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

25.04.2024

# Logische Programmierung & deduktive Datenbanken — Übungsblatt 3 (Rekursion) —

Ihre Lösungen zu den Hausaufgaben (Aufgabe 1 und 2) geben Sie bitte über die Übungs-Plattform in StudIP ab. Einsendeschluss ist Mittwoch, der 1. Mai, $16^{00}$ (die Aufgaben werden in der Übung am Donnerstag besprochen). Aufgabe 3 wird dort auch besprochen: Sie sollten vorher darüber nachdenken, müssen aber nichts abgeben.

## Hausaufgaben

### Aufgabe 1 (Hausaufgabe, 3 Punkte)

Schreiben Sie ein Prädikat `ggt(N,M,T)`, das den größten gemeinsamen Teiler `T` zweier gegebener natürlicher Zahlen `N` und `M` berechnet. Der einfachste Version des bekannten „Euklidischen Algorithmus“ beruht auf folgenden Tatsachen:

- $\mathsf{ggT}(n, n) = n$.
- Wenn $n > m$, dann $\mathsf{ggT}(n, m) = \mathsf{ggT}(n - m, m)$.

  Beweis: Sei $g = \mathsf{ggT}(n, m)$. Da $g$ Teiler von $n$ und $m$ ist, gibt es natürliche Zahlen $a$ und $b$ mit $n = g * a$ und $m = g * b$. Weil $g$ der größte gemeinsame Teiler ist, sind $a$ und $b$ teilerfremd. Nun ist $g$ natürlich auch ein Teiler von $n - m = g * (a - b)$. Hätten $m = g * b$ und $n - m = g * (a - b)$ einen weiteren gemeinsamen Teiler außer $g$, müssen $b$ und $a - b$ einen gemeinsamen Teiler $t$ haben. Dann hätte aber auch $a = (a - b) + b$ diesen Teiler $t$, im Widerspruch zur Voraussetzung, dass $a$ und $b$ teilerfremd sind.
- Umgekehrt gilt: Wenn $n < m$, dann $\mathsf{ggT}(n, m) = \mathsf{ggT}(n, m - n)$.

Hinweise:

- Die Bedingung `N > M` funktioniert auch in Prolog. Sie setzt voraus, dass die Variablen `N` und `M` bereits an einen Wert gebunden sind.
- Die Differenz berechnen Sie mit `D is N - M`. Allgemein ist `is` ein Prädikat, das links eine Variable und rechts einen arithmetischen Ausdruck erwartet. Es bindet die Variable an den Wert des arithmetischen Ausdrucks. Alle Variablen rechts müssen bereits an eine Zahl gebunden sein, damit der Ausdruck ausgewertet werden kann. Die Variable links sollte normalerweise ungebunden sein (noch keinen Wert haben). Es ist aber möglich, links auch eine Zahl (bzw. eine an eine Zahl gebundene Variable) zu haben, dann prüft das Prädikat, ob diese Zahl den richtigen Wert hat.

- Beachten Sie, dass man einmal gebundenen Variablen in Prolog keinen neuen Wert zuweisen kann. (Bei einem rekursiven Aufruf bekommt man aber einen neuen Satz Variablen.)
- Sie sollten in den Bedingungen aller Regeln prüfen, dass die Argumente des Prädikates $> 0$ sind, um die Terminierung zu garantieren.

**Aufgabe 2 (Hausaufgabe, 3 Punkte)**

Es sei nochmals die EMP-DEPT-Datenbank betrachtet:

[https://users.informatik.uni-halle.de/~brass/lp24/prolog/empdept.pl]

In dieser Aufgabe wird nur die Angestellten-Tabelle benötigt:

| EMP | | | | | |
|---|---|---|---|---|---|
| EMPNO | ENAME | JOB | MGR | SAL | DEPTNO |
| 7369 | SMITH | CLERK | 7902 | 800 | 20 |
| 7499 | ALLEN | SALESMAN | 7698 | 1600 | 30 |
| 7521 | WARD | SALESMAN | 7698 | 1250 | 30 |
| 7566 | JONES | MANAGER | 7839 | 2975 | 20 |
| 7654 | MARTIN | SALESMAN | 7698 | 1250 | 30 |
| 7698 | BLAKE | MANAGER | 7839 | 2850 | 30 |
| 7782 | CLARK | MANAGER | 7839 | 2450 | 10 |
| 7788 | SCOTT | ANALYST | 7566 | 3000 | 20 |
| 7839 | KING | PRESIDENT | | 5000 | 10 |
| 7844 | TURNER | SALESMAN | 7698 | 1500 | 30 |
| 7876 | ADAMS | CLERK | 7788 | 1100 | 20 |
| 7900 | JAMES | CLERK | 7698 | 950 | 30 |
| 7902 | FORD | ANALYST | 7566 | 3000 | 20 |
| 7934 | MILLER | CLERK | 7782 | 1300 | 10 |

In der vierten Spalte steht die Angestellten-Nummer des direkten Vorgesetzten des Angestellten. Die Wurzel der Hierarchie ist der Angestellte mit dem „`JOB`“ (dritte Spalte) „`PRESIDENT`“. Schreiben Sie ein Prädikat `level(EMPNO, ENAME, LEVEL)`, das zu jedem Angestellten die Angestellten-Nummer, den Namen, und die Entfernung (Anzahl Kanten) vom Präsidenten der Firma liefert. Der Präsident selbst hat den `LEVEL 0`, seine direkten Untergebenen die Stufe `1`, u.s.w.

# Zur Wiederholung

## Aufgabe 3 (Wiederholungs-Fragen)

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

a) Was ist aus Sicht der Logik die Aufgabe eines Prolog-Systems?

b) Was ist der Unterschied zwischen `'abc'` und `"abc"`? Allgemeiner: Welche unterschiedlichen Repräsentationen von Zeichenketten gibt es in Prolog, und was sind die jeweiligen Vor- und Nachteile? Berücksichtigen Sie auch moderne Prolog-Systeme, die einen eigenen „String“-Datentyp haben.

c) Der Normalfall von „Atomen“ (Bezeichnern) in Prolog sind Folgen von Buchstaben und Ziffern und Unterstrich „`_`“, die mit einem Kleinbuchstaben beginnen. Was zählt außerdem noch als Atom?

d) Welche Schreibweisen für ganzzahlige Konstanten kennen Sie außer der normalen Dezimalschreibweise? Wissen Sie, wie man den ASCII-Code eines Zeichens aufschreiben kann?

e) Was sind Terme in Prolog?

f) Mit welchen Datenstrukturen kann man ein Prolog-Programm beschreiben? Im Skript steht das unter dem Stichwort „Abstrakte Syntax“.

g) Welche drei Parameter beschreiben einen Operator? Wie definiert man einen Operator? Wie kann man die aktuell definierten Operatoren abfragen?

h) Was ist die Standard-Syntax von `1+2+3`? Der Operator `+` hat den Typ `yfx` (d.h. ist linksassoziativ).

i) Warum ist es wichtig, dass in der Standard-Syntax zwischen Funktionssymbol und „`(`“ kein Leerzeichen steht?

j) Welchem Term entspricht die Liste „`[1,2,3]`“ in Prolog?

## Übungsaufgaben (gemeinsam in der Übung zu bearbeiten)

### Aufgabe 4 (Präsenzaufgabe)

a) Wandeln Sie die folgende aussagenlogische Formel (mit den null-stelligen Prädikaten `p` und `q`) in Klauseln um:

$$\neg(\mathtt{p} \wedge \neg\mathtt{q}) \wedge (\mathtt{p} \vee \mathtt{q}).$$

Geben Sie ein Modell an, falls eins existiert.

b) Wandeln Sie die folgende Formel in Klauseln um:

$$\Big(\exists\, \mathtt{Y}\ \forall\, \mathtt{X}\ \mathtt{p}(\mathtt{X},\mathtt{Y}) \vee \neg(\mathtt{q}(\mathtt{X}) \vee \mathtt{r}(\mathtt{Y}))\Big) \wedge \mathtt{r}(1)$$

Für welche Variable(n) werden Skolem-Konstanten bzw. Funktionen eingeführt?

Schreiben Sie die Klauseln

- als Disjunktionen und
- als Fakten und Regeln

Geben Sie ein Herbrandmodell der Klauseln an, falls eines existiert.

c) Gegeben seien folgende Klauseln:

- $\mathtt{q}(\mathtt{X}) \vee \neg\mathtt{p}(\mathtt{X})$.
- $\mathtt{p}(\mathtt{a})$.
- $\neg\mathtt{q}(\mathtt{X})$.

Leiten Sie daraus die leere Klausel mit der Resolventenmethode (Resolution) her. Wenn das gelingt, bedeutet es, dass z.B. $\exists\ \mathtt{X}\ \mathtt{q}(\mathtt{X})$ eine logische Folgerung aus folgender Formelmenge ist:

- $\mathtt{q}(\mathtt{X}) \leftarrow \mathtt{p}(\mathtt{X})$.
- $\mathtt{p}(\mathtt{a})$.

Die Aufteilung in negierte Folgerung und gegebene Formeln kann natürlich auch anders sein.

### Aufgabe 5 (Präsenzaufgabe)

Probieren Sie folgende Anfragen in Prolog aus (arithmetische Vergleiche):

- `1 < 2.`
  Für das Prädikat `<` kann man die übliche Infix-Schreibweise verwenden.
- `<(1, 2).`
  Die Standard-Syntax $\mathtt{p}(\mathtt{t}_1, \ldots, \mathtt{t}_n)$ ist aber auch möglich.
- `10 < 5.`
- `X < 10.`
  Das gibt eine Fehlermeldung. Das Prädikat kann nur ausgewertet werden, wenn die Werte links und rechts bekannt sind.
- `X = 7, X < 10.`
  Der Wert von `X` ist hier bekannt, wenn der Vergleich ausgeführt wird. Die Variable ist zu diesem Zeitpunkt an den Wert `7` gebunden. Prolog wertet die Anfragen immer von links nach rechts aus (entsprechend auch Unteranfragen in Regelrümpfen).
- `X < 10, X = 7.`
  Das funktioniert wieder nicht. Zum Zeitpunkt der Auswertung von `X < 10` ist der Wert von `X` noch nicht bekannt.

Warum ist das Verhalten in den letzten beiden Anfragen aus logischer Sicht ein Problem?

Beim „Constraint Logic Programming“ würden dagegen aktuell noch nicht auswertbare „Constraints“ (Bedingungen) in einen „Constraint-Speicher“ verschoben und später oder auf andere Art berücksichtigt.

### Aufgabe 6 (Präsenzaufgabe)

Schreiben Sie ein Prädikat

```
fak(N, F),
```

das für eine gegebene natürliche Zahl `N` den Wert `F = N!` berechnet. Zu Erinnerung:

$$n! = 1 * 2 * 3 * \cdots * n$$

ist die Anzahl der Permutationen einer $n$-elementigen Menge.

Arithmetische Ausdrücke können Sie in Prolog mit dem Prädikat `is` auswerten. Z.B.:

```
X is Y * 2
```

Dies setzt die Variable `X` auf das Doppelte des Wertes von `Y`. Variablen auf der rechten Seite (hier `Y`) müssen an einen Wert gebunden sein, bevor das Prädikat `is` aufgerufen wird.

## Zum Selbstudium

### Aufgabe 7 (Zusatzaufgabe)

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.

a) Das Buch „Logic, Programming und Prolog (2nd Ed.)“ von Ulf Nilson und Jan Małuszyński gibt es als kostenloses PDF im Internet:

   [http://www.ida.liu.se/~ulfni53/lpp/]

   Es gibt auch einen Foliensatz dazu.

b) Ein aktueller Übersichtsartikel zu deduktiven Datenbanken (fast schon ein Buch) ist Todd J. Green, Shan Shan Huang, Boon Thau Loo, Wenchao Zhou: „Datalog and Recursive Query Processing“. Foundations and Trends in Databases, Vol. 5, No. 2 (2012), 105-195.

   [https://columbiadb.github.io/files/papers/datalog.pdf]

   Laden Sie sich diesen Artikel herunter und lesen Sie die ersten Seiten der Einleitung (106–107) als Motivation. Sie können natürlich gerne auch mehr lesen.

c) Eine Version des Buches „Constraint Logic Programming using ECL$^i$PS$^e$“ von Krzysztof R. Apt und Mark Wallace (2006) gibt es hier:

   [https://www.researchgate.net/publication/
   220693610_Constraint_logic_programming_using_ECLiPSe]

   Ich weiss nicht, ob das beabsichtigt ist: Das Buch wird aktuell noch verkauft.