3.2 Semantics of the Lambda Calculus

Mittwoch, 8. Juni 2016 11:00

We now define the operational semantics of the lambda calculus by introducing an interpreter for lambda terms that uses 3 reduction rules: α-reduction, β-reduction, δ-reduction

α-reduction: renaming bound variables

Def 3.2.1 (α-Reduction) (Slide 54)

The relation $\rightarrow_\alpha \subseteq \Lambda \times \Lambda$ is the smallest relation with

- $\lambda x.t \rightarrow_\alpha \lambda y.t[x/y]$ if $y \notin free(t)$
- if $t_1 \rightarrow_\alpha t_2$, then $(t_1\ r) \rightarrow_\alpha (t_2\ r)$, $(r\ t_1) \rightarrow_\alpha (r\ t_2)$, and $(\lambda y.t_1) \rightarrow_\alpha (\lambda y.t_2)$.

Ex:

$$\lambda x y.x y \rightarrow_\alpha \lambda x z.x z$$
$$\rightarrow_\alpha \lambda v z.v z$$
$$\rightarrow_\alpha \ldots$$

$$\lambda x.x y \not\rightarrow_\alpha \lambda x.x z$$

No renaming of free variables

β-reduction is used to apply a λ-abstraction to a term: $(\lambda x.\, t)\ r$ results in t where all free occurrences of x are replaced by r

<u>Def 3.2.2</u> (β-Reduction) (Slide 54)

The relation $\rightarrow_\beta \subseteq \Lambda \times \Lambda$ is the smallest relation with

- $(\lambda x.\, t)\ r \rightarrow_\beta t\,[x/r]$
- if $t_1 \rightarrow_\beta t_2$, then $(t_1\ r) \rightarrow_\beta (t_2\ r)$, $(r\ t_1) \rightarrow_\beta (r\ t_2)$, and $(\lambda y.\, t_1) \rightarrow_\beta (\lambda y.\, t_2)$

Ex: $(\lambda x.\, x)\ zero \rightarrow_\beta x\,[x/zero] = zero$

$(\lambda x y.\, x y)\ y \rightarrow_\beta (\lambda y.\, x y)[x/y] = \lambda y'.\, y\ \ y'$

$$\begin{tikzcd}[column sep=tiny]
 & (\lambda x.\, plus\ x\ 1)\,((\lambda y.\, times\ y\ y)\ 3) \arrow[dl, "\beta"'] \arrow[dr, "\beta"] & \\
plus\,((\lambda y.\, times\, y\, y)\, 3)\ 1 \arrow[dr, "\beta"'] & & (\lambda x.\, plus\ x\ 1)\ (times\ 3\ 3) \arrow[dl, "\beta"] \\
 & plus\ (times\ 3\ 3)\ 1 &
\end{tikzcd}$$

There can be several possibilities to evaluate a

λ-term by β-reduction. Do they always yield the same result?

<u>Def 3.2.3</u> (transitive-reflexive closure, normal form, confluence)

Let $\rightarrow$ be a relation on some set N.

(a) The <u>transitive-reflexive closure</u> $\rightarrow^*$ is the smallest relation such that:

- $t_1 \rightarrow t_2$ implies $t_1 \rightarrow^* t_2$
- $t_1 \rightarrow t_2 \rightarrow^* t_3$ implies $t_1 \rightarrow^* t_3$
- $t_1 \rightarrow^* t_1$

In other words: $t_0 \rightarrow^* t_n$ iff $t_0 \rightarrow t_1 \rightarrow t_2 \rightarrow \ldots \rightarrow t_n$ for $n \geq 0$

(b) An object $q \in N$ is a <u>normal form</u> iff there is no $q' \in N$ with $q \rightarrow q'$.
We say that q is a <u>normal form of t</u> iff $t \rightarrow^* q$ and q is a normal form.

(c) The relation $\rightarrow$ is <u>confluent</u> iff for all $t, q_1, q_2 \in N$ we have: if $t \rightarrow^* q_1$ and $t \rightarrow^* q_2$,
then there exists a $q \in N$ with $q_1 \rightarrow^* q$ and $q_2 \rightarrow^* q$.

$$\begin{tikzcd}
 & t \arrow[dl, "*"'] \arrow[dr, "*"] & \\
q_1 \arrow[dr, "*"'] & & q_2 \arrow[dl, "*"] \\
 & q &
\end{tikzcd}$$

if the green part holds,
then the red part holds as well.

<u>Lemma 3.2.4</u> (Confluence implies unique normal forms)
Let $\to$ be a confluent relation on a set N. Then every $t \in N$ has at most one normal form.

<u>Proof</u>: Let $t \in N$ have 2 normal forms q_1, q_2.

$$\begin{tikzcd}
 & t \arrow[dl, "*"'] \arrow[dr, "*"] & \\
q_1 \arrow[dr, "*"'] & & q_2 \arrow[dl, "*"] \\
 & q &
\end{tikzcd}$$

By confluence, there exists a $q \in N$ with $q_1 \to^* q$, $q_2 \to^* q$.

Since q_1, q_2 are normal forms, we have
$q_1 = q = q_2$. $\boxtimes$

<u>Thm 3.2.5</u> (Confluence of the λ-calculus with β-reduction, Church & Rosser)

$\to_\beta$ is confluent, i.e.,
if $t \to_\beta^* q_1$ and $t \to_\beta^* q_2$,
then there exist $q, q' \in \Lambda$ with
$q_1 \to_\beta^* q$, $q_2 \to_\beta^* q'$, and $q \to_\alpha^* q'$

$$\begin{tikzcd} & t \arrow[dl, "*\ \beta"'] \arrow[dr, "\beta\ *"] & \\ q_1 \arrow[d, "\beta\ *"'] & & q_2 \arrow[d, "*\ \beta"] \\ q \arrow[rr, "*"', "\alpha"] & & q' \end{tikzcd}$$

The last form of reduction rules in the λ-calculus is needed to evaluate terms built with constants from $\mathcal{C}$. In particular, these constants could correspond to pre-defined functions of Haskell.

But: We want to make sure that <u>δ-reduction</u> does not destroy the confluence of the λ-calculus.

<u>Solution</u>: Define δ-reduction by a set of δ-rules.
These rules must have a certain restricted form which ensures that evaluation with β- and δ-reduction remains confluent.

<u>Def 326</u> (δ-Reduction) (Slide 55)

A set of rules δ of the form $c\, t_1 \dots t_n \to r$ with $c \in \mathcal{C}$, $t_1, \dots, t_n, r \in \Lambda$ is a <u>Delta-Rule-Set</u> iff

- $t_1, \dots, t_n, r$ are <u>closed</u> λ-terms $\leftarrow$

otherwise: $\left.\begin{array}{l} c\,0 \to 0 \\ c\,x \to 1 \end{array}\right\}\delta$

Then $c\,0$ $\swarrow_\delta$ $\searrow$

Then
$$\begin{tikzcd}[column sep=small] & c\,0 \arrow[dl, "\delta"'] \arrow[dr, "\delta"] & \\ 0 & & 1 \end{tikzcd}$$

- $t_1, \dots, t_n$ are in $\to_\beta$-normal form and they do not contain any left-hand side of a rule from δ

otherwise: $c((\lambda x.0)\ 1) \to 0$
$\qquad\qquad c\ \ 0 \to 1$

$$\begin{tikzcd}[column sep=small] & c((\lambda x.0)\ 1) \arrow[dl, "\delta"'] \arrow[dr, "\beta"] & & \\ 0 & & c\ 0 \arrow[dr, "\delta"] & \\ & & & 1 \end{tikzcd}$$

otherwise: $c\ 0 \to 0$
$\qquad\qquad c\ d \to 1$
$\qquad\qquad d \to 0$

$$\begin{tikzcd}[column sep=small] & & c\ d \arrow[dl, "\delta"'] \arrow[dr, "\delta"] & \\ & c\ 0 \arrow[dl, "\delta"'] & & 1 \\ 0 & & & \end{tikzcd}$$

- In δ, there are no two rules $c\ t_1 \dots t_n \to r$, $c\ t_1 \dots t_n \dots t_m \to r'$ with $m \geq n$.

otherwise: $c\ 0 \to 0$
$\qquad\qquad c\ 0\ 1 \to 1$

$$\begin{tikzcd}[column sep=small] & c\ 0\ 1 \arrow[dl, "\delta"'] \arrow[dr, "\delta"] & \\ 0\ 1 & & 1 \end{tikzcd}$$

For such a set δ, $\to_\delta$ is the smallest relation with

- $\ell \to_\delta r$ for all $\ell \to r \in \delta$
- if $t_1 \to_\delta t_2$, then $(t_1\ r) \to_\delta (t_2\ r)$, $(r\ t_1) \to_\delta (r\ t_2)$, and $\lambda y.\, t_1 \to_\delta \lambda y.\, t_2$

We define $\longrightarrow_{\beta\delta} = \longrightarrow_\beta \cup \longrightarrow_\delta$.

Ex. for a Delta-Rule-Set:

$\delta = \{ isa_{Succ}\ (Succ\ t) \rightarrow True \mid t \in \Lambda,\ t$ is closed, and in $\longrightarrow_{\beta\delta}$-normal form$\} \cup$

$\{ isa_{Succ}\ Zero \rightarrow False \}$

<u>Thm 3.2.7</u> (Confluence for the λ-calculus with $\rightarrow_\beta$ and $\rightarrow_\delta$)

$\longrightarrow_{\beta\delta}$ is confluent, i.e.,

if $t \xrightarrow{*}_{\beta\delta} q_1$ and $t \xrightarrow{*}_{\beta\delta} q_2$,

then there exist $q, q' \in \Lambda$ with

$q_1 \xrightarrow{*}_{\beta\delta} q$, $q_2 \xrightarrow{*}_{\beta\delta} q'$, and $q \xrightarrow{*}_{\alpha} q'$.

So $\longrightarrow_{\beta\delta}$ defines an <u>operational semantics</u> for the λ-calculus (and $\rightarrow_\alpha$ defines that we regard certain terms as being "equal").

(A denotational semantics for the λ-calculus was only invented 30 years later by D. Scott. Crucial idea: <u>Continuous</u> functions).