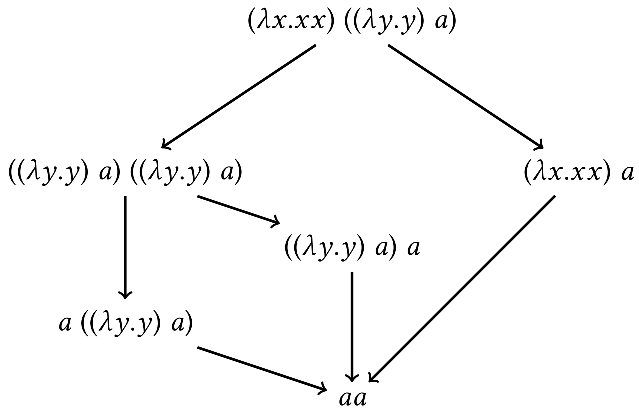
Lambda-calculus and programming language semantics

Thibaut Balabonski @ UPSay
Fall 2023
https://www.lri.fr/~blsk/LambdaCalculus/

Chapter 2: reduction strategies

Reduction graph

There may be several possible reductions for a given term.
The set of all possible reductions can be pictured as a graph



Questions:

- are some paths better than others?
- is there always a result in the end? is it unique?

1 Normalisation

Normal form

A *normal form* is a term that cannot be reduced anymore

Examples

- x
- $\lambda x.xy$
- $x\ (\lambda y.y)\ (\lambda z.zx)$

Counter-examples

- $(\lambda x.x)\ y$
- $x\ ((\lambda y.y)\ (\lambda z.zx))$

If $t \longrightarrow^* t'$ and t' is normal, the term t' is said to be a normal form of t
This defines our informal notion of a *result* of a term

Terms without normal form

$$\begin{array}{rcl} \Omega & = & (\lambda x.xx)\ (\lambda x.xx) \\ & \longrightarrow & (xx)\{x \leftarrow \lambda x.xx\} \\ & = & x\{x \leftarrow \lambda x.xx\}\ x\{x \leftarrow \lambda x.xx\} \\ & = & (\lambda x.xx)\ (\lambda x.xx) \\ & = & \Omega \end{array}$$

Summary:

- reduction of Ω does not terminate
- Ω is a term withour "result"

What about this other example?

$$(\lambda xy.y)\ \Omega\ z$$

Normalization properties

A term t is:

- *strongly normalizing* if every reduction sequence starting from t eventually reaches a normal form

$$(\lambda xy.y)\ ((\lambda z.z)\ (\lambda z.z))$$

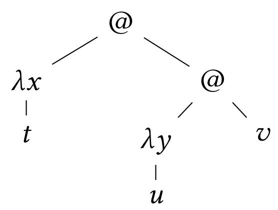
- *weakly normalizing*, or *normalizable*, if there is at least one reduction sequence starting from t and reaching a normal form

$$(\lambda xy.y)\ ((\lambda z.zz)\ (\lambda z.zz))$$

Note: normalization (strong or weak), is an undecidable property (see chapter on λ-computability)

2 Reduction strategies

Reduction orders



Normal order: reduce the most external redex first

- apply functions without reducing the arguments

Applicative order: reduce the most internal redex first

- normalize the arguments before reducing the function application itself

For disjoint redexes: from left to right

Exercise: normal order vs. applicative order

Compare normal order reduction and applicative order reduction of the following terms:

1. $(\lambda xy.x)\ z\ \Omega$
2. $(\lambda x.xx)\ ((\lambda y.y)\ z)$
3. $(\lambda x.x(\lambda y.y))\ (\lambda z.(\lambda a.aa)(z\ b))$

In each case: does another order allow shorter sequences?
Answer

1. Normal order

$$
\begin{array}{ll}
(\lambda xy.x)\ z\ \Omega \\
\longrightarrow & (\lambda y.z)\ \Omega \\
\longrightarrow & z
\end{array}
$$

Applicative order

$$
\begin{array}{ll}
(\lambda xy.x)\ z\ \Omega \\
\longrightarrow & (\lambda y.z)\ \Omega \\
\longrightarrow & (\lambda y.z)\ \Omega \\
\longrightarrow & \ldots
\end{array}
$$

Normal order reduction is as short as possible

2. Normal order

$$
\begin{array}{ll}
(\lambda x.xx)\ ((\lambda y.y)\ z) \\
\longrightarrow & ((\lambda y.y)\ z)\ ((\lambda y.y)\ z) \\
\longrightarrow & z\ ((\lambda y.y)\ z) \\
\longrightarrow & zz
\end{array}
$$

Applicative order

$$
\begin{array}{ll}
(\lambda x.xx)\ ((\lambda y.y)\ z) \\
\longrightarrow & (\lambda x.xx)\ z \\
\longrightarrow & zz
\end{array}
$$

Applicative order reduction is as short as possible

3. Normal order

$$
\begin{array}{ll}
(\lambda x.x(\lambda y.y))\ (\lambda z.(\lambda a.aa)\ (z\ b)) \\
\longrightarrow & (\lambda z.(\lambda a.aa)(z\ b))\ (\lambda y.y) \\
\longrightarrow & (\lambda a.aa)\ ((\lambda y.y)\ b) \\
\longrightarrow & ((\lambda y.y)\ b)\ ((\lambda y.y)\ b) \\
\longrightarrow & b\ ((\lambda y.y)\ b) \\
\longrightarrow & bb
\end{array}
$$

Applicative order

$$
\begin{array}{ll}
(\lambda x.x(\lambda y.y))\ (\lambda z.(\lambda a.aa)(z\ b)) \\
\longrightarrow & (\lambda x.x(\lambda y.y))\ (\lambda z.(z\ b)\ (z\ b)) \\
\longrightarrow & (\lambda z.(z\ b)\ (z\ b))\ (\lambda y.y) \\
\longrightarrow & ((\lambda y.y)\ b)\ ((\lambda y.y)\ b) \\
\longrightarrow & b\ ((\lambda y.y)\ b) \\
\longrightarrow & bb
\end{array}
$$

Shortest reduction

$$
\begin{array}{ll}
(\lambda x.x(\lambda y.y))\ (\lambda z.(\lambda a.aa)\ (z\ b)) \\
\longrightarrow & (\lambda z.(\lambda a.aa)(z\ b))\ (\lambda y.y) \\
\longrightarrow & (\lambda a.aa)\ ((\lambda y.y)\ b) \\
\longrightarrow & (\lambda a.aa)\ b \\
\longrightarrow & bb
\end{array}
$$

Normalizing strategy

Property of *normal order* reduction

- If a term t does have a normal form then *normal order* reduction reaches this normal form

(proof in another chapter)

Such a reducion strategy is said to be *normalizing*

3 Confluence

Confluences

Diamond property

$$
\begin{tikzcd}[column sep=small]
& t \arrow[dl] \arrow[dr] & \\
t_1 & & t_2
\end{tikzcd}
\quad \text{implies } \exists u \text{ tq} \quad
\begin{tikzcd}[column sep=small]
t_1 \arrow[dr] & & t_2 \arrow[dl] \\
& u &
\end{tikzcd}
$$

Confluence

$$
\begin{tikzcd}[column sep=small]
& t \arrow[dl, "*"'] \arrow[dr, "*"] & \\
t_1 & & t_2
\end{tikzcd}
\quad \text{implies } \exists u \text{ tq} \quad
\begin{tikzcd}[column sep=small]
t_1 \arrow[dr, "*"] & & t_2 \arrow[dl, "*"'] \\
& u &
\end{tikzcd}
$$

The λ-calculus does not have the diamond property

$$
\begin{tikzcd}[column sep=small]
& (\lambda x.xx)\ ((\lambda y.y)\ a) \arrow[dl] \arrow[dr] & \\
(\lambda x.xx)\ a \arrow[ddr] & & ((\lambda y.y)\ a)\ ((\lambda y.y)\ a) \arrow[d] \\
& & {} \arrow[dl] \\
& aa &
\end{tikzcd}
$$

It is however confluent

Confluence of the λ-calculus

1. One can prove that the λ-calculus is *locally confluent*, which is:

$$
\begin{tikzcd}[column sep=small]
& t \arrow[dl] \arrow[dr] & \\
t_1 & & t_2
\end{tikzcd}
\quad \text{implique } \exists u \text{ tq} \quad
\begin{tikzcd}[column sep=small]
t_1 \arrow[dr, "*"] & & t_2 \arrow[dl, "*"'] \\
& u &
\end{tikzcd}
$$

2. Then one closes every opening diagram

$$
\begin{tikzcd}[column sep=small]
& & & t \arrow[dl] \arrow[dr] & & & \\
& & u_1 \arrow[dl] & & v_1 \arrow[dr] & & \\
& u_2 \arrow[dl] & & & & v_2 \arrow[dr] & \\
\dots & & & & & & \dots
\end{tikzcd}
$$

by repeated application of local confluence.

Counter-example: local confluence does not imply confluence

$$\begin{tikzcd} a & b \arrow[l] \arrow[r, bend left] & c \arrow[l, bend left] \arrow[r] & d \end{tikzcd}$$

This relation is locally confluent, but one cannot close the following diagram

$$\begin{tikzcd} & b \arrow[dl, "*"'] \arrow[dr, "*"] & \\ a & & d \end{tikzcd}$$

Why repeated local confluence is not a proof

$$\begin{tikzcd} & & & t \arrow[dl] \arrow[dr] & & & \\ & & u_1 \arrow[dl] \arrow[dr, "*"'] & & v_1 \arrow[dl, "*"] \arrow[dr] & & \\ & \dots \arrow[dl] & & w & & \dots \arrow[dr] & \\ u_k & & & & & & v_l \end{tikzcd}$$

No guarantee that the opening subdiagrams

$$\begin{tikzcd} & u_1 \arrow[dl, "*"'] \arrow[dr, "*"] & \\ u_k & & w \end{tikzcd} \quad \text{and} \quad \begin{tikzcd} & v_1 \arrow[dl, "*"'] \arrow[dr, "*"] & \\ w & & v_l \end{tikzcd}$$

are smaller than the first diagram!

Confluence of the λ-calculus, *for real*

Proof of Tait and Martin-Löf

Define a relation $\rightrightarrows_\beta$ which:

- is "between" $\longrightarrow_\beta$ and $\longrightarrow_\beta^*$
- has the diamond property

Idea: reduce several redexes in parallel in such a way that, for instance:

$$((\lambda y.y)a)\ ((\lambda y.y)a) \quad \rightrightarrows_\beta \quad aa$$

Proof of Tait and Martin-Löf: structure of the argument

- Since $\rightrightarrows_\beta$ has the diamond property, one deduces that $\rightrightarrows_\beta^*$ has the diamond property
- With $\longrightarrow_\beta \ \subseteq\ \rightrightarrows_\beta\ \subseteq\ \longrightarrow_\beta^*$, one deduces $\rightrightarrows_\beta^* = \longrightarrow_\beta^*$
- therefore $\longrightarrow_\beta^*$ has the diamond property
- and $\longrightarrow_\beta$ is confluent

Defining $\rightrightarrows_\beta$

Base case *"identity" reduction for variables*

$$\frac{}{x \ \rightrightarrows_\beta \ x}$$

Inductive cases *parallel reduction of subterms*

$$\frac{t \ \rightrightarrows_\beta \ t'}{\lambda x.t \ \rightrightarrows_\beta \ \lambda x.t'} \qquad \frac{t_1 \ \rightrightarrows_\beta \ t_1' \qquad t_2 \ \rightrightarrows_\beta \ t_2'}{t_1 \ t_2 \ \rightrightarrows_\beta \ t_1' \ t_2'}$$

Redexes *parallel reduction of the β-redex and its subterms*

$$\frac{t \ \rightrightarrows_\beta \ t' \qquad u \ \rightrightarrows_\beta \ u'}{(\lambda x.t) \ u \ \rightrightarrows_\beta \ t'\{x \leftarrow u'\}}$$

Example of parallel reduction

$$\dfrac{\dfrac{\dfrac{\dfrac{}{y \rightrightarrows_\beta y} \qquad \dfrac{\dfrac{}{z \rightrightarrows_\beta z}}{\lambda z.z \rightrightarrows_\beta \lambda z.z}}{(\lambda y.y) \ (\lambda z.z) \rightrightarrows_\beta \lambda z.z} \qquad \dfrac{}{x \rightrightarrows_\beta x}}{((\lambda y.y) \ (\lambda z.z)) \ x \ \rightrightarrows_\beta \ (\lambda z.z) \ x} \qquad \dfrac{\dfrac{}{w \rightrightarrows_\beta w} \qquad \dfrac{}{a \rightrightarrows_\beta a}}{(\lambda w.w)a \ \rightrightarrows_\beta \ a}}{(\lambda x.((\lambda y.y) \ (\lambda z.z)) \ x) \ ((\lambda w.w) \ a) \ \rightrightarrows_\beta \ (\lambda z.z) \ a}$$

Remark: one reduces only already-present redexes *the resulting term may contain "new" redexes*

Exercise: framing $\rightrightarrows_\beta$

Prove that

$$t \quad \rightrightarrows_\beta \quad t$$

Prove that

$$\longrightarrow_\beta \quad \subseteq \quad \rightrightarrows_\beta$$

Prove that

$$\rightrightarrows_\beta \quad \subseteq \quad \longrightarrow_\beta^*$$

Answer

- $t \rightrightarrows_\beta t$ by induction on t.
  - Case of a variable x. Then by definition $x \rightrightarrows_\beta x$.
  - Case of an application $t_1 \ t_2$. Induction hypotheses: $t_1 \rightrightarrows_\beta t_1$ and $t_2 \rightrightarrows_\beta t_2$. Then by application rule $t_1 \ t_2 \rightrightarrows_\beta t_1 \ t_2$.
  - Case of an abstraction $\lambda x.t$. Induction hypothesis: $t \rightrightarrows_\beta t$. Then by abstraction rule $\lambda x.t \rightrightarrows_\beta \lambda x.t$. □

- $\longrightarrow_\beta \subseteq \rightrightarrows_\beta$ by induction on $\longrightarrow_\beta$.
  - Case of β-reduction at the root $(\lambda x.t) \ u \longrightarrow_\beta t\{x \leftarrow u\}$. By previous result $t \rightrightarrows_\beta t$ and $u \rightrightarrows_\beta u$. Then by redex rule $(\lambda x.t) \ u \rightrightarrows_\beta t\{x \leftarrow u\}$.
  - Case of reduction at the left of an application $t \ u \longrightarrow_\beta t' \ u$ with $t \longrightarrow_\beta t'$. Induction hypothesis: $t \rightrightarrows_\beta t'$. Moreover, by the previous result $u \rightrightarrows_\beta u$. Then by application rule $t \ u \rightrightarrows_\beta t' \ u$.
  - Cases of reduction at the right of an application or under an abstraction similar.

- $\rightrightarrows_\beta \subseteq \longrightarrow^*_\beta$ by induction on $\rightrightarrows_\beta$.
  - Variable rule: $x \rightrightarrows_\beta x$. In particular $x \longrightarrow^0_\beta x$.
  - Abstraction rule: $\lambda x.t \rightrightarrows_\beta \lambda x.t'$ with $t \rightrightarrows_\beta t'$. Induction hypothesis: $t \longrightarrow^*_\beta t'$. Then by recurrence on the length of the sequence $\lambda x.t \longrightarrow^*_\beta \lambda x.t'$.
  - Application rule: $t_1\ t_2 \rightrightarrows_\beta t_1'\ t_2'$ with $t_1 \rightrightarrows_\beta t_1'$ and $t_2 \rightrightarrows_\beta t_2'$. Induction hypotheses: $t_1 \longrightarrow^*_\beta t_1'$ and $t_2 \longrightarrow^*_\beta t_2'$. Then $t_1\ t_2 \longrightarrow^*_\beta t_1'\ t_2 \longrightarrow^*_\beta t_1'\ t_2'$.
  - Redex rule: $(\lambda x.t)\ u \rightrightarrows_\beta t'\{x \leftarrow u'\}$ with $t \rightrightarrows_\beta t'$ and $u \rightrightarrows_\beta u'$. Induction hypotheses: $t \longrightarrow^*_\beta t'$ and $u \longrightarrow^*_\beta u'$. Then $(\lambda x.t)\ u \longrightarrow^*_\beta (\lambda x.t')\ u \longrightarrow^*_\beta (\lambda x.t')\ u' \longrightarrow_\beta t'\{x \leftarrow u'\}$.

Exercise: method of Tait and Martin-Löf

Prove that if $\longrightarrow$ has the diamond property, then its reflexive-transitive closure $\longrightarrow^*$ has the diamond property

Prove that if two relations $\longrightarrow$ and $\rightrightarrows$ are such that

$$\longrightarrow \ \subseteq \ \rightrightarrows \ \subseteq \ \longrightarrow^*$$

then their reflexive-transitive closures $\rightrightarrows^*$ and $\longrightarrow^*$ are equal

Answer

- Assume $\longrightarrow$ has the diamond property. If $b \longleftarrow a \longrightarrow^n c$ then there is d such that $d \longrightarrow^n d \longleftarrow c$ (proof by recurrence on the length n of the sequence on the right). Then, we prove that if $b^k \longleftarrow a \longrightarrow^n c$, then there is d such that $d \longrightarrow^n d^k \longleftarrow c$ (recurrence on k). Then $\longrightarrow^*$ has the diamond property.
- From $\longrightarrow \subseteq \rightrightarrows \subseteq \longrightarrow^*$ we deduce $\longrightarrow^* \subseteq \rightrightarrows^* \subseteq \longrightarrow^{**}$. Remark: $\longrightarrow^{**} = \longrightarrow^*$. Then $\longrightarrow^* \subseteq \rightrightarrows^* \subseteq \longrightarrow^*$, which means $\longrightarrow^* = \rightrightarrows^*$.

Diamond property for parallel reduction

If $s \leftleftarrows t \rightrightarrows r$ then there is u such that $s \rightrightarrows u \leftleftarrows r$

By induction on the derivation of $t \rightrightarrows_\beta r$

- Case $x \rightrightarrows x$. Then $s = x$, and we define $u = x$
- Case $\lambda x.t_0 \rightrightarrows \lambda x.r_0$ with $t_0 \rightrightarrows r_0$. Then $s = \lambda x.s_0$ with $s_0 \leftleftarrows t_0$.

  By induction hypothesis there is u_0 such that $s_0 \rightrightarrows u_0 \leftleftarrows r_0$.

  Therefore $\lambda x.s_0 \ \rightrightarrows\ \lambda x.u_0 \ \leftleftarrows\ \lambda x.r_0$
- Case $t_1 t_2 \ \rightrightarrows\ r_1 r_2$ with $t_1 \rightrightarrows r_1$ and $t_2 \rightrightarrows r_2$. Two cases for $s \leftleftarrows t_1 t_2$.
  - if $s = s_1\ s_2$ with $s_1 \leftleftarrows t_1$ and $s_2 \leftleftarrows t_2$

    by induction hypotheses there are u_1 and u_2 such that $s_1 \rightrightarrows u_1 \leftleftarrows r_1$ and $s_2 \rightrightarrows u_2 \leftleftarrows r_2$,

    therefore $s_1 s_2 \ \rightrightarrows\ u_1 u_2 \ \leftleftarrows\ r_1 r_2$
  - if $s = s_1\{x \leftarrow s_2\}$ with $t_1 = \lambda x.t_1'$ and $s_1 \leftleftarrows t_1'$ et $s_2 \leftleftarrows t_2$,

    then $r_1 = \lambda x.r_1'$ with $t_1' \rightrightarrows r_1'$ and by induction hypotheses there are u_1 and u_2 such that $s_1 \rightrightarrows u_1 \leftleftarrows r_1'$ et $s_2 \rightrightarrows u_2 \leftleftarrows r_2$,

    therefore $u_1\{x \leftarrow u_2\} \ \leftleftarrows\ (\lambda x.r_1')r_2$

    and we conclude if we can show that $s_1\{x \leftarrow s_2\} \ \rightrightarrows\ u_1\{x \leftarrow u_2\}$

    Lemma: if $a \rightrightarrows_\beta a'$ and $b \rightrightarrows_\beta b'$ then $a\{x \leftarrow b\} \rightrightarrows_\beta a'\{x \leftarrow b'\}$ *coming soon*
- Case $(\lambda x.t_1)t_2 \rightrightarrows r_1\{x \leftarrow r_2\}$ with $t_1 \rightrightarrows r_1$ et $t_2 \rightrightarrows r_2$. Two cases for $s \leftleftarrows (\lambda x.t_1)t_2$.
  - if $s = (\lambda x.s_1)s_2$ with $s_1 \leftleftarrows t_1$ and $s_2 \leftleftarrows t_2$ we conclude as above.
  - if $s = s_1\{x \leftarrow s_2\}$ with $s_1 \leftleftarrows t_1$ and $s_2 \leftleftarrows t_2$

    then by induction hypotheses there are u_1 and u_2 such that $s_1 \rightrightarrows u_1 \leftleftarrows r_1$ and $s_2 \rightrightarrows u_2 \leftleftarrows r_2$,

    and we conclude if we can show that $s_1\{x \leftarrow s_2\} \ \rightrightarrows\ u_1\{x \leftarrow u_2\} \ \leftleftarrows\ r_1\{x \leftarrow r_2\}$

    Same lemma

Lemma $\quad a \rightrightarrows_\beta a' \wedge b \rightrightarrows_\beta b' \quad \Longrightarrow \quad a\{x \leftarrow b\} \rightrightarrows_\beta a'\{x \leftarrow b'\}$

By induction on the derivation of $a \rightrightarrows_\beta a'$

- Case $y \rightrightarrows y$.

  Case on x and y.

  - If $x = y$, then $\quad x\{x \leftarrow b\} = b \rightrightarrows b' = x\{x \leftarrow b'\}$
  - If $x \neq y$, then $\quad y\{x \leftarrow b\} = y \rightrightarrows y = y\{x \leftarrow b'\}$

- Case $\lambda y.a_0 \rightrightarrows \lambda y.a_0'$ with $a_0 \rightrightarrows a_0'$.

  Then $(\lambda y.a_0)\{x \leftarrow b\} = \lambda y.(a_0\{x \leftarrow b\})$ and by induction hypothesis $a_0\{x \leftarrow b\} \rightrightarrows a_0'\{x \leftarrow b'\}$.

  Therefore $\lambda y.(a_0\{x \leftarrow b\}) \rightrightarrows \lambda y.(a_0'\{x \leftarrow b'\}) = (\lambda x.a_0')\{x \leftarrow b'\}$

- Case $a_1 a_2 \rightrightarrows a_1' a_2'$ with $a_1 \rightrightarrows a_1'$ et $a_2 \rightrightarrows a_2'$.

  Then $(a_1 a_2)\{x \leftarrow b\} = (a_1\{x \leftarrow b\})(a_2\{x \leftarrow b\})$ and $\quad (a_1' a_2')\{x \leftarrow b'\} = (a_1'\{x \leftarrow b'\})(a_2'\{x \leftarrow b'\})$

  and by induction hypotheses $a_1\{x \leftarrow b\} \rightrightarrows a_1'\{x \leftarrow b'\}$ and $a_2\{x \leftarrow b\} \rightrightarrows a_2'\{x \leftarrow b'\}$.

  Therefore $(a_1 a_2)\{x \leftarrow b\} \rightrightarrows (a_1' a_2')\{x \leftarrow b'\}$

- Case $(\lambda y.a_1)a_2 \rightrightarrows a_1'\{y \leftarrow a_2'\}$ with $a_1 \rightrightarrows a_1'$ and $a_2 \rightrightarrows a_2'$.

  Then $((\lambda y.a_1)a_2)\{x \leftarrow b\} = (\lambda y.a_1\{x \leftarrow b\})(a_2\{x \leftarrow b\})$.

  By induction hypotheses we have $a_1\{x \leftarrow b\} \rightrightarrows_\beta a_1'\{x \leftarrow b'\}$ and $a_2\{x \leftarrow b\} \rightrightarrows_\beta a_2'\{x \leftarrow b'\}$.

  Therefore $(\lambda y.a_1\{x \leftarrow b\})(a_2\{x \leftarrow b\}) \quad \rightrightarrows (a_1'\{x \leftarrow b'\})\{y \leftarrow a_2'\{x \leftarrow b'\}\}$.

  With α-renaming we can choose $y \neq x$ and $y \notin \mathrm{fv}(b')$, therefore by substitution lemma $(a_1'\{x \leftarrow b'\})\{y \leftarrow a_2'\{x \leftarrow b'\}\} = (a_1'\{y \leftarrow a_2'\})\{x \leftarrow b'\}$.

Substitution lemma

If $\quad x \neq y \quad$ and $\quad x \notin \mathrm{fv}(v) \quad$ then

$$t\{x \leftarrow u\}\{y \leftarrow v\} \quad = \quad t\{y \leftarrow v\}\{x \leftarrow u\{y \leftarrow v\}\}$$

Proof by induction on t

- Case of a variable.

  - Case $t = x$. Then $x\{x \leftarrow u\}\{y \leftarrow v\} = u\{y \leftarrow v\}$ and $x\{y \leftarrow v\}\{x \leftarrow u\{y \leftarrow v\}\} = x\{x \leftarrow u\{y \leftarrow v\}\} = u\{y \leftarrow v\}$
  - Case $t = y$. Then $y\{x \leftarrow u\}\{y \leftarrow v\} = y\{y \leftarrow v\} = v$ and $y\{y \leftarrow v\}\{x \leftarrow u\{y \leftarrow v\}\} = v\{x \leftarrow u\{y \leftarrow v\}\} = v$
  - Case $t = z$, otherwise. Then $z\{x \leftarrow u\}\{y \leftarrow v\} = z$ and $s\{y \leftarrow v\}\{x \leftarrow u\{y \leftarrow v\}\} = z$

- Case of an application $t_1\ t_2$. Assume $t_1\{x \leftarrow u\}\{y \leftarrow v\} = t_1\{y \leftarrow v\}\{x \leftarrow u\{y \leftarrow v\}\}$ and $t_2\{x \leftarrow u\}\{y \leftarrow v\} = t_2\{y \leftarrow v\}\{x \leftarrow u\{y \leftarrow v\}\}$ Then

$$\begin{array}{ll} & (t_1\ t_2)\{x \leftarrow u\}\{y \leftarrow v\} \\ = & (t_1\{x \leftarrow u\}\ t_2\{x \leftarrow u\})\{y \leftarrow v\} \\ = & t_1\{x \leftarrow u\}\{y \leftarrow v\}\ t_2\{x \leftarrow u\}\{y \leftarrow v\} \\ = & t_1\{y \leftarrow v\}\{x \leftarrow u\{y \leftarrow v\}\}\ t_2\{y \leftarrow v\}\{x \leftarrow u\{y \leftarrow v\}\} \\ = & (t_1\{y \leftarrow v\}\ t_2\{y \leftarrow v\})\{x \leftarrow u\{y \leftarrow v\}\} \\ = & (t_1\ t_2)\{y \leftarrow v\}\{x \leftarrow u\{y \leftarrow v\}\} \end{array}$$

- Case of an abstraction $\lambda z.t$. Assume $t\{x \leftarrow u\}\{y \leftarrow v\} = t\{y \leftarrow v\}\{x \leftarrow u\{y \leftarrow v\}\}$ and by Barendregt convention $z \neq x$ and $z \neq y$ and $z \notin \mathsf{fv}(u)$ and $z \notin \mathsf{fv}(v)$ (and $z \notin \mathsf{fv}(u\{y \leftarrow v\})$) Then

$$\begin{array}{ll} & (\lambda z.t)\{x \leftarrow u\}\{y \leftarrow v\} \\ = & (\lambda z.(t\{x \leftarrow u\}))\{y \leftarrow v\} \\ = & \lambda z.(t\{x \leftarrow u\}\{y \leftarrow v\}) \\ = & \lambda z.(t\{y \leftarrow v\}\{x \leftarrow u\{y \leftarrow v\}\}) \\ = & (\lambda z.(t\{y \leftarrow v\}))\{x \leftarrow u\{y \leftarrow v\}\} \\ = & (\lambda z.t)\{y \leftarrow v\}\{x \leftarrow u\{y \leftarrow v\}\} \end{array}$$

Corollary: Church-Rosser theorem

If

$$t_1 \quad =_\beta \quad t_2$$

then there is u such that

$$t_1 \quad \longrightarrow_\beta^* \quad u \qquad \text{et} \qquad t_2 \quad \longrightarrow_\beta^* \quad u$$

Consequences

- if t has a normal form n, then $t \longrightarrow^* n$
- any λ-term can has only one normal form
- if two normal forms n and m are syntactically different, then $n \neq_\beta m$

4 Confluence: another proof

Strip lemma

Property of the λ-calculus:

$$\begin{tikzcd} & t \arrow[dl] \arrow[dr, "*"] & \\ t_1 & & t_2 \end{tikzcd} \qquad \text{implies } \exists u \text{ tq} \qquad \begin{tikzcd} t_1 \arrow[dr, "*"] & & t_2 \arrow[dl, "*"'] \\ & u & \end{tikzcd}$$

Idea: identify the redex R that is reduced by the step $t \longrightarrow t_1$. Then track what remains of R in t_2, and reduce every occurrence. (proof later in the chapter)

The strip lemma implies confluence

If $t_1 \longrightarrow^* t_n$ and $t_1 \longrightarrow^* u_1$, then there exists u_n such that $t_n \longrightarrow^* u_n$ and $u_1 \longrightarrow^* u_n$.

Proof by recurrence on the length of the reduction sequence $t_1 \longrightarrow t_2 \longrightarrow t_3 \longrightarrow \ldots \longrightarrow t_n$. Each step uses the strip lemma to make one "strip" in the following diagram.

$$\begin{tikzcd} t_1 \arrow[d] \arrow[r, "*"] & u_1 \arrow[d, dashed, "*"] \\ t_2 \arrow[d] \arrow[r, dashed, "*"] & u_2 \arrow[d, dashed, "*"] \\ t_3 \arrow[d, dotted, no head] \arrow[r, dashed, "*"] & u_3 \arrow[d, dotted, no head] \\ t_n \arrow[r, dashed, "*"] & u_n \end{tikzcd}$$

Residuals

Consider a β-reduction step $t \xrightarrow{p} t'$ of a redex $(\lambda x.u)v$ at position p in t. Positions of t can be tracked in t'. Let q be a position in t, and define $D(q)$ the set of *descendant positions* of q in t'.

- Positions outside of $(\lambda x.u)v$ still exist, unmodified, in t'.

$$D(q) = \{q\} \qquad \text{if } p \text{ is not a prefix of } q$$

- The positions p of the redex $(\lambda x.u)v$ and $p.1$ of the abstraction $\lambda x.u$ have no descendants.
- Every part of u still exist in $u\{x \longleftarrow v\}$. The positions however are slightly modified between t and t' since an application and an abstraction disappeared.

$$D(p.1.0.q) = \{p.q\}$$

(We could argue on what happens to the occurrences of x. Here we choose to keep them in the descendant relation.)

- Every part of v exist in $u\{x \longleftarrow v\}$ in each substituted occurrence of v (whose number can be arbitrary). The new position of each occurrence of v in $u\{x \longleftarrow v\}$ corresponds to the position of an occurrence of x in u.

$$D(p.2.q) = \{p.p_x.q \mid p_x \text{ position of an occurrence of } x \text{ in } u\}$$

A redex R' at position q' in t' is a *residual* of a redex R at position q in t after $t \xrightarrow{p} t'$ if $q' \in D(q)$.

Marked λ-terms

A simple solution to track the residuals of a set of redexes in a given source term is to add some "marks" in our λ-terms. For this we introduce an extension $\underline{\Lambda}$ of the syntax, where λ-abstractions can be underlined. This extended grammar is:

$$\begin{array}{rcll} t & ::= & x & \text{variable} \\ & \mid & t\ t & \text{application} \\ & \mid & \lambda x.t & \text{ordinary abstraction} \\ & \mid & \underline{\lambda} x.t & \text{marked abstraction} \end{array}$$

The β-reduction rule applies for both ordinary λ's and marked $\underline{\lambda}$'s.

$$\begin{array}{rcl} (\lambda x.t)\ u & \longrightarrow_\beta & t\{x \longleftarrow u\} \\ (\underline{\lambda} x.t)\ u & \longrightarrow_\beta & t\{x \longleftarrow u\} \end{array}$$

Free variables, variable renaming and substitution are also extended to treat marked $\underline{\lambda}$'s as ordinary λ's.

$$\begin{array}{rcl} \mathsf{fv}(x) & = & \{x\} \\ \mathsf{fv}(t\ u) & = & \mathsf{fv}(t) \cup \mathsf{fv}(u) \\ \mathsf{fv}(\lambda x.t) & = & \mathsf{fv}(t) \setminus \{x\} \\ \mathsf{fv}(\underline{\lambda} x.t) & = & \mathsf{fv}(t) \setminus \{x\} \end{array}$$

$$\begin{array}{rcll} x\{x \longleftarrow v\} & = & v & \\ y\{x \longleftarrow v\} & = & y & \text{if } y \neq x \\ (t\ u)\{x \longleftarrow v\} & = & t\{x \longleftarrow v\}\ u\{x \longleftarrow v\} & \\ (\lambda y.t)\{x \longleftarrow v\} & = & \lambda y.(t\{x \longleftarrow v\}) & \text{if } y \neq x \text{ and } y \notin \mathsf{fv}(v) \\ (\underline{\lambda} y.t)\{x \longleftarrow v\} & = & \underline{\lambda} y.(t\{x \longleftarrow v\}) & \text{if } y \neq x \text{ and } y \notin \mathsf{fv}(v) \end{array}$$

$$\begin{array}{rcll} \lambda x.t & =_\alpha & \lambda y.(t\{x \longleftarrow y\}) & \text{if } y \notin \mathsf{fv}(t) \\ \underline{\lambda} x.t & =_\alpha & \underline{\lambda} y.(t\{x \longleftarrow y\}) & \text{if } y \notin \mathsf{fv}(t) \end{array}$$

Removing marks

Let $t \in \underline{\Lambda}$ be a marked term. Define $|t|$ the ordinary λ-term obtained by removing all the marks in t.

$$\begin{array}{rcl} |x| & = & x \\ |t\ u| & = & |t|\ |u| \\ |\lambda x.t| & = & \lambda x.|t| \\ |\underline{\lambda} x.t| & = & \lambda x.|t| \end{array}$$

We can trivially check that the marks do not interfere with reduction.

Lemma 1.

$$\text{For any } t, t' \in \underline{\Lambda}, \qquad t \longrightarrow t' \qquad \text{iff} \qquad |t| \longrightarrow |t'|$$

Diagrammatically:

$$\begin{tikzcd} t \arrow[r] \arrow[d, "||"'] & t' \arrow[d, "||"] \\ u \arrow[r, dashed] & u' \end{tikzcd} \qquad \begin{tikzcd} t \arrow[r, dashed] \arrow[d, "||"'] & t' \arrow[d, "||"] \\ u \arrow[r] & u' \end{tikzcd}$$

(solid arrows are assumptions, dashed arrow are deduced)

Reducing marked redexes

Let $t \in \underline{\Lambda}$ be a marked term. Define $\varphi(t)$ the term obtained by reducing all marked redexes in t (and removing any remaining mark).

$$\begin{array}{rcll} \varphi((\underline{\lambda} x.t)\ u) & = & (\varphi(t))\{x \leftarrow \varphi(u)\} & \\ \varphi(x) & = & x & \\ \varphi(t\ u) & = & \varphi(t)\ \varphi(u) & \qquad \text{if } t \text{ does not start with } \underline{\lambda} \\ \varphi(\lambda x.t) & = & \lambda x.\varphi(t) & \\ \varphi(\underline{\lambda} x.t) & = & \lambda x.\varphi(t) & \end{array}$$

Lemma 2. Commutation of φ and substitution.

$$\text{For any } t, u \in \underline{\Lambda}, \qquad \varphi(t\{x \leftarrow u\}) = \varphi(t)\{x \leftarrow \varphi(u)\}$$

Proof by induction on t.

Lemma 3. Commutation of φ and β-reduction.

$$\text{For any } t, t' \in \underline{\Lambda}, \text{ if} \qquad t \longrightarrow t' \qquad \text{then} \qquad \varphi(t) \longrightarrow \varphi(t')$$

Diagrammatically:

$$\begin{tikzcd} t \arrow[r] \arrow[d, "\varphi"'] & t' \arrow[d, "\varphi"] \\ u \arrow[r, dashed, "*"'] & u' \end{tikzcd}$$

Proof by induction on the derivation of $t \longrightarrow t'$, using lemma 2.

Lemma 4. The simultaneous reduction performed by φ can be realized with ordinary β-reduction.

$$\text{For any } t \in \underline{\Lambda}, \qquad |t| \longrightarrow_\beta^* \varphi(t)$$

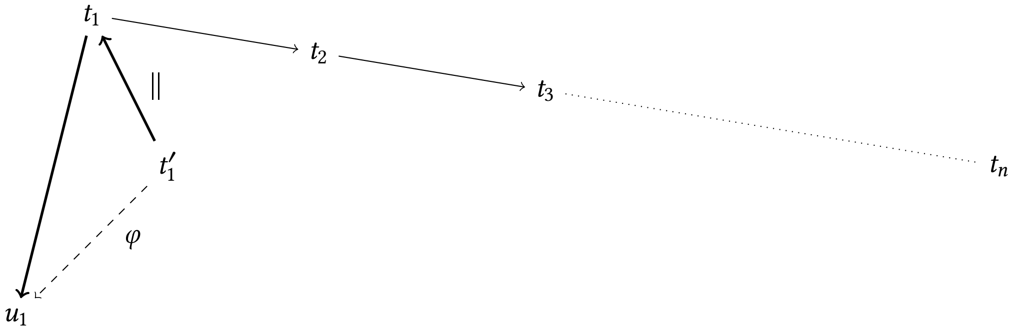
Diagrammatically:

$$\begin{tikzcd} & t \arrow[dl, "||"'] \arrow[dr, "\varphi"] & \\ u \arrow[rr, dashed, "*"'] & & v \end{tikzcd}$$

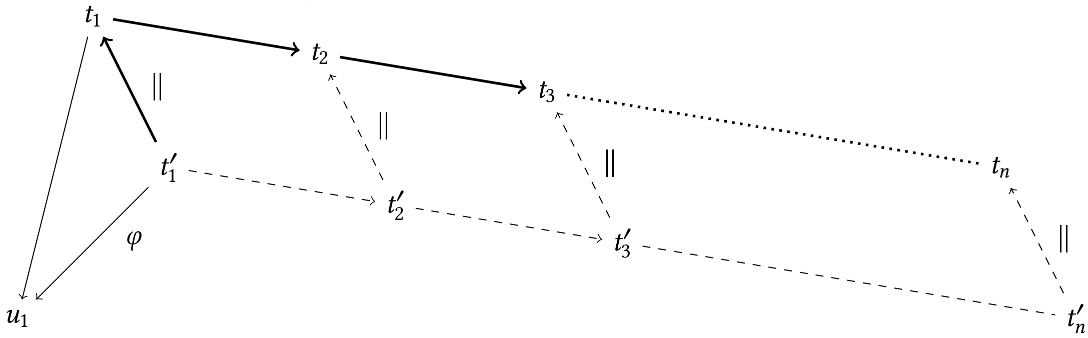
(Proving this lemma is homework!)

Proof of the strip lemma

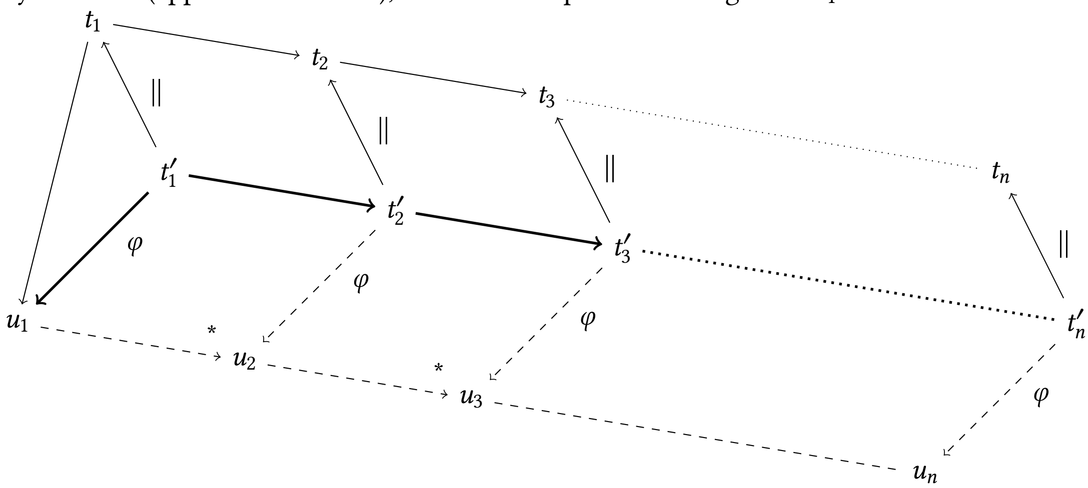
Consider the reduction $t_1 \longrightarrow_\beta u_1$ of a single β-redex $R = (\lambda x.a)\ b$, and a sequence $t_1 \longrightarrow t_2 \longrightarrow t_3 \longrightarrow \ldots \longrightarrow t_n$. Let t_1' be the term obtained from t_1 by marking the λ in R. First remark that $\varphi(t_1')$ is precisly the term u_1 obtained by reducing R in t_1.



Since marks do not interfere with reduction ($n-1$ applications of lemma 1), we can reproduce the sequence $t_1 \longrightarrow^* t_n$ starting from t_1'.



Then by lemma 3 (applied $n-1$ times), we build a sequence starting from u_1.



Finally, by lemma 4 on the last triangle formed with the terms t_n, t_n', u_n, we deduce a reduction sequence from $t_n = |t_n'|$ to $u_n = \varphi(t_n')$.