Logické programování

Michal Beneš

Motto: Pohovořte si se svým počítačem.

Abstrakt : Možná vám připadá zvláštní, proč se na matematickém soustředění bavit o nějakém programování. Ze zkušeností vím, že mnozí matematikové, resp. „mladí matematikové“, nevychází s počítači dobře. Bavit se o programování však neznamená nutně mluvit o počítačích. Na své přednášce vám zkusím představit počítač jako snad až příliš formalistického matematika. Rozhovor s ním nebude snadný, uvidíte však, že bude mnohem zábavnější, než poněkud nesrozumitelná komunikace pomocí zprzněnin anglických slov.

V tomto článku vás chci seznámit se základy jazyka *Prolog* jakožto nejrozšířenějšího (čti „Autor jiný nezná“) jazyka pro logické programování. Prolog se značně liší od tzv. procedurálních jazyků (C, C++, Pascal, Basic, Assembler, Fortran, Karel, ...). Zjednodušeně se dá říci, že narozdíl od těchto jazyků říkáme jen to, co chceme vyřešit a už se nestaráme o to, jak to udělat, tj. nezajímají nás „recepty“ typu: Vezmi tuhle proměnnou, přičti ji k téhle a výsledek ulož na támhleto místo v paměti.

Cvičení 1: *Stojí vám za to navštívit moji přednášku o Prologu ?*

Zdá se tedy, že vše je podezřele jednoduché. Copak stačí Prologu napsat „Vyřeš první úlohu třetí série letošního ročníku PraSete“ a on již vše udělá za nás? Kdepak! Prolog nám odpoví velmi elegantně v tom smyslu, že neví co to znamená vyřešit, co to znamená první a považte, nebude vědět ani, co to znamená PraSe. Za to, že se nestaráme o to, jak problém řešit, platíme tím, že jej musíme velmi přesně specifikovat (rozuměj nadefinovat všechny potřebné termíny).

Podívejme se na příklad, který vypráví skvělý pedagog na MFF UK Rudolf Kryl. Spolek klepen vyslal svoji agentku do malé komunity lidí, aby zjistila některé důležité informace (kdo, kdy, kde a s kým). Část zprávy agentky mohla vypadat takto :

```
muž( žibřid ).                % Byl tam jakýsi žibřid a byl to muž,
manželé( žibřid, kunhuta ).   % byla tam také kunhuta a ti dva byli manželé.
rodič( žibřid, agáta ).       % Žibřid byl rodičem agáty
rodič( žibřid, ivan ).        % a byl také rodičem ivana.
muž( ivan ).                  % Ivan byl muž.
```

Podotýkám, že se jedná o syntakticky správný prologovský program (odhlédneme-li od toho, že ve skutečnosti byste nesměli použít háčky a čárky). Znak `'%'` uvozuje komentář (Prolog ignoruje vše až do konce řádky). Důležitou funkci mají tečky, které označují konec „věty“ (v Prologu se nazývají *klauzule*). Tak, jako v běžné řeči, má zapomenutí tečky za následek nepochopení vašeho sdělení.

Uvědomme si, co uvedený text znamená matematicky. Agentka nám podala informaci o existenci nulárních relací[†] `žibřid`, `kunhuta`, `agáta`, `ivan`. Dále jsme se dozvěděli o unární relaci `muž` a tom, že `žibřid` a `ivan` této relaci vyhovují. Také jsme se dozvěděli o dvou binárních relacích `manželé` a `rodič`.

Pokud takovouto zprávu načteme[‡] do Prologu, můžeme začít klást otázky :

```
rodič( žibřid, agáta ).
rodič( agáta, žibřid ).
manželé( kunhuta, žibřid ).
```

Například první otázkou se ptáme „Je `žibřid` rodičem `agáty`?“. Nikoho nepřekvapí, že v prvním případě dostáváme odpověď `yes` a v druhém `no`. Možná, ale překvapí, že odpověď a třetí otázku je také `no`, přestože cítíme že Kunhuta a Žibřid manželé jsou. Kde se stala chyba?

Chyba se nestala nikde, není totiž tak docela pravda, že `rodič( žibřid, agáta ).` je otázka. Ve skutečnosti jsme Prologu řekli, aby se pokusil dokázat větu „`žibřid` je rodičem `agáty`“. Odpovědi `yes` a `no` tedy musíme chápat jako „Ano, takovou věc lze z daných údajů dokázat.“ a „Ne, takovou věc nelze z daných údajů dokázat.“ nemluvíme o pravdivosti, mluvíme o dokazatelnosti.

Cvičení 2: *Kde se s takovou situací setkáváme ve skutečném životě ?*

Agentka mohla také vyzkoumat mnohem složitější závislosti, k jejich popsání už potřebovala proměnné. Ty v Prologu poznáte podle toho, že začínají velkým písmenem[⋆].

```
syn( S, R ):-muž( S ), rodič( R, S ).
bratr( Kdo, Koho ):-syn( Kdo, X ), rodič( X, Koho ).
```

nebo-li `S` je synem `R`, pokud `S` je muž a zároveň `R` je rodičem `S`, `Kdo` je bratrem `Koho`, pokud `Kdo` je synem nějakého `X`, které je současně rodičem `Koho`.

Cvičení 3: *Co je na takové definici bratra zvláštního ?*

Proměnné lze také použít při zadávání složitějších dotazů :

```
rodič( žibřid, X ).
```

takto se ptáme, jaké musí být `X`, aby ze zadaných faktů bylo možné dokázat „`žibřid` je rodičem `X`“. Dostaneme odpověď `X=agáta`. Nyní Prolog čeká na naši reakci, stiskneme-li klávesu *Enter* dáváme najevo svoji spokojenost a můžeme zadávat další dotazy. Stiskneme-li klávesu středník, dáváme najevo, že bychom rádi dostali nějaké jiné řešení. V našem případě dostaneme po prvním zmáčknutí středníku ještě `X=ivan`, po dalším už jen suché konstatování `no` — pro žádné další `X` již `rodič( žibřid, X )` nelze dokázat.

Necháme nyní Spolek klepen zpracovávat jejich informace Ukažme si ještě, že relace v Prologovském programu nemusí udávat jen vztah mezi nulárními relacemi.

[†]Relace je od slova „vztah“. Například čísla $2, 2, 2$ jsou v relaci „býti rovna“, dvojice čísel $2, 3$ je v binární relaci „býti menší“, v této relaci však již není dvojice čísel $3, 2$. Žibřid z našeho příkladu je v unární relaci „býti muž“. Nulární relaci odpovídá jediný objekt (naopak každý matematický objekt lze považovat za nulární relaci).

[‡]Jmenuje-li soubor obsahující tuto zprávu `zprava.pl`, provedeme to „příkazem“ (Prolog příkazy nemá) „`consult(zprava).`“.

[⋆]Začínat mohou i znakem `'_'`, tím se však až na jednu vyjímku nebudeme zaobírat.

```
vodorovná( úsečka( bod(X1,Y), bod(X2,Y) ) ).
```

Vodorovné jsou ty úsečky, jejichž koncové body mají stejné y-ové souřadnice.

V posledním příkladě vůbec nezáleží na proměnných `X1` a `X2`, v takových případech místo nich použijeme takzvanou anonymní proměnnou, která se značí `'_'`. Anonymní proměnná může nahrazovat cokoliv a při jejím násobném výskytu v jedné formuli dokonce pokaždé něco jiného. Lepší by tedy bylo psát :

```
vodorovná( úsečka( bod(_,Y), bod(_,Y) ) ).
```

Zmíníme se o reprezentaci seznamů prvků v Prologu. Prázdný seznam[†] se značí `[]`. Seznam obsahující prvky `a`, `b`, `c` se značí `[a,b,c]`. Pokud `Ocas` je seznam, pak `[a,b,c|Ocas]` značí seznam obsahující prvky `a`, `b`, `c` následovaný prvky seznamu `Ocas`. Následující výrazy jsou tedy totožné :

```
[a,b,c] [a,b|[c]] [a|[b,c]] [a|[b|[c]]] [a,b,c|[]] [a,b|[c|[]]] [a|[b|[c|[]]]]
```

Jako ilustraci práce se seznamy uvedu relaci pro obracení seznamů :

```
obrácený( [], B, B ).
obrácený( [X|A], Sez, Výs ):-obrácený( A, [X|Sez], Výs ).
obrácený( X, Y ):-obrácený( X, [], Y ).
```

Opíráme se zde o ternální relaci `obrácený` (abychom ji odlišili od binární relace `obrácený` píšeme `obrácený/3`), která znamená asi toto: „můj třetí seznam vznikne tak, že vezmu první, obrátím ho a připojím za něj druhý“.

Cvičení 4: *Co uděláme, chceme-li otočit seznam* `[a,b,c]` *pomocí* `obrácený/2` *?*

Na závěr se zmíním o tom, jak Prolog pracuje s čísly. Původně byla v Prologu implementována jen přirozená čísla, nyní se však můžete často setkat i s interprety, které ovládají čísla desetinná. Výrazy zapisujeme standartním způsobem (například `(1+X)/2`). K porovnávání výrazů slouží relace `<`, `>`, `=:=`, `=\=` (poslední dvě znamenají „rovná se“, „nerovná se“). Pokud se v porovnávaných výrazech objeví proměnná, musí jí již být přiřazen[‡] nějaký výraz.

Relace `is` také testuje rovnost dvou výrazů, na levé straně však musí mít číslo nebo proměnnou, které ještě nebylo nic přiřazeno, nebo které bylo přiřazeno číslo, pro výraz na pravé straně platí stejné podmínky jako u `=:=`. Výhodou `is` proti `=:=` je, že výsledkem `X is V` je to, že výraz `V` je vypočten a jeho hodnota je přiřazena proměnné `X`.

Cvičení 5: *Napiš relaci* `qsort`*, která setřídí daný seznam čísel.*

[†]Pomocí prostředků, které zatím známe, bychom mohli seznam reprezentovat například takto: `nil` označíme prázdný seznam, k reprezentaci delších seznamů použijeme binární relaci tečka (`.`), seznam jehož prvním prvkem je `a` a je následovaný prvky seznamu `Seznam` označíme `.(a,Seznam)`. Výhody dále popsaného zápisu oceníme, již pokud je potřeba zapsat tříprvkový seznam, podle naší konvence by jeho zápis byl totiž `.( a, .( b, .( c, nil ) ) )`.

[‡]Tady není jasné, co to znamená „přiřazen“. Proměnná v Prologu má samozřejmě zcela jiný význam než proměnná v procedurálním jazyku. Vysvětlení toho, co zde myslím slovem „přiřazen“, bude na přednášce, tam však budu používat „vědečtější“ terminologii.

Odpovědi ke cvičením :

1. Ano!
2. V soudnictví.
3. Každý muž, který má nějakého rodiče, je podle takové definice svým bratrem.
4. Napíšeme `obrácený( [a,b,c], Sez ).` Prolog odpoví `Sez=[c,b,a]` po případném stisknutí středníku dostaneme už jen `no`, jinak už tento seznam obrátit nelze.
5. Viz ta nejlepší přednáška.