Łukasz Kowalik
Treść zadania, Opracowanie

Tomasz Czajka
Program

Zawody

U stóp Bajtogóry znajduje się wejście do jaskini. Jaskinia to system komnat połączonych korytarzami. Wejście do jaskini prowadzi bezpośrednio do komnaty nazywanej **wejściową**. *Korytarze nie przecinają się (spotykają się jedynie w komnatach). Dwie komnaty mogą być albo połączone jednym korytarzem, albo mogą nie być połączone wcale (być może można jednak wtedy przejść z jednej do drugiej przechodząc po drodze przez inne komnaty). Korytarz łączy zawsze dwie różne komnaty.*

Postanowiono zorganizować zawody o puchar króla Bajtocji. Celem każdego z zawodników jest przebycie dowolnie wybranej trasy w jaskini i wyjście na zewnątrz w jak najkrótszym czasie. Trasa musi przechodzić przez co najmniej jedną komnatę inną niż wejściowa. Obowiązują tylko dwie zasady: podczas wędrówki po jaskini, każdą komnatę można odwiedzić co najwyżej raz (z wyjątkiem komnaty wejściowej), podobnie tylko raz można przejść każdym z korytarzy.

Do zawodów przygotowuje się słynny grotołaz Bajtała. Bajtała długo trenował w jaskini i dokładnie poznał sieć korytarzy. Dla każdego z korytarzy wyznaczył czasy potrzebne do jego przejścia w każdą stronę. Czas potrzebny do poruszania się po komnatach można zaniedbać. Bajtała chciałby teraz wyznaczyć taką trasę spełniającą wymogi zawodów, którą można przebyć w jak najkrótszym czasie (czas potrzebny do przebycia trasy jest sumą czasów potrzebnych do przejścia korytarzy składających się na trasę).

Zadanie

Pomóż Bajtale! Napisz program, który:

- *wczyta ze standardowego wejścia opis jaskini oraz czasy potrzebne do przejścia poszczególnych korytarzy,*
- *obliczy trasę zgodną z zasadami zawodów, dla której suma czasów przejść korytarzy składających się na trasę jest najmniejsza,*
- *wypisze na standardowe wyjście czas potrzebny do przejścia wyznaczonej trasy.*

Wejście

W pierwszym wierszu wejścia znajdują się dwie liczby całkowite n i m oddzielone pojedynczym odstępem, oznaczające odpowiednio liczbę komnat w jaskini, oraz liczbę łączących je korytarzy, $3 \leqslant n \leqslant 5\,000$, $3 \leqslant m \leqslant 10\,000$. Komnaty są ponumerowane od 1 do n. Komnata wejściowa ma numer 1. W kolejnych m wierszach opisane są poszczególne korytarze. W każdym z tych wierszy znajduje się czwórka liczb naturalnych oddzielonych pojedynczymi odstępami. Czwórka a,b,c,d oznacza, że jaskinie a i b są połączone korytarzem, czas przejścia z jaskini a do b wynosi c, natomiast z b do a wynosi d, $1 \leqslant a,b \leqslant n$, $a \neq b$, $1 \leqslant c,d \leqslant 10\,000$. Możesz założyć, że zawsze istnieje przynajmniej jedna trasa spełniająca wymogi zawodów.

Wyjście

Twój program powinien wypisać w pierwszym i jedynym wierszu wyjścia jedną liczbę całkowitą — minimalny czas potrzebny do przebycia trasy spełniającej warunki zawodów.

Przykład

Dla danych wejściowych:

```
3 3
1 2 4 3
2 3 4 2
1 3 1 1
```

poprawnym wynikiem jest:

```
6
```

Rozwiązanie

Wyjdźmy z jaskini

W treści zadania jest mowa o jaskini, ale i tak każdy wie, że chodzi o graf. Wierzchołkami naszego grafu są komnaty jaskini, to jasne. Ale czym są krawędzie? Wiemy, że dwie komnaty mogą być połączone korytarzem. Każdy z korytarzy Bajtała przemierzył w obie strony, zapisując czasy przejścia. Umówmy się, że każdemu korytarzowi w jaskini odpowiadają w grafie *dwie* krawędzie. Jeśli korytarz łączy komnaty odpowiadające wierzchołkom u i v, jedna z tych krawędzi prowadzi od u do v, natomiast druga odwrotnie, od v do u. Dodatkowo, każdej z krawędzi e przypisano *wagę* $w(e)$ — tzn. liczbę naturalną równą czasowi przejścia korytarza w odpowiednią stronę. Tak opisany graf będziemy oznaczać przez G. Zbiory wierzchołków i krawędzi G będziemy tradycyjnie oznaczać, odpowiednio, przez V i E. Przyjmiemy też, że n i m oznaczają, odpowiednio, liczby wierzchołków i krawędzi grafu G. Zauważmy, że G jest grafem skierowanym, choć bardzo specyficznym: jeśli w G istnieje krawędź od u do v (oznaczamy ją (u,v)), to jest tam również krawędź (v,u).

Spróbujmy teraz sformułować nasze zadanie w języku teorii grafów. W tym celu potrzebujemy kilku definicji.

Dowolny ciąg krawędzi (v_0,v_1), (v_1,v_2), …, (v_{k-1},v_k) nazywamy *ścieżką*. Jeśli $v_0 = v_k$ to taką ścieżkę nazywamy *cyklem*. Gdy wierzchołki v_0, v_1, …, v_k są różne mamy do czynienia ze *ścieżką prostą*. Podobnie, gdy wierzchołki v_0, v_1, …, v_{k-1} są różne i $v_0 = v_k$, to mamy *cykl prosty*. *Wagą* albo *długością* ścieżki nazywamy sumę wag jej krawędzi. Często zamiast pisać „ścieżka o najmniejszej wadze” będziemy używać określenia *najkrótsza ścieżka*.

Teraz, gdy przyswoiliśmy sobie tych kilka prostych pojęć, możemy wreszcie nazwać rzecz po imieniu. Niech v_0 będzie wierzchołkiem odpowiadającym komnacie wejściowej. Zadanie polega na opracowaniu algorytmu, który znajdzie w grafie G najkrótszy (o najmniejszej wadze) cykl prosty przechodzący przez v_0 i składający się z więcej niż dwóch krawędzi (zauważmy, że przechodząc w jaskini trasę odpowiadającą takiemu cyklowi Bajtała co najwyżej raz odwiedza każdą komnatę, z wyjątkiem wejściowej, podobnie co najwyżej raz przechodzi przez każdy korytarz).

Obserwacja

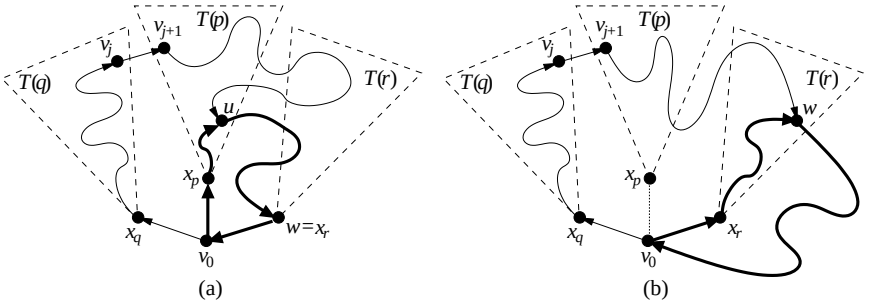
Niech C będzie szukanym cyklem i niech v_0, v_1, ..., v_k będą jego kolejnymi wierzchołkami oraz $v_0 = v_k$. Wybierzmy największe $j \in \{0,1,\ldots,k-1\}$ takie, że ścieżka P_1 przechodząca kolejno przez wierzchołki v_0, ..., v_j jest najkrótszą ścieżką od v_0 do v_j. Oczywiście takie j zawsze istnieje. W najgorszym razie $j = 0$ i jedyna taka ścieżka ma długość 0. Bez trudu zauważamy, że ścieżka $P_2 = v_j, v_{j+1}, \ldots, v_k$ (pamiętamy, że $v_k = v_0$) jest najkrótszą ścieżką od v_j do v_0 omijającą wierzchołki v_1, ..., v_{j-1}.

Wnioski

Bez straty ogólności możemy założyć, że do każdego wierzchołka w grafie prowadzi ścieżka o początku w v_0 (choć treść zadania nie wyklucza istnienia wierzchołków, do których nie prowadzi żaden korytarz, nie będą one miały wpływu na rozwiązanie). W takim razie każdemu wierzchołkowi w grafie, powiedzmy v, możemy przypisać najkrótszą ścieżkę od v_0 do v i oznaczyć ją przez $s(v)$ (gdy takich najkrótszych ścieżek jest wiele, wybieramy dowolną z nich). Niech x_1, ..., x_d będą sąsiadami v_0. Dla $i = 1,\ldots,d$ oznaczmy

$$T(i) = \{v \in V : \text{ścieżka } s(v) \text{ zaczyna się od krawędzi } (v_0, x_i)\}.$$

Oczywiście może się zdarzyć, że $T(i) = \emptyset$ dla niektórych wartości i. Dodatkowo przyjmujemy $T(0) = \{v_0\}$. Nietrudno zauważyć, że zbiory $T(i)$ tworzą podział zbioru V, tzn. każdy wierzchołek jest w dokładnie jednym z tych zbiorów. Przypomnijmy sobie teraz naszą obserwację i zastanówmy się, przez ile zbiorów $T(i)$ może przechodzić szukany przez nas cykl C. Najpierw przyjrzyjmy się ścieżce $P_1 = v_0, \ldots, v_j$. Widzimy, że wszystkie jej wierzchołki, z wyjątkiem v_0, znajdują się w jednym zbiorze $T(q)$, dla $v_1 = x_q$. Teraz zastanówmy się nad drugą ścieżką, tzn. ścieżką $P_2 = v_j, \ldots, v_k$. Pamiętamy, że $v_j \in T(q)$, ale $v_{j+1} \in T(p)$, dla $p \neq q$. Co się dzieje dalej? Załóżmy, że jeden z kolejnych wierzchołków ścieżki należy do zbioru $T(r)$, dla $r \neq p$. Pokażemy, że istnieje wtedy cykl prosty C', składający się z co najmniej 3 krawędzi i zawierający v_0, ale krótszy od C.



Rysunek 1: Zastępowanie cyklu C przez krótszy cykl C' (pogrubiony)

Niech w będzie ostatnim wierzchołkiem na ścieżce P_2, który należy do $T(r)$ oraz niech u będzie ostatnim wierzchołkiem na P_2 przed w, który należy do $T(p)$. Jeśli $w = x_r$ oraz

krawędź wychodząca z w na ścieżce P_2 prowadzi do v_0, możemy otrzymać cykl C' poprzez sklejenie ścieżki $s(u)$ z tą częścią ścieżki P_2, która zaczyna się w u (patrz rys. 1a). W przeciwnym przypadku cykl C' powstaje przez sklejenie ścieżki $s(w)$ z tą częścią P_2, która zaczyna się w w (patrz rys. 1b). W obu przypadkach cykl C' jest krótszy niż C. To jest niemożliwe, a więc wierzchołki $v_{j+1}, \ldots, v_{k-1}$ leżą wszystkie w jednym zbiorze $T(p)$. Więcej, widzimy że ścieżka $v_{j+1}, \ldots, v_k$ jest najkrótszą ścieżką od v_{j+1} do $v_k = v_0$ spośród ścieżek, które przechodzą wyłącznie po wierzchołkach z $T(p)$. Podsumujmy:

Fakt 1 *Najkrótszy cykl prosty o więcej niż dwóch krawędziach i zawierający wierzchołek v_0, składa się z*

- *ścieżki $v_0, \ldots, v_j$ równej ścieżce $s(v_j)$,*
- *krawędzi (v_j, v_{j+1}) takiej, że jeśli $v_1 = x_q$, to $v_{j+1} \in T(p)$, $p \neq q$,*
- *najkrótszej ścieżki od v_{j+1} do v_0 przechodzącej wyłącznie po wierzchołkach $T(p)$.*

Algorytm

Krok pierwszy: tam...

Odkrycie powyższego faktu prowadzi nas już bez przeszkód do algorytmu. Na początek dla każdego wierzchołka w grafie obliczamy jego odległość od v_0, tzn. długość najkrótszej ścieżki od v_0 do tego wierzchołka. W tym celu stosujemy klasyczny algorytm Dijkstry. Jeśli jeszcze go nie poznałeś, zajrzyj koniecznie do jednej z wielu książek, w których jest on dokładnie opisany (książka „Wprowadzenie do algorytmów" Cormena, Leisersona i Rivesta ([17]) jest tu szczególnie godna polecenia). Algorytm Dijkstry wyznacza nie tylko długości najkrótszych ścieżek. Dla każdego wierzchołka v wyznacza także wierzchołek poprzedzający v na pewnej najkrótszej ścieżce od v_0 do v. Nam ta informacja będzie niepotrzebna, ale z jej pomocą możemy łatwo (w czasie liniowym) obliczyć dla każdego wierzchołka $v \neq v_0$ wierzchołek $x_{p(v)}$ taki, że najkrótsza ścieżka od v_0 do v znaleziona przez algorytm Dijkstry zaczyna się od krawędzi $(v_0, x_{p(v)})$. Dodatkowo przyjmujemy $p(v_0) = 0$. Niech ponadto $d(v)$ oznacza obliczoną odległość v od v_0.

Krok drugi: ...z powrotem

Następnie dla każdego wierzchołka $v \neq v_0$ wyznaczamy długość najkrótszej ścieżki do v_0 przechodzącej wyłącznie po wierzchołkach ze zbioru $T(p(v))$. Obojętnie, jaką metodą znajdujemy ścieżki, możemy wymusić to ograniczenie po prostu poprzez ignorowanie krawędzi łączących wierzchołki o różnych wartościach $p(\cdot)$ (choć zostawiamy wszystkie krawędzie wchodzące do i wychodzące z v_0). Aby obliczyć długości ścieżek moglibyśmy dla każdego wierzchołka uruchomić algorytm Dijkstry. Byłoby to jednak nadmiernie czasochłonne, gdyż wystarczy nam jedno uruchomienie tego algorytmu! Jedyne, co musimy zrobić, to odwrócić kierunki krawędzi w grafie i uruchomić algorytm Dijkstry, który w takim *odwróconym* grafie znajdzie długości najkrótszych ścieżek od v_0 do wszystkich innych wierzchołków. Ścieżka od v_0 do v w grafie odwróconym to oczywiście ścieżka od v do v_0 w grafie oryginalnym, a więc dostajemy dokładnie to, co chcieliśmy. Niech $h(v)$ oznacza obliczoną odległość. Przyjmujemy $h(v_0) = 0$.

Krok trzeci: minimum

Krawędź $e = (v,w) \in E$ nazwiemy *pośrednią*, gdy $p(v) \neq p(w)$. Fakt 1 mówi nam, że aby znaleźć długość szukanego najkrótszego cyklu, wystarczy teraz znaleźć taką krawędź pośrednią $e = (v,w)$, że suma $d(v) + w(e) + h(w)$ jest najmniejsza. Wartość tej sumy jest szukaną liczbą.

Uwagi o złożoności

Krok trzeci algorytmu zajmuje oczywiście czas liniowy, a więc całkowita złożoność czasowa jest zdominowana przez złożoność czasową użytej implementacji algorytmu Dijkstry. Najprostsza implementacja tego algorytmu ma złożoność $O(n^2)$, jednakże testy zostały tak dobrane, że takie rozwiązanie nie zdobywało maksymalnej liczby punktów. Taki wynik gwarantowało dopiero użycie w algorytmie Dijkstry kopców binarnych, które zmniejszają pesymistyczny czas obliczeń do $O((n+m)\log n)$.

Program wzorcowy

Rozwiązanie zaprogramowane w programie wzorcowym jest nieco inne od opisanego powyżej, bazuje jednak na tych samych obserwacjach. Zaczynamy tak samo, od wykonania kroku pierwszego, tzn. obliczamy $d(v)$ i $p(v)$ dla wszystkich wierzchołków. Następnie budujemy taki graf G', żeby najkrótsza ścieżka pomiędzy dwoma wyróżnionymi jego wierzchołkami, S i M, miała długość taką samą, jak poszukiwany cykl w G. Oprócz S i M do grafu G' dodajemy jeszcze po jednym wierzchołku v' dla każdego wierzchołka $v \neq v_0$ grafu G. Graf G' zawiera następujące krawędzie (u, v oznaczają dowolne wierzchołki w grafie G takie, że $u, v \neq v_0$ i $d(u), d(v) \neq \infty$):

(e1) krawędź (S,M) z wagą $d(u)+w$ dla każdej $(u,v_0) \in E$ z wagą w, $u \neq x_p(u)$,

(e2) krawędź (S,v') z wagą w dla każdej $(v_0,v) \in E$ z wagą w, $v \neq x_p(v)$,

(e3) krawędź (S,v') z wagą $d(u)+w$ dla każdej $(u,v) \in E$ z wagą w, $x_p(u) \neq x_p(v)$,

(e4) krawędź (u',M) z wagą w dla każdej $(u,v_0) \in E$ z wagą w, $u = x_p(u)$,

(e5) krawędź (u',v') z wagą w dla każdej $(u,v) \in E$ z wagą w, $x_p(u) = x_p(v)$.

Widzimy, że G' jest liniowego rozmiaru względem oryginalnego grafu G, a więc jego skonstruowanie zajmie czas $O(n+m)$, a algorytm Dijkstry zaimplementowany za pomocą kopców binarnych znajdzie długość najkrótszej ścieżki od S do M w czasie $O((n+m)\log n)$.

Pozostaje jedynie uzasadnić, że wyniki otrzymywane przez oba algorytmy są takie same. W tym celu należy pokazać, że najkrótsza ścieżka od S do M w G' ma długość $d(v) + w(e) + h(w)$ dla pewnej krawędzi pośredniej $e = (v,w) \in E$. I odwrotnie, dla każdej krawędzi pośredniej $e = (v,w) \in E$ w grafie G' znajdzie się ścieżka od S do M długości co najwyżej $d(v) + w(e) + h(w)$. Proste dowody tych dwóch faktów pozostawiamy Tobie, Szanowny Czytelniku, jako ćwiczenie.

Rozwiązanie poprawne, ale nieoptymalne

Rozważmy następujący algorytm: dla każdej krawędzi wychodzącej z v_0 i prowadzącej do pewnego wierzchołka x obliczamy za pomocą algorytmu Dijkstry długość najkrótszej ścieżki od x do v_0 w grafie, w którym usunięto krawędź (x, v_0). Ten bardzo prosty algorytm jest oczywiście poprawny, ale w pesymistycznym przypadku, gdy z komnaty wejściowej istnieją korytarze do wszystkich innych komnat, jego złożoność czasowa wynosi $O(n(n+m)\log n)$.

Testy

Wszystkie testy, oprócz dwóch pierwszych, zostały wygenerowane automatycznie. Każdy z testów `zaw2.in`, `zaw3.in`, ..., `zaw10.in` zawierał dwie komnaty połączone korytarzem, do których nie można było dojść z komnaty wejściowej.

- `zaw1.in` — mały test poprawnościowy, $n = 13$, $m = 15$.
- `zaw2.in` — prawie cykl, $n = 32$, $m = 58$.
- `zaw3.in` — prawie graf pełny (prawie wszystkie wierzchołki połączone ze sobą), $n = 42$, $m = 781$.
- `zaw4.in` — dosyć losowy, duże rozgałęzienie z komnaty wejściowej, $n = 102$, $m = 2001$.
- `zaw5.in` — prawie drzewo binarne, $n = 4997$, $m = 4996$.
- `zaw6.in` — krata, dużo możliwości, $n = 2502$, $m = 4997$.
- `zaw7.in` — losowy, duże rozgałęzienie, $n = 3002$, $m = 9996$.
- `zaw8.in` — krata, $n = 5000$, $m = 9993$.
- `zaw9.in` — długi cykl z dołożonymi krawędziami od v_0 do wszystkich pozostałych wierzchołków, $n = 4998$, $m = 9990$.
- `zaw10.in` — prawie cykl, $n = 5000$, $m = 9994$.

Zawody II stopnia

opracowania zadań