

Disclaimer

Aufgaben aus dieser Vorlage stammen aus der Vorlesung *Logik und Logikprogrammierung* und wurden zu Übungszwecken verändert oder anders formuliert! Für die Korrektheit der Lösungen wird keine Gewähr gegeben.

1. Definitionen und Sätze

- (a) Der Korrektheitsatz der Aussagenlogik für den Wahrheitswertebereich B lautet...

Antwort:

- (b) Eine Menge von Formeln Γ heißt erfüllbar, wenn...

Antwort:

- (c) Zwei Formeln α und β heißen äquivalent, wenn...

Antwort:

- (d) Der Kompaktheitsatz der Aussagenlogik lautet...

Antwort:

- (e) Eine Horn Klausel ist eine Formel der Form

Antwort:

2. Wahrheitswertebereiche

- (a) Werte die Formel $\varpi_a = \neg p \wedge \neg \neg p$ im Heytingschen Wahrheitswertebereich $H_{\mathbb{R}}$ aus für die $H_{\mathbb{R}}$ -Belegung B mit $B(p) = \mathbb{R} \setminus \{0\}$

Antwort:

- (b) Überprüfe ob die Formel $\varphi_B = (\neg p \rightarrow \neg p) \rightarrow p$ eine K_3 -Tautologie ist. Ist φ_b eine $B_{\mathbb{R}}$ Tautologie?

Antwort:

- (c) Überprüfe ob die semantische Folgeung $\{p \rightarrow q, q \rightarrow r\} \Vdash_B r \rightarrow \neg p$ gilt.

Antwort:

3. Erfüllbarkeit

- (a) Überprüfe mittels Markierungsalgorithmus, ob die Formel $\varphi_a = (\neg p \vee q) \wedge (t \vee \neg s) \wedge (\neg r \vee s \vee \neg q) \wedge r \wedge (\neg p \vee t) \wedge \neg s \wedge (\neg r \vee p)$ erfüllbar ist.

Antwort:

- (b) Überprüfe mittels SLD Resolution, ob die Formel $\varphi_b = (r \wedge p) \vee \neg t \vee (p \wedge \neg q) \vee \neg p \vee (\neg r \wedge q \wedge t)$ eine Tautologie ist

Antwort:

4. Monotone Formeln: Eine aussagenlogische Formel φ heißt monoton, falls für alle zu φ passenden B -Belegungen B_1, B_2 mit $B_1(p_i) \leq B_2(p_i)$ für alle $i \in \mathbb{N}$ gilt $B_1(\varphi) \leq B_2(\varphi)$. Beispielsweise sind $p_1 \wedge p_2$ und $\neg \neg p_1$ monoton.

- (a) Entscheide, welche der Formeln $\varphi = p_1 \wedge (p_2 \rightarrow p_3)$, $\psi = \neg p_1 \rightarrow p_2$ monoton sind.

Antwort:

- (b) Zeige per vollständiger Induktion über den Formelaufbau, dass aussagenlogische Formeln in denen weder \neg noch \rightarrow vorkommen, monoton sind.

Antwort:

5. Definitionen und Sätze: Sei Σ eine Signatur. Vervollständige die folgenden Definitionen und Sätze.

- (a) Es gilt $\Delta \vdash \varphi$ für eine Σ -Formel φ und eine Menge Δ von Σ -Formeln, falls

Antwort:

- (b) Der Vollständigkeitssatz der Prädikatenlogik lautet...

Antwort:

- (c) Der Satz von Löwenheim-Skolem lautet...

Antwort:

- (d) Die (elementare) Theorie einer Σ -Struktur A ist

Antwort:

6. Natürliches Schließen

- (a) Gebe die Regeln $(\forall - I)$, $(\exists - E)$ und (GfG) inklusive Bedingung an

Antwort:

- (b) Zeige, dass $\forall x \exists y (f(x) = y)$ ein Theorem ist, indem du eine entsprechende Deduktion angibst

Antwort:

- (c) Zeige, dass $\exists x \forall y (f(x) = y)$ nicht allgemeingültig ist

Antwort:

- (d) Zeige, dass die Formel aus c) erfüllbar ist

Antwort:

7. Prädikatenlogische Definierbarkeit: Betrachte im folgenden Graphen als Σ -Struktur, wobei Σ eine Signatur mit einem zweistelligen Relationssymbol E ist.

- (a) Betrachte den (kommt noch) Graphen und die Σ -Formel $\varphi_a = \forall x \exists y \exists z (((E(x, y) \wedge E(y, z)) \vee (E(y, x) \wedge E(z, x))) \wedge y \neq z)$. Gebe eine Kante an, sodass G mit dieser zusätzlichen Kante als Σ_a -Struktur ein Modell der Formel φ_a ist. Begründe deine Antwort.

Antwort:

- (b) Betrachte die folgenden (kommen noch) Graphen G_1 und G_2 . Gebe einen Σ -Satz φ_b an, so dass $G_1 \models \varphi_b$ und $G_2 \not\models \varphi_b$ gilt.

Antwort:

- (c) Gebe einen Σ -Satz φ_c an, so dass für alle Σ -Strukturen A genau dann $A \models \varphi_c$ gilt, wenn E^A eine Äquivalenzrelation ist (d.h. reflexiv, symmetrisch und transitiv).

Antwort:

8. Normalformeln und Unifikatoren

- (a) Betrachte die Formel $\varphi = \forall x (\exists y (R(x, y) \wedge \neg \exists x (R(y, x))))$. Gebe eine Formel ψ_1 in Pränexform an, die äquivalent zu φ ist und eine Formel ψ_2 in Skolemform, die erfüllbarkeitsäquivalent zu φ ist.

Antwort:

- (b) Sei Σ eine Signatur mit zweistelligem Relationssymbol R , zweistelligem Funktionssymbol f , einstelligem Funktionssymbol g und Konstanten a, b . Ermittle mit dem Unifikationsalgorithmus, welche der folgenden Paare atomarer Formeln unifizierbar sind und gebe einen allgemeinsten Unifikator an, falls dieser existiert.

- i) $(R(x, f(y, g(a))), R(a, f(g(x), y)))$
ii) $(R(f(g, x), y), g(y), R(f(y, z), z))$

Antwort:

9. Gegeben sei folgende Wissensbasis:

- über(rot, orange).
- über(orange, gelb).
- über(gelb, grün).
- über(grün, blau).
- über(blau, violett).
- top(X): über(_, X), !, fail.
- top(_).
- oben(X):-über(X,_),top(X).

Wie antwortet ein Prolog System mit dieser Wissensbasis auf die folgenden Fragen:

(a) ?-top(grün).

Antwort:

(b) ?-top(X).

Antwort:

(c) ?-top(rot).

Antwort:

(d) ?-oben(grün).

Antwort:

(e) ?-oben(X).

Antwort:

(f) ?-oben(rot).

Antwort:

10. Man implementiere folgende Prädikate in Form von Prolog Klauseln

(a) Das Prädikat *partition(L, E, Kl, Gr)* soll eine gegebene Liste ganzer Zahlen *L* in zwei Teillisten partitionieren.

- die Liste *Kl* aller Elemente aus *L*, welche kleiner oder gleich *E* sind und
- die Liste *Gr* aller Elemente aus *L*, welche größer als *E* sind

Beispiel: ?-partition([1,2,3,4,5,6], 3, Kl, Gr).

- *Kl* = [1, 2, 3]
- *Gr* = [4, 5, 6]

Antwort:

(b) Das Prädikat *merge(L1, L2, L)* soll zwei sortierte Listen mit ganzen Zahlen *L1* und *L2* zu einer sortierten Liste *L* verschmelzen.

Antwort:

(c) Das Prädikat *listmerge(ListenListe, L)* bekommt eine Liste sortierter Listen *ListenListe* und soll sie zu einer sortierten Liste *L* verschmelzen. Das in Aufgabe b) definierte Prädikat *merge* kann dabei verwendet werden.

Antwort:

(d) Das Prädikat *am_groesten(L, Max)* soll das größte Element *Max* einer Zahlenliste *L* ermitteln. Falls *L* leer ist, soll „nein“ geantwortet werden.

Antwort:

(e) Das Prädikat *am_kuerzesten(ListenListe, L)* soll aus einer Liste von *ListenListe* die kürzeste Liste *L* ermitteln. Dies soll möglichst effizient geschehen:

- Gestalte die Prozedur rechtsrekursiv

- Sehe davon ab, Listenlängen explizit zu ermitteln. Ermittle diese mit einem Hilfsprädikat *kuerzer_als*(*L1*, *L2*)
- Höre mit der Suche auf, sobald eine leere Liste gefunden wurde. Kürzer geht nicht

Falls *ListenListe* leer ist, soll „nein“ geantwortet werden.

Antwort:

11. Ein binärer Suchbaum mit natürlichen Zahlen in den Knoten sei in Prolog wie folgt als strukturierter Term repräsentiert:

- leerer Baum: *nil*
- nichtleerer Baum: *baum*(Wurzel, LinkerUnterbaum, RechterUnterbaum)

Beispiel Baum mit Wurzel 6, Wurzel 4 im linken Unterbaum, Wurzel 7 im rechten Unterbaum und 2,4 und 9 als Blätter:
baum(6, *baum*(4, *baum*(2, *nil*, *nil*), *baum*(5, *nil*, *nil*)), *baum*(7, *nil*, *baum*(0, *nil*, *nil*))).

Man implementiere folgende Prädikate in Prolog

- (a) Das Prädikat *enthalten*(*Baum*, *Zahl*) bekommt einen binären Suchbaum *Baum* sowie eine Zahl *Zahl* und soll entscheiden, ob diese Zahl in Baum enthalten ist und die Antwort „ja“ oder „nein“ liefern.

Antwort:

- (b) Das Prädikat *flatten*(*Baum*, *Liste*) soll aus einem gegebenen Suchbaum *Baum* die Liste *Liste* aller der im Baum enthaltenen Zahlen in aufsteigender sortierter Reihenfolge liefern.

Antwort: