Wojciech Rytter
Treść zadania, Opracowanie

Jakub Radoszewski
Program

OI, Etap II, dzień drugi, 8.02.2008

Mafia

Ostatnio Bajtocją Równikową wstrząsają mafijne porachunki. Do Bajtogrodu, stolicy tego kraju, zjechali się przywódcy mafijni, aby zakończyć spory. W pewnym momencie rozmowy stały się bardzo nerwowe i uczestnicy spotkania wyciągnęli broń. Każdy z nich trzyma w ręku pistolet i celuje w jednego innego uczestnika. Jeśli dojdzie do strzelaniny, to zgodnie z kodeksem honorowym:

- *uczestnicy będą strzelać w pewnej kolejności, przy czym jednocześnie może strzelać tylko jeden z nich,*
- *żaden z nich nie pudłuje, a ofiara natychmiast umiera i nie może strzelać,*
- *każdy z nich strzeli dokładnie jeden raz, o ile wcześniej sam nie zostanie zastrzelony,*
- *żaden uczestnik nie może zmienić wybranego na początku celu, nawet jeżeli osoba, do której celuje, zostanie zastrzelona (w takim wypadku strzał nie zmienia liczby ofiar).*

Przypadkowo sytuację obserwuje przedsiębiorca pogrzebowy, zaprzyjaźniony ze wszystkimi obecnymi. Chce on oszacować możliwą liczbę ofiar: minimalną i maksymalną. Przedsiębiorca pogrzebowy wie, kto w kogo celuje, ale nie zna kolejności strzelania. Twoim zadaniem jest napisanie programu, który te liczby wyliczy.

Zadanie

Napisz program, który:

- *wczyta ze standardowego wejścia opis celów, jakie obrali sobie poszczególni mafiozi,*
- *wyznaczy minimalną i maksymalną liczbę ofiar strzelaniny,*
- *wypisze wynik na standardowe wyjście.*

Wejście

W pierwszym wierszu wejścia zapisana jest liczba uczestników n ($1 \leqslant n \leqslant 1\,000\,000$). Są oni ponumerowani od 1 do n. W drugim wierszu znajduje się n liczb całkowitych $s_1, s_2, \ldots, s_n$ ($1 \leqslant s_i \leqslant n$), pooddzielanych pojedynczymi odstępami. Liczba s_i jest numerem uczestnika, w którego celuje uczestnik nr i. Może się zdarzyć, że $s_i = i$.

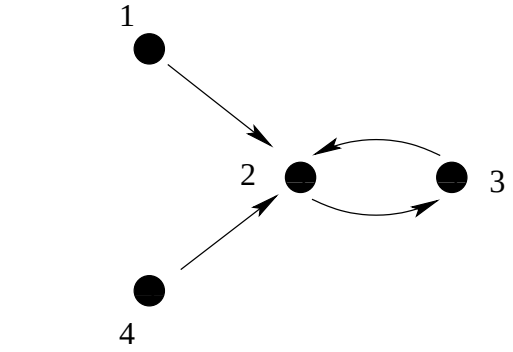
Wyjście

Twój program powinien wypisać w pierwszym i jedynym wierszu wyjścia dwie liczby całkowite, oddzielone pojedynczym odstępem: minimalną i maksymalną liczbę ofiar, będących końcowym wynikiem strzelaniny.

Przykład

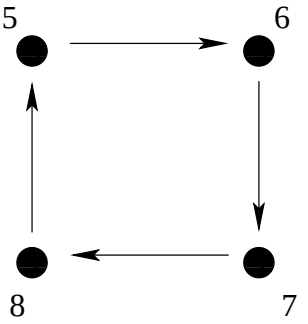
Dla danych wejściowych:

```
8
2 3 2 2 6 7 8 5
```



poprawnym wynikiem jest:

```
3 5
```



Rozwiązanie

Strzelanina w języku grafów

Zacznijmy od zapisania zadania w języku teorii grafów. Po pierwsze pozwoli nam to jaśniej przedstawić problem i skorzystać z rozwiązań znanych w tej dziedzinie. Po drugie uwolnimy się w ten sposób trochę od dość ponurej treści zadania. Niech $G = (V, E)$ będzie grafem skierowanym, w którym wierzchołki V odpowiadają uczestnikom spotkania, a krawędź $(i, j) \in E$ oznacza, że uczestnik i celuje w uczestnika j. Dla krawędzi (i, j) będziemy mówić, że:

- i wskazuje na j albo
- i jest *synem* j, a j jest *ojcem* i, co zapiszemy $parent(i) = j$.

Poszukiwane rozwiązanie zadania to dwa zbiory wierzchołków: o minimalnej oraz maksymalnej liczności, takie że... No właśnie, jeszcze dokładnie nie wiadomo, co to za zbiory — powrócimy do tej sprawy później.

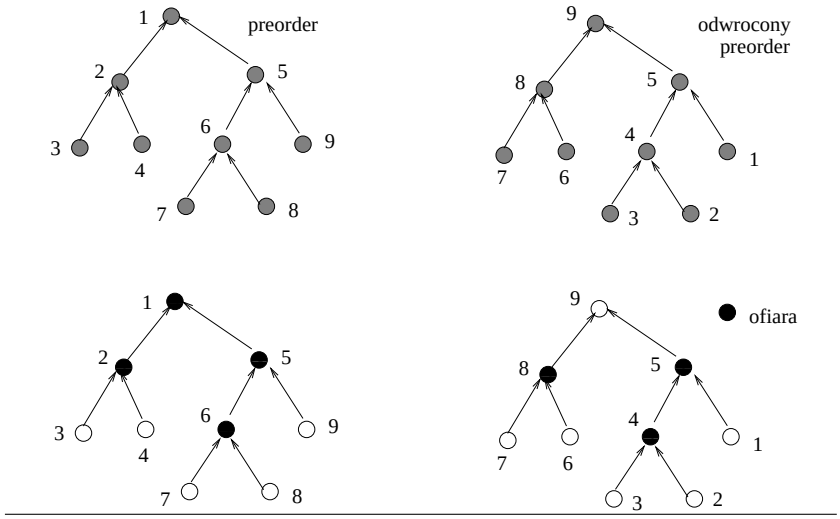
W grafie G każdy wierzchołek ma stopień wyjściowy równy 1, co istotnie wpływa na jego postać. Rozważmy jedną słabo spójną składową G (czyli spójną składową w wersji grafu G, w której ignorujemy skierowanie krawędzi). Jest ona drzewem z jedną dodatkową krawędzią, która łączy korzeń (*root*) z pewnym wierzchołkiem wewnątrz drzewa. Każdą składową możemy zająć się oddzielnie, więc od teraz przyjmiemy, że $G = T = (V, E)$ jest drzewem z dodatkową krawędzią $(root, parent(root))$, którą nazwiemy *złą krawędzią*. Załóżmy też, że graf T ma co najmniej dwa wierzchołki (w przeciwnym przypadku rozwiązanie obu części zadania jest dla niego trywialne).

Niech $|V| = n$. Usuńmy na chwilę złą krawędź z T i ponumerujmy węzły w porządkach *preorder* i *postorder*. Każda z tych numeracji przypisuje węzłom różne wartości ze zbioru

$\{1,2,\ldots n\}$ tak, by dla każdego węzła $i \neq root$ spełnione były warunki:

$$preorder(parent(i)) < preorder(i), \tag{1}$$
$$postorder(parent(i)) > postorder(i). \tag{2}$$

Jest wiele funkcji, które spełniają powyższe wymagania — w szczególności numeracja odwrotna do *preorder* jest pewnym porządkiem *postorder* i odwrotnie (na rys. 1 są pokazane przykładowe numeracje).



Rys. 1: Rozważmy graf powstały ze słabo spójnej składowej przez usunięcie złej krawędzi. W górnym rzędzie, po lewej stronie jest przedstawiona numeracja *preorder*, a po prawej — numeracja *postorder*. W dolnym rzędzie zaznaczono efekty strzelaniny: zaczernione węzły to ofiary, gdyby uczestnicy strzelali odpowiednio w porządku *preorder* i *postorder*.

Minimalna liczba ofiar. *Zbiór niezależny* węzłów grafu definiujemy jako zbiór $N \subseteq V$, taki że

$$i \in N \quad \Rightarrow \quad parent(i) \notin N. \tag{3}$$

Co ciekawe, alternatywna definicja zbioru niezależnego:

$$parent(i) \in N \quad \Rightarrow \quad i \notin N$$

jest równoważna definicji (3).

Można łatwo spostrzec, że zbiór węzłów, które przetrwają strzelaninę, musi być zbiorem niezależnym — żaden z „ocalałych" węzłów nie może celować do innego „ocalałego". Niestety, nie dla każdego zbioru niezależnego da się zaplanować taki przebieg strzelaniny, by ocalały dokładnie wierzchołki z tego zbioru — na przykład pojedynczy wierzchołek jest zbiorem niezależnym, a rzadko kiedy uda się trafić wszystkich oprócz niego. Jednak jeśli mamy maksymalny w sensie zawierania zbiór niezależny zawierający wszystkie liście T (czyli wierzchołki o stopniu wejściowym 0), to taki przebieg już istnieje.

Fakt 1. *Jeśli N jest maksymalnym (w sensie zawierania) zbiorem niezależnym zawierającym wszystkie liście w grafie T, to istnieje taki przebieg strzelaniny, w wyniku którego ocaleją dokładnie węzły ze zbioru N.*

Dowód: Wystarczy zaplanować przebieg strzelaniny tak, by najpierw trafić wszystkie wierzchołki celujące w jakiekolwiek wierzchołki ze zbioru N (oznaczmy zbiór tych wierzchołków „zagrażających" N przez C). Zauważmy, że każdy wierzchołek ze zbioru C może zostać trafiony, gdyż:

- nie mogą to być liście;
- stąd w każdy węzeł $v \in C$ celuje jakiś węzeł $w \in V \setminus C$; oczywiście w nie celuje wówczas w żaden wierzchołek należący do N.

Pozwólmy więc w pierwszej fazie strzelić wszystkim, którzy zlikwidują wierzchołki zagrażające elementom zbioru N. Teraz już zbiór N jest bezpieczny i dalszy (właściwie dowolny) przebieg strzelaniny zawsze pozostawi zbiór niezależny zawierający w sobie zbiór N. Z założenia o maksymalności N wynika, że musi to być dokładnie zbiór N. ■

Na mocy powyższego faktu możemy stwierdzić, że istnieje w T kolejność strzelania, w wyniku której przeżyją wszystkie wierzchołki z pewnego *najliczniejszego* zbioru niezależnego zawierającego wszystkie liście. Łatwo zauważyć, że jest to maksymalna liczba ocalałych, jaką da się uzyskać w T.

W przykładzie przedstawionym na rys. 1 bliski optymalnemu efekt uzyskujemy, „strzelając" w porządku *postorder* (z dokładnością do usuniętej złej krawędzi). Nie jest to przypadek — wykażemy potem, że właśnie porządek postorder pozwala łatwo wyznaczyć najliczniejszy zbiór niezależny.

Maksymalna liczba ofiar. Poszukując maksymalnej liczby ofiar, nie będziemy musieli uciekać się do żadnych zaawansowanych pojęć. Wystarczy zauważyć, że „strzelając" w pewnym porządku preorder[1], można zmaksymalizować liczbę ofiar — da się trafić wszystkich oprócz liści (do których nikt nie celuje, więc nikt ich nie trafi) i być może jednego wierzchołka (nie byłoby tego warunku, gdyby nie zła krawędź).

Algorytm MAX-MIN

Oznaczmy przez *liczba-ofiar*(π) liczbę ofiar, gdy uczestnicy strzelają w kolejności π. Rozważmy porządek postorder wyznaczony w drzewie, tj. w grafie T z pominiętą złą krawędzią. Oznaczmy przez $postorder(V)$ ciąg węzłów posortowany rosnąco względem wartości postorder.

```
1: ALGORYTM MAX-MIN(T)
2:     { Graf T jest pojedynczym drzewem o n węzłach z jedną złą krawędzią. }
3:     { Wynikiem są dwie szukane liczby: }
4:     {   MAX — maksymalna liczba ofiar, }
```

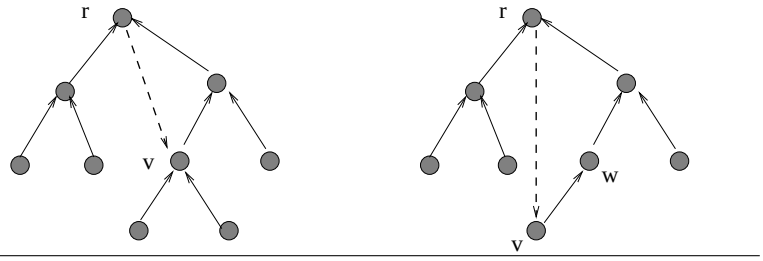
[1] tj. w dowolnym porządku preorder dla pewnego ukorzenienia T — dokładny opis doboru odpowiedniego korzenia znajduje się w dalszej części tekstu.

```
5:  { MIN — minimalna liczba ofiar. }
6:  k := liczba liści w grafie T;
7:  MAX := min(n − k, n − 1);
8:  π := postorder(V);
9:  π′ := cykliczna rotacja π;
10: { π′ tworzymy z π, przesuwając ostatni element na początek. }
11: MIN := min(liczba-ofiar(π), liczba-ofiar(π′));
12: return (MAX, MIN);
```

Wartość MAX jest poprawna

Dowód poprawności tej części algorytmu jest stosunkowo łatwy. Jeśli T jest pojedynczym cyklem zawierającym $n > 1$ węzłów (przypomnijmy, że przypadek $n = 1$ odrzuciliśmy w początkowej analizie), to maksymalna liczba ofiar wynosi $n - 1$. Załóżmy zatem, że T jest nietrywialnym drzewem z cyklem, tzn. ma $k > 0$ liści. Niewątpliwie każdą strzelaninę „przeżyją" wszystkie liście, ponieważ nikt do nich nie celuje.

Zauważmy, że każdy z pozostałych węzłów może zostać trafiony. Faktycznie, gdyby w T nie było złej krawędzi, to w wyniku strzelaniny przeprowadzonej w porządku preorder przy życiu pozostałyby jedynie liście (a dokładniej, liście w podgrafie T niezawierającym złej krawędzi). Zła krawędź $(root, parent(root))$ może zmodyfikować końcowy efekt strzelania — *root* strzela jako pierwszy, eliminuje wierzchołek $parent(root)$, przez co $parent(parent(root))$ może ocaleć. Zauważmy jednak, że wszystko będzie dobrze, jeśli węzeł $parent(parent(root))$ ma jakiegoś syna oprócz $parent(root)$ — wówczas w trakcie strzelaniny w porządku preorder $parent(parent(root))$ zostanie trafiony mimo tego, że $parent(root)$ ginie już w pierwszym strzale (patrz także rys. 2). Jeżeli poprzedni warunek nie zachodzi, to na cyklu zaczynającym się w korzeniu drzewa możemy znaleźć jakiś inny węzeł w taki, że $parent(w)$ ma jeszcze jednego syna $w' \neq w$ (gdyż T jest nietrywialnym drzewem z cyklem). Ponieważ każdy wierzchołek z cyklu może zostać uznany za korzeń drzewa, możemy wybrać także poprzednika w na tym cyklu. Zauważmy, że wówczas po strzelaninie zgodnie z porządkiem preorder pozostaną nietrafione tylko liście (patrz rys. 2).



Rys. 2: Na rysunku po lewej porządek preorder wystarczy, by trafić wszystkie węzły oprócz liści. Na rysunku po prawej najpierw musimy inaczej wybrać korzeń drzewa — powinien nim być wierzchołek, którego ojciec ma „brata" (tzn. ojciec ojca korzenia powinien mieć drugiego syna). Jako korzeń wybieramy zatem poprzednika w na cyklu, czyli w tym przypadku v — wówczas $w = parent(v)$ nie jest „jedynakiem" i „strzelamy" w porządku preorder wyznaczonym w drzewie bez złej krawędzi (v, w).

Wartość MIN jest poprawna

Ta część algorytmu jest mniej trywialna. Możemy jednak wykazać jej poprawność, bazując na prostszym przypadku drzewa bez złej krawędzi. Oznaczmy, jak poprzednio, przez $T = (V, E)$ drzewo ze złą krawędzią (r, p). Ustaliliśmy już, że zbiór „pozostałych przy życiu wierzchołków" musi być najliczniejszym zbiorem niezależnym zawierającym wszystkie liście w grafie.

Najliczniejszy zbiór niezależny w drzewie. Wyrzućmy na moment z grafu T złą krawędź (r, p); niech $T' = T \setminus \{(r, p)\}$. Rozważmy następujący algorytm zachłanny.

Algorytm A**:** Konstruuj zbiór niezależny w T', wybierając do niego za każdym razem najgłębiej położony niewybrany jeszcze wierzchołek, którego żaden z synów nie znajduje się jeszcze w zbiorze.

Okazuje się, że:

Fakt 2. *Algorytm A jest optymalny dla T', to znaczy konstruuje najliczniejszy zbiór niezależny w T'.*

Dowód: Niech N oznacza zbiór niezależny skonstruowany przez A, natomiast N' — dowolny najliczniejszy zbiór niezależny w T'. Pokażemy, że jeżeli $N \neq N'$, to można bez zmniejszenia liczności N' sprowadzić ten zbiór do N, a zatem że N jest również najliczniejszy.

Niech v będzie najgłębiej położonym wierzchołkiem drzewa, który odróżnia N i N', czyli najgłębiej położonym wierzchołkiem ze zbioru $N \oplus N' = (N \setminus N') \cup (N' \setminus N)$. Na mocy kryterium wyboru v, żaden z synów v nie należy ani do N, ani do N'. Ze względu na metodę zachłanną wykorzystywaną w algorytmie A, mamy zatem $v \in N$, czyli $v \notin N'$.

Gdyby ojciec w wierzchołka v nie należał do N' (albo, tym bardziej, gdyby w nie istniał), to $N' \cup \{v\}$ byłby zbiorem niezależnym, co przeczyłoby temu, że N' jest najliczniejszym zbiorem niezależnym. Zamieńmy zatem N' na $N'' = (N' \setminus \{w\}) \cup \{v\}$. Otrzymamy wówczas inny najliczniejszy zbiór niezależny, który ma o jeden więcej wierzchołek wspólny z N (gdyż oczywiście $w \notin N$). Kontynuując ten proces, po maksymalnie $|N|$ krokach bez usuwania wierzchołków otrzymamy zbiór N. To kończy dowód optymalności A. ■

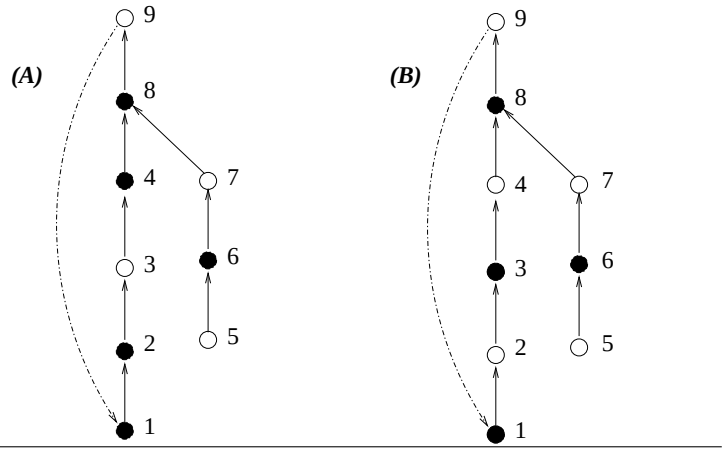
Jak praktycznie wykorzystać algorytm A? Otóż można zauważyć, że *porządek postorder* wyznacza kolejność strzelania, w wyniku której przy życiu pozostaną wierzchołki z dokładnie tego zbioru niezależnego, który jest konstruowany przez *algorytm A* (dowód tego faktu pozostawiamy Czytelnikowi jako ćwiczenie). Związek z algorytmem A w szczególności implikuje jedną istotną własność strzelania w porządku postorder, która przyda nam się w dalszej analizie.

Obserwacja 1. Jeżeli w T' istnieje najliczniejszy zbiór niezależny niezawierający korzenia, to zostanie on skonstruowany w wyniku strzelaniny w porządku postorder.

Najliczniejszy zbiór niezależny w drzewie ze złą krawędzią. Powróćmy teraz do drzewa T ze złą krawędzią (r, p). Zauważmy, że wyznaczony dla niego zbiór niezależny musi być także niezależny ze względu na krawędzie drzewa. Możemy więc rozpocząć od wyznaczenia najliczniejszego zbioru niezależnego N' w drzewie $T \setminus \{(r, p)\}$. Daje to nam jeden z dwóch przypadków:

- do zbioru N' nie należą jednocześnie węzły r i p, więc dołączenie do drzewa złej krawędzi nie psuje niezależności zbioru — to oznacza, że zbiór N' jest niezależny i optymalny także w grafie ze złą krawędzią;
- do zbioru N' należą jednocześnie r i p, więc w drzewie ze złą krawędzią zbiór N' nie jest już zbiorem niezależnym — moglibyśmy wyrzucić jeden z końców krawędzi (r,p), ale nie możemy być do końca pewni, czy dostajemy w ten sposób rozwiązanie optymalne.

Przyjrzyjmy się dokładniej drugiemu spośród powyższych przypadków, w którym obecność p oraz r w N' wymusza na nas wyrzucenie jednego z tych wierzchołków. Zauważmy, że drzewo może mieć kilka różnych najliczniejszych zbiorów niezależnych, więc warto sprawdzić, czy nie można się obejść bez wierzchołka p lub r. Na mocy Obserwacji 1, jeżeli zbiór niezależny N' skonstruowany za pomocą porządku postorder zawiera r, to każdy inny najliczniejszy zbiór niezależny w drzewie także zawiera r. Może jednak istnieć możliwość obejścia się bez p — oznaczmy najliczniejszy zbiór niezależny dla grafu T z usuniętym wierzchołkiem p przez N''. Porównując zbiory N'' i N' z ewentualnie usuniętym jednym z węzłów r, p, dostajemy zatem zbiór N — najliczniejszy zbiór niezależny dla drzewa za złą krawędzią.



Rys. 3: Węzły zaznaczone na czarno oznaczają węzły trafione (A): w porządku postorder $\pi = 1,2,3,4,5,6,7,8,9$ (5 trafień) oraz (B): w porządku $\pi' = 9,1,2,3,4,5,6,7,8$ (4 trafienia).

Skoro już wiemy, jak wyznaczyć najliczniejszy zbiór niezależny, to sprawdźmy, czy robi to algorytm MAX-MIN. Zauważmy, że:

- jeśli węzły strzelają w „zwykłym" porządku postorder π, to jako wynik dostajemy liczność zbioru niezależnego N', ewentualnie z usuniętym węzłem p, jeśli krawędź (r,p) psuła niezależność zbioru N';
- jeśli węzły strzelają w przesuniętym cyklicznie porządku postorder, czyli π', to wyznaczamy optymalny zbiór niezależny w grafie z wstępnie usuniętym węzłem p (r trafia go na samym początku) — jest to zbiór N''.

Widzimy więc, że algorytm bada wszystkie konieczne przypadki i zwraca optymalną z możliwych wartości, czyli jest poprawny!

Na rysunku 3 jest pokazany przypadek, gdy optymalny jest właśnie porządek π'.

Alternatywne nieformalne uzasadnienie dla wersji MIN. Jeżeli Czytelnikowi nie przypadł do gustu dowód poprawności wersji MIN wykorzystujący zbiory niezależne, to na koniec prezentujemy w skrócie alternatywne uzasadnienie tej części algorytmu MAX-MIN.

Zauważmy, że graf T o korzeniu r składa się z cyklu

$$C = r \to v_1 \to \ldots \to v_k \to r$$

i pewnego zbioru D drzew podwieszonych do C. Oznaczmy przez

$$A_{C1} : \; v_1 \to \ldots \to v_k \to r, \;\; A_{C2} : \; r \to v_1 \to \ldots \to v_k$$

dwie interesujące kolejności strzelania na cyklu C oraz niech A_D będzie kolejnością strzelania w drzewach ze zbioru D zgodną z porządkiem postorder.

Obserwacja 2. Algorytm MAX-MIN daje ten sam zbiór ocalałych, co algorytm minimalizujący liczbę ofiar z dwóch kolejności: $(A_D; A_{C1})$, $(A_D; A_{C2})$.

Krótkie uzasadnienie. Zauważmy, że pierwsza z tych kolejności odpowiada π, natomiast druga *prawie* odpowiada π' (gdzie π, π' — jak w algorytmie MAX-MIN). Rozpatrzmy dwa przypadki. Jeśli po wykonaniu A_D wierzchołek r jest martwy, to kolejność $(A_D; A_{C1})$ minimalizuje liczbę ofiar. W tym przypadku jest to po prostu kolejność postorder, optymalna dla drzewa $T \setminus \{(r,p)\}$, a r nie wykonuje strzału, więc zła krawędź niczego nie psuje. W przeciwnym przypadku π' odpowiada pod względem zbioru ofiar kolejności $(A_D; A_{C2})$, tzn. strzał wykonywany w π' przez r możemy przesunąć po A_D.

Obserwacja 3. Istnieje optymalna kolejność strzelania B, w której strzały z A_D padają na samym początku, tj. $B = (A_D; B_C)$.

Wynika to stąd, że każda strategia, w której wybrany liść strzela jako pierwszy, może być zawsze rozszerzona do algorytmu optymalnego. Można to udowodnić indukcyjnie ze względu na liczbę liści.

Wystarczy teraz stwierdzić następujący fakt.

Obserwacja 4. Jeden z ciągów A_{C1}, A_{C2} jest optymalny dla T będącego cyklem, w którym pewne wierzchołki mogą już być martwe, tzn. jeden z ciągów A_{C1}, A_{C2} jest równoważny ciągowi B_C w pewnym algorytmie optymalnym B.

Wynika to stąd, że w takim cyklu dowolny ciąg kolejnych (wzdłuż cyklu) strzelań zaczynający się od martwego wierzchołka (albo jakikolwiek taki ciąg, jeżeli nie istnieją martwe wierzchołki) jest optymalny. W ciągach A_{C1}, A_{C2} strzelanie zaczynamy z sąsiednich wierzchołków v_1 i r, dzięki czemu co najmniej raz otrzymamy ten sam zbiór ofiar, co przy strzelaniu optymalnym.

Złożoność algorytmu

Algorytm wymaga jedynie przejścia grafu w porządku postorder. Jest to prosta operacja wykonywana w czasie $O(n)$, czyli bez problemu radzimy sobie z danymi o zadanym rozmiarze.

Testy

Rozwiązania zawodników były sprawdzane na 13 zestawach testów. W poniższej tabelce znajdują się ich krótkie opisy.

Nazwa	n	Opis
maf1.in	9	mały losowy test
maf2.in	1 000	zróżnicowany średni test losowy (7 cykli, 21 pętli, 74 drzewa z cyklami)
maf3.in	2 000	zróżnicowany średni test losowy (10 cykli, 30 pętli, 169 drzew z cyklami)
maf4.in	3 000	zróżnicowany średni test losowy (17 cykli, 61 pętli, 336 drzew z cyklami)
maf5.in	3 500	zróżnicowany średni test losowy (19 cykli, 120 pętli, 462 drzewa z cyklami)
maf6.in	4 000	zróżnicowany średni test losowy (22 cykle, 97 pętli, 421 drzew z cyklami)
maf7.in	60 006	test złożony z 10 000 6-wierzchołkowych składowych z małym bonusem
maf8.in	10 000	test losowy z dodanymi cyklami (w tym jednowierzchołkowymi)
maf9.in	100 000	test losowy z dodanymi ponad 7 000 wierzchołków na cyklach oraz z 1 200 8-wierzchołkowymi składowymi
maf10.in	400 000	test losowy z dodanymi cyklami, zawierającymi łącznie 150 000 wierzchołków
maf11a.in	1 000 000	test losowy z dodanymi ponad 10 000 9-wierzchołkowych składowych oraz 120 000 wierzchołków leżących na cyklach
maf11b.in	1 000 000	test, w którym kilka wierzchołków ma bardzo duże stopnie wejściowe
maf12a.in	1 000 000	test losowy
maf12b.in	1 000 000	ścieżka długości milion dochodząca do pętli
maf13a.in	1 000 000	test losowy z dodanymi 10 000 10-wierzchołkowych składowych oraz 100 000 wierzchołków na cyklach
maf13b.in	1 000 000	cykl długości milion

W trakcie generowania testów wyszedł na jaw empiryczny fakt, że losowy graf (tj. graf, któremu odpowiada wejście w postaci losowego n-elementowego ciągu nad $\{1,2,\ldots,n\}$) składa się z małej liczby słabo spójnych składowych, które zazwyczaj są nietrywialnymi drzewami z cyklami (i oczywiście pewne z nich muszą być stosunkowo duże). Z tego względu najtrudniejsze testy w zestawie poszerzały testy losowe o słabo spójne składowe będące cyklami (jedno- i wielowierzchołkowymi) oraz o wiele bardzo małych losowych spójnych składowych — celem tych drugich było wykrycie rozmaitych rozwiązań błędnych.