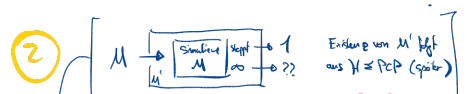
BISHER

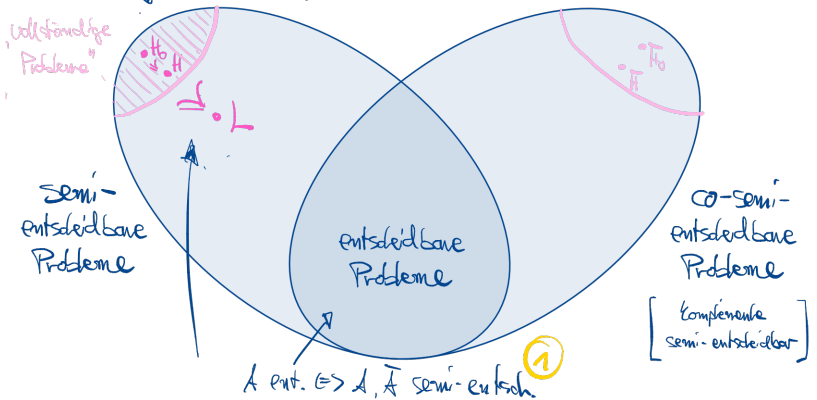


2

$H, H_0 \in \{\text{semi-entscheidbar}\} \setminus \{\text{entscheidbar}\}$

alle Sprachen $\mathcal{P}(\Sigma^*)$

& $H \leq H_0$ ⇒ H n.e. ⇒ H_0 n.e.; H_0 semi-e. ⇒ H semi-e.



A entsch. ⇔ A, $\bar{A}$ semi-entsch. (1)

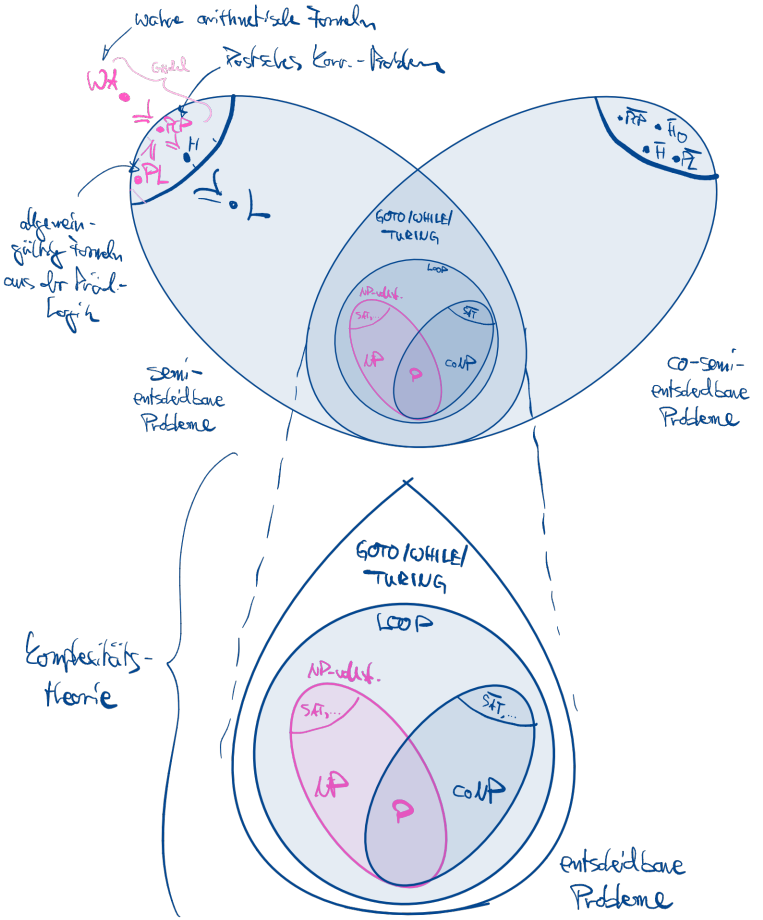
3 $\forall L$ semi-entscheidbar : $L \leq H$ ⇒ „H vollständig"

L semi-entsch. :⇔ $\exists M$ mit $L = T(M)$

Setze $f_L(x) := (\tilde{M}, x)$ $\forall x$ ⇒ $x \in L \Leftrightarrow f_L(x) \in H$ $\forall x$

$\tilde{M}$ verhält sich wie M $\forall x \in T(M)$, und hält endlos falls $x \notin T(M)$

VORSCHAU



VORSCHAU

Der Gödelsche Unvollständigkeitssatz

SATZ

Es gibt Formeln innerhalb der (elementaren) Zahlentheorie / Arithmetik ① die zwar wahr ② aber unbeweisbar ③ sind

① Prädikatenlogische Formeln mit Prädikaten $\{+, *, =\}$ auf $\mathbb{N}$

- $\forall n \exists n$ „$n < n$" $\} = \exists k\ n = n+k+1$ — wegen strikt kleiner „$<$"
- „3 ist Primzahl" $:= \forall x \forall y \left(3 = x*y \Rightarrow x=1 \vee y=3\right)$
- $\neg \forall x \exists y \left(x=0 \vee x*y=1\right)$

  falsch, da $y \in \mathbb{N}$ & Lösung wäre $y = \frac{1}{x}$

Formeln: $term_1 = term_2$, $\neg F$, $F \wedge G$, $F \vee G$, $\forall x F$, $\exists x F$ (für Formeln F, G, Variablen x)

Terme: Konstanten $n \in \mathbb{N}$, Variablen $x \in \mathbb{N}$, $term_1 + term_2$, $term_1 * term_2$

= „Theorie natürlicher Zahlen mit $+, \cdot$"

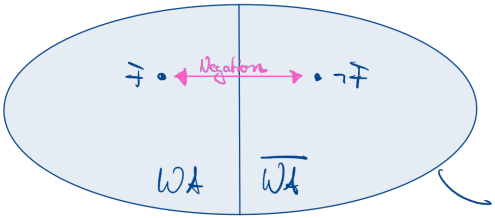
② „F ist wahr": Gleichheit auf $\mathbb{N}$ gilt & logische Aussagen sind wahr (mit Bedeutung von $+, *$ auf $\mathbb{N}$) (mit Bedeutung von $\neg, \wedge, \vee, \forall, \exists$ wie gehabt)

$Wt := \{F$ arithm. Formel $\mid F$ ist wahr (und hat keine freien Variablen$\}$

③ Wt ist nicht entscheidbar

BEW.

$\overline{Wt} = \{F \mid \neg F \in Wt\}$



$Wt \leq \overline{Wt}$ & $\overline{Wt} \leq Wt$ $\}$ $Wt \sim \overline{Wt}$ „äquivalent" (im Bezug auf $\leq$)

mit $f: F \mapsto \neg F$

Wäre Wt semi-entscheidbar, so auch $\overline{Wt}$ ($\overline{Wt} \leq Wt$)

Also wäre Wt entscheidbar. Wir zeigen das im Folgenden: Wt n.e.

$\Rightarrow$ Wt nicht semi-entscheidbar

BEWEIS IDEE

$H \leq Wt$ (dann H n.e. $\Rightarrow Wt$ n.e.)

über den Umweg von PCP:

$H \leq PCP \leq Wt$

PCP

(ab Skript S. 44)

BEISPIEL

gegeben: $\left(\begin{smallmatrix}23\\31\end{smallmatrix}\right)_{i=1}, \left(\begin{smallmatrix}1\\12\end{smallmatrix}\right)_{i=2}, \left(\begin{smallmatrix}31\\1\end{smallmatrix}\right)_{i=3}, \left(\begin{smallmatrix}123\\3\end{smallmatrix}\right)_{i=4}, \left(\begin{smallmatrix}22\\11\end{smallmatrix}\right)_{i=5}$ Wortpaare über Alphabet $\{1,2,3\}$

gesucht: Anordnung sodass durch Verkettung oben und unten das gleiche Wort entsteht

[nicht alle Paare müssen benutzt werden & beliebige Wiederholung erlaubt]

Lösung: $(i_1, i_2, i_3, i_4) = (2,1,2,4)$

("Ja-Instanz")

$$\left[\begin{array}{llll|l} 1 & 23 & 1 & 123 & = 123\,1123 \\ 12 & 31 & 12 & 3 & = 123\,1123 \end{array}\right] \checkmark$$

DEF

$$PCP := \left\{ (x_i, y_i)_{i=1}^{k} \text{ mit } x_i, y_i \in \Sigma^* \;\middle|\; \exists i_1, \dots, i_n \in \{1, \dots, k\} \;(n \geq 1) \;:\; x_{i_1} \circ \dots \circ x_{i_n} = y_{i_1} \circ \dots \circ y_{i_n} \right\}$$

gesuchte Liste von Wortpaaren — Validity der Wörter

- Wortpaare dürfen weggelassen werden $\{i_1, \dots, i_n\} \subset \{1, \dots, k\}$
- Jedes Wortpaar darf beliebig oft vorkommen (oft $n \gg k$)

(z.B. kürzeste Lösung von $\left(\left(\begin{smallmatrix}001\\0\end{smallmatrix}\right), \left(\begin{smallmatrix}01\\011\end{smallmatrix}\right), \left(\begin{smallmatrix}01\\101\end{smallmatrix}\right), \left(\begin{smallmatrix}10\\001\end{smallmatrix}\right)\right)$ hat Länge $n = 66$)

BEWEISIDEE

PCP ist semi-entscheidbar:

Lösungslänge

$(x_i, y_i)_{i=1}^{k}$ → [für aufsteigendes $n \geq 1$ teste alle möglichen Indexfolgen] — stoppt → 1 ; ?? stoppt nicht falls es keine Lösung gibt

Aber ($H \leq PCP$): PCP ist nicht entscheidbar

[d.h. es gibt auch kein anderes M mit $\to M \to$ 1 ⇒ hat Lsg, 0 ⇒ hat keine Lsg.]

über den Umweg von MPCP ($H \leq MPCP \leq PCP$):

DEF

$$MPCP := \left\{ (x_i, y_i)_{i=1}^{k} \text{ mit } x_i, y_i \in \Sigma^* \;\middle|\; \exists i_1, \dots, i_n \in \{1, \dots, k\} \;(n \geq 1) \;:\; x_1 \circ x_{i_1} \circ \dots \circ x_{i_n} = y_1 \circ y_{i_1} \circ \dots \circ y_{i_n} \right\}$$

= Lösung wie in PCP aber mit $i_1 = 1$

LEMMA

$MPCP \leq PCP$

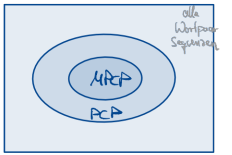
[d.h. $\exists$ berechenbares $f: \underbrace{\Sigma}_{(x_i, y_i)_{i=1}^{k}} \mapsto \underbrace{\Sigma}_{(x_i', y_i')_{i=1}^{k'}}$, s.d. $\xi \in MPCP \Leftrightarrow f(\xi) \in PCP$]

BEW

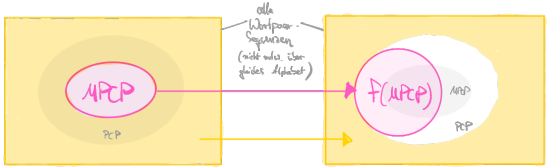
f = id eignet sich nicht:

zwar gilt $\xi \in MPCP \Rightarrow \xi \in PCP$

aber: $\xi \in PCP \not\Rightarrow \xi \in MPCP$



Stattdessen muss f so aussehen:

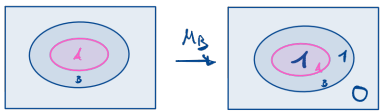


$\mathcal{E} \in$ MPCP $\rightarrow$ $f(\mathcal{E}) \in$ PCP

$\mathcal{E} \notin$ MPCP $\rightarrow$ $f(\mathcal{E}) \notin$ PCP

Erinnerung: Grund für die Def. von $A \leq_f B :\Leftrightarrow x \in A \Leftrightarrow f(x) \in B$ ist, dass wir wollen: B entscheidbar $\Rightarrow$ A entscheidbar

$A \subset B$ ist dafür nicht ausreichend:

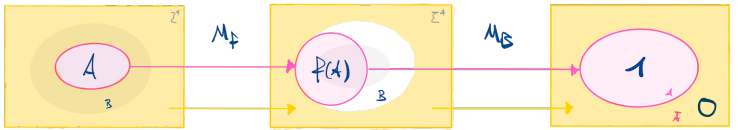


$x \Rightarrow$ [M_B] $\rightarrow 1 \Rightarrow x \in A$; $\rightarrow 0 \Rightarrow x \notin B$

berechnet $1_B(x)$

aber: Nur mit M_B kann man nicht z.B. $A \subset B$ & $B \setminus A \subset \overline{B}$ unterscheiden!

↳ Dazu benutzen wir f, denn $B \setminus A$ wird durch f auf $\overline{B}$ abgebildet



$(M_f; M_B)$ Hintereinander ausführen

berechnet $1_A = 1_B \circ f$

Beweis

Idee: Bilde Wortpaar-Sequenzen $((x_i, y_i))_{i=1}^{k} = \mathcal{E} \mapsto \tilde{\mathcal{E}} = ((\tilde{x}_j, \tilde{y}_j))_{j=1}^{\tilde{k}}$ so ab,

(i) dass Lösungen erhalten bleiben

(ii) dass jede mögliche Lösung für $\tilde{\mathcal{E}}$ auch mit $j=1$ starten muss

Beispiel: $\left(\begin{pmatrix}1\\101\end{pmatrix}, \begin{pmatrix}10\\00\end{pmatrix}, \begin{pmatrix}011\\11\end{pmatrix}\right) =: \mathcal{E} \in$ MPCP (mit $(1,3,2,3)$)

($i=1$, $i=2$, $i=3$)

$$\tilde{\mathcal{E}} = \left(\begin{pmatrix}\#1\#\\\#1\#0\#1\end{pmatrix}, \begin{pmatrix}1\#\\\#1\#0\#1\end{pmatrix}, \begin{pmatrix}1\#0\#\\\#0\#0\end{pmatrix}, \begin{pmatrix}0\#1\#1\#\\\#1\#1\end{pmatrix}, \begin{pmatrix}\$\\\#\$\end{pmatrix}\right)$$

($j=1$, $j=2$, $j=3$, $j=4$, $j=5$)

wird gebraucht, falls $i=1$ zusätzlich in der Mitte vorkommt

$\Rightarrow$ (i) & (ii) erfüllt!

(ii): $j=1$ ist einziges Paar mit führendem $\#$ in $\tilde{x}_j$

(1,3,2,3) Lösung für $\mathcal{E}$ $\rightsquigarrow$ (1,4,3,4,5) Lösung für $\tilde{\mathcal{E}}$

$$\left[\begin{array}{cccc} 1\,011 & 10 & 011 \\ 101\,11 & 00 & 11 \\ 1 & 3 & 2\;3 \end{array}\right] \rightsquigarrow \left[\begin{array}{ccccc} \#1\# & 0\#1\#1\# & 1\#0\# & 0\#1\#1\# & \$ \\ \#1\#0\#1 & \#1\#1 & \#0\#0 & \#1\#1 & \#\$ \\ 1 & 4 & 3 & 4 & 5 \end{array}\right]$$

Im Allgemeinen:

f: Wortpaar-Sequenzen über Σ $\longrightarrow$ Wortpaar-sequenzen über $\tilde{\Sigma} = \Sigma \cup \{\#, \$\}$,

$$\left(\begin{pmatrix} x_1 \\ y_1 \end{pmatrix}, \dots, \begin{pmatrix} x_k \\ y_k \end{pmatrix}\right) \longmapsto \left(\begin{pmatrix} \tilde{x}_1 \\ \grave{y}_1 \end{pmatrix}, \begin{pmatrix} \grave{x}_1 \\ \acute{y}_1 \end{pmatrix}, \dots, \begin{pmatrix} \grave{x}_k \\ \acute{y}_k \end{pmatrix}, \begin{pmatrix} \$ \\ \#\$ \end{pmatrix}\right)$$

wobei $x = ab\cdots c$, $y = ab\cdots c$ $\Rightarrow$
$\tilde{x} := \#a\#b\cdots\#c\#$
$\grave{x} := a\#b\cdots\#c\#$
$\acute{y} := \#a\#b\cdots\#c$

Per Konstruktion: $L \in$ MPCP $\Leftrightarrow f(L) \in$ PCP

$\Rightarrow$ ✓ wegen (i)

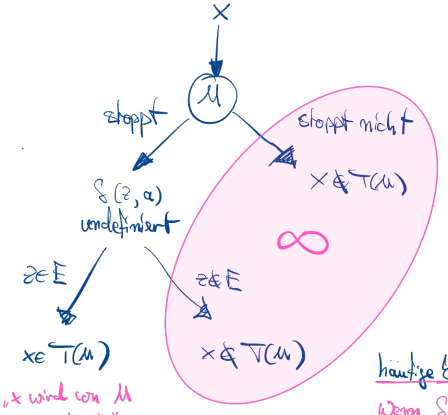
$\Leftarrow$ falls $(i_1, \dots, i_n)$ PCP-Lösung für $f(L)$, dann muss $i_1 = 1$ sein also kann eine entspr. MPCP-Lsg für L konstruiert werden

∎

Lemma: $H \leq$ MPCP $\leq$ PCP ✓ ($\Rightarrow$ PCP nicht entscheidbar)

d.h. $\exists f$ berechenbar : $(M,x) \xmapsto{f} (x_i, y_i)_{i=1}^{k}$ s.t. $\begin{cases} M(x) \text{ stoppt} \\ \Leftrightarrow \\ f(M,x) \in \text{MPCP} \end{cases}$

Beh.



häufige Konvention:
Wenn $\delta(z,a)$ undef. & $z \notin E$ → Endlosschleife

$\Downarrow$

$(M,x) \in H \Leftrightarrow x \in T(M) \Leftrightarrow \exists \alpha, \beta \in \Gamma^*, z_e \in Z, z_e \in E : z_0 x \vdash^* \alpha z_e \beta$

$\Leftrightarrow \exists \alpha_1, \dots \alpha_n, \beta_1, \dots \beta_n \in \Gamma^*, z_1, \dots, z_n \in Z$ ——"—— :

$z_0 x \vdash \alpha_1 z_1 \beta_1 \vdash \cdots \vdash \alpha_n z_n \beta_n \vdash \alpha z_e \beta$ } (*)

SKRIPT

- Repräsentiere Berechnungspfad von M bei Eingabe x als String
  z.B.: „$z_0 x \# \alpha_1 z_1 \beta_1 \# \cdots \# \alpha_n z_n \beta_n \# \alpha z_e \beta$"

- Wähle Wortpaare für MPCP so, dass

  Wortpaar-Sequenz $\in$ MPCP $\Leftrightarrow$ Berechnungspfad erzeugt durch δ führt zu z_e

Idee: Repräsentiere mögliche δ-Übergänge als Tupel $(\alpha z \beta, \alpha' z' \beta')$ wenn $\alpha z \beta \overset{\delta}{\vdash} \alpha' z' \beta'$ & füge # als Trennzeichen ein:

$$\underbrace{\binom{\#}{\#z_0x\#}, \binom{z_0x\#}{\alpha_1 z_1 \beta_1 \#}, \cdots, \binom{\alpha_n z_n \beta_n \#}{\alpha z_e \beta \#}, \binom{\alpha z_e \beta \# \#}{\#}}_{\text{Berechnungspfad}}$$

Problem: Wir würden ∞-viele PCP-Paare benötigen, da $\alpha, \beta \in \Gamma^*$ beliebig lang sein können & Γ darf nur auf berechenbare Weise von (M, x) abhängen

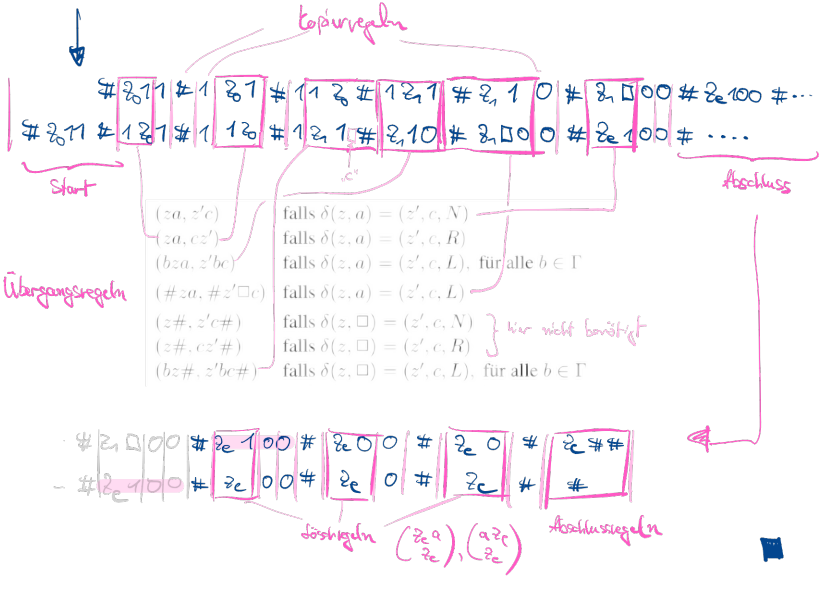
Beobachtung: In $\alpha z \beta \vdash \alpha' z' \beta'$ wird nur ein Symbol geändert in β & Lesekopf wird bewegt

$\Rightarrow$ können $\binom{\alpha z \beta}{\alpha' z' \beta'}$ aufteilen in konstante Teile & veränderte Teile

Beispiel: Addiere 1 zu Binärstring (Skript S. 12)

z.B. 11 → [M] → 100
(3 → +1 → 4)

$\hookrightarrow$ „$z_0 11 \# 1 z_0 1 \# 11 z_0 \# 1 z_1 1 \# z_1 10 \# z_1 \square 00 \# z_e 100$"



Zwischenergebnis: $\mathcal{H} \leq MPCP \leq PCP$

SATZ

MPCP & PCP sind nicht entscheidbar

aber: MPCP & PCP semi-entscheidbar [schrittweises Testen aller möglichen Kombinationen]

$\mathcal{H}$ semi-entscheidbar [d.h. $\exists$ „universelle TM" U mit Input (M,x), die stoppt falls M stoppt]

Für den Beweis von Gödel's Unvollständigkeitssatz (VL n.e.) fehlt nur noch:

SATZ

PCP $\leq$ WA (Skript S. 52f)

[d.h. $\exists f$ berechenbar: $K = ((x_i, y_i))_{i=1}^{k} \mapsto F$ arithm. Formel s.d. $K \in PCP \Rightarrow F$ wahr, $K \notin PCP \Rightarrow F$ nicht wahr]

BEWEIS

Beispiel: $\left(\begin{pmatrix}1\\101\end{pmatrix}, \begin{pmatrix}10\\00\end{pmatrix}, \begin{pmatrix}011\\11\end{pmatrix}\right)$ mit Lösung $(1,3,2,3)$

① Finde Funktionen $f_1, f_2, f_3, g_1, g_2, g_3$ s.d.

$$\underbrace{f_3(f_2(f_3(f_1(0))))}_{(*)} = g_3(g_2(g_3(g_1(0)))) \iff ((x_i, y_i))_{i=1}^{3} \text{ hat Lösung } (1,3,2,3)$$

② Wir schreiben (*) als wahre arithmetische Formel

zu ①: Wir können die Verkettung zweier Binärstrings t, x erzeugen durch eine Funktion der Form

$f_x(t) := 10\ldots0 * t + x$

(10…0: Anzahl der „0'en" := Wortlänge von x; $*$: stellt sicher, dass $t \& x$ bei der Addition nicht vermischt werden; $+$: arithmetische Operation)

[z.B. $t = 10$, $x = 101$ $\Rightarrow t \circ x = 10101 = 1000 \cdot 10 + 101 = f_{101}(10)$]

Wir schreiben im Folgenden

$f_i(t) := f_{x_i}(t)$, $g_i(t) := f_{y_i}(t)$

$\binom{x_1}{y_1} = \binom{1}{101} \rightsquigarrow f_1(t) = 10t + 1,\ g_1(t) = 1000t + 101$

$\binom{x_2}{y_2} = \binom{10}{00} \rightsquigarrow f_2(t) = 100 \cdot t + 10,\ g_2(t) = 100 \cdot t$

$\binom{x_3}{y_3} = \binom{011}{11} \rightsquigarrow f_3(t) = 1000 \cdot t + 011,\ g_3(t) = 100 \cdot t + 11$

$f_3(f_2(f_3(f_1(0)))) = 101110011 = g_3(g_2(g_3(g_1(0))))$

$f_1(0) = 1$, $f_3(f_1(0)) = 1011$, $f_2(f_3(f_1(0))) = 101110$

$g_1(0) = 101$, $g_3(g_1(0)) = 10111$, $g_2(g_3(g_1(0))) = 101100$

Im Allgemeinen:

$$\left(\binom{x_i}{y_i}\right)_{i=1}^{k} \in PCP \Leftrightarrow \begin{cases} \exists n \exists i_1,\dots,i_n \in \{1,\dots,k\} \\ f_{i_n}(f_{i_{n-1}}(\cdots f_{i_1}(0))\cdots) \\ = g_{i_n}(g_{i_{n-1}}(\cdots g_{i_1}(0))\cdots) \end{cases} \quad (**)$$

zu (2): Wir schreiben (**) schrittweise als arithm. Formel

Für $w \in \mathbb{N}$ und Funktionen $f, \tilde{f}, g, \tilde{g}, f_i, g_i : \mathbb{N} \to \mathbb{N}$ $(i \in \{1,2,3\})$

- „$w = \tilde{f}(f(w_0))$": $\exists w_1 \left[f(w_0) = w_1\right] \wedge \left[\tilde{f}(w_1) = w\right]$
- „$\tilde{g}(g(w_0)) = \tilde{f}(f(w_0))$": $\exists w_1, w_2, v_1, v_2 : w_2 = v_2 \wedge \left[f(w_0) = w_1 \wedge g(v_0) = v_1\right] \wedge \left[\tilde{f}(w_1) = w_2 \wedge \tilde{g}(v_1) = v_2\right]$
- „$\exists i_1, i_2 \in \{1,2,3\}$ sd. $f_{i_2}(f_{i_1}(w_0)) = g_{i_2}(g_{i_1}(v_0))$": $\exists w_1, w_2, v_1, v_2$

$$w_2 = v_2 \wedge \begin{bmatrix} (f_1(w_0) = w_1 \wedge g_1(v_0) = v_1) \\ \vee \\ (f_2(w_0) = w_1 \wedge g_2(v_0) = v_1) \\ \vee \\ (f_3(w_0) = w_1 \wedge g_3(v_0) = v_1) \end{bmatrix} \wedge \begin{bmatrix} (f_1(w_1) = w_2 \wedge g_1(v_1) = v_2) \\ \vee \\ (f_2(w_1) = w_2 \wedge g_2(v_1) = v_2) \\ \vee \\ (f_3(w_1) = w_2 \wedge g_3(v_1) = v_2) \end{bmatrix}$$

- (**): $\exists n \exists w_0,\dots,w_n, v_0,\dots,v_n : w_0 = v_0 = 0 \wedge w_n = v_n \wedge$

$$\begin{bmatrix} \rightarrow \wedge \leftarrow \\ \vdots \\ \vee \\ \vdots \\ \leftarrow \wedge \rightarrow \end{bmatrix} \wedge \cdots \wedge \begin{bmatrix} (f_1(w_i) = w_{i+1} \wedge g_1(v_i) = v_{i+1}) \\ \vdots \\ \vee \\ \vdots \\ (f_k(w_i) = w_{i+1} \wedge g_k(v_i) = v_{i+1}) \end{bmatrix} \wedge \cdots \wedge \begin{bmatrix} \rightarrow \wedge \leftarrow \\ \vdots \\ \vee \\ \vdots \\ \leftarrow \wedge \rightarrow \end{bmatrix}$$

$w_0 \to w_1$ $\quad w_i \to w_{i+1}$ $\quad w_{n-1} \to w_n$

⇓

Um eine arithmetische Formel zu erhalten, können wir „…" zwischen „∧" & „∨" ersetzen durch „∀" & „∃".

Aber: „$\exists (u_0,\dots,u_n)$" ist nicht Teil der Arithmetik!

Hierzu benutzen wir:

Resultat aus der Zahlentheorie (Script S.53)

Für jede Familie $(u_0,\dots,u_n)$ gibt es $a,b \in \mathbb{N}$ und ein arithmetischer Term $\beta_i(a,b)$ s.d. $u_i = \beta_i(a,b)$ $\forall i = 0,\dots,n$

⇓ (**)

PCP ≤ Wℋ wird hergestellt durch die Abbildung

$((x_i,y_i))_{i=1}^{k} \mapsto$

$\exists n \exists a \exists b \exists a' \exists b'$: $\underbrace{\beta_0(a,b) = \beta_0(a',b')}_{u_0 = v_0 = 0} = 0 \wedge \underbrace{\beta_n(a,b) = \beta_n(a',b')}_{u_n = v_n}$

$\wedge \forall i<n \; \exists j \le k$: $\underbrace{f_j(\beta_i(a,b)) = \beta_{i+1}(a,b) \wedge g_j(\beta_i(a',b')) = \beta_{i+1}(a',b')}_{f_j(u_i) = u_{i+1} \wedge g_j(v_i) = v_{i+1}}$

⇓

Wℋ ist nicht entscheidbar (Gödel)

Zum Abschluss (des Teils „Berechenbarkeit") noch ein Resultat zur (offenen) Prädikatenlogik (Church, Turing 1936):

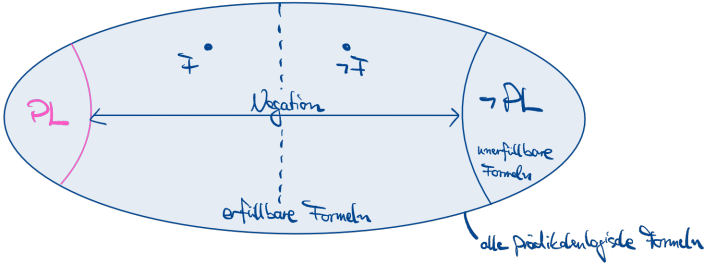
Satz

$PL := \{ F \text{ prädikatenlogische Formel} \mid F \text{ allgemeingültig} \}$

[F ist wahr in jedem Modell (Konkretisierung der Funktions- und Prädikatensymbole) sowie unter jeder Belegung aller freien Variablen]

ist nicht entscheidbar

[Bsp. für $F \in PL$: $F = (\exists y \forall x P(x,y) \Rightarrow \forall x \exists y P(x,y))$]



BEM

PL & ¬PL sind semi-entscheidbar, da z.B. Resolution ein Verfahren angibt, das stoppt falls $\neg F \in PL$

BEWEIS

Wir zeigen PCP ≤ PL, d.h.

$\exists f$ berechenbar: $k = ((x_i, y_i))_{i=1}^{k} \mapsto F_k$ s.d. $k \in PCP \Leftrightarrow F_k \in PL$

Idee:

- Repräsentiere die Erzeugung von Binärstrings $(x_{i_1} \circ \cdots \circ x_{i_n}$ & $y_{i_1} \circ \cdots \circ y_{i_n})$ durch Verschachtelung zweier beliebiger Funktionen f_0, f_1
- Überprüfe potentielle Lösung mithilfe eines zweistelligen Prädikats P

Wichtig: Anders als im Beweis von PCP ≤ Wkt, können wir nicht einfach f_0, f_1 festlegen, d.h. wir können nicht verlangen, dass z.B. $f_0(f_1(a)) = 10$, denn das wäre schon ein Modell (F_k muss aber allgemeingültig sein).

Stattdessen: Wir bilden die benötigte PCP-Struktur ab auf die Struktur der Formel:

Binärstrings ⇝ Prädikatenlog. Formeln

„0", „1", „ε"	Funktionen f_0, f_1, Konstante a
„100"	$f_0(f_0(f_1(a))) = (f_0 \circ f_0 \circ f_1)(a)$
(*) gleiche Indexreihenfolge in Verkettung der x_i & y_i	$P(u,v)$ wahr
(**) x-String = y-String	$\exists z\, P(z,z)$

Konstruktion:

- f_0, f_1 seien beliebige (prädikatenlogische) Funktionen
- für $z \in \{0,1\}^*$ sei f_z die Hintereinanderausführung von f_0, f_1 entsprechend der Reihenfolge von „0" & „1" in z. (z.B. $f_{100} = f_0 \circ f_0 \circ f_1$)
- Die Wahrheit von P(u,v) falls u,v aus f_0, f_1 in gleicher Reihenfolge aus x_i, y_i erzeugt werden wird durch eine vorgeschaltete Prämisse $A_k(P)$ erzwungen:

$$F_k := \underbrace{\left(A_k(P)\right)}_{(*)} \Rightarrow \underbrace{\exists z\, P(z,z)}_{(**)}$$

Def. von $\mathcal{A}$:

„wahr" bedeutet hier „erfüllt"

$$\mathcal{A}_\mathcal{L}(P) := \underbrace{P(f_{x_1}(a), f_{y_1}(a)) \wedge \cdots \wedge P(f_{x_\ell}(a), f_{y_\ell}(a))}_{\text{Es darf mit einem beliebigen Paar } (x_i, y_i) \text{ angefangen werden (bei MPCP wäre hier nur } P(f_{x_1}(a), f_{y_1}(a)))}$$

$\wedge$

$$\forall u,v : \underbrace{\left(P(u,v) \Rightarrow \left(P(f_{x_1}(u), f_{y_1}(v)) \wedge \cdots \wedge P(f_{x_\ell}(u), f_{y_\ell}(v))\right)\right)}_{\text{Es darf irgendein weiteres Paar } (f_{x_i}, f_{y_i}) \text{ angehängt werden}}$$

forciert gleiches Indexpaar für x & y

$\Downarrow$

$\mathcal{A}_\mathcal{L}(P)$ überträgt die PCP-Regeln vom Verketten von Strings auf das Hintereinanderschalten von Funktionen f_0, f_1

d.h. $\mathcal{A}_\mathcal{L}(P) \Rightarrow \exists z\, P(z,z)$ entspricht genau der Existenz einer PCP-Lösung für $\mathcal{L}$, d.h.

$\Downarrow$

$$\mathcal{L} \in \text{PCP} \iff \mathcal{F}_\mathcal{L} \text{ allgemeingültig}$$

$f : ((x_i, y_i))_{i=1}^k = \mathcal{L} \longmapsto \mathcal{F}_\mathcal{L} = \left(\mathcal{A}_\mathcal{L}(P) \Rightarrow \exists z\, P(z,z)\right)$

stellt die Reduktion $\text{PCP} \leq \text{PL}$ her.

∎

ÜBERSICHT

(nicht-triviale) Reduktionen

(1) $\text{MPCP} \leq \text{PCP}$

Idee: Füge Trennsymbole ein, sodass Lösungen erhalten bleiben & mit falschem Paar gestartet werden muss $[\#a\#b\#\,,\ \#a\#b\,,\ a\#b\#\,,\ \dots]$

(2) $\mathcal{H} \leq \text{MPCP}$

Idee: Repräsentiere δ-Übergänge durch PCP-Paare (Kopierpaare, Übergangspaare, Löschpaare, ...) bzw. Berechnungsplatte als PCP-Lösungen

$$\left[\#z_i x\# \cdots \#z_j y\# \rightsquigarrow \begin{pmatrix}\# \\ \#z_i x\#\end{pmatrix}, \begin{pmatrix}z_i a \\ a z_j\end{pmatrix}, \dots, \begin{pmatrix}\# \\ \#\end{pmatrix}, \begin{pmatrix}z_i a \\ z_i\end{pmatrix}, \dots, \begin{pmatrix}z_i \#\# \\ \#\end{pmatrix}\right]$$

(3) $\text{PCP} \leq \text{WA}$

Idee: Repräsentiere Verkettung von Binärstrings explizit durch Verschachtelung von Funktionen $\mathbb{N} \to \mathbb{N}$ der Form $f_x(t) = 10\dots0 \cdot t + x$, sodass PCP-Lösungen $\iff f_{x_m} \circ \cdots \circ f_{x_1}(0) = g_{y_m} \circ \cdots \circ g_{y_1}(0)$ $\rightsquigarrow$ arithm. Formel

(4) $\text{PCP} \leq \text{PL}$

Idee: Konstruiere allgemeingültige Formel, indem Verkettung von Binärstrings implizit durch Verschachtelung zweier beliebiger Funktionen f_0, f_1 repräsentiert werden & PCP-Regeln durch Prämisse erzwungen werden.

$$\left[\mathcal{A}(P) \Rightarrow \exists z\, P(z,z)\right]$$

↑ wahr, wenn $P(u,v)$ PCP-Regeln „überprüft" [$u \hat{=}$ x-string, $v \hat{=}$ y-string]