Entscheidbare und unentscheidbare Probleme

Hans U. Simon (RUB)

mit Modifikationen von

Maike Buchin (RUB)

Lehrstuhl Mathematik und Informatik

Homepage: http://www.ruhr–uni–bochum.de/lmi

Das Grundvokabular der Theorie entscheidbarer und unentscheidbarer Sprachen

Entscheidbarkeit und Semi–Entscheidbarkeit

Eine Sprache $L \subseteq \Sigma^*$ heißt entscheidbar **gdw** die charakteristische Funktion

$$\chi_L(w) := \begin{cases} 1 & \text{falls } w \in L \\ 0 & \text{falls } w \notin L \end{cases}$$

von L berechenbar ist.

Eine Sprache $L \subseteq \Sigma^*$ heißt semi–entscheidbar **gdw** die „halbe“ charakteristische Funktion

$$\chi'_L(w) := \begin{cases} 1 & \text{falls } w \in L \\ \text{„undefiniert“} & \text{falls } w \notin L \end{cases}$$

von L berechenbar ist.

Erinnerung: Sprache und Haltebereich einer DTM

Wir erinnern an die Definition der Sprache $T(M)$ und des Haltebereiches $H(M)$ einer DTM M:

$$
\begin{aligned}
T(M) &= \{w \in \Sigma^* \mid \exists z_e \in E, \alpha, \beta \in \Gamma^* : z_0 w \vdash^* \alpha z_e \beta\} \\
H(M) &= \{w \in \Sigma^* \mid \\
&\quad \exists z \in Z, A \in \Gamma, \alpha, \beta \in \Gamma^* : z_0 w \vdash^* \alpha z A \beta, \delta(z, A) = \text{„undefiniert“}\}
\end{aligned}
$$

Hierbei setzen wir folgendes voraus:

- $\delta(z_e, A) =$„undefiniert“ für alle $z_e \in E$.
- $\delta(z, A)$ mit $z \in Z \setminus E$ darf undefiniert sein (muss aber nicht).

Somit gilt

$$T(M) \subseteq H(M) \subseteq \Sigma^* \ .$$

Entscheidbarkeit (fortgesetzt)

Ein nicht-akzeptierender Stoppzustand ist ein Zustand $z \in Z \setminus E$ mit $\delta(z, A) =$„undefiniert“ für alle $A \in \Gamma$.

Satz: L ist entscheidbar **gdw** es eine DTM M gibt mit

$$T(M) = L \text{ und } H(M) = \Sigma^* \ .$$

Beweis: Es wird ausgenutzt, dass Produzieren der Ausgabe 1 ersetzt werden kann durch Übergang in einen (akzeptierenden) Endzustand — und umgekehrt. Produzieren der Ausgabe 0 kann ersetzt werden durch Übergang in einen (nicht-akzeptierenden) Stoppzustand außerhalb von E — und umgekehrt.

Semi–Entscheidbarkeit (fortgesetzt)

Satz: L ist semi–entscheidbar **gdw** es eine DTM M gibt mit $T(M) = L$.

Beweis: Es wird wieder ausgenutzt, dass Produzieren der Ausgabe „1“ ersetzt werden kann durch Übergang in einen (akzeptierenden) Endzustand — und umgekehrt.

Folgerung: Jede entscheidbare Sprache ist (erst recht) semi-entscheidbar.

Sprachen und Entscheidungsprobleme

(Binäre) Entscheidungsprobleme sind Probleme, welche nur die Antworten JA oder NEIN zulassen.

Sprachen und Entscheidungsprobleme sind zwei Seiten der gleichen Münze:

- Ein Entscheidungsproblem kann auch aufgefasst werden als die Sprache aller Eingabeinstanzen, welche zur Antwort JA führen.
- Eine Sprache kann auch als das Problem aufgefasst werden zu entscheiden, ob eine Eingabeinstanz zur Sprache gehört (Wortproblem).

Wir werden daher im Folgenden „Sprache“ und „Problem“ zuweilen synonym verwenden.

Erinnerung: Abzählbarkeit

Erinnerung: Eine nichtleere Menge A ist abzählbar **gdw** wenn wir ihre Elemente durchnummerieren können, d.h., wenn die Elemente sich bijektiv (1–zu–1) auf $\mathbb{N}_0$ oder (falls A endlich ist) auf eine endliche Teilmenge von $\mathbb{N}_0$ abbilden lassen.

Äquivalent hierzu können wir fordern, dass eine Abbildung $f : \mathbb{N}_0 \to A$ existiert, so dass

$$A = \{f(0), f(1), f(2), \ldots\} \ .$$

Beachte, dass $f(i) = f(j)$ für $i \neq j$ zulässig ist (sonst würden endliche Mengen A ausgeschlossen).

- Jede Teilmenge einer abzählbaren Menge ist ebenfalls abzählbar.
- Da die Wortmenge Σ^* über einem endlichen Alphabet Σ abzählbar ist, ist jede formale Sprache $L \subseteq \Sigma^*$ eine abzählbare Menge.

Aufzählbarkeit

intuitiv: „Aufzählbarkeit“ = „algorithmisch durchführbare Abzählbarkeit“.

Formale Definition: Eine Sprache L heißt (rekursiv) aufzählbar **gdw** $L = \emptyset$ oder es gibt eine total berechenbare (= total definierte und berechenbare) Abbildung f mit $L = \{f(0), f(1), f(2), \ldots\}$.

Wir wenden diese Definition sinngemäß auch auf Mengen an, deren Elemente „auf natürliche Weise“ als Strings kodiert werden können (so dass die Menge als formale Sprache auffassbar ist).

Eine DTM zur Berechnung von f nennen wir eine „Abzählmaschine“ für L.

- Σ^* ist aufzählbar.
- Es gibt nicht aufzählbare formale Sprachen (Beispiele hierfür später).
- Eine Teilmenge einer aufzählbaren Menge ist nicht notwendig aufzählbar.
- $\mathbb{N}_0 \times \mathbb{N}_0$ ist aufzählbar

Eine Abzählmaschine für $\{0,1\}^*$

Betrachte die Abbildung $f_* : \mathbb{N}_0 \to \{0,1\}^*$, die n abbildet auf den n–ten Binärstring der unendlichen Liste

$$\varepsilon, 0, 1, 00, 01, 10, 11, 000, \ldots \quad . \tag{1}$$

Also

$f_*(0) = \varepsilon, f_*(1) = 0, f_*(2) = 1, f_*(3) = 00, f_*(4) = 01, f_*(5) = 10, f_*(6) = 11, \ldots$

Die Berechnung von $f_*(n)$ ist (ohne auf Effizienz zu achten) offensichtlich:

- Zähle auf Band A einen Binärzähler hoch.
- Zähle parallel dazu auf Band B einen Zähler hoch, der die Binärstrings gemäß (1) durchläuft.
- Wenn der Zähler von Band A auf n steht, dann enthält Band B den Binärstring $f^*(n)$.

f_* ist bijektiv und ihre Umkehrabbildung ist nach dem gleichen Schema berechenbar.

Eine Abzählmaschine für $\mathbb{N}_0 \times \mathbb{N}_0$

Betrachte die bijektive Abbildung $d : \mathbb{N}_0 \to \mathbb{N}_0 \times \mathbb{N}_0$, die n abbildet auf das n-te Zahlenpaar der unendlichen Liste

$$\underbrace{(0,0)}_{x+y=0}, \underbrace{(0,1),(1,0)}_{x+y=1}, \underbrace{(2,0),(1,1),(0,2)}_{x+y=2}, \underbrace{(3,0),(2,1),(1,2),(0,3)}_{x+y=3}, \ldots \quad . \tag{2}$$

Also

$$d(0) = (0,0), d(1) = (1,0), d(2) = (0,1), d(3) = (2,0), d(4) = (1,1), \ldots \quad .$$

Die Umkehrabbildung $c : \mathbb{N}_0 \times \mathbb{N}_0 \to \mathbb{N}_0$ ordnet dann einem Zahlenpaar $(x,y) \in \mathbb{N}_0 \times \mathbb{N}_0$ eine eindeutige Nummer $c(x,y) \in \mathbb{N}_0$ zu.

Mit Hilfe eines Zählers zum Auflisten der Paare gemäß (2) und eines parallel dazu inkrementierten Binärzählers sind d und c (ohne auf Efizienz zu achten) offensichtlich berechenbar.

Veranschaulichung der Nummerierung von Zahlenpaaren

Die Nummern $c(x, y)$ ergeben sich auch aus folgender Tabelle:

	y=0	y=1	y=2	y=3	y=4	$\cdots$
x=0	0	1	3	6	10	$\cdots$
x=1	2	4	7	11	16	$\cdots$
x=2	5	8	12	17	23	$\cdots$
x=3	9	13	18	24	31	$\cdots$
x=4	14	19	25	32	40	$\cdots$

$\cdots$

Es gilt (Denksportaufgabe!):

$$c(x, y) = \binom{x + y + 1}{2} + x$$

Äquivalenz von Aufzählbarkeit und Semi–Entscheidbarkeit

Satz Eine Sprache L ist aufzählbar **gdw** L semi–entscheidbar ist.

Wir haben zwei Beweisrichtungen:

1. Transformation einer Abzählmaschine für L in eine DTM zur Berechnung von χ'_L.
2. Transformation einer DTM zur Berechnung von χ'_L in eine Abzählmaschine für L (vorausgesetzt $L \neq \emptyset$).

1. Beweisrichtung

Gegeben: eine Abzählmaschine M für L, die eine Funktion f mit $f(\mathbb{N}_0) = L$ berechnet

Gesucht: eine DTM M' zur Berechnung von $\chi'(w)$

Methode: Für $i = 0, 1, 2, \ldots$ mache folgendes:

1. Berechne $f(i)$ mit Hilfe von M.
2. Falls $f(i) = w$, dann gib 1 aus und stoppe.

- Ein Eingabewort $w \in L$ taucht für mindestens einen Index i als $w = f(i)$ in der Abzählung auf und führt zur Ausgabe 1.
- Für $w \notin L$ gerät M' in eine Endlosschleife.

2. Beweisrichtung

Gegeben: eine DTM M' zur Berechnung von χ'_L

Gesucht: eine Abzählmaschine M für L, die eine Funktion $f(n)$ mit $f(\mathbb{N}_0) = L$ berechnet

Erinnerung: total berechenbare bijektive Abbildungen $f_* : \mathbb{N}_0 \rightarrow \Sigma^*$, $d : \mathbb{N}_0 \rightarrow \mathbb{N}_0 \times \mathbb{N}_0$

Methode:
1. Zu Eingabe n berechne $d(n) = (n_1, n_2) \in \mathbb{N}_0 \times \mathbb{N}_0$.
2. Simuliere n_2 Schritte von M' angesetzt auf Eingabe $f_*(n_1)$.
3. Falls M' in dieser Zeit Ausgabe „1“ produziert, dann gib $f(n) := f_*(n_1)$ aus; andernfalls gib einen „Default–String“ $w \in L$ aus.

Korrektheit: Offensichtlich gilt $f(n) \in L$ für alle $n \in \mathbb{N}_0$, d.h., es werden wirklich nur Wörter aus L aufgezählt.
Zudem ist f surjektiv, d.h., jedes Wort $x \in L$ kommt in der Aufzählung vor. Wieso ? (Begründung in der Vorlesung)

Überblick zur Semi–Entscheidbarkeit

Folgende Aussagen zu einer Sprache $L \subseteq \Sigma^*$ sind äquivalent:

- L ist aufzählbar.
- L ist semi–entscheidbar.
- L hat eine DTM als Akzeptor.
- L hat eine NTM als Akzeptor.
- L ist vom Typ 0.

Abschluss–Eigenschaften

Satz 1: Die Klasse der entscheidbaren Sprachen ist abgeschlossen unter den Operationen „$\cup, \cap, \neg, \cdot, *$“.

Satz 2: Die Klasse der semi–entscheidbaren Sprachen ist abgeschlossen unter den Operationen „$\cup, \cap, \cdot, *$“, aber (wie wir später noch zeigen werden) nicht unter der Operation „$\neg$“.

- Für semi-entscheidbare Sprachen (= Typ 0 Sprachen) sind uns diese Abschlusseigenschaften von den Typ 0 Sprachen her bereits bekannt.
- Der Nachweis der Abschluss–Eigenschaften kann (relativ leicht) geführt werden, indem zwei gegebene DTMs für L_1 und L_2 benutzt werden, um DTMs für

$$L_1 \cup L_2, L_1 \cap L_2, \bar{L}_1, L_1 \cdot L_2, L_1^*$$

zusammenzubasteln (Syntheseprobleme).

Zweimal „halb“ macht „ganz“

Satz: Eine Sprache L ist entscheidbar **gdw** L und $\bar{L}$ semi–entscheidbar sind.

$\Rightarrow$:

Wenn L entscheidbar ist, dann ist auch $\bar{L}$ entscheidbar.

Da jede entscheidbare Sprache erst recht semi–entscheidbar ist, sind folgerichtig dann L und $\bar{L}$ semi–entscheidbar.

$\Leftarrow$:

Wenn L und $\bar{L}$ semi–entscheidbar sind, sagen wir $\chi'_{\bar{L}}$ und χ'_L werden durch DTMs M_0 und M_1 berechnet, dann ist χ_L nach folgendem Muster berechenbar:

Für $t = 0, 1, 2, \ldots$ mache folgendes:

1. Simuliere M_0 und M_1 jeweils auf Eingabe w für t Schritte.
2. Sowie eine der DTMs, sagen wir M_i, eine 1 ausgibt und stoppt, dann gib i aus und stoppe ebenfalls.

Eine binäre Kodierung von Turing–Maschinen

Arbeitsalphabet und Zustandsmenge können stets so gewählt werden, dass jedes Symbol und jeder Zustand eine Nummer erhält:

$$\begin{aligned} \Gamma &= \{A_0, \ldots, A_{r-1}\} \\ Z &= \{z_0, \ldots, z_{s-1}\} \end{aligned}$$

Dabei ist z_0 der Startzustand und oBdA gibt es nur einen Endzustand, und das ist z_{s-1}.

Ebenso können die Richtungsangaben nummeriert werden:

$$d_0 = L, d_1 = R, d_2 = N$$

Ein Eintrag $\delta(z_i, A_j) = (z_{i'}, A_{j'}, d_k)$ der Turing–Tafel kann dann durch den String

$$\#\#bin(i)\#bin(j)\#bin(i')\#bin(j')\#bin(k)$$

kodiert werden.

Die Turing–Tafel ist dann kodiert durch die Konkatenation der Codewörter ihrer Einträge (wobei diese, sagen wir, zeilenweise durchlaufen werden).

Die komplette TM ist schließlich kodiert durch die Konkatention von:

- Präambel bin(r)#bin(s)
- Kodierung der Turing-Tafel
- Endmarkierung ###, welche dafür sorgt, dass kein Codewort Präfix eines anderen Codewortes ist.

Schließlich erhalten wir ein binäres Codewort durch die Substitutionen

$$0 \mapsto 00, 1 \mapsto 01, \# \mapsto 11 \quad .$$

Binäre Kodierung von Turing–Maschinen (fortgesetzt)

Es ist nicht schwer zu zeigen, dass

$$G := \{w \in \{0,1\}^* |\ w \text{ ist Codewort einer DTM}\}$$

entscheidbar ist.

- Falls $w \in G$, dann bezeichne M_w die von w kodierte DTM.
- Falls $w \notin G$, dann bezeichne M_w eine (beliebig aber fest ausgewählte) „Default–DTM“.

Damit haben wir erreicht, dass jedes Binärwort als DTM interpretierbar ist!

Universelle Turing–Maschine

Die universelle Sprache ist definiert wie folgt:

$$U := \{w\#x |\ x \in T(M_w)\}$$

Eine DTM heißt Universelle Turing–Maschine (UTM) **gdw** sie ein Akzeptor von U ist.

Eine UTM ist eine Art „General Purpose Computer“, der auf Eingaben der Form $w\#x$ vorgeht wie folgt:

- Simuliere M_w auf x.
- Akzeptiere $w\#x$ **gdw** M_w ihre Eingabe x akzeptiert.

Konstruktion einer UTM

Satz: Es gibt eine universelle Turing-Maschine (UTM).

Folgerung: U ist semi–entscheidbar.

Beweis des Satzes: Wir konstruieren eine UTM mit 3 Bändern, die wie folgt eingesetzt werden:

- Band 1 dient als „Programmspeicher“. Es enthält dauerhaft den String w, der als Code der DTM M_w interpretiert wird.
- Band 2 enthält die Binärkodierung $\mathrm{bin}(i)$ des aktuellen Zustands z_i von M_w.
- Band 3 dient als „Rechenspeicher“. Es enthält die aktuelle Bandinschrift der 1-Band DTM M_w in kodierter Form. Dabei wird das Symbol $A_j \in \Gamma$ kodiert als String $\mathrm{bin}(j)\$\ldots\$ \in \{0,1,\$\}^\ell$ mit $\ell = \lceil \log r \rceil$, wobei r die Größe des Bandalphabets Γ von M_w bezeichnet. Durch die „Auspolsterung“ mit $\$\ldots\$$ wird erreicht, dass alle Codewörter für Bandalphabetssymbole die selbe Länge ℓ haben.

Konstruktion einer UTM (fortgesetzt)

Die UTM vollzieht eine Schritt–für–Schritt Simulation von M_w. Ein Rechenschritt von M_w gemäß $\delta(z_i, A_j) = (z_{i'}, A_{j'}, R)$ (analog für die anderen Richtungswechsel des Kopfes) verändert die Konfiguration der Rechnung von M_w wie folgt:

$$\cdots z_i A_j A_k \cdots \vdash_{M_w} \cdots A_{j'} z_{i'} A_k \cdots \quad .$$

Die UTM (mit $\mathrm{bin}(i)$ auf Band 2 und dem ersten Symbol von $\mathrm{bin}(j)$ unter dem Lesekopf von Band 3) simuliert diesen Schritt wie folgt:

1. Finde im „Programm“ w auf Band 1 den Teilstring für den Eintrag $\delta(z_i, A_j) = (z_{i'}, A_{j'}, R)$ der Turing-Tafel.
2. Ersetze auf Band 2 „$\mathrm{bin}(i)$“ durch „$\mathrm{bin}(i')$“.
3. Ersetze auf Band 3 „$\mathrm{bin}(j)\$\ldots\$$“ durch „$\mathrm{bin}(j')\$\ldots\$$“ und bewege den Kopf auf das erste Bit dahinter (also das erste Bit von „$\mathrm{bin}(k)\$\ldots\$$“).

Auf diese Weise kann die Schritt–für–Schritt Simulation in Gang gehalten werden.

Erste Beispiele unentscheidbarer Sprachen

Eine nicht semi–entscheidbare Sprache

Die sogenannte Diagonalsprache ist definiert wie folgt:

$$D := \{w \in \{0,1\}^* | \; w \notin T(M_w)\}$$

In Worten: D besteht aus allen (Kodierungen von) DTMs, die ihre eigene Beschreibung (durch ein Codewort) nicht akzeptieren.

Satz: D ist nicht semi–entscheidbar.

Beweis durch Widerspruch: Wir machen die (heuchlerische) Annahme, es gäbe eine DTM M_0 mit $T(M_0) = D$. Betrachte das Codewort w_0 von M_0. Die folgenden Aussagen sind äquivalent:

(1) $w_0 \in T(M_0)$ (2) $w_0 \in D$. (3) $w_0 \notin T(M_0)$.

- Zur Äquivalenz von (1) und (2) nutze aus, dass $T(M_0) = D$.
- Zur Äquivalenz von (2) und (3) nutze die Definition von D aus.
- Die Äquivalenz von (1) und (3) ist ein WIDERSPRUCH.

Reduzierbarkeit

Definition: Betrachte zwei Sprachen $L_1, L_2 \subseteq \Sigma^*$.
L_1 heißt reduzierbar auf L_2 **gdw** eine total berechenbare Abbildung

$$f : \Sigma^* \to \Sigma^*$$

existiert mit der Eigenschaft

$$\forall w \in \Sigma^* : w \in L_1 \Leftrightarrow f(w) \in L_2 \ .$$

f nennen wir in diesem Zusammenhang eine Reduktionsabbildung.

Notation: $L_1 \leq L_2$.

Eigenschaften dieser Relation

Reflexivität: $L \le L$.

Transitivität: Aus $L_1 \le L_2$ und $L_2 \le L_3$ folgt $L_1 \le L_3$.

- Zum Nachweis der Reflexivität benutze die identische Reduktionsabbildung $f(w) = w$.
- Zum Nachweis der Transitivität setze die Reduktionsabbildungen f_1 und f_2 für die Reduktionen $L_1 \le L_2$ und $L_2 \le L_3$ zu einer Reduktionsabbildung $f(w) := f_2(f_1(w))$ für die Reduktion $L_1 \le L_3$ zusammen:

$$w \in L_1 \Leftrightarrow f_1(w) \in L_2 \Leftrightarrow f_2(f_1(w)) \in L_3$$

Eigenschaften (fortgesetzt)

Voraussetzung: $L_1 \leq L_2$ (Reduktionsabbildung f).

Behauptungen: 1. Falls L_2 (semi–)entscheidbar ist, dann ist auch L_1 (semi–)entscheidbar.

2. Falls L_1 nicht (semi–)entscheidbar ist, dann ist auch L_2 nicht (semi–)entscheidbar.

Beweis: 1. Wegen $w \in L_1 \Leftrightarrow f(w) \in L_2$ gilt

$$\chi'_{L_1}(w) = \chi'_{L_2}(f(w)) \ .$$

Abbildung $\chi'_{L_1}(w)$ kann also berechnet werden, indem zunächst $f(w)$ und anschließend $\chi'_{L_2}(f(w))$ berechnet wird. Eine analoge Bemerkung gilt für die Funktion $\chi_{L_1}(w)$.

2. Die zweite Behauptung ist logisch äquivalent zur ersten. (Umkehrschluss: $A \Rightarrow B$ ist logisch äquivalent zu $\neg B \Rightarrow \neg A$.)

Verbreitung „guter und schlechter Nachrichten“

Betrachte eine Reduktionskette

$$L_1 \leq L_2 \leq \cdots \leq L_{k-1} \leq L_k \enspace .$$

- Wenn L_k (semi–)entscheidbar ist, so auch $L_{k-1}, \ldots, L_2, L_1$.
- Wenn L_1 nicht (semi–)entscheidbar ist, so auch $L_2, \ldots, L_{k-1}, L_k$.

Salopp formuliert:

- „Gute Nachrichten“ verbreiten sich entlang von Reduktionsketten von rechts nach links.
- „Schlechte Nachrichten“ verbreiten sich entlang von Reduktionsketten von links nach rechts.

Wir werden diese Denkweise ausnutzen, um aus der Diagonalsprache D weitere nicht entscheidbare bzw. nicht semi–entscheidbare Sprachen abzuleiten.

Eine kleine Sammlung unentscheidbarer Sprachen

Neben dem Komplement der Diagonalsprache

$$\bar{D} = \{w \in \{0,1\}^* |\ w \in T(M_w)\}$$

und der universellen Sprache

$$U = \{w\#x|\ x \in T(M_w)\}$$

betrachten wir noch die folgenden Sprachen:

$$\begin{array}{rcll} H & := & \{w\#x|\ x \in H(M_w)\} & \text{(Halteproblem)} \\ H_0 & := & \{w|\ \varepsilon \in H(M_w)\} & \text{(Halteproblem auf leerem Band)} \\ K & := & \{w|\ w \in H(M_w)\} & \text{(spezielles Halteproblem)} \end{array}$$

Kleine Sammlung (fortgesetzt)

Bei diesen Sprachen geht es also um die folgenden Fragen:

$\bar{D}$: Akzeptiert eine DTM ihre eigene Beschreibung ?

U: Akzeptiert eine DTM ihre Eingabe ?

H: Stoppt eine auf ihre Eingabe angesetzte DTM nach endlich vielen Schritten ?

H_0: Stoppt eine auf das leere Band angesetzte DTM nach endlich vielen Schritten ?

K: Stoppt eine auf ihre eigene Beschreibung angesetzte DTM nach endlich vielen Schritten ?

Alle diese Fragen werden sich als unentscheidbar erweisen.

Kleine Sammlung (fortgesetzt)

Satz: $\bar{D}$ ist unentscheidbar.

Beweis: Wäre $\bar{D}$ entscheidbar, so wäre auch D entscheidbar.
D ist aber noch nicht einmal semi–entscheidbar.

Satz: $K = \{w|\ w \in H(M_w)\}$ ist semi–entscheidbar.

Beweis: $\chi'_K(w)$ kann berechnet werden wie folgt:

1. Gegeben Eingabe w, berechne $w\#w$.
2. Setze die UTM auf $w\#w$ an und gib „1“ aus, sowie die UTM in eine Stoppkonfiguration gerät.

Wegen

$$w \in K \Leftrightarrow w \in H(M_w) \Leftrightarrow w\#w \in H(UTM)\ .$$

führt diese Vorgehensweise zu einer korrekten Berechnung von $\chi'_K(w)$.

Kleine Sammlung (fortgesetzt)

Wir werden die folgende Reduktionskette nachweisen:

$$\bar{D} \leq U \leq H \leq H_0 \leq K$$

Wegen der Art, wie sich gute und schlechte Nachrichten verbreiten, erhalten wir die

Folgerung
$\bar{D}, U, H, H_0, K$ sind zwar semi-entscheidbar aber unentscheidbar.

Bleibt der Nachweis der obigen Reduktionskette.

Reduktion 1

Lemma: $\bar{D} \leq U$.

Verwende Reduktionsabbildung

$$w \mapsto w\#w \quad .$$

Offensichtlich gilt:

$$w \in \bar{D} \;\Leftrightarrow\; w \in T(M_w) \Leftrightarrow w\#w \in U \quad .$$

Reduktion 2

Lemma: $U \leq H$.

Verwende Reduktionsabbildung

$$w\#x \mapsto w'\#x \ .$$

Ziel: Transformiere M_w in eine DTM $M_{w'}$, so dass

$$x \in T(M_w) \ \Leftrightarrow \ x \in H(M_{w'}) \ .$$

Ändere dazu das „Programm" w von M_w zu einem neuen „Programm" w' einer DTM $M_{w'}$ ab:

- $M_{w'}$ simuliert M_w Schritt-für-Schritt,
- außer dass $M_{w'}$ sich in eine Endlosschleife begibt, falls M_w nicht–akzeptierend stoppt.

Reduktion 3

Lemma: $H \leq H_0$.

Verwende Reduktionsabbildung

$$w\#x \mapsto w' \ .$$

Ziel: Transformiere $w\#x$ in das „Programm“ w' einer DTM M', so dass

$$x \in H(M_w) \ \Leftrightarrow \ \varepsilon \in H(M_{w'}) \ .$$

Die gesuchte DTM $M_{w'}$ arbeitet wie folgt:

- Sie löscht ihre Eingabe (sofern diese nicht ohnehin leer ist) und schreibt dann x auf ihr Band.
- Sie simuliert anschließend Schritt-für-Schritt M_w auf Eingabe x.

Reduktion 4

Lemma: $H_0 \leq K$.

Verwende Reduktionsabbildung

$$w \mapsto w' \ .$$

Ziel: Transformiere M_w in eine DTM $M_{w'}$, so dass

$$\varepsilon \in H(M_w) \ \Leftrightarrow \ w' \in H(M_{w'}) \ .$$

Ändere dabei das „Programm“ w von M_w zu einem neuen Programm w' einer DTM $M_{w'}$ ab:

- $M_{w'}$ löscht zunächst ihre Eingabe (zum Beipiel auch die Eingabe w')
- und simuliert dann Schritt-für-Schritt M_w angesetzt auf das leere Band.