

On the confluence of lambda-calculus with conditional rewriting

Frédéric Blanqui (INRIA) * Claude Kirchner (INRIA) †
Colin Riba (LIP - ENS Lyon) ‡

The confluence of untyped λ -calculus with *unconditional* rewriting is now well understood. In this paper, we investigate the confluence of λ -calculus with *conditional* rewriting and provide general results in two directions.

First, when conditional rules are algebraic. This extends results of Müller and Dougherty for unconditional rewriting. Two cases are considered, whether beta-reduction is allowed or not in the evaluation of conditions. Moreover, Dougherty's result is improved from the assumption of strongly normalizing β -reduction to weakly normalizing β -reduction. We also provide examples showing that outside these conditions, modularity of confluence is difficult to achieve.

Second, we go beyond the algebraic framework and get new confluence results using a restricted notion of orthogonality that takes advantage of the conditional part of rewrite rules.

*FIT 3-604, Tsinghua University, Haidian District, Beijing 100084, China

†INRIA Bordeaux - Sud-Ouest, Bât. A29, 351 cours de la Libération, 33405 Talence, France

‡UMR 5668 CNRS ENS-Lyon UCBL INRIA, 46 allée d'Italie, 69364 Lyon Cedex 7, France

Contents

1	Introduction	3
2	Lambda-calculus and conditional rewriting	5
2.1	Terms and rewrite relations	5
2.2	Lambda-calculus	8
2.3	Conditional rewriting	9
2.4	Examples	11
2.4.1	Coherence of lambda-calculus with surjective pairing	12
2.4.2	A term manipulation system	12
2.5	Confluence	13
3	Confluence: from unconditional to conditional rewriting	14
3.1	Confluence of beta-reduction with unconditional rewriting	15
3.1.1	Left-linear rewriting	15
3.1.2	Strongly beta-normalizing terms	16
3.2	Orthogonal conditional rewriting	17
3.3	Overview of the results	18
4	Confluence of beta-reduction with conditional rewriting	18
4.1	Confluence for left-linear semi-closed systems	19
4.2	Confluence on weakly beta-normalizing terms	22
5	Using beta-reduction in the evaluation of conditions	26
5.1	Confluence for left-linear semi-closed systems	28
5.1.1	Preliminaries	28
5.1.2	Confluence of beta-reduction with beta-conditional rewriting	34
5.2	Confluence on weakly beta-normalizing terms	35
6	Orthonormal systems	36
7	Conclusion	40

1 Introduction

Rewriting [DJ90] and λ -calculus [Bar84] are two universal computation models which are both used, with their own advantages, in programming languages design and implementation, as well as for the foundation of logical frameworks and proof assistants. Among other things, λ -calculus allows to manipulate abstractions and higher-order variables, while rewriting is traditionally well suited for defining functions over data-types and for dealing with equality.

Starting from Klop's work on higher-order rewriting and because of their complementarity, many frameworks have been designed with a view to integrate these two formalisms. This integration has been handled either by enriching first-order rewriting with higher-order capabilities, by adding to λ -calculus algebraic features or, more recently, by a uniform integration of both paradigms. In the first case, we find the works on combinatory reduction systems [KOR93] and other higher-order rewriting systems [Wol93, Nip91] each of them subsumed by van Oostrom and van Raamsdonk's axiomatization of HORSs [OR94], and by the every expressive framework of CCERSs [GKK05]. The second case concerns the more atomic combination of λ -calculus with term rewriting [JO91, Bla05] and the last category the rewriting calculus [CK01, BCKL03].

Despite this strong interest in the combination of both concepts, few works have considered *conditional* higher-order rewriting with λ -calculus. This is of particular interest for both computation and deduction. Indeed, conditional rewriting appears to be very convenient when programming with rewrite rules and its combination with higher-order features provides a quite agile background for the combination of algebraic and functional programming. This is also of main use in proof assistants based on the Curry-Howard-de Bruijn isomorphism where, as emphasized in *deduction modulo* [DHK03, Bla05], rewriting capabilities for defining functions and proving equalities automatically is clearly of great interest when making large proof developments. Furthermore, while many confluence proofs often rely on termination and local confluence, in some cases, confluence may be necessary for proving termination (e.g. with type-level rewriting or strong elimination [Bla05]). It is therefore of crucial interest to have also criteria for the preservation of confluence when combining conditional rewriting and β -reduction without assuming the termination of the combined relation. In particular, assuming the termination of just one of the two relations is already of interest.

The earliest work on preservation of confluence when combining typed λ -calculus and first-order rewriting concerns the simple type discipline [BT88] and the result has been extended to polymorphic λ -calculus in [BTG94]. Concerning untyped λ -calculus, the result was shown in [Mül92] for left-linear rewriting. It is extended as a modularity result for higher-order rewriting in [OR94]. In [Dou92], it is shown that left-linearity is not necessary, provided that terms considered are strongly β -normalizable and are well-formed with respect to the declared arity of symbols, a property that we call here *arity compliance*. Higher-order conditional rewriting is studied in [ALS94] and the confluence result relies on joinability of critical pairs, hence on termination of the combined rewrite relation. An approach closer to ours is taken with a form of conditional λ -calculus in [Tak93], and with CCERSs in [GKK05]. In both cases, confluence relies on a form of conditional orthogonality. However, in these works, conditions are abstract predicates on terms, and confluence is achieved by assuming that the satisfaction of these predicates is preserved by reduction. These results do not directly apply in our case, since proving that the satisfaction of conditions is preserved by reduction is actually the most difficult task for confluence, and this requires a precise knowledge of the shape of the conditions. These systems are related to those presented in Section 6. Our results can rather be seen as a form of modularity properties. Concerning confluence of unconditional term rewriting, the early work of [Toy87] has been extended to the higher-order case in [OR94]. In the case of conditional rewriting, if

modularity properties have been investigated in the pure first-order setting (e.g. [Mid91, Gra96]), to the best of our knowledge, there was up to now no result on the preservation of confluence for the *combination with β -reduction*.

In this paper, we study the confluence property of the combination of β -reduction with a confluent conditional rewrite system. This of course should rely on a clear understanding of the conditional rewrite relation under use and, as usual, the way matching is performed and conditions are checked is crucial. We always consider left-hand sides without abstractions. So, rewriting is not in need of higher-order pattern-matching but just relies on syntactic matching.

We begin in Section 2 by presenting our notations and some basic facts on λ -calculus and conditional rewriting. We start from λ -calculus and discuss, via Böhm's theorem, the need of enriching its syntax with symbols defined by rewrite rules. We then present the different kinds of conditional rewriting considered in this paper. We are interested in *join* conditional rewriting: the conditions of rewrite rules are evaluated by testing the *joinability* of terms. Given a conditional rewrite system, we consider two conditional rewrite relations, whether β -reduction is allowed or not in the evaluation of conditions. The case where β -reduction is allowed in the conditions is termed *β -conditional* rewriting. We also discuss the particular case of *normal* rewriting, i.e. when one side of the conditions is made of terms in normal form. We then give two examples of conditional rewrite system. The first one recalls the use of conditional rewriting in the study of λ -calculus with surjective pairing [Vri89]. The second one is a term manipulation system inspired from a program of [Hue86]. We conclude this section by some basic material on confluence.

In Section 3 we state precisely the known results from which this paper starts and give a short overview of our results. The general goal of this paper is to give sufficient conditions for the confluence of β -reduction with β -conditional rewriting (i.e. with β -steps allowed in the evaluation of conditions). Our main objective is the *preservation* of confluence, that is, given a conditional rewrite system, to get confluence β -conditional rewriting combined with β -reduction assuming the confluence of conditional rewriting. Our approach is to generalize known results on the combination of β -reduction with unconditional rewriting. We present in Section 3.1 the two different cases we start with: Müller's result [Mül92] for left-linear rewriting, and Dougherty's result [Dou92] for algebraic rewriting on strongly β -normalizing terms respecting some arity conditions (called *arity compliance*). In each case, we will first consider the case of β -reduction with conditional rewriting (when β -reduction is not allowed in the evaluation of conditions) and then extend these results to β -conditional rewriting. However, Example 5.2 shows that for β -conditional rewriting, we can not go beyond algebraic rewriting with arity conditions. In order to handle rewrite rules which can contain active variables and abstractions in right-hand sides or in conditions, we build on orthogonal conditional rewriting. Known results on the confluence of orthogonal for normal algebraic conditional rewriting are discussed in Section 3.2. We conclude this section by an informal overview of our results. They are summarized in Figure 1, page 6.

The last three sections contain the technical contributions of the paper. We begin in Section 4 by extending Müller's and Dougherty's result to conditional rewriting combined with β -reduction. Müller's result [Mül92] assumes the left-linearity of rewrite rules. Of course, with conditional rewriting, non-linearity can be simulated by linear systems. Extending the result of Müller [Mül92], we prove in Section 4.1 that the confluence of conditional rewriting combined with β -reduction follows from the confluence of conditional rewriting when conditional rules are applicative, left-linear and semi-closed, which means that the conditions of rules cannot test for equality of open terms. In Section 4.2 we adapt Dougherty's method [Dou92] to conditional rewriting and extend it to show that for a large set of *weakly* β -normalizing terms, the left-linearity and semi-closure hypotheses can be dropped provided that rules are algebraic and

terms are arity compliant.

We then turn in Section 5 to the confluence of β -conditional rewriting combined with β -reduction. We show in Example 5.2 that confluence is in general not preserved with non-algebraic rules. When rules are algebraic, we show that arity compliance is a sufficient condition to deduce the confluence of β -conditional rewriting combined with β -reduction from the confluence of conditional rewriting alone. This is done first for left-linear semi-closed systems in Section 5.1, a restriction that we also show to be superfluous when considering only *weakly* β -normalizing terms (Section 5.2).

The case of non-algebraic rules is handled in Section 6. Such rules can contain active variables and abstractions in right-hand sides or in conditions (but still not in left-hand sides). In this case, the confluence of β -conditional rewriting combined with β -reduction does not follow anymore from the confluence of conditional rewriting. We show that confluence holds under a syntactic condition, called *orthonormality*, ensuring that if two rules overlap at a non-variable position, then their conditions cannot be both satisfied. An orthonormal system is therefore an orthogonal system whose orthogonality follows from the confluence of the rewrite relation (recall that with conditional rewriting critical pairs contain conditions ; hence orthogonality depends on the rewrite relation since it depends on the satisfiability of these conditions).

This paper is an extended version of [BKR06]. We assume familiarity with λ -calculus [Bar84] and conditional rewriting [DO90, Ohl02]. We recall the main notions in the next section.

2 Lambda-calculus and conditional rewriting

In this section we present the tools used in this paper and recall some well-known facts.

2.1 Terms and rewrite relations

We consider λ -terms with curried function symbols. Among them we distinguish *applicative terms* that do not contain abstractions, and *algebraic terms* that are applicative terms with no variable in active position.

Definition 2.1 (Terms) *Let Σ be a set of function symbols and \mathcal{X} be a set of variables.*

(i) *The set $\Lambda(\Sigma)$ of λ -terms is defined by the grammar*

$$t, u \in \Lambda(\Sigma) ::= x \mid \lambda x.t \mid tu \mid f,$$

where $x \in \mathcal{X}$ and $f \in \Sigma$. We denote by Λ the set $\Lambda(\emptyset)$ of pure λ -terms.

(ii) *The set of applicative terms is defined by the grammar*

$$t, u ::= x \mid tu \mid f.$$

(iii) *The set of algebraic terms is defined by the grammar*

$$t ::= x \mid f t_1 \dots t_n.$$

As usual, λ -terms are considered equal modulo α -conversion. We denote by $FV(t)$ the set of variables occurring free in the term t . A term is *closed* if it has no free variables and *open* otherwise, it is *linear* if each of its free variables occurs at most once. Given $h \in \mathcal{X} \cup \Sigma$, we write $h\vec{t}$ for $h t_1 \dots t_n$ and let $|\vec{t}| =_{\text{def}} n$. Similarly, we write $\lambda\vec{x}.t$ for $\lambda x_1 \dots \lambda x_n.t$.

§	Left-Hand Sides	Right-Hand Sides	Conditions	Terms	Result
4.1	Algebraic & Linear	Applicative	No equality tests between open terms (Semi-closed, Def. 4.2)	All $(\Lambda(\Sigma))$	$\rightarrow_{\mathcal{R}}$ Confluent $\implies \rightarrow_{\beta \cup \mathcal{R}}$ Confluent (Thm. 4.6)
4.2	Algebraic & Respect an arity \mathbf{a} (Def. 4.8)	Algebraic & Respect the arity \mathbf{a}	Algebraic & Respect the arity \mathbf{a}	Weakly β -normalizing & β nf respect the arity \mathbf{a} ($\mathcal{AN}_{\mathbf{a}}$, Def. 4.9)	$\rightarrow_{\mathcal{R}}$ Confluent $\implies \rightarrow_{\beta \cup \mathcal{R}}$ Confluent (Thm. 4.15)
5.1	Algebraic & Linear & Respect an arity \mathbf{a} (Def. 4.8)	Algebraic & Respect the arity \mathbf{a}	Algebraic & Respect the arity \mathbf{a} & No equality tests between open terms (Semi-closed, Def. 4.2)	Respect the arity \mathbf{a} (Conditionally $(\mathcal{R}, \mathbf{a})$ -stable, Def. 5.5)	$\rightarrow_{\mathcal{R}}$ Confluent $\implies \rightarrow_{\beta \cup \mathcal{R}(\beta)}$ Confluent (Thm. 5.10)
5.2	Algebraic & Respect an arity \mathbf{a} (Def. 4.8)	Algebraic & Respect the arity \mathbf{a}	Algebraic & Respect the arity \mathbf{a}	Weakly β -normalizing & β nf respect the arity \mathbf{a} ($\mathcal{AN}_{\mathbf{a}}$, Def. 4.9)	$\rightarrow_{\mathcal{R}}$ Confluent $\implies \rightarrow_{\beta \cup \mathcal{R}(\beta)}$ Confluent (Thm. 5.12)
6	Algebraic & Linear		Orthonormal (Def. 6.2)	All $(\Lambda(\Sigma))$	$\rightarrow_{\beta \cup \mathcal{R}(\beta)}$ Shallow Confluent (Thm. 6.7)

Figure 1: Overview of the results. Algebraic and applicative terms are defined in Def. 2.1

Example 2.2 *Intuitively, an algebraic term is a curried first-order term with no arity constraint on symbols. For instance the terms filter and $\text{filter } p \times l$ are algebraic, as well as $\text{filter } p \times l \ y \ z$. An applicative term is an algebraic term which may contain variables in head position, such as $x \text{ filter}$. The λ -term $\lambda x.x$ is not applicative (and thus not algebraic).*

A lot of proofs of this paper are made by induction on the structure of λ -terms. However, it is often not convenient to reason directly on their syntax as given by the productions of $\Lambda(\Sigma)$. For instance, knowing that a term t is an application, say $t = u \ v$, gives little information on its behavior: we do not know whether u is an abstraction, in which case t is a β -redex, or whether it is an algebraic term, in which case t may be the instantiated left-hand side of a rewrite rule. It is therefore useful to have an induction principle on λ -terms which makes apparent more informations on their structure. This is provided by the following well-known lemma, due to Wadsworth [Wad71].

Lemma 2.3 ([Wad71]) *Any λ -term $t \in \Lambda(\Sigma)$ can be uniquely written in one of the following forms:*

$$\begin{aligned} & \lambda x_1 \dots \lambda x_m. v \ a_1 \dots a_n & (a) \\ \text{or} & \lambda x_1 \dots \lambda x_m. (\lambda y. b) \ a_0 \ a_1 \dots a_n & (b) \end{aligned}$$

where $n, m \geq 0$ and $v \in \mathcal{X} \cup \Sigma$.

A *substitution* is a map $\sigma : \mathcal{X} \rightarrow \Lambda(\Sigma)$ of finite domain. We denote by $t\sigma$ the capture-avoiding application of the substitution σ to the term t . If σ is the substitution which maps x_i to u_i for all $i \in \{1, \dots, n\}$, then we may write $t[u_1/x_1, \dots, u_n/x_n]$ instead of $t\sigma$.

Definition 2.4 (Rewrite Relations) *A rewrite relation is a binary relation \rightarrow on $\Lambda(\Sigma)$ closed under the following rules, where σ is a substitution:*

$$(\text{ABS}) \frac{t \rightarrow u}{\lambda x. t \rightarrow \lambda x. u} \quad (\text{APPL}) \frac{t \rightarrow u}{t v \rightarrow u v} \quad (\text{APPR}) \frac{t \rightarrow u}{v t \rightarrow v u} \quad (\text{SUBST}) \frac{t \rightarrow u}{t\sigma \rightarrow u\sigma}$$

We denote by \rightarrow^+ the transitive closure of \rightarrow , by $\rightarrow^=$ its reflexive closure, by \rightarrow^* its reflexive and transitive closure, by \leftarrow its inverse and by \leftrightarrow its reflexive symmetric and transitive closure. We write $t \downarrow u$ if there exists v such that $t \rightarrow^* v \leftarrow^* u$ and $t \rightarrow^k u$ if $t \rightarrow^* u$ in at most k steps.

Given two rewrite relations \rightarrow_A and \rightarrow_B , we let $\rightarrow_{A \cup B} =_{\text{def}} \rightarrow_A \cup \rightarrow_B$. We say that a term t is an *A-normal form* if there is no u such that $t \rightarrow_A u$. We let \mathcal{SN}_A , *the set of strongly A-normalizing terms*, be the set of terms on which the relation \rightarrow_A is well-founded and we let \mathcal{WN}_A , *the set of weakly A-normalizing terms*, be the set of terms which rewrite to an A-normal form.

Rewrite relations \rightarrow satisfy the following property: for all $t, u, v \in \Lambda(\Sigma)$,

$$\text{if } t \rightarrow u \quad \text{then } v[t/x] \rightarrow^* v[u/x].$$

In the following, we will often use a stronger property: for all $t, u, v \in \Lambda(\Sigma)$,

$$\text{if } t \rightarrow u \quad \text{then } v[t/x] \rightarrow v[u/x].$$

This is in general false with rewrite relations, but this holds with *parallel* rewrite relations.

Definition 2.5 (Parallel Rewrite Relations) *A parallel rewrite relation is a rewrite relation \triangleright closed under the rules*

$$(\triangleright \text{VAR}) \frac{}{x \triangleright x} \quad (\triangleright \text{SYMB}) \frac{}{f \triangleright f} \quad (\triangleright \text{APP}) \frac{t_1 \triangleright u_1 \quad t_2 \triangleright u_2}{t_1 t_2 \triangleright u_1 u_2}$$

Note that given a parallel rewrite relation \triangleright , we have $\lambda x.t \triangleright \lambda x.u$ if $t \triangleright u$, since by definition parallel rewrite relations are rewrite relations.

Given a rewrite relation \rightarrow and two substitutions σ and σ' , we write $\sigma \rightarrow \sigma'$ if σ and σ' have the same domain and $\sigma(x) \rightarrow \sigma'(x)$ for all $x \in \text{Dom}(\sigma)$.

Proposition 2.6 *If \triangleright is a parallel rewrite relation on $\Lambda(\Sigma)$ then $\sigma \triangleright \sigma'$ implies $v\sigma \triangleright v\sigma'$.*

PROOF. By induction on v .

$v \in \mathcal{X} \cup \Sigma$. If $v = x \in \text{Dom}(\sigma)$ then $v\sigma = \sigma(x) \triangleright \sigma'(x) = v\sigma'$. Otherwise, $v\sigma = v \triangleright v = v\sigma'$ thanks to the rules (\triangleright VAR) and (\triangleright SYMB).

$v = v_1 v_2$. By induction hypothesis we have $v_i\sigma \triangleright v_i\sigma'$ for all $i \in \{1, 2\}$, and we conclude by the rule (\triangleright APP).

$v = \lambda x.v_1$. By induction hypothesis. □

In particular, if $\text{Dom}(\sigma) \cap \text{FV}(v) = \emptyset$ then $v \triangleright v$: parallel rewrite relations are reflexive.

2.2 Lambda-calculus

λ -calculus is characterized by β -reduction. This is the smallest rewrite relation \rightarrow_β on $\Lambda(\Sigma)$ such that

$$(\lambda x.t)u \rightarrow_\beta t[u/x].$$

In order to understand our motivations for studying the combination of λ -calculus with (conditional) rewriting, let us recall some facts about *pure* λ -calculus. It is well-known that integers can be coded within pure λ -calculus. An example of such coding is that of *Church's numerals*. The **Zero** and **Succ** functions are represented by the following terms:

$$\text{Zero} \stackrel{\text{def}}{=} \lambda x.\lambda f.x \quad \text{and} \quad \text{Succ} \stackrel{\text{def}}{=} \lambda n.\lambda x.\lambda f.f(n x f).$$

We can code *iteration* with the term $\text{Iter } x y z \stackrel{\text{def}}{=} z x y$, and for all $n, u, v \in \Lambda$ we have

$$\begin{aligned} \text{Iter } u v \text{ Zero} &= (\lambda x f.x) u v && \xrightarrow{\beta} u \\ \text{Iter } u v (\text{Succ } n) &= (\lambda x f.f(n x f)) u v && \xrightarrow{\beta} v(n u v) = v(\text{Iter } u v n). \end{aligned}$$

However, *recursion* cannot be implemented in constant time (see for instance [Par89]): there is no term $\text{Rec } x y z$ such that there is $k \in \mathbb{N}$ such that for all $u, v, n \in \Lambda$,

$$\text{Rec } u v \text{ Zero} \xrightarrow{\beta}^k u \quad \text{and} \quad \text{Rec } u v (\text{Succ } n) \xrightarrow{\beta}^k v(\text{Rec } u v n) n.$$

In particular, there is no coding of the predecessor function for Church's numerals in constant time. This suggests that practical utilizations of the λ -calculus may require extensions of β -reduction. At this point it is interesting to recall Böhm's theorem. It states that any proper extension of $\beta\eta$ -conversion on the set of weakly β -normalizing pure λ -terms is inconsistent.

Theorem 2.7 (Böhm [Böh68]) *Let \rightarrow_η be the smallest rewrite relation on Λ such that $\lambda x.t x \rightarrow_\eta t$ if $x \notin \text{FV}(t)$. If \simeq is an equivalence relation on Λ which is stable by contexts, contains $\leftrightarrow_{\beta\eta}$ and such that $\simeq \setminus \leftrightarrow_{\beta\eta}$ contains a pair of weakly β -normalizing terms, then for all $t, u \in \Lambda$ we have $t \simeq u$.*

This theorem suggests to find extensions of β -reduction operating on extensions of the set of pure λ -terms Λ . A possibility, that we consider in this paper, is to work with *function symbols* $f \in \Sigma$ defined by *rewrite rules*.

2.3 Conditional rewriting

In this paper, we are interested in *conditional* rewriting. The following example introduces the main ideas. Consider lists built from the empty list `nil` and the constructor `cons`. We use the symbols `true` and `false` to represent the boolean values "true" and "false". We would like to define, via rewriting, a function `filter` such that

- `filter p nil` rewrites to `nil`,
- `filter p (cons t ts)` rewrites to `cons t (filter p ts)` if `p t` rewrites to `true`, and
- `filter p (cons t ts)` rewrites to `filter p ts` if `p t` rewrites to `false`.

This specification can be written using *conditional rewrite rules* (\supset reads *implies*):

$$\begin{array}{l} \text{filter } p \text{ nil} \quad \mapsto \quad \text{nil} \\ p \ x = \text{true} \quad \supset \quad \text{filter } p \text{ (cons } x \text{ xs)} \mapsto \text{cons } x \text{ (filter } p \text{ xs)} \\ p \ x = \text{false} \quad \supset \quad \text{filter } p \text{ (cons } x \text{ xs)} \mapsto \text{filter } p \text{ xs} \end{array} \quad (1)$$

If we try to define a rewrite relation \rightarrow that corresponds to our specification, we get that

$$\text{filter } p \text{ (cons } t \text{ ts)} \rightarrow \text{cons } t \text{ (filter } p \text{ ts)} \quad \text{if} \quad p \ t \rightarrow^* \text{true} . \quad (2)$$

In other words, to define \rightarrow in the step

$$\text{filter } p \text{ (cons } t \text{ ts)} \rightarrow \text{cons } t \text{ (filter } p \text{ ts)} ,$$

we need to test if $p \ t \rightarrow^* \text{true}$, hence to use the relation \rightarrow . This circularity can be broken off by using an inductive definition of conditional rewriting: the relation \rightarrow is stratified in relations $(\rightarrow_i)_{i \in \mathbb{N}}$. The correctness of the definition is ensured by Tarski's fixpoint theorem, which can be applied because, when replacing the symbol $=$ by \rightarrow^* in (1), the obtained formula is *positive* in \rightarrow (it is in fact a Horn clause).

We now turn to formal definitions.

Definition 2.8 (Conditional Rewrite Rules) *A conditional rewrite rule is an expression of the form*

$$d_1 = c_1 \wedge \dots \wedge d_n = c_n \supset l \mapsto r$$

where $d_1, \dots, d_n, c_1, \dots, c_n, l, r \in \Lambda(\Sigma)$ and

- (i) every variable of \vec{d}, \vec{c}, r occurs also in l ,
- (ii) l is an algebraic term which is not a variable.

In conditional rewrite rules, we distinguish

- the left-hand side l , the right-hand side r ;
- the conditions $d_1 = c_1 \wedge \dots \wedge d_n = c_n$.

A rule $d_1 = c_1 \wedge \dots \wedge d_n = c_n \supset l \mapsto r$ is unconditional if $n = 0$. It is left-linear if l is linear.

Since left-hand sides are algebraic terms, rewriting is performed using syntactical first-order matching. Note that the conditions of rewrite rules are not symmetric: the condition $\vec{d} = \vec{c}$ is not the same as $\vec{c} = \vec{d}$.

Given a set \mathcal{R} of conditional rewrite rules, different conditional rewrite relations can be defined, depending on the evaluation of the conditions: by conversion, by joinability or by reduction. This leads respectively to semi-equational, join and oriented conditional rewriting. In this paper, we focus on join conditional rewriting, and call it simply *conditional rewriting*. We also consider the case of join condition rewriting with β -reduction allowed in the evaluation of conditions, and call it *β -conditional rewriting*.

Definition 2.9 (Conditional Rewriting) *Let \mathcal{R} be a set of conditional rewrite rules.*

— *The conditional rewrite relation $\rightarrow_{\mathcal{R}}$ is defined as*

$$\rightarrow_{\mathcal{R}} =_{\text{def}} \bigcup_{i \in \mathbb{N}} \rightarrow_{\mathcal{R}_i} ,$$

where $\rightarrow_{\mathcal{R}_0} =_{\text{def}} \emptyset$ and for all $i \in \mathbb{N}$, $\rightarrow_{\mathcal{R}_{i+1}}$ is the smallest rewrite relation such that for every rule $\vec{d} = \vec{c} \supset l \mapsto_{\mathcal{R}} r$ and every substitution σ ,

$$\text{if } \vec{d}\sigma \downarrow_{\mathcal{R}_i} \vec{c}\sigma \text{ then } l\sigma \rightarrow_{\mathcal{R}_{i+1}} r\sigma .$$

— *The β -conditional rewrite relation $\rightarrow_{\mathcal{R}(\beta)}$ is defined as*

$$\rightarrow_{\mathcal{R}(\beta)} =_{\text{def}} \bigcup_{i \in \mathbb{N}} \rightarrow_{\mathcal{R}(\beta)_i} ,$$

where $\rightarrow_{\mathcal{R}(\beta)_0} =_{\text{def}} \emptyset$ and for all $i \in \mathbb{N}$, $\rightarrow_{\mathcal{R}(\beta)_{i+1}}$ is the smallest rewrite relation such that for every rule $\vec{d} = \vec{c} \supset l \mapsto_{\mathcal{R}} r$ and every substitution σ ,

$$\text{if } \vec{d}\sigma \downarrow_{\beta \cup \mathcal{R}(\beta)_i} \vec{c}\sigma \text{ then } l\sigma \rightarrow_{\mathcal{R}(\beta)_{i+1}} r\sigma .$$

Hence, with conditional rewriting $\rightarrow_{\mathcal{R}}$, β -reduction is not allowed in the evaluation of conditions, while it is allowed with β -conditional rewriting $\rightarrow_{\mathcal{R}(\beta)}$. Note that $\rightarrow_{\mathcal{R}} \subseteq \rightarrow_{\mathcal{R}(\beta)}$. The converse is false, as shown by the following example.

Example 2.10 *Consider the rule*

$$p \ x = \text{true} \ \supset \ \text{filter } p \ (\text{cons } x \ l) \ \mapsto \ \text{cons } x \ (\text{filter } p \ l)$$

issued from the conditional rewrite system (1) and assume that $\text{id } x \mapsto x$. With conditional rewriting we have

$$\text{filter } \text{id} \ (\text{cons } \text{true} \ \text{ts}) \ \rightarrow_{\mathcal{R}} \ \text{cons } \text{true} \ (\text{filter } \text{id} \ \text{ts}) \quad \text{since} \quad \text{id } \text{true} \ \rightarrow_{\mathcal{R}} \ \text{true} .$$

With β -conditional rewriting we also have

$$\text{filter } (\lambda x.x) \ (\text{cons } \text{true} \ \text{ts}) \ \rightarrow_{\mathcal{R}(\beta)} \ \text{cons } \text{true} \ (\text{filter } \lambda x.x \ \text{ts}) \quad \text{since} \quad (\lambda x.x) \ \text{true} \ \rightarrow_{\beta} \ \text{true} ,$$

but the term $\text{filter } (\lambda x.x) \ (\text{cons } \text{true} \ \text{ts})$ is a $\rightarrow_{\mathcal{R}}$ -normal form.

An interesting particular case of join conditional rewriting is *normal rewriting*.

Definition 2.11 (Normal Conditional Rewriting) *Let \mathcal{R} be a conditional rewrite system. If for every rule $\vec{d} = \vec{c} \supset l \mapsto_{\mathcal{R}} r$, the conditions \vec{c} are closed terms in $\rightarrow_{\mathcal{R}}$ -normal form, then we say that $\rightarrow_{\mathcal{R}}$ is a normal conditional rewrite relation.*

In general, for a given conditional system the normal forms w.r.t. join and semi-equational rewriting are not the same (this is a by-product of the fact that semi-equational orthogonal rewriting is confluent, while join orthogonal rewriting is not [BK86, Ohl02], see also Theorem 3.12). The notion of normal conditional rewriting presented in Definition 2.11 is thus specific to join conditional rewriting (it is easy to see that it coincides with normality for oriented rewriting).

An important point with conditional rewriting is the possible undecidability of a rewriting step. This impacts of effectiveness of the notion of normal conditional rewriting.

Remark 2.12 (Decidability) *One-step conditional rewrite relations are in general not decidable. Consider a rule $\vec{d} = \vec{c} \supset l \mapsto r$. Because of the recursive definition of $\rightarrow_{\mathcal{R}}$, to know if $l\sigma \rightarrow_{\mathcal{R}} r\sigma$, we need to reduce the terms $\vec{d}\sigma$ and $\vec{c}\sigma$. This is in general undecidable, even for terminating systems [Kap84] (see also [Ohl02]).*

These facts have consequences on normal rewriting. Given a set of conditional rules, to determine whether it can generate a normal relation, we have to check that a part of the conditions is in normal form. This is in general undecidable, even when the rewrite relation terminates.

We therefore focus on join rewriting because it seems to be a more easily and generally applicable theory than normal rewriting, even if the implementation of conditional rewriting is easier when we already know that the conditional rewrite relation is normal.

Our results on the preservation of confluence impose restrictions on rewrite rules. Some of them concern the terms which can appear in different parts of the rules. This motivates the following definition. Recall from Definition 2.8 that left-hand sides are always assumed to be algebraic.

Definition 2.13 (Applicative and Algebraic Conditional Rewrite Rules) *A conditional rewrite rule $\vec{d} = \vec{c} \supset l \mapsto r$ is*

- right-applicative if r is an applicative term,
- applicative if it is right-applicative and if moreover the terms \vec{d}, \vec{c} are applicative,
- right-algebraic if r is an algebraic term,
- algebraic if it is right-algebraic and if moreover the terms \vec{d}, \vec{c} are algebraic.

A rewrite system \mathcal{R} is right-applicative (resp. applicative, right-algebraic, algebraic) if all its rules are right-applicative (resp. applicative, right-algebraic, algebraic).

In the conditional rewrite system (1), the first rule $\text{filter } p \text{ nil} \mapsto \text{nil}$ is algebraic. The two other rules are right-algebraic. They both use the term $p \ x$ in their conditions, where p is a variable. This term is applicative but not algebraic.

2.4 Examples

We now give some examples of conditional rewrite systems.

2.4.1 Coherence of lambda-calculus with surjective pairing

We begin by recalling one use of conditional rewriting in the study of λ -calculus with surjective pairing. We use $\text{pair } t_1 t_2$ to denote the pairing of t_1 and t_2 . The rewrite rules for binary products are the following:

$$\text{fst } (\text{pair } x_1 x_2) \mapsto_{\pi} x_1 \qquad \text{snd } (\text{pair } x_1 x_2) \mapsto_{\pi} x_2 .$$

It is well-known that the combination of these rules with β -reduction is confluent (see Theorem 3.3, proved in [Mül92]). This follows from the *left-linearity* of the rewrite rules. However, when we add the rule for surjective pairing

$$\text{pair } (\text{fst } x) (\text{snd } x) \mapsto_{\text{SP}} x$$

then the combination of the resulting rewrite relation with β -reduction is not confluent [Klo80]. Note that the rule \mapsto_{SP} is not left-linear: the variable x appears twice in the left-hand side. However, the corresponding conversion is coherent: there are two terms that are not convertible. This has been first shown using semantic methods [Sco75].

In [Vri89], de Vrijer uses semi-equational β -conditional rewriting to give a syntactic proof of the coherence of β -reduction combined with surjective pairing. His rules are those of \mapsto_{π} plus

$$\text{snd } x = y \supset \text{pair } (\text{fst } x) y \mapsto_{\text{lr}} x \qquad \text{fst } x = y \supset \text{pair } y (\text{snd } x) \mapsto_{\text{lr}} x .$$

The resulting relation is confluent modulo an equivalence relation, and this allows de Vrijer to show that λ -calculus plus pairs and surjective pairing is a conservative extension of the pure λ -calculus: for any two *pure* λ -terms $t, u \in \Lambda$,

$$t \leftrightarrow_{\beta} u \quad \text{if and only if} \quad t \leftrightarrow_{\beta \cup \pi \cup \text{SP}} u .$$

2.4.2 A term manipulation system

Our main example is an adaptation of a CAML program of [Hue86]. It defines functions that perform in a term the replacement of the subterm at a given occurrence by another term. Terms are represented by trees whose nodes contain a label and the list of their successor nodes.

This system must be read having in mind the combination of λ -calculus with (join) β -conditional rewriting.

We begin by some basic functions on lists.

$$\begin{array}{ll} \text{car } (\text{cons } x \ l) \mapsto x & \text{cdr } (\text{cons } x \ l) \mapsto l \\ \text{car } \text{nil} \mapsto \text{err} & \text{cdr } \text{nil} \mapsto \text{err} \\ \\ \text{get } l \ \text{zero} \mapsto \text{car } l & \text{length } \text{nil} \mapsto \text{zero} \\ \text{get } l \ (\text{succ } n) \mapsto \text{get } (\text{cdr } l) \ n & \text{length } (\text{cons } x \ l) \mapsto \text{succ } (\text{length } l) \\ \\ & \text{filter } p \ \text{nil} \mapsto \text{nil} \\ p \ x = \text{true} \supset \text{filter } p \ (\text{cons } x \ l) \mapsto \text{cons } x \ (\text{filter } p \ l) \\ p \ x = \text{false} \supset \text{filter } p \ (\text{cons } x \ l) \mapsto \text{filter } p \ l \end{array}$$

Let us define `apply` such that `apply f n l` applies `f` to the `n`th element of `l`. It uses `app` as an

auxiliary function:

$$\begin{aligned}
> (\text{length } l) \ n = \text{true} &\supset \text{apply } f \ n \ l &\mapsto \text{app } f \ n \ l \\
> (\text{length } l) \ n = \text{false} &\supset \text{apply } f \ n \ l &\mapsto \text{err} \\
\text{app } f \ \text{zero } l &&\mapsto \text{cons } (f \ (\text{car } l)) \ (\text{cdr } l) \\
\text{app } f \ (\text{succ } n) \ l &&\mapsto \text{cons } (\text{car } l) \ (\text{app } f \ n \ (\text{cdr } l))
\end{aligned}$$

We represent first-order terms by trees with nodes $\text{node } y \ l$ where y is intended to be a label and l the list of sons. Positions are lists of integers and $\text{occ } u \ t$ tests if u is an occurrence of t . We define it as follows:

$$\begin{aligned}
&\text{occ nil } t &&\mapsto \text{true} \\
> (\text{length } l) \ x = \text{false} &\supset \text{occ } (\text{cons } x \ o) \ (\text{node } y \ l) &\mapsto \text{false} \\
> (\text{length } l) \ x = \text{true} &\supset \text{occ } (\text{cons } x \ o) \ (\text{node } y \ l) &\mapsto \text{occ } o \ (\text{get } l \ x)
\end{aligned}$$

To finish, $\text{replace } t \ o \ s$ replaces by s the subterm of t at occurrence o .

$$\begin{aligned}
\text{occ } u \ t = \text{true} &\supset \text{replace } t \ o \ s &\mapsto \text{rep } t \ o \ s \\
\text{occ } u \ t = \text{false} &\supset \text{replace } t \ o \ s &\mapsto \text{err} \\
\text{rep } t \ \text{nil } s &&\mapsto s \\
\text{rep } (\text{node } y \ l) \ (\text{cons } x \ o) \ s &\mapsto \text{node } y \ (\text{apply } (\lambda z. \text{rep } z \ o \ s) \ x \ l)
\end{aligned}$$

The system *Tree* that consists of the rules defining *car*, *cdr*, *get*, *length* and *occ* is algebraic. The rules of *apply* and *app* are right-applicative and those for *filter* contain in their conditions the variable p in active position. This definition of *rep* involves a λ -abstraction in a right hand side. In Section 6, we prove confluence of the relation $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$ induced by the whole system.

2.5 Confluence

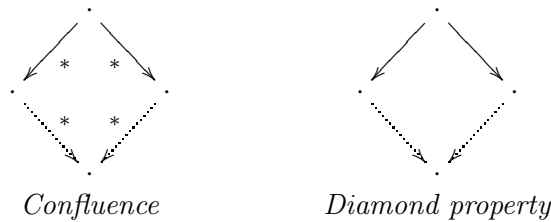
The main property on rewrite relations studied in this paper is *confluence*. The confluence of a relation \rightarrow which has at least two distinct normal forms entails the coherence of the conversion \leftrightarrow . Moreover, it allows to evaluate terms in a modular way: the choice of the subterm to be evaluated first has no impact on the result of the evaluation.

In this section we recall some well-known facts about confluence which will be useful in the following.

A sufficient condition for confluence is the *diamond property*.

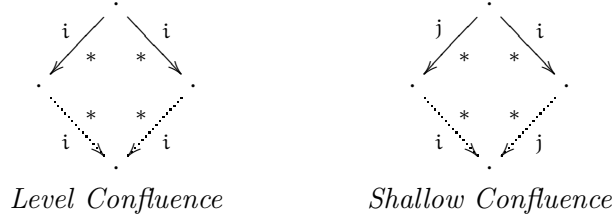
Definition 2.14 (Confluence) *A rewrite relation \rightarrow is confluent if $\leftarrow^* \rightarrow^* \subseteq \rightarrow^* \leftarrow^*$ and has the diamond property if $\leftarrow \rightarrow \subseteq \rightarrow \leftarrow$.*

In diagrammatic form:



The stratification of conditional rewrite relations leads to fine-grained notions of confluence.

Definition 2.15 (Stratified Confluences) Assume that $(\rightarrow_i)_{i \in \mathbb{N}}$ are rewrite relations and let $\rightarrow =_{\text{def}} \bigcup_{i \in \mathbb{N}} \rightarrow_i$. We say that \rightarrow is level confluent if for all $i \geq 0$ we have $\leftarrow_i^* \rightarrow_i^* \subseteq \rightarrow_i^* \leftarrow_i^*$; and shallow confluent if for all $i, j \geq 0$ we have $\leftarrow_i^* \rightarrow_j^* \subseteq \rightarrow_j^* \leftarrow_i^*$.
 In diagrammatic form:



Note that shallow confluence implies level confluence which in turns implies confluence. For instance, in Section 6 we show that $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$ is shallow confluent for some conditional rewrite systems \mathcal{R} called orthonormal. This entails their confluence.

Combinations of rewrite relations. Since we are interested in the confluence of the combination of two rewrite relations (conditional rewriting and λ -calculus), we will use the following notions.

Definition 2.16 (Commutation) A rewrite relation \rightarrow_A commutes with a rewrite relation \rightarrow_B if $\leftarrow_A^* \rightarrow_B^* \subseteq \rightarrow_B^* \leftarrow_A^*$.

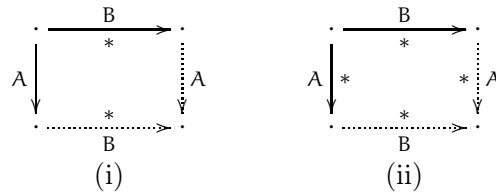
The Hindley-Rosen Lemma is a simple but important tool to prove the confluence of the combination of two rewrite relations.

Lemma 2.17 (Hindley-Rosen) If \rightarrow_A and \rightarrow_B are two confluent rewrite relations that commute then $\rightarrow_{A \cup B}$ is confluent.

The next simple lemma is useful to prove the commutation of two relations.

Lemma 2.18 Let \rightarrow_A and \rightarrow_B be two rewrite relations such that for all $t, u, v \in \Lambda(\Sigma)$, if $u \leftarrow_A t \rightarrow_B v$ then there is a term w such that $u \rightarrow_B^* w \leftarrow_A v$. Then \rightarrow_A commutes with \rightarrow_B .

PROOF. We show (i) by induction on \rightarrow_B^* and deduce (ii) by induction on \rightarrow_A^* .



□

3 Confluence: from unconditional to conditional rewriting

In this section we state precisely the known results from which this paper starts and give a short overview of our results. In Section 3.1 we review the results on the combination of λ -calculus with *unconditional* rewriting that we extend to conditional and β -conditional rewriting in Section 4 and Section 5 respectively. In Section 3.2 we recall a result on the confluence of orthogonal normal rewrite relations, that we generalize to orthonormal β -conditional rewriting in Section 6. We then give a short overview of our results in Section 3.3.

3.1 Confluence of beta-reduction with unconditional rewriting

Our results of Section 4 and 5 on the preservation of confluence for the combination of λ -calculus with (left-algebraic) *conditional* rewriting are extensions of similar results on the combination of λ -calculus with (left-algebraic) *unconditional* rewriting. We concentrate of two cases, both untyped, that we review in this section:

- In Section 3.1.1 we discuss Müller’s result [Mül92] (stated in Theorem 3.3) on left-linear rewriting.
- In Section 3.1.2 we discuss Dougherty’s result [Dou92] (stated in Theorem 3.7) on strongly β -normalizing terms with arity conditions.

3.1.1 Left-linear rewriting

Using the example of surjective pairing [Klo80], we have recalled in Section 2.4.1 that the combination of a confluent non left-linear rewrite system with β -reduction may not be confluent. An example has also been presented in [BTM87], which can be seen as an adaptation of an example due to Huet [Hue80] concerning first-order rewriting.

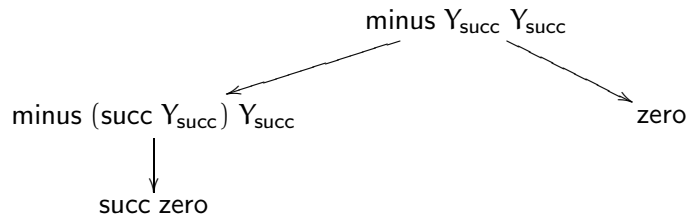
Example 3.1 ([BTM87]) Consider the confluent rewrite system

$$\text{minus } x \ x \ \mapsto_{\text{minus}} \ \text{zero} \qquad \text{minus } (\text{succ } x) \ x \ \mapsto_{\text{minus}} \ (\text{succ } \text{zero}) ,$$

and let

$$Y_{\text{succ}} =_{\text{def}} (\lambda x. \text{succ } (x \ x)) (\lambda x. \text{succ } (x \ x)) .$$

Since $Y_{\text{succ}} \rightarrow_{\beta} \text{succ } Y_{\text{succ}}$, we have the following unjoinable peak:



Remark 3.2 (Interpretation with Böhm trees) A simple interpretation of this system is to see Y_{succ} as representing the "infinite integer" ∞ , the term $\text{minus } Y_{\text{succ}} \ Y_{\text{succ}}$ representing the undefined operation $\infty - \infty$. This interpretation can be made concrete by using Böhm trees [Bar84]. The Böhm tree of the term Y_{succ} is the infinite term

$$\begin{array}{c}
 \text{succ} \\
 | \\
 \text{succ} \\
 | \\
 \vdots
 \end{array}$$

Intuitively, Ex. 3.1 can be seen as an instance of the fact that confluence of non left-linear systems is not preserved when we extend the term algebra (in this case by infinite terms).

As shown in [Mül92], confluence is preserved when rewriting is left-linear. The original result concerns only algebraic systems, but can easily be extended to unconditional rewrite systems with arbitrary right-hand sides.

Theorem 3.3 ([Mül92]) *If \mathcal{R} is a left-linear unconditional rewrite system such that $\rightarrow_{\mathcal{R}}$ is confluent then $\rightarrow_{\beta \cup \mathcal{R}}$ is confluent.*

This result has been generalized to the case of Higher-Order Rewrite Systems [OR94].

3.1.2 Strongly beta-normalizing terms

To handle non left-linear systems, as seen in Example 3.1 we have to forbid infinite terms. This is possible for example by focusing on algebraic rewriting on typed terms. Confluence is preserved for the combination of algebraic rewriting with simply typed λ -calculus [BT88]. This result has been then extended to the polymorphic λ -calculus [BTG89, BTG94, Oka89].

A question arises from these results: besides strong normalization of β -reduction, what is the role of typing in the preservation of confluence? This is studied in [Dou92], which shows that for algebraic rewriting terms must satisfy some *arity conditions*.

Example 3.4 *Consider the rewrite system*

$$\text{id } x \mapsto_{\text{id}} x ,$$

and let $\Omega_{\text{succ}} =_{\text{def}} \lambda x. \text{succ } (x \ x)$. The term

$$t =_{\text{def}} \text{minus } (\text{id } \Omega_{\text{succ}} \ \Omega_{\text{succ}}) (\text{id } \Omega_{\text{succ}} \ \Omega_{\text{succ}})$$

is in β -normal form, hence strongly β -normalizing. Moreover, we can check that the rewrite system $\mapsto_{\text{minus} \cup \text{id}}$ is confluent. However, $\mapsto_{\beta \cup \text{minus} \cup \text{id}}$ is not confluent since t rewrites to the unjoinable peak of Example 3.1:

$$\text{minus } (\text{id } \Omega_{\text{succ}} \ \Omega_{\text{succ}}) (\text{id } \Omega_{\text{succ}} \ \Omega_{\text{succ}}) \xrightarrow{\text{id}}^2 \text{minus } Y_{\text{succ}} \ Y_{\text{succ}} .$$

The problem is that reducing id in the β -normal form $\text{id } \Omega_{\text{succ}} \ \Omega_{\text{succ}}$ leads to a term which is no longer in β -normal form: rewriting does not preserve β -normal forms. The approach taken in [Dou92] is to find *arity conditions* on terms for rewriting to preserve β -normal forms. Consider a symbol $f \in \Sigma$ such that for all $f\vec{l} \mapsto_{\mathcal{R}} r$, we have $|\vec{l}| \leq n$. Then we discard terms of the form $f\vec{t}$ with $|\vec{t}| > n$. For example, the term $\text{id } \Omega_{\text{succ}} \ \Omega_{\text{succ}}$ is not allowed since id takes two arguments whereas its rewrite rule takes only one.

Definition 3.5 (Applicative Arity) *An arity is a function $\mathbf{a} : \Sigma \rightarrow \mathbb{N}$.*

(i) *A term t respects \mathbf{a} if it contains no subterm $f\vec{t}$ with $|\vec{t}| > \mathbf{a}(f)$.*

(ii) *A rewrite system \mathcal{R} respects \mathbf{a} if for all $f\vec{l} \mapsto_{\mathcal{R}} r$, $f\vec{l}$ and r respect \mathbf{a} and moreover $|\vec{l}| = \mathbf{a}(f)$.*

However, the respect of an arity is not stable by β -reduction. For instance, with $\mathbf{a}(\text{id}) = 1$ the term $(\lambda x. x \ \Omega_{\text{succ}} \ \Omega_{\text{succ}}) \text{id}$ respects \mathbf{a} but it β -reduces to $\text{id } \Omega_{\text{succ}} \ \Omega_{\text{succ}}$ which does not respect \mathbf{a} . In order to work with terms which respect an arity \mathbf{a} and whose $\beta\mathcal{R}$ -reducts respect \mathbf{a} too, it is convenient to consider sets of terms respecting \mathbf{a} and which are stable by reduction. This motivates the following definition.

Definition 3.6 ((\mathcal{R}, \mathbf{a})-Stable Terms) *Given an arity \mathbf{a} and a rewrite system \mathcal{R} , a set of terms S is (\mathcal{R}, \mathbf{a})-stable if*

(i) *for all $t \in S$, t respects \mathbf{a} ,*

(ii) for all $t \in S$, if $t \rightarrow_{\beta \cup \mathcal{R}} u$ then $u \in S$,

(iii) for all $t \in S$, if u is a subterm of t then $u \in S$.

Dougherty [Dou92] obtain the preservation of confluence on $(\mathcal{R}, \mathbf{a})$ -stable sets of strongly β -normalizing terms.

Theorem 3.7 ([Dou92]) *If \mathcal{R} is an algebraic confluent unconditional rewrite system that respects an arity \mathbf{a} , then $\rightarrow_{\beta \cup \mathcal{R}}$ is confluent on every $(\mathcal{R}, \mathbf{a})$ -stable set $S \subseteq \mathcal{SN}_\beta$.*

Remark 3.8 *To get the preservation β -normal forms by rewriting it is necessary to restrict to algebraic right-hand sides, since in contrast with algebraic terms, substituting a variable in an applicative term may produce a β -redex. For instance $(xz)[\lambda y.y/x] = (\lambda y.y)z$.*

3.2 Orthogonal conditional rewriting

Orthogonality is a sufficient condition for the confluence of some kinds of conditional rewriting. In this section we recall some known results about the confluence of algebraic orthogonal conditional rewrite systems. They were initially formulated in the framework of first-order conditional rewriting, of which algebraic rewriting is an instance. The main result is the shallow confluence of orthogonal normal conditional rewriting. We generalize it in Section 6 to orthonormal β -conditional rewriting.

For *unconditional* rewriting, orthogonality is a simple syntactic criterion: it entails the confluence of left-linear systems with no critical pairs [Hue80]. With conditional rewriting, things get more complicated since the notion of critical pairs has to take into account the conditions of rewrite rules.

Definition 3.9 (Conditional Critical Pairs) *Let \mathcal{R} be a set of conditional rules and suppose that $\rho_1 : \vec{d} = \vec{c} \supset l \mapsto r$ and $\rho_2 : \vec{d}' = \vec{c}' \supset l' \mapsto r'$ are two renaming of rules in \mathcal{R} such that they have no variable in common. If p is a non-variable occurrence of l and σ is a most general unifier of l_p and l' , then*

$$\vec{d}\sigma = \vec{c}\sigma \quad \wedge \quad \vec{d}'\sigma = \vec{c}'\sigma \quad \supset \quad (l[p \leftarrow r']\sigma, r\sigma)$$

is a conditional critical pair of \mathcal{R} . If ρ_1 and ρ_2 are renaming of the same rule, we assume that p is not the root position of l . A critical pair of the form $\vec{d} = \vec{c} \supset (s, s)$ is called trivial.

The important point is that in a conditional critical pair $\vec{d} = \vec{c} \supset (s, t)$, it is possible that there is no substitution σ such that $\vec{d}\sigma = \vec{c}\sigma$. Thus, critical pairs can be *feasible* or *unfeasible*. According to the kind of conditional rewriting considered (with and without β -reduction in the evaluation of conditions), the satisfaction of conditions is done differently. Therefore, we consider two notions of feasibility.

Definition 3.10 (Feasibility of Conditional Critical Pairs) *A critical pair $\vec{d} = \vec{c} \supset (s, t)$ of a conditional system \mathcal{R} is*

- *feasible if there is a substitution σ such that $\vec{d}\sigma \downarrow_{\mathcal{R}} \vec{c}\sigma$;*
- *β -feasible if there is a substitution σ such that $\vec{d}\sigma \downarrow_{\beta \cup \mathcal{R}(\beta)} \vec{c}\sigma$;*

A critical pair which is not feasible (resp. β -feasible) is said unfeasible (resp. β -unfeasible).

The easiest way to prove unfeasibility of critical pairs is often to use confluence. We come back on this question in Section 6. Both notions of feasibility induce a notion of orthogonality.

Definition 3.11 (Orthogonality) *A set \mathcal{R} of left-linear conditional rewrite rules is*

- orthogonal (*resp.* β -orthogonal) *if all its critical pairs are unfeasible (*resp.* β -unfeasible);*
- weakly orthogonal (*resp.* weakly β -orthogonal) *if all its critical pairs are either trivial or unfeasible (*resp.* β -unfeasible).*

Hence, to test the orthogonality of a conditional system, we have to evaluate the conditions of its critical pairs. According to Remark 2.12, this is in general undecidable.

It is well-known that for normal (and semi-equational) rewriting, weak orthogonality implies confluence. This in general *not* the case for join conditional rewriting, as shown in [BK86].

Theorem 3.12 ([BK86, Ohl02]) *Let \mathcal{R} be a conditional rewrite system. If \mathcal{R} is weakly orthogonal, and moreover is a normal system, then $\rightarrow_{\mathcal{R}}$ is shallow confluent.*

3.3 Overview of the results

The goal of this paper is to give sufficient conditions for the confluence of β -reduction with β -conditional rewriting (i.e. with β -steps allowed in the evaluation of conditions).

More precisely, we seek to obtain results on the *preservation* of confluence, that is to get the confluence of $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$ assuming the confluence of $\rightarrow_{\mathcal{R}}$. Our approach is to generalize the results summarized in Section 3.1 on the combination of β -reduction with unconditional rewriting. We thus consider two different cases:

- First, the extension of Müller’s result [Mül92], when β -reduction is not restricted (we thus need to assume left-linearity, and to extend this notion to conditional rewriting).
- Second, the extension of Dougherty’s result [Dou92], when we restrict to β -normalizing terms (we thus need some arity conditions on terms). In fact, we improve [Dou92] from strongly β -normalizing terms to weakly β -normalizing terms.

In each case, we proceed in two steps. We first consider in Section 4 the case of β -reduction with conditional rewriting $\rightarrow_{\mathcal{R}}$ (when β -reduction is not allowed in the evaluation of conditions). We then extend these results to β -conditional rewriting $\rightarrow_{\mathcal{R}(\beta)}$ in Section 5.

As discussed at the beginning of Section 5 (see Example 5.2), for the extension of both [Mül92] and [Dou92] to β -conditional rewriting, rewrite rules must be algebraic and respect arity conditions. In contrast, the extension of [Mül92] to the simpler case of conditional rewriting holds without these restrictions. This motivates the definition of criteria for the confluence of β -reduction with β -conditional rewriting when rules need not be algebraic nor to respect an arity (recall from Definition 2.8 that left-hand sides are always algebraic in this paper). We propose such a criterion in Section 6, which defines *orthonormal* conditional rewriting, an extension of orthogonal rewriting. We show the *shallow* confluence of β -reduction with β -conditional rewriting for these systems, hence extending Theorem 3.12.

Our results are summarized in Figure 1 page 6.

4 Confluence of beta-reduction with conditional rewriting

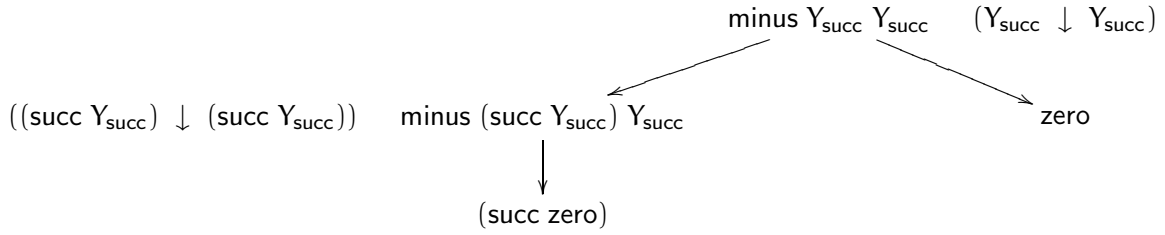
We now turn to conditional rewriting. In this section we focus on the combination of join conditional rewriting $\rightarrow_{\mathcal{R}}$ with β -reduction: we do not allow the use of β -reduction in the evaluation of conditions.

The important point of left-linearity is to prevent *unconditional* rewriting from comparing arbitrary terms. It forbids in particular comparisons of infinite terms such as Y_{succ} . But with conditional rewriting, rewrite rules can make this comparison in their conditions while being left-linear. Hence, starting from Example 3.1, we can define a left-linear conditional rewrite system which makes the commutation of rewriting with β -reduction fail.

Example 4.1 *Consider the conditional system*

$$x = y \supset \text{minus } x \ y \mapsto \text{zero} \qquad x = (\text{succ } y) \supset \text{minus } x \ y \mapsto (\text{succ } \text{zero}) .$$

This system is left-linear, but the conditions can test the equality of open terms. The join conditional rewrite relation issued from this system forms with \rightarrow_{β} the following unjoinable peak:



As for unconditional rewriting in Section 3.1, we consider two ways to overcome the problem: to restrict rewriting (Section 4.1) or to restrict β -reduction (Section 4.2).

4.1 Confluence for left-linear semi-closed systems

In this section we are interested in the extension of Theorem 3.3 ([Mül92]) to conditional rewriting: we want sufficient conditions on rewrite rules for the preservation of confluence on *all untyped terms of $\Lambda(\Sigma)$* . As seen in Example 4.1, for conditional rewriting we have to extend the notion of left-linearity in order to forbid comparison of open terms in the conditions of rewrite rules. To this end we restrict to *semi-closed* conditional rewrite rules.

Definition 4.2 (Semi-Closed Conditional Rewrite Rules) *A conditional rewrite system \mathcal{R} is semi-closed if for all rules*

$$d_1 = c_1 \wedge \dots \wedge d_n = c_n \supset l \mapsto_{\mathcal{R}} r ,$$

the terms c_1, \dots, c_n are applicative and closed.

For example, the system `Tree` of Section 2.4 is left-linear and semi-closed. In a semi-closed rule $\vec{d} = \vec{c} \supset l \mapsto r$, since \vec{c} are closed terms, it is tempting to normalize them and obtain a normal rewrite relation, but as noted in Remark 2.12, results on join rewriting seem more easily applicable.

We show that the confluence of $\rightarrow_{\mathcal{R}}$ implies the confluence of $\rightarrow_{\beta \cup \mathcal{R}}$ for semi-closed left-linear right-applicative systems (Theorem 4.6). Using Hindley-Rosen lemma (Lemma 2.17), this follows from the commutation of conditional rewriting with β -reduction. As in [Mül92], we obtain this commutation as a consequence of the commutation of conditional rewriting with a relation \triangleright_{β} of *parallel β -reduction* (see Definition 2.5). This is shown in Lemma 4.5, which relies on Property 2.6 ($\sigma \triangleright_{\beta} \sigma'$ implies $t\sigma \triangleright_{\beta} t\sigma'$). This property holds for parallel rewrite relations but fails with \rightarrow_{β} .

The parallel β -reduction \triangleright_{β} we use is Tait and Martin-Löf's relation [Bar84, Tak95]. It is defined as follows.

Definition 4.3 (Parallel β -Reduction) We let \triangleright_β be the smallest parallel rewrite relation closed under the rule

$$(\triangleright_\beta) \frac{t_1 \triangleright_\beta u_1 \quad t_2 \triangleright_\beta u_2}{(\lambda x. t_1) t_2 \triangleright_\beta u_1[u_2/x]}$$

We will use some well-known properties of \triangleright_β . If $\sigma \triangleright_\beta \sigma'$ then $s\sigma \triangleright_\beta s\sigma'$; this is the one-step reduction of parallel redexes. We can also simulate β -reduction: $\rightarrow_\beta \subseteq \triangleright_\beta \subseteq \rightarrow_\beta^*$. And third, \triangleright_β enjoys the diamond property: $\triangleleft_\beta \triangleright_\beta \subseteq \triangleright_\beta \triangleleft_\beta$.

The relation \triangleright_β is stronger than the one used in [Mül92]: it can reduce in one step nested β -redexes, while the relation of [Mül92] is simply the smallest parallel rewrite relation containing β -reduction (i.e. the *parallel closure* of \rightarrow_β). The diamond property (which holds for \triangleright_β) fails for the parallel closure of β -reduction precisely because it cannot reduce nested β -redexes in one step. The parallel closure of \rightarrow_β would have been sufficient to obtain Lemma 4.5, but we use the nested relation \triangleright_β because we rely on the diamond property in Section 5.1.

Nested parallelizations (corresponding to complete developments) are already used in [OR94] for their confluence proof of HORSs. However, our method inherits more from [Mül92] than from [OR94], as we use complete developments of \rightarrow_β only, whereas complete developments of \rightarrow_β and of $\rightarrow_{\mathcal{R}}$ are used for the modularity result of [OR94].

The left-linearity assumption is used in the proof of Lemma 4.5 via the following property of linear algebraic terms.

Proposition 4.4 *Let t be an algebraic linear term and σ be a substitution such that $t\sigma \triangleright_\beta u$. There is a substitution σ' such that $u = t\sigma'$ with $\sigma \triangleright_\beta \sigma'$ and $\sigma'(x) = \sigma(x)$ for all $x \notin \text{FV}(t)$.*

PROOF. By induction on t .

$t = x \in \mathcal{X}$. In this case $t\sigma = \sigma(x)$. Take σ' such that $\sigma'(x) = u$ and $\sigma'(y) = \sigma(y)$ for all $y \neq x$.

$t = f \in \Sigma$. In this case $t\sigma = t = u$, hence $\sigma' = \sigma$ fits (recall that \triangleright_β is reflexive).

$t = t_1 t_2$. Since t is algebraic, $t_1\sigma t_2\sigma$ is not a β -redex. It follows that u is of the form $u_1 u_2$ with $(t_1\sigma, t_2\sigma) \triangleright_\beta (u_1, u_2)$. By induction hypothesis, there are two substitutions σ'_1 and σ'_2 such that for each $i \in \{1, 2\}$ we have $\sigma \triangleright_\beta \sigma'_i$, $u_i = t_i\sigma'_i$, and $\sigma'_i(x) = \sigma(x)$ for all $x \notin \text{FV}(t_i)$. Since t is linear, $\text{FV}(t_1) \cap \text{FV}(t_2) = \emptyset$, hence with $\sigma' =_{\text{def}} \sigma'_1 \uplus \sigma'_2$, we have $u = u_1 u_2 = t_1\sigma' t_2\sigma' = t\sigma'$, $\sigma \triangleright_\beta \sigma'$ and $\sigma(y) = \sigma'(y)$ for all $y \notin \text{FV}(t)$. \square

We are now ready to prove the commutation of $\rightarrow_{\mathcal{R}}$ and \triangleright_β . In fact we prove a slightly stronger statement, which can be termed as the "level commutation" of $\rightarrow_{\mathcal{R}}$ and \triangleright_β .

Lemma 4.5 (Commutation of $\rightarrow_{\mathcal{R}_i}$ with \triangleright_β) *If \mathcal{R} is a conditional rewrite system which is semi-closed, left-linear and right-applicative, then \triangleright_β commutes with $\rightarrow_{\mathcal{R}_i}$ for all $i \in \mathbb{N}$:*

$$\begin{array}{ccc} \cdot & \xrightarrow{\mathcal{R}_i} & \cdot \\ \triangleright_\beta \downarrow * & & * \downarrow \triangleright_\beta \\ \cdot & \xrightarrow{\mathcal{R}_i} & \cdot \end{array}$$

PROOF. We reason by induction on $i \in \mathbb{N}$. The base case $i = 0$ is trivial. Let $i \geq 0$ and assume that $\rightarrow_{\mathcal{R}_i}$ commutes with \triangleright_β . We show that $\rightarrow_{\mathcal{R}_{i+1}}$ commutes with \triangleright_β .

We begin by showing that for all $t, u, v \in \Lambda(\Sigma)$, if $u \triangleleft_{\beta} t \rightarrow_{\mathcal{R}_{i+1}} v$ then there is a term w such that $u \rightarrow_{\mathcal{R}_{i+1}}^* w \triangleleft_{\beta} v$. In diagrammatic form:

$$\begin{array}{ccc}
 t & \xrightarrow{\mathcal{R}_{i+1}} & v \\
 \triangleright_{\beta} \downarrow & & \downarrow \triangleright_{\beta} \\
 u & \xrightarrow[\mathcal{R}_{i+1}]{*} & w
 \end{array} \tag{3}$$

We deduce from (3) that $\rightarrow_{\mathcal{R}_{i+1}}$ commutes with \triangleright_{β} by applying Lemma 2.18.

We now show (3) by induction on t . If both reductions $t \triangleright_{\beta} u$ and $t \rightarrow_{\mathcal{R}_{i+1}} v$ occur in a proper subterm of t then we conclude by induction hypothesis. Otherwise there are two cases.

(i) $t = (\lambda x.t_1)t_2$ with $u = u_1[u_2/x]$ and $v = (\lambda x.v_1)v_2$ where $(u_1, u_2) \triangleleft_{\beta} (t_1, t_2) \rightarrow_{\mathcal{R}_{i+1}} (v_1, v_2)$. By induction hypothesis, there are terms w_1 and w_2 such that $u_i \rightarrow_{\mathcal{R}_{i+1}}^* w_i \triangleleft_{\beta} v_i$. We deduce that $u_1[u_2/x] \rightarrow_{\mathcal{R}_{i+1}}^* w_1[w_2/x]$ and that $(\lambda x.v_1)v_2 \triangleright_{\beta} w_1[w_2/x]$.

(ii) t is the \mathcal{R} -redex contracted in the step $t \rightarrow_{\mathcal{R}_{i+1}} v$. In this case, there is a conditional rule $\vec{d} = \vec{c} \supset l \mapsto_{\mathcal{R}} r$ and a substitution σ s.t. $t = l\sigma$ and $v = r\sigma$. Moreover, there are terms \vec{v} such that $\vec{d}\sigma \rightarrow_{\mathcal{R}_i}^* \vec{v} \leftarrow_{\mathcal{R}_i}^* \vec{c}\sigma$. Since \mathcal{R} is semi-closed, the terms \vec{c} are closed and applicative, hence $\vec{c}\sigma = \vec{c}$ and the terms \vec{v} are applicative since \mathcal{R} is right-applicative. Since l is left-linear and algebraic, we deduce from Proposition 4.4 that there is a substitution σ' such that $\sigma \triangleright_{\beta} \sigma'$ and $u = l\sigma'$. It follows Proposition 2.6 that $r\sigma \triangleright_{\beta} r\sigma'$ and $\vec{d}\sigma \triangleright_{\beta} \vec{d}\sigma'$.

To conclude that $w =_{\text{def}} r\sigma'$ fits, it remains to show that $l\sigma' \rightarrow_{\mathcal{R}_{i+1}} r\sigma'$, that is $\vec{d}\sigma' \downarrow_{\mathcal{R}_i} \vec{c}$. We have $\vec{d}\sigma' \triangleleft_{\beta} \vec{d}\sigma \rightarrow_{\mathcal{R}_i}^* \vec{v}$, hence by induction hypothesis there are \vec{w} s.t. $\vec{d}\sigma' \rightarrow_{\mathcal{R}_i}^* \vec{w} \triangleleft_{\beta}^* \vec{v}$. It follows that $\vec{d}\sigma' \rightarrow_{\mathcal{R}_i}^* \vec{v}$, the terms \vec{v} being applicative hence in β -normal form. We thus have $\vec{d}\sigma' \downarrow_{\mathcal{R}_i} \vec{c}$, and conclude that $l\sigma' \rightarrow_{\mathcal{R}_{i+1}} r\sigma'$. \square

A direct application of Hindley-Rosen's Lemma (Lem. 2.17) then offers the preservation of confluence.

Theorem 4.6 (Confluence of $\rightarrow_{\beta \cup \mathcal{R}}$) *Let \mathcal{R} be a semi-closed left-linear right-applicative system. If $\rightarrow_{\mathcal{R}}$ is confluent then so is $\rightarrow_{\beta \cup \mathcal{R}}$.*

Comparison with Müller's work. Our main result on the confluence of β -reduction with conditional rewriting for left-linear semi-closed system (Theorem 4.6) is not a true extension of Theorem 3.3. Indeed, Theorem 3.3 applies to unconditional systems with arbitrary right-hand sides, while Theorem 4.6 requires right-hand sides to be applicative.

The problem is that Lemma 4.5 may fail with non-applicative right-hand sides. Consider the system:

$$h \mapsto \lambda x.x \quad x = h a \supset g x \mapsto a$$

We have $g a \leftarrow_{\beta} g((\lambda x.x)a) \rightarrow_{\mathcal{R}} a$. But since the term $(\lambda x.x)a$ is a \mathcal{R} -normal form, the condition $x \downarrow_{\mathcal{R}} h a$ is not satisfied, and $g a$ is a \mathcal{R} -normal form.

We can extend Theorem 3.3 to normal conditional rewriting, i.e. to systems \mathcal{R} such that in all rules $\vec{d} = \vec{c} \supset l \mapsto_{\mathcal{R}} r$, the closed algebraic terms \vec{c} are required to be in normal form w.r.t. $\rightarrow_{\mathcal{R}}$ (recall from Remark 2.12 that this is undecidable). The proof follows exactly the same scheme as for Theorem 4.6. The only difference lies in the commutation of $\rightarrow_{\mathcal{R}_i}$ with \rightarrow_{β} , for which Lemma 4.5 does not apply.

Theorem 4.7 (Extension of [Mül92]) *Let \mathcal{R} be a left-linear semi-closed system such that $\rightarrow_{\mathcal{R}}$ is a normal conditional rewrite relation. If $\rightarrow_{\mathcal{R}}$ is confluent then so is $\rightarrow_{\beta \cup \mathcal{R}}$.*

PROOF. As in Theorem 4.6, the proof relies on the commutation of $\rightarrow_{\mathcal{R}_i}$ with \rightarrow_{β} . Since right-hand sides are not applicative, we can not use Lemma 4.5. However, the commutation of $\rightarrow_{\mathcal{R}_i}$ with \triangleright_{β} is proved using the same general reasoning, excepted for the following point. Assume that $\rightarrow_{\mathcal{R}_i}$ commutes with \triangleright_{β} and that for a rule $\vec{d} = \vec{c} \triangleright l \mapsto_{\mathcal{R}} r$ and a substitution σ we have $l\sigma' \triangleleft_{\beta} l\sigma \rightarrow_{\mathcal{R}_{i+1}} r\sigma$. As $\rightarrow_{\mathcal{R}}$ is normal, we have $\vec{d}\sigma \rightarrow_{\mathcal{R}_i}^* \vec{c}$, and by induction hypothesis there are \vec{c}' such that $\vec{d}\sigma' \rightarrow_{\mathcal{R}_i}^* \vec{c}' \leftarrow_{\beta}^* \vec{c}$. Since \mathcal{R} is semi-closed, the terms \vec{c} are algebraic hence β -normals. It follows that $\vec{d}\sigma' \rightarrow_{\mathcal{R}_i}^* \vec{c}$, hence $l\sigma' \rightarrow_{\mathcal{R}_{i+1}} r\sigma'$. \square

4.2 Confluence on weakly beta-normalizing terms

We now turn to the problem of dropping the left-linearity and semi-closure conditions. We generalize Theorem 3.7 [Dou92] in two ways. First, we adapt it to conditional rewriting. Second, we use weakly β -normalizing terms whose β -normal forms respect an arity, whereas Dougherty uses sets of strongly normalizing arity compliant terms closed under reduction.

As seen in Example 4.1, fixpoint combinators make the commutation of \rightarrow_{β}^* and $\rightarrow_{\mathcal{R}}^*$ fail when rewriting involves equality tests between open terms. When using weakly β -normalizing terms, we can project rewriting on β -normal forms (βnf), thus eliminating fixpoints as soon as they are not significant for the reduction.

Hence, we seek to obtain

$$\begin{array}{ccc}
 s & \xrightarrow{\beta \cup \mathcal{R}} & t \\
 \beta \downarrow^* & & \downarrow^* \beta \\
 \beta\text{nf}(s) & \xrightarrow[\mathcal{R}]{*} & \beta\text{nf}(t)
 \end{array} \tag{4}$$

We rely on the following:

- (i) First, β -normal forms should be stable by rewriting. By Remark 3.8 we must assume that right-hand sides are algebraic, and as seen in Example 3.4, we must use the notion of applicative arity (Definition 3.5).
- (ii) We need normalizing β -derivations to commute with rewriting. This follows from using the leftmost-outermost strategy of λ -calculus.
- (iii) Finally, we assume that conditions are algebraic. Since left-hand sides and right-hand sides are algebraic (by Definition 2.8 and item (i) respectively), this entails that for all rules $\vec{d} = \vec{c} \triangleright l \mapsto_{\mathcal{R}} r$ and all substitutions σ , we have $\beta\text{nf}(\vec{d}\sigma) = \vec{d} \beta\text{nf}(\sigma)$, $\beta\text{nf}(\vec{c}\sigma) = \vec{c} \beta\text{nf}(\sigma)$, $\beta\text{nf}(l\sigma) = l \beta\text{nf}(\sigma)$, $\beta\text{nf}(r\sigma) = r \beta\text{nf}(\sigma)$.

We now have to extend to conditional rewriting the condition of arity on rewrite rules stated in Definition 3.5.(ii).

Definition 4.8 (Applicative Arity for Conditional Rules) *A rule $\vec{d} = \vec{c} \triangleright l \mapsto_{\mathcal{R}} r$ respects an arity $\mathbf{a} : \Sigma \rightarrow \mathbb{N}$ if the terms \vec{d}, \vec{c} respect \mathbf{a} and the unconditional rule $l \mapsto r$ respects \mathbf{a} .*

As seen in Example 3.4, terms and rewrite systems respecting an arity prevent collapsing rules from creating β -redexes.

However, we do not assume that every term at hand respects an arity. If a term has a β -normal form, the leftmost-outermost strategy for β -reduction never evaluates non-normalizing subterms.

It follows that such subterms may not respect any arity without disturbing the projection on β -normal forms. Therefore it is sufficient to require that terms have a β -normal form that respects an arity.

Definition 4.9 *Given an arity $\mathbf{a} : \Sigma \rightarrow \mathbb{N}$, we let $\mathcal{AN}_{\mathbf{a}}$ be the set of terms having a β -normal form, and whose β -normal form respects \mathbf{a} .*

The proof goes through essentially thanks to two points: the well-foundedness of the leftmost-outermost strategy for \rightarrow_{β} on weakly β -normalizing terms [Bar84]; and the fact that this strategy can be described by means of *head* β -reductions, that are easily shown to commute with (parallel) conditional rewriting.

We use a well-founded relation containing head β -reductions. Recall that by Lemma 2.3, any λ -term can be written either

$$\begin{aligned} & \lambda \vec{x}.v \mathbf{a}_0 \mathbf{a}_1 \dots \mathbf{a}_n & (a) \\ \text{or } & \lambda \vec{x}.(\lambda \mathbf{y}.b) \mathbf{a}_0 \mathbf{a}_1 \dots \mathbf{a}_n & (b) \end{aligned}$$

where $v \in \mathcal{X} \cup \Sigma$. We denote *head* β -reductions by \rightarrow_h . They consist of head β -steps:

$$\lambda \vec{x}.(\lambda \mathbf{y}.b) \mathbf{a}_0 \mathbf{a}_1 \dots \mathbf{a}_n \rightarrow_h \lambda \vec{x}.b[\mathbf{a}_0/\mathbf{y}] \mathbf{a}_1 \dots \mathbf{a}_n .$$

We use the relation \succ defined as:

$$\begin{aligned} & \lambda \vec{x}.v \mathbf{a}_0 \mathbf{a}_1 \dots \mathbf{a}_n \succ \mathbf{a}_i & (a) \\ & \lambda \vec{x}.(\lambda \mathbf{y}.b) \mathbf{a}_0 \mathbf{a}_1 \dots \mathbf{a}_n \succ \lambda \vec{x}.b[\mathbf{a}_0/\mathbf{y}] \mathbf{a}_1 \dots \mathbf{a}_n & (b) \end{aligned}$$

where $v \in \mathcal{X} \cup \Sigma$ and $0 \leq i \leq n$ and $n > 0$. Note that in the case (a), \mathbf{a}_i can have free variables among \vec{x} , hence it can also be a subterm of a term α -equivalent to $\lambda \vec{x}.v \vec{a}$; for instance $\lambda x.f x \succ y$ for all $y \in \mathcal{X}$. Recall that \mathcal{WN}_{β} is the set of weakly β -normalizing terms.

Lemma 4.10 *If $s \in \mathcal{WN}_{\beta}$ and $s \succ t$ then $t \in \mathcal{WN}_{\beta}$. Moreover, \succ is well-founded on \mathcal{WN}_{β} .*

PROOF. For the first part, let $s \in \mathcal{WN}_{\beta}$ and $s \succ t$. If s is of the form (b), the first step of the leftmost-outermost derivation normalizing s is t . Hence $t \in \mathcal{WN}_{\beta}$. Otherwise, if t has no β -normal form, then s has no β -normal form.

For the second part, we write $\#(s)$ for the number of \rightarrow_h -steps in the leftmost-outermost derivation starting from s and $|s|$ for the size of s . We show that if $s \succ t$, then $(\#(s), |s|) \succ_{\text{lex}} (\#(t), |t|)$. If s is of the form (b), by the first point $t \in \mathcal{WN}_{\beta}$. Since $s \rightarrow_h t$, we have $\#(s) > \#(t)$. Otherwise, the leftmost-outermost strategy starting from s reduces each \mathbf{a}_i by leftmost-outermost reductions. Hence $\#(s) \geq \#(t)$. But in this case, t is a proper subterm of s , hence $|s| > |t|$. \square

It follows that we can reason by well-founded induction on \succ . For all $i \geq 0$, we use a *nested* parallelization of $\rightarrow_{\mathcal{R}_i}$. It corresponds to the one used in [OR94], that can be seen as a generalization of Tait and Martin-Löf parallel relation. As for \triangleright_{β} and \rightarrow_{β} , in the orthogonal case, a complete development of $\rightarrow_{\mathcal{R}_i}$ can be simulated by *one step* $\triangleright_{\mathcal{R}_i}$ -reduction. This relation is also an adaptation to conditional rewriting of the parallelization used in [Dou92].

Definition 4.11 (Conditional Nested Parallel Relations) *For all $i \geq 0$, let $\triangleright_{\mathcal{R}_i}$ be the smallest parallel rewrite relation closed under the rule*

$$(\triangleright_{\mathcal{R}}) \frac{\vec{d} = \vec{c} \supset l \mapsto_{\mathcal{R}} r \quad l\sigma \rightarrow_{\mathcal{R}_i} r\sigma \quad \sigma \triangleright_{\mathcal{R}_i} \theta}{l\sigma \triangleright_{\mathcal{R}_i} r\theta}$$

Recall that $\lambda\sigma \rightarrow_{\mathcal{R}_i} r\sigma$ is ensured by $\vec{d}\sigma \downarrow_{\mathcal{R}_{i-1}} \vec{c}\sigma$. These relations enjoy some nice properties:

Proposition 4.12 For all $i \geq 0$,

$$(i) \rightarrow_{\mathcal{R}_i} \subseteq \triangleright_{\mathcal{R}_i} \subseteq \rightarrow_{\mathcal{R}_i}^*;$$

$$(ii) s \triangleright_{\mathcal{R}_i} t \implies u[s/x] \triangleright_{\mathcal{R}_i} u[t/x];$$

$$(iii) [s \triangleright_{\mathcal{R}_i} t \ \& \ u \triangleright_{\mathcal{R}_i} v] \implies u[s/x] \triangleright_{\mathcal{R}_i} v[t/x].$$

PROOF. The first point is shown by induction on the definition of $\triangleright_{\mathcal{R}_i}$; the second follows from Proposition 2.6 and the fact that $\triangleright_{\mathcal{R}_i}$ is a parallel rewrite relation. For the last one, we use an induction on $\triangleright_{\mathcal{R}_i}$ in $u \triangleright_{\mathcal{R}_i} v$. If u is v , the result is trivial. If $u \triangleright_{\mathcal{R}_i} v$ was obtained by ($\triangleright_{\text{APP}}$) or ($\triangleright_{\text{ABS}}$), the result follows from induction hypothesis. Otherwise, $u \triangleright_{\mathcal{R}_i} v$ is obtained by $\triangleright_{\mathcal{R}}$. That is, there is a rule $\vec{d} = \vec{c} \supset l \mapsto_{\mathcal{R}} r$ such that $u = \lambda\sigma$, $v = r\theta$, $\sigma \triangleright_{\mathcal{R}_i} \theta$ and $\lambda\sigma \rightarrow_{\mathcal{R}_i} r\sigma$. Since $\rightarrow_{\mathcal{R}_i}$ is a rewrite relation, we have $\lambda\sigma[s/x] \triangleright_{\mathcal{R}_i} r\sigma[s/x]$. By induction hypothesis, we have $\sigma[s/x] \triangleright_{\mathcal{R}_i} \theta[t/x]$. Therefore $\lambda\sigma[s/x] \triangleright_{\mathcal{R}_i} r\theta[t/x]$. \square

We now turn to the *one step* commutation of $\triangleright_{\mathcal{R}_i}$ and \leftarrow_h . This is a direct consequence of the third point of Proposition 4.12. Commutation of \rightarrow_h with (unconditional) rewriting has already been coined in [BFG97].

Lemma 4.13 Let $i \geq 0$. If $u \leftarrow_h s \triangleright_{\mathcal{R}_i} t$ then there exists v such that $u \triangleright_{\mathcal{R}_i} v \leftarrow_h t$:

$$\begin{array}{ccc} s & \xrightarrow{\triangleright_{\mathcal{R}_i}} & t \\ h \downarrow & & \downarrow h \\ u & \xrightarrow{\triangleright_{\mathcal{R}_i}} & v \end{array}$$

PROOF. Assume that $s \leftarrow_h \lambda\vec{x}.(\lambda y.a_0)a_1 \dots a_p \triangleright_{\mathcal{R}_i} t$. Because rules have non-variable algebraic left-hand sides, $t = \lambda\vec{x}.(\lambda y.b_0)b_1 \dots b_p$ with for all $k \in \{0, \dots, p\}$, $a_k \triangleright_{\mathcal{R}_i} b_k$. On the other hand, $s = \lambda\vec{x}.a_0[a_1/y]a_2 \dots a_p$. It follows from Proposition 4.12.(iii) that $a_0[a_1/x] \triangleright_{\mathcal{R}_i} b_0[b_1/x]$ (in *one step*). Hence we have $s \triangleright_{\mathcal{R}_i} \lambda\vec{x}.b_0[b_1/y]b_2 \dots b_p \leftarrow_h t$. \square

The main lemma is the projection of rewriting on β -normal forms, that is, the commutation of diagram (4).

Lemma 4.14 Let $\mathbf{a} : \Sigma \rightarrow \mathbb{N}$ be an arity and \mathcal{R} be an algebraic conditional rewrite system which respects \mathbf{a} . For all $i \in \mathbb{N}$, if $t \in \mathcal{AN}_{\mathbf{a}}$ and $t \rightarrow_{\beta \cup \mathcal{R}_i}^* u$, then $u \in \mathcal{AN}_{\mathbf{a}}$ and $\beta\text{nf}(t) \rightarrow_{\mathcal{R}_i}^* \beta\text{nf}(u)$.

PROOF. We reason by induction on $i \in \mathbb{N}$. The base case $i = 0$ is trivial. We assume that the property holds for $i \geq 0$ and show it for $i + 1$. The proof is in two steps.

(i) We begin by showing that for all $t \in \mathcal{AN}_{\mathbf{a}}$ we have

$$\begin{array}{ccc} t & \xrightarrow{\triangleright_{\mathcal{R}_{i+1}}} & u \\ \beta \downarrow^* & & \downarrow^* \beta \\ \beta\text{nf}(t) & \xrightarrow{\triangleright_{\mathcal{R}_{i+1}}} & \beta\text{nf}(u) \end{array} \quad (5)$$

We reason by induction on \succ using Lemma 2.3.

$t = \lambda\vec{x}.xt_1 \dots t_n$. In this case, $\beta\text{nf}(t) = \lambda\vec{x}.x \beta\text{nf}(t_1) \dots \beta\text{nf}(t_n)$ and $u = \lambda\vec{x}.xu_1 \dots u_n$ with $t_k \triangleright_{\mathcal{R}_{i+1}} u_k$ for all $k \in \{1, \dots, n\}$. As $t \succ t_k$, for all $k \in \{1, \dots, n\}$, by induction hypothesis on \succ we have $u_k \in \mathcal{AN}_a$ and $\beta\text{nf}(t_k) \triangleright_{\mathcal{R}_{i+1}} \beta\text{nf}(u_k)$. Since $\beta\text{nf}(u) = \lambda\vec{x}.x \beta\text{nf}(u_1) \dots \beta\text{nf}(u_n)$, we have $\beta\text{nf}(u) \in \mathcal{AN}_a$ and $\beta\text{nf}(t) \triangleright_{\mathcal{R}_{i+1}} \beta\text{nf}(u)$.

$t = \lambda\vec{x}.ft_1 \dots t_n$. If no rule is reduced at the head of t , the result follows from induction hypothesis on \succ . Otherwise, there is a rule $\vec{d} = \vec{c} \supset l \mapsto r$ such that $t = \lambda\vec{x}.l\sigma\vec{d}$ and $u = \lambda\vec{x}.r\theta\vec{b}$ with $l\sigma \triangleright_{\mathcal{R}_{i+1}} r\theta$ and $\vec{d}\sigma \downarrow_{\mathcal{R}_i} \vec{c}\sigma$. Since l is algebraic, $\beta\text{nf}(t)$ is of the form $\lambda\vec{x}.l\sigma'\vec{d}'$ where $\sigma' = \beta\text{nf}(\sigma)$ and $\vec{d}' = \beta\text{nf}(\vec{d})$. Since $\beta\text{nf}(t)$ respects a , $\vec{d}' = \emptyset$, hence $\vec{d} = \emptyset$ and $t = \lambda\vec{x}.l\sigma$. Therefore, because $l\sigma \triangleright_{\mathcal{R}_{i+1}} r\theta$, we have $\vec{b} = \emptyset$ and $u = \lambda\vec{x}.r\theta$. It remains to show that $u \in \mathcal{AN}_a$ and that $\beta\text{nf}(t) = \lambda\vec{x}.l\sigma' \triangleright_{\mathcal{R}_{i+1}} \beta\text{nf}(u)$. Because l is algebraic, its variables are $\prec^+ l$. We can then apply induction hypothesis on $\sigma \triangleright_{\mathcal{R}_{i+1}} \theta$. It follows that θ has a β -normal form θ' , which respects a and moreover such that $\sigma' \triangleright_{\mathcal{R}_{i+1}} \theta'$. Since r is algebraic, $\lambda\vec{x}.r\theta'$ is the β -normal form of u (which respects a). Hence it remains to show that $l\sigma' \triangleright_{\mathcal{R}_{i+1}} r\theta'$. Because $\sigma' \triangleright_{\mathcal{R}_{i+1}} \theta'$, it suffices to prove that $l\sigma' \rightarrow_{\mathcal{R}_{i+1}} r\theta'$. Thus, we are done if we show that $\vec{d}\sigma' \downarrow_{\mathcal{R}_i} \vec{c}\sigma'$. Since \vec{d} and \vec{c} are algebraic, $\beta\text{nf}(\vec{d}\sigma) = \vec{d}\sigma'$ and $\beta\text{nf}(\vec{c}\sigma) = \vec{c}\sigma'$. Now, since \vec{d} is algebraic and respects a , and since σ' respects a , it follows that $\vec{d}\sigma'$ respects a . The same holds for $\vec{c}\sigma'$. Hence we conclude by applying on $\vec{d}\sigma \downarrow_{\mathcal{R}_i} \vec{c}\sigma$ the induction hypothesis on i .

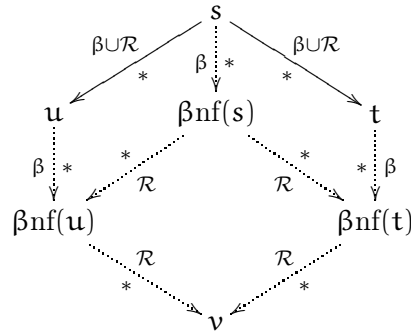
$t = \lambda\vec{x}.(\lambda x.v)wt_1 \dots t_n$. In this case, we head β -normalize t and obtain a term $t' \in \mathcal{AN}_a$. Using the commutation of $\triangleright_{\mathcal{R}_{i+1}}$ and \rightarrow_h , we obtain a term u' such that $t' \triangleright_{\mathcal{R}_{i+1}} u'$. Since $t \succ^+ t'$, we can reason as in the preceding cases.

- (ii) We now show that $t \rightarrow_{\beta \cup \mathcal{R}_i}^* u$ implies $\beta\text{nf}(t) \rightarrow_{\mathcal{R}_i}^* \beta\text{nf}(u)$. We reason by induction on $t \rightarrow_{\beta \cup \mathcal{R}_i}^* u$, using Proposition 4.12.(i). The base case $t = u$ is trivial. Assume that $t \rightarrow_{\beta \cup \mathcal{R}_i} v \rightarrow_{\beta \cup \mathcal{R}_i}^* u$. By induction hypothesis we have $\beta\text{nf}(v) \rightarrow_{\mathcal{R}_i}^* \beta\text{nf}(u)$. There are two cases. If $t \rightarrow_{\beta} v$, then $\beta\text{nf}(t) = \beta\text{nf}(v)$ and we are done. Otherwise we have $t \rightarrow_{\mathcal{R}_i} v$, hence $\beta\text{nf}(t) \rightarrow_{\mathcal{R}_i}^* \beta\text{nf}(v) \rightarrow_{\mathcal{R}_i}^* \beta\text{nf}(u)$ by (i). \square

Preservation of confluence is a direct consequence of the projection on β -normal forms.

Theorem 4.15 *Let $a : \Sigma \rightarrow \mathbb{N}$ be an arity and \mathcal{R} be an algebraic conditional rewrite system which respects a . If $\rightarrow_{\mathcal{R}}$ is confluent on \mathcal{AN}_a , then $\rightarrow_{\beta \cup \mathcal{R}}$ is confluent on \mathcal{AN}_a .*

PROOF. Let s, t, u such that $s \in \mathcal{AN}_a$ and $u \leftarrow_{\beta \cup \mathcal{R}}^* s \rightarrow_{\beta \cup \mathcal{R}}^* t$. By two applications of Lemma 4.14 we get that $\beta\text{nf}(u) \leftarrow_{\mathcal{R}}^* \beta\text{nf}(s) \rightarrow_{\mathcal{R}}^* \beta\text{nf}(t)$, with moreover $\beta\text{nf}(s)$, $\beta\text{nf}(t)$ and $\beta\text{nf}(u) \in \mathcal{AN}_a$. Hence we conclude by $\rightarrow_{\mathcal{R}}$ -confluence on \mathcal{AN}_a . In diagrammatic form,



\square

were already made in Section 4.2 when considering possibly non left-linear rewriting on weakly β -normalizing terms.

The following example presents rules which either are not algebraic or do not respect the arity prescribed by left-hand sides. With these rules property (6) fails and $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$ is not confluent whereas $\rightarrow_{\mathcal{R}}$ and $\rightarrow_{\beta \cup \mathcal{R}}$ are confluent.

Example 5.2 *With the conditional rewrite systems (7), (8), (9) and (10) below,*

(i) *the relations $\rightarrow_{\mathcal{R}}$ and $\rightarrow_{\beta \cup \mathcal{R}}$ are confluent,*

(ii) *property (6) is not satisfied and the relation $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$ is not confluent.*

$$g \ x \ y \mapsto x \ y \qquad g \ x \ c = d \supset f \ x \mapsto a \ x \qquad f \ x \mapsto b \ x \qquad (7)$$

$$x \ c = d \supset f \ x \mapsto a \ x \qquad f \ x \mapsto b \ x \qquad (8)$$

$$id \ x \ c = d \supset f \ x \mapsto a \ x \qquad f \ x \mapsto b \ x \qquad (9)$$

$$h \ x \ y \mapsto id \ x \ y \qquad h \ x \ c = d \supset f \ x \mapsto a \ x \qquad f \ x \mapsto b \ x \qquad (10)$$

where id is defined by $id \ x \mapsto x$.

PROOF.

(i) Since the symbol d is not defined, these systems lead to normal conditional rewrite relations. Since they are left-linear and semi-closed, we can apply Theorem 4.7, and deduce the confluence of $\rightarrow_{\beta \cup \mathcal{R}}$ from the confluence of $\rightarrow_{\mathcal{R}}$. Since they are left-linear systems, if their critical pairs are unfeasible, we can obtain the confluence of $\rightarrow_{\mathcal{R}}$ by Theorem 3.12. Each system has a unique conditional rule and a unique critical pair, issued from the root superposition of this rule with $f \ x \mapsto b \ x$. In each case, the number of occurrences of the symbol c in a term is preserved by $\rightarrow_{\mathcal{R}}$. Moreover, for each instantiation of the conditional rule, the instantiated left-hand side of the condition contains at least one occurrence of c . It follows that it cannot reduce to d , and that the critical pair is unfeasible. Therefore, we obtain the confluence of $\rightarrow_{\mathcal{R}}$ by Theorem 3.12 and we deduce the confluence of $\rightarrow_{\beta \cup \mathcal{R}}$ thanks to Theorem 4.7.

(ii) In each case, the step $f \ \lambda x.d \rightarrow_{\mathcal{R}(\beta)} a \ \lambda x.d$ is not in $\rightarrow_{\beta}^* \rightarrow_{\mathcal{R}}^* \leftarrow_{\beta}^*$ and the following peak is unjoinable

$$a \ \lambda x.d \quad \leftarrow_{\mathcal{R}(\beta)} \quad f \ \lambda x.d \quad \rightarrow_{\mathcal{R}(\beta)} \quad b \ \lambda x.d .$$

□

Note that systems (7) and (8) contain respectively a right-hand side and a condition which are not algebraic, and that systems (9) and (10) contain respectively a right-hand side and a condition that do not respect the arity of id imposed by the rewrite rule $id \ x \mapsto x$.

Note also that (6) is reminiscent of a property required on the substitution calculus used in [OR94]. This would require to see \rightarrow_{β} as the substitution calculus. But this does not fit in our framework, in particular because we consider \rightarrow_{β} and rewriting at the same level. Moreover, the substitution calculus used in [OR94] is required to be complete (i.e. strongly normalizing and confluent), which is not the case here for \rightarrow_{β} .

Outline. We begin in Section 5.1 by the extension of Theorem 4.6 to β -conditional rewriting for left-linear and semi-closed systems. In this case, preservation of confluence only holds on terms respecting an arity (namely *conditionally $(\mathcal{R}, \mathbf{a})$ -stable terms*, see Definition 5.5). This is an extra hypothesis compared to the results of Section 4.1. Then, in Section 5.2 we consider the case of Theorem 4.15. It directly extends to β -conditional rewriting. In both cases, we assume that rules are algebraic and respect an arity. In each case our assumptions ensure that the results of Section 4 apply, hence that $\rightarrow_{\beta \cup \mathcal{R}}$ is confluent whenever $\rightarrow_{\mathcal{R}}$ is confluent. Hence, using Proposition 5.1 we deduce the confluence of $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$ from the confluence of $\rightarrow_{\mathcal{R}}$ and property (6).

Remarks. In [BKR06], we have shown (6) by using a stratification of $\rightarrow_{\mathcal{R}(\beta)}$ in which, instead of having $\rightarrow_{\mathcal{R}(\beta)_0} = \emptyset$ as in Definition 2.9, we had $\rightarrow_{\mathcal{R}(\beta)_0} = \rightarrow_{\beta}$ (it is easy to show that these two base cases induce the same relation $\rightarrow_{\mathcal{R}(\beta)}$).

We proceed here in a slightly different and more general way. We show (6) with $\rightarrow_{\mathcal{R}(\beta)_0} = \emptyset$ and use the following intermediate property: for all $i \in \mathbb{N}$,

$$\begin{array}{ccc}
 \mathbf{t} & \xrightarrow{\beta \cup \mathcal{R}(\beta)_i} & \mathbf{u} \\
 \beta \downarrow * & * & \downarrow \beta \\
 \mathbf{t}' & \xrightarrow[\mathcal{R}_i]{*} & \mathbf{u}'
 \end{array} \tag{11}$$

5.1 Confluence for left-linear semi-closed systems

This section is devoted to the proof of (11) for left-linear semi-closed systems. Using Proposition 5.1 and Theorem 4.6, we then easily deduce the confluence of $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$ when $\rightarrow_{\mathcal{R}}$ is confluent. We postpone the proof of (11) until Lemma 5.9, Section 5.1.2. The material used in the proof is presented and motivated in Section 5.1.1 below.

5.1.1 Preliminaries

The proof of (11) involves some intermediate lemmas and the extension of $(\mathcal{R}, \mathbf{a})$ -stable sets of terms to conditional rewriting. In order to motivate them, we sketch some steps of the proof. Property (11) is proved by induction on $i \in \mathbb{N}$. Assuming the property for $i \in \mathbb{N}$, we discuss it for $i + 1$. We reason by induction on the length of the derivation $\mathbf{t} \xrightarrow{*}_{\beta \cup \mathcal{R}(\beta)_{i+1}} \mathbf{u}$. We present the ingredients used in the different steps of this induction.

The base case. In the base case, we have $\mathbf{t} \xrightarrow{\beta \cup \mathcal{R}(\beta)_{i+1}} \mathbf{u}$ in one step. The case of $\mathbf{t} \rightarrow_{\beta} \mathbf{u}$ is trivial: take $\mathbf{t}' =_{\text{def}} \mathbf{u}' =_{\text{def}} \mathbf{u}$. The case of $\mathbf{t} \xrightarrow{\mathcal{R}(\beta)_{i+1}} \mathbf{u}$ is more involved. We show

$$\begin{array}{ccc}
 \mathbf{t} & \xrightarrow{\mathcal{R}(\beta)_{i+1}} & \mathbf{u} \\
 \beta \downarrow * & * & \downarrow \beta \\
 \mathbf{t}' & \xrightarrow[\mathcal{R}_{i+1}]{*} & \mathbf{u}'
 \end{array} \tag{12}$$

Consider a rule $\vec{d} = \vec{c} \supset l \mapsto_{\mathcal{R}} r$. Recall from Example 5.2 that it must be algebraic and respect an arity. Hence every β -redex occurring in $\vec{d}\sigma$ or $\vec{c}\sigma$ also occurs in $l\sigma$. Then, property (12)

means that there is a β -reduction starting from $l\sigma$ that reduces these redexes and produce a substitution σ' such that

$$l\sigma \rightarrow_{\beta}^* l\sigma' \rightarrow_{\mathcal{R}_{i+1}} r\sigma' \leftarrow_{\beta}^* r\sigma.$$

In other words, if the conditions are satisfied with σ and $\rightarrow_{\beta \cup \mathcal{R}(\beta)_i}$ (i.e. $\vec{d}\sigma \downarrow_{\beta \cup \mathcal{R}(\beta)_i} \vec{c}$, recall that \vec{c} are closed terms since \mathcal{R} is semi-closed), then they are satisfied with σ' and $\rightarrow_{\mathcal{R}_i}$ (i.e. $\vec{d}\sigma' \downarrow_{\mathcal{R}_i} \vec{c}$). Let us look at this more precisely. Assume that $\vec{d}\sigma \downarrow_{\beta \cup \mathcal{R}(\beta)_i} \vec{c}$. Hence there are terms \vec{v} such that

$$\vec{d}\sigma \rightarrow_{\beta \cup \mathcal{R}(\beta)_i}^* \vec{v} \leftarrow_{\beta \cup \mathcal{R}(\beta)_i}^* \vec{c}.$$

By induction hypothesis on i , we get terms \vec{w} and \vec{v}' such that

$$\begin{array}{ccc} \vec{d}\sigma & \xrightarrow[\ast]{\beta \cup \mathcal{R}(\beta)_i} & \vec{v} \xleftarrow[\ast]{\beta \cup \mathcal{R}(\beta)_i} \vec{c} \\ \beta \downarrow \ast & & \ast \downarrow \beta \\ \vec{w} & \xrightarrow[\ast]{\mathcal{R}_i} & \vec{v}' \end{array}$$

In order to conclude, we need a substitution σ' such that $\sigma \rightarrow_{\beta}^* \sigma'$ and $\vec{w} \rightarrow_{\beta}^* \vec{d}\sigma'$. Using the algebraicity of \vec{d} , this follows from Proposition 5.4, which is stated and proved below. The remaining of the proof uses the commutation of \rightarrow_{β} and $\rightarrow_{\mathcal{R}_i}$ (Lemma 4.5) and relies on the semi-closure and the right-applicativity of \mathcal{R} (which follows from its algebraicity). See the proof of Lemma 5.9 in Section 5.1.2 for details.

We need to show that if t is an algebraic term such that $t\sigma \rightarrow_{\beta}^* v$, then there is a substitution σ' such that $\sigma \rightarrow_{\beta}^* \sigma'$ and $v \rightarrow_{\beta}^* t\sigma'$. This is provided by the two following technical propositions. The first one is a generalization of the diamond property of \triangleright_{β} . It is a direct consequence of Lemma 3.2 in [Tak95].

Proposition 5.3 *Let $n \geq 0$ and assume that s, s_1, \dots, s_n are terms such that $s \triangleright_{\beta} s_i$ for all $i \in \{1, \dots, n\}$. Then there is a term s' such that $s \triangleright_{\beta} s'$ and $s_i \triangleright_{\beta} s'$ for all $i \in \{1, \dots, n\}$.*

We deduce the following property. The proof of case (ii) uses the diamond property of \triangleright_{β} .

Proposition 5.4 *Let t_1, \dots, t_n be algebraic terms and let σ be a substitution.*

- (i) *If $t_i\sigma \triangleright_{\beta} u_i$ for all $i \in \{1, \dots, n\}$, then there is a substitution σ' such that $\sigma \triangleright_{\beta} \sigma'$ and $u_i \triangleright_{\beta} t_i\sigma'$ for all $i \in \{1, \dots, n\}$.*
- (ii) *If $t_i\sigma \rightarrow_{\beta}^* u_i$ for all $i \in \{1, \dots, n\}$, then there is a substitution σ' such that $\sigma \rightarrow_{\beta}^* \sigma'$ and $u_i \rightarrow_{\beta}^* t_i\sigma'$ for all $i \in \{1, \dots, n\}$.*

Note that the terms t_1, \dots, t_n need not be linear.

PROOF.

- (i) Since t_i is algebraic, every occurrence of a β -redex in t_i is of the form $p.d$ where p is an occurrence of a variable x in t_i . Since \triangleright_{β} is reflexive, for each $i \in \{1, \dots, n\}$, each $x \in FV(t_i)$ and each $p \in \text{Occ}(x, t_i)$, there is a term $s_{(i,x,p)}$ such that

$$t_i\sigma|_p = \sigma(x) \triangleright_{\beta} s_{(i,x,p)}$$

and for all $i \in \{1, \dots, n\}$,

$$u_i = t_i[p \leftarrow s_{(i,x,p)} \mid x \in \text{FV}(t_i) \wedge p \in \text{Occ}(x, t_i)].$$

By Proposition 5.3, for all $x \in \text{FV}(t_1, \dots, t_n)$, there is a term v_x such that $\sigma(x) \triangleright_\beta v_x$ and $s_{(i,x,p)} \triangleright_\beta v_x$ for all $i \in \{1, \dots, n\}$ and all $p \in \text{Occ}(x, t_i)$. Therefore, for all $i \in \{1, \dots, n\}$ we have

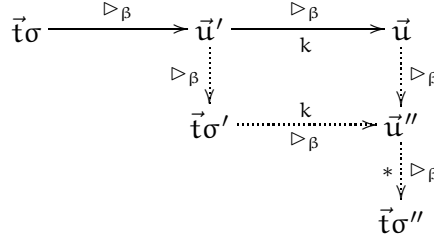
$$u_i \triangleright_\beta t_i[p \leftarrow v_x \mid x \in \text{FV}(t_i) \wedge p \in \text{Occ}(x, t_i)].$$

Let σ' be the substitution of same domain as σ such that $\sigma'(x) = v_x$ for all $x \in \text{FV}(t_1, \dots, t_n)$ and $\sigma'(x) = \sigma(x)$ for all $x \notin \text{FV}(t_1, \dots, t_n)$. Then we have $\sigma \triangleright_\beta \sigma'$ and $u_i \triangleright_\beta t_i \sigma'$ for all $i \in \{1, \dots, n\}$.

- (ii) By induction on $k \in \mathbb{N}$, we show that if $t_i \sigma \triangleright_\beta^k u_i$ for all $i \in \{1, \dots, n\}$, then there is σ' such that $\sigma \triangleright_\beta^* \sigma'$ and $u_i \triangleright_\beta^* t_i \sigma'$ for all $i \in \{1, \dots, n\}$.

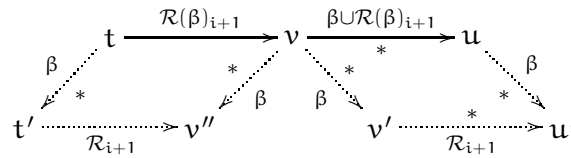
The base case $t_i \sigma \triangleright_\beta^0 u_i$ for all $i \in \{1, \dots, n\}$ is trivial. For the induction case, there are u'_1, \dots, u'_n such that $t_i \sigma \triangleright_\beta^k u'_i \triangleright_\beta u_i$ for all $i \in \{1, \dots, n\}$. Then, by (i), there is σ' such that $\sigma \triangleright_\beta \sigma'$ and $u'_i \triangleright_\beta t_i \sigma'$ for all $i \in \{1, \dots, n\}$. Since \triangleright_β satisfies the diamond property (Proposition 5.3), for all $i \in \{1, \dots, n\}$ there is u''_i such that $t_i \sigma' \triangleright_\beta^k u''_i \triangleleft_\beta u_i$, and by induction hypothesis on k , there is σ'' such that $\sigma' \triangleright_\beta^* \sigma''$ and $u''_i \triangleright_\beta^* t_i \sigma''$ for all $i \in \{1, \dots, n\}$. We deduce that $u_i \triangleright_\beta^* t_i \sigma''$ for all $i \in \{1, \dots, n\}$.

In diagrammatic form:

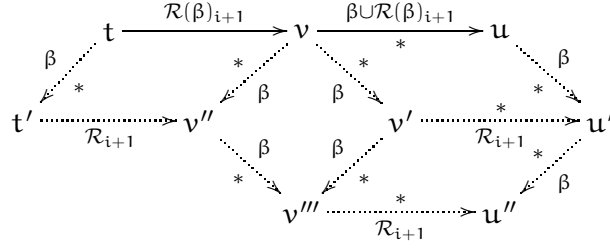


□

The induction case. In the induction case, we have $t \xrightarrow{*}_{\beta \cup \mathcal{R}(\beta)_{i+1}} u$ in more than one step. Hence, this derivation can be written as $t \xrightarrow{\beta \cup \mathcal{R}(\beta)_{i+1}} v \xrightarrow{*}_{\beta \cup \mathcal{R}(\beta)_{i+1}} u$ for some v . If $t \xrightarrow{\beta} v$, then we easily conclude by induction hypothesis on $v \xrightarrow{*}_{\beta \cup \mathcal{R}(\beta)_{i+1}} u$. Otherwise, we have $t \xrightarrow{\mathcal{R}(\beta)_{i+1}} v$ and things get more involved. Using the induction hypothesis on $v \xrightarrow{*}_{\beta \cup \mathcal{R}(\beta)_{i+1}} u$ and the discussion of the above paragraph for $t \xrightarrow{\mathcal{R}(\beta)_{i+1}} v$, we arrive at the following situation:



Using the confluence of \rightarrow_β and the commutation of \rightarrow_β with $\rightarrow_{\mathcal{R}_i}$ (Lemma 4.5), we get



In order to conclude we use the following property: for all $i \in \mathbb{N}$,

$$\begin{array}{ccc}
 t & \xrightarrow{\beta \cup \mathcal{R}_i} & u \\
 \beta \downarrow * & & * \downarrow \beta \\
 t' & \xrightarrow{\mathcal{R}_i} & u'
 \end{array} \quad (13)$$

The intricate case of property (13) is when there is an \mathcal{R}_i -step followed by a β -step:

$$t \rightarrow_{\mathcal{R}_i} v \rightarrow_\beta u.$$

In this case, we have to make sure that the step $t \rightarrow_{\mathcal{R}_i} v$ did not create the β -redex contracted in $v \rightarrow_\beta u$. As seen in Section 3.1, this follows from arity assumptions on terms.

We therefore use terms whose arity is compatible with that of the rewrite system. We need this property to be preserved by $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$, but also by the conditions of rewrite rules: given a semi-closed rule $\vec{d} = \vec{c} \supset l \mapsto_{\mathcal{R}} r$ and a substitution σ , if $l\sigma$ respects $\mathbf{a} : \Sigma \rightarrow \mathbb{N}$, then the terms $r\sigma, \vec{d}\sigma$ should also respect \mathbf{a} . This is the case when r, \vec{d} are algebraic and respect \mathbf{a} . Moreover, in the following we have to make sure that every term at hand satisfy these properties. In particular, if we have a rule $\vec{d} = \vec{c} \supset l \mapsto r$ such that $l\sigma$ and all its reducts respect an arity \mathbf{a} , this has to be the case of $\vec{d}\sigma$ too (the case of \vec{c} follows from semi-closure). Hence, we consider sets of terms which are stable under the rewrite relation $\rightarrow_{\overline{\mathcal{R}}}$ issued from the rewrite system

$$\begin{aligned}
 \mapsto_{\overline{\mathcal{R}}} & \stackrel{\text{def}}{=} \{ (l, \mathbf{d}_i) \mid \mathbf{d}_1 = \mathbf{c}_1 \wedge \dots \wedge \mathbf{d}_n = \mathbf{c}_n \supset l \mapsto_{\mathcal{R}} r \quad \wedge \quad i \in \{1, \dots, n\} \} \\
 & \cup \{ (l, r) \mid \mathbf{d}_1 = \mathbf{c}_1 \wedge \dots \wedge \mathbf{d}_n = \mathbf{c}_n \supset l \mapsto_{\mathcal{R}} r \}.
 \end{aligned}$$

This motivates the following definition, which extends $(\mathcal{R}, \mathbf{a})$ -stability (Definition 3.6) to conditional rewriting.

Definition 5.5 (Conditionally $(\mathcal{R}, \mathbf{a})$ -Stable Terms) *Let $\mathbf{a} : \Sigma \rightarrow \mathbb{N}$ be an arity and \mathcal{R} be a conditional rewrite system. A set of terms S is conditionally $(\mathcal{R}, \mathbf{a})$ -stable if it is $(\overline{\mathcal{R}}, \mathbf{a})$ -stable.*

We now show (13). The proof of this property occupies Proposition 5.6 and Lemma 5.7. Note that we prove Proposition 5.6 for systems whose conditions need not be algebraic. However, this property may fail in presence of right-hand sides which either are not algebraic or do not respect the arity prescribed by the left-hand sides. Note also that we work on conditionally $(\mathcal{R}, \mathbf{a})$ -stable terms.

Proposition 5.6 *Let \mathcal{R} be a left-linear semi-closed system which is right-algebraic and respects $\mathbf{a} : \Sigma \rightarrow \mathbb{N}$, and let $S \subseteq \Lambda(\Sigma)$ be conditionally $(\mathcal{R}, \mathbf{a})$ -stable. For all $i \in \mathbb{N}$ and all $t, u, v \in S$, if*

$t \rightarrow_{\mathcal{R}_i} u \triangleright_{\beta} v$, then there are t' and v' such that $t \triangleright_{\beta} t' \rightarrow_{\mathcal{R}_i}^* v' \triangleleft_{\beta} v$:

$$\begin{array}{ccccc}
 t & \xrightarrow{\mathcal{R}_i} & u & \xrightarrow{\triangleright_{\beta}} & v \\
 \downarrow \triangleright_{\beta} & & & & \downarrow \triangleright_{\beta} \\
 t' & \xrightarrow[\mathcal{R}_i]{*} & & & v'
 \end{array}$$

PROOF. The base case $i = 0$ is trivial, and we assume $i > 0$. We reason by induction on t using Lemma 2.3.

$t = \lambda \vec{x}.x t_1 \dots t_n$. In this case, $u = \lambda \vec{x}.x u_1 \dots u_n$ with $(t_1, \dots, t_n) \rightarrow_{\mathcal{R}_i} (u_1, \dots, u_n)$. Moreover, $v = \lambda \vec{x}.x v_1 \dots v_n$ with $(u_1, \dots, u_n) \triangleright_{\beta} (v_1, \dots, v_n)$. By induction hypothesis, there are (t'_1, \dots, t'_n) and (v'_1, \dots, v'_n) such that

$$(t_1, \dots, t_n) \triangleright_{\beta} (t'_1, \dots, t'_n) \rightarrow_{\mathcal{R}_i}^* (v'_1, \dots, v'_n) \triangleleft_{\beta} (v_1, \dots, v_n).$$

It follows that

$$\lambda \vec{x}.x t_1 \dots t_n \triangleright_{\beta} \lambda \vec{x}.x t'_1 \dots t'_n \rightarrow_{\mathcal{R}_i}^* \lambda \vec{x}.x v'_1 \dots v'_n \triangleleft_{\beta} \lambda \vec{x}.x v_1 \dots v_n.$$

$t = \lambda \vec{x}.f t_1 \dots t_n$. If $u = \lambda \vec{x}.f u_1 \dots u_n$ with

$$(t_1, \dots, t_n) \rightarrow_{\mathcal{R}_i} (u_1, \dots, u_n),$$

then $v = \lambda \vec{x}.f v_1 \dots v_n$ with $(u_1, \dots, u_n) \triangleright_{\beta} (v_1, \dots, v_n)$ and we reason as in the previous case.

Otherwise, there is a rule $\vec{d} = \vec{c} \supset l \mapsto_{\mathcal{R}} r$, a substitution σ and $k \in \{1, \dots, n\}$ such that $t = \lambda \vec{x}.l \sigma t_{k+1} \dots t_n$ and $u = \lambda \vec{x}.r \sigma t_{k+1} \dots t_n$. As t and \mathcal{R} respect α , we have $n = k$, hence $t = \lambda \vec{x}.l \sigma$, $u = \lambda \vec{x}.r \sigma$ and $v = \lambda \vec{x}.w$ with $r \sigma \triangleright_{\beta} w$.

Since r is algebraic, by Proposition 5.4.(i) there is σ' such that $\sigma \triangleright_{\beta} \sigma'$ and $w \triangleright_{\beta} r \sigma'$. As l is linear, by Proposition 4.4 we have $l \sigma \triangleright_{\beta} l \sigma'$. It remains to show that $l \sigma' \rightarrow_{\mathcal{R}_i} r \sigma'$. Since $l \sigma \rightarrow_{\mathcal{R}_i} r \sigma$, there are terms \vec{v} such that $\vec{d} \sigma \rightarrow_{\mathcal{R}_{i-1}}^* \vec{v} \leftarrow_{\mathcal{R}_{i-1}}^* \vec{c}$. Since $\vec{d} \sigma \rightarrow_{\beta}^* \vec{d} \sigma'$, by Lemma 4.5, we obtain terms \vec{v}' such that

$$\vec{d} \sigma \rightarrow_{\beta}^* \vec{d} \sigma' \rightarrow_{\mathcal{R}_{i-1}}^* \vec{v}' \leftarrow_{\beta}^* \vec{v} \leftarrow_{\mathcal{R}_{i-1}}^* \vec{c}.$$

Since terms \vec{c} are applicative and closed, they are algebraic, and since \mathcal{R} is right-algebraic, terms \vec{v} are also algebraic, hence in β -normal form. It follows that $\vec{v} = \vec{v}'$, hence that $\vec{d} \sigma' \downarrow_{\mathcal{R}_{i-1}} \vec{c}$, and we deduce that $l \sigma' \rightarrow_{\mathcal{R}_i} r \sigma'$.

$t = \lambda \vec{x}.(\lambda x.t_0) t_1 \dots t_n$ ($n \geq 1$). Then u is of the form $u = \lambda \vec{x}.(\lambda x.u_0) u_1 \dots u_n$ and we have $(t_0, \dots, t_n) \rightarrow_{\mathcal{R}_i} (u_0, \dots, u_n)$. If $v = \lambda \vec{x}.(\lambda x.v_0) v_1 \dots v_n$ with

$$(u_0, \dots, u_n) \triangleright_{\beta} (v_0, \dots, v_n),$$

then we conclude by induction hypothesis, as in the first case.

Otherwise, $v = \lambda \vec{x}.v_0[v_1/x]v_2 \dots v_n$ with $(u_0, \dots, u_n) \triangleright_{\beta} (v_0, \dots, v_n)$. By induction hypothesis, we have

$$(t_0, \dots, t_n) \triangleright_{\beta} (t'_0, \dots, t'_n) \rightarrow_{\mathcal{R}_i}^* (v'_0, \dots, v'_n) \triangleleft_{\beta} (v_0, \dots, v_n).$$

It follows that by using $(\triangleright\beta)$, $(\triangleright\text{APP})$ we have

$$\begin{array}{ccc} \lambda\vec{x}.(\lambda x.t_0)t_1 \dots t_n & & \lambda\vec{x}.v_0[v_1/x]v_2 \dots v_n \\ \nabla_\beta & & \nabla_\beta \\ \lambda\vec{x}.t'_0[t'_1/x]t'_2 \dots t'_n & \xrightarrow{*}_{\mathcal{R}_i} & \lambda\vec{x}.v'_0[v'_1/x]v'_2 \dots v'_n \end{array}$$

□

Lemma 5.7 *Let \mathcal{R} be a semi-closed left-linear right-algebraic system which respects $\mathbf{a} : \Sigma \rightarrow \mathbb{N}$, and let $S \subseteq \Lambda(\Sigma)$ be conditionally $(\mathcal{R}, \mathbf{a})$ -stable. For all $s, t \in S$, if $s \xrightarrow{*}_{\beta \cup \mathcal{R}_i} t$ then there are s', t' such that $s \xrightarrow{*}_\beta s' \xrightarrow{*}_{\mathcal{R}_i} t' \xleftarrow{*}_\beta t$:*

$$\begin{array}{ccc} s & \xrightarrow{\beta \cup \mathcal{R}_i} & t \\ \beta \downarrow * & * & \downarrow \beta \\ s' & \xrightarrow{*}_{\mathcal{R}_i} & t' \end{array}$$

PROOF. The proof is in three steps.

- (i) We show $\xrightarrow{*}_{\mathcal{R}_i} \triangleright_\beta \subseteq \triangleright_\beta \xrightarrow{*}_{\mathcal{R}_i} \triangleleft_\beta^*$ by induction on the number of \mathcal{R}_i -steps. Assume that $s \xrightarrow{*}_{\mathcal{R}_i} t' \xrightarrow{*}_{\mathcal{R}_i} t \triangleright_\beta u$. By Lemma 5.6, there are v and v' such that $t' \triangleright_\beta v \xrightarrow{*}_{\mathcal{R}_i} v' \triangleleft_\beta u$. By induction hypothesis, there are s' and s'' such that $s \triangleright_\beta s' \xrightarrow{*}_{\mathcal{R}_i} s'' \triangleleft_\beta v$. Then, by Lemma 4.5, there is t'' such that $s'' \xrightarrow{*}_{\mathcal{R}_i} t'' \triangleleft_\beta v'$. Thus, $s \triangleright_\beta s' \xrightarrow{*}_{\mathcal{R}_i} t'' \triangleleft_\beta^* u$.
- (ii) We show $\xrightarrow{*}_{\mathcal{R}_i} \triangleright_\beta^* \subseteq \triangleright_\beta^* \xrightarrow{*}_{\mathcal{R}_i} \triangleleft_\beta^*$ by induction on the number of \triangleright_β -steps. Assume that $s \xrightarrow{*}_{\mathcal{R}_i} t \triangleright_\beta u' \triangleright_\beta^* u$. After (i), there are s' and t' such that $s \triangleright_\beta s' \xrightarrow{*}_{\mathcal{R}_i} t' \triangleleft_\beta^* u'$. By the diamond property of \triangleright_β , there is v such that $t' \triangleright_\beta^* v \triangleleft_\beta^* u$, where $t' \triangleright_\beta^* v$ is no longer than $u' \triangleright_\beta^* u$. Hence, by induction hypothesis, there are s'' and t'' such that $s' \triangleright_\beta^* s'' \xrightarrow{*}_{\mathcal{R}_i} t'' \triangleleft_\beta^* v$. Therefore, $s \triangleright_\beta^* s'' \xrightarrow{*}_{\mathcal{R}_i} t'' \triangleleft_\beta^* u$.
- (iii) We prove $(\triangleright_\beta \cup \xrightarrow{*}_{\mathcal{R}_i})^* \subseteq \triangleright_\beta^* \xrightarrow{*}_{\mathcal{R}_i} \triangleleft_\beta^*$ by induction on the length of $(\triangleright_\beta \cup \xrightarrow{*}_{\mathcal{R}_i})^*$. Assume that $s \xrightarrow{\triangleright_\beta \cup \mathcal{R}_i} t \xrightarrow{*}_{\triangleright_\beta \cup \mathcal{R}_i} u$. There are two cases. First, $s \triangleright_\beta t$. This case follows directly from the induction hypothesis. Second, $s \xrightarrow{*}_{\mathcal{R}_i} t$. By induction hypothesis, there are t' and u' such that $t \triangleright_\beta^* t' \xrightarrow{*}_{\mathcal{R}_i} u' \triangleleft_\beta^* u$. After (ii), there are s' and t'' such that $s \triangleright_\beta^* s' \xrightarrow{*}_{\mathcal{R}_i} t'' \triangleleft_\beta^* t'$. Finally, by Lemma 4.5, there is u'' such that $t'' \xrightarrow{*}_{\mathcal{R}_i} u'' \triangleleft_\beta^* u'$. Hence, $s \triangleright_\beta^* s' \xrightarrow{*}_{\mathcal{R}_i} u'' \triangleleft_\beta^* u$.

We conclude by the fact that $\triangleright_\beta^* = \xrightarrow{*}_\beta$. □

Remark. Note that β -reduction is the only way to obtain a term not respecting \mathbf{a} from a term respecting it. For instance, with $\mathbf{a}(\text{id}) = 1$ the term $(\lambda x.x \ y \ y)\text{id}$ respects \mathbf{a} whereas $\text{id} \ y \ y$ does not respect \mathbf{a} .

Proposition 5.8 *Let \mathcal{R} be an algebraic conditional rewrite system and $t \in \Lambda(\Sigma)$ that both respect $\mathbf{a} : \Sigma \rightarrow \mathbb{N}$. If $t \xrightarrow{*}_{\mathcal{R}(\beta)} u$ then u respects \mathbf{a} .*

PROOF. We reason by induction on t , using Lemma 2.3. The only case which does not directly follow from the induction hypothesis is when $t = \lambda\vec{x}.ft_1 \dots t_n$ and there is a rule $\vec{d} = \vec{c} \supset l \mapsto_{\mathcal{R}} r$, a substitution σ and $k \in \{1, \dots, n\}$ such that $t = \lambda\vec{x}.l\sigma t_{k+1} \dots t_n$. Since t and \mathcal{R} respect \mathbf{a} , we have $k = n$. Hence $u = \lambda\vec{x}.r\sigma$ and u respects \mathbf{a} since r is an algebraic term that respects \mathbf{a} . □

5.1.2 Confluence of beta-reduction with beta-conditional rewriting

We now have all we need to show property (11). As seen in Example 5.2, rules have to be algebraic and arity compliant. We reason by induction on $i \in \mathbb{N}$.

Lemma 5.9 *Let \mathcal{R} be a semi-closed left-linear algebraic system which respects $\mathbf{a} : \Sigma \rightarrow \mathbb{N}$, and let $S \subseteq \Lambda(\Sigma)$ be conditionally $(\mathcal{R}, \mathbf{a})$ -stable. For all $\mathbf{t}, \mathbf{u} \in S$, if $\mathbf{t} \rightarrow_{\beta \cup \mathcal{R}(\beta)_i}^* \mathbf{u}$ then there are \mathbf{t}' , \mathbf{u}' such that $\mathbf{t} \rightarrow_{\beta}^* \mathbf{t}' \rightarrow_{\mathcal{R}_i}^* \mathbf{u}' \leftarrow_{\beta}^* \mathbf{u}$:*

$$\begin{array}{ccc}
 \mathbf{t} & \xrightarrow{\beta \cup \mathcal{R}(\beta)_i} & \mathbf{u} \\
 \beta \downarrow^* & * & \downarrow^* \beta \\
 \mathbf{t}' & \xrightarrow{\mathcal{R}_i} & \mathbf{u}'
 \end{array} \tag{14}$$

PROOF. We show (14) by induction on $i \in \mathbb{N}$. The base case $i = 0$ is trivial. We assume that the property holds for $i \geq 0$ and show it for $i + 1$. The proof is in two steps.

(i) We begin by showing that diagram (15) commutes:

$$\begin{array}{ccc}
 \mathbf{t} & \xrightarrow{\mathcal{R}(\beta)_{i+1}} & \mathbf{u} \\
 \beta \downarrow^* & * & \downarrow^* \beta \\
 \mathbf{t}' & \xrightarrow{\mathcal{R}_{i+1}} & \mathbf{u}'
 \end{array} \tag{15}$$

We reason by induction on \mathbf{t} , using Lemma 2.3. The only case that does not directly follow from the induction hypothesis is when $\mathbf{t} = \lambda \vec{x}. \mathbf{f} \mathbf{t}_1 \dots \mathbf{t}_n$ and there is a rule $\vec{d} = \vec{c} \supset \mathbf{l} \rightarrow_{\mathcal{R}} \mathbf{r}$, a substitution σ and $k \in \{1, \dots, n\}$ such that $\mathbf{t} = \lambda \vec{x}. \mathbf{l} \sigma \mathbf{t}_{k+1} \dots \mathbf{t}_n$ and $\mathbf{u} = \lambda \vec{x}. \mathbf{r} \sigma \mathbf{t}_{k+1} \dots \mathbf{t}_n$ with $\mathbf{l} \sigma \rightarrow_{\mathcal{R}(\beta)_{i+1}} \mathbf{r} \sigma$. Since \mathbf{t} and \mathcal{R} respect \mathbf{a} , we have $k = n$, hence $\mathbf{u} = \lambda \vec{x}. \mathbf{r} \sigma$.

To deduce (15), it remains to show that there is a substitution σ' such that

$$\mathbf{l} \sigma \rightarrow_{\beta}^* \mathbf{l} \sigma' \rightarrow_{\mathcal{R}_{i+1}} \mathbf{r} \sigma' \leftarrow_{\beta}^* \mathbf{r} \sigma .$$

Since $\mathbf{l} \sigma \rightarrow_{\mathcal{R}(\beta)_{i+1}} \mathbf{r} \sigma$, there are terms \vec{v} such that $\vec{d} \sigma \rightarrow_{\beta \cup \mathcal{R}(\beta)_i}^* \vec{v} \leftarrow_{\beta \cup \mathcal{R}(\beta)_i}^* \vec{c}$. By induction hypothesis on i , there are terms \vec{w} and \vec{v}' such that

$$\vec{d} \sigma \rightarrow_{\beta}^* \vec{w} \rightarrow_{\mathcal{R}_i}^* \vec{v}' \leftarrow_{\beta}^* \vec{v} .$$

By Proposition 5.4.(ii), as terms \vec{d} are algebraic there is a substitution σ' such that $\sigma \rightarrow_{\beta}^* \sigma'$ and $\vec{w} \rightarrow_{\beta}^* \vec{d} \sigma'$. By Lemma 4.5 (commutation of $\rightarrow_{\mathcal{R}_i}$ with \rightarrow_{β}), we obtain terms \vec{v}' such that $\vec{d} \sigma' \rightarrow_{\mathcal{R}_i}^* \vec{v}' \leftarrow_{\beta}^* \vec{v}$. It follows that

$$\vec{d} \sigma \rightarrow_{\beta}^* \vec{d} \sigma' \rightarrow_{\mathcal{R}_i}^* \vec{v}' \leftarrow_{\beta}^* \vec{v} \leftarrow_{\beta \cup \mathcal{R}(\beta)_i}^* \vec{c} .$$

Since terms \vec{c} are algebraic and \mathcal{R} is right-applicative, every reduct of \vec{c} by $\rightarrow_{\mathcal{R}(\beta)}$ is β -normal. We thus have $\vec{v}' = \vec{v}$ and by induction hypothesis on i we deduce that $\vec{c} \rightarrow_{\mathcal{R}_i}^* \vec{v}$. It follows that $\vec{d} \sigma' \downarrow_{\mathcal{R}_i} \vec{c}$, hence $\mathbf{l} \sigma' \rightarrow_{\mathcal{R}_{i+1}} \mathbf{r} \sigma'$. We have $\mathbf{l} \sigma \rightarrow_{\beta}^* \mathbf{l} \sigma'$ and $\mathbf{r} \sigma \rightarrow_{\beta}^* \mathbf{r} \sigma'$ since $\sigma \rightarrow_{\beta}^* \sigma'$, hence

$$\mathbf{t} \rightarrow_{\beta}^* \lambda \vec{x}. \mathbf{l} \sigma' \rightarrow_{\mathcal{R}_{i+1}} \lambda \vec{x}. \mathbf{r} \sigma' \leftarrow_{\beta}^* \mathbf{u} .$$

(ii) We now show (14) by induction on the length of $t \rightarrow_{\beta \cup \mathcal{R}(\beta)_{i+1}}^* u$. Assume that

$$t \rightarrow_{\beta \cup \mathcal{R}(\beta)_{i+1}} v \rightarrow_{\beta \cup \mathcal{R}(\beta)_{i+1}}^* u.$$

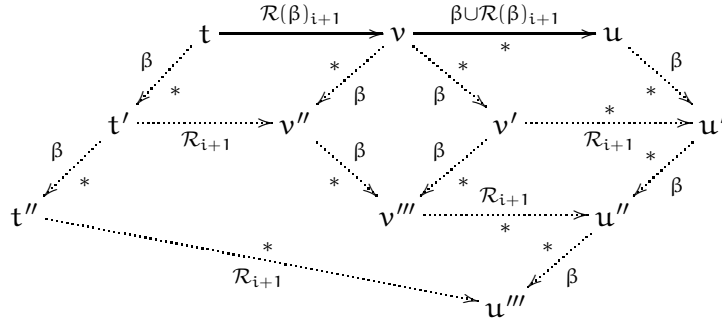
By induction hypothesis, there are v' and u' such that $v \rightarrow_{\beta}^* v' \rightarrow_{\mathcal{R}_{i+1}}^* u' \leftarrow_{\beta}^* u$ and there are two cases. If $t \rightarrow_{\beta} v$, then we are done since $t \rightarrow_{\beta}^* v'$.

Otherwise, we have $t \rightarrow_{\mathcal{R}(\beta)_{i+1}} u$. From (i), there are t' and v'' such that

$$t \rightarrow_{\beta}^* t' \rightarrow_{\mathcal{R}_{i+1}}^* v'' \leftarrow_{\beta}^* v.$$

Now, by confluence of \rightarrow_{β} , there is v''' such that $v'' \rightarrow_{\beta}^* v''' \leftarrow_{\beta}^* v'$. Commutation of \rightarrow_{β} and $\rightarrow_{\mathcal{R}_{i+1}}$ (Lemma 4.5) applied to $v''' \leftarrow_{\beta}^* v' \rightarrow_{\mathcal{R}_{i+1}}^* u'$ gives us a term u'' such that $v''' \rightarrow_{\mathcal{R}_{i+1}}^* u'' \leftarrow_{\beta}^* u'$. We thus have $t' \rightarrow_{\beta \cup \mathcal{R}_{i+1}}^* u''$ and by Lemma 5.7 there are t'' and u''' such that $t'' \rightarrow_{\beta}^* \rightarrow_{\mathcal{R}_{i+1}}^* \leftarrow_{\beta}^* u'''$. Therefore, $t \rightarrow_{\beta}^* \rightarrow_{\mathcal{R}_{i+1}}^* \leftarrow_{\beta}^* u$.

In diagrammatic form,



□

We easily deduce (6) from (14). We get the confluence of $\rightarrow_{\beta \cup \mathcal{R}}$ using Theorem 4.6. By Proposition 5.1, the confluence of $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$ follows from the confluence of $\rightarrow_{\mathcal{R}}$ on conditionally $(\mathcal{R}, \mathbf{a})$ -stable sets of terms.

Theorem 5.10 *Let \mathcal{R} be a semi-closed left-linear algebraic system which respects $\mathbf{a} : \Sigma \rightarrow \mathbb{N}$. Then, on any conditionally $(\mathcal{R}, \mathbf{a})$ -stable set of terms, if $\rightarrow_{\mathcal{R}}$ is confluent then so is $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$.*

PROOF. Since \mathcal{R} is semi-closed, left-linear and right-applicative, confluence of $\rightarrow_{\beta \cup \mathcal{R}}$ follows from confluence of $\rightarrow_{\mathcal{R}}$ by Theorem 4.6. We then conclude by Lemma 5.9 and Proposition 5.1, since conditionally $(\mathcal{R}, \mathbf{a})$ -stable sets of terms are closed under $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$. □

5.2 Confluence on weakly beta-normalizing terms

In this section, we extend to $\rightarrow_{\mathcal{R}(\beta)}$ the results of Section 4.2. The main point is to obtain the lemma corresponding to Lemma 4.14. Moreover, as in Section 5.1, for all $i \in \mathbb{N}$ we project $\rightarrow_{\mathcal{R}(\beta)_i}$ on $\rightarrow_{\mathcal{R}_i}$. We thus want to obtain the following property, which implies (6):

$$\begin{array}{ccc}
 t & \xrightarrow{\beta \cup \mathcal{R}(\beta)_i} & u \\
 \beta \downarrow^* & & \downarrow^* \beta \\
 \beta \text{nf}(t) & \xrightarrow[\mathcal{R}_i]{*} & \beta \text{nf}(u)
 \end{array} \tag{16}$$

We use the same tools as in Section 4.2. We consider weakly β -normalizing terms whose β -normal form respects the arity specified by rewrite rules, and we reason by induction on \succ . We also assume that rewrite rules are algebraic.

We denote by $\triangleright_{\mathcal{R}(\beta)}$ the nested parallelization of join β -conditional rewriting, defined similarly as in Definition 4.11. It satisfies Proposition 4.12 and Lemma 4.13.

We now show (16) using exactly the same method as for showing (4) in Lemma 4.14.

Lemma 5.11 *Let $\mathbf{a} : \Sigma \rightarrow \mathbb{N}$ be an arity and \mathcal{R} be an algebraic conditional rewrite system which respects \mathbf{a} . For all $i \in \mathbb{N}$, if $\mathbf{t} \in \mathcal{AN}_{\mathbf{a}}$ and $\mathbf{t} \rightarrow_{\beta \cup \mathcal{R}(\beta)_i}^* \mathbf{u}$, then $\mathbf{u} \in \mathcal{AN}_{\mathbf{a}}$ and $\beta \text{nf}(\mathbf{t}) \rightarrow_{\mathcal{R}_i}^* \beta \text{nf}(\mathbf{u})$.*

PROOF. We reason exactly as in the proof of Lemma 4.14. We prove the property by induction on $i \in \mathbb{N}$. In the induction case we show that for all $\mathbf{t} \in \mathcal{AN}_{\mathbf{a}}$,

$$\begin{array}{ccc}
 \mathbf{t} & \xrightarrow{\triangleright_{\mathcal{R}(\beta)_{i+1}}} & \mathbf{u} \\
 \beta \downarrow^* & & \downarrow^* \beta \\
 \beta \text{nf}(\mathbf{t}) & \xrightarrow{\triangleright_{\mathcal{R}_{i+1}}} & \beta \text{nf}(\mathbf{u})
 \end{array} \tag{17}$$

We reason by induction on \succ using Lemma 2.3. The only difference with the proof of Lemma 4.14 is the case where $\mathbf{t} = \lambda \vec{x}. \mathbf{f} \mathbf{t}_1 \dots \mathbf{t}_n$ and there is a rule $\vec{\mathbf{d}} = \vec{\mathbf{c}} \supset \mathbf{l} \mapsto \mathbf{r}$ such that $\mathbf{t} = \lambda \vec{x}. \mathbf{l} \sigma \vec{\mathbf{a}}$ and $\mathbf{u} = \lambda \vec{x}. \mathbf{r} \theta \vec{\mathbf{b}}$ with $\mathbf{l} \sigma \triangleright_{\mathcal{R}(\beta)_{i+1}} \mathbf{r} \theta$ and $\vec{\mathbf{d}} \sigma \downarrow_{\beta \cup \mathcal{R}(\beta)_i} \vec{\mathbf{c}} \sigma$. Exactly for the same reasons as in Lemma 4.14, we have $\vec{\mathbf{a}} = \vec{\mathbf{b}} = \emptyset$, $\mathbf{t} = \lambda \vec{x}. \mathbf{l} \sigma$ and $\mathbf{u} = \lambda \vec{x}. \mathbf{r} \sigma$. Moreover, $\beta \text{nf}(\mathbf{t}) = \lambda \vec{x}. \mathbf{l} \sigma'$ and $\beta \text{nf}(\mathbf{u}) = \lambda \vec{x}. \mathbf{r} \theta'$ with $\sigma' =_{\text{def}} \beta \text{nf}(\sigma)$ and $\theta' = \beta \text{nf}(\theta)$, and by induction hypothesis on \succ we have $\sigma' \triangleright_{\mathcal{R}_{i+1}} \theta'$. It remains to show that $\mathbf{l} \sigma' \triangleright_{\mathcal{R}_{i+1}} \mathbf{r} \theta'$. Because $\sigma' \triangleright_{\mathcal{R}_{i+1}} \theta'$, it suffices to prove that $\mathbf{l} \sigma' \rightarrow_{\mathcal{R}_{i+1}} \mathbf{r} \theta'$. Thus, we are done if we show that $\vec{\mathbf{d}} \sigma' \downarrow_{\mathcal{R}_i} \vec{\mathbf{c}} \sigma'$. Since $\vec{\mathbf{d}}$ and $\vec{\mathbf{c}}$ are algebraic, $\beta \text{nf}(\vec{\mathbf{d}} \sigma) = \vec{\mathbf{d}} \sigma'$ and $\beta \text{nf}(\vec{\mathbf{c}} \sigma) = \vec{\mathbf{c}} \sigma'$. Now, since $\vec{\mathbf{d}}$ is algebraic and respects \mathbf{a} , and since σ' respects \mathbf{a} , it follows that $\vec{\mathbf{d}} \sigma'$ respects \mathbf{a} . The same holds for $\vec{\mathbf{c}} \sigma'$. Hence we conclude by applying on $\vec{\mathbf{d}} \sigma \downarrow_{\beta \cup \mathcal{R}(\beta)_i} \vec{\mathbf{c}} \sigma$ the induction hypothesis on i . \square

We deduce the preservation of confluence.

Theorem 5.12 *Let $\mathbf{a} : \Sigma \rightarrow \mathbb{N}$ be an arity and \mathcal{R} be an algebraic conditional rewrite system which respects \mathbf{a} . If $\rightarrow_{\mathcal{R}}$ is confluent on $\mathcal{AN}_{\mathbf{a}}$, then $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$ is confluent on $\mathcal{AN}_{\mathbf{a}}$.*

PROOF. We can reason as described at the beginning of this section, using Proposition 5.1, Theorem 4.15 and Lemma 5.11. A direct proof is also possible, reasoning as for Theorem 4.15. \square

6 Orthonormal systems

In this section, we give a criterion ensuring the confluence of $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$ when conditions and right-hand sides possibly contain abstractions and active variables.

This criterion comes from peculiarities of orthogonality with conditional rewriting. As remarked in Section 3.2, a conditional critical pair can be *feasible* or not. In [Ohl02], it is remarked that results on the confluence of semi-equational and normal orthogonal conditional systems could be extended to systems that have no feasible critical pair. But the results obtained this way are not directly applicable, since proving unfeasibility of critical pairs may require confluence. An example of such situation is the following rewrite system.

Example 6.1 Consider the following two rules, taken from the system presented in Section 2.4.2:

$$\begin{aligned} > (\text{length } l) \ x = \text{false} &\supset \text{occ}(\text{cons } x \ o) (\text{node } y \ l) \mapsto \text{false} \\ > (\text{length } l) \ x = \text{true} &\supset \text{occ}(\text{cons } x \ o) (\text{node } y \ l) \mapsto \text{occ } o \ (\text{get } l \ x) \end{aligned}$$

The only conditional critical pair between them is

$$> (\text{length } l) \ x = \text{true} \wedge > (\text{length } l) \ x = \text{false} \supset (\text{false}, \text{occ } o \ (\text{get } l \ x))$$

The condition of this pair cannot be satisfied by a confluent relation. Hence, if $\rightarrow_{\beta \cup \mathcal{R}(\beta)_i}$ is confluent then we can reason as in Lemma 4.5 and obtain the confluence of $\rightarrow_{\beta \cup \mathcal{R}(\beta)_{i+1}}$.

In this section, we define a class of systems, called *orthonormal*, that allows to generalize this reasoning. As in Example 6.1, confluence can be shown stratified way: the confluence of $\rightarrow_{\beta \cup \mathcal{R}(\beta)_i}$ implies the unfeasibility of critical pairs w.r.t. $\rightarrow_{\beta \cup \mathcal{R}(\beta)_i}$, which in turn entails the confluence of the next stratum $\rightarrow_{\beta \cup \mathcal{R}(\beta)_{i+1}}$. We thus obtain the level confluence of $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$.

Rules of orthonormal systems can have λ -terms in their right-hand sides and conditions. Moreover, no arity assumption is made. Hence, orthonormality ensures the confluence of β -conditional rewriting combined to β -reduction when we cannot deduce it from the confluence of conditional rewriting (see Section 5).

Systems similar to orthonormal systems have already been studied in the first-order case [GM88, KW97]. It is worth relating orthonormal systems with approaches to conditional rewriting in which conditions are arbitrary predicates on terms. For first-order conditional rewriting this approach has been taken in [BK86]. It has been applied to λ -calculus [Tak93], and this is the way conditional rewrite rules are handled in the very expressive framework of CCERSs [GKK05]. Neither of these approaches can directly handle Example 6.1. In each case, confluence is proved under the assumption that the predicates used in conditions are stable by reduction, while proving this property in the case of Example 6.1 requires confluence.

A symbol $f \in \Sigma$ is *defined* if it is the head of the left-hand side of a rule.

Definition 6.2 (Orthonormal Systems) A conditional rewrite system \mathcal{R} is orthonormal if

- (i) it is left-linear;
- (ii) in every rule $\vec{d} = \vec{c} \supset l \mapsto_{\mathcal{R}} r$, the terms in \vec{c} are closed β -normal forms not containing defined symbols;
- (iii) for every critical pair

$$d_1 = c_1 \wedge \dots \wedge d_n = c_n \supset (s, t)$$

there exist distinct $i, j \in \{1, \dots, n\}$ such that $d_i = d_j$ and $c_i \neq c_j$.

Condition (ii) is a simple syntactic and decidable way to ensure that orthonormal systems are normal (recall from Remark 2.12 that normality is in general undecidable). As explained in Example 6.1, assuming the confluence of $\rightarrow_{\beta \cup \mathcal{R}(\beta)_i}$, condition (iii) implies the unfeasibility of critical pairs w.r.t. $\rightarrow_{\beta \cup \mathcal{R}(\beta)_i}$, hence the confluence of the next stratum $\rightarrow_{\beta \cup \mathcal{R}(\beta)_{i+1}}$. This entails the level confluence of $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$. We actually prove in Theorem 6.7 the *shallow* confluence of $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$, which is a stronger property (see Definition 2.15). Theorem 6.7 is thus an extension of the shallow confluence of orthogonal first-order normal conditional rewriting (Theorem 3.12) to orthonormal β -conditional rewriting.

The most important point w.r.t. the results of Section 5 is that orthonormal systems do not need to respect an arity nor to be algebraic.

Example 6.3 *The system presented in Section 2.4.2 is orthonormal.*

We now show that $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$ is shallow confluent when \mathcal{R} is orthonormal. This result is stated and proved in Theorem 6.7 below. We use some intermediate lemmas. The parallel moves property occupies Lemmas 6.5 and 6.6. We begin by showing that the confluence of $\rightarrow_{\beta \cup \mathcal{R}(\beta)_i}$ implies the commutation of \rightarrow_{β}^* and $\rightarrow_{\mathcal{R}(\beta)_{i+1}}^*$.

Lemma 6.4 *Let \mathcal{R} be an orthonormal system. For all $i \in \mathbb{N}$, if $\rightarrow_{\beta \cup \mathcal{R}(\beta)_i}$ is confluent then $\rightarrow_{\mathcal{R}(\beta)_{i+1}}$ commutes with \rightarrow_{β} :*

$$\begin{array}{ccc} \cdot & \xrightarrow{\mathcal{R}(\beta)_{i+1}} & \cdot \\ \beta \downarrow * & & * \downarrow \beta \\ \cdot & \xrightarrow{\mathcal{R}(\beta)_{i+1}} & \cdot \end{array}$$

PROOF. We reason as in Lemma 4.5. We show property (18) below and then deduce the commutation of $\rightarrow_{\mathcal{R}(\beta)_{i+1}}$ and \rightarrow_{β} using Lemma 2.18 and the fact that $\rightarrow_{\beta}^* = \triangleright_{\beta}^*$.

$$\begin{array}{ccc} t & \xrightarrow{\mathcal{R}(\beta)_{i+1}} & v \\ \triangleright_{\beta} \downarrow & & \downarrow \triangleright_{\beta} \\ u & \xrightarrow{\mathcal{R}(\beta)_{i+1}} & w \end{array} \quad (18)$$

The only difference with the proof of Lemma 4.5 is when $t \rightarrow_{\mathcal{R}(\beta)_{i+1}} v$ by contracting a rooted redex. In this case, there is a rule $\vec{d} = \vec{c} \supset l \mapsto_{\mathcal{R}} r$ and a substitution σ such that $t = l\sigma$ and $v = r\sigma$. We show that there is a term w such that $u \rightarrow_{\mathcal{R}(\beta)_{i+1}}^* w \triangleleft_{\beta} r\sigma$. As l is a non-variable linear algebraic term, there is a substitution σ' such that $\sigma \triangleright_{\beta} \sigma'$ and $l\sigma \triangleright_{\beta} l\sigma' = u$. Therefore we have $r\sigma \triangleright_{\beta} r\sigma'$. It remains to show that $l\sigma' \rightarrow_{\mathcal{R}(\beta)_{i+1}} r\sigma'$. Recall that $\vec{d}\sigma \rightarrow_{\beta \cup \mathcal{R}(\beta)_i}^* \vec{c}$. By assumption (confluence of $\rightarrow_{\beta \cup \mathcal{R}(\beta)_i}$), since $\vec{d}\sigma \rightarrow_{\beta}^* \vec{d}\sigma'$ there are \vec{v} such that $\vec{d}\sigma' \rightarrow_{\beta \cup \mathcal{R}(\beta)_i}^* \vec{v} \leftarrow_{\beta \cup \mathcal{R}(\beta)_i}^* \vec{c}$. But \vec{c} are $\beta \cup \mathcal{R}(\beta)$ -normal forms, hence $\vec{v} = \vec{c}$. We conclude that $l\sigma' \rightarrow_{\mathcal{R}(\beta)_{i+1}} r\sigma' \triangleleft_{\beta} r\sigma$. \square

We follow the usual scheme of proofs of confluence of orthogonal conditional rewrite systems [Ohl02]. For all $i \in \mathbb{N}$, we denote by $\rightarrow_{\parallel \mathcal{R}(\beta)_i}$ the smallest parallel rewrite relation containing $\rightarrow_{\mathcal{R}(\beta)_i}$ (see Definition 2.5). Hence, $\rightarrow_{\parallel \mathcal{R}(\beta)_i}$ is strictly included in the nested parallel relation $\triangleright_{\mathcal{R}(\beta)_i}$ used in Section 5.2 (Definition 4.11). The main property is the commutation of $\rightarrow_{\parallel \mathcal{R}(\beta)_i}$ and $\rightarrow_{\parallel \mathcal{R}(\beta)_j}$ for all $i, j \in \mathbb{N}$, which corresponds to the usual parallel moves property. Let $<_{\text{mul}}$ be the multiset extension of the usual ordering on natural numbers. In our case, the parallel moves property is:

Parallel Moves. Given $i, j \in \mathbb{N}$, if $\rightarrow_{\beta \cup \mathcal{R}(\beta)_n}$ commutes with $\rightarrow_{\beta \cup \mathcal{R}(\beta)_m}$ for all n, m such that $\{\mathbf{n}, \mathbf{m}\} <_{\text{mul}} \{\mathbf{i}, \mathbf{j}\}$, then $\rightarrow_{\parallel \mathcal{R}(\beta)_i}$ commutes with $\rightarrow_{\parallel \mathcal{R}(\beta)_j}$.

The proof is decomposed into Lemma 6.5 and Lemma 6.6. In Lemma 6.5, assuming the commutation of $\rightarrow_{\beta \cup \mathcal{R}(\beta)_n}$ and $\rightarrow_{\beta \cup \mathcal{R}(\beta)_m}$ for all n, m such that $\{\mathbf{n}, \mathbf{m}\} <_{\text{mul}} \{\mathbf{i}, \mathbf{j}\}$, we consider, for the commutation of $\rightarrow_{\parallel \mathcal{R}(\beta)_i}$ and $\rightarrow_{\parallel \mathcal{R}(\beta)_j}$, the particular case of a rooted $\mathcal{R}(\beta)_i$ -reduction.

Lemma 6.5 *Let \mathcal{R} be an orthonormal system and $i, j \geq 0$. Assume that $\rightarrow_{\beta \cup \mathcal{R}(\beta)_n}$ commutes with $\rightarrow_{\beta \cup \mathcal{R}(\beta)_m}$ for all n, m such that $\{\mathbf{n}, \mathbf{m}\} <_{\text{mul}} \{\mathbf{i}, \mathbf{j}\}$. Then for all rules $\vec{d} = \vec{c} \supset l \mapsto_{\mathcal{R}} r$, we have*

$$\begin{array}{ccc} l\sigma & \xrightarrow{\mathcal{R}(\beta)_i} & r\sigma \\ \parallel \mathcal{R}(\beta)_j \downarrow & & \downarrow \parallel \mathcal{R}(\beta)_j \\ u & \xrightarrow[\mathcal{R}(\beta)_i]{=} & v \end{array}$$

PROOF. The result holds if $i = 0$ since $\rightarrow_{\mathcal{R}(\beta)_0} = \emptyset$. If $j = 0$, then $u = l\sigma$ and take $v = r\sigma$.

Assume that $i, j > 0$ and write q_1, \dots, q_n for the (disjoint) occurrences in $l\sigma$ of the redexes contracted in $l\sigma \rightarrow_{\|\mathcal{R}(\beta)_j\|} u$. Therefore, for all k , $1 \leq k \leq n$, there exists a rule $\rho_k : \vec{d}_k = \vec{c}_k \supset l_k \mapsto_{\mathcal{R}} r_k$ and a substitution θ_k such that $l\sigma|_{q_k} = l_k\theta_k$. Thus, $u = l\sigma[r_1\theta_1]_{q_1} \dots [r_n\theta_n]_{q_n}$. It is possible to rename variables and assume that $\rho, \rho_1, \dots, \rho_n$ have disjoint variables. Therefore, we can take $\sigma = \theta_1 = \dots = \theta_n$.

Assume that there is a non-variable superposition, i.e. that a q_k is a *non variable* occurrence in l . Hence rules ρ and ρ_k form an instance of a critical pair $\vec{d}'\mu = \vec{c}' \supset (l[r_k]_{q_k}\mu, r\mu)$ and there exists a substitution μ' such that $\sigma = \mu\mu'$. By definition of orthonormal systems, $|\vec{d}'\mu| \geq 2$ and there is $m \neq p$ such that $c'_m \neq c'_p$ and $d'_m\mu = d'_p\mu$. Let us write h for $\max(i, j) - 1$. As $d'_m\mu = d'_p\mu$ we have $d'_m\sigma = d'_p\sigma$ and it follows that

$$c'_m \leftarrow_{\beta \cup \mathcal{R}(\beta)_h}^* d'_m\sigma = d'_p\sigma \rightarrow_{\beta \cup \mathcal{R}(\beta)_h}^* c'_p.$$

But $\{h, h\} <_{\text{mul}} \{i, j\}$ and by assumption $\rightarrow_{\beta \cup \mathcal{R}(\beta)_h}$ is confluent. Therefore we must have $c'_m \downarrow_{\beta \cup \mathcal{R}(\beta)_h} c'_p$. But it is not possible since c'_m and c'_p are distinct normal forms. Hence, conditions of ρ and ρ_k cannot be both satisfied by σ and $\rightarrow_{\beta \cup \mathcal{R}(\beta)_h}$ and it follows that there is no non-variable superposition.

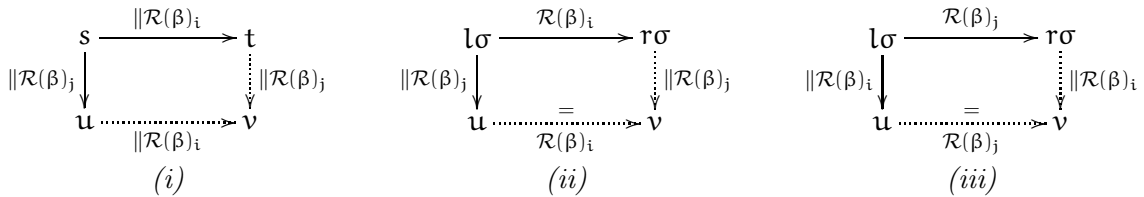
Therefore, each q_k is of the form $u_k.v_k$ where l_{u_k} is a variable x_k . Let σ' be such that $\sigma'(x_k) = \sigma(x_k)[r_k\sigma]_{v_k}$ and $\sigma'(y) = \sigma(y)$ if $y \neq x_k$ for all $1 \leq k \leq n$. Then, $l\sigma \rightarrow_{\|\mathcal{R}(\beta)_j\|} l\sigma'$ and by linearity of l , $u = l\sigma'$. Furthermore, $r\sigma \rightarrow_{\|\mathcal{R}(\beta)_j\|} r\sigma'$. We now show that $l\sigma' \rightarrow_{\mathcal{R}(\beta)_i} r\sigma'$. We have $\vec{d}\sigma \rightarrow_{\beta \cup \mathcal{R}(\beta)_{i-1}}^* \vec{c}$ and $\vec{d}\sigma \rightarrow_{\mathcal{R}(\beta)_j}^* \vec{d}\sigma'$. As $i, j > 0$, we have $\{i-1, j\} <_{\text{mul}} \{i, j\}$. Therefore, by assumption $\rightarrow_{\beta \cup \mathcal{R}(\beta)_{i-1}}$ and $\rightarrow_{\beta \cup \mathcal{R}(\beta)_j}$ commute and there exist terms \vec{c}' such that

$$\vec{d}\sigma' \rightarrow_{\beta \cup \mathcal{R}(\beta)_{i-1}}^* \vec{c}' \leftarrow_{\beta \cup \mathcal{R}(\beta)_{j-1}}^* \vec{c}.$$

As terms \vec{c} are $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$ -normal forms, we have $\vec{c}' = \vec{c}$ and it follows that $l\sigma' \rightarrow_{\mathcal{R}(\beta)_i} r\sigma'$. \square

Now, in Lemma 6.6 we show that the commutation of $\rightarrow_{\|\mathcal{R}(\beta)_i\|}$ and $\rightarrow_{\|\mathcal{R}(\beta)_j\|}$ is ensured by the two particular cases of rooted $\mathcal{R}(\beta)_i$ -reduction and $\mathcal{R}(\beta)_j$ -reduction.

Lemma 6.6 *Let \mathcal{R} be an orthonormal system and $i, j \geq 0$. Property (i) below holds if and only if for all rules $\vec{d} = \vec{c} \supset l \mapsto_{\mathcal{R}} r$, properties (ii) and (iii) hold.*



PROOF. The “only if” statement is trivial. For the “if” case, let s, t, u be three terms such that $u \leftarrow_{\|\mathcal{R}(\beta)_j\|} s \rightarrow_{\|\mathcal{R}(\beta)_i\|} t$. If s is t (resp. u), then take $v = u$ (resp. $v = t$). Otherwise, we reason by induction on the structure of s . If there is a rooted reduction, we conclude by properties (ii) and (iii). Now assume that both reductions are nested. In this case s cannot be a symbol $f \in \Sigma$ nor a variable. If s is an abstraction, we conclude by induction hypothesis. Otherwise s is an application s_1s_2 , and by assumption $u = u_1u_2$ and $t = t_1t_2$ with $u_k \leftarrow_{\|\mathcal{R}(\beta)_j\|} s_k \rightarrow_{\|\mathcal{R}(\beta)_i\|} t_k$. In this case also we conclude by induction hypothesis. \square

Now, an induction on $<_{\text{mul}}$ provides the commutation of $\rightarrow_{\beta \cup \mathcal{R}(\beta)_i}$ and $\rightarrow_{\beta \cup \mathcal{R}(\beta)_j}$ for all $i, j \geq 0$, i.e. the shallow confluence of $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$.

Theorem 6.7 *If \mathcal{R} is an orthonormal system, then $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$ is shallow confluent.*

PROOF. We reason by induction on unordered pairs $\{i, j\}$ seen as multisets and compared with the well-founded relation $<_{\text{mul}}$. We show the commutation of $\rightarrow_{\beta \cup \mathcal{R}(\beta)_i}$ and $\rightarrow_{\beta \cup \mathcal{R}(\beta)_j}$ for all $i, j \geq 0$. The least unordered pair $\{i, j\}$ (considered as a multiset) with respect to $<_{\text{mul}}$ is $\{0, 0\}$. As $\rightarrow_{\beta \cup \mathcal{R}(\beta)_0} = \rightarrow_{\beta}$ by definition, this case holds by confluence of β .

Now, assume that $i > 0$ and that the commutation of $\rightarrow_{\beta \cup \mathcal{R}(\beta)_n}$ and $\rightarrow_{\beta \cup \mathcal{R}(\beta)_m}$ holds for all n, m with $\{n, m\} <_{\text{mul}} \{i, 0\}$. As $\{i-1, i-1\} <_{\text{mul}} \{i, 0\}$, $\rightarrow_{\beta \cup \mathcal{R}(\beta)_{i-1}}$ is confluent and the commutation of $\rightarrow_{\beta \cup \mathcal{R}(\beta)_i}$ with $\rightarrow_{\beta \cup \mathcal{R}(\beta)_0}$ ($= \rightarrow_{\beta}$) follows from Lemma 6.4.

The remaining case is when $i, j > 0$. Using the induction hypothesis, from Lemma 6.5 and Lemma 6.6, we obtain the commutation of $\rightarrow_{\|\mathcal{R}(\beta)_i}$ and $\rightarrow_{\|\mathcal{R}(\beta)_j}$, which in turn implies the commutation of $\rightarrow_{\mathcal{R}(\beta)_i}^*$ and $\rightarrow_{\mathcal{R}(\beta)_j}^*$. Now, as $\{i-1, i-1\} <_{\text{mul}} \{i, j\}$, by Lemma 6.4, \rightarrow_{β} and $\rightarrow_{\mathcal{R}(\beta)_i}$ commute. This way, we also obtain the commutation of \rightarrow_{β} and $\rightarrow_{\mathcal{R}(\beta)_j}$. Then, the commutation of $\rightarrow_{\beta \cup \mathcal{R}(\beta)_i}^*$ and $\rightarrow_{\beta \cup \mathcal{R}(\beta)_j}^*$ easily follows. \square

Example 6.8 *The relation $\rightarrow_{\beta \cup \mathcal{R}(\beta)}$ induced by the system presented in Section 2.4.2 is shallow confluent and thus confluent.*

7 Conclusion

Our results are summarized in Figure 1 page 6.

We provide detailed conditions to ensure modularity of confluence when combining β -reduction and conditional rewriting, either when the evaluation of conditions uses β -reduction or when it does not. This has useful applications on the high-level specification side and for enriching the conversion used in logical frameworks or proof assistants, while still preserving the confluence property.

These results lead us to the following remarks and further research points. The results obtained in Section 4 and 5 for the join conditional rewrite systems extend to the case of oriented systems (hence to normal systems) and to the case of level-confluent semi-equational systems. For semi-equational systems, the proofs follow the same scheme, provided that level-confluence of $\rightarrow_{\mathcal{R}}$ is assumed. However, it would be interesting to know if this restriction can be dropped.

Problems arising from non left-linear rewriting are directly transposed to left-linear conditional rewriting. The semi-closure condition is sufficient to avoid this, and it seems to provide the counterpart of left-linearity for unconditional rewriting. However, two remarks have to be made about this restriction. First, it would be interesting to know if it is a necessary condition and besides, to characterize a class of non semi-closed systems that can be translated into equivalent semi-closed ones. Second, semi-closed terminating join systems behave like normal systems. But normal systems can be easily translated into equivalent non-conditional systems. Moreover such a translation preserves good properties such as left-linearity and non ambiguity. As many practical uses of rewriting rely on terminating systems, semi-closed join systems may be in practice essentially an intuitive way to design rewrite systems that can be then efficiently implemented by non-conditional rewriting.

A wider interesting perspective would be to extend the results to CCERSs [GKK05].

References

- [ALS94] J. AVENHAUS et C. LORÍA-SÁENZ – “Higher Order Conditional Rewriting and Narrowing”, *Proceedings of the 1st International Conference on Constraints in Computational Logics*, LNCS, vol. 845, Springer Verlag, 1994, p. 269–284. 3
- [Bar84] H.P. BARENDREGT – *The Lambda-Calculus, its Syntax and Semantics*, Studies in Logic and the Foundation of Mathematics, North Holland, 1984, Second edition. 3, 5, 15, 19, 23
- [BCKL03] G. BARTHE, H. CIRSTEA, C. KIRCHNER et L. LIQUORI – “Pure Patterns Type Systems”, *Principles of Programming Languages, New Orleans, USA*, ACM, 2003. 3
- [BFG97] F. BARBANERA, M. FERNÁNDEZ et H. GEUVERS – “Modularity of Strong Normalization and Confluence in the Algebraic-lambda-Cube”, *Journal of Functional Programming* **7** (1997), no. 6, p. 613–660. 24
- [BK86] J.A. BERGSTRA et J.W. KLOP – “Conditional rewrite rules: Confluence and termination”, *Journal of Computer and System Sciences* **32** (1986), no. 3, p. 323–362. 11, 18, 37
- [BKR06] F. BLANQUI, C. KIRCHNER et C. RIBA – “On the Confluence of Lambda-Calculus with Conditional Rewriting”, *Proceedings of FoSSaCS’06*, LNCS, vol. 3921, 2006. 5, 28
- [Bla05] F. BLANQUI – “Definitions by Rewriting in the Calculus of Constructions”, *Mathematical Structures in Computer Science* **15** (2005), no. 1, p. 37–92. 3
- [BT88] V. BREAZU-TANNEN – “Combining Algebra and Higher-Order Types”, *Proceedings of LiCS’88*, IEEE Computer Society, 1988. 3, 16
- [BTG89] V. BREAZU-TANNEN et J. GALLIER – “Polymorphic Rewriting Conserves Algebraic Strong-Normalization and Confluence”, *Proceedings of ICALP’89*, 1989. 16
- [BTG94] — , “Polymorphic Rewriting Conserves Algebraic Confluence”, *Information and Computation* **114** (1994), no. 1, p. 1–29. 3, 16
- [BTM87] V. BREAZU-TANNEN et A. MEYER – “Computable Values Can Be Classical”, *Proceedings of POPL’87*, ACM, 1987. 15
- [Böh68] C. BÖHM – “Alcune Proprietà delle Forme β - η -Normali nel λ -K-Calcolo”, Tech. Report 696, Pubblicazioni dell’ Istituto per le Applicazioni del Calcolo, ROMA, 1968. 8
- [CK01] H. CIRSTEA et C. KIRCHNER – “The rewriting calculus — Part I and II”, *Logic Journal of the Interest Group in Pure and Applied Logics* **9** (2001), no. 3, p. 427–498. 3
- [DHK03] G. DOWEK, T. HARDIN et C. KIRCHNER – “Theorem Proving Modulo”, *Journal of Automated Reasoning* **31** (2003), no. 1, p. 33–72. 3
- [DJ90] N. DERSHOWITZ et J.-P. JOUANNAUD – “Rewrite Systems”, *Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics (B)* (J. VAN LEEUWEN, éd.), North-Holland, 1990, p. 243–320. 3

- [DO90] N. DERSHOWITZ et M. OKADA – “A rationale for conditional equational programming”, *Theoretical Computer Science* **75** (1990), p. 111–138. 5
- [Dou92] D.J. DOUGHERTY – “Adding Algebraic Rewriting to the Untyped Lambda Calculus”, *Information and Computation* **101** (1992), no. 2, p. 251–267. 3, 4, 15, 16, 17, 18, 22, 23
- [GKK05] J. GLAUERT, D. KESNER et Z. KHASIDASHVILI – “Expression Reduction Systems and Extensions: An Overview”, *Processes, Terms and Cycles: Steps to the Road of Infinity, Essays Dedicated to Jan Willem Klop, on the Occasion of His 60th Birthday*, LNCS, vol. 3838, Springer, 2005, p. 496–553. 3, 37, 40
- [GM88] E. GIOVANNETTI et C. MOISO – “Notes on the Elimination of Conditions”, *Proceedings of CTRS’87*, LNCS, vol. 308, Springer, 1988, p. 91–97. 37
- [Gra96] B. GRAMLICH – “On Termination and Confluence Properties of Disjoint and Constructor-Sharing Conditional Rewrite Systems”, *Theoretical Computer Science* **165** (1996), no. 1, p. 97–131. 4
- [Hue80] G. HUET – “Confluent Reductions: Abstract Properties and Applications to Term Rewriting Systems”, *Journal of the Association for Computing Machinery* **27** (1980), no. 4, p. 797–821. 15, 17
- [Hue86] — , “Formal Structures for Computation and Deduction”, Technical report, INRIA, Rocquencourt, 1986. 4, 12
- [JO91] J.-P. JOUANNAUD et M. OKADA – “Executable Higher-Order Algebraic Specification Languages”, *Proceedings of LiCS’91*, IEEE Computer Society, 1991. 3
- [Kap84] S. KAPLAN – “Conditional Rewrite Rules”, *Theoretical Computer Science* **33** (1984), p. 175–193. 11
- [Klo80] J.W. KLOP – *Combinatory Reduction Systems*, Mathematical Center Tracts, vol. 127, CWI, 1980, PhD Thesis. 12, 15
- [KOR93] J.W. KLOP, V. VAN OOSTROM et F. VAN RAAMSDONK – “Combinatory Reduction Systems: Introduction and Survey”, *Theoretical Computer Science* **121** (1993), p. 279–308. 3
- [KW97] U. KÜHLER et C.-P. WIRTH – “Conditional Equational Specifications of Data Types with Partial Operations for Inductive Theorem Proving”, *Proceedings of RTA ’97*, LNCS, vol. 1232, Springer, 1997, p. 38–52. 37
- [Mid91] A. MIDDELDORP – “Confluence of the Disjoint Union of Conditional Term Rewriting Systems”, *Proceedings of CTRS’91*, LNCS, vol. 516, 1991, p. 295–306. 4
- [Mül92] F. MÜLLER – “Confluence of the Lambda Calculus with Left Linear Algebraic Rewriting”, *Information Processing Letters* **41** (1992), p. 293–299. 3, 4, 12, 15, 16, 18, 19, 20, 22
- [Nip91] T. NIPKOW – “Higher-Order Critical Pairs”, *Proceedings of LiCS’91*, IEEE Computer Society, 1991. 3

- [Ohl02] E. OHLEBUSCH – *Advanced Topics in Term Rewriting*, Springer, April 2002. 5, 11, 18, 36, 38
- [Oka89] M. OKADA – “Strong Normalizability for the Combined System of the Typed Lambda-Calculus and an Arbitrary Convergent Term Rewrite System”, *Proceedings of ISSAC’89*, ACM, 1989, p. 357–363. 16
- [OR94] V. VAN OOSTROM et F. VAN RAAMSDONK – “Weak Orthogonality Implies Confluence: the Higher-Order Case”, *Proceedings of LFCS’94*, LNCS, vol. 813, 1994. 3, 16, 20, 23, 27
- [Par89] M. PARIGOT – “On the Representation of Data in Lambda-Calculus”, *Proceedings of CSL’89*, LNCS, vol. 440, 1989, p. 309–321. 8
- [Sco75] D. SCOTT – “Lambda Calculus and Recursion Theory”, *Proc. of the third Scandinavian Logic Symposium* (S. KANGER, éd.), North Holland, 1975, p. 154–193. 12
- [Tak93] M. TAKAHASHI – “Lambda-Calculi with Conditional Rules”, *Proceedings of TLCA’93*, LNCS, Springer-Verlag, 1993, p. 406–417. 3, 37
- [Tak95] —, “Parallel Reductions in λ -Calculus”, *Information and Computation* **118** (1995), p. 120–127. 19, 29
- [Toy87] Y. TOYAMA – “On the Church-Rosser Property for the Direct Sum of Term Rewriting Systems”, *Journal of the Association for Computing Machinery* **34** (1987), no. 1, p. 128–143. 3
- [Vri89] R.C. DE VRIJER – “Extending the Lambda Calculus with Surjective Pairing is Conservative”, *Proceedings of LiCS’89*, IEEE Computer Society, 1989, p. 204–215. 4, 12
- [Wad71] C.P. WADSWORTH – “Semantics and Pragmatics of the Lambda-Calculus”, Thèse, Oxford University, 1971. 7
- [Wol93] D.A. WOLFRAM – *The Clausal Theory of Types*, Cambridge Tracts in Theoretical Computer Science, vol. 21, Cambridge University Press, 1993. 3