Rozhledy matematicko-fyzikální

Stanislav Trávníček

Úloha o kartách

Rozhledy matematicko-fyzikální, Vol. 86 (2011), No. 1, 17–23

Persistent URL: http://dml.cz/dmlcz/146397

Úloha o kartách

Stanislav Trávníček, PřF UP, Olomouc

Abstract. The paper describes a method of stacking a pack of cards to perform a card trick. Two problems are posed, and their mathematical principle is explained. The problems are also computer-programmed using two different algorithms.

Úvod

Myšlenkovým zdrojem tohoto článku jsou dvě nevýznamné události, které zde v úvodu stručně komentuji.

1° To jsem byl ještě žák ZŠ, když nám jeden starší chlapec ukazoval tento karetní trik: Vzal připravený balíček karet a začal je vykládat tak, že vždy jednu kartu dal dospodu balíčku a jednu vyložil na stůl a tak stále dokola. A hle, všech 32 karet bylo na stole vyloženo v pořadí E♡, K♡, D♡, ... až nakonec ..., 9♣, 8♣, 7♣. Je tu vlastně jen jedno tajemství – jak balíček připravit. (Pro větší srozumitelnost už zde nebudeme pojednávat o hracích kartách, ale o kartách označených postupně čísly 1, 2, 3, ... a v balíčku nemusí být 32 karet, ale $n \in \mathbb{N}$ karet.)

Pátral jsem tehdy, jak balíček připravit, a došel jsem k tomuto způsobu: Dal jsem do řádky 32 lístečků popsaných postupně 1, 2,..., 32, které reprezentovaly prázdný balíček. Pak jsem kartu 1 položil na lísteček 2, kartu 2 na lísteček 4,... až kartu 16 na lísteček 32 (volná místa byla rezervována na karty, které budou kladeny dospodu) a pak jsem postupoval opět od začátku „balíčku“, vynechal lísteček 1 a kartu 17 dal až na lísteček 3, vynechal místo 5 a kartu 18 dal na lísteček 7 atd. stále dokola, až na kartu 31 vyšel lísteček 17 a na 32 vyšlo poslední volné místo, a to bylo na lístečku 1. Balíček jsem pak poskládal tak, že vespod byla karta z lístečku 32 atd., až nahoře byla karta položená původně na lístečku 1. Fungovalo to. A pak jsem vše zasunul hodně dozadu do paměti.

2° O dvacet let později koncem 60. let jsem byl už tatínek dvou malých dětí a učil jsem je různá říkadla a též dětská rozpočitadla. Připomeňme si např.

Na koho to slovo padne,
ten musí jít z kola pryč.

Přitom jsem si uvědomil, že rozpočítávání dětí o to, kdo začne hru, je tajemný obřad, děti jsou napjaty, kdo půjde „z kola pryč“ a kdo nakonec zůstane, ale přitom pro daný počet dětí a dané rozpočitadlo je vlastně výsledek dán už v okamžiku zahájení rozpočítání.

V článku [1] pro RMF *Na koho to slovo padne* jsem odvodil pravidlo $\varphi(n,r)$, které k dětem s indexy $0,1,\ldots,n-1$ a k rozpočitadlu o r slabikách (např. ve výše uvedeném rozpočitadle je $r=15$), kde rozpočítání začíná u 0, určilo index dítěte, které po rozpočítání zbude. Přitom se využívají kongruence a čísla „dětí“ jsou vlastně reprezentanty zbytkových tříd – proto se počítá od 0. Toto pravidlo lze vyjádřit ve tvaru

$$\varphi(n,r) = (r + (r + (\ldots + (r + (r + (r)_2)_3)_4 \ldots)_{n-2})_{n-1})_n,$$

nebo ve tvaru vhodném pro „ruční“ počítání

$$\varphi(n,r) = (r_n + (r_{n-1} + (\ldots + (r_4 + (r_3 + r_2)_3)_4 \ldots)_{n-2})_{n-1})_n,$$

kde $r_m = r \ (mod\ m)$, tj. zbytek při dělení čísla r číslem m. Přitom index dítěte, které jde „z kola pryč“ jako k-té, je $z^{(k)}$, kde

$$\begin{aligned}
z^{(1)} &= (r-1)_n,\\
z^{(2)} &= (r + (r-1)_{n-1})_n,\\
z^{(3)} &= (r + (r + (r-1)_{n-2})_{n-1})_n,\\
&\vdots\\
z^{(k)} &= (r + (r + \cdots + (r-1)_{n-k+1} \ \ldots)_{n-1})_n, \qquad (1)
\end{aligned}$$

Ukázalo se, že tento matematický model φ odvozený pro rozpočitadla se po menší úpravě hodí i pro přípravu karetního triku uvedeného v 1°, pokud výrok „k-té dítě jde z kola pryč“ nahradíme výrokem „k-tou kartu vyložíme na stůl“ (stačí položit $r=2$).

Oba naznačené přístupy – v 1° i 2° – jsou docela vhodné i pro programové zpracování úloh o kartách, tj. jak připravit balíček karet, abychom dosáhli potřebného výsledku při jejich vyložení. Budeme se zabývat dvěma úlohami.

Úloha 1. Máme balíček n karet s čísly $1,2,3,\ldots,n$. Najděte uspořádání balíčku karet tak, aby pro zadané p a při způsobu vykládání „p karet dospodu, jedna vyložit“ byly všechny karty nakonec vyloženy v pořadí $1,2,3,\ldots,n$.

Řešení: Použijeme metodu podle 2°, tedy základem řešení bude vzorec (1) pro $k = 1, 2, \ldots, n$ pro rozpočitadlo o $r = p + 1$ slabikách. Při otázce programu „Kterou kartu vyložit?“ tedy volíme r; zvolíme-li např. $r = 6$, znamená to, že budeme vždy dávat pět karet dospodu a šestou kartu vyložíme.

Naprogramování vzorců (1) je těžce zvládnutelné jen zdánlivě, protože se v tomto případě nabízí vhodný programový prostředek – rekurze; zprostředkuje ji funkce $V(X, Y)$ s použitím kongruencí.

Uspořádání výpočtu: V cyklu `K := 1 to N` počítáme postupně příslušná $Z\,(= z^{(k)}) \in \{0, 1, 2, \ldots, n-1\}$. Pro $K = 1$ dostáváme, že jako první vykládáme kartu s indexem $Z\,(= z^{(1)})$, což znamená, že na místo, které má v balíčku index Z, vložíme 1: `Balicek[Z] := K` $(= 1)$. Podobně se postupuje v daném cyklu i pro další $K = 2, 3, \ldots, N$. Všechno ostatní je už rutinní postup.

Po umístění karty N vystoupí na obrazovku výsledné uspořádání karet v balíčku. Na prvním místě je číslo karty, která má být v balíčku první shora, pak číslo karty, která je v balíčku druhá shora, atd., na posledním místě je dolní karta balíčku.

```
program Karty1;
uses Crt;
var
  N, R, Z, K: Integer;
  {N pocet karet; R pocet slabik; Z místo v balicku;
  K promenna cyklu}
  Balicek: array [1..32] of Integer;

  function V(X,Y: Integer): Integer;
  begin
     if X = 1 then V := (R-1) mod (N-Y)
     else V := (R + V(X-1, Y+1)) mod (N-Y);
  end;

begin Karty1
  ClrScr;
  Write('Pocet karet: ');
  ReadLn(N);
  Write('Kterou vylozit: ');
  ReadLn(R);
  WriteLn;
  for K := 1 to N do
```

```
  begin
     Z := V(K,0);
     Balicek[Z] := K;
  end;
  {vystup vysledku:}
  for K := 1 to N do
  begin
     WriteLn(K:3,'.  ',Balicek[K-1]:3);
     if (K mod 8) = 0 then WriteLn
  end;
  ReadLn
end  {program Karty1}
```

V další úloze si situaci trochu zkomplikujeme.

Úloha 2. Máme balíček n karet s čísly $1, 2, 3, \ldots, n$ a dále máme k-tici $p_1, p_2, \ldots, p_k$ a l-tici $q_1, q_2, \ldots, q_l$ přirozených čísel. Uvažujme posloupnost (P_m), v níž se k-tice $p_1, p_2, \ldots, p_k$ periodicky opakují, a posloupnost (Q_m), v níž se l-tice $q_1, q_2, \ldots, q_l$ rovněž periodicky opakují. Najděte uspořádání balíčku karet tak, aby pro vyložení karet způsobem: „pro $i = 1, 2, \ldots$ je P_i karet dáno postupně dospodu balíčku střídavě s tím, že Q_i karet se postupně vyloží na stůl“, byly všechny karty nakonec vyloženy v pořadí $1, 2, 3, \ldots, n$.

Příklad

Mějme $p_1, p_2, p_3 = 3, 2, 1$, $q_1, q_2 = 2, 3$. Vyložení karet podle podmínek úlohy 2 má proběhnout takto:
$P_1 = p_1 = 3$ karty dospodu balíčku, $Q_1 = q_1 = 2$ vyložit: $1, 2$;
$P_2 = p_2 = 2$ karty dospodu, $Q_2 = q_2 = 3$ vyložit: $3, 4, 5$,
$P_3 = p_3 = 1$ karta dospodu, $Q_3 = q_1 = 2$ vyložit: $6, 7$;
$P_4 = p_1 = 3$ karty dospodu, $Q_4 = q_2 = 3$ vyložit: $8, 9, 10$;
$P_5 = p_2 = 2$ karty dospodu, $Q_5 = q_1 = 2$ vyložit: $11, 12$;
$P_6 = p_3 = 1$ karta dospodu, $Q_6 = q_2 = 3$ vyložit: $13, 14, 15$; atd.

Řešení: Počítač můžeme pověřit pilnou prací, a proto je myšlenkově jednodušší balíček karet přímo naskládat (podle pravidla (p, q) formulovaného v zadání úlohy 2) způsobem jako v 1°.

Postup: Vytvoříme prázdný balíček, resp. balíček naplněný 32 prázdnými místy (počet je volitelný), a pak v cyklu postupně zasouváme karty $1, 2, 3, \ldots, 32$ na ta místa, kam patří. Jak? Nejprve v balíčku vynecháme

P_1 karet (tj. ponecháme v souboru `Balicek` na prvních p_1 místech nulovou hodnotu), pak do následujících Q_1 míst, které dosud mají nulovou hodnotu, vložíme karty s čísly $1, 2, \ldots, Q_1$. Pak postupujeme dál s rostoucím indexem `Ind` a na dalších P_2 místech, kde najdeme nulovou hodnotu, ji ponecháme, a na Q_2 dalších míst, kde najdeme nulovou hodnotu, vložíme čísla $Q_1 + 1, \ldots, Q_1 + Q_2$. Tak pokračujeme stále dokola (cyklus `repeat`) postupně pro další P_i, Q_j (v programu `P[JP]`, `Q[JQ]`) v cyklech `while`. Ukazatel `Ind` na kartu, s níž právě operujeme, sice stále roste, ale příslušné místo v balíčku zůstává díky příkazu `Ind mod N` stále v patřičném rozmezí 0 až $N - 1$.

Nejprve je však třeba vyřešit vstupy. Volíme `N` počet karet a v souboru `Balicek` pro ně rezervujeme místa s indexy `Ind` od 0 (horní karta v balíčku) až po $N - 1$ (spodní karta). Stejně jako v předchozím programu je hodnotou `Balicek[I]` číslo karty, která je v balíčku na místě $I + 1$; kartu, kterou vykládáme jako I-tou, tedy najdeme v `Balicek[I-1]` (viz tisk výsledku). Počet k čísel p_i i počet l čísel q_j může být libovolný, proto volíme jejich zadání jako k-tice, resp. l-tice s koncovým znakem (např. 0). V našem příkladu volíme vstupy p_i jako $3, 2, 1, 0$; počet p_i označíme `PP` $(= 3)$. Vstupy q_j volíme jako $2, 3, 0$ a jejich počet označíme `QQ` $(= 2)$. Posloupnosti (P_m), (Q_m) budeme realizovat pomocí indexů `JP`, `JQ` $= 1, 2, \ldots$ a pomocí příkazů

```
Inc(JP);
if JP > PP then JP := 1;
```

resp.

```
Inc(JQ);
if JQ > QQ then JQ := 1;
```

Po umístění karty N vystoupí na obrazovku výsledné uspořádání karet v balíčku.

```
program Karty2;
uses Crt;
var
  N: Integer;  {pocet karet}
  P, Q: array [1..9] of Integer; {pocet skupin, P dospodu,
                                  Q vylozit}
  PP, QQ: Integer; {velikost skupin P,Q}
  JP, JQ: Integer; {indexy skupin v polich P,Q}
  I, J, Ind, IP, IQ: Integer;
```

```
  {I index cyklu, J postupne umistovana cisla;
  Ind index vysetrovaneho mista;
  IP index od 1 do P[JP]; IQ index od 1 do Q[JQ];}
  Balicek: array [0..31] of Integer;

begin {Karty2}
  ClrScr;
  {vstupy}
  Write('Pocet karet: ');
  ReadLn(N);
  WriteLn('Dospodu stridave (koncovy znak 0): ');
  PP := 0;
  repeat
    Inc(PP);
    ReadLn(P[PP])
  until P[PP] = 0; Dec(PP);
  WriteLn('Vylozit stridave (koncovy znak 0): ');
  QQ := 0;
  repeat
    Inc(QQ);
    ReadLn(Q[QQ])
  until Q[QQ] = 0; Dec(QQ);
  WriteLn;
  {nulovani balicku}
  for I := 0 to N do Balicek[I] := 0;
  {cyklus naskladani karet do balicku}
  J := 1; Ind := 0; JP := 1; JQ := 1;
  repeat
     {P[JP] karet dospodu}
     IP := 1;
     while (IP <= P[JP]) and (J <= N) do
     begin
       if (Balicek[Ind mod N] = 0) then Inc(IP);
       Inc(Ind)
     end;
     Inc(JP);
     if JP > PP then JP := 1;
     {Q[JQ] karet vylozit}
     IQ := 1;
```

```
      while (IQ <= Q[JQ]) and (J <= N) do
      begin
         if (Balicek[Ind mod N] = 0) then
         begin
            Balicek[Ind mod N] := J;
            Inc(J); Inc(IQ)
         end;
         Inc(Ind)
      end;
      Inc(JQ);
      if JQ > QQ then JQ := 1;
   until J > N;
   {vystup vysledku:}
   for I := 1 to N do
   begin
      WriteLn(I:3, '.',Balicek[I-1]:3,'  ');
      if (I mod 8) = 0 then WriteLn
   end;
   ReadLn
end {program Karty2}
```

Jestliže máme např. 16 karet ($N = 16$) a chceme je připravit na systém vyložení „jedna dospodu, jedna vyložit“, pak v prvním programu `Karty1` volíme

Kterou vyložit: 2

nebo ve druhém programu `Karty2` volíme

Dospodu střídavě: 1 a pak koncový znak 0,

Vyložit střídavě: 1 a koncový znak 0.

V obou případech dostaneme balíček v uspořádání:

16 1 9 2 13 3 10 4 15 5 11 6 14 7 12 8

Literatura

[1] Trávníček, S.: Na koho to slovo padne. *Rozhledy matematicko-fyzikální* **49**, č. 10 (1970–71), s. 433–438.