Übung zur Vorlesung

Automatisierte Logik und Programmierung

Prof. Chr. Kreitz

Universität Potsdam, Theoretische Informatik — Wintersemester 2016/17

Blatt 3 — Abgabetermin: 24.11.2016

Das dritte Übungsblatt soll dazu dienen, Erfahrungen mit dem λ–Kalkül zu sammeln.
Wir werden mögliche Lösungen zu Beginn der Veranstaltung am 24.11.2016 besprechen

---

Aufgabe 3.1 (Gleichheit)

Beweisen Sie folgende Formeln mithilfe der Refinement Logik einschließlich der Gleichheitsregeln:

3.1–a $(\forall n)(\forall m)(m{=}n \Rightarrow s(n){=}s(m))$

3.1–b $(g(g(g(a))){=}a \wedge g(g(g(g(a)))){=}a) \Rightarrow g(a){=}a$

3.1–c $(plus(n,0){=}n \wedge (\forall n)(\forall m)(plus(n,s(m)){=}s(plus(n,m)))) \Rightarrow plus(s(0),s(s(0))){=}s(s(s(0)))$

Aufgabe 3.2 (λ–Kalkül: Boolesche Algebra)

Definieren Sie mithilfe von T, F und $\mathsf{if}\ b\ \mathsf{then}\ s\ \mathsf{else}\ t$ die folgenden Booleschen Operatoren:

3.2–a "and", die logische Konjunktion

3.2–b "or", die logische Disjunktion

3.2–c "neg", die logische Negation

3.2–d "imp", die logische Implikation

und geben Sie diese anschließend als reine λ-Terme (ohne Verwendung abkürzender Definitionen) an. Überlegen Sie sich, wie man die Korrektheit Ihrer Definitionen formal nachweisen könnte und skizzieren Sie für jeden Ihrer Operatoren einen Korrektheitsbeweis.

Aufgabe 3.3 (λ–Kalkül: Ganzzahlfunktionen)

3.3–a Geben Sie einen λ-Term subtract an mit der Eigenschaft $\mathsf{subtract}\ \overline{m}\ \overline{n} = \overline{m-n}$

3.3–b Beschreiben Sie einen λ-Term less_or_equal mit $\mathsf{less\_or\_equal}\ \overline{m}\ \overline{n} = \begin{cases} \mathsf{T} & \text{falls } m{\leq}n \\ \mathsf{F} & \text{sonst} \end{cases}$

3.3–c Geben Sie einen λ-Term max an mit der Eigenschaft $\mathsf{max}\ \overline{m}\ \overline{n} = \begin{cases} \overline{n} & \text{falls } m{\leq}n \\ \overline{m} & \text{sonst} \end{cases}$

Skizzieren Sie jeweils einen Korrektheitsbeweis.

Aufgabe 3.4 (λ–Kalkül: Korrektheit von Repräsentationen)

Die Darstellung der natürlichen Zahlen wurde mit Hilfe der *Church Numerals* ($\overline{n} \equiv \lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^n\ \mathtt{x}$) beschrieben. Beweisen Sie die Korrektheit der folgenden Operationen

3.4–a $\mathsf{PRs}[b,h] \equiv \lambda n.n\ h\ b$

Zeigen Sie, daß für $f \equiv \mathsf{PRs}[b,h]$ gilt: $f\ \overline{0} = b$ und $f\ (s\ n) = h\ (f\ n)$

3.4–b $\mathsf{p} \equiv \lambda \mathtt{n}.\ (\mathtt{n}\ (\lambda \mathtt{f\_x}.\ \langle \mathtt{s}, \mathtt{match\ f\_x\ with}\ \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f\ x}\rangle)\ \langle \lambda \mathtt{z}.\overline{0},\ \overline{0}\rangle).2$ (sehr aufwendig)

Zeigen Sie, daß p die Vorgängerfunktion repräsentiert, d.h. für alle n gilt $\mathsf{p}\ \overline{n} = \overline{n{-}1}$.

Lösung 3.1

Ziel dieser Aufgabe ist es, ein gewisses Gefühl für den Umgang mit Gleichheit zu bekommen.

3.1–a $(\forall n)(\forall m)(m{=}n \Rightarrow s(n){=}s(m))$:

```
⊢ ∀n,m:𝕌.  (m=n ⇒ s(n)=s(m))                          BY allR THEN allR
1.  n:𝕌, m:𝕌 ⊢ m=n ⇒ s(n)=s(m)                        BY impliesR
1.1.  n:𝕌, m:𝕌, m=n ⊢ s(n)=s(m)                      BY applyEq
1.1.1.  n:𝕌, m:𝕌, m=n ⊢ s=s                          BY reflexivity
1.1.2.  n:𝕌, m:𝕌, m=n ⊢ n=m                          BY symmetry
1.1.2.1.  n:𝕌, m:𝕌, m=n ⊢ m-n                        BY axiom
```

3.1–b $(\forall a)\ ((g(g(g(a)))=a \wedge g(g(g(g(g(a)))))=a) \Rightarrow g(a)=a)$

```
⊢ (∀a)  ((g(g(g(a)))=a ∧ g(g(g(g(g(a)))))=a) ⇒ g(a)=a)                BY allR THEN impliesR
1.  a:𝕌, g(g(g(a)))=a ∧ g(g(g(g(g(a)))))=a ⊢ g(a)=a                    BY andL
1.1.  a:𝕌, g(g(g(a)))=a, g(g(g(g(g(a)))))=a ⊢ g(a)=a                  BY transitivity g(g(g(g(g(g(a))))))
1.1.1.  a:𝕌, g(g(g(a)))=a, g(g(g(g(g(a)))))=a ⊢ g(a)=g(g(g(g(g(g(a)))))) BY subst g(g(g(g(g(a)))))=a
1.1.1.1 a:𝕌, g(g(g(a)))=a, g(g(g(g(g(a)))))=a ⊢ g(g(g(g(g(a)))))=a    BY axiom
1.1.1.2 a:𝕌, g(g(g(a)))=a, g(g(g(g(g(a)))))=a ⊢ g(a)=g(a)             BY reflexivity
1.1.2.  a:𝕌, g(g(g(a)))=a, g(g(g(g(g(a)))))=a ⊢ g(g(g(g(g(g(a)))))=a  BY subst g(g(g(a)))=a
1.1.2.1.  a:𝕌, g(g(g(a)))=a, g(g(g(g(g(a)))))=a ⊢ g(g(g(a)))=a        BY axiom
1.1.2.2.  a:𝕌, g(g(g(a)))=a, g(g(g(g(g(a)))))=a ⊢ g(g(g(a)))=a        BY axiom
```

3.1–c $(plus(n,0){=}n \wedge (\forall n)(\forall m)(plus(n,s(m)){=}s(plus(n,m)))) \Rightarrow plus(s(0),s(s(0))){=}s(s(s(0)))$

```
⊢ (∀n)(plus(n,0)=n) ∧ (∀n)(∀m)(plus(n,s(m))=s(plus(n,m))) ⇒ plus( s(0),s(s(0)) ) = s(s(s(0)))
                                                          BY impliesR THEN andL
1.  (∀n)(plus(n,0)=n), (∀n)(∀m)(plus(n,s(m))=s(plus(n,m))) ⊢ plus(s(0),s(s(0))) = s(s(s(0)))
                                                          BY subst plus(s(0),s(s(0)))=s(plus(s(0),s(0)))
1.1.  (∀n)(plus(n,0)=n), (∀n)(∀m)(plus(n,s(m))=s(plus(n,m))) ⊢ plus(s(0),s(s(0)))=s(plus(s(0),s(0)))
                                                          BY allL 2 's(0)' THEN allL 3 's(0)' THEN axiom
1.2.  (∀n)(plus(n,0)=n), (∀n)(∀m)(plus(n,s(m))=s(plus(n,m))) ⊢ s(plus(s(0),s(0))) = s(s(s(0)))
                                                          BY subst plus(s(0),s(0))=s(plus(s(0),0))
1.2.1 (∀n)(plus(n,0)=n), (∀n)(∀m)(plus(n,s(m))=s(plus(n,m))) ⊢ plus(s(0),s(0))=s(plus(s(0),0))
                                                          BY allL 2 's(0)' THEN allL 3 '0' THEN axiom
1.2.2.  (∀n)(plus(n,0)=n), (∀n)(∀m)(plus(n,s(m))=s(plus(n,m))) ⊢ s(s(plus(s(0),0))) = s(s(s(0)))
                                                          BY subst plus(s(0),0)=s(0)
1.2.2.1 (∀n)(plus(n,0)=n), (∀n)(∀m)(plus(n,s(m))=s(plus(n,m))) ⊢ plus(s(0),0)=s(0)
                                                          BY allL 1 's(0)' THEN axiom
1.2.2.2.  (∀n)(plus(n,0)=n), (∀n)(∀m)(plus(n,s(m))=s(plus(n,m))) ⊢ s(s(s(0))) = s(s(s(0)))
                                                          BY reflexivity
```

Lösung 3.2

Da es sich um boolesche Operatoren handelt, kann die Lösung entweder durch Betrachtung der Wahrheitstafeln oder durch eine Analyse der Bedeutung der Operatoren gefunden werden.

a	b	$a \wedge b$	$a \vee b$	$\neg a$	$a \Rightarrow b$
T	T	T	T	F	T
T	F	F	T	F	F
F	T	F	T	T	T
F	F	F	F	T	T

3.2–a Definition von "and" (logische Konjunktion):

Eine kurze Überlegung führt zu folgender Fallunterscheidung:

1. Ist A wahr, so ist $A \wedge B$ wahr, genau dann wenn B wahr ist.
2. Ist A falsch, so ist $A \wedge B$ sowieso falsch.

Damit können wir and wie folgt definieren:

$$\begin{aligned} \mathsf{and} &\equiv \lambda a.\lambda b.\ \ \mathsf{if\ a\ then\ b\ else\ F} \\ &\equiv \lambda a.\lambda b.\ a\ b\ \mathsf{F} \\ &\equiv \lambda a.\lambda b.\ a\ b\ (\lambda x.\lambda y.y) \end{aligned}$$

Um die Korrektheit dieser Definition zu beweisen, müssen wir das Ergebnis der Berechnung von $a \wedge b$ für die vier möglichen Kombinationen boolescher Eingaben berechnen und mit den erwarteten Werten in der Wahrheitstafel vergleichen.

Dies kann durch Reduktion im λ-Kalkül geschehen, also z.B. durch die Berechnung von $\mathsf{T} \wedge \mathsf{T}$ (hierbei müsste T herauskommen), oder durch den Nachweis von $\mathsf{T} \wedge \mathsf{T} = \mathsf{T}$ im Refinementkalkül für λ-Ausdrücke. Wir führen den Beweis durch Reduktion

$$\mathsf{T} \wedge \mathsf{T} \xrightarrow{2} \mathsf{if\ T\ then\ T\ else\ F} \equiv \mathsf{T\ T\ F} \equiv (\lambda x.\lambda y.x)\ \mathsf{T\ F} \xrightarrow{2} \mathsf{T}$$

$$\mathsf{T} \wedge \mathsf{F} \xrightarrow{2} \mathsf{if\ T\ then\ F\ else\ F} \equiv \mathsf{T\ F\ F} \equiv (\lambda x.\lambda y.x)\ \mathsf{F\ F} \xrightarrow{2} \mathsf{F}$$

$$\mathsf{F} \wedge \mathsf{T} \xrightarrow{2} \mathsf{if\ F\ then\ T\ else\ F} \equiv \mathsf{F\ T\ F} \equiv (\lambda x.\lambda y.y)\ \mathsf{T\ F} \xrightarrow{2} \mathsf{F}$$

$$\mathsf{F} \wedge \mathsf{F} \xrightarrow{2} \mathsf{if\ F\ then\ F\ else\ F} \equiv \mathsf{F\ F\ F} \equiv (\lambda x.\lambda y.y)\ \mathsf{F\ F} \xrightarrow{2} \mathsf{F}$$

Die Beweise im Beweiskalkül sind analog (vgl. Folie 18 aus EInheit 5)

3.2–b "or", die logische Disjunktion:

Analog zu den obigen Überlegungen kommen wir zu folgender Definition für or:

$$\begin{aligned} \mathsf{or} &\equiv \lambda a.\lambda b.\mathsf{if\ a\ then\ T\ else\ b} \\ &\equiv \lambda a.\lambda b.\ a\ \mathsf{T}\ b \\ &\equiv \lambda a.\lambda b.\ a\ (\lambda x.\lambda y.x)\ b \end{aligned}$$

Die Beweise sind analog zu denen in Teil a)

3.2–c "neg", die logische Negation:

$$\begin{aligned} \mathsf{neg} &\equiv \lambda a.\mathsf{if\ a\ then\ F\ else\ T} \\ &\equiv \lambda a.\ a\ (\lambda x.\lambda y.y)\ (\lambda x.\lambda y.x) \end{aligned}$$

3.2–d "imp", die logische Implikation:

$$\begin{aligned} \mathsf{imp} &\equiv \lambda a.\lambda b.\mathsf{if\ a\ then\ b\ else\ T} \\ &\equiv \lambda a.\lambda b.\ a\ b\ (\lambda x.\lambda y.x) \end{aligned}$$

Lösung 3.3

3.3–a subtract: Definitionsgemäß unterscheiden wir zwei Fälle für den zweiten Eingabeparameter $\overline{y}$: $y = 0$ oder $y > 0$. Im ersten Fall geben wir den ersten Eingabeparameter ($\overline{x}$) zurück, im zweiten gehen wir davon aus, daß wir für den Vorgänger von $\overline{y}$ das Ergebnis kennen und wenden auf dieses die Vorgängerfunktion an. Dies klingt nun verdächtig nach einem Fall für den einfachen Rekursionsoperator "PRs[$base$,h]":

Für "$base$" muß somit der erste Parameter "$\overline{x}$" herhalten und für "h" demzufolge "p". Das Ganze sieht dann so aus:

$$\mathsf{subtract} \equiv \lambda \mathtt{x}.\,\lambda \mathtt{y}.\,\mathsf{PRs}[\mathtt{x},\mathsf{p}]\ \ \mathtt{y} \equiv \lambda x.\lambda y.(\lambda n.n\ \mathsf{p}\ x)y \xrightarrow{*} \lambda x.\lambda y.y\ \mathsf{p}\ x$$

Die folgende alternative Btrachtung führt direkt zu dem gleichen, reduzierten, Term:
Subtraktion von n ist n-fache Anwendung der Vorgängerfunktion p. *Da das Church Numeral $\overline{n}$ als die n-fache Anwendung einer Funktion auf ein Argument codiert ist, kann der Term direkt angegeben werden als* sub $\equiv \lambda m.\lambda n.\ n\ \mathsf{p}\ m$ *(Achtung: implizite Linksklammerung).*

Kontrolle: sub $\overline{m}\ \overline{n}$ $\equiv (\lambda m.\lambda n.\ n\ \mathsf{p}\ m)\ \overline{m}\ \overline{n}$ $\longrightarrow \overline{n}\ \mathsf{p}\ \overline{m}$ $\equiv (\lambda f.\lambda x.f^n\ x)\ \mathsf{p}\ \overline{m}$ $\longrightarrow \mathsf{p}^n\ \overline{m}$ $\xrightarrow{*} \overline{m-n}$. *Letzteres ist der Fall weil* p $\overline{m} = \overline{m-1}$ *gilt.*

3.3–b less_or_equal: Als gestandener Informatiker sollte man sich klar machen können, daß $m \leq n$ äquivalent zu $m - n \leq 0$ (bzw. $m \dot{-} n = 0$ für nicht–negative Zahlen) ist. Damit kann man die Definition für less_or_equal auch so hinschreiben:

$$\mathsf{less\_or\_equal}\ \overline{m}\ \overline{n} \equiv \begin{cases} \mathsf{T}, & \text{falls } m \dot{-} n = 0 \\ \mathsf{F}, & \text{sonst} \end{cases}$$

An dieser Stelle sollte es beim geneigten Leser bereits klingeln: das hierfür notwendige Werkzeug haben wir nämlich längst zur Hand. Es ist dies die "zero"–Funktion in Kombination mit dem eben definierten "subtract". Damit gelangen wir also zu folgendem Resultat:

$$\mathsf{less\_or\_equal} \equiv \lambda \mathtt{m}.\,\lambda \mathtt{n}.\mathsf{zero}\ \ (\mathsf{subtract}\ \ \mathtt{m}\ \ \mathtt{n})$$

Hier darf wiederum den kompletten λ–Term aufschreiben, wem's Freude bereitet...

3.3–c max: Hier suggeriert die Definition die Anwendung eines Conditionals gemäß folgender Formulierung:

"**Wenn** $m \leq n$ gilt, **dann** ist max $\overline{m}\ \overline{n}$ gleich $\overline{n}$, **ansonsten** ist max $\overline{m}\ \overline{n}$ gleich $\overline{m}$."

Damit gelangen wir zu folgendem Term:

$$\begin{aligned} \mathsf{max} &\equiv \lambda \mathtt{m}.\,\lambda \mathtt{n}.\mathsf{if}\ \mathsf{less\_or\_equal}\ \ \mathtt{m}\ \ \mathtt{n}\ \ \mathsf{then}\ \mathtt{n}\ \ \mathsf{else}\ \mathtt{m} \\ &\equiv \lambda \mathtt{m}.\,\lambda \mathtt{n}.\mathsf{cond}(\mathsf{less\_or\_equal}\ \ \mathtt{m}\ \ \mathtt{n};\mathtt{n};\mathtt{m}) \\ &\equiv \lambda \mathtt{m}.\,\lambda \mathtt{n}.(\mathsf{less\_or\_equal}\ \ \mathtt{m}\ \ \mathtt{n})\ \ \mathtt{n}\ \ \mathtt{m} \end{aligned}$$

Insgesamt sollte man hierbei erkannt haben, wie wichtig die definitorische Erweiterung ist, wenn man vernünftig mit dem λ–Kalkül arbeiten will. Wer das (immer noch) nicht glaubt, der kann ja mal versuchen, dieselbe Herleitung mit den »nackten« λ–Termen durchzuexerzieren...

Lösung 3.4

3.4–a Rekursionsgleichungen für $f \equiv \mathsf{PRs}[b,h]$

$\vdash f\ \overline{0} = b$	BY unfold
$\vdash (\lambda \mathtt{n}.\ \ \mathtt{n}\ h\ b)\ \overline{0} = b$	BY reduce
$\vdash \overline{0}\ h\ b = b$	BY unfold
$\vdash (\lambda \mathtt{f}.\lambda \mathtt{x}.\mathtt{x})\ h\ b = b$	BY reduce THEN reduce THEN reflexivity

$\vdash f\ (s\ n) = h\ (f\ n)$	BY unfold THEN reduce
$\vdash (s\ n)\ h\ b = h\ (f\ n)$	BY unfold
$\vdash (\lambda \mathtt{f}.\lambda \mathtt{x}.\mathtt{f}\ (n\ \mathtt{f}\ \mathtt{x}))\ h\ b = h\ (f\ n)$	BY reduce THEN reduce
$\vdash h\ (n\ h\ b) = h\ (f\ n)$	BY symmetry THEN unfold
$\vdash h\ ((\lambda \mathtt{n}.\ \ \mathtt{n}\ h\ b)\ n) = h\ (n\ h\ b)$	BY reduce THEN reflexivity

3.4–b $\mathsf{p}\,\overline{n} = \overline{m} \quad \equiv \quad \overline{n} = \mathsf{s}\,\overline{m}$:

Dieser Beweis ist ein wenig umfangreicher. Er wird in zwei Teile aufgespaltet:

1. Der Hauptbeweis
2. Ein Lemma über die Reduktion eines Teilredex'

Wir überlegen zunächst, wie die Reduktion von $\mathsf{p}\,\overline{n}$ unabhägig von einem konkreten n prinzipiell aussieht.

Vorüberlegung:

$$
\begin{aligned}
\mathsf{p}\ \overline{n} \quad &\equiv \quad (\lambda \mathtt{n}.\ (\mathtt{n}\ (\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)\ \langle \lambda \mathtt{z}.\overline{0},\ \overline{0}\rangle).2)\ \overline{n} \\
&\longrightarrow \quad (\overline{n}\ (\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)\ \langle \lambda \mathtt{z}.\overline{0},\ \overline{0}\rangle).2 \\
&\equiv \quad (\ (\lambda \mathtt{f}.\lambda \mathtt{x}.\ \mathtt{f}^n\ \mathtt{x})\ (\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)\ \langle \lambda \mathtt{z}.\overline{0},\ \overline{0}\rangle\ ).2 \\
&\longrightarrow \quad (\ (\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)^n\ \langle \lambda \mathtt{z}.\overline{0},\ \overline{0}\rangle\ ).2
\end{aligned}
$$

Nun können wir zwei Fälle für diesen Redex unterscheiden:

Erster Fall — $n = 0$**.** Dann gilt:

$$
\begin{aligned}
\mathsf{p}\ \overline{0} \quad &\equiv \quad (\ (\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)^0\ \langle \lambda \mathtt{z}.\overline{0},\ \overline{0}\rangle\ ).2 \\
&\equiv \quad \langle \lambda \mathtt{z}\ \overline{0},\ \overline{0}\rangle.2 \\
&\longrightarrow \quad \overline{0}
\end{aligned}
$$

Zweiter Fall — $n = m + 1$**.** Dann gilt:

$$
\begin{aligned}
\mathsf{p}\ \overline{m+1} \quad &\equiv \quad (\ (\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)^{m+1}\ \langle \lambda \mathtt{z}.\overline{0},\ \overline{0}\rangle\ ).2 \\
&\equiv \quad (\quad (\ \lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)^m \\
&\qquad\quad (\ (\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)\quad \langle \lambda \mathtt{z}.\ \overline{0},\ \overline{0}\rangle\ )\ ).2 \\
&\longrightarrow \quad (\quad (\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)^m \\
&\qquad\quad \langle \mathtt{s},\ \text{match } \langle \lambda \mathtt{z}.\ \overline{0},\ \overline{0}\rangle \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle\ ).2 \\
&\longrightarrow \quad (\ (\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)^m\ \langle \mathtt{s},\ (\lambda \mathtt{z}.\ \overline{0})\ \overline{0}\rangle\ ).2 \\
&\longrightarrow \quad (\ (\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)^m\ \langle \mathtt{s},\ \overline{0}\rangle\ ).2 \\
&\qquad \text{siehe Lemma (*)} \\
&\longrightarrow \quad \langle \mathtt{s},\ \mathtt{s}^m\ \overline{0}\rangle.2 \\
&\longrightarrow \quad \mathtt{s}^m\ \overline{0} \\
&\longrightarrow \quad \overline{m}
\end{aligned}
$$

Induktionsbeweis für Lemma (*) —
$(\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)^n\ \langle \mathtt{s},\ \overline{0}\rangle \equiv \langle \mathtt{s},\ \mathtt{s}^n\ \overline{0}\rangle$:

Anfang — $n = 0$:
$(\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)^0\ \langle \mathtt{s},\ \overline{0}\rangle \equiv \langle \mathtt{s},\ \overline{0}\rangle \equiv \langle \mathtt{s},\ \mathtt{s}^0\ \overline{0}\rangle$ $\surd$

Schritt: es gelte $(\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)^n\ \langle \mathtt{s},\ \overline{0}\rangle \equiv \langle \mathtt{s},\ \mathtt{s}^n\ \overline{0}\rangle$.

Dann folgt:

$$
\begin{array}{rl}
 & (\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)^{n+1}\ \ \langle \mathtt{s},\overline{0}\rangle \\
\longrightarrow & \ (\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle) \\
 & \qquad (\ (\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)^n\ \ \langle \mathtt{s},\ \overline{0}\rangle\ ) \\
\text{wegen Annahme} & \\
\equiv & (\lambda \mathtt{z}.\ \langle \mathtt{s},\ \text{match } \mathtt{z} \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle)\ \ \langle \mathtt{s},\ \mathtt{s}^n\ \overline{0}\rangle \\
\longrightarrow & \langle \mathtt{s},\ \text{match } \langle \mathtt{s},\ \mathtt{s}^n\ \overline{0}\rangle \text{ with } \langle \mathtt{f},\mathtt{x}\rangle \mapsto \mathtt{f}\ \mathtt{x}\rangle \\
\longrightarrow & \langle \mathtt{s},\ \mathtt{s}\ (\mathtt{s}^n\ \overline{0})\rangle \\
\equiv & \langle \mathtt{s},\ \mathtt{s}^{n+1}\ \overline{0}\rangle
\end{array}
$$

Es gibt einen kürzeren Beweis ... wenn ich Zeit finde, schreibe ich ihn auf