Constraint Programmierung

Prof. Dr. Thom Frühwirth, Marc Meister

WS4 – Blatt 1
Besprechung am 2. Nov., 14 Uhr, O27/2201
Vorlesungsseite `http://www.informatik.uni-ulm.de/pm/veranstaltungen/ws2004/cp/`

Aufgabe 1. Geben Sie für folgende Paare jeweils einen allgemeinsten und einen nicht allgemeinsten Unifikator an. Wo das nicht möglich ist, nennen Sie den Grund für die Nichtunifizierbarkeit.

(a)	$p(a, X, c, Y, Z)$	und	$p(Y, X, c, a, W)$
(b)	$p(Y, g(Y, f(Y)), g(X, a))$	und	$p(f(U), V, g(h(U, V, W), U))$
(c)	$p(g(X), h(a, X), h(Y, Y))$	und	$p(U, h(a, g(V)), h(V, g(U)))$
(d)	$p(h(f(X), a), X, h(f(f(X)), f(f(X))))$	und	$p(h(V, a), U, h(W, f(V)))$

Aufgabe 2. Geben Sie für jede der folgenden Formeln an, welche der Eigenschaften Allgemeingültigkeit, Erfüllbarkeit, Falsifizierbarkeit, Unerfüllbarkeit sie hat:

a) $\neg(p(a) \rightarrow \exists X\, p(X))$
b) $(\exists X\, p(X)) \rightarrow p(a)$
c) $(\exists Y \forall X\, p(X, Y)) \rightarrow (\forall X \exists Y\, p(X, Y))$

Aufgabe 3. Beweisen oder widerlegen Sie folgende Behauptungen:

a) $\{\varphi 1, \varphi 2\} \models \psi$ gdw $\varphi 1 \models \psi$ und $\varphi 2 \models \psi$.
b) Für beliebige Sätze φ, ψ gilt $\varphi \models \psi$ genau dann wenn $\varphi \not\models \neg\psi$.
c) Wenn φ ein Satz ist, der kein Modell hat, dann gilt $\varphi \models \psi$ für jeden beliebigen Satz ψ. Geben Sie Beispiele für Sätze φ mit dieser Eigenschaft an.
d) Für Mengen von Sätzen $\mathcal{M} \subseteq \mathcal{M}'$ und Sätze φ gilt $\mathcal{M} \models \varphi$ genau dann wenn $\mathcal{M}' \models \varphi$.

Aufgabe 4. Let ϕ and ψ be formulae. Then $\phi \models \psi$ if and only if $\models \phi \rightarrow \psi$.
This claim contains three kinds of "implications" on different language levels. Identify them.

Aufgabe 5. Machen Sie sich mit SICStus Prolog, das im Linux-Pool installiert ist, vertraut:

- Setzen Sie die passende Umgebung mit `option sicstus`.
- Auf der Seite `www.sics.se/sicstus/` (oder in der lokalen Dokumentation) finden Sie Infos zur Sprache, lesen Sie das Kapitel "How to Run Prolog".
- Installieren Sie den Emacs-Support für SICStus.
- Schreiben Sie ein "Hello world!" Programm in SICStus Prolog, z.B.
  `start :- write('Hello world!').`
  Compilieren Sie das Quellprogramm in Emacs und rufen Sie es mit `start.` auf