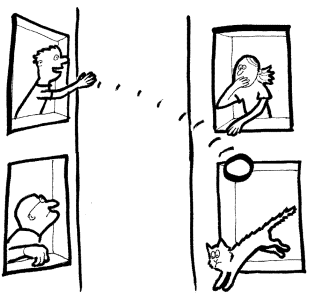
INFORMATIKA

Mirka a Jirka na šikmé ploše

VÁCLAV ZEMEK

Gymnázium Prachatice



V minulém příběhu [3] jsme sledovali Mirku a Jirku, jak nezištně pomáhají učitelkám mateřské školy šetřit školním materiálem. Nyní se ale dostali na šikmou plochu. Chtějí vyzkoušet svoji zručnost a přehodit míček mezi okny sousedních domů. Nedbají na možnost poškození cizí věci – okna pana Nováka či jiného souseda. Riziko je však malé, neb na to jdou vědecky.

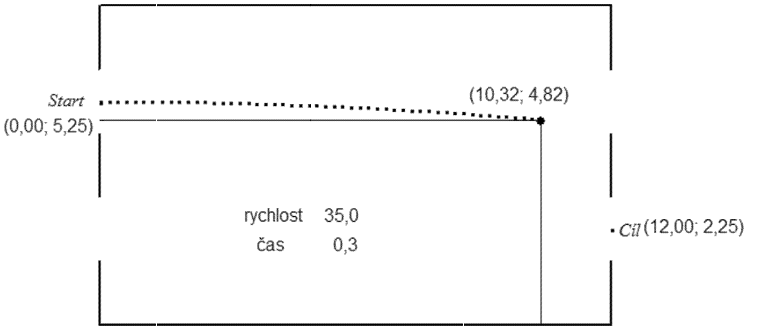
Uveďme nyní parametry problému. Okna jsou vysoká 1,5 m, svislé osy oken jsou vzdáleny 12 m. Středy oken Mirky a Jirky jsou ve výškách 2,25 m, resp. 5,25 m nad okolním terénem (vodorovnou rovinou). Jirka hází míček ze středu svého okna vodorovným směrem v rovině svislých os oken. Mirka věří, že zachytí míček, pokud bude jejím oknem procházet v libovolné výšce. Je totiž také neobyčejně pohybově nadaná.

Vědecké řešení problému vyšlo z této jejich nezapomenutelné debaty.
Mirka: Míček je malý, tuhý, pokus provedeme za bezvětří. Zkusme využít model vodorovného vrhu v homogenním gravitačním poli Země. Pohyb míčku se skládá z rovnoměrného pohybu vodorovným směrem a volného pádu. Pro rovnoměrný pohyb ve směru osy x platí rovnice $x = vt$. Volný pád ve směru osy y z výšky h vyjádříme rovnicí

$$y = h - \frac{gt^2}{2}.$$

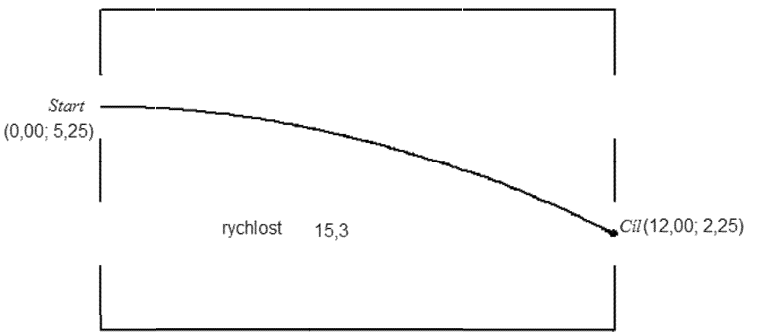
Jirka: Když využiji tyto rovnice, lze v Cabri geometrii pohyb míčku simulovat pohybem bodu o souřadnicích x, y. Změnou rychlosti měníme tra-

jektorii pohybu míčku. Vyberu nejdříve zkusmo nějakou rychlost. Pomocí nástroje stopa *sleduj* trajektorii. Jako bys pozorovala tryskáč na obloze (obr. 1).



Obr. 1

Mirka: Pozor! Míříš do okna pana Nováka. Ještě zlomek sekundy a neštěstí je hotovo. Nevíš, že bydlím v okně pod ním?
Jirka: Ještě jsem stačil míček zastavit. Reálně by to šlo hůře. Budu měnit rychlost tak, až trajektorie bude procházet místem, kde je možné míček chytit. K tomu využiji nástroj množina. Do tvého okna se strefuji pro rychlosti v intervalu od 13,7 m/s do 17,7 m/s. Pokud je rychlost nižší, dopadá míček pod okno, při větší rychlosti nad něj. Přímo do středu okna, kde je chytání pro tebe nejpohodlnější, míří míček při rychlosti 15,3 m/s. Nyní ještě dostat tu správnou rychlost "do ruky" (obr. 2).



Obr. 2

Mirka: Zkusím raději ověřit výsledky získané v Cabri geometrii. Rychlost vypočítáme ze soustavy rovnic pro pohyb míčku:

$$y = h - \frac{gt^2}{2} \Rightarrow t = \sqrt{\frac{2h-2y}{g}},\ x = vt \Rightarrow v = x\sqrt{\frac{g}{2h-2y}}.$$

Do vztahu pro rychlost dosadíme $h = 5{,}5$, $g = 9{,}8$ a souřadnice některých míst, kterými má trajektorie procházet: $x = 12$, $y_0 = 2{,}25$, $y_1 = 1{,}5$, $y_2 = 3$. Ideální počáteční rychlost, při které míček dopadá do středu okna, je

$$v_0 = 12 \cdot \sqrt{\frac{9{,}8}{2 \cdot 5{,}25 - 2 \cdot 2{,}25}} = \frac{14\sqrt{30}}{5} \approx 15{,}3 \text{ m/s}.$$

Na spodní rám okna míří míček při rychlosti

$$v_1 = 12 \cdot \sqrt{\frac{9{,}8}{2 \cdot 5{,}25 - 2 \cdot 1{,}5}} = \frac{28\sqrt{6}}{5} \approx 13{,}7 \text{ m/s}.$$

Trochu povyskočit si budu muset při rychlosti

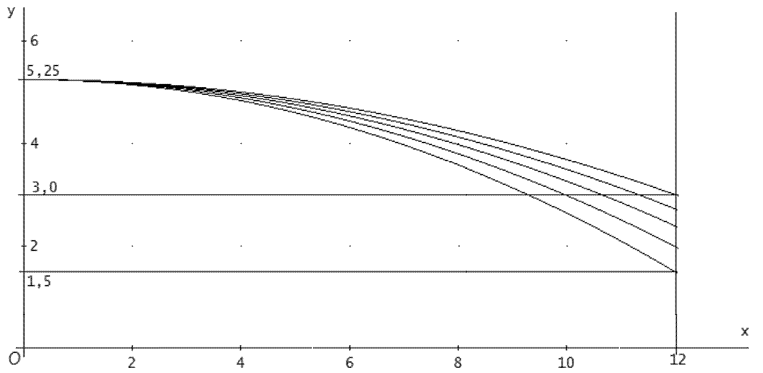
$$v_2 = 12 \cdot \sqrt{\frac{9{,}8}{2 \cdot 5{,}25 - 2 \cdot 3}} = \frac{28\sqrt{10}}{5} \approx 17{,}7 \text{ m/s}.$$

Jirka: Trajektorie pohybu míčku mi připomíná část paraboly.
Mirka: Ano, rovnice, které jsme využívali, jsou vlastně parametrickým vyjádřením paraboly. Parametrem je čas t. Jestliže čas z rovnic vyloučíme dosazením $t = \frac{x}{v}$ do druhé rovnice, dostaneme rovnici $y = -\frac{g}{2v^2}x^2 + h$. Je jasné, že jde o kvadratickou funkci, jejímž grafem je parabola. V této úloze ovšem uvažujeme jen její část pro $x \in \langle 0; 12 \rangle$.
Jirka: Díky za vysvětlení. Ještě to ověřím v programu Derive. Zobrazím čtyři paraboly s vhodnými počátečními rychlostmi. Využiji příkaz na obr. 3.

$$\text{VECTOR}\left(5.25 - \frac{9.8 \cdot x^2}{2 \cdot v^2},\ v,\ 13.7,\ 17.7,\ 1\right)$$

Obr. 3

Parametr se tímto příkazem ve výrazu $5,25 - \frac{9,8}{2v^2}x^2$ mění od 13,7 do 17,7 s krokem 1.



Obr. 4

Myslím, že nelze dále pochybovat. Míček musím hodit rychlostí mezi 13,7 m/s a 17,7 m/s a dál už vše záleží na tvé obratnosti.

Výsledek reálného pokusu není znám. Zkuste se nenápadně zeptat pana Nováka.

(Autorkou úvodní ilustrace je *Mgr. Jaroslava Čermáková* ze Suchých Lazců.)

Literatura

[1] *Charvát, J – Houf, J. – Boček, L.*: Matematika pro gymnázia – Rovnice a nerovnice, dotisk 3. vydání. Prometheus, Praha, 1999.

[2] *Kuřina, F.*: Umění vidět v matematice. SPN, Praha, SPN, 1989.

[3] *Zemek, V.*: Geometrická úloha řešená s přítelem počítačem. MFI, r. 17 (2007/2008), č. 9.

Barevné sloupy (Úlohy z MO – kategorie P, 23. část)

PAVEL TÖPFER

Matematicko-fyzikální fakulta UK, Praha

V našem seriálu zajímavých úloh z Matematické olympiády – kategorie P se tentokrát zastavíme u jedné z teoretických úloh ústředního kola loňského 57. ročníku MO (školní rok 2007/2008). Námět úlohy navrhli organizátoři olympiády z Fakulty matematiky, fyziky a informatiky Univerzity Komenského v Bratislavě, formulaci jsme připravili na naší Matematicko-fyzikální fakultě Univerzity Karlovy v Praze. Autorem původního řešení a vzorového programu, z něhož vychází i tento článek, je *Petr Škoda* z MFF UK Praha. Úloha pojednává o minimalizaci počtu barev, jimiž se podle stanovených pravidel mohou obarvit sloupy v chrámu. Jak uvidíme, jedná se vlastně o grafovou úlohu, v níž je ale velmi důležité umět při řešení vhodný graf správně zvolit. Začneme jako obvykle zadáním soutěžní úlohy, které jsme zde oproti původnímu pouze trochu formulačně upravili.

⋆⋆⋆

Oslavy 75. narozenin královny Šeherezády se blíží a v paláci se to jen hemží. Vyměňují se tapisérie, pokládají se nové koberce a přemalovávají se obrazy. Radní se rozhodli, že královnu překvapí novou duhovou výzdobou chrámu. Chtějí nechat natřít sloupy v chrámu různými barvami, aby z toho jen oči přecházely. Pro dosažení co nejlepšího výsledného dojmu je třeba, aby v žádné řadě sloupů neměly dva sloupy stejnou barvu. Královská pokladna však není bezedná, a proto je třeba nakoupit co nejméně druhů barev.

Soutěžní úloha

Chrám si můžeme představit jako šachovnici o velikosti $N \times N$. Na čtvercových polích šachovnice stojí celkem M sloupů, tzn. na některých polích sloup stojí a na některých ne. Každý sloup stojí na jiném poli šachovnice. Vaším úkolem je obarvit tyto sloupy co nejmenším počtem K barev tak, aby žádné dva sloupy stojící ve stejném řádku nebo ve stejném sloupci šachovnice neměly tutéž barvu.

Formát vstupu

Na prvním řádku vstupu program dostane dvě přirozená čísla N a M, představující velikost šachovnice $1 \leqq N \leqq 500$ a počet sloupů v chrámu $0 \leqq M \leqq N^2$. Následuje M řádků popisujících polohu jednotlivých sloupů. Na i-tém z nich najdete dvě celá čísla R_i a S_i, řádek $1 \leqq R_i \leqq N$ a sloupec $1 \leqq S_i \leqq N$ toho pole šachovnice, na kterém stojí i-tý sloup.

Formát výstupu

První řádek výstupu bude obsahovat minimální počet barev K potřebný na obarvení sloupů v chrámu. Na následujících M řádků vypište barvy jednotlivých sloupů. Na i-tý z těchto řádků napište číslo barvy B_i, $1 \leqq B_i \leqq K$, kterou má být obarven i-tý sloup. Zřejmě bude existovat více optimálních obarvení (barvy lze například mezi sebou libovolně permutovat), vypište proto jedno libovolné z nich.

⋆⋆⋆

Abychom si zadání úlohy lépe objasnili, začneme dvěma ilustrujícími příklady. V prvním z nich jsou v chrámu pouze čtyři sloupy umístěné ve vrcholech čtverce.

Vstup

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

Na obarvení sloupů stačí dvě barvy, vždy protilehlé sloupy dostanou stejnou barvu.

Výstup

```
2
1
2
1
2
```

Druhý příklad je trochu složitější, v chrámu je rozmístěno celkem 8 sloupů a k jejich obarvení jsou zapotřebí tři barvy, jak ukazuje následu-

jící příklad vstupních dat a jim odpovídající možný výstup (existuje i jiné řešení).

Vstup

```
4 8
1 1
1 3
1 4
2 1
3 3
4 1
4 2
4 3
```

Výstup

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

Mnoho řešitelů správně odhadne, že se jedná o grafovou úlohu. Obvykle vytvoří graf o M vrcholech, v němž každý vrchol představuje jeden sloup v chrámu. Hrana v tomto grafu spojuje každou dvojici vrcholů, které odpovídají sloupům umístěným ve stejném řádku nebo ve stejném sloupci šachovnice. V takovém grafu potom potřebujeme obarvit všechny jeho vrcholy minimálním počtem barev tak, aby žádné dva vrcholy spojené hranou nebyly obarveny stejnou barvou. Takto postavená úloha na obarvení grafu je ovšem velmi obtížná, takže tudy správné řešení našeho problému nevede.

Zadanou úlohu raději převedeme na grafový problém jiným, méně obvyklým způsobem. Sestrojíme graf o $2N$ vrcholech, ve kterém vrcholy odpovídají řádkům a sloupcům šachovnice. Hrana v tomto grafu bude spojovat vrcholy reprezentující i-tý řádek a j-tý sloupec právě tehdy, když

na poli šachovnice o souřadnicích $[i, j]$ stojí sloup. V tomto grafu tedy nebudeme obarvovat vrcholy, ale hrany.

Podle uvedené definice mají dvě hrany v grafu společný vrchol právě tehdy, jestliže odpovídají sloupům umístěným ve stejném řádku nebo ve stejném sloupci šachovnice. Takové sloupy v chrámu ale musí být obarveny různými barvami, takže také hrany mající v našem grafu společný vrchol musí být obarveny různými barvami. Pro vyřešení úlohy tedy potřebujeme obarvit všechny hrany grafu minimálním počtem barev tak, aby u každého vrcholu grafu byly barvy všech hran, které z vrcholu vycházejí, navzájem různé.

Je zřejmé, že budeme potřebovat alespoň tolik barev, kolik je maximální stupeň vrcholu grafu, tzn. maximální počet hran, které vycházejí z některého vrcholu (což je jinými slovy maximální počet sloupů, které leží ve stejném řádku nebo ve stejném sloupci šachovnice). Ukážeme, že tento počet barev je zároveň postačující a je tedy naší hledanou hodnotou K. Navíc ukážeme, jak lze takové obarvení rychle nalézt.

Začneme s grafem bez hran, jednotlivé hrany do něj budeme postupně v libovolném pořadí přidávat a zároveň je budeme obarvovat K barvami. Podívejme se, co se stane při přidávání nějaké hrany e. Označme si jako C_e množinu barev přiřazených hranám, které již dříve byly přidány do grafu a mají s hranou e nějaký společný vrchol. Mohou nastat dvě možnosti. Pokud se v množině C_e nevyskytuje všech K barev, obarvíme přidávanou hranu e libovolnou ze zbývajících barev a jsme hotovi. V opačném případě je situace o něco složitější. Protože hrana e má celkově alespoň K sousedních hran, z každého z jejích koncových vrcholů musí vycházet minimálně jedna již obarvená hrana (platí totiž, že K je maximální stupeň vrcholu v našem grafu). Označme C_1 a C_2 množiny barev, jimiž jsou obarveny hrany vycházející z koncových vrcholů hrany e. Vybereme dvojici barev A a B takovou, že A je v C_1 a není v C_2, B je v C_2 a není v C_1. Takové barvy jistě existují, neboť množina C_1 neobsahuje všech K barev (obsahuje jich nejvýše $K-1$), množina C_2 rovněž neobsahuje všech K barev, a přitom sjednocení obou množin C_1 a C_2 obsahuje všech K barev (je totiž rovno množině C_e).

Nyní přebarvíme některé hrany grafu tak, abychom mohli přidávanou hranu e obarvit buď barvou A, nebo barvou B. Na chvíli budeme uvažovat pouze ty hrany grafu, které mají přiřazenu barvu A nebo barvu B. Protože z jednoho vrcholu mohou vést nejvýše dvě hrany obarvené některou z barev A nebo B, v grafu nám zbudou pouze cesty a kružnice. Navíc

v koncových vrcholech hrany e začínají některé z cest, protože u obou koncových vrcholů hrany e jedna z barev A a B chybí. Pokud se jedná o dvě různé cesty, prohodíme barvy A a B na jedné z těchto cest. Tím určitě neporušíme u žádného vrcholu grafu podmínku na různost barev. Nyní bude jedna z barev A nebo B zcela chybět mezi sousedy hrany e, takže touto barvou můžeme hranu e obarvit.

Uvažme nyní, zda se může stát, že budou koncové vrcholy hrany e spojeny v grafu cestou tvořenou hranami obarvenými pouze barvami A a B. Předpokládejme, že tomu tak je. Připomeňme si, že vrcholy našeho grafu odpovídají řádkům a sloupcům šachovnice a hrany vedou vždy mezi jedním z řádků a jedním ze sloupců. Na cestě spojující koncové vrcholy hrany e se tedy pravidelně střídají vrcholy odpovídající řádkům a sloupcům šachovnice a proto tato cesta obsahuje lichý počet hran. Potom ale její první a poslední hrana musí mít stejnou barvu (na cestě se pravidelně střídají hrany barvy A a hrany barvy B), což není možné.

Tím jsme ukázali, že vždy dokážeme přebarvit některé hrany grafu tak, abychom mohli korektně obarvit nově přidávanou hranu. Po přidání všech M hran do grafu tak dostaneme správné obarvení všech hran grafu nejvýše K barvami.

Na závěr zbývá odhadnout časovou složitost navrženého algoritmu. Algoritmus postupně přidává do grafu M hran a při každém přidání hrany musí přebarvit až $O(N)$ dříve obarvených hran (nikdy totiž nepřebarvujeme více než dvě hrany u jednoho vrcholu). Celková časová složitost našeho algoritmu je proto $O(N \cdot M)$.

```
program BarveniSloupu;

const MaxN = 50;      {maximální rozměr šachovnice}

var N: integer;       {skutečný rozměr šachovnice}
    M: integer;       {počet sloupů = počet hran grafu}
    K: integer;       {výsledný počet barev}
    Hrana: array[1..MaxN*MaxN] of             {hrany grafu}
              record
              Barva: integer;                 {barva hrany}
              Vrchol: array[1..2] of integer; {krajní
                                               vrcholy}
              end;
```

```
    Stupen: array[1..2*MaxN] of integer;   {stupně vrcholů}
       {vrcholy grafu odpovídající řádkům šachovnice mají
        čísla 1..N, vrcholy grafu odpovídající sloupcům
        mají čísla N+1..2N}
    UVrcholu: array[1..2*MaxN, 1..maxn] of integer;
       {UVrcholu[v,b] = číslo hrany, která vychází z vrcholu
        'v' a má barvu 'b'}

procedure ObarveniHrany(e, c: integer);
{obarvení hrany 'e' barvou 'c'}
begin
   Hrana[e].Barva := c;
   UVrcholu[Hrana[e].Vrchol[1], c] := e;
   UVrcholu[Hrana[e].Vrchol[2], c] := e;
end;

procedure PrebarveniCesty(v, barva1, barva2: integer);
{přebarvení cesty vedoucí z vrcholu 'v', tvořené hranami
obarvenými barvami 'barva1' a 'barva2'; hrana vedoucí přímo
z 'v' má 'barva1'}
var e, sousedni: integer;
begin
   if UVrcholu[v, barva1] > 0 then
      begin
      e := UVrcholu[v, barva1];
      {odbarvíme hranu 'e', zrušíme na ní obarvení 'barva1':}
      UVrcholu[Hrana[e].Vrchol[1], Hrana[e].Barva] := 0;
      UVrcholu[Hrana[e].Vrchol[2], Hrana[e].Barva] := 0;
      sousedni := Hrana[e].Vrchol[1];  {soused vrcholu 'v'
                                        na hraně 'e'}
      if sousedni = v then sousedni := Hrana[e].Vrchol[2];
      PrebarveniCesty(sousedni, barva2, barva1);

      ObarveniHrany(e, barva2);  {hraně 'e' přiřadíme
                                  'barva2'}
      end;
end;
```

```
var i, j, x, y: integer;       {pomocné proměnné pro řízení
                                cyklů}
    obarveno: boolean;         {příznak, že je hrana obarvena}
    barva1, barva2: integer;   {barvy na přebarvované cestě
                                v grafu}

begin
   {Načtení vstupních dat:}
   readln(N, M);
   for i:= 1 to M do
      begin
      readln(x, y);
      Hrana[i].Vrchol[1] := x;
      Hrana[i].Vrchol[2] := y+N;
      Stupen[x] := Stupen[x]+1;
      Stupen[y+N] := Stupen[y+N]+1;
      end;

   {Určení počtu barev:}
   K := 0;
   for i:= 1 to 2*N do
      if Stupen[i] > K then
         K := Stupen[i];

   {Obarvování hran:}
   for i:= 1 to M do          {postupně přidáváme hrany}
      begin
      obarveno := false;
      for j:= 1 to K do
      {zkusíme, zda lze hranu přímo obarvit barvou 'j'}
         if (UVrcholu[Hrana[i].Vrchol[1], j] = 0) and
            (UVrcholu[Hrana[i].Vrchol[2], j] = 0) then
            begin
            ObarveniHrany(i, j);
            obarveno := true;   {hrana je obarvena}
            break;              {další barvy už nezkoušíme}
            end;
```

```
      if not obarveno then   {budeme muset přebarvovat cestu
                              v grafu}
         begin
         {určení vhodných dvou barev, které jsou každá jen u
          jednoho z krajních vrcholů právě přidávané hrany:}
         for j:= 1 to K do
            if (UVrcholu[Hrana[i].Vrchol[1], j] > 0) and
               (UVrcholu[Hrana[i].Vrchol[2], j] = 0) then
               begin
               barva1 := j;
               break;
               end;
         for j:= 1 to K do
            if (UVrcholu[Hrana[i].Vrchol[1], j] = 0) and
               (UVrcholu[Hrana[i].Vrchol[2], j] > 0) then
               begin
               barva2 := j;
               break;
               end;
         {přebarvení cesty a obarvení nové hrany:}
         PrebarveniCesty(Hrana[i].Vrchol[1], barva1, barva2);
         ObarveniHrany(i, barva1);
         end
      end;

   {Vypsání výsledku:}
   writeln(K);
   for i:= 1 to M do
      writeln(Hrana[i].Barva);
end.
```