego od realnego wyniku minimum.

Algorytm ma jeden główny parametr, który wpływa na prawdopodobieństwo przyjęcia gorszego wyniku. Jest nim temperatura. Im wyższa, tym prawdopodobieństwo przyjęcia gorszego wyniku jest również większe. Dla coraz niższych temperatur algorytm jest coraz bardziej zbliżany w sposobie działania do typowych metod iteracyjnych.

Pod koniec pracy algorytmu temperatura jest tak niska, że gorsze rozwiązanie zostanie przyjęte z prawdopodobieństwem bliskim **zeru.**



Rysunek 1: Pseudokod algorytmu symulowanego wyżarzania dla problemu komiwojażera.

Powyższy pseudokod symulowanego wyżarzania implementuje tak zwaną **„stochastyczną wspinaczkę górską”.** Polega ona na tym, że cały algorytm uruchamia pętlę i ciągle przesuwa się w kierunku „rosnącej” wartości (*rosnącej* należy rozumieć jako tej *lepszej, poszukiwanej, bardziej pożądanej*). Pętla wykonująca algorytm kończy się, gdy zostanie osiągnięty „szczyt” (warunek zakończenia algorytmu, który jest iteracyjnie obliczany).

Wariant stochastyczny jest takim wariantem wspinaczki górskiej, która bierze pod uwagę istnienie wielu ekstremów. Takie podejście powoduje, że jest obliczane prawdopodobieństwo oscylacji rozwiązania wokół aktualnie najlepszego znalezionego rozwiązania. Dlaczego? Dlatego, że aktualna znaleziona wartość **może** być ekstremum **lokalnym,** a nie **globalnym.**

Taki sposób rozważania problemu można opisać zdaniem *„Jeśli wspiąłeś się na szczyt wyżyny, wspinaj się dalej”.*

Kryterium zakończenia działania algorytmu może być wybrane dowolnie wedle uznania. Przykładowymi warunkami zakończenia algorytmu symulowanego wyżarzania mogą być:

* Minięcie odgórnie określonej ilości czasu działania algorytmu,
* Wykonanie pewnej liczby iteracji,
* Spadek temperatury poniżej minimalnej temperatury wyżarzania,

1. **Opis najważniejszych klas w projekcie.**

**3.1. Klasa Graph.cs**

Jest to klasa reprezentująca załadowany graf do pamięci komputera. Posiada ona takie atrybuty jak macierz kosztów *costMatrix*, najlepszy koszt znalezionego cyklu Hamiltona *BestCycleCost*, liczba wierzchołków grafu *numOfCities*. Dzięki tej klasie możliwe jest załadowanie grafu z pliku tekstowego do programu.

**3.2. Klasa SimulatedAnnealing.cs**

Ta klasa odpowiada za implementację algorytmu Symulowanego Wyżarzania. Dziedziczy ona po klasie **Graph.cs** i dodatkowo definiuje takie atrybuty jak generator liczb pseudolosowych (*randomGenerator*) dla zasymulowania zjawiska fizycznego jakim jest wyżarzanie, współczynnik temperatury *temperatureCoefficient*, której wartość jest potrzebna do zmiany temperatury wyżarzania i przyjmuje ona wartość z zakresu **(0; 1]**, zmienna reprezentująca aktualną temperaturę *currentTemperature*, minimalną temperaturę wyżarzania *tMin*, która gdy zostanie osiągnięta wyżarzanie zostanie przerwane oraz pomocniczą zmienną, która reprezentuje wartość cyklu Hamiltona dla aktualnej permutacji wierzchołków grafu *tempCost*.

Najważniejsze metody to:

* **private double GenerateProbability()** – metoda, która jest odpowiedzialna za wyliczanie prawdopodobieństwa zgodnie z ukazanym na pseudokodzie wzorem (wykorzystując wcześnie wspomnianą *wspinaczkę stochastyczną*).
* **private double GenerateRandomProbability() –** metoda odpowiedzialna za wygenerowanie prawdopodobieństwa z przedziału [0; 1],
* **private void GeneratePermutation()** – metoda, która generuje permutację cyklu Hamiltona, przestawiając dwa losowo wybrane wierzchołki w cyklu,
* **private int GetPathLength(int[] indexMatrix)** – metoda, która oblicza koszt cyklu Hamiltona na podstawie tablicy, w której zawarte są indeksy wierzchołków grafu, po których kolejno przechodzi „komiwojażer”,
* **public void StartSA(double minTemperature, double maxTemperature, double tCoefficient) –** ta metoda jest programową implementacją symulowanego wyżarzania. Na początku tworzy pierwszy cykl Hamiltona w pętli, wybierając kolejno rosnący ciąg indeksów wierzchołków w grafie. Następnie w pętli wykonywane jest „wyżarzanie”, czyli wyznaczanie coraz to lepszych rozwiązań w postaci coraz to mniej kosztownych kosztów cyklu Hamiltona.

**3.3. Klasa Stack.cs**

Ta klasa jest programową implementacją struktury stosu. Zostały w niej zdefiniowane typowe metody dla tej struktury:

* **public void Push(int number)** – metoda, która dodaje na szczyt stosu nową liczbę,
* **public void Pop()** – metoda, która strąca ze stosu liczbę,
* **public void Clear()** – metoda usuwająca wszystkie elementy znajdujące się na stosie.

1. **Eksperyment.**

W procesie wykonywania pomiarów posłużyłem się precyzyjnym czasomierzem **Stopwatch,** który zawarty jest w przestrzeni **System.Diagnostics.**

Do wykonania eksperymentu posłużyłem się macierzami grafów:

* **br17.atsp** (39),
* **ft53.atsp** (6905),
* **ftv170.atsp** (2755),
* **rbg323.atsp** (1326),
* **rbg358.atsp** (1163),
* **rbg403.atsp** (2465),
* **dantzig42.tsp** (699),
* **brazil58.tsp** (25395),
* **gr120.tsp** (6942),

W nawiasach zostały zapisane optymalne koszty cyklu Hamiltona dla pliku.

Dla każdej, przetestowanej wartości N zostało wygenerowanych **100** testów, a następnie wyniki działania algorytmu zostały uśrednione (średni koszt uzyskanego cyklu Hamiltona). Następnie dla uzyskanego średniego wyniku został obliczony błąd względny pomiaru, który wyraża się wzorem:

|fzn – fopt| / fopt

gdzie:

fzn – najlepsza obliczona wartość podczas działania algorytmu,

fopt – wartość optymalna, będąca najlepszym znanym rozwiązaniem.

**4.1. Wyniki pomiarowe.**

Za parametry testowe dla **100** prób pomiarowych przyjęto początkową temperaturę wyżarzania **T0 = 1000** stopni, za minimalną **Tmin = 0.01** stopni, a współczynnik wyżarzania **Tcoefficient = 0.9999**. Poniżej zostały ukazane wyniki uzyskane podczas testów wyżej wymienionych macierzy dla tych parametrów:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nazwa pliku** | **Liczba wierzchołków grafu** | **Optymalny koszt cyklu Hamiltona** | **Uzyskana średnia waga cyklu** | **Błąd względny** | **Średni czas wykonywania algorytmu w [ms]** |
| **br17.atsp** | 17 | 39 | 39.00 | **0%** | 19.07 |
| **ft53.atsp** | 53 | 6905 | 8118.52 | **17.57%** | 28.81 |
| **ftv170.atsp** | 171 | 2755 | 6029.84 | **118.84%** | 66.22 |
| **rbg323.atsp** | 323 | 1326 | 1875.82 | **41.44%** | 123.94 |
| **rbg358.atsp** | 358 | 1163 | 1861.65 | **60.07%** | 137.77 |
| **rbg403.atsp** | 403 | 2465 | 3021.7 | **22.58%** | 156.27 |
| **dantzig42.tsp** | 42 | 699 | 818.8 | **17.14%** | 26.00 |
| **brazil58.tsp** | 58 | 25395 | 30882.78 | **21.61%** | 30.37 |
| **gr120.tsp** | 120 | 6942 | 10444.2 | **50.45%** | 49.99 |

Tabela 1: Wyniki testów dla 100 prób pomiarowych dla wskazanych macierzy.

Rysunek 1: Średni czas wykonywania algorytmu SA dla wykonanych 100 prób testowych.

Rysunek 2: Średni błąd względny dla 100 przeprowadzonych testów wskazanych macierzy.

Powyższy wykres można zinterpretować w taki sposób, że dla macierzy o liczbie wierzchołków **171 (ftv170.atsp)** istnieje bardzo niskie prawdopodobieństwo wybrania optymalnej ścieżki między dwoma wierzchołkami. Taka sytuacja sprawia, że w tej macierzy istnieje bardzo dużo cykli Hamiltona o bardzo pesymistycznych wagach. Dlatego też średni błąd względny dla tego grafu tak znacząco odbiega od pozostałych wyników.

Dla wszystkich badanych macierzy przeprowadzono jeden test, w którym ustalone zostały parametry wyżarzania na:

* Temperatura początkowa wyżarzania **T0 = 100000 stopni,**
* Temperatura końcowa wyżarzania **Tmin = 0.0001 stopni**,
* Współczynnik wyżarzania **Tcoefficient = 0.99999**.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nazwa pliku** | **Liczba wierzchołków grafu** | **Optymalny koszt cyklu Hamiltona** | **Uzyskana waga cyklu** | **Błąd względny** | **Czas wykonywania algorytmu w ms** |
| **br17.atsp** | 17 | 39 | 39 | **0%** | 372.21 |
| **ft53.atsp** | 53 | 6905 | 7865 | **13.92%** | 592.18 |
| **ftv170.atsp** | 171 | 2755 | 4492 | **63.05%** | 1341.06 |
| **rbg323.atsp** | 323 | 1326 | 1529 | **15.31%** | 2498.29 |
| **rbg358.atsp** | 358 | 1163 | 1386 | **19.17%** | 2773.57 |
| **rbg403.atsp** | 403 | 2465 | 2576 | **4.50%** | 3131.82 |
| **dantzig42.tsp** | 42 | 699 | 710 | **1.57%** | 511.98 |
| **brazil58.tsp** | 58 | 25395 | 26845 | **5.71%** | 603.80 |
| **gr120.tsp** | 120 | 6942 | 8080 | **16.40%** | 1019.20 |

Tabela 2: Wyniki testów dla dokładniejszej próby wyżarzania.

Rysunek 3: Czas wykonywania algorytmu dla dokładniejszej próby wyżarzania.

Rysunek 4: Błąd względny dla dokładniejszej próby wyżarzania.

Jak można zauważyć, procentowy błąd względny przy zastosowaniu lepszych parametrów wyżarzania radykalnie zmalał. Można odnotować też fakt, że macierzy o **171** wierzchołkach błąd względny dalej jest wysoki.

Wykonany został także pojedynczy pomiar testowy dla macierzy o **403** wierzchołkach (**rbg403.atsp**), ustawiając temperaturę początkową **T0 = 1000** stopni oraz minimalną temperaturę wyżarzania **Tmin = 0.01** stopni. Poniżej został ukazany wykres zmiany błędu względnego w zależności o współczynnika wyżarzania:

Tabela 3: Wyniki testów symulowanego wyżarzania dla różnych współczynników wyżarzania.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Wartość współczynnika wyżarzania** | **Błąd względny w [%]** | **Optymalna wartość cyklu Hamiltona** | **Uzyskana wartość cyklu Hamiltona** | **Czas wykonywania algorytmu w [ms]** |
| **0.5** | 219.68 | 2465 | 7880 | 0.031 |
| **0.6** | 215.58 | 2465 | 7779 | 0.042 |
| **0.7** | 208.68 | 2465 | 7609 | 0.057 |
| **0.8** | 204.75 | 2465 | 7512 | 0.087 |
| **0.9** | 189.01 | 2465 | 7124 | 0.18 |
| **0.99** | 120.85 | 2465 | 5444 | 1.68 |
| **0.999** | 62.84 | 2465 | 4014 | 15.94 |
| **0.9999** | 20.24 | 2465 | 2964 | 156.18 |
| **0.99999** | 6.00 | 2465 | 2613 | 1559.56 |
| **0.999999** | 0.65 | 2465 | 2481 | 15606.43 |

Rysunek 5: Wykres zmiany błędu względnego wyrażonego w {%} w zależności od zmiany współczynnika wyżarzania.

Rysunek 6: Czas wykonywania algorytmu symulowanego wyżarzania w zależności od zmiany współczynnika wyżarzania.

**4.2. Porównanie wyników z algorytmami dokładnymi.**

Dokonano porównania czasów znajdowania rozwiązań optymalnych algorytmu Brute Force, programowania dynamicznego oraz symulowanego wyżarzania. Przyjęto, że pomiar czasu wykonywania algorytmu symulowanego wyżarzania dobiegnie końca wtedy, gdy zostanie znaleziona właśnie optymalna wartość (warunek kończący wykonywanie algorytmu został zmodyfikowany w taki sposób, że jeśli znaleziony cykl Hamiltona przyjmuje optymalną wartość (która jest znana) to następuje przerwanie wykonywania algorytmu i zatrzymanie czasomierza). Zatem zmierzony czas dla symulowanego wyżarzania jest czasem, po którym zostało znalezione rozwiązanie **optymalne.**

Ponieważ badane instancje grafów są bardzo małe (mniejsze niż **30** wierzchołków grafu) przyjęto za parametry dla symulowanego wyżarzania wartości:

* **T0 = 100** stopni,
* **Tmin = 0.0001** stopnia,
* Współczynnik schładzania **Tcoefficient = 0.99999999**.

Dla przeglądu zupełnego skończono pomiary czasowe na macierzy o **13** wierzchołkach (**tsp\_13.txt**), ponieważ dla większych macierzy dla tego algorytmu czas wykonywania był zbyt długi. Z kolei dla programowania dynamicznego testy rozpoczęto od macierzy o **14** wierzchołkach (**tsp\_14.txt**), gdyż dla mniejszych instancji czasy wykonywania algorytmu były bliskie 0 sekund. Dla symulowanego wyżarzania zostały przeprowadzone testy dla wszystkich mniejszych instancji grafów.

Poniżej zostały zaprezentowane wyniki dla tych trzech algorytmów:

Tabela 4: Zależności czasowe między rozmiarem grafu, a czasem rozwiązywania TSP przy użyciu algorytmów BF, DP oraz SA.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nazwa pliku** | **Liczba wierzchołków grafu** | **Czas znajdywania optymalnego rozwiązania przez BF w [ms]** | **Czas znajdywania optymalnego rozwiązania przez DP w [ms]** | **Czas znajdywania optymalnego rozwiązania przez SA w [ms]** |
| tsp\_3.txt | 3 | 0.037 | --- | 0.55 |
| tsp\_4.txt | 4 | 0.04 | --- | 0.68 |
| tsp\_6\_2.txt | 6 | 0.053 | --- | 0.68 |
| tsp\_10.txt | 10 | 51.15 | --- | 5.06 |
| tsp\_12.txt | 12 | 6314.35 | --- | 280.07 |
| tsp\_13.txt | 13 | 80019.76 | --- | 488.48 |
| tsp\_14.txt | 14 | --- | 6.34 | 4051.01 |
| tsp\_15.txt | 15 | --- | 14.35 | 13907.9 |
| data16.txt | 16 | --- | 35.71 | 21179.55 |
| br17.atsp | 17 | --- | 95.51 | 688.51 |
| data18.txt | 18 | --- | 274.84 | 31936.44 |
| tsp\_20.txt | 20 | --- | 1415.76 | 49891.51 |
| gr21.tsp | 21 | --- | 3430.76 | 108.11 |
| tsp\_22.txt | 22 | --- | 7557.96 | 54485.85 |
| tsp\_23.txt | 23 | --- | 17656.16 | 56996.78 |
| gr24.tsp | 24 | --- | 41841.83 | 20190.59 |

Rysunek 7: Porównanie czasowe dla algorytmów przeglądu zupełnego, programowania dynamicznego oraz symulowanego wyżarzania w zależności od rozmiaru instancji grafu.

Uzyskane wyniki dla algorytmu symulowanego wyżarzania są bardzo ciekawe. Nadzwyczajne wyniki, które zostały uzyskane są dla macierzy **br17.atsp** (**17** wierzchołków), **gr21.tsp** (**21** wierzchołków) oraz **gr24.tsp** (**24** wierzchołki). Uzyskane takie czasy dla tych macierzy mają swoje uzasadnienie w specyfice grafów:

* Dla macierzy **br17.atsp** istnieją krawędzie łączące dwa wierzchołki o wadze **0**, a to sprawia, że istnieje kilka **optymalnych** cykli Hamiltona i to jednoznacznie przekłada się na wyższe prawdopodobieństwo wyliczenia optymalnego cyklu Hamiltona szybciej,
* Macierz **gr21.tsp** oraz **gr24.tsp** są macierzami, które reprezentują problem symetrycznego problemu komiwojażera – koszt dojścia z wierzchołka **A** do wierzchołka **B** ma taką samą wartość jak droga z wierzchołka **B** do wierzchołka **A**. Taka sytuacja sprawia, że także mamy wyższe prawdopodobieństwo obliczenia optymalnego cyklu Hamiltona szybciej.

**4.3. Analiza wydajności symulowanego wyżarzania.**

Jak widać wyniki otrzymane po wykonaniu algorytmu SA są zależne od liczebności instancji oraz parametrów zaimplementowanych w programie, którymi są współczynnik wyżarzania (schładzania), temperatura początkowa wyżarzania oraz minimalna temperatura wyżarzania. Zmiana współczynnika schładzania z **0.5** na **0.9** nie wnosi większej różnicy w jakości rozwiązania, natomiast dla wartości **0.99** i większych diametralnie poprawia się jakość wyniku. Jest to tak zwany *punkt przełamania*– różnica między wynikami ulega znacznemu uwidocznieniu. Przy zmianie współczynnika wyżarzania z **0.9** na **0.99** błąd względny badanej instancji problemu maleje prawie **70** punktów procentowych – jest to różnica ogromna. Oczywiście z perspektywy pomiarów, posiadając optymalny wynik problemu TSP, łatwo jest sprawdzić czy badany problem jest poprawny. Jeżeli się nie posiada takich danych, można odpowiednio ustawić parametry algorytmu (w szczególności badać wyniki problemu dla różnych wartości współczynnika wyżarzania), aby algorytm wykonywał się odpowiednio długo i potem sprawdzić czy uzyskiwane wyniki powtarzają się wielokrotnie. Jeżeli tak, to z pewną dozą ufności można założyć, w szczególności dla niewielkich grafów (do **40** wierzchołków), iż uzyskany wynik jest ścieżką **optymalną**, a zatem stanowi minimum globalne funkcji. Przykładowo – dla wykonanych powyżej testów dla macierzy o **17** wierzchołkach (**br17.atsp**) udało się uzyskać cykl optymalny w czasie **19** milisekund. Z kolei po zwiększeniu „jakości” parametrów wyżarzania (tj. zwiększeniu temperatury początkowej wyżarzania, temperatury minimalnej wyżarzania i **radykalnym** zbliżeniu współczynnika wyżarzania do wartości 1) dla macierzy o **403** wierzchołkach (**rbg403.atsp**)uzyskano wynik działania algorytmu, który jest zaledwie **0.65%** razagorszy od optymalnego przy dosyć zadowalającym czasie działania algorytmu (niecałe **16** sekund). To dowodzi temu, iż algorytm symulowanego wyżarzania jest bardzo efektywnym algorytmem do badania większych instancji grafów w Problemie Komiwojażera.

Wartym do zauważenia jest fakt, iż dla ustalonych wartości wyżarzania, notowane czasy mają przyrost **liniowy.** To jest dodatkowy atut tego algorytmu heurystycznego.

1. **Wnioski oraz podsumowanie.**

Zadanie projektowe miało na celu zapoznać studenta z tematyką algorytmów przeszukiwania lokalnego, jakimi są m.in. przeszukiwanie z zakazami (Tabu Search) oraz zaimplementowane symulowane wyżarzanie (Simulated Annealing).   
Algorytm symulowanego wyżarzania jest algorytmem, dla którego kluczowy jest dobór parametrów. W algorytmie przeglądu zupełnego (Brute Force) wystarczyło wczytać graf do pamięci komputera i obliczyć ścieżkę, identyczna sytuacja była w przypadku algorytmu programowania dynamicznego, a to ze względu na fakt, iż te algorytmy są algorytmami, które zwracają **zawsze** wynik **optymalny**, zatem gdy te algorytmy zakończą swoje działanie mamy stuprocentową pewność, iż uzyskane wyniki będą wynikami możliwie najlepszymi. Jednak mają one jedną, poważną wadę, która niemalże całkowicie przekreśla ich użyteczność – złożoność obliczeniowa. Algorytm Brute Force dla grafów o liczbie wierzchołków **13** i więcej jest już algorytmem bezużytecznym – liczba permutacji wierzchołków, która jest wymagana do sprawdzenia rośnie superwykładniczo (złożoność **O(n!)**).

Dla programowania dynamicznego sytuacja jest delikatnie lepsza, ponieważ zamysłem programowania dynamicznego jest podział problemu na wiele atomowych pod problemów i zapamiętywanie rozwiązań tych pod problemów – w przypadku tej metody możliwym było poprawienie szybkości znajdowania rozwiązania i ostatecznie zwiększenie liczby wierzchołków grafu do liczby **25.**

Dla symulowanego wyżarzania istnieje ogromne prawdopodobieństwo, że dla większych problemów (np. 100 wierzchołków i więcej) nie znajdziemy rozwiązania optymalnego w akceptowalnym czasie, jednakże możliwym jest uzyskanie wyniku co najmniej bardzo dobrego. Rozwiązując problem dla np. macierzy o **1000** wierzchołków, przy dobraniu odpowiednich parametrów działania algorytmu możliwe jest uzyskanie dosyć dobrego rozwiązania w akceptowalnym czasie.

Podsumowując – algorytm symulowanego wyżarzania jest bardzo dobrym algorytmem znajdującym rozwiązania przybliżone w akceptowalnym czasie. Implementując go do różnych, bardziej złożonych problemów (jakim jest na przykład Problem Komiwojażera) musimy się liczyć z tym, że uzyskiwane wyniki w znakomitej większości przypadków **nie będą** rozwiązaniami optymalnymi.