Aufgaben aus den Übungsgruppen 8(Lösungsvorschläge)

1 Berechenbarkeitstheorie

Aufgabe 8.1 *(Wahr oder Falsch?)*
Sind die folgenden Aussagen wahr oder falsch? Begründen Sie!

1. Die berechenbaren Sprachen sind Teilmenge der semi-berechenbaren Sprachen. ■ wahr □ falsch
2. Wenn A berechenbar ist, dann ist auch $\bar{A}$ berechenbar. ■ wahr □ falsch
3. Wenn A semi-berechenbar ist, dann ist auch $\bar{A}$ semi-berechenbar. □ wahr ■ falsch
4. Wenn A und $\bar{A}$ semi-berechenbar sind, dann ist A berechenbar. ■ wahr □ falsch
5. Wenn A und $\bar{A}$ semi-berechenbar sind, dann ist $\bar{A}$ berechenbar. ■ wahr □ falsch
6. $A \preceq B$ bedeutet, dass B auf A reduzierbar ist. □ wahr ■ falsch
7. $A \preceq B$ bedeutet, dass A auf B reduzierbar ist. ■ wahr □ falsch
8. Sei A semi-entscheidbar und sei $A \preceq B$. Dann ist auch B semi-entscheidbar. □ wahr ■ falsch
9. Sei $A \preceq B$ und $B \preceq A$, dann ist $A = B$. □ wahr ■ falsch
10. Sei f eine Reduktion von A auf B. Dann ist f^{-1} eine gültige Reduktion von B auf A. □ wahr ■ falsch
11. Sei A semi-entscheidbar und sei $A \preceq B$. Dann ist B entscheidbar. □ wahr ■ falsch
12. Sei $A \preceq B$. Dann gilt auch $\bar{B} \preceq \bar{A}$. □ wahr ■ falsch
13. Sei A nicht entscheidbar und sei $A \preceq B$. Dann ist auch B nicht entscheidbar. ■ wahr □ falsch
14. Sei B nicht semi-entscheidbar und sei $A \preceq B$. Dann ist A nicht semi-entscheidbar. □ wahr ■ falsch
15. Sei $A \preceq B$. Dann gilt auch $\bar{A} \preceq \bar{B}$. ■ wahr □ falsch
16. Sei B semi-entscheidbar und sei $A \preceq B$. Dann ist auch A semi-entscheidbar. ■ wahr □ falsch
17. Sei $A \subseteq B$ und B entscheidbar ist, dann ist auch A entscheidbar. □ wahr ■ falsch
18. Sei $A \subseteq B$ und A unentscheidbar, dann ist auch B unentscheidbar. □ wahr ■ falsch

Aufgabe 8.2 *(Abgeschlossenheit unter Komplement)*
Seien A und B entscheidbar und seien C und D rekursiv aufzählbar. Zeigen oder widerlegen Sie:

1. $A \setminus B$ ist entscheidbar.
2. $C \setminus D$ ist rekursiv aufzählbar.

Lösungsvorschlag 8.2

1. Wahr. Wir wissen, dass A und B entscheidbar sind. Somit existieren Turingmaschinen M_1 und M_2 die χ_A und χ_B berechnen. Mit Hilfe dieser Turingmaschinen, konstruieren wir eine Turingmaschine M_3, die $A \setminus B$ entscheidet. Auf Input x simuliert M_3 zunächst M_1 auf x. Da M_1 χ_A berechnet, hält diese Simulation auf jeden Fall und M_1 gibt entweder 0 oder 1 zurück. Falls M_1 0 zurück gibt, gibt auch M_3 0 zurück. Andernfalls simuliert M_3 M_2 auf x. Auch diese Simulation hält auf jeden Fall. Gibt M_2 0 zurück, so gibt M_3 1 zurück. Andernfalls gibt M_3 0 zurück. Offensichtlich berechnet M_3 $\chi_{A \setminus B}$.
2. Falsch. Gegenbeispiel: $C = \{0,1\}^*$ und $D = H_0$. Aus der Vorlesung wissen wir, dass sowohl C als auch D rekursiv aufzählbar sind. Allerdings gilt $\{0,1\}^* \setminus H_0 = \bar{H}_0$ und wir wissen auch, dass $\bar{H}_0$ nicht rekursiv aufzählbar ist.

Aufgabe 8.3 *(Vereinigungen und Schnitte)*
Seinen A_0, A_1, A_2, ... entscheidbar und B_0, B_1, B_2, ... semi-entscheidbar. Zeigen oder widerlegen Sie:

1. $\forall t \in \mathbb{N} : \bigcap_{i=0}^{t} A_i$ ist berechenbar.
2. $\forall t \in \mathbb{N} : \bigcap_{i=0}^{t} B_i$ ist semi-berechenbar.
3. $\forall t \in \mathbb{N} : \bigcup_{i=0}^{t} A_i$ ist berechenbar.
4. $\forall t \in \mathbb{N} : \bigcup_{i=0}^{t} B_i$ ist semi-berechenbar.
5. $\bigcap_{i \in \mathbb{N}} A_i$ ist berechenbar.
6. $\bigcap_{i \in \mathbb{N}} B_i$ ist semi-berechenbar.

7. $\bigcup_{i \in \mathbb{N}} A_i$ ist berechenbar.
8. $\bigcup_{i \in \mathbb{N}} B_i$ ist semi-berechenbar.

Lösungsvorschlag 8.3

1. Wahr. Da die charakteristischen Funktionen aller A_i berechenbar sind, da alle A_i berechenbar sind, können wir die charakteristische Funktion des Schnittes berechnen: Wir berechnen für gegebenes x also alle ξ_{A_i} nacheinander und geben genau dann 1 zurück, wenn alle charakteristischen Funktionen 1 zurückgegeben haben.
2. Wahr. Wir berechnen jede positive charakteristische Funktion $\xi {B_i}^+$ und geben genau dann 1 zurück, wenn die Berechnung von allen $\xi {B_i}^+$ terminiert. Falls eines der $\xi {B_i}^+$ divergiert, dann divergiert auch die Berechnung der positiven charakteristischen Funktion des Schnitts.
3. Wahr. Da die charakteristischen Funktionen aller A_i berechenbar sind, da alle A_i berechenbar sind, können wir die charakteristische Funktion der Vereinigung berechnen: Wir berechnen für gegebenes x also alle ξ_{A_i} nacheinander und geben genau dann 1 zurück, wenn irgendeine charakteristischen Funktionen 1 zurückgegeben hat.
4. Wahr. Wir berechnen parallel alle $\xi {B_i}^+$ falls irgendwann für ein i 1 ausgegeben wird, terminieren wir und geben ebenfalls 1 aus. Andernfalls divergieren wir. Damit haben wir eine Berechnungsvorschrift für eine positive charakteristische Funktion auf der Vereinigung.
5. Falsch. Wir bauen uns entscheidbare A_i, die geschnitten mit dem Komplement des speziellen Halteproblems $\bar{H}_0$ ergeben:
$$A_i = \begin{cases} \mathbb{N} & \text{falls } i \in \bar{H}_0 \\ \mathbb{N} \setminus \{i\} & \text{falls } i \in \bar{H}_0 \end{cases}$$
Die A_i selbst sind entscheidbar: Die charakteristischen Funktionen sind entweder konstant 1 oder auf genau einem festen Wert 0 und sonst überall 1. (Man kann zwar nicht entscheiden, auf welchen i welcher Fall auftritt, aber beide Fälle, die auftreten können, sind entscheibar.)
Eine Zahl ist nun genau dann in $\bigcap_{i \in \mathbb{N}} A_i$ enthalten, wenn sie nirgends in einem A_i rausgeschnitten wurde, also genau dann, wenn sie in $\bar{H}_0$ liegt. Da aber $\bar{H}_0$ nicht berechenbar ist, ist dies ein Gegenbeispiel zur Behauptung.
6. Falsch. Das Beispiel aus der vorherigen Teilaufgabe ist auch hier ein Gegenbeispiel, da wenn A_i berechenbar ist, A_i auch semi-berechenbar sein muss und $\bar{H}_0$ auch nicht semi-berechenbar ist.
7. Falsch. Wir bauen uns entscheidbare A_i, die geschnitten $\bar{H}_0$ ergeben:
$$A_i = \begin{cases} \{i\} & \text{falls } i \in \bar{H}_0 \\ \emptyset & \text{falls } i \in \bar{H}_0 \end{cases}$$
Die A_i selbst sind entscheidbar: Die charakteristischen Funktionen sind entweder konstant 1 oder auf genau einem festen Wert 0 und sonst überall 1. (Man kann zwar nicht entscheiden, auf welchen i welcher Fall auftritt, aber beide Fälle, die auftreten können, sind entscheibar.) Eine Zahl ist nun genau dann in $\bigcup_{i \in \mathbb{N}} A_i$ enthalten, wenn sie in irgendeinem A_i vorkommt, also genau dann, wenn sie in $\bar{H}_0$ ist. Da aber $\bar{H}_0$ nicht entscheidbar ist, ist dies ein Gegenbeispeil zur Behauptung.
8. Falsch. Das Gegenbeispiel aus der vorherigen Teilaufgabe gilt auch hier.

Aufgabe 8.4 *(Reduktionen)*

Sei $\varphi_M : \{0,1\}^* \to \{0,1\}^*$ eine Funktion, die für ein $x \in \{0,1\}^*$ den Output der Turingmaschine M auf Input x berechnet. Hält M auf x nicht, so ist $\varphi_M(x)$ undefiniert. Seien die folgenden Probleme gegeben:

- $A = \{w \mid M_w \text{ hält auf } w\}$
- $B = \{w \mid M_w \text{ gibt für alle Inputs 0 aus}\}$
- $C = \left\{(w, w') \,\middle|\, im(\varphi_{M_w}) = im(\varphi_{M_{w'}})\right\}$
- $D = \{w \mid |im(\varphi_{M_w})| > 1\}$
- $E = \{w \mid M_w \text{ hält auf } w \text{ nicht oder } M_w \text{ gibt auf } w \text{ 0 aus}\}$
- $F = \left\{w \,\middle|\, \exists x \in \{0,1\}^*, \text{ sodass } M_w \text{ auf Input } x\ x \text{ ausgibt}\right\}$

Bearbeiten Sie nun die folgenden Aufgaben:

1. Zeigen Sie: $A \preceq B$
2. Zeigen Sie: $\bar{A} \preceq B$
3. Zeigen Sie: $B \preceq C$
4. Zeigen Sie: $A \preceq D$
5. Zeigen Sie: $E \preceq \bar{A}$
6. Zeigen Sie: $A \preceq F$
7. *Bonus:* Zeigen Sie: $F \preceq A$

Lösungsvorschlag 8.4

1. *Kommentar: A ist H_0, B ist V_0.*
   Wir konstruieren eine TM $M_{w'}$ wie folgt: $M_{w'}$ ignoriert ihren Input, simuliert M_w auf w und gibt 0 aus. Wir definieren $f(w) = w'$. Offensichtlich ist f berechenbar. Es bleibt zu zeigen, dass f tatsächlich die gewünschte Reduktionsfunktion darstellt.
   - Sei $w \in A$. Dann hält M_w auf Input w. Somit hält die Simulation in $M_{w'}$ immer und $M_{w'}$ gibt immer 0 aus. Also gilt $f(w) = w' \in B$.
   - Sei $w \notin A$. Dann hält M_w auf Input w nicht. Somit hält die Simulation in $M_{w'}$ nicht und daher hält auch $M_{w'}$ nicht. Insbesondere gibt $M_{w'}$ also auch nicht für alle Inputs 0 aus. Somit gilt $f(w) = w' \notin B$.
2. *Kommentar: A ist H_0, B ist V_0.*
   Wir konstruieren eine TM $M_{w'}$ wie folgt: Auf Input x simuliert $M_{w'}$ M_w auf w für x Schritte. Falls M_w in x Schritten nicht hält, gibt $M_{w'}$ 0 aus. Sonst gibt $M_{w'}$ 1 aus. Wir definieren $f(w) = w'$. Offensichtlich ist f berechenbar. Es bleibt zu zeigen, dass f tatsächlich die gewünschte Reduktionsfunktion darstellt.
   - Sei $w \in \bar{A}$. Dann hält M_w auf Input w niemals. Daher hält die Simulation von M_w auf w in $M_{w'}$ für alle Inputs von $M_{w'}$ nicht. Somit gibt $M_{w'}$ für alle Inputs 0 aus. Also gilt $f(w) = w' \in B$.
   - Sei $w \notin \bar{A}$. Dann hält M_w auf Input w. Somit existiert ein $t \in \mathbb{N}$, sodass M_w innerhalb von t Schritten auf w hält. Somit gibt $M_{w'}$ für alle Inputs $\geq t$ 1 aus. Daher gilt $f(w) = w' \notin B$.
3. *Kommentar: B ist V_0, C ist V.*
   Sei M_e die TM, die die konstante 0-Funktion berechnet. Wir definieren $f(w) = (w, e)$. Offensichtlich ist f berechenbar. Es bleibt zu zeigen, dass f tatsächlich die gewünschte Reduktionsfunktion darstellt.
   - Sei $w \in B$. Dann gibt M_w für alle Inputs 0 aus. Also gilt $im(\varphi_{M_w}) = \{0\} = im(\varphi_{M_e})$. Somit gilt $f(w) = (w, e) \in C$.
   - Sei $w \notin B$. Dann gibt es einen Input, sodass M_w nicht 0 ausgibt. Also gilt $im(\varphi_{M_w}) \neq \{0\} = im(\varphi_{M_e})$ und somit $f(w) = (w, e) \notin C$.
4. Wir konstruieren eine TM $M_{w'}$ wie folgt: Auf Input x überprüft $M_{w'}$, ob $w = x$ gilt. Falls ja, simuliert $M_{w'}$ M_w auf a und gibt anschließend 0 zurück. Andernfalls gibt $M_{w'}$ direkt 1 aus. Wir definieren $f(w) = w'$. Offensichtlich ist f berechenbar. Es bleibt zu zeigen, dass f tatsächlich die gewünschte Reduktionsfunktion darstellt.
   - Sei $w \in A$. Dann hält die Simulation von M_w auf w. Somit gibt $M_{w'}$ auf Input w 0 und für alle anderen Inputs 1 aus. Also $im(\varphi_{M_{w'}}) = \{0, 1\}$ und somit $|im(\varphi_{M_{w'}})| = 2 > 1$. Daher $f(w) = w' \in D$.
   - Sei $w \notin A$. Dann hält die Simulation von M_w auf w nicht. Somit divergiert $M_{w'}$ auf Input w und gibt für alle anderen Inputs 1 aus. Also $im(\varphi_{M_{w'}}) = \{1\}$ und somit $|im(\varphi_{M_{w'}})| = 1 \not> 1$. Daher $f(w) = w' \notin D$.
5. $A \preceq F$: Wir konstruieren eine TM $M_{w'}$ wie folgt: $M_{w'}$ ignoriert ihren Input, simuliert M_w auf w und speichert den Output von M_w in v. Falls $v = 0$, geht $M_{w'}$ in eine Endlosschleife. Andernfalls gibt $M_{w'}$ 1 aus. Wir definieren $f(w) = w'$. Offensichtlich ist f berechenbar. Es bleibt zu zeigen, dass f tatsächlich die gewünschte Reduktionsfunktion darstellt.
   - Sei $w \in E$. Dann gerät $M_{w'}$ in eine Endlosschleife (entweder weil die Simulation nicht hält oder weil M_w auf w 0 ausgegeben hat und $M_{w'}$ in eine Endlosschleife geht). Somit hält $M_{w'}$ insbesondere für Input w nicht. Also $f(w) = w' \in \bar{A}$.
   - Sei $w \notin E$. Dann hält M_w auf w und gibt etwas anderes als 0 aus. Somit gilt niemals $v = 0$ und daher gibt $M_{w'}$ immer 1 aus. Insbesondere hält $M_{w'}$ also immer und somit auch auf Input w. Daher gilt $f(w) = w' \notin \bar{A}$.
6. Wir konstruieren eine TM $M_{w'}$ wie folgt: Auf Input x simuliert $M_{w'}$ M_w auf w und gibt x aus. Wir definieren $f(w) = w'$. Offensichtlich ist f berechenbar. Es bleibt zu zeigen, dass f tatsächlich die gewünschte Reduktionsfunktion darstellt.
   - Sei $w \in A$. Dann hält M_w auf w und somit gibt $M_{w'}$ immer ihren Input wieder aus. Daher gilt $f(w) = w' \in F$.
   - Sei $w \notin A$. Dann hält M_w auf w und daher divergiert $M_{w'}$ für alle Inputs. Insbesondere gibt $M_{w'}$ also auch nicht ihren Input aus. Somit gilt $f(w) = w' \notin F$.
7. Wir konstruieren eine TM $M_{w'}$ wie folgt: Auf Input x zählt $M_{w'}$ „diagonal" Paare (t, y) auf. Für jedes dieser Paare simuliert $M_{w'}$ M_w auf y für t Schritte und speichert den Output in v. Falls $v = y$ gilt, dann

verlässt $M_{w'}$ die Aufzähl-Schleife. Wir definieren $f(w) = w'$. Offensichtlich ist f berechenbar. Es bleibt zu zeigen, dass f tatsächlich die gewünschte Reduktionsfunktion darstellt.

- Sei $w \in F$. Dann existiert ein x, sodass M_w auf Input x x ausgibt. Dann gibt es ein $t \in \mathbb{N}$, sodass M_w auf Input x in t Schritten hält und x ausgibt. Somit wird die Aufzähl-Schleife beim Paar (x, t) verlassen. Also hält $M_{w'}$ für alle Inputs und insbesondere auch für Input w. Somit gilt $f(w) = w' \in A$.
- Sei $w \notin F$. Dann gilt für alle möglichen Inputs x, dass M_w nicht x ausgibt. Somit wird die Aufzähl-Schleife niemals verlassen. Also hält $M_{w'}$ niemals und insbesondere auch nicht für Input w. Somit gilt $f(w) = w' \notin A$.

2 Komplexitätstheorie

Aufgabe 8.5 *(Determinismus vs. Nichtdeterminismus)*
Zeigen oder widerlegen Sie die folgende Aussagen:

1. $\mathsf{DSpace}(2^{O(n)}) = \mathsf{NSpace}(2^{O(n)})$
2. $\mathsf{DTime}(log\ n) = \mathsf{DSpace}(log\ n)$
3. $\mathsf{DTime}(2^{O(log\ n)}) = \mathsf{P}$
4. $\mathsf{NTime}(O(log\ n)) = \mathsf{DTime}(O(n^2))$

Lösungsvorschlag 8.5

1. Wahr. Denn es gilt $\mathsf{DSpace}(2^{O(n)}) \subseteq \mathsf{NSpace}(2^{O(n)})$ nach Satz C (VL 19, Folie 14). Ebenfalls aus Satz C folgt, dass $\mathsf{NSpace}(2^{O(n)}) \subseteq \mathsf{DSpace}(2^{O(n)^2})$. Es gilt: $\mathsf{DSpace}(2^{O(n)^2}) = \mathsf{DSpace}(2^{2 \cdot O(n)}) = \mathsf{DSpace}(2^{O(n)})$. Daher gilt $\mathsf{DSpace}(2^{O(n)}) \subseteq \mathsf{NSpace}(2^{O(n)})$ und $\mathsf{NSpace}(2^{O(n)}) \subseteq \mathsf{DSpace}(2^{O(n)})$ und somit $\mathsf{DSpace}(2^{O(n)}) = \mathsf{NSpace}(2^{O(n)})$.
2. Falsch. Zwar gilt $\mathsf{DTime}(log\ n) \subseteq \mathsf{DSpace}(log\ n)$ trivialerweise, aber $\mathsf{DSpace}(log\ n) \not\subseteq \mathsf{DTime}(log\ n)$ (Gegenbeispiel: Binärcounter. Geht in $\mathsf{DSpace}(log\ n)$, aber nicht in $\mathsf{DTime}(log\ n)$)
3. Wahr. Denn (c ist eine Konstante)
$$\mathsf{DTime}(2^{O(log\ n)}) \hat{=} \mathsf{DTime}(2^{c \cdot log\ n}) = \mathsf{DTime}(2^{(log\ n)^c}) = \mathsf{DTime}(n^c) = \bigcup_{c \in \mathbb{N}} \mathsf{DTime}(n^c) = \mathsf{P}$$
4. Falsch. Denn $L := \{w1 \mid w \in \{0,1\}^*\} \in \mathsf{DTime}(O(n^2))$. Jedoch gilt $L \notin \mathsf{NTime}(O(log\ n))$, denn eine $O(log\ n)$-zeitbeschränkte TM kann die Eingabe nicht vollständig lesen.

Aufgabe 8.6 *(Inklusionshierarchie)*
Sortieren Sie die folgenden Mengen bezüglich Inklusion: $\mathsf{NSpace}(O(3^n))$, P, $\mathsf{DSpace}(O(7^n))$, RE, NP, $\mathsf{DTime}(O(n^5))$, $\mathsf{DSpace}(O(logn))$, $\mathsf{NSpace}(O(log\ n))$, REC. Welche der Inklusionen sind echt?

Lösungsvorschlag 8.6

$$\begin{tikzcd}
 & \mathsf{DTime}(O(n^5)) \arrow[dr, phantom, "\subsetneq"] & & & \mathsf{DSpace}(O(7^n)) \arrow[dr, phantom, "\subsetneq"] & & \\
\mathsf{DSpace}(O(log\ n)) \arrow[r, phantom, "\subseteq"] & \mathsf{NSpace}(O(log\ n)) \arrow[r, phantom, "\subseteq"] & \mathsf{P} \arrow[r, phantom, "\subseteq"] & \mathsf{NP} \arrow[ur, phantom, "\subsetneq"] \arrow[dr, phantom, "\subsetneq"] & & \mathsf{REC} \arrow[r, phantom, "\subsetneq"] & \mathsf{RE} \\
 & & & & \mathsf{NSpace}(O(3^n)) \arrow[ur, phantom, "\subsetneq"] & &
\end{tikzcd}$$

Aufgabe 8.7 *(Wahr oder Falsch?)*
Sind die folgenden Aussagen wahr oder falsch? Begründen Sie!

1. Ist $\mathsf{P} = \mathsf{NP}$, dann ist jedes Problem in NP auch NP-schwer. ☐ wahr ■ falsch
2. Jede Sprache in NP ist entscheidbar. ■ wahr ☐ falsch
3. Ist $A \preceq_P B$ und $B \in \mathsf{NP}$, dann ist auch $A \in \mathsf{NP}$. ■ wahr ☐ falsch
4. Ist A NP-vollständig und $A \preceq_P B$, dann ist auch B NP-vollständig. ☐ wahr ■ falsch
5. Ist A NP-vollständig und $A \preceq_P B$, dann ist auch B NP-schwer. ■ wahr ☐ falsch
6. Es gibt eine Sprache L, sodass $L \in \mathsf{P} \ \wedge \ \bar{L} \notin \mathsf{P}$. ☐ wahr ■ falsch
7. Es gibt eine Funktion f, sodass $\mathsf{DSpace}(f) = REG$ (Menge aller reg. Sprachen) ■ wahr ☐ falsch

8. Es gibt eine Funktion f, sodass: $\mathsf{DSpace}(f) = \mathsf{REC}$ (Menge aller entsch. Sprachen) ☐ wahr ■ falsch
9. Jede Sprache in $\mathsf{DSpace}(O(1))$ ist regulär. ■ wahr ☐ falsch
10. Für alle Sprachen L, L' gilt: Wenn $L \subseteq L'$ und $L' \in \mathsf{NP}$, dann ist auch $L \in \mathsf{NP}$. ☐ wahr ■ falsch
11. Für alle Sprachen L, L' gilt: Wenn $L \subseteq L'$ und $L \in \mathsf{P}$, dann ist auch $L' \in \mathsf{P}$. ☐ wahr ■ falsch
12. Wenn $L \in \mathsf{NP}$, dann ist L auch entscheidbar. ■ wahr ☐ falsch

Aufgabe 8.8 *(Polynomialzeit-Reduktionen)*
Sei DNF die Menge aller aussagenlogischer Formeln in disjunktiver Normalform. Betrachten Sie das folgende Problem:

$$L = \{F \in DNF \mid F \text{ is keine Tautologie}\}$$

Zeigen Sie, dass L NP-schwer ist.

Lösungsvorschlag 8.8
Wir zeigen $SAT \preceq_P L$. Die Grundidee ist es, die Regel von *De Morgan* auszunutzen, denn negiert man eine Formel in CNF, so erhält man eine Formel in DNF. Offensichtlich gilt außerdem: F ist unerfüllbar $\Leftrightarrow$ $\neg F$ ist eine Tautologie.
Sei F in CNF. Konstruiere eine Formel F' wie folgt: Ersetze alle Variablen durch ihre Negation, ersetze alle $\wedge$ durch $\vee$ und alle $\vee$ durch $\wedge$ (letztendlich gilt also $F' = \neg F$, wobei die Regel von *De Morgan* erschöpfend angewendet wurde). Wir definieren $f(F) = F'$. Offensichtlich ist f in polynomieller Zeit berechenbar.
Es bleibt zu zeigen, dass f die gewünschte Reduktionsfunktion ist:

- Sei $F \in SAT$. Dann enthält jede Klausel mindestens ein wahres Literal. Somit hat jeder Konjunktionsterm von F' mindestens ein falsches Literal. Offensichtlich ist F' in DNF. Also gilt $F' \in L$.
- Sei $F \notin SAT$. Dann gibt es mindestens eine Klausel, die kein wahres Literal enthält. Somit gibt es in F' mindestens einen Konjunktionsterm, der nur wahre Literale enthält. Da alle Konjunktionsterme durch Disjunktionen verbunden werden, ist F' eine Tautologie. Somit gilt $F' \notin L$.

Aufgabe 8.9 *(P vs. NP)*
Sie und ihr Freund haben sich im Rahmen Ihrer Klausurvorbereitung mit einer recht komplizierten Sprache L beschäftigt. Ihnen selbst ist es gelungen zu zeigen, dass $L \in \mathsf{NP}$ gilt. In der Zeit ist es Ihrem Freund gelungen folgende Sprache $L' = \{w \mid M_w \textit{ berechnet eine lineare Funktion}\}$ auf die Sprache L zu reduzieren. Als ihr Tutor ihre beiden Ergebnisse sieht, ist er begeistert. „Damit könnte sich das P vs. NP Problem lösen lassen.“, ruft er. Lösen Sie mit Hilfe von L das P vs. NP Problem, das heißt, zeigen Sie $\mathsf{P} = \mathsf{NP}$ oder $\mathsf{P} \neq \mathsf{NP}$

Lösungsvorschlag 8.9
Nach dem Satz von Rice ist L' unentscheidbar. Aufgrund der Reduktion gilt dies auch für L. Da L in NP ist, ist L aber auch entscheidbar, ein Widerspruch. Aus diesem Widerspruch lässt sich nun sowohl $\mathsf{P} = \mathsf{NP}$, als auch $\mathsf{P} \neq \mathsf{NP}$ folgern.

Aufgabe 8.10 *(Was wäre wenn?)*
Nehmen wir an, wir hätten $SAT \in \mathsf{P}$ gezeigt. Zeigen Sie, dass man dann auch zu einer gegebenen Formel in CNF eine erfüllende Belegung deterministisch in Polynomialzeit finden kann, sofern eine solche Belegung existiert.

Lösungsvorschlag 8.10
Sei F in CNF und besitze die Variablen $x_1, \ldots, x_n$.
Man prüfe zunächst, ob F erfüllbar ist. Falls nein, kann keine erfüllende Belegung gefunden werden, weshalb nichts weiter zu tun ist. Andernfalls setze man $x_1 = x_2$ und überprüfe, ob F immer noch erfüllbar ist. Falls nein, muss $x_2 = \neg x_1$ gelten. Dann fahre man suzessive damit fort, die restlichen Variablen durch x_1 oder $\neg x_1$ darzustellen, wobei nach jeder Änderung erneut überprüft werden muss, ob F erfüllbar ist. Gibt es eine Variable x_i, sodass sowohl für $x_i = x_1$ als auch für $x_i = \neg x_1$ F nicht erfüllbar ist, muss bei den vergangenen Variablen mindestens eine falsch gesetzt worden sein. Man ändere diese und fahre fort.
Hat man alle Variablen $x_2, \ldots, x_n$ in Abhängigkeit von x_1 dargestellt, so wähle man $x_1 = 1$ und überprüfe, ob dies eine erfüllbare Belegung ist (in Polynomialzeit möglich). Falls ja, ist dies die gesuchte erfüllende Belegung.

Andernfalls, muss $x_1 = 0$ gelten.
Da $SAT \in P$ nach Annahme gilt, ist dies insgesamt in polynomieller Zeit möglich.