Praktische Informatik III - Konzepte von Programmiersprachen

Prof. Dr. Rita Loogen
vertreten von: Mischa Dieterle

Fachbereich Mathematik und Informatik
Philipps-Universität Marburg

19.01.2009

Inhalt

Inhalt

Inhalt

Freie vs. gebundene Variablen

- In λ-Ausdrücken können freie und gebundene Variablen vorkommen
- Beispiel: $\lambda\ x.E$: jedes Auftreten von x in E ist *gebunden*
- Alle nicht gebundenen Variablen heißen *frei*.

Freie vs. gebundene Variablen

- In λ-Ausdrücken können freie und gebundene Variablen vorkommen
- Beispiel: $\lambda\ x.E$: jedes Auftreten von x in E ist *gebunden*
- Alle nicht gebundenen Variablen heißen *frei*.

Freie vs. gebundene Variablen

- In λ-Ausdrücken können freie und gebundene Variablen vorkommen
- Beispiel: $\lambda\ x.E$: jedes Auftreten von x in E ist *gebunden*
- Alle nicht gebundenen Variablen heißen *frei*.

Definition 9.2 (Freie und gebundene Variablen)

$$free, bound : Exp \longrightarrow \wp(Var)$$

definieren die Mengen der *freien bzw. gebundenen Variablen in einem λ-Ausdruck*:

1. $free(x) = \{x\}$
   $bound(x) = \emptyset$
2. $free(E_1\ E_2) = free(E_1) \cup free(E_2)$
   $bound(E_1\ E_2) = bound(E_1) \cup bound(E_2)$
3. $free(\lambda x.E) = free(E) \setminus \{x\}$
   $bound(\lambda x.E) = bound(E) \cup \{x\}$

Definition 9.2 (Freie und gebundene Variablen)

$$free, bound : Exp \longrightarrow \wp(Var)$$

definieren die Mengen der *freien bzw. gebundenen Variablen in einem λ-Ausdruck*:

1. $free(x) = \{x\}$
   $bound(x) = \emptyset$
2. $free(E_1\ E_2) = free(E_1) \cup free(E_2)$
   $bound(E_1\ E_2) = bound(E_1) \cup bound(E_2)$
3. $free(\lambda x.E) = free(E) \setminus \{x\}$
   $bound(\lambda x.E) = bound(E) \cup \{x\}$

Definition 9.2 (Freie und gebundene Variablen)

$$free, bound : Exp \longrightarrow \wp(Var)$$

definieren die Mengen der *freien bzw. gebundenen Variablen in einem λ-Ausdruck*:

1. $free(x) = \{x\}$
   $bound(x) = \emptyset$
2. $free(E_1\ E_2) = free(E_1) \cup free(E_2)$
   $bound(E_1\ E_2) = bound(E_1) \cup bound(E_2)$
3. $free(\lambda x.E) = free(E) \setminus \{x\}$
   $bound(\lambda x.E) = bound(E) \cup \{x\}$

Definition 9.2 (Freie und gebundene Variablen)

$$free, bound : Exp \longrightarrow \wp(Var)$$

definieren die Mengen der *freien bzw. gebundenen Variablen in einem λ-Ausdruck*:

1. $free(x) = \{x\}$
   $bound(x) = \emptyset$
2. $free(E_1\ E_2) = free(E_1) \cup free(E_2)$
   $bound(E_1\ E_2) = bound(E_1) \cup bound(E_2)$
3. $free(\lambda x.E) = free(E) \setminus \{x\}$
   $bound(\lambda x.E) = bound(E) \cup \{x\}$

Beispiel:

Definition 9.2 (Freie und gebundene Variablen)

1. $free(x) = \{x\}$ $\quad$ $bound(x) = \emptyset$
2. $free(E_1\ E_2) = free(E_1) \cup free(E_2)$ $\quad$ $bound(E_1\ E_2) = bound(E_1) \cup bound(E_2)$
3. $free(\lambda x.E) = free(E) \setminus \{x\}$ $\quad$ $bound(\lambda x.E) = bound(E) \cup \{x\}$

- $free(\lambda x.y\ x) = \{y\}$ und $bound(\lambda\ x.y\ x) = \{x\}$.
- $free(\lambda x.\lambda y.x) = \emptyset$ und $bound(\lambda\ x.\lambda y.x) = \{x, y\}$.
- $free(x\ \lambda x.y) = \{x, y\}$ und $bound(x\ \lambda x.y) = \{x\}$
  - ohne gebundenes Vorkommen von x

Beispiel:

Definition 9.2 (Freie und gebundene Variablen)

1. $free(x) = \{x\}$ $\quad$ $bound(x) = \emptyset$
2. $free(E_1\ E_2) = free(E_1) \cup free(E_2)$ $\quad$ $bound(E_1\ E_2) = bound(E_1) \cup bound(E_2)$
3. $free(\lambda x.E) = free(E) \setminus \{x\}$ $\quad$ $bound(\lambda x.E) = bound(E) \cup \{x\}$

- $free(\lambda x.y\ x) = \{y\}$ und $bound(\lambda\ x.y\ x) = \{x\}$.
- $free(\lambda x.\lambda y.x) = \emptyset$ und $bound(\lambda\ x.\lambda y.x) = \{x, y\}$.
- $free(x\ \lambda x.y) = \{x, y\}$ und $bound(x\ \lambda x.y) = \{x\}$
    - ohne gebundenes Vorkommen von x

Beispiel:

Definition 9.2 (Freie und gebundene Variablen)

1. $free(x) = \{x\}$ $\quad bound(x) = \emptyset$
2. $free(E_1\ E_2) = free(E_1) \cup free(E_2)$ $\quad bound(E_1\ E_2) = bound(E_1) \cup bound(E_2)$
3. $free(\lambda x.E) = free(E) \setminus \{x\}$ $\quad bound(\lambda x.E) = bound(E) \cup \{x\}$

- $free(\lambda x.y\ x) = \{y\}$ und $bound(\lambda\ x.y\ x) = \{x\}$.
- $free(\lambda x.\lambda y.x) = \emptyset$ und $bound(\lambda\ x.\lambda y.x) = \{x, y\}$.
- $free(x\ \lambda x.y) = \{x, y\}$ und $bound(x\ \lambda x.y) = \{x\}$
  - ohne gebundenes Vorkommen von x

Definition 9.3 (Geschlossene λ-Ausdrücke)

Ein λ-Ausdruck E heißt *geschlossen* oder *Kombinator*, falls $\mathit{free}(E) = \emptyset$.

- Die λ-Ausdrücke S, K und I sind Kombinatoren.
- Jeder λ-Ausdruck kann äquivalent in einen Ausdruck überführt werden, der nur mittels Applikation aus den Kombinatoren S, K und I gebildet wird.

Definition 9.3 (Geschlossene λ-Ausdrücke)

Ein λ-Ausdruck E heißt *geschlossen* oder *Kombinator*, falls $\mathit{free}(E) = \emptyset$.

- Die λ-Ausdrücke S, K und I sind Kombinatoren.
- Jeder λ-Ausdruck kann äquivalent in einen Ausdruck überführt werden, der nur mittels Applikation aus den Kombinatoren S, K und I gebildet wird.

Definition 9.3 (Geschlossene λ-Ausdrücke)

Ein λ-Ausdruck E heißt *geschlossen* oder *Kombinator*, falls $\mathit{free}(E) = \emptyset$.

- Die λ-Ausdrücke S, K und I sind Kombinatoren.
- Jeder λ-Ausdruck kann äquivalent in einen Ausdruck überführt werden, der nur mittels Applikation aus den Kombinatoren S, K und I gebildet wird.

α-äquivalenz

- Gebundene Variablen sind Platzhalter für Funktionsparameter.
- Umbenennen gebundener Variablen ändert Bedeutung nicht.

  $\lambda x.x$ und $\lambda y.y$ bezeichnen dieselbe Funktion

- Ausdrücke, die sich nur in den Namen gebundener Variablen unterscheiden sind *α-äquivalent*:

$$\lambda x.x =_\alpha \lambda y.y =_\alpha \lambda z.z.$$

α-äquivalenz

- Gebundene Variablen sind Platzhalter für Funktionsparameter.
- Umbenennen gebundener Variablen ändert Bedeutung nicht.

$\lambda x.x$ und $\lambda y.y$ bezeichnen dieselbe Funktion

- Ausdrücke, die sich nur in den Namen gebundener Variablen unterscheiden sind *α-äquivalent*:

$$\lambda x.x =_\alpha \lambda y.y =_\alpha \lambda z.z.$$

α-äquivalenz

- Gebundene Variablen sind Platzhalter für Funktionsparameter.
- Umbenennen gebundener Variablen ändert Bedeutung nicht.

$\lambda x.x$ und $\lambda y.y$ bezeichnen dieselbe Funktion

- Ausdrücke, die sich nur in den Namen gebundener Variablen unterscheiden sind *α-äquivalent*:

$$\lambda x.x =_\alpha \lambda y.y =_\alpha \lambda z.z.$$

α-äquivalenz

- Gebundene Variablen sind Platzhalter für Funktionsparameter.
- Umbenennen gebundener Variablen ändert Bedeutung nicht.

$\lambda x.x$ und $\lambda y.y$ bezeichnen dieselbe Funktion

- Ausdrücke, die sich nur in den Namen gebundener Variablen unterscheiden sind *α-äquivalent*:

$$\lambda x.x =_\alpha \lambda y.y =_\alpha \lambda z.z.$$

Inhalt

1. Der λ-Kalkül
   - Freie und gebundene Variablen
   - Substitution und β-Reduktion
   - Die Reduktionssemantik des λ-Kalküls
   - Normalformen und Konfluenz

Substitution

- $E[x/M]$: Ersetzung aller freien Vorkommen von x in E durch M
- Achtung: Konflikte zwischen freien Variablen in M und λ-Bindungen in E möglich
- Beispiel: $E = \lambda y.(x\ y)$ und $M = y$: $\lambda y.(x\ y)[x/y]$
  - $\neq \lambda y.(y\ y)$, da y in E gebunden und frei in M
  - Ersetze y in E durch z: $\lambda z.(x\ z)[x/y]$

$$\lambda z.(x\ z)[x/y] = \lambda z.(y\ z)$$

Substitution

- $E[x/M]$: Ersetzung aller freien Vorkommen von x in E durch M
- Achtung: Konflikte zwischen freien Variablen in M und λ-Bindungen in E möglich
- Beispiel: $E = \lambda y.(x\ y)$ und $M = y$: $\lambda y.(x\ y)[x/y]$
  - $\neq \lambda y.(y\ y)$, da y in E gebunden und frei in M
  - Ersetze y in E durch z: $\lambda z.(x\ z)[x/y]$

$$\lambda z.(x\ z)[x/y] = \lambda z.(y\ z)$$

Substitution

- $E[x/M]$: Ersetzung aller freien Vorkommen von x in E durch M
- Achtung: Konflikte zwischen freien Variablen in M und λ-Bindungen in E möglich
- Beispiel: $E = \lambda y.(x\ y)$ und $M = y$: $\lambda y.(x\ y)[x/y]$
  - $\neq \lambda y.(y\ y)$, da y in E gebunden und frei in M
  - Ersetze y in E durch z: $\lambda z.(x\ z)[x/y]$

$$\lambda z.(x\ z)[x/y] = \lambda z.(y\ z)$$

Substitution

- $E[x/M]$: Ersetzung aller freien Vorkommen von x in E durch M
- Achtung: Konflikte zwischen freien Variablen in M und λ-Bindungen in E möglich
- Beispiel: $E = \lambda y.(x\ y)$ und $M = y$: $\lambda y.(x\ y)[x/y]$
    - $\neq \lambda y.(y\ y)$, da y in E gebunden und frei in M
    - Ersetze y in E durch z: $\lambda z.(x\ z)[x/y]$

$$\lambda z.(x\ z)[x/y] = \lambda z.(y\ z)$$

Substitution

- $E[x/M]$: Ersetzung aller freien Vorkommen von x in E durch M
- Achtung: Konflikte zwischen freien Variablen in M und λ-Bindungen in E möglich
- Beispiel: $E = \lambda y.(x\ y)$ und $M = y$: $\lambda y.(x\ y)[x/y]$
  - $\neq \lambda y.(y\ y)$, da y in E gebunden und frei in M
  - Ersetze y in E durch z: $\lambda z.(x\ z)[x/y]$

$$\lambda z.(x\ z)[x/y] = \lambda z.(y\ z)$$

Substitution

- $E[x/M]$: Ersetzung aller freien Vorkommen von x in E durch M
- Achtung: Konflikte zwischen freien Variablen in M und λ-Bindungen in E möglich
- Beispiel: $E = \lambda y.(x\ y)$ und $M = y$: $\lambda y.(x\ y)[x/y]$
  - $\neq \lambda y.(y\ y)$, da y in E gebunden und frei in M
  - Ersetze y in E durch z: $\lambda z.(x\ z)[x/y]$

$$\lambda z.(x\ z)[x/y] = \lambda z.(y\ z)$$

Definition 9.4 (Substitution)

Seien $x \in Var$, $E, M \in Exp$. Dann wird das Resultat

$$E[x/M]$$

der *Substitution* aller freien Vorkommen von x in E durch M wie folgt induktiv definiert:

1. $y[x/M] := \begin{cases} M & \text{falls } x = y \\ y & \text{sonst.} \end{cases}$
2. $(E_1\ E_2)[x/M] := (E_1[x/M]\ E_2[x/M])$.
3. $(\lambda y.E)[x/M] :=$
$$\begin{cases} \lambda x.E & \text{falls } y = x \\ \lambda y.(E[x/M]) & \text{falls } y \notin free(M) \\ \lambda z.(E[y/z][x/M]) & \text{sonst,} \\ & \text{wobei } z \notin free(E) \cup free(M) \text{ neu.} \end{cases}$$

Ergebnis nur bis auf α-Äquivalenz eindeutig.

Definition 9.4 (Substitution)

Seien $x \in Var$, $E, M \in Exp$. Dann wird das Resultat

$$E[x/M]$$

der *Substitution* aller freien Vorkommen von x in E durch M wie folgt induktiv definiert:

1. $y[x/M] := \begin{cases} M & \text{falls } x = y \\ y & \text{sonst.} \end{cases}$
2. $(E_1\ E_2)[x/M] := (E_1[x/M]\ E_2[x/M])$.
3. $(\lambda y.E)[x/M] :=$
$$\begin{cases} \lambda x.E & \text{falls } y = x \\ \lambda y.(E[x/M]) & \text{falls } y \notin free(M) \\ \lambda z.(E[y/z][x/M]) & \text{sonst,} \\ & \text{wobei } z \notin free(E) \cup free(M) \text{ neu.} \end{cases}$$

Ergebnis nur bis auf α-Äquivalenz eindeutig.

Definition 9.4 (Substitution)

Seien $x \in Var$, $E, M \in Exp$. Dann wird das Resultat

$$E[x/M]$$

der *Substitution* aller freien Vorkommen von x in E durch M wie folgt induktiv definiert:

1. $y[x/M] := \begin{cases} M & \text{falls } x = y \\ y & \text{sonst.} \end{cases}$
2. $(E_1\ E_2)[x/M] := (E_1[x/M]\ E_2[x/M])$.
3. $(\lambda y.E)[x/M] :=$
$$\begin{cases} \lambda x.E & \text{falls } y = x \\ \lambda y.(E[x/M]) & \text{falls } y \notin free(M) \\ \lambda z.(E[y/z][x/M]) & \text{sonst,} \\ & \text{wobei } z \notin free(E) \cup free(M) \text{ neu.} \end{cases}$$

Ergebnis nur bis auf α-Äquivalenz eindeutig.

Definition 9.4 (Substitution)

Seien $x \in \mathit{Var}$, $E, M \in \mathit{Exp}$. Dann wird das Resultat

$$E[x/M]$$

der *Substitution* aller freien Vorkommen von x in E durch M wie folgt induktiv definiert:

1. $y[x/M] := \begin{cases} M & \text{falls } x = y \\ y & \text{sonst.} \end{cases}$
2. $(E_1\ E_2)[x/M] := (E_1[x/M]\ E_2[x/M])$.
3. $(\lambda y.E)[x/M] :=$
$$\begin{cases} \lambda x.E & \text{falls } y = x \\ \lambda y.(E[x/M]) & \text{falls } y \notin \mathit{free}(M) \\ \lambda z.(E[y/z][x/M]) & \text{sonst,} \\ & \text{wobei } z \notin \mathit{free}(E) \cup \mathit{free}(M) \text{ neu.} \end{cases}$$

Ergebnis nur bis auf α-Äquivalenz eindeutig.

Definition 9.4 (Substitution)

Seien $x \in Var$, $E, M \in Exp$. Dann wird das Resultat

$$E[x/M]$$

der *Substitution* aller freien Vorkommen von x in E durch M wie folgt induktiv definiert:

1. $y[x/M] := \begin{cases} M & \text{falls } x = y \\ y & \text{sonst.} \end{cases}$
2. $(E_1\ E_2)[x/M] := (E_1[x/M]\ E_2[x/M])$.
3. $(\lambda y.E)[x/M] :=$
$$\begin{cases} \lambda x.E & \text{falls } y = x \\ \lambda y.(E[x/M]) & \text{falls } y \notin free(M) \\ \lambda z.(E[y/z][x/M]) & \text{sonst,} \\ & \text{wobei } z \notin free(E) \cup free(M) \text{ neu.} \end{cases}$$

Ergebnis nur bis auf α-Äquivalenz eindeutig.

Beispiel (Substitution):

Definition 9.4 (Substitution)

1. $y[x/M] := \begin{cases} M & \text{falls } x = y \\ y & \text{sonst.} \end{cases}$
2. $(E_1\ E_2)[x/M] := (E_1[x/M]\ E_2[x/M])$.
3. $(\lambda y.E)[x/M] := \begin{cases} \lambda x.E & \text{falls } y = x \\ \lambda y.(E[x/M]) & \text{falls } y \notin \mathit{free}(M) \\ \lambda z.(E[y/z][x/M]) & \text{sonst,} \\ & \text{wobei } z \notin \mathit{free}(E) \cup \mathit{free}(M) \text{ neu.} \end{cases}$

$$((\lambda y.x)(\lambda x.x)x)[x/y]$$

Beispiel (Substitution):

Definition 9.4 (Substitution)

1. $$y[x/M] := \begin{cases} M & \text{falls } x = y \\ y & \text{sonst.} \end{cases}$$
2. $(E_1\ E_2)[x/M] := (E_1[x/M]\ E_2[x/M])$.
3. $$(\lambda y.E)[x/M] := \begin{cases} \lambda x.E & \text{falls } y = x \\ \lambda y.(E[x/M]) & \text{falls } y \notin \mathit{free}(M) \\ \lambda z.(E[y/z][x/M]) & \text{sonst,} \\ & \text{wobei } z \notin \mathit{free}(E) \cup \mathit{free}(M) \text{ neu.} \end{cases}$$

$$((\lambda y.x)(\lambda x.x)x)[x/y] \quad = \quad ((\lambda y.x)[x/y](\lambda x.x)[x/y]x[x/y]) < 2 >$$

Beispiel (Substitution):

Definition 9.4 (Substitution)

1. $y[x/M] := \begin{cases} M & \text{falls } x = y \\ y & \text{sonst.} \end{cases}$
2. $(E_1\ E_2)[x/M] := (E_1[x/M]\ E_2[x/M])$.
3. $(\lambda y.E)[x/M] := \begin{cases} \lambda x.E & \text{falls } y = x \\ \lambda y.(E[x/M]) & \text{falls } y \notin \mathit{free}(M) \\ \lambda z.(E[y/z][x/M]) & \text{sonst,} \\ & \text{wobei } z \notin \mathit{free}(E) \cup \mathit{free}(M) \text{ neu.} \end{cases}$

$$\begin{aligned} ((\lambda y.x)(\lambda x.x)x)[x/y] &= ((\lambda y.x)[x/y](\lambda x.x)[x/y]x[x/y]) \\ &= ((\lambda z.y) < 3.3 > \end{aligned}$$

Beispiel (Substitution):

Definition 9.4 (Substitution)

1. $y[x/M] := \begin{cases} M & \text{falls } x = y \\ y & \text{sonst.} \end{cases}$
2. $(E_1\ E_2)[x/M] := (E_1[x/M]\ E_2[x/M])$.
3. $(\lambda y.E)[x/M] := \begin{cases} \lambda x.E & \text{falls } y = x \\ \lambda y.(E[x/M]) & \text{falls } y \notin \mathit{free}(M) \\ \lambda z.(E[y/z][x/M]) & \text{sonst,} \\ & \text{wobei } z \notin \mathit{free}(E) \cup \mathit{free}(M) \text{ neu.} \end{cases}$

$$\begin{aligned} ((\lambda y.x)(\lambda x.x)x)[x/y] &= ((\lambda y.x)[x/y](\lambda x.x)[x/y]x[x/y]) \\ &= ((\lambda z.y)(\lambda x.x) < 3.1 > \end{aligned}$$

Beispiel (Substitution):

Definition 9.4 (Substitution)

1. $y[x/M] := \begin{cases} M & \text{falls } x = y \\ y & \text{sonst.} \end{cases}$
2. $(E_1\ E_2)[x/M] := (E_1[x/M]\ E_2[x/M])$.
3. $(\lambda y.E)[x/M] := \begin{cases} \lambda x.E & \text{falls } y = x \\ \lambda y.(E[x/M]) & \text{falls } y \notin \mathit{free}(M) \\ \lambda z.(E[y/z][x/M]) & \text{sonst,} \\ & \text{wobei } z \notin \mathit{free}(E) \cup \mathit{free}(M) \text{ neu.} \end{cases}$

$$\begin{aligned} ((\lambda y.x)(\lambda x.x)x)[x/y] &= ((\lambda y.x)[x/y](\lambda x.x)[x/y]x[x/y]) \\ &= ((\lambda z.y)(\lambda x.x)y < 1 > \end{aligned}$$

Beispiel (Substitution):

Definition 9.4 (Substitution)

1. $y[x/M] := \begin{cases} M & \text{falls } x = y \\ y & \text{sonst.} \end{cases}$
2. $(E_1\ E_2)[x/M] := (E_1[x/M]\ E_2[x/M])$.
3. $(\lambda y.E)[x/M] := \begin{cases} \lambda x.E & \text{falls } y = x \\ \lambda y.(E[x/M]) & \text{falls } y \notin \mathit{free}(M) \\ \lambda z.(E[y/z][x/M]) & \text{sonst,} \\ & \text{wobei } z \notin \mathit{free}(E) \cup \mathit{free}(M) \text{ neu.} \end{cases}$

$$\begin{aligned} ((\lambda y.x)(\lambda x.x)x)[x/y] &= ((\lambda y.x)[x/y](\lambda x.x)[x/y]x[x/y]) \\ &= ((\lambda z.y)(\lambda x.x)y)) \end{aligned}$$

Auswertung der Applikation

Definition 9.5 (β-Reduktion)

$$(\lambda x.E)\ M \rightarrow_\beta E[x/M]$$

Beispiel Kombinatoren (β-Reduktion):

Definition 9.5 (β-Reduktion)

$$(\lambda x.E)\ M \rightarrow_\beta E[x/M]$$

$$S\ I\ I \equiv \quad (\lambda x\ y\ z.x\ z\ (y\ z))(\lambda u.u)(\lambda v.v)$$

Beispiel Kombinatoren (β-Reduktion):

Definition 9.5 (β-Reduktion)

$$(\lambda x.E)\ M \rightarrow_\beta E[x/M]$$

$$\begin{aligned} S\ I\ I \equiv &\quad (\lambda x\ y\ z.x\ z\ (y\ z))(\lambda u.u)(\lambda v.v) \\ \Rightarrow_\beta &\quad \text{Anwendung der } \beta\text{-Reduktion im Kontext} \end{aligned}$$

Beispiel Kombinatoren (β-Reduktion):

Definition 9.5 (β-Reduktion)

$$(\lambda x.E)\ M \rightarrow_\beta E[x/M]$$

$$\begin{aligned} S\ I\ I \equiv &\quad (\lambda x\ y\ z.x\ z\ (y\ z))(\lambda u.u)(\lambda v.v) \\ \Rightarrow_\beta &\quad (\lambda y\ z.(\lambda u.u)\ z\ (y\ z))(\lambda v.v) \end{aligned}$$

Beispiel Kombinatoren (β-Reduktion):

Definition 9.5 (β-Reduktion)

$$(\lambda x.E)\ M \rightarrow_\beta E[x/M]$$

$$\begin{array}{rll} S\ I\ I \equiv & & (\lambda x\ y\ z.x\ z\ (y\ z))(\lambda u.u)(\lambda v.v) \\ & \Rightarrow_\beta & (\lambda y\ z.(\lambda u.u)\ z\ (y\ z))(\lambda v.v) \\ & \Rightarrow_\beta & (\lambda y\ z.(z\ (y\ z)))(\lambda v.v) \end{array}$$

Beispiel Kombinatoren (β-Reduktion):

Definition 9.5 (β-Reduktion)

$$(\lambda x.E)\ M \rightarrow_\beta E[x/M]$$

$$\begin{aligned}
S\ I\ I \equiv &\quad (\lambda x\ y\ z.x\ z\ (y\ z))(\lambda u.u)(\lambda v.v) \\
\Rightarrow_\beta &\quad (\lambda y\ z.(\lambda u.u)\ z\ (y\ z))(\lambda v.v) \\
\Rightarrow_\beta &\quad (\lambda y\ z.(z\ (y\ z)))(\lambda v.v) \\
\Rightarrow_\beta &\quad \lambda z.(z\ ((\lambda v.v)\ z))
\end{aligned}$$

Beispiel Kombinatoren (β-Reduktion):

Definition 9.5 (β-Reduktion)

$$(\lambda x.E)\ M \rightarrow_\beta E[x/M]$$

$$\begin{aligned}
S\ I\ I \equiv \quad & (\lambda x\ y\ z.x\ z\ (y\ z))(\lambda u.u)(\lambda v.v) \\
\Rightarrow_\beta \quad & (\lambda y\ z.(\lambda u.u)\ z\ (y\ z))(\lambda v.v) \\
\Rightarrow_\beta \quad & (\lambda y\ z.(z\ (y\ z)))(\lambda v.v) \\
\Rightarrow_\beta \quad & \lambda z.(z\ (\textcolor{red}{(\lambda v.v)\ z})) \\
\Rightarrow_\beta \quad & \lambda z.(z\ z)
\end{aligned}$$

Beispiel Kombinatoren (β-Reduktion):

Definition 9.5 (β-Reduktion)

$$(\lambda x.E)\ M \rightarrow_\beta E[x/M]$$

$$\begin{aligned}
S\ I\ I \equiv\ & (\lambda x\ y\ z.x\ z\ (y\ z))(\lambda u.u)(\lambda v.v) \\
\Rightarrow_\beta\ & (\lambda y\ z.(\lambda u.u)\ z\ (y\ z))(\lambda v.v) \\
\Rightarrow_\beta\ & (\lambda y\ z.(z\ (y\ z)))(\lambda v.v) \\
\Rightarrow_\beta\ & \lambda z.(z\ ((\lambda v.v)\ z)) \\
\Rightarrow_\beta\ & \lambda z.(z\ z)
\end{aligned}$$

Beispiel Church-Zahlen (β-Reduktion):

Definition

(β-Reduktion)

$$(\lambda x.E)\ M \rightarrow_\beta E[x/M]$$

$$suc\ 1 \ \equiv \ (\lambda n\ f\ x.f\ (n\ f\ x))(\lambda g\ y.g\ y)$$

Beispiel Church-Zahlen (β-Reduktion):

Definition

(β-Reduktion)

$$(\lambda x.E)\ M \rightarrow_\beta E[x/M]$$

$$\begin{aligned} suc\ 1 \;\equiv\;& (\lambda n\ f\ x.f\ (n\ f\ x))(\lambda g\ y.g\ y) \\ \Rightarrow_\beta\;& \lambda f\ x.f\ ((\lambda g\ y.g\ y)\ f\ x) \end{aligned}$$

Beispiel Church-Zahlen (β-Reduktion):

Definition

(β-Reduktion)

$$(\lambda x.E)\ M \rightarrow_\beta E[x/M]$$

$$\begin{aligned} suc\ 1 \ \equiv &\quad (\lambda n\ f\ x.f\ (n\ f\ x))(\lambda g\ y.g\ y) \\ \Rightarrow_\beta &\quad \lambda f\ x.f\ ((\lambda g\ y.g\ y)\ f\ x) \\ \Rightarrow_\beta &\quad \lambda f\ x.f\ ((\lambda y.f\ y)\ x) \end{aligned}$$

Beispiel Church-Zahlen (β-Reduktion):

Definition

(β-Reduktion)

$$(\lambda x.E)\ M \rightarrow_\beta E[x/M]$$

$$\begin{aligned}
suc\ 1 \ \equiv &\quad (\lambda n\ f\ x.f\ (n\ f\ x))(\lambda g\ y.g\ y)\\
\Rightarrow_\beta &\quad \lambda f\ x.f\ ((\lambda g\ y.g\ y)\ f\ x)\\
\Rightarrow_\beta &\quad \lambda f\ x.f\ ((\lambda y.f\ y)\ x)\\
\Rightarrow_\beta &\quad \lambda f\ x.f\ (f\ x)
\end{aligned}$$

Beispiel Church-Zahlen (β-Reduktion):

Definition

(β-Reduktion)

$$(\lambda x.E)\ M \rightarrow_\beta E[x/M]$$

$$\begin{aligned}
suc\ 1 \ \equiv &\quad (\lambda n\ f\ x.f\ (n\ f\ x))(\lambda g\ y.g\ y) \\
\Rightarrow_\beta &\quad \lambda f\ x.f\ ((\lambda g\ y.g\ y)\ f\ x) \\
\Rightarrow_\beta &\quad \lambda f\ x.f\ ((\lambda y.f\ y)\ x) \\
\Rightarrow_\beta &\quad \lambda f\ x.f\ (f\ x) \equiv 2
\end{aligned}$$

Inhalt

Reduktionssemantik

- Basis: α–Konversion und β-Reduktion
- Definiert:
  - **Reduktionsrelation** - festlegung von Auswertungen im Kontext (unabhängig von den konkreten Reduktionsregeln)
  - **Äquivalenzrelation** - festlegung einer Gleichheit auf λ-Ausdrücken

Definition 9.6 (Reduktionsrelation; reflexive, transitive Hülle; Äquivalenzrelation) *(1)*

Sei $\rightarrow \subseteq Exp \times Exp$ eine Relation auf *Exp*.

(i) $\rightarrow$ induziert die *Reduktionsrelation*

$$\Rightarrow \subseteq Exp \times Exp$$

durch

- $\rightarrow \subseteq \Rightarrow$
- Kongruenzregeln (Reduktion im Kontext):
  - Mit $E_1 \Rightarrow E_1'$ gilt auch für beliebige $E_2 \in Exp$: $(E_1\ E_2) \Rightarrow (E_1'\ E_2)$.
  - Mit $E_2 \Rightarrow E_2'$ gilt auch für beliebige $E_1 \in Exp$: $(E_1\ E_2) \Rightarrow (E_1\ E_2')$.
  - Mit $E \Rightarrow E'$ gilt auch für beliebige $x \in Var$: $\lambda x.E \Rightarrow \lambda x.E'$.

$\Rightarrow$ definiert einen einzelnen Auswertungsschritt.

Definition 9.6 (Reduktionsrelation; reflexive, transitive Hülle; Äquivalenzrelation) *(2)*

Sei $\rightarrow \subseteq Exp \times Exp$ eine Relation auf *Exp*.

(ii) Die *reflexive, transitive Hülle*

$$\Rightarrow^* \subseteq Exp \times Exp$$

von $\Rightarrow$ ist die kleinste Relation mit

- $\Rightarrow \subseteq \Rightarrow^*$
- $E \Rightarrow^* E$ für alle $E \in Exp$. (Reflexivität)
- $E_1 \Rightarrow^* E_2$ und $E_2 \Rightarrow^* E_3$ impliziert $E_1 \Rightarrow^* E_3$. (Transitivität)

$\Rightarrow^*$ definiert Folgen aufeinanderfolgender Auswertungsschritte.

Definition 9.6 (Reduktionsrelation; reflexive, transitive Hülle; Äquivalenzrelation) *(3)*

Sei $\rightarrow \subseteq Exp \times Exp$ eine Relation auf Exp.

(iii) $\Rightarrow$ induziert eine *Äquivalenzrelation* auf Exp

$$\overset{*}{\Leftrightarrow} \subseteq Exp \times Exp$$

für die gilt:

- $\Rightarrow \subseteq \overset{*}{\Leftrightarrow}$
- $E \overset{*}{\Leftrightarrow} E$ für alle $E \in Exp$. (Reflexivität)
- $E_1 \overset{*}{\Leftrightarrow} E_2$ impliziert $E_2 \overset{*}{\Leftrightarrow} E_1$. (Symmetrie)
- $E_1 \overset{*}{\Leftrightarrow} E_2$ und $E_2 \overset{*}{\Leftrightarrow} E_3$ impliziert $E_1 \overset{*}{\Leftrightarrow} E_3$. (Transitivität)

$\overset{*}{\Leftrightarrow}$ ist die kleinste Äquivalenzrelation, die $\Rightarrow$ umfasst.

Definition 9.7 (α-Konversion und -Äquivalenz)

1. $\rightarrow_\alpha \subseteq Exp \times Exp$ ist definiert durch:

$$\lambda x.E \rightarrow_\alpha \lambda y.E[x/y] \text{ falls } y \notin free(E).$$

"Umbenennung gebundener Variablen"

2. $=_\alpha \; = \; \overset{*}{\Leftrightarrow}_\alpha$.

- Betrachte nachfolgend α-äquivalente Ausdrücke als gleich.
- Betrachte $Exp_{/=_\alpha}$ statt Exp.
- Der β-Reduktion/-Äquivalenz ($\overset{*}{\Leftrightarrow}_\beta$) liegt die Relation $\rightarrow_\alpha \cup \rightarrow_\beta$ zugrunde.

Definition 9.7 (α-Konversion und -Äquivalenz)

1. $\rightarrow_\alpha \subseteq Exp \times Exp$ ist definiert durch:

$$\lambda x.E \rightarrow_\alpha \lambda y.E[x/y] \text{ falls } y \notin free(E).$$

   "Umbenennung gebundener Variablen"

2. $=_\alpha \; = \; \overset{*}{\Leftrightarrow}_\alpha$.

- Betrachte nachfolgend α-äquivalente Ausdrücke als gleich.
- Betrachte $Exp_{/=_\alpha}$ statt Exp.
- Der β-Reduktion/-Äquivalenz ($\overset{*}{\Leftrightarrow}_\beta$) liegt die Relation $\rightarrow_\alpha \cup \rightarrow_\beta$ zugrunde.

Definition 9.7 (α-Konversion und -Äquivalenz)

1. $\rightarrow_\alpha \subseteq \mathit{Exp} \times \mathit{Exp}$ ist definiert durch:

$$\lambda x.E \rightarrow_\alpha \lambda y.E[x/y] \text{ falls } y \notin \mathit{free}(E).$$

   "Umbenennung gebundener Variablen"

2. $=_\alpha \; = \; \overset{*}{\Leftrightarrow}_\alpha$.

- Betrachte nachfolgend α-äquivalente Ausdrücke als gleich.
- Betrachte $\mathit{Exp}_{/=_\alpha}$ statt Exp.
- Der β-Reduktion/-Äquivalenz ($\overset{*}{\Leftrightarrow}_\beta$) liegt die Relation $\rightarrow_\alpha \cup \rightarrow_\beta$ zugrunde.

Definition 9.7 (α-Konversion und -Äquivalenz)

1. $\rightarrow_\alpha \subseteq Exp \times Exp$ ist definiert durch:

$$\lambda x.E \rightarrow_\alpha \lambda y.E[x/y] \text{ falls } y \notin free(E).$$

   "Umbenennung gebundener Variablen"

2. $=_\alpha \; = \; \overset{*}{\Leftrightarrow}_\alpha$.

- Betrachte nachfolgend α-äquivalente Ausdrücke als gleich.
- Betrachte $Exp_{/=_\alpha}$ statt Exp.
- Der β-Reduktion/-Äquivalenz ($\overset{*}{\Leftrightarrow}_\beta$) liegt die Relation $\rightarrow_\alpha \cup \rightarrow_\beta$ zugrunde.

Definition 9.7 (α-Konversion und -Äquivalenz)

1. $\rightarrow_\alpha \subseteq Exp \times Exp$ ist definiert durch:

$$\lambda x.E \rightarrow_\alpha \lambda y.E[x/y] \text{ falls } y \notin free(E).$$

   “Umbenennung gebundener Variablen“

2. $=_\alpha \; = \; \overset{*}{\Leftrightarrow}_\alpha$.

- Betrachte nachfolgend α-äquivalente Ausdrücke als gleich.
- Betrachte $Exp_{/=_\alpha}$ statt *Exp*.
- Der β-Reduktion/-Äquivalenz ($\overset{*}{\Leftrightarrow}_\beta$) liegt die Relation $\rightarrow_\alpha \cup \rightarrow_\beta$ zugrunde.

Inhalt

Definition 9.8 (Normalform)

Ein Ausdruck $E \in Exp_{/=_\alpha}$, für den kein $E' \in Exp_{/=_\alpha}$ existiert mit $E \Rightarrow_\beta E'$ heißt in *Normalform*.

- Normalformen sind die *"Ergebnisse"* von Auswertungen
- Unterscheide:
  - abbrechende, zu Normalformen führende Auswertungen
  - nicht-terminierende Berechnungen

Definition 9.8 (Normalform)

Ein Ausdruck $E \in Exp_{/=_\alpha}$, für den kein $E' \in Exp_{/=_\alpha}$ existiert mit $E \Rightarrow_\beta E'$ heißt in *Normalform*.

- Normalformen sind die *"Ergebnisse"* von Auswertungen
- Unterscheide:
  - abbrechende, zu Normalformen führende Auswertungen
  - nicht-terminierende Berechnungen

Definition 9.8 (Normalform)

Ein Ausdruck $E \in Exp_{/=_\alpha}$, für den kein $E' \in Exp_{/=_\alpha}$ existiert mit $E \Rightarrow_\beta E'$ heißt in *Normalform*.

- Normalformen sind die *"Ergebnisse"* von Auswertungen
- Unterscheide:
  - abbrechende, zu Normalformen führende Auswertungen
  - nicht-terminierende Berechnungen

Definition 9.8 (Normalform)

Ein Ausdruck $E \in Exp_{/=_\alpha}$, für den kein $E' \in Exp_{/=_\alpha}$ existiert mit $E \Rightarrow_\beta E'$ heißt in *Normalform*.

- Normalformen sind die *"Ergebnisse"* von Auswertungen
- Unterscheide:
  - abbrechende, zu Normalformen führende Auswertungen
  - nicht-terminierende Berechnungen

Beispiel (Auswertungsfolgen)

- $(\lambda x.(\lambda y.x)\ a)\ z \Rightarrow_\beta (\lambda y.z)\ a \Rightarrow_\beta z$
  $\leadsto$ z ist Normalform von $(\lambda x.(\lambda y.x)\ a)\ z$.
- $(\lambda x.x\ x)\ \lambda x.x\ x$
  $\Rightarrow (\lambda x.x\ x)\ \lambda x.x\ x \Rightarrow \ldots$ nichtterminierende Berechnung
  Es existiert keine Normalform.

Beispiel (Auswertungsfolgen)

- $(\lambda x.(\lambda y.x)\ a)\ z \Rightarrow_\beta (\lambda y.z)\ a \Rightarrow_\beta z$
  $\rightsquigarrow$ z ist Normalform von $(\lambda x.(\lambda y.x)\ a)\ z$.
- $(\lambda x.x\ x)\ \lambda x.x\ x$
  $\Rightarrow (\lambda x.x\ x)\ \lambda x.x\ x \Rightarrow \ldots$ nichtterminierende Berechnung
  Es existiert keine Normalform.

Beispiel (Auswertungsfolgen)

- $(\lambda x.(\lambda y.x)\ a)\ z \Rightarrow_\beta (\lambda y.z)\ a \Rightarrow_\beta z$
  $\rightsquigarrow$ z ist Normalform von $(\lambda x.(\lambda y.x)\ a)\ z$.
- $(\lambda x.x\ x)\ \lambda x.x\ x$
  $\Rightarrow (\lambda x.x\ x)\ \lambda x.x\ x \Rightarrow \ldots$ nichtterminierende Berechnung
  Es existiert keine Normalform.

Beispiel (Auswertungsfolgen)

- $(\lambda x.(\lambda y.x)\ a)\ z \Rightarrow_\beta (\lambda y.z)\ a \Rightarrow_\beta z$
  $\rightsquigarrow$ z ist Normalform von $(\lambda x.(\lambda y.x)\ a)\ z$.
- $(\lambda x.x\ x)\ \lambda x.x\ x$
  $\Rightarrow (\lambda x.x\ x)\ \lambda x.x\ x \Rightarrow \ldots$ nichtterminierende Berechnung
  Es existiert keine Normalform.

Beispiel (Auswertungsfolgen)

- $(\lambda x.(\lambda y.x)\ a)\ z \Rightarrow_\beta (\lambda y.z)\ a \Rightarrow_\beta z$
  $\leadsto$ z ist Normalform von $(\lambda x.(\lambda y.x)\ a)\ z$.
- $(\lambda x.x\ x)\ \lambda x.x\ x$
  $\Rightarrow (\lambda x.x\ x)\ \lambda x.x\ x \Rightarrow \ldots$ nichtterminierende Berechnung
  Es existiert keine Normalform.

Beispiel (Auswertungsfolgen)

- $(\lambda x.(\lambda y.x)\ a)\ z \Rightarrow_\beta (\lambda y.z)\ a \Rightarrow_\beta z$
  $\rightsquigarrow$ z ist Normalform von $(\lambda x.(\lambda y.x)\ a)\ z$.
- $(\lambda x.x\ x)\ \lambda x.x\ x$
  $\Rightarrow (\lambda x.x\ x)\ \lambda x.x\ x \Rightarrow \ldots$ nichtterminierende Berechnung
  Es existiert keine Normalform.

Beispiel (Auswertungsfolgen)

- $(\lambda x.(\lambda y.x)\ a)\ z \Rightarrow_\beta (\lambda y.z)\ a \Rightarrow_\beta z$
  $\leadsto$ z ist Normalform von $(\lambda x.(\lambda y.x)\ a)\ z$.
- $(\lambda x.x\ x)\ \lambda x.x\ x$
  $\Rightarrow (\lambda x.x\ x)\ \lambda x.x\ x \Rightarrow \ldots$ nichtterminierende Berechnung
  Es existiert keine Normalform.

Beispiel (Auswertungsfolgen)

- $(\lambda x.(\lambda y.x)\ a)\ z \Rightarrow_\beta (\lambda y.z)\ a \Rightarrow_\beta z$
  $\rightsquigarrow$ z ist Normalform von $(\lambda x.(\lambda y.x)\ a)\ z$.
- $(\lambda x.x\ x)\ \lambda x.x\ x$
  $\Rightarrow (\lambda x.x\ x)\ \lambda x.x\ x \Rightarrow \ldots$ nichtterminierende Berechnung
  Es existiert keine Normalform.

Church-Rosser-Eigenschaft:

Die Reihenfolge, in der β-Reduktionen durchgeführt werden, hat keinen Einfluss auf das Ergebnis...

Satz 9.1 (Church-Rosser, Konfluenz)

Für alle $E, E_1, E_2 \in \mathit{Exp}$ gilt:
Falls $E \Rightarrow_\beta^* E_1$ und $E \Rightarrow_\beta^* E_2$, so existiert ein $E' \in \mathit{Exp}$ mit

$$E_1 \Rightarrow_\beta^* E' \text{ und } E_2 \Rightarrow_\beta^* E'.$$

Man sagt: Die Reduktionsrelation ist *konfluent*.

Auch genannt: Diamanteigenschaft

$$\begin{array}{ccc} & E & \\ \swarrow^{*} & & \searrow^{*} \\ E_1 & & E_2 \\ \searrow^{*} & & \swarrow^{*} \\ & E' & \end{array}$$

Church-Rosser-Eigenschaft:

Die Reihenfolge, in der β-Reduktionen durchgeführt werden, hat keinen Einfluss auf das Ergebnis, sofern die Reduktionsfolge terminiert.

Satz 9.1 (Church-Rosser, Konfluenz)

Für alle $E, E_1, E_2 \in Exp$ gilt:
Falls $E \Rightarrow_\beta^* E_1$ und $E \Rightarrow_\beta^* E_2$, so existiert ein $E' \in Exp$ mit

$$E_1 \Rightarrow_\beta^* E' \text{ und } E_2 \Rightarrow_\beta^* E'.$$

Man sagt: Die Reduktionsrelation ist *konfluent*.

Auch genannt: Diamanteigenschaft

$$\begin{tikzcd} & E \arrow[dl, "*"] \arrow[dr, "*"] & \\ E_1 \arrow[dr, "*"] & & E_2 \arrow[dl, "*"] \\ & E' & \end{tikzcd}$$

Church-Rosser-Eigenschaft:

Die Reihenfolge, in der β-Reduktionen durchgeführt werden, hat keinen Einfluss auf das Ergebnis, sofern die Reduktionsfolge terminiert.

Satz 9.1 (Church-Rosser, Konfluenz)

Für alle $E, E_1, E_2 \in Exp$ gilt:
Falls $E \Rightarrow^*_\beta E_1$ und $E \Rightarrow^*_\beta E_2$, so existiert ein $E' \in Exp$ mit

$$E_1 \Rightarrow^*_\beta E' \text{ und } E_2 \Rightarrow^*_\beta E'.$$

Man sagt: Die Reduktionsrelation ist *konfluent*.

Auch genannt: Diamanteigenschaft

$$\begin{tikzcd}
 & E \arrow[dl, "*"] \arrow[dr, "*"] & \\
E_1 \arrow[dr, "*"'] & & E_2 \arrow[dl, "*"] \\
 & E' &
\end{tikzcd}$$

Beweisansatz

- Beweis sehr aufwändig!
- Vorgehensweise:
  - Zeige zunächst lokale Konfluenz: $$\begin{tikzcd} & E \arrow[dl] \arrow[dr] & \\ E_1 \arrow[dr, "*"] & & E_2 \arrow[dl, "*"] \\ & E' & \end{tikzcd}$$
  - ... und schließe damit auf die allgemeine Aussage

Beweisansatz

- Beweis sehr aufwändig!
- Vorgehensweise:
  - Zeige zunächst lokale Konfluenz:
$$\begin{tikzcd}[column sep=small] & E \arrow[dl] \arrow[dr] & \\ E_1 \arrow[dr, "*"] & & E_2 \arrow[dl, "*"'] \\ & E' & \end{tikzcd}$$
  - ... und schließe damit auf die allgemeine Aussage

Beweisansatz

- Beweis sehr aufwändig!
- Vorgehensweise:
  - Zeige zunächst lokale Konfluenz:
    $$\begin{tikzcd} & E \arrow[dl] \arrow[dr] & \\ E_1 \arrow[dr, "*"] & & E_2 \arrow[dl, "*"'] \\ & E' & \end{tikzcd}$$
  - ... und schließe damit auf die allgemeine Aussage

Korollar 9.2

Für alle $E_1, E_2 \in \mathit{Exp}$ gilt:

$$E_1 \overset{*}{\Leftrightarrow}_\beta E_2 \text{ gilt genau dann, wenn } \exists E_3 \in \mathit{Exp} : E_1 \Rightarrow^* E_3 \wedge E_2 \Rightarrow^* E_3.$$

Korollar 9.2

Für alle $E_1, E_2 \in Exp$ gilt:

$$E_1 \overset{*}{\Leftrightarrow}_\beta E_2 \text{ gilt genau dann, wenn } \exists E_3 \in Exp : E_1 \Rightarrow^* E_3 \wedge E_2 \Rightarrow^* E_3.$$

Beweis.

induktiv über die Definition von $\overset{*}{\Leftrightarrow}_\beta$:

1. Fall: $E_1 \Rightarrow^* E_2 \curvearrowright$ Wähle $E_3 = E_2$. $\qquad \Rightarrow \subseteq \overset{*}{\Leftrightarrow}$
2. Fall: $E_1 = E_2$. ✓ (Reflexivität)
3. Fall: $E_1 \overset{*}{\Leftrightarrow} E_2$ wegen $E_2 \overset{*}{\Leftrightarrow} E_1$. Nutze die IV. (Symmetrie)
4. Fall: $E_1 \overset{*}{\Leftrightarrow} E_2$ wegen $E_1 \overset{*}{\Leftrightarrow} E_3$ und $E_3 \overset{*}{\Leftrightarrow} E_2$. (Transitivität)
   Aus der IV. folgt die Existenz von E_1' und E_2' mit

$$\begin{tikzcd}
E_1 \arrow[dr, "*"] & \overset{*}{\Leftrightarrow} & E_3 \arrow[dl, "*"'] \arrow[dr, "*"] & \overset{*}{\Leftrightarrow} & E_2 \arrow[dl, "*"'] \\
& E_1' \arrow[dr, "*"] & & E_2' \arrow[dl, "*"'] & \\
& & E' & &
\end{tikzcd}$$

laut Induktionsvoraussetzung

mit Church-Rosser-Eigenschaft für E_3

□

Korollar 9.2

Für alle $E_1, E_2 \in Exp$ gilt:

$$E_1 \overset{*}{\Leftrightarrow}_\beta E_2 \text{ gilt genau dann, wenn } \exists E_3 \in Exp : E_1 \Rightarrow^* E_3 \wedge E_2 \Rightarrow^* E_3.$$

Beweis.

induktiv über die Definition von $\overset{*}{\Leftrightarrow}_\beta$:

1. Fall: $E_1 \Rightarrow^* E_2 \curvearrowright$ Wähle $E_3 = E_2$. $\qquad \Rightarrow \subseteq \overset{*}{\Leftrightarrow}$
2. Fall: $E_1 = E_2$. ✓ (Reflexivität)
3. Fall: $E_1 \overset{*}{\Leftrightarrow} E_2$ wegen $E_2 \overset{*}{\Leftrightarrow} E_1$. Nutze die IV. (Symmetrie)
4. Fall: $E_1 \overset{*}{\Leftrightarrow} E_2$ wegen $E_1 \overset{*}{\Leftrightarrow} E_3$ und $E_3 \overset{*}{\Leftrightarrow} E_2$. (Transitivität)
   Aus der IV. folgt die Existenz von E_1' und E_2' mit

$$\begin{tikzcd}
E_1 \arrow[dr, "*"] & \overset{*}{\Leftrightarrow} & E_3 \arrow[dl, "*"'] \arrow[dr, "*"] & \overset{*}{\Leftrightarrow} & E_2 \arrow[dl, "*"'] \\
& E_1' \arrow[dr, "*"] & & E_2' \arrow[dl, "*"'] & \\
& & E' & &
\end{tikzcd}$$

laut Induktionsvoraussetzung

mit Church-Rosser-Eigenschaft für E_3

Korollar 9.2

Für alle $E_1, E_2 \in \mathit{Exp}$ gilt:

$$E_1 \overset{*}{\Leftrightarrow}_\beta E_2 \text{ gilt genau dann, wenn } \exists E_3 \in \mathit{Exp} : E_1 \Rightarrow^* E_3 \wedge E_2 \Rightarrow^* E_3.$$

Beweis.

induktiv über die Definition von $\overset{*}{\Leftrightarrow}_\beta$:

1. Fall: $E_1 \Rightarrow^* E_2 \curvearrowright$ Wähle $E_3 = E_2$. $\Rightarrow \subseteq \overset{*}{\Leftrightarrow}$
2. Fall: $E_1 = E_2$. ✓ (Reflexivität)
3. Fall: $E_1 \overset{*}{\Leftrightarrow} E_2$ wegen $E_2 \overset{*}{\Leftrightarrow} E_1$. Nutze die IV. (Symmetrie)
4. Fall: $E_1 \overset{*}{\Leftrightarrow} E_2$ wegen $E_1 \overset{*}{\Leftrightarrow} E_3$ und $E_3 \overset{*}{\Leftrightarrow} E_2$. (Transitivität)
   Aus der IV. folgt die Existenz von E_1' und E_2' mit

$$\begin{tikzcd}
E_1 \arrow[dr, "*"] & \overset{*}{\Leftrightarrow} & E_3 \arrow[dl, "*"'] \arrow[dr, "*"] & \overset{*}{\Leftrightarrow} & E_2 \arrow[dl, "*"'] \\
& E_1' \arrow[dr, "*"] & & E_2' \arrow[dl, "*"'] & \\
& & E' & &
\end{tikzcd}$$

laut Induktionsvoraussetzung

mit Church-Rosser-Eigenschaft für E_3

Korollar 9.2

Für alle $E_1, E_2 \in Exp$ gilt:

$$E_1 \overset{*}{\Leftrightarrow}_\beta E_2 \text{ gilt genau dann, wenn } \exists E_3 \in Exp : E_1 \Rightarrow^* E_3 \wedge E_2 \Rightarrow^* E_3.$$

Beweis.

induktiv über die Definition von $\overset{*}{\Leftrightarrow}_\beta$:

1. Fall: $E_1 \Rightarrow^* E_2 \curvearrowright$ Wähle $E_3 = E_2$. $\quad \Rightarrow \subseteq \overset{*}{\Leftrightarrow}$
2. Fall: $E_1 = E_2$. ✓ (Reflexivität)
3. Fall: $E_1 \overset{*}{\Leftrightarrow} E_2$ wegen $E_2 \overset{*}{\Leftrightarrow} E_1$. Nutze die IV. (Symmetrie)
4. Fall: $E_1 \overset{*}{\Leftrightarrow} E_2$ wegen $E_1 \overset{*}{\Leftrightarrow} E_3$ und $E_3 \overset{*}{\Leftrightarrow} E_2$. (Transitivität)

   Aus der IV. folgt die Existenz von E_1' und E_2' mit

$$\begin{tikzcd}[column sep=small]
E_1 \arrow[dr, "*"] & \overset{*}{\Leftrightarrow} & E_3 \arrow[dl, "*"'] \arrow[dr, "*"] & \overset{*}{\Leftrightarrow} & E_2 \arrow[dl, "*"'] \\
& E_1' \arrow[dr, "*"] & & E_2' \arrow[dl, "*"'] & \\
& & E' & &
\end{tikzcd}$$

laut Induktionsvoraussetzung

mit Church-Rosser-Eigenschaft für E_3

□

Korollar 9.2

Für alle $E_1, E_2 \in Exp$ gilt:

$$E_1 \overset{*}{\Leftrightarrow}_\beta E_2 \text{ gilt genau dann, wenn } \exists E_3 \in Exp : E_1 \Rightarrow^* E_3 \wedge E_2 \Rightarrow^* E_3.$$

Beweis.

induktiv über die Definition von $\overset{*}{\Leftrightarrow}_\beta$:

1. Fall: $E_1 \Rightarrow^* E_2 \curvearrowright$ Wähle $E_3 = E_2$. $\qquad \Rightarrow \subseteq \overset{*}{\Leftrightarrow}$
2. Fall: $E_1 = E_2$. ✓ (Reflexivität)
3. Fall: $E_1 \overset{*}{\Leftrightarrow} E_2$ wegen $E_2 \overset{*}{\Leftrightarrow} E_1$. Nutze die IV. (Symmetrie)
4. Fall: $E_1 \overset{*}{\Leftrightarrow} E_2$ wegen $E_1 \overset{*}{\Leftrightarrow} E_3$ und $E_3 \overset{*}{\Leftrightarrow} E_2$. (Transitivität)
   Aus der IV. folgt die Existenz von E_1' und E_2' mit

$$\begin{array}{ccccccccc} E_1 & & \overset{*}{\Leftrightarrow} & & E_3 & & \overset{*}{\Leftrightarrow} & & E_2 \\ & \searrow^* & & \swarrow^* & & \searrow^* & & \swarrow^* & \\ & & E_1' & & & & E_2' & & \\ & & & \searrow^* & & \swarrow^* & & & \\ & & & & E' & & & & \end{array}$$

laut Induktionsvoraussetzung

mit Church-Rosser-Eigenschaft für E_3 $\square$

Korollar 9.3 (Eindeutigkeit der Normalform)

Jeder Ausdruck $E \in Exp$ besitzt höchstens eine Normalform $E' \in Exp_{/=_\alpha}$
(Eindeutigkeit ist bis auf Umbenennung gebundener Variablen)

Korollar 9.3 (Eindeutigkeit der Normalform)

Jeder Ausdruck $E \in Exp$ besitzt höchstens eine Normalform $E' \in Exp_{/=_\alpha}$ (Eindeutigkeit ist bis auf Umbenennung gebundener Variablen)

Beweis.

Annahme, E besitze zwei Normalformen E_1 und E_2, d.h.

$$E_1 \; {}^* \!\Leftarrow E \Rightarrow^* E_2 \curvearrowright E_1 \overset{*}{\Leftrightarrow} E_2.$$

Aus dem vorherigen Korollar folgt die Existenz eines E' mit $E_1 \Rightarrow^* E' \; {}^* \!\Leftarrow E_2$. Da E_1 und E_2 Normalformen sind, kann nur $E_1 = E' = E_2$ gelten. □

Beispiel

Seien
$$\begin{array}{lcl} K & = & \lambda x\, y.x \qquad . \\ I & = & \lambda x.x \\ \Delta & = & (\lambda x.x\, x)\, \lambda x.x\, x \end{array}$$

- Der Ausdruck $K\ I\ \Delta$ hat die Normalform. I
- Der Ausdruck Δ hat keine Normalform.
- Nicht jede Reduktionsfolge von $K\ I\ \Delta$ führt zur Normalform.

$\Longrightarrow$ Reduktionsstrategien:

- **normal order** (leftmost outermost)
- **applicative-order** (rightmost innermost)

Die normal-order Strategie ist vollständig, d.h. bestimmt die Normalform falls existent.

Beispiel

Seien
$$K = \lambda x\, y.x \quad .$$
$$I = \lambda x.x$$
$$\Delta = (\lambda x.x\, x)\, \lambda x.x\, x$$

- Der Ausdruck $K\ I\ \Delta$ hat die Normalform. I
- Der Ausdruck Δ hat keine Normalform.
- Nicht jede Reduktionsfolge von $K\ I\ \Delta$ führt zur Normalform.

$\Longrightarrow$ Reduktionsstrategien:

- **normal order** (leftmost outermost)
- **applicative-order** (rightmost innermost)

Die normal-order Strategie ist vollständig, d.h. bestimmt die Normalform falls existent.

Beispiel

Seien
$$\begin{aligned} K &= \lambda x\, y.x \quad . \\ I &= \lambda x.x \\ \Delta &= (\lambda x.x\, x)\, \lambda x.x\, x \end{aligned}$$

- Der Ausdruck $K\ I\ \Delta$ hat die Normalform. I
- Der Ausdruck Δ hat keine Normalform.
- Nicht jede Reduktionsfolge von $K\ I\ \Delta$ führt zur Normalform.

$\Longrightarrow$ Reduktionsstrategien:

- **normal order** (leftmost outermost)
- **applicative-order** (rightmost innermost)

Die normal-order Strategie ist vollständig, d.h. bestimmt die Normalform falls existent.

Beispiel

Seien
$$\begin{array}{lcl} K & = & \lambda x\, y.x \quad . \\ I & = & \lambda x.x \\ \Delta & = & (\lambda x.x\, x)\, \lambda x.x\, x \end{array}$$

- Der Ausdruck $K\ I\ \Delta$ hat die Normalform. I
- Der Ausdruck Δ hat keine Normalform.
- Nicht jede Reduktionsfolge von $K\ I\ \Delta$ führt zur Normalform.

$\Longrightarrow$ Reduktionsstrategien:

- **normal order** (leftmost outermost)
- **applicative-order** (rightmost innermost)

Die normal-order Strategie ist vollständig, d.h. bestimmt die Normalform falls existent.

Beispiel

Seien
$$\begin{array}{lcl} K & = & \lambda x\, y.x \\ I & = & \lambda x.x \\ \Delta & = & (\lambda x.x\, x)\, \lambda x.x\, x \end{array}$$

- Der Ausdruck $K\ I\ \Delta$ hat die Normalform. I
- Der Ausdruck Δ hat keine Normalform.
- Nicht jede Reduktionsfolge von $K\ I\ \Delta$ führt zur Normalform.

$\Longrightarrow$ Reduktionsstrategien:

- **normal order** (leftmost outermost)
- **applicative-order** (rightmost innermost)

Die normal-order Strategie ist vollständig, d.h. bestimmt die Normalform falls existent.

Beispiel

Seien

$$\begin{array}{lcl} K & = & \lambda x\, y.x \qquad . \\ I & = & \lambda x.x \\ \Delta & = & (\lambda x.x\ x)\ \lambda x.x\ x \end{array}$$

- Der Ausdruck $K\ I\ \Delta$ hat die Normalform. I
- Der Ausdruck Δ hat keine Normalform.
- Nicht jede Reduktionsfolge von $K\ I\ \Delta$ führt zur Normalform.

$\Longrightarrow$ Reduktionsstrategien:

- **normal order** (leftmost outermost)
- **applicative-order** (rightmost innermost)

Die normal-order Strategie ist vollständig, d.h. bestimmt die Normalform falls existent.