Lambda calculs et catégories

Paul-André Melliès

Master Parisien de Recherche en Informatique

Ecole Normale Supérieure
Synopsis of the lecture

1 – The lambda-calculus

2 – The simply-typed lambda-calculus

3 – Elements of rewriting theory

4 – The lambda-calculus with explicit substitutions
First part

Lambda-calculus

The calculus of functions
The pure $\lambda$-calculus

Terms

\[ M ::= x \mid MN \mid \lambda x. M \]

The $\beta$-reduction:

\[ (\lambda x. M) N \rightarrow M[x := N] \]

The $\eta$-expansion:

\[ M \rightarrow \lambda x. (M x) \]

Remark: every term is considered up to renaming $\equiv_\alpha$ of the bound variables, typically:

\[ \lambda x. \lambda y. x \equiv_\alpha \lambda z. \lambda y. z \]
Occurrences

The set of occurrences of a \( \lambda \)-term \( M \) is defined by induction:

\[
\begin{align*}
\text{occ}(x) &= \{ \varepsilon \} \\
\text{occ}(MN) &= \{ \varepsilon \} \cup \{ 1 \cdot o \mid o \in \text{occ}(M) \} \cup \{ 2 \cdot o \mid o \in \text{occ}(N) \} \\
\text{occ}(\lambda x.M) &= \{ \varepsilon \} \cup \{ 1 \cdot o \mid o \in \text{occ}(M) \}
\end{align*}
\]

Note that every occurrence of the \( \lambda \)-term \( M \) is labelled by

\[
\begin{align*}
&\text{an application node } \text{App} \\
&\text{a binder } \lambda x \\
&\text{a variable } x
\end{align*}
\]
Free variables

The set of free variables of a $\lambda$-term is defined by induction:

- $FV(x) = \{ x \}$
- $FV(MN) = FV(M) \cup FV(N)$
- $FV(\lambda x.M) = FV(M) \setminus \{ x \}$

Every occurrence of a variable $x$ in a $\lambda$-term is

- either free
- or bound by a binder $\lambda x$ above it in the $\lambda$-term.
Church-Rosser theorem

Also called confluence theorem.

Given two \( \beta \)-rewriting paths

\[
f : M \rightarrow^* P \quad g : M \rightarrow^* Q
\]

there exists a \( \lambda \)-term \( N \) and two \( \beta \)-rewriting paths \( f' \) and \( g' \) completing the diagram as

\[
\begin{array}{c}
M \\
\downarrow f \quad \downarrow g \\
P \quad Q \\
\uparrow g' \quad \uparrow f'
\end{array}
\]
The simply-typed $\lambda$-calculus

It is possible to **type** the expressions of the $\lambda$-calculus using simple types $A, B$ constructed by the grammar:

\[
A, B ::= \alpha \mid A \Rightarrow B.
\]

A **typing context** $\Gamma$ is a finite sequence

\[
\Gamma = (x_1: A_1, ..., x_n: A_n)
\]

where each $x_i$ is a variable and each $A_i$ is a simple type.

A **sequent** is a triple

\[
x_1 : A_1, ..., x_n : A_n \vdash P : B
\]

where

\[
x_1 : A_1, ..., x_n : A_n
\]

is a typing context, $P$ is a $\lambda$-term and $B$ is a simple type.
The simply-typed $\lambda$-calculus

<table>
<thead>
<tr>
<th>Rule</th>
<th>Context</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable</td>
<td>$x : A \vdash x : A$</td>
</tr>
<tr>
<td>Abstraction</td>
<td>$\Gamma, x : A \vdash P : B \quad \Delta \vdash Q : A$</td>
</tr>
<tr>
<td>Application</td>
<td>$\Gamma, \Delta \vdash PQ : B$</td>
</tr>
<tr>
<td>Weakening</td>
<td>$\Gamma \vdash P : B$</td>
</tr>
<tr>
<td>Contraction</td>
<td>$\Gamma, x : A, y : A \vdash P : B \quad \Delta \vdash P[x, y \leftarrow z] : B$</td>
</tr>
<tr>
<td>Exchange</td>
<td>$\Gamma, x : A, y : B, \Delta \vdash P : C \quad \Gamma, y : B, x : A, \Delta \vdash P : C$</td>
</tr>
</tbody>
</table>
Subject reduction

A $\lambda$-term $P$ is simply typed when there exists a sequent

$$\Gamma \vdash P : A$$

which may be obtained by a derivation tree.

One establishes that the set of simply typed $\lambda$-terms is closed under $\beta$-réduction:

**Subject Reduction:**

If $\Gamma \vdash P : A$ and $P \rightarrow^\beta Q$, then $\Gamma \vdash Q : A$. 
Strong normalization

A $\lambda$-term $P$ is strongly normalizing when there exists no infinite sequence of $\beta$-reductions:

$$P \rightarrow^\beta P_1 \rightarrow^\beta P_2 \rightarrow^\beta \cdots \rightarrow^\beta P_n \rightarrow^\beta \cdots$$

Strong normalization:

Every simply typed $\lambda$-term $P$ is strongly normalizing.

In particular, the $\lambda$-term $\Delta\Delta$ loops:

$$\Delta\Delta \rightarrow^\beta \Delta\Delta \rightarrow^\beta \cdots$$

is not simply typed.
**Curry-Howard (1)**

<table>
<thead>
<tr>
<th>Rule</th>
<th>Logic Formulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable</td>
<td>( A \vdash A )</td>
</tr>
<tr>
<td>Abstraction</td>
<td>( \Gamma, A \vdash B ) [ \frac{}{\Gamma \vdash A \Rightarrow B} ]</td>
</tr>
<tr>
<td>Application</td>
<td>( \Gamma \vdash A \Rightarrow B, \Delta \vdash A ) [ \frac{}{\Gamma, \Delta \vdash B} ]</td>
</tr>
<tr>
<td>Weakening</td>
<td>( \Gamma \vdash B ) [ \frac{}{\Gamma, \Delta \vdash B} ]</td>
</tr>
<tr>
<td>Contraction</td>
<td>( \Gamma, A, A \vdash B ) [ \frac{}{\Gamma, A \vdash B} ]</td>
</tr>
<tr>
<td>Exchange</td>
<td>( \Gamma, A, B, \Delta \vdash C ) [ \frac{}{\Gamma, B, A, \Delta \vdash C} ]</td>
</tr>
</tbody>
</table>
Curry-Howard (1)

**Variable**

\[ x : A \vdash x : A \]

**Abstraction**

\[ \Gamma, x : A \vdash P : B \]

\[ \Gamma \vdash \lambda x. P : A \Rightarrow B \]

**Application**

\[ \Gamma \vdash P : A \Rightarrow B \quad \Delta \vdash Q : A \]

\[ \Gamma, \Delta \vdash PQ : B \]

**Weakening**

\[ \Gamma \vdash P : B \]

\[ \Gamma, x : A \vdash P : B \]

**Contraction**

\[ \Gamma, x : A, y : A \vdash P : B \]

\[ \Gamma, z : A \vdash P[x, y \leftarrow z] : B \]

**Exchange**

\[ \Gamma, x : A, y : B, \Delta \vdash P : C \]

\[ \Gamma, y : B, x : A, \Delta \vdash P : C \]
Algebraic Church-Rosser Theorem

Given two $\beta$-rewriting paths

\[ f : M \rightarrow^* P \quad g : M \rightarrow^* Q \]

there exists a $\lambda$-term $N$ and two $\beta$-rewriting paths $f'$ and $g'$ completing the diagram as

where $\sim$ denotes the permutation equivalence on rewriting paths.

Theorem established by Jean-Jacques Lévy in 1978
**Redex**

**Definition.** A $\beta$-redex is a pair $(M, o)$ consisting of

- a $\lambda$-term $M$
- an occurrence of the $\lambda$-term $M$ such that

$$M_{|o} = (\lambda x.P) Q$$

is a $\beta$-reduction pattern.
The two redexes $u : M \to P$ and $v : N \to Q$ are disjoint.
The redex \( u \) erases the redex \( v : M \rightarrow P \).
The redex $u$ duplicates the redex $v : M \rightarrow P$. 
Rewriting paths modulo permutations

An important problem of rewriting theory: compare the several paths which rewrite a $\lambda$-term $P$ into its normal form $Q$.

Corollary

Every two rewriting paths to the normal form

$$f, g : P \rightarrow Q$$

are equal modulo a series of redex permutations.
A 2-dimensional hole

The two redexes \( u \) and \( v \) are not equivalent modulo permutation.
The 2-dimensional hole continued

The two paths $u \cdot w$ and $v \cdot w$ are equivalent modulo permutation.
Geometry of rewriting

A standardization theorem will be established in the course
The \( \lambda \)-calculus with de Bruijn indices

Variable

\[ \Gamma, A \vdash 1 : A \]

Abstraction

\[ \frac{\Gamma, A \vdash P : B}{\Gamma \vdash \lambda P : A \Rightarrow B} \]

Application

\[ \frac{\Gamma \vdash P : A \Rightarrow B \quad \Gamma \vdash Q : A}{\Gamma \vdash PQ : B} \]

Weakening

\[ \frac{\Gamma \vdash P : B}{\Gamma, A \vdash P [\uparrow] : B} \]

where \( P [\uparrow] \) denotes the \( \lambda \)-term \( P \) where each free variable has been incremented.
The $\lambda$-calculus with explicit substitutions

Terms \hspace{1cm} M ::= 1 \mid MN \mid \lambda M \mid M[s]

Substitutions \hspace{1cm} s ::= id \mid \uparrow \mid M \cdot s \mid s \circ t

Key idea: replace the $\beta$-rule of the $\lambda$-calculus

$$(\lambda x. M) N \rightarrow M[x := N]$$

by the Beta-rule of the $\lambda\sigma$-calculus

$$(\lambda M) N \rightarrow M[N \cdot id]$$

where the substitution is explicit – and thus similar to a closure.
The eleven rewriting rules of the $\lambda\sigma$-calculus

<table>
<thead>
<tr>
<th>Rule</th>
<th>Pattern</th>
<th>Reduces To</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beta</td>
<td>$(\lambda M)N$</td>
<td>$M[N \cdot id]$</td>
</tr>
<tr>
<td>App</td>
<td>$(MN)[s]$</td>
<td>$M[s]N[s]$</td>
</tr>
<tr>
<td>Abs</td>
<td>$(\lambda M)[s]$</td>
<td>$\lambda (M[1 \cdot (s \circ \uparrow)])$</td>
</tr>
<tr>
<td>Clos</td>
<td>$M[s][t]$</td>
<td>$M[s \circ t]$</td>
</tr>
<tr>
<td>VarCons</td>
<td>$1[M \cdot s]$</td>
<td>$M$</td>
</tr>
<tr>
<td>VarId</td>
<td>$1[id]$</td>
<td>$1$</td>
</tr>
<tr>
<td>Map</td>
<td>$(M \cdot s) \circ t$</td>
<td>$M[t] \cdot (s \circ t)$</td>
</tr>
<tr>
<td>IdL</td>
<td>$id \circ s$</td>
<td>$s$</td>
</tr>
<tr>
<td>Ass</td>
<td>$(s_1 \circ s_2) \circ s_3$</td>
<td>$s_1 \circ (s_2 \circ s_3)$</td>
</tr>
<tr>
<td>ShiftCons</td>
<td>$\uparrow \circ (M \cdot s)$</td>
<td>$s$</td>
</tr>
<tr>
<td>ShiftId</td>
<td>$\uparrow \circ id$</td>
<td>$\uparrow$</td>
</tr>
</tbody>
</table>
The eleven critical pairs of the $\lambda\sigma$-calculus

<table>
<thead>
<tr>
<th>Critical Pair</th>
<th>Rule</th>
<th>Left Hand Side</th>
<th>Right Hand Side</th>
</tr>
</thead>
<tbody>
<tr>
<td>$App + Beta$</td>
<td>$\xleftarrow{App}$</td>
<td>$\lambda M<a href="N%5Bs%5D">s</a>$</td>
<td>$\lambda M)(N)[s]$</td>
</tr>
<tr>
<td>$Clos + App$</td>
<td>$\xleftarrow{Clos}$</td>
<td>$(MN)[s \circ t]$</td>
<td>$(MN)[s][t]$</td>
</tr>
<tr>
<td>$Clos + Abs$</td>
<td>$\xleftarrow{Clos}$</td>
<td>$(\lambda M)[s \circ t]$</td>
<td>$(\lambda M)[s][t]$</td>
</tr>
<tr>
<td>$Clos + VarId$</td>
<td>$\xleftarrow{Clos}$</td>
<td>$1[id \circ s]$</td>
<td>$1[id][s]$</td>
</tr>
<tr>
<td>$Clos + VarCons$</td>
<td>$\xleftarrow{Clos}$</td>
<td>$1[(M \cdot s) \circ t]$</td>
<td>$1[M \cdot s][t]$</td>
</tr>
<tr>
<td>$Clos + Clos$</td>
<td>$\xleftarrow{Clos}$</td>
<td>$M[s][t \circ t']$</td>
<td>$M[s][t][t']$</td>
</tr>
<tr>
<td>$Ass + Map$</td>
<td>$\xleftarrow{Ass}$</td>
<td>$(M \cdot s) \circ (t \circ t')$</td>
<td>$((M \cdot s) \circ t) \circ t'$</td>
</tr>
<tr>
<td>$Ass + IdL$</td>
<td>$\xleftarrow{Ass}$</td>
<td>$id \circ (s \circ t)$</td>
<td>$(id \circ s) \circ t$</td>
</tr>
<tr>
<td>$Ass + ShiftId$</td>
<td>$\xleftarrow{Ass}$</td>
<td>$\uparrow \circ (id \circ s)$</td>
<td>$(\uparrow \circ id) \circ s$</td>
</tr>
<tr>
<td>$Ass + ShiftCons$</td>
<td>$\xleftarrow{Ass}$</td>
<td>$\uparrow \circ ((M \cdot s) \circ t)$</td>
<td>$(\uparrow \circ (M \cdot s)) \circ t$</td>
</tr>
<tr>
<td>$Ass + Ass$</td>
<td>$\xleftarrow{Ass}$</td>
<td>$(s \circ s') \circ (t \circ t')$</td>
<td>$((s \circ s') \circ t) \circ t'$</td>
</tr>
</tbody>
</table>
A dangerous critical pair

This critical pair leads to a counter-example to strong normalization of the simply-typed $\lambda\sigma$-calculus.