Łukasz Kowalik
Treść zadania, Opracowanie

Marcin Stefaniak, Piotr Stańczyk
Program

OI, Etap II, dzień 2, 10–02–2005

Kości

Gra w kości jest grą dwuosobową, w pełni losową. W ostatnim czasie zdobywa ona w Bajtocji rosnącą popularność. W stolicy tego kraju istnieje nawet specjalny klub dla jej wielbicieli. Bywalcy klubu przyjemnie spędzają czas rozmawiając ze sobą i od czasu do czasu rozgrywają partyjkę swojej ulubionej gry z losowo napotkanym graczem. Osoby, które wygrają najwięcej rozgrywek danego dnia, zyskują miano **szczęściarza**. *Zdarza się, że wieczór w klubie upływa w spokojnej atmosferze i rozgrywanych jest niewiele partii. Wtedy nawet jedna wygrana może wystarczyć, aby zostać szczęściarzem.*

Pewnego razu ten zaszczytny tytuł wywalczył sobie straszny pechowiec Bajtazar. Był on tym tak zaskoczony, że całkowicie zapomniał, ile partii wygrał. Zastanawia się teraz, jak wielkie było jego szczęście i czy może pech go wreszcie opuścił. Wie kto z kim i ile partii grał tego wieczora. Nie wie jednak, jakie były wyniki. Bajtazar chce wiedzieć, jaka była najmniejsza liczba wygranych partii, która mogła dać tytuł szczęściarza. Pomóż zaspokoić ciekawość Bajtazara!

Zadanie

Napisz program, który:

- *wczyta ze standardowego wejścia dla każdej rozegranej partii parę uczestniczących w niej graczy,*
- *znajdzie najmniejszą liczbę k, taką że istnieje układ wyników rozgrywek, w którym każdy gracz wygrywa co najwyżej k partii,*
- *wypisze na wyjście liczbę k i wyniki wszystkich partii w znalezionym układzie.*

Wejście

W pierwszym wierszu wejścia znajduje się para liczb całkowitych n i m oddzielonych pojedynczym odstępem, $1 \leqslant n \leqslant 10\,000$, $0 \leqslant m \leqslant 10\,000$; n oznacza liczbę graczy, a m liczbę rozgrywek. Gracze są ponumerowani od 1 do n. W kolejnych m wierszach znajdują się pary numerów graczy, oddzielone pojedynczym odstępem, opisujące ciąg rozgrywek. Ta sama para może pojawiać się wiele razy w podanym ciągu.

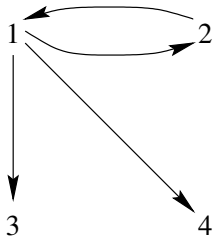
Wyjście

Pierwszy wiersz wyjścia powinien zawierać znalezioną liczbę k. Dla każdej pary numerów graczy a, b podanej w i-tym wierszu wejścia, w i-tym wierszu wyjścia powinna pojawić się liczba 1 gdy gracz o numerze a wygrywa z graczem o numerze b w znalezionym układzie wyników rozgrywek, lub 0 w przeciwnym przypadku.

Przykład

Dla danych wejściowych:

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



poprawnym wynikiem jest:

```
1
0
0
0
1
```

Rozwiązanie

Z bajtockiego na nasze

Jak to zwykle bywa w przypadku zadań olimpijskich, dobrze jest zacząć rozwiązanie zadania od przetłumaczenia jego treści z bajtockiej anegdoty na problem abstrakcyjny, wyrażony za pomocą prostych, precyzyjnie zdefiniowanych pojęć.

Już na pierwszy rzut oka widać, że graczy oraz rozgrywki można przedstawić w postaci swego rodzaju grafu, w którym wierzchołki oznaczają graczy, a krawędzie reprezentują rozgrywki. W odróżnieniu od zwykłego grafu, mamy tu jednak do czynienia z sytuacją, gdy dwa wierzchołki mogą być połączone wieloma krawędziami (bo ci sami gracze mogli grać ze sobą wiele razy). Taki graf nazywamy *multigrafem*.

W zadaniu musimy określić zwycięzców poszczególnych rozgrywek. Jak zapisać w multigrafie, że w pewnej rozgrywce gracz *u* wygrał z graczem *v*? Dość naturalnym jest przyjęcie umowy, że krawędź odpowiadająca rozgrywce, w której wygrał gracz *u*, będzie skierowana w kierunku gracza *v* (równie naturalne jest przyjęcie umowy dokładnie przeciwnej).

Definicja 1 Multigraf, w którym wszystkie krawędzie mają określony kierunek nazywamy *multigrafem zorientowanym* lub inaczej *orientacją* multigrafu. Liczbę krawędzi wychodzących z wierzchołka *u* będziemy nazywać *stopniem wyjściowym u* i oznaczać przez $\deg^+(u)$. Największy ze stopni wyjściowych wierzchołków multigrafu zorientowanego nazwiemy *stopniem wyjściowym multigrafu*. Orientację o stopniu wyjściowym nieprzekraczającym *d* będziemy nazywać *d-orientacją*.

Teraz już możemy wyrazić problem z zadania w języku grafów: *dla zadanego multigrafu poszukujemy orientacji, której stopień wyjściowy jest najmniejszy z możliwych*.

Rozwiązanie wzorcowe

Pomysł, na którym opiera się rozwiązanie wzorcowe jest bardzo prosty.

Algorytm 1 (poszukiwanie optymalnej orientacji)

1. Rozpoczynamy od znalezienia dowolnej orientacji (np. każdej krawędzi multigrafu nadajemy kierunek losowo). Taka orientacja oczywiście nie musi być optymalna. Nic to, w końcu nie od razu Kraków zbudowano.

2. Przystępujemy do *poprawiania* aktualnie posiadanej orientacji.

   (a) Znajdujemy wierzchołek v o największym stopniu wyjściowym — nazwiemy go *wierzchołkiem maksymalnym*, a jego stopień oznaczymy przez D.

   (b) Szukamy w multigrafie takiego wierzchołka w, do którego prowadzi ścieżka z wierzchołka v (możemy się na niej poruszać tylko zgodnie z kierunkiem krawędzi) i który ma stopień wyjściowy nie większy niż $D-2$. Do poszukiwania możemy użyć algorytmu przeszukiwania w głąb (patrz np. [17]).

   (c) Jeśli taki wierzchołek w nie istnieje, to kończymy algorytm. W przeciwnym razie przechodzimy do kolejnego punktu

   (d) Odwracamy kierunki wszystkich krawędzi na znalezionej ścieżce z v do w. Zauważmy, że w ten sposób:

   - zmniejszyliśmy stopień wyjściowy wierzchołka v o 1 (jest teraz równy $D-1$),
   - zwiększyliśmy stopień wyjściowy wierzchołka w o 1 (może być teraz równy najwyżej $D-1$),
   - nie zmieniliśmy stopni wyjściowych pozostałych wierzchołków.

   A więc jesteśmy odrobinę bliżej celu: albo udało nam się zmniejszyć stopień wyjściowy multigrafu, albo chociaż zmniejszyliśmy liczbę wierzchołków o stopniu wyjściowym D.

   (e) Powracamy do punktu 2(a).

Widzimy, że algorytm kończy się, gdy wszystkie wierzchołki, do których można dojść z wierzchołka maksymalnego v mają stopnie wyjściowe równe $\deg^+(v)$ lub $\deg^+(v)-1$. Otrzymaną wówczas orientację multigrafu uważamy za optymalną.

Poprawność rozwiązania wzorcowego

Jeśli nie jesteście przekonani, że orientacja otrzymana po zakończeniu algorytmu jest orientacją o najmniejszym możliwym stopniu wyjściowym, to macie rację — to nie jest oczywiste. Widzimy co prawda, że orientacji tej nie można już poprawić naszą metodą, ale może istnieje lepsze rozwiązanie, które znajduje się zupełnie inaczej? Okazuje się, że jednak nie istnieje, a wynika to z poniższego lematu.

Lemat 1 *Niech H będzie orientacją multigrafu G oraz niech v będzie wierzchołkiem H o największym stopniu wyjściowym. Jeśli wszystkie wierzchołki, do których można dojść z v w multigrafie zorientowanym H mają stopnie wyjściowe większe od* $\deg^+(v)-2$*, to stopień wyjściowy każdej orientacji multigrafu G jest równy co najmniej* $\deg^+(v)$.

Dowód Niech W będzie zbiorem tych wierzchołków, do których można dojść w multigrafie H z wierzchołka v. Załóżmy, że wszystkie wierzchołki W mają stopnie wyjściowe większe od $\deg^+(v)-2$.

Zauważmy, że każda krawędź grafu H wychodząca z wierzchołka należącego do zbioru W wchodzi do wierzchołka, który także należy do zbioru W (tzn. jeśli $u \in W$ oraz (u,w) jest krawędzią w H, to $w \in W$). Dodatkowo z każdego wierzchołka ze zbioru W wychodzi co najmniej $\deg^+(v)-1$ krawędzi, a z wierzchołka v (również należącego do zbioru W) wychodzi $\deg^+(v)$ krawędzi. Stąd liczba krawędzi łączących wierzchołki ze zbioru W jest równa co najmniej $(\deg^+(v)-1)\cdot|W|+1$.

Przypuśćmy, że dla multigrafu G istnieje orientacja J, której stopień wyjściowy nie przekracza $\deg^+(v)-1$. Policzmy ponownie krawędzie przebiegające pomiędzy wierzchołkami zbioru W. Tym razem otrzymujemy, że jest ich najwyżej $(\deg^+(v)-1)\cdot|W|$, gdyż z każdego spośród $|W|$ wierzchołków wychodzi najwyżej $(\deg^+(v)-1)$ krawędzi (na dodatek w orientacji J nie wszystkie krawędzie wychodzące z wierzchołków zbioru W muszą prowadzić do W, ale to tylko oznacza, że nasze szacowanie może być zawyżone). W ten sposób doszliśmy do wniosku sprzecznego z otrzymanym wcześniej ograniczeniem, stąd założenie o stopniu wyjściowym orientacji J było błędne i musi on wynosić co najmniej $\deg^+(v)$. ■

Efektywność rozwiązania wzorcowego

Jak dotąd przekonaliśmy się, że jeśli nasz algorytm zakończy się, to zwróci poprawny wynik. Teraz czas na uzasadnienie, że algorytm rzeczywiście zakończy się i to w miarę szybko. Niech $H_0, H_1, \ldots$ będą kolejnymi orientacjami grafu G wygenerowanymi po kolejnych poprawach i niech $D_0, D_1, \ldots$ oznaczają stopnie wyjściowe tych orientacji. Przez $\deg_i^+(w)$ oznaczymy stopień wyjściowy wierzchołka w w orientacji H_i. Zauważmy, że prawdziwe jest następujące spostrzeżenie:

Lemat 2 *Dla dowolnego wierzchołka w multigrafu G, jeśli w jest maksymalny w pewnej orientacji H_j, to dla każdej orientacji H_i, gdzie $j<i$, zachodzi* $\deg_i^+(w) \geqslant D_i - 1$.

Dowód Udowodnimy przez indukcję względem i, że wszystkie wierzchołki, które były maksymalne w orientacji H_j dla $j<i$, w orientacji H_i mają stopień wyjściowy równy D_i-1 lub D_i. Oczywiście dla $i=0$ lemat jest spełniony, bo nie istnieją wierzchołki, które byłyby maksymalne we wcześniejszych fazach.

Rozważmy teraz $i>0$ oraz wierzchołek w, który był kiedyś maksymalny, i niech $j<i$ będzie maksymalnym indeksem orientacji, w której w był maksymalny.

Przypadek 1. Jeżeli $j=i-1$, to w poprzedniej fazie algorytmu w był wierzchołkiem maksymalnym i albo poprawiając orientację H_j obniżyliśmy jego stopień wyjściowy o 1 (przy czym stopień wyjściowy orientacji mógł zmaleć lub pozostać niezmieniony), albo obniżyliśmy stopień wyjściowy innego wierzchołka, a $\deg_{i-1}^+(w) = \deg_i^+(w)$ (podobnie $D_{i-1}=D_i$). W obu przypadkach zachodzi $deg_i^+(w) \geqslant D_i - 1$.

Przypadek 2. Jeżeli $j<i-1$, to z założenia indukcyjnego i sposobu, w jaki zdefiniowaliśmy j, mamy $\deg_{i-1}^+(w) = D_{i-1}-1$. Stąd w czasie poprawiania orientacji H_{i-1} stopień wyjściowy wierzchołka w nie uległ zmianie i $\deg_i^+(w) = D_i$ (gdy obniżyliśmy stopień wyjściowy orientacji przechodząc z H_{i-1} do H_i) lub $deg_i^+(w) = D_i - 1$ (gdy poprawa nie zmniejszyła stopnia wyjściowego orientacji). ■

Z lematu wynika, że jeśli wierzchołek w stał się maksymalny w pewnej orientacji H_j, to od tego czasu *jego stopień wyjściowy nigdy nie rośnie*. W procesie poprawiania orientacji H_i dla $i \geqslant j$ wzrasta bowiem tylko stopień wierzchołka o stopniu równym najwyżej $D_i - 2$. A *ile razy może maleć* stopień wyjściowy w? Na pewno mniej niż wynosi jego stopień w multigrafie G. Tymczasem podczas każdej poprawy orientacji maleje stopień wyjściowy jednego wierzchołka maksymalnego. Stąd początkową orientację możemy poprawiać najwyżej $\sum_v \deg(v)$ razy.

Powyższą sumę możemy łatwo policzyć. Wystarczy zauważyć, że każdą krawędź multigrafu G liczymy w niej dwa razy[1], a więc $\sum_v \deg(v) = 2m$, jeśli przez m oznaczymy liczbę krawędzi multigrafu G. Stąd wiemy, że algorytm wykonuje $O(m)$ razy poprawę orientacji.

Pozostaje już tylko zastanowić się, ile czasu wymaga jedna poprawa. Algorytm wyszukiwania ścieżki (przeszukiwanie w głąb) zajmuje czas $O(m+n)$, gdzie n jest liczbą wierzchołków grafu. W podobnym czasie możemy znaleźć wierzchołek maksymalny obliczając stopnie wyjściowe wszystkich wierzchołków przy zadanej orientacji.

Choć nie wpłynie to znacząco na sumaryczną złożoność algorytmu, możemy wierzchołki maksymalne znajdować efektywniej. W tym celu liczymy raz, na początku, stopnie wyjściowe wszystkich wierzchołków i tworzymy tablicę d, w której na pozycji i mamy listę wierzchołków stopnia i. Dzięki temu wyszukiwanie wierzchołka największego stopnia zajmuje czas stały — po prostu wybieramy pierwszy element z odpowiedniej listy. Musimy tylko pamiętać, aby po każdej poprawie uaktualniać wartość największego stopnia (gdy lista wierzchołków dotychczas maksymalnych stanie się pusta) oraz żeby przenieść do odpowiednich list wierzchołki, których stopnie wyjściowe uległy zmianie.

Podsumowując, cały algorytm działa w czasie $O(m \cdot (m+n))$, czyli kwadratowo zależnym od liczby krawędzi multigrafu.

Rozwiązanie z przyszłością

Problem można rozwiązać również w inny sposób — dostrzegając w nim specjalny przypadek zagadnienia *maksymalnego przepływu w sieciach*. Problem przepływu w sieciach ma bardzo naturalne sformułowanie i znajduje szereg różnorodnych zastosowań.

Dane dla problemu to graf skierowany $G = (V,E)$, w którym z każdą krawędzią (u,v) związana jest liczba $c(u,v)$ nazywana *przepustowością krawędzi*. Przyjmujemy, że $c(u,v) = 0$, gdy $(u,v) \notin E$. Dwa wierzchołki grafu są wyróżnione: pierwszy oznaczamy przez s i nazywamy *źródłem*, a drugi przez t i nazywamy *ujściem*.

Definicja 2 *Przepływem* w grafie G nazywamy funkcję $f : V \times V \to \mathbf{R}$ spełniającą następujące warunki:

- dla wszystkich $u, v \in V$ zachodzi $f(u,v) \leqslant c(u,v)$,
- dla wszystkich $u, v \in V$ zachodzi $f(u,v) = -f(v,u)$,
- dla wszystkich $u \in V \setminus \{s,t\}$ zachodzi $\sum_{v \in V} f(u,v) = 0$.

Wartością przepływu nazywamy liczbę $|f| = \sum_{v \in V} f(s,v)$.

[1] Fakt ten nosi nazwę *lematu o uściskach dłoni*, patrz [26]

Nieformalnie, możemy myśleć o naszym grafie jak o systemie kanałów, w których płynie woda. Pierwszy warunek mówi, że przez krawędź nie może przepłynąć więcej wody niż wynosi jej przepustowość. Warunki drugi i trzeci mówią, że do każdego wierzchołka (oprócz s i t) wpływa tyle samo wody co z niego wypływa. Innymi słowy, woda wypływa ze źródła i spływa do ujścia, a w pozostałych miejscach nie może jej ani przybyć, ani ubyć. Zgodnie z naszą interpretacją wartość przepływu to ilość wody płynącej w sieci kanałów (czyli wypływającej ze źródła).

Zadanie polega na znalezieniu przepływu o maksymalnej wartości. Szczegółowy opis problemu można znaleźć w książce [17]. Znajduje się tam również szereg bardzo interesujących algorytmów dla tego zagadnienia. Najprostszy z nich, algorytm Forda-Fulkersona, działa w czasie $O(|E| \cdot |f^*|)$, gdzie f^* jest maksymalnym przepływem, przy założeniu, że wszystkie przepustowości krawędzi są liczbami całkowitymi. Dodatkowo, dla grafu o całkowitych przepustowościach algorytm znajduje przepływ całkowitoliczbowy, w którym dla dowolnych wierzchołków u, v liczba $f(u,v)$ jest całkowita.

Pokażemy teraz jak można *zredukować* problem znajdowania d-orientacji multigrafu do znajdowania maksymalnego przepływu w grafie. Redukcja problemu A do problemu B polega na takim przedstawieniu danych dla problemu A, by można było do niego zastosować algorytm rozwiązywania problemu B i z otrzymanego wyniku prosto i szybko uzyskać wynik dla problemu A. Nasza redukcja będzie więc polegać na przedstawieniu multigrafu w postaci sieci w ten sposób, by ze znalezionego optymalnego przepływu w tej sieci można było odtworzyć d-orientację multigrafu.

Rozważmy multigraf $M = (V_M, E_M)$ i utwórzmy sieć $G_d = (V, E)$. Zdefiniujmy

$$V = V_M \cup \{s,t\} \cup \{v_{xy} : E_M \text{ zawiera choć jedną krawędź } xy\},$$

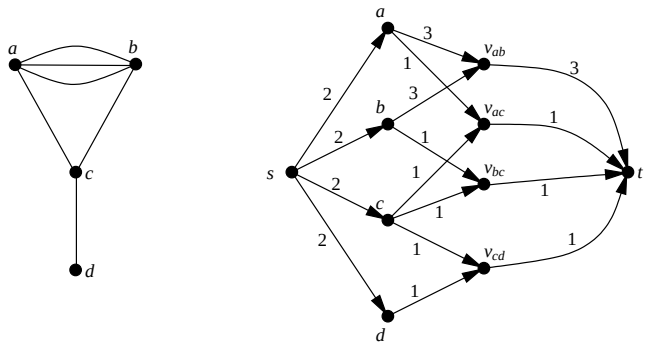
czyli V zawiera wierzchołki zbioru V_M, dwa nowe wierzchołki s (źródło) i t (ujście) oraz po jednym wierzchołku dla każdej pary wierzchołków x, y połączonych w M choć jedną krawędzią. Teraz czas na opisanie zbioru krawędzi E. W zbiorze tym umieszczamy krawędź (s,x) o przepustowości d dla każdego wierzchołka $x \in V_M$. Następnie dorzucamy krawędzie: (x, v_{xy}), (y, v_{xy}), oraz (v_{xy}, t) dla każdej pary wierzchołków x, y, połączonych choć jedną krawędzią w M — wszystkim tym krawędziom przypisujemy przepustowość równą liczbie krawędzi łączących wierzchołki x i y w multigrafie M.

Na rysunku jest przedstawiony przykład multigrafu i sieci utworzonej na jego podstawie (dla $d = 2$).

Zauważmy, że prawdziwy jest następujący lemat.

Lemat 3 *Multigraf M ma d-orientację wtedy i tylko wtedy, gdy maksymalny przepływ w sieci G_d wynosi $|E_M|$.*

Dowód Załóżmy najpierw, że multigraf M ma d-orientację H. Zauważmy, że maksymalny przepływ w sieci G_d nie może przekraczać $|E_M|$, gdyż taka jest suma przepustowości krawędzi wchodzących do ujścia t. Teraz pokażemy, jak można skonstruować przepływ w G_d o wartości $|E_M|$. Zaczynamy od przepływu zerowego f (przypisujemy $f(u,v) = 0$ wszystkim krawędziom G_d). Potem dla każdej krawędzi (x,y) skierowanej w orientacji H od x do y powiększamy o 1 wartości przepływu na krawędziach: (s,x), (x, v_{xy}) oraz (v_{xy}, t). Zauważmy, że taka modyfikacja nie narusza warunków poprawności przepływu z definicji 2 (w szczególności nie przekraczamy przepustowości krawędzi (s,x)) i jednocześnie powoduje



Multigraf (z lewej) i sieć przepływowa służąca do wyznaczania jego 2-orientacji (z prawej — liczby oznaczają przepustowości krawędzi).

wzrost wartości przepływu o 1. Po przeprowadzeniu jej dla wszystkich krawędzi multigrafu M otrzymujemy przepływ o wartości $|E_M|$.

Teraz przyjmijmy, że w grafie G_d istnieje przepływ całkowitoliczbowy f o wartości $|E_M|$. Na jego podstawie skonstruujemy d-orientację multigrafu M. W tym celu rozważmy dowolną parę wierzchołków x, y, które są połączone w M, i oznaczmy przez k liczbę krawędzi łączących te wierzchołki. Skoro przepływ ma wartość $|E_M|$, to oczywiście wszystkie krawędzie wchodzące do ujścia t muszą być „pełne" i stąd $f(v_{xy}, t) = k$. W takim razie z drugiego i trzeciego warunku z definicji 2 zastosowanych dla wierzchołka v_{xy} wynika, że $f(x, v_{xy}) + f(y, v_{xy}) = k$. Wybierzmy dowolne $f(x, v_{xy})$ spośród k krawędzi łączących x i y i zorientujmy je jako wychodzące z x, a pozostałe $f(y, v_{xy})$ krawędzi jako wychodzące z y. W ten sposób określimy kierunek wszystkich krawędzi M, a stopień wyjściowy żadnego wierzchołka nie będzie przekraczał d, co wynika stąd, iż do każdego wierzchołka $x \in V_M$ wpływa najwyżej d jednostek z wierzchołka s (definiując G_d określiliśmy $f(s, v) = d$), a więc najwyżej tyle jednostek może wypływać i $f(x, v_{xy}) \leqslant d$. ■

Z lematu 3 dowiadujemy się, jak dla pewnej liczby d sprawdzić, czy multigraf M ma d-orientację, a jeśli ją ma, to jak ją znaleźć.

Algorytm 2 (znajdowanie d-orientacji)

1. Budujemy graf G_d.
2. Za pomocą algorytm Forda-Fulkersona znajdujemy maksymalny przepływ f w sieci G_d.
3. Jeśli $|f| = |E_M|$, to w sposób podany w dowodzie lematu konstruujemy z przepływu f d-orientację. W przeciwnym razie stwierdzamy, że multigraf M nie ma d-orientacji.

Całość działa w czasie $O(|E| \cdot |E_M|) = O((|E_M| + |V_M|) \cdot |E_M|)$. Pozostaje jeszcze odszukanie optymalnej (minimalnej) wartości d, dla której istnieje d-orientacja multigrafu M. Zauważmy, że liczba d musi należeć do przedziału $[1, |E_M|]$, więc zastosujemy wyszukiwanie binarne w tym przedziale.

Algorytm 3 (znajdowanie optymalnej orientacji)

1. Definiujemy przedział poszukiwań $[l, p] = [1, |E_M|]$.
2. Podstawiamy $d = \lfloor (l+p)/2 \rfloor$ i uruchamiamy Algorytm 2.
3. Jeśli algorytm 2 zwraca d-orientację multigrafu, to za przedział poszukiwań podstawiamy $[l, p] = [l, \lfloor (l+p)/2 \rfloor]$. W przeciwnym razie musimy szukać w przedziale $[l, p] = [\lfloor (l+p)/2 \rfloor + 1, p]$.
4. Jeśli poszukiwanie zostało zawężone do przedziału zawierającego jedną wartość, to kończymy algorytm. W przeciwnym razie wracamy do punktu 2.

Widzimy, że wystarczy uruchomić algorytm 2 najwyżej $\log_2(|E_M|)$ razy, by znaleźć optymalną wartość d. To daje nam końcowy algorytm o złożoności $O((|E_M| + |V_M|) \cdot |E_M| \cdot \log |E_M|)$.
Zadanie (ciekawe i nietrudne): Spróbuj, Drogi Czytelniku, sformułować algorytm znajdowania optymalnej d-orientacji, który wywoła algorytm Forda-Fulkersona jedynie $O(\log d^*)$ razy, gdzie d^* jest poszukiwanym maksymalnym stopniem wyjściowym orientacji.

Stosując metodę przepływów uzyskaliśmy w efekcie algorytm o nieco gorszej złożoności niż rozwiązanie wzorcowe. Jest to jednak podejście z przyszłością. Jeśli bowiem zamiast algorytmu Forda-Fulkersona zastosujemy bardziej zaawansowany algorytm[2], to można pokazać, że operacja znajdowania maksymalnego przepływu zajmie w tym szczególnym przypadku jedynie czas $O(|E_M|^{3/2})$.

Testy

Wygenerowano 26 testów, wszystkie z użyciem generatora liczb pseudolosowych. W każdym teście kolejność partii i graczy została pseudolosowo wymieszana. Większość testów została zgrupowana po dwa testy. Multigrafy zdefiniowane w testach można podzielić na cztery kategorie:

(1) graf dwudzielny, niesymetryczny: jedna część (wielkości rzędu $\sqrt{n}$) mała i gęsta, druga część duża i rzadka,

(2) jedna długa ścieżka zakończona gęstym kłębkiem,

(3) wielokrotnie powtórzony układ trzech wierzchołków, z których jeden jest połączony potrójnymi krawędziami z dwoma pozostałymi,

(4) graf jednorodnie losowy.

W poniższej tabeli liczby n i m oznaczają odpowiednio liczbę zawodników i rozgrywek (wierzchołków — V_M i krawędzi V_M multigrafu), natomiast d oznacza minimalną liczbę wygranych gwarantującą tytuł szczęściarza.

[2] Mowa tu o algorytmie Dinica, opisanym w książce [20]

Nazwa	n	m	d	Opis
kos1a.in	5	8	2	rodzaj (1)
kos1b.in	90	180	2	rodzaj (3)
kos2a.in	100	100	2	rodzaj (1)
kos2b.in	300	600	2	rodzaj (3)
kos3a.in	100	1 000	12	rodzaj (1)
kos3b.in	498	996	2	rodzaj (3)
kos4a.in	1 000	1 000	3	rodzaj (1)
kos4b.in	699	1 398	2	rodzaj (3)
kos5a.in	1 000	10 000	12	rodzaj (1)
kos5b.in	999	1 998	2	rodzaj (3)
kos6a.in	10 000	10 000	3	rodzaj (1)
kos6b.in	1 500	3 000	2	rodzaj (3)
kos7a.in	10	30	5	rodzaj (2)
kos7b.in	100	300	4	rodzaj (4)
kos8a.in	100	200	11	rodzaj (2)
kos8b.in	100	1 000	10	rodzaj (4)
kos9a.in	100	1 000	91	rodzaj (2)
kos9b.in	50	10 000	200	rodzaj (4)
kos10a.in	1 000	1 000	2	rodzaj (2)
kos10b.in	100	3 000	30	rodzaj (4)
kos11a.in	1 000	10 000	91	rodzaj (2)
kos11b.in	1 000	8 000	9	rodzaj (4)
kos12a.in	10 000	10 000	2	rodzaj (2)
kos12b.in	4 000	10 000	3	rodzaj (4)
kos13.in	10 000	9 999	1	duże grafy rzadkie
kos14.in	10 000	10 000	1	duże grafy rzadkie

Zawody III stopnia

opracowania zadań