Problem komiwojażera

Problem komiwojażera przedstawia się następująco. Komiwojażer wyjeżdża z miasta nr 1, przejeżdża przez pewien zbiór miast, po czym wraca do miasta, z którego wyjechał. Im szybciej pokona tę drogę, tym lepiej. Nasze zadanie polega na znalezieniu najkrótszego cyklu przechodzącego przez wszystkie miasta dokładnie raz. Opiszę, jak znaleźć długość najkrótszej drogi, jednakże posługując się tymi metodami, otrzymuje się także trasę komiwojażera.

Problem można oczywiście rozwiązać, przeszukując wszystkie możliwości. Dla 2 miast jest jedna możliwość. Dla 3 miast dwie możliwości $\{ABCA; ACBA\}$, dla czterech miast liczba możliwości wynosi 6: $\{ABCDA; ACDBA; ACBDA; ADCBA; ABDCA; ADBCA\}$. Ogólnie mamy $(n-1)!$ możliwych przypadków do przeanalizowania. Dla 49 miast liczba możliwości będzie sięgała 48! przypadków, czyli w zaokrągleniu około: $1{,}241391559 \cdot 10^{61}$. Przy założeniu, że jesteśmy w stanie liczyć miliard kombinacji na sekundę, program działałby $2 \cdot 10^{34}$ razy tyle co wiek naszego Wszechświata.

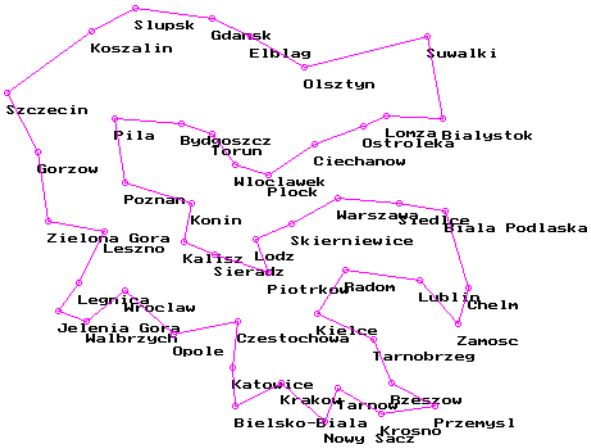
Problem ten można jednak rozwiązać metodą programowania dynamicznego. Oznaczmy przez $D(x,y)$ długość drogi z miasta x do miasta y. Niech $L(x,S)$ oznacza długość najkrótszej drogi z miasta x, przechodzącej przez wszystkie miasta ze zbioru S – przez każde dokładnie raz – kończącej się na mieście nr 1. Jeśli oznaczymy zbiór wszystkich miast jako X, to $L(1, X \setminus \{1\})$ jest interesującym nas rozwiązaniem problemu komiwojażera. $L(x,S)$ możemy łatwo obliczyć. Komiwojażer z miasta x pojedzie najpierw do jednego miasta ze zbioru S, a później możliwie krótką drogą przejedzie przez resztę miast ze zbioru S, aż do miasta nr 1. Tak więc $L(x,S) = \min\limits_{y \in S} \big(D(x,y) + L(y, S \setminus \{y\})\big)$.

Metoda programowania dynamicznego jest szybsza od przeszukiwania wszystkich możliwości. Złożoność czasowa jest rzędu $n^2 2^n$. Niestety metoda ta wymaga sporej pamięci (złożoność pamięciowa jest rzędu $n2^n$, już dla 22 miast potrzeba ponad 2 MB pamięci operacyjnej). Dodatkowo trudno jest taki algorytm optymalizować.

Przeszukiwanie wszystkich możliwości można jednak usprawnić, co, jak się okazuje, daje lepsze rezultaty. Załóżmy więc, że mamy najkrótszą dotychczas znalezioną drogę komiwojażera – cykl A. Mamy też jakąś drogę B przechodzącą przez k miast, gdzie $k < n$. Jeśli długość drogi B jest większa od długości cyklu A, to nie warto, oczywiście, już jej rozważać. W ten sposób nie będziemy analizować wszystkich możliwości.

Oczywiście usprawnienie to można także ulepszyć. Spróbujmy oszacować, o ile co najmniej wydłuży naszą drogę B dodanie do niej pozostałych $n-k$ miast. Są różne metody – jedne zabierają dużo czasu, inne z kolei dają mniej dokładny wynik. W tym przypadku opłaca się przeznaczyć więcej czasu na dokładniejsze oszacowanie. Na miastach skrajnych drogi B oraz miastach nieprzyłączonych do naszej drogi budujemy minimalne drzewo spinające (MST). Długość takiego drzewa jest niewiększa od długości drogi, która będzie domykać naszą drogę. Co za tym idzie, cykl powstały z drogi B wraz z dołączonymi do niej pozostałymi miastami nie będzie dłuższy od sumy długości drogi B i długości MST. Pozostaje więc tylko zbudowanie MST. W tym celu posługuję się tzw. algorytmem Kruskala (dokładny opis oraz dowód poprawności algorytmu znajduje się w książce K.A. Ross, Ch.R.B. Wright, „Matematyka dyskretna"). Kolejne przyspieszenie można uzyskać na czasie budowania MST. Program budowałby MST dla tych samych miast wiele razy, można więc je zapamiętywać. W tym celu użyłem techniki haszowania – zapamiętywałem MST w tablicy list. Oczywiście nie zapamiętałem wszystkich drzew. Po pierwsze zabrakłoby pamięci operacyjnej, po drugie zaś i przede wszystkim nie opłaca się ich zapamiętywać, gdyż listy wydłużyłyby się za bardzo i wyszukiwanie interesujących nas MST trwałoby zbyt długo. Dlatego też zapamiętywałem tylko MST składające się z co najwyżej 5 miast.

Na koniec można jeszcze zauważyć, iż wszystkie dotychczasowe optymalizacje opierały się na założeniu, iż mam już jakąś drogę przybliżoną. Przed uruchomieniem głównej procedury program liczył więc drogę przybliżoną. W tym celu posługiwałem się algorytmami opisanymi w książce M.M. Sysło, N. Deo, J.S. Kowalik, „Algorytmy optymalizacji dyskretnej" oraz opisanymi w niej tzw. 2- oraz 3-optymalizacjami. W ten sposób uzyskana droga przybliżona była dłuższa od cyklu minimalnego o około 0–6%. Dane do programu zostały wprowadzone w formie współrzędnych na mapie. Długości między miastami były zaś odległościami pomiędzy punktami na mapie. Ostatecznie program dla 49 miast wojewódzkich według starego podziału administracyjnego Polski działał około 2 godzin 30 minut. Drogę przybliżoną liczył około 48 sekund. Długość drogi przybliżonej wynosiła około 3699 km, a długość drogi optymalnej wynosiła około 3580 km.



Aktualny światowy rekord pochodzi z 2001 roku. Ustanowili go D. Applegate, R. Bixby, V. Chvátal, i W. Cook, którzy rozwiązali problem komiwojażera dla 15112 miast. Tę i inne informacje dotyczące problemu komiwojażera można znaleźć na stronie internetowej: `www.math.princeton.edu/tsp/index.html`

Krzysztof MROCZEK, Liceum Przymierza Rodzin w Warszawie – **BRĄZOWY MEDAL**