Merkzettel für „Theorie der Programmierung“

Marco Ammon

5. Oktober 2018

Termersetzungssysteme

Terminierung

Polynomordnungen

Konfluenz

Critical Pairs

λ-Kalkül

Ungetypt

- β-Reduktion
$$(\lambda x.t)s \to_\beta t[s/x]$$
- η-Reduktion
$$\lambda x.yx \to_\eta y$$

Auswertungsstrategien

- applikativ (*leftmost-innermost*) $\to_a$ *Def. 3.13 (33)*
  - $\lambda x.t \to_a \lambda x.t'$, wenn $t \to_a t'$
  - $ts \to_a t's$, wenn $t \to_a t'$
  - $ts \to_a ts'$, wenn $s \to_a s'$ und t normal ist
  - $(\lambda x.t)s \to_a t[s/x]$, wenn t und s normal sind
  - effizient
- normal (*leftmost-outermost*) $\to_n$ *Def. 3.14 (34)*
  - $(\lambda x.t)s \to_n t[s/x]$
  - $\lambda x.t \to_n \lambda x.t'$, wenn $t \to_n t'$
  - $ts \to_n t's$m wenn $t \to_n t'$ und t keine λ-Abstraktion ist
  - $ts \to_n ts'$, wenn $s \to_n s'$ und t normal und keine λ-Abstraktion ist
  - terminiert immer, falls Normalform existiert (nach Standardisierungssatz) *Satz 3.17 (35)*

Einfach getypt ($\lambda \rightarrow$)

- Church: Annotation der Variablen mit Typen, nur herleitbare Terme hinschreibbar *S. 39*
- Curry: Alle Terme hinschreibbar, dann Aussondern der nicht typisierbaren
- Typregeln: *S. 39*

$$(Ax)\ \frac{}{\Gamma \vdash x : \alpha}\ x : \alpha \in \Gamma$$

$$(\rightarrow_e)\ \frac{\Gamma \vdash t : \alpha \rightarrow \beta \qquad \Gamma \vdash s : \alpha}{\Gamma \vdash ts : \beta}$$

$$(\rightarrow_i)\ \frac{\Gamma, x : \alpha \vdash t : \beta}{\Gamma \vdash \lambda x.t : \alpha \rightarrow \beta}$$

- Typisierungsprobleme *S. 40*
  - Typcheck: „Gilt $\Gamma \vdash t : \alpha$?“
  - Typinferenz: „Was ist das beste α / Existiert α mit $\Gamma \vdash t : \alpha$?“
  - Type inhabitation: „Existiert t mit $\Gamma t : \alpha$?“
- Inversionslemma *Lem. 3.29 (41)*
  1. $\Gamma \vdash x : \alpha \Rightarrow x : \alpha \in \Gamma$
  2. $\Gamma \vdash ts : \beta \Rightarrow \exists \alpha$ mit $\Gamma \vdash t : \alpha \rightarrow \beta$ und $\Gamma \vdash s : \alpha$
  3. $\Gamma \vdash \lambda x.t : \gamma \Rightarrow \gamma = \alpha \rightarrow \beta$ mit $\Gamma, x : \alpha \vdash t : \beta$
- Typinferenz *S. 41*
  - Typsubstitution σ ist Lösung von $\Gamma \vdash t : \alpha$, wenn $\Gamma\sigma \vdash t : \alpha\sigma$ herleitbar
  - Substitutionen: σ_1 allgemeiner als $\sigma_2 \Leftrightarrow \exists \tau . \sigma_1 \tau = \sigma_2$ *GLoIn, S. 38*
  - Prinzipaltyp von Γ, t ist $\sigma(a)$ für allgemeinste Lösung σ von $\Gamma \vdash t : a$ (a frisch)
  - Algorithmus W (Hindley/Milner) *Alg. 3.31 (42)*
    - Menge PT von Typgleichungen

$$\begin{aligned} PT(\Gamma; x; \alpha) &= \{a \doteq b \mid x : \beta \in \Gamma\} \\ PT(\Gamma; \lambda x.t; \alpha) &= PT((\Gamma; x : a); t; b) \cup \{a \rightarrow b \doteq \alpha\} \text{ mit } a, b \text{ frisch} \\ PT(\Gamma; ts; \alpha) &= PT(\Gamma; t; a \rightarrow \alpha) \cup PT(\Gamma; s; a) \text{ mit } a \text{ frisch} \end{aligned}$$

    - Typinferenz des Terms u mit leerem Kontext:

$$\varepsilon := PT(\emptyset; u; a)$$

    $\Rightarrow$ Prinzipaltyp von u: mgu$(\varepsilon)(a)$

- Subjektreduktion: Wenn $\Gamma \vdash t : \alpha$ und $t \rightarrow_\beta^* s$, dann auch $\Gamma \vdash s : \alpha$, aber nicht umgekehrt! *Satz 3.38 (45)*

Induktive Datentypen

Mengenkonstruktionen

Def. 4.12 (56)
Lem. 4.13 (57)

$$
\begin{aligned}
X_1 \times X_2 &= \{(x_1, x_2)\,|x_i \in X_i \text{ für } i = 1,2\} && (\text{„struct“}) \\
X_1 + X_2 &= \{(i,x)|i = 1,2, x \in X_i\} && (\text{„union“}) \\
1 &= \{*\} && \text{„()“ in Haskell}
\end{aligned}
$$

Seien $f_i : X_i \to Y_i$, $g_i : X_i \to Z$ und $h_i : Z \to X_i$ mit $i \in \{1,2\}$.

$$
\begin{aligned}
f_1 \times f_2 &: X_1 \times X_2 \to Y_1 \times Y_2, & (f_1 \times f_2)(x_1, x_2) &= (f_1(x_1), f_2(x_2)) \\
f_1 + f_2 &: X_1 + X_2 \to Y_1 + Y_2, & (f_1 + f_2)(i,x) &= (i, f_i(x)) \\
[g_1, g_2] &: X_1 + X_2 \to Z, & [g_1, g_2])(i,x) &= g_i(x) \\
\langle h_1, h_2 \rangle &: Z \to X_1 \times X_2, & \langle h_1, h_2 \rangle(z) &= (h_1(z), h_2(z)) \\
in_i &: X_i \to X_1 + X_2, & in_i(x) &= (i,x) \\
\pi_i &: X_1 \times X_2 \to X_i, & \pi_i(x_1, x_2) &= x_i \\
1 &: 1 \to 1
\end{aligned}
$$

Es gilt

$$
\begin{aligned}
[g_1, g_2] \circ in_i &= g_i \\
f_1 + f_2 &= [in_1 \circ f_1, in_2 \circ f_2] \\
[r \circ in_1, r \circ in_2] &= r \text{ für } r : X_1 + X_2 \to Z \\
\pi_i \circ \langle h_1, h_2 \rangle &= h_i \\
f_1 \times f_2 &= \langle f_1 \circ \pi_1, f_2 \circ \pi_2 \rangle \\
\langle \pi_i \circ f, \pi_2 \circ f \rangle &= f \text{ für } f : Z \to X_1 \times X_2
\end{aligned}
$$

Strukturelle Induktion

- über einsortige Datentypen *S. 63*
  - Induktionsanfang: „Anfangs“-Konstruktor (etwa Nil)
  - Induktionsschritt: alle anderen Konstruktoren (etwa $cons$)
- über mehrsortige Datentypen *S. 64*
  - Funktionen müssen immer auf allen Datentypen definiert werden

Kodatentypen

- Definition über „Destruktoren“ etwa hd und tl

Koinduktion

- Bisimulation $R \subseteq A^\omega \times A^\omega$, wenn für alle $(s,t) \in R$ gilt: *Def. 4.39 (74)*

$$
\begin{gathered}
hd\ s = hd\ t \\
(tl\ s)\ R\ (tl\ t)
\end{gathered}
$$

- Wenn R eine Bisimulation ist, gilt $sRt \Rightarrow s = t$ *Satz 4.40 (74)*

Kodatentypen mit Alternativen

System F

Curry

- Typen: *Def. 5.1 (84)*

$$\alpha, \beta \coloneqq a \mid \alpha \to \beta \mid \forall a.\alpha \qquad (a \in V)$$

- Typisierung: *Def. 5.1 (84)*

$$(Ax)\ \frac{}{\Gamma \vdash x : \alpha}\ (x : \alpha \in \Gamma)$$

$$(\to_e)\ \frac{\Gamma \vdash t : \alpha \to \beta \qquad \Gamma \vdash s : \alpha}{\Gamma \vdash ts : \beta}$$

$$(\to_i)\ \frac{\Gamma, x : \alpha \vdash t : \beta}{\Gamma \vdash \lambda x.t : \alpha \to \beta}$$

$$(\forall_i)\ \frac{\Gamma \vdash s : \alpha \qquad a \notin FV(\Gamma)}{\Gamma \vdash s : \forall a.\alpha}$$

$$(\forall_e)\ \frac{\Gamma \vdash s : \forall a.\alpha}{\Gamma \vdash s : (\alpha[\beta/a])}$$

Church-Kodierung

S. 84

- Natürliche Zahlen

$$\begin{aligned} \mathbb{N} &\coloneqq \forall a.(a \to a) \to a \to a \\ zero &: \mathbb{N} \\ zero &= \lambda fx.x \\ suc &: \mathbb{N} \to \mathbb{N} \\ suc &= \lambda nfx.f(nfx) \\ fold &: \forall a.(a \to a) \to a \to \mathbb{N} \to a \\ fold &= \lambda fxn.nfx \\ add &: \mathbb{N} \to \mathbb{N} \to \mathbb{N} \\ add &= \lambda n.fold\ suc\ n \end{aligned}$$

- Paare

$$\begin{aligned} (a \times b) &\coloneqq \forall r.(a \to b \to r) \to r \\ pair &: \forall ab.a \to b \to (a \times b) \\ pair &= \lambda xyf.fxy \\ fst &: \forall ab.(a \times b) \to a \\ fst &= \lambda p.p(\lambda xy.x) \\ snd &: \forall ab.(a \times b) \to b \\ snd &= \lambda p.p(\lambda xy.y) \end{aligned}$$

- Summen

$$\begin{aligned}
(a+b) &\coloneqq \forall r.(a \to r) \to (b \to r) \to r \\
inl &: \forall ab.a \to (a+b) \\
inl &= \lambda xfg.fx \\
inr &: \forall ab.b \to (a+b) \\
inr &= \lambda yfg.gy \\
case &: \forall abs.(a \to s) \to (b \to s) \to (a+b) \to s \\
case &= \lambda fgs.sfg
\end{aligned}$$

- Listen

$$\begin{aligned}
List\ a &\coloneqq \forall r.r \to (a \to r \to r) \to r \\
Nil &: \forall a.Lista\ a \\
Nil &= \lambda uf.u \\
Cons &: \forall a.a \to List\ a \to List\ a \\
Cons &= \lambda xluf.fx(luf) \\
len &: \forall a.List\ a \to \mathbb{N} \\
len &= \lambda l.l\ zero\ (\lambda xr.suc\ r)
\end{aligned}$$

ML-Polymorphie

- Einschränkung von System F durch $\forall$ nur auf oberster Ebene sowie Mehrfachinstanziierung polymorpher Funktionen nur in *let*-Konstrukt *S. 88*
- Typen

$$\alpha, \beta \coloneqq a \mid \alpha \to \beta$$

- Typschemata

$$S \coloneqq \forall a_1, \ldots, a_k.\alpha \qquad (k \geq 0)$$

- Terme

$$t, s \coloneqq x \mid t\ s \mid \lambda x.t \mid let\ x = t\ in\ s$$

- Kontexte

$$\begin{aligned}
\Gamma &= (x_1 : S_1, \ldots, x_n : S_n) \\
Cl(\Gamma, \alpha) &= \forall a_1, \ldots, a_k.\alpha \qquad \text{für } FV(\alpha) \backslash FV(\Gamma) = \{a_1, \ldots, a_k\}
\end{aligned}$$

- Typisierungsregeln *S. 88*

$$(\forall_e)\ \frac{}{\Gamma \vdash x : \alpha[\beta_1/a_1, \ldots, \beta_k/a_k]}\ (x : \forall a_1, \ldots, \forall a_k.\alpha) \in \Gamma$$

$$(\to_i)\ \frac{\Gamma, x : \alpha \vdash t : \beta}{\Gamma \vdash \lambda x.t : \alpha \to \beta}$$

$$(\to_e)\ \frac{\Gamma \vdash t : \alpha \to \beta \qquad \Gamma \vdash s : \alpha}{\Gamma \vdash ts : \beta}$$

$$(\text{let})\ \frac{\Gamma \vdash t : \alpha \qquad \Gamma, x : Cl(\Gamma, \alpha) \vdash s : \beta}{\Gamma \vdash \text{let } x = t \text{ in } s : \beta}$$

- Inversionslemma *S. 89*
  1. Wenn $\Gamma \vdash x : \alpha$, dann existieren Typen β_i und ein Typschema $S = \forall a_1 \dots \forall a_k.\gamma$, so dass $(x : S) \in \Gamma$ und $\alpha = \gamma[\beta_1/a_1, \dots, \beta_k/a_k]$
  2. Wenn $\Gamma \vdash (\text{ let } x = s \text{ in } t) : \alpha$, dann existiert ein Typ β nut $\Gamma \vdash s : \beta$ und $\Gamma, x : Cl(\Gamma, \beta) \vdash t : \alpha$
- erweiterter Algorithmus W mit Menge PT von Typgleichungen

$$\begin{aligned} PT(\Gamma; x; \alpha) &= \{a \doteq \gamma[a'_1/a_1, \dots, a'_k/a_k] | (x : \forall a_1, \dots, \forall a_k.\gamma) \in \Gamma\} \\ PT(\Gamma; \lambda x.t; \alpha) &= PT((\Gamma; x : a); t; b) \cup \{a \to b \doteq \alpha\} \text{ mit } a, b \text{ frisch} \\ PT(\Gamma; ts; \alpha) &= PT(\Gamma; t; a \to \alpha) \cup PT(\Gamma; s; a) \text{ mit } a \text{ frisch} \\ PT(\Gamma; (\text{let } x = s \text{ in } t); \alpha) &= PT(\Gamma\sigma, x : Cl(\Gamma\sigma, \sigma(b)); t; \alpha\sigma) \end{aligned}$$

  wobei $\sigma = mgu(PT(\Gamma; s; b))$ mit b frisch

Minimierung von deterministischen endlichen Automaten

1. Entferne aus Q alle nicht erreichbaren Zustände
2. Initialisiere R auf $\{(q_1, q_2) \mid q_1 \in F \Leftrightarrow q_2 \in F\}$
3. Suche ein Paar $(q_1, q_2) \in R$ und einen Buchstaben $a \in \Sigma$ mit

$$(\delta(a, q_1), \delta(a, q_2)) \notin R$$

   Wenn kein solches Paar gefunden wird, gehe zu Schritt 4. Andernfalls entferne (q_1, q_2) aus R und fahre bei 3. fort.
4. Identifiziere alle Zustandspaare in R.

Unifikationsalgorithmus (Martelli/Montanari)

$$\begin{array}{lll} S \cup \{x \doteq x\} & \to S & \text{(delete)} \\ S \cup \{f(E_1, \dots, E_n) \doteq f(D_1, \dots, D_n)\} & \to S \cup \{E_1 \doteq D_1, \dots, E_n \doteq D_n\} & \text{(decomp)} \\ S \cup \{f(E_1, \dots, E_n) \doteq g(D_1, \dots, D_k)\} & \to \bot \text{ (für } f \neq g) & \text{(conflict)} \\ S \cup \{E \doteq x\} & \to S \cup \{x \doteq E\} \text{ (für } E \text{ keine Variable)} & \text{(orient)} \\ S \cup \{x \doteq E\} & \to \begin{cases} \bot (\text{für } x \in FV(E), x \neq E) \\ S[E/x] \cup \{x \doteq E\} (\text{für } x \notin FV(E), x \in FV(S)) \end{cases} & \text{(occurs)/(eli} \end{array}$$