Projektowanie Efektywnych Algorytmów

Projekt

20/12/2022

259047 Szymon Leja

(4)Symulowane wyzarzanie

|  |  |
| --- | --- |
| Spis treści | strona |
| Sformułowanie zadania | 2 |
| Metoda | 3 |
| Schemat blokowy algorytmu | 5 |
| Dane testowe | 5 |
| Procedura badawcza | 6 |
| Wyniki | 9 |
| Analiza wyników i wnioski | 20 |
|  |  |
|  |  |

1. Sformułowanie zadania  
     
   Problem komiwojażera to problem, który polega na znalezieniu minimalnego cyklu Hamiltona w grafie, gdzie cykl Hamiltona to cykl w grafie, który odwiedza każdy wierzchołek tylko raz.   
     
   Można rozgraniczyć problem na dwa warianty w zależności od grafu opisującego problem – mianowicie, możemy rozpatrywać graf symetryczny, w którym przejście między wierzchołkiem A do wierzchołka B ma taki sam koszt jak przejście z wierzchołka B do wierzchołka A – symetryczny problem komiwojażera. Drugim wariantem, jest asymetryczny problem komiwojażera, koszt przejścia między dwoma wierzchołkami nie musi być taki sam.  
     
   W celu rozwiązania owego problemu stosuję się dwa typy algorytmów, dokładne i heurystyczne. Algorytmy dokładne w każdym wypadku zwrócą optymalną (minimalną) ścieżkę w grafie, natomiast czas wykonania takiego algorytmu zazwyczaj będzie bardzo duży i dla dużych instancji grafów wręcz niemożliwy lub po prostu nieopłacalny do wykonania. Algorytmy heurystyczne to grupa algorytmów, które nie dają pewności, że znaleziona ścieżka jest optymalna, natomiast czas wykonania algorytmu nawet dla dużych instancji jest znacząco niższy lub w ogóle możliwy do wykonania.  
     
   Zadanie polegało na opracowaniu, napisaniu i zbadaniu algorytmu symulowanego wyżarzania, który zalicza się do algorytmów heurystycznych. Złożoność algorytmu symulowanego wyżarzania, zależy od wartości, które zostaną dobrane do jego wykonania.
2. Metoda  
     
     
     
   Symulowane wyżarzanie jest algorytmem, który jest zainspirowany zjawiskami zachodzącymi w metalurgii – zwiększoną plastycznością metalu gdy ten jest bardziej rozgrzany. W algorytmie występuję parametr owej temperatury od której zależy czy gorsze uzyskane rozwiązanie zostanie przyjęte jako lepsze – im wyższa temperatura tym większa szansa na przyjęcie takiego wyniku, ma to na celu wyjście z minimum lokalnego naszego algorytmu.  
     
   Algorytm zaczyna rozwiązywanie problemu z losowym rozwiązaniem, jest wyliczany jego koszt i następnie przeszukuję przestrzeń lokalną w celu znalezienia bardziej optymalnego rozwiązania jedną z metod przeglądu sąsiedztwa. Jeżeli zostało napotkane lepsze rozwiązanie to zawsze zostaje uznane za najlepsze, jednak istnieję pewne prawdopodobieństwo, zależne od zadanej temperatury, że zostanie wybrane gorsze. Taki przegląd odbywa się w epoce, gdzie długość epoki to ilość takich przeglądów.  
   Następnie algorytm „wychładza” swoją temperaturę jedną z metod wychładzania.  
   Algorytm działa do czasu wychłodzenia do temperatury minimalnej.  
     
     
     
     
     
     
   Metody przeglądu sąsiedztwa zaimplementowane w algorytmie to:   
   **2-zmiana** – zamiana ze sobą dwóch wierzchołków, np:  
     
    1 => 2 => 3 => 4 => 5  
   zamienia się w:  
    **4**= > 2 => 3 => **1** => 5  
     
    **zamiana po łuku** – zamiana wierzchołków od początku do końca łuku o losowo wybranych końcach:  
    1 => 2 => 3 => 4 => 5  
   zamienia się w:  
    **4** => **3** => **2** => **1** => 5  
     
     
   Metody schładzania zaimplementowane w algorytmie to :  
   schładzanie **geometryczne –** opisane poniższym wzorem:  
     schemat chłodzenia **Boltzmanna**  
      
   **T** to temperatura, a **Ep** to numer epoki.

1: Podanie grafu, temperatury początkowej ***T0***, długości epoki ***Ep***  
2: Znalezienie losowego rozwiązania i wyliczenie jego kosztu - MK   
*3:* **Dopóki** *T0 > 1*

4: **POWTARZAJ** *Ep* razy

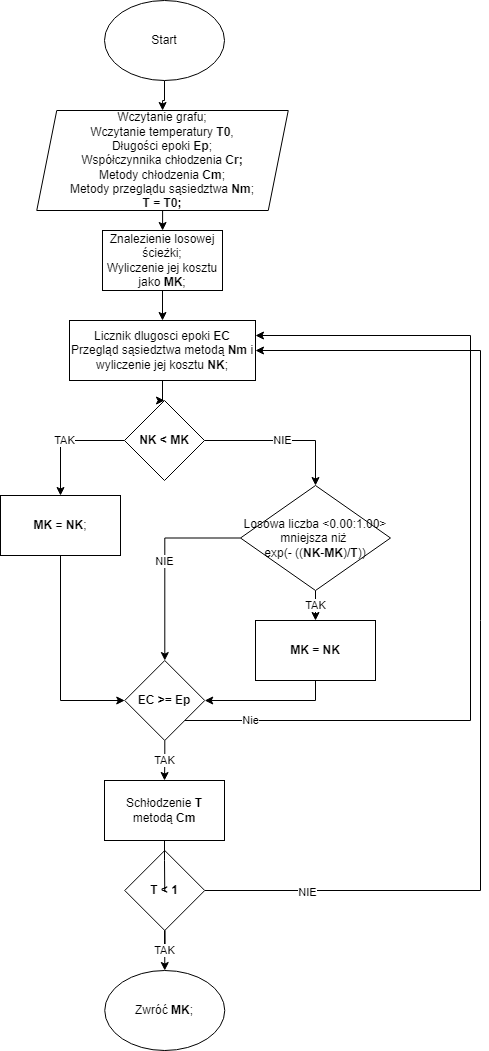
5: Przegląd sąsiedztwa wybraną metodą i wyliczenie nowego kosztu NK

6: **JEŻELI** *NK* < *MK* minimalne

7: *MK* = *NK*

8: **W P.P.** wylosuj liczbę między <0.0:1.0> i sprawdź czy jest mniejsza od jeśli tak to *MK* = *NK*

9: Obniż temperaturę wybraną metodą

1. Schemat blokowy algorytmu

1. Dane testowe  
     
   Do sprawdzenia poprawności algorytmu zastosowano następujący zestaw instancji:  
     
   - tsp\_17  
   - tsp\_26  
   - tsp\_34  
   - tsp\_39  
   - tsp\_45  
   - tsp\_53  
   - tsp\_65  
   - tsp\_70  
   - tsp\_71  
   - gr\_96.tsp  
   - kroA\_100.tsp  
   - krop\_124.atsp  
   - kroB\_150.tsp  
   - pr\_152.tsp  
   - ftv\_170.atsp  
   - kroB\_200.tsp  
   - rbg\_323.atsp  
   - u2319.tsp  
     
   Pochodzące ze strony :   
   http://jaroslaw.mierzwa.staff.iiar.pwr.wroc.pl/pea-stud/tsp/  
   http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/  
     
   Testy były wykonywane na platformie:  
     
   AMD Ryzen 5 5600X 4.5 GHz,   
   16GB RAM DDR4 3800MHz CL16,   
   Kingston M.2 PCIe Gen4 NVMe KC3000  
   Windows 11  
     
     
   Czasy zostały mierzone przy pomocy standardowej biblioteki chrono w języku C++
2. Procedura badawcza  
     
   Liczba plików testowych została ograniczona z powodu dużej ilości zmiennych, które były testowane. Testy sprawdzały czas, koszt i straty w zależności od liczby wierzchołków, współczynnika wychładzania, długości epoki i temperatury startowej.   
     
   Program został uruchomiony z plikiem konfiguracyjnym *config.ini*, zawierający procedurę testową – każda linia była w następującym formacie:  
     
   *<nazwa pliku>* <*ilość testów> <optymalny koszt>*  
   Zawartość pliku config.ini:  
   *tsp\_17.txt 30 39  
   tsp\_26.txt 30 937   
   tsp\_34.txt 30 1286   
   tsp\_39.txt 30 1530   
   tsp\_45.txt 30 1613   
   tsp\_53.txt 30 6905   
   tsp\_65.txt 30 1839   
   tsp\_70.txt 30 38673   
   tsp\_71.txt 30 1950   
   gr\_96.tsp 30 55209   
   kroA\_100.tsp 30 21282   
   krop\_124.atsp 30 36230   
   kroB\_150.tsp 30 26130   
   pr\_152.tsp 30 73682   
   ftv\_170.atsp 30 2755   
   kroB\_200.tsp 30 29437   
   rbg\_323.atsp 30 1326*  
     
   Dodatkowo został przetestowany jednorazowo plik:  
   u\_2319.txt 1 234256  
   Każdy plik został przetestowany w konfiguracji:  
   - 2-Zmiana + wychładzanie geometryczne  
   - 2-Zmiana + wychładzanie Boltzmanna  
   - Zmiana po łuku + wychładzanie geometryczne  
   - Zmiana po łuku + wychładzanie Boltzmanna  
     
   Dodatkowo, każdy taki plik był testowany zapętlony, ze zwiększającą się temperaturą początkową i współczynnikiem wychładzania, gdzie wzór na temperaturę początkową to   
   gdzie N to liczba wierzchołków, a I to liczba iteracji. Współczynnik wychładzania zmieniał swoją wartość na taką, ze zwiększoną dokładnością, np.:  
   Iteracja 1 => Współczynnik = 0.9  
   Iteracja 2 => Współczynnik = 0.99  
     
   Na początku testowania została przeprowadzony test kontrolny każdego pliku sprawdzający wyniki dla T = 1000 i Współczynnika Cr = 0.999.  
     
   W celu zbadania wpływu długości epoki na wyniki, dodawano 1/5 docelowej wartości długości epoki do aktualnej długości epoki – gdzie docelowa długość epoki to:  
   Każde uzyskanie kosztu dla wybranych przez program parametrów było wykonywane 30 razy w celu uzyskania średniej i wyeliminowania możliwości „anomalii” – testy wykonywały się ponad 27 godzin, uzyskano 3100 linii w excelu w formacie:  
   <Nazwa pliku> <Powtorzenia algorytmu> <Optymalny koszt> <Schemat chłodzenia> <Sposób zamiany> <Długość epoki> <Temperatura startowa> <Współczynnik wychładzania> <Czas n> <Koszt n> <Strata n> <Średnia strata> <Średni czas>  
     
   u\_2319.txt;1;234256;boltzmann;quick;0;46380;boltzmann;121433;890811;280.272;280.272;121433  
     
   Link do pliku excel:  
   https://drive.google.com/file/d/17a9Ao7pfWzTXfUttc82WotUQ3XPzvBex/view?usp=sharing
3. Wyniki

2. Straty w zależności od ilości wierzchołków dla T0= 1000 i Cr = 0.999, Ep=0

1.Czas wykonywania w zależności od ilości wierzchołków dla T0= 1000 i Cr = 0.999, Ep=0

4. Strata w zależności od ilości wierzchołków dla T0=700 \*N i Ep=0 - Boltzmann

3. Strata w zależności od ilości wierzchołków dla T0=700 \*N i Cr = 0.999999, Ep=0 - chłodzenie geometryczne

6. Czas wykonywania w zależności od ilości wierzchołków dla T0=700 \*N, Ep=0 i Cr = 0.999999 - Chłodzenie Geometryczne

5. Strata w zależności od ilości wierzchołków dla T0=700 \*N i Cr = 0.999999, Ep=0

7. Czas wykonywania w zależności od ilości wierzchołków dla T0=700 \*N, Ep=0 - Chłodzenie Boltzmann

8. Czas wykonywania w zależności od ilości wierzchołków dla T0=700 \*N, Ep=0 i Cr = 0.999999

10. Czas dla pliku rbg323.atsp dla Ep=10000 w zależności od T0

9. Czas dla pliku rbg323.atsp dla Cr=0.999, Ep=10000 w zależności od T0

12. Straty dla pliku rbg323.atsp dla Cr=0.999, Ep=10000 w zależności od T0

11. Straty dla pliku rbg323.atsp dla Cr=0.999, Ep=10000 w zależności od T0

15. Straty dla pliku rbg323.atsp dla Cr=0.999, Ep=10000 w zależności od T0

13. Czas dla pliku rbg323.atsp dla Cr=0.999, Ep=10000 w zależności od T0

16. Zależność czasu wykonywania dla pliku rbg323.atsp od Wspólczynnika wychładzania dla T0=1250 i Ep=10000

17. Zależność średniej straty dla pliku rbg323.atsp od Wspólczynnika wychładzania dla T0=1250 i Ep=10000

19. Zależność długości epoki do czasu dla pliku rbg323.atsp dla Cr= 0.999 i T0 = 1250

18. Zależność długości epoki do czasu dla pliku rbg323.atsp dla Cr= 0.999 i T0 = 1250

20. Zależność długości epoki do czasu dla pliku rbg323.atsp dla Cr= 0.999 i T0 = 1250

21.Zależność długości epoki do śr. straty dla pliku rbg323.atsp dla Cr= 0.999 i T0 = 1250

23. Zależność długości epoki do śr. straty dla pliku rbg323.atsp dla Cr= 0.999 i T0 = 1250

24. Zależność długości epoki do śr. straty dla pliku rbg323.atsp dla Cr= 0.999 i T0 = 1250

1. Analiza wyników i wnioski  
     
   Algorytm symulowanego wyżarzania działa znacząco szybciej od algorytmów, które zostały do tej pory zrealizowane. Jego złożoność zależy od dużej ilości zmiennych, takich jak temperatura początkowa, współczynnik wychładzania, długość epoki czy sam model wychładzania. Na powyższych wykresach widać, że każdy z parametrów istotnie wpływa na wynik i czas wykonania algorytmu, przy połączeniu odpowiednich parametrów możemy uzyskać dość dokładne wyniki nawet dla większych instancji. Dla pliku rbg323.atsp udało mi się uzyskać błąd na poziomie 12.91% przy czasie wykonania ok. 39 sekund przy parametrach Cr= 0.999999, Ep = 38760 i T0 = 226100.   
   Dla czytelności wykresów wyniki zostały przedstawione jedynie na największej testowanej instancji, ale wyniki dla pozostałych są analogiczne, zawierające inne błędy czy anomalię, które nie wpływają istotnie na analizę danych.  
     
   Złożoność obliczeniowa dla stałych parametrów wejściowych oprócz wielkości instancji złożoność wynosi O(n) (*wykres 1*), gdy bierzemy pod uwagę zmienną temperaturę początkową wynosi ona O(log(T0)) (*wykres 8*). Dodatkowo biorąc pod uwagę współczynnik wychładzania dla modelu chłodzenia geometrycznego złożoność wynosi O() (*wykres 16*)  
   Biorąc pod uwagę długość epoki złożoność obliczeniowa algorytmu wynosi O(Ep), jest liniowo zależna. (*wykres 20*)  
   Łącząc wszystkie parametry złożoność wynosi O().  
   Gdzie:  
    – wielkość instancji  
    – temperatura początkowa  
    – długość epoki  
    – współczynnik wychładzania  
     
   Analizując zastosowane metody wyboru rozwiązania w sąsiedztwie można zauważyć, że metoda 2-zamiany jest minimalnie szybsza i przynosi mniejsze straty, obydwie metody są metodami przeglądu sąsiedztwa typu greedy. (*Wykresy 9, 16, 24)*.  
     
   Zastosowane metody chłodzenia wyraźnie różnią się dokładnością i potrzebnym czasem do uzyskania rozwiązania. Metoda Boltzmanna jest wyraźnie szybsza, ale dużo mniej dokładna. Najlepszy wynik dla Boltzmanna dla pliku rbg323.atsp jaki udało mi się uzyskać to średnia strata na poziomie 47.98% w czasie ok. 1.03s, wyniki o takim samym błędzie udało mi się uzyskać również metodą geometryczną, ale w czasie ok. 4s – ale co ciekawe dla innych parametrów metoda geometryczna uzyskała błąd dla tego pliku ok. 13% w czasie ok. 3.9s. Metoda geometryczna ma więcej parametrów, więc ciężej znaleźć te optymalne, które dadzą dobry wynik w odpowiednim czasie. Skumulowane wyniki można sprawdzić pod linkiem podanym w sekcji procedura badawcza. (*Wykresy 24, 23, 21, 19*)  
     
   Na podstawie uzyskanych danych można stwierdzić, że najlepszym sposobem na wybór metody początkowej jest dostosowanie jej do wielkości instancji, w przypadku statycznej temperatury dla każdej instancji możemy zaobserwować, że wyniki dla coraz większych instancji i tej samej temperatury są coraz gorsze, a jeżeli uzależnimy jej wybór od wielkości instancji to wtedy błąd jest dużo mniejszy, kosztem czasu wykonania algorytmu.  
   Analogicznie do wyboru temperatury początkowej działa wybór współczynnika wychładzania dla metody geometrycznej, im większy jest tym mniejsze błędy uzyskujemy, kosztem czasu. *(Wykresy 1, 2, 3, 4)*  
     
   Długość epoki określa długość wewnętrznej iteracji pętli, dla wysokiej temperatury długa epoka może utrudnić nam wybór poprawnego rozwiązania, ponieważ będzie częściej akceptować te gorsze, natomiast dla mniejszych temperatur może ułatwić zadanie, ponieważ przed zaakceptowaniem rozwiązania będziemy wielokrotnie je sprawdzać z małą szansą na zaakceptowanie gorszego, w obu przypadkach czas działania algorytmu się zwiększa. *(Wykresy 18-24)*