Prof. Javier Esparza
Kevin Kappelmann, Simon Roßkopf, Lukas Stevens
Technische Universität München
Lehrstuhl für Theoretische Informatik

# Einführung in die Theoretische Informatik

Sommersemester 2023 – Hausaufgabenblatt 10

- Bitte beachten Sie, dass in dieser Vorlesung generell Antworten mit Begründung gefordert werden, solange die Aufgabe nicht explizit das Gegenteil sagt.
- Zum Bestehen diese Blattes müssen Sie 50% der Punkte erreichen.
- Update: es werden die Aufgaben 1 (a–d) korrigiert.

**Aufgabe H10.1.** (*Falsch/Wahr*) 4 Punkte

Entscheiden Sie ob die folgenden Aussagen wahr oder falsch sind. Begründen Sie ihre Behauptung. Ein ausführlicher formaler Beweis ist nicht gefordert, aber eine umfassende Begründung.

(a) Sei $L \subseteq \Sigma^*$ eine unentscheidbare Sprache, $w \in \Sigma^*$. $L$ lässt sich mittels $f : \Sigma^* \to \Sigma^*$ mit

$$f(x) = \begin{cases} w, \text{wenn } x \in L \vee x = w \\ x, \text{sonst} \end{cases}$$

auf $L \cup \{w\}$ reduzieren.

(b) Sei $L \subseteq \Sigma^*$ eine unentscheidbare Sprache, $w \in \Sigma^*$. $L \cup \{w\}$ ist unentscheidbar.

(c) Sei $\Sigma = \{a\}$. Alle Sprachen über $\Sigma$ sind entscheidbar.

(d) Sei $L^*$ eine entscheidbare Sprache. $L$ ist entscheidbar.

*Lösungsskizze.*

(a) Falsch, die Reduktion hat mehrere Probleme:

- Falls $w \notin L$ gilt $f(w) \in L \cup \{w\}$
- Aus der Definition von $f$ ist die Berechenbarkeit von $f$ nicht ersichtlich, da $x \in L$ nicht berechenbar ist.

(b) Wahr, wenn $w \in L$, ist $L \cup \{w\} = L$ und daher unentscheidbar. Wenn $w \notin L$, zeigen wir die Unentscheibarkeit per Reduktion. Hierfür benutzen wir die folgende Reduktionsfunktion

$$f(x) = \begin{cases} v, \text{wenn } x = w \\ x, \text{sonst} \end{cases}$$

wobei $v \neq w$ ein Wort nicht in $L$ ist. Ein solches Wort $v$ existiert. Beweis per Widerspruch: Nehme an, es gäbe kein solches $v$. Dann wäre $L \cup \{w\} = \Sigma^*$. Dann wäre aber $L$ entscheidbar (Zur Berechnung der charakteristischen Funktion prüfe ob die Eingabe $w$ ist, wenn ja gib 0, sonst 1, zurück). Widerspruch.

(c) Falsch, es gibt überabzählbar viele Sprachen über $\Sigma$, aber nur abzählbar viele entscheidbare Sprachen (da es nur abzählbar viele TMs gibt).

(d) Falsch. Gegenbeispiel: Wähle $\Sigma = \{a\}$, $L^* = \Sigma^*$. Sei $L'$ irgendeine unentscheidbare Menge über $\Sigma$ (siehe (c)). Setze $L = L' \cup \{a\}$. Nach (b) ist $L$ unentscheidbar und es gilt $L^* = \Sigma^*$.

**Aufgabe H10.2.** (*Hmm*) 4 Punkte

Für die folgenden Aufgaben dürfen Sie annehmen, dass das Halteproblem für WHILE- und GOTO-Programme auf Eingabe 0 unentscheidbar ist. Wir nummerieren die Anweisungen eines Programms fortlaufend, wobei wir die Bedingung eines IFs bzw. WHILEs als eigene Anweisung betrachten.

(a) Ist es entscheidbar, ob bei der Ausführung eines WHILE-Programms $P$ auf Eingabe 0 die zweite Anweisung mindestens einmal ausgeführt wird?

(b) Ist es entscheidbar, ob bei der Ausführung eines GOTO-Programms $P$ auf Eingabe 0 die zweite Anweisung mindestens einmal ausgeführt wird?

*Lösungsskizze.*

(a) Diese Eigenschaft ist entscheidbar. Wir machen eine Fallunterscheidung auf der ersten Anweisung des Programms. Wenn die erste Anweisung eine Zuweisung ist, dann wird die zweite Anweisung immer erreicht. Bei einem WHILE wird der Körper nie ausgeführt, da wir 0 als Eingabe annehmen. Die zweite Anweisung des Programms, welche sich im Körper des WHILEs befinden muss, wird dann nie ausgeführt. Bei einem IF ist die Bedingung immer war, und die erste Anweisung im THEN-Arm wird ausgeführt, welche die zweite Anweisung des gesamten Programms ist.

(b) Wir zeigen, dass diese Eigenschaft unentscheidbar ist. Für ein gegebenes Programm $P$ konstruieren wir das Programm `GOTO L3; L2 : x := x + 1; L3: P; GOTO L2` wobei wir $x_0$ mit $x$ abkürzen und $L2, L3$ neue Sprungmarker sind, die nicht von $P$ verwendet werden. Die zweite Anweisung `x := x + 1` wird genau dann mindestens einmal ausgeführt, wenn $P$ mit Eingabe 0 terminiert. Wenn wir also die gegebene Eigenschaft entscheiden könnten, dann könnten wir auch das Halteproblem für Eingabe 0 für GOTO-Programme entscheiden, welches jedoch unentscheidbar ist.

**Aufgabe H10.3.** (*Verwhilen*) 3 + 3 Punkte

Es seien $f, g : \mathbb{N} \to \{0, 1\}$ zwei berechenbare, partielle Funktionen. Wir definieren die Menge der Summen von Eingaben, auf denen $f$ und $g$ mit Ausgabe 1 terminieren, wie folgt:

$$S \coloneqq \{x + y \mid x, y \in \mathbb{N} \wedge f(x) = g(y) = 1\}.$$

(a) Zeigen Sie, dass die partielle Funktion $f + g$, definiert als

$$f + g(z) \coloneqq \begin{cases} 1, & \text{falls } z \in S \\ \bot, & \text{sonst} \end{cases},$$

ebenfalls berechenbar ist. Sie dürfen hierfür annehmen, dass sie Programme $P_f, P_g$ für $f, g$ in einer beliebigen, Turing-vollständigen Sprache besitzen. Erläutern Sie, wie Sie dann ein Programm $P_{f+g}$ für $f + g$ schreiben können.

(b) Lösen Sie nun die vorherige Aufgabe konkret für WHILE-Programme. D.h. gegeben WHILE-Programme $P_f, P_g$ für $f, g$, erstellen Sie ein neues WHILE-Programm $P_{f+g}$ für $f + g$. In ihrem Programm dürfen Sie bei Zuweisungen und in Bedingungen beliebige arithmetische Ausdrücke $(+, -, \leq)$ verwenden. Weiterhin sind alle aussagenlogischen Operatoren $(\wedge, \neg, \vee)$ und Gleichheit $(=)$ in Bedingungen erlaubt. *Tipp:* Verwenden Sie Korollar 5.26.

*Lösungsskizze.*

(a) Da $P_f$ bzw. $P_g$ unter Umständen für eine mögliche Zerlegung $x, y$ des Inputs $z = x + y$ nicht terminieren, für eine andere Zerlegung $x', y'$ hingegen schon, können wir die Programme nicht einfach auf jede mögliche Zerlegung laufen lassen. Die Idee ist nun, dass wir stattdessen die Programme $P_f, P_g$ stets nur für eine beschränkte Anzahl $k$ von Schritten für jede Zerlegung laufen lassen. Hierfür modifizieren wir die Programme, sodass Sie nach jeder Instruktion einen Zähler erhöhen und dann überprüfen, ob dieser Zähler bereits $k$ überschreitet. Sollte nach $k$ Schritten keiner der Zerlegungen bei beiden Programmen zum Erfolg führen, so erhöhen wir $k$ um eins und beginnen von vorne.

(b) Es seien $P_f, P_g$ WHILE-Programme für $f, g$. Nach Korollar 5.26 können wir o.B.d.A annehmen, dass $P_f =$ `WHILE` $C_f$ `DO` $B_f$ `END` und $P_g =$ `WHILE` $C_g$ `DO` $B_g$ `END` gilt, wobei $B_f, B_g$ schleifenfrei sind. Dies ermöglicht es uns, den Zähler $k$ aus vorheriger Idee einfach am Ende der While-Schleife zu erhöhen und in der While-Bedingung zu überprüfen.

Wir benennen die Variablen im Programm so um, dass es keine Überschneidungen zwischen $P_f$ und $P_g$ gibt. Weiterhin seien $i_f, i_g$ die Eingabevariablen von $P_f$ bzw. $P_g$ und $i_{f+g}$ die Eingabe des Programms. Analog seien $o_f, o_g$ die Ausgabevariablen von $P_f$ bzw. $P_g$ und $o_{f+g}$ die Ausgabe des Programms. Es seien $Z_f, Z_g$ Programme, die jeweils alle von $f, g$ verwendeten Variablen, mit Ausnahme des Inputs, auf 0 setzen.

```
1  WHILE o_{f+g} = 0 DO
2      i_f := 0
3      WHILE i_f ≤ i_{f+g} ∧ o_{f+g} = 0 DO
4          i_g := i_{f+g} - i_f
5          Z_f
6          WHILE C_f ∧ k_f ≤ k DO
7              B_f
8              k_f := k_f + 1
9          END
10         Z_g
11         WHILE C_g ∧ k_g ≤ k DO
12             B_g
13             k_g := k_g + 1
14         END
15         IF ¬C_f ∧ o_f = 1 ∧ ¬C_g ∧ o_g = 1 DO
16             // P_f und P_g akzeptieren ihre Eingaben nach k
                  Iterationen
17             o_{f+g} := 1
18         ELSE
19             o_{f+g} := 0
20         END
21         i_f := i_f + 1
22     END
23     k := k + 1
24 END
```

**Aufgabe H10.4.** (*Dr. Evilangelo*) 3 + 3 + 5 Punkte + Schokoladenpreis

Dr. Evilsparza hat es geschafft: seine Schleifomaten sind allmächtig und er kann sich endlich zur

Ruhe setzen. Am Freitagabend hat er nun gemütlich – wie man es als kulturaffiner Bösewicht im Ruhestand eben so tut – ZDF-aspekte geschaut. Dort ging es um zeitgenössische Kunst und die Ausaneindersetzung der Künstler:innen mit ihrer Vergangenheit. Das fand der Superschurke so inspirierend, dass er kurzum seine ehemalige Werkstube in ein Atelier verwandelte. Er möchte es den Künstler:innen auf aspekte gleichtun und sich kunstvoll mit seiner Vergangenheit als Maschinenbösewicht auseinandersetzen.

Doch so ganz scheint es mit der Kunst leider nicht zu funktionieren: es fehlt ihm die notwendige Technik. Da kommt ihm eine Idee: er entwickelt einfach eine Maschine, die das Malen für ihn übernimmt. Eine geniale Verbindung seiner Vergangenheit und seines modernen Ichs! Seine Erfindung nennt er den *Malomat*.

Ein *Malomat* $M$ ist ähnlich zu einer *deterministischen* Turingmaschine, besitzt allerdings anstatt eines unendlich langen, eindimensionalen Bandes eine unendlich große, zweidimensionale Leinwand und kann sich zusätzlich auf und ab bewegen, dafür aber nie stillstehen. Formal ist ein Malomat ein 4-Tupel $(Q, \Gamma, \delta, q_0)$, wobei

(a) $Q \neq \emptyset$ die endliche Zustandsmenge,

(b) $\Gamma \neq \emptyset$ die endliche Farbpalette,

(c) $\delta : Q \times \Gamma \to Q \times \Gamma \times \{\uparrow, \rightarrow, \downarrow, \leftarrow\}$ die Übergangsfunktion und

(d) $q_0 \in Q$ der Startzustand ist.

Analog zu deterministischen Turingmaschinen können Malomaten auch graphisch als gelabelte Graphen dargestellt werden (siehe Beispiel unten).

Ein Malomat erhält als Eingabe eine voreingefärbte Leinwand. Diese lässt sich durch eine Funktion $f : \mathbb{Z}^2 \to \Gamma$ darstellen. Eine Konfiguration eines Malomats ist dann ein Tripel $(q, p, f)$, wobei $q \in Q$ der aktuelle Zustand, $p \in \mathbb{Z}^2$ die aktuelle Position (des Kopfs/Pinsels) und $f : \mathbb{Z}^2 \to \Gamma$ die aktuelle Leinwand ist. Die Übergangsrelation $\to_M$ ist dann wie folgt definiert: Falls $\delta(q, f(x, y)) = (q', c, D)$, dann

$$(q, (x, y), f) \to_M \begin{cases} (q', (x, y+1), f'), & \text{falls } D = \uparrow \\ (q', (x+1, y), f'), & \text{falls } D = \rightarrow \\ (q', (x, y-1), f'), & \text{falls } D = \downarrow \\ (q', (x-1, y), f'), & \text{falls } D = \leftarrow \end{cases},$$

wobei

$$f'(x', y') \coloneqq \begin{cases} c, & \text{falls } x = x' \text{ und } y = y' \\ f(x', y'), & \text{sonst} \end{cases}.$$

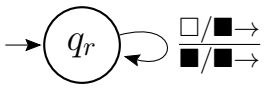
Die Initialposition eines Malomats ist dabei stets der Ursprung $(0, 0) \in \mathbb{Z}^2$. Beachten Sie: Malomaten sind fleißige Maler – sie halten nämlich nie.

Dr. Evilangelo hat nun verschiedene schwarz-weiß Gemälde für seine weißen Leinwände entworfen. Wir betrachen also die Farbpalette $\Gamma \coloneqq \{\square, \blacksquare\}$ (weiß und schwarz) und die weiße Eingabeleinwand $f_w : \mathbb{Z}^2 \to \Gamma, (x, y) \mapsto \square$. Wir sagen, dass ein *Malomat* $M$ *ein Gemälde* $G \subseteq \mathbb{Z}^2$ *malt*, falls $M$ mit Farbpalette $\Gamma$ bei Eingabe $f_w$

(a) jede beliebige, fixierte Position $(x, y) \in G$ nach endlich vielen Schritten schwarz einfärbt,

(b) keine schwarz eingefärbte Position jemals weiß übermalt und

(c) keine Position $(x, y) \in \mathbb{Z}^2 \setminus G$ jemals schwarz einfärbt.




**Beispiel:** Der folgende Malomat malt das Gemälde $\mathbb{N} \times \{0\}$.

Helft Dr. Evilangelo nun die folgenden Gemälde zu malen. Beschreibt für jede der folgenden Konstruktion auch eure Idee informell. Die Korrektheit der Konstruktionen müsst ihr nicht beweisen.

(a) Geben Sie einen Malomat *graphisch* an, der das Gemälde $G_1 \coloneqq \{(n, n) \mid n \in \mathbb{N} \text{ und } n \text{ ist gerade}\}$ malt. Ihr Malomat darf *maximal 5 Zustände* benutzen.

(b) Geben Sie einen Malomat *graphisch* an, der das Gemälde $G_2 \coloneqq \{(i, i) \mid i \in \mathbb{Z}\}$ malt. Ihr Malomat darf *maximal 6 Zustände* benutzen.

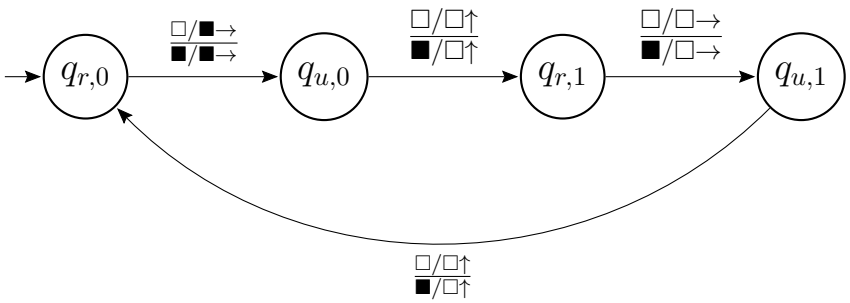
(c) Geben Sie einen Malomat *graphisch* an, der das Gemälde $V \coloneqq \mathbb{Z}^2$ malt. Ihr Malomat darf *maximal 15 Zustände* benutzen.

**Hinweis:** Ursprünglich wurde hier vom Gemälde $\mathbb{N}^2$ gesprochen. Dies war ein Fehler – Dr. Evilangelo möchte natürlich die ganze Leinwand und nicht nur einen Teil davon bemalt sehen. Lösungen für das Gemälde $\mathbb{N}^2$ werden für den Bonus berücksichtigt, sind allerdings vom folgenden Wettbewerb ausgeschlossen.

Dr. Evilangelos Helfer – Mr. Kevin Malomann – hat bereits eine Lösung mit 12 Zuständen gefunden. Dr. Evilangelo findet eine so große Anzahl an Zuständen jedoch höchst verschwenderisch. Seine Kunst soll nicht nur ansprechend, sondern auch nachhaltig sein. Er schreibt daher einen Preis aus: Die Künstlergruppe des Malomats mit der kleinsten Anzahl an Zuständen gewinnt einen Schokoladenpreis, überreicht in der berühmten Theovorlesung von Dr. Evilangelo. Wenn ihr teilnehmen möchtet, sendet eure Lösung per Email an theoleitung@in.tum.de, Betreff: Einreichung Malomat.

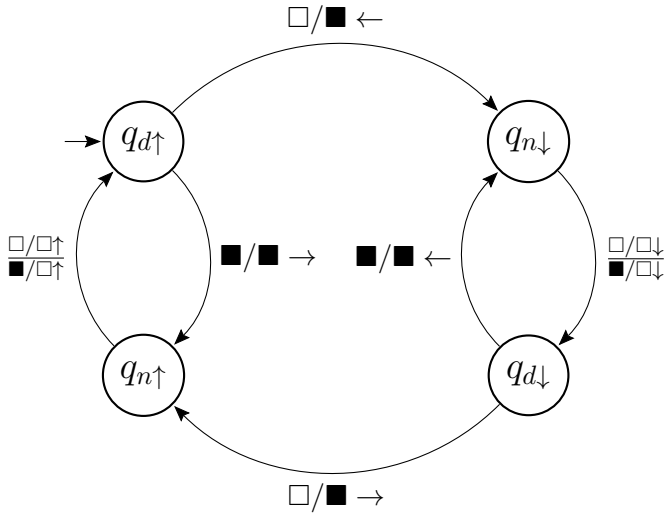
*Lösungsskizze.*

(a)



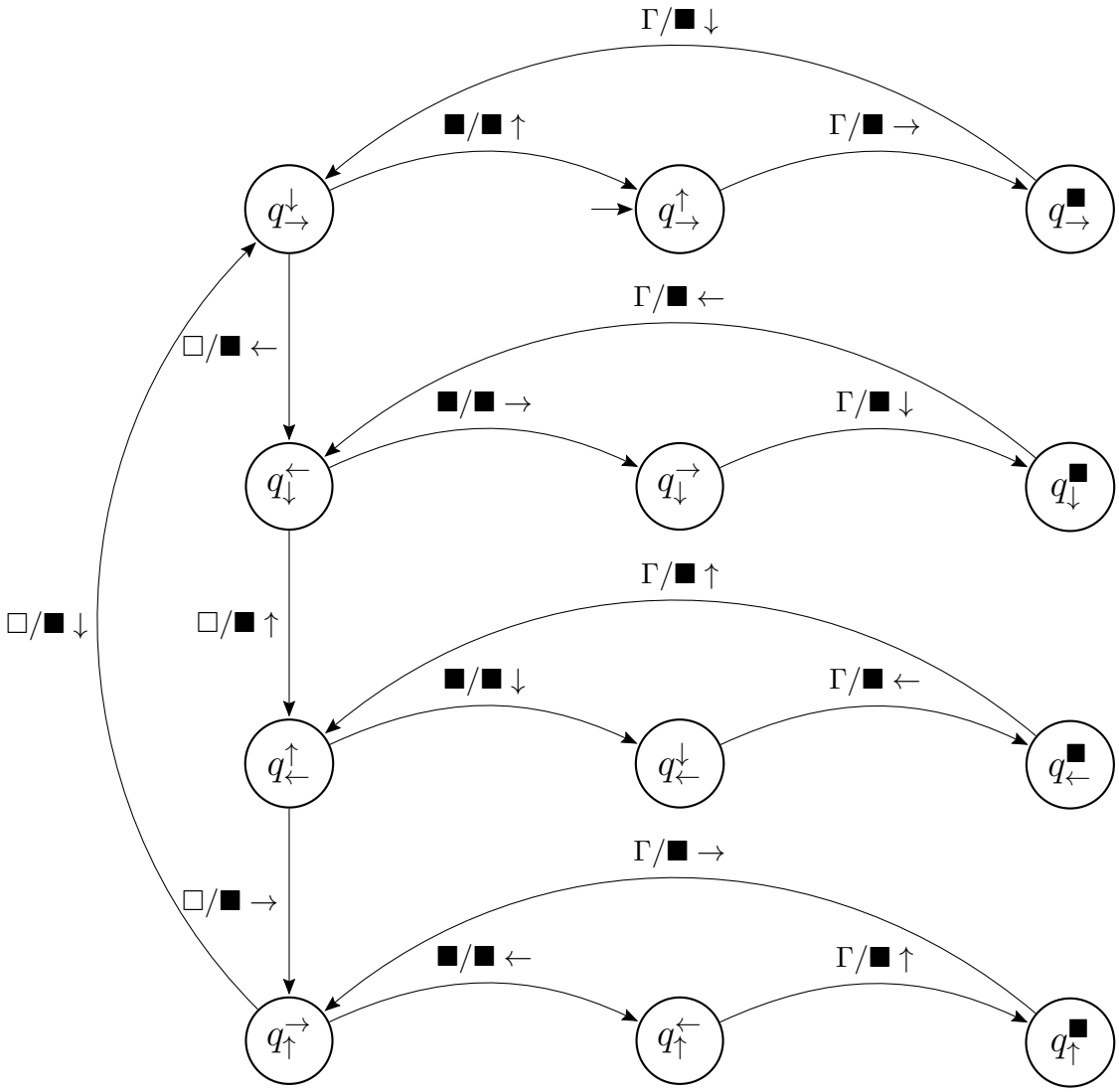

Idee: Es soll die Diagonale nach oben rechts mit geraden Koordinaten gemalt werden. Die Initialposition $(0, 0)$ muss schwarz eingefärbt werden. Danach muss der Pinsel sich vier Schritte bewegen (rechts, rauf, rechts, rauf), ohne die Leinwand schwarz zu färben. Anschließend kann der Vorgang von neuem gestartet werden.

(b)




Idee: Es soll die Diagonale nach unten links und oben rechts gemalt werden. Wir wandern abwechselnd die bereits gemalte Diagonale nach ganz oben und nach ganz unten und fügen dabei jeweils ein Kästchen am Ende hinzu. Im Zustand $q_{d\uparrow}$ befinden wir uns auf der Diagonalen und und wandern aktuell hoch. Sehen wir ein weißes Kästchen, so bemalen wir es und wechseln in den Wandermodus nach unten. Ansonsten wechseln wir in den Zustand $q_{n\uparrow}$ und wandern einen Schritt nach rechts. Im Zustand $q_{n\uparrow}$ befinden wir uns nicht auf der Diagonalen und wandern aktuell hoch. Wir bewegen uns einen Schritt nach oben, um zum nächsten Kästchen auf der Diagonalen zu gelangen und wechseln zurück in $q_{d\uparrow}$. Die Zustände $q_{d\downarrow}$ und $q_{n\downarrow}$ verhalten sich analog, nur wandern wir dabei die Diagonale nach unten.

(c)




Idee: Markiere zunächst den Ursprung und wander nach rechts. Dann wandere im Uhrzeigersinn spiralförmig um den bereits ausgemalten Block und erweitere ihn dabei. Jede Zeile des Automatens ist für einer dieser Richtungen zuständig. Für jede Richtung testen wir, ob wir noch in die aktuelle Richtung den Block erweitern können (es steht bereits ein schwarzes Kästchen in Richtung des Blocks), oder wir den Rand des Blockes im Bezug zur aktuellen Richtung erreicht haben (es steht ein weißes Kästchen in Richtung des Blocks) und deshalb in die nächste Richtung gewechselt werden muss.