Akademia Górniczo-Hutnicza
im. Stanisława Staszica w Krakowie

AGH University of Krakow

Problem komiwojażera

Bartosz Konopka

Part I

Wstęp teoretyczny

Czym jest problem komiwojażera?

Definicja formalna
Zagadnienie optymalizacyjne, polegające na znalezieniu minimalnego cyklu Hamiltona w pełnym grafie ważonym.

Definicja intuicyjna
Problem polegający na tym, że w pełnym grafie ważonym, chcemy odwiedzić każdy wierzchołek dokładnie raz i wrócić do punktu w którym zaczynaliśmy podróż, robiąc to z jak najmniejszym kosztem.

Przykład

Miasta: Kutno, Warszawa, Poznań, Kraków
Odległości:

-	Kutno	Warszawa	Poznań	Kraków
Kutno	0	130	180	300
Warszawa	130	0	320	350
Poznań	180	320	0	360
Kraków	300	350	360	0

Należy znaleźć najkrótszą trasę zaczynającą się np. z Kutna, przechodzącą jednokrotnie przez wszystkie pozostałe miasta i wracającą do Kutna.

Zastosowanie

- Logistyka
- Trasowanie sieci komunikacyjnych
- Planowanie tras
- Procesy produkcyjne

Podział problemu komiwojażera

Problem komiwojażera podzielić można na dwa rodzaje:

- **Symetryczny problem komiwojażera (STSP)** - polegający na tym, że dla dowolnych wierzchołków A i B odległość z A do B jest taka sama jak z B do A.
- **Asymetryczny problem komiwojażera (ATSP)** - w którym odległość między wierzchołkami A i B oraz B i A może być różna.

Problem komiwojażera jako problem NP-trudny

- Problem komiwojażera jest problemem NP-trudnym, co oznacza, że nie da się znaleźć rozwiązania optymalnego w czasie wielomianowym.
- Główną trudnością problemu jest duża liczba danych do analizy. W przypadku symetrycznego problemu komiwojażera dla n miast liczba kombinacji wynosi

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

tak więc dla 20 miast uzyskujemy wynik

$$\frac{19!}{2} \approx 6x10^{16}$$

.

Ograniczenie dolne problemu komiwojażera

Dla małych grafów zastosować można rozwiązania optymalne, takie jak sprawdzenie wszystkich wariantów, czy algorytm Helda-Karpa. Jednakże dla większych przypadków wykorzystuje się algorytmy heurystyczne, czyli takie dla których nie mamy gwarancji znalezienia rozwiązania optymalnego. W przypadku takich algorytmów oprócz rozważenia ich złożoności czasowej, określamy również ich skuteczność.

Aby określić średnią skuteczność algorytmów heurystycznych wprowadzamy pojęcie granicy dolnej, a następnie porównujemy średnią skuteczność. Na przykład jeżeli algorytm jest średnio o połowę gorszy niż ograniczenie dolne, stosunek będzie wynosił 1,5.

Jedną z popularnych metod obliczania dolnego ograniczenia jest wykorzystanie minimalnego drzewa rozpinającego.

Przykładowe algorytmy

Algorytm	Dokładność	Złożoność czasowa	śr optymalność
"brute force"	Tak	$O(n!)$	-
Algorytm Helda-Karpa	Tak	$O(n^2 2^n)$	-
Algorytm najbliższego sąsiada	Nie	$O(n^2)$	1.25
Algorytm najmniejszej krawędzi	Nie	$O(n^2 logn)$	1.17
Algorytm Christofidesa	Nie	worst - $O(n^3)$	około 1.1, <1.5

Part II

Algorytm brute force

Algorytm brute force

Algorytm brute force w problemie komiwojażera polega na przeglądaniu wszystkich możliwych tras przez wszystkie wierzchołki i wybieraniu najkrótszej. Na początku generowane są wszytskie permutacje wierzchołków, obliczane są odległości między wierzchołkami dla każdej permutacji, a następnie wybierana jest najkrótsza odległość. Algorytm ten ma złożoność obliczeniową $O(n!)$, dlatego sprawdza się tylko w przypadku bardzo małych grafów.

Part III

Algorytm Helda-Karpa

Algorytm Helda-Karpa

To dokładny algorytm wykorzystujący programowanie dynamiczne. Jego czasowa złożoność wynosi $O(n^2 2^n)$. Mimo że to gorszy przypadek niż złożoność wielomianowa, jest znacznie efektywniejszy od algorytmu przeglądającego wszystkie możliwości, którego złożoność wynosi $O(n!)$.

Działanie algorytmu

Załóżmy, że mamy graf liczący n wierzchołków ponumerowanych 1, 2, . . . , n. Wierzchołek 1 niech będzie wierzchołkiem początkowym. Jako d_{ij} oznaczmy odległość między wierzchołkami i oraz j (są to dane wejściowe). Oznaczmy jako $D(S, p)$ optymalną długość ścieżki wychodzącej z punktu 1 i przechodzącej przez wszystkie punkty zbioru S tak, aby zakończyć się w punkcie p (p musi należeć do S). Przykładowo, zapis $D(\{2, 5, 6\}, 5)$ to optymalna długość ścieżki wychodzącej z punktu 1, przechodzącej przez punkty 2 i 6, kończącej się w punkcie 5. Liczbę punktów w zbiorze S oznaczmy jako s.
Wartość D(S, p) wyznaczamy następująco:
Jeśli $s = 1$, to $D(S, p) = d_{1,p}$,
Jeśli $s > 1$, to $D(S, p) = min_{x \epsilon (S - \{p\})}(D(S - \{p\}, x) + d_{x,p})$.
Musimy za każdym razem ustalać, który punkt powinien być przedostatni na trasie (który punkt ma poprzedzać punkt p).

Part IV

Algorytm najbliższego sąsiada

Działanie

Jest to algorytm wykorzystujący strategię zachłanną.
Algorytm rozpoczyna działanie od wybranego wierzchołka (nazwijmy go wierzchołkiem początkowym) i polega na kolejnym przechodzeniu do najbliższego nieodwiedzonego sąsiada ostatnio dodanego wierzchołka. W bardziej formalnym zapisie algorytm działa w następujący sposób:

- Wierzchołek początkowy oznaczamy jako odwiedzony i ustawiamy jako aktualny.
- Znajdujemy najkrótszą spośród krawędzi łączących aktualny wierzchołek z jeszcze nieodwiedzonymi wierzchołkami.
- Dołączamy do rozwiązania krawędź znalezioną w punkcie 2.
- Wierzchołek będący drugim końcem krawędzi znalezionej w punkcie 2 oznaczamy jako odwiedzony i ustawiamy jako aktualny.
- Jeśli są jeszcze nieodwiedzone wierzchołki, przechodzimy do punktu 2.
- Dołączamy krawędź łączącą ostatnio dodany wierzchołek z wierzchołkiem początkowym. Zamykamy w ten sposób cykl.

Part V

Algorytm najmniejszej krawędzi

Działanie

Jest to algorytm wykorzystujący strategię zachłanną, jednak w inny sposób, niż algorytm najbliższego sąsiada.
Algorytm działa podobnie do algorytmu Kruskala poszukującego minimalnego drzewa rozpinającego. Polega on na kolejnym dołączaniu do rozwiązania najkrótszych spośród dopuszczalnych krawędzi. Działanie algorytmu można zapisać następująco:

- Posortuj wszystkie krawędzie rosnąco według ich wag, umieść je w kolejce.
- Pobierz z kolejki krawędź o najmniejszej wadze, usuń ją z kolejki.
- Sprawdź, czy dołączenie tej krawędzi do rozwiązania nie spowoduje utworzenia cyklu (nie dotyczy ostatniej iteracji) lub powstania wierzchołka, z którego wychodzą trzy krawędzie. Jeśli nie, dołącz krawędź do rozwiązania.
- Jeśli liczba krawędzi dołączonych do rozwiązania jest równa liczbie wierzchołków, zakończ działanie algorytmu. W przeciwnym razie przejdź do punktu 2.

Part VI

Algorytm Christofidesa

Działanie

Algorytm jest 1,5-optymalny, to znaczy, że znalezione rozwiązanie będzie nie gorsze niż 1,5 rozwiązania optymalnego.

- Dla grafu G stwórzmy minimalne drzewo rozpinające T.
- Niech O będzie zbiorem wierzchołków o nieparzystym stopniu w T Znajdźmy minimalne skojarzenie doskonałe M na wierzchołkach O spośród krawędzi grafu pełnego G.
- Niech H będzie multigrafem utworzonym z M i T.
- Wyznaczmy cykl Eulera w grafie H (graf H jest eulerowski, ponieważ ma wszystkie wierzchołki parzystego stopnia).
- Z cyklu Eulera zróbmy cykl Hamiltona poprzez pomijanie odwiedzonych wierzchołków (skracanie).