2.5.27 Teorie. *Axiomatická teorie* $\mathcal{T}$ v predikátové logice je dána volbou jazyka predikátové logiky a množinou sentencí T. O prvcích množiny T mluvíme jako o *axiomech* teorie $\mathcal{T}$. *Model teorie* $\mathcal{T}$ je každá interpretace, ve které jsou všechny axiomy z T pravdivé.

Příklad: Jazyk predikátové logiky se skládá z jednoho binárního predikátového symbolu R a axiomy jsou následující tři sentence:

$$\forall x\, R(x,x);\ \forall x\, \forall y (R(x,y) \Rightarrow R(y,x));$$

$$\forall x\, \forall y\, \forall z\, ((R(x,y) \wedge R(y,z)) \Rightarrow R(x,z)).$$

Pak modelem této teorie je každá neprázdná množina U, na které je predikát R interpretován jako binární relace, která je reflexivní, symetrická a tranzitivní; tudíž jedná se o množinu a na ní danou relaci ekvivalence R.

2.5.28 Teorie s rovností je teorie, kde kromě množiny predikátových symbolů Pred máme ještě binární predikátový symbol $=$, který má vlastnosti rovnost a vždy se interpretuje jako rovnost objektů „světa“, tj. universa U. (Poznamenejme, že v případě teorie s rovností už může být množina predikátových symbolů prázdná.)

2.5.29 Použití rezoluční metody v programovacím jazyce Prolog. Převzato z přednášky prof. Štěpánkové (PROLOG, PROgramování v LOGice.)

Program v Prologu je teorie v predikátové logice. Přesněji: *Logický program* je množina klauzulí s právě jedním pozitivním literálem. (Klauzulím, které obsahují právě jeden pozitivní literál, se také říká *Hornovy* klauzule.)

Logický program tvoří

- *fakta* — elementární pravdivá tvrzení vyjádřené jako jeden atom.
- *Pravidla – podmíněná tvrzení* „jestliže jsou splněny současně všechny podmínky tvořící tělo pravidla, pak platí i hlava p“ (hlava je vždy popsána jediným atomem).

Program se spouští pomocí dotazu.

2.5.30 Poznámka. V Prologu se pravidla většinou píší ve formě implikace, ale „naopak“, tj. nejprve se napíše hlava pravidla p, pak symbol $:-$ (který má význam implikace zprava do leva) a pak teprve předpoklady implikace.

2.5.31 Příklad. Jazyk predikátové logiky se skládá z

- $\mathsf{Pred} = \{\texttt{rodic}, \texttt{zena}, \texttt{muz}, \texttt{matka}\}$, kde `rodic` a `matka` jsou binární predikátové symboly a `zena` a `muz` jsou unární predikátové symboly.
- $\mathsf{Kons} = \{\texttt{boleslav}, \texttt{drahomira}, \texttt{ludmila}, \texttt{spytihnev}, \texttt{vaclav}, \texttt{vratislav}\}$.

Program:

- Fakta:

$$\texttt{rodic}(\texttt{ludmila}, \texttt{spytihnev}),\quad \texttt{rodic}(\texttt{ludmila}, \texttt{vratislav}),$$
$$\texttt{rodic}(\texttt{drahomira}, \texttt{vaclav}),\quad \texttt{rodic}(\texttt{drahomira}, \texttt{boleslav}),$$
$$\texttt{rodic}(\texttt{vratislav}, \texttt{vaclav}),\quad \texttt{rodic}(\texttt{vratislav}, \texttt{boleslav}).$$
$$\texttt{zena}(\texttt{ludmila}),\ \texttt{zena}(\texttt{drahomira}),\ \texttt{muz}(\texttt{vaclav}),\ \texttt{muz}(\texttt{boleslav}).$$

- Pravidla:

$$\texttt{matka}(x,y)) :- \forall x\, \forall y\, ((\texttt{zena}(x) \wedge \texttt{rodic}(x,y))$$

Toto pravidlo je ekvivalentní klauzuli

$$\forall x\, \forall y\, (\neg\texttt{zena}(x) \vee \neg\texttt{rodic}(x,y) \vee \texttt{matka}(x,y)).$$

Dotaz může být $\exists z\, \texttt{matka}(z, \texttt{vaclav})$?

K programu přidáme negaci dotazu, tedy fakt

$$\neg\exists z\, \texttt{matka}(z, \texttt{vaclav}) \models\!\models \forall z\, \neg\texttt{matka}(z, \texttt{vaclav})$$

a použijeme rezoluční metodu ke zjištění, zda program spolu s negací dotazu tvoří nesplnitelnou množinu (pak odpověď je ano), nebo splnitelnou množinu (pak odpověď je ne).

Není těžké zjistit, že

- Resolventa klauzulí

  $$\forall z\, \neg\texttt{matka}(z, \texttt{vaclav}) \text{ a } \forall x\, \forall y\, (\neg\texttt{zena}(x) \vee \neg\texttt{rodic}(x, y) \vee \texttt{matka}(x, y)))$$

  (druhá klauzule je pravidlo programu) je klauzule

  $$\forall x\, (\neg\texttt{zena}(x) \vee \neg\texttt{rodic}(x, \texttt{vaclav})).$$

- Rezolventa klauzulí

  $$\forall x\, (\neg\texttt{zena}(x) \vee \neg\texttt{rodic}(x, \texttt{vaclav})) \text{ a } \texttt{zena}(\texttt{drahomira})$$

  (druhá klauzule je fakt programu) je klauzule

  $$\neg\texttt{rodic}(\texttt{drahomira}, \texttt{vaclav}).$$

- Rezolventa klauzulí

  $$\neg\texttt{rodic}(\texttt{drahomira}, \texttt{vaclav}) \text{ a } \texttt{rodic}(\texttt{drahomira}, \texttt{vaclav})$$

  (druhá klauzule je fakt programu) je prázdná klauzule **F**.

Zjistili jsme, že program spolu s negaci dotazu tvoří nesplnitelnou množinu klauzulí, a proto odpověď Prologu je **ano**.

2.6 Přirozená dedukce v predikátové logice

Upozorňujeme čtenáře, že zde uvádíme jen velmi „letmé“ přiblížení přirozené dedukce. Jedná se o rozšíření přirozené dedukce z výrokové logiky o odvozovací pravidla pro kvantifikátory $\forall$ a $\exists$.

Pravidla pro kvantifikátory jsou v podstatě následující:

E-pravidlo pro $\forall$:
$$\begin{array}{|l} \forall x\, \varphi(x) \\ \hline \varphi(t) \end{array}$$

I-pravidlo pro $\forall$:
$$\begin{array}{|l} \varphi(x) \\ \hline \forall x\, \varphi(x) \end{array}$$

I-pravidlo pro $\exists$:
$$\begin{array}{|l} \varphi(t) \\ \hline \exists x\, \varphi(x) \end{array}$$

E-pravidlo pro $\exists$:
$$\begin{array}{|l} \exists x\, \varphi(x) \\ \quad\begin{array}{|l} \varphi(y) \\ \hline \psi \end{array} \\ \hline \psi \end{array}$$

Aby tato čtyři odvozovací pravidla byla opravdu „správně“, tj. aby závěry pravidel byly sémantickým důsledkem předpokladů pravidla, musí být ještě splněno:

- V E-pravidle pro $\forall$ a I-pravidle pro $\exists$ musí být term t „volný pro x“, tj. musí obsahovat pouze proměnné, které jsou volné ve formuli $\phi[x := t]$, kde jsme za proměnnou x dosadili term t.
- V I-pravidle pro $\forall$ a E-pravidle pro $\exists$ nesmí být proměnná x volná v žádném aktivním předpokladu.

I v predikátové logice definujeme odvození a logický důsledek stejně jako v 1.9.2 a 1.9.3. A co je důležité, i v predikátové logice, podobně jako ve výrokové logice, platí věta o úplnosti 1.9.4.

2.6.1 Odvození. Posloupnost sentencí φ_1, φ_2, ..., φ_n se nazývá *odvození z předpokladů S* právě tehdy, když

- každá sentence φ_i je buď předpoklad (tj. $\varphi_i \in S$), nebo je pomocný předpoklad, nebo vznikla z předcházejících sentencí pomocí některého odvozovacího pravidla;
- všechny pomocné předpoklady jsou již pasivní.

2.6.2 Logický důsledek. Sentence φ je *logický důsledek množiny předpokladů S*, též *logicky vyplývá z S*, právě tehdy, když existuje odvození z S takové, že $\varphi_n = \varphi$. Zapisujeme $S \vdash \varphi$.

2.6.3 Věta o úplnosti. Pro každou množinu sentencí S a sentenci φ platí

$$S \vdash \varphi \quad \text{právě tehdy, když} \quad S \models \varphi.$$