Theoretische Informatik und Logik – Musterklausur

Sascha Klüppelholz

Algebraische und logische Grundlagen der Informatik

Institut für Theoretische Informatik

20.07.2018

MK1: Definitionsaufgabe – Grundbegriffe

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

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente

MK1: Definitionsaufgabe – Grundbegriffe

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

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente

MK1: Definitionsaufgabe – Grundbegriffe

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

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente

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“, und „Term“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente

MK1: Definitionsaufgabe – Grundbegriffe

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

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente

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“, und „Term“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente

MK1: Definitionsaufgabe – Grundbegriffe

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

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente

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“, und „Term“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente

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“, und „Term“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente

MK1: Definitionsaufgabe – Grundbegriffe

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

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente

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“, und „Term“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente

MK1: Definitionsaufgabe – Grundbegriffe

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

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente

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“, und „Term“ als bekannt voraussetzen dürfen:

1. konjunktive Normalform
2. prädikatenlogische Klausel
3. allgemeinster Unifikator
4. Variante einer prädikatenlogischen Klausel
5. prädikatenlogische Resolvente

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.

MK2: 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.

MK2: 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.

MK2: 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.

MK2: 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.

MK2: 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.

MK2: 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.

MK2: 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
    WHILE x_7 != 0 DO
        x_7 := 0; x_8 := 0
    END
    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.

MK2: 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.

MK2: 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
⋮	⋮	⋮	⋮	⋮	⋮

MK2: 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

MK2: 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

MK2: 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.

MK2: 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.

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

MK2: 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.

MK2: 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
```

MK3: 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.

MK3: Resolutionsverfahren – Normalform, Klauselform

1. Schritt Normalformen bilden:

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

$$\equiv$$

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

$$\equiv$$

$$\neg H \equiv$$

MK3: Resolutionsverfahren – Normalform, Klauselform

1. Schritt Normalformen bilden:

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

MK3: Resolutionsverfahren – Normalform, Klauselform

1. Schritt Normalformen bilden:

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

MK3: Resolutionsverfahren – Normalform, Klauselform

1. Schritt Normalformen bilden:

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

MK3: Resolutionsverfahren – Normalform, Klauselform

1. Schritt Normalformen bilden:

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

MK3: Resolutionsverfahren – Normalform, Klauselform

1. Schritt Normalformen bilden:

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

MK3: 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:

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

MK3: Resolutionsverfahren

3. Resolution:

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

MK3: 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 $\{z \mapsto c, y \mapsto f(d)\}$:

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

MK3: 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.

MK3: 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 $\{F, G\} \not\models H$.

MK4: Postsches Korrespondenzproblem

In der Vorlesung wurde das Postsche Korrespondenzproblem (**PCP**) vorgestellt:

Gegeben: eine endliche Folge von Wortpaaren $(x_1, y_1), \ldots, (x_k, y_k)$ über Σ^*.

Frage: Gibt es eine Folge von Zahlen $i_1, \ldots, i_\ell$, so dass gilt

$$x_{i_1} \ldots x_{i_\ell} = y_{i_1} \ldots y_{i_\ell}$$

wobei $\ell > 0$ ist und $i_j \in \{1, \ldots, k\}$ für alle $j = 1, \ldots, \ell$?

---

1. Bestimmen Sie alle Lösungen für das folgende PCP: $(bc, bcd), (b, b), (da, a)$.
2. Zeigen Sie, dass die folgende Instanz keine Lösung hat:
   $(bab, abb), (abb, bb), (ba, bab)$.
3. Sei $\mathbf{PCP}^{\Sigma}_{\leqslant n}$ die Menge aller lösbaren PCP-Instanzen über dem Alphabet Σ, in deren Wortpaaren nur Wörter der Länge n oder weniger vorkommen. Ist $\mathbf{PCP}^{\Sigma}_{\leqslant n}$ entscheidbar oder unentscheidbar? Begründen Sie Ihre Antwort.

MK4: Postsches Korrespondenzproblem

1. Bestimmen Sie alle Lösungen für das folgende PCP: $(bc, bcd), (b, b), (da, a)$.

MK4: Postsches Korrespondenzproblem

1. Bestimmen Sie alle Lösungen für das folgende PCP: $(bc, bcd), (b, b), (da, a)$.

   Wir starten mit folgenden drei Beobachtungen:

   - eine Lösung kann niemals mit (da, a) beginnen
   - nach (bc, bcd) muss stets ein (da, a) folgen
   - es können jederzeit beliebig viele (b, b) vorkommen

MK4: Postsches Korrespondenzproblem

1. Bestimmen Sie alle Lösungen für das folgende PCP: $(bc, bcd), (b, b), (da, a)$.

   Wir starten mit folgenden drei Beobachtungen:

   - eine Lösung kann niemals mit (da, a) beginnen
   - nach (bc, bcd) muss stets ein (da, a) folgen
   - es können jederzeit beliebig viele (b, b) vorkommen

   Mit $1 \mapsto (bc, bcd), 2 \mapsto (b, b), 3 \mapsto (da, a)$ beschreibt die Sprache $L((2|13)^+)$ alle Lösungen des gegebenen PCPs.

MK4: Postsches Korrespondenzproblem

2. Zeigen Sie, dass die folgende Instanz keine Lösung hat:
   $(bab, abb), (abb, bb), (ba, bab)$.

MK4: Postsches Korrespondenzproblem

2. Zeigen Sie, dass die folgende Instanz keine Lösung hat:
   $(bab, abb), (abb, bb), (ba, bab)$.

   jede Lösung muss mit (ba, bab) beginnen
   danach kann nur (bab, abb) folgen
   aber auch dann kann weiter nur (bab, abb) folgen

MK4: Postsches Korrespondenzproblem

2. Zeigen Sie, dass die folgende Instanz keine Lösung hat:
   $(bab, abb), (abb, bb), (ba, bab)$.

   jede Lösung muss mit (ba, bab) beginnen
   danach kann nur (bab, abb) folgen
   aber auch dann kann weiter nur (bab, abb) folgen

   $\Rightarrow$ Die untere Zeile bleibt folglich stets ein Zeichen länger als die obere Zeile.

MK4: Postsches Korrespondenzproblem

2. Zeigen Sie, dass die folgende Instanz keine Lösung hat:
   $(bab, abb), (abb, bb), (ba, bab)$.

   jede Lösung muss mit (ba, bab) beginnen
   danach kann nur (bab, abb) folgen
   aber auch dann kann weiter nur (bab, abb) folgen

   $\Rightarrow$ Die untere Zeile bleibt folglich stets ein Zeichen länger als die obere Zeile.

   $\Rightarrow$ Das PCP hat keine Lösung.

MK4: Postsches Korrespondenzproblem

3. Sei $\mathbf{PCP}^{\Sigma}_{\leqslant n}$ die Menge aller lösbaren PCP-Instanzen über dem Alphabet Σ, in deren Wortpaaren nur Wörter der Länge n oder weniger vorkommen. Ist $\mathbf{PCP}^{\Sigma}_{\leqslant n}$ entscheidbar oder unentscheidbar? Begründen Sie Ihre Antwort.

MK4: Postsches Korrespondenzproblem

3. Sei $\mathbf{PCP}^{\Sigma}_{\leqslant n}$ die Menge aller lösbaren PCP-Instanzen über dem Alphabet Σ, in deren Wortpaaren nur Wörter der Länge n oder weniger vorkommen. Ist $\mathbf{PCP}^{\Sigma}_{\leqslant n}$ entscheidbar oder unentscheidbar? Begründen Sie Ihre Antwort.

   Es gibt nur endlich viele Wortpaare von Wörtern der Länge höchstens n über Σ.

MK4: Postsches Korrespondenzproblem

3. Sei $\mathbf{PCP}^{\Sigma}_{\leqslant n}$ die Menge aller lösbaren PCP-Instanzen über dem Alphabet Σ, in deren Wortpaaren nur Wörter der Länge n oder weniger vorkommen. Ist $\mathbf{PCP}^{\Sigma}_{\leqslant n}$ entscheidbar oder unentscheidbar? Begründen Sie Ihre Antwort.

   Es gibt nur endlich viele Wortpaare von Wörtern der Länge höchstens n über Σ.
   Es gibt nur endliche viele PCP-Instanzen, die nur solche Wortpaare enthalten.

MK4: Postsches Korrespondenzproblem

3. Sei $\mathbf{PCP}^{\Sigma}_{\leqslant n}$ die Menge aller lösbaren PCP-Instanzen über dem Alphabet Σ, in deren Wortpaaren nur Wörter der Länge n oder weniger vorkommen. Ist $\mathbf{PCP}^{\Sigma}_{\leqslant n}$ entscheidbar oder unentscheidbar? Begründen Sie Ihre Antwort.

   Es gibt nur endlich viele Wortpaare von Wörtern der Länge höchstens n über Σ. Es gibt nur endliche viele PCP-Instanzen, die nur solche Wortpaare enthalten. Damit ist $\mathbf{PCP}^{\Sigma}_{\leqslant n}$ endlich, also regulär, und damit entscheidbar.

MK5: Kann man das überhaupt entscheiden?

Welche der folgenden Probleme sind unentscheidbar? Begründen Sie Ihre Antwort.

1. Gegeben eine Turing-Maschine $\mathcal{M}_1$, wird $L(\mathcal{M}_1)$ von einer deterministischen 2-Band Turing-Maschine entschieden?
2. Gegeben eine Turing-Maschine $\mathcal{M}_2$ und ein Wort w, hat die Turing-Maschine nach maximal $23 \cdot |w|$ Schritten jedes Zeichen des Eingabewortes w mindestens einmal gelesen?
3. Gegeben eine Turing-Maschine $\mathcal{M}_3$ und eine natürliche Zahl $n \in \mathbb{N}$, akzeptiert $\mathcal{M}_3$ nur Wörter der Länge $\leqslant n$?

MK5: Kann man das überhaupt entscheiden?

Welche der folgenden Probleme sind unentscheidbar? Begründen Sie Ihre Antwort.

1. Gegeben eine Turing-Maschine $\mathcal{M}_1$, wird $L(\mathcal{M}_1)$ von einer deterministischen 2-Band Turing-Maschine entschieden?
2. Gegeben eine Turing-Maschine $\mathcal{M}_2$ und ein Wort w, hat die Turing-Maschine nach maximal $23 \cdot |w|$ Schritten jedes Zeichen des Eingabewortes w mindestens einmal gelesen?
3. Gegeben eine Turing-Maschine $\mathcal{M}_3$ und eine natürliche Zahl $n \in \mathbb{N}$, akzeptiert $\mathcal{M}_3$ nur Wörter der Länge $\leqslant n$?

MK5: Kann man das überhaupt entscheiden?

Welche der folgenden Probleme sind unentscheidbar? Begründen Sie Ihre Antwort.

1. Gegeben eine Turing-Maschine $\mathcal{M}_1$, wird $L(\mathcal{M}_1)$ von einer deterministischen 2-Band Turing-Maschine entschieden?
2. Gegeben eine Turing-Maschine $\mathcal{M}_2$ und ein Wort w, hat die Turing-Maschine nach maximal $23 \cdot |w|$ Schritten jedes Zeichen des Eingabewortes w mindestens einmal gelesen?
3. Gegeben eine Turing-Maschine $\mathcal{M}_3$ und eine natürliche Zahl $n \in \mathbb{N}$, akzeptiert $\mathcal{M}_3$ nur Wörter der Länge $\leqslant n$?

Dieses Problem ist unentscheidbar, denn es ist das Wortproblem der Sprache

$$K = \{\, \mathrm{enc}(\mathcal{M}_1) \mid L(\mathcal{M}_1) \text{ ist entscheidbar} \,\}$$

MK5: Kann man das überhaupt entscheiden?

Welche der folgenden Probleme sind unentscheidbar? Begründen Sie Ihre Antwort.

1. Gegeben eine Turing-Maschine $\mathcal{M}_1$, wird $L(\mathcal{M}_1)$ von einer deterministischen 2-Band Turing-Maschine entschieden?
2. Gegeben eine Turing-Maschine $\mathcal{M}_2$ und ein Wort w, hat die Turing-Maschine nach maximal $23 \cdot |w|$ Schritten jedes Zeichen des Eingabewortes w mindestens einmal gelesen?
3. Gegeben eine Turing-Maschine $\mathcal{M}_3$ und eine natürliche Zahl $n \in \mathbb{N}$, akzeptiert $\mathcal{M}_3$ nur Wörter der Länge $\leqslant n$?

Dieses Problem ist unentscheidbar, denn es ist das Wortproblem der Sprache

$$K = \{\, \mathrm{enc}(\mathcal{M}_1) \mid L(\mathcal{M}_1) \text{ ist entscheidbar} \,\}$$

Entscheidbarkeit ist eine nicht-triviale Eigenschaft formaler Sprachen.

MK5: Kann man das überhaupt entscheiden?

Welche der folgenden Probleme sind unentscheidbar? Begründen Sie Ihre Antwort.

1. Gegeben eine Turing-Maschine $\mathcal{M}_1$, wird $L(\mathcal{M}_1)$ von einer deterministischen 2-Band Turing-Maschine entschieden?
2. Gegeben eine Turing-Maschine $\mathcal{M}_2$ und ein Wort w, hat die Turing-Maschine nach maximal $23 \cdot |w|$ Schritten jedes Zeichen des Eingabewortes w mindestens einmal gelesen?
3. Gegeben eine Turing-Maschine $\mathcal{M}_3$ und eine natürliche Zahl $n \in \mathbb{N}$, akzeptiert $\mathcal{M}_3$ nur Wörter der Länge $\leqslant n$?

Dieses Problem ist unentscheidbar, denn es ist das Wortproblem der Sprache

$$K = \{\, \mathrm{enc}(\mathcal{M}_1) \mid L(\mathcal{M}_1) \text{ ist entscheidbar} \,\}$$

Entscheidbarkeit ist eine nicht-triviale Eigenschaft formaler Sprachen.
Nach dem Satz von Rice folgt die Unentscheidbarkeit.

MK5: Kann man das überhaupt entscheiden?

Welche der folgenden Probleme sind unentscheidbar? Begründen Sie Ihre Antwort.

1. Gegeben eine Turing-Maschine $\mathcal{M}_1$, wird $L(\mathcal{M}_1)$ von einer deterministischen 2-Band Turing-Maschine entschieden?
2. Gegeben eine Turing-Maschine $\mathcal{M}_2$ und ein Wort w, hat die Turing-Maschine nach maximal $23 \cdot |w|$ Schritten jedes Zeichen des Eingabewortes w mindestens einmal gelesen?
3. Gegeben eine Turing-Maschine $\mathcal{M}_3$ und eine natürliche Zahl $n \in \mathbb{N}$, akzeptiert $\mathcal{M}_3$ nur Wörter der Länge $\leqslant n$?

MK5: Kann man das überhaupt entscheiden?

Welche der folgenden Probleme sind unentscheidbar? Begründen Sie Ihre Antwort.

1. Gegeben eine Turing-Maschine $\mathcal{M}_1$, wird $L(\mathcal{M}_1)$ von einer deterministischen 2-Band Turing-Maschine entschieden?
2. Gegeben eine Turing-Maschine $\mathcal{M}_2$ und ein Wort w, hat die Turing-Maschine nach maximal $23 \cdot |w|$ Schritten jedes Zeichen des Eingabewortes w mindestens einmal gelesen?
3. Gegeben eine Turing-Maschine $\mathcal{M}_3$ und eine natürliche Zahl $n \in \mathbb{N}$, akzeptiert $\mathcal{M}_3$ nur Wörter der Länge $\leqslant n$?

Dieses Problem ist entscheidbar.

MK5: Kann man das überhaupt entscheiden?

Welche der folgenden Probleme sind unentscheidbar? Begründen Sie Ihre Antwort.

1. Gegeben eine Turing-Maschine $\mathcal{M}_1$, wird $L(\mathcal{M}_1)$ von einer deterministischen 2-Band Turing-Maschine entschieden?
2. Gegeben eine Turing-Maschine $\mathcal{M}_2$ und ein Wort w, hat die Turing-Maschine nach maximal $23 \cdot |w|$ Schritten jedes Zeichen des Eingabewortes w mindestens einmal gelesen?
3. Gegeben eine Turing-Maschine $\mathcal{M}_3$ und eine natürliche Zahl $n \in \mathbb{N}$, akzeptiert $\mathcal{M}_3$ nur Wörter der Länge $\leqslant n$?

Dieses Problem ist entscheidbar.

Gegeben eine Turing-Maschine $\mathcal{M}_2$ und ein Eingabewort w:

simuliere $\mathcal{M}_2$ auf w für höchstens $23 \cdot |w|$ Schritte und

prüfe, ob der Lesekopf alle Zeichen von w besucht hat.

MK5: Kann man das überhaupt entscheiden?

Welche der folgenden Probleme sind unentscheidbar? Begründen Sie Ihre Antwort.

1. Gegeben eine Turing-Maschine $\mathcal{M}_1$, wird $L(\mathcal{M}_1)$ von einer deterministischen 2-Band Turing-Maschine entschieden?
2. Gegeben eine Turing-Maschine $\mathcal{M}_2$ und ein Wort w, hat die Turing-Maschine nach maximal $23 \cdot |w|$ Schritten jedes Zeichen des Eingabewortes w mindestens einmal gelesen?
3. Gegeben eine Turing-Maschine $\mathcal{M}_3$ und eine natürliche Zahl $n \in \mathbb{N}$, akzeptiert $\mathcal{M}_3$ nur Wörter der Länge $\leqslant n$?

MK5: Kann man das überhaupt entscheiden?

Welche der folgenden Probleme sind unentscheidbar? Begründen Sie Ihre Antwort.

1. Gegeben eine Turing-Maschine $\mathcal{M}_1$, wird $L(\mathcal{M}_1)$ von einer deterministischen 2-Band Turing-Maschine entschieden?
2. Gegeben eine Turing-Maschine $\mathcal{M}_2$ und ein Wort w, hat die Turing-Maschine nach maximal $23 \cdot |w|$ Schritten jedes Zeichen des Eingabewortes w mindestens einmal gelesen?
3. Gegeben eine Turing-Maschine $\mathcal{M}_3$ und eine natürliche Zahl $n \in \mathbb{N}$, akzeptiert $\mathcal{M}_3$ nur Wörter der Länge $\leqslant n$?

Dieses Problem ist unentscheidbar, und zwar bereits für jede feste Wahl von n.

MK5: Kann man das überhaupt entscheiden?

Welche der folgenden Probleme sind unentscheidbar? Begründen Sie Ihre Antwort.

1. Gegeben eine Turing-Maschine $\mathcal{M}_1$, wird $L(\mathcal{M}_1)$ von einer deterministischen 2-Band Turing-Maschine entschieden?
2. Gegeben eine Turing-Maschine $\mathcal{M}_2$ und ein Wort w, hat die Turing-Maschine nach maximal $23 \cdot |w|$ Schritten jedes Zeichen des Eingabewortes w mindestens einmal gelesen?
3. Gegeben eine Turing-Maschine $\mathcal{M}_3$ und eine natürliche Zahl $n \in \mathbb{N}$, akzeptiert $\mathcal{M}_3$ nur Wörter der Länge $\leqslant n$?

Dieses Problem ist unentscheidbar, und zwar bereits für jede feste Wahl von n.

Für $n = 2$ ist die Menge

$$M_2 = \{\, \mathrm{enc}(\mathcal{M}_3) \mid L(\mathcal{M}_3) \subseteq \Sigma^{\leqslant 2} \,\}$$

unentscheidbar (Satz von Rice).

MK6: Komplexitätsklassen

1. Definieren Sie die Komplexitätsklasse $\mathrm{ExpTime}$. Dabei dürfen Sie die Begriffe „DTM“, „zeitbeschränkt“, und „DTime“ als bekannt voraussetzen.
2. Zeigen Sie mit einem direkten Beweis $\mathrm{NP} \subseteq \mathrm{ExpTime}$, ohne dabei eine der bekannten Inklusionen zwischen Komplexitätsklassen

   $$\mathrm{NP} \subseteq \mathrm{PSpace} \subseteq \mathrm{ExpTime}$$

   zu benutzen. Ausnahme: Sie dürfen die Beziehung $\mathrm{NL} \subseteq \mathrm{P}$ verwenden.
3. Sei $\mathbf{A} \in \mathrm{NPSpace}$, $\mathbf{B} \in \mathrm{ExpTime}$. Zeigen Sie: $\overline{\mathbf{A}} \cap \mathbf{B} \in \mathrm{ExpTime}$.

MK6: Komplexitätsklassen

1. Definieren Sie die Komplexitätsklasse $\mathrm{ExpTime}$. Dabei dürfen Sie die Begriffe „DTM“, „zeitbeschränkt“, und „DTime“ als bekannt voraussetzen.

MK6: Komplexitätsklassen

1. Definieren Sie die Komplexitätsklasse $\mathrm{ExpTime}$. Dabei dürfen Sie die Begriffe „DTM“, „zeitbeschränkt“, und „DTime“ als bekannt voraussetzen.

Für eine Sprache **L** $\subseteq \Sigma^*$ gilt: **L** $\in \mathrm{ExpTime}$ genau dann, wenn es eine deterministische Turingmaschine $\mathcal{M}$ gibt, die **L** in exponentiell beschränkter Zeit (in der Größe von **L**) entscheidet.

MK6: Komplexitätsklassen (Beweis für $\mathrm{NP} \subseteq \mathrm{ExpTime}$)

MK6: Komplexitätsklassen (Beweis für $\mathrm{NP} \subseteq \mathrm{ExpTime}$)

Seien **L** $\in \mathrm{NP}$, $w \in \Sigma^*$, und $\mathcal{M} = \langle Q, \Sigma, \Gamma, \delta, q_0, Q_F \rangle$ die NTM, die **L** entscheidet.

ExpTime-Entscheidungverfahren für **L**:

1. Konstruktion des *Konfigurationsgraphen* $G_{\mathcal{M}}$ von $\mathcal{M}$ (exponentielle Zeit)
2. Erreichbarkeitsanalyse in $G_{\mathcal{M}}$ (polynomielle Zeit)

MK6: Komplexitätsklassen (Beweis für $\mathrm{NP} \subseteq \mathrm{ExpTime}$)

Seien **L** $\in \mathrm{NP}$, $w \in \Sigma^*$, und $\mathcal{M} = \langle Q, \Sigma, \Gamma, \delta, q_0, Q_F \rangle$ die NTM, die **L** entscheidet.

ExpTime-Entscheidungverfahren für **L**:

1. Konstruktion des *Konfigurationsgraphen* $G_{\mathcal{M}}$ von $\mathcal{M}$ (exponentielle Zeit)
2. Erreichbarkeitsanalyse in $G_{\mathcal{M}}$ (polynomielle Zeit)

Insgesamt ergibt sich also ein ExpTime-Entscheidungsverfahren für **L**, denn es gibt eine erreichbare Konfiguration $c = \langle q, \cdot, \cdot \rangle$ mit $q \in Q_F$ gdw. $w \in$ **L**.

MK6: Komplexitätsklassen (Beweis für $\mathrm{NP} \subseteq \mathrm{ExpTime}$)

Seien **L** $\in \mathrm{NP}$, $w \in \Sigma^*$, und $\mathcal{M} = \langle Q, \Sigma, \Gamma, \delta, q_0, Q_F \rangle$ die NTM, die **L** entscheidet.

ExpTime-Entscheidungverfahren für **L**:

1. Konstruktion des *Konfigurationsgraphen* $G_{\mathcal{M}}$ von $\mathcal{M}$ (exponentielle Zeit)
2. Erreichbarkeitsanalyse in $G_{\mathcal{M}}$ (polynomielle Zeit)

MK6: Komplexitätsklassen (Beweis für $\mathrm{NP} \subseteq \mathrm{ExpTime}$)

Seien **L** $\in \mathrm{NP}$, $w \in \Sigma^*$, und $\mathcal{M} = \langle Q, \Sigma, \Gamma, \delta, q_0, Q_F \rangle$ die NTM, die **L** entscheidet.

ExpTime-Entscheidungverfahren für **L**:

1. Konstruktion des *Konfigurationsgraphen* $G_{\mathcal{M}}$ von $\mathcal{M}$ (exponentielle Zeit)
2. Erreichbarkeitsanalyse in $G_{\mathcal{M}}$ (polynomielle Zeit)

Zur **Größe/Konstruktion** von $G_{\mathcal{M}}$:

$\mathcal{M}$ ist polynomiell zeitbeschränkt

MK6: Komplexitätsklassen (Beweis für $\mathrm{NP} \subseteq \mathrm{ExpTime}$)

Seien **L** $\in \mathrm{NP}$, $w \in \Sigma^*$, und $\mathcal{M} = \langle Q, \Sigma, \Gamma, \delta, q_0, Q_F \rangle$ die NTM, die **L** entscheidet.

ExpTime-Entscheidungverfahren für **L**:

1. Konstruktion des *Konfigurationsgraphen* $G_{\mathcal{M}}$ von $\mathcal{M}$ (exponentielle Zeit)
2. Erreichbarkeitsanalyse in $G_{\mathcal{M}}$ (polynomielle Zeit)

Zur **Größe/Konstruktion** von $G_{\mathcal{M}}$:

$\mathcal{M}$ ist polynomiell zeitbeschränkt

$\Rightarrow$ $\mathcal{M}$ kann nur polynomiell viele Bandpositionen $1, \ldots, \ell$ benutzen

MK6: Komplexitätsklassen (Beweis für $\mathrm{NP} \subseteq \mathrm{ExpTime}$)

Seien **L** $\in \mathrm{NP}$, $w \in \Sigma^*$, und $\mathcal{M} = \langle Q, \Sigma, \Gamma, \delta, q_0, Q_F \rangle$ die NTM, die **L** entscheidet.

ExpTime-Entscheidungverfahren für **L**:

1. Konstruktion des *Konfigurationsgraphen* $G_{\mathcal{M}}$ von $\mathcal{M}$ (exponentielle Zeit)
2. Erreichbarkeitsanalyse in $G_{\mathcal{M}}$ (polynomielle Zeit)

Zur **Größe/Konstruktion** von $G_{\mathcal{M}}$:

$\mathcal{M}$ ist polynomiell zeitbeschränkt

$\Rightarrow$ $\mathcal{M}$ kann nur polynomiell viele Bandpositionen $1, \ldots, \ell$ benutzen

$\Rightarrow$ insgesamt exponentiell viele mögliche Bandinhalte ($|\Gamma|^{\ell}$)

MK6: Komplexitätsklassen (Beweis für $\mathrm{NP} \subseteq \mathrm{ExpTime}$)

Seien **L** $\in \mathrm{NP}$, $w \in \Sigma^*$, und $\mathcal{M} = \langle Q, \Sigma, \Gamma, \delta, q_0, Q_F \rangle$ die NTM, die **L** entscheidet.

ExpTime-Entscheidungverfahren für **L**:

1. Konstruktion des *Konfigurationsgraphen* $G_{\mathcal{M}}$ von $\mathcal{M}$ (exponentielle Zeit)
2. Erreichbarkeitsanalyse in $G_{\mathcal{M}}$ (polynomielle Zeit)

Zur **Größe/Konstruktion** von $G_{\mathcal{M}}$:

$\mathcal{M}$ ist polynomiell zeitbeschränkt

$\Rightarrow$ $\mathcal{M}$ kann nur polynomiell viele Bandpositionen $1, \ldots, \ell$ benutzen

$\Rightarrow$ insgesamt exponentiell viele mögliche Bandinhalte ($|\Gamma|^{\ell}$)

$\Rightarrow$ $|Q| \cdot |\Gamma|^{\ell} \cdot |\ell|$ mögliche Konfigurationen und
$(|Q| \cdot |\Gamma|^{\ell} \cdot |\ell|)^2 = (|Q|^2 \cdot |\Gamma|^{2\ell} \cdot |\ell|^2)$ mögliche Konfigurationswechsel in $G_{\mathcal{M}}$

MK6: Komplexitätsklassen (Beweis für $\mathrm{NP} \subseteq \mathrm{ExpTime}$)

Seien **L** $\in \mathrm{NP}$, $w \in \Sigma^*$, und $\mathcal{M} = \langle Q, \Sigma, \Gamma, \delta, q_0, Q_F \rangle$ die NTM, die **L** entscheidet.

ExpTime-Entscheidungverfahren für **L**:

1. Konstruktion des *Konfigurationsgraphen* $G_{\mathcal{M}}$ von $\mathcal{M}$ (exponentielle Zeit)
2. Erreichbarkeitsanalyse in $G_{\mathcal{M}}$ (polynomielle Zeit)

Zur **Größe/Konstruktion** von $G_{\mathcal{M}}$:

$\mathcal{M}$ ist polynomiell zeitbeschränkt

$\Rightarrow$ $\mathcal{M}$ kann nur polynomiell viele Bandpositionen $1, \ldots, \ell$ benutzen

$\Rightarrow$ insgesamt exponentiell viele mögliche Bandinhalte ($|\Gamma|^\ell$)

$\Rightarrow$ $|Q| \cdot |\Gamma|^\ell \cdot |\ell|$ mögliche Konfigurationen und
$(|Q| \cdot |\Gamma|^\ell \cdot |\ell|)^2 = (|Q|^2 \cdot |\Gamma|^{2\ell} \cdot |\ell|^2)$ mögliche Konfigurationswechsel in $G_{\mathcal{M}}$

Erreichbarkeitsanalyse in $G_{\mathcal{M}}$:

sei s die (eindeutige) Startkonfiguration in $G_{\mathcal{M}}$

MK6: Komplexitätsklassen (Beweis für $\mathrm{NP} \subseteq \mathrm{ExpTime}$)

Seien **L** $\in \mathrm{NP}$, $w \in \Sigma^*$, und $\mathcal{M} = \langle Q, \Sigma, \Gamma, \delta, q_0, Q_F \rangle$ die NTM, die **L** entscheidet.

ExpTime-Entscheidungverfahren für **L**:

1. Konstruktion des *Konfigurationsgraphen* $G_{\mathcal{M}}$ von $\mathcal{M}$ (exponentielle Zeit)
2. Erreichbarkeitsanalyse in $G_{\mathcal{M}}$ (polynomielle Zeit)

Zur **Größe/Konstruktion** von $G_{\mathcal{M}}$:

$\mathcal{M}$ ist polynomiell zeitbeschränkt

$\Rightarrow$ $\mathcal{M}$ kann nur polynomiell viele Bandpositionen $1, \ldots, \ell$ benutzen

$\Rightarrow$ insgesamt exponentiell viele mögliche Bandinhalte ($|\Gamma|^{\ell}$)

$\Rightarrow$ $|Q| \cdot |\Gamma|^{\ell} \cdot |\ell|$ mögliche Konfigurationen und
$(|Q| \cdot |\Gamma|^{\ell} \cdot |\ell|)^2 = (|Q|^2 \cdot |\Gamma|^{2\ell} \cdot |\ell|^2)$ mögliche Konfigurationswechsel in $G_{\mathcal{M}}$

Erreichbarkeitsanalyse in $G_{\mathcal{M}}$:

sei s die (eindeutige) Startkonfiguration in $G_{\mathcal{M}}$

Erreichbarkeit einer akzeptierenden Konfiguration ($\mathrm{NL} \subseteq \mathrm{P} \subseteq \mathrm{ExpTime}$)

MK6: Komplexitätsklassen

3. Sei $\mathbf{A} \in \mathrm{NPSpace}$, $\mathbf{B} \in \mathrm{ExpTime}$. Zeigen Sie: $\overline{\mathbf{A}} \cap \mathbf{B} \in \mathrm{ExpTime}$.

MK6: Komplexitätsklassen

3. Sei $\mathbf{A} \in \mathrm{NPSPACE}$, $\mathbf{B} \in \mathrm{EXPTIME}$. Zeigen Sie: $\overline{\mathbf{A}} \cap \mathbf{B} \in \mathrm{EXPTIME}$.

Zunächst $\mathbf{A} \in \mathrm{EXPTIME}$:

es gilt $\mathbf{A} \in \mathrm{NPSPACE} = \mathrm{PSPACE}$ (Satz von Savitch)

wegen $\mathrm{PSPACE} \subseteq \mathrm{EXPTIME}$, ist auch $\mathbf{A} \in \mathrm{EXPTIME}$

MK6: Komplexitätsklassen

3. Sei $\mathbf{A} \in \mathrm{NPSPACE}$, $\mathbf{B} \in \mathrm{EXPTIME}$. Zeigen Sie: $\overline{\mathbf{A}} \cap \mathbf{B} \in \mathrm{EXPTIME}$.

Zunächst $\mathbf{A} \in \mathrm{EXPTIME}$:

es gilt $\mathbf{A} \in \mathrm{NPSPACE} = \mathrm{PSPACE}$ (Satz von Savitch)

wegen $\mathrm{PSPACE} \subseteq \mathrm{EXPTIME}$, ist auch $\mathbf{A} \in \mathrm{EXPTIME}$

Dann $\overline{\mathbf{A}} \in \mathrm{EXPTIME}$:

$\mathrm{EXPTIME}$ als deterministische Klasse ist abgeschlossen unter Komplementen

MK6: Komplexitätsklassen

3. Sei $\mathbf{A} \in \mathrm{NPSpace}$, $\mathbf{B} \in \mathrm{ExpTime}$. Zeigen Sie: $\overline{\mathbf{A}} \cap \mathbf{B} \in \mathrm{ExpTime}$.

Zunächst $\mathbf{A} \in \mathrm{ExpTime}$:

es gilt $\mathbf{A} \in \mathrm{NPSpace} = \mathrm{PSpace}$ (Satz von Savitch)

wegen $\mathrm{PSpace} \subseteq \mathrm{ExpTime}$, ist auch $\mathbf{A} \in \mathrm{ExpTime}$

Dann $\overline{\mathbf{A}} \in \mathrm{ExpTime}$:

$\mathrm{ExpTime}$ als deterministische Klasse ist abgeschlossen unter Komplementen

Schließlich $\overline{\mathbf{A}} \cap \mathbf{B} \in \mathrm{ExpTime}$:

Sind $\mathcal{M}_{\overline{\mathbf{A}}}$ und $\mathcal{M}_{\mathbf{B}}$ exponentiell zeitbeschränkte DTMs, die $\overline{\mathbf{A}}$ bzw. $\mathbf{B}$ entscheiden. Dann entscheiden wir $\overline{\mathbf{A}} \cap \mathbf{B}$ wie in exponentiell beschränkter Zeit wie folgt:

MK6: Komplexitätsklassen

3. Sei $\mathbf{A} \in \mathrm{NPSPACE}$, $\mathbf{B} \in \mathrm{EXPTIME}$. Zeigen Sie: $\overline{\mathbf{A}} \cap \mathbf{B} \in \mathrm{EXPTIME}$.

Zunächst $\mathbf{A} \in \mathrm{EXPTIME}$:

es gilt $\mathbf{A} \in \mathrm{NPSPACE} = \mathrm{PSPACE}$ (Satz von Savitch)

wegen $\mathrm{PSPACE} \subseteq \mathrm{EXPTIME}$, ist auch $\mathbf{A} \in \mathrm{EXPTIME}$

Dann $\overline{\mathbf{A}} \in \mathrm{EXPTIME}$:

$\mathrm{EXPTIME}$ als deterministische Klasse ist abgeschlossen unter Komplementen

Schließlich $\overline{\mathbf{A}} \cap \mathbf{B} \in \mathrm{EXPTIME}$:

Sind $\mathcal{M}_{\overline{\mathbf{A}}}$ und $\mathcal{M}_{\mathbf{B}}$ exponentiell zeitbeschränkte DTMs, die $\overline{\mathbf{A}}$ bzw. $\mathbf{B}$ entscheiden. Dann entscheiden wir $\overline{\mathbf{A}} \cap \mathbf{B}$ wie in exponentiell beschränkter Zeit wie folgt:

Für Eingabe *w*:

simuliere $\mathcal{M}_{\overline{\mathbf{A}}}$ auf *w*: falls $\mathcal{M}_{\overline{\mathbf{A}}}$ verwirft, verwerfe, andernfalls

MK6: Komplexitätsklassen

3. Sei $\mathbf{A} \in \mathrm{NPSpace}$, $\mathbf{B} \in \mathrm{ExpTime}$. Zeigen Sie: $\overline{\mathbf{A}} \cap \mathbf{B} \in \mathrm{ExpTime}$.

Zunächst $\mathbf{A} \in \mathrm{ExpTime}$:

es gilt $\mathbf{A} \in \mathrm{NPSpace} = \mathrm{PSpace}$ (Satz von Savitch)

wegen $\mathrm{PSpace} \subseteq \mathrm{ExpTime}$, ist auch $\mathbf{A} \in \mathrm{ExpTime}$

Dann $\overline{\mathbf{A}} \in \mathrm{ExpTime}$:

$\mathrm{ExpTime}$ als deterministische Klasse ist abgeschlossen unter Komplementen

Schließlich $\overline{\mathbf{A}} \cap \mathbf{B} \in \mathrm{ExpTime}$:

Sind $\mathcal{M}_{\overline{\mathbf{A}}}$ und $\mathcal{M}_{\mathbf{B}}$ exponentiell zeitbeschränkte DTMs, die $\overline{\mathbf{A}}$ bzw. $\mathbf{B}$ entscheiden. Dann entscheiden wir $\overline{\mathbf{A}} \cap \mathbf{B}$ wie in exponentiell beschränkter Zeit wie folgt:

Für Eingabe *w*:

simuliere $\mathcal{M}_{\overline{\mathbf{A}}}$ auf *w*: falls $\mathcal{M}_{\overline{\mathbf{A}}}$ verwirft, verwerfe, andernfalls

simuliere $\mathcal{M}_{\mathbf{B}}$ auf *w*: falls $\mathcal{M}_{\mathbf{B}}$ akzeptiert, akzeptiere, andernfalls verwerfe.

MK7: Datalog

Gegeben ist das folgende Datalog-Programm P:

$$
\begin{aligned}
L(x, y) &\leftarrow s(x, y) \\
L(x, z) &\leftarrow s(x, y) \wedge L(y, z) \\
Q(y) &\leftarrow t(x, y, z) \wedge L(y, x) \wedge L(y, z)
\end{aligned}
$$

1. Berechnen Sie für die folgenden Fakten schrittweise die Mengen $T_P^0, T_P^1, T_P^2, \ldots$. Wann wird der Fixpunkt erreicht?

$$s(0, 1), s(1, 2), s(2, 3), s(3, 4), t(1, 0, 2), t(3, 1, 4)$$

2. Geben Sie einen Ableitungsbaum für $Q(0)$ an.

MK7: Datalog

Gegeben ist das folgende Datalog-Programm P:

$$
\begin{aligned}
L(x, y) &\leftarrow s(x, y) \\
L(x, z) &\leftarrow s(x, y) \wedge L(y, z) \\
Q(y) &\leftarrow t(x, y, z) \wedge L(y, x) \wedge L(y, z)
\end{aligned}
$$

Fakten: $s(0, 1), s(1, 2), s(2, 3), s(3, 4), t(1, 0, 2), t(3, 1, 4)$.

MK7: Datalog

Gegeben ist das folgende Datalog-Programm P:

$$
\begin{aligned}
L(x, y) &\leftarrow s(x, y) \\
L(x, z) &\leftarrow s(x, y) \wedge L(y, z) \\
Q(y) &\leftarrow t(x, y, z) \wedge L(y, x) \wedge L(y, z)
\end{aligned}
$$

Fakten: $s(0,1), s(1,2), s(2,3), s(3,4), t(1,0,2), t(3,1,4)$.

$$
\begin{aligned}
T_P^0 &= \varnothing \\
T_P^1 &= \{s(0,1), s(1,2), s(2,3), s(3,4), t(1,0,2), t(3,1,4)\}
\end{aligned}
$$

MK7: Datalog

Gegeben ist das folgende Datalog-Programm P:

$$
\begin{aligned}
L(x,y) &\leftarrow s(x,y) \\
L(x,z) &\leftarrow s(x,y) \wedge L(y,z) \\
Q(y) &\leftarrow t(x,y,z) \wedge L(y,x) \wedge L(y,z)
\end{aligned}
$$

Fakten: $s(0,1), s(1,2), s(2,3), s(3,4), t(1,0,2), t(3,1,4)$.

$$
\begin{aligned}
T_P^0 &= \varnothing \\
T_P^1 &= \{s(0,1), s(1,2), s(2,3), s(3,4), t(1,0,2), t(3,1,4)\} \\
T_P^2 &= T_P^1 \cup \{L(0,1), L(1,2), L(2,3), L(3,4)\}
\end{aligned}
$$

MK7: Datalog

Gegeben ist das folgende Datalog-Programm P:

$$
\begin{aligned}
L(x,y) &\leftarrow s(x,y) \\
L(x,z) &\leftarrow s(x,y) \wedge L(y,z) \\
Q(y) &\leftarrow t(x,y,z) \wedge L(y,x) \wedge L(y,z)
\end{aligned}
$$

Fakten: $s(0,1), s(1,2), s(2,3), s(3,4), t(1,0,2), t(3,1,4)$.

$$
\begin{aligned}
T_P^0 &= \varnothing \\
T_P^1 &= \{s(0,1), s(1,2), s(2,3), s(3,4), t(1,0,2), t(3,1,4)\} \\
T_P^2 &= T_P^1 \cup \{L(0,1), L(1,2), L(2,3), L(3,4)\} \\
T_P^3 &= T_P^2 \cup \{L(0,2), L(1,3), L(2,4)\}
\end{aligned}
$$

MK7: Datalog

Gegeben ist das folgende Datalog-Programm P:

$$
\begin{aligned}
L(x,y) &\leftarrow s(x,y) \\
L(x,z) &\leftarrow s(x,y) \wedge L(y,z) \\
Q(y) &\leftarrow t(x,y,z) \wedge L(y,x) \wedge L(y,z)
\end{aligned}
$$

Fakten: $s(0,1), s(1,2), s(2,3), s(3,4), t(1,0,2), t(3,1,4)$.

$$
\begin{aligned}
T_P^0 &= \varnothing \\
T_P^1 &= \{s(0,1), s(1,2), s(2,3), s(3,4), t(1,0,2), t(3,1,4)\} \\
T_P^2 &= T_P^1 \cup \{L(0,1), L(1,2), L(2,3), L(3,4)\} \\
T_P^3 &= T_P^2 \cup \{L(0,2), L(1,3), L(2,4)\} \\
T_P^4 &= T_P^3 \cup \{L(0,3), L(1,4), Q(0)\}
\end{aligned}
$$

MK7: Datalog

Gegeben ist das folgende Datalog-Programm P:

$$
\begin{aligned}
L(x,y) &\leftarrow s(x,y) \\
L(x,z) &\leftarrow s(x,y) \wedge L(y,z) \\
Q(y) &\leftarrow t(x,y,z) \wedge L(y,x) \wedge L(y,z)
\end{aligned}
$$

Fakten: $s(0,1), s(1,2), s(2,3), s(3,4), t(1,0,2), t(3,1,4)$.

$$
\begin{aligned}
T_P^0 &= \varnothing \\
T_P^1 &= \{s(0,1), s(1,2), s(2,3), s(3,4), t(1,0,2), t(3,1,4)\} \\
T_P^2 &= T_P^1 \cup \{L(0,1), L(1,2), L(2,3), L(3,4)\} \\
T_P^3 &= T_P^2 \cup \{L(0,2), L(1,3), L(2,4)\} \\
T_P^4 &= T_P^3 \cup \{L(0,3), L(1,4), Q(0)\} \\
T_P^5 &= T_P^4 \cup \{L(0,4), Q(1)\}
\end{aligned}
$$

MK7: Datalog

Gegeben ist das folgende Datalog-Programm P:

$$
\begin{aligned}
L(x,y) &\leftarrow s(x,y) \\
L(x,z) &\leftarrow s(x,y) \wedge L(y,z) \\
Q(y) &\leftarrow t(x,y,z) \wedge L(y,x) \wedge L(y,z)
\end{aligned}
$$

Fakten: $s(0,1), s(1,2), s(2,3), s(3,4), t(1,0,2), t(3,1,4)$.

$$
\begin{aligned}
T_P^0 &= \varnothing \\
T_P^1 &= \{s(0,1), s(1,2), s(2,3), s(3,4), t(1,0,2), t(3,1,4)\} \\
T_P^2 &= T_P^1 \cup \{L(0,1), L(1,2), L(2,3), L(3,4)\} \\
T_P^3 &= T_P^2 \cup \{L(0,2), L(1,3), L(2,4)\} \\
T_P^4 &= T_P^3 \cup \{L(0,3), L(1,4), Q(0)\} \\
T_P^5 &= T_P^4 \cup \{L(0,4), Q(1)\} \\
T_P^6 &= T_P^5 = T_P^\infty
\end{aligned}
$$

MK7: Datalog

Gegeben ist das folgende Datalog-Programm P:

$$
\begin{aligned}
L(x, y) &\leftarrow s(x, y) \\
L(x, z) &\leftarrow s(x, y) \wedge L(y, z) \\
Q(y) &\leftarrow t(x, y, z) \wedge L(y, x) \wedge L(y, z)
\end{aligned}
$$

Fakten: $s(0,1), s(1,2), s(2,3), s(3,4), t(1,0,2), t(3,1,4)$.

2. Geben Sie einen Ableitungsbaum für $Q(0)$ an.

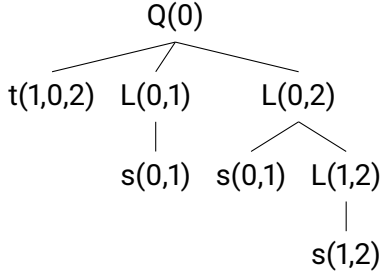
MK7: Datalog

Gegeben ist das folgende Datalog-Programm P:

$$
\begin{aligned}
L(x, y) &\leftarrow s(x, y) \\
L(x, z) &\leftarrow s(x, y) \wedge L(y, z) \\
Q(y) &\leftarrow t(x, y, z) \wedge L(y, x) \wedge L(y, z)
\end{aligned}
$$

Fakten: $s(0,1), s(1,2), s(2,3), s(3,4), t(1,0,2), t(3,1,4)$.

2. Geben Sie einen Ableitungsbaum für $Q(0)$ an.



MK8: Richtig oder Falsch?

Welche der folgenden Aussagen sind jeweils wahr oder falsch? Begründen Sie Ihre Antworten.

1. Sind **A**, **B** zwei formale Sprachen mit $\overline{\mathbf{A}} \leqslant_m \mathbf{B}$, dann gilt auch $\mathbf{A} \leqslant_m \overline{\mathbf{B}}$.
2. coNP ist das mengentheoretische Komplement von NP, d.h. $\mathrm{coNP} = \{L \subseteq \Sigma^* \mid L \notin \mathrm{NP}\}$.
3. Erfüllbarkeit prädikatenlogischer Formeln ist semi-entscheidbar, aber nicht entscheidbar.
4. Jede QBF-Formel ist auch eine prädikatenlogische Formel.
5. **SAT** ist in polynomieller Zeit auf **Clique** reduzierbar.
6. Das Resolutionsverfahren ist ein deterministisches, polynomiell zeitbeschränktes Verfahren, um Unerfüllbarkeit prädikatenlogischer Formeln zu entscheiden.
7. Es ist entscheidbar, ob eine gegebene Turing-Maschine die Kolmogorov-Komplexität des Eingabewortes berechnet.
8. Jede prädikatenlogische Formel mit Gleichheit ist semantisch äquivalent zu einer prädikatenlogischen Formel ohne Gleichheit.

MK8: Richtig oder Falsch?

1. Sind **A**, **B** zwei formale Sprachen mit $\overline{\mathbf{A}} \leqslant_m \mathbf{B}$, dann gilt auch $\mathbf{A} \leqslant_m \overline{\mathbf{B}}$.

MK8: Richtig oder Falsch?

1. Sind **A**, **B** zwei formale Sprachen mit $\overline{\mathbf{A}} \leqslant_m \mathbf{B}$, dann gilt auch $\mathbf{A} \leqslant_m \overline{\mathbf{B}}$.

Ja, da jede Many-One-Reduktion von $\overline{\mathbf{A}}$ auf **B** auch eine Many-One-Reduktion von **A** auf $\overline{\mathbf{B}}$ ist.

MK8: Richtig oder Falsch?

2. coNP ist das mengentheoretische Komplement von NP, d.h. $\mathrm{coNP} = \{L \subseteq \Sigma^* \mid L \notin \mathrm{NP}\}$.

MK8: Richtig oder Falsch?

2. coNP ist das mengentheoretische Komplement von NP, d.h. $\mathrm{coNP} = \{L \subseteq \Sigma^* \mid L \notin \mathrm{NP}\}$.

Nein, $\mathrm{coNP} = \{L \subseteq \Sigma^* \mid \Sigma^* \setminus L \in \mathrm{NP}\}$.

MK8: Richtig oder Falsch?

3. Erfüllbarkeit prädikatenlogischer Formeln ist semi-entscheidbar, aber nicht entscheidbar.

MK8: Richtig oder Falsch?

3. Erfüllbarkeit prädikatenlogischer Formeln ist semi-entscheidbar, aber nicht entscheidbar.

Nein, da sonst sowohl Erfüllbarkeit als auch Unerfüllbarkeit semi-entscheidbar und damit entscheidbar wären.

MK8: Richtig oder Falsch?

4. Jede QBF-Formel ist auch eine prädikatenlogische Formel.

MK8: Richtig oder Falsch?

4. Jede QBF-Formel ist auch eine prädikatenlogische Formel.

Nein, da Quantoren in QBF-Formeln über aussagenlogischen Variablen quantifizieren, und Quantoren in prädikatenlogischen Formeln über Objektvariablen quantifizieren.

MK8: Richtig oder Falsch?

5. **SAT** ist in polynomieller Zeit auf **Clique** reduzierbar.

MK8: Richtig oder Falsch?

5. **SAT** ist in polynomieller Zeit auf **Clique** reduzierbar.

Ja, da **SAT** $\in \mathrm{NP}$ und **Clique** eine NP-vollständige Sprache ist.

MK8: Richtig oder Falsch?

6. Das Resolutionsverfahren ist ein deterministisches, polynomiell zeitbeschränktes Verfahren, um Unerfüllbarkeit prädikatenlogischer Formeln zu entscheiden.

MK8: Richtig oder Falsch?

6. Das Resolutionsverfahren ist ein deterministisches, polynomiell zeitbeschränktes Verfahren, um Unerfüllbarkeit prädikatenlogischer Formeln zu entscheiden.

Nein, das Resolutionsverfahren muss bei erfüllbaren Klauseln nicht terminieren.

MK8: Richtig oder Falsch?

7. Es ist entscheidbar, ob eine gegebene Turing-Maschine die Kolmogorov-Komplexität des Eingabewortes berechnet.

MK8: Richtig oder Falsch?

7. Es ist entscheidbar, ob eine gegebene Turing-Maschine die Kolmogorov-Komplexität des Eingabewortes berechnet.

Ja, denn die Kolmogorov-Komplexität ist nicht berechenbar, und damit berechnet keine Turing-Maschine diese.

MK8: Richtig oder Falsch?

8. Jede prädikatenlogische Formel mit Gleichheit ist semantisch äquivalent zu einer prädikatenlogischen Formel ohne Gleichheit.

MK8: Richtig oder Falsch?

8. Jede prädikatenlogische Formel mit Gleichheit ist semantisch äquivalent zu einer prädikatenlogischen Formel ohne Gleichheit.

Nein, denn für jede erfüllbare prädikatenlogische Formel ohne Gleichheit gibt es ein abzählbares unendliches Modell (Satz von Löwenheim-Skolem), $\forall x, y.\, x \approx y$ hat jedoch nur Modelle der Größe 1.