Theoretische Informatik und Logik – Musterklausur

Maximilian Marx

Wissensbasierte Systeme
Institut für Theoretische Informatik

2021-08-11

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

Eine prädikatenlogische Formel F ist in konjunktiver Normalform, wenn sie von der Form ist:

$$Q_1 Q_2 \cdots Q_n .\ G,$$

wobei $G = \bigwedge_{i=1}^{m} \bigvee_{j=1}^{\ell_i} L_i^j$ eine quantorenfreie Formel mit L_i^j beliebigen Literalen ($i = 1, \ldots, m, j = 1, \ldots, \ell_i$) und $Q_1, Q_2, \ldots Q_n$ beliebige Quantoren sind.

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

Eine prädikatenlogische Klausel ist eine Menge $\{L_1, \ldots, L_n\}$, wobei $L_1, \ldots, L_n$ Literale sind.

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

Ein Unifikator für ein Unifikationsproblem $G = \{s_1 \doteq t_1, \ldots, s_n \doteq t_n\}$ (wobei $s_1, \ldots, s_n, t_1, \ldots, t_n$ Terme sind) ist eine Substitution σ, so dass

$$s_1\sigma = t_1\sigma, \ldots, s_n\sigma = t_n\sigma$$

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

Ein Unifikator für ein Unifikationsproblem $G = \{s_1 \doteq t_1, \ldots, s_n \doteq t_n\}$ (wobei $s_1, \ldots, s_n, t_1, \ldots, t_n$ Terme sind) ist eine Substitution σ, so dass

$$s_1\sigma = t_1\sigma, \ldots, s_n\sigma = t_n\sigma$$

Ist σ ein Unifikator für G, dann ist σ ein allgemeinster Unifikator für G, falls für alle Unifikatoren θ von G gilt: $\sigma \preceq \theta$, d.h. es gibt eine Substitution λ mit $\sigma \circ \lambda = \theta$.

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

Eine Klausel K' ist Variante einer Klausel K, falls K' aus K durch (gleichförmige) Umbenennung von Variablen entsteht.

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

Seien $K_1 = \{A_1, \ldots, A_n, L_1, \ldots, L_k\}$ und $K_2 = \{\neg A'_1, \ldots, \neg A'_m, L'_1, \ldots, L'_\ell\}$, wobei $A_1, \ldots, A_n, A'_1, \ldots, A'_m$ beliebige Atome und $L_1, \ldots, L_k, L'_1, \ldots, L'_\ell$ beliebige Literale.

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

Seien $K_1 = \{A_1, \ldots, A_n, L_1, \ldots, L_k\}$ und $K_2 = \{\neg A'_1, \ldots, \neg A'_m, L'_1, \ldots, L'_\ell\}$, wobei $A_1, \ldots, A_n, A'_1, \ldots, A'_m$ beliebige Atome und $L_1, \ldots, L_k, L'_1, \ldots, L'_\ell$ beliebige Literale.

Die prädikatenlogische Resolvente von K_1 und K_2 ist $\{L_1\sigma, \ldots, L_k\sigma, L'_1\sigma, \ldots, L'_\ell\sigma\}$, wobei σ ein allgemeinster Unifikator für $\{A_1, \ldots, A_n, A'_1, \ldots, A'_m\}$ ist.

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

Ein Entscheider für eine Sprache $L \subseteq \Sigma^*$ ist eine deterministische Turingmaschine, die für jede Eingabe $w \in \Sigma^*$ hält und genau dann akzeptiert, wenn $w \in L$ gilt.

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

Das Halteproblem für eine Turingmaschine $\mathfrak{M}$ ist die Sprache $\{\text{enc}(\mathfrak{M})\#\#\,\text{enc}(w) \mid \mathfrak{M} \text{ hält bei Eingabe } w\}$.

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

Eine $O(f)$-zeitbeschränkte Turingmaschine $\mathcal{M}$ ist eine Turingmaschine, für die es ein $g \in O(f)$ gibt, so dass $\mathcal{M}$ bei Eingabe $w \in \Sigma^*$ stets nach höchstens $g(|w|)$ Schritten hält.

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

Eine totale Funktion $f : \mathbb{N} \to \mathbb{N}$ ist berechenbar, wenn es eine deterministische Turingmaschine gibt, die für jedes $w \in \Sigma^*$ mit dem Bandinhalt $f(w)\sqcup\sqcup\ldots$ anhält.

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

Eine berechenbare Funktion $f : \Sigma^* \to \Sigma^*$ ist eine polynomielle Many-One-Reduktion von einer Sprache **P** auf eine Sprache **Q**, wenn es ein Polynom $p(x)$ und eine $O(p)$-zeitbeschränkte Turingmaschine gibt, die f berechnet,

MK1: Definitionsaufgabe – Grundbegriffe

Definieren Sie folgende Begriffe, wobei Sie die Begriffe „(prädikatenlogische) Formel“, „Atom“, „Literal“, „Quantor“, „Skolemform“, „Substitution“, „Term“ und „deterministische Turingmaschine“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente
6. Entscheider
7. Halteproblem
8. $O(f)$-zeitbeschränkte Turingmaschine
9. berechenbare Funktion
10. polynomielle Many-One-Reduktion

Eine berechenbare Funktion $f : \Sigma^* \to \Sigma^*$ ist eine polynomielle Many-One-Reduktion von einer Sprache **P** auf eine Sprache **Q**, wenn es ein Polynom $p(x)$ und eine $O(p)$-zeitbeschränkte Turingmaschine gibt, die f berechnet, so dass für alle Wörter $w \in \Sigma^*$ gilt:

$$w \in \mathbf{P} \quad \text{genau dann wenn} \quad f(w) \in \mathbf{Q}$$

MK2: Unentscheidbarkeit des PKP

Geben Sie in Ihren eigenen Worten eine kurze Zusammenfassung der wesentlichen Ideen und Schritte für den in der Vorlesung dargestellten Beweis zur Reduktion des Halteproblems von Turingmaschinen auf das Postsche Korrespondenzproblem an.

MK2: Unentscheidbarkeit des PKP

Geben Sie in Ihren eigenen Worten eine kurze Zusammenfassung der wesentlichen Ideen und Schritte für den in der Vorlesung dargestellten Beweis zur Reduktion des Halteproblems von Turingmaschinen auf das Postsche Korrespondenzproblem an.

Zunächst wird das Halteproblem auf ein modifiziertes PCP (MPCP) mit festem Anfangspaar reduziert. Dazu wird der Lauf einer deterministischen Turingmaschine als Wort kodiert: Einzelne Konfigurationen sind Wörter der Form $v\ q\ w$ und ein Lauf ist ein Wort $\#c_0\#c_1\#\ldots$ aus solchen Konfigurationen. Das MPCP beginnt mit dem Wortpaar $\#$ und $\#q_0w\#$ für den Startzustand q_0 und das Eingabewort w. Weitere Wortpaare werden so gewählt, dass die korrekte Fortsetzung des kürzeren Wortes dazu führt, dass im längeren Wort die Nachfolgekonfiguration erscheint. Wenn eine Haltekonfiguration erreicht wird, dann ermöglichen weitere Wortpaare, die beiden Wörter gleich zu machen und nur dann hat das MPCP eine Lösung. Die Reduktion von MPCP zu PCP gelingt durch Einfügung neuer Hilfszeichen $\#$ vor und nach jedem Symbol des MPCP, wobei das obere Wort jeweils um ein $\#$ voraus ist. Nur beim gewünschten Startpaar beginnen beide Wörter mit $\#$, während ein neues Schlusspaar den Unterschied der $\#$ wieder ausgleicht.

MK3: While- und Loop Berechnenbarkeit

```
x0 := x1
x6 := x3
WHILE x3 != 0 DO
    x5 := x5 + x2
    x3 := x3 - 1
    x7 := x3
    x8 := 1
    WHILE x7 != 0 DO
        x7 := 0; x8 := 0
    END
    WHILE x8 != 0 DO
        x8 := x4
        x9 := 1
        WHILE x8 != 0 DO
            x8 := 0; x9 := 0
            x3 := x6
            x4 := x4 - 1
            x0 := x0 + x5
            x5 := 0
        END
        WHILE x9 != 0 DO
            x9 := 0
        END
    END
END
```

1. Welchen Wert x_0 berechnet das Programm für die Eingabe $x_1 = 23, x_2 = 42, x_3 = 2, x_4 = 3$?
2. Welche Funktion $f(x_1, x_2, x_3, x_4)$ berechnet das Programm?
3. Geben Sie ein zu obigem Programm äquivalentes LOOP-Programm an, oder begründen Sie, warum kein solches existiert.

MK3: While- und Loop Berechnenbarkeit

```
x0 := x1
x6 := x3
WHILE x3 != 0 DO
    x5 := x5 + x2
    x3 := x3 - 1
    x7 := x3
    x8 := 1
    WHILE x7 != 0 DO
        x7 := 0; x8 := 0
    END
    WHILE x8 != 0 DO
        x8 := x4
        x9 := 1
        WHILE x8 != 0 DO
            x8 := 0; x9 := 0
            x3 := x6
            x4 := x4 - 1
            x0 := x0 + x5
            x5 := 0
        END
        WHILE x9 != 0 DO
            x9 := 0
        END
    END
END
```

1. Welchen Wert x_0 berechnet das Programm für die Eingabe $x_1 = 23, x_2 = 42, x_3 = 2, x_4 = 3$?
2. Welche Funktion $f(x_1, x_2, x_3, x_4)$ berechnet das Programm?
3. Geben Sie ein zu obigem Programm äquivalentes LOOP-Programm an, oder begründen Sie, warum kein solches existiert.

MK3: While- und Loop Berechnenbarkeit

```
x0 := 23
x6 := 2
WHILE x3 != 0 DO
    x5 := x5 + x2
    x3 := x3 - 1
    x7 := x3
    x8 := 1
    WHILE x7 != 0 DO
        x7 := 0; x8 := 0
    END
    WHILE x8 != 0 DO
        x8 := x4
        x9 := 1
        WHILE x8 != 0 DO
            x8 := 0; x9 := 0
            x3 := x6
            x4 := x4 - 1
            x0 := x0 + x5
            x5 := 0
        END
        WHILE x9 != 0 DO
            x9 := 0
        END
    END
END
```

1. Welchen Wert x_0 berechnet das Programm für die Eingabe $x_1 = 23, x_2 = 42, x_3 = 2, x_4 = 3$?
2. Welche Funktion $f(x_1, x_2, x_3, x_4)$ berechnet das Programm?
3. Geben Sie ein zu obigem Programm äquivalentes LOOP-Programm an, oder begründen Sie, warum kein solches existiert.

MK3: While- und Loop Berechnenbarkeit

```
x0 := 23

WHILE x3 != 0 DO
    x5 := x5 + x2
    x3 := x3 - 1
    x7 := x3
    x8 := 1
    WHILE x7 != 0 DO
        x7 := 0; x8 := 0
    END
    WHILE x8 != 0 DO
        x8 := x4
        x9 := 1
        WHILE x8 != 0 DO
            x8 := 0; x9 := 0
            x3 := 2
            x4 := x4 - 1
            x0 := x0 + x5
            x5 := 0
        END
        WHILE x9 != 0 DO
            x9 := 0
        END
    END
END
```

1. Welchen Wert x_0 berechnet das Programm für die Eingabe $x_1 = 23, x_2 = 42, x_3 = 2, x_4 = 3$?
2. Welche Funktion $f(x_1, x_2, x_3, x_4)$ berechnet das Programm?
3. Geben Sie ein zu obigem Programm äquivalentes LOOP-Programm an, oder begründen Sie, warum kein solches existiert.

MK3: While- und Loop Berechnenbarkeit

```
x0 := 23

WHILE x3 != 0 DO
    x5 := x5 + x2
    x3 := x3 - 1
    x7 := x3
    x8 := 1
    WHILE x7 != 0 DO
        x7 := 0; x8 := 0
    END
    WHILE x8 != 0 DO
        x8 := x4

        WHILE x8 != 0 DO
            x8 := 0;
            x3 := 2
            x4 := x4 - 1
            x0 := x0 + x5
            x5 := 0
        END



    END
END
```

1. Welchen Wert x_0 berechnet das Programm für die Eingabe $x_1 = 23, x_2 = 42, x_3 = 2, x_4 = 3$?
2. Welche Funktion $f(x_1, x_2, x_3, x_4)$ berechnet das Programm?
3. Geben Sie ein zu obigem Programm äquivalentes LOOP-Programm an, oder begründen Sie, warum kein solches existiert.

MK3: While- und Loop Berechnenbarkeit

```
x0 := 23

WHILE x3 != 0 DO
    x5 := x5 + 42
    x3 := x3 − 1
    x7 := x3
    x8 := 1
    WHILE x7 != 0 DO
        x7 := 0; x8 := 0
    END
    WHILE x8 != 0 DO
        x8 := x4

        WHILE x8 != 0 DO
            x8 := 0;
            x3 := 2
            x4 := x4 − 1
            x0 := x0 + x5
            x5 := 0
        END



    END
END
```

1. Welchen Wert x_0 berechnet das Programm für die Eingabe $x_1 = 23, x_2 = 42, x_3 = 2, x_4 = 3$?
2. Welche Funktion $f(x_1, x_2, x_3, x_4)$ berechnet das Programm?
3. Geben Sie ein zu obigem Programm äquivalentes LOOP-Programm an, oder begründen Sie, warum kein solches existiert.

MK3: While- und Loop Berechnenbarkeit

```
x0 := 23

WHILE x3 != 0 DO
    x5 := x5 + 42
    x3 := x3 − 1
    x7 := x3
    x8 := 1
    WHILE x7 != 0 DO
        x7 := 0; x8 := 0
    END
    WHILE x8 != 0 DO
        x8 := x4

        IF x8 != 0 THEN
            x8 := 0;
            x3 := 2
            x4 := x4 − 1
            x0 := x0 + x5
            x5 := 0
        END IF



    END
END
```

1. Welchen Wert x_0 berechnet das Programm für die Eingabe $x_1 = 23, x_2 = 42, x_3 = 2, x_4 = 3$?
2. Welche Funktion $f(x_1, x_2, x_3, x_4)$ berechnet das Programm?
3. Geben Sie ein zu obigem Programm äquivalentes LOOP-Programm an, oder begründen Sie, warum kein solches existiert.

MK3: While- und Loop Berechnenbarkeit

```
x_0 := 23

WHILE x_3 != 0 DO
    x_5 := x_5 + 42
    x_3 := x_3 - 1
    x_7 := x_3
    x_8 := 1
    IF x_7 != 0 THEN
        x_7 := 0; x_8 := 0
    END IF
    WHILE x_8 != 0 DO
        x_8 := x_4

        IF x_8 != 0 THEN
            x_8 := 0;
            x_3 := 2
            x_4 := x_4 - 1
            x_0 := x_0 + x_5
            x_5 := 0
        END IF


    END
END
```

1. Welchen Wert x_0 berechnet das Programm für die Eingabe $x_1 = 23, x_2 = 42, x_3 = 2, x_4 = 3$?
2. Welche Funktion $f(x_1, x_2, x_3, x_4)$ berechnet das Programm?
3. Geben Sie ein zu obigem Programm äquivalentes LOOP-Programm an, oder begründen Sie, warum kein solches existiert.

MK3: While- und Loop Berechnenbarkeit

$x_0 := 23$

WHILE $x_3 \; != 0$ **DO**
  $x_5 := x_5 + 42$
  $x_3 := x_3 - 1$
  $x_7 := x_3$
  $x_8 := 1$
  IF $x_7 \; != 0$ **THEN**
    $x_7 := 0; x_8 := 0$
  END IF
  WHILE $x_8 \; != 0$ **DO**
    $x_8 := x_4$

    IF $x_8 \; != 0$ **THEN**
      $x_8 := 0;$
      $x_3 := 2$
      $x_4 := x_4 - 1$
      $x_0 := x_0 + x_5$
      $x_5 := 0$
    END IF

  END
END

x_0	x_3	x_4	x_5	x_7	x_8
0	2	3	0	0	0
23	2	3	0	0	0
23	2	3	42	0	0
23	1	3	42	0	0
23	1	3	42	1	0
23	1	3	42	1	1
23	1	3	42	0	0
23	1	3	84	0	0
23	0	3	84	0	0
23	0	3	84	0	0
23	0	3	84	0	1
23	0	3	84	0	3
23	0	3	84	0	0
23	2	3	84	0	3
23	2	2	84	0	3
107	2	3	84	0	3
107	2	3	0	0	3
⋮	⋮	⋮	⋮	⋮	⋮

MK3: While- und Loop Berechnenbarkeit

```
x0 := 23

WHILE x3 != 0 DO
    x5 := x5 + 42
    x3 := x3 − 1
    x7 := x3
    x8 := 1
    IF x7 != 0 THEN
        x7 := 0; x8 := 0
    END IF
    WHILE x8 != 0 DO
        x8 := x4

        IF x8 != 0 THEN
            x8 := 0;
            x3 := 2
            x4 := x4 − 1
            x0 := x0 + x5
            x5 := 0
        END IF



    END
END
```

x_0	x_3	x_4	x_5	x_7	x_8
0	2	3	0	0	0
107	2	3	42	0	3
107	1	3	42	0	3
107	1	3	42	1	3
107	1	3	42	1	1
107	1	3	42	0	0
107	1	3	42	0	0
107	0	3	84	0	0
107	1	3	84	0	0
107	1	3	84	0	1
107	1	3	84	0	3
107	1	3	84	0	0
107	2	3	84	0	0
107	2	2	84	0	0
191	2	2	84	0	0
191	2	2	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮

MK3: While- und Loop Berechnenbarkeit

```
x0 := 23

WHILE x3 != 0 DO
    x5 := x5 + 42
    x3 := x3 − 1
    x7 := x3
    x8 := 1
    IF x7 != 0 THEN
        x7 := 0; x8 := 0
    END IF
    WHILE x8 != 0 DO
        x8 := x4

        IF x8 != 0 THEN
            x8 := 0;
            x3 := 2
            x4 := x4 − 1
            x0 := x0 + x5
            x5 := 0
        END IF




    END
END
```

x_0	x_3	x_4	x_5	x_7	x_8
0	2	3	0	0	0
107	2	3	42	0	3
107	1	3	42	0	3
107	1	3	42	1	3
107	1	3	42	1	1
107	1	3	42	0	0
107	1	3	42	0	0
107	0	3	84	0	0
107	1	3	84	0	0
107	1	3	84	0	1
107	1	3	84	0	3
107	1	3	84	0	0
107	2	3	84	0	0
107	2	2	84	0	0
191	2	2	84	0	0
191	2	2	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮
275	2	2	84	0	0

MK3: While- und Loop Berechnenbarkeit

```
x0 := 23

WHILE x3 != 0 DO
    x5 := x5 + 42
    x3 := x3 − 1
    x7 := x3
    x8 := 1
    IF x7 != 0 THEN
        x7 := 0; x8 := 0
    END IF
    WHILE x8 != 0 DO
        x8 := x4

        IF x8 != 0 THEN
            x8 := 0;
            x3 := 2
            x4 := x4 − 1
            x0 := x0 + x5
            x5 := 0
        END IF


    END
END
```

1. Welchen Wert x_0 berechnet das Programm für die Eingabe $x_1 = 23, x_2 = 42, x_3 = 2, x_4 = 3$?
2. Welche Funktion $f(x_1, x_2, x_3, x_4)$ berechnet das Programm?
3. Geben Sie ein zu obigem Programm äquivalentes LOOP-Programm an, oder begründen Sie, warum kein solches existiert.

MK3: While- und Loop Berechnenbarkeit

```
x0 := 23

WHILE x3 != 0 DO
    x5 := x5 + 42
    x3 := x3 − 1
    x7 := x3
    x8 := 1
    IF x7 != 0 THEN
        x7 := 0; x8 := 0
    END IF
    WHILE x8 != 0 DO
        x8 := x4

        IF x8 != 0 THEN
            x8 := 0;
            x3 := 2
            x4 := x4 − 1
            x0 := x0 + x5
            x5 := 0
        END IF


    END
END
```

1. Welchen Wert x_0 berechnet das Programm für die Eingabe $x_1 = 23, x_2 = 42, x_3 = 2, x_4 = 3$?
2. Welche Funktion $f(x_1, x_2, x_3, x_4)$ berechnet das Programm?
3. Geben Sie ein zu obigem Programm äquivalentes LOOP-Programm an, oder begründen Sie, warum kein solches existiert.

Das Programm berechnet die Funktion $f(x_1, x_2, x_3, x_4) = x_1 + x_2 \cdot x_3 \cdot x_4$.

MK3: While- und Loop Berechnenbarkeit

```
x0 := 23

WHILE x3 != 0 DO
    x5 := x5 + 42
    x3 := x3 − 1
    x7 := x3
    x8 := 1
    IF x7 != 0 THEN
        x7 := 0; x8 := 0
    END IF
    WHILE x8 != 0 DO
        x8 := x4

        IF x8 != 0 THEN
            x8 := 0;
            x3 := 2
            x4 := x4 − 1
            x0 := x0 + x5
            x5 := 0
        END IF



    END
END
```

1. Welchen Wert x_0 berechnet das Programm für die Eingabe $x_1 = 23, x_2 = 42, x_3 = 2, x_4 = 3$?
2. Welche Funktion $f(x_1, x_2, x_3, x_4)$ berechnet das Programm?
3. Geben Sie ein zu obigem Programm äquivalentes LOOP-Programm an, oder begründen Sie, warum kein solches existiert.

MK3: While- und Loop Berechnenbarkeit

```
x0 := 23

WHILE x3 != 0 DO
    x5 := x5 + 42
    x3 := x3 − 1
    x7 := x3
    x8 := 1
    IF x7 != 0 THEN
        x7 := 0; x8 := 0
    END IF
    WHILE x8 != 0 DO
        x8 := x4

        IF x8 != 0 THEN
            x8 := 0;
            x3 := 2
            x4 := x4 − 1
            x0 := x0 + x5
            x5 := 0
        END IF


    END
END
```

1. Welchen Wert x_0 berechnet das Programm für die Eingabe $x_1 = 23, x_2 = 42, x_3 = 2, x_4 = 3$?
2. Welche Funktion $f(x_1, x_2, x_3, x_4)$ berechnet das Programm?
3. Geben Sie ein zu obigem Programm äquivalentes LOOP-Programm an, oder begründen Sie, warum kein solches existiert.

```
x0 := x1
LOOP x3 DO
    LOOP x4 DO
        x0 := x0 + x2
    END
END
```

MK4: Pseudopolynomielle Probleme

Erklären Sie kurz und mit eigenen Worten, was ein pseudopolynomielles Problem ist und veranschaulichen Sie dies an einem Beispiel. Diskutieren Sie kurz, welche Bedeutung diese Eigenschaft für die praktische Lösbarkeit eines Problems hat.

MK4: Pseudopolynomielle Probleme

Erklären Sie kurz und mit eigenen Worten, was ein pseudopolynomielles Problem ist und veranschaulichen Sie dies an einem Beispiel. Diskutieren Sie kurz, welche Bedeutung diese Eigenschaft für die praktische Lösbarkeit eines Problems hat.

Pseudopolynomielle Probleme sind NP-schwere Entscheidungsprobleme, die in polynomieller Zeit gelöst werden können, wenn die in Probleminstanzen vorkommenden Zahlen unär kodiert werden.
Ein Beispiel ist das Problem SubSet-Sum (Knapsack). Eine Eingabe dieses Problems besteht in einer Menge von Gegenständen, denen Zahlen (Gewichte und Werte) zugeordnet sind. Das Problem lässt sich mit dynamischer Programmierung in polynomieller Zeit bezüglich des Betrags der Zahlen (der Gewichte) lösen, ist aber NP-vollständig, wenn Zahlen wie üblich binär kodiert sind.
Praktisch bedeutet das, dass derartige Probleme trotz ihrer theoretischen NP-Schwere durchaus effizient lösbar sein können, wenn die vorkommenden Zahlen nicht zu groß werden. Andere Teile der Eingabe dürfen hingegen wachsen.

MK5: Resolutionsverfahren

Gegeben sind die folgenden prädikatenlogischen Formeln:

$$F = \forall x, y, z.\ \Big(p\big(f(x), y\big) \rightarrow q\big(z, g(y)\big)\Big)$$
$$G = \neg\exists u, v.\ \Big(p\big(u, f(v)\big)\Big)$$
$$H = \forall x, y.\ \Big(q\big(x, g(f(y))\big)\Big)$$

Prüfen Sie mit Hilfe des Resolutionsverfahrens, ob $\{F, G\} \models H$ gilt. Geben Sie bei jeder Resolventenbildung die verwendeten Klauseln und den verwendeten allgemeinsten Unifikator an.

MK5: Resolutionsverfahren – Normalform, Klauselform

1. Schritt Normalformen bilden:

$$F = \forall x, y, z.\ \Big(p\big(f(x), y\big) \rightarrow q\big(z, g(y)\big)\Big)$$

$$\equiv$$

$$G = \neg\exists u, v.\ \Big(p\big(u, f(v)\big)\Big)$$

$$\equiv$$

$$\neg H \equiv$$

MK5: Resolutionsverfahren – Normalform, Klauselform

1. Schritt Normalformen bilden:

$$
\begin{aligned}
F &= \forall x, y, z.\ \Big(p\big(f(x), y\big) \rightarrow q\big(z, g(y)\big)\Big) \\
&\equiv \forall x, y, z.\ \Big(\neg p\big(f(x), y\big) \vee q\big(z, g(y)\big)\Big) \\
G &= \neg \exists u, v.\ \Big(p\big(u, f(v)\big)\Big) \\
&\equiv \\
\neg H &\equiv
\end{aligned}
$$

MK5: Resolutionsverfahren – Normalform, Klauselform

1. Schritt Normalformen bilden:

$$
\begin{aligned}
F &= \forall x, y, z.\ \Big(p\big(f(x), y\big) \rightarrow q\big(z, g(y)\big)\Big) \\
&\equiv \forall x, y, z.\ \Big(\neg p\big(f(x), y\big) \vee q\big(z, g(y)\big)\Big) \\
\\
G &= \neg \exists u, v.\ \Big(p\big(u, f(v)\big)\Big) \\
&\equiv \forall u, v.\ \Big(\neg p\big(u, f(v)\big)\Big) \\
\neg H &\equiv
\end{aligned}
$$

MK5: Resolutionsverfahren – Normalform, Klauselform

1. Schritt Normalformen bilden:

$$
\begin{aligned}
F &= \forall x, y, z.\ \Big(p\big(f(x), y\big) \to q\big(z, g(y)\big)\Big) \\
&\equiv \forall x, y, z.\ \Big(\neg p\big(f(x), y\big) \vee q\big(z, g(y)\big)\Big) \\
\\
G &= \neg\exists u, v.\ \Big(p\big(u, f(v)\big)\Big) \\
&\equiv \forall u, v.\ \Big(\neg p\big(u, f(v)\big)\Big) \\
\neg H &\equiv \neg\forall x, y.\ q\big(x, g(f(y))\big)
\end{aligned}
$$

MK5: Resolutionsverfahren – Normalform, Klauselform

1. Schritt Normalformen bilden:

$$
\begin{aligned}
F &= \forall x, y, z.\ \Big(p\big(f(x), y\big) \to q\big(z, g(y)\big)\Big) \\
&\equiv \forall x, y, z.\ \Big(\neg p\big(f(x), y\big) \vee q\big(z, g(y)\big)\Big) \\
\\
G &= \neg\exists u, v.\ \Big(p\big(u, f(v)\big)\Big) \\
&\equiv \forall u, v.\ \Big(\neg p\big(u, f(v)\big)\Big) \\
\neg H &\equiv \neg\forall x, y.\ q\big(x, g(f(y))\big) \\
&\equiv \exists x, y.\ \neg q\big(x, g(f(y))\big)
\end{aligned}
$$

MK5: Resolutionsverfahren – Normalform, Klauselform

1. Schritt Normalformen bilden:

$$
\begin{aligned}
F &= \forall x, y, z.\ \Big(p\big(f(x), y\big) \to q\big(z, g(y)\big)\Big) \\
&\equiv \forall x, y, z.\ \Big(\neg p\big(f(x), y\big) \vee q\big(z, g(y)\big)\Big) \\
\\
G &= \neg\exists u, v.\ \Big(p\big(u, f(v)\big)\Big) \\
&\equiv \forall u, v.\ \Big(\neg p\big(u, f(v)\big)\Big) \\
\neg H &\equiv \neg\forall x, y.\ q\big(x, g(f(y))\big) \\
&\equiv \exists x, y.\ \neg q\big(x, g(f(y))\big) \xrightarrow{\text{Skolemisierung}} \neg q\big(c, g(f(d))\big)
\end{aligned}
$$

MK5: Resolutionsverfahren – Normalform, Klauselform

1. Schritt Normalformen bilden:

$$
\begin{aligned}
F &= \forall x, y, z.\ \Big(p\big(f(x), y\big) \to q\big(z, g(y)\big)\Big) \\
&\equiv \forall x, y, z.\ \Big(\neg p\big(f(x), y\big) \vee q\big(z, g(y)\big)\Big) \\
\\
G &= \neg\exists u, v.\ \Big(p\big(u, f(v)\big)\Big) \\
&\equiv \forall u, v.\ \Big(\neg p\big(u, f(v)\big)\Big) \\
\neg H &\equiv \neg\forall x, y.\ q\big(x, g(f(y))\big) \\
&\equiv \exists x, y.\ \neg q\big(x, g(f(y))\big) \xrightarrow{\text{Skolemisierung}} \neg q\big(c, g(f(d))\big)
\end{aligned}
$$

2. Klauselform:

$$
\Big\{ \underbrace{\big\{\neg p\big(f(x), y\big), q\big(z, g(y)\big)\big\}}_{(1)}, \underbrace{\big\{\neg p\big(u, f(v)\big)\big\}}_{(2)}, \underbrace{\big\{\neg q\big(c, g(f(d))\big)\big\}}_{(3)} \Big\}
$$

MK5: Resolutionsverfahren

3. Resolution:

$$\Big\{\underbrace{\big\{\neg p\big(f(x),y\big),q\big(z,g(y)\big)\big\}}_{(1)},\underbrace{\big\{\neg p\big(u,f(v)\big)\big\}}_{(2)},\underbrace{\big\{\neg q\big(c,g(f(d))\big)\big\}}_{(3)}\Big\}$$

MK5: Resolutionsverfahren

3. Resolution:

$$\Big\{\underbrace{\big\{\neg p\big(f(x), y\big), q\big(z, g(y)\big)\big\}}_{(1)}, \underbrace{\big\{\neg p\big(u, f(v)\big)\big\}}_{(2)}, \underbrace{\big\{\neg q\big(c, g(f(d))\big)\big\}}_{(3)}\Big\}$$

Damit ergibt sich die folgende Resolvente aus (1) und (3) mit $\{z \mapsto c, y \mapsto f(d)\}$:

$$\underbrace{\big\{\neg p\big(f(x), f(d)\big)\big\}}_{(4)}$$

MK5: Resolutionsverfahren

3. Resolution:

$$\Big\{\underbrace{\big\{\neg p\big(f(x), y\big), q\big(z, g(y)\big)\big\}}_{(1)}, \underbrace{\big\{\neg p\big(u, f(v)\big)\big\}}_{(2)}, \underbrace{\big\{\neg q\big(c, g(f(d))\big)\big\}}_{(3)}\Big\}$$

Damit ergibt sich die folgende Resolvente aus (1) und (3) mit $\{z \mapsto c, y \mapsto f(d)\}$:

$$\underbrace{\big\{\neg p\big(f(x), f(d)\big)\big\}}_{(4)}$$

Es lassen sich keine weiteren Resolventen bilden, d. h. das Resolutionsverfahren terminiert, ohne die leere Klausel abzuleiten.

MK5: Resolutionsverfahren

3. Resolution:

$$\Big\{\underbrace{\big\{\neg p\big(f(x),y\big),q\big(z,g(y)\big)\big\}}_{(1)},\underbrace{\big\{\neg p\big(u,f(v)\big)\big\}}_{(2)},\underbrace{\big\{\neg q\big(c,g(f(d))\big)\big\}}_{(3)}\Big\}$$

Damit ergibt sich die folgende Resolvente aus (1) und (3) mit $\{z \mapsto c, y \mapsto f(d)\}$:

$$\underbrace{\big\{\neg p\big(f(x),f(d)\big)\big\}}_{(4)}$$

Es lassen sich keine weiteren Resolventen bilden, d. h. das Resolutionsverfahren terminiert, ohne die leere Klausel abzuleiten.

Also ist $\{F, G, \neg H\}$ erfüllbar, d.h. es gilt $\{F, G\} \not\models H$.

MK6: Konsequenzrelation

Erklären Sie kurz und in eigenen Worten, wie die Konsequenzrelation einer Logik modelltheoretisch definiert werden kann. Geben Sie an, wie diese allgemeine Definition bei einer konkreten Beispiellogik angewendet wird.

MK6: Konsequenzrelation

Erklären Sie kurz und in eigenen Worten, wie die Konsequenzrelation einer Logik modelltheoretisch definiert werden kann. Geben Sie an, wie diese allgemeine Definition bei einer konkreten Beispiellogik angewendet wird.

Die Modelltheorie einer Logik legt fest, welche Formeln der Logik in welchen ihrer Modellen gelten. Gilt eine Formel F in einem Modell I, dann ist I ein Modell für F. Ist T eine Formelmenge, dann ist I ein Modell für T, falls I ein Modell für jede Formel $F \in T$ ist. Damit können wir logische Konsequenz definieren: Eine Formel F folgt aus einer Formelmenge T, wenn jedes Modell I von T auch ein Modell für F ist. Die Konsequenzrelation $\models$ ist die Menge aller solcher Paare von Formelmengen T und Formeln F, bei denen F aus T folgt.
Diese Defintion wird zum Beispiel in der Aussagenlogik angewendet. Dort sind Formeln aussagenlogische Formeln und Modelle Variablenzuweisungen. Eine Formel wird durch eine Variablenzuweisung erfüllt, wenn sie ihr den Wahrheitswert „wahr“ zuweist. Damit ergibt sich zum Beispiel, dass aus der Formelmenge $\{p, q\}$ die Formel $F = p \wedge q$ folgt.