Shadows of Computation, Lecture 5

Will Troiani

September 2022

1 The untyped λ-calculus

Definition 1.1. *Let $\mathscr{V}$ be a (countably) infinite set of variables, and let $\mathscr{L}$ be the language consisting of $\mathscr{V}$ along with the special symbols*

$$\lambda \qquad . \qquad (\qquad)$$

Let $\mathscr{L}^*$ be the set of words of $\mathscr{L}$, more precisely, an element $w \in \mathscr{L}^*$ is a finite sequence $(w_1, ..., w_n)$ where each w_i is in $\mathscr{L}$, for convenience, such an element will be written as $w_1...w_n$. Now let Λ_p denote the smallest subset of $\mathscr{L}^*$ such that

- *if $x \in \mathscr{V}$ then $x \in \Lambda_p$,*
- *if $M, N \in \Lambda_p$ then $(MN) \in \Lambda_p$,*
- *if $x \in \mathscr{V}$ and $M \in \Lambda_p$ then $(\lambda x.M) \in \Lambda_p$*

*Λ_p is the set of **preterms**. A preterm M such that $M \in \mathscr{V}$ is a **variable**, if $M = (M_1 M_2)$ for some preterms M_1, M_2, then M is an **application**, and if $M = (\lambda x, M')$ for some $x \in \mathscr{V}$ and $M' \in \Lambda_p$ then M is an **abstraction**.*

In practice, it becomes unwieldy to use this notation for the preterms exactly, and so the following notation is adopted:

Definition 1.2.
- *For preterms M_1, M_2, M_3, the preterm $M_1M_2M_3$ means $((M_1M_2)M_3))$,*
- *For variables x, y and a preterm M, the preterm $\lambda xy.M$ means $(\lambda x.(\lambda y.M))$.*

The variables x which appear in the subpreterm M of a preterm $\lambda x.M$ are viewed as "markers for substitution", (see Remark 1.9). For this reason, a distinction is made between the variable x and the variable y in, for example, the preterm $\lambda x.xy$:

Definition 1.3. *Given a preterm M, let $FV(M)$ be the following set of variables, defined recursively*

- *if $M = x$ where x is a variable then $FV(M) = \{x\}$,*
- *if $M = M_1 M_2$ then $FV(M) = FV(M_1) \cup FV(M_2)$,*
- *if $M = \lambda x.M'$ then $FV(M) = FV(M') \setminus \{x\}$.*

*A variable $x \in FV(M)$ is a **free variable** of M, a variable x which appears in M but is not a free variable is a **bound variable**.*

As mentioned, bound variables will be viewed as "markers for substitution", so we define the following equivalence relation on Λ_p which relates a preterm M to M' if M can be obtained by replacing every bound occurrence of a variable x in M' with another variable y:

Definition 1.4. *For any term M, let $M[x := y]$ be the preterm given by replacing every bound occurrence of x in M with y. Define the following equivalence relation on Λ_p: $M \sim_\alpha M'$ if there exists $x, y \in \mathscr{V}$ such that $M[x := y] = M'$, where no free variable of M becomes bound in $M[x := y]$. In such a case, we say that M is α-**equivalent** to M'.*

Remark 1.5. The reason why we need to let x and y be such that no free variable of M becomes bound in $M[x := y]$ is so that a preterm such as $\lambda x.y$ does not get identified with the preterm $\lambda y.y$.

We are now in a position to define the underlying language of λ-calculus:

Definition 1.6. *Let $\Lambda = {}^{\Lambda_p}/_{\sim_\alpha}$ be the set of λ-**terms**. The set of **free variables** of a λ-term $[M]$ is $FV(M)$, which can be shown to be well defined. For convenience, M will be written instead of $[M]$.*

Now the dynamics of the computation of λ-terms will be defined.

Definition 1.7. ***Single step** β-**reduction** $\to_\beta$ is the smallest relation on Λ satisfying:*

- *the **reduction axiom**:*
  - *for all variables x and λ-terms M, M', $(\lambda x.M)M' \to_\beta M[x := M']$, where $M[x := M']$ is the term given by replacing every free occurrence of x in M with M',*
- *the following **compatibility axioms**:*
  - *if $M \to_\beta M'$ then $(MN) \to_\beta (M'N)$ and $(NM) \to_\beta (NM')$,*
  - *if $M \to_\beta M'$ then for any variable x, $\lambda x.M \to_\beta \lambda x M'$.*

*A subterm of the form $(\lambda x.M)M'$ is a β-**redex**, and $(\lambda x.M)M'$ **single step** β-**reduces** to $M[x := M']$.*

Remark 1.8. Strictly, single step β reduction should be defined on preterms and then shown that a well defined relation is induced on terms, but this level of detail has been omitted for the sake of clarity.

Remark 1.9. The reducition axiom shows precisely in what sense a bound variable is a "marker for substitution". For example, $(\lambda x.x)M \to_\beta M$ and $(\lambda y.y)M \to_\beta M$, which is why $\lambda x.x$ is identified with $\lambda y.y$.

It is through single step β-reduction that computation may be performed. In fact, λ-calculus is capable of performing natural number addition:

Example 1.10. Define the following λ-terms:

- ONE $:= \lambda fx.fx$,
- TWO $:= \lambda fx.ffx$,
- THREE $:= \lambda fx.fffx$,
- PLUS $:= \lambda mnfx.mf(nfx)$

then

$$\begin{aligned} PLUS\ ONE\ TWO &= (\lambda mnfx.\underline{m}f(nfx))\underline{(\lambda fx.fx)}(\lambda fx.ffx) \\ &\to_\beta (\lambda nfx.(\lambda fx.\underline{f}x)\underline{f}(nfx))(\lambda fx.ffx) \\ &\to_\beta (\lambda nfx.(\lambda x.f\underline{x})\underline{(nfx)})(\lambda fx.ffx) \\ &\to_\beta (\lambda nfx.f\underline{n}fx)\underline{(\lambda fx.ffx)} \\ &\to_\beta (\lambda fx.f(\lambda fx.\underline{ff}x)\underline{f}x) \\ &\to_\beta (\lambda fx.f(\lambda x.ff\underline{x})\underline{x}) \\ &\to_\beta (\lambda fx.fffx) = THREE \end{aligned}$$

where each step is obtained by substituting the right most underlined λ-term inplace of the left most underlined variable.

Historically, is this how Church first defined computable functions.

There is also η-expansion, which is defined similarly.

Definition 1.11. ***Single step η-expansion*** $\longrightarrow_\eta$ *is the smallest, compatible relation on* Λ *satisfying:*

$$M \longrightarrow_\eta \lambda x.Mx \tag{1.1}$$

where x is a variable not in the free variable set of M. ***Multi step η-expansion*** *is the reflexive closure of single step η-expansion.* ***η-equivalence*** *is the reflexive, symmetric symmetric closure of multi step η-expansion.*

$\beta\eta$-equivalence *is the union of η-equivalence and β-equivalence.*

2 Simply typed λ-calculus

In the simply-typed lambda calculus [29, Chapter 3] there is an infinite set of *atomic types* and the set $\Phi_\to$ of *simple types* is built up from the atomic types using $\to$. Let Λ' denote the set of untyped lambda calculus preterms in these variables, as defined in [29, Chapter 1]. We define a subset $\Lambda'_{wt} \subseteq \Lambda'$ of *well-typed* preterms, together with a function $t : \Lambda'_{wt} \longrightarrow \Phi_\to$ by induction:

- all variables $x : \sigma$ are well-typed and $t(x) = \sigma$,
- if $M = (P\,Q)$ and P, Q are well-typed with $t(P) = \sigma \to \tau$ and $t(Q) = \sigma$ for some σ, τ then M is well-typed and $t(M) = \tau$,
- if $M = \lambda x\,.\,N$ with N well-typed, then M is well-typed and $T(M) = t(x) \to t(N)$.

We define $\Lambda'_\sigma = \{M \in \Lambda'_{wt} \,|\, t(M) = \sigma\}$ and call these *preterms of type* σ. Next we observe that $\Lambda'_{wt} \subseteq \Lambda'$ is closed under the relation of α-equivalence on Λ', as long as we understand α-equivalence type by type, that is, we take

$$\lambda x\,.\,M =_\alpha \lambda y\,.\,M[x := y]$$

as long as $t(x) = t(y)$. Denoting this relation by $=_\alpha$, we may therefore define the sets of *well-typed lambda terms* and *well-typed lambda terms of type* σ, respectively:

$$\Lambda_{wt} = \Lambda'_{wt}/ =_\alpha \tag{2.1}$$

$$\Lambda_\sigma = \Lambda'_\sigma/ =_\alpha \ . \tag{2.2}$$

Note that Λ_{wt} is the disjoint union over all $\sigma \in \Phi_\to$ of Λ_σ. We write $M : \sigma$ as a synonym for $[M] \in \Lambda_\sigma$, and call these equivalence classes *terms of type* σ. Since terms are, by definition, α-equivalence classes, the expression $M = N$ henceforth means $M =_\alpha N$ unless indicated otherwise. We denote the set of free variables of a term M by $\mathrm{FV}(M)$.

3 The category of λ-terms

We define a category $\mathcal{L}$ whose objects are the types of simply-typed lambda calculus, and whose morphisms are the terms of that calculus. The natural desiderata for such a category are that the fundamental algebraic structure of lambda calculus, function application and lambda abstraction, should be realised by categorical algebra.

Following Church's original presentation our lambda calculus only contains function types and $\Phi_\to$ denotes the set of simple types. We write Λ_σ for the set of α-equivalence classes of lambda terms of type σ, and we write $=_{\beta\eta}$ for the equivalence relation generated by $\beta\eta$ equivalence.

Definition 3.1 (Category of lambda terms). The category $\mathcal{L}$ has objects

$$\mathrm{ob}(\mathcal{L}) = \Phi_{\to} \cup \{\mathbf{1}\}$$

and morphisms given for types $\sigma, \tau \in \Phi_{\to}$ by

$$\begin{aligned}
\mathcal{L}(\sigma, \tau) &= \Lambda_{\sigma\to\tau}/{=_{\beta\eta}} \\
\mathcal{L}(\mathbf{1}, \sigma) &= \Lambda_{\sigma}/{=_{\beta\eta}} \\
\mathcal{L}(\sigma, \mathbf{1}) &= \{\star\} \\
\mathcal{L}(\mathbf{1}, \mathbf{1}) &= \{\star\}\,,
\end{aligned}$$

where $\star$ is a new symbol. For $\sigma, \tau, \rho \in \Phi_{\to}$ the composition rule is the function

$$\mathcal{L}(\tau, \rho) \times \mathcal{L}(\sigma, \tau) \longrightarrow \mathcal{L}(\sigma, \rho) \tag{3.1}$$

$$(N, M) \longmapsto \lambda x^{\sigma}\,.\,(N(Mx)) \tag{3.2}$$

where $x \notin \mathrm{FV}(N) \cup \mathrm{FV}(M)$. We write the composite as $N \circ M$. In the remaining special cases the composite is given by the rules

$$\mathcal{L}(\tau, \rho) \times \mathcal{L}(\mathbf{1}, \tau) \longrightarrow \mathcal{L}(\mathbf{1}, \rho)\,, \qquad N \circ M = (N\,M)\,, \tag{3.3}$$

$$\mathcal{L}(\mathbf{1}, \rho) \times \mathcal{L}(\mathbf{1}, \mathbf{1}) \longrightarrow \mathcal{L}(\mathbf{1}, \rho)\,, \qquad N \circ \star = N\,, \tag{3.4}$$

$$\mathcal{L}(\mathbf{1}, \rho) \times \mathcal{L}(\sigma, \mathbf{1}) \longrightarrow \mathcal{L}(\sigma, \rho)\,, \qquad N \circ \star = \lambda t^{\sigma}\,.\,N\,, \tag{3.5}$$

where in the final rule $t \notin \mathrm{FV}(N)$. Notice that these functions, although their rules depend on representatives of equivalence classes, are none-the-less well defined.

For terms M, N the expression $M = N$ always means equality of terms (that is, up to α-equivalence) and we write $M =_{\beta\eta} N$ if we want to indicate equality up to $\beta\eta$-equivalence (for example as morphisms in the category $\mathcal{L}$). Since the free variable set of a lambda term is not invariant under β-reduction, some care is necessary in defining the category $\mathcal{L}_Q$ below. Let $\twoheadrightarrow_{\beta}$ denote multi-step β-reduction [29, Definition 1.3.3].

Lemma 3.2. *If $M \twoheadrightarrow_{\beta} N$ then $\mathrm{FV}(N) \subseteq \mathrm{FV}(M)$.*

Definition 3.3. Given a term M we define

$$\mathrm{FV}_{\beta}(M) = \bigcap_{N =_{\beta} M} \mathrm{FV}(N)$$

where the intersection is over all terms N which are β-equivalent to M.

Clearly if $M =_{\beta} M'$ then $\mathrm{FV}_{\beta}(M) = \mathrm{FV}_{\beta}(M')$.

Lemma 3.4. *Given terms $M : \sigma \to \rho$ and $N : \sigma$ we have*

$$\mathrm{FV}_{\beta}((MN)) \subseteq \mathrm{FV}_{\beta}(M) \cup \mathrm{FV}_{\beta}(N)\,.$$

Lemma 3.5. *Given $M : \sigma \to \rho$ and $N : \tau \to \sigma$ we have*

$$\mathrm{FV}_\beta(M \circ N) \subseteq \mathrm{FV}_\beta(M) \cup \mathrm{FV}_\beta(N)\,. \tag{3.6}$$

Given a set Q of variables we write Λ_σ^Q for the set of lambda terms M of type σ with $\mathrm{FV}(M) \subseteq Q$. Let $=_{\beta\eta}$ denote the induced relation on this subset of Λ_σ.

Lemma 3.6. *For any type σ and set Q of variables the image of the injective map*

$$\Lambda_p^Q / =_{\beta\eta} \longrightarrow \Lambda_p / =_{\beta\eta} \tag{3.7}$$

is the set of equivalence classes of terms M with $\mathrm{FV}_\beta(M) \subseteq Q$.

Proof. Since the simply-typed lambda calculus is strongly normalising [29, Theorem 3.5.1] and confluent [29, Theorem 3.6.3] there is a unique normal form $\widehat{M}$ in the β-equivalence class of M, and $\mathrm{FV}_\beta(M) = \mathrm{FV}(\widehat{M})$. Hence if $\mathrm{FV}_\beta(M) \subseteq Q$ then $\mathrm{FV}(\widehat{M}) \subseteq Q$ and so M is in the image of (3.7). □

Definition 3.7. For a set of variables Q we define a subcategory $\mathcal{L}_Q \subseteq \mathcal{L}$ by

$$\mathrm{ob}(\mathcal{L}_Q) = \mathrm{ob}(\mathcal{L}) = \Phi_\to \cup \{\mathbf{1}\}$$

and for types σ, ρ

$$\begin{aligned}
\mathcal{L}_Q(\sigma, \rho) &= \{M \in \mathcal{L}(\sigma, \rho) \mid \mathrm{FV}_\beta(M) \subseteq Q\}\,,\\
\mathcal{L}_Q(\mathbf{1}, \sigma) &= \{M \in \mathcal{L}(\mathbf{1}, \sigma) \mid \mathrm{FV}_\beta(M) \subseteq Q\}\,,\\
\mathcal{L}_Q(\sigma, \mathbf{1}) &= \mathcal{L}(\sigma, \mathbf{1}) = \{\star\}\,,\\
\mathcal{L}_Q(\mathbf{1}, \mathbf{1}) &= \mathcal{L}(\mathbf{1}, \mathbf{1}) = \{\star\}\,.
\end{aligned}$$

Note that the last two lines have the same form using the convention that $\mathrm{FV}_\beta(\star) = \emptyset$. We denote the inclusion functor by $I_Q : \mathcal{L}_Q \longrightarrow \mathcal{L}$. We write $\mathcal{L}_{cl}$ for $\mathcal{L}_Q$ when $Q = \emptyset$ and call this the category of **closed** lambda terms.

We prove that we have a category.

The following calculation shows that $\mathrm{id}_\sigma \in \mathcal{L}(\sigma, \sigma)$ is an identity at σ. Observe that for a term $M : \sigma \to \tau$, we have

$$\begin{aligned}
\lambda t^\sigma\,.\,(M(\mathrm{id}_\sigma\, t)) &= \lambda t^\sigma\,.\,(M((\lambda x^\sigma\,.\,x)t))\\
&=_\beta \lambda t\,.\,(Mt)\\
&=_\eta M\,,
\end{aligned}$$

and similarly $\lambda s^\tau\,.\,(\mathrm{id}_\tau(Ms)) =_{\beta\eta} M$. Moreover, $\star$ is clearly an identity at $\mathbf{1}$. For associativity there are a few cases to check:

- Consider a diagram of objects and morphisms in $\mathcal{L}$ of the form:

(3.8) $$\delta \xleftarrow{P} \rho \xleftarrow{N} \tau \xleftarrow{M} \sigma\,.$$

$$\begin{aligned} P \circ (N \circ M) &= \lambda y^{\sigma}\,.\,(P(N \circ M\,y)) \\ &= \lambda y^{\sigma}\,.\,(P((\lambda x^{\sigma}\,.\,(N(Mx)))y)) \\ &=_{\beta} \lambda y^{\sigma}\,.\,(P(N(My))) \\ &=_{\beta} (P \circ N) \circ M\,. \end{aligned}$$

- Consider a diagram of objects and morphisms in $\mathcal{L}$ of the form

(3.9) $$\delta \xleftarrow{P} \rho \xleftarrow{N} \tau \xleftarrow{M} \mathbf{1}\,.$$

$$\begin{aligned} P \circ (N \circ M) &= P \circ (NM) \\ &= (P(NM)) \\ &= (\lambda y^{\tau}\,.\,(P(Ny))M) \\ &= (P \circ N) \circ M\,. \end{aligned}$$

- Consider a diagram of objects and morphisms in $\mathcal{L}$ of the form

(3.10) $$\delta \xleftarrow{P} \rho \xleftarrow{N} \mathbf{1} \xleftarrow{\star} \sigma\,.$$

$$\begin{aligned} (P \circ N) \circ \star &= (PN) \circ \star \\ &= \lambda t^{\sigma}\,.\,(PN) \\ &= \lambda t^{\sigma}(P((\lambda z^{\sigma}\,.\,N)t)) \\ &= P \circ (N \circ \star)\,. \end{aligned}$$

- Consider a diagram of objects and morphisms in $\mathcal{L}$ of the form

(3.11) $$\delta \xleftarrow{P} \mathbf{1} \xleftarrow{\star} \tau \xleftarrow{M} \sigma\,.$$

$$\begin{aligned} (P \circ \star) \circ M &= (\lambda t^{\tau}\,.\,P) \circ M \\ &= \lambda q^{\sigma}\,.\,((\lambda t^{\tau}\,.\,P)(Mq)) \\ &= \lambda q^{\sigma}\,.\,P \\ &= P \circ (\star \circ M)\,. \end{aligned}$$

The other cases are trivial.

References

[1] M. Atiyah, *Duality in mathematics and physics*, in: lecutre notes from Institut de Matematica de la Universitat de Barcelona (IMUB), (2007) available at: `https://fme.upc.edu/ca/arxius/butlleti-digital/riemann/071218_conferencia_atiyah-d_article.pdf`.

[2] M. Borisavljevic, K. Dosen, Z. Petric, *On Permuting Cut with Contraction*, preprint arXiv:math/9911065v1.

[3] A. Church, *A set of postulates for the foundation of logic*, Annals of Mathematics **33**, no.2 pp.346–366 (1932).

[4] H. B. Curry and R. Feys, *Combinatory logic, volume I*, Studies in Logic and the Foundations of Mathematics, North-Holland, Amsterdam, (1958).

[5] K. Došen, *Deductive completeness*, Bulletin of symbolic logic 2.3, pp.243–283 (1996).

[6] K. Došen, *Abstraction and application in adjunction*, arXiv preprint math/0111061 (2001).

[7] R. Dyckhoff, L. Pinto, *Permutability of proofs in intuitionistic sequent calculi*, Theoretical Computer Science **212**, pp.141-155, (1999).

[8] J. H. Gallier, *Constructive logics Part I: A tutorial on proof systems and typed lambda-calculi*, Theoretical Computer Science, 110(2) pp.249–339, (1993).

[9] G. Gentzen, *Untersuchungen über das logische Schliessen*, Mathematische Zeitschrift **39** (1935) 176–210, 405–431, translation in *The collected papers of Gerhard Gentzen*, edited by M. E. Szabo, 1969.

[10] J.-Y. Girard, *The Blind Spot: lectures on logic*, European Mathematical Society, (2011).

[11] J.-Y. Girard, Y. Lafont, and P. Taylor, *Proofs and Types*, Cambridge Tracts in Theoretical Computer Science 7 ,Cambridge University Press, 1989.

[12] H. Herbelin, *A Lambda-Calculus structure isomorphic to a Gentzen-style sequent calculus structure*, in L. Pacholski and J. Tiuryn, editors, *Computer Science Logic*, 8th workshop, CSL'94, volume 933 of Lecture Notes in Computer Science, pp.61–75, Springer-Verlag (1995).

[13] C. Hermida and B. Jacobs, *Fibrations with indeterminates: Contextual and functional completeness for polymorphic lambda calculi*, Math. Structures Comput. Sci. 5 (1995), 501–531.

[14] A. Heyting, *Intuitionism, an introduction*, Studies in Logic and the Foundations of Mathematics, North-Holland, (1956). Third edition (1971).

[15] W. A. Howard, *The formulae-as-types notion of construction*, in Seldin and Hindley *To H.B.Curry: essays on Combinatory logic, Lambda calculus and Formalism*, Academic press (1980).

[16] C. B. Jay, N. Ghani, *The virtues of eta-expansion*, J. Functional Programming **1** (1): 1–000, Cambridge University Press, (1993).

[17] S. C. Kleene, *Two papers on the predicate calculus*, Memoirs of the American Mathematical Society, **10**, (1952).

[18] J. Lambek, *Functional completeness of cartesian categories*, Annals of Mathematical Logic *6.3-4* pp.259–292, (1974).

[19] J. Lambek and P.J. Scott, *Introduction to higher-order categorical logic*, Camrbidge Studies in Advanced Mathematics, Cambridge University Press, (1986).

[20] F.W. Lawvere, *Adjointness in foundations*, Dialectica **23** No. 3/4, pp.281–296, (1969).

[21] S. Mac Lane, *The Lambda Calculus and Adjoint Functors*, Logic, Meaning and Computation, Springer Netherlands, pp.181–184 (2001).

[22] G. Mints, *Normal forms for sequent derivations*, in: P. Odifreddi (Ed.), Kreiseliana, A. K. Peters, Wellesley, Massachusetts, 1996, pp. 469–492; also part of Stanford Univ. Report CSLI-94-193, November 1994.

[23] S. Negri, *Varieties of linear calculi*, Journal of Philosophical Logic **31**, pp.569–590, (2002).

[24] S. Negri and J. von Plato, *Structural proof theory*, Cambridge University Press, (2008).

[25] G. Pottinger, *Normalization as a homomorphic image of cut-elimination*, Annals of Mathematical Logic **12** pp.323–357, (1977).

[26] D. Prawitz, *Natural deduction: a proof-theoretical study*, Almqvist & Wicksell, Stockholm (1965).

[27] D. Prawitz, *Philosophical aspects of proof theory*, Contemporary philosophy: a new survey, **1**, pp.235–277, Martinus Nijhoff Publishers, The Hague/Boston/London (1981).

[28] P. Selinger, *Lecture notes on the lambda calculus*, preprint [arXiv:0804.3434], (2008).

[29] M. Sørensen and P. Urzyczyn, *Lectures on the Curry-Howard isomorphism*, Studies in Logic and the Foundations of Mathematics Vol. 149, Elsevier New York, (2006).

[30] A. S. Troelstra, *Marginalia on Sequent Calculi*, Studia Logica **62**, pp.291–303, (1999).

[31] A .S. Troelstra and D. van Dalen, *Constructivism in Mathematics*, Vol. 1, Studies in Logic and the Foundations of Mathematics, 121, Amsterdam: North-Holland, (1988).

[32] A. S. Troelstra, H. Schwichtenberg, *Basic proof theory*, Cambridge University Press, Cambridge, (1996).

[33] P. L. Wadler, *Proofs are programs: 19th century logic and 21st century computing*, Manuscript (2000).

[34] A. M. Ungar, *Normalization, cut-elimination and the theory of proofs*, Center for the Study of Language and Information Lecture Notes No. 28, (1992).

[35] J. Zucker, *The correspondence between cut-elimination and normalization*, Annals of Mathematical Logic **7** 1–112 (1974).