Implementation of Substitution in the AUTOMATH Verification Program

L.S. Jutting

AUTOMATH	de Bruijn 1967-now	proofchecking feasable (which is more than decidable)
Verification Program	Zandleven 1971-1976	- simulating pointers in Burroughs Algol - controlling memory access - interactive - big (30,000 lines)
Question	Coquand, Huet 1986	How is substitution implemented?

AUTOMATH

Based on 1. Definitions.

2. λ-calculus.

1. Application of definitions : δ-reduction

if $d(x_n, \ldots, x_0) := D$

then $d(A_n, \ldots, A_0) > s^{x_n \ldots x_0}_{A_n \ldots A_0} D$

2. Application of functions to arguments: β-reduction

$(\lambda x : A . BC) > s^{x}_{C} B$

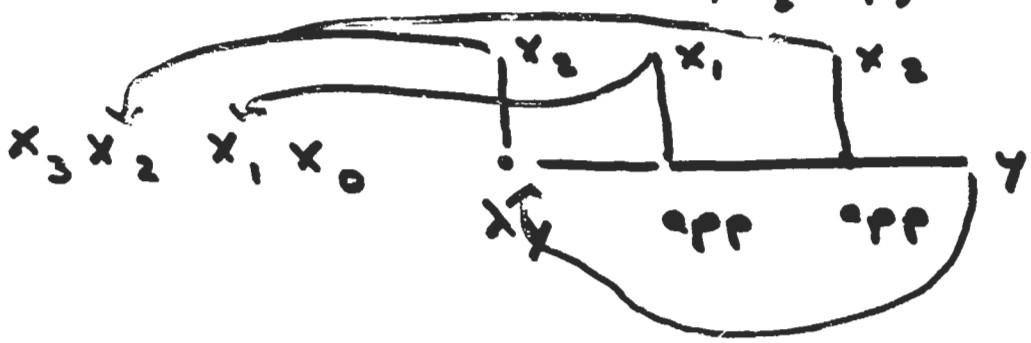
Difficulties with Substitution

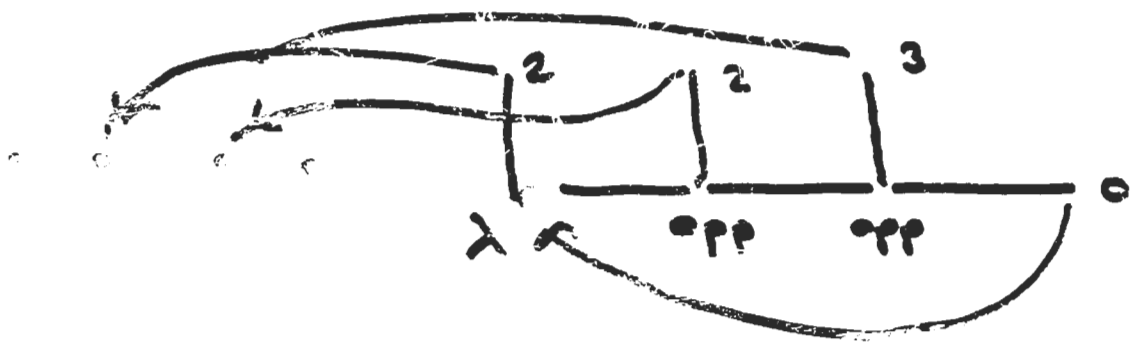
1. α-conversion.

2. copying.

Solutions 1. de Bruijn indices.

(x_3, x_2, x_1, x_0) $\lambda y{:}x_2((y\,x_2)\,x_1)$





( ) $\lambda 2.((0\,3)\,2)$

I : Definition Language

(PAL · SEMIPAL)

variables : $0, 1, 2, 3, \ldots$

constants : $p, q, r, \ldots$ $\in \mathcal{P}$

expressions:

$\uparrow_d$ | variables

$p(\bar{A})$ for any sequence $\bar{A} = \ldots A_n, \ldots A_1, A_0$ of expressions.

<u>Simultaneous substitution:</u>

if $\bar{B} = \ldots B_n, \ldots B_1, B_0$ is a sequence of expressions

then

$$s_{\bar{B}}\, i = B_i$$

$$s_{\bar{B}}\, p(\bar{A}) = p(s_{\bar{B}}\, \bar{A})$$

<u>Thm:</u> $s_{\bar{B}}\, s_{\bar{C}} = s_{s_{\bar{B}}\bar{C}}$

<u>δ-reduction</u>

Δ a (partial) function : $\mathcal{P} \to \mathcal{T}$

$$p(\bar{A}) > s_{\bar{A}}\, \Delta(p)$$

<u>contexts</u> : functions $\Gamma : k\omega \to \mathcal{T}_d$

$k \geq 0$ $\quad \mathcal{T}_d$ is the set of terms

$$\ldots, B_n, \ldots B_1, B_0, \ldots, A_n, \ldots A_1, A_0 \qquad C$$

$$\langle \ldots \quad \text{context } \Gamma \quad \ldots \ldots \rangle$$

C should be interpreted w.r.t. Γ

Operations on contexts:

Extension: $\delta_{\bar{A}}$

Cutting : γ_ω

<u>interpretation</u>: $I(C, \Gamma) \in \mathcal{T}_d$

$$I(i, \Gamma) = \begin{cases} I(\Gamma(i), \gamma_\omega \Gamma) & \text{if } dom(\Gamma) \neq \emptyset \\ i & \text{if } dom(\Gamma) = \emptyset \end{cases}$$

$$I(p(\bar{A}), \Gamma) = p(I(\bar{A}, \Gamma))$$

<u>Thm</u> $\quad I(C, \delta_{\bar{A}} \Gamma) = s_{I(\bar{A}, \Gamma)} C$

Application to <u>δ-reduction</u>

$$I(p(\bar{A}), \Gamma) >_{\delta}$$

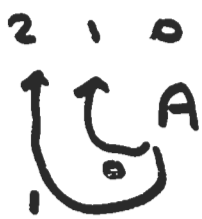
$$I(\Delta(p), \delta_{\bar{A}} \Gamma)$$

II: λ-calculus

$\mathcal{T}_\lambda$
- variables: 0, 1, 2, 3, ...
- application: (AB)
- abstraction: λA.B

Substitution S^n_A

$\lambda B.C$



$$S^n_A\, i = \begin{cases} i & \text{if } i \neq n \\ u^{n+1}_0 A & \text{if } i = n \end{cases}$$

$$S^n_A (BC) = (S^n_A B \;\; S^n_A C)$$

$$S^n_A \lambda B.\underline{C} = \lambda S^n_A B.\, S^{n+1}_A C$$

Updating u^n_k

$\lambda B.C$

remains 0

$$u^n_k\, i = \begin{cases} i & \text{if } i < k \\ n+i & \text{if } i \geq k \end{cases}$$

$$u^n_k (BC) = (u^n_k B \; u^n_k C)$$

$$u^n_k \lambda B.C = \lambda u^n_k B.\, u^n_{k+1} C$$

<u>Thm's</u> 1. $U^n_k U^m_\ell = \begin{cases} U^m_\ell U^n_{k-m} & k \geq \ell+m \\ U^{n+m}_\ell & \ell \leq k \leq \ell+m \end{cases}$

2. $S^n_A U^m_\ell = \begin{cases} U^m_\ell S^{n-m}_A & n \geq \ell+m \\ U^m_\ell & \ell \leq n < \ell+m \end{cases}$

3. $S^n_A S^m_B = S^m_{S^{n-m-1}_A B} S^n_A \qquad n > m$

<u>β-reduction</u> $(\lambda A.B\,C) > S^A_C B$

(remark: Here the context should be extended)

$\omega \to (T \times \omega)$

<u>Contexts</u>: partial functions $\Gamma \to \cancel{(T \times \omega)}$

$\#\,dom(\Gamma)$ finite

$\langle A, \ell \rangle \cdot \langle B, m \rangle \cdot \quad C$

C should be interpreted w.r.t. Γ.

Operations on contexts:

Extension: ε^k

(placing extra dots: $\cdot \quad \cdot \quad \cdot$)

Cutting: γ_k

(removing the final entries)

Substitution: $\sigma^n_{\langle A, \ell \rangle}$ (for $n \notin dom(\Gamma)$)

(extending the domain to n)

<u>interpretation</u> $I(C, \Gamma) \in \mathcal{T}_\lambda$

$$I(i, \Gamma) = \begin{cases} U_0^{i+\ell+1} I(B, \gamma_{i+\ell+1} \Gamma) & \text{if } i \in dom(\Gamma),\ \Gamma(i) = \langle B, \ell \rangle \\ i & \text{if } i \notin dom(\Gamma) \end{cases}$$

$$I((AB), \Gamma) = (I(A, \Gamma)\ I(B, \Gamma))$$

$$I(\lambda A.B, \Gamma) = \lambda I(A, \Gamma).\ I(B, \varepsilon' \Gamma)$$

<u>Thm's</u> 1. $I(U_k^n A, \varepsilon^k \Gamma) = U_k^n I(A, \varepsilon^k \gamma^n \Gamma)$

<u>Corr</u> $I(U_0^n A, \Gamma) = U_0^n I(A, \gamma^n \Gamma)$

2. $I(A, \sigma^n_{\langle B, \ell \rangle} \Gamma) = S^n_{B_0}\ I(A, \Gamma)$

where $B_0 = I(U_0^\ell B, \gamma^n \Gamma)$

Application to β-reduction

Let

$\cdots \langle\rangle\langle\rangle\cdot\langle\rangle\cdot\langle\rangle\cdot \qquad (AB)$

$$I(A) \geq I'(\lambda A_1.A_2)$$

$\cdots \langle\rangle\langle\rangle\cdot\langle\rangle\cdot\langle\rangle\cdot\left\{\begin{array}{l}\langle\rangle\langle\rangle\langle\rangle\langle\rangle \\ \xleftarrow[\ell]{\qquad\qquad}\end{array}\right. \quad \lambda A_1.A_2$

then $I((AB)) \geq I''(A_2)$

$\cdots \langle\rangle\langle\rangle\cdot\langle\rangle\cdot\langle\rangle\cdot\left\{\langle\rangle\langle\rangle\langle\rangle\langle\rangle\langle B,\ell\rangle \quad A_2\right.$

or: If $I(A,\Gamma) \geq I(\lambda A_1.A_2,\Gamma')$

$$\Gamma = \gamma^{\ell}\Gamma'$$

then $I((AB),\Gamma) \geq I(A_2, S^{0}_{\langle B,L\rangle}\,\varepsilon'\Gamma')$

Combination of I & II

This is straightforward.
Takes some time.

Combination of Zendian implementations.

[The same
A paper will appear.

Moreover:

Also typing (of variables, constants and expressions) can be implemented using this device.

Discussion of results

1. Zandleven implementation is adequate to describe substitution & typing
2. Zandleven implementation is adequate to describe outside β-reduction & δ-reduction
3. Zandleven implementation does not require copying, and therefore reduces the required memory space.
4. As to aspects of performance I have no hard facts. The AUTOMATH checker is rather good for a program over 10 years old.

L.S. Jutting
dept. of Mathematics & Computing Science
Univ. of Technology, Eindhoven
the Netherlands.