Prof. Dr. Stefan Brass
Institut für Informatik
MLU Halle-Wittenberg

06.05.2022

Logische Programmierung & deduktive Datenbanken — Übungsblatt 4 (Listen) —

Ihre Lösungen zu den Hausaufgaben a) bis d) geben Sie bitte über die Übungs-Plattform in StudIP ab. Einsendeschluss ist der 12. Mai, 10^{00} (die Aufgaben werden anschließend in der Übung besprochen).

Hausaufgaben

a) Definieren Sie ein Prädikat `ablist(L)`, das wahr ist, wenn `L` eine Liste nur aus den Elementen `a` und `b` ist. Z.B. soll `ablist([a,a,b,a])` wahr sein, aber `ablist([a,b,c,a])` falsch sein. Es ist nur gefordert, dass keine anderen Elemente außer `a` und `b` vorkommen, nicht, dass diese beiden enthalten sein müssen. Z.B. soll `ablist([])` wahr sein.

b) Schreiben Sie ein Prädikat `sum(L, S)`, das die Zahlen in einer Liste `L` aufsummiert. Z.B. soll der Aufruf `sum([1,2,3], S)` die Lösung `S = 6` liefern. Mit `number(E)` können Sie testen, ob ein Element `E` der Liste eine Zahl ist. Ihr Prädikat soll fehlschlagen (false liefern), wenn ein Element der Liste keine Zahl ist.

c) Schreiben Sie ein Prädikat `insert(E, LIn, LOut)`, das ein Element `E` in die sortierte Liste `LIn` an die richtige Stelle einfügt. Z.B. soll `insert(3,[1,4,5],X)` das Ergebnis `X = [1,3,4,5]` liefern. Sie können davon ausgehen, dass `LIn` nur Zahlen enthält und `E` eine Zahl ist. Falls `E` schon in `LIn` vorkommt, soll es nicht nochmals eingefügt werden.

d) Schreiben Sie ein Prädikat `mysort(LIn, LOut)`, das die Liste `LIn` sortiert, indem es die Elemente mittels `insert` aus Aufgabe c) nacheinander in eine anfangs leere Liste einfügt. Duplikate werden dabei eliminiert.

Zur Wiederholung

e) Was würden Sie in einer mündlichen Prüfung auf die folgenden Fragen zur Prolog-Syntax und zu Listen antworten?

- Was wird `write(+(1,2))` ausgeben?
- Was wird `display(1+2)` ausgeben?
- Wird die Anfrage `1+1 = 2` mit `true` oder `false` beantwortet?

- Mit welchem Funktionssymbol und welcher Konstanten werden Listen in Prolog intern repräsentiert?
- Geben Sie drei verschiedene Schreibweisen für die Liste "`[1,2,3]`" in Prolog an.
- In welchen Prolog-Term wird Notation `[F|R]` intern übersetzt? Wofür stehen `F` und `R` hier?
- Wie wird das Prädikat `append` in Prolog definiert? Geben Sie die Regeln an.
- Funktioniert `append` auch zum Aufspalten einer Liste?
- Definieren Sie ein Prädikat `member(X, L)` zum Test, ob `X` ein Element der Liste `L` ist.
- Definieren Sie ein Prädikat `length(L, N)`, das die Länge `N` einer Liste `L` berechnet.
- Was muss man in Prolog beachten, damit rekursiv definierte Prädikate terminieren?
- Wie sehen Regeln in Prolog aus? Definieren Sie die Begriffe "Regel", "Kopf einer Regel", "Rumpf einer Regel", "Kopfliteral einer Regel", "Rumpfliteral einer Regel", "Regel über ein Prädikat p".

f) Was würden Sie in einer mündlichen Prüfung auf die folgenden Fragen zur Unifikation antworten?

- Definieren Sie den Begriff "Substitution".
- Was ist das Ergebnis der Anwendung der Substitution $\{X/Y,\ Y/a\}$ angewendet auf $p(X, Y, Z, b)$?
- Definieren Sie den Begriff "Unifikator" und "Allgemeinster Unifikator" ("most general unifier", mgu).
- Ist der allgemeinste Unifikator zweier Literale immer eindeutig bestimmt?
- Was ist ein allgemeinster Unifikator von $p(X, a)$ und $p(Y, Z)$? Was wäre ein Beispiel für einen nicht allgemeinsten Unifikator?
- Berechnen Sie einen allgemeinsten Unifikator von $p(X, X)$ und $p(f(Y), f(a))$.
- Wie spielen Unifikatoren für Prolog eine Rolle? Wo werden sie gebraucht? Wie kann man sie mit Prolog berechnen?
- Was unterscheidet die Unifikation von Zuweisungen in klassischen Programmiersprachen?
- Was ist der "Occur Check"? Warum ist er für die Unifikation wichtig? Welches Problem gibt es? Warum lassen viele Prolog-Systeme ihn weg? Gibt es noch alternative Lösungen?
- Wie kann man testen, ob ein Prolog-System den Occur-Check verwendet? Geben Sie ein einfaches Beispiel-Programm an.

Zum Selbstudium

g) Schauen Sie sich die folgenden Webseiten an. Sie brauchen für diese Aufgabe nichts abzugeben. Ziel ist, dass Sie einen Eindruck davon gewinnen, was es im Internet zum Thema dieser Vorlesung gibt, und dabei für sich nützliche Quellen entdecken.

- Tutorial zu Prolog von John R. Fisher "`prolog :- tutorial`":

  [https://www.cpp.edu/~jrfisher/www/prolog_tutorial/]

  [https://www.cpp.edu/~jrfisher/www/prolog_tutorial/pt_framer.html]

- Eine Online-Version des Buches "Adventure in Prolog" von Dennis Merritt gibt es hier:

  [http://www.amzi.com/AdventureInProlog/apreface.php]

  Das Buch ist 1990 im Springer-Verlag erschienen. Es ist ein Prolog-Tutorial, das ein Textadventure-Spiel als durchgehendes Beispiel verwendet. Die Folien zu einem Prolog-Kurs von Dennis Merritt finden Sie hier:

  [https://github.com/AmziLS/prolog_course/blob/master/prolog_course.pptx]

- Es gibt dort auch ein ehemals kommerzielles Prolog, das jetzt Open Source ist:

  [http://www.amzi.com/index.php]

  Videos zur Einbindung von Amzi-Prolog in die Eclipse-Entwicklungsumgebung finden Sie hier:

  [http://www.amzi.com/videos/]

- Ein Eclipse-Plugin für SWI-Prolog (PDT 3.1 von der Uni Bonn) gibt es hier:

  [https://sewiki.iai.uni-bonn.de/research/pdt/docs/start]

  Die Installation ist in diesem Video beschrieben:

  [https://www.youtube.com/watch?v=5LlsH9MxnWs]

- Wenn Sie Video-Tutorials mögen, könnten Sie z.B. das folgende Tutorial zur Listen-Notation von "EducationAboutStuff" probieren (etwas langatmig):

  [https://www.youtube.com/watch?v=Jrf6CKc2c0k]

  Hier ist ein anderes Tutorial (von "mycurlycode"):

  [https://www.youtube.com/watch?v=WKoM4L74XzY]

  Das folgende Video (von "bSimple") enthält eine Lösung zu Aufgabe a):

  [https://www.youtube.com/watch?v=PUyccUj405k]