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

16.04.2022

Logische Programmierung & deduktive Datenbanken — Übungsblatt 1 (Einführung) —

Ihre Lösungen zu den Hausaufgaben a) bis c) geben Sie bitte über die Übungs-Plattform in StudIP ab. Einsendeschluss ist der 21. April, 10^{00} (die Aufgaben werden anschließend in der Übung besprochen). Aufgabe d) wird dort auch besprochen: Sie sollten vorher darüber nachdenken, müssen aber nichts abgeben.

Hausaufgaben

Es sei die Beispiel-Datenbank aus der Einführungs-Vorlesung betrachtet. Damit es für die Prolog-Fakten besser passt, wurden die Tabellennamen in die Singular-Form in Kleinschreibung überführt. Entsprechend wurden auch Aufgabentypen und Aufgabenthemen als normale Prolog Konstanten („Atome“) geschrieben. Sie finden eine Prolog-Datei mit den entsprechenden Fakten unter folgender Web-Adresse:

[https://www.informatik.uni-halle.de/~brass/lp22/prolog/db.pl]

- `student`: enthält die Daten von Studierenden der Vorlesung.
  - `SID`: „Studierenden-ID“ (eindeutige Nummer).
  - `VORNAME`, `NACHNAME`: Vor- und Nachname.
  - `EMAIL`: Email-Adresse (Der Nullwert wird als leeres Atom `\` dargestellt).

student			
SID	VORNAME	NACHNAME	EMAIL
101	Lisa	Weiss	···
102	Michael	Grau	NULL
103	Daniel	Sommer	···
104	Iris	Winter	···

- `aufgabe`: Beschreibt Aufgaben mit ihren Daten.
  - `ATYP`: Typ/Kategorie der Aufgabe.
    Z.B. `h`: Hausaufgabe, `z`: Zwischenklausur, `e`: Endklausur.
  - `ANR`: Aufgabennummer (eindeutig nur innerhalb des Typs).
  - `THEMA`: Thema der Aufgabe.
  - `MAXPT`: Maximale/volle Punktzahl der Aufgabe.

aufgabe			
ATYP	ANR	THEMA	MAXPT
h	1	ER	10
h	2	SQL	10
z	1	SQL	14

- `bewertung`: Abgegebene Lösungen zu einer Aufgabe (mit der erreichten Punktzahl).
  - `SID`: Student, der die Lösung abgegeben hat.
    Dies referenziert eine Zeile in `student`, d.h. es ist garantiert, dass es ein `student`-Fakt mit der gleichen `SID` gibt (Fremdschlüssel).
  - `ATYP`, `ANR`: Identifikation der Aufgabe.
    Zusammen identifiziert die beiden Werte ein Fakt des Prädikats `aufgabe`.
  - `PUNKTE`: Punkte, die der Student/die Studentin für die Lösung bekommen hat.
  - Falls es keinen Eintrag für einen Studenten und eine Aufgabe gibt, wurde die Aufgabe nicht abgegeben.

bewertung			
SID	ATYP	ANR	PUNKTE
101	H	1	10
101	H	2	8
101	Z	1	12
102	H	1	9
102	H	2	9
102	Z	1	10
103	H	1	5
103	Z	1	7

a) Definieren Sie ein Prädikat

```
lisas_ha(ANR, PUNKTE, MAXPT),
```

das die Hausaufgabenpunkte der Studentin „Lisa Weiss“ enthält.

Die Namen der Argumente dienen nur zur Erläuterung der Bedeutung. Sie können selbstverständlich beliebige Variablennamen verwenden. Da Variablennamen nur innerhalb einer Regel bekannt sind, hätte der Aufrufer des Prädikates ohnehin keine Möglichkeit, herauszufinden, wie die Argumente in Ihrer Regel heißen.

b) Nehmen wir an, wir wüßten, dass Lisa Weiss und Daniel Sommer die einzigen Informatiker unter den Studierenden sind. Definieren Sie ein Prädikat

```
inf_ha(Nachname, ANr, Punkte),
```

um die Hausaufgaben-Punkte dieser Studierenden im Blick zu behalten.

Sie dürfen selbstverständlich auch beliebige Hilfsprädikate einführen. Damit können Sie Codeduplizierung vermeiden und Ihr Programm leichter änderbar machen.

c) Definieren Sie ein Prädikat

`gut(VORNAME, NACHNAME)`,

das Studierende enthält, die sowohl in Hausaufgabe 1, als auch in Hausaufgabe 2 mindestens 8 Punkte bekommen haben.

Den numerischen Vergleich können Sie in Prolog als `P >= 8` schreiben. Dazu muss die Variable `P` weiter links im Regelrumpf schon einmal vorgekommen sein, und dabei an einen Wert gebunden sein. D.h. dieses eingebaute Prädikat kann zwei gegebene Zahlen auf die $\geq$-Beziehung testen, aber es kann natürlich nicht die unendlich vielen Zahlen aufzählen, die größergleich 8 sind. Prolog arbeitet die gegebenen Bedingungen im Regelrumpf immer von links nach rechts ab.

Beim „Constraint Logic Programming“ können Bedingungen wie solche arithmetischen Vergleiche dagegen verschoben werden, wenn sie noch nicht auswertbar sind, und es können auch Folgerungen gezogen werden, ohne dass Werte für alle Variablen bekannt sind.

Zur Wiederholung

d) Was würden Sie in einer mündlichen Prüfung auf die folgenden Fragen antworten?

- Was bedeutet eigentlich deklarative Programmierung? Warum ist SQL eine deklarative Sprache?
- Welche Vorteile hat deklarative Programmierung? (Sie könnten das z.B. anhand von SQL-Anfragen erläutern, die Sie mit entsprechenden Java-Programmen vergleichen. In einigen Jahren wurde der Vergleich in der Einführungs-Vorlesung zu Datenbanken praktisch ausprobiert.)
- Nennen Sie mindestens ein Prolog-System.
- Welche logischen Operatoren können in Prolog-Regeln verwendet werden (soweit bisher an Beispielen in der Vorlesung gezeigt)? Wie sind Regeln aufgebaut?
- Wie kann man den Inhalt einer relationalen Datenbank in Prolog darstellen?
- Wie unterscheiden sich Variablen und Konstanten in Prolog syntaktisch?
- Wann kann man die Anführungszeichen `'...'` weglassen?
- Wie sieht die anonyme Variable aus? Was unterscheidet sie von normalen Variablen? Welchen Schutz gibt es in Prolog gegen Tippfehler in Variablen?
- Wie kann man in Prolog eine Datei mit Fakten und Regeln laden?
- Wie verhält sich Prolog, wenn man eine Anfrage stellt, die mehr als eine Antwort hat?
- Wie kann man ein Prolog-System verlassen, d.h. die Kommandoschleife des Interpreters beenden? (Es heisst nicht „`quit`“.)
- Kann Prolog in Endlosschleifen geraten? Geben Sie ggf. ein Beispiel. Was kann man dann tun?

Zum Selbstudium

e) Schauen Sie sich die folgenden Webseiten an. Sie brauchen nicht alles zu verstehen, sondern sich nur einen ersten Überblick verschaffen, was es gibt. Es ist geplant, dass auf jedem Übungsblatt einige Quellen genannt werden, die für Sie interessant sein könnten.

- „Prolog“ in der Wikipedia:

  [https://de.wikipedia.org/wiki/Prolog_(Programmiersprache)]

- „Learn Prolog Now!“:

  [http://www.learnprolognow.org/].

  Schauen Sie sich insbesondere Abschnitt 1.1 „Some Simple Examples“ der „Free Online Version“ an. Es gibt auch eine Version mit Auswertungsmöglichkeit:

  [http://lpn.swi-prolog.org/lpnpage.php?pagetype=html&pageid=lpn-htmlse1]

- Es empfiehlt sich, ein Prolog-System auf dem eigenen Rechner zu installieren. Es gibt viele freie Prolog-Systeme, die sich mehr oder weniger an den Prolog-Standard halten. Ein bekanntes und problemloses System ist:

  [http://www.swi-prolog.org/]

  Im Laufe des Semesters werden wir uns auch andere Prolog-Systeme anschauen, die vielleicht eine höhere Performance oder mehr Möglichkeiten im Bereich „Constraint Logic Programming“ bieten. Aber für den Einstieg ist SWI-Prolog sicher eine gute Option. Falls Sie auf Ihrem Rechner nichts installieren wollen, können Sie auch die Online-Version testen:

  [http://swish.swi-prolog.org/]

Sie brauchen für diese Aufgabe nichts abzugeben, aber sollten schon einige Zeit investieren (es sind ja zwei Stunden pro Woche für Hausaufgaben eingeplant, und die Aufgaben a) bis c) sollten schnell erledigt sein). Notieren Sie sich auch, was Sie in der Selbstudiums-Woche in der vorlesungsfreien Zeit nochmals anschauen wollen.