Übung 5

Logische Programmierung

SLD - Resolution

Gegeben sei das folgende Prolog-Programm $\mathcal{P}$. Das Prädikat `len(Xs,Y)` ist genau dann wahr, wenn `Xs` eine Liste von atomaren Formeln ist, von denen `Y` viele nicht beweisbar sind.

```
p(s(X),X).

len([], 0).
len([X|Xs], Y) :- not(X),!,len(Xs,Z),Y is Z+1.
len([_|Xs], Y) :- len(Xs,Y).
```

Geben Sie den SLD-Baum an, der bei der Anfrage "`?- len([p(X,0),p(0,X)],Y).`" entsteht. Sie können gleiche Teile mit "_____ " _____" abkürzen.

Tipp: Überlegen Sie sich zunächst, durch welche zwei Klauseln das Prädikat `not/1` vordefiniert ist.

SLD - Resolution

Gegeben sei folgendes Prolog-Programm, das hintereinanderstehende Duplikate aus Listen löscht:

```
erasePairs([],[]).
erasePairs([X,X|XS],YS) :- !,erasePairs(XS,YS).
erasePairs([X|XS],[X|YS]) :- erasePairs(XS,YS).
```

a) Zeichnen Sie den SLD-Baum **ohne Beachtung des Cuts** zur Anfrage:

```
erasePairs([1,2,2],YS).
```

b) Markieren Sie die Teile Ihres SLD-Baums aus Aufgabe a), die der Cut löscht.

Herbrand Modelle

Gegeben sei die folgende Formel φ über der Signatur (Σ, Δ) mit $\Sigma = \Sigma_0 = \{\mathsf{a}\}$ und $\Delta = \Delta_1 = \{\mathsf{p}\}$:

$$\forall X \; (\; (\; \forall X \; \mathsf{p}(X) \;) \rightarrow \mathsf{p}(X) \;)$$

a) Überführen Sie φ zunächst in Pränex- und dann in Skolem-Normalform.

b) Geben sie alle Strukturen an, die ein Herbrand-Modell von φ sind.
Wieviele solche Strukturen gibt es?

Herbrand Modelle

Sei $\varphi = \mathsf{p}(\mathsf{b},\mathsf{a}) \wedge \forall X, Y\ (\mathsf{p}(X,Y) \rightarrow \mathsf{p}(X,\mathsf{f}(Y)))$ eine Formel über der Signatur (Σ, Δ) mit $\Sigma_0 = \{\mathsf{a},\mathsf{b}\}$, $\Sigma_1 = \{\mathsf{f}\}$ und $\Delta_2 = \{\mathsf{p}\}$. Zusätzlich sei $\psi = \exists X\ \mathsf{p}(X,\mathsf{f}(\mathsf{f}(\mathsf{a})))$.

a) Überführen Sie φ und $\neg\psi$ in hierzu jeweils äquivalenten Klauselmengen $\mathcal{K}(\varphi)$ und $\mathcal{K}(\neg\psi)$.

b) Um zu prüfen, ob $\varphi \models \psi$ gilt, leiten Sie die leere Klausel $\square$ mit Hilfe von Input-Resolution aus der Klauselmenge $\mathcal{K}(\varphi) \cup \mathcal{K}(\neg\psi)$ her.

c) Geben Sie ein Herbrandmodell S für die Formel φ an.

d) Ist Ihr Modell S auch ein Modell der Formel $\mathsf{p}(\mathsf{a},\mathsf{b})$? Falls nein, existiert ein Herbrandmodell von φ, das auch Modell von $\mathsf{p}(\mathsf{a},\mathsf{b})$ ist?

Unifikation

In dieser Aufgabe sollen allgemeinste Unifikatoren bestimmt werden. Sie sollten diese Aufgabe ohne Hilfe eines Rechners lösen, da Sie zur Lösung von Aufgaben dieses Typs auch in der Klausur keinen Rechner zur Verfügung haben.
Nutzen Sie den Algorithmus zur Berechnung des allgemeinsten Unifikators (MGU) aus der Vorlesung, um die folgenden Termpaare auf Unifizierbarkeit zu testen.
Geben Sie neben dem Endergebnis σ auch die Unifikatoren $\sigma_1, \sigma_2, \ldots, \sigma_n$ für die direkten Teilterme der beiden Terme an. Sollte ein σ_i nicht existieren, so begründen Sie kurz, warum die Unifikation fehlschlägt. Geben Sie in diesem Fall an, ob es sich um einen *clash failure* oder einen *occur failure* handelt.

(i) **f(X,Y,Z)** und **f(g(Y,Y),g(Z,Z),a)**

(ii) **g(f(a,X),Y,h(a))** und **g(Y,Z,X)**

(iii) **f(h(X),g(Y),X,Y)** und **f(Z,g(Z),a,Z)**

(iv) **f(g(X),Z,Z)** und **f(Y,Y,X)**

(v) **f(X,h(Y),Y)** und **f(g(Z),X,Z)**

Beweisbäume

Betrachten Sie die Anfrage `?- t(c,Z).` auf folgendem Prolog-Programm:

```
t(X,c) :- t(X,b).
t(X,X) :- p(X,a),t(c,X).
t(X,b) :- t(c,X).
t(c,b).
```

a) Geben Sie den zugehörigen Beweisbaum (SLD-Baum) bis einschließlich Höhe 3. Die Höhe eines Baums ist der längste Pfad von der Wurzel bis zu einem Blatt (ein binärer Baum, welcher nur aus einem Blatt besteht, hat also die Höhe 0). Markieren Sie unendliche Pfade mit ∞ und Fehlschläge mit $(fail)$. Geben Sie alle Lösungen (Antwortsubstitutionen) zur obigen Anfrage an.

b) Strukturieren Sie das gegebene Programm so in ein logisch äquivalentes Programm um, dass Prolog mit seiner Auswertungsstrategie **mindestens eine** Lösung zur gegebenen Anfrage findet. Der Beweisbaum (SLD-Baum) muss nicht endlich sein!

Hinweis: Bei dieser Umstrukturierung dürfen Sie nur die Reihenfolge der Prolog-Klauseln verändern.

Beweisbäume

Betrachten Sie die Anfrage `?- q(Z,s(0)).` auf folgendem Prolog-Programm:

```
q(X, Y) :- q(s(X), Y).
q(0, Y) :- h(s(Y), Y).
q(s(X), X).
h(X, X) :- q(s(X), X).
```

a) Geben Sie den zugehörigen Beweisbaum (SLD-Baum) bis einschließlich Höhe 3. Die Höhe eines Baums ist der längste Pfad von der Wurzel bis zu einem Blatt (ein binärer Baum, welcher nur aus einem Blatt besteht, hat also die Höhe 0). Markieren Sie unendliche Pfade mit ∞ und Fehlschläge mit $(fail)$. Geben Sie alle Lösungen (Antwortsubstitutionen) zur obigen Anfrage an.

b) Strukturieren Sie das gegebene Programm so in ein logisch äquivalentes Programm um, dass Prolog mit seiner Auswertungsstrategie **mindestens eine** Lösung zur gegebenen Anfrage findet. Der Beweisbaum (SLD-Baum) muss nicht endlich sein! Sie brauchen den SLD Baum nicht angeben.

Hinweis: Bei dieser Umstrukturierung dürfen Sie nur die Reihenfolge der Prolog-Klauseln verändern.

Prolog

Gegeben sei folgendes Prolog-Programm $\mathcal{P}$:

```
inc(A,[],[]).
inc(A,[X|XS],[Y|YS]) :- Y =:= A+X, inc(A,[X|XS],[Y|YS]).
```

a) Warum bricht die Anfrage `inc(5,[1,2,3,4,5],XS)` mit einem Programmfehler ab?

b) Ändern Sie das Programm so ab, dass es alle Anfragen der Form

`inc(a,[`q_1`,...,`q_n`],XS)`

für `a,`q_1`,...,`q_n $\in \mathbb{N}$ mit der Antwortsubstitution `XS/[`r_1`,...,`r_n`]` beantwortet, so dass $r_i \in \mathbb{N}$ die Zahl `a+`q_i ist.