Übungen 5: Operatoren, Daten- und Kontrollfluss

Programmiertechniken in der Computerlinguistik I · Wintersemester 2001/2002

Um keine Zeit mit Tippen und Tippfehlerkorrektur zu verschwenden, findest du die Programmtexte dieser Übung unter http://www.ifi.unizh.ch/cl/siclemat/lehre/ws0102/pcl1/uebung5.txt

1. Anfragen

Sind folgende Anfragen beweisbar oder nicht? Wenn ja, mit welcher Variablenbindung? Überlege zuerst und überprüfe deine Resultate am Interpreter – der Tracing-Modus kann dir beim Verstehen helfen.

```
a) ?- term(a,B) = term(b,b); a = B.
b) ?- \+ term(A,b) = term(a,c).
c) ?- (\+ \+ term(X) = term(a)), X = b.
d) ?- ','(=(B,b),=(a,A)).
e) ?- ( A = b, !, fail ; C = c).
```

2. Once, Twice …

Wir haben das Prädikat once/1 kennengelernt, das als Argument ein Prädikat nimmt und dieses höchstens einmal gelingen lässt. Manchmal wäre ein Prädikat nützlich, das höchstens zwei Lösungen berechnet. Alfred P. hat sich das ebenfalls gedacht und nach kurzem Überlegen folgendes Prädikat twice/1 niedergeschrieben:

```
twice(Goal) :-
   once(Goal).
twice(Goal) :-
   once(Goal).
```

Wie findest du diese Idee? Überlege zuerst und teste twice/1 danach am Prolog-Interpreter. Zum Beispiel anhand dieses Prgrämmchens:

```
person(hans).
person(gerda).
person(frieda).

?- twice(person(X)).
```

3. Vom Stutzen und Verzweigen

Wieviele Lösungen ergibt die Anfrage ?- p(X). Überlege zuerst, zeichne einen Mini-Beweisbaum und überprüfe dann deine Einschätzung am Prolog-Interpreter.

```
p(p1).
p(p2) :- q(_), !, r(_).
p(p3).

q(q1).
q(q2).

r(r1).
r(r2).
```

4. Sentence Frame Grammar – Satzmuster

Mit Hilfe von *Sentence Frame Grammar (Satzmuster-Grammatik)* kann man Sätze mit gleichartiger syntaktischer Struktur zusammenfassen. Den Satz "Peter likes Mary" könnte man mit dem Satzmuster "Pn Vt Pn" beschreiben, wobei Pn für Eigennamen und Vt für transitives Verb steht. In Prolog kann man das leicht implementieren, indem man sich folgende Regel vorstellt:
"Die Wörter W1, W2, W3 bilden ein Satz des Englischen, falls W1 und W3 Eigennamen sind und W2 ein transitives Verb."

a) Implementiere das Satzmuster "Pn Vt Pn" als Klausel des Prädikats satz/3. Für die syntaktischen Kategorien wie "Pn" oder "Vt" verwendest du am besten einfache Fakten wie `pn(peter).` oder `vt(likes).` Diese Fakten bilden das Lexikon (Wortbestand). Teste dein satz/3 mit der Anfrage ?- satz(W1, W2, W3). und mache manuelles Backtracking.

b) Implementiere die 2 weiteren Satzmuster "Det N Vt Pro" und "Pro Vt Det N ". Dazu gehört mindestens folgendes Lexikon:

Wortart	**Wörter**
Det (*determiner*, Artikel)	the, a
Vt (*transitive verb*, transitives Verb)	like, likes
Pro (*pronoun*, Pronomen)	he, they, him, them
N (*noun*, Nomen)	birds, king

c) Schreibe ein Prädikat alle_saetze/0, das durch einen Failure-Driven-Loop alle möglichen Sätze auf den Bildschirm herausschreibt. (Verwende write/1 zum Herausschreiben von Termen und nl/0 zum Erzeugen von Zeilenwechseln.)

d) Damit nur noch grammatische Sätze entstehen, müssen zwischen den Wörtern bestimmte Regeln und Übereinstimmungen bestehen:

- gleicher Numerus für Subjekt und Verb
- gleicher Numerus für Artikel und Nomen
- Subjekt steht im Nominativ
- Objekt steht im Akkusativ

Für das Satzmuster "Pro Vt Det N" ergibt sich folgende Regel:
"Die Wörter W1, W2, W3, W4 bilden einen englischen Satz, falls W1 ein Pronomen im Nominativ, W2 ein transitives Verb, W3 ein Artikel und W4 ein Nomen ist, wobei sowohl Pronomen und Verb wie Artikel und Nomen im Numerus übereinstimmen müssen."

Damit dieses Satzmuster implementiert werden kann, braucht es Numerus- und Kasusinformation im Lexikon. Erweitere das Lexikon entsprechend und ergänze die beiden Satzmuster so, dass nur noch syntaktisch korrekte englische Sätze entstehen. Überprüfe mit alle_satze/0.