Under consideration for publication in Math. Struct. in Comp. Science

Redexes are stable in the λ-calculus

JEAN-JACQUES LÉVY

INRIA,
Campus de l'École polytechnique,
91120 Palaiseau, France.

Received 14 May 2013

In the theory of sequential programming languages, Berry's stability property is an important step claiming that values have unique origins in their calculations. It has been shown that Bohm trees are stable in the λ-calculus, meaning that there is a unique minimum prefix of any lambda term which computes its Bohm tree. Moreover this property is also true for finite prefixes of Bohm trees. The proof relies on Curry's standardization theorem as initially pointed out by Plotkin in his famous article *LCF considered as a programming language*. In this paper we will show that the stability property also holds for redexes. Namely for any redex family, there is a unique minimum calculation and a unique redex which computes this family. This property was already known in the study of optimal reductions, but we stress here on stability and want to show that stability is inside the basic objects of calculations. The proof will be based on nice commutations between residuals and creations of new redexes. Our tool for proving this property will be the labeled lambda calculus used in the study of optimal reductions.

1. Introduction

The theory of sequential programming languages relies on complex structures such as Scott's domains (Cadiou, 1972; Vuillemin, 1973; Plotkin, 1977; Berry, 1978), Kahn-Plotkin concrete domains and CDS (Berry and Curien, 1982; Berry and Curien, 1985), game semantics (Abramsky et al., 2000), etc. However, Berry's stability property is a simple property of monotonic functions on lattices. The monotonic function f is stable if whenever x and y are compatible, we have $f(x \sqcap y) = f(x) \sqcap f(y)$ where $x \sqcap y$ is the greatest lower bound of x and y. By x and y compatible, we meant that they have a common majorant. This implies on well-founded lattices that there is a unique minimal minorant of any x producing the same value as $f(x)$. Therefore stability is a first approximation of sequentiality, since one cannot compute a given value by two distinct minimal ways. The prototype of a non stable function is the well-known parallel-or *por* such that $por(\perp, \perp) = \perp$, $por(tt, \perp) = tt$, $por(\perp, tt) = tt$.

Stability does exist in the λ-calculus. Plotkin(Plotkin, 1977) proved that PCF (a λ-calculus with conditional and arithmetic) is stable, thus *por* is not definable

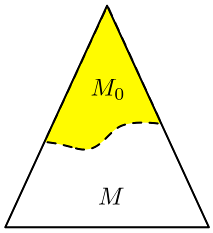


Fig. 1. "has a (head) normal form" is a stable predicate.

in PCF. In the pure λ-calculus, one can also prove that there is no context $C[\ ,\ ]$ such that $C[\Omega, \Omega]$ has no normal form, but $C[\Omega, I]$ and $C[I, \Omega]$ have normal forms, where $\Delta = \lambda x.xx$, $\Omega = \Delta\Delta$ and $I = \lambda x.x$. Indeed, with Curry's standardization theorem, we know that a normal form is always reached by the leftmost-outermost reduction (normal reduction). Therefore either the normal reduction of $C[M, N]$ ignores M and N and $C[\Omega, \Omega]$ has a normal form; or the normal reduction starts by M, and then $C[\Omega, I]$ cannot have a normal form; or it starts by N and then $C[I, \Omega]$ has no normal form. The deterministic feature of the normal reduction enforces the stability property of the predicate "has a normal form".

This property is also true for head normal forms and for their finite and infinite expansions, namely Bohm trees (Barendregt, 1984). For any λ-term M with a head normal form, there is a unique minimum prefix M_0 which has a head normal form, see figure 1. To be more precise, we define prefixes in a λ-calculus augmented by a constant $\_$ and say that M is a prefix of N if N matches M except in occurrences of this new constant. For instance $M = (\lambda f.fI(f\Omega))(\lambda x.I)$ has I as normal form. Its minimum prefix is $M_0 = (\lambda f.f\_(f\_))(\lambda x.I)$ which also reduces to I. Similarly, for Bohm trees, if a is a finite approximation of the Bohm tree of M, there is a unique minimum prefix M_0 of M which also admits a as finite approximation of its Bohm tree. Proofs of this property can be found in (Berry, 1978; Lévy, 1978).

The stability property relies on the standardization theorem which entails the correctness of the normal reduction strategy. But a more precise cause could exist inside redexes. This is the goal of this contribution. In section 2, we refresh notions of labels, residuals and creation of redexes in the λ-calculus. We also introduce a revised labeled λ-calculus. In sections 3 and 4 we review permutation equivalence on reductions, historical redexes and redex families. In section 5 we state our aimed stability properties on redexes. We conclude in section 6.

2. The labeled λ-calculus, residuals and created redexes

We follow the notations used in (Lévy, 2009). The set Λ of λ-terms, ranged over by M, N, ... contains variables x, applications (MN) and abstractions $(\lambda x.M)$. Beta-conversion is $(\lambda x.M)N \to M\{x := N\}$. Several steps (maybe none) of beta-conversion are written $\twoheadrightarrow$ and called reductions. We give names ρ, σ, ... to reductions and we write $\rho : M \twoheadrightarrow N$ to specify the initial and final terms of reduction

ρ. The reduction graph $\mathcal{R}(M)$ of any lambda term M is the graph of reducts of M connected by reduction steps. Paths in this graph are reductions between reducts of M. The reductions ρ and σ are coinitial if they start from a same term M. They are cofinal if they end on a same N. Let $\rho \in \mathcal{R}(M)$ be the following reduction from M to N

$$\rho : M = M_0 \stackrel{R_1}{\to} M_1 \stackrel{R_2}{\to} M_2 \cdots \stackrel{R_n}{\to} M_n = N$$

contracting redex R_i in M_{i-1} at each step ($1 \leq i \leq n$). We may also write simply $\rho = R_1 R_2 \ldots R_n$ when M and N are clear from the context. When $n = 0$, then ρ is the empty reduction o_M or simply o. Redexes may be tracked over reductions. Let R be a redex in M. We write R/ρ for the set of residuals of R in M by reduction $\rho \in \mathcal{R}(M)$. For instance, if we underline R and its residuals, we have

$$\begin{aligned} \rho &: \Delta(\underline{Ix}) \to (\underline{Ix})\ (\underline{Ix}) \\ \sigma &: \underline{\Delta(Ix)} \to \underline{\Delta x} \\ \tau &: \underline{\Delta(Ix)} \to (Ix)(Ix) \end{aligned}$$

We refer to Curry&Feys (Curry and Feys, 1958), Barendregt (Barendregt, 1984), or (Lévy, 1978) for a precise definition of residuals by use of the annoying notion of occurrences. Notice that residuals could differ along two coinitial and cofinal reductions. Take $I(Ix) \to Ix$. Finally ρ creates R if there is no R' such that $R \in R'/\rho$.

We now define the labeled λ-calculus, which was first introduced in the Rome Symposium on λ-calculus and Computer Science Theory, 1975, organized by Corrado Böhm (Lévy, 1975). The aim of this calculus is to give names to redexes and subterms in order to track their origin. The set Λ_ℓ of labeled terms contains the usual λ-terms but with every subterm equipped with a label. Beta-conversion modifies labels on borders of contractums of redexes.

$$\begin{array}{llr} \alpha, \beta, \ldots \quad ::= \quad a, b, c, \ldots \mid \lceil \alpha \rceil \mid \lfloor \alpha \rfloor \mid \alpha\beta & & \text{labels} \\ M, N, \ldots \quad ::= \quad x^\alpha \mid (MN)^\alpha \mid (\lambda x.M)^\alpha & & \text{labeled terms} \\ ((\lambda x.M)^\alpha N)^\beta \to ((M\{x := N^{\lfloor \alpha \rfloor}\})^{\lceil \alpha \rceil})^\beta & & \text{labeled beta-conversion} \\ (U^\alpha)^\beta = U^{\alpha\beta} \qquad x^\alpha\{x := N\} = N^\alpha & & U \text{ without external label} \end{array}$$

Atomic labels are letters a, b, c, … overlined label $\lceil \alpha \rceil$ and underlined label $\lfloor \alpha \rfloor$. Label $\alpha\beta$ is composite. Substitution is defined as usual in the λ-calculus. The only difference is when substituting a variable in which case the label is changed. Notice also that we slightly modified the calculus of (Lévy, 1975) by taking mirror images of previous labels. This new presentation seems more intuitive. Finally the name of a redex is the label of its function part.

$$\mathsf{name}(((\lambda x.M)^\alpha N)^\beta) = \alpha$$

Proposition 2.1. $S \in R/\rho$ implies $\mathsf{name}(R) = \mathsf{name}(S)$.

Proposition 2.2. If R creates S, then $\mathsf{name}(R)$ is strictly contained in $\mathsf{name}(S)$.

Theorem 2.3. The labeled λ-calculus is confluent.

The proofs follow from the definitions and standard methods, see (Lévy, 1975). The only interesting proposition is for the creation of redexes. There are only three cases. First when a function is passed to the left of an application as in:

$$(\lambda x.\cdots(x^\beta\, N)\cdots)^\alpha\,(\lambda y.M)^\gamma \to \cdots((\lambda y.M)^{\gamma\lfloor\alpha\rfloor\beta} N')\cdots$$

or when the curryfied function takes its first argument:

$$((\lambda x.(\lambda y.M)^\gamma)^\alpha N)^\beta P \to (\lambda y.M')^{\gamma\lceil\alpha\rceil\beta} P$$

or when a function is applied to the identity at the left of an application:

$$((\lambda x.x^\gamma)^\alpha(\lambda y.M)^\delta)^\beta N \to (\lambda y.M)^{\delta\lfloor\alpha\rfloor\gamma\lceil\alpha\rceil\beta} N$$

In every case, the name α of the contracted redex becomes atomic in the name of the new redex and is therefore strictly contained inside. The confluence property of the labeled λ-calculus is also true when one restricts beta-conversion to a given set of redex names, since residuals keep names. Moreover strong normalization holds when this set of names is finite, see (Lévy, 1978).

Let $\mathsf{INIT}(M)$ be true when all labels in M are distinct letters. Then names of created redexes cannot be simple letters, which gives the following easy corollary.

Corollary 2.4. If $\mathsf{INIT}(M)$ and $\rho : M \twoheadrightarrow N$, then $S \in R/\rho$ iff $\mathsf{name}(R) = \mathsf{name}(S)$.

3. Parallel reductions, permutations of reductions

Parallel beta-conversion $\xrightarrow{\mathcal{F}}$ of a set $\mathcal{F}$ of redexes is defined as a finite development of $\mathcal{F}$ where finite developments follow this theorem.

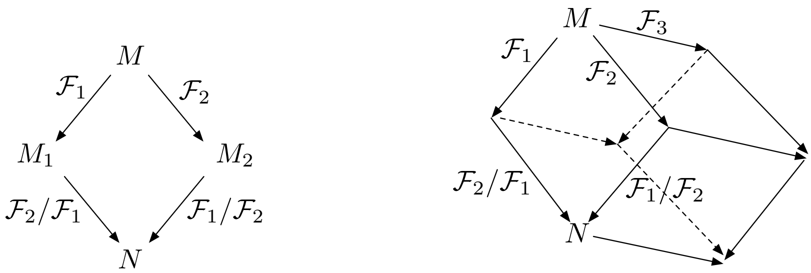
Theorem 3.1 (Finite developments). Let $\mathcal{F}$ be a set of redexes in M, consider relative reductions which contract only residuals of redexes in $\mathcal{F}$. Then:

(i) there is no infinite reduction relative to $\mathcal{F}$;
(ii) the developments of $\mathcal{F}$ (maximal relative reductions) end all at the same term;
(iii) the residuals of any redex R in M are the same by all developments of $\mathcal{F}$.

The theorem is due to Curry. Its proof is quite simple by using confluency and strong normalization of the labeled λ-calculus as in (Lévy, 1978). Parallel moves and the cube lemma are two corollaries.

Corollary 3.2 (Parallel moves). Let $\mathcal{F}_1$ and $\mathcal{F}_2$ be two sets of redexes in M. If $M_1 \xleftarrow{\mathcal{F}_1} M \xrightarrow{\mathcal{F}_2} M_2$, then there is a term N such that $M_1 \xrightarrow{\mathcal{G}_2} N \xleftarrow{\mathcal{G}_1} M_2$ with $\mathcal{G}_1 = \mathcal{F}_1/\mathcal{F}_2$ and $\mathcal{G}_2 = \mathcal{F}_2/\mathcal{F}_1$.

Let $\mathcal{F}_1 \sqcup \mathcal{F}_2 = \mathcal{F}_1(\mathcal{F}_2/\mathcal{F}_1)$. An alternative statement of this lemma would be that $\mathcal{F}_1 \sqcup \mathcal{F}_2$ and $\mathcal{F}_2 \sqcup \mathcal{F}_1$ are cofinal. The proof is obvious by considering two developments of $\mathcal{F}_1 \cup \mathcal{F}_2$, one starting by contracting $\mathcal{F}_1$ and then $\mathcal{F}_2/\mathcal{F}_1$, the second contracting $\mathcal{F}_2$ and then $\mathcal{F}_1/\mathcal{F}_2$.



(a) (b)

Fig. 2. (a) Parallel moves; (b) Cube lemma.

Corollary 3.3 (Cube lemma). Let $\mathcal{F}_1$, $\mathcal{F}_2$ and $\mathcal{F}_3$ be three sets of redexes in M. Then $\mathcal{F}_3/(\mathcal{F}_1 \sqcup \mathcal{F}_2) = \mathcal{F}_3/(\mathcal{F}_2 \sqcup \mathcal{F}_1)$.

Two reductions are equivalent by permutations when they differ by several uses of the lemma of parallel moves and erasures of empty steps.

Definition 3.4. The *permutation equivalence* $\sim$ on reductions in $\mathcal{R}(M)$ is inductively defined by:

(i) $\mathcal{F}_1 \sqcup \mathcal{F}_2 \sim \mathcal{F}_2 \sqcup \mathcal{F}_1$
(ii) $\emptyset \sim o$ and $o \sim \emptyset$
(iii) $\rho \sim \sigma$ implies $\tau\rho \sim \tau\sigma$
(iv) $\rho \sim \sigma$ implies $\rho\tau \sim \sigma\tau$
(v) $\rho \sim \sigma \sim \tau$ implies $\rho \sim \tau$.

Alternative definitions of this equivalence can be found in (van Oostrom and de Vrijer, 2002; van Oostrom and de Vrijer, 2003). This relation is defined on parallel reductions, but it also relates regular reductions since their elementary steps can be considered as contractions of singleton sets of redexes. Residuals of reductions can also be defined inductively as follows (see figure 3).

Definition 3.5. Let ρ and σ be two reductions starting at M. The *residual ρ/σ of reduction ρ* by reduction σ is inductively defined by:

(i) if $\rho = \mathcal{F}$ and $\sigma = \mathcal{G}$, then $\rho/\sigma = \mathcal{F}/\mathcal{G}$
(ii) $\rho/(\sigma_1\sigma_2) = (\rho/\sigma_1)/\sigma_2$
(iii) $(\rho_1\rho_2)/\sigma = (\rho_1/\sigma)(\rho_2/(\sigma/\rho_1)$

When ρ and σ are two coinitial reductions, we pose $\rho \sqcup \sigma = \rho(\sigma/\rho)$, and write $\emptyset^k$ for k steps of empty-set contractions ($k \geq 0$). The following properties hold for the permutation equivalence.

Proposition 3.6. Let ρ and σ be two coinitial reductions ($\rho, \sigma \in \mathcal{R}(M)$) :

(i) $\rho \sqcup \sigma \sim \sigma \sqcup \rho$
(ii) $\rho \sim \sigma$ iff $\forall \tau \in \mathcal{R}(M)\;\; \tau/\rho = \tau/\sigma$
(iii) $\rho \sim \sigma$ iff $\rho/\sigma = \emptyset^k$ and $\sigma/\rho = \emptyset^\ell$

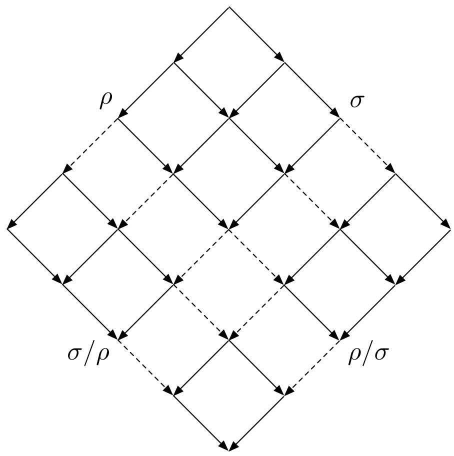


Fig. 3. ρ/σ is residual of reduction ρ by reduction σ.

(iv) $\rho\sigma \sim \rho\tau$ iff $\sigma \sim \tau$

Standard reductions work from left to right. A redex is to the left of another redex if the first is to the left or contains the second redex. The following reduction

$$\rho : M_0 \overset{R_1}{\to} M_1 \overset{R_2}{\to} M_2 \cdots \overset{R_n}{\to} M_n$$

is standard iff for all i,j such that $0 < i < j \leq n$, redex R_j is not residual of R'_j to the left of R_i in M_{i-1} along ρ.

Proposition 3.7. For any ρ, there is a unique σ standard such that $\rho \sim \sigma$.

The proof follows from the usual proof of Curry's standardization theorem. The uniqueness is easy to prove, see (Lévy, 1978). Therefore an equivalence class of reductions is characterized by its unique standard reduction. Finally let ρ be a prefix of coinitial reduction σ iff $\rho\tau \sim \sigma$ for some τ. We then write $\rho \leq \sigma$. This prefix ordering gives the lattice of reductions for a given $\mathcal{R}(M)$ reduction graph. The interested reader is refered to (Lévy, 1978; Barendregt, 1984; Berry and Lévy, 1979; Huet and Lévy, 1991).

4. Histories and redex families

An *historical redex* (*hredex* in short) is a pair $\langle \rho, R \rangle$ where $\rho : M \twoheadrightarrow N$ and R is a redex in N. Redex R in M is hredex $\langle o, R \rangle$ where o is the empty reduction. Intuitively, history of redexes has to be consistent with permutations of reduction steps. The contractions of two independent redexes are insensitive in the creation history of a redex. For instance we have $(\lambda x.xy)(II) \twoheadrightarrow Iy$ by contractions of two distinct redexes. Redex Iy is created by these two redexes independently of the order in which they are contracted. But when we have $\rho : \Omega \to \Omega$ and $\sigma : \Omega \to \Omega \to \Omega$, then $o \not\sim \rho \not\sim \sigma$. Therefore $\langle o, \Omega \rangle$, $\langle \rho, \Omega \rangle$ and $\langle \sigma, \Omega \rangle$ are distinct hredexes. A third

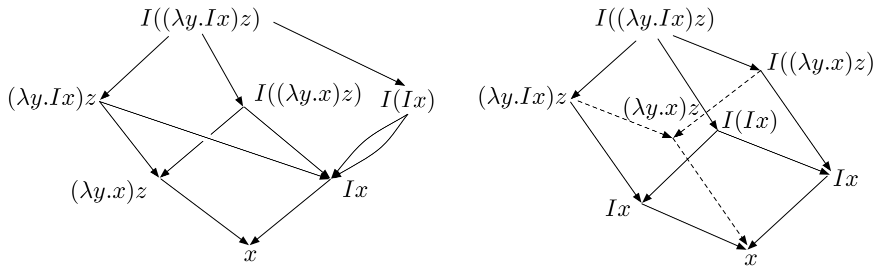


Fig. 4. Hredexes and permutations of reductions.

example considers $M = I((\lambda y.Ix)z)$ with its three redexes $R = I((\lambda y.Ix)z)$, $S = Ix$ and $K = (\lambda y.Ix)z$. The reduction graph has no lattice structure, but the structure induced by the prefix ordering is indeed a lattice. Let $\rho : I((\lambda y.Ix)z) \to (\lambda y.Ix)z \to Ix$ and $\sigma : I((\lambda y.Ix)z) \to I((\lambda y.x)z) \to Ix$, the hredex $\langle \rho, Ix \rangle$ is a residual of $\langle o, S \rangle$, but $\langle \sigma, Ix \rangle$ is a residual of $\langle o, R \rangle$. This disambiguates the residual relations induced on terms, see figure 4.

We now define formally the residual relation on hredexes by taking care of the histories of hredexes.

Definition 4.1. We say that hredex $\langle \sigma, S \rangle$ is a residual of hredex $\langle \rho, R \rangle$ when there is τ such that $\rho\tau \sim \sigma$ and $S \in R/\tau$. We then write $\langle \rho, R \rangle \leqslant \langle \sigma, S \rangle$.

Proposition 4.2. Let ρ and σ be coinitial reductions:

(i) $\langle \rho, R \rangle \leqslant \langle \sigma, S \rangle$ iff $\rho \leq \sigma$ and $S \in R/(\sigma/\rho)$.

(ii) $\rho \sim \sigma$ and $R = S$ iff $\langle \rho, R \rangle \leqslant \langle \sigma, S \rangle \leqslant \langle \rho, R \rangle$

(iii) $\langle \rho, R \rangle \leqslant \langle \sigma, S \rangle \leqslant \langle \tau, T \rangle$ implies $\langle \rho, R \rangle \leqslant \langle \tau, T \rangle$

(iv) $\langle \rho, R \rangle \leqslant \langle \tau, T \rangle$ and $\langle \sigma, S \rangle \leqslant \langle \tau, T \rangle$ implies there is T' (unique) such that $\langle \rho, R \rangle \leqslant \langle \rho \sqcup \sigma, T' \rangle$, $\langle \sigma, S \rangle \leqslant \langle \rho \sqcup \sigma, T' \rangle$ and $\langle \rho \sqcup \sigma, T' \rangle \leqslant \langle \tau, T \rangle$.

These properties are easy consequences of the previous definition. Residuals of hredexes are consistent with permutations of reductions. They form a nice pre-order with equivalence by permutations and equality of redex occurrences as associated equivalence. It also has a conditional upper semi-lattice structure and uniqueness of the least upper bound is due to the functionality of the inverse of the residual relation, since a redex is at most residual of a single redex for any given reduction.

The last step of our construction is to consider created redexes and the family relations between them. Our favourite example is the one of figure 5. Two redexes are initially in M. A third one Ix is created in the reduction graph. We observe that we only can connect its occurrences by zigzagging with residuals of hredexes. For instance Ix in $(Ix)(IIx)$ is connected to Ix in $(IIx)(Ix)$ by a zigzag of residuals through $\Delta(Ix)$. We therefore define families of hredexes as the symmetric and transitive closures of the residual relation on hredexes.

Definition 4.3. Hredexes $\langle \rho, R \rangle$ and $\langle \sigma, S \rangle$ are family-related, written $\langle \rho, R \rangle \approx$

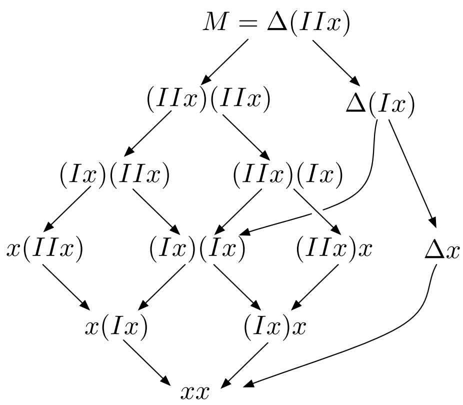


Fig. 5. An example of redex families.

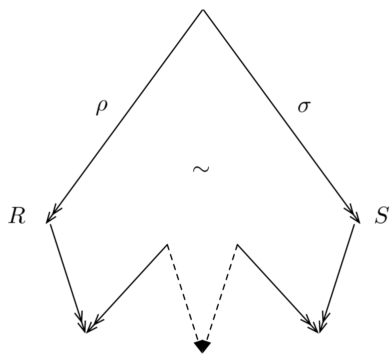


Fig. 6. $\langle \rho, R\rangle \approx \langle \sigma, S\rangle$: the hredexes $\langle \rho, R\rangle$ and $\langle \sigma, S\rangle$ are family-related.

$\langle \sigma, S\rangle$, iff $\langle \rho, R\rangle \leqslant \langle \sigma, S\rangle$ or $\langle \sigma, S\rangle \leqslant \langle \rho, R\rangle$ or $\langle \rho, R\rangle \approx \langle \tau, T\rangle \approx \langle \sigma, S\rangle$ for some $\langle \tau, T\rangle$. (see figure 6)

5. Stability of redexes

We now study the origins of hredexes and show that they have a unique origin in each redex family, ensuring a stability property for redexes as stated in the introduction of this article. First consider redexes with no history. Let R and S be redexes in M and $\rho : M \twoheadrightarrow N$ where $\rho = R \sqcup S$. We also have $\sigma : M \twoheadrightarrow N$ with $\sigma = S \sqcup R$. Let T be a redex in N. There are several cases: either T already exists in M, either T is created by both R and S, either T is created by one of them. But, in the last case, the origin is unique, T is exclusively created by R or exclusively created by S. To be more precise, we have the two following properties.

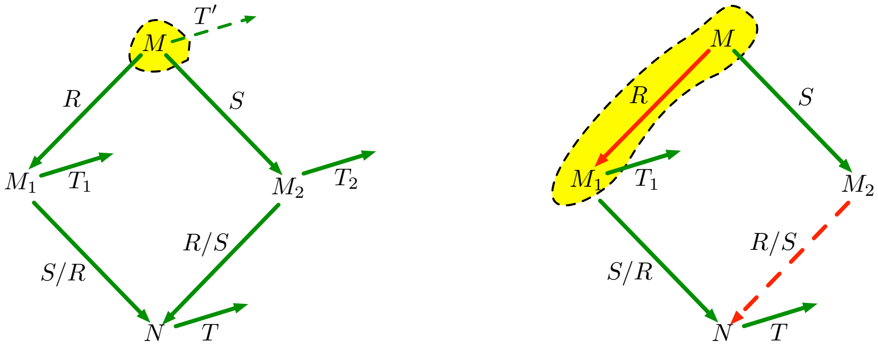


Fig. 7. Stability of redexes

Proposition 5.1. Let $R \neq S$. Then $T \in T_1/(S/R)$ and $T \in T_2/(R/S)$ implies $T \in T'/(R \sqcup S)$ for some T'. (see figure 7)

Corollary 5.2. Let $R \neq S$. If $T \in T_1/(S/R)$ and R creates T_1, then there exists R' in R/S such that R' creates T.

This last property can be seen as a nice commutation between creation and residuals. The proof of the proposition again uses the labeled λ-calculus. Assume that $\mathsf{INIT}(M)$ is true. Let a and b be names of R and S ($a \neq b$ since $R \neq S$). By proposition 1, redexes T, T_1, T_2 have the same name α. If T_1 is residual of T' in M. Then $T \in T'/(R \sqcup S)$. If T_1 is created by R, then a is strictly contained in α by proposition 2. As R is the only contracted redex in M, the name b of S cannot be underlined or overlined in α. So b is not stricly inside α. This makes impossible the creation of T_2 by S. Therefore $T_2 \in T'/S$ for some redex T' in M. Thus $T \in T'/(S \sqcup R) = T'/(R \sqcup S)$. The corollary is a logical consequence of the proposition. The only difficulty is to prove that only one redex in R/S creates T. Since R/S is a set of disjoint redexes, only one of them can create a new redex. (Only nested redexes can cooperate in the creation of new redexes.) Finally the stability property is also true for parallel steps. Again if two sets of redexes are disjoint, then there is the stability property. The proof is similar and uses the labeled λ-calculus.

Proposition 5.3. Let $\mathcal{F} \cap \mathcal{G} = \emptyset$. Then $T \in T_1/(\mathcal{G}/\mathcal{F})$ and $T \in T_2/(\mathcal{F}/\mathcal{G})$ implies $T \in T'/(\mathcal{F} \sqcup \mathcal{G})$ for some T'.

Corollary 5.4. If $\mathcal{F} \cap \mathcal{G} = \emptyset$, $T \in T_1/(\mathcal{G}/\mathcal{F})$ and $\mathcal{F}$ creates T_1, then $\mathcal{F}/\mathcal{G}$ creates T.

This is corollary 5.2 for the case of redex families, although the proofs are more difficult, see (Asperti and Laneve, 1995; Lévy, 1978). There is still a lack of an easy proof for it. The usual one goes through the labeled λ-calculus and an extraction process for creation of hredexes. The statement of the property is as follows.

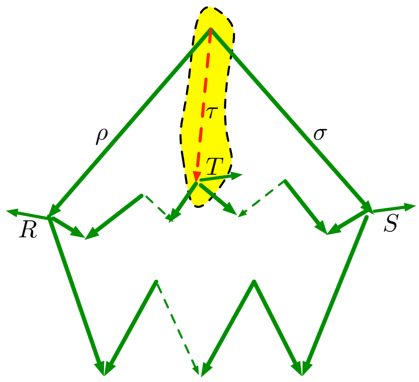


Fig. 8. Stability of redex families

Theorem 5.5. In each redex family class, there is a unique hredex $\langle \tau, T \rangle$ with τ standard reduction of minimum length.

In the theory of optimal reductions, we called this hredex the canonical representative of its family. Intuitively every step in τ is necessary to create T. Philosophically, this representative represents the unique origin τ of its family. There is no alternative way to create it. Therefore the stability property is also valid inside redex families.

In the rest of this section we sketch the proof of this theorem.

First the canonical representative is defined by eliminating every redex S useless in the creation of any hredex $\langle \rho_1 S \rho_2, R \rangle$. This happens when there is $\langle \rho_2', R' \rangle$ such that $\langle \rho_1 S \rho_2, R \rangle \leqslant \langle \rho_1 (S \sqcup \rho_2'), R'' \rangle \geqslant \langle \rho_1 \rho_2', R' \rangle$ and S not used in ρ_2'. It only makes sense when $\rho_1 S \rho_2$ is a standard reduction. Therefore following (Asperti and Laneve, 1995), we define an extraction relation when $\rho_1 S \rho_2 R$ (including last step R) is standard.

Definition 5.6. The extraction relation $\rhd$ is defined by one of the following cases:

(i) $S \sqcup \rho \rhd \rho$ if ρ is fully in a subterm disjoint from redex $S = (\lambda x.A)B$.

(ii) $S \sqcup \rho \rhd \rho$ if ρ is in the function body A of S.

(iii) $S\rho \rhd \rho'$ if ρ is in the i-th instance of the argument B of S and $\rho \| S = \rho'/S$ where $\rho \parallel S$ is reduction ρ done simultaneously in all instances of B.

(iv) $\sigma\rho \rhd \sigma\rho'$ if $\rho \rhd \rho'$

Definition 5.7. Hredex $\langle \rho, R \rangle$ is *canonical* if ρR is standard and normal form of $\rhd$.

One can show that the extraction relation is confluent (although not necessary in the overall proof). Clearly extraction keeps hredexes in the same family, namely $\langle \sigma, S \rangle \approx \langle \rho, R \rangle$ when $\sigma S \rhd \rho R$. Notice also that when σ is standard and $\sigma \rhd \rho$, then ρ is also a standard reduction with strictly smaller length. It remains to show that the canonical hredex is unique in each redex family. This is the difficult part of the proof and it relies on the labeled λ-calculus. We begin with a remark about labeled standard reductions.

Proposition 5.8. Let $\rho : M \twoheadrightarrow N$. If $R = ((\lambda x.A)^\alpha B)^\beta$ is a redex in N and ρR is standard, then $(\lambda x.A)^a$ (where $\alpha = a\alpha'$) and any subterm Q in N to the right of $(\lambda x.A)^a$ is of the form $Q = P\{x_1 := P_1^{\lfloor\gamma_1\rfloor}\} \cdots \{x_n := P_n^{\lfloor\gamma_n\rfloor}\}$ where P is a subterm of M.

This proposition is immediate along standard reductions since subterms to the right of (or equal to) the function part of the contracted redex come from subterms already to the right of the previously contracted redex. At most their free variables can be instantiated. As corollary, when $\mathsf{INIT}(M)$, at any step of every standard reduction starting from M, in any abstraction to the right of (or equal to) the function part of the contracted redex, the paths from binders to occurrences of bound variables are exactly the paths in M and therefore labeled with atomic letters. Another corollary is that there are no overlined labels at right of $(\lambda x.A)^a$.

Proposition 5.9. Let $\mathsf{INIT}(M)$ and $\rho, \sigma \in \mathcal{R}(M)$. If $\langle \rho, R\rangle$ and $\langle \sigma, S\rangle$ are canonical hredexes, then $\mathsf{name}(R) = \mathsf{name}(S)$ implies $\langle \rho, R\rangle = \langle \sigma, S\rangle$.

In (Asperti and Laneve, 1995), this uniqueness proof uses paths in M constructed from the structure of $\mathsf{name}(R)$. In (Lévy, 1978), subcontexts (i.e. prefixes of labeled subterms) are used. A full theory of subcontext families is also developed as for redexes. Although illuminating, this theory is not fully needed. Here we only use proposition 5.8 and generators of redexes.

Prefixes of terms were described in section 1. A term M is a prefix of N, written $M \preceq N$ when N matches M except for some or several $\_$ and let $M\,|\,u$ be the subterm of M at occurrence u (occurrences are sequences of 1 and 2 indicating the path from the root of term M to its subterm $M|u$). Let ρ be a labeled reduction starting from M, the generator $\mathsf{gen}(\rho, R)$ of hredex $\langle \rho, R\rangle$ is the smallest prefix P of the smallest subterm $M\,|\,u$ (u maximum) such that $P \twoheadrightarrow Q$ and Q contains R, in fact its top redex prefix $((\lambda x.\_)^\alpha \_)$ which is sufficient to characterize R. A step-by-step definition of this generator can be effectively given.

Definition 5.10. Let $M \xrightarrow{R} N$ be a reduction step contracting $R = ((\lambda x.A)^\alpha B)^b = M|u$, redex at occurrence u in M. Let (v, Q) be subcontext Q at occurrence v in N ($Q \preceq N|v$). The generator $\mathsf{gen}(R, (v, Q))$ of (v, Q) is the following subcontext in M:

(i) when (v, Q) does not overlap boundaries of the contractum of R, the definition follows the definition of residuals of redexes. Thus $\mathsf{gen}(R, (v, Q)) = (v', Q)$ where $(v, Q) \in (v', Q)/R$.
(ii) when (v, Q) overlaps the top boundary of the contractum, then $Q|u' = Q_1^{\lceil\alpha\rceil b}$ and $u = vu'$, $Q_1 \preceq A$. Then $\mathsf{gen}(R, (v, Q)) = (v, Q\{u' := ((\lambda x.Q_1)^\alpha \_)^b\})$.
(iii) when (v, Q) overlaps the bottom boundary of the contractum, then $v = uv'$, $A|v' = Q_1$ and $Q \preceq Q_1\{x := Q_2^{\lfloor\alpha\rfloor}\}$ for some Q_2 prefix of B ($Q_2 \preceq B$). We

take Q_2 minimum. Then $\mathsf{gen}(R,(v,Q)) = (u, (\lambda x.\, v' \circ Q_1)^\alpha Q_2)^b$ where $v' \circ Q_1$ is the minimum prefix of A containing Q_1.

(iv) when (v, Q) overlaps both top and bottom boundaries of the contractum, this case is the union of the two previous ones.

When ρ is standard, this definition of generator corresponds to the canonical reduction. A reduction is canonical for $\langle \rho, R \rangle$ iff it is canonical for its generator at any step. The rest of the reduction is contained in the generator. Moreover when $\mathsf{INIT}(M)$, it is possible to backward reconstruct the canonical hredex $\langle \rho', R' \rangle$ of $\langle \rho, R \rangle$ by considering the generators of R. Thanks to proposition 5.8, we iterate on generators. The last step of the canonical reduction is defined by the rightmost overlined label in the generator which can be expanded in its previous-step generator. When there is no more overlined label, one considers innermost underlined labels. Proposition 5.8 says that paths from the binder of the last contracted redex are labeled by atomic letters. As $\mathsf{INIT}(M)$ is true, one can find the innermost binder in the initial term corresponding to the function part of the last contracted redex. Again one can compute the previous-step generator and proceed backwards in the reduction. Hence the canonical reduction only depends on the name of the hredex $\langle \rho, R \rangle$.

This concludes the proof of theorem 5.5.

6. Conclusion

Stability is everywhere in the λ-calculus, on sequential models, on Bohm trees, on redex families, on redexes. Stability means a unique deterministic origin of computed values or of computing objects. Sequentiality is a more elaborated concept, more interested by the prospective of calculus. Several interpretations of the λ-calculus are also sequential, for instance Bohm trees (Berry, 1978). It seems harder to define sequentiality for computing agents such as redexes or hredexes, but it would be interesting to get such a notion.

As said in the introduction, stability is also present in many other sequential languages, such as Plotkin's PCF, Kahn-Macqueen networks, or more realistic programming languages as we did for incremental computations of makefiles in Vesta (Abadi et al., 1996). Clearly the theory developed in this paper for the λ-calculus also applies to these cases.

Finally, this theory also applies to formalisms with critical pairs or with non-determinist features, but we have then to restrict to each compatible subclasses, in order to avoid conflicts, as described in event structures (Winskel, 1983; Laneve, 1994; Boudol, 1985).

Acknowledgements

I met Corrado Böhm at Swansea in 1974 and, since then, I admire his illuminating personality, his continual interest for the λ-calculus, and his famous separability

theorem (with use of *the magic combinators* $\lambda x x_1 x_2 \cdots x_n . x x_1 x_2 \cdots x_n$) fundamental in the theory of the λ-calculus and programming languages.

References

Abadi, M., Lampson, B., and Lévy, J.-J. (1996). Analysis and caching of dependencies. In *Proceedings of the 1996 ACM International Conference on Functional Programming*, pages 83–91.

Abramsky, S., Jagadeesan, R., and Malacaria, P. (2000). Full abstraction for PCF. *Information and Computation*, 167:409–470.

Amadio, R. and Curien, P.-L. (1998). *Domains and Lambda-calculi.* Cambridge University Press. ISBN 0-521-62277-8.

Asperti, A. and Laneve, C. (1995). Paths, computations and labels in the lambda-calculus. *Theoretical Computer Science*, 142(2):272–288.

Barendregt, H. (1977). *Handbook of Mathematical Logic*, chapter The Type Free Lambda Calculus. North-Holland. ISBN 0-7204-2285-X.

Barendregt, H. (1984). *The Lambda Calculus, Its Syntax and Semantics.* North-Holland. ISBN-13: 978-0-444-87508-2.

Berry, G. (1978). *Modèles stables du lambda-calcul.* PhD thesis, Paris 7.

Berry, G. and Curien, P.-L. (1982). Sequential algorithms on concrete data structures. *Theoretical Computer Science*, 20:265–321.

Berry, G. and Curien, P.-L. (1985). *Algebraic methods in Semantics*, chapter Theory and Practice of Sequential Algorithms: the Kernel of the Programming Language CDS. Cambridge University Press.

Berry, G. and Lévy, J.-J. (1979). Minimal and optimal computations of recursive programs. *JACM*, 26(1):148–175.

Boudol, G. (1985). *Algebraic methods in Semantics*, chapter Computational semantics of term rewriting systems. Cambridge University Press.

Cadiou, J.-M. (1972). *Recursive definitions of partial functions and their computations.* PhD thesis, Stanford University.

Church, A. (1941). *The Calculi of Lambda-Conversion.* Princeton University Press – Oxford University Press.

Curry, H. and Feys, R. (1958). *Combinatory Logic*, volume 1. North-Holland.

Curry, H., Feys, R., and Seldin, J. (1972). *Combinatory Logic*, volume 2. North-Holland.

Hindley, R. (1997). *Basic Simple Type Theory.* Cambridge University Press.

Huet, G. and Lévy, J.-J. (1991). *Computational Logic; Essays in Honor of Alan Robinson*, chapter Computations in Orthogonal Rewriting Systems I and II, pages 395–443. The MIT Press.

Klop, J.-W. (1980). Combinatory reduction systems. Mathematical Centre Tracts 127, CWI, Amsterdam. PhD thesis.

Laneve, C. (1994). Distributive evaluations of lambda-calculus. *Fundamenta Informaticae*, 20(4):333–352.

Lévy, J.-J. (1975). An algebraic interpretation of the $\lambda\beta$-calculus and a labeled λ-calculus. In Böhm, C., editor, *Lambda-Calculus and Computer Science Theory*, volume 37 of *Lecture Notes in Computer Science*, pages 147–165. Springer.

Lévy, J.-J. (1978). *Réductions correctes et optimales dans le lambda calcul.* PhD thesis, Univ. of Paris 7. (In French).

Lévy, J.-J. (1980). *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, chapter Optimal reductions in the lambda-calculus, pages 159–191. Academic Press.

Lévy, J.-J. (2009). *From Semantics to Computer Science: Essays in Honour of Gilles Kahn*, chapter Generalized Finite developments, pages 185–204. Cambridge University Press.

Maranget, L. (1992). *La stratégie paresseuse.* PhD thesis, Univ. Paris 7.

Melliès, P.-A. (1997). A factorisation theorem in rewriting theory. In *Proceedings of the 7th Conference on Category Theory and Computer Science*, number 1290 in Lecture Notes in Computer Science, pages 49–68, Santa Margherita Ligure. Springer.

Plotkin, G. (1977). LCF considered as a programming language. *Theoretical Computer Science*, 5.

van Oostrom, V. and de Vrijer, R. (2002). Four equivalent equivalences of reductions. In *WRS'02*, volume 70(6) of *ENTCS*.

van Oostrom, V. and de Vrijer, R. (2003). *Terese, Term Rewriting Systems*, chapter Equivalence of reductions. Cambridge University Press.

Vuillemin, J. (1973). *Proof Techniques for Recursive Programs.* PhD thesis, Stanford University.

Winskel, G. (1983). Event structure semantics for CCS and related languages. Technical report, DAIMI – Aarhus University.