INFORMATIKA

Life na ostrově

STANISLAV TRÁVNÍČEK
Přírodovědecká fakulta UP, Olomouc

Hru Life vymyslel známý matematik *John Conway*; s první rozsáhlejší informací o ní přišla u nás kniha [1]. Hra v zjednodušené formě simuluje to, co se odehrává v živé přírodě u nějaké populace organizmů. Generace zde střídá generaci a populace roste, proměňuje se, stabilizuje nebo zaniká. Tak je tomu i ve hře Life.

Hra se odehrává na neohraničené čtvercové síti, tedy na čtverečkovaném papíru; jedinec populace obsazuje vždy právě jeden čtvereček (element sítě) a může mít až 8 sousedů (vodorovně, svisle i šikmo v obou směrech). Na počátku je zadáno rozložení (konfigurace) počáteční (nulté) generace a hra se pak řídí těmito pravidly přechodu z generace na generaci:

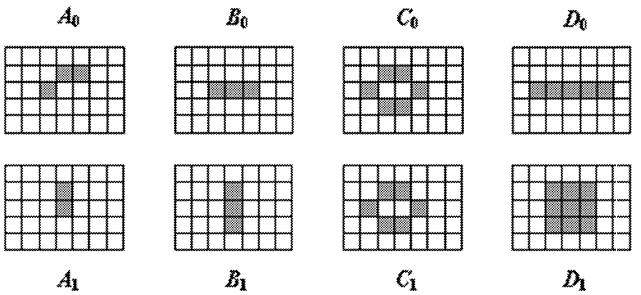
- *pravidlo přežití:* každý jedinec, který má ve výchozí generaci 2 nebo 3 sousedy, přežije do příští generace;
- *pravidlo zániku:* každý jedinec, který ve výchozí generaci nemá souseda nebo má 1 souseda, zanikne (v důsledku samoty); jedinec, který má více než 3 sousedy, také zanikne (pro nedostatek životního prostoru v důsledku lokálního přemnožení organizmů);
- *pravidlo zrození:* na prázdném místě, které má ve výchozí generaci právě 3 sousedy, vznikne nový jedinec.

Celá výchozí generace se vždy proměňuje najednou.

Příklady

Na obrázku 1 ve výsecích nekonečné sítě jsou v 1. řádku nulté generace nějakých kolonií organizmů, ve 2. řádku příslušné 1. generace vzniklé

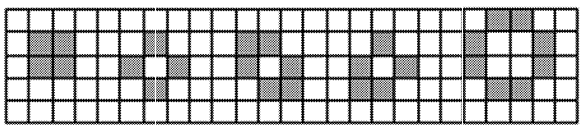
podle pravidel přechodu. Lehce zjistíme, že v případě A kolonie zanikne ve 2. generaci. Ve případě B bude 2. generace shodná s nultou, tedy dostáváme periodicky se měnící populaci s periodou 2. V případě C budou všechny generace shodné, populace má stabilní tvar. V případě D nelze další průběh změn ani „délku existence“ populace předpovědět, musí se to vyzkoušet. (Zde přejde populace v 6. generaci do periodického stavu, tvořeného v každé další generaci dvěma útvary typu B_0 a dvěma B_1.)



Obr. 1

Jsou možné tyto konce hry:
- populace zanikne;
- nebo zaujme stabilní rozložení;
- nebo se periodicky proměňuje;
- nebo se stále a neperiodicky mění.

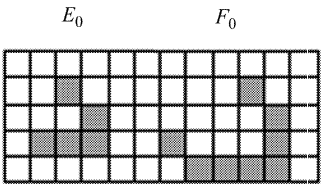
Na obr. 2 jsou znázorněny některé nejjednodušší stabilní tvary skupiny organizmů.



Obr. 2

Jestliže tuto hru hrajeme „na ostrově“, tj. např. na obdélníku 20×30, pak poslední uvedený případ asi nenastane, i když populace může přežít mnoho set generací, aniž by skončila některým z prvních tří případů. Podstatný rozdíl je totiž v tom, že zatímco u původní hry může populace

neomezeně putovat, na ostrově jí v tom „moře“ zabraňuje, protože zde předpokládáme, že v „moři“ se nenaplní pravidlo zrození (tj. nový jedinec může vzniknout jen na souši – na ostrově). Na obr. 3 jsou dva příklady takových kočovných populací.



Obr. 3

Populace E se proměňuje a ve 4. generaci se objeví přesunutá vpravo dolů o jeden element sítě; při neohraničené síti postupuje tedy každou čtvrtou generaci stále dál a dál. Populace F kočuje ještě rychleji. Každou druhou generaci se objeví překlopená dolů a o jeden element vpravo, takže každou čtvrtou generaci se objeví v původní podobě F_0, ale o dva elementy vpravo. Zajímavé je chování těchto kočovníků na ostrově. Populace F, když narazí na pravý břeh, se od břehu „odrazí“ a promění se v populaci typu E. Pokud byl mezi F_0 a pravým břehem lichý počet prázdných elementů, putuje vzniklá populace E šikmo vlevo dolů, byl-li počet prázdných elementů sudý, putuje E šikmo vlevo nahoru. Narazí-li E na libovolný břeh, změní se na stabilní „čtverec 2×2“.

Sám John Conway prostudoval tuto hru do velké hloubky a je velmi zajímavé se s jeho výsledky seznámit alespoň v rozsahu uvedeném v [1]. Můžeme si sami vymýšlet libovolné nulté generace a pak sledovat, jak se populace vyvíjí dále. Počet konfigurací nulté generace je na dosti velkém ostrově tak obrovský, že každý hráč může prožít objevitelské pocity, když mu před očima procházejí jednotlivé generace. Někdy se populace velmi rozšíří, aby nakonec zmizela, jindy se rozdělí na izolované části, z nichž každá si žije svým životem, některé části se pak zase začnou rozšiřovat nebo putovat a rozvrátí klidný život jiné části, pokud se k ní dostatečně přiblíží, atd. Často lze sledovat i mnoho set generací. Ze všech dalších známých poznatků o hře Life přidejme už jenom tento: Vývoj populací není stabilní v tomto smyslu: i malá změna v nulté generaci může vést (a zpravidla vede) k naprosto odlišnému celkovému vývoji i konci. (Např. když linku v D_0 prodloužíme o 1, zanikne populace v 10. generaci.)

Je asi jasné, že pravý požitek ze hry máme tehdy, když hru naprogramujeme pro počítač. Pak je také srozumitelné, proč je vhodné uvažovat o populacích jen na ostrově – neohraničené obrazovky neexistují. Autor tohoto článku tuto hru naprogramoval poprvé už před více než 30 lety pro počítač Minsk 22. Protože však jediným vhodným výstupem byla tehdy řádková tiskárna, bylo možné (z důvodu spotřeby papíru) jen vyzkoušet funkčnost při zobrazení několika generací, takže vlastně šlo jen o cvičení v programování. Až dnešní rychlé osobní počítače s barevným displejem přinášejí hráči ten správný požitek.

Čtenáři, který by dostal chuť si hru Life naprogramovat (i přesto, že jsme z jejího bohatství zajímavých poznatků mohli vybrat jen to zcela nejnutnější), nabídneme nyní jako podnět několik výsledků programové analýzy z existujícího programu Life, který funguje a v provozu se osvědčil. Samozřejmě však lze postupovat i jinak.

Podívejme se nejprve na uživatelskou tvář programu:

Menu:

1. Nová akce – standard
2. Zpět na předchozí začátek
3. Volba ostrova
4. Zpět na předchozí ostrov
5. Zadání ze souboru
6. Uložení zadání do souboru
7. O hře Life
8. Konec práce programu

V 7. bodě menu je uvedena informace o hře Life a dále pak tento návod k používání programu:

„Režim práce si vyberete v menu. Při volbě standardní hry se zobrazí maximální ostrov (62×38) a v jednom ze středních polí znak X. Tímto znakem se pohybuje pomoci kurzorových kláves. Jedince populace zadáte Mez a stejně ho můžete i zrušit. Hromadného nastavení linky či obdélníku můžete dosáhnout užitím Tab v koncových bodech požadované linky resp. v protějších vrcholech obdélníku. Nastavení ukončíte klávesou Enter.

Při volbě 2. bodu menu se vám zobrazí původní nastavení z právě ukončené hry, které máte ještě možnost měnit. Při volbě 3. bodu můžete nastavit velikost ostrova. Při volbě 4. bodu se nastaví ostrov předchozí velikosti. V dalších bodech menu máte možnost výchozí nastavení uložit do souboru

nebo si nahrát postaveni ze souboru již dříve uloženého. Tyto soubory mají jméno Life-xxx.dat, kde místo znaků xxx zadáte tři číslice nebo písmena.

V průběhu hry můžete užitím kurzorových kláves měnit způsob hry – volit hru po krocích nebo hru plynulou. Hra se automaticky zastaví, když populace přejdou do stavu stabilního, periodického s periodou 2 nebo při zániku populace. Po takovém zastavení programu se po stisku libovolné klávesy přejde do menu. Přerušit práci a přejít do menu lze klávesou Esc. Výtah z tohoto návodu máte k dispozici i při hře – na servisním panelu.

Ať vám vycházejí pěkné obrázky a ať se vám líbí čas při této hře strávený.“

Ostrovy jsou obdélníkové a skládají se z elementů (čtverečků) uspořádaných do řádků a sloupců. Největší možný ostrov je dán deklarací

```
const
  RaMax  = 38;   {maximum radku: 38}
  SlMax  = 62;   {maximum sloupcu: 62};
```

tyto rozměry vycházejí z rozměru obrazovky 640×480 pixelů a z rozměru elementu pole 10×10 (z toho vždy 1 „černý“ pixel se používá na vytvoření hranice „šedých“ elementů o rozměrech 9×9 pixelů), přičemž ostrov je obrouben „modrým mořem“, tj. rámem o šířce 1 elementu a v dolním pruhu obrazovky je „servisní panel“. Vodorovný rozměr (délku ostrova) lze volit v rozmezí $10 - 62$ elementů, svislý rozměr (šířku ostrova) v rozmezí $5 - 38$ elementů. V případě volby menšího ostrova vyplňuje moře celou zbývající plochu obrazovky mimo ostrov a servisní panel. Ostrov se tedy jeví jako modře orámovaná černá obdélníková síťka na šedém podkladu. V případě, že při vytváření nulté generace je v elementu zadán jedinec populace, změní se šedý obsah čtverečku na červený (a v dalších generacích se stále střídá červená se zelenou).

Pro elementy ostrova jsou definována 4 bytová pole:

```
Pole: array [0..3, 0..SlMax + 1, 0..RaMax + 1] of Byte;
```

Zadání nulté generace se provádí do Pole[0, X, Y], které se během celé hry uchovává pro případ příští možné volby 2. bodu menu. Jedinec populace v místě [X, Y] je vyjádřen jedničkou v příslušném místě Pole[0, X, Y], jinak je tu nula. Ve všech polích jsou tedy řádky i sloupce „moře“ trvale

vynulovány (při maximálním ostrovu jsou to dva krajní řádky a dva krajní sloupce).

Informace, které pro nastavení nulté generace dává servisní panel, jsou zřejmé z příslušné procedury:

```
procedure ServisniPanel1;
begin {ServisniPanel1}
  OutTextXY(10, 405, 'Servisni Panel:');
  OutTextXY(145, 405, 'sipky -- pohyb X po ostrove');
  OutTextXY(145, 420, 'MEZ   -- nastaveni a ruseni
                              organizmu');
  OutTextXY(145, 435, 'TAB   -- nastaveni skupiny
                              organizmu');
  OutTextXY(145, 450, 'ENTER -- ukonceni nastaveni');
  OutTextXY(145, 465, 'ESC   -- preruseni prace a navrat
                              do menu');
  OutTextXY(572, 405, 'L i f e')
end;  {ServisniPanel1}
```

Nastavení nulté generace řídí

```
procedure Nastaveni;
var
  KonecNastaveni: Boolean;
begin {Nastaveni}
  KonecNastaveni := false;
  repeat
    Key := GetKey;
    case Key of
    _Up  : if YZde > Y0 then DelY := -1;
    _Down: if YZde < YM then DelY := 1;
    _Left: if XZde > X0 then DelX := -1;
    _Right: if XZde < XM then DelX := 1;
    _Tab :
         begin
           if Zapln = false then
             PripravaNaZaplneni
           else
```

```
          begin
            ZaplnObdelnik;
            Zapln := false
          end
        end;
    _Mez :
        begin
          DoPole;
          if Zapln then
          begin
            ZaplnObdelnik;
            Zapln := false
          end
        end;
    _CR  : KonecNastaveni := true;
    _Esc :
        begin
          Likviduj := true;
          KonecNastaveni := true;
          Zapln := false
        end
    end;
    if (Key in [_Up, _Down, _Left, _Right]) then NaPole
  until KonecNastaveni
end;  {Nastaveni}
```

Stisk kurzorových kláves přesunuje znak X prostřednictvím procedury *NaPole*, Mez nastavuje nebo ruší nastavení populace v elementu, funkce CR a Esc je zřejmá. Tab usnadňuje nastavování dlouhých linií nebo obdélníků. Při prvním stisku Tab se provede procedura *PripravaNaZaplneni*, která označí zvolený element, zaznamená jeho souřadnice a nastaví Booleovskou proměnnou Zapln na true. Při druhém stisku Tab (na jiném elementu) se zaplní požadovaný obdélník a Zapln se nastaví na false (vedle druhé volby Tab funguje takto i Mez).

Na začátku hry (po nastavení nulté generace a stisku CR) se obsah Pole[0, X, Y] přesune do Pole[1, X, Y], které se během hry zobrazuje na displeji; stane se tak v proceduře

```
procedure StartGenerace;
begin {StartGenerace}
  VymazX;
  MazServis(145, 465, 405, 472);
  PresunPole(0, 1);  IndG := 0;
  Generace := 0;
  Krokovat := true;
  ServisniPanel3;
  VystupGenerace
end;  {StartGenerace}
```

Servisní panel je ošetřen procedurami

```
procedure ServisniPanel2;
begin {ServisniPanel2}
  OutTextXY(145, 405, 'nastaven rezim Krokovani -');
  OutTextXY(155, 420, '(dale MEZ nebo jinou klavesou)')
end;  {ServisniPanel2}

procedure ServisniPanel3;
begin {ServisniPanel3}
  ServisniPanel2;
  OutTextXY(145, 435, 'zmena rezimu sipkami');
  OutTextXY(145, 465, 'GENERACE:')
end;  {ServisniPanel3}

procedure ServisniPanel4;
begin {ServisniPanel4}
  OutTextXY(145, 405, 'nastaven rezim Plynuly chod')
end;  {ServisniPanel4}
```

Střídání barev je připraveno deklarací:

```
BaGen: array [0..1] of Integer = (4, 2); {barva generace}
```

Hlavní procedurou hry je vytvoření další generace; ta se vždycky vytváří v poliPole[1, X, Y], tj. po výstupu celé jedné generace na displej je nutno toto pole uvolnit pro další generaci a to tak, že obsah Pole[2, X, Y]

přesuneme do Pole[3, X, Y], a pak ještě Pole[1, X, Y] do Pole[2, X, Y]. Uchování tří generací umožňuje automatické zastavení hry; je-li obsah vytvořeného Pole[1, X, Y] stejný jako obsah Pole[2, X, Y], znamená to, že populace přešla do stabilního stavu, a je-li obsah Pole[1, X, Y] shodný s obsahem Pole[3, X, Y], probíhají v populaci jen periodické změny s periodou 2, což je perioda nejčastější (případné větší periody si musí pohlídat hráč). Procedura *VytvarejGenerace* zajišťuje vytváření další generace podle uvedených pravidel přechodu a zároveň zařídí postupný výstup nové generace na displej, tj. výstup všech elementů v potřebné barvě, kterými se přemaže předchozí stav.

```
procedure VytvarejGenerace;
var
  SVG: Integer;
begin {VytvarejGenerace}
  PresunPole(2, 3);
  PresunPole(1, 2);
  IndG := 1-IndG;
  for X := X0 to XM do
    for Y := Y0 to YM do
    begin
      SVG := 0;
      SVG := Pole[2,X-1,Y-1]+Pole[2,X,Y-1]+Pole[2,X+1,Y-1]+
             Pole[2,X-1,Y+1]+Pole[2,X,Y+1]+Pole[2,X+1,Y+1]+
             Pole[2,X-1,Y] + Pole[2,X+1,Y];
      case SVG of
        2:     {element se nemeni}
          begin
            Pole[1,X,Y] := Pole[2,X,Y];
            if Pole[1,X,Y] = 0 then
               Blok(X,Y,BaNast[0])
            else
               Blok(X,Y,BaGen[IndG]);
          end;
        3:     {jedinec vznikne nebo zustane}
          begin
            Pole[1,X,Y] := 1;
               Blok(X,Y,BaGen[IndG]);
```

```
          end;
        else   {jedinec zanikne}
          begin
            Pole[1,X,Y] := 0;
            Blok(X,Y,BaNast[0])
          end
      end
    end
end;  {VytvarejGenerace}
```

Procedura *Blok* provádí výstup elementu na obrazovku.

```
procedure Blok(XXB, YYB: Integer; BBB: Word);
var
  IBB, JBB, IIB, JJB: Integer;
begin {Blok}
  IIB := 10*XXB;
  JJB := 10*YYB;
  for IBB := IIB to IIB+8 do
    for JBB := JJB to JJB+8 do
      PutPixel(IBB, JBB, BBB)
end;  {Blok}
```

Procedura *VytvarejGenerace* je vnořena do procedury

```
procedure PokracujGenerace;
begin {PokracujGenerace}
  ZdaKrokovat;  if not Likviduj then
  begin
    Inc(Generace);
    MazServis(220, 300, 465, 472);
    VystupGenerace;
    VytvarejGenerace
  end
end;  {PokracujGenerace}
```

Procedura *VystupGenerace* doplňuje do servisního panelu číslo právě vytvářené generace.

```
procedure VystupGenerace;
var
  Gen: string;
begin {VystupGenerace}
  Str(Generace, Gen);
  OutTextXY(220, 465, Gen)
end;  {VystupGenerace}
```

Ukončení hry je organizováno touto sekvencí programu:

```
if not Likviduj then
begin
  StartGenerace;
  repeat
    PokracujGenerace  until Zastav
end;
if not Likviduj then
begin
  ServisniPanel5;
  ClearBuffer;
  Postuj
end;
```

Procedura *ServisniPanel5* hodnotí výsledek:

```
procedure ServisniPanel5;
begin {ServisniPanel5}
  case JakyKonec of
    1: OutTextXY(260, 465, '- zanik populace');
    2: OutTextXY(260, 465, '- setrvaly stav');
    3: OutTextXY(260, 465, '- periodicke opakovani')
  end
end;  {ServisniPanel5}
```

Funkce *Zastav* pracuje, jak již bylo zmíněno výše:

```
function Zastav: Boolean;
```

```
var
  IZ, JZ: Integer;
begin {Zastav}
  Zastav := true;
  JakyKonec := 1;
  if JsouStejna(1, 2) then
  begin
    JakyKonec := 2;
    exit
  end;
  if (JsouStejna(1, 3) and (Generace <> 1)) then
  begin
    JakyKonec := 3;
    exit
  end;
  for IZ := X0 to XM do
    for JZ := Y0 to YM do
      if Pole[1, IZ, JZ] <> 0 then
          Zastav := false;
  if Likviduj then
    Zastav := true
end;  {Zastav}
```

Komentovaný program Life má přes 1000 řádků, což je ovšem pro MFI rozsah příliš velký. Uvedli jsme zde proto jen návrh řešení klíčových míst, podle nichž si – jak lze doufat – může čtenář udělat představu o celém programu a pokusit se jej sestavit tak nebo jinak k potěšení svému i svých bližních.

Literatura

[1] *Gardner, M.:* Matematičeskije dosugi. Mir, Moskva 1972.

Cabri vkročilo do třetí dimenze

ANTONÍN VRBA

Praha

V době, kdy si počítačový program Cabri II získával oblibu mezi učiteli planimetrie v mnoha zemích včetně té naší, pracovali již ve francouzském Grenoblu jeho autoři na obdobném programu pro podporu výuky stereometrie a deskriptivní geometrie. První verze programu Cabri 3D byla uvedena do praxe na přelomu let 2004/2005. Autoři zvolili metodu dodělávání za pochodu, takže pak následovalo asi s půlroční periodicitou ještě několik aktualizací, které vždy znamenaly významné rozšíření možností. Poslední aktualizace je z listopadu 2006 a nese název Cabri 3D v2 (verze 2.0.0 – 279). V tomto článku budeme komentovat některé vlastnosti a možnosti využití tohoto programu v posledně uvedené verzi. Očekáváme, že většina čtenářů je seznámena s Cabri II, takže se občas na některé jeho vlastnosti odvoláme, aniž bychom zacházeli do podrobností. Ano, mohli bychom zde výklad ilustrovat černobílými statickými obrázky, ale neuděláme to. Čtenář má totiž možnost stáhnout si zkušební verzi Cabri 3D z internetu [1] a vyzkoušet si program v plné dynamičnosti a v barvách. (Zkušební verze funguje 30 dní jako plně funkční a teprve pak přejde do omezeného demonstračního režimu).

Cabri 3D umožňuje vytvářet geometrické objekty v trojrozměrném prostoru, provádět stereometrické konstrukce, měřit geometrické veličiny a zobrazovat stereometrické situace v několika typech promítání. Program má interaktivní a dynamické vlastnosti podobné jako jeho planimetrický předchůdce Cabri II, tzn. objekty lze přemísťovat nebo deformovat se zachováním vzájemných vazeb a souvislostí. Cabri 3D prezentuje ovšem modely trojrozměrných geometrických situací promítnutím na dvojrozměrnou obrazovku. Prostorové iluze se dosahuje výtvarnými prostředky, zejména zeslabením vzdálenějších objektů, plastickým vybarvením ploch a zobrazením viditelnosti překrývajících se objektů. Grafické aspekty si uživatel může ve značné míře upravovat. K dispozici je několik typů pravoúhlého, kosoúhlého a středového promítání. Uživatel může také měnit směr pohledu, ze kterého pozoruje danou geometrickou situaci, tj. měnit vzájemnou polohu promítací soustavy (průmětny – obrazovky a středu, resp. směru promítání) vůči zobrazovaným objektům, přičemž se zachovává pů-

vodně zvolený typ promítání. V jednom dokumentu lze otevřít několik oken s různými typy projekce téže situace, přitom akce, které se provádějí v jednom okně, se simultánně realizují i v ostatních oknech. Význam pro podporu výuky deskriptivní geometrie je tu nesporný.

Objekty, které lze přímo vytvářet, jsou bod, přímka, úsečka, polopřímka, vektor, kružnice, kuželosečka, rovina, polorovina, (rovinný) úhel, mnohoúhelník (i nekonvexní nebo sám sebe protínající), různé typy mnohostěnů, koule, kužel, válec. K dispozici jsou pak konstrukční nástroje pro vytváření průsečíků a průsečnic, rovnoběžných a kolmých přímek a rovin, středu a roviny souměrnosti dvojice bodů, nanášení délky a pod., některé i ve více variantách. Další nástroje realizují shodná zobrazení v prostoru (stejnolehlost zde na rozdíl od Cabri II k dispozici není). Ke geometrickým objektům lze připojovat názvy. Jednotlivé nástroje jsou opatřeny nápovědou a kromě toho se při použití nástroje během zadávání vstupních parametrů zobrazuje na pokračování komentář, např. „Rovina jdoucí: tento bod ...“, „ ... a nový bod (v prostoru) ...“, „ ... a bod C“. Přitom můžeme sledovat postupně vznikající a stabilizující se objekt. Z méně obvyklých nástrojů stojí za zmínku vytvoření mnohostěnu jako konvexního obalu bodů, úseček, mnohoúhelníků a mnohostěnů, odříznutí části mnohostěnu rovinou a zejména rozložení mnohostěnu v jeho síť. Do dokumentů Cabri 3D lze vkládat i textová pole.

V Cabri 3D lze měřit délky (úseček, vektorů, kružnic, kruhových oblouků a elips), obvody mnohoúhelníků, vzdálenosti (dvou bodů, bodu od přímky, bodu od roviny a dvou přímek), velikosti rovinných úhlů a odchylky (dvou různoběžek a přímky od roviny), obsahy (mnohoúhelníků, kruhů, elips), povrchy (mnohostěnů a koulí) a objemy (mnohostěnů, koulí, kuželů a válců – i s eliptickou podstavou a šikmou osou). Pro ostatní případy je nutno si vypomoci buď konstrukcí převádějící měření na některý z uvedených případů (např. odchylku dvou rovin změříme jako odchylku dvou přímek k nim kolmých), nebo si vzpomenout na vzorec a využít kalkulačku. (To se týká povrchu válce a kužele, které tu nejsou měřitelné přímo. U kosého válce a kužele jde ovšem o problém obtížně numericky řešitelný, ale ani pro rotační válec a kužel není v Cabri 3D povrch měřen.) Není zde variabilita jednotek jako v Cabri II, vše se měří v cm, cm^2, cm^3 a stupních, což pro školní účely postačuje, až na to, že chybí radiány. (Zajímavé je, že kalkulačka interpretuje vstupní hodnotu goniometrických funkcí ve stupních, pokud je znak stupně u hodnoty uveden, a v radiánech, pokud hodnota jednotkou opatřena není.) U každého číselného údaje lze určit,

bude-li zobrazen ve formě desetinného zlomku (a určit počet desetinných míst), nebo (pokud lze takto vyjádřit) v algebraickém tvaru $\frac{a+b\sqrt{c}}{d}$, kde a, b, c, d jsou celá čísla.

Kalkulačka v Cabri 3D nenapodobuje reálnou kalkulačku s tlačítky, je to jen dvojice okének, do jednoho se vkládají z klávesnice výrazy a v druhém se průběžně prezentují výsledky. Do výrazů lze vkládat i numerické veličiny z nákresny, zejména naměřené hodnoty nebo již dříve vypočtené hodnoty výrazů. Výsledky lze vytáhnout na nákresnu a při změnách obrázku způsobujících změny vložených hodnot se pak dynamicky přepočítávají. Geometrické veličiny kalkulačka zpracovává i s jednotkami, ale ve shodě s měřením jen na úrovni základních jednotek, nepřepočítává tedy např. metry na centimetry nebo dokonce na palce, jako to uměla kalkulačka v Cabri II. Na rozdíl od Cabri II nelze také poklepáním na výsledek rekonstruovat příslušný výraz.

V Cabri 3D lze zobrazit souřadnice bodů a vektorů (vzhledem ke kartézské soustavě dané základním reperem, který se zobrazuje v obrázcích). Souřadnice lze vkládat i do kalkulačky, ale jen po jednotlivých složkách. Pro standardní stereometrické výpočty bychom očekávali, že kalkulátor zvládá i základní operace s vektory, ale kupodivu tomu tak není, i když to nemůže být velký technický problém. Šikovnou pomůckou je možnost editovat souřadnice bodu (s příslušným stupněm volnosti) a umisťovat tak body do pozic s danými souřadnicemi. U rovin a koulí si můžeme nechat zobrazit jejich rovnice v obvyklém tvaru. Přímka je v Cabri 3D reprezentována dvojicí lineárních rovnic (tj. rovnicemi dvou rovin, jichž je průsečnicí). To se bohužel pro výuku základů analytické geometrie v prostoru, která je dnes založena na parametrických rovnicích přímky vycházejících z bodu a směrového vektoru, příliš nehodí. Přitom vygenerování parametrických rovnic přímky by patrně bylo technicky jednodušší, než u dvojice lineárních rovnic. Autoři přislíbili, že pro příští aktualizaci programu rovnici přímky přepracují.

Kromě „ručního“ přemísťování objektů lze v Cabri 3D naprogramovat i automatický pohyb objektů. Pro každý bod, který má volnost pohybu po úsečce nebo po kružnici, můžeme nastavit rychlost (s numerickou indikací) a smysl pohybu a pak pohyb celé soustavy (který může indukovat pohyby dalších odvozených objektů) spustit. To je podstatný pokrok ve srovnání s nepřesně determinovaným natahováním pružin, které automatizovaly pohyb v Cabri II.

Pohybující se objekty mohou zanechávat stopu. Oproti dvojrozměrnému případu v Cabri II je však využití stopy objektu pohybujícího se ve třech dimenzích méně vděčné, neboť trojrozměrná stopa je ve dvojrozměrné projekci zpravidla méně přehledná. Zřejmě z tohoto důvodu lze v Cabri 3D stopu pořizovat pouze pro omezený okruh objektů: body, úsečky, vektory, přímky a kružnice. Na rozdíl od Cabri II je zde stopa stabilním (i když ne dynamickým) objektem a ukládá se do souboru s ostatními objekty. Obdoba dynamické množiny známé z Cabri II zde k dispozici není. Významnou novinkou je však využití nástroje Stopa k hledání bodů, které anulují výrazy s geometrickými veličinami.

Historii konstrukce lze v Cabri 3D rekapitulovat dvěma způsoby. Jednotlivé kroky můžeme rušit a opět vracet do libovolné hloubky (na rozdíl od Cabri II) pomocí funkcí Zpět a Znovu, podobnou možnost nabízí i zvláštní okno pro přehrávání konstrukce. První způsob je podrobnější, týká se i manipulace s okny, přesouvání objektů v nákresně, změn stylu apod., druhý způsob sleduje jen vznik nových objektů. V obou případech můžeme z libovolného stádia konstrukce pokračovat jiným způsobem než původně. Kromě využití pro prezentaci postupu konstrukce daného obrázku lze tyto možnosti využít i pro odhalování a opravování chyb ve složitějších konstrukcích.

Z nástrojů, které se osvědčily v Cabri II, chybí v Cabri 3D kromě již zmíněných dynamických množin a podobných zobrazení (stejnolehlosti), dva, které citelně postrádám. Jednak testování vzájemné polohy objektů, které by se velmi hodilo vzhledem k deformacím situace promítáním, a pak možnost vytvářet makrokonstrukce.

Obrázky vytvořené v Cabri 3D lze tisknout přímo z programu. Dále je lze vkládat jako statické bitmapové ilustrace i do dokumentů standardních textových, grafických a prezentačních programů. Do dokumentů vytvořených v programech Microsoft Word a Microsoft PowerPoint je lze vkládat dokonce při zachování možnosti manipulovat s pohyblivými objekty a pozorovat naprogramované pohyby. K prohlížení stačí mít kromě Wordu, resp. PowerPointu nainstalován jen zásuvný modul (plug-in), jehož instalační program je volně ke stažení na [1], není tedy zapotřebí mít program Cabri 3D. Stejné možnosti jsou i pro prohlížení dynamických obrázků z Cabri 3D na webových stránkách.

Program se dodává v cca 20 jazykových variantách včetně češtiny a slovenštiny. V české variantě nebylo bohužel možno z technických důvodů rozlišovat mezi kruhem a kružnicí, koulí a kulovou plochou apod., kde to

nerozlišuje ani originální varianta. V komunikaci s programem proto místy dochází k odchylkám od terminologické normy, např. obsah kružnice, průsečnice koulí a pod. Na webu [1] je k dispozici cca 100 ukázkových příkladů, tutor a uživatelská příručka. Když jsem lokalizoval program a doprovodné materiály do češtiny, shledal jsem v originální uživatelské příručce řadu nedostatků a nakonec jsem raději sepsal vlastní příručku [2]. Připomínky k funkci programu, k počeštění i k příručce uvítám. Distributorem Cabri 3D pro Česko je Akermann Electronic [3]. Program byl akreditován ministerstvem jako vhodný pro školství.

Stereometrii a deskriptivní geometrii je dnes i přes proklamace o důležitosti prostorové představivosti věnováno na našich středních školách velmi málo času. V jednoduchých případech asi lépe než počítač poslouží jako názorná pomůcka několik špejlí a polystyrénových desek. Vzniká otázka, bude-li stát školám za to, aby si program kupovaly, není totiž levný. Cenová politika firmy Cabrilog se mi zdá poněkud asociální. Uvědomíme-li si celosvětové rozšíření Cabri II, které určitě znamenalo i úspěch komerční, mohly by snad dnes ceny jejich produktů být už vstřícnější. (Obzvlášť podivné mi připadá, že zákazníci, kteří zakoupili starší verzi a podpořili tím vlastně vývoj programu v rané fázi, musejí připlatit za aktualizaci na verzi Cabri 3D v2.) Cabri 3D je ovšem program lahůdkový a všichni počítačově geometričtí labužníci si ho určitě rádi dopřejí. Cabri II má několik mladších příbuzných programů, které sice neumějí úplně všechno to, co originál, ale pro potřeby planimetrie zejména na základních školách stačí, přičemž jsou podstatně levnější nebo zcela zdarma. U Cabri 3D o žádném podobném programu nevím a vzhledem k technické náročnosti realizace 3D geometrie ani neočekávám, že by levná napodobenina brzy vznikla. Určitou bariérou rozšíření na skromněji vybavené školy mohou být nároky, které 3D grafika klade na výkonnost procesoru i grafické karty.

Cabri 3D je program pro podporu výuky stereometrie a deskriptivní geometrie. Využití najde na středních a vysokých školách, kde se tyto předměty pěstují do větší hloubky. Je také užitečným nástrojem pro tvorbu tištěných a elektronických materiálů pro tyto discipliny.

Literatura

[1] www.cabri.com
[2] www.pf.jcu.cz/cabri
[3] www.akermann.cz