Theoretische Informatik

Einheit 3

Elementare Berechenbarkeitstheorie

1. Aufzählbarkeit und Entscheidbarkeit
2. Universelle Maschinen
3. Beweistechniken für unlösbare Probleme

Untersuchung von Fragen der Berechenbarkeit unabhängig vom Modell

Berechenbarkeitstheorie

Untersuchung von Fragen der Berechenbarkeit unabhängig vom Modell

- **Was kann überhaupt berechnet werden?**
  - Welche Funktionen sind berechenbar?
  - Welche Programmeigenschaften kann man entscheiden (testen)?

Berechenbarkeitstheorie

Untersuchung von Fragen der Berechenbarkeit unabhängig vom Modell

- **Was kann überhaupt berechnet werden?**
  - Welche Funktionen sind berechenbar?
  - Welche Programmeigenschaften kann man entscheiden (testen)?
- **Wie kann man Lösungen wiederverwenden?**
  - Verwendung von Abschlußeigenschaften
  - Transformation in ein anderes Problem (Reduktion)

Berechenbarkeitstheorie

Untersuchung von Fragen der Berechenbarkeit unabhängig vom Modell

- **Was kann überhaupt berechnet werden?**
  - Welche Funktionen sind berechenbar?
  - Welche Programmeigenschaften kann man entscheiden (testen)?
- **Wie kann man Lösungen wiederverwenden?**
  - Verwendung von Abschlußeigenschaften
  - Transformation in ein anderes Problem (Reduktion)
- **Wo liegen die Grenzen?**
  - Terminierung, Korrektheit, Äquivalenz, Optimalität von Programmen

Berechenbarkeitstheorie

Untersuchung von Fragen der Berechenbarkeit unabhängig vom Modell

- **Was kann überhaupt berechnet werden?**
  - Welche Funktionen sind berechenbar?
  - Welche Programmeigenschaften kann man entscheiden (testen)?
- **Wie kann man Lösungen wiederverwenden?**
  - Verwendung von Abschlußeigenschaften
  - Transformation in ein anderes Problem (Reduktion)
- **Wo liegen die Grenzen?**
  - Terminierung, Korrektheit, Äquivalenz, Optimalität von Programmen
- **Welche Beweistechniken gibt es?**
  - Konstruktion von abstrakten Gegenbeispielen durch Diagonalisierung
  - Problemreduktion
  - Direkte Beweise (Busy Beaver)

Berechenbarkeitstheorie – Vorgehensweise

- **Konzepte modellunabhängig präzisieren**
  - Berechenbare Funktionen
  - (Semi-)entscheidbare und aufzählbare Mengen

Berechenbarkeitstheorie – Vorgehensweise

- **Konzepte modellunabhängig präzisieren**
  - Berechenbare Funktionen
  - (Semi-)entscheidbare und aufzählbare Mengen
- **Grundeigenschaften aus Modellen herleiten**
  - Zusammenhänge zwischen den Konzepten
  - Abschlußeigenschaften
  - Codierung von Programmen als Daten
  - Existenz universeller Funktionen

Berechenbarkeitstheorie – Vorgehensweise

- **Konzepte modellunabhängig präzisieren**
  - Berechenbare Funktionen
  - (Semi-)entscheidbare und aufzählbare Mengen
- **Grundeigenschaften aus Modellen herleiten**
  - Zusammenhänge zwischen den Konzepten
  - Abschlußeigenschaften
  - Codierung von Programmen als Daten
  - Existenz universeller Funktionen
- **Theorie nur auf dieser Basis weiterführen**
  - Hier: die wichtigsten Unmöglichkeitsaussagen

Theoretische Informatik

Einheit 3.1

Aufzählbarkeit und Entscheidbarkeit

1. Präzisierung der Begriffe
2. Zusammenhänge
3. Abschlußeigenschaften

Berechenbarkeit

- **Berechenbarkeit auf Worten**
  - $f{:}X^* \to Y^*$ berechenbar, falls f Turing-berechenbar

Berechenbarkeit

- **Berechenbarkeit auf Worten**
  - $f{:}X^*{\to}Y^*$ berechenbar, falls f Turing-berechenbar
- **Berechenbarkeit auf Wort-Tupeln** $f{:}X^*{\times}X^*{\to}Y^*$
  - $f{:}X^*{\times}X^*{\to}Y^*$ berechenbar, falls $f'{:}(X{\cup}\{\#\})^*{\to}Y^*$
    mit $f'(v\#w) = f(v,w)$ berechenbar

Berechenbarkeit

- **Berechenbarkeit auf Worten**
  - $f{:}X^*{\to}Y^*$ berechenbar, falls f Turing-berechenbar
- **Berechenbarkeit auf Wort-Tupeln** $f{:}X^*{\times}X^*{\to}Y^*$
  - $f{:}X^*{\times}X^*{\to}Y^*$ berechenbar, falls $f'{:}(X{\cup}\{\#\})^*{\to}Y^*$
    mit $f'(v\#w) = f(v,w)$ berechenbar
- **Berechenbarkeit auf Zahlen**
  - $f{:}\mathbb{N}{\to}\mathbb{N}$ berechenbar, falls $f_r{:}X^*{\to}X^*$
    mit $f_r(w) = r(f(\mathrm{r}^{-1}(w)))$ berechenbar
  - $r{:}\mathbb{N}{\to}X^*$ bijektive Repräsentation von Zahlen als Worte

Berechenbarkeit

- **Berechenbarkeit auf Worten**
  - $f{:}X^*{\to}Y^*$ berechenbar, falls f Turing-berechenbar
- **Berechenbarkeit auf Wort-Tupeln** $f{:}X^*{\times}X^*{\to}Y^*$
  - $f{:}X^*{\times}X^*{\to}Y^*$ berechenbar, falls $f'{:}(X{\cup}\{\#\})^*{\to}Y^*$
    mit $f'(v\#w) = f(v,w)$ berechenbar
- **Berechenbarkeit auf Zahlen**
  - $f{:}\mathbb{N}{\to}\mathbb{N}$ berechenbar, falls $f_r{:}X^*{\to}X^*$
    mit $f_r(w) = r(f(\mathrm{r}^{-1}(w)))$ berechenbar
  - $r{:}\mathbb{N}{\to}X^*$ bijektive Repräsentation von Zahlen als Worte
- **Berechenbarkeit auf Zahlentupeln und -listen**
  - $f{:}\mathbb{N}^k{\to}\mathbb{N}$ berechenbar, falls $f'{:}\mathbb{N}{\to}\mathbb{N}$
    mit $f'\langle x_1, x_2, .., x_k\rangle^k = f(x_1, x_2, .., x_k)$ berechenbar
  - $f{:}\mathbb{N}^*{\to}\mathbb{N}$ berechenbar, falls $f'{:}\mathbb{N}{\to}\mathbb{N}$
    mit $f'\langle x_1, x_2, .., x_k\rangle^* = f(x_1x_2...x_k)$ berechenbar
  - $\langle\rangle^k{:}\mathbb{N}^k{\to}\mathbb{N}$, $\langle\rangle^*{:}\mathbb{N}^*{\to}\mathbb{N}$ Standard-Tupelfunktionen

Aufzählbarkeit und Entscheidbarkeit – intuitiv

- **Entscheidbarkeit einer Menge**
  - Wir können eine Maschine konstruieren, die testet,
    ob ein bestimmtes Element zur Menge gehört oder nicht

Aufzählbarkeit und Entscheidbarkeit – intuitiv

- **Entscheidbarkeit einer Menge**
  - Wir können eine Maschine konstruieren, die testet,
    ob ein bestimmtes Element zur Menge gehört oder nicht
- **Semi-Entscheidbarkeit einer Menge**
  - Wir können eine Maschine konstruieren, die testet,
    ob ein bestimmtes Element zur Menge gehört,
    aber im Mißerfolgsfall eventuell niemals eine Antwort gibt

Aufzählbarkeit und Entscheidbarkeit – intuitiv

- **Entscheidbarkeit einer Menge**
  - Wir können eine Maschine konstruieren, die testet, ob ein bestimmtes Element zur Menge gehört oder nicht
- **Semi-Entscheidbarkeit einer Menge**
  - Wir können eine Maschine konstruieren, die testet, ob ein bestimmtes Element zur Menge gehört, aber im Mißerfolgsfall eventuell niemals eine Antwort gibt
- **Aufzählbarkeit einer Menge**
  - Wir können eine Maschine konstruieren, welche die Elemente der Menge schrittweise generiert, also z.B. bei Eingabe der Zahl n (oder einer Codierung) das n-te Element der Menge ausgibt

Aufzählbarkeit und Entscheidbarkeit – präzisiert

- $M \subseteq X^*$ **entscheidbar**
  - $\chi_M : X^* \to \{0,1\}^*$ berechenbar, wobei $\chi_M(w) = \begin{cases} 1 & \text{falls } w \in M, \\ 0 & \text{sonst} \end{cases}$

Aufzählbarkeit und Entscheidbarkeit – präzisiert

- $M \subseteq X^*$ **entscheidbar**
  - $\chi_M : X^* \to \{0,1\}^*$ berechenbar, wobei $\chi_M(w) = \begin{cases} 1 & \text{falls } w \in M, \\ 0 & \text{sonst} \end{cases}$
- $M \subseteq X^*$ **semi-entscheidbar**
  - $\psi_M : X^* \to \{0,1\}^*$ berechenbar, wobei $\psi_M(w) = \begin{cases} 1 & \text{falls } w \in M, \\ \perp & \text{sonst} \end{cases}$

Aufzählbarkeit und Entscheidbarkeit – präzisiert

- $M \subseteq X^*$ **entscheidbar**
  - $\chi_M : X^* \to \{0,1\}^*$ berechenbar, wobei $\chi_M(w) = \begin{cases} 1 & \text{falls } w \in M, \\ 0 & \text{sonst} \end{cases}$
- $M \subseteq X^*$ **semi-entscheidbar**
  - $\psi_M : X^* \to \{0,1\}^*$ berechenbar, wobei $\psi_M(w) = \begin{cases} 1 & \text{falls } w \in M, \\ \perp & \text{sonst} \end{cases}$
- $M \subseteq X^*$ **(rekursiv) aufzählbar**
  - $M = \emptyset$ oder es gibt eine totale, berechenbare Funktion $f : \{1\}^* \to X^*$

    mit $M = \text{range}(f) = \{v \in X^* \mid \exists w \in \{1\}^*\ f(w) = v\}$

Aufzählbarkeit und Entscheidbarkeit – präzisiert

- $M \subseteq X^*$ **entscheidbar**
  - $\chi_M : X^* \to \{0,1\}^*$ berechenbar, wobei $\chi_M(w) = \begin{cases} 1 & \text{falls } w \in M, \\ 0 & \text{sonst} \end{cases}$
- $M \subseteq X^*$ **semi-entscheidbar**
  - $\psi_M : X^* \to \{0,1\}^*$ berechenbar, wobei $\psi_M(w) = \begin{cases} 1 & \text{falls } w \in M, \\ \perp & \text{sonst} \end{cases}$
- $M \subseteq X^*$ **(rekursiv) aufzählbar**
  - $M = \emptyset$ oder es gibt eine totale, berechenbare Funktion $f : \{1\}^* \to X^*$ mit $M = \text{range}(f) = \{v \in X^* \mid \exists w \in \{1\}^*\ f(w) = v\}$
- **Berechenbarkeit von Mengen $M \subseteq \mathbb{N}$ analog**

Aufzählbarkeit – Anmerkungen

- **Jede endliche Menge ist aufzählbar**
  - $M = \{x_0, ..., x_n\}$ ist Wertebereich von f mit $f(n) = \begin{cases} x_i & \text{falls } i \leq n, \\ x_0 & \text{sonst} \end{cases}$
  - f ist primitiv rekursiv, also berechenbar

Aufzählbarkeit – Anmerkungen

- **Jede endliche Menge ist aufzählbar**
  - $M = \{x_0, ..., x_n\}$ ist Wertebereich von f mit $f(n) = \begin{cases} x_i & \text{falls } i \leq n, \\ x_0 & \text{sonst} \end{cases}$
  - f ist primitiv rekursiv, also berechenbar
- **Aufzählungen sind nicht notwendigerweise injektiv**
  - Die Funktion g mit $g(n) = \begin{cases} n+1 & \text{falls } n \text{ gerade}, \\ n & \text{sonst} \end{cases}$

    zählt alle ungeraden Zahlen genau zweimal auf

Aufzählbarkeit – Anmerkungen

- **Jede endliche Menge ist aufzählbar**
  - $M = \{x_0, ..., x_n\}$ ist Wertebereich von f mit $f(n) = \begin{cases} x_i & \text{falls } i \leq n, \\ x_0 & \text{sonst} \end{cases}$
  - f ist primitiv rekursiv, also berechenbar
- **Aufzählungen sind nicht notwendigerweise injektiv**
  - Die Funktion g mit $g(n) = \begin{cases} n+1 & \text{falls } n \text{ gerade}, \\ n & \text{sonst} \end{cases}$

    zählt alle ungeraden Zahlen genau zweimal auf
- **Jede Menge hat verschiedene Aufzählungen**
  - Die Funktion h mit $h(n) = 2n+1$

    zählt alle ungeraden Zahlen genau einmal auf

Aufzählbarkeit – Anmerkungen

- **Jede endliche Menge ist aufzählbar**
  - $M = \{x_0, ..., x_n\}$ ist Wertebereich von f mit $f(n) = \begin{cases} x_i & \text{falls } i \leq n, \\ x_0 & \text{sonst} \end{cases}$
  - f ist primitiv rekursiv, also berechenbar
- **Aufzählungen sind nicht notwendigerweise injektiv**
  - Die Funktion g mit $g(n) = \begin{cases} n+1 & \text{falls } n \text{ gerade}, \\ n & \text{sonst} \end{cases}$

    zählt alle ungeraden Zahlen genau zweimal auf
- **Jede Menge hat verschiedene Aufzählungen**
  - Die Funktion h mit $h(n) = 2n+1$

    zählt alle ungeraden Zahlen genau einmal auf
- **Es gibt viele äquivalente Charakterisierungen**
  - Aufzählbar — semi-entscheidbar
  - Werte- oder Haltebereich einer (evtl. partiellen) berechenbaren Funktion

Charakterisierungen von Aufzählbarkeit

Für $M \subseteq \mathbb{N}$ sind folgende Aussagen äquivalent

Charakterisierungen von Aufzählbarkeit

Für $M \subseteq \mathbb{N}$ sind folgende Aussagen äquivalent

1. M ist aufzählbar

Charakterisierungen von Aufzählbarkeit

Für $M \subseteq \mathbb{N}$ sind folgende Aussagen äquivalent

1. **M ist aufzählbar**
2. **$M = \text{range}(f)$ für ein berechenbares $f{:}\mathbb{N} \to \mathbb{N}$**
   – $\text{range}(f) = \{f(i) \mid i \in \mathbb{N}\}$, f nicht notwendigerweise total

Charakterisierungen von Aufzählbarkeit

Für $M \subseteq \mathbb{N}$ sind folgende Aussagen äquivalent

1. **M ist aufzählbar**
2. **$M = \text{range}(f)$ für ein berechenbares $f{:}\mathbb{N} \to \mathbb{N}$**
   – $\text{range}(f) = \{f(i) \mid i \in \mathbb{N}\}$, f nicht notwendigerweise total
3. **M ist semi-entscheidbar**

Charakterisierungen von Aufzählbarkeit

Für $M \subseteq \mathbb{N}$ sind folgende Aussagen äquivalent

1. **M ist aufzählbar**
2. **$M = \text{range}(f)$ für ein berechenbares $f{:}\mathbb{N}\to\mathbb{N}$**
   – $\text{range}(f) = \{f(i) \mid i \in \mathbb{N}\}$, f nicht notwendigerweise total
3. **M ist semi-entscheidbar**
4. **$M = \text{domain}(f)$ für ein berechenbares $f{:}\mathbb{N}\to\mathbb{N}$**
   – $\text{domain}(f) = \{i \in \mathbb{N} \mid f(i) \neq \bot\}$

Charakterisierungen von Aufzählbarkeit

Für $M \subseteq \mathbb{N}$ sind folgende Aussagen äquivalent

1. **M ist aufzählbar**
2. **$M = \text{range}(f)$ für ein berechenbares $f{:}\mathbb{N}\to\mathbb{N}$**
   – $\text{range}(f) = \{f(i) \mid i \in \mathbb{N}\}$, f nicht notwendigerweise total
3. **M ist semi-entscheidbar**
4. **$M = \text{domain}(f)$ für ein berechenbares $f{:}\mathbb{N}\to\mathbb{N}$**
   – $\text{domain}(f) = \{i \in \mathbb{N} \mid f(i) \neq \bot\}$

Aussage gilt analog für $M \subseteq X^*$, $M \subseteq \mathbb{N}^k$, ...

Beweis der Äquivalenz durch Ringschluss

- **M aufzählbar $\Rightarrow$ M=range(f) für ein berechenbares f**

Beweis der Äquivalenz durch Ringschluss

- **M aufzählbar $\Rightarrow$ M=range(f) für ein berechenbares f**
  - Falls $M=\emptyset$ dann ist $M = \text{range}(f_\perp)$, wobei $f_\perp(i) = \perp$ für alle $i \in \mathbb{N}$ ✓

Beweis der Äquivalenz durch Ringschluss

- **M aufzählbar $\Rightarrow$ M=range(f) für ein berechenbares f**
  - Falls $M=\emptyset$ dann ist $M = \text{range}(f_\perp)$, wobei $f_\perp(i) = \perp$ für alle $i \in \mathbb{N}$ ✓
  - Andernfalls $M = \text{range}(f)$ für ein berechenbares, totales f ✓

Beweis der Äquivalenz durch Ringschluss

- **M aufzählbar $\Rightarrow$ M=range(f) für ein berechenbares f**
  - Falls $M=\emptyset$ dann ist $M = \text{range}(f_\perp)$, wobei $f_\perp(i) = \perp$ für alle $i \in \mathbb{N}$ ✓
  - Andernfalls $M = \text{range}(f)$ für ein berechenbares, totales f ✓
- **M=range(f) für ein berechenbares f $\Rightarrow$ M semi-entscheidbar**

Beweis der Äquivalenz durch Ringschluss

- **M aufzählbar $\Rightarrow$ M=range(f) für ein berechenbares f**
  - Falls $M=\emptyset$ dann ist $M = \text{range}(f_\perp)$, wobei $f_\perp(i) = \perp$ für alle $i \in \mathbb{N}$ ✓
  - Andernfalls $M = \text{range}(f)$ für ein berechenbares, totales f ✓
- **M=range(f) für ein berechenbares f $\Rightarrow$ M semi-entscheidbar**
  - Es sei M=range(f) für ein berechenbares f

Beweis der Äquivalenz durch Ringschluss

- **M aufzählbar $\Rightarrow$ M=range(f) für ein berechenbares f**
  - Falls $M=\emptyset$ dann ist $M = \text{range}(f_\perp)$, wobei $f_\perp(i) = \perp$ für alle $i \in \mathbb{N}$ ✓
  - Andernfalls $M = \text{range}(f)$ für ein berechenbares, totales f ✓
- **M=range(f) für ein berechenbares f $\Rightarrow$ M semi-entscheidbar**
  - Es sei M=range(f) für ein berechenbares f
  - Es ist $\psi_M(w) = \begin{cases} 1 & \exists x, i.\ f(x) \text{ hält nach } i \text{ Schritten mit } f(x) = w, \\ \perp & \text{sonst} \end{cases}$
  - Damit ist ψ_M berechenbar und M semi-entscheidbar ✓

Beweis der Äquivalenz durch Ringschluss

- **M aufzählbar $\Rightarrow$ M=range(f) für ein berechenbares f**
  - Falls $M=\emptyset$ dann ist $M = \text{range}(f_\perp)$, wobei $f_\perp(i) = \perp$ für alle $i \in \mathbb{N}$ ✓
  - Andernfalls $M = \text{range}(f)$ für ein berechenbares, totales f ✓
- **M=range(f) für ein berechenbares f $\Rightarrow$ M semi-entscheidbar**
  - Es sei M=range(f) für ein berechenbares f
  - Es ist $\psi_M(w) = \begin{cases} 1 & \exists x, i.\ f(x) \text{ hält nach } i \text{ Schritten mit } f(x) = w, \\ \perp & \text{sonst} \end{cases}$
  - Damit ist ψ_M berechenbar und M semi-entscheidbar ✓
- **M semi-entscheidbar $\Rightarrow$ M=domain(f) für ein berechenbares f**

Beweis der Äquivalenz durch Ringschluss

- **M aufzählbar $\Rightarrow$ M=range(f) für ein berechenbares f**
  - Falls $M=\emptyset$ dann ist $M = \text{range}(f_\perp)$, wobei $f_\perp(i) = \perp$ für alle $i \in \mathbb{N}$ ✓
  - Andernfalls $M = \text{range}(f)$ für ein berechenbares, totales f ✓
- **M=range(f) für ein berechenbares f $\Rightarrow$ M semi-entscheidbar**
  - Es sei M=range(f) für ein berechenbares f
  - Es ist $\psi_M(w) = \begin{cases} 1 & \exists x, i.\ f(x) \text{ hält nach } i \text{ Schritten mit } f(x) = w, \\ \perp & \text{sonst} \end{cases}$
  - Damit ist ψ_M berechenbar und M semi-entscheidbar ✓
- **M semi-entscheidbar $\Rightarrow$ M=domain(f) für ein berechenbares f**
  - Es sei M semi-entscheidbar.

Beweis der Äquivalenz durch Ringschluss

- **M aufzählbar $\Rightarrow$ M=range(f) für ein berechenbares f**
  - Falls $M=\emptyset$ dann ist $M = \text{range}(f_\perp)$, wobei $f_\perp(i) = \perp$ für alle $i \in \mathbb{N}$ ✓
  - Andernfalls $M = \text{range}(f)$ für ein berechenbares, totales f ✓
- **M=range(f) für ein berechenbares f $\Rightarrow$ M semi-entscheidbar**
  - Es sei M=range(f) für ein berechenbares f
  - Es ist $\psi_M(w) = \begin{cases} 1 & \exists x, i.\ f(x) \text{ hält nach } i \text{ Schritten mit } f(x) = w, \\ \perp & \text{sonst} \end{cases}$
  - Damit ist ψ_M berechenbar und M semi-entscheidbar ✓
- **M semi-entscheidbar $\Rightarrow$ M=domain(f) für ein berechenbares f**
  - Es sei M semi-entscheidbar.
  - Dann ist ψ_M berechenbar und $M=\{i \in \mathbb{N} \mid \psi_M(i) = 1\} = \text{domain}(\psi_M)$ ✓

Beweis der Äquivalenz durch Ringschluss (II)

- **M=domain(f) für ein berechenbares f $\Rightarrow$ M aufzählbar**

Beweis der Äquivalenz durch Ringschluss (II)

- **M=domain(f) für ein berechenbares f $\Rightarrow$ M aufzählbar**
  - Es sei $M = \text{domain}(f) = \{i \in \mathbb{N} \mid f(i) \neq \perp\}$ für ein berechenbares f

Beweis der Äquivalenz durch Ringschluss (II)

- **M=domain(f) für ein berechenbares f $\Rightarrow$ M aufzählbar**
  - Es sei $M = \text{domain}(f) = \{i \in \mathbb{N} \mid f(i) \neq \bot\}$ für ein berechenbares f
  - Falls $M = \emptyset$, dann ist M per Definition aufzählbar ✓

Beweis der Äquivalenz durch Ringschluss (II)

- **M=domain(f) für ein berechenbares f $\Rightarrow$ M aufzählbar**
  - Es sei $M = \text{domain}(f) = \{i \in \mathbb{N} \mid f(i) \neq \perp\}$ für ein berechenbares f
  - Falls $M = \emptyset$, dann ist M per Definition aufzählbar ✓
  - Andernfalls gibt es ein $n_0 \in \mathbb{N}$ mit $f(n_0) \neq \perp$ und eine TM τ mit $f = h_\tau$

Beweis der Äquivalenz durch Ringschluss (II)

- **M=domain(f) für ein berechenbares f $\Rightarrow$ M aufzählbar**
  - Es sei $M = \text{domain}(f) = \{i \in \mathbb{N} \mid f(i) \neq \perp\}$ für ein berechenbares f
  - Falls $M = \emptyset$, dann ist M per Definition aufzählbar ✓
  - Andernfalls gibt es ein $n_0 \in \mathbb{N}$ mit $f(n_0) \neq \perp$ und eine TM τ mit $f = h_\tau$
  - Wir konstruieren eine Turingmaschine τ' mit $M = \text{range}(h_{\tau'})$

    Bei Eingabe einer (Repräsentation der) Zahl n arbeite τ' wie folgt

Beweis der Äquivalenz durch Ringschluss (II)

- **M=domain(f) für ein berechenbares f $\Rightarrow$ M aufzählbar**
  - Es sei $M = \text{domain}(f) = \{i \in \mathbb{N} \mid f(i) \neq \perp\}$ für ein berechenbares f
  - Falls $M = \emptyset$, dann ist M per Definition aufzählbar ✓
  - Andernfalls gibt es ein $n_0 \in \mathbb{N}$ mit $f(n_0) \neq \perp$ und eine TM τ mit $f = h_\tau$
  - Wir konstruieren eine Turingmaschine τ' mit $M = \text{range}(h_{\tau'})$

    Bei Eingabe einer (Repräsentation der) Zahl n arbeite τ' wie folgt

    1. Berechne i, j so daß $n = \langle i, j \rangle$

Beweis der Äquivalenz durch Ringschluss (II)

- **M=domain(f) für ein berechenbares f $\Rightarrow$ M aufzählbar**
  - Es sei $M = \text{domain}(f) = \{i \in \mathbb{N} \mid f(i) \neq \perp\}$ für ein berechenbares f
  - Falls $M = \emptyset$, dann ist M per Definition aufzählbar ✓
  - Andernfalls gibt es ein $n_0 \in \mathbb{N}$ mit $f(n_0) \neq \perp$ und eine TM τ mit $f = h_\tau$
  - Wir konstruieren eine Turingmaschine τ' mit $M = \text{range}(h_{\tau'})$

    Bei Eingabe einer (Repräsentation der) Zahl n arbeite τ' wie folgt

    1. Berechne i, j so daß $n = \langle i, j \rangle$
    2. Simuliere die Berechnung von τ bei Eingabe i für (maximal) j Schritte

       (Erzeuge Anfangskonfiguration, simuliere $\hat{\delta}_\tau$ auf separatem Band)

Beweis der Äquivalenz durch Ringschluss (II)

- **M=domain(f) für ein berechenbares f $\Rightarrow$ M aufzählbar**
  - Es sei $M = \text{domain}(f) = \{i \in \mathbb{N} \mid f(i) \neq \perp\}$ für ein berechenbares f
  - Falls $M = \emptyset$, dann ist M per Definition aufzählbar ✓
  - Andernfalls gibt es ein $n_0 \in \mathbb{N}$ mit $f(n_0) \neq \perp$ und eine TM τ mit $f = h_\tau$
  - Wir konstruieren eine Turingmaschine τ' mit $M = \text{range}(h_{\tau'})$

    Bei Eingabe einer (Repräsentation der) Zahl n arbeite τ' wie folgt

    1. Berechne i, j so daß $n = \langle i, j \rangle$
    2. Simuliere die Berechnung von τ bei Eingabe i für (maximal) j Schritte
       (Erzeuge Anfangskonfiguration, simuliere $\hat{\delta}_\tau$ auf separatem Band)
    3. Falls τ in j Schritten anhält, gebe i aus. Andernfalls gebe n_0 aus

Beweis der Äquivalenz durch Ringschluss (II)

- **M=domain(f) für ein berechenbares f $\Rightarrow$ M aufzählbar**
  - Es sei $M = \text{domain}(f) = \{i \in \mathbb{N} \mid f(i) \neq \perp\}$ für ein berechenbares f
  - Falls $M = \emptyset$, dann ist M per Definition aufzählbar ✓
  - Andernfalls gibt es ein $n_0 \in \mathbb{N}$ mit $f(n_0) \neq \perp$ und eine TM τ mit $f = h_\tau$
  - Wir konstruieren eine Turingmaschine τ' mit $M = \text{range}(h_{\tau'})$

    Bei Eingabe einer (Repräsentation der) Zahl n arbeite τ' wie folgt

    1. Berechne i, j so daß $n = \langle i, j \rangle$
    2. Simuliere die Berechnung von τ bei Eingabe i für (maximal) j Schritte
       (Erzeuge Anfangskonfiguration, simuliere $\hat{\delta}_\tau$ auf separatem Band)
    3. Falls τ in j Schritten anhält, gebe i aus. Andernfalls gebe n_0 aus
  - τ' hält auf jeder Eingabe, also ist $h_{\tau'}$ total

Beweis der Äquivalenz durch Ringschluss (II)

- **M=domain(f) für ein berechenbares f $\Rightarrow$ M aufzählbar**
  - Es sei $M = \text{domain}(f) = \{i \in \mathbb{N} \mid f(i) \neq \perp\}$ für ein berechenbares f
  - Falls $M = \emptyset$, dann ist M per Definition aufzählbar ✓
  - Andernfalls gibt es ein $n_0 \in \mathbb{N}$ mit $f(n_0) \neq \perp$ und eine TM τ mit $f = h_\tau$
  - Wir konstruieren eine Turingmaschine τ' mit $M = \text{range}(h_{\tau'})$

    Bei Eingabe einer (Repräsentation der) Zahl n arbeite τ' wie folgt

    1. Berechne i, j so daß $n = \langle i, j \rangle$
    2. Simuliere die Berechnung von τ bei Eingabe i für (maximal) j Schritte
       (Erzeuge Anfangskonfiguration, simuliere $\hat{\delta}_\tau$ auf separatem Band)
    3. Falls τ in j Schritten anhält, gebe i aus. Andernfalls gebe n_0 aus
  - τ' hält auf jeder Eingabe, also ist $h_{\tau'}$ total
  - $M \subseteq \text{range}(h_{\tau'})$: Sei $i \in M$. Dann gibt es ein j so daß τ nach j Schritten hält.

    Damit $h_{\tau'}(\langle i, j \rangle) = i$, also $i \in \text{range}(h_{\tau'})$

Beweis der Äquivalenz durch Ringschluss (II)

- **M=domain(f) für ein berechenbares f $\Rightarrow$ M aufzählbar**
  - Es sei $M = \text{domain}(f) = \{i \in \mathbb{N} \mid f(i) \neq \perp\}$ für ein berechenbares f
  - Falls $M = \emptyset$, dann ist M per Definition aufzählbar ✓
  - Andernfalls gibt es ein $n_0 \in \mathbb{N}$ mit $f(n_0) \neq \perp$ und eine TM τ mit $f = h_\tau$
  - Wir konstruieren eine Turingmaschine τ' mit $M = \text{range}(h_{\tau'})$

    Bei Eingabe einer (Repräsentation der) Zahl n arbeite τ' wie folgt

    1. Berechne i, j so daß $n = \langle i, j \rangle$
    2. Simuliere die Berechnung von τ bei Eingabe i für (maximal) j Schritte
       (Erzeuge Anfangskonfiguration, simuliere $\hat{\delta}_\tau$ auf separatem Band)
    3. Falls τ in j Schritten anhält, gebe i aus. Andernfalls gebe n_0 aus
  - τ' hält auf jeder Eingabe, also ist $h_{\tau'}$ total
  - $M \subseteq \text{range}(h_{\tau'})$: Sei $i \in M$. Dann gibt es ein j so daß τ nach j Schritten hält.
    Damit $h_{\tau'}(\langle i, j \rangle) = i$, also $i \in \text{range}(h_{\tau'})$
  - $M \supseteq \text{range}(h_{\tau'})$: Sei $i \in \text{range}(h_{\tau'})$. Falls $i = n_0$ dann $i \in M$ nach Voraussetzung.
    Andernfalls gibt es ein j so daß τ nach j Schritten hält.
    Also $i \in \text{domain}(h_\tau) = M$. ✓

Beispiele entscheidbarer / aufzählbarer Mengen

Sei f**:**$\mathbb{N}\to\mathbb{N}$ **total berechenbar,** g**:**$\mathbb{N}\to\mathbb{N}$ **berechenbar**

Beispiele entscheidbarer / aufzählbarer Mengen

Sei f**:**$\mathbb{N}\to\mathbb{N}$ **total berechenbar,** g**:**$\mathbb{N}\to\mathbb{N}$ **berechenbar**

- **graph(f) $= \{(i,j) \mid f(i) = j\}$ ist entscheidbar**
  - Bei Eingabe (i,j) berechne $f(i)$ (hält immer) und vergleiche mit j

Beispiele entscheidbarer / aufzählbarer Mengen

Sei f**:**$\mathbb{N}\to\mathbb{N}$ **total berechenbar,** g**:**$\mathbb{N}\to\mathbb{N}$ **berechenbar**

- $\mathbf{graph}(f) = \{(i,j) \mid f(i) = j\}$ **ist entscheidbar**
  - Bei Eingabe (i,j) berechne $f(i)$ (hält immer) und vergleiche mit j
- $\mathbf{range}(f) = \{j \mid \exists i \in \mathbb{N} f(i) = j\}$ **ist aufzählbar**
  - Charakterisierung #2

Beispiele entscheidbarer / aufzählbarer Mengen

Sei f:$\mathbb{N}\to\mathbb{N}$ total berechenbar, g:$\mathbb{N}\to\mathbb{N}$ berechenbar

- **graph(f) $= \{(i,j) \mid f(i) = j\}$ ist entscheidbar**
  - Bei Eingabe (i,j) berechne $f(i)$ (hält immer) und vergleiche mit j
- **range(f) $= \{j \mid \exists i \in \mathbb{N} f(i) = j\}$ ist aufzählbar**
  - Charakterisierung #2
- **range(g) $= \{j \mid \exists i \in \mathbb{N} g(i) = j\}$ ist aufzählbar**
  - Charakterisierung #2

Beispiele entscheidbarer / aufzählbarer Mengen

Sei f:$\mathbb{N}\to\mathbb{N}$ total berechenbar, g:$\mathbb{N}\to\mathbb{N}$ berechenbar

- **$\mathbf{graph}(f) = \{(i,j) \mid f(i) = j\}$ ist entscheidbar**
  - Bei Eingabe (i,j) berechne $f(i)$ (hält immer) und vergleiche mit j
- **$\mathbf{range}(f) = \{j \mid \exists i \in \mathbb{N} f(i) = j\}$ ist aufzählbar**
  - Charakterisierung #2
- **$\mathbf{range}(g) = \{j \mid \exists i \in \mathbb{N} g(i) = j\}$ ist aufzählbar**
  - Charakterisierung #2
- **$\mathbf{graph}(g) = \{(i,j) \mid g(i) = j\}$ ist aufzählbar**
  - Bei Eingabe (i,j) berechne $g(i)$ (hält nicht immer) und vergleiche

Beispiele entscheidbarer / aufzählbarer Mengen

Sei f:$\mathbb{N}\to\mathbb{N}$ total berechenbar, g:$\mathbb{N}\to\mathbb{N}$ berechenbar

- **$\mathbf{graph}(f) = \{(i,j) \mid f(i) = j\}$ ist entscheidbar**
  - Bei Eingabe (i,j) berechne $f(i)$ (hält immer) und vergleiche mit j
- **$\mathbf{range}(f) = \{j \mid \exists i \in \mathbb{N} f(i) = j\}$ ist aufzählbar**
  - Charakterisierung #2
- **$\mathbf{range}(g) = \{j \mid \exists i \in \mathbb{N} g(i) = j\}$ ist aufzählbar**
  - Charakterisierung #2
- **$\mathbf{graph}(g) = \{(i,j) \mid g(i) = j\}$ ist aufzählbar**
  - Bei Eingabe (i,j) berechne $g(i)$ (hält nicht immer) und vergleiche
- **$\mathbf{domain}(g) = \{i \mid g(i)$ hält$\}$ ist aufzählbar**
  - Charakterisierung #4

Aufzählbarkeit vs. Entscheidbarkeit

$$M \subseteq \mathbb{N} \text{ entscheidbar } \Leftrightarrow M \text{ und } \overline{M} \text{ aufzählbar}$$

Aufzählbarkeit vs. Entscheidbarkeit

$M \subseteq \mathbb{N}$ entscheidbar $\Leftrightarrow$ M und $\overline{M}$ aufzählbar

$\Rightarrow$ Es sei M entscheidbar. Dann ist $\chi_M : \mathbb{N} \to \mathbb{N}$ berechenbar.

Aufzählbarkeit vs. Entscheidbarkeit

$M \subseteq \mathbb{N}$ entscheidbar $\Leftrightarrow$ M und $\overline{M}$ aufzählbar

$\Rightarrow$ Es sei M entscheidbar. Dann ist $\chi_M : \mathbb{N} \to \mathbb{N}$ berechenbar.

Es ist $\psi_M(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=1, \\ \perp & \text{sonst} \end{cases}$ und $\psi_{\overline{M}}(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=0, \\ \perp & \text{sonst} \end{cases}$

Aufzählbarkeit vs. Entscheidbarkeit

$M \subseteq \mathbb{N}$ entscheidbar $\Leftrightarrow$ M und $\overline{M}$ aufzählbar

$\Rightarrow$ Es sei M entscheidbar. Dann ist $\chi_M : \mathbb{N} \to \mathbb{N}$ berechenbar.

Es ist $\psi_M(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=1, \\ \perp & \text{sonst} \end{cases}$ und $\psi_{\overline{M}}(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=0, \\ \perp & \text{sonst} \end{cases}$

Damit sind ψ_M und $\psi_{\overline{M}}$ berechenbar, also M und $\overline{M}$ aufzählbar ✓

Aufzählbarkeit vs. Entscheidbarkeit

$M \subseteq \mathbb{N}$ entscheidbar $\Leftrightarrow$ M und $\overline{M}$ aufzählbar

$\Rightarrow$ Es sei M entscheidbar. Dann ist $\chi_M : \mathbb{N} \to \mathbb{N}$ berechenbar.

Es ist $\psi_M(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=1, \\ \perp & \text{sonst} \end{cases}$ und $\psi_{\overline{M}}(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=0, \\ \perp & \text{sonst} \end{cases}$

Damit sind ψ_M und $\psi_{\overline{M}}$ berechenbar, also M und $\overline{M}$ aufzählbar ✓

$\Leftarrow$ Seien M und $\overline{M}$ aufzählbar.

Aufzählbarkeit vs. Entscheidbarkeit

$M \subseteq \mathbb{N}$ entscheidbar $\Leftrightarrow$ M und $\overline{M}$ aufzählbar

$\Rightarrow$ Es sei M entscheidbar. Dann ist $\chi_M : \mathbb{N} \to \mathbb{N}$ berechenbar.

Es ist $\psi_M(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=1, \\ \perp & \text{sonst} \end{cases}$ und $\psi_{\overline{M}}(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=0, \\ \perp & \text{sonst} \end{cases}$

Damit sind ψ_M und $\psi_{\overline{M}}$ berechenbar, also M und $\overline{M}$ aufzählbar ✓

$\Leftarrow$ Seien M und $\overline{M}$ aufzählbar.

Falls $M=\emptyset$ oder $\overline{M}=\emptyset$, so ist M trivialerweise entscheidbar ✓

Aufzählbarkeit vs. Entscheidbarkeit

$M \subseteq \mathbb{N}$ entscheidbar $\Leftrightarrow$ M und $\overline{M}$ aufzählbar

$\Rightarrow$ Es sei M entscheidbar. Dann ist $\chi_M : \mathbb{N} \to \mathbb{N}$ berechenbar.

Es ist $\psi_M(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=1, \\ \perp & \text{sonst} \end{cases}$ und $\psi_{\overline{M}}(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=0, \\ \perp & \text{sonst} \end{cases}$

Damit sind ψ_M und $\psi_{\overline{M}}$ berechenbar, also M und $\overline{M}$ aufzählbar ✓

$\Leftarrow$ Seien M und $\overline{M}$ aufzählbar.

Falls $M=\emptyset$ oder $\overline{M}=\emptyset$, so ist M trivialerweise entscheidbar ✓

Andernfalls $M=\text{range}(f)$ und $\overline{M}=\text{range}(g)$ wobei f, g total berechenbar.

Aufzählbarkeit vs. Entscheidbarkeit

$M \subseteq \mathbb{N}$ entscheidbar $\Leftrightarrow$ M und $\overline{M}$ aufzählbar

$\Rightarrow$ Es sei M entscheidbar. Dann ist $\chi_M{:}\mathbb{N}\to\mathbb{N}$ berechenbar.

Es ist $\psi_M(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=1, \\ \perp & \text{sonst} \end{cases}$ und $\psi_{\overline{M}}(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=0, \\ \perp & \text{sonst} \end{cases}$

Damit sind ψ_M und $\psi_{\overline{M}}$ berechenbar, also M und $\overline{M}$ aufzählbar ✓

$\Leftarrow$ Seien M und $\overline{M}$ aufzählbar.

Falls $M{=}\emptyset$ oder $\overline{M}{=}\emptyset$, so ist M trivialerweise entscheidbar ✓

Andernfalls $M{=}\text{range}(f)$ und $\overline{M}{=}\text{range}(g)$ wobei f, g total berechenbar.

Wir konstruieren eine Turingmaschine τ mit $\chi_M{=}h_\tau$

– Bei Eingabe von n berechne $j = \min\{i \mid f(i){=}n \vee g(i){=}n\}$

– Falls $f(j){=}n$, gebe 1 aus, ansonsten 0

Aufzählbarkeit vs. Entscheidbarkeit

$M \subseteq \mathbb{N}$ entscheidbar $\Leftrightarrow$ M und $\overline{M}$ aufzählbar

$\Rightarrow$ Es sei M entscheidbar. Dann ist $\chi_M : \mathbb{N} \to \mathbb{N}$ berechenbar.

Es ist $\psi_M(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=1, \\ \perp & \text{sonst} \end{cases}$ und $\psi_{\overline{M}}(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=0, \\ \perp & \text{sonst} \end{cases}$

Damit sind ψ_M und $\psi_{\overline{M}}$ berechenbar, also M und $\overline{M}$ aufzählbar ✓

$\Leftarrow$ Seien M und $\overline{M}$ aufzählbar.

Falls $M=\emptyset$ oder $\overline{M}=\emptyset$, so ist M trivialerweise entscheidbar ✓

Andernfalls $M=\text{range}(f)$ und $\overline{M}=\text{range}(g)$ wobei f, g total berechenbar.

Wir konstruieren eine Turingmaschine τ mit $\chi_M = h_\tau$

– Bei Eingabe von n berechne $j = \min\{i \mid f(i)=n \ \vee \ g(i)=n\}$

– Falls $f(j)=n$, gebe 1 aus, ansonsten 0

τ hält für jedes n, da $n \in \text{range}(f)$ oder $n \in \text{range}(g)$ $(n \in M \ / \ n \notin M)$

Aufzählbarkeit vs. Entscheidbarkeit

$M \subseteq \mathbb{N}$ entscheidbar $\Leftrightarrow$ M und $\overline{M}$ aufzählbar

$\Rightarrow$ Es sei M entscheidbar. Dann ist $\chi_M : \mathbb{N} \to \mathbb{N}$ berechenbar.

Es ist $\psi_M(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=1, \\ \perp & \text{sonst} \end{cases}$ und $\psi_{\overline{M}}(n) = \begin{cases} 1 & \text{falls } \chi_M(n)=0, \\ \perp & \text{sonst} \end{cases}$

Damit sind ψ_M und $\psi_{\overline{M}}$ berechenbar, also M und $\overline{M}$ aufzählbar ✓

$\Leftarrow$ Seien M und $\overline{M}$ aufzählbar.

Falls $M=\emptyset$ oder $\overline{M}=\emptyset$, so ist M trivialerweise entscheidbar ✓

Andernfalls $M=\text{range}(f)$ und $\overline{M}=\text{range}(g)$ wobei f, g total berechenbar.

Wir konstruieren eine Turingmaschine τ mit $\chi_M = h_\tau$

– Bei Eingabe von n berechne $j = \min\{i \mid f(i)=n \ \vee \ g(i)=n\}$

– Falls $f(j)=n$, gebe 1 aus, ansonsten 0

τ hält für jedes n, da $n \in \text{range}(f)$ oder $n \in \text{range}(g)$ $(n \in M \ / \ n \notin M)$

Für $n \in M$ gilt $f(j)=n$ für ein j, also $h_\tau(n)=1$, ansonsten $h_\tau(n)=0$ ✓

Aufzählbarkeit vs. Entscheidbarkeit

- **Jede entscheidbare Menge ist aufzählbar**

Aufzählbarkeit vs. Entscheidbarkeit

- **Jede entscheidbare Menge ist aufzählbar**
  - Die Umkehrung gilt nicht (Beispiel folgt später)

Aufzählbarkeit vs. Entscheidbarkeit

- **Jede entscheidbare Menge ist aufzählbar**
  - Die Umkehrung gilt nicht (Beispiel folgt später)
- **Ist M endlich, so ist M entscheidbar und aufzählbar**

Aufzählbarkeit vs. Entscheidbarkeit

- **Jede entscheidbare Menge ist aufzählbar**
  - Die Umkehrung gilt nicht (Beispiel folgt später)
- **Ist M endlich, so ist M entscheidbar und aufzählbar**
  - Für $M = \{x_1, ..., x_n\}$ ist $\chi_M(n) = \begin{cases} 1 & n{=}x_1 \vee \ldots \vee n{=}x_n \ , \\ 0 & \text{sonst} \end{cases}$
  - χ_M ist berechenbar, also ist M entscheidbar ✓

Aufzählbarkeit vs. Entscheidbarkeit

- **Jede entscheidbare Menge ist aufzählbar**
  - Die Umkehrung gilt nicht (Beispiel folgt später)
- **Ist M endlich, so ist M entscheidbar und aufzählbar**
  - Für $M = \{x_1, ..., x_n\}$ ist $\chi_M(n) = \begin{cases} 1 & n{=}x_1 \lor \ldots \lor n{=}x_n \,, \\ 0 & \text{sonst} \end{cases}$
  - χ_M ist berechenbar, also ist M entscheidbar ✓
- $\boldsymbol{M \subseteq \mathbb{N}}$ **aufzählbar** $\Leftrightarrow$ **es gibt ein entscheidbares** $\boldsymbol{M' \subseteq \mathbb{N} \times \mathbb{N}}$

  mit $\boldsymbol{M = \{y \in \mathbb{N} \mid \exists n \in \mathbb{N}\ (n, y) \in M'\}}$

Aufzählbarkeit vs. Entscheidbarkeit

- **Jede entscheidbare Menge ist aufzählbar**
  - Die Umkehrung gilt nicht (Beispiel folgt später)
- **Ist M endlich, so ist M entscheidbar und aufzählbar**
  - Für $M = \{x_1, ..., x_n\}$ ist $\chi_M(n) = \begin{cases} 1 & n{=}x_1 \lor \ldots \lor n{=}x_n \,, \\ 0 & \text{sonst} \end{cases}$
  - χ_M ist berechenbar, also ist M entscheidbar ✓
- $\boldsymbol{M \subseteq \mathbb{N}}$ **aufzählbar** $\Leftrightarrow$ **es gibt ein entscheidbares** $\boldsymbol{M' \subseteq \mathbb{N} \times \mathbb{N}}$

  mit $\boldsymbol{M = \{y \in \mathbb{N} \mid \exists n \in \mathbb{N}\ (n, y) \in M'\}}$

  $\Rightarrow$: Es sei $M \subseteq \mathbb{N}$ aufzählbar

Aufzählbarkeit vs. Entscheidbarkeit

- **Jede entscheidbare Menge ist aufzählbar**
  - Die Umkehrung gilt nicht (Beispiel folgt später)
- **Ist M endlich, so ist M entscheidbar und aufzählbar**
  - Für $M = \{x_1, ..., x_n\}$ ist $\chi_M(n) = \begin{cases} 1 & n{=}x_1 \lor \ldots \lor n{=}x_n \,, \\ 0 & \text{sonst} \end{cases}$
  - χ_M ist berechenbar, also ist M entscheidbar ✓
- $\boldsymbol{M \subseteq \mathbb{N}}$ **aufzählbar** $\Leftrightarrow$ **es gibt ein entscheidbares** $\boldsymbol{M' \subseteq \mathbb{N} \times \mathbb{N}}$ **mit** $\boldsymbol{M = \{y \in \mathbb{N} \mid \exists n \in \mathbb{N}\ (n, y) \in M'\}}$

  $\Rightarrow$: Es sei $M \subseteq \mathbb{N}$ aufzählbar
  - Falls $M = \emptyset$, so ist $M = \{y \in \mathbb{N} \mid \exists n \in \mathbb{N}\ (n, y) \in \emptyset\}$ ✓

Aufzählbarkeit vs. Entscheidbarkeit

- **Jede entscheidbare Menge ist aufzählbar**
  - Die Umkehrung gilt nicht (Beispiel folgt später)
- **Ist M endlich, so ist M entscheidbar und aufzählbar**
  - Für $M = \{x_1, ..., x_n\}$ ist $\chi_M(n) = \begin{cases} 1 & n{=}x_1 \lor \ldots \lor n{=}x_n \,, \\ 0 & \text{sonst} \end{cases}$
  - χ_M ist berechenbar, also ist M entscheidbar ✓
- $\boldsymbol{M \subseteq \mathbb{N}}$ **aufzählbar** $\Leftrightarrow$ **es gibt ein entscheidbares** $\boldsymbol{M' \subseteq \mathbb{N} \times \mathbb{N}}$ **mit** $\boldsymbol{M = \{y \in \mathbb{N} \mid \exists n \in \mathbb{N}\ (n, y) \in M'\}}$

  $\Rightarrow$: Es sei $M \subseteq \mathbb{N}$ aufzählbar
  - Falls $M{=}\emptyset$, so ist $M = \{y \in \mathbb{N} \mid \exists n \in \mathbb{N}\ (n, y) \in \emptyset\}$ ✓
  - Andernfalls ist $M{=}\text{range}(f)$ für ein berechenbares, totales f und $\text{range}(f) = \{y \in \mathbb{N} \mid \exists n \in \mathbb{N}\ (n, y) \in \text{graph}(f)\}$ ✓

Aufzählbarkeit vs. Entscheidbarkeit

- **Jede entscheidbare Menge ist aufzählbar**
  - Die Umkehrung gilt nicht (Beispiel folgt später)
- **Ist M endlich, so ist M entscheidbar und aufzählbar**
  - Für $M = \{x_1, ..., x_n\}$ ist $\chi_M(n) = \begin{cases} 1 & n{=}x_1 \lor \ldots \lor n{=}x_n \ , \\ 0 & \text{sonst} \end{cases}$
  - χ_M ist berechenbar, also ist M entscheidbar ✓
- $\boldsymbol{M \subseteq \mathbb{N}}$ **aufzählbar** $\Leftrightarrow$ **es gibt ein entscheidbares** $\boldsymbol{M' \subseteq \mathbb{N} \times \mathbb{N}}$ **mit** $\boldsymbol{M = \{y \in \mathbb{N} \mid \exists n \in \mathbb{N}\ (n, y) \in M'\}}$

  $\Rightarrow$: Es sei $M \subseteq \mathbb{N}$ aufzählbar
  - Falls $M{=}\emptyset$, so ist $M = \{y \in \mathbb{N} \mid \exists n \in \mathbb{N}\ (n, y) \in \emptyset\}$ ✓
  - Andernfalls ist $M{=}\text{range}(f)$ für ein berechenbares, totales f und $\text{range}(f) = \{y \in \mathbb{N} \mid \exists n \in \mathbb{N}\ (n, y) \in \text{graph}(f)\}$ ✓

  $\Leftarrow$: Es sei $M = \{y \in \mathbb{N} \mid \exists n \in \mathbb{N}\ (n, y) \in M'\}$ für ein entscheidbares M'

Aufzählbarkeit vs. Entscheidbarkeit

- **Jede entscheidbare Menge ist aufzählbar**
  - Die Umkehrung gilt nicht (Beispiel folgt später)
- **Ist M endlich, so ist M entscheidbar und aufzählbar**
  - Für $M = \{x_1, ..., x_n\}$ ist $\chi_M(n) = \begin{cases} 1 & n{=}x_1 \lor \ldots \lor n{=}x_n \,, \\ 0 & \text{sonst} \end{cases}$
  - χ_M ist berechenbar, also ist M entscheidbar ✓
- $\boldsymbol{M \subseteq \mathbb{N}}$ **aufzählbar** $\Leftrightarrow$ **es gibt ein entscheidbares** $\boldsymbol{M' \subseteq \mathbb{N} \times \mathbb{N}}$ **mit** $\boldsymbol{M = \{y \in \mathbb{N} \mid \exists n \in \mathbb{N}\ (n, y) \in M'\}}$

  $\Rightarrow$: Es sei $M \subseteq \mathbb{N}$ aufzählbar
  - Falls $M{=}\emptyset$, so ist $M = \{y \in \mathbb{N} \mid \exists n \in \mathbb{N}\ (n, y) \in \emptyset\}$ ✓
  - Andernfalls ist $M{=}\text{range}(f)$ für ein berechenbares, totales f und $\text{range}(f) = \{y \in \mathbb{N} \mid \exists n \in \mathbb{N}\ (n, y) \in \text{graph}(f)\}$ ✓

  $\Leftarrow$: Es sei $M = \{y \in \mathbb{N} \mid \exists n \in \mathbb{N}\ (n, y) \in M'\}$ für ein entscheidbares M'
  - Dann ist $\psi_M(y) = \text{sign}\ (\min\{n \in \mathbb{N} \mid (n, y) \in M'\}{+}1)$
    $= \text{sign}\ (\min\{n \in \mathbb{N} \mid \chi_{M'}(n, y){=}1\}){+}1$ ✓

Abschlusseigenschaften

Sei M, $M' \subseteq X^*$, f berechenbar und total, g berechenbar

Abschlusseigenschaften

Sei M, $M' \subseteq X^*$, f berechenbar und total, g berechenbar

- **Sind M, M' aufzählbar, dann auch**

  $M \cup M'$, $M \cap M'$, $M \circ M'$, M^*, $g(M)$, $g^{-1}(M)$

  Vereinigung, Durchschnitt, Konkatenation, Konkatenationsabschluß, Urbild

  Abschluß unter Komplement oder Differenz gilt nicht

Abschlusseigenschaften

Sei M, $M' \subseteq X^*$, f berechenbar und total, g berechenbar

- **Sind M, M' aufzählbar, dann auch**

  $M \cup M'$, $M \cap M'$, $M \circ M'$, M^*, $g(M)$, $g^{-1}(M)$

  Vereinigung, Durchschnitt, Konkatenation, Konkatenationsabschluß, Urbild

  Abschluß unter Komplement oder Differenz gilt nicht

- **Sind M, M' entscheidbar, dann auch**

  $M \cup M'$, $M \cap M'$, $M \setminus M'$, $\overline{M}$, $M \circ M'$, M^*, $f^{-1}(M)$

  Vereinigung, Durchschnitt, Differenz, Komplement, Konkatenation, -abschluß, Urbild

$\overline{M} = \{w \in X^* \mid w \notin M\}$ $\quad M \circ M' = \{wv \mid w \in M \wedge v \in M'\}$ $\quad M^* = \{w_1..w_n \mid w_i \in M\}$

$f^{-1}(M) = \{w \in X^* \mid f(w) \in M\}$ $\quad f(M) = \{f(w) \mid w \in M\}$

Abschlusseigenschaften

Sei M, $M' \subseteq X^*$, f berechenbar und total, g berechenbar

- **Sind M, M' aufzählbar, dann auch**

  $M \cup M'$, $M \cap M'$, $M \circ M'$, M^*, $g(M)$, $g^{-1}(M)$

  Vereinigung, Durchschnitt, Konkatenation, Konkatenationsabschluß, Urbild

  Abschluß unter Komplement oder Differenz gilt nicht

- **Sind M, M' entscheidbar, dann auch**

  $M \cup M'$, $M \cap M'$, $M \setminus M'$, $\overline{M}$, $M \circ M'$, M^*, $f^{-1}(M)$

  Vereinigung, Durchschnitt, Differenz, Komplement, Konkatenation, -abschluß, Urbild

Aussage gilt analog für Teilmengen von $\mathbb{N}$, $\mathbb{N}^k$, ...

$\overline{M} = \{w \in X^* \mid w \notin M\}$ $\quad M \circ M' = \{wv \mid w \in M \wedge v \in M'\}$ $\quad M^* = \{w_1..w_n \mid w_i \in M\}$

$f^{-1}(M) = \{w \in X^* \mid f(w) \in M\}$ $\quad f(M) = \{f(w) \mid w \in M\}$

Beweis der Abschlusseigenschaften

- Es seien M, M' aufzählbar mit Funktionen f und f'

  $\boldsymbol{M \cup M'}$:

  $\boldsymbol{M \cap M'}$:

  $\boldsymbol{M \circ M'}$:

  $\boldsymbol{M^*}$:

  $\boldsymbol{g(M)}$:

  $\boldsymbol{g^{-1}(M)}$:

Beweis der Abschlusseigenschaften

- Es seien M, M' aufzählbar mit Funktionen f und f'

$\boldsymbol{M \cup M'}$: Für $h(w) = \begin{cases} f(v) & \text{falls } w{=}0v, \\ f'(v) & \text{falls } w{=}1v \end{cases}$ ist $\text{range}(h) = M \cup M'$

$\boldsymbol{M \cap M'}$:

$\boldsymbol{M \circ M'}$:

$\boldsymbol{M^*}$:

$\boldsymbol{g(M)}$:

$\boldsymbol{g^{-1}(M)}$:

Beweis der Abschlusseigenschaften

- Es seien M, M' aufzählbar mit Funktionen f und f'

$\boldsymbol{M \cup M'}$: Für $h(w) = \begin{cases} f(v) & \text{falls } w{=}0v, \\ f'(v) & \text{falls } w{=}1v \end{cases}$ ist $\text{range}(h) = M \cup M'$

$\boldsymbol{M \cap M'}$: $\psi_{M \bigcap M'}(w) = \psi_M(w) * \psi_{M'}(w)$

$\boldsymbol{M \circ M'}$:

$\boldsymbol{M^*}$:

$\boldsymbol{g(M)}$:

$\boldsymbol{g^{-1}(M)}$:

Beweis der Abschlusseigenschaften

- Es seien M, M' aufzählbar mit Funktionen f und f'

$\boldsymbol{M \cup M'}$: Für $h(w) = \begin{cases} f(v) & \text{falls } w{=}0v, \\ f'(v) & \text{falls } w{=}1v \end{cases}$ ist $\text{range}(h) = M \cup M'$

$\boldsymbol{M \cap M'}$: $\psi_{M \cap M'}(w) = \psi_M(w) * \psi_{M'}(w)$

$\boldsymbol{M \circ M'}$: Für $h(w\#v) = f(w) \circ f'(v)$ ist $\text{range}(h) = M \circ M'$

$\boldsymbol{M^*}$:

$\boldsymbol{g(M)}$:

$\boldsymbol{g^{-1}(M)}$:

Beweis der Abschlusseigenschaften

- Es seien M, M' aufzählbar mit Funktionen f und f'

$\boldsymbol{M \cup M'}$: Für $h(w) = \begin{cases} f(v) & \text{falls } w{=}0v, \\ f'(v) & \text{falls } w{=}1v \end{cases}$ ist $\text{range}(h) = M \cup M'$

$\boldsymbol{M \cap M'}$: $\psi_{M \bigcap M'}(w) = \psi_M(w) * \psi_{M'}(w)$

$\boldsymbol{M \circ M'}$: Für $h(w\#v) = f(w) \circ f'(v)$ ist $\text{range}(h) = M \circ M'$

$\boldsymbol{M^*}$: Für $h(w_1\# \ldots \#w_n) = f(w_1) \circ \ldots \circ f(w_n)$ ist $\text{range}(h) = M^*$

$\boldsymbol{g(M)}$:

$\boldsymbol{g^{-1}(M)}$:

Beweis der Abschlusseigenschaften

- Es seien M, M' aufzählbar mit Funktionen f und f'

$\boldsymbol{M \cup M'}$: Für $h(w) = \begin{cases} f(v) & \text{falls } w{=}0v, \\ f'(v) & \text{falls } w{=}1v \end{cases}$ ist $\text{range}(h) = M \cup M'$

$\boldsymbol{M \cap M'}$: $\psi_{M \bigcap M'}(w) = \psi_M(w) * \psi_{M'}(w)$

$\boldsymbol{M \circ M'}$: Für $h(w\#v) = f(w) \circ f'(v)$ ist $\text{range}(h) = M \circ M'$

$\boldsymbol{M^*}$: Für $h(w_1\# \ldots \#w_n) = f(w_1) \circ \ldots \circ f(w_n)$ ist $\text{range}(h) = M^*$

$\boldsymbol{g(M)}$: Für $h(w) = g(f(w))$ ist $\text{range}(h) = g(M)$

$\boldsymbol{g^{-1}(M)}$:

Beweis der Abschlusseigenschaften

- Es seien M, M' aufzählbar mit Funktionen f und f'

$\boldsymbol{M \cup M'}$: Für $h(w) = \begin{cases} f(v) & \text{falls } w{=}0v, \\ f'(v) & \text{falls } w{=}1v \end{cases}$ ist $\text{range}(h) = M \cup M'$

$\boldsymbol{M \cap M'}$: $\psi_{M \bigcap M'}(w) = \psi_M(w) * \psi_{M'}(w)$

$\boldsymbol{M \circ M'}$: Für $h(w\#v) = f(w) \circ f'(v)$ ist $\text{range}(h) = M \circ M'$

$\boldsymbol{M^*}$: Für $h(w_1\# \ldots \#w_n) = f(w_1) \circ \ldots \circ f(w_n)$ ist $\text{range}(h) = M^*$

$\boldsymbol{g(M)}$: Für $h(w) = g(f(w))$ ist $\text{range}(h) = g(M)$

$\boldsymbol{g^{-1}(M)}$: $\psi_{g^{-1}(M)}(w) = \psi_M(g(w))$

Beweis der Abschlusseigenschaften

- Es seien M, M' aufzählbar mit Funktionen f und f'

  $\boldsymbol{M \cup M'}$: Für $h(w) = \begin{cases} f(v) & \text{falls } w{=}0v, \\ f'(v) & \text{falls } w{=}1v \end{cases}$ ist $\text{range}(h) = M \cup M'$

  $\boldsymbol{M \cap M'}$: $\psi_{M \cap M'}(w) = \psi_M(w) * \psi_{M'}(w)$

  $\boldsymbol{M \circ M'}$: Für $h(w\#v) = f(w) \circ f'(v)$ ist $\text{range}(h) = M \circ M'$

  $\boldsymbol{M^*}$: Für $h(w_1\# \ldots \#w_n) = f(w_1) \circ \ldots \circ f(w_n)$ ist $\text{range}(h) = M^*$

  $\boldsymbol{g(M)}$: Für $h(w) = g(f(w))$ ist $\text{range}(h) = g(M)$

  $\boldsymbol{g^{-1}(M)}$: $\psi_{g^{-1}(M)}(w) = \psi_M(g(w))$

- Entscheidbare Mengen sind abgeschlossen unter $\cup$, $\cap$, $\setminus$, $\bar{\ }$, $\circ$, *, f^{-1}

  Beweis in Übungen

Theoretische Informatik

Einheit 3.2

Universelle Maschinen

1. Standardnumerierung berechenbarer Funktionen
2. Universelle Funktion
3. Grundeigenschaften berechenbarer Funktionen

Noch Offene Fragen

- **Gibt es unentscheidbare Mengen?**
  - Unentscheidbar aber aufzählbar?
  - Nicht aufzählbar'?

Noch Offene Fragen

- **Gibt es unentscheidbare Mengen?**
  - Unentscheidbar aber aufzählbar?
  - Nicht aufzählbar'?
- **Gibt es unberechenbare Funktionen?**

Noch Offene Fragen

- **Gibt es unentscheidbare Mengen?**
  - Unentscheidbar aber aufzählbar?
  - Nicht aufzählbar'?
- **Gibt es unberechenbare Funktionen?**
- **Wie beweist man Unlösbarkeit?**
  - Kardinalitätsargument: es gibt mehr Funktionen als Programme
  - Konkretes Gegenbeispiel konstruieren

Noch Offene Fragen

- **Gibt es unentscheidbare Mengen?**
  - Unentscheidbar aber aufzählbar?
  - Nicht aufzählbar'?
- **Gibt es unberechenbare Funktionen?**
- **Wie beweist man Unlösbarkeit?**
  - Kardinalitätsargument: es gibt mehr Funktionen als Programme
  - Konkretes Gegenbeispiel konstruieren
- **Was benötigt man für diese Argumente?**
  - Präzisierung der Grundannahmen zur Berechenbarkeit
  - Nachweis, daß diese Grundannahmen erfüllt sind

Grundannahmen über Berechnungen

- **Programme und Daten sind als Zahlen codierbar**
  - Programme und Daten werden als Worte dargestellt
  - Worte, die Programme darstellen, können durchnumeriert werden
  - φ_i: Berechnete Funktion des Programms i $(\varphi_i : \mathbb{N} \to \mathbb{N})$
  - Φ_i: Rechenzeitfunktion zum Programm i $(\text{domain}(\Phi_i) = \text{domain}(\varphi_i))$

Grundannahmen über Berechnungen

- **Programme und Daten sind als Zahlen codierbar**
  - Programme und Daten werden als Worte dargestellt
  - Worte, die Programme darstellen, können durchnumeriert werden
  - φ_i: Berechnete Funktion des Programms i $(\varphi_i : \mathbb{N} \to \mathbb{N})$
  - Φ_i: Rechenzeitfunktion zum Programm i $(\text{domain}(\Phi_i) = \text{domain}(\varphi_i))$
- **Computer sind universelle Maschinen**
  - Bei Eingabe beliebiger Programme und Daten berechnen sie das Ergebnis
  - Die Funktion $u : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ mit $u(i, n) = \varphi_i(n)$ ist berechenbar

Grundannahmen über Berechnungen

- **Programme und Daten sind als Zahlen codierbar**
  - Programme und Daten werden als Worte dargestellt
  - Worte, die Programme darstellen, können durchnumeriert werden
  - φ_i: Berechnete Funktion des Programms i $(\varphi_i : \mathbb{N} \to \mathbb{N})$
  - Φ_i: Rechenzeitfunktion zum Programm i $(\text{domain}(\Phi_i) = \text{domain}(\varphi_i))$
- **Computer sind universelle Maschinen**
  - Bei Eingabe beliebiger Programme und Daten berechnen sie das Ergebnis
  - Die Funktion $u : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ mit $u(i, n) = \varphi_i(n)$ ist berechenbar
- **Man kann Programme effektiv zusammensetzen**
  - Die Nummer des entstehenden Programms kann berechnet werden
  - Es gibt eine berechenbare totale Funktion h mit $\varphi_{h(i,j)} = \varphi_i \circ \varphi_j$

Grundannahmen über Berechnungen

- **Programme und Daten sind als Zahlen codierbar**
  - Programme und Daten werden als Worte dargestellt
  - Worte, die Programme darstellen, können durchnumeriert werden
  - φ_i: Berechnete Funktion des Programms i $(\varphi_i : \mathbb{N} \to \mathbb{N})$
  - Φ_i: Rechenzeitfunktion zum Programm i $(\text{domain}(\Phi_i) = \text{domain}(\varphi_i))$
- **Computer sind universelle Maschinen**
  - Bei Eingabe beliebiger Programme und Daten berechnen sie das Ergebnis
  - Die Funktion $u : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ mit $u(i, n) = \varphi_i(n)$ ist berechenbar
- **Man kann Programme effektiv zusammensetzen**
  - Die Nummer des entstehenden Programms kann berechnet werden
  - Es gibt eine berechenbare totale Funktion h mit $\varphi_{h(i,j)} = \varphi_i \circ \varphi_j$
- **Rechenzeit ist entscheidbar**
  - Man kann für beliebige $i, n, t \in \mathbb{N}$ testen ob $\Phi_i(n) = t$ ist oder nicht

Numerierung von Turingmaschinen

- **Codierung aller Alphabete in einem Alphabet $\hat{\Gamma}$**

Numerierung von Turingmaschinen

- **Codierung aller Alphabete in einem Alphabet $\hat{\Gamma}$**
- **Codiere Zustandsüberführungsfunktion δ als Wort in $\hat{\Gamma}^*$**

Numerierung von Turingmaschinen

- **Codierung aller Alphabete in einem Alphabet $\hat{\Gamma}$**
- **Codiere Zustandsüberführungsfunktion δ als Wort in $\hat{\Gamma}^*$**
- **Codiere Komponenten der Turingmaschine τ als Wort in $\hat{\Gamma}^*$**

Numerierung von Turingmaschinen

- **Codierung aller Alphabete in einem Alphabet $\hat{\Gamma}$**
- **Codiere Zustandsüberführungsfunktion δ als Wort in $\hat{\Gamma}^*$**
- **Codiere Komponenten der Turingmaschine τ als Wort in $\hat{\Gamma}^*$**
- **Numeriere Worte über $X = \{x_1, .., x_n\}$ gemäß der lexikographischen Ordnung**

$$\epsilon < x_1 < \ldots < x_n < x_1x_1 < x_1x_2 < \ldots < x_nx_n < x_1x_1x_1 < \ldots$$

Numerierung von Turingmaschinen

- **Codierung aller Alphabete in einem Alphabet** $\hat{\Gamma}$
- **Codiere Zustandsüberführungsfunktion** δ **als Wort in** $\hat{\Gamma}^*$
- **Codiere Komponenten der Turingmaschine** τ **als Wort in** $\hat{\Gamma}^*$
- **Numeriere Worte über** $X = \{x_1, .., x_n\}$ **gemäß der lexikographischen Ordnung**

  $$\epsilon < x_1 < \ldots < x_n < x_1x_1 < x_1x_2 < \ldots < x_nx_n < x_1x_1x_1 < \ldots$$

- **Numeriere Worte, die Turingmaschinen codieren**
  - Benutze Numerierung aller Worte
  - Man kann testen, ob ein Wort $w \in \hat{\Gamma}^*$ ein Turingprogramm beschreibt
  - $\boldsymbol{\tau_i}$: Turingmaschine τ mit $w_\tau = \nu(n_\tau(i))$ "die i-te Turingmaschine"
  - **Gödelnummer** der Turingmaschine τ: Zahl i mit $\tau = \tau_i$

Numerierung berechenbarer Funktionen

- **Berechenbare Funktionen auf Worten**
  - $\hat{\varphi}_i \equiv h_{\tau_i}$: "die von der i-ten Turingmaschine berechnete Funktion"

Numerierung berechenbarer Funktionen

- **Berechenbare Funktionen auf Worten**
  - $\hat{\varphi}_i \equiv h_{\tau_i}$: "die von der i-ten Turingmaschine berechnete Funktion"
  - $\boldsymbol{t_i}$: Schrittzahlfunktion der Turingmaschine τ_i

$$t_i(w) = \begin{cases} m & \text{falls Berechnung von } \tau_i(w) \text{ in } m \text{ Schritten terminiert,} \\ \perp & \text{sonst} \end{cases}$$

Numerierung berechenbarer Funktionen

- **Berechenbare Funktionen auf Worten**
  - $\hat{\varphi}_i \equiv h_{\tau_i}$: "die von der i-ten Turingmaschine berechnete Funktion"
  - $\boldsymbol{t_i}$: Schrittzahlfunktion der Turingmaschine τ_i

$$t_i(w) = \begin{cases} m & \text{falls Berechnung von } \tau_i(w) \text{ in } m \text{ Schritten terminiert,} \\ \perp & \text{sonst} \end{cases}$$

- **Berechenbare Funktionen auf Zahlen**
  - $\boldsymbol{\varphi_i} \equiv r^{-1} \circ \hat{\varphi}_i \circ r : \mathbb{N} \to \mathbb{N}$ "die i-te berechenbare Funktion"

    $r{:}\mathbb{N} \to X^*$ bijektive Repräsentation von Zahlen als Worte
  - $\boldsymbol{\Phi_i} \equiv t_i \circ r : \mathbb{N} \to \mathbb{N}$ "Schrittzahlfunktion von φ_i"

Numerierung berechenbarer Funktionen

- **Berechenbare Funktionen auf Worten**
  - $\hat{\varphi}_i \equiv h_{\tau_i}$: "die von der i-ten Turingmaschine berechnete Funktion"
  - $\boldsymbol{t_i}$: Schrittzahlfunktion der Turingmaschine τ_i

$$t_i(w) = \begin{cases} m & \text{falls Berechnung von } \tau_i(w) \text{ in } m \text{ Schritten terminiert,} \\ \perp & \text{sonst} \end{cases}$$

- **Berechenbare Funktionen auf Zahlen**
  - $\boldsymbol{\varphi_i} \equiv r^{-1} \circ \hat{\varphi}_i \circ r : \mathbb{N} \to \mathbb{N}$ "die i-te berechenbare Funktion"

    $r{:}\mathbb{N} \to X^*$ bijektive Repräsentation von Zahlen als Worte
  - $\boldsymbol{\Phi_i} \equiv t_i \circ r : \mathbb{N} \to \mathbb{N}$ "Schrittzahlfunktion von φ_i"

- **Eigenschaften von $\boldsymbol{\varphi}$ und $\boldsymbol{\Phi}$**
  - φ is surjektiv, aber nicht bijektiv
  - $\text{domain}(\Phi_i) = \text{domain}(\varphi_i)$ (Φ_i terminiert auf den gleichen Eingaben wie φ_i)
  - $\{(i, n, t) \mid \Phi_i(n) = t\}$ ist entscheidbar "Rechenzeit ist entscheidbar"

Numerierung berechenbarer Funktionen

- **Berechenbare Funktionen auf Worten**
  - $\hat{\varphi}_i \equiv h_{\tau_i}$: "die von der i-ten Turingmaschine berechnete Funktion"
  - $\boldsymbol{t_i}$: Schrittzahlfunktion der Turingmaschine τ_i

$$t_i(w) = \begin{cases} m & \text{falls Berechnung von } \tau_i(w) \text{ in } m \text{ Schritten terminiert,} \\ \perp & \text{sonst} \end{cases}$$

- **Berechenbare Funktionen auf Zahlen**
  - $\boldsymbol{\varphi_i} \equiv r^{-1} \circ \hat{\varphi}_i \circ r : \mathbb{N} \to \mathbb{N}$ "die i-te berechenbare Funktion"

    $r{:}\mathbb{N} \to X^*$ bijektive Repräsentation von Zahlen als Worte
  - $\boldsymbol{\Phi_i} \equiv t_i \circ r : \mathbb{N} \to \mathbb{N}$ "Schrittzahlfunktion von φ_i"

- **Eigenschaften von $\boldsymbol{\varphi}$ und $\boldsymbol{\Phi}$**
  - φ is surjektiv, aber nicht bijektiv
  - $\text{domain}(\Phi_i) = \text{domain}(\varphi_i)$ (Φ_i terminiert auf den gleichen Eingaben wie φ_i)
  - $\{(i, n, t) \mid \Phi_i(n) = t\}$ ist entscheidbar "Rechenzeit ist entscheidbar"

Die Numerierung berechenbarer Funktionen ist nur surjektiv

Universelle Turingmaschinen

Kann man alle Turingprogramme auf einer einzigen Maschine ausführen?

Universelle Turingmaschinen

Kann man alle Turingprogramme auf einer einzigen Maschine ausführen?

- **Universelle Maschinen**
  - τ_u ist universell, wenn $h_{\tau_u}(w_\tau, v) = h_\tau(v)$ für jede TM τ und jedes $v \in X^*$

Universelle Turingmaschinen

Kann man alle Turingprogramme auf einer einzigen Maschine ausführen?

- **Universelle Maschinen**
  - τ_u ist universell, wenn $h_{\tau_u}(w_\tau, v) = h_\tau(v)$ für jede TM τ und jedes $v \in X^*$
  - Insbesondere $h_{\tau_u}(r(i), v) = h_{\tau_i}(v)$ für alle i,v (r:$\mathbb{N} \to X^*$ Zahlendarstellung)

Universelle Turingmaschinen

Kann man alle Turingprogramme auf einer einzigen Maschine ausführen?

- **Universelle Maschinen**
  - τ_u ist universell, wenn $h_{\tau_u}(w_\tau, v) = h_\tau(v)$ für jede TM τ und jedes $v \in X^*$
  - Insbesondere $h_{\tau_u}(r(i), v) = h_{\tau_i}(v)$ für alle i,v (r:$\mathbb{N} \to X^*$ Zahlendarstellung)
- **Universelle Funktionen**
  - u:$\mathbb{N} \times \mathbb{N} \to \mathbb{N}$ ist universell, wenn $u(i, n) = \varphi_i(n)$ für alle $i, n \in \mathbb{N}$

Universelle Turingmaschinen

Kann man alle Turingprogramme auf einer einzigen Maschine ausführen?

- **Universelle Maschinen**
  - τ_u ist universell, wenn $h_{\tau_u}(w_\tau, v) = h_\tau(v)$ für jede TM τ und jedes $v \in X^*$
  - Insbesondere $h_{\tau_u}(r(i), v) = h_{\tau_i}(v)$ für alle i,v (r:$\mathbb{N} \to X^*$ Zahlendarstellung)
- **Universelle Funktionen**
  - u:$\mathbb{N} \times \mathbb{N} \to \mathbb{N}$ ist universell, wenn $u(i, n) = \varphi_i(n)$ für alle $i, n \in \mathbb{N}$
- **Gibt es universelle Maschinen?**

Universelle Turingmaschinen

Kann man alle Turingprogramme auf einer einzigen Maschine ausführen?

- **Universelle Maschinen**
  - τ_u ist universell, wenn $h_{\tau_u}(w_\tau, v) = h_\tau(v)$ für jede TM τ und jedes $v \in X^*$
  - Insbesondere $h_{\tau_u}(r(i), v) = h_{\tau_i}(v)$ für alle i,v (r:$\mathbb{N} \to X^*$ Zahlendarstellung)
- **Universelle Funktionen**
  - u:$\mathbb{N} \times \mathbb{N} \to \mathbb{N}$ ist universell, wenn $u(i, n) = \varphi_i(n)$ für alle $i, n \in \mathbb{N}$
- **Gibt es universelle Maschinen?**
  - Die Numerierung n_τ ist berechenbar

Universelle Turingmaschinen

Kann man alle Turingprogramme auf einer einzigen Maschine ausführen?

- **Universelle Maschinen**
  - τ_u ist universell, wenn $h_{\tau_u}(w_\tau, v) = h_\tau(v)$ für jede TM τ und jedes $v \in X^*$
  - Insbesondere $h_{\tau_u}(r(i), v) = h_{\tau_i}(v)$ für alle i,v (r:$\mathbb{N} \to X^*$ Zahlendarstellung)
- **Universelle Funktionen**
  - u:$\mathbb{N} \times \mathbb{N} \to \mathbb{N}$ ist universell, wenn $u(i, n) = \varphi_i(n)$ für alle $i, n \in \mathbb{N}$
- **Gibt es universelle Maschinen?**
  - Die Numerierung n_τ ist berechenbar
  - Turingprogramme lassen sich simulieren

Universelle Turingmaschinen

Kann man alle Turingprogramme auf einer einzigen Maschine ausführen?

- **Universelle Maschinen**
  - τ_u ist universell, wenn $h_{\tau_u}(w_\tau, v) = h_\tau(v)$ für jede TM τ und jedes $v \in X^*$
  - Insbesondere $h_{\tau_u}(r(i), v) = h_{\tau_i}(v)$ für alle i,v (r:$\mathbb{N} \to X^*$ Zahlendarstellung)
- **Universelle Funktionen**
  - u:$\mathbb{N} \times \mathbb{N} \to \mathbb{N}$ ist universell, wenn $u(i, n) = \varphi_i(n)$ für alle $i, n \in \mathbb{N}$
- **Gibt es universelle Maschinen?**
  - Die Numerierung n_τ ist berechenbar
  - Turingprogramme lassen sich simulieren
  - Baue universelle Maschine mit $\nu \circ n_\tau$ und Einzelschrittsimulation

Universelle Turingmaschinen

Kann man alle Turingprogramme auf einer einzigen Maschine ausführen?

- **Universelle Maschinen**
  - τ_u ist universell, wenn $h_{\tau_u}(w_\tau, v) = h_\tau(v)$ für jede TM τ und jedes $v \in X^*$
  - Insbesondere $h_{\tau_u}(r(i), v) = h_{\tau_i}(v)$ für alle i,v (r:$\mathbb{N} \to X^*$ Zahlendarstellung)
- **Universelle Funktionen**
  - u:$\mathbb{N} \times \mathbb{N} \to \mathbb{N}$ ist universell, wenn $u(i, n) = \varphi_i(n)$ für alle $i, n \in \mathbb{N}$
- **Gibt es universelle Maschinen?**
  - Die Numerierung n_τ ist berechenbar
  - Turingprogramme lassen sich simulieren
  - Baue universelle Maschine mit $\nu \circ n_\tau$ und Einzelschrittsimulation

Details z.B. in Hopcroft, Motwani, Ullman, Seite 387–389

Das Übersetzungslemma

Turingmaschinen sind effektiv kombinierbar

Das Übersetzungslemma

Turingmaschinen sind effektiv kombinierbar

- **Kombiniere τ_1 und τ_2 zu τ mit $h_\tau = h_{\tau_1} \circ h_{\tau_2}$**

Das Übersetzungslemma

Turingmaschinen sind effektiv kombinierbar

- **Kombiniere τ_1 und τ_2 zu τ mit $h_\tau = h_{\tau_1} \circ h_{\tau_2}$**
  - Umbenennung der Zustände von τ_2
  - Springe vom "Endzustand" von τ_1 zum Anfangszustand von τ_2

Das Übersetzungslemma

Turingmaschinen sind effektiv kombinierbar

- **Kombiniere τ_1 und τ_2 zu τ mit $h_\tau = h_{\tau_1} \circ h_{\tau_2}$**
  - Umbenennung der Zustände von τ_2
  - Springe vom "Endzustand" von τ_1 zum Anfangszustand von τ_2
  - Programm w_τ kann aus w_{τ_1} und w_{τ_2} berechnet werden

Das Übersetzungslemma

Turingmaschinen sind effektiv kombinierbar

- **Kombiniere τ_1 und τ_2 zu τ mit $h_\tau = h_{\tau_1} \circ h_{\tau_2}$**
  - Umbenennung der Zustände von τ_2
  - Springe vom "Endzustand" von τ_1 zum Anfangszustand von τ_2
  - Programm w_τ kann aus w_{τ_1} und w_{τ_2} berechnet werden
  - Gödelnummer k von τ kann aus denen für τ_1 und τ_2 berechnet werden

Das Übersetzungslemma

Turingmaschinen sind effektiv kombinierbar

- **Kombiniere τ_1 und τ_2 zu τ mit $h_\tau = h_{\tau_1} \circ h_{\tau_2}$**
  - Umbenennung der Zustände von τ_2
  - Springe vom "Endzustand" von τ_1 zum Anfangszustand von τ_2
  - Programm w_τ kann aus w_{τ_1} und w_{τ_2} berechnet werden
  - Gödelnummer k von τ kann aus denen für τ_1 und τ_2 berechnet werden
- **Kombiniere φ_i und φ_j zu φ_k mit $\varphi_k = \varphi_i \circ \varphi_j$**

Das Übersetzungslemma

Turingmaschinen sind effektiv kombinierbar

- **Kombiniere τ_1 und τ_2 zu τ mit $h_\tau = h_{\tau_1} \circ h_{\tau_2}$**
  - Umbenennung der Zustände von τ_2
  - Springe vom "Endzustand" von τ_1 zum Anfangszustand von τ_2
  - Programm w_τ kann aus w_{τ_1} und w_{τ_2} berechnet werden
  - Gödelnummer k von τ kann aus denen für τ_1 und τ_2 berechnet werden
- **Kombiniere φ_i und φ_j zu φ_k mit $\varphi_k = \varphi_i \circ \varphi_j$**
  - Index k kann aus i und j berechnet werden

Das Übersetzungslemma

Turingmaschinen sind effektiv kombinierbar

- **Kombiniere τ_1 und τ_2 zu τ mit $h_\tau = h_{\tau_1} \circ h_{\tau_2}$**
  - Umbenennung der Zustände von τ_2
  - Springe vom "Endzustand" von τ_1 zum Anfangszustand von τ_2
  - Programm w_τ kann aus w_{τ_1} und w_{τ_2} berechnet werden
  - Gödelnummer k von τ kann aus denen für τ_1 und τ_2 berechnet werden
- **Kombiniere φ_i und φ_j zu φ_k mit $\varphi_k = \varphi_i \circ \varphi_j$**
  - Index k kann aus i und j berechnet werden
  - Es gibt eine berechenbare totale Funktion h mit $\varphi_{h(i,j)} = \varphi_i \circ \varphi_j$

Das Übersetzungslemma

Turingmaschinen sind effektiv kombinierbar

- **Kombiniere τ_1 und τ_2 zu τ mit $h_\tau = h_{\tau_1} \circ h_{\tau_2}$**
  - Umbenennung der Zustände von τ_2
  - Springe vom "Endzustand" von τ_1 zum Anfangszustand von τ_2
  - Programm w_τ kann aus w_{τ_1} und w_{τ_2} berechnet werden
  - Gödelnummer k von τ kann aus denen für τ_1 und τ_2 berechnet werden
- **Kombiniere φ_i und φ_j zu φ_k mit $\varphi_k = \varphi_i \circ \varphi_j$**
  - Index k kann aus i und j berechnet werden
  - Es gibt eine berechenbare totale Funktion h mit $\varphi_{h(i,j)} = \varphi_i \circ \varphi_j$
- **Allgemeinste Version: SMN Theorem**

Das Übersetzungslemma

Turingmaschinen sind effektiv kombinierbar

- **Kombiniere τ_1 und τ_2 zu τ mit $h_\tau = h_{\tau_1} \circ h_{\tau_2}$**
  - Umbenennung der Zustände von τ_2
  - Springe vom "Endzustand" von τ_1 zum Anfangszustand von τ_2
  - Programm w_τ kann aus w_{τ_1} und w_{τ_2} berechnet werden
  - Gödelnummer k von τ kann aus denen für τ_1 und τ_2 berechnet werden
- **Kombiniere φ_i und φ_j zu φ_k mit $\varphi_k = \varphi_i \circ \varphi_j$**
  - Index k kann aus i und j berechnet werden
  - Es gibt eine berechenbare totale Funktion h mit $\varphi_{h(i,j)} = \varphi_i \circ \varphi_j$
- **Allgemeinste Version: SMN Theorem**
  - Es gibt eine berechenbare totale Funktion s mit $\varphi_{s\langle m,n\rangle}(i) = \varphi_m\langle n, i\rangle$

Das Übersetzungslemma

Turingmaschinen sind effektiv kombinierbar

- **Kombiniere τ_1 und τ_2 zu τ mit $h_\tau = h_{\tau_1} \circ h_{\tau_2}$**
  - Umbenennung der Zustände von τ_2
  - Springe vom "Endzustand" von τ_1 zum Anfangszustand von τ_2
  - Programm w_τ kann aus w_{τ_1} und w_{τ_2} berechnet werden
  - Gödelnummer k von τ kann aus denen für τ_1 und τ_2 berechnet werden
- **Kombiniere φ_i und φ_j zu φ_k mit $\varphi_k = \varphi_i \circ \varphi_j$**
  - Index k kann aus i und j berechnet werden
  - Es gibt eine berechenbare totale Funktion h mit $\varphi_{h(i,j)} = \varphi_i \circ \varphi_j$
- **Allgemeinste Version: SMN Theorem**
  - Es gibt eine berechenbare totale Funktion s mit $\varphi_{s\langle m,n\rangle}(i) = \varphi_m\langle n, i\rangle$

Technisches Resultat mit wenig eigener Bedeutung

Zusammenfassung: Kernaxiome der Berechenbareitstheorie

- **Berechenbare Funktionen sind effektiv numerierbar**
  - $\varphi_i : \mathbb{N} \to \mathbb{N}$: berechnete Funktion des Programms i
  - Φ_i: Rechenzeitfunktion zum Programm i
  - $\mathbf{domain(\Phi_i) = domain(\varphi_i)}$

Zusammenfassung: Kernaxiome der Berechenbareitstheorie

- **Berechenbare Funktionen sind effektiv numerierbar**
  - $\varphi_i : \mathbb{N} \to \mathbb{N}$: berechnete Funktion des Programms i
  - Φ_i: Rechenzeitfunktion zum Programm i
  - $\mathbf{domain}(\boldsymbol{\Phi_i}) = \mathbf{domain}(\boldsymbol{\varphi_i})$
- **Die Menge $\{(i, n, t) \mid \Phi_i(n) = t\}$ ist entscheidbar**

Zusammenfassung: Kernaxiome der Berechenbareitstheorie

- **Berechenbare Funktionen sind effektiv numerierbar**
  - $\varphi_i : \mathbb{N} \to \mathbb{N}$: berechnete Funktion des Programms i
  - Φ_i: Rechenzeitfunktion zum Programm i
  - $\mathbf{domain}(\boldsymbol{\Phi_i}) = \mathbf{domain}(\boldsymbol{\varphi_i})$
- **Die Menge $\{(i, n, t) \mid \Phi_i(n) = t\}$ ist entscheidbar**
- **Die universelle Funktion ist berechenbar**
  - $u : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ mit $u(i, n) = \varphi_i(n)$ ist berechenbar

Zusammenfassung: Kernaxiome der Berechenbareitstheorie

- **Berechenbare Funktionen sind effektiv numerierbar**
  - $\varphi_i : \mathbb{N} \to \mathbb{N}$: berechnete Funktion des Programms i
  - Φ_i: Rechenzeitfunktion zum Programm i
  - $\mathbf{domain}(\boldsymbol{\Phi_i}) = \mathbf{domain}(\boldsymbol{\varphi_i})$
- **Die Menge $\{(i, n, t) \mid \Phi_i(n) = t\}$ ist entscheidbar**
- **Die universelle Funktion ist berechenbar**
  - $u : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ mit $u(i, n) = \varphi_i(n)$ ist berechenbar
- **Programme sind effektiv kombinierbar**
  - Es gibt eine berechenbare totale Funktion h mit $\varphi_{h(i,j)} = \varphi_i \circ \varphi_j$
  - Es gibt eine berechenbare totale Funktion s mit $\varphi_{s\langle m,n\rangle}(i) = \varphi_m\langle n, i\rangle$

Zusammenfassung: Kernaxiome der Berechenbareitstheorie

- **Berechenbare Funktionen sind effektiv numerierbar**
  - $\varphi_i : \mathbb{N} \to \mathbb{N}$: berechnete Funktion des Programms i
  - Φ_i: Rechenzeitfunktion zum Programm i
  - $\mathbf{domain}(\boldsymbol{\Phi_i}) = \mathbf{domain}(\boldsymbol{\varphi_i})$
- **Die Menge $\{(i, n, t) \mid \Phi_i(n) = t\}$ ist entscheidbar**
- **Die universelle Funktion ist berechenbar**
  - $u : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ mit $u(i, n) = \varphi_i(n)$ ist berechenbar
- **Programme sind effektiv kombinierbar**
  - Es gibt eine berechenbare totale Funktion h mit $\varphi_{h(i,j)} = \varphi_i \circ \varphi_j$
  - Es gibt eine berechenbare totale Funktion s mit $\varphi_{s\langle m,n\rangle}(i) = \varphi_m\langle n, i\rangle$

Alles weitere folgt aus diesen Axiomen

Wichtige entscheidbare und aufzählbare Mengen

- $\{(i, n, t) \mid \Phi_i(n) = t\}$ **ist entscheidbar**

Wichtige entscheidbare und aufzählbare Mengen

- $\{(i, n, t) \mid \Phi_i(n) = t\}$ **ist entscheidbar**
  - Kernaxiom der Berechenbarkeitstheorie

Wichtige entscheidbare und aufzählbare Mengen

- $\{(i, n, t) \mid \Phi_i(n) = t\}$ **ist entscheidbar**
  - Kernaxiom der Berechenbarkeitstheorie
- $\{(i, n, y) \mid \varphi_i(n) = y\}$ **ist aufzählbar**

Wichtige entscheidbare und aufzählbare Mengen

- $\{(i, n, t) \mid \Phi_i(n) = t\}$ **ist entscheidbar**
  - Kernaxiom der Berechenbarkeitstheorie
- $\{(i, n, y) \mid \varphi_i(n) = y\}$ **ist aufzählbar**
  - Graph der universellen Funktion

Wichtige entscheidbare und aufzählbare Mengen

- $\{(i,n,t) \mid \Phi_i(n) = t\}$ **ist entscheidbar**
  - Kernaxiom der Berechenbarkeitstheorie
- $\{(i,n,y) \mid \varphi_i(n) = y\}$ **ist aufzählbar**
  - Graph der universellen Funktion
- $H = \{(i,n) \mid \varphi_i(n) \neq \perp\}$ **ist aufzählbar**

Wichtige entscheidbare und aufzählbare Mengen

- $\{(i,n,t) \mid \Phi_i(n) = t\}$ **ist entscheidbar**
  - Kernaxiom der Berechenbarkeitstheorie
- $\{(i,n,y) \mid \varphi_i(n) = y\}$ **ist aufzählbar**
  - Graph der universellen Funktion
- $H = \{(i,n) \mid \varphi_i(n) \neq \perp\}$ **ist aufzählbar**
  - Haltebereich der universellen Funktion

Wichtige entscheidbare und aufzählbare Mengen

- $\{(i,n,t) \mid \Phi_i(n) = t\}$ **ist entscheidbar**
  - Kernaxiom der Berechenbarkeitstheorie
- $\{(i,n,y) \mid \varphi_i(n) = y\}$ **ist aufzählbar**
  - Graph der universellen Funktion
- $H = \{(i,n) \mid \varphi_i(n) \neq \perp\}$ **ist aufzählbar**
  - Haltebereich der universellen Funktion
- $S = \{i \mid \varphi_i(i) \neq \perp\}$ **ist aufzählbar**

Wichtige entscheidbare und aufzählbare Mengen

- $\{(i,n,t) \mid \Phi_i(n) = t\}$ **ist entscheidbar**
  - Kernaxiom der Berechenbarkeitstheorie
- $\{(i,n,y) \mid \varphi_i(n) = y\}$ **ist aufzählbar**
  - Graph der universellen Funktion
- $H = \{(i,n) \mid \varphi_i(n) \neq \perp\}$ **ist aufzählbar**
  - Haltebereich der universellen Funktion
- $S = \{i \mid \varphi_i(i) \neq \perp\}$ **ist aufzählbar**
  - Haltebereich von $\lambda i.u(i,i)$

Theoretische Informatik

Einheit 3.3

Beweistechniken für unlösbare Probleme

1. Diagonalisierung
2. Monotoniegargumente
3. Problemreduktion
4. Der Satz von Rice

Grenzen der Berechenbarkeit

Wie beweist man die Unlösbarkeit eines Problems?

Grenzen der Berechenbarkeit

Wie beweist man die Unlösbarkeit eines Problems?

- **Diagonalisierung**
  - Zeige, daß eine Funktion von jeder berechenbaren Funktion an mindestens einer Stelle abweicht, also selbst nicht berechenbar sein kann

Grenzen der Berechenbarkeit

Wie beweist man die Unlösbarkeit eines Problems?

- **Diagonalisierung**
  - Zeige, daß eine Funktion von jeder berechenbaren Funktion an mindestens einer Stelle abweicht, also selbst nicht berechenbar sein kann
- **Wachstums- und Monotonieargumente**
  - Zeige, daß eine Funktion stärker wächst als jede berechenbare Funktion

Grenzen der Berechenbarkeit

Wie beweist man die Unlösbarkeit eines Problems?

- **Diagonalisierung**
  - Zeige, daß eine Funktion von jeder berechenbaren Funktion an mindestens einer Stelle abweicht, also selbst nicht berechenbar sein kann
- **Wachstums- und Monotonieargumente**
  - Zeige, daß eine Funktion stärker wächst als jede berechenbare Funktion
- **Reduktionsmethode und Abschlußeigenschaften**
  - Zeige, daß Lösung des Problems zu einer Lösung eines bekanntermaßen unlösbaren Problems führen würde

Grenzen der Berechenbarkeit

Wie beweist man die Unlösbarkeit eines Problems?

- **Diagonalisierung**
  - Zeige, daß eine Funktion von jeder berechenbaren Funktion an mindestens einer Stelle abweicht, also selbst nicht berechenbar sein kann
- **Wachstums- und Monotonieargumente**
  - Zeige, daß eine Funktion stärker wächst als jede berechenbare Funktion
- **Reduktionsmethode und Abschlußeigenschaften**
  - Zeige, daß Lösung des Problems zu einer Lösung eines bekanntermaßen unlösbaren Problems führen würde
- **Anwendung allgemeiner theoretischer Resultate**
  - Unlösbarkeit folgt direkt aus bekannten Sätzen

Diagonalisierung

- **Ziel**
  - Zeige, daß eine unendliche Menge M eine Eigenschaft P nicht besitzt
  - z.B. Entscheidbarkeit, Aufzählbarkeit, Abzählbarkeit

Diagonalisierung

- **Ziel**
  - Zeige, daß eine unendliche Menge M eine Eigenschaft P nicht besitzt
  - z.B. Entscheidbarkeit, Aufzählbarkeit, Abzählbarkeit
- **Methodik**

Diagonalisierung

- **Ziel**
  - Zeige, daß eine unendliche Menge M eine Eigenschaft P nicht besitzt
  - z.B. Entscheidbarkeit, Aufzählbarkeit, Abzählbarkeit
- **Methodik**
  - Wir nehmen an, M habe die Eigenschaft P

Diagonalisierung

- **Ziel**
  - Zeige, daß eine unendliche Menge M eine Eigenschaft P nicht besitzt
  - z.B. Entscheidbarkeit, Aufzählbarkeit, Abzählbarkeit
- **Methodik**
  - Wir nehmen an, M habe die Eigenschaft P
  - Konstruiere ein Element x, das von allen Elementen von M verschieden ist

Diagonalisierung

- **Ziel**
  - Zeige, daß eine unendliche Menge M eine Eigenschaft P nicht besitzt
  - z.B. Entscheidbarkeit, Aufzählbarkeit, Abzählbarkeit
- **Methodik**
  - Wir nehmen an, M habe die Eigenschaft P
  - Konstruiere ein Element x, das von allen Elementen von M verschieden ist
  - Zeige, daß $x \in M$ aufgrund der Annahme gelten muß

Diagonalisierung

- **Ziel**
  - Zeige, daß eine unendliche Menge M eine Eigenschaft P nicht besitzt
  - z.B. Entscheidbarkeit, Aufzählbarkeit, Abzählbarkeit
- **Methodik**
  - Wir nehmen an, M habe die Eigenschaft P
  - Konstruiere ein Element x, das von allen Elementen von M verschieden ist
  - Zeige, daß $x \in M$ aufgrund der Annahme gelten muß
  - Aus dem Widerspruch folgt, daß die Annahme nicht gelten kann

Diagonalisierung

- **Ziel**
  - Zeige, daß eine unendliche Menge M eine Eigenschaft P nicht besitzt
  - z.B. Entscheidbarkeit, Aufzählbarkeit, Abzählbarkeit
- **Methodik**
  - Wir nehmen an, M habe die Eigenschaft P
  - Konstruiere ein Element x, das von allen Elementen von M verschieden ist
  - Zeige, daß $x \in M$ aufgrund der Annahme gelten muß
  - Aus dem Widerspruch folgt, daß die Annahme nicht gelten kann
- **Konstruktion des neuen Elementes**

  Cantor'sches Diagonalverfahren:

Diagonalisierung

- **Ziel**
  - Zeige, daß eine unendliche Menge M eine Eigenschaft P nicht besitzt
  - z.B. Entscheidbarkeit, Aufzählbarkeit, Abzählbarkeit
- **Methodik**
  - Wir nehmen an, M habe die Eigenschaft P
  - Konstruiere ein Element x, das von allen Elementen von M verschieden ist
  - Zeige, daß $x \in M$ aufgrund der Annahme gelten muß
  - Aus dem Widerspruch folgt, daß die Annahme nicht gelten kann
- **Konstruktion des neuen Elementes**

  Cantor'sches Diagonalverfahren:
  - Trage alle Elemente von M als Zeilen einer Tabelle auf

Diagonalisierung

- **Ziel**
  - Zeige, daß eine unendliche Menge M eine Eigenschaft P nicht besitzt
  - z.B. Entscheidbarkeit, Aufzählbarkeit, Abzählbarkeit
- **Methodik**
  - Wir nehmen an, M habe die Eigenschaft P
  - Konstruiere ein Element x, das von allen Elementen von M verschieden ist
  - Zeige, daß $x \in M$ aufgrund der Annahme gelten muß
  - Aus dem Widerspruch folgt, daß die Annahme nicht gelten kann
- **Konstruktion des neuen Elementes**

  Cantor'sches Diagonalverfahren:
  - Trage alle Elemente von M als Zeilen einer Tabelle auf
  - Konstruiere x auf Diagonale mit Abweichung an jedem Punkt

Diagonalisierung

- **Ziel**
  - Zeige, daß eine unendliche Menge M eine Eigenschaft P nicht besitzt
  - z.B. Entscheidbarkeit, Aufzählbarkeit, Abzählbarkeit
- **Methodik**
  - Wir nehmen an, M habe die Eigenschaft P
  - Konstruiere ein Element x, das von allen Elementen von M verschieden ist
  - Zeige, daß $x \in M$ aufgrund der Annahme gelten muß
  - Aus dem Widerspruch folgt, daß die Annahme nicht gelten kann
- **Konstruktion des neuen Elementes**

  Cantor'sches Diagonalverfahren:
  - Trage alle Elemente von M als Zeilen einer Tabelle auf
  - Konstruiere x auf Diagonale mit Abweichung an jedem Punkt
  - Also kann x nicht als Zeile vorkommen

Diagonalbeweise I: Überabzählbarkeit von $\mathbb{N}\to\mathbb{N}$

- **Die Menge $\mathbb{N}\to\mathbb{N}$ "überabzählbar" unendlich**
  - Die Menge aller Funktionen über $\mathbb{N}$ kann nicht durchnumeriert werden

Diagonalbeweise I: Überabzählbarkeit von $\mathbb{N}\to\mathbb{N}$

- **Die Menge $\mathbb{N}\to\mathbb{N}$ "überabzählbar" unendlich**
  - Die Menge aller Funktionen über $\mathbb{N}$ kann nicht durchnumeriert werden
  - **Annahme**: $\mathbb{N}\to\mathbb{N}$ ist abzählbar

Diagonalbeweise I: Überabzählbarkeit von $\mathbb{N}\to\mathbb{N}$

- **Die Menge $\mathbb{N}\to\mathbb{N}$ "überabzählbar" unendlich**
  - Die Menge aller Funktionen über $\mathbb{N}$ kann nicht durchnumeriert werden
  - **Annahme**: $\mathbb{N}\to\mathbb{N}$ ist abzählbar
  - Dann können alle Funktionen über $\mathbb{N}$ in eine Tabelle eingetragen werden

	0	1	2	3	4	...
f_0	$f_0(0)$	$f_0(1)$	$f_0(2)$	$f_0(3)$	$f_0(4)$	...
f_1	$f_1(0)$	$f_1(1)$	$f_1(2)$	$f_1(3)$	$f_1(4)$	...
f_2	$f_2(0)$	$f_2(1)$	$f_2(2)$	$f_2(3)$	$f_2(4)$	...
f_3	$f_3(0)$	$f_3(1)$	$f_3(2)$	$f_3(3)$	$f_3(4)$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

Diagonalbeweise I: Überabzählbarkeit von $\mathbb{N}\to\mathbb{N}$

- **Die Menge $\mathbb{N}\to\mathbb{N}$ "überabzählbar" unendlich**
  - Die Menge aller Funktionen über $\mathbb{N}$ kann nicht durchnumeriert werden
  - **Annahme**: $\mathbb{N}\to\mathbb{N}$ ist abzählbar
  - Dann können alle Funktionen über $\mathbb{N}$ in eine Tabelle eingetragen werden

	0	1	2	3	4	...
f_0	$f_0(0)+1$	$f_0(1)$	$f_0(2)$	$f_0(3)$	$f_0(4)$	...
f_1	$f_1(0)$	$f_1(1)$	$f_1(2)$	$f_1(3)$	$f_1(4)$	...
f_2	$f_2(0)$	$f_2(1)$	$f_2(2)$	$f_2(3)$	$f_2(4)$	...
f_3	$f_3(0)$	$f_3(1)$	$f_3(2)$	$f_3(3)$	$f_3(4)$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

  - Definiere eine neue Funktion $f{:}\mathbb{N}\to\mathbb{N}$ durch $f(x) = f_x(x)+1$

Diagonalbeweise I: Überabzählbarkeit von $\mathbb{N}\to\mathbb{N}$

- **Die Menge $\mathbb{N}\to\mathbb{N}$ "überabzählbar" unendlich**
  - Die Menge aller Funktionen über $\mathbb{N}$ kann nicht durchnumeriert werden
  - **Annahme**: $\mathbb{N}\to\mathbb{N}$ ist abzählbar
  - Dann können alle Funktionen über $\mathbb{N}$ in eine Tabelle eingetragen werden

	0	1	2	3	4	...
f_0	$f_0(0)+1$	$f_0(1)$	$f_0(2)$	$f_0(3)$	$f_0(4)$	...
f_1	$f_1(0)$	$f_1(1)+1$	$f_1(2)$	$f_1(3)$	$f_1(4)$	...
f_2	$f_2(0)$	$f_2(1)$	$f_2(2)$	$f_2(3)$	$f_2(4)$	...
f_3	$f_3(0)$	$f_3(1)$	$f_3(2)$	$f_3(3)$	$f_3(4)$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

  - Definiere eine neue Funktion $f{:}\mathbb{N}\to\mathbb{N}$ durch $f(x) = f_x(x)+1$

Diagonalbeweise I: Überabzählbarkeit von $\mathbb{N}\to\mathbb{N}$

- **Die Menge $\mathbb{N}\to\mathbb{N}$ "überabzählbar" unendlich**
  - Die Menge aller Funktionen über $\mathbb{N}$ kann nicht durchnumeriert werden
  - **Annahme**: $\mathbb{N}\to\mathbb{N}$ ist abzählbar
  - Dann können alle Funktionen über $\mathbb{N}$ in eine Tabelle eingetragen werden

	0	1	2	3	4	...
f_0	$f_0(0)+1$	$f_0(1)$	$f_0(2)$	$f_0(3)$	$f_0(4)$	...
f_1	$f_1(0)$	$f_1(1)+1$	$f_1(2)$	$f_1(3)$	$f_1(4)$	...
f_2	$f_2(0)$	$f_2(1)$	$f_2(2)+1$	$f_2(3)$	$f_2(4)$	...
f_3	$f_3(0)$	$f_3(1)$	$f_3(2)$	$f_3(3)$	$f_3(4)$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

  - Definiere eine neue Funktion $f{:}\mathbb{N}\to\mathbb{N}$ durch $f(x) = f_x(x)+1$

Diagonalbeweise I: Überabzählbarkeit von $\mathbb{N}\to\mathbb{N}$

- **Die Menge $\mathbb{N}\to\mathbb{N}$ "überabzählbar" unendlich**
  - Die Menge aller Funktionen über $\mathbb{N}$ kann nicht durchnumeriert werden
  - **Annahme**: $\mathbb{N}\to\mathbb{N}$ ist abzählbar
  - Dann können alle Funktionen über $\mathbb{N}$ in eine Tabelle eingetragen werden

	0	1	2	3	4	...
f_0	$f_0(0)+1$	$f_0(1)$	$f_0(2)$	$f_0(3)$	$f_0(4)$	...
f_1	$f_1(0)$	$f_1(1)+1$	$f_1(2)$	$f_1(3)$	$f_1(4)$	...
f_2	$f_2(0)$	$f_2(1)$	$f_2(2)+1$	$f_2(3)$	$f_2(4)$	...
f_3	$f_3(0)$	$f_3(1)$	$f_3(2)$	$f_3(3)+1$	$f_3(4)$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

  - Definiere eine neue Funktion $f{:}\mathbb{N}\to\mathbb{N}$ durch $f(x) = f_x(x)+1$

Diagonalbeweise I: Überabzählbarkeit von $\mathbb{N}\to\mathbb{N}$

- **Die Menge $\mathbb{N}\to\mathbb{N}$ "überabzählbar" unendlich**
  - Die Menge aller Funktionen über $\mathbb{N}$ kann nicht durchnumeriert werden
  - **Annahme**: $\mathbb{N}\to\mathbb{N}$ ist abzählbar
  - Dann können alle Funktionen über $\mathbb{N}$ in eine Tabelle eingetragen werden

	0	1	2	3	4	...
f_0	$f_0(0)+1$	$f_0(1)$	$f_0(2)$	$f_0(3)$	$f_0(4)$	...
f_1	$f_1(0)$	$f_1(1)+1$	$f_1(2)$	$f_1(3)$	$f_1(4)$	...
f_2	$f_2(0)$	$f_2(1)$	$f_2(2)+1$	$f_2(3)$	$f_2(4)$	...
f_3	$f_3(0)$	$f_3(1)$	$f_3(2)$	$f_3(3)+1$	$f_3(4)$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

  - Definiere eine neue Funktion $f{:}\mathbb{N}\to\mathbb{N}$ durch $f(x) = f_x(x)+1$
  - f ist offensichtlich total, kann aber in der Tabelle nicht vorkommen

Diagonalbeweise I: Überabzählbarkeit von $\mathbb{N}\to\mathbb{N}$

- **Die Menge $\mathbb{N}\to\mathbb{N}$ "überabzählbar" unendlich**
  - Die Menge aller Funktionen über $\mathbb{N}$ kann nicht durchnumeriert werden
  - **Annahme**: $\mathbb{N}\to\mathbb{N}$ ist abzählbar
  - Dann können alle Funktionen über $\mathbb{N}$ in eine Tabelle eingetragen werden

	0	1	2	3	4	...
f_0	$f_0(0)+1$	$f_0(1)$	$f_0(2)$	$f_0(3)$	$f_0(4)$	...
f_1	$f_1(0)$	$f_1(1)+1$	$f_1(2)$	$f_1(3)$	$f_1(4)$	...
f_2	$f_2(0)$	$f_2(1)$	$f_2(2)+1$	$f_2(3)$	$f_2(4)$	...
f_3	$f_3(0)$	$f_3(1)$	$f_3(2)$	$f_3(3)+1$	$f_3(4)$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

  - Definiere eine neue Funktion $f{:}\mathbb{N}\to\mathbb{N}$ durch $f(x) = f_x(x)+1$
  - f ist offensichtlich total, kann aber in der Tabelle nicht vorkommen
  - Ansonsten wäre $f{=}f_i$ für ein i und $f_i(i) = f(i) = f_i(i)+1$ ✓

Existenz unberechenbarer Funktionen

- **Abstraktes Argument: es gibt zu viele Funktionen**

Existenz unberechenbarer Funktionen

- **Abstraktes Argument: es gibt zu viele Funktionen**
  - Die Menge der berechenbaren Funktionen in $\mathbb{N} \to \mathbb{N}$ ist abzählbar
  - Die Menge $\mathbb{N} \to \mathbb{N}$ is nicht abzählbar

Existenz unberechenbarer Funktionen

- **Abstraktes Argument: es gibt zu viele Funktionen**
  - Die Menge der berechenbaren Funktionen in $\mathbb{N} \to \mathbb{N}$ ist abzählbar
  - Die Menge $\mathbb{N} \to \mathbb{N}$ is nicht abzählbar
  - Es gibt mehr Funktionen als es berechenbare Funktionen geben kann

Existenz unberechenbarer Funktionen

- **Abstraktes Argument: es gibt zu viele Funktionen**
  - Die Menge der berechenbaren Funktionen in $\mathbb{N}\to\mathbb{N}$ ist abzählbar
  - Die Menge $\mathbb{N}\to\mathbb{N}$ is nicht abzählbar
  - Es gibt mehr Funktionen als es berechenbare Funktionen geben kann
  - Es gibt nichtberechenbare Funktionen auf $\mathbb{N}\to\mathbb{N}$ ✓

Existenz unberechenbarer Funktionen

- **Abstraktes Argument: es gibt zu viele Funktionen**
  - Die Menge der berechenbaren Funktionen in $\mathbb{N}\to\mathbb{N}$ ist abzählbar
  - Die Menge $\mathbb{N}\to\mathbb{N}$ is nicht abzählbar
  - Es gibt mehr Funktionen als es berechenbare Funktionen geben kann
  - Es gibt nichtberechenbare Funktionen auf $\mathbb{N}\to\mathbb{N}$ ✓
- **Konkretes Argument: Angabe eines Beispiels**

Existenz unberechenbarer Funktionen

- **Abstraktes Argument: es gibt zu viele Funktionen**
  - Die Menge der berechenbaren Funktionen in $\mathbb{N}\to\mathbb{N}$ ist abzählbar
  - Die Menge $\mathbb{N}\to\mathbb{N}$ is nicht abzählbar
  - Es gibt mehr Funktionen als es berechenbare Funktionen geben kann
  - Es gibt nichtberechenbare Funktionen auf $\mathbb{N}\to\mathbb{N}$ ✓
- **Konkretes Argument: Angabe eines Beispiels**
  - Das Halteproblem $H = \{(i,n) \mid \varphi_i(n) \neq \bot\}$ ist unentscheidbar (Beweis folgt)

Existenz unberechenbarer Funktionen

- **Abstraktes Argument: es gibt zu viele Funktionen**
  - Die Menge der berechenbaren Funktionen in $\mathbb{N}\to\mathbb{N}$ ist abzählbar
  - Die Menge $\mathbb{N}\to\mathbb{N}$ is nicht abzählbar
  - Es gibt mehr Funktionen als es berechenbare Funktionen geben kann
  - Es gibt nichtberechenbare Funktionen auf $\mathbb{N}\to\mathbb{N}$ ✓
- **Konkretes Argument: Angabe eines Beispiels**
  - Das Halteproblem $H = \{(i,n) \mid \varphi_i(n) \neq \bot\}$ ist unentscheidbar (Beweis folgt)
  - Die charakteristische Funktion $\chi_H : \mathbb{N}\to\mathbb{N}$ ist nicht berechenbar ✓

Existenz unberechenbarer Funktionen

- **Abstraktes Argument: es gibt zu viele Funktionen**
  - Die Menge der berechenbaren Funktionen in $\mathbb{N}\to\mathbb{N}$ ist abzählbar
  - Die Menge $\mathbb{N}\to\mathbb{N}$ is nicht abzählbar
  - Es gibt mehr Funktionen als es berechenbare Funktionen geben kann
  - Es gibt nichtberechenbare Funktionen auf $\mathbb{N}\to\mathbb{N}$ ✓
- **Konkretes Argument: Angabe eines Beispiels**
  - Das Halteproblem $H = \{(i,n) \mid \varphi_i(n) \neq \bot\}$ ist unentscheidbar (Beweis folgt)
  - Die charakteristische Funktion $\chi_H : \mathbb{N}\to\mathbb{N}$ ist nicht berechenbar ✓
  - Weitere konkrete Beispiele folgen

Diagonalbeweise II: Unentscheidbarkeit des Halteproblems

Annahme: $\boldsymbol{H = \{(i,n) \mid \varphi_i(n) \neq \perp\}}$ **ist entscheidbar**

Diagonalbeweise II: Unentscheidbarkeit des Halteproblems

Annahme: $H = \{(i,n) \mid \varphi_i(n) \neq \perp\}$ ist entscheidbar

– Dann ist $\chi_H : \mathbb{N} \to \mathbb{N}$ berechenbar, wobei $\chi_H(i,n) = \begin{cases} 1 & \text{wenn } \varphi_i(n) \text{ hält} \\ 0 & \text{sonst} \end{cases}$

Diagonalbeweise II: Unentscheidbarkeit des Halteproblems

Annahme: $H = \{(i,n) \mid \varphi_i(n) \neq \perp\}$ ist entscheidbar

– Dann ist $\chi_H : \mathbb{N} \to \mathbb{N}$ berechenbar, wobei $\chi_H(i,n) = \begin{cases} 1 & \text{wenn } \varphi_i(n) \text{ hält} \\ 0 & \text{sonst} \end{cases}$

	0	1	2	3	4	...
φ_0	$\times$	$\times$	$\times$	$\perp$	$\times$	...
φ_1	$\perp$	$\perp$	$\times$	$\times$	$\times$	...
φ_2	$\times$	$\times$	$\perp$	$\times$	$\times$	...
φ_3	$\perp$	$\times$	$\perp$	$\times$	$\perp$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

Diagonalbeweise II: Unentscheidbarkeit des Halteproblems

Annahme: $H = \{(i,n) \mid \varphi_i(n) \neq \perp\}$ ist entscheidbar

– Dann ist $\chi_H : \mathbb{N} \to \mathbb{N}$ berechenbar, wobei $\chi_H(i,n) = \begin{cases} 1 & \text{wenn } \varphi_i(n) \text{ hält} \\ 0 & \text{sonst} \end{cases}$

– Definiere eine neue Funktion $f : \mathbb{N} \to \mathbb{N}$ durch

$$f(n) := \begin{cases} 0 & \text{wenn } \varphi_n(n) \text{ nicht hält} \\ \perp & \text{sonst} \end{cases}$$

	0	1	2	3	4	...
φ_0	$\times$	$\times$	$\times$	$\perp$	$\times$	...
φ_1	$\perp$	$\perp$	$\times$	$\times$	$\times$	...
φ_2	$\times$	$\times$	$\perp$	$\times$	$\times$	...
φ_3	$\perp$	$\times$	$\perp$	$\times$	$\perp$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

Diagonalbeweise II: Unentscheidbarkeit des Halteproblems

Annahme: $H = \{(i,n) \mid \varphi_i(n) \neq \perp\}$ ist entscheidbar

– Dann ist $\chi_H : \mathbb{N} \to \mathbb{N}$ berechenbar, wobei $\chi_H(i,n) = \begin{cases} 1 & \text{wenn } \varphi_i(n) \text{ hält} \\ 0 & \text{sonst} \end{cases}$

– Definiere eine neue Funktion $f : \mathbb{N} \to \mathbb{N}$ durch

$$f(n) := \begin{cases} 0 & \text{wenn } \varphi_n(n) \text{ nicht hält} \\ \perp & \text{sonst} \end{cases}$$

	0	1	2	3	4	...
φ_0	$\perp$	$\times$	$\times$	$\perp$	$\times$	...
φ_1	$\perp$	$\perp$	$\times$	$\times$	$\times$	...
φ_2	$\times$	$\times$	$\perp$	$\times$	$\times$	...
φ_3	$\perp$	$\times$	$\perp$	$\times$	$\perp$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

Diagonalbeweise II: Unentscheidbarkeit des Halteproblems

Annahme: $H = \{(i,n) \mid \varphi_i(n) \neq \bot\}$ ist entscheidbar

– Dann ist $\chi_H : \mathbb{N} \to \mathbb{N}$ berechenbar, wobei $\chi_H(i,n) = \begin{cases} 1 & \text{wenn } \varphi_i(n) \text{ hält} \\ 0 & \text{sonst} \end{cases}$

– Definiere eine neue Funktion $f : \mathbb{N} \to \mathbb{N}$ durch

$$f(n) := \begin{cases} 0 & \text{wenn } \varphi_n(n) \text{ nicht hält} \\ \bot & \text{sonst} \end{cases}$$

	0	1	2	3	4	...
φ_0	$\bot$	$\times$	$\times$	$\bot$	$\times$	...
φ_1	$\bot$	$\times$	$\times$	$\times$	$\times$	...
φ_2	$\times$	$\times$	$\bot$	$\times$	$\times$	...
φ_3	$\bot$	$\times$	$\bot$	$\times$	$\bot$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

Diagonalbeweise II: Unentscheidbarkeit des Halteproblems

Annahme: $H = \{(i,n) \mid \varphi_i(n) \neq \perp\}$ ist entscheidbar

– Dann ist $\chi_H : \mathbb{N} \to \mathbb{N}$ berechenbar, wobei $\chi_H(i,n) = \begin{cases} 1 & \text{wenn } \varphi_i(n) \text{ hält} \\ 0 & \text{sonst} \end{cases}$

– Definiere eine neue Funktion $f : \mathbb{N} \to \mathbb{N}$ durch

$$f(n) := \begin{cases} 0 & \text{wenn } \varphi_n(n) \text{ nicht hält} \\ \perp & \text{sonst} \end{cases}$$

	0	1	2	3	4	...
φ_0	$\perp$	$\times$	$\times$	$\perp$	$\times$	...
φ_1	$\perp$	$\times$	$\times$	$\times$	$\times$	...
φ_2	$\times$	$\times$	$\times$	$\times$	$\times$	...
φ_3	$\perp$	$\times$	$\perp$	$\times$	$\perp$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

Diagonalbeweise II: Unentscheidbarkeit des Halteproblems

Annahme: $H = \{(i,n) \mid \varphi_i(n) \neq \perp\}$ ist entscheidbar

– Dann ist $\chi_H : \mathbb{N} \to \mathbb{N}$ berechenbar, wobei $\chi_H(i,n) = \begin{cases} 1 & \text{wenn } \varphi_i(n) \text{ hält} \\ 0 & \text{sonst} \end{cases}$

– Definiere eine neue Funktion $f : \mathbb{N} \to \mathbb{N}$ durch

$$f(n) := \begin{cases} 0 & \text{wenn } \varphi_n(n) \text{ nicht hält} \\ \perp & \text{sonst} \end{cases}$$

	0	1	2	3	4	...
φ_0	$\perp$	$\times$	$\times$	$\perp$	$\times$	...
φ_1	$\perp$	$\times$	$\times$	$\times$	$\times$	...
φ_2	$\times$	$\times$	$\times$	$\times$	$\times$	...
φ_3	$\perp$	$\times$	$\perp$	$\perp$	$\perp$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

Diagonalbeweise II: Unentscheidbarkeit des Halteproblems

Annahme: $H = \{(i,n) \mid \varphi_i(n) \neq \perp\}$ ist entscheidbar

– Dann ist $\chi_H : \mathbb{N} \to \mathbb{N}$ berechenbar, wobei $\chi_H(i,n) = \begin{cases} 1 & \text{wenn } \varphi_i(n) \text{ hält} \\ 0 & \text{sonst} \end{cases}$

– Definiere eine neue Funktion $f : \mathbb{N} \to \mathbb{N}$ durch

$$f(n) := \begin{cases} 0 & \text{wenn } \varphi_n(n) \text{ nicht hält} \\ \perp & \text{sonst} \end{cases}$$

	0	1	2	3	4	...
φ_0	$\perp$	$\times$	$\times$	$\perp$	$\times$	...
φ_1	$\perp$	$\times$	$\times$	$\times$	$\times$	...
φ_2	$\times$	$\times$	$\times$	$\times$	$\times$	...
φ_3	$\perp$	$\times$	$\perp$	$\perp$	$\perp$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

– Dann ist f berechenbar, denn $f(n) = \mu_z[\chi_H(n,n) = 0]$

Diagonalbeweise II: Unentscheidbarkeit des Halteproblems

Annahme: $H = \{(i,n) \mid \varphi_i(n) \neq \perp\}$ ist entscheidbar

– Dann ist $\chi_H : \mathbb{N} \to \mathbb{N}$ berechenbar, wobei $\chi_H(i,n) = \begin{cases} 1 & \text{wenn } \varphi_i(n) \text{ hält} \\ 0 & \text{sonst} \end{cases}$

– Definiere eine neue Funktion $f : \mathbb{N} \to \mathbb{N}$ durch

$$f(n) := \begin{cases} 0 & \text{wenn } \varphi_n(n) \text{ nicht hält} \\ \perp & \text{sonst} \end{cases}$$

	0	1	2	3	4	...
φ_0	$\perp$	$\times$	$\times$	$\perp$	$\times$	...
φ_1	$\perp$	$\times$	$\times$	$\times$	$\times$	...
φ_2	$\times$	$\times$	$\times$	$\times$	$\times$	...
φ_3	$\perp$	$\times$	$\perp$	$\perp$	$\perp$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

– Dann ist f berechenbar, denn $f(n) = \mu_z[\chi_H(n,n) = 0]$

– Also gibt es ein i mit $f = \varphi_i$

Diagonalbeweise II: Unentscheidbarkeit des Halteproblems

Annahme: $H = \{(i,n) \mid \varphi_i(n) \neq \perp\}$ ist entscheidbar

– Dann ist $\chi_H : \mathbb{N} \to \mathbb{N}$ berechenbar, wobei $\chi_H(i,n) = \begin{cases} 1 & \text{wenn } \varphi_i(n) \text{ hält} \\ 0 & \text{sonst} \end{cases}$

– Definiere eine neue Funktion $f : \mathbb{N} \to \mathbb{N}$ durch

$$f(n) := \begin{cases} 0 & \text{wenn } \varphi_n(n) \text{ nicht hält} \\ \perp & \text{sonst} \end{cases}$$

	0	1	2	3	4	...
φ_0	$\perp$	$\times$	$\times$	$\perp$	$\times$	...
φ_1	$\perp$	$\times$	$\times$	$\times$	$\times$	...
φ_2	$\times$	$\times$	$\times$	$\times$	$\times$	...
φ_3	$\perp$	$\times$	$\perp$	$\perp$	$\perp$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

– Dann ist f berechenbar, denn $f(n) = \mu_z[\chi_H(n,n) = 0]$

– Also gibt es ein i mit $f = \varphi_i$

– Aber für dieses i gilt: $\varphi_i(i)$ hält $\Leftrightarrow$ $f(i)$ hält $\Leftrightarrow$ $\varphi_i(i)$ hält nicht

Diagonalbeweise II: Unentscheidbarkeit des Halteproblems

Annahme: $H = \{(i,n) \mid \varphi_i(n) \neq \perp\}$ ist entscheidbar

– Dann ist $\chi_H : \mathbb{N} \to \mathbb{N}$ berechenbar, wobei $\chi_H(i,n) = \begin{cases} 1 & \text{wenn } \varphi_i(n) \text{ hält} \\ 0 & \text{sonst} \end{cases}$

– Definiere eine neue Funktion $f : \mathbb{N} \to \mathbb{N}$ durch

$$f(n) := \begin{cases} 0 & \text{wenn } \varphi_n(n) \text{ nicht hält} \\ \perp & \text{sonst} \end{cases}$$

	0	1	2	3	4	...
φ_0	$\perp$	$\times$	$\times$	$\perp$	$\times$	...
φ_1	$\perp$	$\times$	$\times$	$\times$	$\times$	...
φ_2	$\times$	$\times$	$\times$	$\times$	$\times$	...
φ_3	$\perp$	$\times$	$\perp$	$\perp$	$\perp$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

– Dann ist f berechenbar, denn $f(n) = \mu_z[\chi_H(n,n) = 0]$

– Also gibt es ein i mit $f = \varphi_i$

– Aber für dieses i gilt: $\varphi_i(i)$ hält $\Leftrightarrow$ $f(i)$ hält $\Leftrightarrow$ $\varphi_i(i)$ hält nicht

– Dies ist ein Widerspruch, also ist die Annahme "H entscheidbar" falsch ✓

Diagonalbeweise II: Unentscheidbarkeit des Halteproblems

Annahme: $H = \{(i,n) \mid \varphi_i(n) \neq \bot\}$ ist entscheidbar

– Dann ist $\chi_H : \mathbb{N} \to \mathbb{N}$ berechenbar, wobei $\chi_H(i,n) = \begin{cases} 1 & \text{wenn } \varphi_i(n) \text{ hält} \\ 0 & \text{sonst} \end{cases}$

– Definiere eine neue Funktion $f : \mathbb{N} \to \mathbb{N}$ durch

$$f(n) := \begin{cases} 0 & \text{wenn } \varphi_n(n) \text{ nicht hält} \\ \bot & \text{sonst} \end{cases}$$

	0	1	2	3	4	...
φ_0	$\bot$	$\times$	$\times$	$\bot$	$\times$	...
φ_1	$\bot$	$\times$	$\times$	$\times$	$\times$	...
φ_2	$\times$	$\times$	$\times$	$\times$	$\times$	...
φ_3	$\bot$	$\times$	$\bot$	$\bot$	$\bot$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

– Dann ist f berechenbar, denn $f(n) = \mu_z[\chi_H(n,n) = 0]$

– Also gibt es ein i mit $f = \varphi_i$

– Aber für dieses i gilt: $\varphi_i(i)$ hält $\Leftrightarrow$ $f(i)$ hält $\Leftrightarrow$ $\varphi_i(i)$ hält nicht

– Dies ist ein Widerspruch, also ist die Annahme "H entscheidbar" falsch ✓

Terminierung von Programmen ist nicht testbar

Diagonalbeweise III: Total berechenbare Funktionen sind nicht aufzählbar

Annahme: $\mathcal{R}_\varphi = \{i \mid \varphi_i \text{ total}\}$ **ist aufzählbar**

Diagonalbeweise III: Total berechenbare Funktionen sind nicht aufzählbar

Annahme: $\mathcal{R}_\varphi = \{i \mid \varphi_i \text{ total}\}$ ist aufzählbar

- Dann gibt es eine berechenbare totale Funktion f mit $\text{range}(f) = \mathcal{R}_\varphi$

Diagonalbeweise III: Total berechenbare Funktionen sind nicht aufzählbar

Annahme: $\mathcal{R}_\varphi = \{i \mid \varphi_i \text{ total}\}$ ist aufzählbar

- Dann gibt es eine berechenbare totale Funktion f mit $\text{range}(f) = \mathcal{R}_\varphi$
- Dann lassen sich alle Funktionen aus $\mathcal{R}$ wie folgt in eine Tabelle eintragen

	0	1	2	3	4	...
$\varphi_{f(0)}$	$\varphi_{f(0)}(0)$	$\varphi_{f(0)}(1)$	$\varphi_{f(0)}(2)$	$\varphi_{f(0)}(3)$	$\varphi_{f(0)}(4)$	...
$\varphi_{f(1)}$	$\varphi_{f(1)}(0)$	$\varphi_{f(1)}(1)$	$\varphi_{f(1)}(2)$	$\varphi_{f(1)}(3)$	$\varphi_{f(1)}(4)$	...
$\varphi_{f(2)}$	$\varphi_{f(2)}(0)$	$\varphi_{f(2)}(1)$	$\varphi_{f(2)}(2)$	$\varphi_{f(2)}(3)$	$\varphi_{f(2)}(4)$	...
$\varphi_{f(3)}$	$\varphi_{f(3)}(0)$	$\varphi_{f(3)}(1)$	$\varphi_{f(3)}(2)$	$\varphi_{f(3)}(3)$	$\varphi_{f(3)}(4)$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

Diagonalbeweise III: Total berechenbare Funktionen sind nicht aufzählbar

Annahme: $\mathcal{R}_\varphi = \{i \mid \varphi_i \text{ total}\}$ ist aufzählbar

- Dann gibt es eine berechenbare totale Funktion f mit $\text{range}(f) = \mathcal{R}_\varphi$
- Dann lassen sich alle Funktionen aus $\mathcal{R}$ wie folgt in eine Tabelle eintragen

	0	1	2	3	4	...
$\varphi_{f(0)}$	$\varphi_{f(0)}(0)+1$	$\varphi_{f(0)}(1)$	$\varphi_{f(0)}(2)$	$\varphi_{f(0)}(3)$	$\varphi_{f(0)}(4)$	...
$\varphi_{f(1)}$	$\varphi_{f(1)}(0)$	$\varphi_{f(1)}(1)$	$\varphi_{f(1)}(2)$	$\varphi_{f(1)}(3)$	$\varphi_{f(1)}(4)$	...
$\varphi_{f(2)}$	$\varphi_{f(2)}(0)$	$\varphi_{f(2)}(1)$	$\varphi_{f(2)}(2)$	$\varphi_{f(2)}(3)$	$\varphi_{f(2)}(4)$	...
$\varphi_{f(3)}$	$\varphi_{f(3)}(0)$	$\varphi_{f(3)}(1)$	$\varphi_{f(3)}(2)$	$\varphi_{f(3)}(3)$	$\varphi_{f(3)}(4)$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

- Definiere eine neue Funktion $h{:}\mathbb{N}\to\mathbb{N}$ durch $h(n) = \varphi_{f(n)}(n)+1$

Diagonalbeweise III: Total berechenbare Funktionen sind nicht aufzählbar

Annahme: $\mathcal{R}_\varphi = \{i \mid \varphi_i \text{ total}\}$ ist aufzählbar

- Dann gibt es eine berechenbare totale Funktion f mit $\text{range}(f) = \mathcal{R}_\varphi$
- Dann lassen sich alle Funktionen aus $\mathcal{R}$ wie folgt in eine Tabelle eintragen

	0	1	2	3	4	...
$\varphi_{f(0)}$	$\varphi_{f(0)}(0)+1$	$\varphi_{f(0)}(1)$	$\varphi_{f(0)}(2)$	$\varphi_{f(0)}(3)$	$\varphi_{f(0)}(4)$	...
$\varphi_{f(1)}$	$\varphi_{f(1)}(0)$	$\varphi_{f(1)}(1)+1$	$\varphi_{f(1)}(2)$	$\varphi_{f(1)}(3)$	$\varphi_{f(1)}(4)$	...
$\varphi_{f(2)}$	$\varphi_{f(2)}(0)$	$\varphi_{f(2)}(1)$	$\varphi_{f(2)}(2)$	$\varphi_{f(2)}(3)$	$\varphi_{f(2)}(4)$	...
$\varphi_{f(3)}$	$\varphi_{f(3)}(0)$	$\varphi_{f(3)}(1)$	$\varphi_{f(3)}(2)$	$\varphi_{f(3)}(3)$	$\varphi_{f(3)}(4)$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

- Definiere eine neue Funktion $h{:}\mathbb{N}\to\mathbb{N}$ durch $h(n) = \varphi_{f(n)}(n)+1$

Diagonalbeweise III: Total berechenbare Funktionen sind nicht aufzählbar

Annahme: $\mathcal{R}_\varphi = \{i \mid \varphi_i \text{ total}\}$ ist aufzählbar

- Dann gibt es eine berechenbare totale Funktion f mit $\text{range}(f) = \mathcal{R}_\varphi$
- Dann lassen sich alle Funktionen aus $\mathcal{R}$ wie folgt in eine Tabelle eintragen

	0	1	2	3	4	...
$\varphi_{f(0)}$	$\varphi_{f(0)}(0)+1$	$\varphi_{f(0)}(1)$	$\varphi_{f(0)}(2)$	$\varphi_{f(0)}(3)$	$\varphi_{f(0)}(4)$	...
$\varphi_{f(1)}$	$\varphi_{f(1)}(0)$	$\varphi_{f(1)}(1)+1$	$\varphi_{f(1)}(2)$	$\varphi_{f(1)}(3)$	$\varphi_{f(1)}(4)$	...
$\varphi_{f(2)}$	$\varphi_{f(2)}(0)$	$\varphi_{f(2)}(1)$	$\varphi_{f(2)}(2)+1$	$\varphi_{f(2)}(3)$	$\varphi_{f(2)}(4)$	...
$\varphi_{f(3)}$	$\varphi_{f(3)}(0)$	$\varphi_{f(3)}(1)$	$\varphi_{f(3)}(2)$	$\varphi_{f(3)}(3)$	$\varphi_{f(3)}(4)$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

- Definiere eine neue Funktion $h{:}\mathbb{N}\to\mathbb{N}$ durch $h(n) = \varphi_{f(n)}(n)+1$

Diagonalbeweise III: Total berechenbare Funktionen sind nicht aufzählbar

Annahme: $\mathcal{R}_\varphi = \{i \mid \varphi_i \text{ total}\}$ ist aufzählbar

- Dann gibt es eine berechenbare totale Funktion f mit $\text{range}(f) = \mathcal{R}_\varphi$
- Dann lassen sich alle Funktionen aus $\mathcal{R}$ wie folgt in eine Tabelle eintragen

	0	1	2	3	4	...
$\varphi_{f(0)}$	$\varphi_{f(0)}(0)+1$	$\varphi_{f(0)}(1)$	$\varphi_{f(0)}(2)$	$\varphi_{f(0)}(3)$	$\varphi_{f(0)}(4)$	...
$\varphi_{f(1)}$	$\varphi_{f(1)}(0)$	$\varphi_{f(1)}(1)+1$	$\varphi_{f(1)}(2)$	$\varphi_{f(1)}(3)$	$\varphi_{f(1)}(4)$	...
$\varphi_{f(2)}$	$\varphi_{f(2)}(0)$	$\varphi_{f(2)}(1)$	$\varphi_{f(2)}(2)+1$	$\varphi_{f(2)}(3)$	$\varphi_{f(2)}(4)$	...
$\varphi_{f(3)}$	$\varphi_{f(3)}(0)$	$\varphi_{f(3)}(1)$	$\varphi_{f(3)}(2)$	$\varphi_{f(3)}(3)+1$	$\varphi_{f(3)}(4)$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

- Definiere eine neue Funktion $h{:}\mathbb{N}\to\mathbb{N}$ durch $h(n) = \varphi_{f(n)}(n)+1$

Diagonalbeweise III: Total berechenbare Funktionen sind nicht aufzählbar

Annahme: $\mathcal{R}_\varphi = \{i \mid \varphi_i \text{ total}\}$ ist aufzählbar

- Dann gibt es eine berechenbare totale Funktion f mit $\text{range}(f) = \mathcal{R}_\varphi$
- Dann lassen sich alle Funktionen aus $\mathcal{R}$ wie folgt in eine Tabelle eintragen

	0	1	2	3	4	...
$\varphi_{f(0)}$	$\varphi_{f(0)}(0)+1$	$\varphi_{f(0)}(1)$	$\varphi_{f(0)}(2)$	$\varphi_{f(0)}(3)$	$\varphi_{f(0)}(4)$	...
$\varphi_{f(1)}$	$\varphi_{f(1)}(0)$	$\varphi_{f(1)}(1)+1$	$\varphi_{f(1)}(2)$	$\varphi_{f(1)}(3)$	$\varphi_{f(1)}(4)$	...
$\varphi_{f(2)}$	$\varphi_{f(2)}(0)$	$\varphi_{f(2)}(1)$	$\varphi_{f(2)}(2)+1$	$\varphi_{f(2)}(3)$	$\varphi_{f(2)}(4)$	...
$\varphi_{f(3)}$	$\varphi_{f(3)}(0)$	$\varphi_{f(3)}(1)$	$\varphi_{f(3)}(2)$	$\varphi_{f(3)}(3)+1$	$\varphi_{f(3)}(4)$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

- Definiere eine neue Funktion $h{:}\mathbb{N}\to\mathbb{N}$ durch $h(n) = \varphi_{f(n)}(n)+1$
- h ist offensichtlich total und berechenbar, denn $h(n) = u(f(n), n)+1$

Diagonalbeweise III: Total berechenbare Funktionen sind nicht aufzählbar

Annahme: $\mathcal{R}_\varphi = \{i \mid \varphi_i \text{ total}\}$ ist aufzählbar

- Dann gibt es eine berechenbare totale Funktion f mit $\text{range}(f) = \mathcal{R}_\varphi$
- Dann lassen sich alle Funktionen aus $\mathcal{R}$ wie folgt in eine Tabelle eintragen

	0	1	2	3	4	...
$\varphi_{f(0)}$	$\varphi_{f(0)}(0)+1$	$\varphi_{f(0)}(1)$	$\varphi_{f(0)}(2)$	$\varphi_{f(0)}(3)$	$\varphi_{f(0)}(4)$	...
$\varphi_{f(1)}$	$\varphi_{f(1)}(0)$	$\varphi_{f(1)}(1)+1$	$\varphi_{f(1)}(2)$	$\varphi_{f(1)}(3)$	$\varphi_{f(1)}(4)$	...
$\varphi_{f(2)}$	$\varphi_{f(2)}(0)$	$\varphi_{f(2)}(1)$	$\varphi_{f(2)}(2)+1$	$\varphi_{f(2)}(3)$	$\varphi_{f(2)}(4)$	...
$\varphi_{f(3)}$	$\varphi_{f(3)}(0)$	$\varphi_{f(3)}(1)$	$\varphi_{f(3)}(2)$	$\varphi_{f(3)}(3)+1$	$\varphi_{f(3)}(4)$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

- Definiere eine neue Funktion $h{:}\mathbb{N}\to\mathbb{N}$ durch $h(n) = \varphi_{f(n)}(n)+1$
- h ist offensichtlich total und berechenbar, denn $h(n) = u(f(n), n)+1$
- Also gibt es ein $i \in \mathcal{R}_\varphi$ mit $h = \varphi_i$ und damit ein $j \in \mathbb{N}$ mit $i = f(j)$

Diagonalbeweise III: Total berechenbare Funktionen sind nicht aufzählbar

Annahme: $\mathcal{R}_\varphi = \{i \mid \varphi_i \text{ total}\}$ ist aufzählbar

- Dann gibt es eine berechenbare totale Funktion f mit $\text{range}(f) = \mathcal{R}_\varphi$
- Dann lassen sich alle Funktionen aus $\mathcal{R}$ wie folgt in eine Tabelle eintragen

	0	1	2	3	4	...
$\varphi_{f(0)}$	$\varphi_{f(0)}(0)+1$	$\varphi_{f(0)}(1)$	$\varphi_{f(0)}(2)$	$\varphi_{f(0)}(3)$	$\varphi_{f(0)}(4)$	...
$\varphi_{f(1)}$	$\varphi_{f(1)}(0)$	$\varphi_{f(1)}(1)+1$	$\varphi_{f(1)}(2)$	$\varphi_{f(1)}(3)$	$\varphi_{f(1)}(4)$	...
$\varphi_{f(2)}$	$\varphi_{f(2)}(0)$	$\varphi_{f(2)}(1)$	$\varphi_{f(2)}(2)+1$	$\varphi_{f(2)}(3)$	$\varphi_{f(2)}(4)$	...
$\varphi_{f(3)}$	$\varphi_{f(3)}(0)$	$\varphi_{f(3)}(1)$	$\varphi_{f(3)}(2)$	$\varphi_{f(3)}(3)+1$	$\varphi_{f(3)}(4)$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...

- Definiere eine neue Funktion $h{:}\mathbb{N}\to\mathbb{N}$ durch $h(n) = \varphi_{f(n)}(n)+1$
- h ist offensichtlich total und berechenbar, denn $h(n) = u(f(n), n)+1$
- Also gibt es ein $i \in \mathcal{R}_\varphi$ mit $h = \varphi_i$ und damit ein $j \in \mathbb{N}$ mit $i = f(j)$
- Für dieses j gilt $\varphi_{f(j)}(j) = h(j) = \varphi_{f(j)}(j)+1$ ✓

Andere unlösbare Probleme mit Diagonalbeweisen

- **Selbstanwendbarkeitsproblem:**
  - $S = \{i \mid \varphi_i(i) \neq \perp\}$ unentscheidbar, aber aufzählbar

Andere unlösbare Probleme mit Diagonalbeweisen

- **Selbstanwendbarkeitsproblem:**
  - $S = \{i \mid \varphi_i(i) \neq \perp\}$ unentscheidbar, aber aufzählbar
- **Entscheidungsproblem:**
  - $E = \{(i,j) \mid \varphi_i(j) = 1\}$ unentscheidbar, aber aufzählbar

Andere unlösbare Probleme mit Diagonalbeweisen

- **Selbstanwendbarkeitsproblem:**
  - $S = \{i \mid \varphi_i(i) \neq \bot\}$ unentscheidbar, aber aufzählbar
- **Entscheidungsproblem:**
  - $E = \{(i,j) \mid \varphi_i(j) = 1\}$ unentscheidbar, aber aufzählbar
- **Monotone Funktionen:**
  - $MON = \{i \mid \forall k.\ \varphi_i(k) < \varphi_i(k+1)\}$ nicht aufzählbar

Andere unlösbare Probleme mit Diagonalbeweisen

- **Selbstanwendbarkeitsproblem:**
  - $S = \{i \mid \varphi_i(i) \neq \bot\}$ unentscheidbar, aber aufzählbar
- **Entscheidungsproblem:**
  - $E = \{(i,j) \mid \varphi_i(j) = 1\}$ unentscheidbar, aber aufzählbar
- **Monotone Funktionen:**
  - $MON = \{i \mid \forall k.\ \varphi_i(k) < \varphi_i(k+1)\}$ nicht aufzählbar
- **Entscheidungsfunktionen:**
  - $EF = \{i \mid \forall j.\ \varphi_i(j) \in \{0,1\}\}$ nicht aufzählbar

Monotoniearguments

- **Ziel**
  - Zeige, daß eine Funktion f eine Eigenschaft P nicht besitzt
  - z.B. primitiv rekursiv, berechenbar, maximale Komplexität (Rechenzeit)

Monotoniearguments

- **Ziel**
  - Zeige, daß eine Funktion f eine Eigenschaft P nicht besitzt
  - z.B. primitiv rekursiv, berechenbar, maximale Komplexität (Rechenzeit)
- **Methodik**
  - Zeige daß f stärker wächst als jede Funktion mit Eigenschaft P

Monotoniearguments

- **Ziel**
  - Zeige, daß eine Funktion f eine Eigenschaft P nicht besitzt
  - z.B. primitiv rekursiv, berechenbar, maximale Komplexität (Rechenzeit)
- **Methodik**
  - Zeige daß f stärker wächst als jede Funktion mit Eigenschaft P
    - Induktive Analyse des Wachstumsverhaltens von f
    - Analyse des maximalen Wachstums von Funktionen mit Eigenschaft P

Monotoniearguments

- **Ziel**
  - Zeige, daß eine Funktion f eine Eigenschaft P nicht besitzt
  - z.B. primitiv rekursiv, berechenbar, maximale Komplexität (Rechenzeit)
- **Methodik**
  - Zeige daß f stärker wächst als jede Funktion mit Eigenschaft P
    - Induktive Analyse des Wachstumsverhaltens von f
    - Analyse des maximalen Wachstums von Funktionen mit Eigenschaft P
  - f kann also nicht selbst Eigenschaft P besitzen

Monotoniearguments

- **Ziel**
  - Zeige, daß eine Funktion f eine Eigenschaft P nicht besitzt
  - z.B. primitiv rekursiv, berechenbar, maximale Komplexität (Rechenzeit)
- **Methodik**
  - Zeige daß f stärker wächst als jede Funktion mit Eigenschaft P
    - Induktive Analyse des Wachstumsverhaltens von f
    - Analyse des maximalen Wachstums von Funktionen mit Eigenschaft P
  - f kann also nicht selbst Eigenschaft P besitzen
- **Beispiele**
  - Die Ackermann Funktion ist nicht primitiv-rekursiv
  - Die Busy-Beaver Funktion ist nicht berechenbar folgt
  - Komplexitätsanalysen

Das Busy-Beaver Problem

Biber stauen Bäche, indem sie Holzstücke in den Bach tragen. Fleißige Biber tragen mehr Holzstücke zusammen als faule. Größere Biber können mehr leisten als kleine.

Die Busy-Beaver Funktion liefert die Länge der längsten ununterbrochenen Staumauer, die ein Biber zusammentragen kann, ohne daß schon eine Teilmauer vorhanden war.

Das Busy-Beaver Problem

Biber stauen Bäche, indem sie Holzstücke in den Bach tragen. Fleißige Biber tragen mehr Holzstücke zusammen als faule. Größere Biber können mehr leisten als kleine.

Die Busy-Beaver Funktion liefert die Länge der längsten ununterbrochenen Staumauer, die ein Biber zusammentragen kann, ohne daß schon eine Teilmauer vorhanden war.

- **Beschreibe Biber durch Turingmaschinen**
  - Holzstücke werden durch das Symbol | beschrieben
  - $\tau = (\{1..n\}, \{|\}, \{|, b\}, \delta, 1, b)$ heißt Busy-Beaver TM der Größe n
  - $\mathrm{BBT}(n)$ sei die Menge aller Busy-Beaver Turingmaschinen der Größe n

Das Busy-Beaver Problem

Biber stauen Bäche, indem sie Holzstücke in den Bach tragen. Fleißige Biber tragen mehr Holzstücke zusammen als faule. Größere Biber können mehr leisten als kleine.

Die Busy-Beaver Funktion liefert die Länge der längsten ununterbrochenen Staumauer, die ein Biber zusammentragen kann, ohne daß schon eine Teilmauer vorhanden war.

- **Beschreibe Biber durch Turingmaschinen**
  - Holzstücke werden durch das Symbol | beschrieben
  - $\tau = (\{1..n\}, \{|\}, \{|, b\}, \delta, 1, b)$ heißt Busy-Beaver TM der Größe n
  - $\mathrm{BBT}(n)$ sei die Menge aller Busy-Beaver Turingmaschinen der Größe n
- **Beschreibe Produktivität von Bibern**
  - $\text{Produktivität}(\tau) = \begin{cases} n & \text{wenn } h_\tau(\epsilon) = |^n \\ 0 & \text{wenn } \tau \text{ bei Eingabe } \epsilon \text{ nicht hält} \end{cases}$

Das Busy-Beaver Problem

Biber stauen Bäche, indem sie Holzstücke in den Bach tragen. Fleißige Biber tragen mehr Holzstücke zusammen als faule. Größere Biber können mehr leisten als kleine.

Die Busy-Beaver Funktion liefert die Länge der längsten ununterbrochenen Staumauer, die ein Biber zusammentragen kann, ohne daß schon eine Teilmauer vorhanden war.

- **Beschreibe Biber durch Turingmaschinen**
  - Holzstücke werden durch das Symbol | beschrieben
  - $\tau = (\{1..n\}, \{|\}, \{|, b\}, \delta, 1, b)$ heißt Busy-Beaver TM der Größe n
  - $\mathrm{BBT}(n)$ sei die Menge aller Busy-Beaver Turingmaschinen der Größe n
- **Beschreibe Produktivität von Bibern**
  - $\mathrm{Produktivität}(\tau) = \begin{cases} n & \text{wenn } h_\tau(\epsilon) = |^n \\ 0 & \text{wenn } \tau \text{ bei Eingabe } \epsilon \text{ nicht hält} \end{cases}$
- **Beschreibe maximal mögliche Leistung von Bibern**
  - $BB(n) = \max\{\mathrm{Produktivität}(\tau) \mid \tau \in \mathrm{BBT}(n)\}$

Das Busy-Beaver Problem

Biber stauen Bäche, indem sie Holzstücke in den Bach tragen. Fleißige Biber tragen mehr Holzstücke zusammen als faule. Größere Biber können mehr leisten als kleine.
Die Busy-Beaver Funktion liefert die Länge der längsten ununterbrochenen Staumauer, die ein Biber zusammentragen kann, ohne daß schon eine Teilmauer vorhanden war.

- **Beschreibe Biber durch Turingmaschinen**
  - Holzstücke werden durch das Symbol | beschrieben
  - $\tau = (\{1..n\}, \{|\}, \{|, b\}, \delta, 1, b)$ heißt Busy-Beaver TM der Größe n
  - $\mathrm{BBT}(n)$ sei die Menge aller Busy-Beaver Turingmaschinen der Größe n
- **Beschreibe Produktivität von Bibern**
  - $\mathrm{Produktivität}(\tau) = \begin{cases} n & \text{wenn } h_\tau(\epsilon) = |^n \\ 0 & \text{wenn } \tau \text{ bei Eingabe } \epsilon \text{ nicht hält} \end{cases}$
- **Beschreibe maximal mögliche Leistung von Bibern**
  - $BB(n) = \max\{\mathrm{Produktivität}(\tau) \mid \tau \in \mathrm{BBT}(n)\}$

Ist die Busy-Beaver Funktion berechenbar?

Busy-Beaver Problem: Intuitive Analyse

- **Beispiel einer BBT(2) Maschine**

$\delta =$

s	a	s'	a'	P
1	|	2	|	r
1	b	2	|	l
2	|	2	|	h
2	b	1	|	l

Busy-Beaver Problem: Intuitive Analyse

- **Beispiel einer BBT(2) Maschine**

$\delta =$

s	a	s'	a'	P
1	|	2	|	r
1	b	2	|	l
2	|	2	|	h
2	b	1	|	l

Arbeitsweise:

Busy-Beaver Problem: Intuitive Analyse

- **Beispiel einer BBT(2) Maschine**

$\delta =$

s	a	s'	a'	P
1	|	2	|	r
1	b	2	|	l
2	|	2	|	h
2	b	1	|	l

Arbeitsweise: $b1b$

Busy-Beaver Problem: Intuitive Analyse

- **Beispiel einer BBT(2) Maschine**

$\delta =$

s	a	s'	a'	P
1	|	2	|	r
1	b	2	|	l
2	|	2	|	h
2	b	1	|	l

Arbeitsweise: $b1b$

$\mapsto |2b$

Busy-Beaver Problem: Intuitive Analyse

- **Beispiel einer BBT(2) Maschine**

$\delta =$

s	a	s'	a'	P
1	|	2	|	r
1	b	2	|	l
2	|	2	|	h
2	b	1	|	l

Arbeitsweise:

$$b1b$$
$$\mapsto |2b$$
$$\mapsto b1||$$

Busy-Beaver Problem: Intuitive Analyse

- **Beispiel einer BBT(2) Maschine**

$\delta =$

s	a	s'	a'	P
1	|	2	|	r
1	b	2	|	l
2	|	2	|	h
2	b	1	|	l

Arbeitsweise:

$$b1b$$
$$\mapsto |2b$$
$$\mapsto b1||$$
$$\mapsto b2b||$$

Busy-Beaver Problem: Intuitive Analyse

- **Beispiel einer BBT(2) Maschine**

$\delta =$

s	a	s'	a'	P
1	|	2	|	r
1	b	2	|	l
2	|	2	|	h
2	b	1	|	l

Arbeitsweise:

$$
\begin{aligned}
& b1b \\
\mapsto\ & |2b \\
\mapsto\ & b1|| \\
\mapsto\ & b2b|| \\
\mapsto\ & b1b|||
\end{aligned}
$$

Busy-Beaver Problem: Intuitive Analyse

- **Beispiel einer BBT(2) Maschine**

$\delta =$

s	a	s'	a'	P
1	|	2	|	r
1	b	2	|	l
2	|	2	|	h
2	b	1	|	l

Arbeitsweise:

$$
\begin{aligned}
& b1b \\
\mapsto\ & |2b \\
\mapsto\ & b1|| \\
\mapsto\ & b2b|| \\
\mapsto\ & b1b||| \\
\mapsto\ & |2|||
\end{aligned}
$$

Busy-Beaver Problem: Intuitive Analyse

- **Beispiel einer BBT(2) Maschine**

$\delta =$

s	a	s'	a'	P
1	|	2	|	r
1	b	2	|	l
2	|	2	|	h
2	b	1	|	l

Arbeitsweise:

$$
\begin{aligned}
& b1b \\
\mapsto\ & |2b \\
\mapsto\ & b1|| \\
\mapsto\ & b2b|| \\
\mapsto\ & b1b||| \\
\mapsto\ & |2||| \\
& \text{stop}
\end{aligned}
$$

Busy-Beaver Problem: Intuitive Analyse

- **Beispiel einer BBT(2) Maschine**

$\delta =$

s	a	s'	a'	P
1	|	2	|	r
1	b	2	|	l
2	|	2	|	h
2	b	1	|	l

Arbeitsweise:

$$
\begin{aligned}
& b1b \\
\mapsto\ & |2b \\
\mapsto\ & b1|| \\
\mapsto\ & b2b|| \\
\mapsto\ & b1b||| \\
\mapsto\ & |2||| \\
& \text{stop}
\end{aligned}
$$

– Produktivität ist 3 (4, wenn man alle Holzstücke zählt)

Busy-Beaver Problem: Intuitive Analyse

- **Beispiel einer BBT(2) Maschine**

$\delta =$

s	a	s'	a'	P
1	|	2	|	r
1	b	2	|	l
2	|	2	|	h
2	b	1	|	l

Arbeitsweise:

$$
\begin{aligned}
& b1b \\
\mapsto\ & |2b \\
\mapsto\ & b1|| \\
\mapsto\ & b2b|| \\
\mapsto\ & b1b||| \\
\mapsto\ & |2||| \\
& \text{stop}
\end{aligned}
$$

– Produktivität ist 3 (4, wenn man alle Holzstücke zählt)

- **$BB(n)$ bekannt für kleine n:**

n	1	2	3	4	5	6	...
	1	4	6	13	$\geq$4098	$\geq 6.4 * 10^{462}$	

Busy-Beaver Problem: Intuitive Analyse

- **Beispiel einer BBT(2) Maschine**

$\delta =$

s	a	s'	a'	P
1	|	2	|	r
1	b	2	|	l
2	|	2	|	h
2	b	1	|	l

Arbeitsweise:

$b1b$
$\mapsto |2b$
$\mapsto b1||$
$\mapsto b2b||$
$\mapsto b1b|||$
$\mapsto |2|||$
stop

– Produktivität ist 3 (4, wenn man alle Holzstücke zählt)

- ***BB*(*n*) bekannt für kleine *n*:**

n	1	2	3	4	5	6	...
	1	4	6	13	$\geq$4098	$\geq 6.4 * 10^{462}$	

- **Vollständige Analyse nicht möglich**

– $|\mathrm{BBT}(n)|$ ist $(|S|*|\Gamma|*|\{r,l\}|)^{(|S|*|\Gamma|-1)} * (|S|*|\Gamma|*|\{h\}|) * (|S|*|\Gamma|) = (4n)^{2n}$

$|\mathrm{BBT}(1)|$=16, $|\mathrm{BBT}(2)|$=4096, $|\mathrm{BBT}(3)|$=2985984, ...

Busy-Beaver Problem: Intuitive Analyse

- **Beispiel einer BBT(2) Maschine**

$\delta =$

s	a	s'	a'	P
1	|	2	|	r
1	b	2	|	l
2	|	2	|	h
2	b	1	|	l

Arbeitsweise:

$b1b$
$\mapsto |2b$
$\mapsto b1||$
$\mapsto b2b||$
$\mapsto b1b|||$
$\mapsto |2|||$
stop

– Produktivität ist 3 (4, wenn man alle Holzstücke zählt)

- **$BB(n)$ bekannt für kleine n:**

n	1	2	3	4	5	6	...
	1	4	6	13	$\geq$4098	$\geq 6.4*10^{462}$	

- **Vollständige Analyse nicht möglich**
  – $|\text{BBT}(n)|$ ist $(|S|*|\Gamma|*|\{r,l\}|)^{(|S|*|\Gamma|-1)} * (|S|*|\Gamma|*|\{h\}|) * (|S|*|\Gamma|) = (4n)^{2n}$
  $|\text{BBT}(1)|$=16, $|\text{BBT}(2)|$=4096, $|\text{BBT}(3)|$=2985984, ...
  – Anzahl möglicher Bandkonfigurationen einer TM ist unbegrenzt

Das Busy-Beaver Problem ist unlösbar

- **BB ist streng monoton:** $i > j \Leftrightarrow \mathrm{BB}(i) > \mathrm{BB}(j)$

Das Busy-Beaver Problem ist unlösbar

- **BB ist streng monoton:** $\boldsymbol{i>j \;\Leftrightarrow\; \mathrm{BB}(i)>\mathrm{BB}(j)}$
  - Für alle n gilt $\mathrm{BB}(n+1)>\mathrm{BB}(n)$
    - Schreibe in Zustand 1 ein | und beginne mit der $\mathrm{BB}(n)$-Maschine

Das Busy-Beaver Problem ist unlösbar

- **BB ist streng monoton: $i>j \Leftrightarrow \mathrm{BB}(i)>\mathrm{BB}(j)$**
  - Für alle n gilt $\mathrm{BB}(n+1)>\mathrm{BB}(n)$
    - Schreibe in Zustand 1 ein | und beginne mit der BB(n)-Maschine
  - $i>j \Leftrightarrow \mathrm{BB}(i)>\mathrm{BB}(j)$ folgt nun durch Induktion über $i-j$

Das Busy-Beaver Problem ist unlösbar

- **BB ist streng monoton:** $\boldsymbol{i>j \;\Leftrightarrow\; \mathrm{BB}(i)>\mathrm{BB}(j)}$
  - Für alle n gilt $\mathrm{BB}(n+1)>\mathrm{BB}(n)$
    - Schreibe in Zustand 1 ein | und beginne mit der $\mathrm{BB}(n)$-Maschine
  - $i>j \;\Leftrightarrow\; \mathrm{BB}(i)>\mathrm{BB}(j)$ folgt nun durch Induktion über $i-j$
- **Für alle n gilt:** $\boldsymbol{\mathrm{BB}(n+8)\geq 2n}$

Das Busy-Beaver Problem ist unlösbar

- **BB ist streng monoton:** $\boldsymbol{i>j \;\Leftrightarrow\; \mathrm{BB}(i)>\mathrm{BB}(j)}$
  - Für alle n gilt $\mathrm{BB}(n+1)>\mathrm{BB}(n)$
    - Schreibe in Zustand 1 ein | und beginne mit der BB(n)-Maschine
  - $i>j \;\Leftrightarrow\; \mathrm{BB}(i)>\mathrm{BB}(j)$ folgt nun durch Induktion über $i-j$
- **Für alle n gilt:** $\boldsymbol{\mathrm{BB}(n+8)\geq 2n}$
  - Mit n Zuständen kann man n Striche generieren

Das Busy-Beaver Problem ist unlösbar

- **BB ist streng monoton:** $\boldsymbol{i>j \;\Leftrightarrow\; \mathrm{BB}(i)>\mathrm{BB}(j)}$
  - Für alle n gilt $\mathrm{BB}(n+1)>\mathrm{BB}(n)$
    - Schreibe in Zustand 1 ein | und beginne mit der BB(n)-Maschine
  - $i>j \;\Leftrightarrow\; \mathrm{BB}(i)>\mathrm{BB}(j)$ folgt nun durch Induktion über $i-j$
- **Für alle n gilt:** $\boldsymbol{\mathrm{BB}(n+8)\geq 2n}$
  - Mit n Zuständen kann man n Striche generieren
  - Mit 8 Zuständen kann man Striche verdoppeln (vgl τ_4 aus Kapitel 6.1)

Das Busy-Beaver Problem ist unlösbar

- **BB ist streng monoton:** $\boldsymbol{i>j \;\Leftrightarrow\; \mathrm{BB}(i)>\mathrm{BB}(j)}$
  - Für alle n gilt $\mathrm{BB}(n+1)>\mathrm{BB}(n)$
    - Schreibe in Zustand 1 ein | und beginne mit der BB(n)-Maschine
  - $i>j \;\Leftrightarrow\; \mathrm{BB}(i)>\mathrm{BB}(j)$ folgt nun durch Induktion über $i-j$
- **Für alle n gilt:** $\boldsymbol{\mathrm{BB}(n+8)\geq 2n}$
  - Mit n Zuständen kann man n Striche generieren
  - Mit 8 Zuständen kann man Striche verdoppeln (vgl τ_4 aus Kapitel 6.1)
- **BB berechenbar** $\boldsymbol{\Rightarrow\; \mathrm{BB}(n+2k)\geq \mathrm{BB}(\mathrm{BB}(n))}$ **für ein** $\boldsymbol{k}$

Das Busy-Beaver Problem ist unlösbar

- **BB ist streng monoton:** $\boldsymbol{i>j \;\Leftrightarrow\; \mathrm{BB}(i)>\mathrm{BB}(j)}$
  - Für alle n gilt $\mathrm{BB}(n+1)>\mathrm{BB}(n)$
    - Schreibe in Zustand 1 ein | und beginne mit der BB(n)-Maschine
  - $i>j \;\Leftrightarrow\; \mathrm{BB}(i)>\mathrm{BB}(j)$ folgt nun durch Induktion über $i-j$
- **Für alle n gilt:** $\boldsymbol{\mathrm{BB}(n+8)\geq 2n}$
  - Mit n Zuständen kann man n Striche generieren
  - Mit 8 Zuständen kann man Striche verdoppeln (vgl τ_4 aus Kapitel 6.1)
- **BB berechenbar** $\boldsymbol{\Rightarrow\; \mathrm{BB}(n+2k)\geq \mathrm{BB}(\mathrm{BB}(n))}$ **für ein** $\boldsymbol{k}$
  - Wähle $k :=$ Anzahl der Zustände der TM über $\Gamma=\{|,b\}$, die BB berechnet

Das Busy-Beaver Problem ist unlösbar

- **BB ist streng monoton: $i>j \Leftrightarrow \mathrm{BB}(i)>\mathrm{BB}(j)$**
  - Für alle n gilt $\mathrm{BB}(n+1)>\mathrm{BB}(n)$
    - Schreibe in Zustand 1 ein | und beginne mit der BB(n)-Maschine
  - $i>j \Leftrightarrow \mathrm{BB}(i)>\mathrm{BB}(j)$ folgt nun durch Induktion über $i-j$
- **Für alle n gilt: $\mathrm{BB}(n+8)\geq 2n$**
  - Mit n Zuständen kann man n Striche generieren
  - Mit 8 Zuständen kann man Striche verdoppeln (vgl τ_4 aus Kapitel 6.1)
- **BB berechenbar $\Rightarrow \mathrm{BB}(n+2k)\geq \mathrm{BB}(\mathrm{BB}(n))$ für ein k**
  - Wähle $k :=$ Anzahl der Zustände der TM über $\Gamma=\{|,b\}$, die BB berechnet
  - Mit n Zuständen generiere n Striche
  - Mit k Zuständen berechne jetzt $\mathrm{BB}(n)$
  - Mit weiteren k Zuständen berechne $\mathrm{BB}(\mathrm{BB}(n))$

Das Busy-Beaver Problem ist unlösbar

- **BB ist streng monoton: $i>j \Leftrightarrow \mathrm{BB}(i)>\mathrm{BB}(j)$**
  - Für alle n gilt $\mathrm{BB}(n+1)>\mathrm{BB}(n)$
    - Schreibe in Zustand 1 ein | und beginne mit der BB(n)-Maschine
  - $i>j \Leftrightarrow \mathrm{BB}(i)>\mathrm{BB}(j)$ folgt nun durch Induktion über $i-j$
- **Für alle n gilt: $\mathrm{BB}(n+8)\geq 2n$**
  - Mit n Zuständen kann man n Striche generieren
  - Mit 8 Zuständen kann man Striche verdoppeln (vgl τ_4 aus Kapitel 6.1)
- **BB berechenbar $\Rightarrow \mathrm{BB}(n+2k)\geq \mathrm{BB}(\mathrm{BB}(n))$ für ein k**
  - Wähle $k :=$ Anzahl der Zustände der TM über $\Gamma=\{|,b\}$, die BB berechnet
  - Mit n Zuständen generiere n Striche
  - Mit k Zuständen berechne jetzt $\mathrm{BB}(n)$
  - Mit weiteren k Zuständen berechne $\mathrm{BB}(\mathrm{BB}(n))$
- **BB kann nicht berechenbar sein**

Das Busy-Beaver Problem ist unlösbar

- **BB ist streng monoton: $i>j \Leftrightarrow \mathrm{BB}(i)>\mathrm{BB}(j)$**
  - Für alle n gilt $\mathrm{BB}(n+1)>\mathrm{BB}(n)$
    - Schreibe in Zustand 1 ein | und beginne mit der BB(n)-Maschine
  - $i>j \Leftrightarrow \mathrm{BB}(i)>\mathrm{BB}(j)$ folgt nun durch Induktion über $i-j$
- **Für alle n gilt: $\mathrm{BB}(n+8)\geq 2n$**
  - Mit n Zuständen kann man n Striche generieren
  - Mit 8 Zuständen kann man Striche verdoppeln (vgl τ_4 aus Kapitel 6.1)
- **BB berechenbar $\Rightarrow \mathrm{BB}(n+2k)\geq \mathrm{BB}(\mathrm{BB}(n))$ für ein k**
  - Wähle $k :=$ Anzahl der Zustände der TM über $\Gamma=\{|,b\}$, die BB berechnet
  - Mit n Zuständen generiere n Striche
  - Mit k Zuständen berechne jetzt $\mathrm{BB}(n)$
  - Mit weiteren k Zuständen berechne $\mathrm{BB}(\mathrm{BB}(n))$
- **BB kann nicht berechenbar sein**
  - Sonst gibt es ein k, so daß für alle n: $\mathrm{BB}(n+8+2k) \geq \mathrm{BB}(\mathrm{BB}(n+8)) \geq \mathrm{BB}(2n)$

Das Busy-Beaver Problem ist unlösbar

- **BB ist streng monoton: $i>j \Leftrightarrow \mathrm{BB}(i)>\mathrm{BB}(j)$**
  - Für alle n gilt $\mathrm{BB}(n+1)>\mathrm{BB}(n)$
    - Schreibe in Zustand 1 ein | und beginne mit der BB(n)-Maschine
  - $i>j \Leftrightarrow \mathrm{BB}(i)>\mathrm{BB}(j)$ folgt nun durch Induktion über $i-j$
- **Für alle n gilt: $\mathrm{BB}(n+8)\geq 2n$**
  - Mit n Zuständen kann man n Striche generieren
  - Mit 8 Zuständen kann man Striche verdoppeln (vgl τ_4 aus Kapitel 6.1)
- **BB berechenbar $\Rightarrow \mathrm{BB}(n+2k)\geq \mathrm{BB}(\mathrm{BB}(n))$ für ein k**
  - Wähle $k :=$ Anzahl der Zustände der TM über $\Gamma=\{|,b\}$, die BB berechnet
  - Mit n Zuständen generiere n Striche
  - Mit k Zuständen berechne jetzt $\mathrm{BB}(n)$
  - Mit weiteren k Zuständen berechne $\mathrm{BB}(\mathrm{BB}(n))$
- **BB kann nicht berechenbar sein**
  - Sonst gibt es ein k, so daß für alle n: $\mathrm{BB}(n+8+2k) \geq \mathrm{BB}(\mathrm{BB}(n+8)) \geq \mathrm{BB}(2n)$
  - Für $n=2k+9$ widerspräche $\mathrm{BB}(4k+17) \geq \mathrm{BB}(4k+18)$ der Monotonie ✓

Problemreduktion

- **Ziel: Wiederverwendung bekannter Ergebnisse**
  - Zur Lösung eines Problems P bzw. zum Nachweis seiner Unlösbarkeit

Problemreduktion

- **Ziel: Wiederverwendung bekannter Ergebnisse**
  - Zur Lösung eines Problems P bzw. zum Nachweis seiner Unlösbarkeit
  - Unlösbar $\hat{=}$ unentscheidbar, nicht aufzählbar, nicht in Zeit t lösbar, ...

Problemreduktion

- **Ziel: Wiederverwendung bekannter Ergebnisse**
  - Zur Lösung eines Problems P bzw. zum Nachweis seiner Unlösbarkeit
  - Unlösbar $\hat{=}$ unentscheidbar, nicht aufzählbar, nicht in Zeit t lösbar, ...
- **Methodik zum Nachweis der Unlösbarkeit**
  - Transformiere P in ein anderes Problem P', das als unlösbar bekannt ist

Problemreduktion

- **Ziel: Wiederverwendung bekannter Ergebnisse**
  - Zur Lösung eines Problems P bzw. zum Nachweis seiner Unlösbarkeit
  - Unlösbar $\hat{=}$ unentscheidbar, nicht aufzählbar, nicht in Zeit t lösbar, ...
- **Methodik zum Nachweis der Unlösbarkeit**
  - Transformiere P in ein anderes Problem P', das als unlösbar bekannt ist
  - Zeige, daß jede Lösung für P in eine Lösung für P' transformiert würde

Problemreduktion

- **Ziel: Wiederverwendung bekannter Ergebnisse**
  - Zur Lösung eines Problems P bzw. zum Nachweis seiner Unlösbarkeit
  - Unlösbar $\hat{=}$ unentscheidbar, nicht aufzählbar, nicht in Zeit t lösbar, ...
- **Methodik zum Nachweis der Unlösbarkeit**
  - Transformiere P in ein anderes Problem P', das als unlösbar bekannt ist
  - Zeige, daß jede Lösung für P in eine Lösung für P' transformiert würde
  - P kann also nicht lösbar sein

Problemreduktion

- **Ziel: Wiederverwendung bekannter Ergebnisse**
  - Zur Lösung eines Problems P bzw. zum Nachweis seiner Unlösbarkeit
  - Unlösbar $\hat{=}$ unentscheidbar, nicht aufzählbar, nicht in Zeit t lösbar, ...
- **Methodik zum Nachweis der Unlösbarkeit**
  - Transformiere P in ein anderes Problem P', das als unlösbar bekannt ist
  - Zeige, daß jede Lösung für P in eine Lösung für P' transformiert würde
  - P kann also nicht lösbar sein
- **Methodik zur Konstruktion einer Lösung**
  - Transformiere ein anderes Problem P', das als lösbar bekannt ist, in P

Problemreduktion

- **Ziel: Wiederverwendung bekannter Ergebnisse**
  - Zur Lösung eines Problems P bzw. zum Nachweis seiner Unlösbarkeit
  - Unlösbar $\hat{=}$ unentscheidbar, nicht aufzählbar, nicht in Zeit t lösbar, ...
- **Methodik zum Nachweis der Unlösbarkeit**
  - Transformiere P in ein anderes Problem P', das als unlösbar bekannt ist
  - Zeige, daß jede Lösung für P in eine Lösung für P' transformiert würde
  - P kann also nicht lösbar sein
- **Methodik zur Konstruktion einer Lösung**
  - Transformiere ein anderes Problem P', das als lösbar bekannt ist, in P
  - Zeige, wie eine Lösung für P' in eine Lösung für P transformiert wird

Problemreduktion

- **Ziel: Wiederverwendung bekannter Ergebnisse**
  - Zur Lösung eines Problems P bzw. zum Nachweis seiner Unlösbarkeit
  - Unlösbar $\hat{=}$ unentscheidbar, nicht aufzählbar, nicht in Zeit t lösbar, ...
- **Methodik zum Nachweis der Unlösbarkeit**
  - Transformiere P in ein anderes Problem P', das als unlösbar bekannt ist
  - Zeige, daß jede Lösung für P in eine Lösung für P' transformiert würde
  - P kann also nicht lösbar sein
- **Methodik zur Konstruktion einer Lösung**
  - Transformiere ein anderes Problem P', das als lösbar bekannt ist, in P
  - Zeige, wie eine Lösung für P' in eine Lösung für P transformiert wird
- **Hilfsmittel: Reduzierbarkeit $P' \leq P$**
  - $P' \leq P$, falls $P' = f^{-1}(P) = \{x \mid f(x) \in P\}$ für ein total-berechenbares f
  - "P' ist reduzierbar auf P"

Problemreduktion

- **Ziel: Wiederverwendung bekannter Ergebnisse**
  - Zur Lösung eines Problems P bzw. zum Nachweis seiner Unlösbarkeit
  - Unlösbar $\hat{=}$ unentscheidbar, nicht aufzählbar, nicht in Zeit t lösbar, ...
- **Methodik zum Nachweis der Unlösbarkeit**
  - Transformiere P in ein anderes Problem P', das als unlösbar bekannt ist
  - Zeige, daß jede Lösung für P in eine Lösung für P' transformiert würde
  - P kann also nicht lösbar sein
- **Methodik zur Konstruktion einer Lösung**
  - Transformiere ein anderes Problem P', das als lösbar bekannt ist, in P
  - Zeige, wie eine Lösung für P' in eine Lösung für P transformiert wird
- **Hilfsmittel: Reduzierbarkeit $P' \leq P$**
  - $P' \leq P$, falls $P' = f^{-1}(P) = \{x \mid f(x) \in P\}$ für ein total-berechenbares f
  - "P' ist reduzierbar auf P" (Begriff gilt für Teilmengen von Zahlen, Worten, ...)

Beweisführung durch Reduktion

- $P' \leq P$ **bedeutet "P' ist leichter als P"**

Beweisführung durch Reduktion

- $P' \leq P$ **bedeutet "P' ist leichter als P"**
  - Ist P lösbar, dann kann P' wie folgt gelöst werden

Beweisführung durch Reduktion

- $P' \leq P$ **bedeutet "P' ist leichter als P"**
  - Ist P lösbar, dann kann P' wie folgt gelöst werden
    - Bei Eingabe x bestimme $f(x)$ (f ist die Reduktionsfunktion)

Beweisführung durch Reduktion

- $P' \leq P$ **bedeutet "P' ist leichter als P"**
  - Ist P lösbar, dann kann P' wie folgt gelöst werden
    - Bei Eingabe x bestimme $f(x)$ (f ist die Reduktionsfunktion)
    - Löse $f(x)$ mit der Lösungsmethode für P

Beweisführung durch Reduktion

- $P' \leq P$ **bedeutet "P' ist leichter als P"**
  - Ist P lösbar, dann kann P' wie folgt gelöst werden
    - Bei Eingabe x bestimme $f(x)$ (f ist die Reduktionsfunktion)
    - Löse $f(x)$ mit der Lösungsmethode für P
    - Es gilt $x \in P' \Leftrightarrow f(x) \in P$, also überträgt sich das Ergebnis

Beweisführung durch Reduktion

- **$P' \leq P$ bedeutet "P' ist leichter als P"**
  - Ist P lösbar, dann kann P' wie folgt gelöst werden
    - Bei Eingabe x bestimme $f(x)$ (f ist die Reduktionsfunktion)
    - Löse $f(x)$ mit der Lösungsmethode für P
    - Es gilt $x \in P' \Leftrightarrow f(x) \in P$, also überträgt sich das Ergebnis
- **Aus $P' \leq P$ und P entscheidbar folgt P' entscheidbar**
  - Übertragung von Entscheidbarkeit: $\chi_{P'}(x) = \chi_{f^{-1}(P)}(x) = \chi_P(f(x))$

Beweisführung durch Reduktion

- **$P' \leq P$ bedeutet "P' ist leichter als P"**
  - Ist P lösbar, dann kann P' wie folgt gelöst werden
    - Bei Eingabe x bestimme $f(x)$ (f ist die Reduktionsfunktion)
    - Löse $f(x)$ mit der Lösungsmethode für P
    - Es gilt $x \in P' \Leftrightarrow f(x) \in P$, also überträgt sich das Ergebnis
- **Aus $P' \leq P$ und P entscheidbar folgt P' entscheidbar**
  - Übertragung von Entscheidbarkeit: $\chi_{P'}(x) = \chi_{f^{-1}(P)}(x) = \chi_P(f(x))$
- **Aus $P' \leq P$ und P aufzählbar folgt P' aufzählbar**
  - Übertragung von Aufzählbarkeit: $\psi_{P'}(x) = \psi_{f^{-1}(P)}(x) = \psi_P(f(x))$

Beispiele von Problemreduktion

- $\boldsymbol{S = \{i \mid \varphi_i(i) \neq \perp\} \;\leq\; H = \{(i,n) \mid \varphi_i(n) \neq \perp\}}$

  "Das Selbstanwendbarkeitsproblem ist leichter als das Halteproblem"

Beispiele von Problemreduktion

- $\boldsymbol{S = \{i \mid \varphi_i(i) \neq \bot\} \;\leq\; H = \{(i,n) \mid \varphi_i(n) \neq \bot\}}$

  "Das Selbstanwendbarkeitsproblem ist leichter als das Halteproblem"

  – Es gilt $i \in S \Leftrightarrow (i,i) \in H$.

Beispiele von Problemreduktion

- $\boldsymbol{S = \{i \mid \varphi_i(i) \neq \bot\} \;\leq\; H = \{(i,n) \mid \varphi_i(n) \neq \bot\}}$

  "Das Selbstanwendbarkeitsproblem ist leichter als das Halteproblem"

  – Es gilt $i \in S \Leftrightarrow (i,i) \in H$.

  – Wähle $f(i) := (i,i)$. Dann ist f total-berechenbar und $S = f^{-1}(H)$

Beispiele von Problemreduktion

- $\boldsymbol{S = \{i \mid \varphi_i(i) \neq \bot\} \;\leq\; H = \{(i,n) \mid \varphi_i(n) \neq \bot\}}$

  "Das Selbstanwendbarkeitsproblem ist leichter als das Halteproblem"

  – Es gilt $i \in S \Leftrightarrow (i,i) \in H$.

  – Wähle $f(i) := (i,i)$. Dann ist f total-berechenbar und $S = f^{-1}(H)$

  – Man kann auch H auf S reduzieren (aufwendig)

Beispiele von Problemreduktion

- $\boldsymbol{S = \{i \mid \varphi_i(i) \neq \perp\} \;\leq\; H = \{(i,n) \mid \varphi_i(n) \neq \perp\}}$

  "Das Selbstanwendbarkeitsproblem ist leichter als das Halteproblem"

  – Es gilt $i \in S \Leftrightarrow (i,i) \in H$.

  – Wähle $f(i) := (i,i)$. Dann ist f total-berechenbar und $S = f^{-1}(H)$

  – Man kann auch H auf S reduzieren (aufwendig)

- $\boldsymbol{\overline{H} = \{(i,n) \mid \varphi_i(n) = \perp\} \;\leq\; PROG_z = \{i \mid \varphi_i = z\}}$

Beispiele von Problemreduktion

- $\boldsymbol{S = \{i \mid \varphi_i(i) \neq \bot\} \;\leq\; H = \{(i,n) \mid \varphi_i(n) \neq \bot\}}$

  "Das Selbstanwendbarkeitsproblem ist leichter als das Halteproblem"

  – Es gilt $i \in S \Leftrightarrow (i,i) \in H$.

  – Wähle $f(i) := (i,i)$. Dann ist f total-berechenbar und $S = f^{-1}(H)$

  – Man kann auch H auf S reduzieren (aufwendig)

- $\boldsymbol{\overline{H} = \{(i,n) \mid \varphi_i(n) = \bot\} \;\leq\; PROG_z = \{i \mid \varphi_i = z\}}$

  – Es gilt $(i,n) \in \overline{H} \;\Leftrightarrow\; \forall t \in \mathbb{N}.\Phi_i(n) \neq t$.

Beispiele von Problemreduktion

- $\boldsymbol{S = \{i \mid \varphi_i(i) \neq \bot\} \;\leq\; H = \{(i,n) \mid \varphi_i(n) \neq \bot\}}$

  "Das Selbstanwendbarkeitsproblem ist leichter als das Halteproblem"

  – Es gilt $i \in S \Leftrightarrow (i,i) \in H$.

  – Wähle $f(i) := (i,i)$. Dann ist f total-berechenbar und $S = f^{-1}(H)$

  – Man kann auch H auf S reduzieren (aufwendig)

- $\boldsymbol{\overline{H} = \{(i,n) \mid \varphi_i(n) = \bot\} \;\leq\; PROG_z = \{i \mid \varphi_i = z\}}$

  – Es gilt $(i,n) \in \overline{H} \;\Leftrightarrow\; \forall t \in \mathbb{N}.\Phi_i(n) \neq t$.

  – Da $\Phi = \{(i,n,t) \mid \Phi_i(n) = t\}$ entscheidbar ist, gibt es ein j mit

$$\varphi_j(i,n,t) = \chi_\Phi(i,n,t) = \begin{cases} 1 & \text{falls } (i,n,t) \in \Phi \\ 0 & \text{sonst} \end{cases}$$

Beispiele von Problemreduktion

- $\boldsymbol{S = \{i \mid \varphi_i(i) \neq \bot\} \;\leq\; H = \{(i,n) \mid \varphi_i(n) \neq \bot\}}$

  "Das Selbstanwendbarkeitsproblem ist leichter als das Halteproblem"

  – Es gilt $i \in S \Leftrightarrow (i,i) \in H$.

  – Wähle $f(i) := (i,i)$. Dann ist f total-berechenbar und $S = f^{-1}(H)$

  – Man kann auch H auf S reduzieren (aufwendig)

- $\boldsymbol{\overline{H} = \{(i,n) \mid \varphi_i(n) = \bot\} \;\leq\; PROG_z = \{i \mid \varphi_i = z\}}$

  – Es gilt $(i,n) \in \overline{H} \;\Leftrightarrow\; \forall t \in \mathbb{N}.\Phi_i(n) \neq t$.

  – Da $\Phi = \{(i,n,t) \mid \Phi_i(n) = t\}$ entscheidbar ist, gibt es ein j mit

  $$\varphi_j(i,n,t) = \chi_\Phi(i,n,t) = \begin{cases} 1 & \text{falls } (i,n,t) \in \Phi \\ 0 & \text{sonst} \end{cases}$$

  – Nach dem SMN Theorem gibt es eine total-berechenbare Funktion f mit $\varphi_{f(i,n)}(t) = \varphi_j(i,n,t)$

Beispiele von Problemreduktion

- $\boldsymbol{S = \{i \mid \varphi_i(i) \neq \bot\} \;\leq\; H = \{(i,n) \mid \varphi_i(n) \neq \bot\}}$

  "Das Selbstanwendbarkeitsproblem ist leichter als das Halteproblem"

  – Es gilt $i \in S \Leftrightarrow (i,i) \in H$.

  – Wähle $f(i) := (i,i)$. Dann ist f total-berechenbar und $S = f^{-1}(H)$

  – Man kann auch H auf S reduzieren (aufwendig)

- $\boldsymbol{\overline{H} = \{(i,n) \mid \varphi_i(n) = \bot\} \;\leq\; PROG_z = \{i \mid \varphi_i = z\}}$

  – Es gilt $(i,n) \in \overline{H} \;\Leftrightarrow\; \forall t \in \mathbb{N}.\Phi_i(n) \neq t$.

  – Da $\Phi = \{(i,n,t) \mid \Phi_i(n) = t\}$ entscheidbar ist, gibt es ein j mit

  $$\varphi_j(i,n,t) = \chi_\Phi(i,n,t) = \begin{cases} 1 & \text{falls } (i,n,t) \in \Phi \\ 0 & \text{sonst} \end{cases}$$

  – Nach dem SMN Theorem gibt es eine total-berechenbare Funktion f mit $\varphi_{f(i,n)}(t) = \varphi_j(i,n,t)$

  – Es folgt $(i,n) \in \overline{H} \;\Leftrightarrow\; \forall t \in \mathbb{N}.\Phi_i(n) \neq t \;\Leftrightarrow\; \forall t \in \mathbb{N}.\varphi_{f(i,n)}(t) = 0 \;\Leftrightarrow\; \varphi_{f(i,n)} = z$

  $\Leftrightarrow\; f(i,n) \in PROG_z$

Beispiele von Problemreduktion

- $\boldsymbol{S = \{i \mid \varphi_i(i) \neq \bot\} \;\leq\; H = \{(i,n) \mid \varphi_i(n) \neq \bot\}}$

  "Das Selbstanwendbarkeitsproblem ist leichter als das Halteproblem"

  – Es gilt $i \in S \Leftrightarrow (i,i) \in H$.

  – Wähle $f(i) := (i,i)$. Dann ist f total-berechenbar und $S = f^{-1}(H)$

  – Man kann auch H auf S reduzieren (aufwendig)

- $\boldsymbol{\overline{H} = \{(i,n) \mid \varphi_i(n) = \bot\} \;\leq\; PROG_z = \{i \mid \varphi_i = z\}}$

  – Es gilt $(i,n) \in \overline{H} \;\Leftrightarrow\; \forall t \in \mathbb{N}.\Phi_i(n) \neq t$.

  – Da $\Phi = \{(i,n,t) \mid \Phi_i(n) = t\}$ entscheidbar ist, gibt es ein j mit

  $$\varphi_j(i,n,t) = \chi_\Phi(i,n,t) = \begin{cases} 1 & \text{falls } (i,n,t) \in \Phi \\ 0 & \text{sonst} \end{cases}$$

  – Nach dem SMN Theorem gibt es eine total-berechenbare Funktion f mit $\varphi_{f(i,n)}(t) = \varphi_j(i,n,t)$

  – Es folgt $(i,n) \in \overline{H} \;\Leftrightarrow\; \forall t \in \mathbb{N}.\Phi_i(n) \neq t \;\Leftrightarrow\; \forall t \in \mathbb{N}.\varphi_{f(i,n)}(t) = 0 \;\Leftrightarrow\; \varphi_{f(i,n)} = z$

  $\Leftrightarrow\; f(i,n) \in PROG_z$

  also $\overline{H} = f^{-1}(PROG_z)$

Problemreduktion mit Abschlusseigenschaften

- **Ziel: Wiederverwendung bekannter Ergebnisse**
  - Zur Lösung eines Problems P bzw. zum Nachweis seiner Unlösbarkeit

Problemreduktion mit Abschlusseigenschaften

- **Ziel: Wiederverwendung bekannter Ergebnisse**
  - Zur Lösung eines Problems P bzw. zum Nachweis seiner Unlösbarkeit
- **Methodik**
  - Zeige, daß Lösung für P ein unlösbares Problem P' lösen würde

Problemreduktion mit Abschlusseigenschaften

- **Ziel: Wiederverwendung bekannter Ergebnisse**
  - Zur Lösung eines Problems P bzw. zum Nachweis seiner Unlösbarkeit
- **Methodik**
  - Zeige, daß Lösung für P ein unlösbares Problem P' lösen würde
  - Zeige, wie Lösung eines bekannten Problems P' zur Lösung von P' führt

Problemreduktion mit Abschlusseigenschaften

- **Ziel: Wiederverwendung bekannter Ergebnisse**
  - Zur Lösung eines Problems P bzw. zum Nachweis seiner Unlösbarkeit
- **Methodik**
  - Zeige, daß Lösung für P ein unlösbares Problem P' lösen würde
  - Zeige, wie Lösung eines bekannten Problems P' zur Lösung von P' führt
- **Hilfsmittel: Abschlußeigenschaften**
  - M, M' entscheidbar, dann auch $M \cup M'$, $M \cap M'$, $M \setminus M'$, $\overline{M}$, $f^{-1}(M)$
  - M, M' aufzählbar, dann auch $M \cup M'$, $M \cap M'$, $g(M)$, $g^{-1}(M)$
  - M entscheidbar $\Leftrightarrow$ M und $\overline{M}$ aufzählbar
  - Reduzierbarkeit ist ein besonders mächtiger Spezialfall

Problemreduktion mit Abschlusseigenschaften

- **Ziel: Wiederverwendung bekannter Ergebnisse**
  - Zur Lösung eines Problems P bzw. zum Nachweis seiner Unlösbarkeit
- **Methodik**
  - Zeige, daß Lösung für P ein unlösbares Problem P' lösen würde
  - Zeige, wie Lösung eines bekannten Problems P' zur Lösung von P' führt
- **Hilfsmittel: Abschlußeigenschaften**
  - M, M' entscheidbar, dann auch $M \cup M'$, $M \cap M'$, $M \backslash M'$, $\overline{M}$, $f^{-1}(M)$
  - M, M' aufzählbar, dann auch $M \cup M'$, $M \cap M'$, $g(M)$, $g^{-1}(M)$
  - M entscheidbar $\Leftrightarrow$ M und $\overline{M}$ aufzählbar
  - Reduzierbarkeit ist ein besonders mächtiger Spezialfall
- **Umkehrung der Abschlußeigenschaften**
  - M nicht entscheidbar $\Rightarrow$ $\overline{M}$ nicht entscheidbar
  - M aufzählbar, nicht entscheidbar $\Rightarrow$ $\overline{M}$ weder aufzählbar noch entscheidbar
  - M entscheidbar, $M \cup M'$ nicht entscheidbar $\Rightarrow$ M' nicht entscheidbar
  - M entscheidbar, $M \backslash M'$ nicht entscheidbar $\Rightarrow$ M' nicht entscheidbar

$\vdots$ $\vdots$

Resultate aus Abschlusseigenschaften

- $\{(i,n) \mid \varphi_i(n){=}\bot\}$ **ist nicht aufzählbar**
  - $\{(i,n) \mid \varphi_i(n){=}\bot\}$ ist das Komplement von $H = \{(i,n) \mid \varphi_i(n){\neq}\bot\}$
  - H ist aufzählbar aber unentscheidbar, also kann $\overline{H}$ nicht aufzählbar sein

Resultate aus Abschlusseigenschaften

- $\{(i,n) \mid \varphi_i(n)=\perp\}$ **ist nicht aufzählbar**
  - $\{(i,n) \mid \varphi_i(n)=\perp\}$ ist das Komplement von $H = \{(i,n) \mid \varphi_i(n)\neq\perp\}$
  - H ist aufzählbar aber unentscheidbar, also kann $\overline{H}$ nicht aufzählbar sein
- $\{i \mid \varphi_i(i)=\perp\}$ **ist nicht aufzählbar**
  - $\{i \mid \varphi_i(i)=\perp\}$ ist das Komplement von $S = \{i \mid \varphi_i(i)\neq\perp\}$
  - S ist aufzählbar aber unentscheidbar, also kann $\overline{S}$ nicht aufzählbar sein

Resultate aus Abschlusseigenschaften

- $\{(i,n) \mid \varphi_i(n)=\perp\}$ **ist nicht aufzählbar**
  - $\{(i,n) \mid \varphi_i(n)=\perp\}$ ist das Komplement von $H = \{(i,n) \mid \varphi_i(n)\neq\perp\}$
  - H ist aufzählbar aber unentscheidbar, also kann $\overline{H}$ nicht aufzählbar sein
- $\{i \mid \varphi_i(i)=\perp\}$ **ist nicht aufzählbar**
  - $\{i \mid \varphi_i(i)=\perp\}$ ist das Komplement von $S = \{i \mid \varphi_i(i)\neq\perp\}$
  - S ist aufzählbar aber unentscheidbar, also kann $\overline{S}$ nicht aufzählbar sein
- $PROG_z = \{i \mid \varphi_i = z\}$ **ist nicht aufzählbar**
  - Es gilt $\overline{H} \leq PROG_z$ und $\overline{H}$ ist nicht aufzählbar

Resultate aus Abschlusseigenschaften

- $\{(i,n) \mid \varphi_i(n)=\perp\}$ **ist nicht aufzählbar**
  - $\{(i,n) \mid \varphi_i(n)=\perp\}$ ist das Komplement von $H = \{(i,n) \mid \varphi_i(n)\neq\perp\}$
  - H ist aufzählbar aber unentscheidbar, also kann $\overline{H}$ nicht aufzählbar sein
- $\{i \mid \varphi_i(i)=\perp\}$ **ist nicht aufzählbar**
  - $\{i \mid \varphi_i(i)=\perp\}$ ist das Komplement von $S = \{i \mid \varphi_i(i)\neq\perp\}$
  - S ist aufzählbar aber unentscheidbar, also kann $\overline{S}$ nicht aufzählbar sein
- $PROG_z = \{i \mid \varphi_i = z\}$ **ist nicht aufzählbar**
  - Es gilt $\overline{H} \leq PROG_z$ und $\overline{H}$ ist nicht aufzählbar
- $PF_\varphi = \{i \mid \varphi_i \text{ partiell}\}$ **ist unentscheidbar**
  - PF_φ ist das Komplement von $\mathcal{R}_\varphi = \{i \mid \varphi_i \text{ total}\}$
  - $\mathcal{R}_\varphi$ ist nicht aufzählbar, also kann $PF_\varphi = \overline{\mathcal{R}_\varphi}$ nicht entscheidbar sein

Welche Verifikationsprobleme sind entscheidbar?

- **Halteproblem**
  - Kann man von einem beliebigen Programm entscheiden, ob es bei bestimmten Eingaben hält oder nicht?
  - Bereits als unentscheidbar nachgewiesen

Welche Verifikationsprobleme sind entscheidbar?

- **Halteproblem**
  - Kann man von einem beliebigen Programm entscheiden, ob es bei bestimmten Eingaben hält oder nicht?
  - Bereits als unentscheidbar nachgewiesen
- **Korrektheitsproblem**
  - Kann man von einem beliebigen Programm entscheiden, ob es eine bestimmte Funktion berechnet oder nicht?
  - Für die Nullfunktion bereits als unentscheidbar nachgewiesen
  - Gilt ähnliches für andere Funktionen?

Welche Verifikationsprobleme sind entscheidbar?

- **Halteproblem**
  - Kann man von einem beliebigen Programm entscheiden, ob es bei bestimmten Eingaben hält oder nicht?
  - Bereits als unentscheidbar nachgewiesen
- **Korrektheitsproblem**
  - Kann man von einem beliebigen Programm entscheiden, ob es eine bestimmte Funktion berechnet oder nicht?
  - Für die Nullfunktion bereits als unentscheidbar nachgewiesen
  - Gilt ähnliches für andere Funktionen?
- **Spezifikationsproblem**
  - Kann man von einem beliebigen Programm entscheiden, ob es eine gegebene Spezfikation erfüllt oder nicht?

Welche Verifikationsprobleme sind entscheidbar?

- **Halteproblem**
  - Kann man von einem beliebigen Programm entscheiden, ob es bei bestimmten Eingaben hält oder nicht?
  - Bereits als unentscheidbar nachgewiesen
- **Korrektheitsproblem**
  - Kann man von einem beliebigen Programm entscheiden, ob es eine bestimmte Funktion berechnet oder nicht?
  - Für die Nullfunktion bereits als unentscheidbar nachgewiesen
  - Gilt ähnliches für andere Funktionen?
- **Spezifikationsproblem**
  - Kann man von einem beliebigen Programm entscheiden, ob es eine gegebene Spezfikation erfüllt oder nicht?
- **Äquivalenzproblem**
  - Kann man entscheiden, ob zwei beliebige Programme die gleiche Funktion berechnen oder nicht?

Welche Verifikationsprobleme sind entscheidbar?

- **Halteproblem**
  - Kann man von einem beliebigen Programm entscheiden, ob es bei bestimmten Eingaben hält oder nicht?
  - Bereits als unentscheidbar nachgewiesen
- **Korrektheitsproblem**
  - Kann man von einem beliebigen Programm entscheiden, ob es eine bestimmte Funktion berechnet oder nicht?
  - Für die Nullfunktion bereits als unentscheidbar nachgewiesen
  - Gilt ähnliches für andere Funktionen?
- **Spezifikationsproblem**
  - Kann man von einem beliebigen Programm entscheiden, ob es eine gegebene Spezfikation erfüllt oder nicht?
- **Äquivalenzproblem**
  - Kann man entscheiden, ob zwei beliebige Programme die gleiche Funktion berechnen oder nicht?

Gibt es eine allgemeine Antwort?

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Beweis durch Reduktion auf $S = \{i \mid \varphi_i(i) \neq \perp\}$

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Beweis durch Reduktion auf $S = \{i \mid \varphi_i(i) \neq \bot\}$

– Betrachte $g \equiv \lambda x.\bot$

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Beweis durch Reduktion auf $S = \{i \mid \varphi_i(i) \neq \bot\}$

– Betrachte $g \equiv \lambda x.\bot$

– Falls $g \notin P$, so wähle $h \in P$ beliebig und definiere

$$h'(i,x) = \begin{cases} h(x) & \text{falls } \varphi_i(i) \neq \bot \\ \bot & \text{sonst} \end{cases}$$

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Beweis durch Reduktion auf $S = \{i \mid \varphi_i(i) \neq \bot\}$

– Betrachte $g \equiv \lambda x.\bot$

– Falls $g \notin P$, so wähle $h \in P$ beliebig und definiere

$$h'(i,x) = \begin{cases} h(x) & \text{falls } \varphi_i(i) \neq \bot \\ \bot & \text{sonst} \end{cases}$$

– Dann ist h' berechenbar und nach dem SMN Theorem gibt es ein total-berechenbares f mit $h'(i,x) = \varphi_{f(i)}(x)$

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Beweis durch Reduktion auf $S = \{i \mid \varphi_i(i) \neq \bot\}$

– Betrachte $g \equiv \lambda x.\bot$

– Falls $g \notin P$, so wähle $h \in P$ beliebig und definiere

$$h'(i,x) = \begin{cases} h(x) & \text{falls } \varphi_i(i) \neq \bot \\ \bot & \text{sonst} \end{cases}$$

– Dann ist h' berechenbar und nach dem SMN Theorem gibt es ein total-berechenbares f mit $h'(i,x) = \varphi_{f(i)}(x)$

– Es folgt: $i \in S$

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Beweis durch Reduktion auf $S = \{i \mid \varphi_i(i) \neq \bot\}$

– Betrachte $g \equiv \lambda x.\bot$

– Falls $g \notin P$, so wähle $h \in P$ beliebig und definiere

$$h'(i,x) = \begin{cases} h(x) & \text{falls } \varphi_i(i) \neq \bot \\ \bot & \text{sonst} \end{cases}$$

– Dann ist h' berechenbar und nach dem SMN Theorem gibt es ein total-berechenbares f mit $h'(i,x) = \varphi_{f(i)}(x)$

– Es folgt: $i \in S \Rightarrow \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = h(x)$

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Beweis durch Reduktion auf $S = \{i \mid \varphi_i(i) \neq \bot\}$

– Betrachte $g \equiv \lambda x.\bot$

– Falls $g \notin P$, so wähle $h \in P$ beliebig und definiere

$$h'(i,x) = \begin{cases} h(x) & \text{falls } \varphi_i(i) \neq \bot \\ \bot & \text{sonst} \end{cases}$$

– Dann ist h' berechenbar und nach dem SMN Theorem gibt es ein total-berechenbares f mit $h'(i,x) = \varphi_{f(i)}(x)$

– Es folgt: $i \in S \Rightarrow \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = h(x) \Rightarrow \varphi_{f(i)} = h \in P$

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Beweis durch Reduktion auf $S = \{i \mid \varphi_i(i) \neq \bot\}$

– Betrachte $g \equiv \lambda x.\bot$

– Falls $g \notin P$, so wähle $h \in P$ beliebig und definiere

$$h'(i,x) = \begin{cases} h(x) & \text{falls } \varphi_i(i) \neq \bot \\ \bot & \text{sonst} \end{cases}$$

– Dann ist h' berechenbar und nach dem SMN Theorem gibt es ein total-berechenbares f mit $h'(i,x) = \varphi_{f(i)}(x)$

– Es folgt: $i \in S \Rightarrow \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = h(x) \Rightarrow \varphi_{f(i)} = h \in P \Rightarrow f(i) \in \mathcal{L}_P$

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Beweis durch Reduktion auf $S = \{i \mid \varphi_i(i) \neq \bot\}$

– Betrachte $g \equiv \lambda x.\bot$

– Falls $g \notin P$, so wähle $h \in P$ beliebig und definiere

$$h'(i,x) = \begin{cases} h(x) & \text{falls } \varphi_i(i) \neq \bot \\ \bot & \text{sonst} \end{cases}$$

– Dann ist h' berechenbar und nach dem SMN Theorem gibt es ein total-berechenbares f mit $h'(i,x) = \varphi_{f(i)}(x)$

– Es folgt: $i \in S \Rightarrow \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = h(x) \Rightarrow \varphi_{f(i)} = h \in P \Rightarrow f(i) \in \mathcal{L}_P$

$i \notin S$

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Beweis durch Reduktion auf $S = \{i \mid \varphi_i(i) \neq \bot\}$

– Betrachte $g \equiv \lambda x.\bot$

– Falls $g \notin P$, so wähle $h \in P$ beliebig und definiere

$$h'(i,x) = \begin{cases} h(x) & \text{falls } \varphi_i(i) \neq \bot \\ \bot & \text{sonst} \end{cases}$$

– Dann ist h' berechenbar und nach dem SMN Theorem gibt es ein total-berechenbares f mit $h'(i,x) = \varphi_{f(i)}(x)$

– Es folgt: $i \in S \Rightarrow \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = h(x) \Rightarrow \varphi_{f(i)} = h \in P \Rightarrow f(i) \in \mathcal{L}_P$

$i \notin S \Rightarrow \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = \bot$

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Beweis durch Reduktion auf $S = \{i \mid \varphi_i(i) \neq \bot\}$

– Betrachte $g \equiv \lambda x.\bot$

– Falls $g \notin P$, so wähle $h \in P$ beliebig und definiere

$$h'(i,x) = \begin{cases} h(x) & \text{falls } \varphi_i(i) \neq \bot \\ \bot & \text{sonst} \end{cases}$$

– Dann ist h' berechenbar und nach dem SMN Theorem gibt es ein total-berechenbares f mit $h'(i,x) = \varphi_{f(i)}(x)$

– Es folgt: $i \in S \Rightarrow \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = h(x) \Rightarrow \varphi_{f(i)} = h \in P \Rightarrow f(i) \in \mathcal{L}_P$

$i \notin S \Rightarrow \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = \bot \Rightarrow \varphi_{f(i)} = g \notin P$

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Beweis durch Reduktion auf $S = \{i \mid \varphi_i(i) \neq \bot\}$

– Betrachte $g \equiv \lambda x.\bot$

– Falls $g \notin P$, so wähle $h \in P$ beliebig und definiere

$$h'(i,x) = \begin{cases} h(x) & \text{falls } \varphi_i(i) \neq \bot \\ \bot & \text{sonst} \end{cases}$$

– Dann ist h' berechenbar und nach dem SMN Theorem gibt es ein total-berechenbares f mit $h'(i,x) = \varphi_{f(i)}(x)$

– Es folgt: $i \in S \Rightarrow \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = h(x) \Rightarrow \varphi_{f(i)} = h \in P \Rightarrow f(i) \in \mathcal{L}_P$

$i \notin S \Rightarrow \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = \bot \Rightarrow \varphi_{f(i)} = g \notin P \Rightarrow f(i) \notin \mathcal{L}_P$

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Beweis durch Reduktion auf $S = \{i \mid \varphi_i(i) \neq \bot\}$

– Betrachte $g \equiv \lambda x.\bot$

– Falls $g \notin P$, so wähle $h \in P$ beliebig und definiere

$$h'(i,x) = \begin{cases} h(x) & \text{falls } \varphi_i(i) \neq \bot \\ \bot & \text{sonst} \end{cases}$$

– Dann ist h' berechenbar und nach dem SMN Theorem gibt es ein total-berechenbares f mit $h'(i,x) = \varphi_{f(i)}(x)$

– Es folgt: $i \in S \Rightarrow \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = h(x) \Rightarrow \varphi_{f(i)} = h \in P \Rightarrow f(i) \in \mathcal{L}_P$

$i \notin S \Rightarrow \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = \bot \Rightarrow \varphi_{f(i)} = g \notin P \Rightarrow f(i) \notin \mathcal{L}_P$

– Insgesamt $i \in S \Leftrightarrow f(i) \in \mathcal{L}_P$, also $S \leq \mathcal{L}_P$

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Beweis durch Reduktion auf $S = \{i \mid \varphi_i(i) \neq \bot\}$

– Betrachte $g \equiv \lambda x.\bot$

– Falls $g \notin P$, so wähle $h \in P$ beliebig und definiere

$$h'(i,x) = \begin{cases} h(x) & \text{falls } \varphi_i(i) \neq \bot \\ \bot & \text{sonst} \end{cases}$$

– Dann ist h' berechenbar und nach dem SMN Theorem gibt es ein total-berechenbares f mit $h'(i,x) = \varphi_{f(i)}(x)$

– Es folgt: $i \in S \;\Rightarrow\; \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = h(x) \;\Rightarrow\; \varphi_{f(i)} = h \in P \;\Rightarrow\; f(i) \in \mathcal{L}_P$

$i \notin S \;\Rightarrow\; \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = \bot \;\Rightarrow\; \varphi_{f(i)} = g \notin P \;\Rightarrow\; f(i) \notin \mathcal{L}_P$

– Insgesamt $i \in S \;\Leftrightarrow\; f(i) \in \mathcal{L}_P$, also $S \leq \mathcal{L}_P$

– Da S unentscheidbar ist, muß dies auch für $\mathcal{L}_P$ gelten ✓

Der Satz von Rice

Keine nichttriviale extensionale Eigenschaft berechenbarer Funktionen ist entscheidbar

Für $\emptyset \neq P \subset \mathcal{T}_\mu$ ist $\mathcal{L}_P = \{i \mid \varphi_i \in P\}$ nicht entscheidbar

Beweis durch Reduktion auf $S = \{i \mid \varphi_i(i) \neq \bot\}$

– Betrachte $g \equiv \lambda x.\bot$

– Falls $g \notin P$, so wähle $h \in P$ beliebig und definiere

$$h'(i,x) = \begin{cases} h(x) & \text{falls } \varphi_i(i) \neq \bot \\ \bot & \text{sonst} \end{cases}$$

– Dann ist h' berechenbar und nach dem SMN Theorem gibt es ein total-berechenbares f mit $h'(i,x) = \varphi_{f(i)}(x)$

– Es folgt: $i \in S \;\Rightarrow\; \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = h(x) \;\Rightarrow\; \varphi_{f(i)} = h \in P \;\Rightarrow\; f(i) \in \mathcal{L}_P$

$i \notin S \;\Rightarrow\; \forall x.\, h'(i,x) = \varphi_{f(i)}(x) = \bot \;\Rightarrow\; \varphi_{f(i)} = g \notin P \;\Rightarrow\; f(i) \notin \mathcal{L}_P$

– Insgesamt $i \in S \;\Leftrightarrow\; f(i) \in \mathcal{L}_P$, also $S \leq \mathcal{L}_P$

– Da S unentscheidbar ist, muß dies auch für $\mathcal{L}_P$ gelten ✓

– Falls $g \in S$ wähle ein beliebiges $h \notin S$ und zeige so $S \leq \overline{P}$ ✓

Anwendungen des Satzes von Rice

- $\boldsymbol{MON = \{i \mid \forall k.\ \varphi_i(k) < \varphi_i(k{+}1)\}}$
  - Monotone Funktionen sind unentscheidbar

Anwendungen des Satzes von Rice

- $\boldsymbol{MON = \{i \mid \forall k.\ \varphi_i(k) < \varphi_i(k{+}1)\}}$
  - Monotone Funktionen sind unentscheidbar
- $\boldsymbol{EF = \{i \mid \forall j.\ \varphi_i(j) \in \{0, 1\}\}}$
  - Entscheidungsfunktionen sind unentscheidbar

Anwendungen des Satzes von Rice

- $\boldsymbol{MON = \{i \mid \forall k.\ \varphi_i(k) < \varphi_i(k{+}1)\}}$
  - Monotone Funktionen sind unentscheidbar
- $\boldsymbol{EF = \{i \mid \forall j.\ \varphi_i(j) \in \{0,1\}\}}$
  - Entscheidungsfunktionen sind unentscheidbar
- $\boldsymbol{PROG_{spec} = \{i \mid \varphi_i}$ **erfüllt Spezifikation** $\boldsymbol{spec\}}$
  - Allgemeines Spezifikationsproblem ist unentscheidbar

Anwendungen des Satzes von Rice

- $\boldsymbol{MON = \{i \mid \forall k.\ \varphi_i(k) < \varphi_i(k{+}1)\}}$
  - Monotone Funktionen sind unentscheidbar
- $\boldsymbol{EF = \{i \mid \forall j.\ \varphi_i(j) \in \{0,1\}\}}$
  - Entscheidungsfunktionen sind unentscheidbar
- $\boldsymbol{PROG_{spec} = \{i \mid \varphi_i}$ **erfüllt Spezifikation** $\boldsymbol{spec\}}$
  - Allgemeines Spezifikationsproblem ist unentscheidbar
- $\boldsymbol{PROG_f = \{i \mid \varphi_i = f\}}$
  - Korrektheitsproblem ist unentscheidbar

Anwendungen des Satzes von Rice

- $\boldsymbol{MON = \{i \mid \forall k.\ \varphi_i(k) < \varphi_i(k{+}1)\}}$
  - Monotone Funktionen sind unentscheidbar
- $\boldsymbol{EF = \{i \mid \forall j.\ \varphi_i(j) \in \{0,1\}\}}$
  - Entscheidungsfunktionen sind unentscheidbar
- $\boldsymbol{PROG_{spec} = \{i \mid \varphi_i}$ **erfüllt Spezifikation** $\boldsymbol{spec\}}$
  - Allgemeines Spezifikationsproblem ist unentscheidbar
- $\boldsymbol{PROG_f = \{i \mid \varphi_i = f\}}$
  - Korrektheitsproblem ist unentscheidbar
- $\boldsymbol{EQ = \{(i,j) \mid \varphi_i = \varphi_j\}}$
  - Äquivalenzproblem ist unentscheidbar

Anwendungen des Satzes von Rice

- $\boldsymbol{MON = \{i \mid \forall k.\ \varphi_i(k) < \varphi_i(k+1)\}}$
  - Monotone Funktionen sind unentscheidbar
- $\boldsymbol{EF = \{i \mid \forall j.\ \varphi_i(j) \in \{0,1\}\}}$
  - Entscheidungsfunktionen sind unentscheidbar
- $\boldsymbol{PROG_{spec} = \{i \mid \varphi_i \text{ erfüllt Spezifikation } spec\}}$
  - Allgemeines Spezifikationsproblem ist unentscheidbar
- $\boldsymbol{PROG_f = \{i \mid \varphi_i = f\}}$
  - Korrektheitsproblem ist unentscheidbar
- $\boldsymbol{EQ = \{(i,j) \mid \varphi_i = \varphi_j\}}$
  - Äquivalenzproblem ist unentscheidbar
- $\boldsymbol{RG = \{(i,j) \mid j \in \mathrm{range}(\varphi_i)\}}$
  - Bildbereiche sind unentscheidbar

Anwendungen des Satzes von Rice

- $\boldsymbol{MON = \{i \mid \forall k.\ \varphi_i(k) < \varphi_i(k+1)\}}$
  - Monotone Funktionen sind unentscheidbar
- $\boldsymbol{EF = \{i \mid \forall j.\ \varphi_i(j) \in \{0,1\}\}}$
  - Entscheidungsfunktionen sind unentscheidbar
- $\boldsymbol{PROG_{spec} = \{i \mid \varphi_i \text{ erfüllt Spezifikation } spec\}}$
  - Allgemeines Spezifikationsproblem ist unentscheidbar
- $\boldsymbol{PROG_f = \{i \mid \varphi_i = f\}}$
  - Korrektheitsproblem ist unentscheidbar
- $\boldsymbol{EQ = \{(i,j) \mid \varphi_i = \varphi_j\}}$
  - Äquivalenzproblem ist unentscheidbar
- $\boldsymbol{RG = \{(i,j) \mid j \in \mathrm{range}(\varphi_i)\}}$
  - Bildbereiche sind unentscheidbar

Keine Programmeigenschaft kann getestet werden
Beweise müssen von Hand geführt werden
Rechnerunterstützung nur in Spezialfällen möglich