# Lambda-Kalkül und Kombinatorische Logik

Peter Schroeder-Heister

Sommersemester 1997

Skriptum von Michael Arndt

1997
Universität Tübingen
Wilhelm-Schickard-Institut für Informatik

# Inhaltsverzeichnis

# 0 Vorwort

Dies ist ein Skriptum zu einer Vorlesung, die ich zuletzt im Sommersemester 1997 gehalten habe. In den ersten beiden Teilen orientiert es sich im wesentlichen am klassischen Lehrbuch von Hindley und Seldin, in den letzten beiden Teilen an Barendregts Kapitel über den getypten $\lambda$-Kalkül im *Handbook of Logic in Computer Science* (Band II). Das Skriptum soll zur Orientierung über das technische Gerüst des Themas dienen. Dementsprechend ist es nicht bis in alle Einzelheiten ausgearbeitet. So wurde auf Stilfragen wenig Rücksicht genommen. Auch wurden elementare, aber langwierige Beweise häufig weggelassen. Erläuternde Passagen zu Sinn und Zweck des $\lambda$-Kalküls sowie einzelner Begriffsbildungen sind ebenfalls nicht aufgezeichnet. Hierzu seien Leser auf die genannten Texte verwiesen.

Ich danke Michael Arndt für die Erstellung des Skriptums. Frau Natali Alt und Herrn Reinhard Kahle danke ich für eine kritische Durchsicht des Textes. Alle verbleibenden inhaltlichen Fehler gehen natürlich zu meinen Lasten.

Peter Schroeder-Heister

# 1 Der ungetypte $\lambda$-Kalkül

## 1.1 Syntax

Gegeben sei eine unendliche Folge von Variablen. (Es ist wichtig, daß eine feste Reihenfolge angenommen wird.) Die metasprachlichen Zeichen dafür seien $x, y, z, x_1, x_2, x_3, \ldots$
Man unterscheidet zwischen zwei Varianten des ungetypten $\lambda$-Kalküls:

- dem reinen $\lambda$-Kalkül, bei dem keine Konstanten gegeben sind.
- dem angewandten $\lambda$-Kalkül, bei dem zusätzlich eine endliche oder unendliche Menge von Konstanten gegeben ist.

(In den ersten zwei Kapiteln wird nur der ungetypte $\lambda$-Kalkül behandelt. Daher wird die Bezeichnung "ungetypt" immer weggelassen.)

**Definition 1.1 (Syntax)**

- *Alle Variablen und Konstanten sind $\lambda$-Terme (*"Atome"*)*
- *Mit $M$ und $N$ ist auch $(MN)$ ein $\lambda$-Term (*"Applikation"*) mit $M$ und $N$ als unmittelbaren Teiltermen*
- *Mit $M$ ist auch $(\lambda x.M)$ ein $\lambda$-Term (*"Abstraktion"*) mit $x$ und $M$ als unmittelbaren Teiltermen*

*Die* Länge *eines Termes $M$ ist die Anzahl der Vorkommen von Atomen in $M$.*
Teilterme *eines Terms sind dieser Term selbst, sowie die Teilterme seiner echten Teilterme.*
*Man schreibt $M[P]$, wenn $P$ an einer bestimmten Stelle als Teilterm in $M$ vorkommt.*
*Im Kontext von $M[P]$ bedeute $M[Q]$, daß man das in $M[P]$ gemeinte Vorkommen von $P$ in $M$ durch $Q$ ersetzt.*
*Ein Vorkommen einer Variable $x$ in einem Term $M$ ist* gebunden*, falls es zu einem Teilterm $\lambda x.P$ von $M$ gehört, ansonsten ist es* frei.
*Falls $x$ ein freies Vorkommen in $M$ hat, heißt $x$* freie Variable *von $M$. Die Menge dieser freien Variablen sei $FV(M)$.*
*$M$ heißt* geschlossen*, wenn $FV(M) = \emptyset$ .*
*Ein geschlossener Term ohne Konstanten heißt* Kombinator.
Metasprachliche Variablen: *$M, N, P, Q, R, S, T, \ldots$ für $\lambda$-Terme; $a, b, c, \ldots$ für Atome.*
*Außenklammern können wegfallen. Bei Klammerung gilt Linksassoziation, d.h. $MNPQ$ meint $((MN)P)Q$. Ferner steht $\lambda x.MN$ für $(\lambda x.(MN))$, $\lambda x_1 \ldots\ x_n.M$ für $\lambda x_1.\lambda x_2.\ldots\lambda x_n.M$.*
*$M \equiv N$ bezeichne die syntaktische Identität von $M$ und $N$.*

**Beispiel 1.2 (Grammatik für Terme)**
*Die Terme des reinen $\lambda$-Kalküls können durch folgende kontextfreie Grammatik charakterisiert werden, wenn Variablen die Form $v'^{\cdots'}$ haben:*

- *Terminalalphabet:* $\{\lambda, ., (, ), v, '\}$
- *Nichtterminalalphabet:* $\{L, V\}$
- *Startsymbol:* $L$
- *Produktionen:* $\begin{array}{lcl} L & \longrightarrow & V \mid (LL) \mid (\lambda V.L) \\ V & \longrightarrow & v \mid V' \end{array}$

**Beispiel 1.3 (Kombinatoren)**

- $I :\equiv \lambda x.x$
- $K :\equiv \lambda xy.x$
- $S :\equiv \lambda xyz.xz(yz)$

**Definition 1.4 (Substitution)**

1. $x[^N/_x] \equiv N$
2. $a[^N/_x] \equiv a$, *falls* $x \not\equiv a$
3. $(PQ)[^N/_x] \equiv (P[^N/_x]Q[^N/_x])$
4. $(\lambda x.P)[^N/_x] \equiv \lambda x.P$
5. $(\lambda y.P)[^N/_x] \equiv \lambda y.P[^N/_x]$, *falls* $x \not\equiv y$ *und nicht:* $y \in FV(N)$ *und* $x \in FV(P)$
6. $(\lambda y.P)[^N/_x] \equiv \lambda z.P[^z/_y][^N/_x]$, *falls* $x \not\equiv y$ *und* $y \in FV(N)$ *und* $x \in FV(P)$, *wobei* $z$ *die erste Variable (in der Aufzählung aller Variablen) mit* $z \notin FV(NP)$

**Beispiel 1.5**
$(\lambda y.x)[^y/_x] \equiv \lambda z.y$ *(falls* $z$ *erste von* $x$ *und* $y$ *verschiedene Variable)*

**Definition 1.6 ($\alpha$-Konversion, Kongruenz)**
*Falls* $y \notin FV(M)$, *so sei* $P[\lambda x.M] \equiv_{1\alpha} P[\lambda y.M[^y/_x]]$ "gebundene Umbenennung"
$P \equiv_\alpha Q$, *falls* $P \equiv P_1 \equiv_{1\alpha} P_2 \equiv_{1\alpha} \cdots \equiv_{1\alpha} P_n \equiv Q$ "$\alpha$-Konversion", "Kongruenz"

**Lemma 1.7**

1. $P \equiv_\alpha Q \implies FV(P) = FV(Q)$

2. *Für jedes* $P$ *und alle* $x_1, \ldots, x_n$ *existiert* $P'$ *mit* $P \equiv_\alpha P'$*, wobei kein* $x_1, \ldots, x_n$ *gebunden in* $P'$

3. $\equiv_\alpha$ *ist Äquivalenzrelation*

Beweis
Übung □

**Lemma 1.8 (Kongruenz von $\equiv_\alpha$)**
*Wenn* $M \equiv_\alpha M'$ *und* $N \equiv_\alpha N'$*, dann* $M[{}^N\!/x] \equiv_\alpha M'[{}^{N'}\!/x]$.

**Definition 1.9 ($\beta$-Kontraktion, $\beta$-Reduktion, $\beta$-Konversion)**
$P[\underbrace{(\lambda x.M)N}_{Redex}] \rhd_{1\beta} P[\underbrace{M[{}^N\!/x]}_{Kontraktum}]$ "$\beta$-Kontraktion"

$P \rhd_\beta Q$*, falls* $P \equiv P_1 \overset{\rhd_{1\beta}}{\equiv_{1\alpha}} P_2 \overset{\rhd_{1\beta}}{\equiv_{1\alpha}} \cdots \overset{\rhd_{1\beta}}{\equiv_{1\alpha}} P_n \equiv Q$ "$\beta$-Reduktion"

*Falls* $P \equiv P_1 \rhd_{1\beta} P_2 \rhd_{1\beta} P_3 \rhd_{1\beta} \cdots$*, dann heißt* $(P_1, P_2, P_3, \ldots)$ $\beta$-Reduktionsfolge *von* $P$.

$P =_\beta Q$*, falls* $P \equiv P_1 \begin{smallmatrix}\equiv_{1\alpha}\\ \rhd_{1\beta}\\ \lhd_{1\beta}\end{smallmatrix} P_2 \begin{smallmatrix}\equiv_{1\alpha}\\ \rhd_{1\beta}\\ \lhd_{1\beta}\end{smallmatrix} \cdots \begin{smallmatrix}\equiv_{1\alpha}\\ \rhd_{1\beta}\\ \lhd_{1\beta}\end{smallmatrix} P_n \equiv Q$ "$\beta$-Konversion", "$\beta$-Gleichheit"

$P$ *ist in* $\beta$-Normalform, *falls* $P$ *kein* $\beta$*-Redex enthält.*
*Falls* $P \rhd_\beta Q$ *und* $Q$ *in* $\beta$*-Normalform ist, dann heißt* $Q$ *eine* $\beta$*-Normalform* von $P$.
$P$ *heißt* (schwach) normalisierbar, *wenn es eine* $\beta$*-Normalform von* $P$ *gibt.*
$P$ *heißt* stark normalisierbar, *wenn es keine unendliche* $\beta$*-Reduktionsfolge von* $P$ *gibt.*

**Beispiel 1.10**

- $(\lambda x.(\lambda y.yx)z)v \rhd_{1\beta} (\lambda x.zx)v \rhd_{1\beta} zv$
  $zv$ *ist* $\beta$*-Normalform von* $(\lambda x.(\lambda y.yx)z)v$.

- $\Omega \equiv (\lambda x.xx)(\lambda x.xx)$ *hat keine* $\beta$*-Normalform:*
  $(\lambda x.xx)(\lambda x.xx) \rhd_{1\beta} (\lambda x.xx)(\lambda x.xx) \rhd_{1\beta} \cdots$
  *Allerdings können* $\lambda$*-Terme, die den* $\Omega$*-Kombinator enthalten, eine* $\beta$*-Normalform besitzen:*
  $(\lambda x.y)\Omega \rhd_{1\beta} y$*. Der Term* $(\lambda x.y)\Omega$ *ist also schwach normalisierbar, jedoch nicht stark normalisierbar, da es eine unendliche Reduktionsfolge gibt:* $(\lambda x.y)\Omega \rhd_{1\beta} (\lambda x.y)\Omega \rhd_{1\beta} \ldots$

**Lemma 1.11**

- *Wenn* $P \equiv_\alpha P'$, $Q \equiv_\alpha Q'$, $P \overset{\rhd_\beta}{=_\beta} Q$*, dann* $P' \overset{\rhd_\beta}{=_\beta} Q'$

- *Wenn* $P \overset{\rhd_\beta}{=_\beta} Q$, *dann* $M[{}^{P}\!/\!_{x}] \overset{\rhd_\beta}{=_\beta} M[{}^{Q}\!/\!_{x}]$
- *Wenn* $P \overset{\rhd_\beta}{=_\beta} Q$, *dann* $P[{}^{N}\!/\!_{x}] \overset{\rhd_\beta}{=_\beta} Q[{}^{N}\!/\!_{x}]$

**Lemma 1.12**
*Die Klasse aller $\beta$-Normalformen läßt sich induktiv definieren durch folgende Regeln:*

1. *Jedes Atom ist eine $\beta$-Normalform.*
2. *Mit $M_1, \ldots, M_n$ ist auch $aM_1 \ldots M_n$ eine $\beta$-Normalform.*
3. *Mit $M$ ist auch $\lambda x.M$ eine $\beta$-Normalform.*

*Das heißt, eine $\beta$-Normalform hat die Form $\lambda x_1 \ldots x_n.aM_1 \ldots M_m$, wobei die $M_i$ dieselbe Form haben.*

Beweis
Sei $M$ eine $\beta$-Normalform. Falls $M \equiv a$, so läßt sich $M$ nach *1.* erzeugen. Falls $M \equiv (PQ)$, dann sind nach Induktionsvoraussetztung $P$ und $Q$ mit den Regeln *1.–3.* zu erzeugen, wobei $P$ keine Abstraktion ist. Also ist $P \equiv a$ oder $P \equiv aM_1 \ldots M_k$. Damit ist $M \equiv aQ$ oder $M \equiv aM_1 \ldots M_kQ$. Das läßt sich nach *2.* erzeugen. Falls $M \equiv \lambda x.P$, dann läßt sich $P$ nach Induktionsvoraussetztung aus *1.–3.* erzeugen, daraus also auch $M$ mit *3.*.
Falls umgekehrt $M$ mit den Regeln *1.–3.* erzeugt ist, dann ist klar, daß $M$ in $\beta$-Normalform ist.
□

**Lemma 1.13**
*Ein beliebiger $\lambda$-Term hat entweder die Form $\lambda x_1 \ldots x_n.aM_1 \ldots M_m$ $(m, n \geq 0)$, oder er hat die Form $\lambda x_1 \ldots x_n.\underbrace{(\lambda x.M)N}_{Kopfredex} M_1 \ldots M_m$ $(m, n \geq 0)$.*

Beweis
Ähnlich wie der Beweis des vorigen Lemmas. □

**Bemerkung**
Wenn ein Term die Form $\lambda x_1 \ldots x_n.aM_1 \ldots M_m$ hat, sagt man auch, daß er sich in *Kopf-Normalform* befindet.

**Theorem 1.14 (Church-Rosser)**

1. *Wenn $P \rhd_\beta M$ und $P \rhd_\beta N$, dann existiert ein Term $T$, so daß $M \rhd_\beta T$ und $N \rhd_\beta T$.*
2. *Wenn $M =_\beta N$, so existiert ein Term $T$, so daß $M \rhd_\beta T$ und $N \rhd_\beta T$.*

**Beispiel 1.15**

$$\begin{tikzcd}
& (\lambda x.(\lambda y.yx)z)v \arrow[dl, "\rhd_{1\beta}"'] \arrow[dr, "\rhd_{1\beta}"] & \\
(\lambda x.zx)v \arrow[dr, "\rhd_{1\beta}"'] & & (\lambda y.yv)z \arrow[dl, "\rhd_{1\beta}"] \\
& zv &
\end{tikzcd}$$

Beweis

von 2. durch Induktion über der Anzahl der Schritte von $M$ nach $N$.

**Anzahl = 0:** trivial

**Anzahl = $n+1$:**

$$\begin{tikzcd}
M \equiv P_1 \arrow[dr, "\rhd_\beta"'] & \cdots & P_n \arrow[dl, "\rhd_\beta"] \arrow[r, phantom, "\substack{\equiv_{1\alpha} \\ \rhd_{1\beta} \\ \lhd_{1\beta}}"] & P_{n+1} \equiv N \\
& T' & &
\end{tikzcd}$$

**Fall 1 ($\equiv_{1\alpha}$):** $T \equiv_{def} T'$, da $P_{n+1} \equiv_{1\alpha} P_n \rhd_\beta T'$.

**Fall 2 ($\lhd_{1\beta}$):** $T \equiv_{def} T'$, da $P_{n+1} \rhd_{1\beta} P_n \rhd_\beta T'$.

**Fall 3 ($\rhd_{1\beta}$):** Da $P_n \rhd_\beta T'$ und $P_n \rhd_{1\beta} P_{n+1}$, existiert nach *1.* ein $T$ mit $T' \rhd_\beta T$, und $P_{n+1} \rhd_\beta T$, also $P_1 \rhd_\beta T$ und $P_n \rhd_\beta T$.

□

Für den Beweis von *1.* werden noch einige Definitionen und Lemmata benötigt.

**Definition 1.16**

*Seien $R$ und $S$ $\beta$-Redexe in $P$, wobei $R \rhd_{1\beta} R'$.*

*Das* Residuum $res(S,R)$ *von $S$ bezüglich $R$ ist wie folgt definiert:*

1. *$R$ und $S$ überlappen sich nicht. Dann sei $res(S,R) = S$.*

2. *$R \equiv S$: Dann sei $res(S,R)$ nicht definiert.*

3. *$R \not\equiv S$ und $S \equiv S[R]$, d.h. $R$ ist echter Teilterm von $S$. Dann sei $res(S,R) = S[R']$.*

*Intuitiv: $res(S,R)$ ist die Gestalt von $S$ nach Kontraktion von $R$.*

**Definition 1.17**
*Sei $\mathcal{R}$=$\{R_1, \ldots, R_n\}$ eine Menge von Redexen in $P$. $R_i$ heißt* minimal*, falls kein $R_j$ echter Teilterm von $R_i$ ist.*
*$P \rhd_{mcd} Q$, falls $Q$ aus $P$ durch folgendes (nichtdeterministisches) Verfahren hervorgeht:*

1. *Es wird ein minimales Element $R_i$ in $\mathcal{R}$ gewählt.*
2. *$R_i$ wird in $P$ $\beta$-kontrahiert.*
3. *$R_j$ wird in $P$ durch $res(R_j, R_i)$ ersetzt für alle $j \neq i$. $P'$ sei der resultierende Term.*
4. *$\mathcal{R}'$ umfasse alle $res(R_j, R_i)$ für $j \neq i$, die Redexe sind ($\mathcal{R}'$ hat damit maximal $n - 1$ Elemente).*
5. *Falls $\mathcal{R}'$ nicht leer, werden die Schritte* 1.–4. *in Bezug auf $\mathcal{R}'$ und $P'$ wiederholt. Ansonsten ergibt sich $Q$ aus $P'$ durch beliebig viele $\alpha$-Reduktionen.*

*Grenzfall: $P \rhd_{mcd} P$*

**Bemerkungen**

1. "mcd" steht für "minimal complete development".
2. $\rhd_{mcd}$ ist nicht transitiv: Es ist $(\lambda x.xy)(\lambda x.x) \rhd_{mcd} (\lambda x.x)y$ und $(\lambda x.x)y \rhd_{mcd} y$, aber nicht $(\lambda x.xy)(\lambda x.x) \rhd_{mcd} y$ (Beachte, daß $res(S, R)$ für $S \equiv R$ nicht definiert ist).
3. $\rhd_{mcd}$ ist relativ zu einer gewählten Menge $\mathcal{R}$ von Redexen, wobei $\mathcal{R}$ für jede $\rhd_{mcd}$-Beziehung verschieden sein kann.

**Beispiel 1.18**
$(\lambda x.xy)(\lambda x.x) \rhd_{mcd} (\lambda x.x)y \; \substack{\not\rhd_{mcd} \\ \rhd_\beta} \; y$

**Lemma 1.19**
*Wenn $P \rhd_{mcd} Q$ und $P \equiv_\alpha P^*$, dann $P^* \rhd_{mcd} Q$.*

**Lemma 1.20**
*Wenn $M \rhd_{mcd} M'$ und $N \rhd_{mcd} N'$, dann $M[{}^{N}\!/x] \rhd_{mcd} M'[{}^{N'}\!/x]$.*

**Lemma 1.21**
*Wenn $P \rhd_{mcd} Q$ und $P \rhd_{mcd} R$, so existiert ein Term $T$, so daß $Q \rhd_{mcd} T$ und $R \rhd_{mcd} T$.*

Beweis
Wegen *Lemma 1.19* können wir annehmen, daß in den gegebenen mcd-Reduktionen keine $\alpha$-Schritte vorkommen. Der Beweis erfolgt durch Induktion über der Struktur von $P$:

1. $P \equiv a$: $P \equiv Q \equiv R \equiv T$

2. $P \equiv \lambda x.P_1$, d.h. $Q \equiv \lambda x.Q_1$ und $R \equiv \lambda x.R_1$ (keine $\alpha$-Schritte!)
   $P_1 \rhd_{mcd} Q_1$ und $P_1 \rhd_{mcd} R_1$. Nach Induktionsvoraussetzung existiert $T_1$, so daß $Q_1 \rhd_{mcd} T_1$ und $R_1 \rhd_{mcd} T_1$. Setze $T \equiv_{def} \lambda x.T_1$.

3. $P \equiv P_1 P_2$ und alle Redexe von $\mathcal{R}$ sind in $P_1, P_2$, dh. $P$ selbst, wird nicht reduziert.
   Dann gilt nach Induktionsvoraussetzung mit

$$\begin{tikzcd} & P_1 \arrow[dl, "\rhd_{mcd}"'] \arrow[dr, "\rhd_{mcd}"] & \\ Q_1 \arrow[dr, "\rhd_{mcd}"'] & & R_1 \arrow[dl, "\rhd_{mcd}"] \\ & T_1 & \end{tikzcd} \quad \text{und} \quad \begin{tikzcd} & P_2 \arrow[dl, "\rhd_{mcd}"'] \arrow[dr, "\rhd_{mcd}"] & \\ Q_2 \arrow[dr, "\rhd_{mcd}"'] & & R_2 \arrow[dl, "\rhd_{mcd}"] \\ & T_2 & \end{tikzcd} \quad \text{auch} \quad \begin{tikzcd} & P_1P_2 \arrow[dl, "\rhd_{mcd}"'] \arrow[dr, "\rhd_{mcd}"] & \\ Q_1Q_2 \arrow[dr, "\rhd_{mcd}"'] & & R_1R_2 \arrow[dl, "\rhd_{mcd}"] \\ & T_1T_2 & \end{tikzcd}$$

   Setze $T \equiv_{def} T_1 T_2$.

4. $P \equiv (\lambda x.M)N$ und das Residuum von $P$ wird bei $P \rhd_{mcd} Q$, nicht jedoch bei $P \rhd_{mcd} R$ kontrahiert, d.h.
   $P \equiv (\lambda x.M)N \rhd_{mcd} (\lambda x.M')N' \rhd_{1\beta} M'[N'/x] \equiv Q \qquad (M \rhd_{mcd} M'$ und $N \rhd_{mcd} N')$
   $P \equiv (\lambda x.M)N \rhd_{mcd} (\lambda x.M'')N'' \equiv R \qquad (M \rhd_{mcd} M''$ und $N \rhd_{mcd} N'')$
   Nach Induktionsvoraussetzung existieren $M^+$ und $N^+$ derart, daß $M', M'' \rhd_{mcd} M^+$ und $N', N'' \rhd_{mcd} N^+$.
   Setze $T \equiv_{def} M^+[N^+/x]$, dann ist $Q \equiv M'[N'/x] \rhd_{mcd} M^+[N^+/x]$ nach Lemma 1.20.
   Ferner $(\lambda x.M'')N'' \rhd_{mcd} (\lambda x.M^*)N^* \rhd_{1\beta} M^*[N^*/x] \equiv_\alpha M^+[N^+/x]$ , wobei wir annehmen, daß ohne $\alpha$-Schritte $M'' \rhd_{mcd} M^*$, $N'' \rhd_{mcd} N^*$, und $M^* \equiv_\alpha M^+$, $M^* \equiv_\alpha M^+$.

5. $P \equiv (\lambda x.M)N$ und beide mcd-Reduktionen kontrahieren das Residuum von $P$, d.h.
   $P \equiv (\lambda x.M)N \rhd_{mcd} (\lambda x.M')N' \rhd_{1\beta} M'[N'/x] \equiv Q$
   $P \equiv (\lambda x.M)N \rhd_{mcd} (\lambda x.M'')N'' \rhd_{1\beta} M''[N''/x] \equiv R$
   Wir argumentieren wie in Fall *4.* und setzen wieder $T \equiv_{def} M^+[N^+/x]$. Mit *Lemma 1.20* ergibt sich dann die Behauptung.

□

Beweis von Theorem 1.14,*1.*
Zunächst ergibt sich aus dem vorhergehenden Lemma durch Induktion:
Wenn $P \rhd_{mcd} M$ und $P \rhd_\beta N$, dann existiert ein Term $T$, so daß $M \rhd_\beta T$ und $N \rhd_{mcd} T$.

$$
\begin{tikzcd}
 & P \arrow[dl, "\rhd_{mcd}"'] \arrow[dr, "\substack{\rhd_{1\beta} \\ \equiv_\alpha}"] & & & \\
M \arrow[dr, "\rhd_{mcd}"'] & & N_1 \arrow[dl, "\rhd_{mcd}"'] \arrow[dr, "\substack{\rhd_{1\beta} \\ \equiv_\alpha}"] & & \\
 & T_1 \arrow[dr, "\rhd_{mcd}"'] & & N_2 \arrow[dl, "\rhd_{mcd}"'] \arrow[dr, "\dots"] & \\
 & & T_2 \arrow[dr, "\dots"'] & & N_n \equiv N \arrow[dl, "\rhd_{mcd}"] \\
 & & & T_n \equiv T &
\end{tikzcd}
$$

Beachte, daß mit $\rhd_{1\beta}$ auch $\rhd_{mcd}$ gilt, und mit $\rhd_{mcd}$ auch $\rhd_\beta$, und daß $\rhd_\beta$ transitiv ist.
Daraus ergibt sich:
Wenn $P \rhd_{1\beta} M$ und $P \rhd_\beta N$, dann existiert ein Term $T$, so daß $M \rhd_\beta T$ und $N \rhd_\beta T$.
Hieraus folgt dann sofort durch Induktion die Behauptung des Theorems.

$$
\begin{tikzcd}
 & & & P \arrow[dl, "\substack{\rhd_{1\beta} \\ \equiv_\alpha}"'] \arrow[dr, "\rhd_\beta"] & \\
 & & P_1 \arrow[dl, "\substack{\rhd_{1\beta} \\ \equiv_\alpha}"'] \arrow[dr, "\rhd_\beta"'] & & N \arrow[dl, "\rhd_\beta"] \\
 & P_2 \arrow[dl, "\dots"'] \arrow[dr, "\rhd_\beta"'] & & T_1 \arrow[dl, "\rhd_\beta"] & \\
M \arrow[dr, "\rhd_\beta"'] & & T_2 \arrow[dl, "\dots"] & & \\
 & T & & &
\end{tikzcd}
$$

□

**Korollar 1.22**

1. *Falls $M$ und $N$ $\beta$-Normalformen von $P$ sind, dann $M \equiv_\alpha N$.*

2. *Wenn $M =_\beta N$ und $N$ $\beta$-Normalform ist, dann $M \rhd_\beta N$.*

3. *Falls $M =_\beta N$, dann gilt: $M$ oder $N$ hat keine, oder beide haben dieselbe $\beta$-Normalform (bis auf Kongruenz).*

4. *$\beta$-gleiche Terme in $\beta$-Normalform sind kongruent.*

**Definition 1.23**
*Eine* L-Reduktionsfolge *ist eine $\beta$-Reduktionsfolge, bei der immer das linkeste Redex im jeweiligen Term kontrahiert wird. Ein Redex $(\lambda x_1.M_1)N_1$ ist dabei linker als $(\lambda x_2.M_2)N_2$ (im betrachteten Term), falls sich $\lambda x_1$ links von $\lambda x_2$ befindet.*
*Eine* QL-Reduktionsfolge *ist eine $\beta$-Reduktionsfolge $(M_1, M_2, M_3, \ldots)$, so daß es zu jedem $M_i$, das nicht letztes Glied der Folge ist, ein $M_j$ und ein $M_{j+1}$ mit $j \geq i$ gibt, so daß beim Übergang von $M_j$ zu $M_{j+1}$ das linkeste Redex in $M_j$ kontrahiert wird.*

**Bemerkung**
L steht für "leftmost", QL für "quasi-leftmost". Eine QL-Reduktionsfolge ist also eine $\beta$-Reduktionsfolge, bei der "immer wieder" das linkeste Redex kontrahiert wird.

**Theorem 1.24**
*Falls ein λ-Term $M$ eine $\beta$-Normalform hat, dann terminiert jede mit $M$ beginnende L-Reduktionsfolge (und damit auch jede QL-Reduktionsfolge).*

Beweis
Vgl. Barendregt (1980), Abschnitt 13.2. □

**Theorem 1.25**
*Es gibt Kombinatoren mit folgender Eigenschaft:*

1. $Yx =_\beta x(Yx)$
2. $Yx \rhd_\beta x(Yx)$

*d.h. $Yx$ ist* Fixpunkt *von $x$ ($Y$ heißt* Fixpunktkombinator*).*

Beweis

- $\Upsilon \equiv_{def} \lambda x.(\lambda y.x(yy))\,\underbrace{(\lambda y.x(yy))}_{M}$ (Curry)
- $\Theta \equiv_{def} (\lambda zx.x(zzx))\,\underbrace{(\lambda zx.x(zzx))}_{N}$ (Turing)

1. $\Upsilon x \rhd_\beta MM \rhd_\beta x(MM) \lhd_\beta x(\Upsilon x)$
2. $\Theta x \rhd_\beta (\lambda x.x(NNx))x \rhd_\beta x(NNx) \equiv x(\Theta x)$

$\Theta$ erfüllt natürlich auch 1., $\Upsilon$ erfüllt aber nicht 2. (warum nicht?) □

**Korollar 1.26**
*Für jedes $N$ und $n \geq 0$ gibt es ein $M$, so daß $My_1 \ldots y_n =_\beta N[{}^{M}\!/x]$*

D.h. jede Gleichung, die $x$ durch einen Term $N$ definiert, in dem $x$ vorkommen kann (und die insofern "rekursiv" ist), hat eine Lösung $M$.

Beweis
Setze $M \equiv_{def} Y(\lambda x y_1 \dots y_n.N)$ für einen Fixpunktkombinator $Y$. □

**Bemerkung**
Wählen wir $\Theta$ für $Y$, d.h. $M \equiv_{def} \Theta(\lambda x y_1 \dots y_n.N)$, dann gilt sogar $M y_1 \dots y_n \rhd_\beta N[{}^M\!/\!_x]$.

**Proposition 1.27**
*$M$ ist ein Fixpunktkombinator (d.h. $Mx =_\beta x(Mx)$) genau dann, wenn $M$ Fixpunkt von $SI$ ist, d.h. $M =_\beta SIM$.*

Beweis (Barendregt (1980), 6.5.3)
$SI =_\beta \lambda yz.z(yz)$
Sei $M$ Fixpunkt von $SI$, d.h. $M =_\beta SIM$. Dann $MF =_\beta SIMF =_\beta F(MF)$, d.h. $M$ ist Fixpunktkombinator.
Sei $Mx =_\beta x(Mx)$. Dann ist $Mx$ nicht in Normalform, da sonst $Mx$ und $x(Mx)$ $\alpha$-kongruent wären. Damit gilt $Mx \rhd_\beta xP$ und $x(Mx) \rhd_\beta xP$ für ein $P$. Ferner gilt $M \rhd_\beta \lambda z.N$, da $M$ als Kombinator nicht mit einer Variable beginnen kann. Damit gilt
$\lambda x.Mx =_\beta \lambda x.(\lambda z.N)x =_\beta \lambda x.N[{}^x\!/\!_z] =_\beta M$ (d.h. $\eta$-Konversion (s.u.) ist für $M$ beweisbar).
Somit $M =_\beta \lambda x.Mx =_\beta \lambda x.x(Mx) =_\beta SIM$ □

**Definition 1.28**
*$P[\lambda x.Mx] \rhd_{1\eta} P[M]$, falls $x \notin FV(M)$* "$\eta$-Kontraktion"

$$P \rhd_{\beta\eta} Q, \text{ falls } P \equiv P_1 \substack{\equiv_{1\alpha} \\ \rhd_{1\beta} \\ \rhd_{1\eta}} P_2 \substack{\equiv_{1\alpha} \\ \rhd_{1\beta} \\ \rhd_{1\eta}} \cdots \substack{\equiv_{1\alpha} \\ \rhd_{1\beta} \\ \rhd_{1\eta}} P_n \equiv Q \qquad \text{"}\beta\eta\text{-Reduktion"}$$

$$P =_{\beta\eta} Q \text{ , falls } P \equiv P_1 \substack{\equiv_{1\alpha} \\ \rhd_{1\beta} \\ \lhd_{1\beta} \\ \rhd_{1\eta} \\ \lhd_{1\eta}} P_2 \substack{\equiv_{1\alpha} \\ \rhd_{1\beta} \\ \lhd_{1\beta} \\ \rhd_{1\eta} \\ \lhd_{1\eta}} \cdots \substack{\equiv_{1\alpha} \\ \rhd_{1\beta} \\ \lhd_{1\beta} \\ \rhd_{1\eta} \\ \lhd_{1\eta}} P_n \equiv Q \qquad \text{"}\beta\eta\text{-Gleichheit"}$$

**Bemerkung**
$\beta\eta$-Gleichheit besagt intuitiv, daß es für die Bedeutung eines Terms nur auf sein Verhalten bei Anwendung auf einen anderen Term ankommt (Extensionalität, vgl. Lemma 1.33).

**Lemma 1.29**
*Lemma 1.11 gilt auch für $\rhd_{\beta\eta}$.*

**Theorem 1.30**

*$\beta\eta$-Reduktion genügt Church-Rosser.*

## 1.2 Die formalen Theorien $\lambda\beta$ und $\lambda\beta\eta$

**Definition 1.31**

*Formeln der Systeme $\lambda\beta$ und $\lambda\beta\eta$ sind alle Gleichungen der Form $M = N$ für $\lambda$-Terme $M, N$.*
*Die Axiome sind:*

$(\rho)$ $M = M$

$(\alpha)$ $\lambda x.M = \lambda y.M[y/x]$, *falls* $y \notin FV(M)$

$(\beta)$ $(\lambda x.M)N = M[N/x]$

$(\eta)$ $\lambda x.Mx = M$, *falls* $x \notin FV(M)$ (nur $\lambda\beta\eta$ !)

*Die Regeln sind:*

$(\sigma)$ $\dfrac{M = N}{N = M}$

$(\tau)$ $\dfrac{M = N \quad N = P}{M = P}$

$(\mu)$ $\dfrac{M = M'}{NM = NM'}$

$(\nu)$ $\dfrac{N = N'}{NM = N'M}$

$(\xi)$ $\dfrac{M = M'}{\lambda x.M = \lambda x.M'}$ *(schwache Extensionalität)*

$\lambda\beta \vdash M = N$ *heißt, daß* $M = N$ *in* $\lambda\beta$ *ableitbar ist.*
$\lambda\beta\eta \vdash M = N$ *heißt, daß* $M = N$ *in* $\lambda\beta\eta$ *ableitbar ist.*
$\lambda\beta_{\triangleright}$ *und* $\lambda\beta\eta_{\triangleright}$ *sind Systeme ohne Regel* $(\sigma)$ *(Symmetrie).*
$\lambda\beta_{\triangleright} \vdash M = N$ *heißt, daß* $M = N$ *in* $\lambda\beta_{\triangleright}$ *ableitbar ist.*
$\lambda\beta\eta_{\triangleright} \vdash M = N$ *heißt, daß* $M = N$ *in* $\lambda\beta\eta_{\triangleright}$ *ableitbar ist.*

**Lemma 1.32**

1. $M \triangleright_{\beta} N$ *genau dann, wenn* $\lambda\beta_{\triangleright} \vdash M = N$
2. $M \triangleright_{\beta\eta} N$ *genau dann, wenn* $\lambda\beta\eta_{\triangleright} \vdash M = N$

3. *$M =_\beta N$ genau dann, wenn $\lambda\beta \vdash M = N$*
4. *$M =_{\beta\eta} N$ genau dann, wenn $\lambda\beta\eta \vdash M = N$*

Beweis

Übung. □

**Lemma 1.33**

*Ersetzt man in der Definition von $\lambda\beta\eta$ das Axiom $(\eta)$ durch*

- $(ext)\ \dfrac{MP = NP \text{ für alle } P}{M = N}$ *oder*
- $(\zeta)\ \dfrac{Mx = Nx}{M = N}$ *falls $x \notin FV(NM)$*

*so sind in $\lambda\beta\eta$ dieselben Gleichungen wie vorher ableitbar.*

Beweis

- "$(\eta) \Longrightarrow (\zeta)$": $(\tau)\dfrac{(\sigma)\dfrac{(\eta)\dfrac{}{\lambda x.Mx = M}}{M = \lambda x.Mx} \quad (\xi)\dfrac{Mx = Nx}{\lambda x.Mx = \lambda x.Nx}}{(\eta) + (\tau)\dfrac{M = \lambda x.Nx}{M = N}}$
- "$(\zeta) \Longrightarrow (ext)$": Wähle als $P$ ein $x \notin FV(MN)$
- "$(ext) \Longrightarrow (\eta)$": $(ext)\ \dfrac{(\beta)\dfrac{}{(\lambda x.Mx)P = MP} \text{ für alle } P}{\lambda x.Mx = M}$ (falls $x \in FV(M)$)

□

## 1.3 λ-Definierbarkeit rekursiver Funktionen

**Definition 1.34**

*Sei* $M^0 N \equiv N$ *und* $M^{n+1} N \equiv M(M^n N)$.

*Dann sind die* Church-Ziffern *wie folgt definiert:* $\underline{n} \equiv_{def} \lambda xy.x^n y$

**Bemerkungen**

1. Vgl. Wittgenstein, Tractatus 6.021: "Die Zahl ist der Exponent einer Operation".

2. Falls $\underline{m} =_\beta \underline{n}$, dann $m = n$, da Church-Ziffern in $\beta$-Normalform sind.

**Definition 1.35**

*Der* $\lambda$*-Term* $P$ *definiert die* $k$*-stellige zahlentheoretische Funktion* $f$*, falls für alle* $m_1, \ldots, m_k$ *gilt, daß* $P\,\underline{m_1} \ldots \underline{m_k} \simeq_\beta \underline{f(m_1, \ldots, m_k)}$*, d.h.* $P\,\underline{\vec{m}} \simeq_\beta \underline{f(\vec{m})}$.

*Dabei bedeutet* $P\,\underline{\vec{m}} \simeq_\beta \underline{n}$*, daß* $\begin{cases} P\,\underline{\vec{m}} =_\beta \underline{n} \iff f(\vec{m}) = n & \textit{falls } f(\vec{m}) \textit{ definiert} \\ P\,\underline{\vec{m}} \textit{ hat keine } \beta\textit{-Normalform} & \textit{falls } f(\vec{m}) \textit{ nicht definiert} \end{cases}$

**Lemma 1.36**

*Es gibt Kombinatoren mit folgenden Eigenschaften:*

1. $N\,\underline{k} =_\beta \underline{k+1}$

2. $V\,\underline{k+1} =_\beta \underline{k}$

3. $DPQ\,\underline{0} =_\beta P$
   $DPQ\,\underline{k+1} =_\beta Q$

4. $RPQ\,\underline{0} =_\beta P$
   $RPQ\,\underline{k+1} =_\beta Q\,\underline{k}(RPQ\,\underline{k})$

Beweis

1. $N \equiv_{def} \lambda uxy.x(uxy)$

3. $D \equiv_{def} \lambda xyz.z(Ky)x$

2. $V \equiv_{def} \lambda x.x(\lambda z.D(N(z\,\underline{0}))(z\,\underline{0}))(D\,\underline{0}\,\underline{0})\,\underline{1}$

   Beweis

   Wir zeigen durch Induktion über $k$: $(\underbrace{\lambda z.D(N(z\,\underline{0}))(z\,\underline{0})}_{P})^{k+1}(D\,\underline{0}\,\underline{0}) =_\beta D\,\underline{k+1}\,\underline{k}$

   Induktionsanfang:

$$P^1 D\,\underline{0}\,\underline{0} =_\beta D(N(D\,\underline{0}\,\underline{0}\,\underline{0}))(D\,\underline{0}\,\underline{0}\,\underline{0}) =_\beta D(N\,\underline{0})\,\underline{0} =_\beta D\,\underline{1}\,\underline{0}$$

Induktionsschritt: Sei $P^{k+1}(D\,\underline{0}\,\underline{0}) =_\beta D\,\underline{k+1}\,\underline{k}$. Dann ist

$$\begin{array}{rcll} P^{k+2}(D\,\underline{0}\,\underline{0}) & =_\beta & P(P^{k+1}(D\,\underline{0}\,\underline{0})) & \\ & =_\beta & P(D\,\underline{k+1}\,\underline{k}) & \text{(Induktionsvoraussetzung)} \\ & =_\beta & D(N(D\,\underline{k+1}\,\underline{k}\,\underline{0}))(D\,\underline{k+1}\,\underline{k}\,\underline{0}) & \\ & =_\beta & D(N\,\underline{k+1})\,\underline{k+1} & \\ & =_\beta & D\,\underline{k+2}\,\underline{k+1} & \end{array}$$

Damit ist

$$\begin{array}{rcl} V\,\underline{k+1} & =_\beta & \underline{k+1}P(D\,\underline{0}\,\underline{0})\,\underline{1} \\ & =_\beta & P^{k+1}(D\,\underline{0}\,\underline{0})\,\underline{1} \\ & =_\beta & D\,\underline{k+1}\,\underline{k}\,\underline{1} \\ & =_\beta & \underline{k} \end{array}$$

□

4. $R \equiv_{def} \Theta(\lambda uxyz.Dx(y(Vz)(uxy(Vz)))z)$
   $R$ ist nach Korollar 1.26 Lösung von $Rxyz =_\beta Dx(y(Vz)(Rxy(Vz)))z$

□

**Theorem 1.37**

*Jede primitiv-rekursive Funktion ist λ-definierbar.*

Beweis

- $0 : \mathcal{N}^0 \longrightarrow \mathcal{N}$ ist λ-definiert durch den Term $\underline{0}$
- $s : \mathcal{N} \longrightarrow \mathcal{N}$ ist λ-definiert durch den Term $N$
- $\Pi_n^i : \mathcal{N}^n \longrightarrow \mathcal{N}$ ist λ-definiert durch den Term $\lambda x_1 \dots x_n.x_i$
- Falls $h : \mathcal{N}^k \longrightarrow \mathcal{N}$ und $g_i : \mathcal{N}^n \longrightarrow \mathcal{N}$ durch $P$ und $Q_i$ λ-definiert sind $(1 \leq i \leq k)$, und weiterhin $f(\vec{m}) = h(g_1(\vec{m}), \dots, g_k(\vec{m}))$ (wobei $\vec{m} = (m_1, \dots, m_n)$), dann wird die Funktion $f : \mathcal{N}^n \longrightarrow \mathcal{N}$ durch den Term $\lambda\vec{x}.P(Q_1\vec{x})\dots(Q_k\vec{x})$ λ-definiert (wobei $\vec{x} = (x_1 \dots x_n)$)
- Falls $g : \mathcal{N}^k \longrightarrow \mathcal{N}$ und $h : \mathcal{N}^{k+2} \longrightarrow \mathcal{N}$ durch die λ-Terme $P$ und $Q$ λ-definiert sind, und $f : \mathcal{N}^{k+1} \longrightarrow \mathcal{N}$ gegeben ist durch:

  1. $f(0, \vec{m}) = g(\vec{m})$
  2. $f(n+1, \vec{m}) = h(n, f(n, \vec{m}), \vec{m})$

  dann wird $f$ λ-definiert durch den Term $\lambda u\vec{x}.R(P\vec{x})(\lambda uv.Quv\vec{x})u$

Beweis durch Induktion über $n$

Induktionsanfang:

$$\begin{aligned}(\lambda u\vec{x}.R(P\vec{x})(\lambda uv.Quv\vec{x})u)\,\underline{0}\,\underline{\vec{m}} &=_\beta R(P\,\underline{\vec{m}})(\lambda uv.Quv\,\underline{\vec{m}})\,\underline{0}\\ &=_\beta P\,\underline{\vec{m}}\\ &=_\beta \underline{g(\vec{m})}\ \ \text{(nach Voraussetzung über } g)\end{aligned}$$

Induktionsschritt:

$$\begin{aligned}(\lambda u\vec{x}.R(P\vec{x})(\lambda uv.Quv\vec{x})u)\,\underline{n+1}\,\underline{\vec{m}} &=_\beta R(P\,\underline{\vec{m}})(\lambda uv.Quv\,\underline{\vec{m}})\,\underline{n+1}\\ &=_\beta (\lambda uv.Quv\,\underline{\vec{m}})\,\underline{n}(R(P\,\underline{\vec{m}})(\lambda uv.Quv\,\underline{\vec{m}})\,\underline{n})\\ &=_\beta Q\,\underline{n}(R(P\,\underline{\vec{m}})(\lambda uv.Quv\,\underline{\vec{m}})\,\underline{n})\,\underline{\vec{m}}\\ &=_\beta Q\,\underline{n}((\lambda u\vec{x}.R(P\vec{x})(\lambda uv.Quv\vec{x})u)\,\underline{n}\,\underline{\vec{m}})\,\underline{\vec{m}}\\ &=_\beta Q\,\underline{n}\,\underline{f(n,\vec{m})}\,\underline{\vec{m}}\ \ \text{(Induktionsvoraussetzung)}\\ &=_\beta \underline{h(n,f(n,\vec{m}),\vec{m})}\ \ \text{(nach Voraussetzung über } h)\end{aligned}$$

□

**Theorem 1.38**

*Jede partiell-rekursive Funktion ist λ-definierbar.*

Beweis

Jedes partiell-rekursive $f$ läßt sich nach Kleene darstellen als $f(\vec{m}) = h(\mu k.g(\vec{m},k) = 0)$, wobei $g, h$ primitiv-rekursive Funktionen sind (Informatik III). Seien $g$ und $h$ durch $P$ und $Q$ λ-definiert. Betrachte die Gleichung

$$(\star)\quad U\vec{x}y =_\beta Dy(U\vec{x}(Ny))(P\vec{x}y)$$

Nach Korollar 1.26 ist $\Theta\,\underbrace{(\lambda u\vec{x}y.Dy(u\vec{x}(Ny))(P\vec{x}y))}_{Z}$ eine Lösung der Gleichung.

Behauptung: $f$ wird λ-definiert durch den Term $\lambda\vec{x}.Q(\Theta Z\vec{x}\,\underline{0})$.

Dazu genügt es zu zeigen: $\Theta Z\,\underline{\vec{m}}\,\underline{0} =_\beta \underline{k_1}$, falls $k_1$ kleinstes $k$ mit $g(\vec{m},k) = 0$.

Wir werden zeigen:

$$(\star\star)\quad \text{Falls } g(\vec{m},k) \neq 0 \text{ für alle } k < k_1, \text{ dann } \Theta Z\vec{m}\,\underline{0} =_\beta D\,\underline{k_1}(\Theta Z\,\underline{\vec{m}}\,\underline{k_1+1})(P\,\underline{\vec{m}}\,\underline{k_1})$$

Hieraus ergibt sich: Wenn $k_1$ kleinstes $k$ mit $g(\vec{m},k) = 0$, dann $\Theta Z\vec{m}\,\underline{0} =_\beta \underline{k_1}$, da $P\,\underline{\vec{m}}\,\underline{k_1} =_\beta \underline{0}$.

Beweis von $(\star\star)$ durch Induktion über $k_1$:

- $k_1 = 0$: $\Theta Z\,\underline{\vec{m}}\,\underline{0} \ =_\beta\ D\,\underline{0}(\Theta Z\,\underline{\vec{m}}\,\underline{1})(P\,\underline{\vec{m}}\,\underline{0})$ mit $(\star)$

- $k_1 > 0$: $$\begin{aligned} \Theta Z\,\underline{\vec{m}}\,\underline{0} &=_\beta D\,\underline{k_1-1}(\Theta Z\,\underline{\vec{m}}\,\underline{k_1})\underbrace{(P\,\underline{\vec{m}}\,\underline{k_1-1})} \qquad \text{(Induktionsvoraussetzung)} \\ &\qquad\qquad =_\beta \underline{l+1} \text{ für ein } l, \\ &\qquad\qquad \text{da } g(\vec{m},k_1-1)\neq 0 \\ &=_\beta \Theta Z\,\underline{\vec{m}}\,\underline{k_1} \\ &=_\beta D\,\underline{k_1}(\Theta Z\,\underline{\vec{m}}\,\underline{k_1+1})(P\,\underline{\vec{m}}\,\underline{k_1}) \quad \text{mit } (\star) \end{aligned}$$

Es bleibt zu zeigen: Wenn $f(\vec{m})$ undefiniert, d.h. wenn $g(\vec{m},k)\neq 0$ für alle $k$ bei gegebenem $\vec{m}$, dann hat $\Theta Z\,\underline{\vec{m}}\,\underline{0}$ keine $\beta$-Normalform. Es gilt:

$$\begin{array}{lllll} \Theta Z\,\underline{\vec{m}}\,\underline{0} & \rhd_\beta & D\,\underline{0}(\Theta Z\,\underline{\vec{m}}\,\underline{1})(P\,\underline{\vec{m}}\,\underline{0}) & \rhd_\beta & \Theta Z\,\underline{\vec{m}}\,\underline{1} \\ & \rhd_\beta & D\,\underline{1}(\Theta Z\,\underline{\vec{m}}\,\underline{2})(P\,\underline{\vec{m}}\,\underline{1}) & \rhd_\beta & \Theta Z\,\underline{\vec{m}}\,\underline{2} \\ & \rhd_\beta & D\,\underline{2}(\Theta Z\,\underline{\vec{m}}\,\underline{3})(P\,\underline{\vec{m}}\,\underline{2}) & \rhd_\beta & \Theta Z\,\underline{\vec{m}}\,\underline{3} \\ & \rhd_\beta & \cdots & & \end{array}$$

Diese Reduktionsfolge ist QL (quasi-leftmost), d.h. es kommt immer wieder vor, daß ein linkester Term kontrahiert wird, nämlich ein Term der Form $DMN\,\underline{l+1}$. Wir haben also eine nichtterminierende QL-Reduktionsfolge. Also hat $\Theta Z\,\underline{\vec{m}}\,\underline{0}$ keine $\beta$-Normalform (Theorem 1.24). Man beachte, daß wir $\Theta$ als Fixpunktkombinator gewählt haben. Damit haben wir statt $=_\beta$ immer $\rhd_\beta$ in der Anwendung von $(\star)$ (vgl. die Bemerkung zu Korollar 1.26). □

**Theorem 1.39**

*Jede $\lambda$-definierbare Funktion ist partiell-rekursiv.*

Beweisskizze

Sei $f$ $n$-stellig und durch $P$ $\lambda$-definiert. Dann gilt:

| | | |
|---|---|---|
| $f(k_1,\ldots,k_n)$ | $=$ | dasjenige $k$, für das die Gleichung $P\,\underline{k_1}\ldots\underline{k_n} = \underline{k}$ die Endformel der kürzesten Ableitung in $\lambda\beta$ ist, die mit einer Formel der Gestalt $P\,\underline{k_1}\ldots\underline{k_n} = \underline{m}$ endet, falls es eine solche Ableitung in $\lambda\beta$ gibt. |
| $f(k_1,\ldots,k_n)$ | | ist sonst undefiniert. |

Nach geeigneter Gödelisierung erweist sich $f$ als partiell-rekursive Funktion. □

### 1.4 Entscheidbarkeit

**Theorem 1.40 (Church 1936)**
*Die Menge $NF_\beta =_{def} \{M : M \text{ hat } \beta\text{-Normalform}\}$ ist nicht entscheidbar.*

Beweisskizze
Wir können die einstelligen partiell-rekursiven Funktionen so abzählen: $f_1, f_2, \ldots$, daß die Funktion $u$ mit $u(m,n) \simeq_{def} f_m(n)$ partiell rekursiv ist. Nun werde $u$ durch $P$ $\lambda$-definiert. Dann gilt: $P\,\underline{m}\,\underline{n}$ hat $\beta$-Normalform g.d.w. $u(m,n)$ ist definiert.
Wäre $NF_\beta$ entscheidbar, wäre $g$ mit

$$g(n) \quad =_{def} \quad \begin{cases} u(n,n)+1 & \text{falls } u(n,n) \text{ definiert} \\ 1 & \text{sonst} \end{cases}$$

eine total-rekursive Funktion. Damit wäre $g = f_k$ für ein $k$, also

$$u(k,k) = f_k(k) = g(k) = u(k,k)+1$$

da $f_k$ total. □

**Theorem 1.41 (Church 1936)**
*$=_\beta$ ist unentscheidbar.*

Beweisskizze
Die zu einem Term $\beta$-konvertiblen Terme lassen sich rekursiv aufzählen. Sei

$$f(m,k) \quad =_{def} \quad \text{Gödelnummer des } k\text{-ten Terms, der zum Term mit Gödelnummer } m\ \beta\text{-konvertibel ist}$$

$$h(m) \quad =_{def} \quad \begin{cases} 0 & \text{falls } m \text{ eine Gödelnummer eines Terms in } \beta\text{-Normalform ist} \\ 1 & \text{sonst} \end{cases}$$

$f$ und $h$ sind primitiv rekursiv. Sie seien durch $F$ und $H$ $\lambda$-definiert.
Betrachte die Gleichung (in $G$): $Gxy =_\beta D\,\underline{1}(Gx(Ny))(H(Fxy))$. Eine Lösung dieser Gleichung ist nach Korollar 1.26: $\Upsilon\lambda gxy.\underbrace{D\,\underline{1}(Gx(Ny))(H(Fxy))}_{V}$. Es gilt dann:
$(\Upsilon\lambda gxy.V)\,\underline{m}\,\underline{0} =_\beta \underline{1}$, falls $m$ Gödelnummer eines Terms ist, der zu einem Term in $\beta$-Normalform konvertibel ist. $(\Upsilon\lambda gxy.V)\,\underline{m}\,\underline{0}$ hat sonst keine $\beta$-Normalform.
Falls nun $=_\beta$ entscheidbar, dann ist $(\Upsilon\lambda gxy.V)\,\underline{\ulcorner M \urcorner}\,\underline{0} =_\beta \underline{1}$ entscheidbar (wobei $\ulcorner M \urcorner$ Gödelnummer von $M$). Also ist $NF_\beta$ entsheidbar. □

**Theorem 1.42 (Church 1936)**
*Die Prädikatenlogik 1. Stufe $\mathcal{PL}$ ist unentscheidbar.*

BEWEISSKIZZE
Da $=_\beta$ unentscheidbar, ist $\lambda\beta$ ein unentscheidbarer Kalkül. Nun gilt:

$$\lambda\beta \vdash M = N \text{ g.d.w. } \mathcal{PL} \vdash (F_1 \wedge \ldots \wedge F_8) \rightarrow E(\overline{\overline{m}}, \overline{\overline{n}})$$

Hierbei sei $E$ ein ausgezeichnetes zweistelliges Prädikat und

$$\left.\begin{array}{lll} \overline{\overline{0}} & \equiv_{def} & z \\ \overline{\overline{1}} & \equiv_{def} & f(z) \\ \overline{\overline{2}} & \equiv_{def} & f(f(z)) \\ & \vdots & \end{array}\right\} \text{für ausgezeichnete } z \text{ und } f$$

$F_i$ sei die Prädikatenlogische Übersetzung der $i$-ten Regel von $\lambda\beta$ durch Gödelnummerierung.
Beispiel: $(\sigma)$ wird übersetzt als $E(\overline{\overline{\overline{\ulcorner M \urcorner}}}, \overline{\overline{\overline{\ulcorner N \urcorner}}}) \rightarrow E(\overline{\overline{\overline{\ulcorner N \urcorner}}}, \overline{\overline{\overline{\ulcorner M \urcorner}}})$
Wenn die Prädikatenlogik entscheidbar wäre, wäre somit $\lambda\beta$ entscheidbar. $\square$

# 2 Kombinatorische Logik

Im folgenden seien $K$ und $S$ vorgegebene Konstanten. Wenn außer diesen noch weitere Konstanten hinzu kommen, heißt das System *angewandt* (sonst *rein*).

**Definition 2.1 (Syntax)**

- *Alle Variablen und Konstanten sind $\mathcal{CL}$-Terme (*Atome*)*
- *Mit $X$ und $Y$ ist auch $(XY)$ ein $\mathcal{CL}$-Term (*Applikation*)*

*Ein* geschlossener *$\mathcal{CL}$-Term enthält keine Variablen.*
*Ein* Kombinator *enthält nur $K$ und $S$ als Atome.*
*$FV(X)$ sei die Menge der Variablen in $X$.*
*Die Substitution von Variablen $Y[^X/z]$ ist in offensichtlicher Weise definiert, da es in $\mathcal{CL}$ keine gebundenen Variablen gibt.*

**Beispiel 2.2**

- *$Sxy(Ky)(KKSS)$ ist ein $\mathcal{CL}$-Term*
- *$S(KS)$ ist ein $\mathcal{CL}$-Term*

**Definition 2.3 (schwache Reduktion, schwache Konversion)**

$$\left.\begin{array}{rcl} U[KXY] & \rhd_{1w} & U[X] \\ U[SXYZ] & \rhd_{1w} & U[XZ(YZ)] \end{array}\right\} \quad (\text{“schwache Kontraktion”})$$

$X \rhd_w Y$, *falls* $X \equiv P_1 \rhd_{1w} P_2 \rhd_{1w} \ldots \rhd_{1w} P_n \equiv Y$ (“schwache Reduktion”)
$X =_w Y$, *falls* $X \equiv P_1 \overset{\rhd_{1w}}{\lhd_{1w}} P_2 \overset{\rhd_{1w}}{\lhd_{1w}} \ldots \overset{\rhd_{1w}}{\lhd_{1w}} P_n \equiv Y$ (“schwache Konversion”)

**Bemerkung**
$\rhd_w$ ist invariant gegenüber Substitution. Es gilt Church-Rosser.

**Definition 2.4**
*Formeln des Systems $\mathcal{CL}w$ sind alle Gleichungen $X = Y$ für $\mathcal{CL}$-Terme $X, Y$.*
*Die Axiome sind:*

($\rho$) $X = X$

($K$) $KXY = X$

($S$) $SXYZ = XZ(YZ)$

*Die Regeln sind:*

$$(\sigma)\ \frac{X = Y}{Y = X}$$

$$(\tau)\ \frac{X = Y \quad Y = Z}{X = Z}$$

$$(\mu)\ \frac{X = X'}{YX = YX'}$$

$$(\nu)\ \frac{Y = Y'}{YX = Y'X}$$

*$\mathcal{CL}w \vdash X = Y$ bedeutet, daß $X = Y$ in $\mathcal{CL}w$ ableitbar ist.*
*$\mathcal{CL}w_{\rhd} \vdash X = Y$ bedeutet, daß $X = Y$ in $\mathcal{CL}w$ ohne $(\sigma)$ ableitbar ist.*

**Lemma 2.5**

- $X =_w Y \iff \mathcal{CL}w \vdash X = Y$
- $X \rhd_w Y \iff \mathcal{CL}w_{\rhd} \vdash X = Y$

**Definition 2.6**
*Für einen $\mathcal{CL}$-Term $X$ ist der $\lambda$-Term $X_\lambda$ wie folgt definiert:*

1. $x_\lambda \equiv_{def} x$
2. $K_\lambda \equiv_{def} \lambda xy.x$
3. $S_\lambda \equiv_{def} \lambda xyz.xz(yz)$
4. $(XY)_\lambda \equiv_{def} X_\lambda Y_\lambda$

*(Wir identifizieren dabei $\alpha$-kongruente Terme.)*

**Lemma 2.7**

- $X \rhd_w Y \implies X_\lambda \rhd_\beta Y_\lambda$
- $X =_w Y \implies X_\lambda =_\beta Y_\lambda$

Beweis
Benutze $\mathcal{CL}w$ bzw. $\lambda\beta$. □

**Bemerkung**
Die Umkehrung gilt nicht. Es gilt z.B. $S_\lambda K_\lambda =_\beta K_\lambda(S_\lambda K_\lambda K_\lambda)$, nicht jedoch $SK =_w K(SKK)$.

**Definition 2.8**
*Für einen $\lambda$-Term $M$ ist der $\mathcal{CL}$-Term $M_{\mathcal{CL}}$ wie folgt definiert:*

1. $x_{\mathcal{CL}} \equiv_{def} x$
2. $(MN)_{\mathcal{CL}} \equiv_{def} M_{\mathcal{CL}} N_{\mathcal{CL}}$
3. $(\lambda x.M)_{\mathcal{CL}} \equiv_{def} [x].M_{\mathcal{CL}}$

*wobei $[x].X$ für $\mathcal{CL}$-Terme $X$ wie folgt definiert ist:*

1. $[x].x \equiv_{def} SKK$ *(abgekürzt: $I \equiv_{def} SKK$)*
2. $[x].X \equiv_{def} KX$ *falls $x \notin FV(X)$*
3. $[x].Xx \equiv_{def} X$ *falls $x \notin FV(X)$*
4. $[x].(XY) \equiv_{def} S([x].X)([x].Y)$ *falls die vorherigen Fälle nicht zutreffen*

**Beispiel 2.9**

$$\begin{aligned} [x].xxz &\equiv S([x].xx)([x].z) \\ &\equiv S(S([x].x)([x].x))(Kz) \\ &\equiv S(SII)(Kz) \end{aligned}$$

**Bemerkung**
$[x].X$ ist eine metasprachliche Operation.
$x \notin FV([x].Y)$. Insofern verhält sich $[x]$ wie ein variablenbindender Operator.

**Lemma 2.10**
$([x].Y)Z \rhd_w Y[^Z/_x]$

BEWEIS
Induktion über der Struktur von $Y$:

1. $Y \equiv x$: $([x].x)Z \equiv IZ \rhd_w Z \equiv x[^Z/_x]$
2. $Y$ ist Atom, $Y \not\equiv x$: $([x].Y)Z \equiv KYZ \rhd_w Y \equiv Y[^Z/_x]$
3. $Y \equiv (UV)$:
   - $x \notin FV(Y)$: $([x].Y)Z \equiv KYZ \rhd_w Y \equiv Y[^Z/_x]$
   - $x \notin FV(U)$, $V \equiv x$: $([x].Y)Z \equiv UZ \equiv Ux[^Z/_x]$

- keiner der vorherigen Fälle:

$$\begin{aligned}([x].Y)Z &\equiv S([x].U)([x].V)Z\\ &\rhd_w ([x].U)Z(([x].V)Z)\\ &\rhd_w (U[^Z\!/x])(V[^Z\!/x]) \quad \text{nach Induktionsvoraussetzung}\\ &\equiv Y[^Z\!/x]\end{aligned}$$

□

**Korollar 2.11 (Kombinatorische Vollständigkeit)**
*Sei $G$ ein Term mit $\{x_1, \ldots, x_n\} \subseteq FV(G)$. Dann gibt es einen Term $T$, in dem $x_1, \ldots, x_n$ nicht vorkommt, so daß $TX_1 \ldots X_n \rhd_w G[^{X_1}\!/x_1] \ldots [^{X_n}\!/x_n]$.*

Beweis
Setze $T \equiv_{def} [x_1]. \ldots [x_n].G$ □

**Bemerkung**
Damit kann man jeden Kombinator $T$, der durch eine Kontraktion $TX_1 \ldots X_n \rhd_w U$ gegeben ist, wobei $U$ nur aus $X_1, \ldots, X_n$ zusammengesetzt ist, in $\mathcal{CL}$ durch einen variablenfreien Term definieren. Mit Hilfe von $S$ und $K$ lassen sich also “alle” Kombinatoren ausdrücken.

**Lemma 2.12**

*(i) Für $\mathcal{CL}$-Terme $X$ gilt: $(X_\lambda)_{\mathcal{CL}} \equiv X$*

*(ii) Für $\lambda$-Terme $M$ gilt: $(M_{\mathcal{CL}})_\lambda =_{\beta\eta} M$*

Beweis

(i) Induktion über der Struktur von $X$:

- $(x_\lambda)_{\mathcal{CL}} \equiv x_{\mathcal{CL}} \equiv x$
- $(K_\lambda)_{\mathcal{CL}} \equiv (\lambda xy.x)_{\mathcal{CL}} \equiv [x].([y].x) \equiv [x].Kx \equiv K$
- $$\begin{aligned}(S_\lambda)_{\mathcal{CL}} &\equiv (\lambda xyz.xz(yz))_{\mathcal{CL}}\\ &\equiv [x].([y].([z].xz(yz)))\\ &\equiv [x].([y].S([z].xz)([z].yz))\\ &\equiv [x].([y].Sxy)\\ &\equiv [x].Sx\\ &\equiv S\end{aligned}$$
- $((XY)_\lambda)_{\mathcal{CL}} \equiv (X_\lambda Y_\lambda)_{\mathcal{CL}} \equiv (X_\lambda)_{\mathcal{CL}}(Y_\lambda)_{\mathcal{CL}} \equiv XY$ nach Induktionsvoraussetzung

(ii) Induktion über der Struktur von $M$:

- $(x_{\mathcal{CL}})_\lambda \equiv x_\lambda \equiv x$
- $((MN)_{\mathcal{CL}})_\lambda \equiv (M_{\mathcal{CL}}N_{\mathcal{CL}})_\lambda \equiv (M_{\mathcal{CL}})_\lambda(N_{\mathcal{CL}})_\lambda =_{\beta\eta} MN$ nach Induktionsvorauss.
- zu zeigen: $((\lambda x.M')_{\mathcal{CL}})_\lambda \equiv ([x].(M')_{\mathcal{CL}})_\lambda =_{\beta\eta} \lambda x.M'$
  Induktion über der Struktur von $M'$:
  - $([x].x_{\mathcal{CL}})_\lambda \equiv I_\lambda \equiv S_\lambda K_\lambda K_\lambda =_\beta \lambda x.x$
  - falls $x \notin FV((UV)_{\mathcal{CL}})$:
    $$\begin{aligned} ([x].(UV)_{\mathcal{CL}})_\lambda &\equiv (K(UV)_{\mathcal{CL}})_\lambda \\ &\equiv K_\lambda((UV)_{\mathcal{CL}})_\lambda \\ &\equiv (\lambda xy.x)((UV)_{\mathcal{CL}})_\lambda \qquad \text{wo } y \notin FV(UV) \\ &=_\beta \lambda y.((UV)_{\mathcal{CL}})_\lambda \\ &=_{\beta\eta} \lambda y.(UV) \qquad \text{nach Induktionsvoraussetzung} \end{aligned}$$
  - falls $x \notin FV(U_{\mathcal{CL}})$ und $V_{\mathcal{CL}} \equiv x$:
    $$\begin{aligned} ([x].(UV)_{\mathcal{CL}})_\lambda &\equiv ([x].(U_{\mathcal{CL}}V_{\mathcal{CL}}))_\lambda \\ &\equiv ([x].(U_{\mathcal{CL}}x))_\lambda \\ &\equiv (U_{\mathcal{CL}})_\lambda \\ &=_{\beta\eta} U \qquad \text{nach Induktionsvoraussetzung} \\ &=_\eta \lambda x.(Ux) \\ &\equiv \lambda x.(UV) \end{aligned}$$
  - sonst:
    $$\begin{aligned} ([x].(UV)_{\mathcal{CL}})_\lambda &\equiv (S([x].U_{\mathcal{CL}})([x].V_{\mathcal{CL}}))_\lambda \\ &\equiv S_\lambda([x].U_{\mathcal{CL}})_\lambda([x].V_{\mathcal{CL}})_\lambda \\ &=_{\beta\eta} S_\lambda(\lambda x.U)(\lambda x.V) \qquad \text{nach Induktionsvoraussetzung} \\ &\equiv (\lambda uvy.uy(vy))(\lambda x.U)(\lambda x.V) \\ &=_\beta \lambda y.(\lambda x.U)y((\lambda x.V)y) \\ &=_\beta \lambda x.(UV) \end{aligned}$$

□

**Bemerkungen**

- Es gilt also mit Lemma 2.7: $M_{\mathcal{CL}} =_w N_{\mathcal{CL}} \Longrightarrow M =_{\beta\eta} N$
- Es gilt aber *nicht*: $(M_{\mathcal{CL}})_\lambda =_\beta M$
  Zum Beispiel ist $((\lambda x.yx)_{\mathcal{CL}})_\lambda \equiv ([x].yx)_\lambda \equiv y_\lambda \equiv y \neq_\beta \lambda x.yx$
  Keine der folgenden Behauptungen gilt:
  $M_{\mathcal{CL}} \rhd_w N_{\mathcal{CL}} \Longrightarrow M \rhd_\beta N \qquad M_{\mathcal{CL}} \rhd_w N_{\mathcal{CL}} \Longleftarrow M \rhd_\beta N$
  $M_{\mathcal{CL}} =_w N_{\mathcal{CL}} \Longrightarrow M =_\beta N \qquad M_{\mathcal{CL}} =_w N_{\mathcal{CL}} \Longleftarrow M =_\beta N$

Man darf also zur Auswertung von $\lambda$-Termen, wenn es nur um $\beta$-Reduktion geht, *nicht* so verfahren: $M \rightsquigarrow M_{\mathcal{CL}} \rhd_w N_{\mathcal{CL}} \rightsquigarrow N$

Das Problem mit " $\Longrightarrow$ " besteht darin, daß $(\eta)$ in $\mathcal{CL}w$ gilt. Denn es ist

$(\lambda x.Mx)_{\mathcal{CL}} \equiv [x].M_{\mathcal{CL}}x \equiv M_{\mathcal{CL}}$ falls $x \notin FV(M)$

Das Problem mit " $\Longleftarrow$ " besteht darin, daß $(\xi)$ in $\mathcal{CL}w$ nicht gilt. Denn es ist

$$\begin{aligned}
[x].Sxyz &\equiv S([x].Sxy)([x].z) \\
&\equiv S(S([x].Sx)([x].y))(Kz) \\
&\equiv S(SS(Ky))(Kz) \\
[x].xz(yz) &\equiv S([x].xz)([x].yz) \\
&\equiv S(S([x].x)([x].z))(K(yz)) \\
&\equiv S(SI(Kz))(K(yz))
\end{aligned}$$

Also ist zwar $Sxyz =_w xz(yz)$, aber nicht $[x].Sxyz =_w [x].xz(yz)$.

- Die Hinzunahme von $(\xi)$ zu $\mathcal{CL}w$ bewirkt *volle* Extensionalität:

  $$(\xi)\,\dfrac{\dfrac{Xx = Yx}{[x].Xx = [x].Yx}}{X = Y}$$

  Während also im $\lambda\beta$-Kalkül die Hinzunahme von $(\eta)$ Extensionalität zur Folge hat, wohingegen $(\xi)$ eo ipso gilt, bewirkt in der Kombinatorischen Logik die Hinzunahme von $(\xi)$ Extensionalität, während $(\eta)$ eo ipso gilt. Das zeigt die Disparatheit beider Systeme.

- Wir definieren $\succ\!-$ durch Erweiterung von $\mathcal{CL}w_{\rhd}$ um

  $$(\xi)\quad \frac{X \succ\!- Y}{[x].X \succ\!- [x].Y}$$

  Dann gilt für $\lambda$-Terme:

  $M \rhd_{\beta\eta} N \;\Longrightarrow\; M_{\mathcal{CL}} \succ\!- N_{\mathcal{CL}}$

  $M =_{\beta\eta} N \;\Longleftarrow\; M_{\mathcal{CL}} \succ\!- N_{\mathcal{CL}}$

  $\eta$-Konversion ist jetzt Fall von $(\rho)$ und gilt in beliebiger Richtung. Daraus ergibt sich:

  $M =_{\beta\eta} N \iff M_{\mathcal{CL}} \succ\!\!\prec N_{\mathcal{CL}}$ , wobei $\succ\!\!\prec$ der symmetrische Abschluß von $\succ\!-$ ist.

  Damit gilt für $\mathcal{CL}$-Terme $X, Y$:

  $X \succ\!\!\prec Y \iff X_\lambda =_{\beta\eta} Y_\lambda$

- Um $\beta$-Gleichheit in $\mathcal{CL}w$ repräsentieren zu können, kann man die Definition von $[x].Y$ so abschwächen, daß $(\eta)$ nicht automatisch gilt, z.B. durch Weglassen der Klausel $[x].Xx \equiv X$ (falls $x \notin FV(X)$) in der Definition von $[x]$.

  Jedoch gilt selbst dann $(\xi)$ nicht:

  $[x].Sxyz \equiv S(S(S(KS)I)(Ky))(Kz)$

$[x].xz(yz) \equiv S(SI(Kz))(K(yz))$
Man kann mit dieser neuen Definition von $[x]$ jedoch zeigen, daß
$\lambda\beta \vdash M = N \iff (\mathcal{CL}w + \otimes) \vdash M_{\mathcal{CL}} = N_{\mathcal{CL}}$
$(\mathcal{CL}w + \otimes) \vdash X = Y \iff \lambda\beta \vdash X_\lambda = Y_\lambda$
wobei $\otimes$ eine Erweiterung von $\mathcal{CL}w$ um ein bestimmtes Regelschema bzw. eine endliche Menge von Axiomen ist (vgl. Hindley/Seldin Ch. 9).
$(S_\lambda)_{\mathcal{CL}} =_w S$ gilt bei der modifizierten Definition von $[x]$ nicht. Eine weitergehende Modifikation ist jedoch möglich, so daß statt Lemma 2.12 jetzt gilt:

(i) $(X_\lambda)_{\mathcal{CL}} \equiv X$

(ii) $(M_{\mathcal{CL}})_\lambda =_\beta M$

Damit ist die Auswertung von übersetzten $\lambda$-Termen in $\mathcal{CL}w$ korrekt, was in der funktionalen Programmierung oft ausgenutzt wird.

- Bezüglich Vollständigkeit gilt folgendes: wir betrachten eine erweiterte Sprache, in der zusätzliche Funktionen ausgewertet werden können (sog. $\delta$-Regeln). Die entsprechenden Reduktionen sind $\rhd_{1\beta\delta}, \rhd_{\beta\delta}, \rhd_{1l\beta\delta}$ ("l" für leftmost). Dann gilt:
  Wenn $M \rhd_{1l\beta\delta} N$, mit $M$ abgeschlossen und nicht von der Form $[x].P$, so ist $M_{\mathcal{CL}} \rhd_{1w\delta} N_{\mathcal{CL}}$.
  Falls ein getyptes System 2. Stufe mit $Int, Bool, Char$ als Grundtypen gegeben ist, in dem ein Fixpunktoperator $\Upsilon$ existiert und $M$ vom Grundtyp ist, dann gilt:
  $M \rhd_{l\beta\delta} N \implies M_{\mathcal{CL}} \rhd_{w\delta} N_{\mathcal{CL}}$
  Das bedeutet, daß man *alles*, was man in $\lambda\beta$ finden kann, durch Übersetzung in $\mathcal{CL}w$ finden kann. Dies wird zum Beispiel in der funktionalen Programmiersprache Miranda ausgenutzt (Turner 1979).
  Es ergibt sich daraus für eine Konstante $c$ eines Grundtyps (die per definitionem in $\beta$-Normalform ist):
  $\lambda\delta \vdash M = c \implies M =_{\beta\delta} c \implies M \rhd_{l\beta\delta} c \implies M_{\mathcal{CL}} \rhd_{w\delta} c \implies \mathcal{CL}w\delta \vdash M_{\mathcal{CL}} = c$
  sowie umgekehrt
  $\mathcal{CL}w\delta \vdash M_{\mathcal{CL}} = c \implies M_{\mathcal{CL}} =_{w\delta} c \implies M_{\mathcal{CL}} \rhd_{w\delta} c \implies \underbrace{(M_{\mathcal{CL}})_\lambda}_{=_\beta M} \rhd_{\beta\delta} c \implies \lambda\beta \vdash M = c$
  Jede Berechnung eines Wertes für $M$ kann also in $\mathcal{CL}w$ durchgeführt werden.

# 3 Der getypte λ-Kalkül

Es gibt zwei Versionen der Typisierung des λ-Kalküls:

- *Curry-Typisierung*: Terme sind die Terme der ungetypten Theorie. Jeder Term hat eine Menge möglicher Typen (*implizite* Typisierung, "type assignment").
- *Church-Typisierung*: Terme haben assoziierte Typen, die damit in der Regel eindeutig sind (*explizite* Typisierung).

Wir behandeln die Curry-Typisierung, und zwar für die einfachste Form, die nur Funktionstypen enthält (Bezeichnung des Kalküls: $\lambda\!\!\rightarrow$). Wir folgen dabei der Darstellung in Barendregt (1992).

**Bemerkung**
Für den getypten λ-Kalkül ($\lambda\!\!\rightarrow$) gilt starke Normalisierung. Daher sind nicht alle rekursiven Funktionen definierbar — die partiellen Funktionen sowieso nicht, aber auch nicht alle totalen. Definiere dazu die Funktion $F$ wie folgt:
$F(n,m) = k \iff$ der $n$-te getypte Term angewandt auf Argument $\underline{m}$ hat die $\beta$-Normalform $\underline{k}$,
$F(n,m) = 0$ sonst (bei geeigneter Aufzählung der getypten Terme).
Dann kann die (totale) Funktion $g(n) =_{def} F(n,n) + 1$ nicht in $\lambda\!\!\rightarrow$ definierbar sein:
Sei $g$ in $\lambda\!\!\rightarrow$ definiert durch den $p$-ten getypten Term. Dann ist $g(p) = F(p,p)$; aber nach Definition ist $g(p) = F(p,p) + 1$, ein Widerspruch.

**Definition 3.1**
*Die Menge der* Typen $T$ *von* $\lambda\!\!\rightarrow$ *ist wie folgt definiert:*

1. Typvariable $\alpha, \alpha', \alpha'', \alpha''', \ldots$ *sind Typen*
2. *Mit* $\sigma$ *und* $\tau$ *ist* $\sigma \rightarrow \tau$ *ein Typ*

*Es steht* $\sigma_1 \rightarrow \sigma_2 \rightarrow \cdots \rightarrow \sigma_{n-1} \rightarrow \sigma_n$ *für* $\sigma_1 \rightarrow (\sigma_2 \rightarrow (\cdots (\sigma_{n-1} \rightarrow \sigma_n) \cdots))$
*Ein* Urteil *hat die Form* $M\!:\!\sigma$ *für einen λ-Term* $M$ *und einen Typ* $\sigma$. *Dabei heißt* $M$ *das* Subjekt *des Urteils.*
*Eine* Deklaration *ist ein Urteil, dessen Subjekt eine Termvariable ist.*
*Eine* Basis $\Gamma$ *ist eine endliche Menge von Deklarationen, deren Subjekte paarweise verschieden sind. Eine* Sequenz *hat die Form* $\Gamma \vdash M\!:\!\sigma$ *für eine Basis* $\Gamma$ *und ein Urteil* $M\!:\!\sigma$.
*Mit "Variable" meinen wir immer "Termvariable".*

**Definition 3.2**
*Sequenzen $\Gamma \vdash M:\sigma$, die ausdrücken, daß das Urteil $M:\sigma$ in der Basis $\Gamma$ gilt, kann man mit folgenden Regeln im Kalkül $\lambda\!\rightarrow$ herleiten:*

- $(I)$ $\Gamma, x:\sigma \vdash x:\sigma$
- $(\rightarrow I)$ $\dfrac{\Gamma, x:\sigma \vdash M:\tau}{\Gamma \vdash (\lambda x.M):\sigma\rightarrow\tau}$
- $(\rightarrow E)$ $\dfrac{\Gamma \vdash M:\sigma\rightarrow\tau \quad \Gamma \vdash N:\sigma}{\Gamma \vdash MN:\tau}$

*Ist $\Gamma \vdash M:\sigma$ in $\lambda\!\rightarrow$ herleitbar, schreiben wir $\Gamma \vdash_{\lambda\rightarrow} M:\sigma$ oder auch $\Gamma \vdash M:\sigma$. Wir identifizieren also häufig Sequenzen mit der Behauptung ihrer Herleitbarkeit. Aus dem Kontext ergibt es sich dann, was gemeint ist.*

**Beispiel 3.3**

- $\vdash \lambda xy.x:\sigma\rightarrow\tau\rightarrow\sigma$

$$\dfrac{\dfrac{\dfrac{}{x:\sigma, y:\tau \vdash x:\sigma}}{x:\sigma \vdash \lambda y.x:\tau\rightarrow\sigma}}{\vdash \lambda x.\lambda y.x:\sigma\rightarrow\tau\rightarrow\sigma}$$

- $\vdash \lambda xyz.xz(yz):(\sigma\rightarrow\tau\rightarrow\tau')\rightarrow(\sigma\rightarrow\tau)\rightarrow\sigma\rightarrow\tau'$ *(S-Kombinator)*

**Bemerkung**
Konstanten können hinzukommen. Entsprechende Konstantendeklarationen gehören dann zu jeder Basis dazu. Ein Beispiel ist der Fixpunktkombinator vom Typ $Y:(\sigma\rightarrow\sigma)\rightarrow\sigma$ für alle $\sigma$ in der Programmiersprache ML.

**Definition 3.4**
*Ein geschlossener Term $M$ heißt* typbar*, falls $\vdash M:\sigma$ für ein $\sigma$. Ein Term $M$ mit freien Variablen $x_1, \ldots, x_n$ heißt typbar, falls $\Gamma \vdash M:\sigma$ für ein $\sigma$, wobei $\Gamma = \{x_1:\sigma_1, \ldots, x_n:\sigma_n\}$ für gewisse $\sigma_1, \ldots, \sigma_n$.*
*Sei $\Gamma = \{x_1:\sigma_1, \ldots, x_n:\sigma_n\}$ eine Basis. Dann sei $\Gamma(x_i) =_{def} \sigma_i$.*
*Sei $V$ eine Menge von Variablen. Dann sei $\Gamma|_V = \{x:\sigma \mid x \in V, \Gamma(x) =_{def} \sigma\}$*
*$dom\,\Gamma =_{def} \{x_1, \ldots, x_n\}$*
*Substitution von Typen: $\sigma[{}^{\tau}\!/\!_{\alpha}]$ bedeutet eine gleichzeitige Ersetzung aller in $\sigma$ vorkommenden $\alpha$ durch $\tau$.*

**Lemma 3.5**

1. $\Gamma \subseteq \Gamma' \implies (\Gamma \vdash M:\sigma \implies \Gamma' \vdash M:\sigma)$ *(Monotonie)*
2. $\Gamma \vdash M:\sigma \implies FV(M) \subseteq dom\,\Gamma$
3. $\Gamma \vdash M:\sigma \implies \Gamma|_{FV(M)} \vdash M:\sigma$
4. $\Gamma \vdash x:\sigma \implies (x:\sigma) \in \Gamma$
5. $\Gamma \vdash MN:\sigma \implies \Gamma \vdash M:\tau \to \sigma$ *und* $\Gamma \vdash N:\tau$ *für ein* $\tau$
6. $\Gamma \vdash (\lambda x.M):\sigma \implies \sigma \equiv \sigma_1 \to \sigma_2$ *für gewisse* $\sigma_1, \sigma_2$ *und* $\Gamma, x:\sigma_1 \vdash M:\sigma_2$
7. $M'$ *ist Teilterm von* $M$ *und* $\Gamma \vdash M:\sigma \implies \Gamma' \vdash M':\sigma'$ *für gewisse* $\Gamma', \sigma'$
8. $\Gamma \vdash M:\sigma \implies \Gamma[{}^{\tau}\!/\!_{\alpha}] \vdash M:\sigma[{}^{\tau}\!/\!_{\alpha}]$
9. $\Gamma, x:\sigma \vdash M:\tau,\ \Gamma \vdash N:\sigma \implies \Gamma \vdash M[{}^{N}\!/\!_{x}]:\tau$
10. $M \rhd_\beta M' \implies (\Gamma \vdash M:\sigma \implies \Gamma \vdash M':\sigma)$ *(Subjektreduktion)*

**Bemerkung**
Umgekehrt (zu Lemma 3.1 (10)) gilt nicht Invarianz gegenüber Expansion $\lhd_\beta$:
$M \lhd_\beta M' \not\Longrightarrow (\Gamma \vdash M:\sigma \implies \Gamma \vdash M':\sigma)$.
Beispiel: $\vdash I:\sigma \to \sigma$, aber $\nvdash KI(\lambda x.xx):\sigma \to \sigma$, obwohl $I \lhd_\beta KI(\lambda x.xx)$

Wir zeigen jetzt, daß alle typbaren Terme stark normalisierbar sind. (Die schwache Normalisierbarkeit wurde schon von Turing gezeigt; die starke Normalisierbarkeit geht auf Tait zurück.) Die Umkehrung dieser Behauptung gilt nicht, wie das Beispiel des nicht typbaren Terms $\lambda x.xx$ zeigt.

**Definition 3.6**
*Sei* $\mathcal{SN}$ *die Menge der stark normalisierbaren* $\lambda$*-Terme.*
*Für Mengen* $A, B$ *von* $\lambda$*-Termen sei*

- $A \to B =_{def} \{M \mid$ *für alle* $N \in A$ *ist* $MN \in B\}$
- $[\![\alpha]\!] =_{def} \mathcal{SN}$ *für alle Typvariablen* $\alpha$
- $[\![\sigma \to \tau]\!] =_{def} [\![\sigma]\!] \to [\![\tau]\!]$

**Definition 3.7** *Eine Menge* $A$ *von Termen heißt* saturiert*, wenn gilt:*

*a)* $A \subseteq \mathcal{SN}$

*b)* $xR_1 \ldots R_n \in A$, *falls* $x$ *Termvariable und* $R_1, \ldots, R_n \in \mathcal{SN}$ $(n \geq 0)$

*c)* $(\lambda x.M)NR_1 \ldots R_n \in A$, *falls* $(M[{}^N\!/\!_x])R_1 \ldots R_n \in A$ *für* $N, R_1, \ldots, R_n \in \mathcal{SN}$ $(n \geq 0)$

$\mathcal{SAT} =_{def} \{A \mid A saturiert\}$

**Lemma 3.8**
*Für jeden Typ* $\sigma$ *von* $\lambda\!\to$ *gilt:* $[\![\sigma]\!]$ *ist saturiert.*

Beweis

1. $\sigma$ ist Typvariable. Zu zeigen: $\mathcal{SN}$ ist saturiert.

   a) $\mathcal{SN} \subseteq \mathcal{SN}$

   b) $xR_1 \ldots R_n \in \mathcal{SN}$, falls $R_1, \ldots, R_n \in \mathcal{SN}$

   c) Sei $(M[{}^N\!/\!_x])R_1 \ldots R_n \in \mathcal{SN}$ mit $N, R_1, \ldots, R_n \in \mathcal{SN}$
   Dann gilt auch: $M \in \mathcal{SN}$, sonst könnte $(M[{}^N\!/\!_x])R_1 \ldots R_n$ nicht stark normalisierbar sein. Wir betrachten $(\lambda x.M)NR_1 \ldots R_n$. Jede Reduktionfolge sieht dann so aus:
   $$\begin{array}{rcl} (\lambda x.M)NR_1 \ldots R_n & \rhd_\beta & \ldots \\ & \rhd_\beta & (\lambda x.M')N'R'_1 \ldots R'_n \\ & \rhd_{1\beta} & M'[{}^{N'}\!/\!_x]R_1 \ldots R_n \\ & \rhd_\beta & \ldots \end{array}$$
   wobei $M \rhd_\beta M'$, $N \rhd_\beta N'$ und $R_i \rhd_\beta R'_i$ für alle $i$.
   Damit erhält man: $(M[{}^N\!/\!_x])R_1 \ldots R_n \rhd_\beta (M'[{}^{N'}\!/\!_x])R'_1 \ldots R'_n \rhd_\beta \cdots$. Da diese Folge terminiert, terminiert auch die erste, d.h. $(\lambda x.M)NR_1 \ldots R_n$ ist stark normalisierbar.

2. Mit $A$ und $B$ ist $A \to B$ saturiert:

   a) Sei $M \in A \to B$. Wegen Definition 3.7 b) gilt $x \in A$ für alle Variablen $x$. Also $Mx \in B$. Da $Mx$ stark normalisierbar, ist $M$ stark normalisierbar.

   b) Sei $M \in A$. Dann ist $M \in \mathcal{SN}$. Dann ist $xR_1 \ldots R_nM \in B$ für $R_i \in \mathcal{SN}$. Damit ist auch $xR_1 \ldots R_n \in A \to B$.

   c) Sei $M \in A$. Dann ist $M \in \mathcal{SN}$.
   Dann $(\lambda x.P)NR_1 \ldots R_nM \in B$, falls $(P[{}^N\!/\!_x])R_1 \ldots R_nM \in B$.
   Dann $(\lambda x.P)NR_1 \ldots R_n \in A \to B$, falls $(P[{}^N\!/\!_x])R_1 \ldots R_n \in A \to B$.

□

**Definition 3.9**
*Eine* Bewertung $\rho$ *ist eine Abbildung* $\rho$ : *Variablen* $\longrightarrow$ $\lambda$-*Terme.*
$[\![M]\!]_\rho = M[{}^{\rho(x_1)}\!/x_1, \ldots, {}^{\rho(x_n)}\!/x_n]$*, wobei* $\{x_1, \ldots, x_n\}$ *Menge* aller *freien Variablen in* $M$*.*
$\rho \models M\!:\!\sigma$ *("*$\rho$ erfüllt $M\!:\!\sigma$*"), falls* $[\![M]\!]_\rho \in [\![\sigma]\!]$.
$\rho \models \Gamma$*, falls* $\rho \models x\!:\!\sigma$ *für alle* $(x\!:\!\sigma) \in \Gamma$.
$\Gamma \models M\!:\!\sigma$*, falls für alle* $\rho$ *gilt:* $\rho \models \Gamma \Longrightarrow \rho \models M\!:\!\sigma$.

**Lemma 3.10**
$\Gamma \vdash M\!:\!\sigma \Longrightarrow \Gamma \models M\!:\!\sigma$.

Beweis
Induktion über der Struktur der Ableitung von $M\!:\!\sigma$.

1. $\Gamma', x\!:\!\sigma \vdash x\!:\!\sigma$ $\surd$
2. $M \equiv PQ$: Dann $\Gamma \vdash P\!:\!\tau \!\to\! \sigma$ und $\Gamma \vdash Q\!:\!\tau$.
   Nach Induktionsvoraussetzung $\Gamma \models P\!:\!\tau \!\to\! \sigma$ und $\Gamma \models Q\!:\!\tau$.
   D.h., falls $\rho \models \Gamma$, dann $[\![P]\!]_\rho \in [\![\tau \!\to\! \sigma]\!]$ und $[\![Q]\!]_\rho \in [\![\tau]\!]$.
   Dann $[\![PQ]\!]_\rho \in [\![\sigma]\!]$ nach Definition von $[\![\ ]\!]$, da $[\![PQ]\!]_\rho = [\![P]\!]_\rho [\![Q]\!]_\rho$.
3. $M \equiv \lambda x.N$: Dann ist $\sigma \equiv \sigma_1 \!\to\! \sigma_2$ und $\Gamma, x\!:\!\sigma_1 \vdash N\!:\!\sigma_2$.
   Nach Induktionsvoraussetzung: $\Gamma, x\!:\!\sigma_1 \models N\!:\!\sigma_2$.
   D.h., falls $\rho \models \Gamma$ und $\rho(x) \in [\![\sigma_1]\!]$, dann $[\![N]\!]_\rho \in [\![\sigma_2]\!]$.
   Dann ist $[\![(\lambda x.N)x]\!]_\rho \in [\![\sigma_2]\!]$, da $[\![\sigma_2]\!]$ saturiert.
   D.h., falls $\rho \models \Gamma$ und $\rho(x) = Q \in [\![\sigma_1]\!]$, dann $[\![\lambda x.N]\!]_\rho Q = [\![(\lambda x.N)x]\!]_\rho \in [\![\sigma_2]\!]$.
   Also $[\![\lambda x.N]\!]_\rho \in [\![\sigma_1 \!\to\! \sigma_2]\!]$.

□

**Theorem 3.11** $\Gamma \vdash M\!:\!\sigma \implies M$ *ist stark normalisierbar.*

Beweis
$\Gamma \vdash M\!:\!\sigma \implies \Gamma \models M\!:\!\sigma \implies (\forall\rho)(\,\rho \models \Gamma \Longrightarrow \rho \models M\!:\!\sigma)$.
Wir wählen $\rho = id$. $id \models \Gamma$ gilt, da $x\!:\![\![\sigma]\!]$ für jedes $\sigma$ (da $\sigma$ saturiert).
Also gilt $M\!:\![\![\sigma]\!]$. $[\![\sigma]\!]$ ist saturiert und damit stark normalisierbar. □

Im folgenden stellen wir einen Algorithmus zur Typisierung von Termen dar. Dieser Algorithmus liefert einen Typ, falls der Term typisierbar ist, und gibt `fail` aus, falls dieser Term nicht typisierbar ist. Auf diese Weise lassen sich folgende beiden Entscheidbarkeitsfragen lösen:

1. Gegeben $M$ und $\sigma$, gilt $\vdash M:\sigma$ ? (Typprüfung, "type checking")

2. Gegeben $M$, gibt es $\sigma$ mit $\vdash M:\sigma$ ? (Typbarkeit, "typability")

Die dritte Entscheidbarkeitsfrage in diesem Kontext

3. Gegeben $\sigma$, gibt es $M$ mit $\vdash M:\sigma$ ? ("inhabitation")

wird durch den Curry-Howard-Isomorphismus gelöst, der am Schluß dieses Kapitels behandelt wird.

Zunächst sind einige Vorbereitungen über Substitution und Unifikation von Typvariablen und Typen notwendig.

**Definition 3.12**
*Eine* Substitution (in Typen) *ist eine Abbildung*
$s$ : *Typvariablen* $\longrightarrow$ *Typen,*
*wobei* $s(\alpha) = \alpha$ *für unendlich viele* $\alpha$. *Wir schreiben* $s$ *auch als* $\{\alpha_1 = \sigma_1, \ldots, \alpha_n = \sigma_n\}$, *falls* $s(\alpha_i) = \sigma_i$.
*Offenbar determiniert* $s$ *eine Abbildung* $\bar{s}$ *von Typen in Typen:*

$$\begin{array}{rcl} \bar{s}(\alpha) & \equiv & s(\alpha) \\ \bar{s}(\sigma \to \tau) & \equiv & \bar{s}(\sigma) \to \bar{s}(\tau) \end{array}$$

*Wir identifizieren* $s$ *und* $\bar{s}$ *und schreiben* $s(\sigma)$ *oder* $\sigma[{}^{\sigma_1}/_{\alpha_1}, \ldots, {}^{\sigma_n}/_{\alpha_n}]$.
*Für Substitutionen* $s_1$ *und* $s_2$ *ist* $s_1 \circ s_2$ *(kurz:* $s_1 s_2$*) in natürlicher Weise definiert. Entsprechend ist* $s_1 \circ s_2(\sigma) \equiv s_1(s_2(\sigma))$.
*Ein* Unifikator *für* $\sigma$ *und* $\tau$ *ist ein* $s$ *mit* $s(\sigma) \equiv s(\tau)$, *ein Unifikator für eine Menge von Gleichungen* $E = \{\sigma_1 = \tau_1, \ldots, \sigma_n = \tau_n\}$ *ist ein* $s$ *mit* $s(\sigma_i) = s(\tau_i)$ *für alle* $i$ $(1 \leq i \leq n)$.
*Ein* allgemeinster Unifikator (mgu — "most general unifier") *von* $\sigma$ *und* $\tau$ *(bzgl.* $E$*) ist ein Unifikator* $s$, *so daß für jeden anderen Unifikator* $s'$ *von* $\sigma$ *und* $\tau$ *(bzgl.* $E$*) gilt:* $s' = s_1 \circ s$ *für eine Substitution* $s_1$.
*Wir schreiben:* $s = mgu(\sigma, \tau)$ *bzw.* $s = mgu(E)$.
$\tau$ *ist* Variante *von* $\sigma$, *falls es* $s_1$ *und* $s_2$ *gibt mit* $s_1(\tau) = \sigma$ *und* $s_2(\sigma) = \tau$.

**Beispiel 3.13**

- $\alpha \to (\beta \to \alpha)$ *und* $\alpha \to (\beta \to \alpha)$ *haben* $\{{}^{\alpha}/_{\beta}\}$ *als mgu.*

- $\beta \to (\alpha \to \beta)$ *und* $(\gamma \to \gamma) \to \delta$ *haben* $\{{}^{\gamma \to \gamma}/_{\beta}, {}^{\alpha \to (\gamma \to \gamma)}/_{\delta}\}$ *als mgu.*

**Bemerkung**
Zwei *mgu*s bezüglich derselben Menge sind immer Varianten voneinander. In diesem Sinn sind also *mgu*s eindeutig.

**Theorem 3.14**
*Es gibt einen Algorithmus, der zu jedem Gleichungssystem $E$ einen mgu von $E$ liefert, falls $E$ unifizierbar ist, und* `fail` *liefert, falls $E$ nicht unifizierbar ist.*

Beweis
Siehe Logikprogrammierung. Ein Verfahren, das auf Herbrand zurückgeht, besteht aus Regeln zur Umformung von Gleichungssystemen $E = \{\sigma_1 = \tau_1, \ldots, \sigma_n = \tau_n\}$:

- $(id)$ $\dfrac{E \ \dot\cup \ \{\sigma = \sigma\}}{E}$
- $(sym)$ $\dfrac{E \ \dot\cup \ \{\sigma = \alpha\}}{E \ \cup \ \{\alpha = \sigma\}}$ falls $\sigma$ keine Variable ist
- $(subst)$ $\dfrac{E \ \dot\cup \ \{\alpha = \sigma\}}{E[\sigma/\alpha] \ \cup \ \{\alpha = \sigma\}}$ falls $\alpha$ nicht in $\sigma$ vorkommt und $\alpha$ in $E$ vorkommt
- $(fail)$ $\dfrac{E \ \dot\cup \ \{\alpha = \sigma\}}{\texttt{fail}}$ falls $\alpha$ in $\sigma$ vorkommt
- $(func)$ $\dfrac{E \ \dot\cup \ \{\tau_1 \to \tau_2 = \sigma_1 \to \sigma_2\}}{E \ \cup \ \{\tau_1 = \sigma_1, \tau_2 = \sigma_2\}}$

Es gilt: die Anwendung dieser Reduktionsregeln auf eine Gleichungsmenge terminiert. Das Resultat ist entweder `fail` oder ist $\{\alpha_1 = \sigma_1, \ldots, \alpha_n = \sigma_n\}$, wobei $\{\alpha_1 = \sigma_1, \ldots, \alpha_n = \sigma_n\}$ der *mgu* von $E$ ist. □

Gleichungssystemen dieser Art ordnen wir nun Sequenzen zu.

**Definition 3.15**
*Das mit der Sequenz $\Gamma \vdash M : \sigma$ assoziierte Gleichungssystem $E(\Gamma \vdash M : \sigma)$ ist wie folgt definiert:*

1. $E(\Gamma \vdash x : \sigma) =_{def} \{\sigma = \Gamma(x)\}$
2. $E(\Gamma \vdash \lambda x.M : \sigma) =_{def} \{\sigma = \alpha \to \beta\} \cup E(\Gamma, x : \alpha \vdash M : \beta)$ *für neue Typvariablen $\alpha, \beta$*
3. $E(\Gamma \vdash MN : \sigma) =_{def} E(\Gamma \vdash M : \alpha \to \sigma) \cup E(\Gamma \vdash N : \alpha)$ *für neue Typvariable $\alpha$*

**Beispiel 3.16**

- $E(x : \alpha \vdash x : \beta) = \{\beta = \alpha\}$

- $$\begin{aligned} E(\vdash \lambda xy.x : \alpha \to \beta) &= \{\alpha \to \beta = \alpha_1 \to \alpha_2\} \cup E(x:\alpha_1 \vdash \lambda y.x : \alpha_2) \\ &= \{\alpha \to \beta = \alpha_1 \to \alpha_2, \alpha_2 = \alpha_3 \to \alpha_4\} \cup E(x:\alpha_1, y:\alpha_3 \vdash x:\alpha_4) \\ &= \{\alpha \to \beta = \alpha_1 \to \alpha_2, \alpha_2 = \alpha_3 \to \alpha_4, \alpha_4 = \alpha_1\} \end{aligned}$$

  *Lösung:*

  $\{\alpha \to \beta = \alpha_1 \to \alpha_2, \alpha_2 = \alpha_3 \to \alpha_4, \alpha_4 = \alpha_1\}$

  $\implies \{\alpha = \alpha_1, \beta = \alpha_2, \alpha_2 = \alpha_3 \to \alpha_4, \alpha_4 = \alpha_1\}$

  $\implies \{\alpha = \alpha_1, \beta = \alpha_3 \to \alpha_4, \alpha_2 = \alpha_3 \to \alpha_4, \alpha_4 = \alpha_1\}$

  $\implies \{\alpha = \alpha_1, \beta = \alpha_3 \to \alpha_1, \alpha_2 = \alpha_3 \to \alpha_1, \alpha_4 = \alpha_1\}$

  *Es gilt also* $\vdash \lambda xy.x : \alpha_1 \to (\alpha_3 \to \alpha_1)$

**Lemma 3.17**

*(i) Sei $s$ Lösung von $E(\Gamma \vdash M:\sigma)$. Dann gilt: $s(\Gamma) \vdash M : s(\sigma)$*

*(ii) Wenn $s(\Gamma) \vdash M : s(\sigma)$, dann gilt: es gibt ein $s'$, das die Typvariablen in $\Gamma$ und $\sigma$ wie $s$ interpretiert, und $s'$ ist Lösung von $E(\Gamma \vdash M:\sigma)$. Dabei können die Typvariablen, die von $s$ und $s'$ verschieden interpretiert werden, aus einer festen Typvariablenmenge $V$ mit $V \cap FV(\Gamma \cup \{\sigma\}) = \emptyset$ gewählt werden.*

Beweis

(i) Induktion über der Struktur von $M$

- $s$ ist Lösung von $E(\Gamma \vdash x:\sigma) \implies s(\sigma) = s(\Gamma(x)) \implies x:s(\sigma)$ kommt in $\Gamma$ vor $\implies$ $s(\Gamma) \vdash x:s(\sigma)$
- $s$ ist Lösung von $E(\Gamma \vdash \lambda x.M:\sigma) \implies s$ ist Lösung von $\{\sigma = \alpha \to \beta\} \cup E(\Gamma, x:\alpha \vdash M:\beta)$
  Nach Induktionsvoraussetzung ist $s(\Gamma), x:s(\alpha) \vdash M:s(\beta)$, damit $s(\Gamma) \vdash \lambda x.M : (s(\alpha) \to s(\beta)) = s(\sigma)$
- $s$ ist Lösung von $E(\Gamma \vdash MN:\sigma) \implies s$ ist Lösung von $E(\Gamma \vdash M:\alpha \to \sigma)$ und $s$ ist Lösung von $E(\Gamma \vdash N:\alpha)$
  Nach Induktionsvoraussetzung ist $s(\Gamma) \vdash M:s(\alpha \to \sigma)$ und $s(\Gamma) \vdash N:s(\alpha)$, damit $s(\Gamma) \vdash MN:s(\sigma)$

(ii) Induktion über der Struktur von $M$

- $s(\Gamma) \vdash x:s(\sigma) \implies (x:s(\sigma)) \in s(\Gamma) \implies s(\sigma) = s(\Gamma(x))$
  $\implies s$ ist Lösung von $E(\Gamma \vdash x:\sigma)$
- $s(\Gamma) \vdash \lambda x.M:s(\sigma) \implies s(\sigma) = \sigma_1 \to \sigma_2$ für gewisse $\sigma_1, \sigma_2$ und $s(\Gamma), x:\sigma_1 \vdash M:\sigma_2$.
  Sei $s'$ wie $s$, jedoch erweitert um $\alpha_1 \mapsto \sigma_1$ und $\alpha_2 \mapsto \sigma_2$ mit $\alpha_1, \alpha_2$ neu. Dann ist

$s'(\Gamma), x\!:\!\alpha_1 \vdash M\!:\!s'(\alpha_2)$. Damit stimmen $s$ und $s'$ auf den in $\Gamma$ und $\sigma$ vorkommenden Typvariablen überein, und $s'$ ist Lösung von $\{\sigma = \alpha_1 \to \alpha_2\}$. Nach Induktionsvoraussetzung gibt es $s''$ als Lösung von $E(\Gamma, x\!:\!\alpha_1 \vdash M\!:\!\alpha_2)$ derart, daß $s''$ mit $s'$ auf den Typvariablen in $\Gamma, \alpha_1, \alpha_2$ übereinstimmt. Ferner kann angenommen werden, daß die Typvariablen, die von $s'$ und $s''$ verschieden interpretiert werden, nicht in $\sigma$ vorkommen, also $s'(\sigma) = s''(\sigma)$.

Damit ist $s''$ also Lösung von $\{\sigma = \alpha_1 \to \alpha_2\} \cup E(\Gamma, x\!:\!\alpha_1 \vdash M\!:\!\alpha_2)$, das heißt von $E(\Gamma \vdash \lambda x.M\!:\!\sigma)$, und stimmt auf den Typvariablen in $\Gamma$ und $\sigma$ mit $s$ überein.

- $s(\Gamma) \vdash MN\!:\!\sigma \implies s(\Gamma) \vdash M\!:\!\tau \to s(\sigma)$ und $s(\Gamma) \vdash N\!:\!\tau$ für ein $\tau$

  Wir definieren $s'$ als $s$, erweitert um $\alpha \mapsto \tau$, wobei $\alpha$ neu. Dann stimmt $s'$ mit $s$ auf den Typvariablen in $\Gamma$ und $\sigma$ überein. Wir haben also $s'(\Gamma) \vdash M\!:\!s'(\alpha) \to s'(\sigma)$ und $s'(\Gamma) \vdash N\!:\!s'(\alpha)$. Nach Induktionsvoraussetzung gibt es Lösungen $s_1''$ und $s_2''$ von $E(\Gamma \vdash M\!:\!\alpha \to \sigma)$ bzw. $E(\Gamma \vdash N\!:\!\alpha)$, die mit $s'$ auf den Typvariablen in $\Gamma, \sigma, \alpha$ übereinstimmen. Ferner können wir annehmen, daß die bei der Konstruktion von $E(\Gamma \vdash M\!:\!\alpha \to \sigma)$ und $E(\Gamma \vdash N\!:\!\alpha)$ neu eingeführten Typvariablen voneinander verschieden sind.

  Dann ist $s_1'' \cup s_2''$ Lösung von $E(\Gamma \vdash M\!:\!\alpha \to \sigma) \cup E(\Gamma \vdash N\!:\!\alpha)$, das heißt also von $E(\Gamma \vdash MN\!:\!\sigma)$, und stimmt mit $s$ auf den Typvariablen in $\Gamma$ und $\sigma$ überein.

$\square$

**Definition 3.18**

*$\sigma$ heißt* Haupttyp *für einen geschlossenen Term $M$, falls $\vdash M\!:\!\sigma$ und ferner:*
*falls $\vdash M\!:\!\sigma$ und $\vdash M\!:\!\sigma'$, dann gibt es eine Substitution $s$, so daß $\sigma' = s(\sigma)$.*
*$(\Gamma, \sigma)$ heißt* Hauptpaar *für $M$, falls $\Gamma \vdash M\!:\!\sigma$ und ferner:*
*falls $\Gamma \vdash M\!:\!\sigma$ und $\Gamma' \vdash M\!:\!\sigma'$, dann gibt es eine Substitution $s$, so daß $\sigma' = s(\sigma)$ und $\Gamma' \supseteq s(\Gamma)$.*

**Theorem 3.19**

*Es gibt einen Algorithmus, der zu jedem geschlossenen Term $M$ einen Haupttyp und zu jedem $M$ ein Hauptpaar liefert, falls $M$ überhaupt einen Typ hat, und ansonsten* `fail` *ausgibt.*

Beweis

Sei $\Gamma_0 =_{def} \{x_1\!:\!\alpha_1, \ldots, x_n\!:\!\alpha_n\}$ und $\sigma_0 =_{def} \beta$, wobei $x_1, \ldots, x_n$ die freien Variablen in $M$ sind. Jede *mgu*-Lösung von $E(\Gamma_0 \vdash M\!:\!\sigma_0)$ ist Hauptpaar von $M$, falls es eine Lösung gibt, sonst wird `fail` ausgegeben.

1. M hat Typ $\iff$ $\Gamma \vdash M\!:\!\sigma$ für gewisse $\Gamma, \sigma$
   $\iff$ $s(\Gamma_0) \vdash M\!:\!s(\sigma_0)$ für gewisses $s$
   $\iff$ $E(\Gamma_0 \vdash M\!:\!\sigma_0)$ ist lösbar (Lemma 3.17 (i))

2. Sei $s$ *mgu*-Lösung von $E(\Gamma_0 \vdash M : \sigma_0)$, dann ist $s(\Gamma_0) \vdash M : s(\sigma_0)$. Sei nun $\Gamma' \vdash M : \sigma'$. Dann ist $\tilde{\Gamma} \vdash M : \sigma'$, wobei $\tilde{\Gamma} =_{def} \Gamma'|_{FV(M)}$. Sei $s'$ so gewählt, daß $s'(\Gamma_0) = \tilde{\Gamma}$, sowie $s'(\sigma_0) = \sigma'$. Dann $s'(\Gamma_0) \vdash M : s'(\sigma_0)$. Dann gilt für ein $s''$, das die Typvariablen in $\Gamma_0$ und $\sigma_0$ wie $s'$ interpretiert: $s''$ ist Lösung von $E(\Gamma_0 \vdash M : \sigma_0)$ (Lemma 3.17 (ii)). Da $s$ *mgu* ist, gilt $s'' = S_1 \circ S$ für ein $s_1$, das heißt $\sigma' = s''(\sigma_0) = S_1(s(\sigma_0))$. Ferner haben wir, daß $\Gamma \supseteq \tilde{\Gamma} = s''(\Gamma_0) = S_1(s(\Gamma_0))$.
   $(s(\Gamma_0), s(\sigma_0))$ ist also Hauptpaar.

□

**Theorem 3.20**
*Typprüfung und Typisierbarkeit sind entscheidbar.*

Beweis

- Sei geschlossenes $M$ gegeben. Der Algorithmus aus Theorem 3.19 liefert einen Haupttyp $\sigma$, wenn $M$ typisierbar ist, sonst `fail`.
- Zur Prüfung von $\vdash M : \sigma'$ für gegebenes $\sigma'$ muß noch geprüft werden, ob $\sigma' = s(\sigma)$ für ein $s$. Das ist leicht algorithmisch möglich.

□

**Theorem 3.21**
*Das Problem, ob es zu einem gegebenen Typ $\sigma$ einen Term gibt, ist entscheidbar.*

Beweis
Es gibt $M$ mit $\vdash M : \sigma \iff$ Es gibt Beweis für $\sigma$ (als Formel) in der positiven Implikationslogik. Dies ergibt sich aus der im folgenden dargestellten sog. “Curry-Howard-Interpretation” (Theorem 3.26). Die positive Implikationslogik ist entscheidbar. □

### Der Curry-Howard-Isomorphismus

**Definition 3.22**
*Typvariablen heißen auch* Aussagevariablen, *Typen auch* (implikationslogische) Formeln.
*Eine endliche Menge von Formeln heißt* Kontext. *Metasprachliche Variablen für Kontexte sind* $\Delta, \Delta', \ldots$
*Die Axiome und Regeln der positiven Implikationslogik $P\!\rightarrow$ sind:*

- $(I)\ \ \Delta, \sigma \vdash \sigma$
- $(\rightarrow I)\ \ \dfrac{\Delta, \sigma \vdash \tau}{\Delta \vdash \sigma \rightarrow \tau}$
- $(\rightarrow E)\ \ \dfrac{\Delta \vdash \sigma \rightarrow \tau \quad \Delta \vdash \sigma}{\Delta \vdash \tau}$

$\Delta \vdash_{P\rightarrow} \sigma$ *bedeutet, daß* $\Delta \vdash \sigma$ *in* $P\!\rightarrow$ *herleitbar ist.*
*Für ein Urteil* $M\!:\!\sigma$ *sei* $(M\!:\!\sigma)^\circ \equiv \sigma$.
*Für eine Basis* $\Gamma = \{x_1\!:\!\sigma_1, \ldots, x_n\!:\!\sigma_n\}$ *sei* $\Gamma^\circ$ *der Kontext* $\{\sigma_1, \ldots, \sigma_n\}$.

**Lemma 3.23**
$\Gamma \vdash_{\lambda\rightarrow} M\!:\!\sigma \implies \Gamma^\circ \vdash_{P\rightarrow} \sigma$

Beweis
Lasse Terme weg. □

**Lemma 3.24**
*Es gibt einen Algorithmus, der zu jedem typbaren Term $M$ eine Ableitung von $\Delta \vdash \sigma$ in $P\!\rightarrow$ liefert derart, daß $\Delta = \Gamma^\circ$ und $\Gamma \vdash_{\lambda\rightarrow} M\!:\!\sigma$.*

Beweis
Theorem 3.19 □

D.h.: Jeder typbare Term $M$ kodiert eine Ableitung in $P\!\rightarrow$. Aus dieser Ableitung lassen sich durch Substitution alle Ableitungen von $\Gamma^\circ \vdash \sigma$ in $P\!\rightarrow$ gewinnen, die Ableitungen von $\Gamma \vdash M\!:\!\sigma$ in $\lambda\!\rightarrow$ entsprechen.

**Lemma 3.25**
*Zu jeder Ableitung von $\Delta \vdash \sigma$ in $P\!\rightarrow$ läßt sich ein Term $M$ und eine Ableitung von $\Gamma \vdash M\!:\!\sigma$ in $\lambda\!\rightarrow$ konstruieren mit $\Gamma^\circ = \Delta$.*

Beweis

Induktion über dem Aufbau der Ableitung von $\Delta \vdash \sigma$ in $P{\rightarrow}$.

- Alle in Anfangssequenzen vorkommenden Formeln $\sigma$ werden durch Typdeklarationen $x{:}\sigma$ ersetzt, wobei die Variable $x$ so gewählt ist, daß allen Vorkommen einer Formel $\sigma$ *dieselbe* Deklaration $x{:}\sigma$ und *verschiedenen* Formeln $\sigma$ und $\tau$ Deklarationen $x{:}\sigma$ und $y{:}\tau$ mit *verschiedenen* Variablen $x$ und $y$ entsprechen.

- Sei in $P{\rightarrow}$ der Schritt

$$(\rightarrow I)\ \frac{\sigma_1, \ldots \sigma_n, \sigma \vdash \tau}{\sigma_1, \ldots \sigma_n \vdash \sigma \rightarrow \tau}$$

  angewendet, wobei für $\sigma_1, \ldots \sigma_n, \sigma \vdash \tau$ schon eine Ableitung in $\lambda{\rightarrow}$ von

$$x_1{:}\sigma_1, \ldots x_n{:}\sigma_n, x{:}\sigma \vdash M{:}\tau$$

  vorliegt. Dann verlängern wir diese unter Verwendung von $(\rightarrow I)$ in $\lambda{\rightarrow}$ zu

$$x_1{:}\sigma_1, \ldots x_n{:}\sigma_n \vdash \lambda x.M{:}\sigma \rightarrow \tau.$$

- Sei in $P{\rightarrow}$ der Schritt

$$(\rightarrow E)\ \frac{\sigma_1, \ldots \sigma_n, \sigma \vdash \sigma \rightarrow \tau \qquad \sigma_1, \ldots \sigma_n \vdash \sigma}{\sigma_1, \ldots \sigma_n \vdash \tau}$$

  angewendet, wobei in $\lambda{\rightarrow}$ für $\sigma_1, \ldots \sigma_n \vdash \sigma \rightarrow \tau$ und $\sigma_1, \ldots \sigma_n \vdash \sigma$ schon Ableitungen von

$$x_1{:}\sigma_1, \ldots x_n{:}\sigma_n \vdash M{:}\sigma \rightarrow \tau$$

  und

$$x_1{:}\sigma_1, \ldots x_n{:}\sigma_n \vdash N{:}\sigma$$

  vorliegen. Man beachte, daß einem Typ $\sigma_i$ genau eine Variable $x_i$ zugeordnet ist. Damit erhalten wir mit $(\rightarrow E)$ eine Ableitung in $\lambda{\rightarrow}$ von

$$x_1{:}\sigma_1, \ldots x_n{:}\sigma_n \vdash MN{:}\tau.$$

$\square$

**Theorem 3.26 (Curry-Howard-Isomorphismus)**
*Sei* $M_P$ *die gemäß Lemma 3.24 zu einem in* $\lambda\!\!\rightarrow$ *typbaren Term* $M$ *gehörende Ableitung in* $P\!\!\rightarrow$*. Sei* $\Pi_\lambda$ *der gemäß Lemma 3.25 zu einer Ableitung in* $P\!\!\rightarrow$ *gehörendeTerm von* $\lambda\!\!\rightarrow$*. Dann gilt:*

*(i)* $(\Pi_\lambda)_P$ *ist eine Ableitung in* $P\!\!\rightarrow$*, aus der sich* $\Pi$ *durch Substitution von Formeln für Aussagevariablen gewinnen läßt.*

*(ii)* $(M_P)_\lambda$ *ist (bis auf Umbenennung von freien und/oder gebundenen Variablen) ein Term, der aus* $M$ *durch Identifizierung von freien oder von gebundenen Variablen entsteht.*

Beweis
Aus den Lemmas 3.24 und 3.25. □

**Bemerkung**
Ein Beispiel für *(ii)* ist der $\lambda$-Term $u(zx)(zy)$, der nach dem Typisierungsalgorithmus durch

$$u\!:\!\alpha\to\alpha\to\beta, z\!:\!\gamma\to\beta, x\!:\!\gamma, y\!:\!\gamma \vdash_{\lambda\to} u(zx)(zy)\!:\!\beta$$

getypt wird. Die korrespondierende Ableitung in $P\!\!\rightarrow$ liefert

$$\alpha\to\alpha\to\beta, \gamma\to\beta, \gamma \vdash \beta.$$

Dieser Ableitung entspricht gemäß Lemma 3.25 ein $\lambda$-Term der Form $u(zx)(zx)$, in dem $x$ und $y$ identifiziert sind. Zur Identifizierung der Variablen kommt es dadurch, daß beim Übergang von $\lambda\!\!\rightarrow$ zu $P\!\!\rightarrow$ Information verlorengeht, die beim Übergang von $P\!\!\rightarrow$ zu $\lambda\!\!\rightarrow$ nicht mehr rekonstruiert werden kann. Das hängt damit zusammen, daß bei der Zuordnung von Variablen zu Formeln (= Typen) in Anfangssequenzen von $\lambda\!\!\rightarrow$ gemäß Lemma 3.25 keine zwei Variablen denselben Typ haben können, d.h. beim Übergang von $\lambda\!\!\rightarrow$ zu $P\!\!\rightarrow$ keine Sequenz der Form $\Gamma, x\!:\!\sigma, y\!:\!\sigma \vdash M\!:\!\tau$ auftreten kann.

Eine andere Fassung des aussagenlogischen Kalküls erlaubt es, ohne diesen Informationsverlust auszukommen. Dazu wird auf die Beweistheorie verwiesen (siehe Trolstra/Schwichtenberg 1996).

Der Curry-Howard-Isomorphismus induziert Reduktions- und Gleichheitsrelationen für Herleitungen, die $\triangleright_\beta$ und $=_\beta$ entsprechen. Diese behandelt man ebenfalls in der Beweistheorie, genauer in der Theorie des natürlichen Schließens.

# 4 Der polymorph getypte $\lambda$-Kalkül

Der polymorph getypte $\lambda$-Kalkül heißt auch *Getypter $\lambda$-Kalkül 2. Stufe* oder *System F*.
Die Motivation: es soll zum Beispiel die Identitätsfunktion $id \equiv \lambda x.x : \alpha \to \alpha$ unabhängig von einem speziellen Typ $\alpha$ sein. Man will also, daß $id \equiv \lambda x.x : \forall\alpha.(\alpha \to \alpha)$

**Definition 4.1**
*Die Menge der Typen des polymorph getypten $\lambda$-Kalküls ($\lambda 2$) ist wie folgt gegeben:*

- *Jede Typvariable ist ein Typ*
- *Mit $\sigma$ und $\tau$ ist auch $(\sigma \to \tau)$ ein Typ*
- *Mit $\alpha$ und $\sigma$ ist auch $\forall\alpha.\sigma$ ein Typ*

*Konvention: $\forall$ bindet stärker als $\to$.*
*$\sigma[\tau/\alpha]$ ist nur erlaubt, falls $\tau$ frei für $\alpha$ in $\sigma$ ist.*

**Definition 4.2**
*Die Typisierung in $\lambda 2$ ist durch folgende Regeln definiert:*

- $(I)$ $\Gamma, x : \sigma \vdash x : \sigma$
- $(\to I)$ $\dfrac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash (\lambda x.M) : \sigma \to \tau}$
- $(\to E)$ $\dfrac{\Gamma \vdash M : \sigma \to \tau \quad \Gamma \vdash N : \sigma}{\Gamma \vdash MN : \tau}$
- $(\forall I)$ $\dfrac{\Gamma \vdash M : \sigma}{\Gamma \vdash M : \forall\alpha.\sigma}$ *falls* $\alpha \notin FV(\Gamma)$
- $(\forall E)$ $\dfrac{\Gamma \vdash M : \forall\alpha.\sigma}{\Gamma \vdash M : \sigma[\tau/\alpha]}$

**Beispiel 4.3**

- $\lambda x.x : \forall\alpha.\alpha \to \alpha$

$$\dfrac{\dfrac{\dfrac{}{x : \sigma \vdash x : \sigma}}{\vdash \lambda x.x : \alpha \to \alpha}}{\vdash \lambda x.x : \forall\alpha.\alpha \to \alpha}$$

- $\lambda xy.y : \forall\alpha\beta.\alpha \to (\beta \to \beta)$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x:\alpha, y:\beta \vdash y:\beta}}{x:\alpha \vdash \lambda y.y : \beta \to \beta}}{\vdash \lambda xy.y : \alpha \to (\beta \to \beta)}}{\vdash \lambda xy.y : \forall\beta.\alpha \to (\beta \to \beta)}}{\vdash \lambda xy.y : \forall\alpha\beta.\alpha \to (\beta \to \beta)}$$

- $\lambda x.xx : \forall\beta.(\forall\alpha.\alpha) \to \beta$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x:\forall\alpha.\alpha \vdash x:\forall\alpha.\alpha}}{x:\forall\alpha.\alpha \vdash x:\alpha \to \beta} \quad \dfrac{\dfrac{}{x:\forall\alpha.\alpha \vdash x:\forall\alpha.\alpha}}{x:\forall\alpha.\alpha \vdash x:\alpha}}{x:\forall\alpha.\alpha \vdash xx:\beta}}{\vdash \lambda x.xx : (\forall\alpha.\alpha) \to \beta}}{\vdash \lambda x.xx : \forall\beta.(\forall\alpha.\alpha) \to \beta}$$

**Definition 4.4**

*Es sei $\sigma \sqsupset \tau$, falls*

- $\tau \equiv \forall\alpha.\sigma$ *($\tau$ ist* Generalisierung *von $\sigma$)*
- $\sigma \equiv \forall\alpha.\sigma_1$ *und* $\tau \equiv \sigma_1[\sigma_2/\alpha]$ *($\tau$ ist* Spezialisierung *von $\sigma$)*

*Es sei $\sigma \sqsupseteq \tau$ genau dann, wenn es $n \geq 0$ gibt, so daß $\sigma \equiv \sigma_1 \sqsupset \cdots \sqsupset \sigma_n \equiv \tau$.*
*Es sei $\sigma^0$ der Typ $\sigma$ ohne Quantorenpräfix (d.h. führende Quantoren sind weggelassen).*

**Beispiel 4.5**

- $(\forall\alpha_1 \ldots \alpha_n.\sigma)^0 \equiv \sigma$
- $(\forall\alpha.(\forall\beta.\beta) \to \alpha)^0 \equiv (\forall\beta.\beta) \to \alpha$

**Bemerkung**

- $\sqsupset$ ist nicht symmetrisch:
  Zwar gilt, daß $\sigma \sqsupset \forall\alpha.\sigma \implies \forall\alpha.\sigma \sqsupset \sigma$,
  aber $\forall\alpha.\sigma_1 \sqsupset \sigma_1[\sigma_2/\alpha] \not\Longrightarrow \sigma_1[\sigma_2/\alpha] \sqsupset \forall\alpha.\sigma_1$.
- Intuitiv bedeutet $\sigma \sqsupseteq \tau$, daß $M:\tau$ aus $M:\sigma$ alleine durch Anwendung von $\forall$-Regeln herleitbar ist, also

$$\left.\begin{array}{c} \Gamma \vdash M:\sigma \\ \vdots \\ \Gamma \vdash M:\tau \end{array}\right\} \text{nur } \forall\text{-Regeln}$$

**Lemma 4.6**

*Sei $\Gamma$ gegeben. Dann ist $\sigma \sqsupseteq \tau$ mit $\sigma \equiv \sigma_1 \sqsupset \cdots \sqsupset \sigma_n \equiv \tau$, wobei keine in $\Gamma$ frei vorkommende Typvariable bei einem Übergang von $\sigma_i$ nach $\sigma_{i+1}$ generalisiert wird, genau dann, wenn*

$$\left.\begin{array}{c}\Gamma \vdash M:\sigma \\ \vdots \\ \Gamma \vdash M:\tau\end{array}\right\} \textit{nur } \forall\textit{-Regeln}$$

Beweis

Definition von $\sqsupseteq$. □

**Lemma 4.7**

- $\Gamma \vdash x:\sigma \implies (\exists \sigma' \sqsupseteq \sigma)\; x:\sigma' \in \Gamma$
- $\Gamma \vdash MN:\tau \implies (\exists \sigma)\; (\exists \tau' \sqsupseteq \tau)\; [\,\Gamma \vdash M:\sigma \to \tau'$ und $\Gamma \vdash N:\sigma]$
- $\Gamma \vdash \lambda x.M:\rho \implies (\exists \sigma)\; (\exists \tau)\; [\,\Gamma, x:\sigma \vdash M:\tau$ und $(\sigma \to \tau) \sqsupseteq \rho]$

**Bemerkung**

Alle anderen Aussagen von Lemma 3.5 außer der Subjektreduktion gelten offensichtlich in $\lambda 2$. Die Subjektreduktion wird im folgenden gesondert bewiesen.

**Lemma 4.8**

*Wenn $(\sigma \to \tau) \sqsupseteq (\sigma' \to \tau')$, dann ist $(\sigma' \to \tau') \equiv s(\sigma \to \tau)$ für eine Substitution $s$, d.h. $\sigma' \to \tau'$ ist spezieller als $\sigma \to \tau$.*

Beweis

Sei $(\sigma \to \tau) \equiv \sigma_1 \sqsupset \cdots \sqsupset \sigma_n \equiv (\sigma' \to \tau')$.
Wir zeigen: $\sigma_i^0 \equiv s_i(\sigma \to \tau)$ für jedes $s_i$ $(1 \leq i \leq n)$.
Damit folgt die Behauptung, da $(\sigma' \to \tau')^0 \equiv (\sigma' \to \tau')$.
Beweis durch Induktion über $n$:

- n=0: $\surd$
- n=m+1: Sei $\sigma_m^0 \equiv s_m(\sigma \to \tau)$.
  - Sei $\sigma_{m+1} \equiv \forall \alpha.\sigma_m$, dann ist $\sigma_{m+1}^0 = \sigma_m^0$, also $s_{m+1} =_{def} s_m$.
  - Sei $\sigma_m \equiv \forall \alpha.\rho$ und $\sigma_{m+1} \equiv \rho[\rho_1/\alpha]$. Dann setze $s_{m+1} =_{def} s_m[\rho_1/\alpha]$, wobei $s_m[\rho_1/\alpha]$ wie $s_m$ ist, modifiziert um $\alpha \mapsto \rho_1$.

$\square$

**Theorem 4.9 (Subjektreduktion)**
*$M \rhd_\beta M' \implies (\Gamma \vdash M:\sigma \implies \Gamma \vdash M':\sigma)$*

Beweis
Wir betrachten den Fall $M \equiv (\lambda x.P)Q \rhd_{1\beta} P[{}^{Q}\!/\!_{x}] \equiv M'$. Daraus ergibt sich der Rest.

$$\begin{array}{ll}
\Gamma \vdash (\lambda x.P)Q:\sigma & \\
\overset{(4.7)}{\Longrightarrow} & (\exists\tau)\,(\exists\sigma' \sqsupseteq \sigma)\,[\,\Gamma \vdash \lambda x.P:\tau\to\sigma' \text{ und } \Gamma \vdash Q:\tau] \\
\overset{(4.7)}{\Longrightarrow} & (\exists\tau)\,(\exists\sigma' \sqsupseteq \sigma)\,(\exists\tau')\,(\exists\sigma'')\,[\,\Gamma, x:\tau' \vdash P:\sigma'' \text{ und } \Gamma \vdash Q:\tau \text{ und } (\tau'\to\sigma'') \sqsupseteq (\tau\to\sigma')] \\
\overset{(4.8)}{\Longrightarrow} & (\exists\tau)\,(\exists\sigma' \sqsupseteq \sigma)\,[\,\underbrace{\Gamma, x:\tau \vdash P:\sigma'}_{\tau,\sigma' \text{ spezieller als } \tau',\sigma''} \text{ und } \Gamma \vdash Q:\tau] \\
\Longrightarrow & (\exists\sigma' \sqsupseteq \sigma)\ \Gamma \vdash P[{}^{Q}\!/\!_{x}]:\sigma' \\
\overset{(4.6)}{\Longrightarrow} & \Gamma \vdash P[{}^{Q}\!/\!_{x}]:\sigma
\end{array}$$

$\square$

**Definition 4.10**
*Eine* Bewertung *in $\mathcal{SAT}$ ist eine Abbildung $\nu$:Typvariablen$\to\mathcal{SAT}$.*
*Wir definieren eine Semantik von Typen relativ zu Bewertungen:*

- $[\![\alpha]\!]_\nu =_{def} \nu(\alpha)$
- $[\![\sigma\to\tau]\!]_\nu =_{def} [\![\sigma]\!]_\nu \to [\![\tau]\!]_\nu$
- $[\![\forall\alpha.\sigma]\!]_\nu =_{def} \bigcap_{A\in\mathcal{SAT}} [\![\sigma]\!]_{\nu[{}^{A}\!/\!_{\alpha}]}$

**Lemma 4.11**
*Für jeden Typ $\sigma$ und jede Bewertung $\nu$ gilt: $[\![\sigma]\!]_\nu$ ist saturiert.*
*(vgl. Lemma 3.8)*

Beweis
Analog zum Beweis von Lemma 3.8. Es wäre noch zu zeigen, daß $\mathcal{SAT}$ unter Durchschnitt abgeschlossen ist. Das ist aber trivial. $\square$

**Definition 4.12**
$\rho,\nu \models M:\sigma \iff_{def} [\![M]\!]_\rho \in [\![\sigma]\!]_\nu$
$\rho,\nu \models \Gamma \iff_{def} \rho,\nu \models x:\sigma$ *für alle* $(x:\sigma)\in\Gamma$
$\Gamma \models M:\sigma \iff_{def} (\forall\rho)\,(\forall\nu)\,[\rho,\nu \models \Gamma \Longrightarrow \rho,\nu \models M:\sigma]$

**Lemma 4.13**
*$\Gamma \vdash M:\sigma \implies \Gamma \models M:\sigma$*

**Theorem 4.14**
$\Gamma \vdash M : \sigma \implies M$ *ist stark normalisierbar.*

Beweis
Da $[\![\sigma]\!]_\nu$ für jedes $\nu$ saturiert, gilt $id, \nu \models \Gamma$ für jedes $\nu$. Also $id, \nu \models M : \sigma$, dh. $M \in [\![\sigma]\!]_\nu$. □

**Bemerkung**

- $\lambda 2$ hat mehr typbare Terme als $\lambda\!\rightarrow$. $\lambda x.xx$ ist ein Beispiel für einen stark normalisierbaren Term, der nicht in $\lambda\!\rightarrow$, aber in $\lambda 2$ typbar ist.
- Frage: Ist Typbarkeit in $\lambda 2$ = Normalisierbarkeit?
  Nein. Jedoch kann jeder Term in Normalform in $\lambda 2$ getypt werden, dh.
  $x_1 : \forall\alpha.\alpha, \ldots, x_n : \forall\alpha.\alpha \vdash M : \sigma$ für ein $\sigma$, wobei $x_1, \ldots, x_n$ freie Variablen in $M$ sind.
- Es ist aber starke Normalisierbarkeit = Typbarkeit in $\lambda\cap$ (System D)