UNIVERSITÄT OSNABRÜCK

Prof. Dr. Joachim Hertzberg, Andreas Nüchter

6. Übungsblatt zur Künstlichen Intelligenz

Wintersemester 2004/2005

Aufgabe 6.1 (Prädikatenlogik) Betrachten Sie die folgenden Sätze:

- John mag alles Essbare.
- Äpfel sind essbar.
- Wurst ist essbar.
- Alles was jemand isst und ihn nicht tötet ist essbar.
- Jeder der durch etwas getötet wird, lebt nicht mehr.
- Bill isst Erdnüsse und lebt noch.
- Susanne isst all das, was Bill isst.

1. Übersetzen Sie diese Sätze in Formeln der Prädikatenlogik.
2. Überführen Sie die Formeln in Klauselform.
3. Beweisen Sie durch Resolution, dass John Ernüsse mag.
4. Wie könnte man Resolution benutzen, um die Frage „Was ist Susanne?“ zu beantworten?

Aufgabe 6.2 (Einführung in Prolog) Programmieren Sie in Prolog die folgenden Prädikate und probieren Sie sie anhand der Wissensbasis `griechischeGoetter.pl` aus. Den File `griechischeGoetter.pl` finden Sie im StudIP.

1. `vater_von/2, mutter_von/2, eltern_von/3, sohn_von/2, tochter_von/2`,
2. `geschwister/2, bruder_von/2, schwester_von/2, onkel_von/2, tante_von/2`,
3. `grossvater_von/2, enkel_von/2, vorfahr/2, nachfahr/2`.

Aufgabe 6.3 (Unifikation) Welche Ausgabe produziert der (Prolog-)Unifikationsalgorithmus für die folgenden Ausdrücke:

1. $p(a)$ und $p(b)$,
2. $p(f(a,b,c))$ und $p(g(a,b,c))$,
3. $p(f(X,Y))$ und $p(f(X))$,

4. $p(f(b, X, b, Y))$ und $p(f(X, b, Y, Z))$,
5. $p(f(g(X)), Y, f(g(a)))$ und $p(f(g(a)), f(Z), f(Z))$.

Aufgabe 6.4 (Listen in Prolog) Eine Liste ist eine geordnete Folge von Elementen beliebiger Länge. Listen werden in eckigen Klammern geschrieben und die Elemente durch Komma getrennt.

Beispiele:

- `[1,2,3,4,5,6]`
- `[Mutter,Vater,Tochter,Sohn]`
- `[[1,1],[2,4],[3,9],[4,16]]`

Das dritte Beispiel zeigt, dass Listen auch geschachtelt werden können, um etwa eine Wertetabelle dazustellen.

Formal ist eine Liste eine rekursive Datenstruktur. Sie besteht entweder aus der leeren Liste `[]` oder einem Kopfelement und einer Restliste `[K|R]`.

Mit Hilfe des Listenoperators | kann man eine bestehende Liste in Kopfelement und Restliste aufteilen oder aus einem Kopfelement und einer Liste eine neue Liste erzeugen, also `[Element|Restliste]`.

Beispiele:

- `[1|[2,3]]` ist die Liste mit dem Kopfelement `1` und der Restliste `[2,3]`. Sie entspricht der Liste `[1,2,3]`.
- `?- Y = [b,c,d], X = [a|Y]` macht aus der 3-elementigen Liste `Y` die 4-elementige Liste `[a,b,c,d]`
- `?- [X|Y] = [a,b,c,d]` zerlegt die Liste `[a,b,c,d]` in den Kopf `X=a` und die Restliste `Y = [b,c,d]`.

Bemerkungen:

- Der Zugriff auf eine Liste ist nur über des erste Element, also den Listenkopf möglich. Soll auf andere Elemente zugegriffen werden, so müssen entsprechende Prädikate definiert werden.
- Listen können beliebige, gemischte Datenobjekte enthalten, z.B., `[name('Max','Meier'),hat,2,'Soehne',und,2,'Töchter']`.

Aufgabe: Schreiben Sie Prolog Klauseln für die folgenden Aufgaben.

1. `mydelete(X,L,R)` ist wahr, falls `R` die Liste ist, die aus Liste `L` durch das Löschen aller Vorkommen von `X` entsteht. Die Reihenfolge der restlichen Listenelemente bleibt unverändert.
2. `mypermutation(P,L)` ist wahr, falls `P` eine Permutation der Liste `L` ist.
3. `myreverse(R,L)` ist wahr, falls `R` genau die Elemente der Liste `L` in umgekehrter Reihenfolge enthält.
4. `sorted(L)` ist wahr, falls die Liste `L` aufsteigend sortiert ist.
5. `mysorted_list(S,L)` ist wahr, falls `S` eine sortierte Liste ist, welche die und nur die Elemente von `L` enthält. Hinweis: Sie können zum Beispiel Permutationssortierverfahren benutzen.
6. `length(L,N)` ist wahr, falls `N` die Länge der Liste `L` ist.