Preuves et Programmes

Philippe Audebaud

Fall 2017

Contents

1 (Pure) Lambda Calculus

1.1 Computing with functions

Main question : How do we do maths ? To answer that we can see:

- Having structures: We want to manipulate numbers, spaces (of points, vectors and functions). (Eidenberg-Mac Lane, Category Theory, 1942). Those things will be the types.
- Build, explore and transform structures (Church's λ-calculus, 1930). Theses things with be programs, proofs.
- Compare «stuff», with equality for instance (equality comes first !) (Voevaski, 2006, Algebraic Topology)
- Provide a framework (rules) to reasoning on theses previous things. This points is somehow the first point that comes when we want to do Maths.

One can notice that it is, in fact, basically, recent research.

1.2 Informal introduction to Church's λ-calculus.

We take a function : $f : \begin{array}{ccc} A & \to & B \\ x & \mapsto & e \end{array}$. Given $a \in A$, $f(a)$ is the image, we replace (kind of) each occurrence of x in e by an occurrence of a. Then we get $e\langle a/x\rangle$. We say that we apply f to a. We can denote $f\ a$ when there is no ambiguity. In terms of λ-calculus, we can define f like that: $f \equiv \lambda x.e$ where $\equiv$ is the definitional equality. Then $f\ a \equiv (\lambda x.e)\ a$.

For the syntax, if there is no parenthesis, all what is after the . is part of the body of the function.

Example 1.1. *Here are some examples*

- *$\lambda x.x$ is the function $x \mapsto x$ is the identity function.*
- *If x and y are two distinct variables the function $\lambda x.y$ as no effect since $y\langle a/x\rangle \equiv y$ and then $(\lambda x.y)\ a \equiv y$.*

To «compute» terms we have to introduce a kind of reduction $\to_\beta$, a binary relation on λ-terms. For instance, $(\lambda x.a)b \to_\beta a\langle b/x\rangle$.

Some terms also seems to be «equivalent». Then we may need some λ-equivalence, let say $=_\alpha$. For instance we would like to say when $y \neq x$ is a fresh variable that $\lambda x.e =_\alpha \lambda y.e\langle y/x\rangle$. To state $\lambda x.a =_\alpha \lambda y.b$, we might need some fresh variable z such that $a\langle z/x\rangle =_\alpha b\langle z/y\rangle$.

To be completed next week.

1.3 A toolbox on λ-calculus.

Let $\mathcal{X}$ be a denumerable set of variables range over $x, y, z, \ldots$.

Definition 1.2 (λ-terms)**.** *A λ-term e is generated by the following grammar $e ::= x \in \mathcal{X} | \lambda x.e | ee$. The set of $lambda$ terms is denoted Λ.*

Definition 1.3 (Free variable)**.** *The set of free variable in a term e, denoted $FV(e)$ is defined inductively by:*

- *if $e = x \in \mathcal{X}$ then $FV(x) \equiv \{x\}$.*
- *if $e = \lambda x.a$ then $FV(e) \equiv FV(a) \backslash \{x\}$.*
- *if $e = a\ b$ then $FV(e) \equiv FV(a) \cup FV(b)$.*
- *if e is closed $FV(e) = \emptyset$.*

Example 1.4.

- $e \equiv \lambda x.x$ *then* $FV(e) = \emptyset$.
- $e \equiv \lambda x.y$ *then* $FV(e) = \{y\}$.

Definition 1.5 (Subsitution). *Given* $x \in \mathcal{X}, a \in \Lambda$ *the substitution of (all the) occurrences of* x *in* e *by* a*, denoted* $e\langle a/x\rangle$ *is:*

- *if* $e \equiv y \in \mathcal{X}\backslash\{x\}$ *then* $y\langle a/x\rangle \equiv y$ *otherwise* $x\langle a/x\rangle \equiv a$.
- $(\lambda y.e)\langle a/x\rangle \equiv \lambda y.e\langle a/x\rangle$
- $(e\ f)\langle a/x\rangle \equiv e\langle a/x\rangle\ f\langle a/x\rangle$.

Definition 1.6 (The $\rightarrow_\beta$ relation). *The binary relation* $\rightarrow_\beta$ *over* Λ *is* $\rightarrow_\beta \subseteq \Lambda \times \Lambda$ *such that* $\rightarrow_\beta \equiv \{((\lambda x.a)\ b, a\langle b/x\rangle) | x \in \mathcal{X}, a \in \Lambda, b \in \Lambda\}$

Example 1.7.

- $(\lambda x.(\lambda y.y)\ a)\ b \rightarrow_\beta ((\lambda y.y)\ a)\langle b/x\rangle \equiv (\lambda y.y)\langle b/x\rangle\ a\langle b/x\rangle \equiv (\lambda y.y)\ a\langle b/x\rangle$
- $(\lambda x.y)\ a \rightarrow_\beta y$
- *Russell paradox:* $(\lambda x.x\ x)(\lambda x.x\ x) \rightarrow_\beta (x\ x)\langle(\lambda x.x\ x)/x\rangle$ *or* $(x\ x)\langle(\lambda y.y\ y)/x\rangle \equiv (\lambda x.x\ x)(\lambda x.x\ x)$ *modulo* $=_\alpha$ *(or exactly that if you consider the case before the "or"). Then it reduces to itself and the reduction does not terminate.*

With the last example we see that we need $\rightarrow_\beta \subseteq \beta_0 \subseteq \beta \equiv \beta_0^*$. For some β_0 and β the β-reduction, $\beta \equiv \rightarrow_\beta^*$ also denoted $\twoheadrightarrow_\beta$.

Definition 1.8 (β_0-contraction). *Let* $a, b \in \Lambda$ *we define* $a\ \beta_0\ b$*:*

- $x\ \beta_0\ x$
- $(\lambda x.u)v\ \beta_0\ u\langle v/x\rangle$
- $(\lambda x.u)\ \beta_0\ (\lambda x.v)$ *if* $u\ \beta_0\ v$
- $(u\ v)\ \beta_0\ (u'\ v)$ *if* $u\ \beta_0\ u'$
- $(u\ v)\ \beta_0\ (u\ v')$ *if* $v\ \beta_0\ v'$