LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Formale Sprachen und Komplexität
Sommersemester 2019

Berechenbarkeitstheorie: Teil III
Unentscheidbarkeit und Reduktionen

Prof. Dr. David Sabel

LFE Theoretische Informatik

Letzte Änderung der Folien: 11. Juli 2019

Inhalt

- Unentscheidbarkeit des Halteproblems
- Technik der Reduktion
- Das Postsche Korrespondenzproblem (PCP)
- Unentscheidbarkeit: Weitere Beispiele
  (vorallem in der Zentralübung)

Entscheidbarkeit

Definition

Eine Sprache $L \subseteq \Sigma^*$ heißt **entscheidbar**, wenn die charakteristische Funktion von L, $\chi_L : \Sigma^* \to \{0, 1\}$ mit

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

berechenbar ist.

algorithmisch: Der χ_L-berechnende Algorithmus terminiert in jedem Fall und liefert ein Ergebnis.

Semi-Entscheidbarkeit

Definition

Eine Sprache heißt **semi-entscheidbar** falls $\chi'_L : \Sigma^* \to \{0, 1\}$ mit

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

berechenbar ist.

algorithmisch: Der χ'_L-berechnende Algorithmus terminiert nur, falls $w \in L$, und läuft anderenfalls endlos.

Satz

Ein Sprache L ist genau dann entscheidbar, wenn L und $\overline{L}$ jeweils semi-entscheidbar sind.

Beweis:

„$\Rightarrow$“ Konstruiere aus TM die χ_L berechnet zwei TMs die χ'_L und $\chi'_{\overline{L}}$ berechnen (einfach).

Korollar

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

Satz

Ein Sprache L ist genau dann entscheidbar, wenn L und $\overline{L}$ jeweils semi-entscheidbar sind.

Beweis:

„$\Rightarrow$“ Konstruiere aus TM die χ_L berechnet zwei TMs die χ'_L und $\chi'_{\overline{L}}$ berechnen (einfach).

„$\Leftarrow$“ Gegeben TMs M_L und $M_{\overline{L}}$, die χ'_L und $\chi'_{\overline{L}}$ berechnen.
Konstruiere TM, die χ_L berechnet:

Starte mit $i = 1$.
Simuliere i-Schritte von M_L.
Wenn diese akzeptiert, dann akzeptiere mit Ausgabe 1.
Ansonsten simuliere i-Schritte von $M_{\overline{L}}$.
Wenn diese akzeptiert, dann akzeptiere mit Ausgabe 0.
Ansonsten erhöhe i um 1 und starte von neuem.

Korollar

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

Rekursive Aufzählbarkeit

Definition

Eine Sprache $L \subseteq \Sigma^*$ heißt **rekursiv aufzählbar**, falls $L = \emptyset$ oder falls es eine totale berechenbare Funktion $f : \mathbb{N} \to \Sigma^*$ gibt, sodass $L = \bigcup_{i \in \mathbb{N}} f(i)$. Man sagt dann „$f$ zählt L auf“.

Rekursive Aufzählbarkeit

Definition

Eine Sprache $L \subseteq \Sigma^*$ heißt **rekursiv aufzählbar**, falls $L = \emptyset$ oder falls es eine totale berechenbare Funktion $f : \mathbb{N} \to \Sigma^*$ gibt, sodass $L = \bigcup_{i \in \mathbb{N}} f(i)$. Man sagt dann „$f$ zählt L auf“.

Lemma

Die Sprache Σ^* ist rekursiv-aufzählbar.

Beweis: Sei $|\Sigma| = b$, $n \in \mathbb{N}$. Interpretiere $w \in \Sigma^*$ als $b+1$-äre Zahl.

- Konstruiere TM, die n in Binärdarstellung auf Eingabeband erhält.
- TM erzeugt auf anderem Band die $b+1$-äre Darstellung der 0
- Anschließend zählt die TM die Zahl auf dem Eingabeband um 1 herunter und die $b+1$-äre Zahl um 1 nach oben.
- Dies wird wiederholt bis auf dem Eingabeband die 0 steht
- Dann steht auf dem anderen Band $f(n)$.

Rekursiv aufzählbar = semi-entscheidbar

Satz

Eine Sprache ist genau dann rekursiv aufzählbar, wenn sie semi-entscheidbar ist.

Beweis:

„$\Rightarrow$“ Sei f die totale, berechenbare Funktion, die L aufzählt.
Dann berechnet der folgende Algorithmus $\chi'_L(w)$:

Für $i = 0, 1, 2, 3, \ldots$ **tue**
  wenn $f(n) = w$ **dann**
    stoppe und gebe 1 aus

„$\Leftarrow$“ ...

Rekursiv aufzählbar = semi-entscheidbar

„$\Leftarrow$“ Sei M eine TM, die χ'_L berechnet.

- Wenn $L = \emptyset$, dann ist L rekursiv-aufzählbar.
- Anderenfalls sei $u \in L$ ein Wort.
  Wir konstruieren TM M', die die L aufzählende Funktion berechnet.
- Sei n eine Eingabe. Wir interpretieren n als $c(x, y)$.
- M' simuliert y Schritte von M bei Eingabe $g(x)$,
  wobei g die Σ^* aufzählende Funktion sei.
- Wenn M nach y Schritten $g(x)$ akzeptiert, dann akzeptiert M'
  mit Ausgabe $g(x)$. Anderenfalls, akzeptiert M' mit Ausgabe u.
- Die von M' berechnete Funktion:

$$f(n) = \begin{cases} w \in L, \text{falls } w = g(\mathit{left}(n)) \text{ und } M \text{ akzeptiert} \\ \qquad\qquad w \text{ in } \mathit{right}(n) \text{ Schritten} \\ u \in L, \text{ sonst} \end{cases}$$

- f zählt L auf, da für jedes Wort w ein x existiert mit $g(x) = w$ und
  ein y existiert, sodass M mit Eingabe w nach y Schritten akzeptiert.

Zusammenfassung: Äquivalente Eigenschaften

Die folgenden Eigenschaften sind äquivalent:

- L ist vom Typ 0.
- L ist semi-entscheidbar.
- L ist rekursiv-aufzählbar.
- Es gibt eine Turingmaschine M, die L akzeptiert (d.h. $L(M) = L$).
- χ'_L ist Turing-, WHILE-, GOTO-berechenbar.
- Es gibt berechenbare Funktionen, die L als Wertebereich (nämlich die L aufzählende Funktion) bzw. als Definitionsbereich (nämlich χ'_L) haben.

Rekursiv aufzählbar $\neq$ abzählbar

- Sprache L ist abzählbar, wenn es eine totale Funktion $f : \mathbb{N} \to L$ gibt, sodass $\bigcup_{i \in \mathbb{N}} f(i) = L$.
- Beachte: Abzählbarkeit fordert nicht das f berechenbar ist!

Gödelisierung von Turingmaschinen

Ziel:

Stelle Turingmaschinenbeschreibung als natürliche Zahl in Binärdarstellung dar:

Grund:

Andere Turingmaschinen können die Beschreibung als Eingabe erhalten, erzeugen, usw.

Gödelisierung von Turingmaschinen (2)

Sei $(Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ eine DTM mit $\Sigma = \{0, 1\}$ und

- $\Gamma = \{a_0, \ldots, a_k\}$ wobei $a_0 = \square$, $a_1 = \#$, $a_2 = 0$, $a_3 = 1$
- $Z = \{z_0, \ldots, z_n\}$
- $E = \{z_n\}$

Für $\delta(z_p, a_i) = (z_q, a_j, D)$ erzeuge Wort über Alphabet $\{0, 1, \#\}$:

$$w_{p,i,q,j,D} = \#\#bin(p)\#bin(i)\#bin(q)\#bin(j)\#bin(D_m)$$

mit $D_m = 0$, falls $D = L$, $D_m = 1$, falls $D = R$, $D_m = 2$, falls $D = N$

Kodierung w_δ: Schreibe alle δ-Worte hintereinander.

Schließlich: Kodiere Alphabet $\{0, 1, \#\}$ durch $\{0 \mapsto 00, 1 \mapsto 01, \# \mapsto 11\}$.

Wende dies auf w_δ an.

Wir bezeichnen mit w_M die so kodierte TM M.

Gödelisierung von Turingmaschinen (3)

- Nicht jedes Wort über $\{0,1\}$ entspricht der Kodierung einer Turingmaschine.
- Sei $\widehat{M}$ eine beliebige aber feste Turingmaschine.
- Definiere für jedes $w \in \{0,1\}^*$ die zugehörige TM M_w:

$$M_w := \begin{cases} M, & \text{wenn } w = w_M \\ \widehat{M}, & \text{sonst} \end{cases}$$

Spezielles Halteproblem

Definition (Spezielles Halteproblem)

Das spezielle Halteproblem ist die Sprache

$$K := \{w \in \{0,1\}^* \mid M_w \text{ hält für Eingabe } w\}$$

Unentscheidbarkeit von K

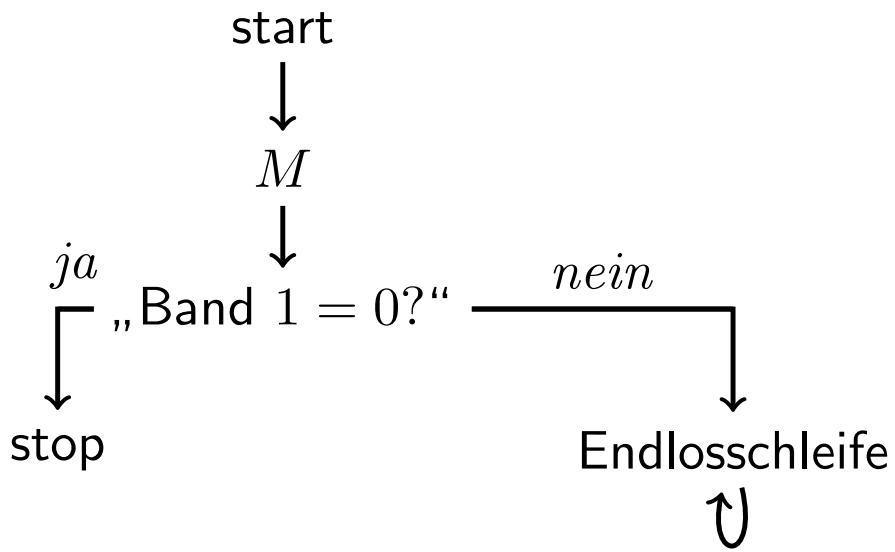
Satz

Das spezielle Halteproblem ist nicht entscheidbar (und damit *unentscheidbar*).

Beweis:

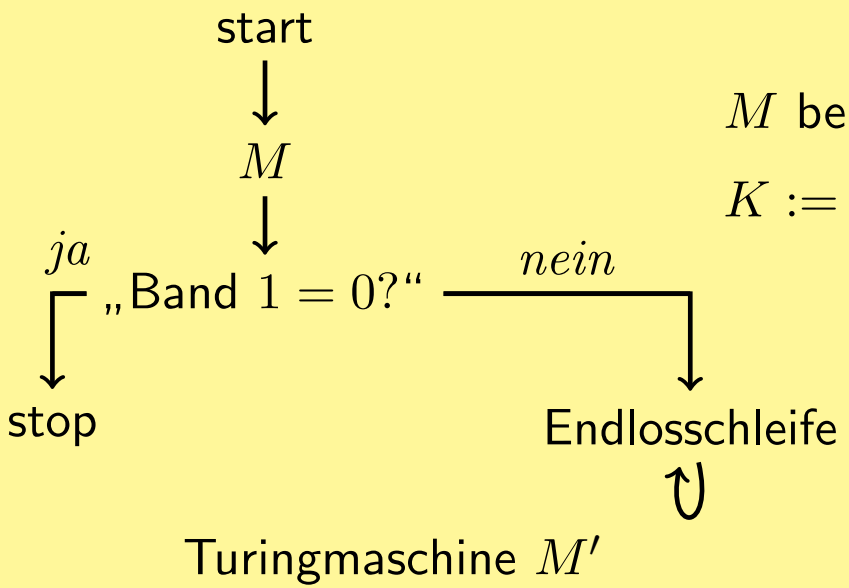
- Annahme: K ist entscheidbar.
- Dann ist χ_K berechenbar, und es gibt TM M, die χ_K berechnet.
- Konstruiere M':

1. M' lässt M ablaufen
2. Wenn M mit 0 auf dem Band endet, dann akzeptiert M'
3. Wenn M mit 1 auf dem Band endet, dann läuft M' in eine Endlosschleife.



Unentscheidbarkeit von K (2)

Zur Erinnerung:



M berechnet χ_K wobei K:

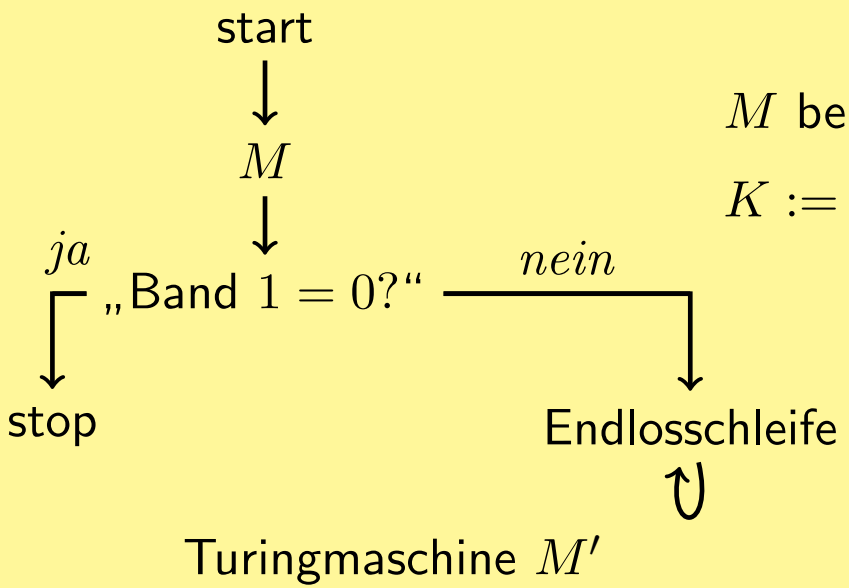
$K := \{w \in \{0,1\}^* \mid M_w \text{ hält für Eingabe } w\}$

Betrachte nun M' auf der Eingabe $w_{M'}$: Es gilt

	M' hält für Eingabe $w_{M'}$
g.d.w.	M angesetzt auf $w_{M'}$ gibt 0 aus
g.d.w.	$\chi_K(w_{M'}) = 0$
g.d.w.	$w_{M'} \notin K$
g.d.w.	M' hält nicht für Eingabe $w_{M'}$

Unentscheidbarkeit von K (2)

Zur Erinnerung:



M berechnet χ_K wobei K:

$K := \{w \in \{0,1\}^* \mid M_w \text{ hält für Eingabe } w\}$

Betrachte nun M' auf der Eingabe $w_{M'}$: Es gilt

	M' hält für Eingabe $w_{M'}$
g.d.w.	M angesetzt auf $w_{M'}$ gibt 0 aus
g.d.w.	$\chi_K(w_{M'}) = 0$
g.d.w.	$w_{M'} \notin K$
g.d.w.	M' hält nicht für Eingabe $w_{M'}$

Unentscheidbarkeit von K (3)

- M' hält für Eingabe $w_{M'} \iff M'$ hält nicht für Eingabe $w_{M'}$
- Widerspruch!
- Annahme war falsch!
- K ist nicht entscheidbar, sondern unentscheidbar.

Diagonalisierungsargument

	w_1	w_2	w_3	w_4	$\dots$
M_{w_1}	ja	$nein$	ja	$\dots$	$\dots$
M_{w_2}	$nein$	$nein$	ja	$\dots$	$\dots$
M_{w_3}	ja	$nein$	ja	$\dots$	$\dots$
M_{w_4}	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$

- Spalten: alle Worte über $\{0,1\}$: $w_1, w_2, \dots$
- Zeilen: alle TMs $M_{w_1}, M_{w_2}, \dots$.
- Eintrag in Zeile i und Spalte j: ja, wenn M_{w_i} Bei Eingabe w_j akzeptiert, $nein$ sonst.

Diagonalisierungsargument

	w_1	w_2	w_3	w_4	$\ldots$
M_{w_1}	ja	$nein$	ja	$\ldots$	$\ldots$
M_{w_2}	$nein$	$nein$	ja	$\ldots$	$\ldots$
M_{w_3}	ja	$nein$	ja	$\ldots$	$\ldots$
M_{w_4}	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
M_{w_D}					

- Spalten: alle Worte über $\{0,1\}$: $w_1, w_2, \ldots$
- Zeilen: alle TMs $M_{w_1}, M_{w_2}, \ldots$.
- Eintrag in Zeile i und Spalte j: ja, wenn M_{w_i} Bei Eingabe w_j akzeptiert, $nein$ sonst.
- Diagonalsprache: $L_D = \{w_i \mid M_{w_i} \text{ hält nicht für Eingabe } w_i\}$
- Sei w_D die TM-Beschreibung von TM M_{w_D}, die χ_{L_D} berechnet.

Diagonalisierungsargument

	w_1	w_2	w_3	w_4	$\ldots$
M_{w_1}	ja	$nein$	ja	$\ldots$	$\ldots$
M_{w_2}	$nein$	$nein$	ja	$\ldots$	$\ldots$
M_{w_3}	ja	$nein$	ja	$\ldots$	$\ldots$
M_{w_4}	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
M_{w_D}	$nein$				

- Spalten: alle Worte über $\{0,1\}$: $w_1, w_2, \ldots$
- Zeilen: alle TMs $M_{w_1}, M_{w_2}, \ldots$.
- Eintrag in Zeile i und Spalte j: ja, wenn M_{w_i} Bei Eingabe w_j akzeptiert, $nein$ sonst.
- Diagonalsprache: $L_D = \{w_i \mid M_{w_i} \text{ hält nicht für Eingabe } w_i\}$
- Sei w_D die TM-Beschreibung von TM M_{w_D}, die χ_{L_D} berechnet.

Diagonalisierungsargument

	w_1	w_2	w_3	w_4	$\dots$
M_{w_1}	ja	$nein$	ja	$\dots$	$\dots$
M_{w_2}	$nein$	$nein$	ja	$\dots$	$\dots$
M_{w_3}	ja	$nein$	ja	$\dots$	$\dots$
M_{w_4}	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
M_{w_D}	$nein$	ja			

- Spalten: alle Worte über $\{0,1\}$: $w_1, w_2, \dots$
- Zeilen: alle TMs $M_{w_1}, M_{w_2}, \dots$.
- Eintrag in Zeile i und Spalte j: ja, wenn M_{w_i} Bei Eingabe w_j akzeptiert, $nein$ sonst.
- Diagonalsprache: $L_D = \{w_i \mid M_{w_i} \text{ hält nicht für Eingabe } w_i\}$
- Sei w_D die TM-Beschreibung von TM M_{w_D}, die χ_{L_D} berechnet.

Diagonalisierungsargument

	w_1	w_2	w_3	w_4	$\dots$
M_{w_1}	ja	$nein$	ja	$\dots$	$\dots$
M_{w_2}	$nein$	$nein$	ja	$\dots$	$\dots$
M_{w_3}	ja	$nein$	ja	$\dots$	$\dots$
M_{w_4}	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
M_{w_D}	$nein$	ja	$nein$		

- Spalten: alle Worte über $\{0,1\}$: $w_1, w_2, \dots$
- Zeilen: alle TMs $M_{w_1}, M_{w_2}, \dots$.
- Eintrag in Zeile i und Spalte j: ja, wenn M_{w_i} Bei Eingabe w_j akzeptiert, $nein$ sonst.
- Diagonalsprache: $L_D = \{w_i \mid M_{w_i} \text{ hält nicht für Eingabe } w_i\}$
- Sei w_D die TM-Beschreibung von TM M_{w_D}, die χ_{L_D} berechnet.

Diagonalisierungsargument

	w_1	w_2	w_3	w_4	$\ldots$	w_D
M_{w_1}	ja	$nein$	ja	$\ldots$	$\ldots$	$\ldots$
M_{w_2}	$nein$	$nein$	ja	$\ldots$	$\ldots$	$\ldots$
M_{w_3}	ja	$nein$	ja	$\ldots$	$\ldots$	$\ldots$
M_{w_4}	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
M_{w_D}	$nein$	ja	$nein$	$\ldots$	$\ldots$	

- Spalten: alle Worte über $\{0,1\}$: $w_1, w_2, \ldots$
- Zeilen: alle TMs $M_{w_1}, M_{w_2}, \ldots$.
- Eintrag in Zeile i und Spalte j: ja, wenn M_{w_i} Bei Eingabe w_j akzeptiert, $nein$ sonst.
- Diagonalsprache: $L_D = \{w_i \mid M_{w_i} \text{ hält nicht für Eingabe } w_i\}$
- Sei w_D die TM-Beschreibung von TM M_{w_D}, die χ_{L_D} berechnet.

Diagonalisierungsargument

	w_1	w_2	w_3	w_4	$\ldots$	w_D
M_{w_1}	ja	$nein$	ja	$\ldots$	$\ldots$	$\ldots$
M_{w_2}	$nein$	$nein$	ja	$\ldots$	$\ldots$	$\ldots$
M_{w_3}	ja	$nein$	ja	$\ldots$	$\ldots$	$\ldots$
M_{w_4}	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
M_{w_D}	$nein$	ja	$nein$	$\ldots$	$\ldots$	**??**

- Spalten: alle Worte über $\{0,1\}$: $w_1, w_2, \ldots$
- Zeilen: alle TMs $M_{w_1}, M_{w_2}, \ldots$.
- Eintrag in Zeile i und Spalte j: ja, wenn M_{w_i} Bei Eingabe w_j akzeptiert, $nein$ sonst.
- Diagonalsprache: $L_D = \{w_i \mid M_{w_i} \text{ hält nicht für Eingabe } w_i\}$
- Sei w_D die TM-Beschreibung von TM M_{w_D}, die χ_{L_D} berechnet.

Reduktion

- Hilfsmittel, um Unentscheidbarkeit nachzuweisen
- Statt Unentscheidbarkeit von Sprache L von Grundauf neu zu beweisen, zeige:

  Wenn man L entscheiden könnte, dann könnte man auch L_0 entscheiden.
- Wenn L_0 bereits als unentscheidbar gezeigt, folgt L ist unentscheidbar.

Reduktion (Definition)

Definition (Reduktion (einer Sprache auf eine andere))

Sei $L_1 \subseteq \Sigma_1^*$ und $L_2 \subseteq \Sigma_2^*$ Sprachen.
Dann sagen wir L_1 ist auf L_2 reduzierbar (geschrieben $L_1 \leq L_2$), falls es eine totale und berechenbare Funktion $f : \Sigma_1^* \to \Sigma_2^*$ gibt, sodass für alle $w \in \Sigma_1^*$ gilt: $w \in L_1 \iff f(w) \in L_2$

Reduktion (Definition)

Definition (Reduktion (einer Sprache auf eine andere))

Sei $L_1 \subseteq \Sigma_1^*$ und $L_2 \subseteq \Sigma_2^*$ Sprachen.
Dann sagen wir L_1 ist auf L_2 reduzierbar (geschrieben $L_1 \leq L_2$), falls es eine totale und berechenbare Funktion $f : \Sigma_1^* \to \Sigma_2^*$ gibt, sodass für alle $w \in \Sigma_1^*$ gilt: $w \in L_1 \iff f(w) \in L_2$

Satz

Wenn $L_1 \leq L_2$ und L_2 entscheidbar (bzw. semi-entscheidbar) ist, dann ist auch L_1 entscheidbar (bzw. semi-entscheidbar).

Reduktion (Definition)

Definition (Reduktion (einer Sprache auf eine andere))

Sei $L_1 \subseteq \Sigma_1^*$ und $L_2 \subseteq \Sigma_2^*$ Sprachen.
Dann sagen wir L_1 ist auf L_2 reduzierbar (geschrieben $L_1 \leq L_2$), falls es eine totale und berechenbare Funktion $f : \Sigma_1^* \to \Sigma_2^*$ gibt, sodass für alle $w \in \Sigma_1^*$ gilt: $w \in L_1 \iff f(w) \in L_2$

Satz

Wenn $L_1 \leq L_2$ und L_2 entscheidbar (bzw. semi-entscheidbar) ist, dann ist auch L_1 entscheidbar (bzw. semi-entscheidbar).

Beweis (nur entscheidbar, semi-entscheidbar analog):
Sei f die $L_1 \leq L_2$ bezeugende (und berechenbare) Funktion.
Da L_2 entscheidbar, ist χ_{L_2} berechenbar. Damit ist auch $\chi_{L_1}(w) := \chi_{L_2}(f(w))$ berechenbar. Offensichtlich gilt

$$\chi(L_1)(w) := \left\{ \begin{array}{ll} 1, & w \in L_1 \\ 0, & w \notin L_1 \end{array} \right\} = \left\{ \begin{array}{ll} 1, & f(w) \in L_2 \\ 0, & f(w) \notin L_2 \end{array} \right\} = \chi_{L_2}(f(w))$$

Unentscheidbarkeit

Mit Kontraposition folgt:

Lemma

Sei $L_1 \leq L_2$ und L_1 ist unentscheidbar.
Dann ist auch L_2 unentscheidbar.

Das ist die Richtung, die wir meistens brauchen:

- L_1 sei eine bekannt unentscheidbare Sprache
- Reduziere L_1 auf L_2 durch Angabe einer berechenbaren Funktion f mit $w \in L_1 \iff f(w) \in L_2$.
- Damit folgt, dass L_2 unentscheidbar ist.

Halteproblem

Definition (Halteproblem)

Das (allgemeine) Halteproblem ist definiert als
$H := \{w\#x \mid \text{TM } M_w \text{ hält für Eingabe } x\}$

Halteproblem

Definition (Halteproblem)

Das (allgemeine) Halteproblem ist definiert als
$H := \{w\#x \mid \text{TM } M_w \text{ hält für Eingabe } x\}$

Satz

Das allgemeine Halteproblem ist unentscheidbar.

Halteproblem

Definition (Halteproblem)

Das (allgemeine) Halteproblem ist definiert als
$H := \{w\#x \mid \text{TM } M_w \text{ hält für Eingabe } x\}$

Satz

Das allgemeine Halteproblem ist unentscheidbar.

Beweis: Wir reduzieren das spezielle Halteproblem auf das allgemeine Halteproblem, und zeigen daher $K \leq H$. Sei $f(w) = w\#w$. Dann gilt

$$\begin{aligned} & w \in K \\ \text{g.d.w.} \quad & M_w \text{ hält für Eingabe } w \\ \text{g.d.w.} \quad & w\#w \in H \\ \text{g.d.w.} \quad & f(w) \in H \end{aligned}$$

f kann durch eine TM berechnet werden. Damit gilt $K \leq H$ und damit folgt aus K unentscheidbar auch H unentscheidbar.

Halteproblem bei leerer Eingabe

Definition

Das Halteproblem auf leerem Band ist die Sprache
$H_0 = \{w \mid M_w \text{ hält für die leere Eingabe}\}$

Halteproblem bei leerer Eingabe

Definition

Das Halteproblem auf leerem Band ist die Sprache
$H_0 = \{w \mid M_w \text{ hält für die leere Eingabe}\}$

Satz

Das Halteproblem auf leerem Band ist unentscheidbar.

Halteproblem bei leerer Eingabe

Definition

Das Halteproblem auf leerem Band ist die Sprache
$H_0 = \{w \mid M_w \text{ hält für die leere Eingabe}\}$

Satz

Das Halteproblem auf leerem Band ist unentscheidbar.

Beweis: Wir reduzieren H auf H_0: Sei $f(w_M \# x) = w_{M_{0,x}}$, wobei TM $M_{0,x}$ erst x auf das Band schreibt, sich dann wie M verhält.

$$
\begin{array}{ll}
 & w_M \# x \in H \\
\text{g.d.w.} & M_w \text{ hält für Eingabe } x \\
\text{g.d.w.} & M_{0,x} \text{ hält für die leere Eingabe} \\
\text{g.d.w.} & w_{M_{0,x}} \in H_0 \\
\text{g.d.w.} & f(w_M \# x) \in H_0
\end{array}
$$

Funktion f kann durch eine TM berechnet werden. Daher gilt $H \leq H_0$. Da H unentscheidbar, ist H_0 unentscheidbar.

Der Satz von Rice

Von Henry Gordon Rice im Jahr 1953 veröffentlicht.

Satz von Rice

Sei $\mathcal{R}$ die Klasse aller Turingberechenbaren Funktionen. Sei $\mathcal{S}$ eine beliebige Teilmenge, sodass $\emptyset \subset \mathcal{S} \subset \mathcal{R}$. Dann ist die Sprache

$$C(\mathcal{S}) = \{w \mid \text{die von } M_w \text{ berechnete Funktion liegt in } \mathcal{S}\}$$

unentscheidbar.

Der Satz zeigt:

- Fast alle interessanten Eigenschaften von TMs sind algorithmisch nicht entscheidbar.
- Z.B. folgt, dass die Sprache
  $$L = \{w \mid M_w \text{ berechnet eine konstante Funktion}\}$$
  nicht entscheidbar ist.

Beweis des Satzes von Rice

Sei $\Omega(x) = \perp$ für alle x.

Zeige:

1. $H_0 \leq C(\mathcal{S})$ für den Fall $\Omega \notin \mathcal{S}$.

   In diesem Fall folgt aus der Unentscheidbarkeit von H_0 die Unentscheidbarkeit von $C(\mathcal{S})$

2. $H_0 \leq C(\mathcal{R} \setminus \mathcal{S})$ falls $\Omega \in \mathcal{S}$.

   In diesem Fall folgt aus der Unentscheidbarkeit von H_0 die Unentscheidbarkeit von $C(\mathcal{R} \setminus \mathcal{S})$ und damit auch die Unentscheidbarkeit von $C(\mathcal{S})$, denn $\overline{C(\mathcal{S})} = C(\mathcal{R} \setminus \mathcal{S})$ und $\forall L : L$ entscheidbar $\iff \overline{L}$ entscheidbar.

Wir beweisen nur 1), da 2) komplett analog geht.

Beweis des Satzes von Rice (2)

Fall: $\Omega \notin \mathcal{S}$. Da $\emptyset \subset \mathcal{S}$, gibt es $q \in \mathcal{S}$, die von einer TM Q berechnet wird.
Konstruktion einer TM M^*: Für TM M und Eingabe y:

1. M^* simuliert M auf leerer Eingabe.
2. Wenn M anhält, dann simuliert M^* die TM Q mit Eingabe y.

Sei f die Funktion, die aus Beschreibung w für TM M_w,
die Beschreibung $f(w)$ von M_w^* erstellt.

$$
\begin{aligned}
\text{Dann gilt: } w \in H_0 &\implies M_w \text{ hält auf leerer Eingabe} \\
&\implies M_w^* \text{ berechnet } q \\
&\implies \text{die von } M_w^* \text{ berechnete Funktion liegt } \mathcal{S} \\
&\implies f(w) \in C(\mathcal{S})
\end{aligned}
$$

$$
\begin{aligned}
\text{und ebenso: } w \notin H_0 &\implies M_w \text{ hält nicht auf leerer Eingabe} \\
&\implies M_w^* \text{ berechnet } \Omega \\
&\implies \text{die von } M_w^* \text{ berechnete Funktion liegt nicht in } \mathcal{S} \\
&\implies f(w) \notin C(\mathcal{S})
\end{aligned}
$$

Da f berechenbar und $w \in H_0 \iff f(w) \in C(\mathcal{S})$ und somit $H_0 \leq C(\mathcal{S})$.

Das Postsche Korrespondenzproblem

Motivation

- Das Postsche Korrespondenzproblem ist ein einfaches aber unentscheidbares Problem
- Wird häufig verwendet, um es auf andere Probleme zu reduzieren und deren Unentscheidbarkeit zu zeigen
- Vorgeschlagen von Emil Post im Jahr 1946

Definition des Postschen Korrespondenzproblems

Definition (Postsches Korrespondenzproblem)

Gegeben sei ein Alphabet Σ und eine Folge von Wortpaaren $K = \{(x_1, y_1), \ldots, (x_k, y_k)\}$ mit $x_i, y_i \in \Sigma^+$. Das Postsche Korrespondenzproblem (PCP) ist die Frage, ob es für die gegebene Folge K eine Folge von Indizes $i_1, \ldots, i_m$ mit $i_j \in \{1, \ldots, k\}$ gibt, sodass $x_{i_1} \cdots x_{i_m} = y_{i_1} \cdots y_{i_m}$ gilt.

PCP ist wie ein Domino-Spiel

Spielsteinarten: $(\begin{bmatrix} x_1 \\ y_n \end{bmatrix}, \ldots, \begin{bmatrix} x_k \\ y_k \end{bmatrix})$

Gesucht: Aneinandereihung der Spielsteine, sodass oben wie unten dasselbe Wort abgelesen werden kann. Dabei dürfen beliebig (aber endlich) viele Spielsteine verwendet werden.

Beispiel:

Sei $K = (\begin{bmatrix} a \\ aba \end{bmatrix}, \begin{bmatrix} baa \\ aa \end{bmatrix}, \begin{bmatrix} ab \\ bb \end{bmatrix})$

PCP ist wie ein Domino-Spiel

Spielsteinarten: $(\begin{bmatrix} x_1 \\ y_n \end{bmatrix}, \ldots, \begin{bmatrix} x_k \\ y_k \end{bmatrix})$

Gesucht: Aneinandereihung der Spielsteine, sodass oben wie unten dasselbe Wort abgelesen werden kann. Dabei dürfen beliebig (aber endlich) viele Spielsteine verwendet werden.

Beispiel:

Sei $K = (\begin{bmatrix} a \\ aba \end{bmatrix}, \begin{bmatrix} baa \\ aa \end{bmatrix}, \begin{bmatrix} ab \\ bb \end{bmatrix})$

$I = (1, 2, 3, 2)$ ist eine Lösung, da

$$\begin{bmatrix} a \\ aba \end{bmatrix} \begin{bmatrix} baa \\ aa \end{bmatrix} \begin{bmatrix} ab \\ bb \end{bmatrix} \begin{bmatrix} baa \\ aa \end{bmatrix} = abaaabbaa$$
$$= abaaabbaa$$

PCP-Beispiel

Instanz $K = (\begin{bmatrix} ab \\ bba \end{bmatrix}, \begin{bmatrix} ba \\ baa \end{bmatrix}, \begin{bmatrix} ba \\ aba \end{bmatrix}, \begin{bmatrix} bba \\ b \end{bmatrix})$

PCP-Beispiel

Instanz $K = (\begin{bmatrix} ab \\ bba \end{bmatrix}, \begin{bmatrix} ba \\ baa \end{bmatrix}, \begin{bmatrix} ba \\ aba \end{bmatrix}, \begin{bmatrix} bba \\ b \end{bmatrix})$

Die kürzeste Lösung benötigt 66 Paare:

(2, 1, 3, 1, 1, 2, 4, 2, 1, 3, 1, 3, 1, 1, 3, 1, 1, 2, 4, 1, 1, 2, 4, 3, 1, 4, 4, 3, 1, 1, 1, 2, 4, 2, 4, 4, 4, 3, 1, 3, 1, 4, 2, 4, 1, 1, 2, 4, 1, 4, 4, 3, 1, 4, 4, 3, 4, 4, 3, 4, 2, 4, 1, 4, 4, 3).

Unentscheidbarkeit von PCP

Beweis in 2 Schritten:

1. MPCP $\leq$ PCP

   MPCP ist das Modifzierte Postsche Korrespondenzproblem:

   Nur Lösungen zulässig, die mit dem ersten Spielstein $\begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$ beginnen

2. $H \leq$ MPCP

Damit folgt aus der Unentscheidbarkeit von H die Unentscheidbarkeit von MPCP und damit die Unentscheidbarkeit von PCP.

Modifziertes PCP

Definition (Modifiziertes Postsches Korrespondenzproblem)

Gegeben sei ein Alphabet Σ und eine Folge von Wortpaaren $K = \{(x_1, y_1), \ldots, (x_k, y_k)\}$ mit $x_i, y_i \in \Sigma^+$. Das Modifizierte Postsche Korrespondenzproblem (MPCP) ist die Frage, ob es für die gegebene Folge K eine Folge von Indizes $1, i_2, \ldots, i_m$ mit $i_j \in \{1, \ldots, k\}$ gibt, sodass $x_{i_1} \cdots x_{i_m} = y_{i_1} \cdots y_{i_m}$ gilt.

MPCP $\leq$ PCP

Lemma

MPCP $\leq$ PCP

Beweis: Gesucht: f mit: K MPCP-lösbar g.d.w. $f(K)$ PCP-lösbar.

MPCP $\leq$ PCP

Lemma

MPCP $\leq$ PCP

Beweis: Gesucht: f mit: K MPCP-lösbar g.d.w. $f(K)$ PCP-lösbar.

Für $w = a_1 \cdots a_n \in \Sigma^+$ sei:

$\bar{w}$=$\#a_1\#a_2\#\cdots\#a_n\#$ $\quad$ $\acute{w}$=$a_1\#a_2\#\cdots\#a_n\#$ $\quad$ $\grave{w}$=$\#a_1\#a_2\#\cdots\#a_n$

MPCP $\leq$ PCP

Lemma

MPCP $\leq$ PCP

Beweis: Gesucht: f mit: K MPCP-lösbar g.d.w. $f(K)$ PCP-lösbar.

Für $w = a_1 \cdots a_n \in \Sigma^+$ sei:

$$\bar{w}{=}\#a_1\#a_2\#\cdots\#a_n\# \qquad \acute{w}{=}a_1\#a_2\#\cdots\#a_n\# \qquad \grave{w}{=}\#a_1\#a_2\#\cdots\#a_n$$

Sei $f(\begin{bmatrix} x_1 \\ y_1 \end{bmatrix}, \ldots, \begin{bmatrix} x_k \\ y_k \end{bmatrix}) = (\underbrace{\begin{bmatrix} \bar{x}_1 \\ \grave{y}_1 \end{bmatrix}}_{(x'_1,y'_1)}, \underbrace{\begin{bmatrix} \acute{x}_1 \\ \grave{y}_1 \end{bmatrix}}_{(x'_2,y'_2)}, \ldots, \underbrace{\begin{bmatrix} \acute{x}_k \\ \grave{y}_k \end{bmatrix}}_{(x'_{k+1},y'_{k+1})}, \underbrace{\begin{bmatrix} \$ \\ \#\$ \end{bmatrix}}_{(x'_{k+2},y'_{k+2})})$

MPCP $\leq$ PCP

Lemma

MPCP $\leq$ PCP

Beweis: Gesucht: f mit: K MPCP-lösbar g.d.w. $f(K)$ PCP-lösbar.

Für $w = a_1 \cdots a_n \in \Sigma^+$ sei:

$$\bar{w}{=}\#a_1\#a_2\#\cdots\#a_n\# \qquad \acute{w}{=}a_1\#a_2\#\cdots\#a_n\# \qquad \grave{w}{=}\#a_1\#a_2\#\cdots\#a_n$$

Sei $f(\begin{bmatrix} x_1 \\ y_1 \end{bmatrix}, \ldots, \begin{bmatrix} x_k \\ y_k \end{bmatrix}) = (\underbrace{\begin{bmatrix} \bar{x}_1 \\ \grave{y}_1 \end{bmatrix}}_{(x'_1,y'_1)}, \underbrace{\begin{bmatrix} \acute{x}_1 \\ \grave{y}_1 \end{bmatrix}}_{(x'_2,y'_2)}, \ldots, \underbrace{\begin{bmatrix} \acute{x}_k \\ \grave{y}_k \end{bmatrix}}_{(x'_{k+1},y'_{k+1})}, \underbrace{\begin{bmatrix} \$ \\ \#\$ \end{bmatrix}}_{(x'_{k+2},y'_{k+2})})$

- $1, i_2, \ldots, i_m$ Lösung für $K \Rightarrow 1, i_2{+}1, \ldots, i_m{+}1, \ldots, k{+}2$ Lösung für $f(K)$.

MPCP $\leq$ PCP

Lemma

MPCP $\leq$ PCP

Beweis: Gesucht: f mit: K MPCP-lösbar g.d.w. $f(K)$ PCP-lösbar.

Für $w = a_1 \cdots a_n \in \Sigma^+$ sei:

$$\bar{w}{=}\#a_1\#a_2\#\cdots\#a_n\# \qquad \acute{w}{=}a_1\#a_2\#\cdots\#a_n\# \qquad \grave{w}{=}\#a_1\#a_2\#\cdots\#a_n$$

Sei $f(\begin{bmatrix} x_1 \\ y_1 \end{bmatrix}, \ldots, \begin{bmatrix} x_k \\ y_k \end{bmatrix}) = (\underbrace{\begin{bmatrix} \bar{x}_1 \\ \grave{y}_1 \end{bmatrix}}_{(x'_1,y'_1)}, \underbrace{\begin{bmatrix} \acute{x}_1 \\ \grave{y}_1 \end{bmatrix}}_{(x'_2,y'_2)}, \ldots, \underbrace{\begin{bmatrix} \acute{x}_k \\ \grave{y}_k \end{bmatrix}}_{(x'_{k+1},y'_{k+1})}, \underbrace{\begin{bmatrix} \$ \\ \#\$ \end{bmatrix}}_{(x'_{k+2},y'_{k+2})})$

- $1, i_2, \ldots, i_m$ Lösung für $K \Rightarrow 1, i_2{+}1, \ldots, i_m{+}1, \ldots, k{+}2$ Lösung für $f(K)$.
- $i_1, \ldots, i_m$ Lösung für $f(K) \Rightarrow i_1, i_2 - 1, \ldots, i_{m-1} - 1$ Lösung für K

  Für Lösungen muss gelten: $i_1 = 1$, $\begin{bmatrix} x_{i_m} \\ y_{i_m} \end{bmatrix} = \begin{bmatrix} \$ \\ \#\$ \end{bmatrix}$ und $\begin{bmatrix} x_{i_j} \\ y_{i_j} \end{bmatrix} = \begin{bmatrix} \acute{x}_{j(r)} \\ \grave{y}_{j(r)} \end{bmatrix}$ für $2 \leq i_j \leq i_{m-1}$

$H \leq$ MPCP

Lemma

$H \leq$ MPCP.

Beweis:

- $m\#w$ mit Turingmaschinenbeschreibung m und Eingabe w
- Erstelle MPCP-Instanz $K = f(m\#w)$, so dass TM M_m auf Eingabe w genau dann anhält, wenn K lösbar.
- Sei $M_m = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$.
- Alphabet für das MPCP $\Gamma \cup Z \cup \{\#\}$.
- Idee Lösung des MPCP simuliert Transitionsfolge der TM.
- Erstes Wortpaar (mit dem jede Lösung anfangen muss):
$$\left[\begin{array}{c} x_1 \\ y_1 \end{array}\right] = \left[\begin{array}{c} \# \\ \#z_0w\# \end{array}\right]$$
- Weitere Paare lassen sich in Gruppen von Regeln aufteilen
- Kopierregeln, Transitionsregeln, Löschregeln, Abschlussregeln

$H \leq$ MPCP: Kopierregeln

- $\left[\begin{array}{c} a \\ a \end{array}\right]$ für alle $a \in \Gamma \cup \{\#\}$

$H \leq$ MPCP: Transitionsregeln

- $\left[\begin{array}{c} za \\ z'c \end{array}\right]$ falls $\delta(z,a) = (z', c, N)$
- $\left[\begin{array}{c} za \\ cz' \end{array}\right]$ falls $\delta(z,a) = (z', c, R)$
- $\left[\begin{array}{c} bza \\ zbc' \end{array}\right]$ falls $\delta(z,a) = (z', c, L)$ für alle $b \in \Gamma$
- $\left[\begin{array}{c} \#za \\ \#z'\square c \end{array}\right]$ falls $\delta(z,a) = (z', c, L)$
- $\left[\begin{array}{c} z\# \\ z'c\# \end{array}\right]$ falls $\delta(z,\square) = (z', c, N)$
- $\left[\begin{array}{c} z\# \\ cz'\# \end{array}\right]$ falls $\delta(z,\square) = (z', c, R)$
- $\left[\begin{array}{c} bz\# \\ z'bc\# \end{array}\right]$ falls $\delta(z,\square) = (z', c, L)$ für alle $b \in \Gamma$

$H \leq$ MPCP: Löschregeln und Abschlussregeln

Löschregeln:

- $\left[\begin{array}{c} az_e \\ z_e \end{array}\right]$ für alle $a \in \Gamma$, $z_e \in E$
- $\left[\begin{array}{c} z_e a \\ z_e \end{array}\right]$ für alle $a \in \Gamma$, $z_e \in E$

Abschlussregeln:

- $\left[\begin{array}{c} z_e \# \# \\ \# \end{array}\right]$ für alle $z_e \in E$

$H \leq$ MPCP: Korrespondenz

Wenn TM akzeptierenden Lauf hat, dann gibt es Folge

$$K_0 \vdash K_1 \vdash \ldots \vdash K_n,$$

wobei $K_0 = z_0 w$ und $K_n = u z_e v$ für ein $z_e \in E$.
Dann hat das MPCP eine Lösung, die oben und unten das Wort

$$\#K_0\#K_1\#\cdots\#K_n\#K_{n+1}\#\cdots\#K_m\#\#$$

erzeugt, wobei $K_m = z_e$ und jedes K_i mit $n+1 \leq i \leq m$ jeweils aus K_{i-1} entsteht durch Löschen eines der benachbarten Zeichen von z_e in $u' z_e v'$ entsteht.

$H \leq$ MPCP: Korrespondenz (2)

Obere Folge hinkt der unteren um eine Konfiguration hinterher
oben: $\#K_1\#K_2\#\cdots\#K_i\#$
unten: $\#K_1\#K_2\#\cdots\#K_i\#K_{i+1}\#$
Verlängerung, um die nächste:

- Kopierregel anwenden bis in die Nähe des Zustands
- Dann Überführungsregel anwenden
- Kopierregel anwenden zum Vervollständigen

Ab K_n: Löschregeln anwenden, um die Symbole auf dem Band zu löschen.
Untere Folge hat $z_e\#$ stehen, dann Abschlussregel anwenden.

Umgekehrte Richtung

Jede Lösung für das MPCP erzeugt eine akzeptierende Konfigurationsfolge.
Schließlich prüfe, dass f berechenbar ist.
Daher folgt: $m\#w \in H \iff MPCPf(m\#w)$ lösbar

Unentscheidbarkeit PCP und MPCP

Satz

Das Postsche Korrespondenzproblem (sowie das modifizierte Postsche Korrespondenzproblem) ist unentscheidbar.

Beweis: Da H unentscheidbar ist und $H \leq \text{MPCP} \leq \text{PCP}$ gilt, folgt, dass MPCP als auch PCP unentscheidbar sind.

01-PCP

Lemma (Unentscheidbarkeit des 01-PCP)

Das Postsche Korrespondenzproblem über dem Alphabet Σ mit $|\Sigma| = 2$ (01-PCP) ist unentscheidbar.

Beweis:

- Reduziere PCP auf 01-PCP
- Sei $\Sigma = \{0, 1\}$.
- Sei $K = (x_1, y_1), \ldots, (x_k, y_k)$ eine Instanz des PCP über dem Alphabet $\{a_1, \ldots, a_j\}$.
- Sei $f(\varepsilon) = \varepsilon$, $f(a_i) = 10^i$, $f(a_i w) = f(a_i) f(w)$ und $f(K) = (f(x_1), f(y_1)), \ldots, (f(x_k), f(y_k))$.
- Dann ist $f(K)$ eine Instanz des 01-PCPs und offensichtlich gilt: $i_1, \ldots, i_n$ ist eine Lösung für K g.d.w. $i_1, \ldots, i_n$ ist eine Lösung für $f(K)$.
- f ist Turingberechenbar und daher folgt PCP $\leq$ 01-PCP

PCP mit unärem Alphabet

Lemma

Das PCP für unäre Alphabete ist entscheidbar.

Beweis:

- Alle Spielsteine von der Form $\left[\begin{array}{c} a^n \\ a^m \end{array}\right]$.
- Wenn für alle (x_i, y_i): $|x_i| < |y_i|$, dann gibt es keine Lösung.
- Wenn für alle (x_i, y_i): $|x_i| > |y_i|$, dann gibt es keine Lösung.
- Wenn $(x_i, y_i) = (a^n, a^{n+r})$ und $(x_j, y_j) = (a^{m+s}, a^m)$ mit $s, r \geq 0$, dann ist das PCP immer lösbar:
  Die Lösung ist $\underbrace{i, \ldots, i}_{s\text{-mal}}, \underbrace{j, \ldots, j}_{r\text{-mal}}$, denn:
  oben $a^{s \cdot n + r \cdot (m+s)}$ und unten $a^{s \cdot (n+r) + r \cdot m}$.
  Daher oben wie unten $(sn + rm + rs)$-viele a's

Anzahl k der Spielsteinarten beschränken

PCP mit k-vielen verschiedenen Spielsteinarten:

- $k = 1$ oder $k = 2$: als entscheidbar gezeigt, im Jahr 1982
- $k \geq 5$: als unentscheidbar gezeigt im Jahr 2015
  (vorher war $k \geq 7$ bekannt (1996)
- $k = 3, 4$: unbekannt

PCP semi-entscheidbar

PCP ist semi-entscheidbar:

- Probiere alle Folgen von i-Spielsteinen aus.
- Lässe i wachsen.

Findet Lösung, wenn eine existiert, in endlich vielen Schritten, aber nichtterminiert, wenn keine Lösung existiert.

Universelle Turingmaschine

Da $H \leq$ PCP folgt auch, dass H semi-entscheidbar ist.

Daher: Es gibt Turingmaschine U, die die sich bei Eingabe $w\#x$ so verhält wie M_w auf Eingabe x.

Die TM U nennt man eine Universelle Turingmaschine:

- verhält sich wie ein Interpreter für Turingmaschinen
- wird durch die Eingabe w programmiert und x ist dann die eigentliche Eingabe für das Programm.

Weitere Unentscheidbarkeitsresultate

In der Zentralübung zeigen wir weitere Unentscheidbarkeitsresultate.
Inbesondere für Grammatik-Probleme, durch Reduktion von PCP auf die Probleme.

Zusammenfassung

- Entscheidbarkeit, Semi-Entscheidbarkeit
- Das Halteproblem ist unentscheidbar!
- Reduktion $L_1 \leq L_2$ als Werkzeug zum Nachweis der Unentscheidbarkeit / Entscheidbarkeit
- PCP als „einfaches“ unentscheidbares Problem