Politechnika Łódzka Instytut Informatyki Stosowanej

Podstawy Sztucznej Inteligencji

Laboratorium

Ćwiczenie 1

Rozwiązanie zagadnienia komiwojażera za pomocą przeszukiwania grafów.

Przygotował:

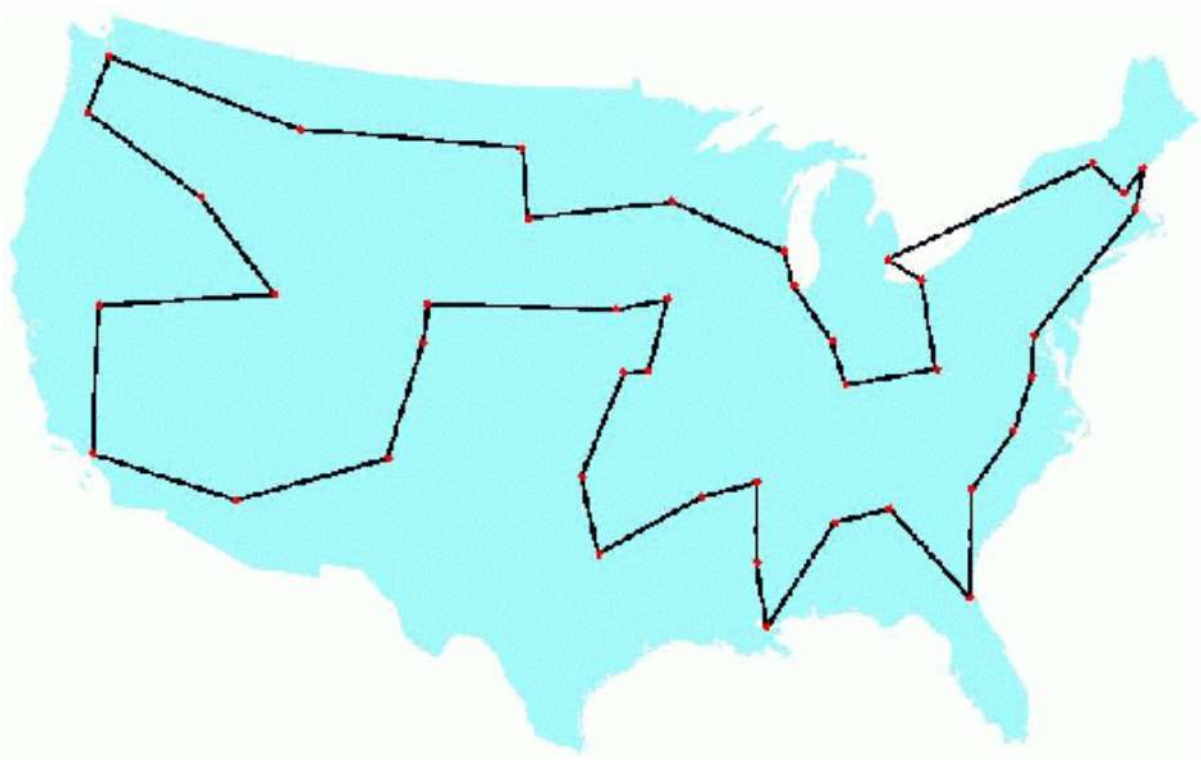
Dr inż. Piotr Urbanek

Wstęp

Problem komiwojażera (ang. Traveling Salesman Problem, TSP) przedstawia się następująco: komiwojażer musi odwiedzić n miast w taki sposób, by wrócił do miasta, z którego wyruszył pokonując jak najmniejszą drogę, musi wyznaczyć więc taką trasę między miastami, aby całkowity koszt jego podróży był najmniejszy.

Został on sformułowany jako zadanie matematyczne w latach 30-tych XX wieku, choć jego historia jest dużo starsza. Już w 1832 roku pewien podręcznik dla komiwojażerów wspominał to zagadnienie i zawierał przykładowe trasy uwzględniające Niemcy i Szwajcarię, choć bez opisu matematycznego problemu.

Używając metod numerycznych trzej uczeni: George Dantzig, Ray Fulkerson i Selmer Johnson przedstawili w 1954 roku optymalne rozwiązanie dla 49 miast w USA.



Obecnie cały czas trwają prace na rozwiązaniem TSP dla coraz większej liczby miast. Zainteresowanych odsyłam na strony internetowe opisujące omawiany problem.

Problem TSP można przestawić za pomocą grafu pełnego, tzn. grafu o stuprocentowym nasyceniu krawędziowym, co oznacza, że każda para wierzchołków

jest połączona krawędzią. Miasta, które musi odwiedzić komiwojażer są wierzchołkami, a drogi łączące te miasta to krawędzie z wagami, symbolizującymi koszt podróży daną drogą.

Problem komiwojażera jest NP-trudny, co oznacza, że nie są znane algorytmy o wielomianowej złożoności obliczeniowej rozwiązujące ten problem (przypuszczalnie takie nie istnieją). Ma on złożoność wykładniczą typu O(n!) i dla n miast liczba wszystkich kombinacji k_n (przy założeniu symetrii problemu, czyli że droga w jedną i w drugą stronę ma taki sam koszt) wyraża się zależnością:

$$k_n = \frac{(n-1)!}{2}$$

Jest ściśle związany z cyklem Hamiltona w grafie, tzn. takim cyklem, który zawiera każdy z wierzchołków danego grafu dokładnie 1 raz.

Omówienie niektórych metod rozwiązania zagadnienia TSP.

Aby rozwiązać TSP należy wyznaczyć cykl Hamiltona o najmniejszej sumie wag krawędzi należących do tego cyklu. Klasyczny sposób rozwiązania tego problemu polegający na sprawdzaniu długości każdego cyklu Hamiltona wymaga sprawdzenia wszystkich permutacji wierzchołków, co w konsekwencji prowadzi do złożoności wykładniczej (n!), praktyczne zastosowanie takiego algorytmu już dla kilkunastu miast jest więc niemożliwe. Nawet przy zastosowaniu powracania, gdy w każdym kroku aktualny koszt porównywany jest z najkrótszym wyznaczonym do tej pory cyklem, nie zmniejszy tej złożoności, choć może skrócić czas wykonania algorytmu w średnim przypadku, gdy najkrótsza trasa zostanie wyznaczona w pierwszych krokach. Jednak złożoność nadal będzie wykładnicza. Alternatywą dla przeglądu wyczerpującego jest zastosowanie **strategii zachłannej** w **algorytmie aproksymacyjnym**, dającym wprawdzie rozwiązanie przybliżone, lecz w czasie O(n2). Dzieje się tak przy założeniu, że funkcja kosztu podróży spełnia warunek trójkąta, to znaczy, że koszt podróży z miasta u do miasta v jest na pewno mniejszy niż suma kosztów podróży z miasta u do w i z w do v, czyli: c(u,v) < c(u,w) + c(v,w). W przypadku problemu komiwojażera nierówność ta jest spełniona, gdyż koszt podróży z miasta u do v jest po prostu

odległością euklidesową między tymi miastami. Odległość euklidesowa (koszt przebycia drogi K) pomiędzy miastami V_{i-1} o współrzędnych x_{i-1}, y_{i-1} a miastem $Vi(x_i, y_i)$ dana jest wzorem:

$$K = \sum_{i=1}^{M} \sqrt{(x_{i-1} - x_i)^2 + (y_{i-1} - y_i)^2}$$

Gdzie M – liczba miast, K – koszt przebycia drogi

Można wykazać, że odległość przybliżona jest co najwyżej dwa razy większa od odległości optymalnej. Aproksymacyjny algorytm wyznaczania najkrótszego cyklu Hamiltona w grafie polega na zastosowaniu **algorytmu wyznaczającego minimalne drzewo rozpinające** (metodą **Kruskala** lub **Prima**). Po zbudowaniu minimalnego drzewa rozpinającego należy przejść miasta metodą preorder (tzn. odwiedzać ojców przed synami) i zapisywać wierzchołki na liście tylko, gdy są odwiedzane po raz pierwszy. Przechodząc drzewo należy równocześnie sumować koszt przebycia krawędzi między dwoma kolejnymi wierzchołkami a na końcu dodać jeszcze koszt przebycia krawędzi między pierwszym i ostatnim wierzchołkiem na liście, ponieważ, zgodnie z założeniami problemu, komiwojażer musi powrócić do miasta, z którego wyruszył.

Jedną z częściej stosowanych metod rozwiązania TSP jest Metoda Najbliższego Sąsiada (N-N Method)

Działanie algorytmu N-N

Algorytm rozpoczyna działanie od wybranego wierzchołka początkowego i polega na kolejnym przechodzeniu do najbliższego nieodwiedzonego sąsiada ostatnio dodanego wierzchołka. W bardziej formalnym zapisie algorytm działa w następujący sposób:

1. Wierzchołek początkowy oznaczamy jako odwiedzony i ustawiamy jako aktualny.
2. Znajdujemy najkrótszą spośród krawędzi łączących aktualny wierzchołek z jeszcze nieodwiedzonymi wierzchołkami.
3. Dołączamy do rozwiązania krawędź znalezioną w punkcie 2.

4. Wierzchołek będący *drugim końcem* krawędzi znalezionej w punkcie 2 oznaczamy jako odwiedzony i ustawiamy jako aktualny.

5. Jeśli są jeszcze nieodwiedzone wierzchołki, przechodzimy do punktu 2.

6. Dołączamy krawędź łączącą ostatnio dodany wierzchołek z wierzchołkiem początkowym. Zamykamy w ten sposób cykl.

Algorytm Dijkstry

Algorytm służący do wyznaczania najkrótszych ścieżek w grafie. Wyznacza najkrótsze ścieżki z jednego wierzchołka (zwanego wierzchołkiem źródłowym) do pozostałych wierzchołków. Algorytm wymaga, aby wagi krawędzi grafu nie były ujemne. Autorem algorytmu jest holenderski naukowiec Edsger Dijkstra.

Algorytm realizuje podejście zachłanne. W każdej iteracji wybierany jest ten spośród nieodwiedzonych wierzchołków, do którego można dotrzeć najmniejszym kosztem. Po wyznaczeniu ścieżki do konkretnego wierzchołka nie zostanie ona zmodyfikowana w trakcie wykonywania dalszej części algorytmu.

W trakcie wykonywania algorytmu dla każdego wierzchołka zostają wyznaczone dwie wartości: koszt dotarcia do tego wierzchołka oraz poprzedni wierzchołek na ścieżce. Na początku działania algorytmu dla wierzchołka źródłowego koszt dotarcia wynosi 0 (już tam jesteśmy), a dla każdego innego wierzchołka nieskończoność (w ogóle nie wiemy, jak się tam dostać). Wszystkie wierzchołki na początku znajdują się w zbiorze Q (są to wierzchołki nieprzejrzane). Następnie algorytm przebiega następująco:

Dopóki zbiór Q nie jest pusty:

- Pobierz ze zbioru Q wierzchołek o najmniejszym koszcie dotarcia. Oznacz go jako v i usuń ze zbioru Q.
- Dla każdej krawędzi wychodzącej z wierzchołka v (oznaczmy ją jako k) wykonaj następujące czynności:
  - Oznacz wierzchołek znajdujący się na drugim końcu krawędzi k jako u.

- Jeśli koszt dotarcia do wierzchołka u z wierzchołka v poprzez krawędź k jest mniejszy od aktualnego kosztu dotarcia do wierzchołka u, to:
  - Przypisz kosztowi dotarcia do wierzchołka u koszt dotarcia do wierzchołka v powiększony o wagę krawędzi k.
  - Ustaw wierzchołek v jako poprzednik wierzchołka u.

Brute force

Najdokładniejszą metodą wyznaczenia optymalnej trasy jest metoda przeszukiwania całej przestrzeni możliwych rozwiązań (**metoda brute force**). Jest ona jednak skuteczna dla niewielkiej liczby miast. W tym przypadku należy wyznaczyć permutację możliwych połączeń z każdego dowolnego miasta (wierzchołka) do reszty miast (wierzchołków). Następnie dla każdej trasy obliczyć koszt K pokonanej drogi i zapisać ją w tablicy T. Po zbadaniu wszystkich kombinacji bez powtórzeń posortować zawartość tablicy T w kolejności rosnącej. Pierwszy element tablicy będzie najlepszym rozwiązaniem.

Wykonanie ćwiczenia

1. Wykonać aplikację zawierającą minimum 3 funkcje obliczające koszt przebycia drogi pomiędzy miastami. Pierwsza funkcja ma realizować metodę brute force a dwie pozostałe wybrane przez wykonującego ćwiczenie algorytmy suboptymalne (np. metodę Dijksrty, czy Najbliższego sąsiada).
2. Czwarta funkcja wykonywanej aplikacji powinna zwracać M miast o wylosowanych współrzędnych (x_i, y_i).
3. Przetestować działanie aplikacji dla N miast, gdzie N=3, 4, ….M
4. Wyniki badania zamieścić w tabelach:

Metoda Brute Force, metoda 2, metoda 3

Liczba miast	Koszt
3	
4	
…..	
M	

Mile widziane będzie zamieszczenie w sprawozdaniu graficznej reprezentacji otrzymanych rozwiązań.

Do sprawozdania dołączyć pliki źródłowe wykonanych aplikacji, wewnątrz których powinien znajdować się opis zawartego kodu.