

Theorie der Programmiersprachen

9. Übung

1. Aufgabe:

Man zeige mittels Grundresolution *und* mittels prädikatenlogischer Resolution, dass sowohl

$$\forall x (\neg P(x) \wedge P(f(a)))$$

als auch

$$\forall x (((P(f(x)) \rightarrow P(x)) \wedge P(f(f(a)))) \wedge \neg P(a))$$

unerfüllbar sind.

2. Aufgabe:

Man formalisiere die Aussagen (a) und (b) als prädikatenlogische Formeln ($S(x, y)$ - x ist Student von y , $G(x)$ - x ist glücklich, $M(x)$ - x mag Logik).

(a) Der Professor ist glücklich, wenn alle seine Studenten Logik mögen.

(b) Der Professor ist glücklich, wenn er keine Studenten hat.

Man zeige durch Grundresolution *und* durch prädikatenlogische Resolution, dass (b) eine Folgerung aus (a) ist. Formulieren Sie dazu $[(a) \wedge \neg(b)]$ in Klauselform.

Gegeben seien zwei Strukturen \mathcal{A} und \mathcal{B} mit:

$$\begin{aligned} U_{\mathcal{A}} &= \{p, s_1, \dots, s_n\} \\ I_{\mathcal{A}}(G) &= \{x \mid x \text{ ist glücklich}\} \\ I_{\mathcal{A}}(M) &= \{x \mid x \text{ mag Logik}\} \\ I_{\mathcal{A}}(S) &= \{(x, y) \mid x \text{ ist Student von } y\} \\ I_{\mathcal{A}}(p) &= \{p\} \\ I_{\mathcal{A}}(q) &= \{s_1\}, q \text{ Skolemkonstante} \\ U_{\mathcal{B}} &= \mathbb{N} = \{1, 2, 3, \dots\} \\ I_{\mathcal{B}}(G) &= \{x \mid x = 1\} \\ I_{\mathcal{B}}(M) &= \{x \mid x > 1\} \\ I_{\mathcal{B}}(S) &= \{(x, y) \mid x > y\} \\ I_{\mathcal{B}}(p) &= \{1\} \\ I_{\mathcal{B}}(q) &= \{2\} \end{aligned}$$

Zeigen Sie durch "Hochgehen" in Ihren Beweisen, dass beide Strukturen kein Modell für die entsprechende Formel sind.

3. Aufgabe:

Man drücke folgende Tatsachen als prädikatenlogische Formeln aus:

- (a) Jeder Drache ist glücklich, wenn alle seine Kinder fliegen können.
- (b) Grüne Drachen können fliegen.
- (c) Ein Drache ist grün, wenn er Kind mindestens eines grünen Drachen ist.

Man zeige durch Grundresolution *und* durch prädikatenlogische Resolution, dass aus (a), (b) und (c) folgt, dass alle grünen Drachen glücklich sind.

4. Aufgabe:

Geben Sie *alle* prädikatenlogischen Resolventen von

$$\{P(f(x)), \neg Q(z), P(z)\} \quad \text{und} \quad \{\neg P(x), R(g(x), a)\}$$

an (x, y, z sind Variablen, a ist eine Konstante).

5. Aufgabe:

Man wende den Unifikationsalgorithmus auf die Literalmenge

$$L = \{P(x, y), P(f(a), g(x)), P(f(z), g(f(z)))\}$$

an.

6. Aufgabe:

Aus Effizienzgründen wird in manchen Implementierungen des Unifikationsalgorithmus' auf den Test "kommt x in t vor" (occur check) verzichtet. Man gebe ein Beispiel einer nicht unifizierbaren, zweielementigen Literalmenge L_1, L_2 an, so dass L_1 und L_2 keine gemeinsamen Variablen enthalten, und ein Unifikationsalgorithmus ohne occur check - je nach Implementierung - in eine unendliche Schleife gerät oder fälschlicherweise "unifizierbar" konstatiert.

7. Aufgabe:

Zeigen Sie, dass der Unifikationsalgorithmus (naiv implementiert) exponentielle Laufzeit haben kann. Hinweis: Man betrachte das Beispiel:

$$L = \{P(x_1, x_2, \dots, x_n), P(f(x_0, x_0), f(x_1, x_1), \dots, f(x_{n-1}, x_{n-1}))\}$$

Man überlege sich eine geeignete Datenstruktur für Literale bzw. Literalmenge, so dass das Unifizieren effizienter durchgeführt werden kann.