3. Church's formal system of lambda-calculus

3.1. Models of computation: introduction to β-reduction

A. Church in 1936 introduced a formal system based on the operations of function abstraction and application, called *the lambda calculus* and defined the notion of computable function in this system. Church's original goal was to construct a formal system for the foundations of mathematics based on functions together with a set of logical notions. When this system was discovered to be inconsistent, Church then separated out the consistent subsystem that is now called lambda calculus and concentrated on it. The language of untyped λ-calculus consists of an infinite set of variables, the abstraction operator λ, parentheses, the application operator $\cdot$ (usually$(t \cdot s)$ is written (ts)). If there is also a set of constants, the calculus is *applied*. We will deal with calculus without constants, called *pure*.

Terms. The variables are terms; if t, s are terms, also (ts) is a term. If t is a term and x is a variable, then $(\lambda x.t)$ is a term. We shorten $(((...(ts)r)...)p)$ with $tsr...p$, and we shorten $(\lambda x(\lambda y(...(\lambda z.(t)))))$ with $\lambda xy...z.t$. Moreover, to avoid confusion with other types of equality, we still use the symbol $=$ to denote the meta-linguistic syntactic identity. Following Barendregt (1984) p. 26 we make this convention:

1. Terms so-called $\alpha-$equivalent, i.e. such that $\lambda x.t =_\alpha \lambda y.t[y/x]$ are identified.

2. We will always assume without loss of generality that in a certain context (proof, definition ecc.) bounded variables have been renamed to be distinct, and distinct from free variables.

This convention is called *Barendregt's variables convention*. Thanks to it, the substitution operation can be defined quite simply as follows:

1. $x[s/x] = s$
2. $y[s/x] = y \;\; x \neq y$
3. $(tr)[s/x] = t[s/x]r[s/x]$
4. $(\lambda x.t)[s/x] = \lambda x.t$
5. $(\lambda y.t)[s/x] = \lambda y.(t[s/x])$

Lemma 7. (Substitution lemma) *Let $x \neq y$ and x is not among the free variables of s. Then:*

$$t[r/x][s/y] = t[s/y][r[s/y]/x]$$

Proof. Induction on t. For instance, if $t = \lambda z.V$, where z is a new variable, for the above convention; from the definition we have:

$$(\lambda z.V)[r/x][s/y] = \lambda z.(V[r/x][s/y]) =$$

$$= \lambda z.(V[s/y][r[s/y]/x]) \;\; (inductive \;\; hypot.) =$$

$$= (\lambda z.V)[s/y][r[s/y]/x]$$

Exercise. Verify the other cases. If t is a variable (a) if $t = x$, the we have $r[s/y] = r[s/y]$; (b) if $t = y$, then we get $s = s$ and (c) if $t = z$ variable different from y and from y we get $z = z$. QED

Duccio Pianigiani, University of Siena, Italy, duccio.pianigiani@unisi.it, 0000-0001-9441-7226

Referee List (DOI 10.36253/fup_referee_list)
FUP Best Practice in Scholarly Publishing (DOI 10.36253/fup_best_practice)

Book References DOI 10.36253/979-12-215-0778-2.references

For *redex* we mean a term like $(\lambda x.t)s$; it can be simplified by repalcing all occurence x in t, with s, that we denote $t[s/x]$ (its *contractum*); the fundamental operation of the λ-calculus, called β-contraction, is the following:

$$(\lambda x.t)s \to_\beta t[s/x]$$

We write $t \to_1 s$, meaning that s is obtained by contracting a single redex of t. We say that t *converts in a term* s, if the latter results from a finite number of $1-$contractions and inverse $1-$contractions (retractions) from t. A term is in *normal form*, if it cannot be further reduced, because it does not contain any *redex*. shall we say that the "reduction", that we denote with $\Rightarrow$, constitutes the reflexive and transitive closure of $\to_1$, while the "conversion" $=$ is the reflexive, transitive and symmetric closure. We get an *extensional* calculus, by adding the η-rule $\lambda x.Ux = U$, for $x \notin FVar(U)$. Observe that the equality between functions is extensional:

$$\forall x(f(x) = g(x)) \to f = g$$

But if $f = y$ and $g = \lambda x.yx$, we get $fU = gU$. However $f \neq g$. The η-rule allows to overcome the obstacle.

The following is easily shown.

Lemma 8. *The following holds:*

1. *if* $t \Rightarrow s$, *then* $r[t/x] \Rightarrow r[s/x]$.
2. *if* $t = s$ *then* $t[r/x] = s[r/x]$.
3. *if* $t = s$ *then* $r[t/x] = r[s/x]$.
4. *if* $t \to_1 s$ *then* $t[r/x] \to_1 s[r/x]$

The original theory introduced by Church (called λ_I-calculus) was different from that we have illustrated, called λ_{K}-calculus. A term of the λ_{I}-calculus has the property that in all its subterms of the form $\lambda x.s$, the variable x must occur free in s at least once. Hence $\mathsf{K} = \lambda xy.x$ is not a well-formed term of this language. To highlight the main difference, consider the term $\mathsf{KI}\Omega$ of the λ_K-calculus, where $\mathsf{I} = \lambda x.x$ and $\Omega = (\lambda x.xx)(\lambda x.xx)$. Depending on the order in which we perform the reduction, the term $\mathsf{KI}\Omega$ can be reduced either to normal form I, or produce an infinite reduction $\mathsf{KI}\Omega \to \mathsf{KI}\Omega \to$ ..... This cannot happen in Church's original calculus, where a term t has normal form iff each its subterm also has normal form.

Confluence. A binary relation $\prec$ is called *confluent* iff:

$$\forall xyz\exists w(x \prec y \wedge x \prec z \Rightarrow y \prec w \wedge z \prec w)$$

We claim that the relation $\Rightarrow$ is confluent and we show this beginning, as is usually done, by demonstrating that it is fulfilled by the relation of *parallel reduction* due to Tait and Martin-Löf (see for instance Barendregt (1984) pp. 60-3), but in the elegant simplified version of Takahashi (1995). We cannot start with the one-reduction because it is known not to be confluent.

Definition 15. (Parallel reduction)

1. $x \to_p x$
2. $\dfrac{t \to_p s}{\lambda x.t \to_p \lambda x.s}$
3. $\dfrac{t \to_p s \quad r \to_p w}{tr \to_p sw}$
4. $\dfrac{t \to_p s \quad r \to_p w}{(\lambda x.t)r \to_p s[w/x]}$

In presence of η-rule, add:

$$\frac{t \to_p s}{\lambda x.tx \to_p s}$$

We show that there exists a term t^T, dependent from t, but not from s, such that if $t \to_p s$, then $s \to_p t^T$ and follows the confluence for $\to_p$. Since $\Rightarrow$ is the transitive closure of $\to_p$, it follows the confluence also for this relation.

Definition 16. *The term t^T is inductively defined as follows:*

1. $x^T = x$
2. $(\lambda x.t)^T = \lambda x.(t)^T$
3. $(rs)^T = r^T s^T \ \ (rs \ \ non \ \ redex)$
4. $((\lambda x.t)s)^T = t^T[s^T/x]$

Theorem 34. *If $t \to_p s$, then $s \to_p t^T$.*

Proof. Induction on the complexity of t.

1. If $t = x$, i.e. $x \to_p s$, then $s = x = t^T$.
2. If $t = \lambda x.r$ and $\lambda x.r \to_p s$, then take $s = \lambda x.u$, where $r \to_p u$ and apply the induction hypothesis, obtaining $u \to_p r^T$, from which $\lambda x.u \to_p \lambda x.r^T$; take finally $t^T = \lambda x.r^T$.
3. If $t = ru$ and $ru \Rightarrow s$ (ru non redex), this means that $s = r'u'$ and $r \to_p r'$, $u \to_p u'$. By induction hypothesis there are r^T and u^T such that $r'u' \to_p t^T$, where $t^T = r^T u^T$.
4. If $t = (\lambda x.r)u \to_p s$ we will have these possibilities: either $s = (\lambda x.r')u'$, or $s = r'[u'/x]$. In both cases $r \to_p r'$ and $u \to_p u'$. By induction hypothesis $r' \to_p r^T$ and $u' \to_p u^T$. In the first case take $t^T = r^T[u^T/x]$. Clearly $s \to_p t^T$. Similarly in the second case.

QED

Corollary 5. *If a normal form for t exists, this is unique.*

Proof. If there were two n_1, n_2, where $t \Rightarrow n_1$ and $t \Rightarrow n_2$, for Church-Rosser there will be a term s to which both n_1 and n_2 converge: being in normal form we will have s, n_1 and n_2 are the same term. QED

In the type-free calculus we have *fixed points* operators. A fixed point operator is a term Fix such that, for all term s, Fix$s = s$(Fix$s$). For instance:

$$\mathcal{Y} = \lambda f.(\lambda x.f(xx))(\lambda x.f(xx))$$

is a fixed point. Observe that, if $\mathcal{W} = \lambda x.F(xx)$, we have :

$$\mathcal{Y}F \Rightarrow \mathcal{W}\mathcal{W} \Rightarrow F(\mathcal{W}\mathcal{W}) = F(\mathcal{Y}F)$$

For instance, if we want to build a term F such that $FXY = FYXF$, we can argue as follows: we need an F such that $F = (\lambda fxy.fxyf)F$; therefore we define $F = \mathcal{Y}(\lambda fxy.fxyf)$. In general we have the following.

Theorem 35. (Fixed point theorem) *Given a term $\mathcal{C}[f, x]$, with free variables f and x, there is a term F such that for all T, $FT = \mathcal{C}[F, T]$*

Proof. As we have seen, take $F = \mathcal{Y}(\lambda fx.\mathcal{C}[f, x]))$.

To ensure that those terms, defines the required function, we reason as follows: let $W = \lambda fxy.fxyf$ (and then $\mathcal{Y}W = W(\mathcal{Y}W)$); then $F = \mathcal{Y}(W) = W(\mathcal{Y}W) = (\lambda fxy.fxyf)(\mathcal{Y}W) = (\lambda fxy.fxyf)F$. QED

Exercise. Let $A = \lambda xy.y(xxy)$ and $\Theta = AA$; show that Θ is a fixed point operator. It is observed that, while it is not valid $\mathcal{Y} \Rightarrow f(\mathcal{Y}f)$, It is valid instead $\Theta \Rightarrow f(\Theta f)$.

Remark 3. *A fundamental difference between the calculus without types and the calculus with types is that in the former the conversion is undecidable, whereas in the latter (as satisfying normalisation) it is decidable.*

The lambda calculus can be presented as an equational theory; we introduce therefore the theory $\widetilde{\lambda}$, defined by the following axioms and rules:

1. $\lambda x.U = \lambda y.U[y/x]$, $y \notin FVar(U)$
2. $(\lambda x.U)V = U[V/x]$
3. $U = U$
4. $\dfrac{U = V}{ZU = ZV}$
5. $\dfrac{U = V}{UZ = VZ}$
6. $\dfrac{U = V \quad V = Z}{U = Z}$
7. $\dfrac{U = V}{V = U}$
8. $\dfrac{U = V}{\lambda x.U = \lambda x.V}$

So, we want to show taht the theory $\widetilde{\lambda}$ is undecidable. Recall that two sets X, Y are called *recursively separable* if there is a recursive set $\mathcal{C}$ such that $X \subseteq \mathcal{C}$ and $Y \cap \mathcal{C} = \emptyset$. A pair of sets of terms is called recursively separable iff the corresponding sets of Gödel numbers are recursively separable. We say that a set of λ-terms X is *closed* with respect to β- conversion, iff $t \in X$ *and* $t = r$ *implies* $r \in X$.

Second fixed point theorem. We write, to simplify the notation, $X^{\#}$, meaning by this $\overline{\ulcorner X \urcorner}$, the numeral of Gödel number of X and we observe that these functions are definable by terms of our calculus $APP(M^{\#}N^{\#}) = (MN)^{\#}$, $NUM(M^{\#}) = (M^{\#})^{\#}$.

Theorem 36. *Let* $W = \lambda x.F(APPx(NUMx))$ *and* $\widetilde{X} = WW^{\#}$*; then*

$$\widetilde{X} = WW^{\#} = F(APPW^{\#}(NUM(W^{\#})) = F(WW^{\#})^{\#} = F\widetilde{X}^{\#}$$

Hence: $\forall F \exists X : FX^{\#} = X$

Proof. Immediate. QED

Theorem 37. (Scott and Curry) *Let* $X, Y \neq \emptyset$ *sets of terms, closed with respect to* β*-conversion. Hence* X, Y *are not recursively separable.*

Proof. Let $t \in X$, $r \in Y$. Let $\mathcal{C}$ a recursive set such that $X \subseteq \mathcal{C}$ e $Y \cap \mathcal{C} = \emptyset$. Let F the term that defines the characteristic function of $\mathcal{C}$, and then $F^{\overline{\ulcorner u \urcorner}} = \overline{0}$ if $u \in \mathcal{C}$ and $F^{\overline{\ulcorner u \urcorner}} = \overline{1}$ if $u \notin \mathcal{C}$. Let finally:

$$\mathcal{G} = \lambda x.\delta(ZERO(Fx))rt$$

where $ZERO\overline{n}$ is the test for zero (see below at p. 65) which returns $\lambda xy.x$, if $n = 0$, and $\lambda xy.y$, if $n > 0$. Observe that, if $u \in \mathcal{C}$, then $\mathcal{G}^{\overline{\ulcorner u \urcorner}} = r$; if instead $u \notin \mathcal{C}$, then $\mathcal{G}^{\overline{\ulcorner u \urcorner}} = t$. But for the previous fixed point theorem, there exists X such that $\mathcal{G}^{\overline{\ulcorner X \urcorner}} = X$. Hence we finally have:

* $X \in \mathcal{C} \Rightarrow X = \mathcal{G}^{\overline{\ulcorner X \urcorner}} = r \Rightarrow X \notin \mathcal{C}$, because $r \in Y$.
* $X \notin \mathcal{C} \Rightarrow X = \mathcal{G}^{\overline{\ulcorner X \urcorner}} = t \Rightarrow X \in \mathcal{C}$, because $t \in X$. QED

Corollary 6. *If* X *is closed for the relation "=", then it is not recursive.*

Proof. In previous theorem take $Y = \overline{X}$ (the complement of X); it follows that if X is closed for β-conversion, then X is not recursive. QED

Corollary 7. *The set* X_{nf} *of terms that have a normal form is recursively enumerabile, but not recursive.*

Proof. Recall that a set is computably enumerable iff can be defined by a formula of the form $\exists x P$, for P recursive. Hence X_{nf} is recursively enumerabile because M has a normal form iff $\exists U \exists p$("U is in normal form and p is a proof of $M = U$"). But it is not recursive (*Exercise*). This is one of the first examples of computably enumerable but not recursive sets. QED

Corollary 8. *The relation "=" is undecidable.*

Proof. Take $X = \{r | \vdash r = (\lambda x.x)\}$. It is closed for β-conversion and therefore not recursive. QED

It follows the undecidability of the *Entscheidungsproblem*: When terms are given Gödel numbers, then "=" corresponds to a relation between natural numbers, i.e. once goedelized, terms become numbers and "=" become a relation between numbers; the numbers can also be represented in a very simple predicative language, for example $0 = x, 1 = f(x), 2 = f(f(x))$.... Axioms of the equational theory of $\lambda - calculus$ can thus be translated into formulas $\alpha_0, ..., \alpha_n$ of the predicate calculus, in such a way that $(\alpha_0 \wedge ... \wedge \alpha_n) \rightarrow E(\overline{m}, \overline{n})$ is provable in the predicate calculus, if $m = \ulcorner s \urcorner$, $n = \ulcorner t \urcorner$ e $t = s$. But if we could decide this formula, we could also decide "=". Indeed, if we could decide all questions of provability in pure predicate logic, then we could decide whether arbitrary lambda-terms are convertible.

Theorem 38. (Böhm's separability theorem 1968) *It t and s are $\beta\eta$-normal forms, and u,v are arbitrary terms, the it is possible to construct terms $r_0, ..., r_n$ and find variables $x_0, ..., x_k$ such that $\lambda x_0 ... \lambda x_k . t r_0 ... r_n = u$ and $\lambda x_0 ... \lambda x_k . s r_0 ... r_n = v$.*

Proof. (See Krivine (1993), p. 88) QED

A consequence is that if $t = s$ is not provable and it is added to the calculus, the calculus crashes.

Let us now begin to assess the expressive power of this calculus. There are many ways in which we can represent numbers, using appropriate terms ("numerals"), in λ-calculus. For instance, these are the *Church numerals*:

$$\overline{n} =_{def} \lambda xy. \overbrace{x(x(x(...(xy)...)}^{n-times}$$

with these numerals, we can define the basic arithmetic operations as follows:

1. $(sum) \quad \oplus = \lambda xyzu.xz(yzu)$
2. $(multiplication) \quad \otimes = \lambda xyz.x(yz)$
3. $(exponentiation) \quad EXP = \lambda xy.yx$

If we abbreviate $\overbrace{x(x(x(...(xy)...)))}^{n-times}$ with $x^n y$, we see for example that:

$$\overline{n}uz = (\lambda wv.w^n v)uz = (\lambda v.u^n v)z = u^n z$$

from which follows:

$$\oplus \overline{n}\overline{m} = (\lambda xyzu.xz(yzu))\overline{n}\overline{m} = \lambda zu.\overline{n}z(\overline{m}zu) =$$
$$\lambda xyzu.z^n(z^m u) = \lambda zu.(z^{m+n}u) = \overline{m+n}$$

The reader can verify by exercise that with respect to Church numerals, the followings are respectively the predecessor function and the test for zero:

1. $P = \lambda xfy.x(\lambda pq.q(pf))(\mathsf{K}y)\mathsf{I}$, $\mathsf{K} = \lambda xy.x$ and $\mathsf{I} = \lambda x.x$.
2. $Zero = \lambda x.x(\lambda y.\mathsf{K}^*)\mathsf{K}$, where $\mathsf{K}^* = \lambda xy.y$.
3. $\top = \mathsf{K}$, $\bot = \mathsf{K}^*$.

That of Church is not the only system of numerals. More abstractly a system of numerals is a sequence $N = n_0, n_1, n_2...$ of closed terms such that there exist terms S and Zero for which it $Sn_k = n_{k+1}$, $Zero\ n_0 = \top$, $Zero\ n_{k+1} = \bot$ (where $\top$ and $\bot$ are truth-values, see below).

Definition 17. *A partial recursive function ψ is definable with respect to N, iff there exists a term F of the lambda claculus such that for all $r_0, ..., r_k$,*

$$F n_{r_0} ... n_{r_0} = n_{\psi(r_0,...,r_k)}$$

We say that N is adequate, if with respect to it we can define all partial recursive functions; equivalently: if we can define the predecessor $P(n_{k+1}) = n_k$.

Here we choose the approach to the numbers and functions of the classical Barendregt (1984).

1. *Booleans.* $\top = \lambda xy.x = \mathsf{K}$, $\bot = \lambda xy.y = \mathsf{K}^*$
2. *Discriminator.* An operator δXPQ such that:

$$\delta XPQ = \begin{cases} P & \text{if } X = \top \\ Q & \text{if } X = \bot \end{cases}$$

   Take $\delta = \lambda xyz.xyz$. Actually $\top PQ = XPQ = P$, $\bot PQ = XPQ = Q$.
3. *Pair and projections.* $\langle P, Q\rangle = \lambda x.xPQ$, $\pi_0 = \lambda xy.x = \mathsf{K}$, $\pi_1 = \lambda xy.y = \mathsf{K}^*$.
   Clearly $\langle P, Q\rangle\pi_0 = P$ and $\langle P, Q\rangle\pi_1 = Q$
4. *Barendregt's numerals.* $\overline{0} = \lambda x.x$, $\overline{n+1} = \langle\bot, \overline{n}\rangle$. For instance $\overline{3} = \langle\bot, \overline{2}\rangle = \langle\bot, \langle\bot, \overline{1}\rangle\rangle = \langle\bot, \langle\bot, \langle\bot, \overline{0}\rangle\rangle\rangle$
5. *Successor.* $S\overline{n} = (\lambda x.\langle\bot, x\rangle)\overline{n}$
6. *Predecessor.* $P\overline{n} = (\lambda x.x\bot)\overline{n}$
7. *Test for zero.* $ZERO = \lambda x.x\top$.

   Observe that

$$\mathsf{ZERO}\overline{0} = (\lambda x.x\top)\overline{0} = \top$$

$$\mathsf{ZERO}\overline{n+1} = (\lambda x.x\top)\langle\bot, \overline{n}\rangle = \bot$$

Recall that in general $\langle X\pi_0, X\pi_1\rangle = X$ does not hold; pair is not "surjective" (C.Mann 1972). We now obtain the *total* recursive functions in this way:

1. *Initial functions.* (*projections*) $U_i = \lambda x_0, ..., x_p.x_i$ (*successore*) $\lambda x.\langle\bot, x\rangle$ (*zero*) $\lambda x.\overline{0}$
2. *Closure under composition.* Let $g, h_0, .., h_m$ be defined by terms $G, H_0, ..., H_m$; we define composition as followe:

$$F = \lambda\overrightarrow{x}.G(H_0\overrightarrow{x})...(H_m\overrightarrow{x})$$

3. *Closure under primitive recursion.* Suppose that:

   (a) $f(0, n) = g(n)$,

   (b) $f(k+1, n) = h(f(k, n), k, n)$

   and let h, g be defined respectively by H, G.
   By *Fixed point theorem*, given a term $\mathcal{C}[f, x]$, there is a term F such that for all T, $FT = \mathcal{C}[F, T]$. As we have seen, we can take $F = \mathcal{Y}(\lambda fx.\mathcal{C}[f, x]))$. Hence, applying this theorem, we have that we can define F such that:

$$Fx\overrightarrow{y} = \delta(\mathsf{ZERO}x)(G\overrightarrow{y})(H(F(Px)\overrightarrow{y})(Px)\overrightarrow{y})$$

4. *Minimalization.* Let $f(n) = \mu m(g(n, m) = 0)$ and let g be defined by G. Let then:

$$Hxy = \delta(\mathsf{ZERO}(Gxy)y(Hx(Sy))$$

   Let finally $F = \lambda x.Hx\overline{0}$. Observe that $F\overline{n} = \overline{0}$, if $G\overline{n}\overline{0} = \overline{0}$ and $F\overline{n} = H\overline{n}\overline{1}$ otherwise; $F\overline{n} = \overline{1}$, if $G\overline{n}\overline{1} = \overline{0}$ and $F\overline{n} = H\overline{n}\overline{2}$ otherwise, etc. if moreover $G\overline{n}\overline{2} = \overline{0}$, then:

$$H\overline{n}\overline{0} = H\overline{n}\overline{1} = H\overline{n}\overline{2} = \delta(\mathsf{ZERO}(G\overline{n}\overline{2})\overline{2}(H\overline{n}\overline{2}) = \overline{2}$$

Let us now address the problem of of representing *partial* functions. We will say that ψ is $\lambda-$definible with respect to a set of terms X, if there exists a term F of the $\lambda - calculus$ such that:

1. if $\psi(n)\downarrow$, then $F\overline{n} = \overline{\psi(n)}$
2. If $\psi(n)\uparrow$, then $F\overline{n} \in X$

The elements of X constitutes the formalization of the concept of "meaningless" terms. But what set X we have to choose? Various purposes are admitted. Church took the terms without normal form, for instance $\Omega = (\lambda x.xx)(\lambda x.xx)$. Barendregt take the unsolvable terms: a closed term M is *solvable*, iff for some n and terms $T_0, ...T_n$, $MT_0, ..., T_n = (\lambda x.x)$. There are terms without normal form but with *head normal-form* $\lambda x_0...\lambda x_n.yM_0...M_n$. A term does not convey any information, if not even possess this normal form. We will see that a term is solvable if and only if it has a head normal form. In this regard, there are general results.

Theorem 39. (Statman 1970) *A set of closed terms X is said "co-Visser set", if $\overline{X}$ is computably enumerable and it is closed for $=$, i.e. if $t \in \overline{X}$ and $t = s$, then $s \in \overline{X}$. If X is a "co-Visser set", then all partial recursive functions can be defined using it as a representative of undefined terms.*

Proposals of Church and Barendregt satisfy this theorem (see Barendregt (1992)). The two different choices by these logicians are actually related:

1. in λ_I calculus: meaningless=unsolvable= no normal form.

2. In λ_K calculus: meaningless=unsolvable= no head normal form.

3.2. Solvability and head normal forms

Let us begin by defining more formally the notion of "solvable term".

Definition 18. *A closed term t is called* solvable *iff there exists a number n and terms $s_0, ..., s_n$ such that $ts_0, ..., s_n = \mathsf{I}$. An arbitrary term $t(x_0, ..., x_k)$ is solvable iff its closure $\lambda x_0...x_k.t$ is solvable.*

For instance, if $\mathcal{Y}$ is the above-mentioned fixed point operator, then it is solvable: $\mathcal{Y}(\mathsf{KI}) = (\mathsf{KI})(\mathcal{Y}(\mathsf{KI})) = \mathsf{I}$.

Lemma 9. *Let t be a term: t is solvable iff there exists a closed substitution instance t^* of it and closed terms $s_0, ..., s_n$ such that $t^*s_0, ..., s_n = \mathsf{I}$.*

Proof. $\Rightarrow$ Consider the closure $\lambda x_0...\lambda x_k.t$, that, by hypothesis, is solvable, i.e. for some $s_0, ..., s_n$, $(\lambda x_0...\lambda x_k.t)s_0, ..., s_n$ (assume $n > k$, perhaps by adding copies of I), Hence $t[s_0/x_0, ..., s_k/x_k]s_{k+1}...s_n = \mathsf{I}$, Put $t^* = t[s_0/x_0, ..., s_k/x_k]$.
$\Leftarrow$ If $t^* = t[s_0/x_0, ..., s_k/x_k]$ is solvable, then clearly also $\lambda x_0...\lambda x_k.t$ is solvable. QED

Lemma 10. *t is solvable iff $\lambda x_0.t$ is solvable.*

Proof. (Sketch) Let $\lambda x_0...\lambda x_k.t$ be the closure of t. By previous lemma t is solvable iff there exist closed terms $s_0, ..., s_k, s_{k+1}, ..., s_n$ such that:

$$t[s_0/x_0, ..., s_k/x_k]s_{k+1}...s_n = \mathsf{I}$$

iff

$$((\lambda x_0.t)s_0)[s_1/x_1, ..., s_k/x_k]s_{k+1}, ..., s_n = \mathsf{I}$$

iff

$$(\lambda x_0.t)[s_1/x_1, ..., s_k/x_k]s_0 s_{k+1}, ..., s_n = \mathsf{I}$$

iff $\lambda x_0.t$ is solvable. QED

Corollary 9. *t is unsolvable iff for all s, we have that ts, $t[s/x]$, $\lambda x.t$ are unsolvable.*

One can prove by an easy induction that each term is either of the form

1. $\lambda x_0 ... \lambda x_n . y t_0 ... t_k$, or of the form
2. $\lambda x_0 ... \lambda x_n . (\lambda y . s) r t_0 ... t_n$

where the sequences $x_0, ..., x_n$ and $t_0, ..., t_k$ might be empty. The redex $(\lambda y.s)r$ is called *head redex.* A contraction of a term t to a term u that eliminates a head redex is called *head reduction* and denoted $t \to_h u$. If in a reduction sequence $t_0 \to_h t_1 \to_h t_2 \to ...$ there is a term t_n of the form $\lambda x_0 ... \lambda_k . y s_0 ... s_m$ (called *head normal form* h.n.f.) such a reduction sequence terminates at t_n. If $t = s$ and s is in h.n.f. then we say that t *has* an h.n.f. It holds that t *has* an h.n.f. iff its head reduction path of terminates.

Lemma 11. *$\lambda x.t$ has an h.n.f. iff t has an h.n.f.*

Proof. Clearly if $\lambda x.t \Rightarrow s$ then s has the form $\lambda x.u$, where $t \Rightarrow u$. QED

Lemma 12. *If $t[s/x]$ has an h.n.f. also t has an h.n.f.*

Proof. Suppose on the contrary that t has no head normal form. It is not hard to show that the following substitution holds: if $t \to_h s$, then $t[r/x] \to_h s[r/x]$, hence $t[s/x]$ has an infinite head-reduction path and therefore has no h.n.f. itself. QED

Theorem 40. *If ts has an h.n.f. then t has an h.n.f too.*

Proof. Let $t \to_h t_0 \to_h t_1 \to_h t_2 ...$ be the head reduction of t and consider these cases:

1. If no term of this sequence has form $\lambda x.u$, the an head-reduction of ts has the form $ts \to_h t_0 s \to_h t_1 s \to_h t_2 s ...$ and by hypothesis it is finite; hence also $t \to_h t_0 \to_h t_1 \to_h t_2 ...$ is finite and then t has an h.n.f.
2. If some term of this sequence has form $\lambda x.u$, let t_k the first of such terms. Hence the head reduction has the form:

$$ts \to_h t_0 s \to_h t_1 s \to_h t_2 s ... \to_h t_{k-1} s \to_h (\lambda x.u)s \to_h u[s/x] \to_h ...$$

However, if ts has an h.n.f. also $u[s/x]$ has an h.n.f., as well as u and $\lambda x.u$ and t, by previous lemmas. QED

Theorem 41. *A term t has an h.n.f. iff its head reduction path terminates.*

Proof. $\Rightarrow$ Recall that the Church-Rosser theorem holds also for β-equality (conversion), by "composing the boxes": if $t = s$ then there exists r such that $t \Rightarrow r$ and $s \Rightarrow r$. Let us suppose that $t = \lambda x.yU$. Hence, by Church-Rosser theorem there exists z such that both $t \Rightarrow z$ and $\lambda x.yU \Rightarrow z$. Hence z must be of the form $\lambda x.yV$ where $U \Rightarrow V$ and by the *Standardization Theorem* at p. 70 there will be a standard reduction $t = t_0 \to^{\delta_0} t_1 \to^{\delta_1} t_2 \to ... \to \lambda x.yV$. Now, if all redexes δ_j contracted in this reduction are head redexes, then this is a terminating head reduction. Otherwise, let δ_i the first *internal* redex reduced at step $t_i \to^{\delta_i} t_{i+1}$. Then t_i must be in head normal form, because otherwise the head redex would remain: but the reduction of t_i to $\lambda x.yV$ in this case would not be standard. Then $t \to_h ... \to_h t_i$ is a terminating head reduction.

$\Leftarrow$ Trivial.

QED

Theorem 42. (Wadsworth 1971) *The term t is solvable iff it has an h.n.f.*

Proof. By previous results being solvable is true for a term as well as for its closure. The same equivalence holds for the property of having a head normal form. Hence we can assume that t is closed. $\Rightarrow$ If $ts_0 ... s_n = \mathsf{I}$ then I is its h.n.f. Hence by previous lemmas t also has itself an h.n.f. $\Leftarrow$ If t has an head normal form $\lambda x_0 ... \lambda x_n . y t_0 ... t_k$, let $y = x_i$. Observe that:

$$(\lambda x_0 ... \lambda x_n . y t_0 ... t_k) \overbrace{(\mathsf{K}^{k+1}\mathsf{I})(\mathsf{K}^{k+1}\mathsf{I})...(\mathsf{K}^{k+1}\mathsf{I})}^{n+1-times}$$

reduces to $(\mathsf{K}^{k+1}\mathsf{I})t_0^* ... t_k^*$ and therefore to I ($\mathsf{K}^m\mathsf{I}$ abbreviates $\mathsf{K}(\mathsf{K}(\mathsf{K}(...(\mathsf{KI})...)))$ m-times). QED

Corollary 10. *Unsolvable terms have no normal form.*

Proof. If t is in normal form, then has also an head normal form and therfore is solvable. QED

Theorem 43. *A term t is unsolvable iff it is hereditarily without normal form, i.e. for all substitution t^* and all $s_0, ..., s_n$, $t^*s_0...s_n$ ha no normal form.*

Proof. $\Rightarrow$ If $t^*s_0...s_n$ has a normal form, then it is solvable. Therefore there exists a closed substitution instance $(t^*s_0...s_n)^*$ of it and terms $r_0, ..., r_k$ such that $(t^*s_0...s_n)^*r_0, ..., r_k = \mathsf{I}$. Thus $(t^{**}s_0^*...s_n^*)r_0, ..., r_k = \mathsf{I}$ and therefore t is solvable. $\Leftarrow$ If t is solvable, then $(t^*s_0...s_n) = \mathsf{I}$ for some $s_0...s_n$. QED

Definition 19. *A partial recursive function ϕ is lambda-definable, iff there exists a term F such that for all natural numbers n:*

$$F\overline{n} = \begin{cases} \overline{\phi(n)} & \text{if } \phi(n) \downarrow \\ no\ h.n.f & \text{otherwise} \end{cases}$$

Lemma 13. (Solvability of numerals) $\overline{m}\mathsf{KII} = \mathsf{I}$.

Proof. Recall that $\overline{0} = \mathsf{I}$ and $\overline{n+1} = \langle \mathsf{K}^*, \overline{n} \rangle$. Hence $\overline{0}\mathsf{KII} = \mathsf{IKII} = \mathsf{I}$.
Moreover $\overline{n+1}\mathsf{KII} = \langle \mathsf{K}^*, \overline{n} \rangle \mathsf{KII} = \mathsf{KK}^*\overline{n}\mathsf{II} = \mathsf{K}^*\mathsf{II} = \mathsf{I}$. QED

Lemma 14. *If F defines a partial function ϕ, then:*

1. *$F\overline{n}\mathsf{KII} = \mathsf{I}$, if $\phi(n) \downarrow$.*
2. *$F\overline{n}\mathsf{KII}$ is unsolvable, otherwise.*

Proof. If $\phi(n) \downarrow$, then $F\overline{n} = \overline{\phi(n)}$; but $\overline{\phi(n)}\mathsf{KII} = \mathsf{I}$. If $\phi(n) \uparrow$, then $F\overline{n}$ is unsolvable. Then it $F\overline{n}s_0...s_n$ is unsolvable for all $s_0, ..., s_n$. QED

Theorem 44. (Closure under composition) *Suppose:*

$$\phi(n) \simeq \chi(\psi_0(n), ..., \psi_m(n))$$

and suppose that $G, H_0, ..., H_m$ respectively define $\chi, \psi_0, ..., \psi_m$. Then:

$$F = \lambda x(H_0 x\mathsf{KII})...(H_m x\mathsf{KII})(G(H_0 x)...(H_m x))$$

defines ϕ.

Proof. If some $\psi_i(n) \uparrow$, then $H_i\overline{n}\mathsf{KII}$ (the "jamming factor") is unsolvable, by the previous lemma, and therefore the whole term F is unsolvable. In case all $\psi_i(n) \downarrow$, then all $H_i\overline{n}\mathsf{KII} = \overline{\psi_i(n)}\mathsf{KII} = \mathsf{I}$ and therefore $F\overline{n} = (G(H_0\overline{n}...H_m\overline{n}))$. QED

To prove closure under minimisation, we must now focus on reduction strategies. We will show in particular that leftmost reductions are *normalizing*, that is, when t has a normal form, then there exists n such that applying n-times the strategy to it leads to the normal form.

Definition 20. (Leftmost reduction) *Call the lambda of a redex $(\lambda x.t)s$ the first occurrence of λ in it. Let δ_0, δ_1 be two occurrences of redex in t. We say that δ_0 is to the left of δ_1, if the λ of δ_0 is to the left of the λ of δ_1. A redex occurrence si called the leftmost redex of a term, if it is to the left of all other redexes.*

Example. In this case δ_0 is the *leftmost*:

$$\underbrace{(\lambda x.\overbrace{(\lambda z.z)x}^{\delta_1}\, x)u}_{\delta_0}(\lambda u.V)$$

To sum up, we write:

1. $t \to_l s$ if s is obtained by contraction of the leftmost redex.
2. $t \to_h s$ if s is obtained by contraction of the head redex.
3. $t \to_i s$ if s is obtained by contraction of an internal redex (we will write $t \to_{pi} s$ for a contraction, both parallel and internal).

The Standardization theorem. Let us call a number p the *position* of a redex δ_i in the reduction step $t_i \to^{\delta_i} t_{i+1}$, if the first symbol of δ_i in t_i is the $p-th$, starting from the left. We denote $p = p_i$. Then the reduction

$$t_0 \to^{\delta_0} t_1 \to^{\delta_1} t_2 \to^{\delta_2} t_3 ....$$

is standard if $p_0 \leq p_1 \leq p_2 \leq ...$, i.e. the sequence of redexes contracted in the reduction moves from left to right. In other words, in this reduction sequence, once a redex is contracted, all redexes whose first symbol is at its left become "frozen". For example, this is a standard reduction:

$$\lambda a.\overbrace{(\lambda b.(\lambda c.c)bb)d}^{1} \to \lambda a.\overbrace{(\lambda c.c)d}^{2}\, d \to \lambda a.dd$$

On the contrary, this is not a standard reduction:

$$\lambda a.(\lambda b.\overbrace{(\lambda c.c)b}^{1}\, b)d \to \lambda a.\overbrace{(\lambda b.bb)d}^{2} \to \lambda a.dd$$

In particular, leftmost reductions $\to_\ell$ and head reductions $\to_h$ are standard.

Theorem 45. (Standardization theorem) *If $t \Rightarrow s$, then there is a standard reduction from t to s.*

We omit the proof, which is performed with the same techniques we will use later in this paragraph (see Sørensen, Urzyczyin (2006) pp. 13-7 and especially Takahashi (1995) p. 124).

Lemma 15. *If there is a standard reduction of t to s and s is in normal form, then all its reductions are leftmost.*

Proof. Suppose that t is reduced to s in standard way, but, by contradiction, at some step a leftmost redex $(\lambda x.p)q$ is not reduced. Hence, since the reduction is standard, this redex is frozen (perhaps p is reduced to p' and q is reduced to q', but $(\lambda x.p')q'$ is never reduced). Hence s cannot be in normal form. QED

Definition 21. *Let us write $t \rhd s$ iff $t = t_0 \to_h t_1 \to_h ... \to_h t_n \to_{pi} s$ and $t_i \to_p s$ for all $i \leq n$.*

Lemma 16. *The following hold:*

1. *If $t \rhd s$ then $\lambda x.t \rhd \lambda x.s$*
2. *If $t \rhd s$ and $r \to_p q$, then $tr \rhd sq$.*
3. *If $t \rhd s$ and $r \rhd q$, then $t[r/z] \rhd s[q/z]$.*

Proof. 1. Recall that $t \rhd s$ iff $t = t_0 \to_h t_1 \to_h ... \to_h t_n \to_{pi} s$ and $t_i \to_p s$ for all $i \leq n$. But $t_n \to_{pi} s$ implies $t_n \to_p s$ and by definition this implies $\lambda x.t_n \to_{pi} \lambda x.s$. Moreover, if $t_i \to_h t_{i+1}$, then $\lambda x.t_i \to_h \lambda x.t_{i+1}$.

2. Now let $t = t_0 \to_h t_1 \to_h ... \to_h t_n \to_{pi} s$ and $t_i \to_p s$ for all $i \leq n$. If at least one t_k is an abstraction, take the first with this property. Hence $t_k r \to_{pi} sq$ and $tr = t_0 r \to_h t_1 r \to_h ... \to_h t_k r \to_{pi} sq$, where each $t_i r \to_p sq$. If no t_k is an abstraction, keep $k = n$.

3. Lastly let us first assume that $t \to_{pi} s$ and $r \rhd q$. We want to show that $t[r/x] \rhd s[q/x]$ by considering two subcases:

   3.1 if $t = \lambda z.(\lambda y.P)Qr_0...r_n$ and $s = \lambda z.(\lambda y.P')Q'r'_0...r'_n$, where for each $i \leq n$, $r_i \to_p r'_i$, $P \to_p P'$, $Q \to_p Q'$. But $t[r/x] \to_{pi} s[q/x]$ (recall that substitution holds for $\to_p$ and that if $r \rhd q$, in particular $r \to_{pi} q$).

   3.2 If $t = \lambda z.yr_0...r_n$, then $s = \lambda z.yr'_0...r'_n$.

   (a) If $x \neq y$, then $t[r/x] \to_{pi} s[q/x]$.

   (b) If $x = y$, notice that $r \to_p q$ might be a contraction and we don't want a head reduction. Then argue by applying substitution for parallel reduction, (2) and (1):

$$t[r/x] = \lambda z.rr_0[r/x]...r_n[r/x] \rhd \lambda z.qr'_0[q/x]...r'_n[q/x] = s[q/x]$$

More precisely:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{r_0 \to_{pi} r'_0 \quad r \to_p q}}{r_0[r/x] \to_{pi} r'_0[q/x] \qquad r \rhd q}}{rr_0[r/x] \rhd qr'_0[q/x] \qquad r_1[r/x] \to_{pi} r'_1[q/x]}}{rr_0[r/x]r_1[r/x] \rhd qr'_0[q/x]r'_1[q/x]}}{\dfrac{\dfrac{\vdots}{rr_0[r/x]...r_n[r/x] \rhd qr'_0[q/x]...r'_n[q/x]}}{\lambda z.rr_0[r/x]...r_n[r/x] \rhd \lambda z.qr'_0[q/x]...r'_n[q/x]}}$$

Hence $t[r/x] \rhd s[q/x]$. Now let us consider the general case $t \rhd s$. Notice now that in the general case we have:

$$\underbrace{t[r/x] = t_0[r/x] \to_h t_1[r/x] \to_h ... \to_h}_{substitution\ for\ h-reduction} \overbrace{t_m[r/x] \to_h ... \to_h t_n[r/x] \to_{pi} s[q/x]}^{points\ 3.1\ and\ 3.2}$$

Observe that for each i, still by substitution in parallel reduction, $t_i[r/x] \to_p s[q/x]$. Hence $t[r/x] \rhd s[q/x]$. QED

Lemma 17. (Factorization) *If $t \to_p s$ then $t = t_0 \to_h t_1 \to_h t_2 \to_h ... \to_h r \to_{pi} s$, for some r.*

Proof. Show that $t \to_p s$ implies $t \rhd s$ (notice that this is a *stronger* statement) by induction on the structure of t, using properties 1,2,3 above:

- If $t \to_p t$, nothing to prove.
- If $t = vr \to_p uq$, and by IH $v \rhd u$, then apply 2.
- If $t = \lambda x.v \to_p \lambda x.u = s$, where $v \to_p u$, then by IH $v \rhd u$ and apply 1.
- If $t = (\lambda x.v)r \to_p u[q/x] = s$ and $v \to_p u$ and $r \to_p q$, then by IH obtain $v \rhd u$ and $r \rhd q$. Now apply 3.

QED

Lemma 18. (Inversion) *If $t \to_{pi} r \to_h s$, for some r, then $t \to_h t'_0 \to_h t'_1 \to_h t'_2 \to_h ... \to_h v \to_{pi} s$, for some v.*

Proof. We must have $t = \lambda z.(\lambda x.P)Qr_0...r_n$, since the second step is a head reduction, $r = \lambda z.(\lambda x.P')Q'r'_0...r'_n$ where $P \to_p P'$, $Q \to_p Q'$ and for all $i \leq n$, $r_i \to_p r'_i$. This means that $s = \lambda z.P'[Q'/x]r'_0...r'_n$. Now put $v = \lambda z.P[Q/x]r_0...r_n$. Clearly this is obtained from t by head-reduction, and in turn is reducible to s by parallel reduction (P could be a redex) and therefore, since $v \to_p s$, the previous lemma apply and the parallel reduction can be factorized in two blocks: first a sequence of head reductions, then a parallel internal reduction. QED

Theorem 46. *If t has a normal form s, then there is a leftmost reduction from t to s.*

Proof. By induction on s. Thanks to the previous lemma we can decompose each step $t_i \to_p t_{i+1}$ of the parallel reduction of t to s in this way: $t_i \to_h ...r \to_{pi} t_{i+1}$ for some r. Hence we have a reduction of this form:

$$t = \overbrace{t_0 \to_h ...r_0 \to_{pi}}^{①}\overbrace{t_1 \to_h ...r_1 \to_{pi}}^{②}\overbrace{t_2 \to_h ...r_2}^{③}... \to_{pi} s$$

But by the previous result the head reduction can be brought at the beginning, so that $t = t_0 \to_h ... \to_h r_k \to_{pi} ... \to_{pi} s$. Hence $r_k = \lambda x.yP_0...P_n$ and $s = \lambda x.yP'_0...P'_n$ where P'_i is the normal form of P_i and by induction hypothesis is obtained by leftmost-reduction. But head-reduction too is a kind of leftmost-reduction. Hence t reduces to s by leftmost-reduction. QED

Now, it is important to point out that similar result holds also for *quasi-leftmost* reductions:

$$t_0 \to^{d_0} t_1 \to^{d_1} t_2 \to^{d_2} ...$$

where for all n there exists an $m \geq n$ such that the redex d_m is leftmost in t_m (where d_i is the redex eliminated at step i).

Theorem 47. *If t has an infinite quasi leftmost reduction, then t has no normal form.*

So we are ready to demonstrate the closure under minimalisation (Barendregt (1984) pp. 178-83).

Definition 22. *Let $\Theta = (\lambda xy.y(xxy))(\lambda xy.y(xxy))$ be the Turing fixed point operator; then let $H = \Theta(\lambda uz.If\ Pz,\ then\ z,\ else\ uz^+)$ and $\mu P = H\overline{0}$.*

The above fixed-point operator has the property that $\Theta X \Rightarrow X(\Theta X)$. Recall that "if x then y, else z" is simply $\lambda xyz.xyz$. Note that $H\overline{n}$ means "If $P\overline{n}$ then $\overline{n}$, else $H\overline{n+1}$". Hence μP is "If $P\overline{0}$ then $\overline{0}$ else $H\overline{1}$".

Theorem 48. *Let P be a term such that $P\overline{n} \Rightarrow \bot$, for all n, where $\bot = K^*$. Then:*

1. *μP has no normal form.*
2. *μP is unsolvable.*

Proof. 1. $H\overline{n} \Rightarrow If\ P\overline{n},\ then\ \overline{n},\ else\ H\overline{n+1}$. Now, if $P\overline{n} \Rightarrow \bot$, for all n, we have the infinite sequence of reduction $H\overline{0} \Rightarrow H\overline{1} \Rightarrow H\overline{2} \Rightarrow ....$ In the passage $H\overline{n} \Rightarrow H\overline{n+1}$ at least one head redex is contracted. Hence μP has an infinite quasi-leftmost reduction. Hence it has no normal form.

2. Let $(\mu P)^*$ a closed substitution instance of (μP). This has the form $\mu(P^*)$ and by 1. has no normal form. Hence it is hereditarily without normal form and therefore μP is unsolvable.

QED

Theorem 49. (Closure under minimalization) *Let $\phi(\overline{n}) \simeq \mu m(\chi(n,m) = 0)$ and suppose that G defines a total function χ. Take $P = \lambda y.\mathsf{Zero}(Gxy))$. Hence $F = \lambda x.\mu(\lambda y.\mathsf{Zero}(Gxy))$ defines ϕ.*

Proof. Consider two cases:

1. if $\phi(n)\downarrow$, then this means that for some m (say, the minimum), $\chi(n,m)=0$. Hence $\phi(n)=m$ and therefore $F\overline{n}=\overline{\phi(n)}=\overline{m}$
2. if $\phi(n)\uparrow$, then for all m, $\mathsf{Zero}(G\overline{nm})\Rightarrow\bot$ and therefore $F\overline{n}=\mu(\lambda y.Zero(G\overline{n}y))$ is unsolvable.

To see that all partial recursive functions are representable, recall now this results of characterization: the class of the partial recursive functions is the least class of partial numeric functions containing all primitive recursive functions and closed under composition and this schema of unbounded minimalization:

> Suppose $g(x,y)$ is a *total* function. Then f is defined by minimization from g if and only if
>
> $$f(x)=\mu b(g(x,b)=0)$$
>
> or $f(x)\uparrow$ if there is no such b.

(The total recursive functions are obtained by replacing this schema by the regular minimization, where g is *regular* iff $\mu b(g(x,b)=0)\downarrow$). QED

3.3. The simply typed lambda calculus $\lambda_{\times,\to}$

First developed by Bertrand Russell in the early 1900s to avoid paradoxes, typed calculus was later develped by Church and Curry. In the approach *à la* Church all terms are furnished with type information: if a term is typeable, it has a unique type. In the approach *à la* Curry, types are assigned to existing untyped terms, using typing rules. Thinking terms as algorithms/programs, type as specification, these two approaches reflect two different paradigms in programming languages. We consider here as constructors $\to$ and $\times$. In this case, among the terms, in addition to abstraction and application, we will also have pairs and projections.

Definition 23. *1. (Set of types $\mathbb{T}$)*

(a) A set of base types $\tau_0,\tau_1,\tau_2...$

(b) if σ, τ are types, also $\sigma\to\tau$, $\sigma\times\tau$ are types

(c) these are the ony types.

2. (Typed terms in Church's style) The set of typed terms $\bigcup\{\lambda_\sigma|\sigma\in\mathbb{T}\}$ is defined as follows:

(a) $\sigma\in\mathbb{T},x\in Var\Rightarrow x^\sigma\in\lambda_\sigma$

(b) $t\in\lambda_{\sigma\to\tau},s\in\lambda_\sigma\Rightarrow(ts)\in\lambda_\tau$

(c) $t\in\lambda_\tau\Rightarrow(\lambda x^\sigma.t)\in\lambda_{\sigma\to\tau}$

(d) $t\in\lambda_\sigma,s\in\lambda_\tau\Rightarrow\langle t,s\rangle\in\lambda_{\sigma\times\tau}$

(e) if $t\in\lambda_{\sigma\times\tau}$, then $\pi_0 t$ and $\pi_1 t$ are respectively in λ_σ and in λ_τ.

As already remarked, sometimes the calculus *à la* Church is introduced by means of rules of assignement of types in a similar way to Curry's style. Instead of assuming that the set of variables is partitioned into disjoint sets indexed by types one uses *environments* Γ to declare types of free variables, and says that a judgement $\Gamma\vdash t:\sigma$ holds[1], if it is derivable from a certain set of typing rules.

1. *Environments.* Are set of the form $\Gamma=\{x_0:\tau_0,...,x_n:\tau_n\}$

[1] Rules are often given 'in sequential form'. Below we present them in the form of rules of natural deduction in the Gentzen-Prawitz style, so that the relation to Proof Theory is immediately clear. The reader accustomed to natural deduction in the style of Gentzen-Prawitz may think of this expression as denoting a derivation t of σ from hypotheses not discharged in Γ.

2. *Judgements.* $\Gamma \vdash t : \sigma$ whose meaning is: t is of type σ in the environment Γ.

We say that a term t is *typable*, if there are Γ and σ such that $\Gamma \vdash t : \sigma$. Typical problems of study are the followings:

1. *Type checking.* Given Γ, t and σ, decide whether $\Gamma \vdash t : \sigma$.

2. *Typability.* To decide whether a term is typable.

3. *Type inhabitation.* To decide, for a given type τ, whether there exists a closed term t such that $\vdash t : \tau$.

The first two problems are decidable in polynomial time; the last is PSPACE complete (see Statman (1979))). Type derivations are built up from assumptions of the form $x : \sigma$ by using the following inference rules.

$$\frac{t : \sigma \quad s : \tau}{\langle t, s\rangle : \sigma \times \tau} \qquad \frac{t : \sigma \times \tau}{\pi_0(t) : \sigma} \qquad \frac{t : \sigma \times \tau}{\pi_1(t) : \tau}$$

$$\frac{\begin{array}{c}[x : \sigma] \\ \vdots \\ t : \tau\end{array}}{\lambda x^{\sigma}.t : \sigma \to \tau} \qquad \frac{t : \sigma \to \tau \quad s : \sigma}{(ts) : \tau}$$

Example. Notice that the types assigned to combinators $\mathsf{K} = \lambda xy.x$ and $\mathsf{S} = \lambda xyz.xz(yz)$, as formulas, correspond to well known tautologies, also intuitionistically valid. Write "$t : \alpha$" to mean "$t \in \lambda_\alpha$".

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x : \alpha \to \beta \to \gamma \quad z : \alpha}{xz : \beta \to \gamma} \quad \dfrac{y : \alpha \to \beta \quad z : \alpha}{yz : \beta}}{xz(yz) : \gamma}}{\lambda z.xz(yz) : (\alpha \to \gamma)}}{\lambda yz.xz(yz) : (\alpha \to \beta) \to (\alpha \to \gamma)}}{\lambda xyz.xz(yz) : (\alpha \to \beta \to \gamma) \to (\alpha \to \beta) \to (\alpha \to \gamma)}$$

Analogously:

$$\dfrac{\dfrac{x : \alpha}{\lambda y.x : (\beta \to \alpha)}}{\lambda xy.x : \alpha \to (\beta \to \alpha)}$$

in analogy with natural deduction, think to an empty application of the introduction rule for implication. Recall that $\mathsf{K} = \lambda xy.x$ is not a term of λ_I-calculus and $\alpha \to (\beta \to \alpha)$ is not accepted in linear as well as relevant logic.

Exercise. Show that Church's numerals have types $(\alpha \to \alpha) \to (\alpha \to \alpha)$.

Proofs as programs. The expert reader will recognize the similarity between these rules and the Gentzen-Prawitz natural deduction propositional rules for the fragment $\wedge, \to$ of the intuitionistic logic. The Curry-Howard isomorphism states that for any derivation in intuitionistic logic there exists a typable λ-term (*à la* Church) and conversely. Actually le language can be exended with an empty type and further constructors in order to reach a full correspondence between typed lambda calculus and intuitionistic logic, where the lambda terms are an appropriate notation for intuitionistic proofs, according to the Brouwer- Heyting-Kolmogorov interpretation of intuitionistic operators. This is the basis of the so-called *proofs-as-programs* paradigm, or *Curry-Howard*

correspondence.

Lambda − calculus	Natural Deduction
$Variables$	$Assumptions$
$Terms$	$Proofs$
$Types$	$Formulas$
$Constructors$	$Connectives$
$Redex$	$Detours$
$Contraction\ \ step$	$Reduction\ \ step\ \ in\ \ normalization$

(1)

The correspondence between $\beta - contraction$ and elimination of *detours* in the normalization algorithm for Natural Deduction is evident:

$$\dfrac{\dfrac{\begin{array}{c}[x:\phi]\\ \vdots\\ t:\psi\end{array}}{\lambda x.t:\phi\to\psi}\,\mathcal{I} \qquad p:\phi}{(\lambda x.t)p:\psi}\,\mathcal{E}$$

Reduces to (where $\mathcal{D}^* = \mathcal{D}[p/x]$):

$$\begin{array}{c} p:\phi\\ \vdots\\ \mathcal{D}^*\\ \vdots\\ t[p/x]:\psi\end{array}$$

Unlike the case of untyped calculus, in the calculus with types *any* term can be reduced to normal form. By the confluence theorem (Church-Rosser) if the normal form exists, this is unique. A term is said to *strongly* normalizzabile when there is no *infinite* sequence of reductions that starts with it. Regarding the fragments of the typed lambda calculus considered above, both of these two results apply: *weak normalization*, i.e. there is at least one strategy reduction leading to the normal form, and *strong normalization*, which means that all reduction sequences are finite (so you get a normal form regardless of the order of application of the rules of contraction). Here we show the *strong* normalization of $\lambda_{\times,\to}$.

This proof is due to Tait and Girard (see Girard, Lafont and Taylor (1989) pp. 41-5 and Troelstra and Schwichtenberg (2000) pp. 157-59) and starts by introducing the notion of reducibility. The set Red_α ("reducible terms of type α") is defined inductively with respect to α as follows:

1. If α is atomic and t is a term of type α, then: $t \in \mathsf{Red}_\alpha$ if there is no infinite reduction sequence starting with t (i.e. t is strongly normalizable).
2. If $\alpha = \sigma \times \tau$ is t a term of type α, then $t \in \mathsf{Red}_\alpha$ if $\pi_0 t \in \mathsf{Red}_\sigma$ and $\pi_1 t \in \mathsf{Red}_\tau$
3. If $\alpha = \sigma \to \tau$ and t a term of type α, then $t \in \mathsf{Red}_\alpha$ if $\forall u \in \mathsf{Red}_\sigma$ we have $tu \in \mathsf{Red}_\tau$.

Now we introduce the notion of neutrality, due to Girard.

Definition 24. *A term t is neutral, if it is neither of the form $\langle u, v\rangle$, nor $\lambda x.t$.*

We show a preliminary result, based on König's lemma ("*An infinite tree finitely ramified, has at least an infinite branch*").

Lemma 19. *If t is strongly normalizable, then exists a number $n_t \in \mathbb{N}$ such that the length of all reduction sequences starting with t is less than n_t (the inverse is clear).*

Proof. Consider the tree $\mathcal{T}$ of all possible reductions of t. Since t has a finite number of subterms the tree is finitely ramified. But if t is strongly normalizable, the tree has no infinite branchs. Hence, by König, $\mathcal{T}$ is finite and therefore there exist an n_t as required. QED

Theorem 50. *The following holds:*

(a) if $t \in \mathsf{Red}_\alpha$, then t is strongly normalizable
(b) if $t \in \mathsf{Red}_\alpha$ and $t \Rightarrow s$, then $s \in \mathsf{Red}_\alpha$.
(c) if t is neutral *and if contracting a redex r of t we always obtain a term $t' \in \mathsf{Red}_\alpha$, then also $t \in \mathsf{Red}_\alpha$.*

Proof. Induction α. In the atomic case (a) follows by definition; as regards instead (b), if $t \in \mathsf{Red}_\alpha$, then t is strongly normalizable; hence, all s such that $t \Rightarrow s$ is strongly normalizable and therefore $s \in \mathsf{Red}_\alpha$; to show (c) we need the previous lemma: each reduction of t goes, by hypothesis, through an $s \in \mathsf{Red}_\alpha$ and such an s is by definition strongly normalizable; So each branch of the tree representing the possible reductions of t is finite. It follows that $n_t = max\{n_s | t \Rightarrow_1 s\} + 1$, and therefore that t is strongly normalizable.

As for the compelx cases:

If $\alpha = \sigma \times \tau$; as regards to (a), suppose that $t \in \mathsf{Red}_\alpha$; then by (IH), since $\pi_0 t \in \mathsf{Red}_\sigma$ and $\pi_1 t \in \mathsf{Red}_\tau$, we will have that $\pi_0 t$ e $\pi_1 t$ are strongly normalizable. Then exists in particular $n_t \leq n_{\pi_0 t}$ and so t is strongly normalizable.

As regards to (b), if $t \Rightarrow s$, then $\pi_0 t \Rightarrow \pi_0 s$ and $\pi_1 t \Rightarrow \pi_1 s$. Suppose that $t \in \mathsf{Red}_{\sigma\times\tau}$. By definition this means that $\pi_0 t \in \mathsf{Red}_\sigma$ and $\pi_1 t \in \mathsf{Red}_\tau$. By (IH) $\pi_0 s \in \mathsf{Red}_\sigma$ and $\pi_1 s \in \mathsf{Red}_\tau$ and therefore by definition $s \in \mathsf{Red}_{\sigma\times\tau}$. As regards to (c), if t is neutral and all s accessible from t in one step are reducible, then we consider $\pi_0 s$ obtained reducing from inside $\pi_0 t$: this is the only possible reduction, not being (for neutrality) $\pi_0 t$ a redex and therefore $t \neq \langle t_0, t_1\rangle$. By hypothesis $s \in \mathsf{Red}_{\sigma\times\tau}$ and therefore $\pi_0 s \in \mathsf{Red}_\sigma$. Hence we have that $\pi_0 t$ is neutral and all $\pi_0 s$ obtained from it in one step is reducible of type σ. Hence we apply (IH) with respect to (c), and conclude that also $\pi_0 t$ is reducible of type σ. The proof for $\pi_1 t$ is symmetric. Hence we have that $\pi_0 t$ is reducible of type σ and $\pi_1 t$ is reducible of type τ and therefore $t \in \mathsf{Red}_{\sigma\times\tau}$.

Let us see the most complex case, namely that in which $\alpha = \sigma \to \tau$. As regards to (a) , if $t \in \mathsf{Red}_{\sigma\to\tau}$ and x is of type σ, we use the inductive hypothesis (on the type) relative to (c): being x neutral and normal, we will have $x \in \mathsf{Red}_\sigma$. But if $t \in \mathsf{Red}_{\sigma\to\tau}$ and $x \in \mathsf{Red}_\sigma$, then by definition $tx \in \mathsf{Red}_\tau$. By applying (IH) on the type relative to (a), we conclude that there exist a n_{tx} finite as in the lemma; henche must exist also $n_t \leq n_{tx}$ and therefore t is strongly normalizable We now quickly see (b); if $t \Rightarrow s$ and t is reducible, take a term $r \in \mathsf{Red}_\sigma$, so that we have $tr \in \mathsf{Red}_\tau$ and $tr \Rightarrow sr$: from (IH) on the type concerning (b), we have that $sr \in \mathsf{Red}_\tau$ and therefore by definition $s \in \mathsf{Red}_{\sigma\to\tau}$.

The proof relative to (c) requires more attention: let t neutral and suppose that all s accessible from t in one step is reducible of type $\sigma \to \tau$; let $u \in \mathsf{Red}_\sigma$ and consider by (IH) on the type relative to (a), that u can be assumed strongly normalizable and therefore there exist n_u as in the lemma. Now we perform a subinduction on n_u: let us see the possible one-step reductions of tu.

1. if $tu \Rightarrow_1 su$, from initial hypothesis $s \in \mathsf{Red}_{\sigma\to\tau}$ and by definition $su \in \mathsf{Red}_\tau$.

2. if $tu \Rightarrow_1 tw$, where $u \Rightarrow_1 w$, then by (IH) on the type relative to (b), we have $w \in \mathsf{Red}_\sigma$; moreover $n_w < n_u$, and therefore by (IH) on n_w we have that $tw \in \mathsf{Red}_\tau$.

We remark that there are no other alternatives, since by neutrality of t, tu cannot be a redex and therefore $t \neq \lambda x.v$. Summarizing, tu is neutral and reduces in one step only to terms in Red_τ. By (IH) on the type relative to (c), we obtain that $tu \in \mathsf{Red}_\tau$ and by definition $t \in \mathsf{Red}_{\sigma\to\tau}$. QED

Now we consider non-neutral terms, and the following holds.

Lemma 20. *If t, s are reducible, the so is also $\langle t, s\rangle$.*

Proof. By (a), t and s are strongly normalizable. Then we use the numbers n_t and n_s and perform an induction on $n_t + n_s$, showing that $\pi_0\langle t, s\rangle$ and $\pi_1\langle t, s\rangle$ are reducible, from which follows that also $\langle t, s\rangle$ is reducible. Indeed, notice that $\pi_0\langle t, s\rangle$ contracts as follows:

1. $\pi_0\langle t, s\rangle \Rightarrow_1 t$ and t is by hypothesis reducible.

2. if $t \Rightarrow_1 r$, then from (b), r is reducible. Moreover $n_r < n_t$, hence we can apply (IH) to $n_r + n_s < n_t + n_s$ and conclude that $\pi_0\langle r, s\rangle$ is reducible.
3. if $s \Rightarrow_1 w$, analogously we obtain that $\pi_0\langle t, w\rangle \Rightarrow_1 t$ is reducible.

With the same criterium we analize $\pi_1\langle t, s\rangle$, concluding that $\pi_0\langle t, s\rangle$ and $\pi_1\langle t, s\rangle$ reduces in one step to a reducible term and therefore by (c) are reducible. Hence $\langle t, s\rangle$ is reducible. QED

Lemma 21. *If for all reducible $s \in \mathsf{Red}_\sigma$ we have that $t[s/x]$ is reducible, then $\lambda x.t$ is reducible.*

Proof. Induction on $n_t + n_s$; we show that $(\lambda x.t)s$ is reducible, if s is reducible. Observe that $(\lambda x.t)s$ is reducible in one step to the following terms:

1. $t[s/x]$, that by hypotesis is reducible.
2. $(\lambda x.r)s$, where $t \Rightarrow_1 r$. Given that t is reducible (from (b)), also r is, and $n_r < n_t$. By (IH) we have that $(\lambda x.r)s$ is reducible.
3. $(\lambda x.t)w$, where $s \Rightarrow_1 w$. Argue as in previous case.

Hence $(\lambda x.t)s$ reduces in one step only to reducible terms, and by (c) is itself reducible. Hence $\lambda x.t$ is reducible, if $t[s/x]$ is, for all s reducible. QED

We obtain the strong normalization result, proving that all the terms are reducible (and therefore, strongly normalizable!). To obtain this additional result, we show that $t[s_0/x_0, ..., s_n/x_n]$ is reducible, for $s_0, ..., s_n$ reducible. In particular, for $s_i = x_i$, we have that $t(x_0, ..., x_n)$ is reducible.

Lemma 22. *$t[s_0/x_0, ..., s_n/x_n]$ is reducible, for $s_0, ..., s_n$ reducible.*

Proof. Induction on the complexity of t:

1. If $t = x_i$, obvious, because $t[s_i/x_i] = s_i$ and s_i is reducible by hypothesis.
2. $t = \pi_0 w$ and by (IH) $w[s_i/x_i]$ is reducible, by definition also $\pi_0 w[s_i/x_i] = t[s_i/x_i]$ is reducible. (the case of π_1 is symmetric).
3. If $t = \langle r, s\rangle$ and by (IH) $r[s_i/x_i]$ $s[s_i/x_i]$ are reducible, then $\langle r[s_i/x_i], s[s_i/x_i]\rangle$ is reducible, from the above results.
4. If $t = rs$ (*Exercise*).
5. If $t = \lambda x.s$ and by (IH) $s[v/x, s_i/x_i]$ is reducible, for all v reducible, the previous emma states that $\lambda x.s[s_i/x_i]$ is reducible.

QED

Corollary 11. *All terms are strongly normalizzabile.*

Proof. Taking $s_i = x_i$ in the previous result, we have that all terms are reducible. QED

Corollary 12. *The relation $t = s$, in the typed calculus, is decidable.*

Proof. The normal forms of t and s are effectively computable.

QED

However the decision method is not elementary, i.e. there is no fixed k such that its complexity is on the order:

$$2^{2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}}$$

iterated $k - times$, as shown in Statman (1979). This is the content of the following theorem (in this version taken from Nerode and Odifreddi (1990) pp. 249-50 and Troelstra and Schwichtenberg (2000) pp. 161-63).

Theorem 51. *For infinitely many terms t any normalizing procedure takes at least a superexponential number of steps.*

Proof. Let us define:

1. $Exp_0(n) = n$
2. $Exp_{k+1}(n) = n^{Exp_k(n)}$
3. $Superexp(n) = Exp_n(n) = n^{n^{\cdot^{\cdot^{\cdot^{n}}}}} \Big\} n+1-times$

Let us consider a Church's numeral $\overline{n}$ and recall that $\overline{nn} = \overline{n^n}$. Recall also that $\overline{n}fx = f^n x$ and therefore $\overbrace{\overline{n}...\overline{n}}^{n+1-times} \; x = f^{Superexp(n)}x$

Notice that at each reduction step a term at most squares its length, e.g. $t = (\lambda x.x^n r)s$ of length about $n + |r| + |s|$ reduces to $s^n r$ on the order $n \cdot |s| + |r| < |t|^2$. How many steps are needed for going from a term of length n^2 to a term of length $Superexp(n)$? Notice that at each step the length increases as follows:

$$n^2, n^{2^2}, n^{2^3} ... n^{2^k} ...$$

Hence to get at step k a value n^{2^k} of the order of $Superexp(n)$ such a k (the number of steps) must be of the order of $Exp_{n-2}(n)$. Notice that:

$$Exp_{n-2}(n) \geq Exp_{n-2}(n-2) = Superexp(n-2)$$

QED

The strong normalization and therefore the decidability, highligt the fact that the typed calculus is weaker than the untyped calculus. Actually, not all computable functions are definable in the typed version. The extended polynomials (generated by projections, constants, sg, $\overline{sg}$, $+$, $\times$, composition) are definable in this calculus. In Schwichtenberg (1976) it is proved that exactly the extended polynomials are definable in it (with numerals expressed *à la Church* and typed as above, where α is a base type). Hence primitive recursion cannot be defined (in fact not even the predecessor function).

A sensitive enrichment in power is represented by the system T of functionals of finite type, dating back to Gödel, that provides a higher type analogue of the notion of primitive recursive computation. Already in Zilsel's lecture of 1938, Gödel had proposed to extend the finitistic mathematics with higher type functionals. This extension comes to light in the context of a proof of consistency of arithmetic and is part of the so-called "Dialectica interpretation", from the name of the journal where appeared the article Gödel (1958) at the 70° of Paul Bernays. The author, however, was not fully satisfied and continued in work there during the next decade (see Gödel (1972) for a later revision). Gödel describes a hierarchy of systems which he calls 'finitary'. These include not only finitary number theory, which he calls *the lowest level of the hierarchy*, but systems which extend beyond this, involving functions of higher type. Following Bernays, he actually distinguished two components in the finitary attitude: the *constructive component*, for which we are allowed to speak of mathematical objects in so far as we can produce them by means of a construction, and the properly *finitistic component*, for which the objects of our statements and our constructions are *intuitive*, spatio-temporal arrangements of elements in a precise sense:

> What Hilbert means by *Anschauung* is substantially Kant's space-time intuition confined, however, to configurations of a finite number of discrete objects (Gödel (1972)).

He argued that the second, too restrictive requirement, must be dropped and certain abstract notions must be admitted in finitistic mathematics, as the primitive recursive functionals of finite types, that generalize the primitive recursive functions. This opens up the possibility of a constructive (although not finitistic) proof of the consistency of arithmetic; what is proposed, is the interpretation of intuitionistic arithmetic HA in a in a quantifier-free theory theory of *functional type* called sytem T (via the so-called Gentzen-Gödel interpretation, the same goes for PA). Already in fragments of arithmetic, it is provable that the consistency of T, implies the consistency of HA and PA. The modern definition of the system T, as a purely typed lambda

calculus, has been given later. This system also provided a point of departure for modern type theory (see e.g. Bishop (1970)). It is confluent and strongly normalizzabile, although, unlike the case of $\lambda_{\rightarrow,\times}$, the strong normalization theorem uses methods that transcend PA and not formalized in it.

The system has a base-type Int. As regards to terms t of type $\mathsf{Int} \rightarrow \mathsf{Int}$, they induce a function $f_t : \mathbb{N} \longrightarrow \mathbb{N}$ such that $t\overline{m} \Rightarrow^* \overline{n}$ if and only if $f_t(m) = n$. A function f for which there exist such an t_f, is called *definable in* T.

Theorem 52. *The class of functions definable in* T *by terms of type* $\mathsf{Int} \rightarrow \mathsf{Int}$ *coincides with those provably total in* PA.

Adding an operation of abstraction on types, as in Girard and Reynold system F, greatly increases the class of representable functions to the class of functions provably total in second order Peano arithmetic (See Girard, Lafont and Taylor (1989) pp. 61-3 and pp. 89-102). This system had a strong influence on the development of type theory.

3.4. The Curry-Howard-Lambek "computational trinitarianism"

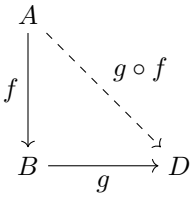
In this chapter, we will discuss the three-way isomorphism between type theory, intuitionistic Proof Theory and *category theory*, widely used in Computer Science, sometimes referred as the *computational trinitarianism*, all three theories being "manifestations of one divine notion of computation" (Harper (2011)). To show this correspondence, it will be necessary to introduce some basic concepts of category theory. This mathematical theory starts with the observation that many properties of mathematical systems can be unified and simplified by a presentation with diagrams of arrows, functors, natural transformations, adjointness etc. Many properties of mathematical constructions may be represented by *universal properties* of diagrams. It is therefore a sort of universal mathematical language, like set theory, that highligts surprising connections among different fields. Category theory started in 1945 with Eilenberg and Mac Lane and has many applications, e.g. in 1950s Grothendieck found applications to algebraic geometry. Eilenberg and Mac Lane actually claimed that category theory should have been considered a generalization of Klein's *Erlangen program*, that placed emphasis on the role of transformation groups and their invariants to classify geometries. According to some authors (see Awodey (1996)) category theory support a form of mathematical structuralism, based on the primitive concept of morphism and distinct from the set-theoretic structuralism of the mathematicians of the Bourbaki group. In 1960s Lawvere applied this theory to logic (see Lawvere (1970) and Lawvere (1969)) and subsequently, Lambek showed in the early 1970s that the proofs of intuitionistic propositional logic and the combinators of typed combinatory logic share a common equational theory which is the one of cartesian closed categories (see Lambek and Scott (1986)). The expression *Curry–Howard–Lambek correspondence* refers to the three way isomorphism between intuitionistic logic, typed lambda calculus and *cartesian closed categories*, with objects being interpreted as types or propositions and morphisms as terms or proofs, so that a morphism $f : \alpha \longrightarrow \beta$ is interpreted as a proof of β from the assumption α. Beyond the below most famous example of correspondence between *Intuitionistic Logic* and *Cartesian Closed Categories*, an interesting development and a suggestion for further study is in the direction of categorical semantic for *Linear Logic* and of the categorical counterpart to linearity. However in that case *Cartesian Closed Categories* are not an adequate a categorical counterpart anymore, but a better choice is that of the so-called symmetric monoidal closed categories (see Abramsky and Tzevelekos (2011)). In fact, this paradigm is the basis for two wide-ranging research programmes: Girard's *Linear Logic* and Martin-Löf's *Intuitionistic Type Theory*. Without attempting to provide an introduction to category theory, we will limit ourselves here to recalling some fundamental concepts that will be useful in the rest of this discussion.

Definition 25. *A category* $\mathcal{C}$ *consists of:*

1. *a collection* $Obj(\mathcal{C})$ *of objects,*
2. *for each objects* A, B *of* $Obj(\mathcal{C})$*, a collection* $\mathcal{C}(A, B)$ *(if this collection is a set, the category is called locally small) of morphisms* $f : A \rightarrow B$*, satisfying the following properties:*
   (a) *there is a distinguished morphism* $Id_A \in \mathcal{C}(A, A)$ *for any object* A *of* $\mathcal{C}$

(b) for each object A, B, D of $\mathcal{C}$ there is an associative mapping $\circ$ such that if $f : A \to B$ and $g : B \to D$, then $g \circ f : A \to D$ (arrows are composable, see figure below),

(c) in particular, if $f : A \to B$, then $Id_B \circ f = f = f \circ Id_A$.



All constructions are given as properties of morphisms between objects. Here some examples of categories:

Category	Objects	arrows
Set	*Sets*	*functions*
Top	*Topological spaces*	*continuous functions*
Vect	*Vectos spaces*	*linear transformations*
Pos	*Posets*	*monotone functions*

(2)

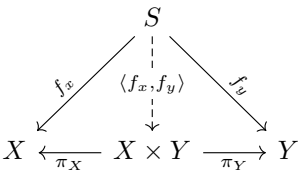
By generalising certain properties of functions (injectivity, surjectivity), we obtain the definition of the following properties of arrows: an arrow (or morphism) $f : A \to B$ is an *isomorphism*, if there is another arrow $g : B \to A$ such that $g \circ f = Id_A$ *and* $f \circ g = Id_B$. Since inverses are unique, we write $g = f^-$. Moreover we have an abstract characterizations of injectivity and surjectivity: an arrow $f : A \to B$ is *monic*, iff given $g, h : C \to A$ we have that $f \circ g = f \circ h$ implies $g = h$. An arrow $f : A \to B$ is instead *epic* , iff given $g, h : B \to D$ we have that $g \circ f = h \circ f$ implies $g = h$. Every isomorphism is both mono and epic. In *Set* the *epic* (right cancellable) arrows are the surjective functions and *monic* (left cancellable) are the injective arrows. Hence arrows both monic and epic are isomorphisms. In other categories this may not be true (il holds in general only that every *iso* is *monic* and *epic*). Rather informally, categories whose *objects* are sets and arrows are functions are called *concrete* and categories whose *class of objects* are (perhaps structured) sets and the class morphisms are (structure-preserving) in turn sets, are called *small*. In the terminology of category theory, *functors* are mapping between categories that map objects to objects in such a way that they preserve domain and codomain, identity arrows and composition, and as categories have morphisms between them –i.e. functors– analogously functors have morphisms between them too, the so-called *natural transformations*. The following are some very basic constructions:

1. *Initial objects*. An object 0 is *initial* in the category $\mathcal{C}$, if for any object A of the category, there is only an arrow $0 \to A$. For example, in the category of sets and functions, since there is only a subset $f \subseteq \emptyset \times A = \emptyset$, for each set A, also there is only one function $\emptyset \to A$, which is $\emptyset$, and therefore the empty set is the only initial object.

2. *Terminal objects*. An object 1 is *terminal* in the category $\mathcal{C}$, if for any object A of the category, there is only an arrow $A \to 1$. For example, in the category of sets and functions the terminal objects are all singletons $\{x\}$: given a set A the function $f(x) = a$, for all $x \in A$, is the only function $f : A \to \{a\}$. In the particular cases of *Mon* and *Grp* (respectively monoids and groups with homomorphisms as arrows), initial objects as well as terminal objects are of the form $\langle \{a\}, *, a \rangle$ with $a * a = a$. In general, in all categories any two initial objects are isomorphic and the same holds for terminal objects.

3. *Elements*. In categories with a terminal object 1, there may be several arrows $1 \to A$, that are called *global elements*, or points.

4. *Products*. A category has *binary products* if for all pairs of objects X, Y, there is an object $X \times Y$ in the category and projections π_X, π_Y such that for all objects S and arrows $f_x : S \to X$ and $f_y : S \to Y$ there is a unique map $\langle f_x, f_y \rangle$ such that the following diagram commutes:

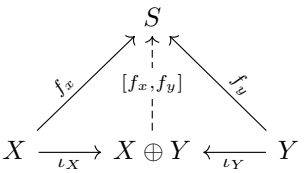


namely $\pi_X \circ \langle f_x, f_y \rangle = f_x$ and $\pi_Y \circ \langle f_x, f_y \rangle = f_y$ and this operation is called the *pairing* of f_x and f_y (dashed arrow means "there is a unique arrow"). A terminal object 1 is a *nullary* product, so, having binary and nullary product we also have unary product, since $A \times 1$ is isomorphic to A.

In the above we say that $\langle \pi_X, \pi_Y \rangle$ is *universal* among pairs arrows from some object, respectively to X and Y, in the sense that any other f_X and f_Y factor uniquely via $\langle f_X, f_Y \rangle$ through $\langle \pi_X, \pi_Y \rangle$. We can therefore say that products are defined by a *universal mapping property*. Other examples of universal mapping properties: terminal and initial objects are defined by how any other object has a unique arrow to or from them.

5. *Coproducts.*

For any category $\mathcal{C}$ the dual (or opposite) category is the category $\mathcal{C}^{op}$ where the objects are the same and the arrows are reversed in direction. The dual notion of *product*, is the *coproduct* with iniections ι_X, ι_Y. If X, Y are objects in a category, a coproduct is an object $X \oplus Y$ with arrows $\iota_X : X \longrightarrow X \oplus Y$ and $\iota_Y : Y \longrightarrow X \oplus Y$ such that for any triplet $X \xrightarrow{f_x} S \xleftarrow{f_y} Y$ we have a unique morphism $[f_x, f_y]$ such that $[f_x, f_y] \circ \iota_X = f_x$ and $[f_x, f_y] \circ \iota_Y = f_y$:

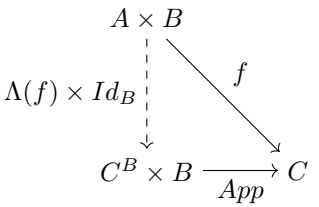


For example, in *Set* the coproduct is:

$$X \oplus Y = \{\langle a, 0\rangle | a \in X\} \cup \{\langle b, 1\rangle | b \in B\}$$

i.e. the disjoint union and the injections $\iota_X(a) = \langle a, 0\rangle$ and $\iota_Y(b) = \langle b, 1\rangle$ and $[f, g](a, 0) = f(a)$, $[f, g](a, 1) = g(a)$. We will not encounter this concept again, but it is important to emphasise one fact. We have decided to deal only with the typed lambda calculus with constructors $\times$ and $\rightarrow$, corresponding to the fragment of intuitionistic propositional calculus with conjunction and implication. We could have extended the Curry-Howard-Lambek isomorphism by adding a type constructor $+$, corresponding to disjunction, i.e. to the coproduct, dual to the product. The correspondence actually extends to coproducts in *bicartesian* closed category, rather than cartesian, namely in cartesian closed categories with the addition of finite coproducts.

6. *Exponentials.* Let us consider for example the category of sets and functions, and the function $f(x, y) : A \times B \rightarrow C$. Now fix $x = a$, so that we have a function $f(a, y) : B \rightarrow C$ i.e. an element of the set C^B of the functions from B to C. We associate to it a map $\Lambda(f) : A \rightarrow C^B$ defined as $a \mapsto f(a, y)$ and uniquely determined by the equation $\Lambda(f)(a)(b) = f(a, b)$. Actually any map $A \rightarrow C^B$ is of this form, coming from some f as above and $\Lambda(f)$ is called *the transpose of f*. Associated to it, there is a map called *evaluation* $App : C^B \times B \rightarrow C$ defined as $App(f, b) = f(b)$. In any category having binary products, an *exponential* for B and C, consists in an object C^B and an arrow App as above, such that the following *universal property* holds: given A and $f : A \times B \longrightarrow C$, there is a unique $\Lambda(f)$ such that $App \circ (\Lambda(f) \times Id_B) = f$.



where the product of morphisms, for $f : A \longrightarrow B$ and $g : C \longrightarrow D$, is the morphism $f \times g : A \times C \longrightarrow B \times D$ defined as $f \times g = \langle f \circ \pi_A, g \circ \pi_C \rangle$, where π_A and π_C are respectively the first and the second projection of $A \times C$. The process of transforming a function of two arguments into a function of one argument is known as *currying*.

The notion of products can be generalised to arbitrary arities. A category is said to have *all finite products* if it has a terminal object and products of any finite cardinality. Actually a terminal object can be seen as a *zero-ary* product: indeed, a zero-ary product is an object t without projections and for any object S there is a unique $S \longrightarrow t$; hence $t = 1$. A *unary* product of A is an object t with a unique projection $t \longrightarrow A$ such that for any $f : S \longrightarrow A$ there is a unique $u : S \longrightarrow t$ such that $\pi \circ u = f$. Take $t = A$ and $\pi = Id_A$. For a formal proof of this, see Mac Lane (1971), p. 73.

Definition 26. *A category is* cartesian closed (CCC) *if it has all finite products and exponentials (i.e. exponent object, evaluation and transposes such that the diagram in the figure above commutes).*

Examples of CCC. Consider a single poset $\langle P, \leq \rangle$ as a category in itself, where $a \to b$ iff $a \leq b$. A terminal object is a largest element 1, product is the infimum $a \wedge b$ and the exponential, usually written $a \Rightarrow b$ satisfies:

$$x \leq (a \Rightarrow b) \; iff \; x \wedge b \leq a$$

that is, $a \Rightarrow b$ is the largest element $x \in P$ such that $x \wedge b \leq a$. If the poset has also an initial object 0 and coproducts $a \vee b$, it is an *Heyting algebra*, i.e. a model of intuitionistic propositional logic, where $p \Rightarrow 0$ is called *pseudo-complement* of p and denoted $\neg p$. If moreover $\neg p \vee p = 1$, then it is a boolean algebra, i.e. a model of classical propositional logic. Another example of CCC is just the category *Pos* of posets and monotone functions. We consider:

1. the poset $P \times Q$ has pairs as its elements, orderd as follows $\langle a, b \rangle \leq \langle c, d \rangle$ iff $a \leq c$ and $b \leq d$. Show by exercise that if $f : X \to P$ and $g : X \to Q$ are monotone, also the projections are monotone.

2. The poset $Q^P = \{f | f : P \to Q, \; f \; monotone\}$ whose elements are ordered pointwise: $f \leq g$ iff for all a, $f(a) \leq g(a)$.

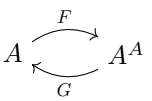
3. *App* and $\Lambda(f)$ defined as above are monotone. Let $p \leq q$ and $\langle f, p \rangle \leq \langle g, q \rangle$, we have:

   $$App(f, p) \leq f(p) \leq f(q) \leq g(q) = App(g, q)$$

   If $x \leq y$, then $\Lambda(f)(x)(p) = f(x, p) \leq f(y, p) = \Lambda(f)(y)(p)$

4. Terminal objects are singleton posets $\langle \{a\}, \{\langle a, a \rangle\} \rangle$. Note that for any poset P there is a unique map $P \to \{a\}$.

Reflexive objects. An object A of a CCC is called *reflexive*, if there are $F : A \to A^A$ and $G : A^A \to A$ such that $F \circ G = Id_{A^A}$



Scott (1980) proved that models of the untyped calculus may be defined as reflexive objects in cartesian closed categories. We address this topic later. Rather, we deal here with calculus *with types*.

Categorical semantic of simply typed lambda calculus. Based on the concepts introduced, we are able to establish a correspondence between Logic, Computability and Categories. We are going to define a *model* of the language of typed lambda-calculus in a cartesian closed category: let therefore $\mathcal{C}$ be a *cartesian closed category*; let us fix an interpretation $[\![\tau]\!]$ of base types τ as objects of the category. For the purposes of an interpretation in cartesian closed categories let us add a *unit type* 1 and a unit value a. The unit value is the only term of type unit (similar to *void* type in certain programming languages). We see that the models of the simply typed lambda calculus with product type and unit are exactly the cartesian closed categories. We say that it is the *internal language* of CCC's. First, we formulate a formal system for the calculus of the $\beta\eta$-conversion as a set of rules. *Judgements* are expressions of the form $\Gamma \vdash t : \alpha$, whose meaning is "t is a term of type α in the context Γ"; this relation is inductively defined by the following rules:

1. Rules that say that the conversion $=$ is an equivalence (i.e. reflexivity, transitivity and simmetry).

2. Unit type

$$\frac{\Gamma \vdash t : 1}{\Gamma \vdash t = a : 1.}$$

3. Product rules:

(a) $$\frac{\Gamma \vdash t = s : \alpha \quad \Gamma \vdash r = h : \beta}{\Gamma \vdash \langle t, r\rangle = \langle s, h\rangle : \alpha \times \beta}$$

(b) $$\frac{\Gamma \vdash s = t : \alpha \times \beta}{\Gamma \vdash \pi_0 s = \pi_0 t : \alpha} \qquad \frac{\Gamma \vdash s = t : \alpha \times \beta}{\Gamma \vdash \pi_1 s = \pi_1 t : \beta}$$

(c) $$\frac{\Gamma \vdash t : \alpha \quad \Gamma \vdash u : \beta}{\Gamma \vdash \pi_0\langle u, t\rangle = u : \alpha} \qquad \frac{\Gamma \vdash t : \alpha \quad \Gamma \vdash u : \beta}{\Gamma \vdash \pi_1\langle u, t\rangle = t : \beta}$$

4. $\eta-$extensional rules:

(a) $$\frac{\Gamma \vdash t : \alpha \to \beta}{\Gamma \vdash \lambda x : \alpha.tx = t : \alpha \to \beta}$$, where x is not among the free variables of t.

(b) $$\frac{\Gamma \vdash t : \alpha \times \beta}{\Gamma \vdash t = \langle \pi_0 t, \pi_1 t\rangle : \alpha \times \beta}$$

5. $\beta-$conversion and application rules:

(a) $$\frac{\Gamma, x : \alpha \vdash t = u : \beta}{\Gamma \vdash (\lambda x : \alpha.t) = (\lambda x : \alpha.u) : \alpha \to \beta}$$

(b) $$\frac{\Gamma \vdash s = t : \alpha \to \beta \quad \Gamma \vdash u = v : \alpha}{\Gamma \vdash su = tr : \beta}$$

(c) $$\frac{\Gamma, x : \alpha \quad \Gamma \vdash s : \alpha}{\Gamma \vdash (\lambda x : \alpha.t)s = t[s/x] : \beta}$$

These rules are valid under the below interpretation in arbitrary cartesian closed category. Fix therefore a CCC and suppose we are given an assignment of an object to each base type. Then we define by induction:

1. $[\![\alpha \to \beta]\!] = [\![\alpha]\!] \to [\![\beta]\!]$ (hence, a morphism between the objects that interpret α and β).

2. $[\![\alpha \times \beta]\!] = [\![\alpha]\!] \times [\![\beta]\!]$.

3. $[\![1]\!] = 1_C$ (where 1_C is a terminal object of the category)

A judgement $\Gamma \vdash t : \alpha$, where Γ has the form $x_0 : \alpha_0, ..., x_n : \alpha_n$ will be interpreted as a morphism:

$$[\![\alpha_0]\!] \times [\![\alpha_1]\!] \times ... \times [\![\alpha_n]\!] \to [\![\alpha]\!]$$

We now list the various terms formation rules (in sequential style) together with their interpretations.

1. *Variables.* $x_0 : \alpha_0, ..., x_n : \alpha_n \vdash x_i : \alpha_i$ will be intepreted in CCC as projections:

$$\pi_i : [\![\alpha_0]\!] \times [\![\alpha_1]\!] \times ... \times [\![\alpha_n]\!] \to [\![\alpha_i]\!]$$

2. *Unit.* $\Gamma \vdash a : 1$ is intepreted as the unique morphism from $[\![\Gamma]\!]$ to 1_C.
3. *Abstraction.*

$$\frac{\Gamma, x : \alpha \vdash t : \beta}{\Gamma \vdash \lambda x.t : \alpha \to \beta}$$

Let f the morphism that intepretes the premiss; then the conclusion is intepreted as the traspose $\Lambda(f)$, so that $App \circ (\Lambda(f) \times Id_{[\![\alpha]\!]}) = f$. Note that, if A, B, C are the objects that intepret respectively α, β, Γ, then:

$$C \times A \xrightarrow{\Lambda(f) \times Id_A} B^A \times A \xrightarrow{App} B$$

and that $f : C \times A \longrightarrow B$ and $\Lambda(f) : C \to B^A$.

4. *Application.*

$$\frac{\Gamma \vdash t : \alpha \to \beta \quad \Gamma \vdash s : \alpha}{\Gamma \vdash ts : \beta}$$

If $f : C \longrightarrow B^A$ and $g : C \longrightarrow A$ are the interpretations of the premisses, then the conclusion is intepreted as $App \circ \langle f, g \rangle$:

$$C \xrightarrow{\langle f,g \rangle} B^A \times A \xrightarrow{App} B$$

Recall that we are in CCC, and therefore if $h : C \longrightarrow P$ and $k : C \longrightarrow Q$, there exists a unique map $\langle h, k \rangle : C \longrightarrow P \times Q$, such that $\pi_0 \circ \langle h, k \rangle = h$ and $\pi_1 \circ \langle h, k \rangle = k$.

5. *Product.*

$$\frac{\Gamma \vdash t : \alpha \quad \Gamma \vdash s : \beta}{\Gamma \vdash \langle t, s \rangle : \alpha \times \beta}$$

If $f : C \longrightarrow A$ and $g : C \longrightarrow B$ are the interpretations of the premisses, take $\langle f, g \rangle : C \longrightarrow A \times B$ as the intepretation of conclusion.

6. *Projections.*

$$\frac{\Gamma \vdash t : \alpha \times \beta}{\Gamma \vdash \pi_0 t : \alpha}$$

If $f : C \longrightarrow A \times B$ intepretes the premiss, then $\pi_0 \circ f : C \longrightarrow A$ intepretes the conclusion:

$$C \xrightarrow{f} A \times B \xrightarrow{\pi_0} A$$

(Analogously for the second projection)

Substitution as composition of morphisms. Let us consider a context $x_0 : \alpha_0, ..., x_n : \alpha_n$, interpreted as the product $A_0 \times ... \times A_n$. A substitution of terms $t_0, ..., t_n$ for the variables $x_0, ..., x_n$, where for all $i \leq n$, $\Delta \vdash t_i : \alpha_i$ and these are interpreted as morphisms $g_i : C \longrightarrow A_i$, is a morphism:

$$C \xrightarrow{\langle g_0, ..., g_n \rangle} A_0 \times ... \times A_n$$

Lemma 23. *If $f : A_0 \times ... \times A_n \times A \longrightarrow B$ interprets $\Gamma, x : \alpha \vdash t : \beta$, and $g : A_0 \times ... \times A_n \longrightarrow A$ the interpretation of $\Gamma \vdash s : \alpha$; then the interpretation of $\Gamma \vdash t[s/x] : \beta$ is $f \circ \langle Id_{A_0 \times ... \times A_n}, g \rangle$:*

$$A_0 \times ... \times A_n \xrightarrow{\langle Id_{A_0 \times ... \times A_n}, g \rangle} A_0 \times ... \times A_n \times A \xrightarrow{f} B$$

Proof. By structural induction on t. QED

Theorem 53. (Soundness of the categorical semantic) *The interpretation is sound, namely, if $\Gamma \vdash t = s : \alpha$ is provable, then $[\![\Gamma \vdash t : \alpha]\!] = [\![\Gamma \vdash s : \alpha]\!]$.*

Proof. We will actually show the most significant cases, leaving the others as exercises.

(i) As for the unit case, the result follows from the fact that there is a unique morphism $[\![\Gamma]\!] \longrightarrow 1$.

(ii) The case 5.a is proved arguing by induction on the hight of derivation: suppose by inductive hypothesis the theorem holds for the premiss, i.e. that $[\![\Gamma, x : \alpha \vdash t : \beta]\!] = [\![\Gamma, x : \alpha \vdash u : \beta]\!]$; hence:

$$[\![\Gamma \vdash \lambda x : \alpha.u : \alpha \to \beta]\!] = \Lambda([\![\Gamma, x : \alpha \vdash u : \beta]\!]) =$$

$$= \Lambda([\![\Gamma, x : \alpha \vdash t : \beta]\!]) = [\![\Gamma \vdash \lambda x : \alpha.t : \alpha \to \beta]\!]$$

The case 5.b is proved in a similar way, reasoning again by induction. As for $\beta-$conversion 5.c, let $\Gamma = x_0 : \alpha_0, ..., x_n : \alpha_n$ and suppose that $\Gamma, x : \alpha \vdash t : \beta$ and $\Gamma \vdash s : \alpha$ are intepreted resp. by $f : A_0 \times ... \times A_n \times A \longrightarrow B$ and $g : A_0 \times ... \times A_n \longrightarrow A$. By definition:

$$[\![\Gamma \vdash (\lambda x : \alpha.t)s : \beta]\!] = App \circ \langle \Lambda(f), g \rangle$$

But this is equal to $App \circ (\Lambda(f) \times Id_A) \circ \langle Id_{A_0 \times ... \times A_n}, g \rangle$:

$$A_0 \times ... \times A_n \xrightarrow{\langle Id_{A_0 \times ... \times A_n}, g \rangle} A_0 \times ... \times A_n \times A \xrightarrow{\Lambda(f) \times Id_A} B^A \times A \xrightarrow{App} B$$

Since in CCC $App \circ (\Lambda(f) \times Id_A) = f$, lastly we get $f \circ \langle Id_{A_0 \times ... \times A_n}, g \rangle$, namely:

$$[\![\Gamma, x : \alpha \vdash t : \beta]\!] \circ \langle Id_{A_0 \times ... \times A_n}, g \rangle$$

But from the substitution lemma it follows that this is equal to $[\![\Gamma \vdash t[s/x] : \beta]\!]$. Hence $[\![\Gamma \vdash (\lambda x : \alpha.t)s : \beta]\!] = [\![\Gamma \vdash t[s/x] : \beta]\!]$.

(iii) As for $\eta - reduction$ 4.a, observe that:

$$[\![\Gamma \vdash \lambda x : \alpha.tx]\!] = \Lambda([\![\Gamma, x : \alpha \vdash tx]\!]) = \Lambda(App([\![\Gamma \vdash t]\!] \times Id_A) = [\![\Gamma \vdash t]\!]$$

As regards 4.b, just observe that:

$$[\![\Gamma \vdash \langle \pi_0 t, \pi_1 t \rangle : \alpha \times \beta]\!] = \langle \pi_0 \circ [\![\Gamma \vdash t : \alpha \times \beta]\!], \pi_1 \circ [\![\Gamma \vdash t : \alpha \times \beta]\!] \rangle =$$

$$= [\![\Gamma \vdash t : \alpha \times \beta]\!]$$

(iv) As regards to product case 3.c. note that:

$$[\![\Gamma \vdash \pi_0(\langle u, v \rangle) : \alpha]\!] = \pi_0 \circ [\![\Gamma \vdash \langle u, v \rangle : \alpha \times \beta]\!] =$$

$$= \pi_0 \circ \langle [\![\Gamma \vdash u : \alpha]\!], [\![\Gamma \vdash v : \beta]\!] \rangle = [\![\Gamma \vdash u : \alpha]\!]$$

(v) Analyse the other cases by exercise.

QED

With regard to *completeness*, we provide a sketch of the proof, which the reader can find in more detail in Abramsky and Tzevelekos (2011), 62-64, which reproduces and simplifies the one in ch.4.8 of Crole (1994). We just sketch the basic intuition. Our aim is to find a category in which true equalities correspond precisely to provable conversions between terms in the calculus. Such a categorical model is the *term model*, or *classifying category*, the "free category" generated by the syntax of the calculus:

1. the *objects* of this category $\mathbb{C}$ are types, plus a terminal object 1;

2. the *morphisms* $\alpha \longrightarrow \beta$ are *equivalence classes* of pairs variable-term x, t such that x, t and y, s are considered equivalent if both $x : \alpha \vdash t : \beta$ and $y : \alpha \vdash s : \beta$ are provable and t can be converted in $s[x/y]$. It is not hard to show that this actually is an equivalence. Special cases are considered, when the first element of the pair is empty: hence we say that $-, t$ and $-, s$ are equivalent if $\vdash t : \beta$ and $\vdash s : \beta$ are provable and t can be converted in s. We denote with $(x|t)$ the equivalence class of the couple x, t. Hence $\mathbb{C}(\alpha, \beta)$ will be the set of equivalence classes $(x|t)$ such that $x : \alpha \vdash t : \beta$ is provable. In particular $\mathbb{C}(1, \beta)$ is the set of equivalence classes $(-|t)$ such that $\vdash t : \beta$ is provable and $\mathbb{C}(\beta, 1)$ is the singleton of the terminal arrow $1_\alpha : \alpha \longrightarrow 1$.

3. Identities are $Id_\alpha = (x|x)$, if $\alpha \neq 1$, and $Id_\alpha = 1_\alpha$, if $\alpha = 1$.

4. The composition of arrows is defined as $(x|t) \circ (y|u) = (y|t[u/x])$, with the special cases of $(-|t) \circ 1_\alpha = (y|t)$, if $\alpha \neq 1$, and $(-|t) \circ 1_\alpha = (-|t)$, if $\alpha = 1$, and of $1_\beta \circ f = 1_\alpha$, where $f \in \mathbb{C}(\alpha, \beta)$.

Some facts:

1. Composition is associative. Indeed, for the substitution lemma at p. 61, the the following terms (a) and (b) are equal:

   (a) $(x|t) \circ ((y|u) \circ (z|y)) = (x|t) \circ (z|u[v/y]) = (z|t[u[v/y]/x])$

   (b) $((x|t) \circ (y|u)) \circ (z|v) = (y|t[u/x]) \circ (z|v) = (z|t[u/x][v/y])$

   So what we have defined is a *category*. It is actually a *cartesian closed* category:

2. The category $\mathbb{C}$ has binary products. We define projections π_i in $\alpha \xleftarrow{\pi_0} \alpha \times \beta \xrightarrow{\pi_1} \beta$ as $\pi_i = (x|\pi_i x)$ and, given a triple $\alpha \xleftarrow{(x|t)} \mu \xrightarrow{(x|u)} \beta$, we define the morphism $\langle (x|t), (x|u) \rangle : \mu \longrightarrow \alpha \times \beta$ as $(x|\langle t, u \rangle)$. Hence, for instance we have $\pi_1 \circ \langle (x|t), (x|u) \rangle = (u|\pi_1 y) \circ (x|\langle t, u \rangle) = (x|\pi_1, u\rangle) = (x|t)$. Readers can practise demonstrating uniqueness.

3. The category $\mathbb{C}$ has exponentials too. Let us define App as $(x|(\pi_0 x)(\pi_1 x))$ and, given $(x|t) : \sigma \times \alpha \longrightarrow \beta$, let us define $\Lambda((x|t)) = (x_0|\lambda x_1.t[\langle x_0, x_1 \rangle / x])$, so that:

$$
\begin{aligned}
App \circ \Lambda((x|t)) \times Id &= App \circ \langle \Lambda((x|t) \circ \pi_0, Id \circ \pi_1 \rangle \\
&= App \circ \langle (x_0|\lambda x_1.t[\langle x_0, x_1 \rangle / x]) \circ (y|\pi_0 y), (y|\pi_1 y) \rangle \\
&= App \circ \langle (y|\lambda x_1.t[\langle \pi_1 y, x_1 \rangle / x]), (y|\pi_1 y) \rangle \\
&= (z|(\pi_0 z)(\pi_1 z)) \circ (y|\langle \lambda x_1.t[\langle \pi_1 y, x_1 \rangle / x], \pi_1 y \rangle) \\
&= \langle y, (\pi_0 X)(\pi_1 X) \rangle = (y|(\lambda x_1.t[\langle \pi_0 y, x_1 \rangle / x](\pi_1 y) \\
&= (u|t[\langle \pi_0 y, \pi_1 y \rangle / x]) = (y|t[y/x]) = (x|t)
\end{aligned}
\tag{3}
$$

   where $X = \langle \lambda x_1.t[\langle \pi_0 y, x_1 \rangle / x], \pi_1 u \rangle$, and $\Lambda((x|t)) : \sigma \longrightarrow \beta^\alpha$; moreover $App : \beta^\alpha \times \alpha \longrightarrow \beta$ and $Id = Id_\alpha$. Note that we have used $\eta-$reduction at the penultimate step. Uniqueness and other cases left to the reader.

Now, using the interpretation of lambda terms in categories that we used in the soundness theorem, we can easily verify that:

$$[\![\Gamma \vdash t : \tau]\!] = (x|t[\pi_0/x_0, ..., \pi_n/x_n])$$

where x does not occur in $\Gamma = \{x_0 : \tau_0, ..., x_n : \tau_n\}$ and $x : \tau_0 \times ... \times \tau_n$. From this follows that if $[\![\Gamma \vdash t : \alpha]\!] = [\![\Gamma \vdash s : \alpha]\!]$ holds in it, then $\Gamma \vdash t = s : \alpha$ is provable and together with soundness, this implies that in this model true equalities coincide with provable conversions.

3.5. Categorical models for untyped lambda calculus

Various semantics for the untyped lambda calculus have been studied. Of these, in particular, the *Scott continuous semantics* is based on the class of reflexive objects in the cartesian closed category CPO whose objects are complete partial orders and morphisms are Scott continuous functions. This category actually has reflexive objects. More in general, a categorical model of the untyped λ-calculus is a reflexive object of a Cartesian closed category. Since in untyped lambda calculus self-application xx is admitted, terms can be seen both as objects as well as functions, so in the study of models we will address the so-called domain equation $D \cong (D \to D)$. Clearly, by Cantor's theorem $(D \to D)$ cannot be the whole space of functions from D to D (the only case in which holds is when D is a singleton). We will illustrate now Scott's method for the concrete cartesian closed category CPO of complete partial orders. However the method works for all concrete cartesian closed categories.

Definition 27. *Let $\langle P, \leq \rangle$ be a poset. A subset $X \subseteq P$ is directed iff for all $x, y \in X$ there exists $z \in X$ such that $x \leq z$ and $y \leq z$. The poset is a cpo iff:*

1. *there is a bottom element $\bot$, i.e. for all $x \in P$, $\bot \leq x$.*

2. *If $X \subseteq P$ is directed, then the supremum $\sqcup X \in P$ exists.*

Definition 28. (Scott topology) *The Scott topology on a cpo $\langle P, \leq \rangle$ is defined as follows. A subset $\mathcal{O} \subseteq P$ is open, iff:*

1. *$x \in \mathcal{O}$ and $x \leq y$, then $y \in \mathcal{O}$ (closure upwards).*

2. *If X is directed and $\sqcup X \in \mathcal{O}$, then $X \cap \mathcal{O} \neq \emptyset$ (inaccessibility by directed joins).*

Since according to this definition the empty set and P are open, and opens are closed under arbitrary unions and if $\mathcal{O}, \mathcal{O}'$ are opens, then $\mathcal{O} \cap \mathcal{O}'$ is open, we have that this definition actually determines a topology in the usual sense. Moreover the sets $U_x = \{z | z \not\leq x\}$ are clearly opens.

Continuous maps.

Theorem 54. . *Let D, D' be two cpo's and $f : D \to D'$. Hence f is continuous in the usual sense (i.e. $f^{-1}(Y) = \{x \in D | f(x) \in Y\}$ is open, when Y is open), iff for all directed $X \subseteq D$, $f(\sqcup X) = \sqcup f(X)$, where $f(X) = \{f(x) | x \in X\}$.*

Proof. . $\Rightarrow$ Let f be continuous. We see that it is also monotonic: actually, if $x \leq y$, but $f(x) \not\leq f(y)$, then $f(x) \in U_{f(y)} = \{z | z \not\leq f(y)\}$. Since $U_{f(y)}$ is open and f is continuous, by definition $f^{-1}(U_{f(y)})$ is open too, and x belong to it. Hence by closure upwards, also y belong to it and $f(y) \in U_{f(y)}$ (contradiction). So f is monotonic. It follows that $x \in X$, $f(x) \leq f(\sqcup X)$ and therefore $\sqcup_{x \in X} f(x) \leq f(\sqcup X)$. Now suppose that the other way round does not hold. Hence $f(\sqcup X) \in U_{\sqcup f(X)}$ by definition, and therefore $\sqcup X \in f^{-1}(U_{\sqcup f(X)})$. Now, since X is directed and $f^{-1}(U_{\sqcup f(X)})$ is open, then $X \cap f^{-1}(U_{\sqcup f(X)})$ is not empty. Take x in this intersection. It follows that $f(x) \in U_{\sqcup f(X)}$ (contradiction). $\Leftarrow$ Let $f(\sqcup X) = \sqcup f(X)$. Since $x \leq y$ implies $y = x \sqcup y$, we have $f(y) = f(x \sqcup y)$ and by hypothesis $f(x \sqcup y) = f(x) \sqcup f(y)$, from which monotonicity, $f(x) \leq f(y)$. Observe that if $X \subseteq D$ is directed and $\sqcup X \in f^{-1}(\mathcal{O})$, then (since by hypothesis $f(\sqcup X) = \sqcup f(X)$) we have $\sqcup f(X) \in \mathcal{O}$. But $f(X)$ is directed and therefore by definition $f(X) \cap \mathcal{O} \neq \emptyset$, from which

follows $X \cap f^{-1}(\mathcal{O}) \neq \emptyset$. But $f^{-1}(\mathcal{O})$ is also upwards closed. It follows that $f^{-1}(\mathcal{O})$ is open and therefore f is continuous.

Remarks. Continuous maps in cpo's are always monotone.

We now show that the category CPO of cpo's and continuous maps is cartesian closed. To this purpose, it is necessary to show some facts, but first we make two further remarks. QED

Now we show that CPO, the category of cpo's and continuous maps as arrows, is cartesian closed. We need these results:

Theorem 55. *The set $[D \to D']$ of continuous maps from D to D' is a cpo, where:*

1. *it is ordered pointwise:*
$$f \leq g \; iff \; \forall d \in D(f(d) \leq g(d))$$
2. *$\bot = D \times \{\bot_{D'}\}$ is the function $x \mapsto \bot_{D'}$,*
3. *If $X \subseteq [D \to D']$ is directed, then $\sqcup X$ is the map $x \mapsto \sqcup\{f(x) | f \in X\}$.*

Proof. We just show that in 3. if $X \subseteq [D \to D']$ is directed, then $\sqcup X$ is continuous. Note that if X is directed, then for all $f, g \in X$ there is an h such that for all $a \in D$, $f(a) \leq h(a)$ and $g(a) \leq h(a)$. Let $f(p) = \sqcup\{f_i(p) | f_i \in X\} = \sqcup X(p)$. Hence:

$$f(\sqcup_j p_j) = \sqcup_i f_i(\sqcup_j p_j) = \sqcup_i \sqcup_j f_i(p_j) = \sqcup_j \sqcup_i f_i(p_j) = \sqcup_j f(p_j)$$

Theorem 56. *If D, D' are cpo's, then $D \times D'$, is a cpo, where:*

1. *the bottom is $\langle \bot, \bot \rangle$*
2. *$\langle a, b \rangle \leq \langle c, d \rangle$ iff $a \leq c$ and $b \leq d$,*
3. *if $X \subseteq D \times D'$ is directed, then $\sqcup X = \langle \sqcup X_0, \sqcup X_1 \rangle$, where:*
   (a) $X_0 = \{x \in D | \langle x, a \rangle \in X, \; for \; some \; a \in D'\}$
   (b) $X_1 = \{x \in D' | \langle a, x \rangle \in X, \; for \; some \; a \in D\}$

Theorem 57. *Let D, D', B be cpo's. Hence $f : (D \times D') \to B$ is continuous iff it is continuous in both arguments separately, i.e. iff both $d \mapsto f(d, d')$ and $d' \mapsto f(d, d')$ are continuous.*

We omit the proofs of these results, rather standard (see Barendregt's handbook pp. 10-3).

Theorem 58. (Continuity of application) *Let $B^A = [A \to B]$ and A, B, C cpo's. Hence the map $App : B^A \times A \to C$ defined as $App(f, a) = f(a)$ is continuous.*

Proof. Let us consider the components $\lambda f.f(x)$ (namely the function $f \mapsto App(f, x)$) and $\lambda x.f(x)$ (namely the function $x \mapsto App(f, x)$). The latter is clearly continuous, as for the former, we argue as follows. Let $X \subseteq B^A$ be directed and let $h = \lambda f.f(x)$; hence:

$$h(\sqcup X) = \lambda f.f(x)(\sqcup X) = (\sqcup X)(x) =$$

$$= \sqcup\{f(x) | f \in X\} = \sqcup\{h(f) | f \in X\} = \sqcup h(X)$$

Hence *App* is continuous in both components and by the previous result, it is continuous itself. QED

Theorem 59. (Continuity of abstraction) *Let $f \in [B \times A \to C]$ and $\Lambda(f)(x) = \lambda y.f(x, y)$ (i.e. the map that send x in $y \mapsto f(x, y)$). It follows that $\Lambda(f)$ is continuous.*

Proof. Observe that $\Lambda(f) : A \to [B \to C]$. Let $X \subseteq A$ be directed; hence:

$$\Lambda(f)(\sqcup X) = \lambda y.f(\sqcup X, y) = \lambda y.\, \sqcup_{x \in X}\, f(x,y) =$$

$$= \sqcup_{x \in X}(\lambda y.f(x,y)) = \sqcup_{x \in X}\Lambda(f)(X)$$

QED

Theorem 60. *Also the mapping $f \mapsto \Lambda(f)$ is continuous.*

Proof. Let $H = \lambda f.\Lambda(f)$. Then for $X \subseteq [B \times A \to C]$ directed. Hence:

$$H(\sqcup X) = \lambda x \lambda y.(\sqcup X)(x,y) = \lambda x \lambda y.(\sqcup_{f \in X} f(x,y)) =$$

$$= \sqcup_{f \in X} \lambda x \lambda y.f(x,y) = \sqcup H(X)$$

QED

So we have that if A, B are cpo's, also $A \times B$ and B^A are cpo's. The singleton cpo $\langle \{a\}, \{\langle a, a\rangle\}\rangle$ works as terminal object. Moreover *App* is continuous and for every continuous $f : A \times B \to C$, there is a unique continuous $\Lambda(f) : A \to [B \to C]$ such that the diagram of CCC. commutes. Hence we have the followiung result:

Corollary 13. CPO *is cartesian closed.*

We now consider reflexive cpo's A, namely equipped with continuous maps:

$$A \underset{G}{\overset{F}{\rightleftarrows}} A^A$$

Such that $F \circ G = Id_{A^A}$, where $A^A = [A \to A]$. We say that $[A \to A]$ is a *retract* of A.

We define an interpretation of terms $[\![t]\!]^\sigma$ on the basis of an evaluation $\sigma : Var \to A$ of variables:

1. $[\![x]\!]^\sigma = \sigma(x)$
2. $[\![ts]\!]^\sigma = F([\![t]\!]^\sigma)([\![s]\!]^\sigma)$
3. $[\![\lambda x.t]\!]^\sigma = G(d \mapsto [\![t]\!]^{\sigma[d/x]})$

This definition is correct, only if we prove the following:

Lemma 24. *The function $\lambda d.[\![t]\!]^{\sigma[d/x]}$, namely $d \mapsto [\![t]\!]^{\sigma[d/x]}$ is continuous.*

Proof. Induction on t. The only interesting case is when $t = \lambda y.s$. By (IH) we have that $h(d,c) = [\![s]\!]^{\sigma[d/x, c/y]}$ is continuous separately in d and c; hence by previous a result also $h(d,c)$ is continuous. Hence:

$$[\![\lambda y.s]\!]^{\sigma[d/x]} = G(c \mapsto h(d,c)) = G(\Lambda(h)(d))$$

namely the composition of continuous maps, itself continuous. QED

Also by induction on the complexity of terms it is provable the following substitution result:

$$[\![t[s/x]]\!]^\sigma = [\![t]\!]^{\sigma[[\![s]\!]^\sigma/x]}$$

From which follows the main result:

Theorem 61. *If $t \Rightarrow s$, then $[\![t]\!]^\sigma = [\![s]\!]^\sigma$.*

Proof. . We see just the case of contraction:

$$[\![(\lambda x.t)s]\!]^\sigma = F([\![(\lambda x.t)]\!]^\sigma)([\![s]\!]^\sigma) =$$

$$= F(G(d \mapsto ([\![t]\!]^{\sigma[d/x]}))([\![s]\!]^\sigma) = (d \mapsto ([\![t]\!]^{\sigma[d/x]})([\![s]\!]^\sigma) =$$

$$= [\![t]\!]^{\sigma[[\![s]\!]^\sigma/x]} = [\![t[s/x]]\!]^\sigma$$

QED

The extensionality axiom $\lambda x.tx$, where $x \notin FVar(t)$, holds only if $G \circ F = Id_A$ also holds, namely if we have an isomorphism $A \cong [A \to A]$. In 1969 Dana Scott introduced the model D_∞ satisying this isomorphism. In the 70s Scott and Plotkin introduced the easier, but non extensional *graph model* $\langle \mathcal{P}(\omega), \subseteq \rangle$ of subsets of natural numbers ordered by set-theoretical inclusion. Both models can be constructed in the category **CPO**. Here we illustrate some property of the graph-model. Actually, this is a cpo, with $\cup$ as supremum. Being a complete lattice, supremum and infimum exist for any set. We start showing some basic properties.

Theorem 62. *A function $f : \mathcal{P}(\omega) \to \mathcal{P}(\omega)$ is continuous iff for every $A \subseteq \omega$:*

$$f(A) = \bigcup_{D \subseteq_{finite} A} f(D)$$

Proof. $\Rightarrow$ by monotonicy if $D \subseteq A$, then $f(D) \subseteq f(A)$, hence $f(A) \supseteq \bigcup_{D \subseteq_{finite} A} f(D)$. For the other way round, consider $A_n = A \cap \{0, ..., n\}$, so that $A = \bigcup_n A_n$. Hence by continuity:

$$f(A) = f(\bigcup_n A_n) = \bigcup_n f(A_n)$$

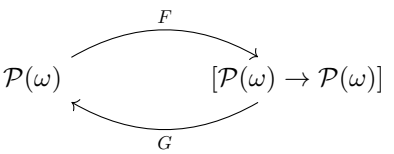
Since the A_n 's are finite, it follows $f(A) \subseteq \bigcup_{D \subseteq_{finite} A} f(D)$

$\Leftarrow$ Suppose that $f(A) = \bigcup_{D \subseteq_{finite} A} f(D)$. Note that if $A \subseteq B$ and $D \subseteq_{finite} A$, then $D \subseteq_{finite} B$, from which $f(A) \subseteq f(B)$. Take a directed set of sets A_0, A_1, A_2.... Hence:

$$f(\bigcup_n A_n) = \bigcup \{f(D) | D \subseteq_{finite} \bigcup_n A_n\} =$$

$$= \bigcup_n \bigcup \{f(D) | D \subseteq_{finite} A_n\} = \bigcup_n f(A_n)$$

Retraction pair. Now we define the retraction pair F, G as follows:



Let $D_x = \{x_0, ..., x_n\}$, where $x = 2^{x_0} + ... + 2^{x_n}$ and $x_0 < ... < x_n$. Let us define:

1. $F(S)(X) = \{m | \exists n(D_m \subseteq X \wedge \langle n, m \rangle \in S)\}$
2. $G(f) = \{\langle n, x \rangle | x \in f(D_n)\}$

Lemma 25. $F \circ G = Id_{[\mathcal{P}(\omega)\to\mathcal{P}(\omega)]}$
Let $f \in [\mathcal{P}(\omega) \to \mathcal{P}(\omega)]$. Hence:

$$F(G(f))(X) = \{x|\exists n(\langle x,n\rangle \in G(f) \wedge D_n \subseteq X)\} =$$

$$= \{x|\exists n(x \in f(D_n) \wedge D_n \subseteq X)\} =$$

$$= \bigcup\{f(D_n)|D_n \subseteq X\} = f(X)$$

Proof. By the previous result. QED

Now we show that F and G are continuous.

Theorem 63. *The maps F and G, defined as above, are both continuous.*

Proof. 1. Let $X \subseteq [\mathcal{P}(\omega) \to \mathcal{P}(\omega)]$. Then:

$$G(\bigsqcup X) = \{\langle n,m\rangle|m \in (\bigsqcup X)(D_n)\} = (by\ definition\ of\ sup.)$$

$$= \{\langle n,m\rangle|m \in \bigcup_{f\in X} f(D_n)\} =$$

$$= \bigcup_{f\in X}\{\langle n,m\rangle|m \in f(D_n)\} =$$

$$= \bigcup_{f\in X} G(f) = \bigsqcup G(X)$$

2. Let $X \subseteq \mathcal{P}(\omega)$. Then:

$$F(Z)(\bigcup_i X_i) = \{a|\exists n(\langle n,a\rangle \in Z \wedge D_n \subseteq \bigcup_i X_i)\} =$$

$$= \{a|\exists_i \exists n(\langle n,a\rangle \in Z \wedge D_n \subseteq X_i)\} =$$

$$= \bigcup_i \{a|\exists n(\langle n,a\rangle \in Z \wedge D_n \subseteq X_i)\} =$$

$$= \bigcup_i F(Z)(X_i)$$

QED

Corollary 14. *$\langle \mathcal{P}(\omega), \subseteq\rangle$ with the retraction pair F, G and the interpretation $[\![x]\!]$ is a model of the type free lambda calculus.*

Note the disparity between the countable amount of terms of the language, and the uncountable amount of object of the model. An interesting development is to restrict the model to the set $\mathcal{E}$ of recursively enumerable sets.

Definition 29. *A continuous $f : \mathcal{P}(\omega) \to \mathcal{P}(\omega)$ is effective iff $G(f)$ is recursively enumerable.*

Theorem 64. *Continuous effective f's, maps $\mathcal{E}$ in $\mathcal{E}$.*

Proof. $x \in f(A)$ iff $\exists n(\langle x,n\rangle \in G(f) \wedge D_n \subseteq A)$ (where D_n is the already mentioned coding of finite sets). Note that if $G(f)$ and A are r.e. the whole formula, being Σ_1 is r.e. Continuous effective f as above are called *enumeration operators.* QED

Theorem 65. *For all interpretations of variables in $\mathcal{E}$, $[\![t]\!]$ is an r.e. set, for all t.*

We remark that the graph model is not extensional (i.e. does not satisfies the $\eta - rule$). Scott introduced another *extensional* model in 1969. It was the first non-trivial model of the untyped Lambda-calculus. We conclude just sketching the basic steps of the construction of Scott's model named D_∞. The equation $D = D \longrightarrow D$ is solved not as an equality, but as an isomorphism. Let $D_0, D_1, D_2...$ be a countable sequence of cpo's with Scott's topology, defined as $D_0 = D$ and $D_{n+1} = [D_n \longrightarrow D_n]$ and let $j_n : D_{n+1} \longrightarrow D_n$ be continuous maps. Each D_n is a finer approximation to a limit than each of the preceding and each D_n will be embedded in D_{n+1} preserving its structure. The elements of this limit set are infinite sequences of functions. The application of an element $b = \langle b_0, b_1, b_2...\rangle$ to an element $a = \langle a_0, a_1, a_2...\rangle$ is defined as:

$$b \cdot a = \langle b_1(a_0), b_2(a_1), b_3(a_2)...\rangle$$

so that self-application is made possible $a \cdot a = \langle a_1(a_0), a_2(a_1), a_3(a_2)...\rangle$. The sequence $\langle D_n, j_n\rangle$ is called *projective* (or *inverse*) *system* and its limit is a cpo called *inverse limit*, also denoted D_∞, where:

$$D_\infty = \{x \in (\bigcup_n D_n)^{\mathbb{N}} | \forall n(x(n) \in D_n) \wedge j_n(x(n+1)) = x(n)\}$$

is a set of infinite sequences of functions (we can write x_n in place od $x(n)$), orderd pointwise: $x \leq y$ if an only if $\forall n(x(n) \leq y(n))$. For directed $X \subseteq D_\infty$ the supremum is defined as $\sqcup X = \lambda n. \sqcup \{x(n) | x \in X\}$. Scott characterises this embedding by means of pairs of functions (i, j) where i is injective and j is surjective.

Definition 30. *A pair of mappings (i, j) is called* projection *(or* embedding-projection*, or* retraction pair*) of D' on D if:*

1. *$i : D \longrightarrow D'$ and $j : D' \longrightarrow D$ are both continuous.*
2. *$j \circ i = Id_D$ and $i \circ j \leq Id_{D'}$.*

The function j is properly called projection *and the function i is called* embedding.

The function i (embedding) is injective, while the function j (projection) is suriective and these functions determine uniquely each other. Embedding-projection pairs are weakening of the concept of isomorphism: point 2. means that going in the direction $D \xrightarrow{i} D' \xrightarrow{j} D$ all information is completely preserved and we are back exactly where we started, while, on the contrary, going in the direction $D' \xrightarrow{j} D \xrightarrow{i} D'$ we may lose some information, i.e. $i(j(x)) \leq x$. Let us start by considering the projection (i_0, j_0) where $i_0 : D_0 \longrightarrow D_1$ and $j_0 : D_1 \longrightarrow D_0$ are defined as $i_0(x)(y) = x$ and $j(f) = f(\bot)$. Now, suppose a projection (i_n, j_n) has been defined, where $i_n : D_n \longrightarrow D_{n+1}$ and $j_n : D_{n+1} \longrightarrow D_n$. Hence it is provable that the following pair (i_{n+1}, j_{n+1}) is a projection too, where:

1. $i_{n+1}(f) = i_n \circ f \circ j_n$
2. $j_{n+1}(g) = j_n \circ g \circ i_n$

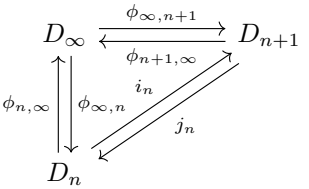
for $f \in D_{n+1}$ and $g \in D_{n+2}$. Our D_∞ is the inverse limit of the sequence $\langle D_m, j_n\rangle_{n\in\mathbb{N}}$, where $\bot = \langle \bot_0, \bot_1, \bot_2....\rangle$, where $\bot_n$ is the least member of D_n and $\sqcup X = \langle \sqcup X_0, \sqcup X_1, \sqcup X_2...\rangle$ with directed $X \subseteq D_\infty$ and $X_n = \{x(n) | x \in X\}$. In particular the projection $(\phi_{n,\infty}, \phi_{\infty,n})$ which embeds D_n into D_∞ is considered, where $\phi_{n,\infty} = \langle \phi_{n,k}(x)\rangle_{k\in\mathbb{N}}$ and $\phi_{\infty,n}(x) = x_n$ and:

$$\phi_{n,m} = \begin{cases} j_m \circ \phi_{n,m+1} & \text{if } n > m \\ i_{m-1} \circ \phi_{n,m-a} & \text{if } n < m \\ Id & \text{otherwise} \end{cases} \tag{4}$$

In other words, if $n < m$, then $\phi_{n,m} = i_{m-1} \circ i_{m-2} \circ i_{m-3} \circ ... \circ i_{n+1} \circ i_n$ and if on the contrary $n > m$, then $\phi_{n,m} = j_m \circ j_{m+1} \circ j_{m+2} \circ ... \circ j_{n-2} \circ j_{n-1}$. Hence $\phi_{n,\infty}$ has the form $y_0, y_1, y_2....$ where:

$$
\begin{aligned}
&y_0 = j_0(j_1(j_2(...(y_{n-1}(x)...)\\
&y_1 = j_1(j_2(...(y_{n-1}(x)...)\\
&\vdots\\
&y_{n-k} = j_{n-k}(...(y_{n-1}(x)...)\\
&\vdots\\
&y_{n-1} = j_{n-1}(x)\\
&y_n = x\\
&y_{n+1} = i_n(x)\\
&y_{n+2} = i_{n+1}(i_n(x))\\
&\vdots\\
&y_{n+k} = i_{n+k-1}(...(i_n(x)...)\\
&\vdots
\end{aligned}
$$

For instance, $\phi_{2,\infty}(x)$ has the form $\langle j_0(j_1(x)), j_1(x), x, i_2(x), i_3(x), i_4(x)...\rangle$, where $j_0(j_1(x)), j_1(x)$ are approximations of x and $i_2(x), i_3(x), i_4(x)...$ are copies of it. Since the range of $\phi_{n,\infty}$ is an isomorphic copy of D_n in D_∞ we can think of $\phi_{n,\infty}(x)$ just as x. Under this convention, since it is not difficult to prove that $\phi_{n,\infty}(x_n) \leq \phi_{n+1,\infty}(x_{n+1})$ and $x = \sqcup_n \phi_{n,\infty}(x(n))$, then we can look at the sequence $x_0, x_1, x_2....$ as a series of successive approximations to x and (up to isomorphisms) $D_0 \subseteq D_1 \subseteq D_2 \subseteq ... \subseteq D_\infty$. Notice that $\phi_{\infty,n+1} = i_n \circ \phi_{\infty,n}$ and $\phi_{n+1,\infty} = \phi_{\infty,n} \circ j_n$, where $\phi_{n,\infty} : D_n \longrightarrow D_\infty$ and $\phi_{\infty,n} : D_\infty \longrightarrow D_n$. Hence if we consider the category of cpo's with projective (or retraction) pairs as morphisms, we see that the following diagram commutes:



It follows that D_∞ is an upper bound for the sequence $D_0, D_1, D_2...$ Without investigating in depth the interesting structure of D_∞, let us emphasise a few facts. The first fact is that if $x \in D_n$, then $i_n(x) \in D_{n+1}$, hence this second element has in D_∞ the form:

$$\langle \phi_{n+1,0}(x), \phi_{n+1,1}(x), ..., \phi_{n+1,n+1}(x), ...\rangle$$

However $\phi_{n+1,m} = \phi_{n,m} \circ j_n$, so the above sequence actually is:

$$\langle \phi_{n,0}(x), \phi_{n,1}(x), ..., x, i_n(x), ..., \phi_{n,m}(x), ...\rangle$$

which is just x. It follow that if $x \in D_n$, then $i_n(x) = x$. With similar arguments we also obtain that if $x \in D_{n+1}$, then $j_n(x) \leq x$. The second fact to which we would like to draw attention is that if $n \leq k$, $x \in D_{n+1}$ and $y \in D_k$, then $x(y_n) = x_{k+1}(y)$. Actually this is immediate for $k = n + 1$, since $i_{n+1}(x) = i_n \circ x \circ j_n$ and therefore $i_{n+1}(x)(y) = i_n(x(j_n(y)) = i_n(x(y_n)) = x(y_n)$ by definition of projection and of the structure of D_∞. Analogously if $n \leq k$, $x \in D_{k+1}$ and $y \in D_n$, then $x_{n+1}(y) = (x(y_k))_n$. We will use these remarks in what follows. The next step consist in defining the application as follows, for $x, y \in D_\infty$:

$$x \cdot y = \sqcup_n \phi_{n,\infty}(x_{n+1}(y_n))$$

and $F(x)(y) = x \cdot y$ and its inverse G, where $G(f) = \sqcup_n G_n(f)$, where $G_n(f)(y) = (f(y))_n$. Application is continuous in both variables, i.e. $x \cdot \sqcup X = \sqcup\{x \cdot y | y \in X\}$ and $\sqcup X \cdot y = \sqcup\{x \cdot y | x \in X\}$.

Theorem 66. *The maps F and G are inverse isomorphisms.*

Proof. In these arguments we use, without explicitly mentioning it, the continuity of the functions involved:

1. $G \circ F = Id$. This follows from the fact that if $y \in D_n$, then $(x \circ y)_n = x_{n+1}(y)$ and this is a consequence of the second of the previous remarks:

$$(x \circ y)_n = (\sqcup_k (x_{k+1}(y)_k)_n = \sqcup_k (x_{k+1}(y)_k)_n = \sqcup_{k \geq n} (x_{k+1}(y)_k)_n = \sqcup_{k \geq n} x_{n+1}(y) = x_{n+1}(y)$$

   Hence $G(F(x)) = \sqcup_n x_{n+1} = x$.
2. $F \circ G = Id$. First observe that, thanks to the above remarks:

$$(G(f)) \cdot x = \sqcup_i (G(f))_i(x_i) = \sqcup_{i \geq n} (G(f))((x_i)_n) = (G(f))(x_n)$$

   Hence by definition $G(f) \cdot x = \sqcup_n (f(x_n))_n$. Still by using continuity of f we obtain:

$$G(f) \cdot x = \sqcup_n (f(x_n))_n = \sqcup_{p \geq n} (f(x_n))_p = f(x)$$

QED

Hence we have obtained a "solution" to the equation $D = [D \longrightarrow D]$, as desired. The functions F and G are used to interpret terms, as in the cases we have already analysed. Moreover, this model is extensional, in the sense that if $a \cdot c = b \cdot c$, for all c, then $a = b$. We show that $a_{n+1} = b_{n+1}$, that is, for all $x \in D_n$, $a_{n+1}(x) = b_{n+1}$: take $c = \phi_{n,\infty}(x)$, so that $\phi_{r,n}(x) = c_n$; but we have seen that $(a \cdot c)_n = a_{n+1}(x)$ and $(b \cdot c)_n = b_{n+1}(x)$ and therefore if $a \cdot c = b \cdot c$ we conclude that $a_{n+1}(x) = b_{n+1}(x)$, for all $x \in D_n$. In particular $a_0 = j_0(a_1) = j_0(b_1) = b_0$ (see Amadio and Curien (1996) 67-76, Hindley and Seldin (2008) 256-270 and Barendregt (1984) 477-486 for a more detailed argument).

3.6. Feasibility in lambda calculus: guide for further study

So far we have talked mainly of definability in untyped lambda calculus of computable functions, but we have established little with regard to lower complexity and to the main complexity classes, only that the functions that can be defined in the simply typed lambda calculus are the extended polynomials. What about the polynomial time computable functions? In line with the purpose of these lecture notes we indicate now some research developments in this area more specifically oriented in the direction of *computational complexity*. This research has developed mainly in two directions. The first one comes from the area of (second order) *Linear Logic*: *Light Linear Logic*, *Light Affine Logic*, *Soft Linear Logic* etc. (see among others Girard (1998), Asperti (1998), Baillot and Mogbil (2004), Lafont (2004), Asperti and Roversi (2002)) propose different characterisations of functions computable in polynomial time. Such functions are representable by proofs within the proofs-as-programs correspondence and strategies are found which normalize them in polynomial time. These logics are *intrinsically polytime* in that any proof expressed in the so-called *proof nets*, or term of a lambda-calculus, can be converted into a normal (cut-free) one in polynomial time in the dimension of some parameter relative to the proof or term. Gentzen's sequent calculus contains, in addition to the logical rules, the so-called structural rules, in particular weakening and contraction (see sections 7.1 and 7.2 for an introduction to this calculus, of cut-elimination and the role of structural rules). Both, although perfectly plausible from a classical point of view, clash with the principles of *Linear Logic*, which is a resource-sensitive logic, where two resources do not count as one. Furthermore, the unrestricted use of the contraction rule actually causes an exponential explosion of complexity of the process of computation, i.e. of cut-elimination. In *Linear Logic*, contracting or weakening a formula is not permitted unless it is authorised by means of two operators ! (of course) and ? (why not). The !-modality corresponds to contraction and weakening on the left hand side, and the ?-modality corresponds to contraction and weakening on the right hand side. However, the cut-elimination complexity is still superexponential, but this

idea of controlling contraction through modal operators can be further developed, resulting in the *Light Linear Logic* that, unlike another variant, called *Bounded Linear Logic*, does not use explicit parameters such as $! \leq n$, which makes the system not purely logical. Rather, a new modality § ("neutral") is in this case added which refers to the unary cases of ! and ?. The associated calculus no longer allows certain laws of *Linear Logic* to be derived, such as $!\alpha \Longrightarrow \alpha$ ("dereliction ") or $!\alpha \Longrightarrow !!\alpha$ ("digging"), which are, however, compensated for by other laws as a weak version of the former $!\alpha \Longrightarrow \S\alpha$ or $\S\alpha \Longrightarrow ?\alpha$. This ends up being sufficient to prevent the uncontrolled explosion of computational complexity. Every polynomial time function is representable by an proof of this calculus, and every proof is normalizable in polynomial time.

Light Affine Logic is a variant of the *Light Linear Logic* that was introduced in Asperti (1998) by admitting the full weakening rule. This move does not undermine progress in complexity, but greatly simplifies the calculus. In Baillot (2004) it is shown that *Intuitionistic Light Affine Logic* provides a typing for lambda-calculus which guarantees that a well-typed program is executable in polynomial time. Echoing an observation by Girard that normalisation in polynomial time is largely independent of types, Terui (2007) introduces instead an *untyped light affine lambda calculus*, which embodies the essentials of this logic, with the remarkable strong normalization property that each term is normalizable in polynomial time regardless the strategy applied. Logic is then re-introduced as a Curry-style type assignment system for this calculus.

We will not deal with this here because of the large number of prerequisites that we would have to set in advance to address this issue in detail. Rather, we conclude by broadly stating what are the insights behind the second direction of the research, that proposes restrictions such that the functions definable thus obtained are those of PTIME and it is more accessible with the tools we have introduced so far: this is for instance the approach that originated in Bellantoni and Cook (1992). In Leivant (1990) and Leivant and Marion (1993) the underling idea is that data are used computationally at different levels of abstraction and this is called *data ramification*, or *data tiering*. The proposal consists of an *implicit* characterisation of complexity classes, i.e. by language restrictions rather than by explicit resource bounds. The first paper proposes a particular formalism in which the arguments of a function on binary strings are divided into two blocks separated by a semi-colon $f(x_0, ..., x_n; y_0, ..., y_k)$. Those occurring to the left of the semi-colon are called *normal*, and variables to the right are called *safe*. Performing a recursion, the previous step value must be placed in *safe* position:

1. $f(0, x; y) = g(x; y)$
2. $f(z * i, x; y) = h_i(z, x; y, f(z, x; y))$ $(i = 0, 1)$

The composition scheme ensures that the recursive value will stay in a safe position and will not be copied into a normal position:

$$f(x; y) = h(g_0(x;), ..., g_m(x;); h_0(x; y), ..., h_k(x; y))$$

This is an example of use of tiering techniques. The polynomial time functions will be exactly those functions which have all normal inputs, i.e. no safe inputs. As for the lambda calculus, the second paper introduce a notion of "tiering" in the framework of a typed lambda-calculi with a base type o, by extending the set of terms of the typed calculus we have considered in the strong normalization theorem with new operators $\mathsf{0}$, $\mathsf{1}$ and predecessor $\mathsf{p}(cx) = x$ (for $c = 0, 1$), all of type $\mathsf{o} \to \mathsf{o}$ and discriminator function d of type $\mathsf{o} \times \mathsf{o} \times \mathsf{o} \times \mathsf{o} \to \mathsf{o}$ and an operator ϵ (the empty string) of type o, and show that the polynomial-time functions are precisely the functions representable with *abstract* input (Church-like numerals) and *concrete* output (represented by binary strings). To explain this idea, let us consider the set of binary strings $\{\mathsf{0}, \mathsf{1}\}^*$ (the free algebra generated by $\mathsf{0}, \mathsf{1}$ and ϵ). Observe that a word e.g. $w = (\mathsf{0}(\mathsf{1}(\mathsf{1}\epsilon)))$ can be also represented abstractly *à la Church* as $w^* = \lambda x_0 \lambda x_1 \lambda x_\epsilon . x_0(x_1(x_1 x_\epsilon)))$. The reader can check that this is a term of type $\omega[\tau] = (\tau \to \tau) \to (\tau \to \tau) \to \tau \to \tau$.

When w^* has type $\omega[\mathsf{o}^q]$, where $\mathsf{o}^q = \overbrace{\mathsf{o} \times ... \times \mathsf{o}}^{q-times}$, we will write $w^{*,q}$. A function $f : \{0, 1\}^k \to \{0, 1\}^r$ is *explicitly* defined in this calculus by a term E of type $\mathsf{o}^k \to \mathsf{o}^r$, if $\mathsf{E}w_0...w_{k-1} =$

$f(w_0...w_{r-1})$. It is instead *2-tiers* definable by a term E of type $\omega[\mathsf{o}_0^q] \to ... \to \omega[\mathsf{o}_{k-1}^q] \to \mathsf{o}$, for some $q_0, ..., q_{k-1}$, if $\mathsf{E}w_0^{*,q_0}...w_{k-1}^{*,q_{k-1}} = f(w_0...w_{r-1})$. Well, it is provable that a function over $\{\mathsf{0},\mathsf{1}\}^*$ is polynomial time computable if and only if it is *2-tier* definable in this calculus.