Turingsche These

> **Die Turing-berechenbaren Funktionen sind genau die im intuitiven Sinne berechenbaren Funktionen.**

Analog für den λ-Kalkül:

Die **Churchsche These**: Die λ-definierbaren Funktionen sind genau die im intuitiven Sinne berechenbaren Funktionen.

Einige unentscheidbare Probleme

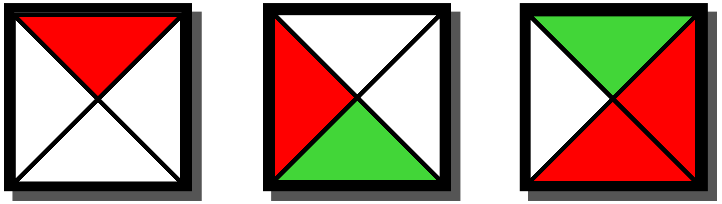
Unentscheidbare Probleme

- schon als unentscheidbar gezeigt:
  - Sprache L_d
  - Komplement von L_d
  - Halteproblem
- außerdem unentscheidbar:
  - einige Kachelprobleme bzw. (2D-)Dominoprobleme
  - Vorkommen einer 1 als Bild einer Funktion
  - Leerheitsproblem für TM
  - 10. Hilbertsches Problem
  - ...

Kachel-/Dominoprobleme

Gegeben:	Eine Menge von r Kacheltypen $\mathcal{R} = \{K_1, K_2, \ldots, K_r\}$, $n \in \mathbb{N}$ (beliebig viele Kacheln von jedem Typ)
Gesucht:	**(A)** Kann man den ersten Quadranten der euklidischen Ebene korrekt kacheln? **(B)** Kann man eine Fläche der Größe $n \times n$ korrekt kacheln?
Antwort:	JA oder NEIN

Beispiel: Kachelproblem

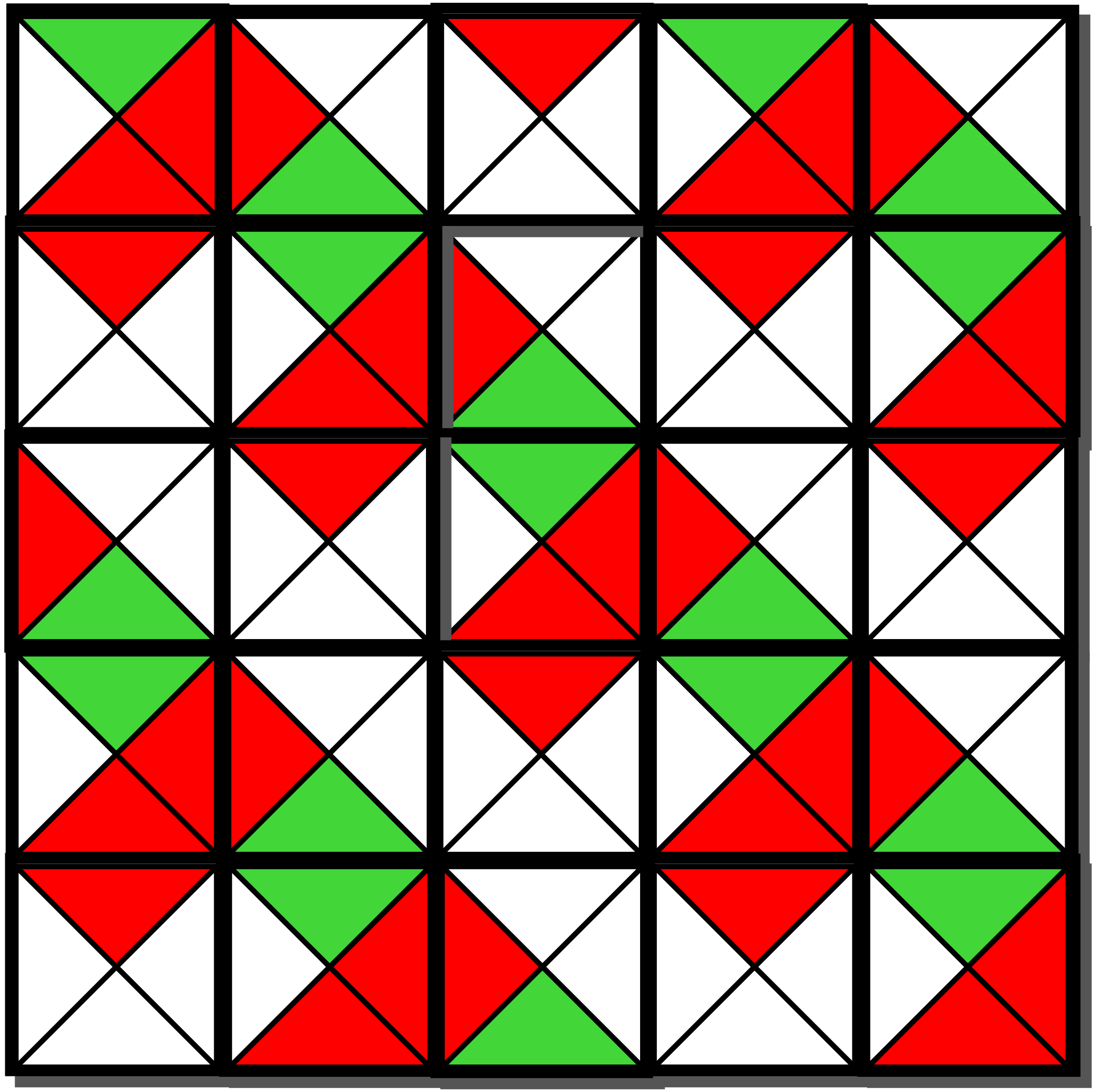


- nur gleichfarbige Seiten dürfen aneinandergelegt werden!
- Steine dürfen nicht gedreht werden!

Theorem: Das Kachelproblem **(A)** ist *unentscheidbar*.

Für **(B)** gibt es zur Zeit deterministische Verfahren nur mit exponentiellem Zeitaufwand, denn das Problem ist $\mathcal{NP}$-vollständig.

Kachelproblem (eine Lösung)



Kommt 1 als Funktionswert vor?

Definition: Mit $\mathcal{M}_T$ sei die Menge aller (Kodierungen von) Turing-Maschinen M bezeichnet, die totale Funktionen $f_M : \Sigma^* \to \{0, 1\}$ berechnen.

Gegeben:	Eine beliebige stets haltende Turing-Maschine
Gesucht:	Wird bei allen Eingaben stets die Ausgabe 0 berechnet?
Antwort:	JA oder NEIN

Beweis

Annahme der Entscheidbarkeit von
$\mathcal{P} := \{M \in \mathcal{M}_T \mid \exists w \in \Sigma^* : f_M(w) = 1\}$
$\Rightarrow$ TM $A_\mathcal{P}$ berechne $\chi_\mathcal{P}$ von $\mathcal{P}$.

Also: $$\chi_\mathcal{P}(\langle M \rangle) = \begin{cases} 1, & \text{falls } M \in \mathcal{P} \\ 0, & \text{sonst} \end{cases}$$

- Betrachte Klasse $M_{B,w}$ von TM, mit:
  - Eingabe $n \in \mathbb{N} \rightarrow$ simuliere genau n Schritte von B bei Eingabe von w,
  - halte an und gib 1 aus, sofern B nach genau n Schritten auf w hält,
  - sonst gib eine 0 aus.

Beweis (Forts.)

$$\chi_{\mathcal{P}}(\langle M_{B,w} \rangle) = 1$$

$$\iff$$

$$M_{B,w} \in \mathcal{P}$$

$$\iff$$

$M_{B,w}$ druckt bei Eingabe von n eine 1

$$\iff$$

B hält auf w nach n Schritten an

$$\iff$$

$$w \in L(B)$$

Dann entscheidet $A_{\mathcal{P}}$ aber das Halteproblem!

Widerspruch !!!

Unentscheidbarkeitsresultate

Theorem: Es gibt keinen Algorithmus, der für eine beliebige Funktion $f_M \in \mathcal{M}_T$ entscheidet, ob $f_M(w) = 1$ für mindestens ein $w \in \Sigma^*$ gilt.

Korollar: Es ist unentscheidbar, ob eine beliebige, durch eine TM definierte, entscheidbare Menge M leer ist.

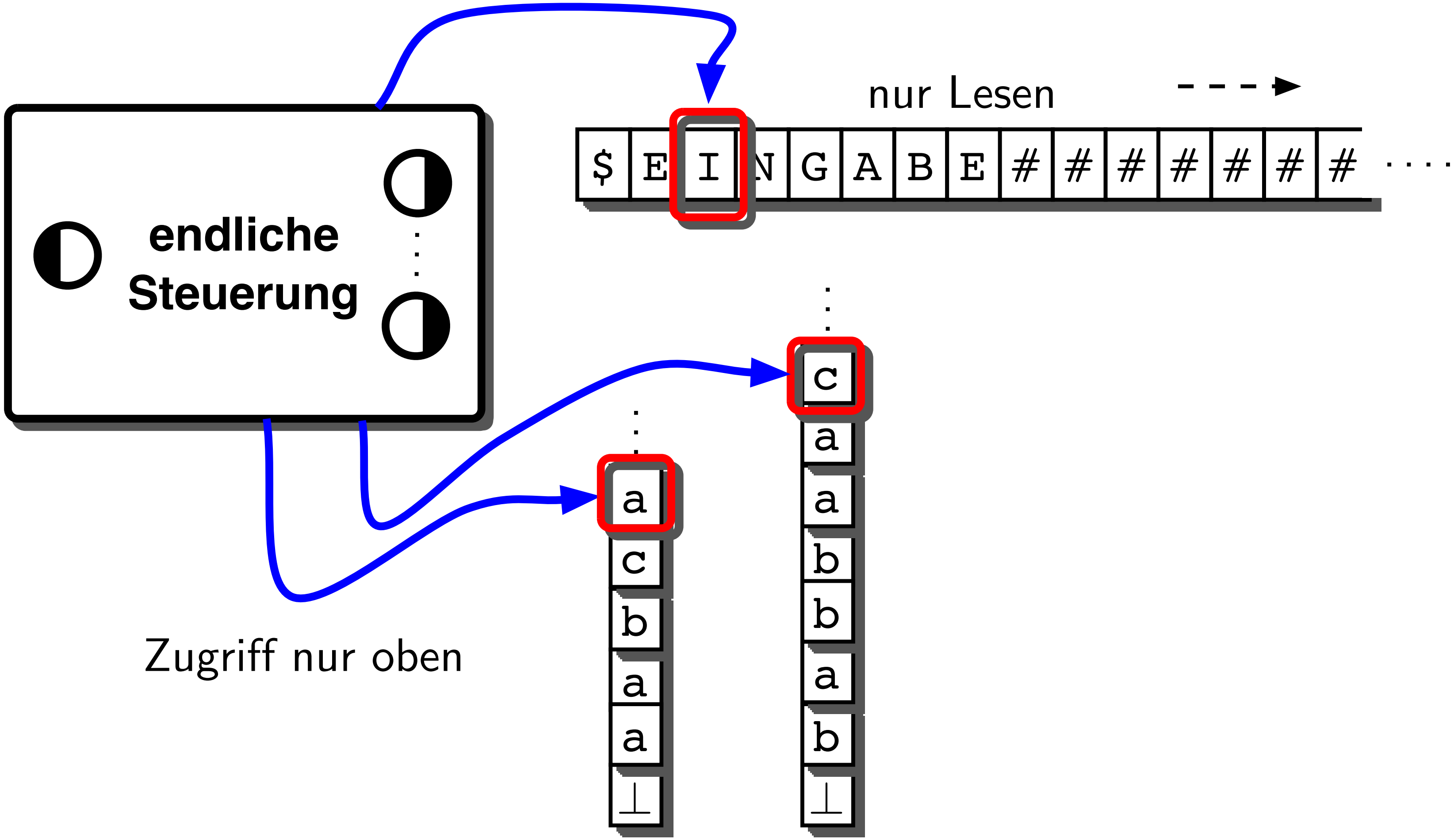
Alternativen zum durch Turing-Maschinen definierten Berechenbarkeitsbegriff

Übersicht

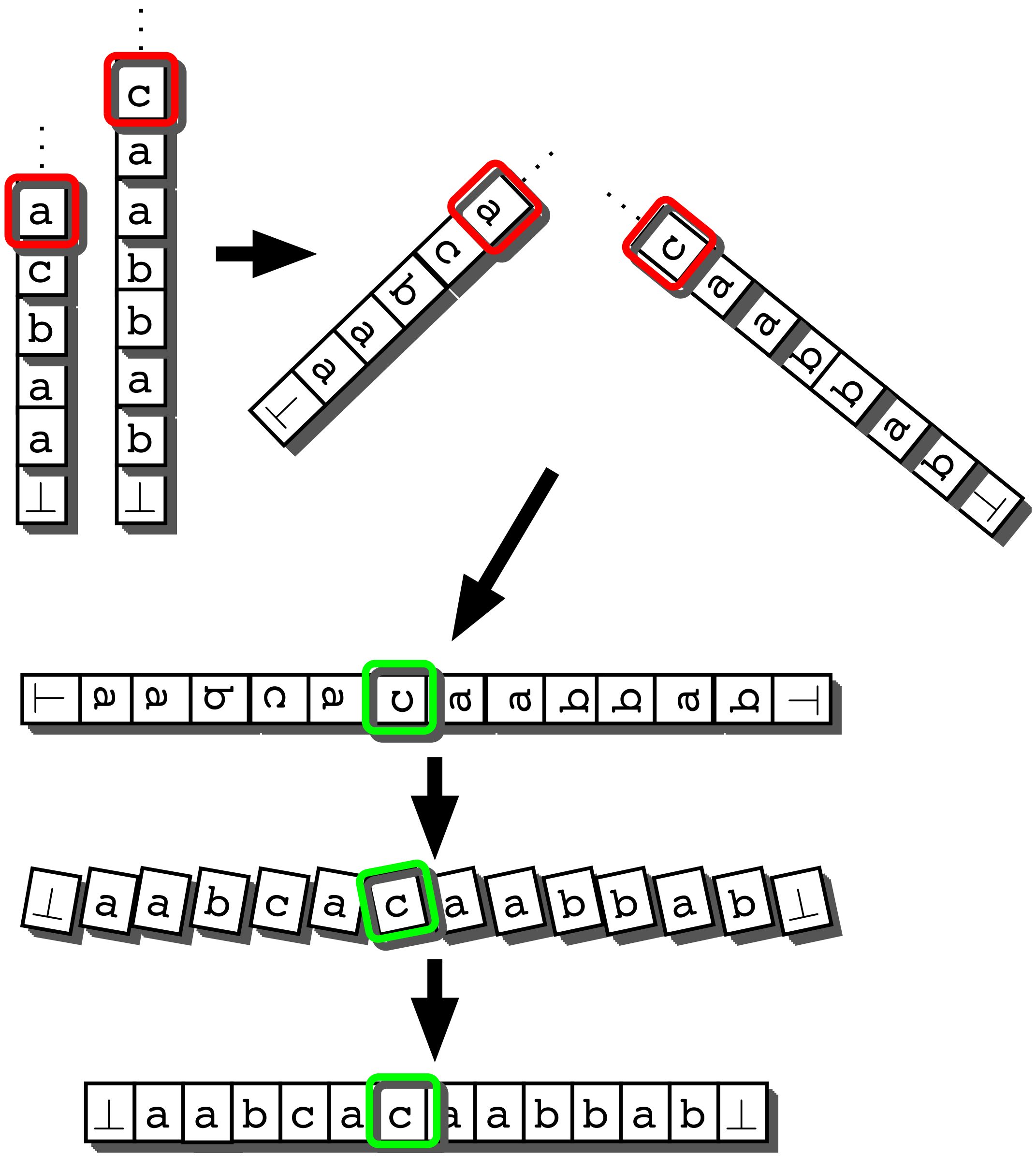
- Automatenmodelle:
  - 2-Kellerautomaten/k-Kellerautomaten
  - Zählerautomaten
- mathematische Modelle:
  - μ-rekursive Definierbarkeit
- realistische Modelle:
  - RAM (Random Access Machine)

2-Kellerautomaten

...zur Akzeptierung von Sprachen:



2 Keller = Turing-Band



Zählerautomaten

Ein k**-Zähler-Automat** ist ein k-Keller-Automat, bei dem das Kelleralphabet für die Keller (zusätzlich zu dem benutzten **Kellerboden- symbol** $\perp$) nur aus einem einzigen Zeichen, z.B. $*$, besteht.

Für $\Gamma = \{y_1, y_2, \ldots, y_{k-1}\}$ wird Kellerinhalt $y_{i_1} y_{i_2} y_{i_3} \ldots y_{i_m}$ durch die Zahl $i_1 \cdot k^{m-1} + i_2 \cdot k^{m-2} + \ldots + i_{m-1} \cdot k + i_m$ in eindeutiger Weise k-när kodiert.

$\mathtt{push}(y_i)$ entspricht der Änderung $z := z \cdot k + i$

$\mathtt{pop}$ entspricht der Änderung $z := \lfloor \frac{z}{k} \rfloor$

$\mathtt{top}= y_i$ entspricht dem Test $i = z \bmod k$

Äquivalenzen

Eine Menge M ist genau dann aufzählbar, wenn sie von einem Automaten der folgenden Art erkannt werden kann:

1. Einer deterministischen Turing-Maschine (DTM).
2. Einer nichtdeterministischen Turing-Maschine (NTM).
3. Einem 2-Keller-Automaten.
4. Einem 2-Zähler-Automaten.

Mathematische Definition der Berechenbarkeit:

primitive- und μ-Rekursion

Prinzip der primitiven Rekursion

Funktionen werden zusammengesetzt

- aus elementaren **Basisfunktionen**
  - konstante Funktionen
  - Nachfolgerfunktion
  - Projektionen (Zugriff auf eine Komponente eines Tupels)
- durch Anwendung von elementaren **Operationen**
  - Substitution (Schachtelung)
  - primitive Rekursion

Basisfunktionen

- **Nachfolgerfunktion:** $S : \mathbb{N} \to \mathbb{N} = \{0, 1, 2, \ldots\}$ mit $S(n) := n + 1$.
- **konstante Funktionen:** $C_q^r : \mathbb{N}^r \to \mathbb{N}$ für $r, q \in \mathbb{N}$ mit:

  $$C_q^r(n_1, n_2, \ldots, n_r) := q$$

  für alle r-Tupel $(n_1, n_2, \ldots, n_r) \in \mathbb{N}^r$. Die nullstellige Konstante $q \in \mathbb{N}$ ist somit: $C_q^0 := q$.
- **Projektionsfunktionen:** $U_i^r : \mathbb{N}^r \to \mathbb{N}$ mit:

  $$U_i^r(n_1, n_2, \ldots, n_r) := n_i$$

  für alle r-Tupel $(n_1, n_2, \ldots, n_r) \in \mathbb{N}^r$.

Operationen (1)

Substitution: Sind $f : \mathbb{N}^r \to \mathbb{N}$ und $g_1, g_2, \ldots, g_r : \mathbb{N}^m \to \mathbb{N}$ in $\mathcal{PR}$, so ist auch die Funktion $h : \mathbb{N}^m \to \mathbb{N}$ mit

$$h(n_1, n_2, \ldots, n_m) := f(g_1(n_1, \ldots, n_m), \ldots, g_r(n_1, \ldots, n_m))$$

in $\mathcal{PR}$.

. . . zum Beispiel: $h : \mathbb{N}^2 \to \mathbb{N}$ mit $h(x, y) := (x^2 \cdot y) + 1$ wird zusammengesetzt aus $S : \mathbb{N} \to \mathbb{N}$ und $mult : \mathbb{N}^2 \to \mathbb{N}$ mit $mult(x, y) := x \cdot y$ als:

$$h(a, b) = S(mult(mult(U_1^2, U_1^2), U_2^2))(a, b) = S(mult(mult(a, a), b))$$

Operationen (2)

primitive Rekursion: Sind $f : \mathbb{N}^r \longrightarrow \mathbb{N}$ und $g : \mathbb{N}^{r+2} \longrightarrow \mathbb{N}$ in $\mathcal{PR}$, so ist auch die folgende Funktion $h : \mathbb{N}^{r+1} \longrightarrow \mathbb{N}$ in $\mathcal{PR}$, die den folgenden *Rekursionsgleichungen* genügt:

$$\begin{array}{lrcl} a) & h(0, n_1, n_2, \ldots, n_r) & := & f(n_1, \ldots, n_r), \\ b) & h(S(n), n_1, \ldots, n_r) & := & g(n, h(n, n_1, \ldots, n_r), n_1, \ldots, n_r), \end{array}$$

h entsteht durch primitive Rekursion aus f und g.

Definition $\mathcal{PR}$

Die Menge $\mathcal{PR}$ der **primitiv rekursiven Funktionen** besteht aus den *Basisfunktionen* und den sich in endlich vielen Schritten durch *Substitution* und *primitive Rekursion* ergebenden Funktionen.

Andere Funktionen gehören nicht zu $\mathcal{PR}$.

Jede primitiv-rekursive Funktion $f \in \mathcal{PR}$ ist eine **totale** Funktion, d.h. sie ist überall definiert. Außerdem ist $f(x)$ stets eindeutig bestimmt.

Beispiel: Addition in $\mathbb{N}$

Die Addition in $\mathbb{N}$ ist durch die folgenden Rekursionsgleichungen definiert:

$add : \mathbb{N}^2 \to \mathbb{N}$ mit

$$\begin{aligned} add(0,n) &:= U_1^1(n), \\ add(S(m),n) &:= g(m, add(m,n), n)), \\ g(x,y,z) &:= S(U_2^3(x,y,z)). \end{aligned}$$

Übliche Kurzform:

$$\begin{aligned} 0 + n &:= n, \\ (m+1) + n &:= (m+n) + 1. \end{aligned}$$

Beispiel: Multiplikation in $I\!N$

Die Multiplikation in $I\!N$, $mult : I\!N^2 \to I\!N$, ist definiert durch:

$$\begin{aligned} mult(0, n) &:= C_0^1(n), \\ mult(S(m), n) &:= g(m, mult(m, n), n), \\ g(x, y, z) &:= add(U_2^3(x, y, z), U_3^3(x, y, z)). \end{aligned}$$

Kurzform:

$$\begin{aligned} 0 \cdot n &:= 0, \\ (m+1) \cdot n &:= m \cdot n + n. \end{aligned}$$

Beispiel: Fakultät

$$\begin{aligned} h(0) &:= C_1^0 \\ h(S(n)) &:= mult(S(U_1^2(n, h(n))), U_2^2(n, h(n))) \end{aligned}$$

Da $mult$ primitiv rekursiv ist, ist somit die Fakultätsfunktion _! $: I\!\!N \to I\!\!N$ als primitiv rekursiv nachgewiesen durch:

$$h(n) = n!$$

Eigenschaften

Theorem: Jede primitiv-rekursive Funktion $f : \mathbb{N}^r \to \mathbb{N}$ ist Turing-berechenbar.

$\boxed{\text{Die Umkehrung des Satzes ist jedoch } \mathbf{falsch!}}$

Denn es gilt:

Theorem: Nicht jede Turing-berechenbare Funktion ist total.

Frage: *Sind die totalen Turing-berechenbaren Funktionen genau die primitiv-rekursiven Funktionen?*

NEIN! Gegenbeispiel: Ackermann-Funktion.

Definition: Ackermann-Funktion

Die **Ackermann-Funktion** ist wie folgt definiert:

- $f(0, n) := n + 1$,
- $f(m + 1, 0) := f(m, 1)$,
- $f(m + 1, n + 1) := f(m, f(m + 1, n))$.

Die Ackermann-Funktion ist *Turing-berechenbar*, ihre Funktionswerte sind für *alle* Argumente eindeutig bestimmt.

Theorem: Die Ackermann-Funktion ist nicht primitiv-rekursiv.

Lemma zu prim. rek. Funktionen

Sei $g : \mathbb{N}^r \to \mathbb{N}$ primitiv-rekursiv. Dann gibt es ein $c \in \mathbb{N}$, so daß

$$g(n_1, n_2, \ldots, n_r) < f(c, n_1 + n_2 + \ldots + n_r)$$

für alle $(n_1, n_2, \ldots, n_r) \in \mathbb{N}^r$.

Zu jeder primitiv rekursiven Funktion gibt es eine Konstante, so daß die Ackermann-Funktion als obere Schranke dienen kann.

Beweis: Ackermann-Funktion

Beweisidee: Angenommen, die f sei primitiv-rekursiv. Dann ist auch die Funktion

$$g(n) := f(n, n)$$

primitiv-rekursiv, denn
$g(n) = f(U_1^2(n, m), U_1^2(n, m)) \in \mathcal{PR}$.
Nach Lemma gibt es ein $c \in \mathbb{N}$ mit

$$g(n) < f(c, n) \text{ für alle } n \in \mathbb{N}.$$

Für den Spezialfall $n = c$ gilt dann aber:

$$g(c) < f(c, c) = g(c).$$

Widerspruch!!!

mächtiger durch μ-Rekursion

Sei $f : \mathbb{N}^{n+1} \to \mathbb{N}$ eine beliebige partielle Funktion und $(x_1, x_2, \ldots, x_n)$ ein beliebiges, festes n-Tupel aus $\mathbb{N}^n$. Dann betrachten wir die Folge von existierenden (!) Funktionswerten:

$$
\begin{aligned}
y_0 &:= f(x_1, x_2, \ldots, x_n, 0), \\
y_1 &:= f(x_1, x_2, \ldots, x_n, 1), \\
&\vdots \\
y_k &:= f(x_1, x_2, \ldots, x_n, k), \\
&\vdots
\end{aligned}
$$

$\boxed{\text{Minimiere } k\text{, so daß } y_k = 0.}$

Definition: μ-Operator

Sei $f : I\!N^{n+1} \rightarrow I\!N$ eine partielle Funktion.
Sei ferner $M := \{i \mid f(x_1, x_2, \ldots, x_n, i) = 0\}$.
Dann ist

$$\mu y[f(x_1, x_2, \ldots, x_n, y) = 0]$$

$$:= \begin{cases} \min(M) & \text{falls } \min(M) \text{ existiert} \\ & \text{und } \forall j < \min(M).(x_1, x_2, \ldots, x_n, j) \in \mathrm{Def}(f) \\ \mathrm{div} & \text{sonst} \end{cases}$$

. . . allgemein: $\mu y[Q(\bar{x}, y)]$ für eine Eigenschaft Q.

Def.: μ-rekursive Funktionen

Eine Funktion f heißt μ**-rekursiv** (**partiell-rekursiv**) genau dann, wenn f entweder

1. eine primitiv-rekursive Grundfunktion ist oder
2. durch Substitution aus μ-rekursiven Funktionen entsteht oder
3. durch primitive Rekursion aus μ-rekursiven Funktionen entsteht oder
4. durch Anwendung der Minimalisierung (des μ-Operators) aus μ-rekursiven Funktionen entsteht.

Die partiell rekursiven Funktionen

Theorem: Die Ackermann-Funktion ist μ-rekursiv.

Theorem: Für eine n-stellige Funktion $f : \mathbb{N}^n \to \mathbb{N}$ sind folgende Aussagen äquivalent:

1. f ist durch eine 1-DTM berechenbar,
2. f ist durch eine k-DTM berechenbar,
3. f ist μ-rekursiv (partiell-rekursiv),
4. f ist durch eine RAM berechenbar.

$\vdots$

Ausblick

- Ein realistisches Maschinenmodell: die RAM;
- erweiterte Grammatiken:
  - kontextsensitive Grammatiken,
  - Phrasenstrukturgrammatiken;