Rozhledy matematicko-fyzikální

Stanislav Trávníček

Modelování adaptace na změny

Rozhledy matematicko-fyzikální, Vol. 89 (2014), No. 3, 11–17

Persistent URL: http://dml.cz/dmlcz/146584

Modelování adaptace na změny

Stanislav Trávníček, PřF UP, Olomouc

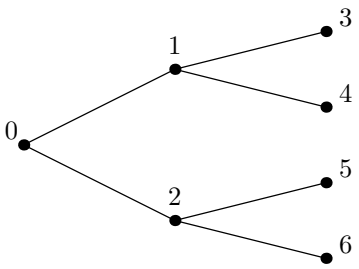
Abstract. The article presents a computer model of an equipment which has the ability to adapt itself to changing conditions.

Schopnost adaptace na životní podmínky a nová adaptace při změně těchto podmínek se nám jeví jako charakteristický rys živoucích bytostí. Dá se však ukázat, že to tak docela neplatí, že takovou adaptaci na dané a změněné podmínky lze modelovat i velmi jednoduchými mechanickými prostředky. Jedno takové zařízení, zhotovené z krabiček od zápalek, popsal ve své knize [1] americký matematik M. Gardner; bylo to ještě v době nepočítačové. Když jsem si v roce 1990 jeho problém formuloval česky, zaujala mě možnost nazvat toto zařízení (s kapkou humoru) *KOZA*: *K*rabičky *O*d *Z*ápalek – *A*daptivní zařízení. Uvidíme, že při troše fantazie si lze skutečně představit, jak nějaká koza hledá potravu. Tvařme se však solidněji a toto zařízení (schopné adaptace) označujme dále zkratkou ZK.

Zmíněné ZK lze samozřejmě modelovat na počítači a v článku [2] jsem se tomu věnoval a seznámil čtenáře s principy a s využitím takového počítačového modelu. Nyní však předkládám přímo program (nazvaný ZK) s jednoduchou verzí ZK, který umožňuje čtenáři samostatně experimentovat a získávat zajímavé poznatky o problému adaptace.

Popis problému

Je dáno schéma cest, v němž jsou tři uzly (0, 1 a 2) a čtyři cíle (3, 4, 5 a 6), viz obr. 1. Naše ZK opakovaně prochází z východiska 0 některou ze čtyř cest k některému ze čtyř cílů, každý takový průchod nazvěme dále *akce*. Cíle jsou buď pozitivní (pro názornost řekněme – ZK tam najde potravu) nebo negativní. Na počátku nemá ZK o kvalitě cílů žádné informace. K cíli se dostane na základě dvou rozhodnutí, vždy v uzlu 0 a podruhé v uzlu 1 nebo v uzlu 2. ZK chce co nejvíce navštěvovat pozitivní cíle, ale občas se podívat i jinam, jestli se i některý z negativních cílů nestal v průběhu posloupnosti akcí cílem pozitivním, a podle toho konat svá rozhodnutí v uzlech.



Obr. 1

Strategie řešení problému

Na počátku není v žádném z uzlů důvod dávat některé cestě přednost, takže *váha rozhodnutí*, jestli při startu první akce volit směr $(0, 1)$ nebo $(0, 2)$ je pro oba případy stejná, podobně i v uzlech 1 a 2. Z pohledu teorie grafů jsou $(0, 1)$ a $(0, 2)$ hrany grafu a váhy rozhodnutí vlastně znamenají ohodnocení hran grafu. Je proto zcela přijatelný počáteční stav, kde všechny hrany mají totéž ohodnocení. Jestliže nyní postupně probíhají jednotlivé akce, sbírá ZK zkušenosti, a to tak, že když dojde k pozitivnímu cíli, celou tu cestu „odmění“ zvýšením ohodnocení obou příslušných hran, ale je-li cíl negativní, celou cestu „potrestá“ snížením ohodnoceni jejích hran.

Je však účelné stanovit minimální ohodnocení jako kladné (kdyby bylo nulové, už by se ZK na tuto cestu nikdy nedostalo, což by mohlo být zlé v případě změny kvality cílů), ale také ohodnocení maximální, které ovlivňuje rychlost adaptace na nové poměry.

V zájmu krátkosti a přehlednosti zdrojového textu předloženého programu jsou vstupní parametry zvoleny přímo v programu, a to minimální ohodnocení jako 1, maximální ohodnocení je dáno proměnnou `MaxH = 5`, počáteční ohodnocení všech hran je dáno proměnnou `Zaklad = 4`. Oba tyto údaje najdeme v úvodu zdrojového textu a lze je při zkoumání adaptace měnit. S troškou znalostí o programování lze jednoduše změnit i kvalitu cílů. Na obr. 2 je obsah obrazovky po 120 akcích, jak je realizuje program ZK. Na počátku byl nastaven cíl 3 jako jediný úspěšný (horní polovina obrazovky), a pak se náhle cíl 3 stal neúspěšný a úspěšným se stal cíl 6 (dolní polovina obrazovky).

Záznamy o akcích jsou tvořeny takto: první trojice čísel (například 0–1–4) je cesta ZK od počátku 0 do cíle, další číslo, 0 nebo 1, ukazuje, zda cesta byla úspěšná (0 = neúspěšná) a poslední písmeno, A nebo N, vyja-

dřuje, zda se ZK touto cestou „poučilo“, tj. došlo ke změně v ohodnocení některé hrany grafu ve výše uvedeném smyslu. Vidíme, že v 1. desítce bylo pro ZK všechno nové, učilo se, ve druhé a třetí desítce už začalo mít a využívat zkušeností (roste počet N), pak už (v tomto konkrétním chodu programu) je A-ček docela málo. Nulová úspěšnost některých cest nebyla pro ZK na závadu, jen testovalo, jestli se nezměnila kvalita cílů zatím neúspěšných.

```
0-1-4:0-A   0-1-4:0-A   0-2-6:0-N   0-1-3:1-N   0-1-3:1-N   0-2-5:0-N
0-2-5:0-A   0-2-5:0-A   0-1-3:1-N   0-1-3:1-N   0-1-3:1-N   0-2-6:0-N
0-1-3:1-A   0-1-4:0-A   0-1-3:1-N   0-1-3:1-N   0-1-3:1-N   0-1-3:1-N
0-2-6:0-A   0-2-6:0-A   0-1-4:0-A   0-1-3:1-N   0-1-3:1-N   0-1-3:1-N
0-1-4:0-A   0-1-3:1-A   0-1-3:1-A   0-1-3:1-N   0-1-3:1-N   0-1-3:1-N
0-2-5:0-A   0-1-3:1-A   0-1-3:1-N   0-1-3:1-N   0-1-3:1-N   0-1-3:1-N
0-1-3:1-A   0-1-3:1-N   0-1-3:1-N   0-2-6:0-N   0-1-3:1-N   0-1-3:1-N
0-1-3:1-A   0-2-6:0-A   0-1-4:0-A   0-1-3:1-N   0-2-6:0-N   0-1-3:1-N
0-1-4:0-A   0-1-3:1-N   0-1-3:1-A   0-1-3:1-N   0-1-4:0-A   0-1-3:1-N
0-1-3:1-A   0-1-3:1-N   0-1-3:1-N   0-1-3:1-N   0-1-3:1-A   0-1-3:1-N

0-1-3:0-A   0-2-6:1-A   0-2-6:1-N   0-1-4:0-N   0-2-6:1-N   0-2-6:1-N
0-1-3:0-A   0-2-6:1-N   0-2-6:1-N   0-2-6:1-N   0-2-5:0-A   0-2-6:1-N
0-2-6:1-A   0-2-6:1-N   0-2-5:0-A   0-2-6:1-N   0-2-6:1-A   0-1-3:0-N
0-2-6:1-A   0-1-4:0-N   0-2-6:1-A   0-2-6:1-N   0-2-6:1-N   0-2-6:1-N
0-1-3:0-A   0-2-6:1-N   0-2-6:1-N   0-2-6:1-N   0-2-6:1-N   0-1-3:0-N
0-2-6:1-A   0-1-4:0-N   0-2-6:1-N   0-1-4:0-N   0-1-4:0-N   0-2-6:1-N
0-2-6:1-A   0-2-6:1-N   0-2-6:1-N   0-2-6:1-N   0-2-6:1-N   0-2-6:1-N
0-1-3:0-A   0-2-5:0-A   0-1-4:0-N   0-2-6:1-N   0-2-6:1-N   0-2-6:1-N
0-1-4:0-N   0-2-6:1-A   0-2-5:0-A   0-1-3:0-N   0-2-6:1-N   0-1-4:0-N
0-2-5:0-A   0-2-6:1-N   0-2-6:1-A   0-2-6:1-N   0-2-6:1-N   0-1-3:0-N
```

Obr. 2

A potom došlo k obrovské změně (tj. pro 61. akci jsem ji takto do programu nastavil): náhle se cíl 3 stal neúspěšný a ZK zkoumalo, jestli je některý z dalších cílů úspěšný (byl to nyní cíl 6). Vidíme, že ZK šlo ještě dvakrát „postaru“ cestou 0–1–3 a pak (náhodou celkem brzy) objevilo úspěšnou cestu 0–2–6, ale úplně jí nevěřilo a ještě se znovu vracelo k předchozí cestě 0–1–3. V 6. desítce získávalo zkušenosti a pak začalo více uplatňovat nový poznatek 0–2–6.

Je samozřejmé, že když je graf cest složitější, jsou výsledky pestřejší; např. mohou existovat dva od sebe odlehlé úspěšné cíle.

Nahlédněme chvíli do kuchyně programování. Ohodnocení hran (a, b) je uloženo v proměnných `Hrana[a,b]`, hodnoty cílů v proměnných `Cil[3]` až `Cil[6]`, čísla uzlů při právě probíhající akci jsou hodnotami `Cesta[0]`, `Cesta[1]`, `Cesta[2]` proměnné `Cesta`. Jejím prostřednictvím jsou pak hrany odměňovány nebo trestány.

Rozhodování v uzlech 0, 1, 2 je založeno na generátoru náhodných čísel. Provádí se takto: sečte se ohodnocení obou navazujících hran, nechť jsou to např. čísla 1 a 3. Tedy součet je 4; pro jednoduchost a větší

přesnost se pak pracuje s desetinásobky, takže se vygeneruje celé číslo do 40 a pokud je toto číslo menší než 10, volí se první hrana, jinak druhá hrana. Aby se předešlo nějaké setrvačnosti generátoru náhodných čísel, používá program až každé páté vygenerované číslo (i když je to možná zbytečná obava).

O každé akci program vypracuje a vytiskne řádkový protokol, jak už jsme se zmínili shora. Abychom mohli program používat bez potřeby vnějších informací, vystoupí potřebné informace na obrazovce.

Vzhledem k použití generátoru náhodných čísel dostaneme při každém běhu programu jiný výsledek, ale adaptace vždy proběhne podobně. Je docela zajímavé se po každém chodu programu přesvědčit, jak tentokrát adaptace probíhala.

```
program ZK;
uses Crt;
type TStr10 = string[10];
const
  Zaklad = 4;
  MaxH = 5;
var
  Hrana  : array [0..2,1..6] of Integer;
  Cil    : array [3..6] of Integer;
  Cesta  : array [0..2] of Integer;
  Radek  : TStr10;
  Zmena: Boolean;
  Ii, Jj, Kk, Ra, Rade, Slou: Integer;
procedure Uvod;
begin                {Úvod}
  ClrScr;  WriteLn; TextAttr := White;
  WriteLn(' Program K O Z A '); WriteLn;
  WriteLn(' Program postupně generuje cesty
            z výchozího uzlu 0  do cíle. ');
  WriteLn;
  WriteLn(' Z výchozího bodu 0 lze postoupit do uzlu 1
            nebo do uzlu 2; ');
  WriteLn(' z uzlu 1 se postoupí do cíle 3 nebo
            do cíle 4,' );
  WriteLn(' z uzlu 3 se postoupí do cíle 5 nebo
            do cíle 6.' );
```

```
  WriteLn;
  WriteLn(' Každý z cílů má hodnotu 1 (úspěch)
            nebo 0 (neúspěch). ');
  WriteLn;
  WriteLn(' V každém z uzlů 0, 1, 2 se rozhoduje
            o dvou směrech; ');
  Writeln(' každý směr má určitou váhu (od 1 do 5)
            a rozhodování o směru ');
  WriteLn(' dalšího postupu respektuje poměr velikosti
            obou vah. ');
  WriteLn;
  WriteLn(' Váhy uzlů se v průběhu mění na základě
            zkušenosti systému: ' );
  WriteLn(' byl-li cíl úspěšný, váhy na této cestě
            se posílí, ');
  WriteLn(' byl-li cíl neúspěšný, váhy na této cestě
            se zeslabí; ');
  WriteLn(' nejmenší možná váha je 1, největší je
            MaxH = 5. ');
  WriteLn(' Na počátku je ve všech uzlech nastavena
            hodnota Zaklad = 4. ');
  WriteLn;
  WriteLn(' V daném nastavení se v polovině procesu mění
            úspěšný cíl z 3 na 6. ');
  WriteLn; Write(' Dále Enter:');
  ReadLn; ClrScr
end;  {Uvod}
procedure NastavCile(C: Integer);
begin
  for Ii := 3 to 6 do Cil[Ii] := 0;
  if C = 0 then Cil[3] := 1
  else  Cil[6] := 1
end;
procedure NastavHrany;
begin
  for Ii := 0 to 2 do
   for Jj := 1 to 6 do
    Hrana[Ii,Jj] := Zaklad
end;
```

```
procedure Akce;
var SA, SB: Integer;
procedure Projdi(K1,K2: Integer);
var AJ: Integer;
begin
  SA := Hrana[K1,K2] + Hrana[K1,K2+1];
  SA := 10 * SA;
  for AJ := 1 to 5 do
    SB := Random(SA) + 1;
  if SB < 10 * Hrana[K1,K2] then
  begin
    Cesta[JJ] := K2;
    Radek := Radek + '-' + Chr(K2+48)
  end else
  begin
    Cesta[JJ] := K2 + 1;
    Radek := Radek + '-' + Chr(K2+49)
  end
end;
begin
  Cesta[0] := 0; Radek := '0';
  for JJ := 1 to 2 do
    Projdi(Cesta[JJ-1],2 * Cesta[JJ-1] + 1);
  Radek := Radek + ':';   Zmena := false;
  if Cil[Cesta[2]] = 1 then
  begin
    Radek := Radek + '1-';
    for Kk := 0 to 1 do
    if Hrana[Cesta[Kk],Cesta[Kk+1]] < MaxH then
    begin
      Inc(Hrana[Cesta[Kk],Cesta[Kk+1]]);
      Zmena:= true
    end
  end else
  begin
    for Kk := 0 to 1 do
     if Hrana[Cesta[Kk],Cesta[Kk+1]] > 1 then
     begin
       Dec(Hrana[Cesta[Kk],Cesta[Kk+1]]);
```

```
        Zmena := true
      end;
     Radek := Radek + '0-'
  end;
  if Zmena then Radek := Radek + 'A'
  else Radek := Radek + 'N'
end;
{**************************************}
begin            {program}
  Randomize;
  Uvod;
  NastavHrany;
  Rade := 0;
  repeat
    Slou := 1;
    NastavCile(Rade);
    repeat
      Ra := Rade;
      for Ii := 1 to 10 do
      begin
        Akce;
        Inc(Ra);
        GotoXY(Slou+1,Ra+1);
        Write(Radek)
      end;
      Inc(Slou,12)
    until Slou > 65;
    Rade := 12
  until Ra > 20;
  ReadLn
end.             {program}
```

Literatura

[1] Gardner, M.: *Matematičeskije dosugi*. Mir, Moskva 1972. (převzato z publikací *New Mathematical Diversions from Scientific American* a *The Unexpected Hanging and Other Mathematical Diversions*).

[2] Trávníček, S.: Zařízení pro modelování adaptace. *MFI* **2** (1992/93), č. 5, s. 257–264.