Prof. Dr. Stefan Brass
Institut für Informatik
Universität Halle

15.05.2006

Deduktive Datenbanken und Logische Programmierung — Blatt 5: Ein einfacher Typprüfer —

Aufgabe 7 — 5 Punkte

Schreiben Sie einen ganz einfachen Typprüfer für Prolog/Datalog-Regeln in Prolog. Zur Vereinfachung sei angenommen, daß die Prädikate der zu prüfenden Regeln nur Integers und Atome als Argumente haben, keine strukturierten Terme. Die Prädikate, die in diesen Regeln verwendet werden dürfen, seien im Typprüfer selbst deklariert in der folgenden Form:

```
pred(komponist(int, atom, atom, int, int)).
pred(stueck(int, int, atom, atom, atom)).
pred(>(int, int)).
pred(h1a(int, int)).
...
```

D.h. die verwendbaren Prädikate sind (vorläufig) in den Typprüfer fest eingebaut. Man soll dann durch Aufruf von

```
read(X), check(X).
```

jeweils eine Regel, ein Fakt (Literal), oder eine Anfrage (Konjunktion von Literalen) eingeben können, und auf Typrichtigkeit prüfen. Die syntaktische Richtigkeit können Sie voraussetzen. Das eingebaute Prädikat `read` liefert die eingegebene Regel auch bereits als Term (Baumstruktur). Beachten Sie, daß wenn Sie `:-` und `,` als normale Funktoren verwenden wollen, Sie diese Operatoren in Anführungszeichen einschließen müssen, z.B. `':-'`. Es reicht aus, wenn Ihr Prädikat `check` fehlschlägt, falls die Eingabe einen Typfehler enthält (d.h. man in diesem Fall die Antwort "no" bekommt und sonst "yes"). Falls Sie wollen, können Sie natürlich auch bessere Fehlermeldungen vorsehen (Ausgabe mit `write`, Zeilenumbruch mit `nl`, erzwungenes Fehlschlagen mit `fail`: Gute Fehlermeldungen verkomplizieren die Aufgabe aber deutlich!).

Zum Beispiel sollte die Eingabe

```
h1a(X, Y) :- komponist(_, 'Wolfgang Amadeus', 'Mozart', X, Y).
```

als korrekt gewertet werden, dagegen die folgenden alle nicht:

- `h1a(X, X) :- komponist(X, Y, Y, 'Wolfgang Amadeus', 'Mozart').`
  (Atom-Konstanten an Integer-Argumentposition)
- `h1a(X1, X2) :- komponist(Nr, X1, X2, Y1, Y2).`
  (`X1` und `X2` müssen nach dem Rumpf-Literal Atome sein, aber nach dem Kopf-Literal Integers)
- `h1a(X, X) :- komponist(Nr, 'Wolfgang Amadeus', 'Mozart', X).`
  (Es gibt kein Prädikat `komponist` mit nur vier Argumenten.)

Die Eingabe

```
h1a(X1, X2) :- komponist(_, 'Wolfgang Amadeus', 'Mozart', Y1, Y2).
```

soll dagegen noch als korrekt gelten, obwohl die Variablen `X1` und `X2` im Rumpf nicht gebunden werden: Dies ist kein Typfehler, sondern eine Verletzung der Bereichsbeschränkung (je nach deklarierten Bindungsmustern, kommt in der Vorlesung noch).

Sie können diese Aufgabe z.B. in drei Schritten lösen:

- Überführen Sie die Eingabe in eine Liste von Literalen.
- Ersetzen Sie die Konstanten in diesen Literalen durch die Typbezeichner `atom` und `int`. Den Test auf die Typen können Sie mit den Prädikaten `atom/1`, `integer/1`, `var/1` durchführen, z.B. wäre `integer(2)` wahr, und integer(X) falsch (wenn `X` noch nicht an eine Zahl gebunden ist). Sie müssen für diese Aufgabe auch Literale in Prädikat und Argumentliste zerlegen, bzw. umgekehrt wieder zusammensetzen. Das geht mit dem Prädikat `=..`, das als Infix-Operator deklariert ist. Z.B. gilt `p(a,b) =.. [p,a,b]` (das Prädikat ist das erste Element der Liste). Nach dem Aufruf `X =.. [p,a,b]` ist `X = p(a,b)`, und nach dem Aufruf `p(a,b) =.. Y` ist `Y = [p,a,b]`.
- Nun müssen Sie nur noch jedes Literal Ihrer Liste mit den gegebenen `pred`-Fakten unifizieren (dafür ist wichtig, daß Sie im zweiten Schritt die Variablen aus der Eingabe intakt gelassen haben, sonst würde die inkonsistente Verwendung der gleichen Variablen mit unterschiedlichen Typen nicht erkannt).

Abgabetermin:

Bitte geben Sie Ihre Lösung bis zum Montag, den **12. Juni 2006**, ab.