**Marcin Pilipczuk** | **Marcin Andrychowicz**
Treść zadania, Opracowanie | Program

**Dostępna pamięć: 128 MB.** *OI, etap III, dzień próbny, 12.03.2013*

# Gra Tower Defense

*Bajtuś gra w grę komputerową* **Tower Defense***. Jego zadaniem jest tak pobudować wieże strażnicze, by strzegły całego jego państwa. W państwie Bajtusia znajduje się wiele miast, a niektóre z nich połączone są dwukierunkowymi drogami. Jeśli Bajtuś postawi w pewnym mieście wieżę strażniczą, wieża ta strzeże tego miasta oraz wszystkich innych miast połączonych z nim bezpośrednią drogą.*

*Gdy Bajtuś zastanawiał się nad rozmieszczeniem wież strażniczych w swoim państwie, do pokoju weszła jego starsza siostra Bajtunia. Bajtunia spojrzała na ekran komputera przedstawiający mapę państwa i po chwili stwierdziła: „Eee, nad czym się tu zastanawiać, przecież $k$ wież wystarczy!”.*

*Bajtuś, zły, że Bajtunia popsuła mu zabawę, przegonił siostrę z pokoju i zaczął się zastanawiać, co począć. Honor nie pozwoli mu teraz zbudować więcej niż $k$ wież. Ma jednak w zanadrzu tajną broń: może wynaleźć technologię, dzięki której będzie mógł budować* **ulepszone** *wieże strażnicze. Ulepszona wieża strażnicza pilnuje nie tylko miasta, w którym została wybudowana, i wszystkich bezpośrednio sąsiadujących miast, lecz także miast położonych trochę dalej. Formalnie, ulepszona wieża wybudowana w mieście $u$ pilnuje miasta $v$, jeśli zachodzi jeden z przypadków:*

- $u = v$;
- *istnieje bezpośrednia droga z $u$ do $v$;*
- *lub istnieje miasto $w$ takie, że istnieją bezpośrednie drogi z $u$ do $w$ oraz z $w$ do $v$.*

*Oczywiście, Bajtuś dalej musi wybudować co najwyżej $k$ wież, będą to jednak ulepszone wieże. Pomóż mu.*

## Wejście

*W pierwszym wierszu standardowego wejścia znajdują się trzy liczby całkowite $n$, $m$ oraz $k$ ($2 \leqslant n \leqslant 500\,000$, $0 \leqslant m \leqslant 1\,000\,000$, $1 \leqslant k \leqslant n$), rozdzielone pojedynczymi odstępami, oznaczające, odpowiednio, liczbę miast i dróg w państwie Bajtusia oraz liczbę $k$ wypowiedzianą przez Bajtunię. Miasta w państwie Bajtusia są ponumerowane liczbami od 1 do $n$. Dalej następuje $m$ wierszy opisujących drogi. W każdym wierszu znajdują się dwie liczby całkowite $a_i$, $b_i$ ($1 \leqslant a_i, b_i \leqslant n$, $a_i \neq b_i$) oznaczające, że miasta o numerach $a_i$ i $b_i$ łączy bezpośrednia dwukierunkowa droga. Każda para miast jest połączona co najwyżej jedną drogą.*

## Wyjście

*Twój program powinien wypisać na standardowe wyjście dwa wiersze opisujące rozmieszczenie ulepszonych wież w państwie Bajtusia. Pierwszy wiersz powinien zawierać liczbę całkowitą $r$ ($1 \leqslant r \leqslant k$), oznaczającą liczbę ulepszonych wież, które powinien zbudować Bajtuś. Drugi wiersz powinien zawierać opis rozmieszczenia tych wież: $r$ parami różnych liczb całkowitych* *oznaczających numery miast, w których należy zbudować ulepszone wieże strażnicze. Numery miast można podać w dowolnej kolejności.*

*W przypadku, gdy istnieje więcej niż jedno rozwiązanie, wystarczy wypisać dowolne z nich. Zwracamy uwagę, że należy wypisać dowolne rozmieszczenie nie więcej niż $k$ ulepszonych wież – nie trzeba używać minimalnej możliwej liczby ulepszonych wież. Możesz założyć, że Bajtunia nie pomyliła się, tzn. że całe państwo Bajtusia można strzec przy pomocy $k$ zwykłych (nieulepszonych) wież. W szczególności oznacza to, że zawsze istnieje rozwiązanie.*

## Przykład

*Dla danych wejściowych:*

```
9 8 3
1 2
2 3
3 4
1 4
3 5
4 6
7 8
8 9
```

*poprawnym wynikiem jest:*

```
3
1 5 7
```

**Testy „ocen”:**

`1ocen:` $n = m = 10$, $k = 5$, *sieć dróg tworzy cykl;*

`2ocen:` $n = 1414$, $m = 998\,991$, $k = 100$, *każda para miast jest połączona drogą; zauważ, że w tym przypadku wystarczyłoby wybudować tylko jedną wieżę;*

`3ocen:` $n = 500\,000$, $m = 499\,999$, $k = 250\,000$, *sieć dróg tworzy ścieżkę.*

# Rozwiązanie

Naturalnym jest, by państwo Bajtusia reprezentować jako nieskierowany graf, w którym wierzchołki odpowiadają miastom, a krawędzie – bezpośrednim drogom między nimi. Przełóżmy więc pozostałe definicje z treści zadania na język teorii grafów.

Niech $G = (V, E)$ będzie grafem o zbiorze wierzchołków $V$ i zbiorze nieskierowanych krawędzi $E$. Powiemy, że dwa wierzchołki $u, v \in V$ są *w odległości nie większej niż* $r$, jeśli w grafie $G$ istnieje ścieżka łącząca $u$ i $v$, zawierająca co najwyżej $r$ krawędzi. Dla liczby całkowitej dodatniej $r$, zbiór wierzchołków $X$ w grafie $G = (V, E)$ nazwiemy $r$*-dominującym*, jeśli dla każdego wierzchołka ze zbioru $V$ istnieje wierzchołek ze zbioru $X$ w odległości nie większej niż $r$. Zbiór 1-dominujący będziemy nazywać zazwyczaj po prostu *dominującym.*

Zauważmy, że $X = V$ jest zbiorem dominującym w grafie $G$. Ciekawym problemem jest jednak znajdowanie *jak najmniejszego* zbioru dominującego lub $r$-dominującego. W szczególności, problem postawienia $k$ (nieulepszonych) wież strażniczych w państwie Bajtusia to, używając właśnie wprowadzonej definicji, problem znalezienia w danym grafie zbioru dominującego o co najwyżej $k$ wierzchołkach. Stwierdzenie Bajtuni możemy więc przetłumaczyć na język teorii grafów jako „W tym grafie istnieje zbiór dominujący o $k$ wierzchołkach!”.

Co zaś zmieniają ulepszone wieże? Zauważmy, że problem postawienia $k$ ulepszonych wież to tak naprawdę problem znalezienia zbioru 2-dominującego o co najwyżej $k$ wierzchołkach. Zadanie, które ma przed sobą Bajtuś, brzmi więc:

> Wiedząc, że w grafie $G$ istnieje zbiór dominujący wielkości $k$, znajdź w $G$ zbiór 2-dominujący wielkości $k$.

## Rozwiązanie wzorcowe

Opiszmy najpierw rozwiązanie wzorcowe, na które wpadła znaczna część uczestników. Spróbujmy postąpić zachłannie: dopóki istnieje choć jedno niepilnowane miasto, postawmy w nim ulepszoną wieżę. A w języku teorii grafów: konstruujemy zbiór wierzchołków $X$, zaczynając od $X = \emptyset$, i, dopóki $X$ nie jest zbiorem 2-dominującym, bierzemy dowolny wierzchołek $v$ w odległości większej niż 2 od wszystkich wierzchołków z $X$ i dodajemy go do zbioru $X$. Oczywiście, w ten sposób otrzymamy zbiór 2-dominujący. Nie jest jednak jasne, że nie postawimy w ten sposób więcej niż $k$ wież.

By to pokazać, rozważmy zbiór dominujący $Z$ wielkości co najwyżej $k$, który miała na myśli Bajtunia. Przeanalizujemy jeden krok naszego algorytmu, w którym do zbioru $X$ dodajemy wierzchołek $v$ (czyli stawiamy ulepszoną wieżę w $v$). Zgodnie z definicją zbioru dominującego, istnieje wierzchołek $z \in Z$ w odległości co najwyżej 1 od $v$. Poczyńmy kluczową obserwację: każdy wierzchołek, pilnowany przez (nieulepszoną) wieżę postawioną w wierzchołku $z$, jest również pilnowany przez ulepszoną wieżę postawioną w $v$. Innymi słowy, w tym kroku zbiór wierzchołków pilnowanych przez postawione dotychczas ulepszone wieże „połyka” wszystkie wierzchołki pilnowane przez (nieulepszoną) wieżę w $z$. W związku z tym w każdym kroku algorytmu mamy do czynienia z innym wierzchołkiem $z$. A ponieważ $|Z| \leqslant k$, więc wykonamy nie więcej niż $k$ kroków, czyli postawimy co najwyżej $k$ ulepszonych wież.

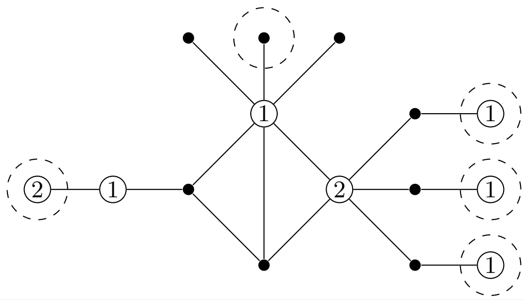
### Rozwiązanie wzorcowe w innym języku

Powyższy opis dla niektórych czytelników może wydać się za mało formalny. Spróbujmy więc uczynić go przejrzystszym. Posłuży nam do tego dodatkowa definicja. Zbiór wierzchołków $Y$ nazwiemy *$r$-rozrzuconym*, jeśli dowolne dwa różne wierzchołki ze zbioru $Y$ są w odległości większej niż $r$. Zbiór $r$-rozrzucony $Y$ jest *maksymalny ze względu na zawieranie*, jeśli nie można do niego dodać już żadnego innego wierzchołka, tj. zbiór $Y \cup \{v\}$ nie jest $r$-rozrzucony dla każdego wierzchołka $v \in V \setminus Y$. Na rysunku 1 pokazany jest przykład zbioru dominującego, 2-dominującego i 3-rozrzuconego.

Poczyńmy następujące dwie proste obserwacje, które powiążą tę definicję ze zbiorami dominującymi.

**Lemat 1.** Jeśli zbiór $Y$ jest maksymalnym ze względu na zawieranie zbiorem $r$-rozrzuconym, to jest też zbiorem $r$-dominującym.

**Dowód:** Załóżmy, że $Y$ jest zbiorem $r$-rozrzuconym, ale nie jest zbiorem $r$-dominującym. Wówczas istnieje wierzchołek $v \in V$ taki, że każdy wierzchołek




Rys. 1: Przykładowy graf z zaznaczonym zbiorem dominującym wielkości 5 (wierzchołki z jedynkami), zbiorem 2-dominującym wielkości 2 (wierzchołki z dwójkami) i zbiorem 3-rozrzuconym wielkości 5 (wierzchołki otoczone przerywaną linią). Jako że przedstawiony zbiór 3-rozrzucony jest oczywiście też zbiorem 2-rozrzuconym, zgodnie z lematem 2, w tym grafie nie istnieje zbiór dominujący o mniej niż pięciu wierzchołkach.

$y \in Y$ jest w odległości większej niż $r$ od $v$; w szczególności $v \notin Y$. Ale wtedy $Y \cup \{v\}$ jest również zbiorem $r$-rozrzuconym, czyli $Y$ nie jest maksymalny ze względu na zawieranie. ■

**Lemat 2.** Jeśli graf $G$ zawiera zbiór $2r$-rozrzucony wielkości $k$, to każdy zbiór $r$-dominujący w grafie $G$ ma co najmniej $k$ wierzchołków.

**Dowód:** Niech $Y$ będzie zbiorem $2r$-rozrzuconym w grafie $G$, a $Z$ zbiorem $r$-dominującym. Weźmy dowolny wierzchołek $y \in Y$. Zgodnie z definicją zbioru $r$-dominującego, istnieje wierzchołek $z \in Z$ w odległości nie większej niż $r$ od wierzchołka $y$. Zauważmy, że nie może się zdarzyć tak, że dwa wierzchołki $y_1, y_2 \in Y$ są w odległości nie większej niż $r$ od jednego wierzchołka $z \in Z$, gdyż wówczas $y_1$ i $y_2$ byłyby w odległości nie większej niż $2r$, co przeczyłoby definicji zbioru $2r$-rozrzuconego. Zatem $|Y| \leqslant |Z|$, co kończy dowód lematu. ■

Zauważmy, że wcześniej omawiany algorytm tak naprawdę w zachłanny sposób konstruuje maksymalny w sensie zawierania zbiór 2-rozrzucony. Lemat 1 pokazuje, że na końcu otrzymamy zbiór 2-dominujący. Lemat 2 pokazuje zaś, że nie będzie miał on więcej niż $k$ wierzchołków. Zwróćmy uwagę na podobieństwo argumentów użytych w dowodzie lematu 2 i poprzedniego opisu poprawności algorytmu: jest to tak naprawdę to samo rozumowanie.

### Implementacja

Zadanie wymagało zaimplementowania omówionego rozwiązania zachłannego tak, by działało w czasie liniowym od wielkości grafu wejściowego. Rozważmy następującą naturalną implementację: gdy dodajemy wierzchołek $v$ do konstruowanego zbioru $X$ (stawiamy ulepszoną wieżę w $v$), przeglądamy wszystkie wierzchołki w odległości co najwyżej 2 od wierzchołka $v$ i odznaczamy je jako „strzeżone”. W kolejnym kroku algorytmu szukamy dowolnego niestrzeżonego jeszcze wierzchołka i stawiamy tam kolejną wieżę.

Jak szybko jesteśmy w stanie przejrzeć wierzchołki strzeżone przez nowo postawioną wieżę? Musimy do tego przejrzeć listę sąsiadów $v$ oraz listę sąsiadów każdego wierzchołka połączonego bezpośrednią drogą z wierzchołkiem $v$. Zauważmy, że jeśli przejrzymy listę sąsiadów wierzchołka $w$ zarówno przy okazji stawiania wieży w wierzchołku $v_1$, jak i w wierzchołku $v_2$, to odległość między $v_1$ i $v_2$ wynosi co najwyżej 2, co jest w sprzeczności z zasadą działania naszego algorytmu. Tak więc przejrzymy listę sąsiadów każdego wierzchołka co najwyżej raz, zatem algorytm działa w czasie liniowym od wielkości grafu.

Rozwiązanie wzorcowe można znaleźć w plikach `gra.cpp`, `gra1.pas` i `gra2.cpp`.

## Testy

Przygotowano 11 zestawów testowych, z których każdy zawiera co najmniej 6 testów. Testy zostały wygenerowane w sposób losowy, zapewniając istnienie pokrycia $k$ nieulepszonymi wieżami.

## Kilka słów o podobnych, ale trudniejszych problemach

Chcielibyśmy wykorzystać zadanie *Gra Tower Defense* jako pretekst, by opowiedzieć o podobnych problemach, którymi zajmują się badacze i które stanowią przedmiot niejednej publikacji naukowej w dziedzinie algorytmiki.

### Jak sprawdzić czy Bajtunia miała rację?

Bardzo ciekawym pytaniem jest, skąd Bajtunia wiedziała, że w państwie Bajtusia wystarczy $k$ wież. I jak Bajtuś mógłby sprawdzić, czy Bajtunia ma rację? W języku teorii grafów nasze pytanie brzmi:

> Mając dany graf $G$ i liczbę $k$, sprawdź, czy w grafie $G$ istnieje zbiór dominujący o co najwyżej $k$ wierzchołkach.

Okazuje się, że jest to problem NP-trudny, co oznacza, że nie spodziewamy się algorytmu rozwiązującego go, który działałby w czasie wielomianowym od wielkości grafu $G$. Co więcej, jest to też problem co najmniej tak trudny jak problemy w być może dość egzotycznej klasie $W[2]$, co z kolei oznacza (pomijając tutaj skomplikowaną definicję tej klasy), że raczej nie spodziewamy się algorytmu działającego istotnie szybciej niż algorytm kompletnie naiwny, sprawdzający wszystkie możliwe rozwiązania i działający w czasie mniej więcej $O(n^k)$.

Jedną z metod, jakimi staramy się radzić sobie z problemami bardzo trudnymi, jest *aproksymacja*: zamiast próbować rozwiązać problem dokładnie, postarajmy się dać może niekoniecznie optymalne, ale dość dobre rozwiązanie. W naszym problemie przekłada się to na następujące sformułowanie:

> Mając dany graf $G$ i liczbę $k$, stwierdź, że w $G$ nie istnieje zbiór dominujący o co najwyżej $k$ wierzchołkach, lub podaj zbiór dominujący o co najwyżej $\alpha k$ wierzchołkach.

Parametr $\alpha$ nazywamy *współczynnikiem aproksymacji*; im jest on mniejszy, tym algorytm jest lepszy – lepiej przybliża nam prawdziwe rozwiązanie.

Nie jest bardzo trudno pokazać (zachęcamy Czytelnika do próby samodzielnego dowodu), że dość naturalny algorytm zachłanny (który buduje wieżę w takim wierzchołku, by strzegła jak najwięcej dotychczas niestrzeżonych wierzchołków) jest algorytmem aproksymacyjnym o współczynniku aproksymacji $\alpha = 1 + \ln n$, gdzie ln oznacza logarytm naturalny. Innymi słowy, w grafie, w którym istnieje zbiór dominujący wielkości $k$, algorytm ten wyznaczy zbiór dominujący wielkości co najwyżej $k(1 + \ln n)$. Okazuje się, że lepiej się nie da, co (przy pewnych dość standardowych założeniach teoriozłożonościowych) wykazał Uriel Feige w 1996 roku [41].

Podsumowując, nie mamy pojęcia, jak Bajtunia odkryła, że można strzec całe państwo Bajtusia przy pomocy tylko $k$ nieulepszonych wież.

**Problem remiz strażackich**

W naszym zadaniu zastosowaliśmy inny kierunek aproksymacji: zamiast stawiać więcej wież strażniczych, Bajtuś budował ulepszone wieże. To przybliża nas do następującego problemu, który najczęściej formułowany jest w języku budowy remiz strażackich.

Mamy dany graf $G = (V, E)$, w którym wierzchołki grafu odpowiadają różnym ważnym częściom miasta lub państwa, a krawędzie – połączeniom między nimi. Chcemy postawić w niektórych wierzchołkach remizy strażackie. Naszym celem jest, by do każdego wierzchołka naszego grafu straż pożarna dojeżdżała możliwie najszybciej. Formalnie, jeśli $X$ będzie zbiorem wierzchołków, w których zbudujemy remizy, to miarą jakości naszego rozwiązania jest

$$\max_{v \in V} \min_{x \in X} \text{odległość}(v, x).$$

Innymi słowy, dla każdego wierzchołka $v$ patrzymy, gdzie jest najbliższa remiza do wierzchołka $v$, i znajdujemy ten wierzchołek, który ma najdalej do najbliższej remizy.

Oczywiście, idealnym rozwiązaniem byłoby zbudować remizę w każdym wierzchołku naszego grafu. Niestety, mamy ograniczony budżet: możemy zbudować tylko $k$ remiz. Jak więc je rozmieścić?

Zauważmy, że problem Bajtusia to tak naprawdę problem rozmieszczenia $k$ remiz tak, by straż pożarna była w odległości 1 od każdego wierzchołka, jeśli Bajtuś używa nieulepszonych wież, a w odległości 2 w przypadku wież ulepszonych. Spróbujmy więc uogólnić nasze rozwiązanie. Załóżmy, że ktoś nam powiedział, że można rozmieścić te $k$ remiz tak, by straż pożarna była w odległości co najwyżej $r$ od każdego wierzchołka grafu. Rozważmy następujący algorytm, analogiczny do rozwiązania wzorcowego: tak długo, jak istnieje wierzchołek, od którego najbliższa remiza jest w odległości większej niż $2r$, stawiamy w jednym takim wierzchołku remizę. Omówione wcześniej rozumowanie przechodzi właściwie bez zmian: postawimy co najwyżej $k$ remiz.

Dostajemy więc algorytm o współczynniku aproksymacji 2 dla problemu remiz strażackich. Zauważmy, że nie możemy liczyć na algorytm o lepszym współczynniku aproksymacji, gdyż wówczas taki algorytm mógłby (dokładnie) rozwiązywać problem zbioru dominującego. A ten problem, jak już mówiliśmy, jest bardzo trudny.

Omówiony algorytm został zauważony już w latach 80. ubiegłego wieku [40]. Od tego czasu naukowcy badali różne warianty problemu remiz strażackich, analizując, które z nich mają równie dobre algorytmy aproksymacyjne. Jednym z najciekawszych kierunków badań jest rozważanie wariantu *z pojemnościami*.

Załóżmy, że w naszym grafie istnieje jeden wierzchołek $v$, z którego prowadzi bezpośrednia droga do każdego innego wierzchołka grafu. Wówczas optymalnym rozwiązaniem jest postawić jedną remizę strażacką w $v$. Czy to jednak jest dobre rozwiązanie? Jeśli miasto, reprezentowane przez nasz graf, jest naprawdę duże, to może nie wystarczyć, mimo, że każdy wierzchołek jest blisko remizy. W dużym mieście może wybuchnąć kilka pożarów w różnych częściach miasta, i strażaków z jednej remizy będzie za mało, by je wszystkie ugasić.

Dodajmy więc do naszego zadania założenie o *pojemnościach*: jedna remiza strażacka jest w stanie pilnować co najwyżej $L$ wierzchołków. Formalnie, nasze zadanie zdefiniowane jest następująco.

> Mając dany graf $G$ i liczby $k$ oraz $L$, znajdź zbiór wierzchołków $X$ wielkości $k$ oraz funkcję $f\colon V \to X$ tak, by dla każdego $x \in X$ zachodziło $|f^{-1}(x)| \leqslant L$ oraz by zminimalizować wielkość
>
> $$\max_{v \in V} \text{odległość}(v, f(v)).$$

Funkcja $f$ przyporządkowuje każdemu wierzchołkowi $v$ remizę $f(v)$, która strzeże $v$. Warunek $|f^{-1}(x)| \leqslant L$ oznacza, że remiza w wierzchołku $x$ strzeże co najwyżej $L$ wierzchołków.

Problem z pojemnościami okazuje się istotnie trudniejszy. Khuller i Sussmann w 2000 roku pokazali algorytm 6-aproksymacyjny dla tego wariantu [42]. Jeśli dodatkowo utrudnimy sobie zadanie i założymy, że pojemności mogą się różnić między wierzchołkami (tzn. każdy wierzchołek $x \in V$ ma daną pojemność $L(x)$, i remiza postawiona w wierzchołku $x$ może strzec co najwyżej $L(x)$ wierzchołków), wówczas otrzymujemy problem, do którego pierwszy znany algorytm o stałym współczynniku aproksymacji został odkryty dopiero w 2012 roku [43], a obecnie najlepszy znany algorytm ma współczynnik aproksymacji 9 [44].