Prof. Dr. Stefan Brass
Institut für Informatik
MLU Halle-Wittenberg 18.04.2019

Logische Programmierung & deduktive Datenbanken — Übungsblatt 3 (Mathematiker-Stammbaum) —

Ihre Lösungen zu den Hausaufgaben h) bis k) schicken Sie bitte per EMail an den Dozenten (mit "`[lp19]`" in der Betreff-Zeile). Einsendeschluss ist der 24. April (die Aufgaben werden in der Übung am 25. April besprochen). Aufgabe b) wird dort auch besprochen: Sie sollten vorher darüber nachdenken, müssen aber nichts abgeben.

Zum Selbstudium

a) 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 WWW zum Thema dieser Vorlesung gibt, und dabei für sich nützliche Quellen entdecken.

- 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.

- 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.

  [http://blogs.evergreen.edu/sosw/files/2014/04/Green-Vol5-DBS-017.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.

- Eine Version des Buches "Constraint Logic Programming using ECL^{i}PS^e" von Krzysztof R. Apt und Mark Wallace (2006) gibt es hier:

  [https://pdfs.semanticscholar.org/329c/
  0913723ebc282d021cdab9fa32b5400c7f0a.pdf]

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

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

- Was ist eine Signatur (der mehrsortigen Prädikatenlogik erster Stufe)? Wozu dient sie und was sind die wesentlichen Komponenten?
- Was ist eine Interpretation?
- Was ist eine Variablenbelegung?
- Definieren Sie den Begriff "Term" (der Prädikatenlogik erster Stufe).
- Wie sind Formeln der Prädikatenlogik erster Stufe aufgebaut?
- Was bedeutet es, dass eine Formelmenge F eine Formel G logisch impliziert? D.h. wie ist dieser Begriff definiert?
- Was bedeutet es, das zwei Formeln F und G äquivalent sind? Nennen Sie einige Äquivalenzen.
- Wie kann man die Frage, ob eine Formelmenge F eine Formel G logisch impliziert auf einen Konsistenztest zurückführen? Beweiser durch Widerspruch arbeiten nach diesem Prinzip.
- Was sind Literale (positive und negative Literale)?
- Definieren Sie den Begriff "Klausel".
- Kann man jede Formel der Prädikatenlogik erster Stufe in eine Menge von Klauseln überführen?
- Was ist die Idee der Skolemnisierung?
- Definieren Sie den Begriff Substitution. Was unterscheidet Substitutionen von Variablenbelegungen?
- Was sind Herbrand-Interpretationen? Was unterscheidet sie von allgemeinen Interpretationen? Warum sind sie nützlich?
- Erläutern Sie, wie man die Frage, ob eine Formelmenge F eine Formel G logisch impliziert auf die Existenz eines Herbrand-Modells einer Klauselmenge zurückführen kann.

Präsenzaufgaben

c) Gegeben sei eine Datenbank mit den folgenden Relationen:

- `emp(EmpNo, EName, Job, DeptNo)`: Angestellte, hier reduziert auf vier Spalten
- `dept(DeptNo, DName, Loc)`: Abteilungen.

Es sei nun die folgende Fremdschlüssel-Bedingung betrachtet:

$$\forall\, \mathtt{X}, \mathtt{Y}, \mathtt{Z}, \mathtt{D}\, \Big(\mathtt{emp}(\mathtt{X}, \mathtt{Y}, \mathtt{Z}, \mathtt{D}) \rightarrow \exists\, \mathtt{N}, \mathtt{L}\, \mathtt{dept}(\mathtt{D}, \mathtt{N}, \mathtt{L})\Big).$$

Benutzen Sie die in der Vorlesung angegebenen Äquivalenz-Transformationen, um zu zeigen, dass die Formel äquivalent zu der folgenden Formel ist:

$$\forall\, \mathtt{D}\, \exists\, \mathtt{N}, \mathtt{L}\, \forall\, \mathtt{X}, \mathtt{Y}, \mathtt{Z}\, \Big(\mathtt{emp}(\mathtt{X}, \mathtt{Y}, \mathtt{Z}, \mathtt{D}) \rightarrow \mathtt{dept}(\mathtt{D}, \mathtt{N}, \mathtt{L})\Big).$$

Welche Skolem-Funktionen werden für diese Formel eingeführt?

d) Wandeln Sie die folgende Formel in Klauseln um:

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

Schreiben Sie die Klauseln

- als Disjunktionen und
- als Fakten und Regeln

Geben Sie ein Herbrandmodell der Klauseln an.

e) Definieren Sie in Prolog ein Prädikat `prim(N)`, das genau dann wahr ist, wenn die natürliche Zahl `N` eine Primzahl ist.

Es ist nur verlangt, dass das Prädikat für eine gegebene Zahl `N` den korrekten Wahrheitswert hat, z.B. müssen `prim(2)` und `prim(3)` gelten, aber `prim(4)` nicht. Es ist nicht verlangt, dass der Aufruf `prim(N)` mit einer Variablen `N` möglich ist, und `N` dann nacheinander an `2`, `3`, u.s.w. gebunden wird. Wenn Sie Ihr Prädikat so programmieren wollen, dürfen Sie das natürlich tun.

Hinweise:

- Die ganzzahlige Division schreibt man in Prolog `div`. Wenn Sie z.B. 7 durch 3 teilen wollen, schreiben Sie `X is 7 div 3`. Anschließend ist `X = 2`.
- Den Divisionsrest können Sie entsprechend mit dem Operator `mod` bestimmen, z.B. liefert `Y is 7 mod 3` die Variablenbindung `Y = 1`.
- Natürlich kann man auch Zahlen addieren oder subtrahieren, z.B. `Y is X+1`.
- Man kann Zahlen bzw. arithmetische Ausdrücke vergleichen mit den Operatoren `<`, `=<`, `>=`, `>`. Ein Beispiel ist die Bedingung `I < N`.

- Falls Sie testen wollen, dass das Argument eine ganze Zahl ist, können Sie das mit der Bedingung `integer(N)` tun. Sie können sich aber auch einfach darauf verlassen, dass das Prädikat mit einem Argument des richtigen Typs aufgerufen wird. Spätestens bei arithmetischen Operationen wie `div` würde `is` einen Typfehler melden.
- In SWI-Prolog und GNU-Prolog aber nicht im Prolog-Standard (und nicht z.B. in ECLiPSe) gibt es ein Prädikat `between(Low,High,I)` das mit ganzzahligen Werten für `Low` und `High` aufgerufen werden muss, und `I` dann nacheinander an `Low`, `Low+1`, ..., `High` bindet. Sie können sich auch leicht selbst so ein Prädikat definieren:

```
myBetween(Low, High, Low) :-
        Low =< High.
myBetween(Low, High, Value) :-
        Next is Low + 1,
        Next =< High,
        myBetween(Next, High, Value).
```

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

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

  Beweis: Sei $g = \mathtt{ggT}(n, m)$. 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 $\mathtt{ggT}(n, m) = \mathtt{ggT}(n, m - n)$.

Sie sollten in den Bedingungen aller Regeln prüfen, dass die Argumente des Prädikates > 0 sind, um die Terminierung zu garantieren.

g) 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.

Hausaufgabe

Wer eine Doktorarbeit schreibt, wird dabei in der Regel von einem Professor betreut, dem Doktorvater (es gibt auch fälle mit mehr als einem Betreuer). Manche Doktoranden werden später selbst Professor und betreuen ihrerseits Doktoranden. Durch diese Beziehungen entsteht eine Art Stammbaum, der sehr verästelt sein kann und sich über mehrere Kontinente erstrecken kann. Das "Mathematics Genealogy Project"

[http://www.genealogy.ams.org/]

hat sich zum Ziel gesetzt, diese Daten für Doktorarbeiten auf dem Gebiet der Mathematik zu erfassen. Wir haben offenbar eine ältere Version dieser Daten, und stellen sie auf der Webseite zu dieser Vorlesung als Prolog-Fakten zur Verfügung. Dabei werden zwei Prädikate benutzt:

- `doctorate(ID, Name, Degree, University, Year, Country, KindOfThesis, Thesis).`
  [http://www.informatik.uni-halle.de/~brass/lp19/doctorate.facts]
- `advisor(ID, AdvisorID).`
  [http://www.informatik.uni-halle.de/~brass/lp19/advisor.facts]

Formulieren Sie die folgenden Anfragen in Prolog, entweder direkt als Prolog-Anfrage, oder als Prädikat. Sie können natürlich nach Bedarf Hilfsprädikate deklarieren:

h) Schreiben Sie ein "Vorfahr"-Prädikat `ancestor`, das IDs verwendet (also die transitive Hülle der Relation `advisor` berechnet.

i) Darauf basierend entwickeln Sie nun ein Prädikat `ancestorN`, das Namen verwendet.

Hinweis: In den Daten wurde bedauerlicherweise die alte String-Repräsentation verwendet, die Ausgabe würde dann als Liste von ASCII-Codes erfolgen. Mit dem Prädikat `string_to_list(S, L)` ist eine Umwandlung in beiden Richtungen möglich — also auch von der Liste von ASCII-Codes `L` in einen String `S`. Eine Umwandlung zwischen Atomen und Listen von ASCII-Codes ist mit dem Prädikat `name(A,L)` möglich. Vermutlich ist Ihr Prädikat am einfachsten zu verwenden, wenn es Atome als Argumente benutzt, und nicht Listen von ASCII-Codes. Sie können auch beides alternativ unterstützen.

j) Was sind die akademischen Nachkommen (direkte und indirekte Doktoranden) von Gustav Peter Lejeune Dirichlet, die in den Vereinigten Staaten nach 1959 promoviert wurden?

Hinweis: `X > Y` funktioniert in Prolog, wenn `X` und `Y` vorher, also weiter links, an einen Wert gebunden wurden.

- Gibt es Doktoranden, die vor ihrem Doktorvater promoviert wurden?

- Berechnen Sie zu jedem indirekten Doktorvater von Harold Stephen Finkelstein die Anzahl der dazwischenliegenden akademischen Generationen. Beispiel: Zwischen Finkelstein und Finkelstein liegen null Generationen, zwischen Finkelstein und seinen direkten Doktorvätern liegt eine Generation.

  Hinweis: `X is Y+1` bindet `X` an den Wert von `Y+1`. Dazu muss der Wert von `Y` bereits bekannt sein.

k) Schreiben Sie ein Prädikat, das den Ahnenbaum eines Mathematikers "graphisch" ausgibt, indem Doktorväter um jeweils 4 Leerzeichen weiter eingerückt werden.

Hinweis: Das Prädikat `tab(N)` gibt `N` Leerzeichen aus, das Prädikat `nl` gibt eine neue Zeile aus, und das Prädikat `write(T)` druckt den Term `T`.

- Schreiben Sie ein Prädikat für die Anzahl der direkten Doktoranden eines Doktorvaters.

  Hinweis: Der Aufruf `setof(A, B, C)` setzt `C` auf die Liste aller `A`s, für die `B` gilt, und zwar ohne Duplikate, d.h. $C = \{A|B\}$. Der Aufruf `length(L, N)` setzt `N` auf die Länge der Liste `L`.

(Diese Hausaufgaben wurden von PD Dr. Alexander Hinneburg und Dr. Henning Thielemann aus unserer Arbeitsgruppe entwickelt.)