

Sprawozdanie z laboratorium:
Metaheurystyki i Obliczenia Inspirowane Biologicznie

Część I: Algorytmy optymalizacji lokalnej, problem STSP

12 listopada 2017

Prowadzący: dr hab. inż. Maciej Komosiński

Autorzy: **Patryk Gliszczynski** inf117228 ISWD patryk.gliszczynski@student.put.poznan.pl
Mateusz Ledzianowski inf117226 ISWD mateusz.ledzianowski@student.put.poznan.pl

Zajęcia w środy, 15:10.

Oświadczam/y, że niniejsze sprawozdanie zostało przygotowane wyłącznie przez powyższych autora/ów, a wszystkie elementy pochodzące z innych źródeł zostały odpowiednio zaznaczone i są cytowane w bibliografii.

Udział autorów

- PG zaimplementował..., przeprowadził eksperyment..., opisał..., przygotował...
- ML zaimplementowała..., przeprowadziła eksperyment..., opisała..., przygotowała...

1 Symetryczny problem komiwojażera (STSP)

1.1 Opis problemu

Symetryczny problem komiwojażera modeluje sytuację znaną z rzeczywistego świata, w której osoba ma odwiedzić konkretne miejsca w dowolnej kolejności i wrócić do miejsca początkowego tak, aby pokonać jak najkrótszą drogę. Z tego typu zadaniem mierzą się przede wszystkim wszyscy dostawcy, listonosze, akwizytorzy. W symetrycznym problemie komiwojażera odległości pomiędzy dwoma miejscami są takie same w obie strony. Problem nie daje możliwości tworzenia dróg jednokierunkowych, a także budowana sieć dróg jest grafem pełnym.

1.2 Złożoność

W tak postawionym problemie, istnieje różnych $\theta(n!)$ rozwiązań, gdzie n oznacza liczbę miejsc do odwiedzenia. Miejsca możemy odwiedzać w dowolnej kolejności, więc jeśli zostaną one ponumerowane od 1 do n , każda permutacja n -elementowa może reprezentować pełne rozwiązanie. Rozwiązanie w postaci permutacji możemy odczytywać w taki sposób, że z miejsca na pozycji i , przemieszczamy się do miejsca na pozycji $i + 1$, pamiętając o tym, żeby z miejsca na pozycji n wrócić do startowego o indeksie 1. Przestrzeń rozwiązań jest więc bardzo duża i trudno jest przejrzeć je wszystkie. Jeśli bylibyśmy w stanie sprawdzać 1'000'000'000 rozwiązań w czasie 1 sekundy, rozwiązania dokładnego dla $n = 16$, szukalibyśmy przez ok. 6h, a znalezienie go dla $n = 20$ zajęłoby 77 lat.

1.3 Rozwiązanie losowe

Ponieważ rozwiązania można reprezentować w postaci permutacji, da się w łatwy i szybki sposób wygenerować losowe początkowe rozwiązanie dla wielu innych algorytmów poprzez wygenerowanie losowej permutacji. Złożoność generowania permutacji to $\theta(n)$, gdzie n jest jej długością. Aby wygenerować losową permutację należy zastosować poniższą procedurę:

1. Wypełnij tablicę liczbami od 1 do n .
2. $i := n$.
3. Zamień element z pozycji $i - 1$ z elementem na losowej wcześniejszej lub tej samej pozycji (od 0 do $i - 1$).
4. $i := i - 1$.
5. Jeżeli $i > 1$, wróć do kroku 2.

1.4 Heurystyka

Dla problemu komiwojażera istnieje prosta heurystyka o złożoności $\theta(n^2)$, dająca zadowalające wyniki - przeciętnie odległe od rozwiązania optymalnego o 10 – 15%. [2] Polega ona na wykonaniu poniższych kroków:

1. Wybierz losowe miasto początkowe.
2. Znajdź najbliższe nieodwiedzone miasto i udaj się tam.
3. Jeśli pozostały jeszcze jakieś nieodwiedzone miasta, idź do kroku 2.
4. Wróć do początkowego miasta.

1.5 Instancje problemu

Problem jest bardzo powszechny i istnieje wiele gotowych instancji, których można użyć w badaniach. Wybraliśmy 8 instancji z udostępnionego przez uniwersytet „Heidelberg” zbioru. [1] Są to:

1. berlin52,
2. ch130,
3. eil51,
4. kroA100,
5. lin105,
6. pr76,
7. rd100,
8. tsp225.

Przy doborze konkretnych instancji zależało nam na tym, aby nie były one zbyt duże (przeważnie do 200 miast), a także miały znalezione rozwiązania optymalne, były różnorodne oraz podane w formacie EUC_2D, czyli za pomocą współrzędnych na dwuwymiarowej płaszczyźnie Euklidesowej.

2 Optymalizacja lokalna

2.1 Wstęp

Algorytm optymalizacji lokalnej pozwala na poszukiwanie lepszych rozwiązań od aktualnie znalezionego, tak długo, aż w zadanym sąsiedztwie nie można go już bardziej poprawić. Dzięki takiej optymalizacji, nie trzeba przeszukiwać całej przestrzeni rozwiązań, a osiągać dobre wyniki.

2.2 Operatory sąsiedztwa

Aby stosować przeszukiwanie lokalne, należy określić sąsiedztwo dla każdego rozwiązania. Jest to przestrzeń rozwiązań, które są podobne do posiadanego. Jednym z możliwych sąsiedztw jest OPT-2.

2.2.1 OPT-2

Sąsiedztwo OPT-2 definiuje się poprzez zamianę kolejności wierzchołków na dwóch pozycjach lub odwrócenie kolejności odwiedzania miast na łuku pomiędzy dwoma pozycjami. W zależności od tego, czy rozważany problem komiwojażera jest symetryczny, czy nie - różne sąsiedztwa sprawdzają się z odmienną skutecznością. Rozważając, które z sąsiedztw wybrać, należy zastanowić się, które z nich w najmniejszym stopniu wpływa na funkcję celu.

Zamiana miast Sąsiedztwo OPT-2 polegające na zamianie miast można zaimplementować poprzez prostą zamianę elementów na pozycji i -tej i j -tej. Jest ono skuteczniejsze dla problemu asymetrycznego komiwojażera, ponieważ zamiana łuków spowodowałaby większą różnicę w funkcji celu – inne byłyby nie tylko cztery drogi, prowadzące do dwóch zamienianych miast, lecz także wszystkie drogi na łuku.

Odwrócenie łuku W przypadku odwrócenia łuku pomiędzy miastami na pozycjach i -tej i j -tej, funkcja celu ulegnie zmianie spowodowanej tylko modyfikacją dwóch dróg prowadzących do miast i -tego i j -tego z jednej strony, a z drugiej odwrócenie łuku, gdy drogi w obie strony mają taką samą długość, niczego nie zmieni.

2.3 Greedy

Jednym z algorytmów przeszukiwania lokalnego jest algorytm Greedy. Polega on na tym, że jeśli w momencie przeszukiwania sąsiedztwa aktualnego rozwiązania, znajdzie rozwiązanie lepsze, natychmiast je wybiera jako aktualne i szuka następnego w jego sąsiedztwie, dopóki istnieje rozwiązanie poprawiające aktualne.

2.4 Steepest

Algorytm Steepest różni się od poprzednika tym, że gdy przegląda swoje sąsiedztwo, nie wybiera jako aktualne rozwiązanie, pierwszego znalezionego, lepszego rozwiązania, ale zawsze przegląda wszystkich sąsiadów i wybiera najlepszego spośród nich, dzięki czemu porusza się „po najbardziej stromym zboczu” na wykresie funkcji celu.

Rysunek 1: Porównanie średnich rozwiązań na różnych instancjach.

Rysunek 2: Porównanie najlepszych znalezionych rozwiązań przez algorytmy na różnych instancjach.

Rysunek 3: Porównanie czasu działania algorytmów na poszczególnych instancjach.

Rysunek 4: Porównanie efektywności algorytmów na poszczególnych instancjach.

Rysunek 5: Porównanie algorytmów Greedy Search i Steepest pod względem liczby kroków do zatrzymania.

3 Eksperymenty

3.1 Odległość od optimum

3.1.1 Przypadek średni

...

3.1.2 Przypadek najlepszy

...

3.2 Czas działania

...

3.3 Efektywność algorytmów

3.3.1 Wybrana miara

...

3.3.2 Wyniki

...

3.4 Średnia liczba kroków

...

Rysunek 6: Porównanie algorytmów Greedy Search i Steepest pod względem liczby przeszukiwanych rozwiązań.

Rysunek 7: Porównanie jakości rozwiązań początkowych i końcowych przez algorytmy Greedy Search i Steepest.

Rysunek 8: Porównanie jakości rozwiązań algorytmów Greedy Search i Steepest w zależności od liczby uruchomień tych algorytmów dla różnych rozwiązań początkowych.

Rysunek 9: Porównanie odległości znajdowanych rozwiązań przez algorytmy od rozwiązania optymalnego.

3.5 Średnia liczba przeszukanych rozwiązań

...

3.6 Greedy Search

3.6.1 Jakość rozwiązania początkowego a końcowego

...

3.6.2 Wielokrotne uruchamianie dla różnych rozwiązań początkowych

...

3.7 Porównanie rozwiązań

3.7.1 Miara odległości rozwiązań od rozwiązania optymalnego

...

3.7.2 Wyniki

...

4 Podsumowanie

4.1 Wnioski

...

4.2 Trudności

...

4.3 Propozycje udoskonaleń

Literatura

- [1] Universität Heidelberg. Discrete and combinatorial optimization. <http://comopt.ifl.uni-heidelberg.de/software/TSPLIB95/>.
- [2] Christian Nilsson. Heuristics for the traveling salesman problem. <https://web.tuke.sk/fei-cit/butka/hop/htsp.pdf>.