Rewriting with Extensional Polymorphic λ-calculus

Roberto Di Cosmo[1] and Delia Kesner[2]

[1] LIENS (CNRS) - DMI
Ecole Normale Supérieure
45, Rue d'Ulm
75005 Paris - France, E-mail: `dicosmo@dmi.ens.fr`
[2] CNRS and LRI
Bât 490, Université de Paris-Sud
91405 ORSAY Cedex, France, E-mail: `kesner@lri.lri.fr`

Abstract. We provide a confluent and strongly normalizing rewriting system, based on expansion rules, for the *extensional* second order typed lambda calculus with product and *unit* types: this system corresponds to the Intuitionistic Positive Calculus with implication, conjunction, quantification over proposition and the constant *True*. This result is an important step towards a new theory of reduction based on expansion rules, and gives a natural interpretation to the notion of second order η-long normal forms used in higher order resolution and unification, that are here just the normal forms of our reduction system.

1 Introduction

Typed lambda calculus provides a convenient framework for studying functional programming and offers a natural formalism to deal with proofs in intuitionistic logic. It comes traditionally equipped with a fundamental computational mechanism, which is the β equality $(\lambda x.M)N = M[N/x]$, and with a minimal tool for reasoning about programs, which is the η *extensional* equality $\lambda x.Mx = M$. This basic calculus can then be extended by adding further types, like products, unit and second order types, each coming with its own computational mechanism and/or its extensional equalities.

This work provides a confluent and strongly normalizing rewriting system for second order lambda calculus equipped with extensionality axioms for the product and for the arrow type, no longer oriented as *contractive* rules, but, according to the now growing practice [Aka93, Dou93, DCK94b, Cub92, DCK94a, DCP95, JG92], as *expansive* rewriting rules.

Using expansive rewrite rules, in a controlled fashion, allows us to obtain a simple canonical system that can be used to decide equality, has the second order Huet's $\beta\eta$-long normal forms as normal forms and is naturally compatible with rules that break confluence when added to systems with contractive rules for extensional equalities. This is the case, for example, of the special *unit* type T and the rule $M : T \longrightarrow * : T$ that one needs to define the extensional first order lambda calculus associated to Cartesian Closed Categories. Another example is the extensional first order lambda calculus enriched with a confluent algebraic rewriting system, where confluence is also broken by contractive rules [DCK94a].

As it has been noticed before, the expansive interpretation of extensional rules is not new, but has been neglected for a long time, so that the theory of lambda calculus with expansion rules is far from being fully developed: even for η-contraction, that has been extensively studied for a long time, a satisfactory treatment has been provided only recently [Geu92].

A first positive result for polymorphic systems is presented in [DCP95], but just for the first order expansive η rule: unfortunately, the proof technique used there, which is based essentially on η postponement, does not extend to polymorphic η expansion nor to expansive surjective pairing.

Here we provide a proof of confluence and normalization for the full polymorphic system with the extensionality of arrow, product and universal types turned into expansive rules. Since the rewriting relation with controlled expansive rules is not a congruence and is not stable by substitution, we cannot use the traditional reducibility proofs, that strongly rely on these properties.

Our proof is based on a *modification* of Girard's reducibility candidates, as first suggested for the first order calculus in [Jay92], that requires some careful reordering of the traditional lemmas in Girard's proof, but works homogeneously for products as well as arrow and polymorphic types. A related approach has been independently taken in [Gha95] for the fragment without products and terminal object: there a different modification of reducibility is introduced, which seems very promising.

It is worth to notice that the expansive interpretation of extensional equalities does not fit the various higher order schemes proposed for example in [Klo80, JO91, Nip90], as those schemes do not allow the left hand side of the reduction rules to be a single higher-order variable.

We also present a simple argument that allows to add in a fully modular way the special rule for the *unit* type to any rewriting system, preserving normalization.

Overall, this provides us with the first canonical rewriting system for the full polymorphic extension of the first order lambda calculus associated to Cartesian Closed Categories, an important step towards a full theory of expansive extensionality.

1.1 Brief Survey

Several proof techniques have been developed to tackle the expansionary interpretation of the extensional equalities, and show that it yields a confluent and normalizing system in the first order case. One idea is to try to separate the expansion rules from the rest of the reduction, and then try to show some kind of modularity of the reduction systems. In [Aka93] this is done by means of the following property:

Lemma 1. *Let S and R be confluent and strongly normalizing reductions, s.t.*

$$\forall M, N \quad (M \stackrel{R}{\Longrightarrow} N) \quad \textit{implies} \quad (M^S \stackrel{R}{\Longrightarrow}{}^{+} N^S)$$

where M^S is the S normal form of M, then $S \cup R$ is also confluent and strongly normalizing.

The lemma applies to S taken as the expansions and R the usual reductions. In [DCK94b], we reduced confluence and strong normalization of the full expansionary system to that of the traditional one without expansions using another modular technique based on a translation.

A different non-modular approach is taken in [JG92] and [Dou93], where the proofs of strong normalization are based on an extension of the traditional techniques of reducibility and allow to handle also the peculiarity of the expansion rules. But that is not all, since one is left to prove weak confluence separately, which is not an easy task in the presence of expansion rules, as the rewrite relation is not a congruence (see [DCK94b] for details).

An even different technique is used in [Cub92], where confluence is shown by a careful study of the residuals in the reduction.

Finally, in [DCP95], one finds a simple general lemma, that appears also in [Ges90], and that is today the most satisfactory tool to handle the addition of expansion rules to first order systems, since it is fully modular and extremely easy to apply.

Lemma 2. *Let $\langle \mathcal{A}, R, S\rangle$ be an Abstract Reduction System, where R-reduction is strongly normalizing. Let the following commutation hold*

$$\forall a, b, c, d \in \mathcal{A} \quad \begin{array}{ccc} a & \xrightarrow{R} & c \\ {\scriptstyle S}\downarrow & & \downarrow\!\!\downarrow{\scriptstyle S} \\ b & \overset{R}{\underset{+}{\twoheadrightarrow}} & d \end{array}$$

Then S^* and R^+ commute, so, in particular, S^* and R^* commute.

Since the union of two confluent reductions that commute is confluent, by Hindley-Rosen's lemma (see [Bar84]), this lemma gives an easy way of proving confluence modularly: in particular, if we take expansive η and SP as S, and the rest of the system as R, all *first order* systems cited above satisfy the lemma, so confluence comes up immediately from the separate confluence of the expansions alone and of the non expansive subsystems.

As for normalization, if R preserves S normal forms, and S and R are both confluent and normalizing, it is easy to see that commutation also entails the hypothesis of Akama's lemma, so also normalization comes up modularly. This is the case in all first order systems: since the type of any subterm do not evolve during reduction, expansive normal forms are preserved, so one gets immediately strong normalization for the full system.

For these reasons, we can consider that the treatment of expansive rules in first order systems is nowadays fully satisfactory. Unfortunately, in the presence of second order quantification, the type of a subterm *can evolve* during evaluation, and this fact allows us to build very simple examples suggesting that the modular approaches [Aka93, DCK94b, DCP95] cannot be satisfactorily extended to the second order case, and we have to go back and have a better look at reducibility candidates.

Expansions and polymorphism are not modular The following simple example shows that we cannot use the modular techniques developed up to now to separate the complexities introduced by expansion rules and polymorphic typing.

Example 1. Let $M = (\Lambda X.(x[X \to X])(\lambda y : X.y))[A \times B]$ where $x : \forall Z.(Z \to A)$ and A and B are base types. Then, the term M is a normal form w.r.t expansion rules, but its immediate β^2 reduct is not:

$$M' = (x[A \times B \to A \times B])(\lambda y : A \times B.y)$$

In fact, M' reduces to the term

$$M'' = (x[A \times B \to A \times B])(\lambda y : A \times B.\langle \pi_1(y), \pi_2(y)\rangle)$$

Now, there is no way to reduce M to M'', which is the (η, SP)-normal form of M', without using expansions: this means that the hypothesis of lemma 1 are not satisfied, and we fail to obtain normalization modularly (also lemma 2 alone is no good for normalization).

This same example can be used to show how the use of expansor terms of [DCK94b] is neither viable.

Our approach Since the modular techniques are not viable, we focused on the reducibility predicates defined in [JG92] for the first order calculi with expansion rules, and we adapt the strong normalization proof to handle both second order quantification and expansion rules, by a careful reorganisation of the traditional arguments.

One fundamental difference between the proof for the simply typed and the proof for polymorphic lambda calculus is that one does not work with just one reducibility candidate, but with all reducibility candidates at once. This requires to deal with many subtle points in the second order case that do not appear in the first order case.

The paper is organized as follows: we define the calculus in section 2 and we show weak confluence of the full reduction system in section 3. We then give the proof of strong normalization in section 4, add the Top type in section 5 and we finally conclude with some ideas for further work.

We will use some standard notions from the theory of rewriting system, such as redex, normal form, confluence, weak confluence and strong normalization. See [Bar84] for references.

2 The Calculus

We consider a denumerable set of atomic types and a denumerable set of type variables. The set of types of our calculus can be defined by the following grammar:

$$A ::= \iota \ \mid\ X \ \mid\ A \times A \ \mid\ A \to A \ \mid\ \forall X.A$$

where ι ranges over the set of atomic types and X over the set of type variables.

Variables are typed by the following axiom:

$$x_1 : A_1, \ldots, x_n : A_n \vdash x_i : A_i \qquad (1 \leq i \leq n)$$

where the x_j's are pairwise distinct.

And terms are typed by the following rules:

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x : A.M : A \to B} \qquad \frac{\Gamma \vdash M : A \to B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B}$$

$$\frac{\Gamma \vdash M_1 : A_1 \quad \Gamma \vdash M_2 : A_2}{\Gamma \vdash \langle M_1, M_2 \rangle : A_1 \times A_2} \quad \frac{\Gamma \vdash M : B_1 \times B_2}{\Gamma \vdash \pi_1(M) : B_1} \quad \frac{\Gamma \vdash M : B_1 \times B_2}{\Gamma \vdash \pi_2(M) : B_2}$$

$$\frac{\Gamma \vdash M : A \quad \text{X not free in } \Gamma}{\Gamma \vdash \Lambda X.M : \forall X.A} \qquad \frac{\Gamma \vdash M : \forall X.A}{\Gamma \vdash M[B] : A[B/X]}$$

We write $M : A$ is M has type A under some context Γ. The set of free variables of a term M, denoted $FV(M)$ is defined as usual. If M is a term and θ a substitution, $M\theta$ is the term M where each $x \in FV(M)$ is replaced by $\theta(x)$. Substitutions are *typed*, in the sense that the substituted variable and term must have the same type.

We identify terms up to α-conversion, *i.e.* renaming of bound variables. The reduction rules are the following (we refer the interested reader to [DCK94b] for a discussion of the restrictions on the expansion rules):

(β) $(\lambda x : A.M)N \xrightarrow{\beta} M[N/x]$

(β^2) $(\Lambda X.M)[A] \xrightarrow{\beta^2} M[A/X]$

(π_i) $\pi_i\langle M_1, M_2\rangle \xrightarrow{\pi_i} M_i$, for $i = 1, 2$

(δ) $M \xrightarrow{\delta} \langle \pi_1(M), \pi_2(M)\rangle$, if $\begin{cases} M : A \times B \\ M \text{ is not a pair} \end{cases}$

(η) $M \xrightarrow{\eta} \lambda x : A.Mx$, if $\begin{cases} x \notin FV(M) \\ M : A \to C \\ M \text{ is not a } \lambda\text{-abstraction} \end{cases}$

(η^2) $M \xrightarrow{\eta^2} \Lambda X.M[X]$, if $\begin{cases} X \text{ is not free in } M \\ X \text{ is not free in the type of any free variable of } M \\ M : \forall Y.A \\ M \text{ is not a } \Lambda\text{-abstraction} \end{cases}$

The one-step reduction relation between terms, denoted $\Longrightarrow$, is defined as the closure of the reduction rules $\beta, \beta^2, \eta, \eta^2, \pi_1, \pi_2, \delta$ for all the contexts *except* in the application and projection cases, *i.e*:

- If $M \Longrightarrow M'$, then $MN \Longrightarrow M'N$ except in the case $M \xrightarrow{\eta} M'$
- If $M \Longrightarrow M'$, then $M[A] \Longrightarrow M'[A]$ except in the case $M \xrightarrow{\eta^2} M'$
- If $M \Longrightarrow M'$, then $\pi_i(M) \Longrightarrow \pi_i(M')$ except in the case $M \xrightarrow{\delta} M'$

Notation 3 *The transitive and the reflexive transitive closure of* $\Longrightarrow$ *are noted* $\Longrightarrow^+$ *and* $\Longrightarrow^*$ *respectively.*

3 Weak Confluence

Since we already know that the first order fragment of the calculus is confluent [Aka93, DCK94b, JG92, Dou93, Cub92], this task is greatly simplified: we are left to examine only the critical pairs that arise from the use of the second order rules β^2 and η^2.

We first prove a substitution lemma for *types*:

Lemma 4. *If* $Q \Longrightarrow Q'$*, then* $Q[A/X] \Longrightarrow Q'[A/X]$

Proof. By a simple case analysis.

Then, we can proceed to prove the following

Lemma 5 Critical Pairs. *If* $M \longrightarrow M_1$ *and* $M \Longrightarrow M_2$*, there is* M_3 *such that* $M_1 \Longrightarrow^* M_3$ *and* $M_2 \Longrightarrow^* M_3$*.*

Proof. We analyse only the cases where $M \longrightarrow M_1$ is an $\xrightarrow{\eta^2}$ or $\xrightarrow{\beta^2}$, all the other cases can be found in [DCK94b].

1.

$$\begin{tikzcd}
(\Lambda X.Q)[A] \arrow[r, Rightarrow] \arrow[d, "\beta^2"'] & (\Lambda X.Q')[A] \arrow[d, "\beta^2"'] \\
Q[A/X] \arrow[r, Rightarrow, "\text{by lemma 4}"'] & Q'[A/X]
\end{tikzcd}$$

2.

$$\begin{tikzcd}
(\Lambda X.Q)[A] \arrow[r, "\eta"] \arrow[d, "\beta^2"'] & \lambda y : B.((\Lambda X.Q)[A])y \arrow[d, "\beta^2"'] \\
Q[A/X] & \lambda y : B.(Q[A/X])y
\end{tikzcd}$$

If $Q[A/X]$ is not a λ-abstraction, then $Q[A/X] \xrightarrow{\eta} \lambda y : B.(Q[A/X])y$, otherwise $Q[A/X] = \lambda z.N$ and we have

$$\lambda y : B.(\lambda z.N)y \xrightarrow{\beta} \lambda y : B.(N[y/z]) =_\alpha \lambda z.N = Q[A/X]$$

3.

$$\begin{tikzcd}
(\Lambda X.Q)[A] \arrow[r, "\delta"] \arrow[dd, "\beta^2"'] & \langle \pi_1((\Lambda X.Q)[A]), \pi_2((\Lambda X.Q)[A]) \rangle \arrow[d, "\beta^2"'] \\
& \langle \pi_1(Q[A/X]), \pi_2((\Lambda X.Q)[A]) \rangle \arrow[d, "\beta^2"'] \\
Q[A/X] & \langle \pi_1(Q[A/X]), \pi_2(Q[A/X]) \rangle
\end{tikzcd}$$

If $Q[A/X]$ is not a pair, then $Q[A/X] \xrightarrow{\delta} \langle \pi_1(Q[A/X]), \pi_2(Q[A/X]) \rangle$, otherwise $Q[A/X] = \langle N_1, N_2 \rangle$ and we have

$$\langle \pi_1(\langle N_1, N_2 \rangle), \pi_2(\langle N_1, N_2 \rangle) \rangle \xrightarrow{\pi_1} \langle N_1, \pi_2(\langle N_1, N_2 \rangle) \rangle \xrightarrow{\pi_2} \langle N_1, N_2 \rangle = Q[A/X]$$

4.

$$\begin{tikzcd} (\Lambda X.Q)[A] \arrow[r, "\eta^2"] \arrow[d, "\beta^2"'] & \Lambda Y.((\Lambda X.Q)[A])[Y] \arrow[d, "\beta^2"'] \\ Q[A/X] & \Lambda Y.(Q[A/X])[Y] \end{tikzcd}$$

If $Q[A/X]$ is not a Λ-abstraction, then $Q[A/X] \xrightarrow{\eta^2} \Lambda Y.(Q[A/X])[Y]$, otherwise $Q[A/X] = \Lambda Z.N$ and we have

$$\Lambda Y.(\Lambda Z.N)y \xrightarrow{\beta^2} \Lambda Y.(N[Y/Z]) =_\alpha \Lambda Z.N = Q[A/X]$$

5.

$$\begin{tikzcd} Q \arrow[r, Rightarrow] \arrow[d, "\eta^2"'] & Q' \arrow[d, "\eta^2"'] \\ \Lambda Y.Q[Y] \arrow[r, Rightarrow] & \Lambda Y.Q'[Y] \end{tikzcd}$$

This is because if $Q \Longrightarrow Q'$ is an internal reduction, it is not a root expansion and then $Q[Y] \Longrightarrow Q'[Y]$ which implies also $\Lambda Y.Q[Y] \Longrightarrow \Lambda Y.Q'[Y]$

6.

$$\begin{tikzcd} (\lambda y.M)N \arrow[r, "\beta"] \arrow[d, "\eta^2"'] & M[N/y] \\ \Lambda Y.((\lambda y.M)N)[Y] & \end{tikzcd}$$

If $M[N/y]$ is not a Λ-abstraction, then $M[N/y] \xrightarrow{\eta^2} \Lambda Y.(M[N/y])[Y]$, otherwise $M[N/y] = \Lambda Z.L$ and we have:

$$\Lambda Y.(\Lambda Z.L)[Y] \xrightarrow{\beta^2} \Lambda Y.L[Y/Z] =_\alpha \Lambda Z.L = M[N/y]$$

Finally, the proof of the weak confluence for the calculus proceeds exactly as in Theorem 4.11 of [DCK94b].

4 Strong normalization

In this section we prove strong normalization for the full system by means of a notion of reducibility modified as in [JG92].

4.1 Definitions

Definition 6 (neutral terms) *A term $t{:}U$ is neutral if it is not a λ-abstraction, a type abstraction or a pair.*

Definition 7 (longest reduction path for a term, or rank) *Let u be a term, then $\nu(u)$ denotes the length of the longest reduction path starting from u. Notice that, by König's Lemma, if u is strongly normalisable then $\nu(u)$ is finite.*

Definition 8 *A reducibility candidate of type U is a set R of terms of type U with the following properties.*

(CR1) *if $t \in R$, then t is strongly normalisable.*
(CR2) *if $t \in R$ and $t \longrightarrow t'$, then $t' \in R$.*
(CR3) *if t is neutral and for all t' s.t. $t \longrightarrow t'$* **other than a basic expansion** *we have that $t' \in R$, then $t \in R$.*

The modification written in bold in CR3 was first suggested in [Jay92] to adapt Girard's reducibility proof for the simply typed λ-calculus with expansion rules.

To discuss its implications, let us recall here the steps that one follows in a traditional reducibility proof for second order calculi. First, one defines the notion of reducibility candidate, remarks that the set of strongly normalizing terms is a reducibility candidate, defines the notion of product and function space for reducibility candidates, and shows that the product and function spaces of any given reducibility candidate is still a reducibility candidate. Then one defines a notion of reducibility candidate *with parameters* $RED[\overrightarrow{R}/\overrightarrow{X}]$ for all types T and shows by induction on the types that any reducibility candidate with parameters is a reducibility candidate. Finally, using a bunch of technical lemmas, one shows by structural induction on the terms that every instance of a term t (and in particular t itself) is reducible, hence strongly normalizable.

In the presence of expansions, we have to alter the order of several lemmas, and it is not evident that the set of strongly normalizing terms form a reducibility candidate in general, while this is trivial with Girard's formulation of CR3.

Fortunately, it will be enough to show that the set of strongly normalizing terms of *atomic or variable types only* are a reducibility candidate.

Proposition 9. *The set of strongly normalizable terms of atomic type or variable type are a reducibility candidate.*

Proof. Just notice that on terms of atomic type or variable type the modified (CR3) condition is equal to Girard's original (CR3) condition, because no root expansion is possible, so the proof goes through as in the usual case without expansions.

Remark. A reducibility candidate R, if it exists, is never empty as CR3 implies that it always contains the variables of type U, because they are neutral and there is no reduction other than basic expansions leaving them.

We will often write in the sequel s.n. for strongly normalizing.

Lemma 10 Normalization lemma.

(i) *If $t : A \times B$ is a term such that $\pi_1(t) : A$ and $\pi_2(t)$ are s. n. then so is t.*
(ii) *If $t : A \to B$ is a term and $x : A$ is a variable not free in t such that $tx : B$ is s. n. then t is s. n.*
(iii) *If $t : \forall Y.A$ is a term and X is a type variable not free in t such that $t[X] : A[X/Y]$ is s. n. then t is s. n..*
(iv) *If $t : A$ is a s. n. term, then $\lambda x.t$ and $\Lambda X.t$ are s. n. too.*

Proof. By induction on the maximal reduction lengths of the strongly normalizing terms.

If R and S are reducibility candidates of types U and V, we can define the following sets:

$t \in R \times S \Longleftrightarrow \pi_1(t) \in U$ and $\pi_2(t) \in V \qquad t \in R \to S \Longleftrightarrow$ for all $u \in R, tu \in S$

Lemma 11 Pairing. *Let R_1 and R_2 be reducibility candidates of types A_1 and A_2 and let $u : A_1 \in R_1$ and $v : A_2 \in R_2$. Then $\pi_i\langle u, v\rangle : A_i \in R_i$.*

Proof. By induction on the normalization lengths of u and v, using (CR3), as usual.

Theorem 12. *If R_1 and R_2 are reducibility candidates of types U_1 and U_2, then $R_1 \times R_2$ is a reducibility candidate of type $U_1 \times U_2$.*

Proof. The proof is still done by induction on the type. It is rather different from the usual one, so we show it in full in the Appendix.

4.2 Reducibility with parameters

Let T be a type, and $\overrightarrow{X}$ be a set of type variables that contains at least all the free type variables of T. For $\overrightarrow{U}$ a sequence of types of the same length, let $T[\overrightarrow{U}/\overrightarrow{X}]$ be the type obtained by simultaneous substitution of the X's with the U's, and $\overrightarrow{R}$ a sequence of reducibility candidates of corresponding types.

Definition 13 *The set $RED_T[\overrightarrow{R}/\overrightarrow{X}]$ of reducible terms of type T is defined by induction on the type T as follows (where the $\overrightarrow{R}$ are reducibility candidates of type $\overrightarrow{U}$).*

- if T is atomic, $RED_T[\overrightarrow{R}/\overrightarrow{X}]$ is the set of s. n. terms of type T
- if T is X_i, $RED_T[\overrightarrow{R}/\overrightarrow{X}]$ is R_i
- if T is $U \times V$ then $RED_T[\overrightarrow{R}/\overrightarrow{X}]$ is $RED_U[\overrightarrow{R}/\overrightarrow{X}] \times RED_V[\overrightarrow{R}/\overrightarrow{X}]$
- if T is $U \to V$ then $RED_T[\overrightarrow{R}/\overrightarrow{X}]$ is $RED_U[\overrightarrow{R}/\overrightarrow{X}] \to RED_V[\overrightarrow{R}/\overrightarrow{X}]$

– if T is $\forall Y.W$ then $RED_T[\overrightarrow{R}/\overrightarrow{X}]$ is the set of terms t of type $T[\overrightarrow{U}/\overrightarrow{X}]$ such that, for every type V and reducibility candidate S of this type, $t[V] \in RED_W[\overrightarrow{R}/\overrightarrow{U}, S/Y]$

Lemma 14 Universal abstraction. *If for every type V and candidate S, $RED_W[\overrightarrow{R}/\overrightarrow{X}, S/Y]$ is a reducibility candidate and $v[V/Y] \in RED_W[\overrightarrow{R}/\overrightarrow{X}, S/Y]$, then $\Lambda Y.v \in RED_{\forall Y.W}[\overrightarrow{R}/\overrightarrow{X}]$*

Proof. We need to show that $(\Lambda Y.v)[V] \in RED_W[\overrightarrow{R}/\overrightarrow{X}, S/Y]$ for every type V and candidate S of type V. We can argue by induction on $\nu(v)$, because v is strongly normalizable as $v = v[Y/Y]$ is in $RED_W[\overrightarrow{R}/\overrightarrow{X}, SN_Y/Y]$, which is a reducibility candidate by hypothesis. Since $(\Lambda Y.v)[V]$ is neutral, we will use CR3 for $RED_W[\overrightarrow{R}/\overrightarrow{X}, S/Y]$.

Converting a redex of $(\Lambda Y.v)[V]$ without using basic expansions can yield:

– $v[V/Y]$, which is in $RED_W[\overrightarrow{R}/\overrightarrow{X}, S/Y]$ by hypothesis
– $(\Lambda Y.v')[V]$ with v' one step from v; now, $v[V/Y]$ reduces to $v'[V/Y]$[3], so we can apply induction on $\nu(v)$ and get $(\Lambda Y.v')[V] \in RED_W[\overrightarrow{R}/\overrightarrow{X}, S/Y]$

The results follows then by CR3 for $RED_W[\overrightarrow{R}/\overrightarrow{X}]$.

In the following lemma we will talk about *subtypes* of a given type, where the subtypes of $A \to B$ are given by $A \to B$ itself together with the subtypes of A and B, and similarly for $A \times B$, while the subtypes of $\forall X.A$ are given by $\forall X.A$ itself together with the subtypes of A. This notion also provide a natural order over types.

Lemma 15 Abstraction.
Let $x : U$ and $v : V$, and suppose that $RED_W[\overrightarrow{R}/\overrightarrow{X}]$ is a reducibility candidate for all W subtypes of U or V (inclusive). If for all $u \in RED_U[\overrightarrow{R}/\overrightarrow{X}]$ we have $v[u/x] \in RED_V[\overrightarrow{R}/\overrightarrow{X}]$, then $\lambda x.v \in RED_{U \to V}[\overrightarrow{R}/\overrightarrow{X}]$

Proof. Assume the property is true for all subtypes of U and V.

To show that $\lambda x.v \in RED_{U \to V}[\overrightarrow{R}/\overrightarrow{X}]$, we need to show that $(\lambda x.v)u \in RED_V[\overrightarrow{R}/\overrightarrow{X}]$ for all $u \in RED_U[\overrightarrow{R}/\overrightarrow{X}]$, and we will do so by using CR3 for $RED_V[\overrightarrow{R}/\overrightarrow{X}]$.

We know that $x : U$ is in $RED_U[\overrightarrow{R}/\overrightarrow{X}]$ (remark 4.1).

So $v = v[x/x]$ is in $RED_V[\overrightarrow{R}/\overrightarrow{X}]$, hence strongly normalizable by CR1 and we can argue by induction on $\nu(u) + \nu(v)$ to show that all terms one step from $(\lambda x.v)u$ are reducible.

The term $(\lambda x.v)u$ converts to

[3] Since type substitution does not alter the structure of a term, we can perform on $v[V/Y]$ all the reductions we could perform on v, or maybe more as new expansion redexes can arise, but surely not less.

– $v[u/x]$ that is in $RED_V[\overrightarrow{R}/\overrightarrow{X}]$ by hypothesis.

– $(\lambda x.v')u$ with v' one step from v. To apply the induction hypothesis using the fact that $\nu(v') < \nu(v)$, and get $(\lambda x.v')u \in RED_V[\overrightarrow{R}/\overrightarrow{X}]$, we need to show now that $v'[u/x]$ is in $RED_V[\overrightarrow{R}/\overrightarrow{X}]$. If $v'[u/x]$ is one step from $v[u/x]$, then this comes from CR2 for $RED_V[\overrightarrow{R}/\overrightarrow{X}]$.

  If $v'[u/x]$ is not one step from $v[u/x]$, then v' is obtained from v by an expansion of an occurrence of x and u is either a pair, a λ-abstraction or a type abstraction. Notice now that if u is a pair, $\langle \pi_1(u), \pi_2(u)\rangle$ is reducible by Lemma 11; if u is an abstraction of type $U = C \to D$ then by definition of $u \in RED_U[\overrightarrow{R}/\overrightarrow{X}]$, $ut \in RED_D[\overrightarrow{R}/\overrightarrow{X}]$ for all $t \in RED_C[\overrightarrow{R}/\overrightarrow{X}]$, and by induction hypothesis on the type we have that $\lambda y.uy \in RED_{C\to D}[\overrightarrow{R}/\overrightarrow{X}]$; finally, if u is a type abstraction of type $U = \forall X.W$, then by definition of $u \in RED_U[\overrightarrow{R}/\overrightarrow{X}]$, for every type V and reducibility candidate S of this type, $u[X][V/X] = u[V] \in RED_W[\overrightarrow{R}/\overrightarrow{U}, S/Y]$, which is a reducibiblity candidate by hypothesis, so we have that $\Lambda X.u[X] \in RED_{\forall X.D}[\overrightarrow{R}/\overrightarrow{X}]$ by lemma 14.
  Let us now use $\eta(u)$ for either $\langle \pi_1(u), \pi_2(u)\rangle$ with u a pair or $\lambda y.uy$ with u a λ-abstraction or $\Lambda X.u[X]$ with u a type abstraction. What we have just shown means that by hypothesis $v[\eta(u)/x] \in RED_V[\overrightarrow{R}/\overrightarrow{X}]$, but it is straightforward to see that $\eta(u) \twoheadrightarrow u$, so we can build a reduction sequence $v[\eta(u)/x] \twoheadrightarrow v'[u/x]$, and then use repeatedly CR2 for $RED_V[\overrightarrow{R}/\overrightarrow{X}]$ to deduce $v'[u/x] \in RED_V[\overrightarrow{R}/\overrightarrow{X}]$, as neeeded.
– $(\lambda x.v)u'$ with u' one step from u. Then $u' \in RED_U[\overrightarrow{R}/\overrightarrow{X}]$ by CR2, $\nu(u') < \nu(u)$ and $v[u'/x] \in RED_V[\overrightarrow{R}/\overrightarrow{X}]$ by repeated applications of CR2, as it is some steps from v[u/x]. So we can apply again the induction hypothesis.

Since $(\lambda x.v)u$ is neutral and it converts to terms in $RED_V[\overrightarrow{R}/\overrightarrow{X}]$ only, it is in $RED_V[\overrightarrow{R}/\overrightarrow{X}]$ too. Hence $\lambda x.v$ is in $RED_{U\to V}[\overrightarrow{R}/\overrightarrow{X}]$ by definition.

Theorem 16. *$RED_T[\overrightarrow{R}/\overrightarrow{X}]$ is a reducibility candidate of type $T[\overrightarrow{U}/\overrightarrow{X}]$.*

Proof. We proceed by structural induction on the type T. We show only the cases for the arrow and universal types, which are the ones that change with expansions.

(Arrow types) Let T be $U_1 \to U_2$. We know by induction hypothesis that $RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}]$, $RED_{U_2}[\overrightarrow{R}/\overrightarrow{X}]$ and in fact all $RED_W[\overrightarrow{R}/\overrightarrow{X}]$ for W subtype of the U_i, are reducibility candidates, and we will now show that $RED_{U_1\to U_2}[\overrightarrow{R}/\overrightarrow{X}] = RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}] \to RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}]$ is a reducibility candidate.

– (CR1) if $t \in RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}] \to RED_{U_2}[\overrightarrow{R}/\overrightarrow{X}]$, then let x be a variable of type U_1. Since $x \in$ *any* reducibility candidate of type U_1, (remark 4.1), in particular $x \in RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}]$ and we get that $(tx) \in RED_{U_2}[\overrightarrow{R}/\overrightarrow{X}]$ by definition, hence (tx) is strongly normalisable by CR1 for $RED_{U_2}[\overrightarrow{R}/\overrightarrow{X}]$, which is a reducibility candidate by induction hypothesis. This suffices to show that t is strongly normalisable, by lemma 10.

– (CR2) if $t \longrightarrow t'$, we need to show that $(t'u) \in RED_{U_2}[\overrightarrow{R}/\overrightarrow{X}]$ for all term $u \in RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}]$. Take $u \in RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}]$; we have $(tu) \in RED_{U_2}[\overrightarrow{R}/\overrightarrow{X}]$ by definition of reducibility candidate. If $(tu) \longrightarrow (t'u)$, then $(t'u) \in RED_{U_2}[\overrightarrow{R}/\overrightarrow{X}]$ by CR2 for $RED_{U_2}[\overrightarrow{R}/\overrightarrow{X}]$, which is a reducibility candidate by induction hypothesis. Otherwise, $t \longrightarrow \lambda x : U_1.tx$, but $(tu) = (tx)[u/x]$ is in $RED_{U_2}[\overrightarrow{R}/\overrightarrow{X}]$ for any $u \in RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}]$ by induction hypothesis, so we can conclude by applying lemma 15.

– (CR3) t is neutral and all t' one step from t other than basic expansions are in $RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}] \rightarrow RED_{U_2}[\overrightarrow{R}/\overrightarrow{X}]$. In order to show $t \in RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}] \rightarrow RED_{U_2}[\overrightarrow{R}/\overrightarrow{X}]$, we need to show that $(tu) \in RED_{U_2}[\overrightarrow{R}/\overrightarrow{X}]$ for all term $u \in RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}]$.
By induction hypothesis on $RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}]$, we get u is strongly normalisable, so we can argue by induction on $\nu(u)$.
In one step, and without performing basic expansions, (tu) converts to:
  - $(t'u)$ with t' one step from t.
    As $t' \in RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}] \rightarrow RED_{U_2}[\overrightarrow{R}/\overrightarrow{X}]$, we get $(t'u) \in RED_{U_2}[\overrightarrow{R}/\overrightarrow{X}]$ by definition.
  - (tu') with u' one step from u.
    By induction hypothesis on $RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}]$, $u' \in RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}]$ and $\nu(u') < \nu(u)$, so $(tu') \in RED_{U_2}[\overrightarrow{R}/\overrightarrow{X}]$ by the induction hypothesis on u.

(Universal types) Let $T = \forall Y.W$.

– (CR1) if $t \in RED_{\forall Y.W}[\overrightarrow{R}/\overrightarrow{X}]$, then let X be an arbitrary type variable and S be an arbitrary reducibility candidate of type X (for example, the strongly normalizable terms of type X). Then $t[X] \in RED_W[\overrightarrow{R}/\overrightarrow{X}, S/Y]$ by definition, so by induction hypothesis we know that $t[X]$ is strongly normalizable. Then we can conclude by lemma 10 that t is strongly normalisable.

– (CR2) let $t \longrightarrow t'$; for all types V and reducibility candidate S of this type, we have that $t[V] \in RED_W[\overrightarrow{R}/\overrightarrow{X}, S/Y]$, so if $t[V] \longrightarrow t'[V]$, then $t'[V] \in RED_W[\overrightarrow{R}/\overrightarrow{X}, S/Y]$ by induction hypothesis on W.
Otherwise $t \longrightarrow \Lambda X.t[X] = t'$: we know by inductive hypothesis that for all types V and reducibility candidate S of this type $RED_W[\overrightarrow{R}/\overrightarrow{X}, S/Y]$ is a reducibility candidate and that $t[Y][V/Y] = t[V] \in RED_W[\overrightarrow{R}/\overrightarrow{X}, S/Y]$, so we can apply lemma 14 and we get $t' = \Lambda X.t[X] \in RED_{\forall Y.W}[\overrightarrow{R}/\overrightarrow{X}]$, and finally we get $t'[V] \in RED_W[\overrightarrow{R}/\overrightarrow{X}, S/Y]$ by definition.
In any case, $t'[V] \in RED_W[\overrightarrow{R}/\overrightarrow{X}]$, so by definition $t' \in RED_{\forall Y.W}[\overrightarrow{R}/\overrightarrow{X}]$.

– (CR3) t is neutral and all t' one step from t other than basic expansions are in $RED_T[\overrightarrow{R}/\overrightarrow{X}]$. Take V and S: if we apply a conversion other than a basic expansion to $t[V]$, the only possible result is $t'[V]$ (since t is neutral), and t' cannot be a basic expansion of t (because t is applied). Now, this means that t' is in $RED_T[\overrightarrow{R}/\overrightarrow{X}]$, so by definition $t'[V]$ is in $RED_W[\overrightarrow{R}/\overrightarrow{X}, S/Y]$, and by (CR3) for W we get that $t[V]$ is in $RED_W[\overrightarrow{R}/\overrightarrow{X}, S/Y]$, so we can conclude that $t \in RED_T[\overrightarrow{R}/\overrightarrow{X}]$.

We shall need now one more lemma to deduce reducibility of a term from reducibility of its subterms.

Lemma 17. $RED_{T[V/Y]}[\overrightarrow{R}/\overrightarrow{X}] = RED_T[\overrightarrow{R}/\overrightarrow{X}, RED_V[\overrightarrow{R}/\overrightarrow{X}]/Y]$

Proof. By induction on T.

Lemma 18 Universal application.
If $t \in RED_{\forall Y.W}[\overrightarrow{R}/\overrightarrow{X}]$, then $t[V] \in RED_{W[V/Y]}[\overrightarrow{R}/\overrightarrow{X}]$ for every type V.

Proof. By hypothesis, $t[V] \in RED_W[\overrightarrow{R}/\overrightarrow{X}, S/Y]$ for every candidate S of type V. Taking $S = RED_V[\overrightarrow{R}/\overrightarrow{X}]$, the result follows by lemma 17.

The reducibility theorem

As in [GLT90], we say here that a term t of type T is *reducible* if it is in $RED_T[\overrightarrow{SN}/\overrightarrow{X}]$, where $\overrightarrow{X}$ are the free type variables of T and SN_i is the set of strongly normalizable terms of type X_i.
In the proof of the theorem, there is the need of a stronger induction hypothesis, from which the strong normalization follows by putting $u_i = x_i$ and $R_i = SN_i$.

Proposition 19. *Let $t : T$ be any term, whose free variables are contained in $x_1 : U_1, \ldots, x_n : U_n$, and all the free variable of T, U_1, $\ldots U_n$ are among X_1, $\ldots X_m$. If R_1, $\ldots R_m$ are reducibility candidates of types V_1, $\ldots V_m$, and u_1, $\ldots u_m$ are terms of types $U_1[\overrightarrow{V}/\overrightarrow{X}]$, $\ldots U_m[\overrightarrow{V}/\overrightarrow{X}]$ which are in $RED_{U_1}[\overrightarrow{R}/\overrightarrow{X}]$, $\ldots RED_{U_n}[\overrightarrow{R}/\overrightarrow{X}]$, then $t[\overrightarrow{V}/\overrightarrow{X}][\overrightarrow{u}/\overrightarrow{x}] \in RED_T[\overrightarrow{R}/\overrightarrow{X}]$.*

Proof. By induction on t as usual.

Theorem 20. $\xrightarrow{\beta^2\eta^2\pi*}$ *is strongly normalizing.*

Proof. Let t be any term, and U be its type. All its free variables are in any reducibility candidate by remark 4.1, so that $t = t[\overrightarrow{X}/\overrightarrow{X}][x_1/x_1, \ldots, x_n/x_n] \in RED_{U[\overrightarrow{X}/\overrightarrow{X}]}[\overrightarrow{SN}/\overrightarrow{X}]$ by the previous lemma. Then it is strongly normalizing by CR1.

5 Adding the Top type *modularly*

It is possible to extend the previous reducibility proof to take into account also the special *unit* type T and the Top rule

$$M : T \longrightarrow * : T \qquad (\text{if } M \not\equiv *)$$

that one needs for various systems (see [CDC91], for a discussion of the motivation for the Top rule), but that would not be a wise approach to this rule: indeed, its great simplicity will allow to add it modularly to a given canonical system, without any need to go again through a long proof involving the rest of the calculus.

Without going into much formal details, we want to give here a very simple argument of general applicability that shows how Top preserves strong normalization of left linear reduction relations, and hence also confluence under some additional assumptions. We are quite informal here about the meaning of left linearity for a reduction relation, but this notion can be formalized properly once one choses an appropriate framework for describing the reduction relation (like for example the Combnatory Reduction Systems).

Proposition 21 Preservation of Strong Normalization with Top. *Given a strongly normalizing left-linear reduction relation R, generated by rules that do not contain the special term $*$ in their left hand side, then $R + Top$ is still strongly normalizing.*

Proof. The argument of the proof is indeed very simple: take any infinite reduction Π in $R + Top$, and notice that (since Top alone is clearly strongly normalizing) this reduction must contain infinite R steps. Now, one can easily build an infinite reduction in R alone by using the infinite number of R steps from the Π reduction: it suffices to remark that any Top reduction step followed by an R reduction step in Π $C[M] \xrightarrow{Top} C[*] \xrightarrow{R} M'$ can be postponed as in $C[M] \xrightarrow{R} M'' \xrightarrow{Top}{}^{*} M'$, as the rules on R are left-linear and do not transform $*$ (basically, rules in R can only pass $*$ around untouched, or delete it).

Clearly, if R is also confluent, and Top does not break local confluence, then, as an easy corollary of Newman's Lemma, $R + Top$ is also confluent.

It is then quite immediate to derive the following

Corollary 22. *The second order lambda calculus with expansive rules for products, arrow and universal types, and the Top rule is strongly normalizing and confluent.*

6 Conclusions and future work

We explained why expansion rules and the second order β rule cannot be handled separately in a modular way and we showed how to handle them together by a careful restructuration of the usual reducibility candidate method. This work suggests that the expansionary approach to extensional rules is viable even in a second order context, and is a firm step towards a redesign of the extensional rules in more complex systems like Fω or the Calculus of Constructions. Expansion rules behave better than contractions in the presence of additional axioms (like the one for *unit* or for a fixpoint combinator) and give a natural characterization of the notion of η-long normal forms.

Finally, we would like to suggest that the use of expansion rules for extensional equalities is very promising in order to improve the modularity results on the combination of the lambda calculus with first or higher order algebraic rewriting systems (presented for example in [BTM86, BT88, BTG94] and [JO91]): these works use the traditional contractive interpretation of η, and cannot handle higher order rewrite rules like the one given above for the *unit* type, while such rule is not problematic at all in the presence of expansion rules.

Acknowledgements

We are endebted to Neil Ghani for the fundamental remark that we only need the set of strongly normalizing terms of basic types for the proof to go through.

References

[Aka93] Yohji Akama. On Mints' reductions for ccc-Calculus. In *Typed Lambda Calculus and Applications*, number 664 in LNCS, pages 1–12. Springer Verlag, 1993.

[Bar84] Henk Barendregt. *The Lambda Calculus; Its syntax and Semantics (revised edition)*. North Holland, 1984.

[BT88] Val Breazu-Tannen. Combining algebra and higher-order types. In *Proceedings, Third Annual Symposium on Logic in Computer Science*, pages 82–90, Edinburgh, Scotland, July 5–8 1988. IEEE Computer Society.

[BTG94] Val Breazu-Tannen and Jean Gallier. Polymorphic rewiting preserves algebraic confluence. *Information and Computation*, 114:1–29, 1994.

[BTM86] Val Breazu-Tannen and Albert R. Meyer. Polymorphism is conservative over simple types (preliminary report). In *Proceedings, Symposium on Logic in Computer Science*, pages 7–17, Cambridge, Massachusetts, June 16–18 1986. IEEE Computer Society.

[CDC91] Pierre-Louis Curien and Roberto Di Cosmo. A confluent reduction system for the λ-calculus with surjective pairing and terminal object. In Leach, Monien, and Artalejo, editors, *Intern. Conf. on Automata, Languages and Programming (ICALP)*, volume 510 of *Lecture Notes in Computer Science*, pages 291–302. Springer-Verlag, July 1991.

[Cub92] Djordje Cubric. On free CCC. Distributed on the `types` mailing list, 1992.

[DCK94a] Roberto Di Cosmo and Delia Kesner. Combining first order algebraic rewriting systems, recursion and extensional lambda calculi. In Serge Abiteboul and Eli Shamir, editors, *Intern. Conf. on Automata, Languages and Programming (ICALP)*, volume 820 of *Lecture Notes in Computer Science*, pages 462–472. Springer-Verlag, July 1994.

[DCK94b] Roberto Di Cosmo and Delia Kesner. Simulating expansions without expansions. *Mathematical Structures in Computer Science*, 4:1–48, 1994. A preliminary version is available as Technical Report LIENS-93-11/INRIA 1911.

[DCP95] Roberto Di Cosmo and Adolfo Piperno. Expanding extensional polymorphism. In Mariangiola Dezani-Ciancaglini and Gordon Plotkin, editors, *Typed Lambda Calculus and Applications*, volume 902 of *Lecture Notes in Computer Science*, pages 139–153, April 1995.

[Dou93] Daniel J. Dougherty. Some lambda calculi with categorical sums and products. In *Proc. of the Fifth International Conference on Rewriting Techniques and Applications (RTA)*, 1993.

[Ges90] Alfons Geser. *Relative termination.* Dissertation, Fakultät für Mathematik und Informatik, Universität Passau, Germany, 1990. Also available as: Report 91-03, Ulmer Informatik-Berichte, Universität Ulm, 1991.

[Geu92] Herman Geuvers. The church-rosser property for $\beta\eta$-reduction in typed λ-calculi. In *7thProceedings of the Symposium on Logic in Computer Science (LICS)*, pages 453–460, 1992.

[Gha95] Neil Ghani. Extensionality and polymorphism. University of Edimburgh, Submitted, 1995.

[GLT90] Jean-Yves Girard, Yves Lafont, and Paul Taylor. *Proofs and Types.* Cambridge University Press, 1990.

[Jay92] Colin Barry Jay. Long $\beta\eta$ normal forms and confluence (revised). Technical Report 44, LFCS - University of Edinburgh, August 1992.

[JG92] Colin Barry Jay and Neil Ghani. The Virtues of Eta-expansion. Technical Report ECS-LFCS-92-243, LFCS, 1992. University of Edimburgh, preliminary version of [JG95].

[JG95] Colin Barry Jay and Neil Ghani. The Virtues of Eta-expansion. *Journal of Functional Programming*, 5(2):135–154, April 1995.

[JO91] Jean-Pierre Jouannaud and Mitsuhiro Okada. A computation model for executable higher-order algebraic specification languages. In *Proceedings, Sixth Annual IEEE Symposium on Logic in Computer Science*, pages 350–361, Amsterdam, The Netherlands, July 15–18 1991. IEEE Computer Society Press.

[Klo80] Jan Willem Klop. Combinatory reduction systems. *Mathematical Center Tracts*, 27, 1980.

[Nip90] Tobias Nipkow. A critical pair lemma for higher-order rewrite systems and its application to λ^*. *First Annual Workshop on Logical Frameworks*, 1990.

A Strong normalization

Theorem 12 *If R_1 and R_2 are reducibility candidates of types U_1 and U_2, then $R_1 \times R_2$ is a reducibility candidate of type $U_1 \times U_2$.*

Proof. Assume that R_1 and R_2 are reducibility candidates of type U_1 and U_2, respectively.

– (CR1) if $t \in R_1 \times R_2$, then $\pi_i(t)$ is strongly normalisable by the hypothesis on R_i, since $\pi_i(t) \in R_i$ by definition. Hence t is strongly normalisable by lemma 10.

- (CR2) if $t \longrightarrow t'$ not via a basic expansion, then $\pi_1(t) \longrightarrow \pi_1(t')$ and $\pi_2(t) \longrightarrow \pi_2(t')$.
  As $t \in R_1 \times R_2$, then $\pi_1(t) \in R_1$ and $\pi_2(t) \in R_2$. By hypothesis CR2 for R_1 and R_2 we get $\pi_1(t') \in R_1$ and $\pi_2(t') \in R_2$, hence, by definition, $t' \in R_1 \times R_2$.
  In the case $t \longrightarrow \langle \pi_1(t), \pi_2(t) \rangle$, we must prove that $\pi_i \langle \pi_1(t), \pi_2(t) \rangle \in R_i$. By definition of $R_1 \times R_2$, $\pi_1(t)$ is in R_1 and $\pi_2(t)$ is in R_2, so we can apply lemma 11.
- (CR3) t is neutral and every t' one step from t other than basic expansions are in $R_1 \times R_2$.
  We need to show $\pi_1(t) \in R_1$ and $\pi_2(t) \in R_2$.
  Now notice that applying a one step reduction to $\pi_i(t)$ which is not a basic expansion can only result in some $\pi_i(t')$ as t is neutral. Furthermore, this t' is not a basic expansion of t because it is in an influential position. So, t' is in $R_1 \times R_2$, and then by definition $\pi_1(t') \in R_1$ and $\pi_2(t') \in R_2$. Since the $\pi_i(t)$ are neutral and every term one step from them other than basic expansions is in R_i, the hypothesis for R_1 and R_2 ensure $\pi_1(t) \in R_1$ and $\pi_2(t) \in R_2$. So $t \in R_1 \times R_2$ by definition.