Theoretische Informatik II

Einheit 6.4

Andere Berechenbarkeitsmodelle

1. Typ-0 Berechenbarkeit
2. λ-Kalkül
3. Church'sche These

Typ-0 Berechenbarkeit

- $G = (N,T,P,\sigma)$ **Typ-0 Grammatik**
  - N, T: Alphabete (Nonterminalsymbole / Terminalsymbole)
  - P: Menge von Produktionen $\alpha \rightarrow \beta$ (α enthält Nonterminalsymbole)
  - σ: Startsymbol

Typ-0 Berechenbarkeit

- $G = (N,T,P,\sigma)$ **Typ-0 Grammatik**
  - N, T: Alphabete (Nonterminalsymbole / Terminalsymbole)
  - P: Menge von Produktionen $\alpha \rightarrow \beta$ (α enthält Nonterminalsymbole)
  - σ: Startsymbol
- **Ableitbarkeit** $\boldsymbol{v \rightarrow_G^* w}$
  - $v = u_1\alpha_1 z_1 \rightarrow u_1\beta_1 z_1 = u_2\alpha_2 z_2 \rightarrow u_2\beta_2 z_2 \rightarrow \ldots \rightarrow u_n\beta_n z_n = w$

Typ-0 Berechenbarkeit

- $G = (N,T,P,\sigma)$ **Typ-0 Grammatik**
  - N, T: Alphabete (Nonterminalsymbole / Terminalsymbole)
  - P: Menge von Produktionen $\alpha \to \beta$ (α enthält Nonterminalsymbole)
  - σ: Startsymbol
- **Ableitbarkeit** $\boldsymbol{v \to_G^* w}$
  - $v = u_1\alpha_1 z_1 \to u_1\beta_1 z_1 = u_2\alpha_2 z_2 \to u_2\beta_2 z_2 \to \ldots \to u_n\beta_n z_n = w$
- **Erzeugte Sprache** $\boldsymbol{L(G)} = \{w \in T^* \mid \sigma \to_G^* w\}$

Typ-0 Berechenbarkeit

- $G = (N,T,P,\sigma)$ **Typ-0 Grammatik**
  - N, T: Alphabete (Nonterminalsymbole / Terminalsymbole)
  - P: Menge von Produktionen $\alpha \to \beta$ (α enthält Nonterminalsymbole)
  - σ: Startsymbol
- **Ableitbarkeit** $\boldsymbol{v \to_G^* w}$
  - $v = u_1\alpha_1 z_1 \to u_1\beta_1 z_1 = u_2\alpha_2 z_2 \to u_2\beta_2 z_2 \to \ldots \to u_n\beta_n z_n = w$
- **Erzeugte Sprache** $\boldsymbol{L(G)} = \{w \in T^* \mid \sigma \to_G^* w\}$
- $\boldsymbol{L \subseteq T^*}$ **Typ-0 berechenbar**
  - L wird von einer Typ-0 Grammatik G erzeugt $(L = L(G))$
  - G "zählt Elemente von L auf"

Typ-0 Berechenbarkeit

- $G = (N,T,P,\sigma)$ **Typ-0 Grammatik**
  - N, T: Alphabete (Nonterminalsymbole / Terminalsymbole)
  - P: Menge von Produktionen $\alpha \to \beta$ (α enthält Nonterminalsymbole)
  - σ: Startsymbol
- **Ableitbarkeit** $\boldsymbol{v \to_G^* w}$
  - $v = u_1\alpha_1 z_1 \to u_1\beta_1 z_1 = u_2\alpha_2 z_2 \to u_2\beta_2 z_2 \to \ldots \to u_n\beta_n z_n = w$
- **Erzeugte Sprache** $\boldsymbol{L(G)} = \{w \in T^* \mid \sigma \to_G^* w\}$
- $\boldsymbol{L \subseteq T^*}$ **Typ-0 berechenbar**
  - L wird von einer Typ-0 Grammatik G erzeugt $(L = L(G))$
  - G "zählt Elemente von L auf"
- $\boldsymbol{f: X^* \to Y^*}$ **Typ-0 berechenbar**
  - $L_f = \{v\#w \mid f(v) = w\}$ Typ-0 berechenbar

L Typ-0 berechenbar $\Leftrightarrow$ L semi-entscheidbar Satz Q

$\Rightarrow$ **Simuliere Produktionsregeln der Grammatik G**

- Schreibe Wort w auf Band 1 und σ auf Band 2
- In Phase i schreibe alle Worte von $L(G)$ auf Band 2, die in i Schritten ableitbar sind und teste, ob w auf Band 2 vorkommt
- Im Erfolgsfall gebe 1 aus, andernfalls beginne Phase i+1
- Programm terminiert nicht, wenn $w \notin L$ (Ergebnis ist $\perp$)

L Typ-0 berechenbar $\Leftrightarrow$ L semi-entscheidbar Satz Q

$\Rightarrow$ Simuliere Produktionsregeln der Grammatik G

- Schreibe Wort w auf Band 1 und σ auf Band 2
- In Phase i schreibe alle Worte von $L(G)$ auf Band 2, die in i Schritten ableitbar sind und teste, ob w auf Band 2 vorkommt
- Im Erfolgsfall gebe 1 aus, andernfalls beginne Phase i+1
- Programm terminiert nicht, wenn $w \notin L$ (Ergebnis ist $\perp$)

$\Leftarrow$ Simuliere Abarbeitung der Turingmaschine τ

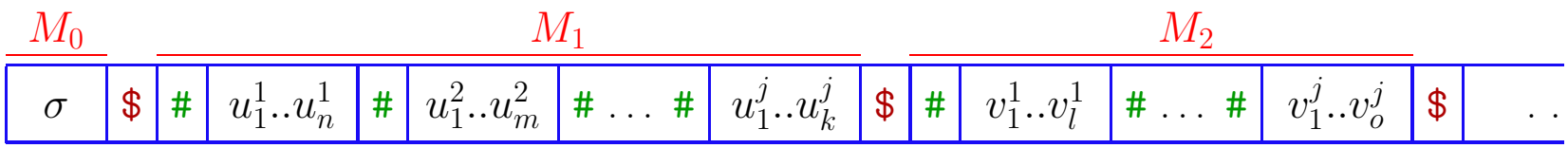
- Codiere Konfigurationen (s,u,v) als Worte usv
- Simuliere Konfigurationsübergänge als Regeln mit Begrenzern
- Entferne Begrenzer, wenn (Simulation von) τ angehalten hat

Beweis: Typ-0 berechenbar $\Rightarrow$ semi-entscheidbar

- **Für $G = (N,T,P,\sigma)$ definiere Mengen M_i**
  - $M_0 = \{\sigma\}$, $M_{i+1} = \{u | \exists v \in M_i \; v \rightarrow_G u\}$
  - M_i ist die Menge der in i Schritten ableitbaren Worte

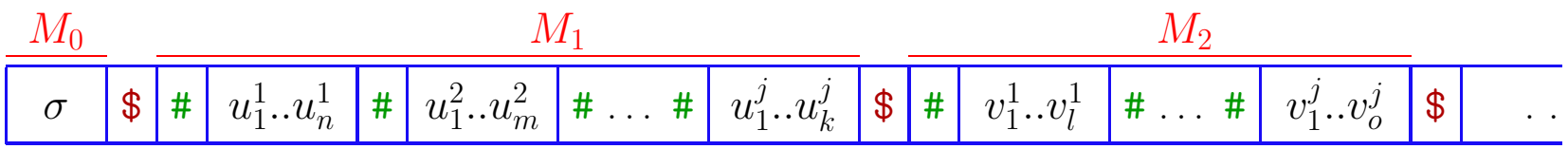
Beweis: Typ-0 berechenbar $\Rightarrow$ semi-entscheidbar

- **Für $G = (N,T,P,\sigma)$ definiere Mengen M_i**
  - $M_0 = \{\sigma\}$, $M_{i+1} = \{u | \exists v \in M_i \; v \rightarrow_G u\}$
  - M_i ist die Menge der in i Schritten ableitbaren Worte
- **Beschreibe Mehrband-TM für Test $w \in L(G)$**
  - $L(G) = \bigcup M_i \cap T^*$ also $w \in L(G) \Leftrightarrow \exists i \; w \in M_i$
  - Schreibe w auf Band 1, erzeuge M_i auf Band 2 und vergleiche
  - Trennung zwischen Worten bzw. M_i und M_{i+1} durch Symbole `#`, `$`



Beweis: Typ-0 berechenbar $\Rightarrow$ semi-entscheidbar

- **Für $G = (N,T,P,\sigma)$ definiere Mengen M_i**
  - $M_0 = \{\sigma\}$, $M_{i+1} = \{u | \exists v \in M_i \; v \rightarrow_G u\}$
  - M_i ist die Menge der in i Schritten ableitbaren Worte
- **Beschreibe Mehrband-TM für Test $w \in L(G)$**
  - $L(G) = \bigcup M_i \cap T^*$ also $w \in L(G) \Leftrightarrow \exists i \; w \in M_i$
  - Schreibe w auf Band 1, erzeuge M_i auf Band 2 und vergleiche
  - Trennung zwischen Worten bzw. M_i und M_{i+1} durch Symbole `#`, `$`



- **Iterative Erzeugung der M_i durch TM**
  - Codiere Produktionsregeln als Unterprogramme der Turingmaschine
  - Beginne mit $M_0 = \{\sigma\}$
  - Wende Regeln auf Worte von M_i an, schreibe Resultate ans Bandende
  - Beginne Vergleich mit w, wenn M_{i+1} vollständig erzeugt

Anmerkungen zum Beweis

- **TM berechnet partiell-charakteristische Funktion $\psi_{L(G)}$**
  - Erfolgreicher Vergleich bedeutet $w \in M_i \subseteq L(G)$ (Ergebnis 1)
  - $w \notin L(G)$ bedeutet $w \notin M_i$ für alle i: keine Terminierung (Ergebnis $\perp$)

Anmerkungen zum Beweis

- **TM berechnet partiell-charakteristische Funktion $\psi_{L(G)}$**
  - Erfolgreicher Vergleich bedeutet $w \in M_i \subseteq L(G)$ (Ergebnis 1)
  - $w \notin L(G)$ bedeutet $w \notin M_i$ für alle i: keine Terminierung (Ergebnis $\perp$)
- **Technische Details aufwendig**
  - Codierung der Regeln als Programme
  - Suchmechanismen: Anfang von M_i (`$`), nächstes Wort in M_i (`#`)
  - Erzeugung aller ableitbaren Worte durch Bestimmung aller anwendbaren Regeln (ggf. Anwendbarkeit einer Regel auf mehrere Teilworte)
  - Vergleich von w mit Worten auf Band 2, korrekte Ausgabe erzeugen

Anmerkungen zum Beweis

- **TM berechnet partiell-charakteristische Funktion $\psi_{L(G)}$**
  - Erfolgreicher Vergleich bedeutet $w \in M_i \subseteq L(G)$ (Ergebnis 1)
  - $w \notin L(G)$ bedeutet $w \notin M_i$ für alle i: keine Terminierung (Ergebnis $\perp$)
- **Technische Details aufwendig**
  - Codierung der Regeln als Programme
  - Suchmechanismen: Anfang von M_i (`$`), nächstes Wort in M_i (`#`)
  - Erzeugung aller ableitbaren Worte durch Bestimmung aller anwendbaren Regeln (ggf. Anwendbarkeit einer Regel auf mehrere Teilworte)
  - Vergleich von w mit Worten auf Band 2, korrekte Ausgabe erzeugen
- **Optimierungen möglich**
  - Erzeuge M_{i+1} auf drittem Band
  - Überschreibe M_i, wenn M_{i+1} vollständig erzeugt

$\vdots$

Anmerkungen zum Beweis

- **TM berechnet partiell-charakteristische Funktion $\psi_{L(G)}$**
  - Erfolgreicher Vergleich bedeutet $w \in M_i \subseteq L(G)$ (Ergebnis 1)
  - $w \notin L(G)$ bedeutet $w \notin M_i$ für alle i: keine Terminierung (Ergebnis $\perp$)
- **Technische Details aufwendig**
  - Codierung der Regeln als Programme
  - Suchmechanismen: Anfang von M_i (`$`), nächstes Wort in M_i (`#`)
  - Erzeugung aller ableitbaren Worte durch Bestimmung aller anwendbaren Regeln (ggf. Anwendbarkeit einer Regel auf mehrere Teilworte)
  - Vergleich von w mit Worten auf Band 2, korrekte Ausgabe erzeugen
- **Optimierungen möglich**
  - Erzeuge M_{i+1} auf drittem Band
  - Überschreibe M_i, wenn M_{i+1} vollständig erzeugt

$\vdots$

Projekt: Details heraussuchen, kurze Ausarbeitung schreiben

Beweis: semi-entscheidbar $\Rightarrow$ Typ-0 berechenbar

- **Zeige: Jede Funktion h_τ ist Typ-0 berechenbar**

Beweis: semi-entscheidbar $\Rightarrow$ Typ-0 berechenbar

- **Zeige: Jede Funktion h_τ ist Typ-0 berechenbar**
  - Allgemeinere Aussage: jede Turingmaschine kann simuliert werden

Beweis: semi-entscheidbar $\Rightarrow$ Typ-0 berechenbar

- **Zeige: Jede Funktion h_τ ist Typ-0 berechenbar**
  - Allgemeinere Aussage: jede Turingmaschine kann simuliert werden
  - Spezieller Beweis ist dann: L semi-entscheidbar

Beweis: semi-entscheidbar $\Rightarrow$ Typ-0 berechenbar

- **Zeige: Jede Funktion h_τ ist Typ-0 berechenbar**
  - Allgemeinere Aussage: jede Turingmaschine kann simuliert werden
  - Spezieller Beweis ist dann: L semi-entscheidbar

    $\Rightarrow$ ψ_L Turing-berechenbar

Beweis: semi-entscheidbar $\Rightarrow$ Typ-0 berechenbar

- **Zeige: Jede Funktion h_τ ist Typ-0 berechenbar**
  - Allgemeinere Aussage: jede Turingmaschine kann simuliert werden
  - Spezieller Beweis ist dann: L semi-entscheidbar

    $\Rightarrow$ ψ_L Turing-berechenbar

    $\Rightarrow$ $L_{\psi_L} = \{w\#1 | w \in L\}$ Typ-0 berechenbar

Beweis: semi-entscheidbar $\Rightarrow$ Typ-0 berechenbar

- **Zeige: Jede Funktion h_τ ist Typ-0 berechenbar**
  - Allgemeinere Aussage: jede Turingmaschine kann simuliert werden
  - Spezieller Beweis ist dann: L semi-entscheidbar

$$\Rightarrow \psi_L \text{ Turing-berechenbar}$$
$$\Rightarrow L_{\psi_L} = \{w\#1 | w \in L\} \text{ Typ-0 berechenbar}$$
$$\Rightarrow L = \{w | w \in L\} \text{ Typ-0 berechenbar}$$

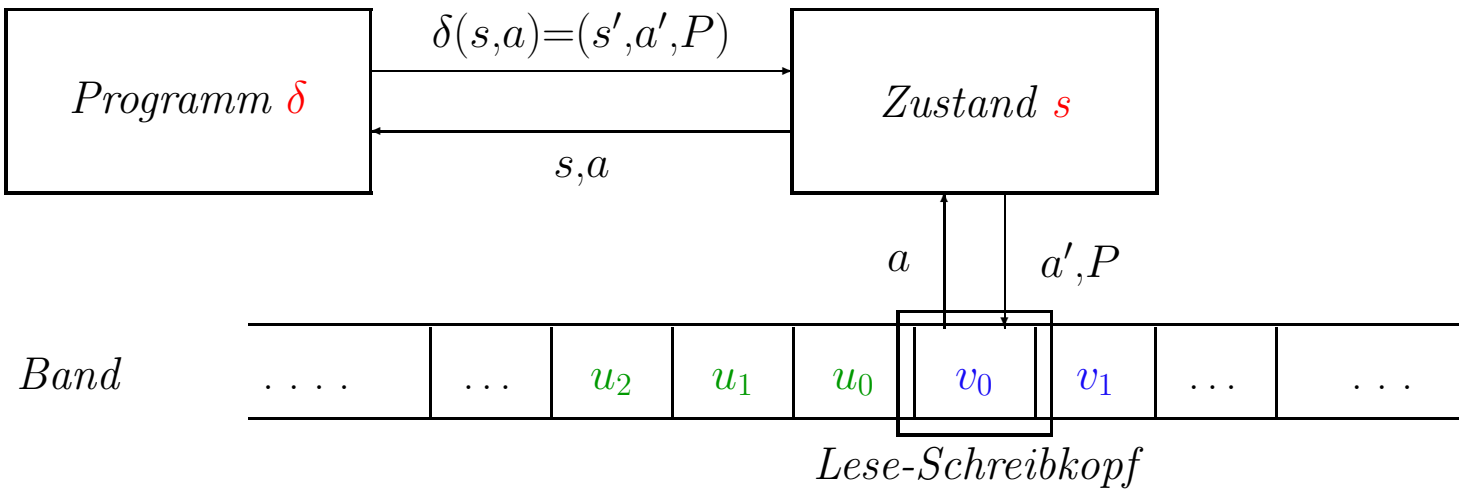
Beweis: semi-entscheidbar $\Rightarrow$ Typ-0 berechenbar

- **Zeige: Jede Funktion h_τ ist Typ-0 berechenbar**
  - Allgemeinere Aussage: jede Turingmaschine kann simuliert werden
  - Spezieller Beweis ist dann: L semi-entscheidbar
    $\Rightarrow$ ψ_L Turing-berechenbar
    $\Rightarrow$ $L_{\psi_L} = \{w\#1 | w \in L\}$ Typ-0 berechenbar
    $\Rightarrow$ $L = \{w | w \in L\}$ Typ-0 berechenbar
- **Idee: Generiere alle Konfigurationen von τ**
  - Erzeuge alle möglichen Eingabeworte und Anfangskonfigurationen
  - Codiere Konfigurationsübergänge von τ als Regeln
  - Simuliere Ausgabe durch Löschen von Nonterminalsymbolen

Beweis: semi-entscheidbar $\Rightarrow$ Typ-0 berechenbar

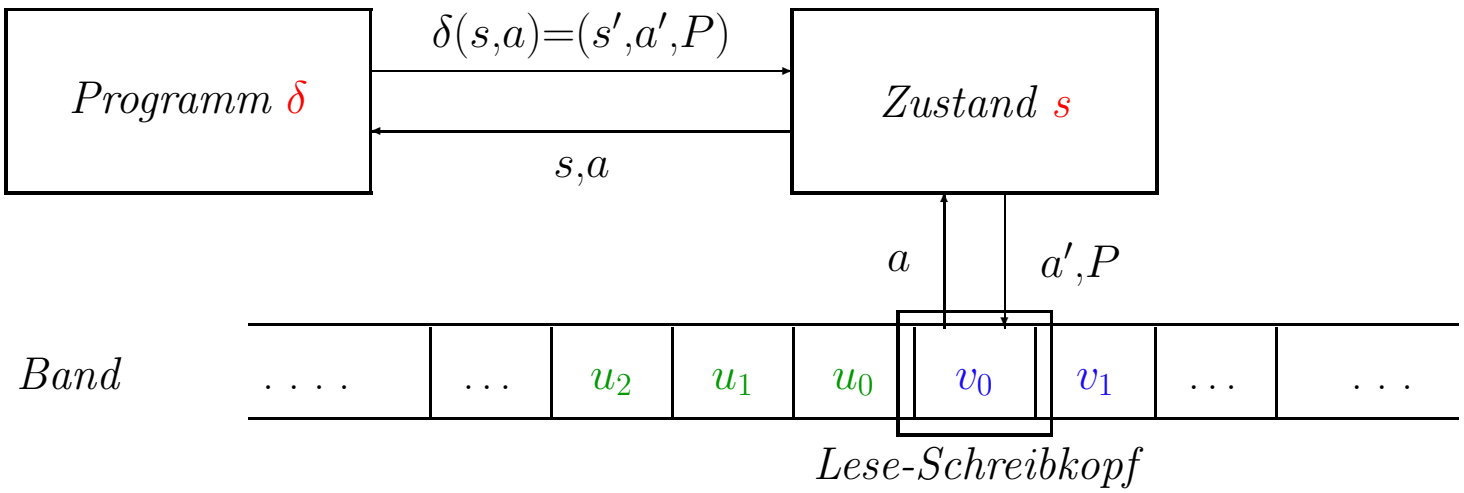
- **Zeige: Jede Funktion h_τ ist Typ-0 berechenbar**
  - Allgemeinere Aussage: jede Turingmaschine kann simuliert werden
  - Spezieller Beweis ist dann: L semi-entscheidbar
    $\Rightarrow$ ψ_L Turing-berechenbar
    $\Rightarrow$ $L_{\psi_L} = \{w\#1 | w \in L\}$ Typ-0 berechenbar
    $\Rightarrow$ $L = \{w | w \in L\}$ Typ-0 berechenbar
- **Idee: Generiere alle Konfigurationen von τ**
  - Erzeuge alle möglichen Eingabeworte und Anfangskonfigurationen
  - Codiere Konfigurationsübergänge von τ als Regeln
  - Simuliere Ausgabe durch Löschen von Nonterminalsymbolen
- **Abzuleitende Worte in simulierender Grammatik**
  - Worte der Form $w \# @ u\, s\, v \$$
  - w Eingabewort, (s,u,v) Konfiguration von τ bei Verarbeitung von w
  - @, \$ Trennsymbole

Rückblick: Konfigurationen in Wort-Darstellung



- **Konfigurationstripel** (s,u,v)
  - s aktueller Zustand, u, v String links/rechts vom Kopf

Rückblick: Konfigurationen in Wort-Darstellung



- **Konfigurationstripel (s,u,v)**
  - s aktueller Zustand, u, v String links/rechts vom Kopf
- **Konfigurationsübergänge**
  - $\delta(s,a) = (s',a',l)$ liefert $\hat{\delta}(s,u_0u,av) = (s',u,u_0a'v)$
  - $\delta(s,a) = (s',a',r)$ liefert $\hat{\delta}(s,u,av) = (s',a'u,v)$

Regeln der Grammatik

- **Erzeugung von Anfangskonfigurationen**
  - Regeln zur Erzeugung von Worten der Form $w\#@s_0w\$$ für $w \in X^*$

Regeln der Grammatik

- **Erzeugung von Anfangskonfigurationen**
  - Regeln zur Erzeugung von Worten der Form $w\#@s_0w\$$ für $w \in X^*$
- **Simulation der Konfigurationsübergänge**
  - Regeln der Form $s\,a\,v_1 \mapsto a'\,s'\,v_1$ für $v_1 \in \Gamma$ und $\delta(s,a) = (s',a',r)$
  - Regeln der Form $s\,a\,\$ \mapsto a'\,s'\,b\,\$$ für $\delta(s,a) = (s',a',r)$
  - Regeln der Form $u_0\,s\,a \mapsto s'\,u_0\,a'$ für $u_0 \in \Gamma$ und $\delta(s,a) = (s',a',l)$
  - Regeln der Form $@\,s\,a \mapsto @\,s'\,b\,a'$ für $\delta(s,a) = (s',a',l)$
  - Regeln der Form $s\,a \mapsto q_f\,a'$ für $\delta(s,a) = (s',a',h)$

Regeln der Grammatik

- **Erzeugung von Anfangskonfigurationen**
  - Regeln zur Erzeugung von Worten der Form $w\#@s_0w\$$ für $w \in X^*$
- **Simulation der Konfigurationsübergänge**
  - Regeln der Form $s\,a\,v_1 \mapsto a'\,s'\,v_1$ für $v_1 \in \Gamma$ und $\delta(s,a) = (s',a',r)$
  - Regeln der Form $s\,a\,\$ \mapsto a'\,s'\,b\,\$$ für $\delta(s,a) = (s',a',r)$
  - Regeln der Form $u_0\,s\,a \mapsto s'\,u_0\,a'$ für $u_0 \in \Gamma$ und $\delta(s,a) = (s',a',l)$
  - Regeln der Form $@\,s\,a \mapsto @\,s'\,b\,a'$ für $\delta(s,a) = (s',a',l)$
  - Regeln der Form $s\,a \mapsto q_f\,a'$ für $\delta(s,a) = (s',a',h)$
- **Schlußregeln (Ausgabe)**
  - Regeln zum Löschen der Symbole @, \$ im Kontext q_f
  - Regeln müssen q_f nach links und dann nach rechts schieben

Regeln der Grammatik

- **Erzeugung von Anfangskonfigurationen**
  - Regeln zur Erzeugung von Worten der Form $w\#@s_0w\$$ für $w \in X^*$
- **Simulation der Konfigurationsübergänge**
  - Regeln der Form $s\,a\,v_1 \mapsto a'\,s'\,v_1$ für $v_1 \in \Gamma$ und $\delta(s,a) = (s',a',r)$
  - Regeln der Form $s\,a\,\$ \mapsto a'\,s'\,b\,\$$ für $\delta(s,a) = (s',a',r)$
  - Regeln der Form $u_0\,s\,a \mapsto s'\,u_0\,a'$ für $u_0 \in \Gamma$ und $\delta(s,a) = (s',a',l)$
  - Regeln der Form $@\,s\,a \mapsto @\,s'\,b\,a'$ für $\delta(s,a) = (s',a',l)$
  - Regeln der Form $s\,a \mapsto q_f\,a'$ für $\delta(s,a) = (s',a',h)$
- **Schlußregeln (Ausgabe)**
  - Regeln zum Löschen der Symbole @, \$ im Kontext q_f
  - Regeln müssen q_f nach links und dann nach rechts schieben
- **Grammatik erzeugt die Sprache** $\{w\#v | h_\tau(w) = v\}$
  - Details z.B. in Erk-Priese, Seite 199–201

Grundlage funktionaler Programmiersprachen

Grundlage funktionaler Programmiersprachen

- **Einfacher mathematischer Mechanismus**
  - Funktionen werden definiert und angewandt
  - Die Beschreibung des Funktionsverhaltens ist der Name der Funktion
  - Funktionswerte werden ausgerechnet durch Einsetzen von Werten

Grundlage funktionaler Programmiersprachen

- **Einfacher mathematischer Mechanismus**
  - Funktionen werden definiert und angewandt
  - Die Beschreibung des Funktionsverhaltens ist der Name der Funktion
  - Funktionswerte werden ausgerechnet durch Einsetzen von Werten
- **Leicht zu verstehen**
  - Definition einer Funktion: `f(x) = 2*x+3`

Grundlage funktionaler Programmiersprachen

- **Einfacher mathematischer Mechanismus**
  - Funktionen werden definiert und angewandt
  - Die Beschreibung des Funktionsverhaltens ist der Name der Funktion
  - Funktionswerte werden ausgerechnet durch Einsetzen von Werten
- **Leicht zu verstehen**
  - Definition einer Funktion: `f(x) = 2*x+3`
  - Auswertung der Funktion: `f(4) = 2*4+3 = 11`

Grundlage funktionaler Programmiersprachen

- **Einfacher mathematischer Mechanismus**
  - Funktionen werden definiert und angewandt
  - Die Beschreibung des Funktionsverhaltens ist der Name der Funktion
  - Funktionswerte werden ausgerechnet durch Einsetzen von Werten
- **Leicht zu verstehen**
  - Definition einer Funktion: $\texttt{f} \mathrel{\hat{=}} \texttt{x} \mapsto \texttt{2*x+3}$
  - Auswertung der Funktion: `f(4) = 2*4+3 = 11`

  Name der Funktion ist irrelevant

Grundlage funktionaler Programmiersprachen

- **Einfacher mathematischer Mechanismus**
  - Funktionen werden definiert und angewandt
  - Die Beschreibung des Funktionsverhaltens ist der Name der Funktion
  - Funktionswerte werden ausgerechnet durch Einsetzen von Werten
- **Leicht zu verstehen**
  - Definition einer Funktion: `f` $\hat{=}$ `\x. 2*x+3` Abstraktion von `x`
  - Auswertung der Funktion: `f(4) = 2*4+3 = 11`

Name der Funktion ist irrelevant

Grundlage funktionaler Programmiersprachen

- **Einfacher mathematischer Mechanismus**
  - Funktionen werden definiert und angewandt
  - Die Beschreibung des Funktionsverhaltens ist der Name der Funktion
  - Funktionswerte werden ausgerechnet durch Einsetzen von Werten
- **Leicht zu verstehen**
  - Definition einer Funktion: `f` $\hat{=}$ λ`x. 2*x+3` λ-Notation
  - Auswertung der Funktion:

Name der Funktion ist irrelevant

Grundlage funktionaler Programmiersprachen

- **Einfacher mathematischer Mechanismus**
  - Funktionen werden definiert und angewandt
  - Die Beschreibung des Funktionsverhaltens ist der Name der Funktion
  - Funktionswerte werden ausgerechnet durch Einsetzen von Werten
- **Leicht zu verstehen**
  - Definition einer Funktion: `f` $\hat{=}$ `λx. 2*x+3` — λ-Notation
  - Auswertung der Funktion: `(λx. 2*x+3)(4)` — Applikation

Name der Funktion ist irrelevant

Grundlage funktionaler Programmiersprachen

- **Einfacher mathematischer Mechanismus**
  - Funktionen werden definiert und angewandt
  - Die Beschreibung des Funktionsverhaltens ist der Name der Funktion
  - Funktionswerte werden ausgerechnet durch Einsetzen von Werten
- **Leicht zu verstehen**
  - Definition einer Funktion: `f` $\hat{=}$ `λx. 2*x+3` — λ-Notation
  - Auswertung der Funktion: `(λx. 2*x+3)(4)` $\xrightarrow{\beta}$ 11 — Applikation + Reduktion

Name der Funktion ist irrelevant

λ-Kalkül – Syntax und Semantik

- **λ-Terme**
  - Variablen x
  - $\lambda x\,.\,t$, wobei x Variable und t λ-Term — λ-Abstraktion
    Vorkommen von x in t werden gebunden
  - $f\ t$, wobei t und f λ-Terme — Applikation
  - (t), wobei t λ-Term

λ-Kalkül – Syntax und Semantik

- **λ-Terme**
  - Variablen x
  - $\lambda x\,.\,t$, wobei x Variable und t λ-Term — λ-Abstraktion
    Vorkommen von x in t werden gebunden
  - $f\ t$, wobei t und f λ-Terme — Applikation
  - (t), wobei t λ-Term
- **Konventionen**
  - Applikation bindet stärker als λ-Abstraktion
  - Applikation ist links-assoziativ: $f\ t_1\ t_2 \mathrel{\hat{=}} (f\ t_1)\ t_2$
  - Notation $f(t_1, \ldots, t_n)$ entspricht iterierter Applikation $f\ t_1\ \ldots t_n$

λ-Kalkül – Syntax und Semantik

- **λ-Terme**
  - Variablen x
  - $\lambda x\,.\,t$, wobei x Variable und t λ-Term — λ-Abstraktion
    Vorkommen von x in t werden gebunden
  - $f\ t$, wobei t und f λ-Terme — Applikation
  - (t), wobei t λ-Term
- **Konventionen**
  - Applikation bindet stärker als λ-Abstraktion
  - Applikation ist links-assoziativ: $f\ t_1\ t_2 \mathrel{\hat{=}} (f\ t_1)\ t_2$
  - Notation $f(t_1,\ldots,t_n)$ entspricht iterierter Applikation $f\ t_1\ \ldots t_n$
- **Auswertung von λ-Termen**
  - Ersetze Funktionsparameter durch Funktionsargumente
  - Reduktion $(\lambda x\,.\,t)(b) \xrightarrow{\beta} t[b/x]$
  - Substitution $t[b/x]$: ersetze freie Vorkommen von x in t durch b

Substitution und Reduktion am Beispiel

$$(\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x)$$

Substitution und Reduktion am Beispiel

$$(\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x)$$

$$\longrightarrow \quad \lfloor \lambda f.\lambda x.\ n\ f\ (f\ x) \rfloor \left[\lambda f.\lambda x.x \,/\, n\right]$$

Substitution und Reduktion am Beispiel

$$(\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x}))\ (\lambda \mathtt{f}.\lambda \mathtt{x}.\mathtt{x})$$

$$\longrightarrow \quad \lambda \mathtt{f}.\lfloor \lambda \mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x}) \rfloor \left[\lambda \mathtt{f}.\lambda \mathtt{x}.\mathtt{x} \,/\, \mathtt{n}\right]$$

Substitution und Reduktion am Beispiel

$$(\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x)$$

$$\longrightarrow \quad \lambda f.\lambda x.\ \lfloor n\ f\ (f\ x) \rfloor \left[\lambda f.\lambda x.x \,/\, n\right]$$

Substitution und Reduktion am Beispiel

$$(\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x)$$

$$\longrightarrow \quad \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x)$$

Substitution und Reduktion am Beispiel

$$(\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x)$$

$$\longrightarrow \quad \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x)$$

Substitution und Reduktion am Beispiel

$$(\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x)$$

$$\longrightarrow \quad \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x)$$

Substitution und Reduktion am Beispiel

$$(\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x)$$

$$\longrightarrow \quad \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x)$$

Substitution und Reduktion am Beispiel

$$(\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x)$$

$$\longrightarrow \quad \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x)$$

$$\longrightarrow \quad \lfloor \lambda x.x \rfloor [f\,/\,f]$$

Substitution und Reduktion am Beispiel

$$(\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x)$$

$$\longrightarrow \quad \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x)$$

$$\longrightarrow \quad \lambda x.x$$

Substitution und Reduktion am Beispiel

$$(\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x)$$

$$\longrightarrow\quad \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x)$$

$$\longrightarrow\quad (\lambda x.x)\ (f\ x)$$

Substitution und Reduktion am Beispiel

$$(\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x)$$

$$\longrightarrow\quad \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x)$$

$$\longrightarrow\quad \lambda x.(\lambda x.x)\ (f\ x)$$

Substitution und Reduktion am Beispiel

$$(\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x)$$

$$\longrightarrow\quad \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x)$$

$$\longrightarrow\quad \lambda f.\lambda x.(\lambda x.x)\ (f\ x)$$

Substitution und Reduktion am Beispiel

$$(\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x)$$

$$\longrightarrow \quad \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x)$$

$$\longrightarrow \quad \lambda f.\lambda x.(\lambda x.x)\ (f\ x)$$

Substitution und Reduktion am Beispiel

$$(\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x)$$

$$\longrightarrow\quad \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x)$$

$$\longrightarrow\quad \lambda f.\lambda x.(\lambda x.x)\ (f\ x)$$

Substitution und Reduktion am Beispiel

$$
\begin{aligned}
& (\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x) \\
\longrightarrow\quad & \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x) \\
\longrightarrow\quad & \lambda f.\lambda x.(\lambda x.x)\ (f\ x) \\
\longrightarrow\quad & \lfloor x \rfloor [f\ x / x]
\end{aligned}
$$

Substitution und Reduktion am Beispiel

$$
\begin{aligned}
& (\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x) \\
\longrightarrow\quad & \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x) \\
\longrightarrow\quad & \lambda f.\lambda x.(\lambda x.x)\ (f\ x) \\
\longrightarrow\quad & f\ x
\end{aligned}
$$

Substitution und Reduktion am Beispiel

$$
\begin{aligned}
& (\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x) \\
\longrightarrow\quad & \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x) \\
\longrightarrow\quad & \lambda f.\lambda x.(\lambda x.x)\ (f\ x) \\
\longrightarrow\quad & \lambda x.\ f\ x
\end{aligned}
$$

Substitution und Reduktion am Beispiel

$$
\begin{aligned}
& (\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x) \\
\longrightarrow\quad & \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x) \\
\longrightarrow\quad & \lambda f.\lambda x.(\lambda x.x)\ (f\ x) \\
\longrightarrow\quad & \lambda f.\lambda x.\ f\ x
\end{aligned}
$$

Substitution und Reduktion am Beispiel

$$
\begin{aligned}
& (\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x) \\
\longrightarrow\quad & \lambda f.\lambda x.\ (\lambda f.\lambda x.x)\ f\ (f\ x) \\
\longrightarrow\quad & \lambda f.\lambda x.(\lambda x.x)\ (f\ x) \\
\longrightarrow\quad & \lambda f.\lambda x.\ f\ x
\end{aligned}
$$

$$\Downarrow$$

$$(\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.x) \xrightarrow{3} \lambda f.\lambda x.\ f\ x$$

Darstellung Boolescher Operatoren im λ-Kalkül

$$\begin{array}{lcl} \textsf{T} & \equiv & \lambda \mathtt{x}\,.\,\lambda \mathtt{y}\,.\,\mathtt{x} \\ \textsf{F} & \equiv & \lambda \mathtt{x}\,.\,\lambda \mathtt{y}\,.\,\mathtt{y} \\ \textsf{if } b \textsf{ then } s \textsf{ else } t & \equiv & b\,s\,t \end{array}$$

Darstellung Boolescher Operatoren im λ-Kalkül

$$\begin{aligned} \mathsf{T} &\equiv \lambda x.\lambda y.x \\ \mathsf{F} &\equiv \lambda x.\lambda y.y \\ \textsf{if } b \textsf{ then } s \textsf{ else } t &\equiv b\,s\,t \end{aligned}$$

Konditional ist invers zu T und F

$$\begin{aligned} & \textsf{if T then } s \textsf{ else } t \\ \equiv\ & \mathsf{T}\,s\,t \\ \equiv\ & (\lambda x.\lambda y.x)\,s\,t \\ \longrightarrow\ & (\lambda y.s)\,t \\ \longrightarrow\ & s \end{aligned}$$

Darstellung Boolescher Operatoren im λ-Kalkül

$$\begin{aligned} \mathsf{T} &\equiv \lambda x.\lambda y.x \\ \mathsf{F} &\equiv \lambda x.\lambda y.y \\ \textsf{if } b \textsf{ then } s \textsf{ else } t &\equiv b\,s\,t \end{aligned}$$

Konditional ist invers zu T und F

$$\begin{aligned} & \textsf{if T then } s \textsf{ else } t \\ \equiv\ & \mathsf{T}\,s\,t \\ \equiv\ & (\lambda x.\lambda y.x)\,s\,t \\ \longrightarrow\ & (\lambda y.s)\,t \\ \longrightarrow\ & s \end{aligned}$$

$$\begin{aligned} & \textsf{if F then } s \textsf{ else } t \\ \equiv\ & \mathsf{F}\,s\,t \\ \equiv\ & (\lambda x.\lambda y.y)\,s\,t \\ \longrightarrow\ & (\lambda y.y)\,t \\ \longrightarrow\ & t \end{aligned}$$

Bildung und Analyse von Paaren

$$\langle s\,,t\rangle \equiv \lambda \mathtt{p}.\ \mathtt{p}\,s\,t$$

$$pair.\mathbf{1} \equiv pair\ (\lambda \mathtt{x}.\lambda \mathtt{y}.\mathtt{x})$$

$$pair.\mathbf{2} \equiv pair\ (\lambda \mathtt{x}.\lambda \mathtt{y}.\mathtt{y})$$

$$\mathbf{let}\ \langle x\,,y\rangle = pair\ \mathbf{in}\ t \equiv pair\ (\lambda x.\lambda y.t)$$

Bildung und Analyse von Paaren

$$
\begin{array}{lcl}
\langle s\,,t\rangle & \equiv & \lambda \mathtt{p}.\ \ \mathtt{p}\,s\,t \\
pair.\mathbf{1} & \equiv & pair\ \ (\lambda \mathtt{x}.\lambda \mathtt{y}.\mathtt{x}) \\
pair.\mathbf{2} & \equiv & pair\ \ (\lambda \mathtt{x}.\lambda \mathtt{y}.\mathtt{y}) \\
\mathbf{let}\ \langle x\,,y\rangle = pair\ \mathbf{in}\ t & \equiv & pair\ \ (\lambda x.\lambda y.t)
\end{array}
$$

Analyseoperator ist invers zur Paarbildung

$$
\begin{array}{rcl}
 & & \mathbf{let}\ \langle x\,,y\rangle = \langle u\,,v\rangle\ \mathbf{in}\ t \\
 & \equiv & \langle u\,,v\rangle\,(\lambda x.\lambda y.t) \\
 & \equiv & (\lambda \mathtt{p}.\ \ \mathtt{p}\,u\,v)\,(\lambda x.\lambda y.t) \\
 & \longrightarrow & (\lambda x.\lambda y.t)\,u\,v \\
 & \longrightarrow & (\lambda y.t[u/x])\,u\,v \\
 & \longrightarrow & t[u,v/x,y])
\end{array}
$$

λ-berechenbare Funktionen auf $\mathbb{N}$

- **Darstellung von Zahlen durch iterierte Terme**
  - Semantisch: wiederholte Anwendung von Funktionen

λ-berechenbare Funktionen auf $\mathbb{N}$

- **Darstellung von Zahlen durch iterierte Terme**
  - Semantisch: wiederholte Anwendung von Funktionen
  - Repräsentiere die Zahl n durch den Term $\underbrace{\lambda \texttt{f}.\lambda \texttt{x}.\ \texttt{f}\ (\texttt{f}\ .\ .\ (\texttt{f}\ \texttt{t})\ .\ .)}_{n\text{-}mal}$

λ-berechenbare Funktionen auf $\mathbb{N}$

- **Darstellung von Zahlen durch iterierte Terme**
  - Semantisch: wiederholte Anwendung von Funktionen
  - Repräsentiere die Zahl n durch den Term $\underbrace{\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}\,(\mathtt{f}\,..(\mathtt{f}\,\mathtt{t})..)}_{n\text{-}mal}$
  - Notation: $\overline{\boldsymbol{n}} \quad \equiv \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^n\,\mathtt{x}$

λ-berechenbare Funktionen auf $\mathbb{N}$

- **Darstellung von Zahlen durch iterierte Terme**
  - Semantisch: wiederholte Anwendung von Funktionen
  - Repräsentiere die Zahl n durch den Term $\underbrace{\lambda \mathtt{f}.\lambda \mathtt{x}.\, \mathtt{f}\,(\mathtt{f}\,..(\mathtt{f}\,\mathtt{t})..)}_{n\text{-}mal}$
  - Notation: $\overline{\boldsymbol{n}} \equiv \lambda \mathtt{f}.\lambda \mathtt{x}.\, \mathtt{f}^n\,\mathtt{x}$
  - Bezeichnung: **Church Numerals**

λ-berechenbare Funktionen auf $\mathbb{N}$

- **Darstellung von Zahlen durch iterierte Terme**
  - Semantisch: wiederholte Anwendung von Funktionen
  - Repräsentiere die Zahl n durch den Term $\underbrace{\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}\ (\mathtt{f}\ .\ .\ (\mathtt{f}\ \mathtt{t})\ .\ .)}_{n\text{-}mal}$
  - Notation: $\overline{\boldsymbol{n}} \equiv \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^n\ \mathtt{x}$
  - Bezeichnung: **Church Numerals**
- $\boldsymbol{f}$**:**$\mathbb{N}^{\boldsymbol{n}} \longrightarrow \mathbb{N}$ **λ-berechenbar:**
  - Es gibt einen λ-Term t mit $f(x_1, ..., x_n) = m \Leftrightarrow t\ \overline{x_1}\ \ldots \overline{x_n} = \overline{m}$

λ-berechenbare Funktionen auf $\mathbb{N}$

- **Darstellung von Zahlen durch iterierte Terme**
  - Semantisch: wiederholte Anwendung von Funktionen
  - Repräsentiere die Zahl n durch den Term $\underbrace{\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}\ (\mathtt{f}\ .\ .\ (\mathtt{f}\ \mathtt{t})\ .\ .)}_{n\text{-}mal}$
  - Notation: $\overline{\boldsymbol{n}} \equiv \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^n\ \mathtt{x}$
  - Bezeichnung: **Church Numerals**
- $\boldsymbol{f}$**:**$\mathbb{N}^{\boldsymbol{n}} \longrightarrow \mathbb{N}$ **λ-berechenbar:**
  - Es gibt einen λ-Term t mit $f(x_1, ..., x_n) = m \Leftrightarrow t\ \overline{x_1}\ \ldots \overline{x_n} = \overline{m}$
- **Operationen müssen Termvielfachheit verändern**
  - z.B. **add** $\overline{\boldsymbol{m}}$ $\overline{\boldsymbol{n}}$ muß als Wert immer den Term $\overline{\boldsymbol{m+n}}$ ergeben

Programmierung im λ-Kalkül

- **Nachfolgerfunktion:** $\mathtt{s} \equiv \lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x})$

Programmierung im λ-Kalkül

- **Nachfolgerfunktion:** $\mathtt{s} \equiv \lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x})$
  - Zeige: Der Wert von $\mathtt{s}\ \overline{n}$ ist der Term $\overline{n+1}$

Programmierung im λ-Kalkül

- **Nachfolgerfunktion:** $\mathtt{s} \equiv \lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x})$
  - Zeige: Der Wert von $\mathtt{s}\ \overline{n}$ ist der Term $\overline{n+1}$

$$\mathtt{s}\ \overline{n} \quad \equiv \quad (\lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x}))\ (\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})$$

Programmierung im λ-Kalkül

- **Nachfolgerfunktion:** $\mathtt{s} \equiv \lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x})$
  - Zeige: Der Wert von $\mathtt{s}\ \overline{n}$ ist der Term $\overline{n+1}$

$$\begin{aligned}\mathtt{s}\ \overline{n} \quad &\equiv \quad (\lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x}))\ (\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x}) \\ &\longrightarrow\ \lambda\mathtt{f}.\lambda\mathtt{x}.\ (\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x})\end{aligned}$$

Programmierung im λ-Kalkül

- **Nachfolgerfunktion:** $\mathtt{s} \equiv \lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x})$
  - Zeige: Der Wert von $\mathtt{s}\ \overline{n}$ ist der Term $\overline{n+1}$

$$
\begin{aligned}
\mathtt{s}\ \overline{n} \quad &\equiv \quad (\lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x}))\ (\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ (\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ (\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ (\mathtt{f}\ \mathtt{x})
\end{aligned}
$$

Programmierung im λ-Kalkül

- **Nachfolgerfunktion:** $\mathtt{s} \equiv \lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x})$
  - Zeige: Der Wert von $\mathtt{s}\ \overline{n}$ ist der Term $\overline{n+1}$

$$
\begin{aligned}
\mathtt{s}\ \overline{n} \quad &\equiv \quad (\lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x}))\ (\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ (\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ (\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ (\mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\ (\mathtt{f}\ \mathtt{x})
\end{aligned}
$$

Programmierung im λ-Kalkül

- **Nachfolgerfunktion:** $\mathtt{s} \equiv \lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x})$
  - Zeige: Der Wert von $\mathtt{s}\ \overline{n}$ ist der Term $\overline{n+1}$

$$
\begin{aligned}
\mathtt{s}\ \overline{n} \quad &\equiv \quad (\lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x}))\ (\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ (\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ (\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ (\mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\ (\mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^{n+1}\ \mathtt{x}
\end{aligned}
$$

Programmierung im λ-Kalkül

- **Nachfolgerfunktion:** $\mathtt{s} \equiv \lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x})$
  - Zeige: Der Wert von $\mathtt{s}\ \overline{n}$ ist der Term $\overline{n+1}$

$$
\begin{aligned}
\mathtt{s}\ \overline{n} \quad &\equiv \quad (\lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x}))\ (\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ (\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ (\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ (\mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\ (\mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^{n+1}\ \mathtt{x} \qquad\qquad \equiv \quad \overline{n+1}
\end{aligned}
$$

Programmierung im λ-Kalkül

- **Nachfolgerfunktion:** $\mathbf{s} \equiv \lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x)$
  - Zeige: Der Wert von $\mathbf{s}\ \overline{n}$ ist der Term $\overline{n+1}$

$$
\begin{aligned}
\mathbf{s}\ \overline{n} \quad &\equiv \quad (\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.\ f^n\, x) \\
&\longrightarrow \lambda f.\lambda x.\ (\lambda f.\lambda x.\ f^n\, x)\ f\ (f\ x) \\
&\longrightarrow \lambda f.\lambda x.\ (\lambda x.\ f^n\, x)\ (f\ x) \\
&\longrightarrow \lambda f.\lambda x.\ f^n\ (f\ x) \\
&\longrightarrow \lambda f.\lambda x.\ f^{n+1}\ x \qquad\qquad \equiv \quad \overline{n+1}
\end{aligned}
$$

- **Addition:** $\mathbf{add} \equiv \lambda m.\lambda n.\lambda f.\lambda x.\ m\ f\ (n\ f\ x)$

Programmierung im λ-Kalkül

- **Nachfolgerfunktion:** $\mathtt{s} \equiv \lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x})$
  - Zeige: Der Wert von $\mathtt{s}\ \overline{n}$ ist der Term $\overline{n+1}$

$$
\begin{aligned}
\mathtt{s}\ \overline{n} \quad &\equiv \quad (\lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x}))\ (\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ (\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ (\lambda\mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ (\mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^n\ (\mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{f}^{n+1}\ \mathtt{x} \qquad\qquad \equiv \quad \overline{n+1}
\end{aligned}
$$

- **Addition:** $\mathtt{add} \equiv \lambda\mathtt{m}.\lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x})$
- **Multiplikation:** $\mathtt{mul} \equiv \lambda\mathtt{m}.\lambda\mathtt{n}.\lambda\mathtt{f}.\lambda\mathtt{x}.\ \mathtt{m}\ (\mathtt{n}\ \mathtt{f})\ \mathtt{x}$

Programmierung im λ-Kalkül

- **Nachfolgerfunktion:** $\mathsf{s} \equiv \lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x)$
  - Zeige: Der Wert von $\mathsf{s}\ \overline{n}$ ist der Term $\overline{n+1}$

$$
\begin{aligned}
\mathsf{s}\ \overline{n} \quad &\equiv \quad (\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.\ f^n\, x) \\
&\longrightarrow \lambda f.\lambda x.\ (\lambda f.\lambda x.\ f^n\, x)\ f\ (f\ x) \\
&\longrightarrow \lambda f.\lambda x.\ (\lambda x.\ f^n\, x)\ (f\ x) \\
&\longrightarrow \lambda f.\lambda x.\ f^n\ (f\ x) \\
&\longrightarrow \lambda f.\lambda x.\ f^{n+1}\ x \qquad\qquad \equiv \quad \overline{n+1}
\end{aligned}
$$

- **Addition:** $\mathsf{add} \equiv \lambda m.\lambda n.\lambda f.\lambda x.\ m\ f\ (n\ f\ x)$
- **Multiplikation:** $\mathsf{mul} \equiv \lambda m.\lambda n.\lambda f.\lambda x.\ m\ (n\ f)\ x$
- **Test auf Null:** $\mathsf{zero} \equiv \lambda n.\ n\ (\lambda n.\mathsf{F})\ \mathsf{T}$

Programmierung im λ-Kalkül

- **Nachfolgerfunktion:** $\mathsf{s} \equiv \lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x)$
  - Zeige: Der Wert von $\mathsf{s}\ \overline{n}$ ist der Term $\overline{n+1}$

$$
\begin{aligned}
\mathsf{s}\ \overline{n} \quad &\equiv \quad (\lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x))\ (\lambda f.\lambda x.\ f^n\, x) \\
&\longrightarrow \lambda f.\lambda x.\ (\lambda f.\lambda x.\ f^n\, x)\ f\ (f\ x) \\
&\longrightarrow \lambda f.\lambda x.\ (\lambda x.\ f^n\, x)\ (f\ x) \\
&\longrightarrow \lambda f.\lambda x.\ f^n\ (f\ x) \\
&\longrightarrow \lambda f.\lambda x.\ f^{n+1}\ x \qquad\qquad \equiv \quad \overline{n+1}
\end{aligned}
$$

- **Addition:** $\mathsf{add} \equiv \lambda m.\lambda n.\lambda f.\lambda x.\ m\ f\ (n\ f\ x)$
- **Multiplikation:** $\mathsf{mul} \equiv \lambda m.\lambda n.\lambda f.\lambda x.\ m\ (n\ f)\ x$
- **Test auf Null:** $\mathsf{zero} \equiv \lambda n.\ n\ (\lambda n.\mathsf{F})\ \mathsf{T}$
- **Vorgängerfunktion:**

  $\mathsf{p} \equiv \lambda n.\ (n\ (\lambda f x.\ \langle \mathsf{s}, \mathsf{let}\ \langle f, x\rangle = f x\ \mathsf{in}\ f\ x\rangle)\ \langle \lambda z.\overline{0}, \overline{0}\rangle).2$

Auswertung der Additionsfunktion

- **Zeige: add $\overline{m}$ $\overline{n}$ reduziert zu $\overline{m+n}$**

Auswertung der Additionsfunktion

- **Zeige: add $\overline{m}$ $\overline{n}$ reduziert zu $\overline{m+n}$**

$$\textbf{add}\ \overline{m}\ \overline{n} \quad \equiv \quad (\lambda \mathtt{m}.\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{m}\ \overline{n}$$

Auswertung der Additionsfunktion

- **Zeige: add $\overline{m}$ $\overline{n}$ reduziert zu $\overline{m+n}$**

$$\begin{aligned} \textbf{add}\ \overline{m}\ \overline{n} &\equiv (\lambda \mathtt{m}.\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{m}\ \overline{n} \\ &\longrightarrow (\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{n} \end{aligned}$$

Auswertung der Additionsfunktion

- **Zeige: add $\overline{m}$ $\overline{n}$ reduziert zu $\overline{m+n}$**

$$
\begin{aligned}
\textbf{add}\ \overline{m}\ \overline{n} \quad &\equiv \quad (\lambda \mathtt{m}.\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{m}\ \overline{n} \\
&\longrightarrow \quad (\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{n} \\
&\longrightarrow \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\overline{n}\ \mathtt{f}\ \mathtt{x})
\end{aligned}
$$

Auswertung der Additionsfunktion

- **Zeige: add $\overline{m}$ $\overline{n}$ reduziert zu $\overline{m+n}$**

$$
\begin{aligned}
\textbf{add}\ \overline{m}\ \overline{n} \quad &\equiv \quad (\lambda \mathtt{m}.\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{m}\ \overline{n} \\
&\longrightarrow \quad (\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{n} \\
&\longrightarrow \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\equiv \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ (\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,\mathtt{x})\ \mathtt{f}\ (\overline{n}\ \mathtt{f}\ \mathtt{x})
\end{aligned}
$$

Auswertung der Additionsfunktion

- **Zeige: add $\overline{m}$ $\overline{n}$ reduziert zu $\overline{m+n}$**

$$
\begin{aligned}
\textbf{add}\ \overline{m}\ \overline{n} \quad &\equiv \quad (\lambda \mathtt{m}.\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{m}\ \overline{n} \\
&\longrightarrow \quad (\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{n} \\
&\longrightarrow \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\equiv \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ (\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,\mathtt{x})\ \mathtt{f}\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ (\lambda \mathtt{x}.\ \mathtt{f}^m\,\mathtt{x})\ (\overline{n}\ \mathtt{f}\ \mathtt{x})
\end{aligned}
$$

Auswertung der Additionsfunktion

- **Zeige: add $\overline{m}$ $\overline{n}$ reduziert zu $\overline{m+n}$**

$$
\begin{aligned}
\textbf{add}\ \overline{m}\ \overline{n} \quad &\equiv \quad (\lambda \mathtt{m}.\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{m}\ \overline{n} \\
&\longrightarrow \quad (\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{n} \\
&\longrightarrow \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\equiv \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ (\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,\mathtt{x})\ \mathtt{f}\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ (\lambda \mathtt{x}.\ \mathtt{f}^m\,\mathtt{x})\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,(\overline{n}\ \mathtt{f}\ \mathtt{x})
\end{aligned}
$$

Auswertung der Additionsfunktion

- **Zeige: add $\overline{m}$ $\overline{n}$ reduziert zu $\overline{m+n}$**

$$
\begin{aligned}
\textbf{add}\ \overline{m}\ \overline{n} &\equiv (\lambda \mathtt{m}.\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{m}\ \overline{n} \\
&\longrightarrow (\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{n} \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\equiv \lambda \mathtt{f}.\lambda \mathtt{x}.\ (\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,\mathtt{x})\ \mathtt{f}\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ (\lambda \mathtt{x}.\ \mathtt{f}^m\,\mathtt{x})\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,(\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\equiv \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,((\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ \mathtt{f}\ \mathtt{x})
\end{aligned}
$$

Auswertung der Additionsfunktion

- **Zeige: add $\overline{m}$ $\overline{n}$ reduziert zu $\overline{m+n}$**

$$
\begin{aligned}
\textbf{add}\ \overline{m}\ \overline{n} &\equiv (\lambda \mathtt{m}.\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{m}\ \overline{n} \\
&\longrightarrow (\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{n} \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\equiv \lambda \mathtt{f}.\lambda \mathtt{x}.\ (\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,\mathtt{x})\ \mathtt{f}\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ (\lambda \mathtt{x}.\ \mathtt{f}^m\,\mathtt{x})\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,(\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\equiv \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,((\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ \mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,((\lambda \mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ \mathtt{x})
\end{aligned}
$$

Auswertung der Additionsfunktion

- **Zeige: add $\overline{m}$ $\overline{n}$ reduziert zu $\overline{m+n}$**

$$
\begin{aligned}
\textbf{add}\ \overline{m}\ \overline{n} &\equiv (\lambda \mathtt{m}.\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{m}\ \overline{n} \\
&\longrightarrow (\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{n} \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\equiv \lambda \mathtt{f}.\lambda \mathtt{x}.\ (\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,\mathtt{x})\ \mathtt{f}\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ (\lambda \mathtt{x}.\ \mathtt{f}^m\,\mathtt{x})\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,(\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\equiv \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,((\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ \mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,((\lambda \mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ \mathtt{x}) \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,(\mathtt{f}^n\,\mathtt{x})
\end{aligned}
$$

Auswertung der Additionsfunktion

- **Zeige: add $\overline{m}$ $\overline{n}$ reduziert zu $\overline{m+n}$**

$$
\begin{aligned}
\textbf{add}\ \overline{m}\ \overline{n} &\equiv (\lambda \mathtt{m}.\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{m}\ \overline{n} \\
&\longrightarrow (\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\mathtt{n}\ \mathtt{f}\ \mathtt{x}))\ \overline{n} \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f}\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\equiv \lambda \mathtt{f}.\lambda \mathtt{x}.\ (\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,\mathtt{x})\ \mathtt{f}\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ (\lambda \mathtt{x}.\ \mathtt{f}^m\,\mathtt{x})\ (\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,(\overline{n}\ \mathtt{f}\ \mathtt{x}) \\
&\equiv \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,((\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ \mathtt{f}\ \mathtt{x}) \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,((\lambda \mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ \mathtt{x}) \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,(\mathtt{f}^n\,\mathtt{x}) \\
&\longrightarrow \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^{m+n}\,\mathtt{x}
\end{aligned}
$$

Auswertung der Additionsfunktion

- **Zeige: add $\overline{m}$ $\overline{n}$ reduziert zu $\overline{m+n}$**

$$
\begin{aligned}
\textbf{add}\ \overline{m}\ \overline{n} \quad &\equiv \quad (\lambda \mathtt{m}.\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{m\ f\ (n\ f\ x)})\ \overline{m}\ \overline{n} \\
&\longrightarrow \quad (\lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f\ (n\ f\ x)})\ \overline{n} \\
&\longrightarrow \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ \overline{m}\ \mathtt{f\ (}\overline{n}\ \mathtt{f\ x)} \\
&\equiv \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ (\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,\mathtt{x})\ \mathtt{f\ (}\overline{n}\ \mathtt{f\ x)} \\
&\longrightarrow \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ (\lambda \mathtt{x}.\ \mathtt{f}^m\,\mathtt{x})\ (\overline{n}\ \mathtt{f\ x)} \\
&\longrightarrow \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,(\overline{n}\ \mathtt{f\ x)} \\
&\equiv \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,((\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ \mathtt{f\ x)} \\
&\longrightarrow \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,((\lambda \mathtt{x}.\ \mathtt{f}^n\,\mathtt{x})\ \mathtt{x)} \\
&\longrightarrow \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^m\,(\mathtt{f}^n\,\mathtt{x)} \\
&\longrightarrow \quad \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^{m+n}\,\mathtt{x} \qquad\qquad \equiv \quad \overline{m+n}
\end{aligned}
$$

Rekursion im λ-Kalkül

Y-Kombinator: $\mathbf{Y} \equiv \lambda \mathtt{f}.\,(\lambda \mathtt{x}.\,\mathtt{f}\,(\mathtt{x}\,\mathtt{x}))\,(\lambda \mathtt{x}.\,\mathtt{f}\,(\mathtt{x}\,\mathtt{x}))$

- **Y ist Fixpunktkombinator**
  - $\mathbf{Y}\ t = t\ (\mathbf{Y}\ t)$ für beliebige Terme t

Rekursion im λ-Kalkül

Y-Kombinator: $\mathbf{Y} \equiv \lambda \mathtt{f}.\,(\lambda \mathtt{x}.\,\mathtt{f}\,(\mathtt{x}\,\mathtt{x}))\,(\lambda \mathtt{x}.\,\mathtt{f}\,(\mathtt{x}\,\mathtt{x}))$

- **Y ist Fixpunktkombinator**
  - $\mathbf{Y}\ t = t\ (\mathbf{Y}\ t)$ für beliebige Terme t

$$
\begin{array}{lcl}
\mathbf{Y}\ t & \equiv & \lambda \mathtt{f}.\,(\lambda \mathtt{x}.\,\mathtt{f}\ (\mathtt{x}\ \mathtt{x}))\,(\lambda \mathtt{x}.\,\mathtt{f}\ (\mathtt{x}\ \mathtt{x}))\ t \\
 & \longrightarrow & (\lambda \mathtt{x}.\,t\ (\mathtt{x}\ \mathtt{x}))\ (\lambda \mathtt{x}.\,t\ (\mathtt{x}\ \mathtt{x})) \\
 & \longrightarrow & t\ (\ (\lambda \mathtt{x}.\,t\ (\mathtt{x}\ \mathtt{x}))\ (\lambda \mathtt{x}.\,t\ (\mathtt{x}\ \mathtt{x}))\ ) \\
t\ (\mathbf{Y}\ t) & \equiv & t\ (\lambda \mathtt{f}.\,(\lambda \mathtt{x}.\,\mathtt{f}\ (\mathtt{x}\ \mathtt{x}))\,(\lambda \mathtt{x}.\,\mathtt{f}\ (\mathtt{x}\ \mathtt{x}))\ t) \\
 & \longrightarrow & t\ (\ (\lambda \mathtt{x}.\,t\ (\mathtt{x}\ \mathtt{x}))\ (\lambda \mathtt{x}.\,t\ (\mathtt{x}\ \mathtt{x}))\ )
\end{array}
$$

Rekursion im λ-Kalkül

Y-Kombinator: $\mathsf{Y} \equiv \lambda \mathtt{f}.\,(\lambda \mathtt{x}.\,\mathtt{f}\,(\mathtt{x}\,\mathtt{x}))\,(\lambda \mathtt{x}.\,\mathtt{f}\,(\mathtt{x}\,\mathtt{x}))$

- **Y ist Fixpunktkombinator**
  - $\mathsf{Y}\,t = t\,(\mathsf{Y}\,t)$ für beliebige Terme t

$$
\begin{array}{lcl}
\mathsf{Y}\ t & \equiv & \lambda \mathtt{f}.\,(\lambda \mathtt{x}.\,\mathtt{f}\ (\mathtt{x}\ \mathtt{x}))\,(\lambda \mathtt{x}.\,\mathtt{f}\ (\mathtt{x}\ \mathtt{x}))\ t \\
 & \longrightarrow & (\lambda \mathtt{x}.\,t\ (\mathtt{x}\ \mathtt{x}))\ (\lambda \mathtt{x}.\,t\ (\mathtt{x}\ \mathtt{x})) \\
 & \longrightarrow & t\ (\ (\lambda \mathtt{x}.\,t\ (\mathtt{x}\ \mathtt{x}))\ (\lambda \mathtt{x}.\,t\ (\mathtt{x}\ \mathtt{x}))\ ) \\
t\ (\mathsf{Y}\ t) & \equiv & t\ (\lambda \mathtt{f}.\,(\lambda \mathtt{x}.\,\mathtt{f}\ (\mathtt{x}\ \mathtt{x}))\,(\lambda \mathtt{x}.\,\mathtt{f}\ (\mathtt{x}\ \mathtt{x}))\ t) \\
 & \longrightarrow & t\ (\ (\lambda \mathtt{x}.\,t\ (\mathtt{x}\ \mathtt{x}))\ (\lambda \mathtt{x}.\,t\ (\mathtt{x}\ \mathtt{x}))\ )
\end{array}
$$

- **Rekursion darstellbar als**

  letrec $f\,(x) = t \equiv \mathsf{Y}\,(\lambda f\,.\,\lambda x\,.\,t)$

Ausdruckskraft des λ-Kalküls

Alle μ-rekursiven Funktione sind λ-berechenbar

- **Nachfolgerfunktion s:** `s` $\equiv$ λ`n.`λ`f.`λ`x. n f (f x)`

Ausdruckskraft des λ-Kalküls

Alle μ-rekursiven Funktione sind λ-berechenbar

- **Nachfolgerfunktion s:** $\mathtt{s} \equiv \lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x})$
- **Projektionsfunktionen pr^n_m** $\mathtt{pr}^n_m \equiv \lambda \mathtt{x}_1 . . \lambda \mathtt{x}_n .\ \mathtt{x}_m$

Ausdruckskraft des λ-Kalküls

Alle μ-rekursiven Funktione sind λ-berechenbar

- **Nachfolgerfunktion s:** $\mathtt{s} \equiv \lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x})$
- **Projektionsfunktionen pr^n_m** $\mathtt{pr}^n_m \equiv \lambda \mathtt{x}_1 . . \lambda \mathtt{x}_n . \mathtt{x}_m$
- **Konstantenfunktion c^n_m:** $\mathtt{c}^n_m \equiv \lambda \mathtt{x}_1 . . \lambda \mathtt{x}_n . \overline{\mathtt{m}}$

Ausdruckskraft des λ-Kalküls

Alle μ-rekursiven Funktione sind λ-berechenbar

- **Nachfolgerfunktion s:** $\mathtt{s} \equiv \lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x})$
- **Projektionsfunktionen pr^n_m** $\mathtt{pr}^n_m \equiv \lambda \mathtt{x}_1 . . \lambda \mathtt{x}_n .\ \mathtt{x}_m$
- **Konstantenfunktion c^n_m:** $\mathtt{c}^n_m \equiv \lambda \mathtt{x}_1 . . \lambda \mathtt{x}_n .\ \overline{\mathtt{m}}$
- **Komposition $f \circ (g_1 ... g_n)$:**
  - $\circ \equiv \lambda \mathtt{f} . \lambda \mathtt{g}_1 . . \lambda \mathtt{g}_n .\ \lambda \mathtt{x} .\ \mathtt{f}\ (\mathtt{g}_1 \mathtt{x}) . . (\mathtt{g}_n \mathtt{x})$

Ausdruckskraft des λ-Kalküls

Alle μ-rekursiven Funktione sind λ-berechenbar

- **Nachfolgerfunktion s:** $\mathbf{s} \equiv \lambda n.\lambda f.\lambda x.\ n\ f\ (f\ x)$
- **Projektionsfunktionen pr^n_m** $\mathbf{pr}^n_m \equiv \lambda x_1..\lambda x_n.\ x_m$
- **Konstantenfunktion c^n_m:** $\mathbf{c}^n_m \equiv \lambda x_1..\lambda x_n.\ \overline{m}$
- **Komposition $f \circ (g_1 ... g_n)$:**
  - $\circ \equiv \lambda f.\lambda g_1..\lambda g_n.\ \lambda x.\ f\ (g_1 x)..(g_n x)$
- **Primitive Rekursion $Pr[f, g]$:**
  - $\mathbf{PR} \equiv \lambda f.\lambda g.$
    letrec $h(x) = \lambda y.$ **if zero** y **then** $f\ x$ **else** $g\ x\ (\mathbf{p}\ y)\ (h\ x\ (\mathbf{p}\ y))$

Ausdruckskraft des λ-Kalküls

Alle μ-rekursiven Funktione sind λ-berechenbar

- **Nachfolgerfunktion s:** $\mathsf{s} \equiv \lambda \mathtt{n}.\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{n}\ \mathtt{f}\ (\mathtt{f}\ \mathtt{x})$
- **Projektionsfunktionen pr^n_m** $\mathsf{pr}^n_m \equiv \lambda \mathtt{x}_1 .. \lambda \mathtt{x}_n .\ \mathtt{x}_m$
- **Konstantenfunktion c^n_m:** $\mathsf{c}^n_m \equiv \lambda \mathtt{x}_1 .. \lambda \mathtt{x}_n .\ \overline{\mathtt{m}}$
- **Komposition $f \circ (g_1 ... g_n)$:**
  - $\circ \equiv \lambda \mathtt{f}.\lambda \mathtt{g}_1 .. \lambda \mathtt{g}_n .\ \lambda \mathtt{x}.\ \mathtt{f}\ (\mathtt{g}_1 \mathtt{x}) .. (\mathtt{g}_n \mathtt{x})$
- **Primitive Rekursion $Pr[f, g]$:**
  - $\mathsf{PR} \equiv \lambda \mathtt{f}.\lambda \mathtt{g}.$
    letrec $\mathtt{h}(\mathtt{x}) = \lambda \mathtt{y}.$ **if zero** $\mathtt{y}$ **then** $\mathtt{f}\ \mathtt{x}$ **else** $\mathtt{g}\ \mathtt{x}\ (\mathsf{p}\ \mathtt{y})\ (\mathtt{h}\ \mathtt{x}\ (\mathsf{p}\ \mathtt{y}))$
- **Minimierung $\mu[f]$:**
  - $\mathsf{Mu} \equiv \lambda \mathtt{f}.\lambda \mathtt{x}.$
    (**letrec** $\mathtt{min}(\mathtt{y}) =$ **if zero** $(\mathtt{f}\ \mathtt{x}\ \mathtt{y})$ **then** $\mathtt{y}$ **else** $\mathtt{min}\ (\mathsf{s}\ \mathtt{y}))\ \overline{0}$

Weitere Modelle

- **Nichtdeterministische Turingmaschine**
  - Arbeitsweise wie gewöhnliche Turingmaschine
  - Zustandsüberführungsfunktion erlaubt alternative Resultate

Weitere Modelle

- **Nichtdeterministische Turingmaschine**
  - Arbeitsweise wie gewöhnliche Turingmaschine
  - Zustandsüberführungsfunktion erlaubt alternative Resultate
- **Abakus**
  - Erweiterung des mechanischen Abakus: beliebig viele Stangen und Kugeln
  - Zwei Operationen: Kugel hinzunehmen / Kugel wegnehmen

Weitere Modelle

- **Nichtdeterministische Turingmaschine**
  - Arbeitsweise wie gewöhnliche Turingmaschine
  - Zustandsüberführungsfunktion erlaubt alternative Resultate
- **Abakus**
  - Erweiterung des mechanischen Abakus: beliebig viele Stangen und Kugeln
  - Zwei Operationen: Kugel hinzunehmen / Kugel wegnehmen
- **Markov-Algorithmen**
  - Wie Typ-0 Grammatiken, aber mit fester Strategie für Regelanwendung
  - Verarbeitet Eingabeworte, statt mit einem Startsymbol zu beginnen

Weitere Modelle

- **Nichtdeterministische Turingmaschine**
  - Arbeitsweise wie gewöhnliche Turingmaschine
  - Zustandsüberführungsfunktion erlaubt alternative Resultate
- **Abakus**
  - Erweiterung des mechanischen Abakus: beliebig viele Stangen und Kugeln
  - Zwei Operationen: Kugel hinzunehmen / Kugel wegnehmen
- **Markov-Algorithmen**
  - Wie Typ-0 Grammatiken, aber mit fester Strategie für Regelanwendung
  - Verarbeitet Eingabeworte, statt mit einem Startsymbol zu beginnen
- **Arithmetische Repräsentierbarkeit**
  - Spezifikation von Funktionen in arithmetisch-logischem Kalkül
  - f ist repräsentierbar, wenn das Ein-/Ausgabeverhalten von f eindeutig durch eine Formel spezifiziert werden kann
  - Eindeutigkeit muß ausschließlich aus logischen Axiomen beweisbar sein

Die Church'sche These

- **Alle Berechenbarkeitsmodelle sind äquivalent**
  - Keines kann mehr berechnen als Turingmaschinen
  - Es ist keine intuitiv berechenbare Funktion bekannt, die nicht von Turingmaschinen berechnet werden kann

Die Church'sche These

- **Alle Berechenbarkeitsmodelle sind äquivalent**
  - Keines kann mehr berechnen als Turingmaschinen
  - Es ist keine intuitiv berechenbare Funktion bekannt, die nicht von Turingmaschinen berechnet werden kann
- **Church'sche These:**

  Die Klasse der Turing-berechenbaren Funktionen stimmt mit der Klasse der intuitiv berechenbaren Funktionen überein

Die Church'sche These

- **Alle Berechenbarkeitsmodelle sind äquivalent**
  - Keines kann mehr berechnen als Turingmaschinen
  - Es ist keine intuitiv berechenbare Funktion bekannt, die nicht von Turingmaschinen berechnet werden kann
- **Church'sche These:**

  Die Klasse der Turing-berechenbaren Funktionen stimmt mit der Klasse der intuitiv berechenbaren Funktionen überein
  - Unbeweisbare, aber wahrscheinlich richtige Behauptung
  - Arbeitshypothese für theoretische Argumente
    - man darf in Beweisen "intuitive" Programme angeben