Confluence in non-left-linear higher-order theories

Gaspard Férey, Jean-Pierre Jouannaud

Confluence in non-left-linear higher-order theories

Gaspard Férey
Deducteam, ENS de Paris-Saclay
email: gaspard.ferey@gmail.com

Jean-Pierre Jouannaud
Deducteam, Université Paris-Saclay
Email:jeanpierre.jouannaud@gmail.com

***Abstract*—We develop techniques based on van Oostrom's decreasing diagrams that reduce confluence proofs to the checking of critical pairs for higher-order rewrite rules extending beta-reduction on pure lambda-terms. We show that confluence is preserved for a large subset of terms that contains all pure lambda terms. Our results are applied to famous Klop's examples of non-confluent behaviours in lambda-calculi with convergent rewrite rules, on the one hand, and to fragments of an encoding in a dependent type theory augmented with rewrite rules of the Calculus of Constructions with polymorphic universes. on the other hand.**

I. Introduction

AGDA, COQ and DEDUKTI are implementations of dependent type theory that all allow for user-defined rewrite rules. Adding rewrite rules makes it difficult to prove consistency, unless the rewrite rules are required to satisfy some very specific format that includes inductive types, such as the general schema [4], which imposes critical-pair critical-pair free left-linear rewrite rules. More permissive attempts are non-conclusive yet [5].

The two essential properties of a type theory, consistency and decidability of type checking, follow from three simpler ones: type preservation, strong normalization and confluence. In dependent type theories however, confluence is often needed to prove type preservation and strong normalization, making all three properties interdependent if termination is used in the confluence proof. This circularity can be broken in two ways: by proving all properties together within a single induction [9]; or by proving confluence on untyped terms first, and then successively type preservation, confluence on typed terms, and strong normalization. We develop the latter way here, focusing on untyped confluence.

Techniques for showing confluence of higher-order dependent theories developed in [7], [8] are restricted to those for which the left-hand sides of rules are linear. We consider here the case of non-left-linear rewrite rules. We have indeed a specific interest in the following examples of non-left-linear rules that pop-up when encoding type-theoretic universes:

$$\max(X, X) \to X$$
$$\mathbf{F}(X, X, Y) \to Y$$
$$\mathrm{u}(X, \mathrm{c}(X, Y)) \to Y$$

But non left-linear rules have a bad rewriting behaviour in presence of β-reductions. A counter-example to termination in the simply typed λ-calculus is given in [17]. Numerous counter-examples to confluence in the pure λ-calculus are given in [11], including surjective pairing which came up as a surprise at that time. A striking one is provided by the rule $f(X, X) \to a$, whose interaction with a fixpoint combinator generates diverging reductions that cannot be joined: confluence of the pure λ-calculus can't be preserved when adding rewrite rules whose left-hand sides are non-linear.

Indeed, non-left-linear rules don't behave as left-linear ones. Consider the first-order rules $f(X, Y) \to a$ and $c \to g(c)$. Then, $f(c, c)$ rewrites to a using the first rule. It also rewrites to $f(c, g(c))$ using the second rule. Since the first rule still applies to $f(c, g(c))$, we get the same result a, although a bit more painfully. Now, assume the first rule is $f(X, X) \to a$. Then, $f(c, c)$ still rewrites to a, but $f(c, g(c))$ does not anymore. We need to rewrite the first subterm c first to get another copy of $g(c)$. So, rewriting below a redex may destroy that redex, and, as a result, the so-called ancestor peaks do not commute nicely anymore, requiring an additional rewrite step from $f(c, g(c))$, hence destroying the so-called strong confluence property, or, in modern terms, van Oostrom's decreasing diagram property. Adding now the new rule $f(X, g(X)) \to b$ breaks confluence since $f(c, c)$ can now rewrite to a and b, both in normal form.

One can also say that adding the rule $c \to g(c)$ to the system $\{f(X, X) \to a, f(X, g(X)) \to b\}$, which is confluent since terminating and critical pair free, breaks its confluence. Since that rule mimics a fixpoint, adding the untyped lambda calculus to that systems destroys its confluence as well. The point is that these two rules have a *hidden* critical pair, since c and $g(c)$ are now equated by the fixpoint rule. In case all those critical pairs are joinable, as is not the case in this example, then confluence is preserved [14]. Since the proof uses van Oostrom's decreasing diagrams, the ancestor peak in the previous paragraph must be made decreasing. This is done by using a rewrite relation, called *sub-rewriting*, such that $f(u, v) \longrightarrow a$ and $f(u, g(v)) \longrightarrow b$, provided u and v can be equated with the added fixpoint rule, which has the effect of linearizing the appropriate instances of the non-linear rules. The confluence proof is then done by induction on a stratification of terms by *layers* which are used as labels for van Oostrom's decreasing diagrams.

In the previous example, the outside redex $f(c, c)$ and the inside redexes c belong to two different layers, one being *below* the other. The point is that sub-rewriting, here, is layer non-increasing, a requirement for using the previous technique. The beta rule being layer increasing, this technique falls apart. Indeed, adding non-left-linear rules does not preserve confluence of the pure lambda calculus, as shown by Klop.

The question then becomes: on which subset(s) of terms is confluence preserved ? and what kind of stratification can be used that makes beta be layer non-increasing ? We shall define meaningfull such subsets by means of a very simple typing system based on integers as base types which ensures that non-linear variables of a left-hand side of rules are substituted by terms of a strictly smaller layer than the left-hand side instance itself. These integers will define the layers we need to carry out the technique. It turns out that this generalizes another technique, called confinement, whose idea is to have two layers: a bottom layer made of pure *first-order* terms, which cannot contain beta-redexes as subterms, and an upper layer for the other terms [1]. Non-linear rewrite rules are then restricted to the bottom layer, hence must be first-order, higher-order rules being left-linear.

The paper is organized as follows. Section II describes our model of higher-order rewriting and Section III introduces an abstract notion of term layering satisfying some axioms. We show in Section IV that confluence of the λ-calculus is preserved by adding rules whose critical pairs are joinable by a decreasing diagram. The proof is done by induction on layers, the induction step being carried out thanks to van Oostrom's decreasing diagram theorem. In Section V, we describe our technique for defining layers in practice, and apply it to Klop's examples as well as to ours. Significance of the results is discussed in conclusion.

II. The higher-order rewriting calculus $\lambda\mathcal{F}$ [7]

Given an *untyped* annotated lambda calculus generated by a vocabulary made of three pairwise disjoint sets, a signature $\mathcal{F}$ of *function symbols*, a set $\mathcal{X}$ of *variables*, and a set $\mathcal{Z}$ of *meta-variables*, we are interested in the calculus $\lambda\mathcal{F}$, whose reduction relation extends the β-rule of the underlying λ-calculus by a set R of user-defined rewrite rules built over that vocabulary.

A. Terms in $\lambda\mathcal{F}$

$\lambda\mathcal{F}$ is a mix of the pure annotated lambda-calculus and Klop's combinatory reduction systems [11], that fits with incarnations of dependent type theory such as Agda, Coq and Dedukti, which allow for rewrite rules. Terms are those of the annotated lambda calculus enriched with *$\mathcal{F}$-headed* terms of the form $f(\overline{u})$ with $f \in \mathcal{F}$ and *meta-terms* of the form $Z[\overline{v}]$ with $Z \in \mathcal{Z}$. Only variables can be abstracted over. Elements of the vocabulary have arities, denoted by vertical bars as in $|f|$. Variables have arity zero, meta-variables have a maximum arity. The set of terms, $\mathcal{T}_{\lambda\mathcal{F}}$, is defined by the grammar:

$$u, v := x \in \mathcal{X} \mid (u\ v) \mid \lambda x{:}u.v \mid f(\overline{u}) \mid Z[\overline{v}]$$
$$\text{where } f \in \mathcal{F},\ |\overline{u}| = |f|,\ Z \in \mathcal{Z} \text{ and } |\overline{v}| \leq |Z|$$

Our abstraction operator $''\lambda x : .''$ has arity 2: our syntax is slightly richer than usual in order to enable abstracting step by step derivation in typed lambda calculi by derivations in $\lambda\mathcal{F}$. The calculus without annotation being of course itself an abstraction of $\lambda\mathcal{F}$, all results presented here adapt straightforwardly to that calculus by forgetting annotations. We will use this facility unannounced in examples originating from non-dependent typed lambda calculi.

Following usage, we don't duplicate parentheses, writing $f(x\ y)$ for $f((x\ y))$, and use brackets instead of parentheses for meta-variables. We use the small letters $f, g, h, \ldots$ for function symbols and $x, y, z, \ldots$ for variables, and reserve capital letters $X, Y, Z, \ldots$ for meta-variables. When convenient, a small letter like x may denote any variable in $\mathcal{X} \cup \mathcal{Z}$.

We use the notations $|\_|$ for various quantities besides arities, e.g., the length of a list, the size of an expression, the cardinality of a set, $[m..n]$ for the list of natural numbers from m to n, and $\overline{u}[m..n]$ for the finite sublist $u_m \ldots u_n$ of $\overline{u}$.

Unlike function symbols and Klop's meta-variables, meta-variables here have an arity which is not fixed, but bounded, a handy feature used in Dedukti that avoids writing complex applications of variable length as in $(X\ a)$ and $(((X\ a)b)c)$.

Positions in terms are words over the natural numbers (assuming $|\lambda x : .| = 2$), using $\cdot$ for concatenation, Λ for the empty word, P/p for $\{q : p \cdot q \in P\}$, $\leq_{\mathcal{P}}$ for the prefix order (*above*), $\geq_{\mathcal{P}}$ for its inverse (*below*), $>_{\mathcal{P}}$ for the strict part of $\geq_{\mathcal{P}}$, $p\#q$ for $\neg(>_{\mathcal{P}} \vee \leq_{\mathcal{P}})$ (*parallel*), and $p >_{\mathcal{P}} Q$ for $\forall q \in Q :\ p >_{\mathcal{P}} Q$.

Given a term M, we use $\mathcal{V}ar(M)$ and $M\mathcal{V}ar(M)$ for its sets of free variables and of meta-variables respectively, $M(p)$ for its symbol at positions p, and $\mathcal{P}os(M)$, $\mathcal{V}\mathcal{P}os(M)$, $M\mathcal{P}os(M)$ for the following respective sets of positions of M: all positions, the positions of free variables, and of meta-variables. A term M is *ground* if $\mathcal{V}ar(M) = \varnothing$, *closed* if $M\mathcal{V}ar(M) = \varnothing$, and linear if $|M\mathcal{P}os(M)| = |M\mathcal{V}ar(M)|$.

As usual, meta-variables, hence terms in $\mathcal{T}_{\lambda\mathcal{F}}$, belong to the meta-language. On the other hand, variables belong to the language, whose set of closed terms is denoted by $\mathcal{T}$.

B. Substitutions

A *substitution* is a mapping from a finite set of variables and meta-variables to terms, written $\sigma = \{x_1 \mapsto M_1, \ldots, x_n \mapsto M_n\}$, or $\sigma = \{\overline{x} \mapsto \overline{M}\}$, where $ar(M_i) \geq ar(x_i)$, which extends to an *arity-preserving* and *capture-avoiding* homomorphism on terms as defined next. Let $\mathcal{D}om(\sigma) = \{x_1, \ldots, x_n\} \subseteq \mathcal{X} \cup \mathcal{Z}$ be the *domain* of σ while $\mathcal{R}an(\sigma) = \bigcup_{i=1}^{n} \mathcal{V}ar(M_i) \cup M\mathcal{V}ar(M_i)$ is its *image*. By convention, we write $\sigma(x_i)$ for M_i. As in λ-calculus, substituting in terms requires renaming bound variables to avoid capturing free ones. Using post-fixed notation, applying a substitution to a term is therefore defined as follows: $x_i\sigma = M_i$; $y\sigma = y$ if $y \notin \mathcal{D}om(\sigma)$; $f(\bar{t})\sigma = f(\bar{t}\sigma)$; $(u\ v)\sigma = (u\sigma\ v\sigma)$; and $(\lambda x.u)\sigma = \lambda z.u(\{x \mapsto z\})\sigma$ where $z \notin \mathcal{D}om(\sigma) \cup \mathcal{R}an(\sigma)$ (x needs not be renamed in z if it already satisfies the condition). Assuming now $Z \mapsto \lambda\overline{x}.s \in \sigma$, where s is not an abstraction, the additional rule for meta-variables, inspired from Klop's definition of substitutions in the case of fixed arities [11], is as follows:

1) $|\overline{u}| = m \leq n = |\overline{x}|$, then
   $Z[\overline{u}]\sigma = \lambda\overline{x}[m{+}1..n].s\{\overline{x}[1..m] \mapsto \overline{u}\sigma\}$,
   (replacement of missing arguments of Z is delayed)

2) $m > n$, then (since $ar(s) \geq |Z| - |\overline{x}| \geq |\overline{u}| - |\overline{x}| > 0$) $s = Y[\overline{t}]$ and $\overline{u} = \overline{v}\,\overline{w}$ with $|\overline{v}| = n$ and $|Y| - |\overline{t}| \geq |\overline{w}|$, in which case $Z[\overline{u}]\sigma = Y[\overline{t}\{\overline{x} \mapsto \overline{v}\sigma\}, \overline{w}\sigma]$.

The result $t\sigma$ of substituting the term t is called an *instance* of t, and the operation itself an *instantiation*. The substitution σ is *ground* (resp., *closed*) when so are all M_i's. A substitution σ can be *restricted to* or *deprived from* (meta-)variables in some set V, written $\sigma_{|V}$ and $\sigma_{\setminus V}$ respectively.

Let for example s be the term $f(X((x\ \ y), y))$, where X has two arguments, $(x\ \ y)$ and y, and σ be the substitution $\{X \mapsto \lambda x'y'z'.g(x', y', z'), y \mapsto a\}$. Then, we get $s\sigma = f(\lambda z'.g((x\ \ y)\sigma, y\sigma, z'\sigma)) = f(\lambda z'.g((x\ \ a), a, z'))$. Let us now compare with the instance by σ of the term $u = f((X\ \ (x\ \ y))\ \ y)$, in which X is applied successively to $(x\ y)$ and y. Then $u\sigma = f((\lambda x'y'z'.g(x', y', z')\ \ (x\ \ a))\ \ a)$. We can see that $u\sigma$ reduces to $s\sigma$ in two β-steps: substitutions of meta-variables hides those reductions.

Substitutions are extended to sequences of terms and to substitutions as expected. We use $\sigma_{|\mathcal{Y}}$ for the restriction of σ to $\mathcal{Y} \cap \mathcal{D}om(\sigma)$ and postfix notation for the application of σ to a term t, writing $t\sigma$, or to a vector of terms $\overline{t}$, writing $\overline{t}\sigma$, or to a substitution τ, writing $\tau\sigma$, and call $t\sigma$ (resp., $\overline{t}\sigma$, $\tau\sigma$) the *instance* of t (resp., $\overline{t}, \tau$) by σ. The notation $\mathcal{P}os(\sigma)$ will have the obvious meaning of a sequence of $\mathcal{D}om(\sigma)$-indexed sets of positions.

C. Splitting

Given a term u and a list $P = \{p_i\}_{i=1}^{i=n}$ of parallel positions in u, we define the term obtained by *splitting* u along P as $\underline{u}_P = u[Z_1(\overline{x_1})]_{p_1} \dots [Z_n(\overline{x_n})]_{p_n}$ (u is cut below P) and its associated substitution by $\overline{u}^P = \{Z_i \mapsto \lambda\overline{x_i}.u|_{p_i}\}_{i=1}^{i=n}$ (u is cut above P), where, for all $i \in [1, n]$, $\overline{x_i}$ is the list of all variables of $u|_{p_i}$ bound in u above p_i and Z_i is a fresh meta-variable of arity (exactly) $|\overline{x_i}|$. The definition of substitution for meta-variables ensures that $\underline{u}_P\overline{u}^P = u$.

Splitting a term t at the set of parallel positions K of its meta-variables, yields a linear term t^{lin} called *linearization* of t, the substitution $\overline{t}^K$ being a mapping from $M\mathcal{V}ar(t^{lin})$ to $M\mathcal{V}ar(t)$.

Our use of splitting in this paper will be systematic unless it alters readability for no good reason. This invention permitted by Klop's notion of meta-variable, is the only technique we know of which allows to manipulate terms with binders safely, in case renaming of variables needs to take place independently in a term and in its context, as it will happen here.

D. Reductions

Arrow signs used for rewriting will often be decorated, below by a name, and above by a position p or set of positions P, as in $s \xrightarrow[R]{p} t$ or by a property that this position or set of positions satisfies, as in $u \xrightarrow[R]{<_{\mathcal{P}} p} v$, $u \xrightarrow[R]{\geq_{\mathcal{P}} p} v$ or $u \xrightarrow[R]{\#P} v$.

Given a reduction relation $\longrightarrow$, $s \xleftarrow[L \to R]{p} u \xrightarrow[G \to D]{q} t$ is a *local peak*, of which there are 3 categories: *disjoint* if $p \# q$, *ancestor* if $p \geq_{\mathcal{P}} q \cdot F_L$, and *overlapping* if $q \in p \cdot \mathcal{FP}os(L)$. Confluence proofs are usually based on the analysis of local peaks.

Two different kinds of reductions coexist in $\lambda\mathcal{F}$, functional and higher-order reductions. Both are meant to operate on closed terms. However, rewriting open terms will sometimes be needed, in which case rewriting is intended to rewrite all their closed instances at once.

E. Functional reductions

Functional reduction is the relation on terms generated by the rule $(\lambda x.u\ \ v) \underset{\beta_\alpha}{\longrightarrow} u\{x \mapsto v\}$. The usually omitted α-index stresses that renaming bound variables, called α-*conversion*, is built-in. As is customary [16], the particular case for which v is a variable is denoted by β^0. Instantiating a β^0-step may yield a full β-step. For example, $s = (\lambda x.(\lambda y.g(y)\ \ x)\ \ a) \xrightarrow[\beta^0]{1\cdot 1} (\lambda x.g(x)\ \ a) \xrightarrow[\beta]{\Lambda} g(a)$ while $s \xrightarrow[\beta]{\Lambda} (\lambda y.g(y)\ \ a) \xrightarrow[\beta]{\Lambda} g(a)$. This is our main motivation for using Klop's notion of substitution for meta-variables, whose benefits will appear all along the paper.

F. Higher-order reductions

Higher-order reductions result from rules whose left-hand sides are higher-order patterns in Miller's or Nipkow's sense [15], although they need not be typed here:

Definition 1 (Pattern): A *pre-redex* of arity n in a term L is an unapplied meta-term $Z[\overline{x}]$ whose arguments $\overline{x}$ are n pairwise distinct variables. A *pre-pattern* is a ground β-normal term all of whose meta-variables occur in pre-redexes. A *pattern* is a pre-pattern which is not a pre-redex.

It is important to assume, as does Nipkow, that pre-patterns are β-normal. Note that erasing types from a Nipkow's pattern yields a pattern in our sense, since his pre-redexes being of base type, they cannot be applied.

Observe that pre-redexes in pre-patterns can only occur at parallel positions, whose set plays a key rôle:

Definition 2 (Fringe): The *fringe* F_L of a pre-pattern L is the set of parallel positions of its pre-redexes. We denote by $\mathcal{FP}os(L) = \{p \in \mathcal{P}os(L) : p <_{\mathcal{P}} F_L\}$ the (non-empty) set of *functional positions* of the pre-pattern L, and by $M\mathcal{V}ar(L, o)$, for $o \in F_L$, the meta-variable Z such that $L|_o = Z[\overline{x}]$. We also define $F_\beta = \{11, 2\}$ for convenience.

Example 1: The term $L = f(\lambda xyz.g(X[x, y, z], X[x, y]))$ is a pattern. Its pre-redexes are the terms $X[x, y, z]$ and $X[x, y]$. Its fringe is the set $F_L = \{1^5, 1^4 \cdot 2\}$. The term $(f(\lambda xyz.g(X[x, y, z])\ \ (a\ \ X))$ is also a pattern, its fringe is the set $\{1^6, 2^2\}$. Terms $f(\lambda x.X[x, x])$, $f(X[a])$, $f(X[Y])$, and $f(X\ Y)$, are no patterns.

Note that the set of functional positions coincides with the usual notion for first-order terms.

We can now define higher-order rules and rewriting:

Definition 3 (Rule): A (higher-order) *rule* is a triple $i : L \to R$, whose (possibly omitted) *index* i is a natural number, left-hand side L is a pattern, and right-hand side R is a ground β-normal term such that $M\mathcal{V}ar(R) \subseteq M\mathcal{V}ar(L)$. The rule is *left-linear* if L is linear.

So, rules are pairs of (specific) ground terms. They may have meta-variables, but don't admit free variables. This allows

to clearly separate the object language (which has no meta-variables), from the meta-language (which has meta-variables). Rules, critical pairs and and splittings belong to the meta-language, which serves analyzing the properties of the language. The role taken by free variables in first-order rules is therefore taken here by meta-variables of arity zero.

Definition 4 (Higher-order untyped rewriting): Given an open term u, a position $p \in \mathcal{P}os(u)$, and a rule $i : L \to R$, then u rewrites with i at p, written $u \xrightarrow[i]{p} v$, iff $u|_p = L\gamma$ for some substitution γ, and $v = u[X[\overline{x}]]_p\{X \mapsto \lambda\overline{x}.R\gamma\} = u[R\gamma]_p$, where $\overline{x}$ is the list of variables of $u|_p$ which are bound above the position p in u. We write $u \xrightarrow[\mathcal{R}]{p} v$ for $\exists i \in \mathcal{R}.u \xrightarrow[i]{p} v$.

Let's now make our splitting notations fully explicit. Whenever $u \xrightarrow[i]{p} v$, we have by definition:

- $\underline{u}_p = u[X[\overline{x}]]_p$ and $\overline{u}^p = \{X \mapsto \lambda\overline{x}.u|_p\}$ with $\overline{x}$ variables bound above p in u
- $u = \underline{u}_p\overline{u}^p = \underline{u}_p\{X \mapsto \lambda\overline{x}.u|_p\} = \underline{u}_p\{X \mapsto \lambda\overline{x}.L\gamma\}$
- $v = \underline{u}_p\{X \mapsto \lambda\overline{x}.R\gamma\}$, hence $\underline{v}_p = \underline{u}_p$, $\overline{v}^p = \{X \mapsto \lambda\overline{x}.R\gamma\}$ and $v|_p = R\gamma$.

Example 2: Let $L = der(\lambda x.sin(F[x])) \to R = \lambda x.cos(F[x])$, and take for σ the identity substitution $\{F \mapsto \lambda x.x\}$. Then, $L\sigma = der(sin(x))$ and $R\sigma = cos(x)$, hence $der(sin(x)) \longrightarrow cos(x)$.

In sharp contrast with Nipkow [15], we observe that we need not matching modulo β^0 explicitely, since the needed β^0-steps are now hidden in Klop's definition of substitution for meta-variables. Nor do we assume that u, or v, is β-normal, or even β-normal up to position p. We cannot for two reasons: β-normal forms may not exist on the one hand, and on the other hand, we need monotonicity and stability properties:

Lemma 1 (Splitting Above): Let $s \xrightarrow[L\to R]{q} t$. Then, $\overline{s}^q \xrightarrow[L\to R]{} \sigma$ and $t = \underline{s}_q\sigma$.

Lemma 2 (Monotonicity): Let $s \xrightarrow[L\to R]{p} t$ and u a term such that $q \in \mathcal{P}os(u)$. Then, $u[s]_q \xrightarrow[L\to R]{q\cdot p} u[t]_q$.

Lemma 3 (Stability): Let $s \xrightarrow[L\to R]{p} t$ and σ a substitution. Then $s\sigma \xrightarrow[L\to R]{p} t\sigma$.

Monotonicity and stability extend to rewriting at a set of parallel positions P.

Lemma 4 (Substitution Lemma): Let $u \xrightarrow[\mathcal{R}]{p} v$ and $\sigma \xrightarrow[\mathcal{R}]{} \tau$. Then, $u\sigma \xrightarrow[\mathcal{R}]{} v\tau$.

G. Rewrite theories

Definition 5: A *$\lambda\mathcal{F}$-rewrite theory* is a pair $(\mathcal{F}, \mathcal{R})$ made of a user's signature $\mathcal{F}$ and a set $\mathcal{R}$ of higher-order rewrite rules on that signature, defining the rewrite relation of $\lambda\mathcal{F}$ as $\xrightarrow[\mathcal{R}\cup\beta_\alpha]{}$, also written $\xrightarrow[\mathcal{R}\beta]{}$.

Rewrite theories are used in recent version of AGDA and COQ [5], and in DEDUKTI [3] to define the conversion rule of the calculus, which is, as is customary, untyped (but type-preserving and operating on typed terms). The rewrite relation implemented in DEDUKTI is indeed the one we just described, Klop's notion of substitution for meta-variables being implemented via a dedicated mechanism.

In this paper, we are interested in rewrite theories whose left-hand sides of rules may be non-linear, and whose rewrite relation is therefore non-confluent. Yet, non-linear rewrite theories occur in practice, for example when encoding polymorphic universes in DEDUKTI.

The precise problem we address is to approximate the subset of closed terms on which confluence of beta-reductions is preserved, assuming the rules in $\mathcal{R}$ satisfy some reasonable assumptions. Finding a meaningful one is an open problem.

H. Critical peaks

Our goal is to reduce confluence in $\lambda\mathcal{F}$ to the checking of finitely many minimal overlapping peaks of $\mathcal{R}$ called critical:

Definition 6 (Critical peak): Given rules $L \to R, G \to D \in \mathcal{R}$ and position $p \in \mathcal{FP}os(L)$ such that $L|_p = G$ has most general unifier σ, then $R\sigma \xleftarrow[L\to R]{\Lambda} L\sigma = L\sigma[G|_p\sigma]_p \xrightarrow[G\to D]{p} L\sigma[D\sigma]_p$ is a *critical peak* of $G \to D$ onto $L \to R$ at position p.

Note that this definition is exactly the same as that of a first-order critical peak, the only difference being the unification algorithm used. Once more, this is a benefit of Klop's notion of substitution for meta-variables.

Higher-order unification of (untyped) patterns is described in [7] for linear patterns, and its extension to non-linear ones is sketched without proof, but follows the standard path. We will take the existence and computation of a most general unifier for granted. Assuming that rewrite rules are typed checked, they are in AGDA, COQ, DEDUKTI, then these assumptions follow indeed from typed pattern unification.

III. LAYERED REWRITING

Stratification of terms is at the core of our technique for proving confluence of rewriting on nthe set $\mathcal{T}$ of closed terms. First, we introduce a first level of *confined* terms (and rewrites) before to give the requirements for the whole hierarchy of closed terms. Rewriting can therefore be classified by levels according to that hierarchy. We then introduce sub-rewriting, in which a non-linear variable in a rule may be instantiated by different terms provided they are joinable in a strictly lower level, and state its basic properties, including a reduction of its overlapping peaks to the higher-order critical peaks we just defined. We end up recalling orthogonal λ-reductions.

A. Confinement

Confinement was introduced in [1], the only work we are aware of, apart Klop's thesis [12], that addresses the problem of confluence of a set of higher-order rules containing possibly non-left-linear rules. These rules, called confined, can only rewrite terms belonging to a subset of first-order terms. Being first-order, the confluence of these rules can be checked by standard methods. Blending them with left-linear higher-order rules is however a non-trivial problem.

The signature $\mathcal{F}$ is therefore split into two disjoint subsets, $\mathcal{F}_c$, the set of *confined* function symbols, and $\mathcal{F}_u$, the set of

unconfined function symbols, so that $\mathcal{F}=\mathcal{F}_c \cup \mathcal{F}_u$. We denote by $T_c \subseteq \mathcal{T}_{\lambda\mathcal{F}}$ the set of *confined expressions*, i.e., the first-order terms built over $\mathcal{F}_c \cup \mathcal{X}$.

Accordingly, the rewrite system $\mathcal{R}$ is split into two disjoint sets of rules, R_c and R_u, so that $\mathcal{R}=R_c \cup R_u$. Rules in R_c are *confined*, they operate on the set of *confined* first-order expressions. We don't actually need to know what these rules are, we won't need them but the rewrite relation they generate on confined terms. Rules in R_u are higher-order rules, possibly non-left-linear, which operate on non-confined expressions.

B. Term layering

From now on, we assume a subset $\mathcal{L} \subset \mathcal{T}$ of so-called *layered terms* partitioned into pairwise disjoint sets $\mathcal{L}_n$ of terms annotated with $n \in \mathbb{N}$. We write $\mathcal{L}_{\leq n} := \bigcup_{k \leq n} \mathcal{L}_k$. Variables need have a level too, hence are pairs of the form $x : n$. Substitutions are then written $\{x_i : n_i \mapsto u_i\}_i$, where n_i is a level. A meta-variable in a rule has no level, it will be determined when matching a given closed term against that rule. More generally, non-closed terms have no level. In particular, the left-hand and right-hand sides of a rule have no level.

Definition 7: A substitution σ is *well-layered* if $\forall x : n \mapsto u \in \sigma :\ u \in \mathcal{L}_{\leq n}$. We denote by WLS their set.

We assume the following properties of layered terms::

- **(H0)** $\mathcal{L}_0 = \mathcal{T}_c$
- **(H1)** $t \in \mathcal{L}_n$ and $t \rhd u$ implies $u \in \mathcal{L}_{\leq n}$
- **(H2)** $t \in \mathcal{L}_n$ and $t \underset{\mathcal{R}\beta}{\longrightarrow} u$ implies $u \in \mathcal{L}_{\leq n}$
- **(H3)** $t \in \mathcal{L}_n$ and $\sigma \in WLS$ implies $t\sigma \in \mathcal{L}_{\leq n}$
- **(H4)** $L \to R \in \mathcal{R}$ and $L^{lin}\sigma \in \mathcal{L}_n$ implies $\forall p \in F^{nl}_L :\ X^p\sigma \in \mathcal{L}_{<n}$

Consider for example the rule $F(X,X,Y) \to Y$, and let $u \in \mathcal{L}_m, v \in \mathcal{L}_{\leq m}, w \in \mathcal{L}_n$. Then, according to (H1), (H4) and (H3), the term $F(u,v,w)$ belongs to L_{m+1} if $m \geq n$ and to L_n if $n \geq m+1$.

As can be guessed, our goal is to describe techniques for showing that the derivation relation $\underset{\mathcal{R}\beta}{\longrightarrow}$, which cannot be confluent on $\mathcal{T}$ if $\mathcal{R}$ contains non-left-linear rules, is nevertheless confluent on $\mathcal{L}$. To make sense, $\mathcal{L}$ should be a large enough subset of closed terms. In particular, it should contain all instances of L and R by a well-layered substitution σ whenever $L \to R \in \mathcal{R}$ and $L\sigma$ and $R\sigma$ are closed terms. We shall see that $\mathcal{L}$ can be made indeed very large. In particular, it will contain the whole set of pure λ-terms.

C. Level Rewriting

Let $\longrightarrow$ a rewriting relation, possibly rewriting several redexes at once. Its steps can be labeled by a property of their levels:

Definition 8 (Level rewriting): Let $u \underset{L\to R}{\overset{P}{\longrightarrow}} v$ for some $u \in \mathcal{L}$, set of positions P and rule $L \to R$. We say that u rewrites to v

- *at level* n, written $u \underset{L\to R,n}{\overset{P}{\longrightarrow}} v$, if $\forall p \in P :\ u|_p \in \mathcal{L}_n$,
- *below level* n, written $u \underset{L\to R,<n}{\overset{P}{\longrightarrow}} v$, if $\forall p \in P :\ u|_p \in \mathcal{L}_{<n}$

The rule name $L \to R$ may be omitted, or replaced by $\mathcal{R}$ when convenient. Using the wording *layered reductions* is reserved to plain rewriting, in which case P is a singleton set.

Note that we don't care about the level of the whole term u, only the redexes' levels play a rôle.

Lemma 5: Let $\longrightarrow$ be a monotonic, stable relation on terms. Then $\underset{n}{\longrightarrow}$ and $\underset{\leq n}{\longrightarrow}$ are monotonic, and $\underset{\leq n}{\longrightarrow}$ is stable

D. Sub-rewriting

A main idea of the confluence proof to come is to proceed by induction on the level of terms in $\mathcal{L}$. In order to prove confluence at level $n+1$, the induction hypothesis will therefore guarantee confluence at all levels up to n. This remark is at the heart of the sub-rewriting relation.

Sub-rewriting was introduced in [13] and developed further in [14]. Assuming an instance of a left-hand side of rule belongs to some level, its non-linear variables must be instantiated by terms of a strictly lower level by **(H4)**. This allows to *equalize* different values of a non-linear variable by rewriting recursively at a strictly lower level. Although concrete levels in [14] are completely different from the abstract levels defined here, as well as from the concrete levels defined in Section V, similar techniques apply. Properties are proved when new.

From now on, we may denote by $\longrightarrow$ the relation $\underset{\mathcal{R}\beta}{\longrightarrow}$.

Definition 9 (Sub-rewriting): A term $u \in \mathcal{L}$ *sub-rewrites* to a term v at position $p \in \mathcal{P}os(u)$ for some rule $L \to R$, written $u \overset{p}{\underset{\mathcal{R}}{\rhd\!\!\longrightarrow}} v$, if there exists a substitution θ such that $u \xrightarrow{\geq_{\mathcal{P}} p \cdot F^{nl}_L} u[L\theta]_p \underset{L\to R}{\overset{p}{\longrightarrow}} u[R\theta]_p = v$.

The term $u|_p$ is called a *sub-rewriting redex* of u at p, the substitution θ is an *equalizer* of L, and the rewrite steps from $u|_p$ to $L\theta$ constitute the corresponding *equalization* steps.

We can of course categorize sub-rewriting by levels, with the expected notation. Note that in case $L \to R \in R_c$, then sub-rewriting coincides with plain rewriting, saving us from using a specific notation for confined rewriting.

A simple, important property of a sub-rewriting redex, which illustrates the informal introduction of the definition, is that it is an instance of a linearized left-hand side of rule, and that equalization steps operate at a strictly lower level. This is expresed by the following property:

Lemma 6: Let $u \overset{p}{\underset{\mathcal{R}}{\rhd\!\!\longrightarrow}} v$. Then there exist a rewrite rule $L \to R \in \mathcal{R}$, substitutions σ, τ, δ, and an integer n such that:

1) $u \overset{p}{\underset{L\to R,n}{\rhd\!\!\longrightarrow}} v$
2) $\mathcal{D}om(\sigma) = \mathcal{MV}ar^l(L), \mathcal{D}om(\tau) = \mathcal{MV}ar^{nl}(L^{lin})$ and $\mathcal{D}om(\delta) = \mathcal{MV}ar^{nl}(L)$
3) $u|_p = L^{lin}(\sigma \cup \tau)$ and $v = u[R(\sigma \cup \delta)]_p$
4) $\forall X^p \in \mathcal{MV}ar^{nl}(L^{lin})\ (\tau(X^p) \underset{<n}{\twoheadrightarrow} \delta(X))$

Proof. 1 follows from **(H1)**. 2 and 3 are clear, 4 is trivial if $L \to R \in R_c$, and follows from **(H4)** and **(H2)** otherwise. $\Diamond$

Lemma 7 (Monotonicity): Let $s[t]_p$ and $s[u]_p$ be layered terms such that $t \overset{q}{\underset{\mathcal{R},n}{\rhd\!\!\longrightarrow}} u$. Then $s[t] \overset{p\cdot q}{\underset{\mathcal{R},n}{\rhd\!\!\longrightarrow}} s[u]$.

Proof. By monotonicity of higher-order rewriting and **(H1)**. ◊

Lemma 8 (Stability): Let $t \xrightarrow[\mathcal{R},n]{p} u$ and assume that σ is well-layered. Then $t\sigma \xrightarrow[\mathcal{R},\leq n]{p} u\sigma$.

Proof. By stability of higher-order rewriting and **(H3)**. ◊

E. Sub-rewriting peaks

A sub-rewriting equalization step uses rules fron R_c, R_u, as well as the beta rule. As a consequence, the characterization of a sub-rewriting overlapping peak $s \xleftarrow[L\to R]{p} u \xrightarrow[G\to D]{p\cdot q} t$ such that $q <_{\mathcal{P}} F_L$ involves unifying $L|_q$ with G modulo $\mathcal{R}\cup\beta$. In general, there would be no set of most general unifiers and confluence would then not be checkable. Our goal is therefore to find conditions that reduce sub-rewriting overlapping peaks to critical higher-order peaks.

A first condition allowing us to reduce unification of two patterns modulo $\mathcal{R}\cup\beta$ to higher-order unification of those patterns is that the equalization processes from u to s and u to t preserve L^{lin} and G^{lin}. This the rôle of:

Definition 10 (Linear independence): A set of rules $\mathcal{R}$ is *linearly independent* if any two rules $L \to R, G \to D \in \mathcal{R}$ satisfy the *linear independence* property:

$\forall p \in \mathcal{P}os(L)$ such that L^{lin} and G^{lin} are renamed apart and unify at p, then

1) $\forall q \in \mathcal{FP}os(G)$ such that $p\cdot q \geq_{\mathcal{P}} F^{nl}_L$, $G|_q$ is unifiable with no linearized left-hand side of a rule in $\mathcal{R}$;
2) $\forall p\cdot q \in \mathcal{FP}os(L)$ such that $q \geq_{\mathcal{P}} F^{nl}_G$, $L|_{p\cdot q}$ is not unifiable with any linearized left-hand side of rule in $\mathcal{R}$.

Any pattern L unifies with a renaming of itself at the root, but in that case, it is easy to see that no check is to be performed. Likewise, any two linear rules satisfy linear independence, and, since patterns are β-normal and the β-rule is left-linear, $\mathcal{R}\cup\beta$ is linearly independent if so is $\mathcal{R}$.

Example 3: Let $\mathcal{R} = \{f(X, f(X, Y) \to R, f(Z, Z) \to D\}$. Checking linear independence requires considering all three possibilities of unifying one (linearized) left-hand side with a subterm of the (linearized) other:

1) $f(X, f(X, Y))$ with itself at position 2, giving the solvable equation $f(X^1, f(X^{2\cdot1}, Y)) = f(X'^{2\cdot1}, Y')$. No check is needed here, the condition is satisfied;
2) $f(Z, Z)$ with $f(X, f(X, Y))$ at position 2, giving the solvable equation $f(Z^1, Z^2) = f(X^{2\cdot1}, Y))$. Again, no check is to be performed.
3) $f(Z, Z)$ with $f(X, f(X, Y))$ at position Λ, giving the solvable equation $f(Z^1, Z^2) = f(x^1, f(X^{2\cdot1}, Y))$. Then, we need to check whether $f(X, Y)$ is unifiable with any one of the two (linearized) left-hand sides, and indeed, it is with both $f(Z^1, Z^2)$ and $f(X'^1, f(X'^{2\cdot1}, Y'))$.

$\mathcal{R}$ is thus not linearly independent, but replacing $f(X, f(X, Y))$ by $f(X, g(X, Y)$ yields a linearly independent set.

Our new assumption is therefore that

- **(H5)** $\mathcal{R}$ is linearly independent.

We need yet another property of the linear unification problem $L^{lin}|_p = G^{lin}$:

Definition 11: Two rules $L \to R, G \to D$ such that the unification problem $L^{lin}|_p = G^{lin}$ has a solution σ which satisfies some equation of the form $X^o = u[X^q]$, where $u[X^q]$ is a non-variable term, are said to be in *linear occur-check*.

In that definition, X^o and X^q, following our notations, are two meta-variables renaming a same meta-variable X from L or G.

The existence of a linear occur check implies of course that $L|_p$ and G are not unifiable, the solutions of such equations being infinite rational terms. In the absence of linear occur check, let σ be the most general unifier of $L^{lin}|_p = G^{lin}$. The *occur-check* order on variables is defined by $X <_{oc} Y$ iff $X^o\sigma$ is a strict subterm of $Y^q\sigma$ for some positions o, q.

- **(H6)** No two rules of R_u are in linear occur check.

Note that if one of $L|p$ and G is linear, then $L \to R$ and $G \to D$ cannot be in linear occur-check.

A last assumption resembles Toyama's variable condition [2]:

- **(H7)** $L\to R$ satisfies TVC at $p\in\mathcal{FP}os(L)$ if the context $L[\_]_p$ and the subterm $L|_p$ of L at p have no meta-variable in common.

Under these assumptions, we can show that overlapping sub-rewriting peaks reduce to instances of critical peaks:

Lemma 9 (Overlapping peak lemma):

Let $\mathcal{R}$ be a rewriting system satisfying assumptions **(H1)** to **(H6)**, and confluent on $\mathcal{L}_{<n}$. Let $s \xleftarrow[L\to R,n]{q} u \xrightarrow[G\to D,m]{q\cdot p} t$ be a closed overlapping sub-rewriting local peak. Then, the equation $L|_p = G$ has most general unifier ρ s.t.

1) $v \xleftarrow[L\to R,n']{\Lambda} L\rho = L\rho[D\rho]_p \xrightarrow[G\to D,m']{p} w$ is a critical peak of $G \to D$ onto $L \to R$ at p
2) $s \underset{<n}{\longrightarrow\!\!\!\!\!\rightarrow} v\delta$ and $t \underset{<n}{\longrightarrow\!\!\!\!\!\rightarrow} w\delta$ for some closed substitution δ ($t \underset{<m}{\longrightarrow\!\!\!\!\!\rightarrow} w\delta$ if $L\to R$ satisfies **(H7)** at p)

Proof. By monotonicity, we can assume that $q = \Lambda$ wlog. The proof is illustrated at Figure 9.

By Lemma 6, $u = L^{lin}\sigma\tau$ with $\mathcal{D}om(\sigma) \subseteq \mathcal{MV}ar^{nl}(L)$, $\mathcal{D}om(\tau) \subseteq \mathcal{MV}ar^{l}(L)$, $\sigma \underset{<n}{\longrightarrow\!\!\!\!\!\rightarrow} \sigma'$ (by **(H3)**, **(H1)** and **(H2)**), and $u \longrightarrow\!\!\!\!\!\rightarrow L(\sigma' \cup \tau)s' \xrightarrow[L\to R]{\Lambda} R(\sigma' \cup \tau) = s$. By the same token, $u|_p = G^{lin}\gamma\theta$ with $\mathcal{D}om(\gamma) \subseteq \mathcal{MV}ar^{nl}(G)$, $\mathcal{D}om(\theta) \subseteq \mathcal{MV}ar^{l}(G)$, $\gamma \underset{<m}{\longrightarrow\!\!\!\!\!\rightarrow} \gamma'$ and $u|_p \longrightarrow\!\!\!\!\!\rightarrow G(\gamma'\cup\theta)t' \xrightarrow[G\to D]{\Lambda} D(\gamma'\cup\theta) = t$. It follows that $L^{lin}|p$ and G^{lin} unify with most general unifier σ. Let H be the term $L\sigma = L\sigma[G\sigma]_p$, and F_H its fringe.

All steps in the derivation from u to $L(\sigma' \cup \tau)$ must be at positions below F_H, since otherwise, the first one falsifying this property would falsify linear independence of $\mathcal{R}\cup\beta$, which follows from **(H5)**. This is true as well of the derivation from u to $L(\sigma \cup \tau)[G(\gamma' \cup \theta)]_p$. Therefore, s' and t' coincide with H at all positions above F_H, and since H is linear, $s' = H\delta$ and $t' = H\varphi$ for some substitutions δ and φ respectively.

We construct now derivations from s' and from t' to a common reduct u' which is at the same time an instance of H (since H is linear), of L, and of $L[G]_p$, hence unifying $L|_p$ and G. To this end, we need to equalize, for each meta

variable $X \in \mathcal{MV}ar(L) \cup \mathcal{MV}ar(G)$, and for each of its occurrences $X^o \in \mathcal{MV}ar(H)$, the values of $X^o\delta$ and $X^o\varphi$ when they differ, hence defining a term $H\rho = L\rho = L\rho[G\rho]_p$ (we use here the fact that H, L, G have no meta-variable name in common). Note that, despite the fact that we have equalized the non-linear variables only, a linear variable Y may have different values by δ and φ in case it is a variable of, say G, sitting above a non-linear variable X of L in the overlap.

The construction is by induction on the number of meta-variables of H that have a different value by δ and φ. If there is none, we are done. Otherwise, by **(H6)**, we select a maximal subset S of meta-variables of H which are both equivalent and minimal in the occur-check order. Such a subset can't be reduced to linear meta-variables, since the only way for such a Y to have different values by δ and τ is to depend upon a non-linear meta-variable X, and since Y must be minimal in the occur-check order, then $Y = X$ and X must belong to S by maximality. Therefore, by **(H4)** and **(H3)**, for all variables $X \in S$, $X\delta \in \mathcal{T}_{<n}$ and $X\varphi \in \mathcal{T}_{<m}$. Since all these terms occur at parallel positions, and $m \leq n$ by **(H1)**, we can use our confluence assumption for all those terms at once, hence defining ρ for all variables in S. Note that the derivation from s' (resp., t') operates at levels strictly below n. So does of course the derivation from t' to u', but in case **(H7)** is satisfied, all meta-variables of H that possibly have different values by δ and φ are meta-variables from G^{lin}, hence belong to $\mathcal{T}_{<m}$, and the derivation operates at levels strictly smaller than m. We then conclude the construction by induction hypothesis, yielding $U' = H\rho = L\rho = L\rho[G\rho]_p$, the derivations from s' and t' to u' satisfying the announced level requirements.

Since u' is an instance of both L and $L[G]_p$ then G unifies with L at G with most general unifier ρ, hence giving rise to the announced critical pair, so that $u' = L\sigma\rho$. We are left with routine calculations for commuting the two derivations originating from s' (resp., t'), and conclude the proof. ◊

F. Orthogonal functional reductions

Tait's proof of confluence for the lambda calculus is based on using what he calls parallel β-reductions, that are called orthogonal β-reductions in [7], a terminology we also prefer. Writing a term u as $\overline{u}^K\underline{u}_K$ for some set K of parallel positions of u, the idea is that an orthogonal rewrite step rewrites simultaneously in $\overline{u}^K$ and $\underline{u}_K$. Tait chooses an arbitrary splitting set K, while we choose a canonical one. This choice is a matter of convenience, parallel rewriting with left-linear non-overlapping rules being strongly confluent for all choices.

Definition 12: Let P be a set of m parallel positions, $K = \{k_i\}_i$ be a set of n positions such that $\forall k \in K\ \exists p \in P : k >_{\mathcal{P}} p$, and Q be a set of n sets of positions $\{Q_i\}_i$. Then we define $P \otimes_K Q = P \cup \{k_i \cdot q : i \in [1..n] \text{ and } q \in Q_i\}$.

Consider a set of positions P. We can always write $P = O \otimes_K Q$, where $O = P_{min}$ and $K = (P \setminus O)_{min}$, Q being then uniquely defined. The triple (O, K, P) is called *canonical splitting* of P.

Definition 13: Orthogonal β-rewriting the term u to the term v at a set of positions $P \subseteq \mathcal{P}os(u)$, denoted $u \overset{P}{\otimes\Longrightarrow} v$, is the smallest relation that contains parallel β-rewriting at P when P is a set of parallel positions and that otherwise satisfies the following property: let (O, K, Q) be a canonical splitting of P, then $v = w\theta$, where $\overline{u}^K \overset{O}{\Longrightarrow} w$ and $\underline{u}_K \overset{Q}{\otimes\Longrightarrow} \theta$.

Orthogonal functional reductions can of course be categorized by levels with the expected notation.

IV. Confluence

Preparation being now done, we can develop the confluence proof. We first recall the basics of van Oostrom's decreasing diagrams technique, before to carry out that proof by induction on the levels. Various properties of local sub-rewriting peaks are considered first, that is, disjoint peaks, ancestor peaks, and critical peaks, before to investigate local functional peaks and local heterogeneous peaks. The objective will always be the same: show that the considered local peak has a decreasing diagram. We will then be ready for the confluence proof itself.

As a very first important task, we need to make clear which rewrite relation is used in the confluence proof. Since

$$\xrightarrow[(\mathcal{R}\beta),n]{} \subseteq \underset{\leq n}{\otimes\Longrightarrow} \cup \underset{\mathcal{R},n}{\rhd\!\!\longrightarrow} \subseteq \xrightarrow[(\mathcal{R}\beta),\geq n]{}\!\!\!\!\!\!\rightarrow .$$

the rewrite relation in $\lambda\mathcal{F}$ is confluent on $\mathcal{L}$ iff the relation $\bigcup_{n\geq 0} \underset{\leq n}{\otimes\Longrightarrow} \cup \underset{\mathcal{R},n}{\rhd\!\!\longrightarrow}$ is confluent on $\mathcal{L}$. We will actually use an improved version of the latter:

Lemma 10: $\overset{O}{\underset{\leq n}{\otimes\Longrightarrow}} \subseteq \overset{P}{\underset{n}{\otimes\Longrightarrow}} \underset{\beta,<n}{\longrightarrow\!\!\!\!\!\!\rightarrow}$ with $P \subseteq O$.

Proof. Let $u \overset{O}{\underset{\leq n}{\otimes\Longrightarrow}} v$. The proof is by induction on $|O|$. If O is empty, the result is clear. Otherwise, $Q = \{o \in O : u|_o \in \mathcal{L}_n\} \neq \varnothing$. Note that $Q \subseteq O_{min}$. Then, $u \overset{Q}{\underset{n}{\otimes\Longrightarrow}} u' \overset{O\setminus Q}{\underset{\leq n}{\otimes\Longrightarrow}}$, using **(H3)** for those redexes in u' that have been instantiated by the rewrite at Q. We conclude by induction hypothesis. ◊

So, the rewrite relation we are interested in is indeed:

$$\bigcup_{n\geq 0} \underset{\beta,<n}{\longrightarrow\!\!\!\!\!\!\rightarrow} \cup \underset{\beta,n}{\otimes\Longrightarrow} \cup \underset{\mathcal{R},n}{\rhd\!\!\longrightarrow}$$

A. Decreasing diagrams

In the following, we consider rewrite relations all of whose elementary steps are equipped with a label belonging to some well-founded set.

Definition 14 (Decreasing diagram [19]): Given a labeled relation $\longrightarrow$ on an abstract set equipped with a well-founded order $\rhd$, we denote by $DS(m, n)$ the set of *decreasing rewrite sequences* of the form $u \xrightarrow{\delta}\!\!\!\!\!\!\rightarrow v$ or $u \xrightarrow{\gamma}\!\!\!\!\!\!\rightarrow s \xrightarrow{n} t \xrightarrow{\delta}\!\!\!\!\!\!\rightarrow v$ such that the labels in γ and δ are strictly smaller in the order than, respectively, m, and, m or n. The steps labeled by γ, n and δ, are called the *side* steps, *facing* step and *middle* steps of the decreasing sequence, respectively.

Given a local peak $v \xleftarrow{m} u \xrightarrow{n} w$, a *decreasing (rewrite) diagram* is a pair of derivations from v and w to some common term t, belonging to $DS(m, n)$ and $DS(n, m)$, respectively.

Decreasing rewrite diagrams are represented in Figure 1 and abbreviated as DDs. Note that a facing step of a decreasing

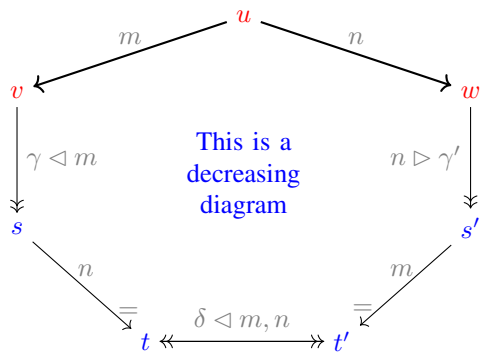

Fig. 1. Decreasing diagram

diagram may be missing, its side steps are then absorbed by the middle ones. A decreasing diagrams is *simple* if it reduces to its middle steps. A more general notion of decreasing diagram appears in [20], we won't need it here.

Theorem 1 ([19]): A labeled relation is Church-Rosser if all its local peaks have DDs.

The careful reader could object that our rewriting relations work modulo $=_\alpha$ on terms, rather than on $=_\alpha$-equivalence classes of terms, and therefore we should have explicit $=_\alpha$-steps, with a minimal label as required in the corresponding extension of van Oostrom's theorem to rewriting modulo [10]. Renaming of bound variables being so specific an equivalence, we allow ourselves this little glitch, as is customary.

Our coming confluence proof is by induction on levels, assuming the relation $\underset{\mathcal{R}\beta,<n}{\longrightarrow}$ is confluent. Local peaks to be considered are between those three relations mentioned before, and will always involve (at least) one step at level exactly n.

The label of a rewrite step will be a pair whose first argument is the level of that step, and the second, the name of the rule that is used (a technique called rule labeling). Arguments will be compared lexicographically, hence DDs when considering the level only will again be DDs with pairs as labels. Rule names will be compared in a well-founded order $>_\mathcal{R}$, such that the β-rule's name 0 is maximal in $>_\mathcal{R}$. Rule names being non-zero natural numbers, we feel sorry that this might be felt counter-intuitive.

B. Decreasing diagrams for disjoint peaks

Disjoint peaks commute for all monotonic relations, as is the case here of $\bigcup_{n\geq 0} \underset{\beta,<n}{\longrightarrow\!\!\!\!\!\rightarrow} \cup \underset{n}{\otimes\!\!\!\!\!\Longrightarrow} \cup \underset{\mathcal{R},n}{\triangleright\!\!\!\!\!\longrightarrow}$, yielding a diagram reduced to its facing steps, which is therefore decreasing.

C. Decreasing diagrams for functional peaks

They are of three kinds that we consider in turn.

Two orthogonal rewrite steps originating from the same term commute [18]: $s \underset{n}{\Longleftarrow\!\!\!\!\!\otimes} u \underset{n}{\otimes\!\!\!\!\!\Longrightarrow} t$ implies $s \underset{\leq m'}{\otimes\!\!\!\!\!\Longrightarrow} u \underset{\leq n'}{\Longleftarrow\!\!\!\!\!\otimes} t$ with $n' \leq n$ and $m' \leq m$ by **(H3)**, which is a decreasing diagram.

The same argument applies to peaks $s \underset{\beta,<n}{\longleftarrow\!\!\!\!\!\!\!\!\!\!\!\!} u \underset{n}{\otimes\!\!\!\!\!\Longrightarrow} t$.

The induction hypothesis takes care of $\underset{\beta,<n}{\longrightarrow\!\!\!\!\!\rightarrow}$-local peaks.

D. Decreasing diagrams for ancestor sub-rewriting peaks

We consider now the case where a sub-rewriting rewrite step takes place below some meta-variable X of a rule $L \to R$ used by the other rewrite. These peaks are therefore of two kinds, depending whether the meta-variable X occurs linearly or non-linearly in L.

Lemma 11: Assume $\underset{\mathcal{R}\beta,<n}{\longrightarrow}$ is confluent, and let $u \underset{\mathcal{R},n}{\overset{p}{\longleftarrow\!\!\!\!\!\triangleleft}} t \underset{\mathcal{R},m}{\overset{p\cdot o\cdot q}{\triangleright\!\!\!\!\!\longrightarrow}} v$ with $L(o) = X$ being a linear variable of L. Then u and v are joinable by a DD.

Proof. In that case, $m \leq n$ and the two steps commute, as for a disjoint peak, since equalization steps don't apply to linear variables. The obtain diagram is therefore decreasing.

Lemma 12: Assume $n \neq 0$, $\underset{\mathcal{R}\beta,<n}{\longrightarrow}$ is confluent, and let $s \underset{\mathcal{R},n}{\overset{p}{\longleftarrow\!\!\!\!\!\triangleleft}} u \underset{\mathcal{R},m}{\overset{p\cdot 0\cdot q}{\triangleright\!\!\!\!\!\longrightarrow}} v$ with $L(o) = X$ being a non-linear meta-variable of L. Then u and v are joinable by a DD.

Proof. By monotonicity, we can assume wlog that $p = \Lambda$. By lemma 6, $u = L^{lin}\sigma \underset{<n}{\overset{\geq_\mathcal{P} F_L}{\longrightarrow\!\!\!\!\!\rightarrow}} s' = L\tau \underset{n}{\overset{\Lambda}{\longrightarrow}} s = R\tau$. By assumption, $X\sigma \underset{m}{\overset{q}{\longrightarrow}} X\gamma$. By **(H4)**, $m < n$. By confluence assumption, $X\tau \underset{<n}{\longrightarrow\!\!\!\!\!\rightarrow} v \underset{<n}{\longleftarrow\!\!\!\!\!\!\!\!\!\!\!\!} X\gamma$. Let θ be identical to τ except for X for which its value is v. Then, $s = R\tau \underset{<n}{\longrightarrow\!\!\!\!\!\rightarrow} R\theta \underset{k}{\overset{\Lambda}{\longleftarrow\!\!\!\!\!\!\!\!\!\!\!\!}} L\theta \underset{<n}{\longleftarrow\!\!\!\!\!\!\!\!\!\!\!\!} t$. Since $\tau \longrightarrow \theta$, $k \leq n$ by **(H3)**. We have got a DD. ◊

E. Decreasing diagrams for heterogeneous peaks

We now consider local peaks made of a rewrite step with $\mathcal{R}$ on one side, and functional steps on the other side. Many cases must be considered. We start with $\underset{\beta,<n}{\longrightarrow\!\!\!\!\!\rightarrow}$ which is easy:

Lemma 13: Assume $n \neq 0$, $\underset{\mathcal{R}\beta,<n}{\longrightarrow}$ confluent, and let $s \underset{L\to R,n}{\overset{p}{\longleftarrow\!\!\!\!\!\triangleleft}} u \underset{\beta,<n}{\longrightarrow\!\!\!\!\!\rightarrow} t$. Then s and t are joinable be a DD.

Proof. We prove that $s \underset{\beta,<n}{\longrightarrow\!\!\!\!\!\rightarrow} v \underset{L\to R}{\overset{p}{\longleftarrow\!\!\!\!\!\triangleleft}} t$ for some v, a DD, by induction on the number of β-steps. Since β-steps operate on $\mathcal{L}_{<n}$, they cannot occur at positions above p. The induction step has therefore three cases, whether the β-step occurs at a position $q\#p$, below a linear variable of L (same proof as for Lemma 11), or below a non-linear variable of L (same proof as for Lemma 12).

We are left with a different kind of a local peak involving an orthogonal β-step.

Lemma 14: Assume $n \neq 0$, $\underset{\mathcal{R}\beta,<n}{\longrightarrow}$ confluent, and let $s \underset{L\to R,m}{\overset{p}{\longleftarrow\!\!\!\!\!\triangleleft}} u \underset{k}{\overset{O}{\otimes\!\!\!\!\!\Longrightarrow}} t$, with $max(m,k) = n$. Then, we have two different cases:

- $s \underset{k}{\otimes\!\!\!\!\!\Longrightarrow} \underset{<k}{\longrightarrow\!\!\!\!\!\rightarrow} \underset{\mathcal{R},n}{\overset{p}{\longleftarrow}} t$ if $k < n$;
- $s \underset{n}{\otimes\!\!\!\!\!\Longrightarrow} \underset{<n}{\longrightarrow\!\!\!\!\!\rightarrow} \underset{\mathcal{R},m}{\overset{p}{\Longleftarrow}} t$ if $k = n$.

Further, both diagrams are decreasing.

Proof. Since the pattern L is β-normal and the β-rule is left-linear, the positions in O can only be disjoint from p, below a variable of L, or strictly above p, so that $O = O_\# \uplus O_l \uplus$

$O_{nl} \uplus O_p$. Accordingly, we split the orthogonal step into four, so that $u \underset{k}{\overset{O_\#}{\otimes\!\!\Longrightarrow}} u' \underset{k}{\overset{O_l}{\otimes\!\!\Longrightarrow}} s' \underset{k}{\overset{O_{nl}}{\otimes\!\!\Longrightarrow}} w \underset{k}{\overset{O_p}{\otimes\!\!\Longrightarrow}} t$, and by **(H1)**, $k \geq m$ hence $n = k$ if $O_p \neq \varnothing$.

Disjoint steps commute, hence $s \underset{k}{\overset{O_\#}{\otimes\!\!\Longrightarrow}} s' \underset{\mathcal{R},m}{\overset{p}{\Longleftarrow}} u'$. The step at p commutes as well with the steps below linear variables of L, hence $s' \underset{k}{\overset{O'_l}{\otimes\!\!\Longrightarrow}} v' \underset{\mathcal{R},m}{\overset{p}{\Longleftarrow}} v$. O'_l may of course be different from O_l, but $O'_l \# O_\#$ since the former are below p and the latter incomparable to p. As for the next step, we get $v' \underset{k}{\overset{O'_{nl}}{\otimes\!\!\Longrightarrow}} w' \underset{\mathcal{R},m}{\overset{p}{\Longleftarrow}} w$. Again, O'_{nl} may differ from O_l, but again, $O'_{nl} \#(O_\# \uplus O_l)$, since different variables occur at unrelated positions in a term. The last step is slightly different, since positions in O_p stand above p, hence above $O_l \uplus O_{nl}$ as well as above $O'_l \uplus O'_{nl}$. Since the β-rewrite rule is left-linear but not right-linear in general, we get $w' \underset{k}{\overset{O_p}{\otimes\!\!\Longrightarrow}} t' \underset{\mathcal{R},m}{\overset{p}{\Longleftarrow}} t$. To be formal, we should use splitting here.

Let now $O' = O_\# \uplus O'_l \uplus O'_{nl} \uplus O_p$, Then, $s \underset{k}{\overset{O'}{\otimes\!\!\Longrightarrow}} t'$. We therefore get $w' \underset{n}{\overset{O'}{\otimes\!\!\Longrightarrow}} t' \underset{\mathcal{R},m}{\overset{p}{\Longleftarrow}} t$ if $k = n$ and $w' \underset{k}{\overset{O'}{\otimes\!\!\Longrightarrow}} w' \underset{\mathcal{R},m}{\overset{p}{\Longleftarrow}} t$ if $n > k$. In the former case, the diagram is decreasing because the β-rule is maximal in $\rhd$. In the latter, levels suffice to show decreasingness. $\Diamond$

F. Decreasing diagrams for overlapping sub-rewriting peaks

Lemma 9 tells us that any overlapping peak reduces to a closed instance of a critical higher-order peak. To get a decreasing diagram for an overlapping peaks, we therefore need to replace the instance of the critical higher-order peak by its corresponding decreasing diagram, assuming it has one. What we can compute, on the other hand, is a joinability diagram for the critical peak, we therefore need to instantiate that diagram before sticking it in, and get a decreasing diagram for the overlapping peak. Instantiating any joinability diagram by a well-layered substitution, however, may not result in a decreasing diagram, even if this joinability diagram *lokks* decreasing, since instantiating a rewrite step may change its level. And, of course, the joinability diagram rewrites open terms, which don't belong to $\mathcal{L}$, its rewrite steps don't have levels. This explains the need for the coming definition:

Definition 15: Assume that $s \underset{L\to R}{\overset{\Lambda}{\longleftarrow}} u \underset{G\to D}{\overset{p}{\longrightarrow}} t$ is a critical peak. The joinability diagram $s \underset{\mathcal{R}\beta}{\overset{}{\twoheadrightarrow}} \underset{\mathcal{R}\beta}{\overset{}{\twoheadleftarrow}} t$ is said to be *stable decreasing* if, for all closed substitutions σ such that u is layered, $s\sigma \overset{\mathcal{R}}{\twoheadrightarrow} \overset{\mathcal{R}}{\twoheadleftarrow} t\sigma$ is a DD for $s\sigma \underset{L\to R}{\overset{\Lambda}{\longleftarrow}} u\sigma \underset{G\to D}{\overset{p}{\longrightarrow}} t\sigma$.

Lemma 15 (Critical peak lemma):

Let $\mathcal{R}$ a rewrite system satisfying **(H1)** to **(H7)** such that $\underset{\mathcal{R}\beta,<n}{\longrightarrow}$ is confluent, all of whose critical peaks are joinable by a stable DD if **(H7)** is satisfied, otherwise a stable simple DD.

Then, overlapping sub-rewriting peaks $s \underset{\mathcal{R},n}{\overset{p}{\longleftarrow\!\!\lhd}} u \underset{\mathcal{R},m}{\overset{q}{\rhd\!\!\longrightarrow}} t$ are joinable by a DD.

Proof. We can assume wlog that $q \in p \cdot \mathcal{FP}os(L)$. By Lemma 9, there are terms u', v, w such that $s \underset{<n}{\overset{}{\twoheadrightarrow}} v$, $t \underset{<n}{\overset{}{\twoheadrightarrow}} w$ ($t \underset{<m}{\overset{}{\twoheadrightarrow}} w$ if **(H7)** is not satisfied), and $v \underset{n}{\overset{\Lambda}{\longleftarrow}} u' \underset{m}{\overset{p}{\longrightarrow}} w$ is an instance of the critical pair of $G \to D$ onto $L \to R$ at position p. By assumption and stability, this instance has a DD.

By **(H2)**, $m \leq n$. We assume first that $m < n$.

If **(H7)** is not satisfied, then the critical pair's DD has no facing step, and the whole joinability diagram $s \underset{<n}{\longrightarrow} v \underset{<n}{\twoheadrightarrow} \underset{<n}{\longleftarrow} w \underset{<n}{\twoheadrightarrow} t$ is therefore decreasing.

If **(H7)** is satisfied, the DD has a facing step, hence the whole joinability diagram $s \underset{<n}{\longrightarrow} v \overset{<n}{\twoheadrightarrow} \underset{<n}{\longleftarrow} \underset{n}{\longleftarrow} \underset{<n}{\longleftarrow} w \underset{<m}{\longleftarrow}$ is decreasing again since $m < n$.

The slightly different case $m = n$ is left to the reader. $\Diamond$

G. Confluence proof

We are now ready for carrying out the confluence proof. We have considered all kinds of local peaks of the relation $\bigcup_n \underset{\beta,<n}{\overset{}{\twoheadrightarrow}} \cup \underset{\beta,n}{\overset{}{\otimes\!\!\Longrightarrow}} \cup \underset{\mathcal{R},n}{\overset{}{\rhd\!\!\longrightarrow}}$, and all have a decreasing diagram with respect to the labelling on elementary proof steps. Hence, by van Oostrom's theorem, this relation is confluent on $\mathcal{L}$, provided it is confluent on $\mathcal{L}_0$, and therefore, $\underset{\mathcal{R}\beta}{\longrightarrow}$ is confluent on $\mathcal{L}$ iff it is confluent on $\mathcal{L}_0$. This shows the main result of this paper:

Theorem 2: Let $\mathcal{R} = R_c \cup R_{\mathcal{U}}$ be a set of rewrite rules satisfying our assumptions **(H1)** to **(H7)**. Assume further that R_c defines a confluent rewrite relation on confined terms, and that all critical peaks $s \underset{L\to R}{\overset{\Lambda}{\longleftarrow}} u \underset{G\to D}{\overset{p}{\longrightarrow}}$ of $\mathcal{R}$ involving at least one rule of $R_{\mathcal{U}}$ have a stable DD with respect to our order on proof steps. Then, $\lambda\mathcal{F}$ has a confluent rewrite relation $\underset{\mathcal{R}\beta}{\longrightarrow}$.

This result generalizes several other recent results.

Assume that R_u is a set of left-linear rules. Then, two layers of terms are enough, **(H1)**, **(H2)** and **(H3)** are trivially satisfied since all non-confined terms inhabit $\mathcal{L}_1$ and **(H4)** is void. Futher, all decreasing diagrams wrt rule-labelling are stable decreasing, since in this case the level of a rewrite step is entirely determined by the head of the redex, hence does not change by instantiation. We then obtain the result of [1].

Assume further that R_c is empty. Then the set of confined terms, $\mathcal{L}_0$, is the empty set, and the second layer $\mathcal{L}_1$ is the set $\mathcal{T}$ of all closed terms. This shows that confluence of the pure λ-calculus is preserved when adding left-linear higher-order rewrite rules whose critical pairs have decreasing joinability diagrams with respect to rule labeling. Indeed, linear independence **(H5)** and absence of linear occur-checks **(H6)** are always satisfied by a set of left-linear rewrite rules, as pointed out in Section III-E, and **(H7)** is void in case all rules are left-linear. This result is to be compared with a similar result obtained in [7], whose proof requires R_u to be terminating on pure λ-terms, the order on rule names being exactly the same. It generalizes also the confluence result of [15], which is restricted to simply typed λ-terms and a set R_u of terminating rules, apart from the fact that the joinability diagram needs not be decreasing wrt rule labelling, since it is already decreasing wrt higher-order rewriting.

This shows the strength of Theorem 2.

H. Checking joinability diagrams for stable decreasingness

In addition to the assumptions **(H1)** to **(H7)**, we have assumed that each critical peak has a joinability diagram that is stable decreasing, that is, all of whose ground instances are decreasing diagrams. This does not come for free. Computing a joinability diagram is easy. Showing it is stable decreasing is a different matter, and will have to be checked one by one. In practice, this is not that hard, and we suggest here sufficient conditions that can be easily implemented.

Let $s \underset{i:L\to R}{\overset{\Lambda}{\leftarrow}} u \underset{j:G\to D}{\overset{p}{\rightarrow}} t$ be a critical peak. We assume now that the meta-variables, or rather the pre-redexes, in the critical peak have themselves levels, let LA be a level assignment for them so that u has a level. Then, by **(H2)**, s and t and all their reducts have levels as well, hence, $s \underset{i,n}{\overset{\Lambda}{\leftarrow}} u \underset{j,m}{\overset{p}{\rightarrow}} t$ in LA, with $n \geq m$ by **(H2)**.

We consider first the case where the joinability diagram has the form $s \underset{k}{\overset{o}{\rightarrow}} \cdot \underset{l}{\overset{q}{\leftarrow}}$. Then, in LA, $s \underset{k,m'}{\overset{o}{\rightarrow}} \cdot \underset{l,n'}{\overset{q}{\leftarrow}}$, and since $u \in \mathcal{L}_n$, $m \geq m'$ and $n \geq n'$ by **(H1)**. For this joinability diagram to be decreasing, we need either one of the three following conditions: (i)$n > m', n \geq n'$, or (ii) $n = m, n > n', n \geq m'$, or (iii) $n = m = n' = m'$ and $\{i, j\} \rhd_{mul} \{k, l\}$, where the index mul denotes the multiset extension of the order on rule names.

Consider (i) first. By **(H2)**, $n \geq n'$, but $n > m'$ needs by checked. In practice, this follows from the levels assigned to the signature by the level typing system described in Section V. Consider now (ii). Again, $n \geq m'$ follows by **(H2)**, and $n > n'$ should be checked. For (iii), it is enough to check the condition on rule names. So, this joinability diagram is decreasing if for every level assignments, either one of the three following conditions is satisfied: (i) $n > m'$ (ii) $n > n'$ (iii) $\{i, j\} \rhd_{mul} \{k, l\}$. Of course, (iii) is nice since it does not refer to the levels. But it may not always work, so the other two are useful too.

This reasoning can of course be extended to more complex joinability diagrams. In particular, condition (iii) extends easily: let $i_1, \ldots, i_n$ be the multiset of rule names occurring in the joinability diagram. Then $\{i, j\} \rhd \{i_1, \ldots, i_n\}$ ensures that this joinability diagram is stable decreasing.

V. Layering

The aim of the layering construction is to eliminate the terms breaking confluence while retaining as many as possible of those that don't. We should further keep in mind that this development is meant to be implemented, and should therefore cooperate with our favourite dependent type system, whose aim is also to eliminate (non-typable) terms. We will therefore syntactically define layers by means of a typing system which can easily be blended with the type-checking procedure of the type system.

A. Term layering

A first step toward a stratification of terms satisfying our assumptions is to have an annotation of variables with the natural number layer they belong to. Since we need to distinguish free from bound variables, we shall write an abstraction as in $\lambda x^n : u.t$, where n is a non-zero level, and on the other hand declare a free variable x as in $x : n$, as already said. In the latter case, $n = 0$ is possible in order to have variables in confined terms. When taking a subterm of $\lambda x^n : t.u$, we need of course to declare the level n of the variable x which may now occur free in t.

Symbols need be assigned a more complex layering expression so as to allow mixing layers together in a controlled manner. Consider for instance the case of the non-linear rewrite rule $\mathbf{F}(X, X) \to t$. Well-layered instances of the left-hand side of this rule are expected to replace X by terms of a strictly lower level. A layering restriction ensuring this property is $\forall t, u \in \mathcal{L}_{\leq n}, \mathbf{F}(t, u) \in \mathcal{L}_{n+1}$. In the general case, we need annotating all symbols $\mathbf{G}$ of arity p with a tuple of $p+1$ levels $(k_0, \ldots, k_p)$. In order to satisfy **(H1)**, it is of course critical to ensure that the ultimate codomain's level is greater than all levels in the tuple.

Definition 16: A *level-type* of arity n is a $n+1$-tuple of natural numbers $(k_0, \ldots, k_n)$, written $k_0 \to \cdots \to k_n$, such that $\forall i, k_n \geq k_i$.

Definition 17: A *layering assignment* is composed of

- For each free variable $x \in \mathcal{X}$, an assignment $x : n$ to some layer $n > 0$;
- For each confined symbol $\mathbf{G} \in \mathcal{F}_c$ of arity n, the assignment $\mathbf{G} : 0 \to \cdots \to 0$, level-type of arity n;
- For each non-confined symbol $\mathbf{G} \in \mathcal{F}_u$ of arity n, the assignment $\mathbf{G} : k_0 \to \cdots \to k_n$, level-type of arity n such that $\forall i \neq n : k_i \geq 0$, and $k_n > 0$.

Definition 18: Assuming given a layering assignment, we define the following layering type system assigning terms to layers:

$$\frac{x : n}{x \in \mathcal{L}_n} \qquad \frac{t \in \mathcal{L}_n \qquad u \in \mathcal{L}_k}{(t\ u) \in \mathcal{L}_{\max(n,k,1)}} \qquad \frac{t \in \mathcal{L}_{\leq n} \qquad u \in \mathcal{L}_{\leq n}}{\lambda x^n : t.u \in \mathcal{L}_n}$$

$$\frac{\mathbf{G} : k_1 \to \cdots \to k_p \to n \qquad \forall i : u_i \in \mathcal{L}_{\leq k_i}}{\mathbf{G}(u_1, \ldots, u_p) \in \mathcal{L}_n}$$

When typing an abstraction, we need of course to modify the layering assignment for typing u by adding the assignment $x : n$ and removing, if necessary, the previous assignment for the same variable x.

Example 4: Terms of the pure λ-calculus are well-layered, and all belong to layer $\mathcal{L}_n$ if all their variables, whether free or bound, are assigned the same layer n. We can therefore have one copy of the pure λ-calculus at each layer. We can also have λ-terms that mix layers. For instance assuming A, B are terms inhabiting layer k, the annotated λ-term $\Delta_n = \lambda x^n : A.(x\ x)$ inhabits layer n provided $k \leq n$, and $(\Delta_n\ \Delta_n)$ then inhabits layer n as well.

Assuming now the unary symbol $\mathbf{F}:n \to n+1$ declared in the layering assignment, the term $\lambda y^{n+1}:A.\,(y\ \ \mathbf{F}(\lambda x^n:B.x))$ belongs to layer $\mathcal{L}_{n+1}$ for all $A \in \mathcal{L}_{\leq n+1}$ and $B \in \mathcal{L}_{\leq n}$. The same term becomes ill-layered if $B \in \mathcal{L}_{n+1}$. $\Diamond$

The following properties hold for all layering systems:

Lemma 16: Let $t \in \mathcal{L}_n$.

1) (Partition): $\{L_m\}_{m\geq 0}$ is a partition with $\mathcal{L}_0 = \mathcal{T}_c$;
2) (Subterm) If $t \rhd u$, then $u \in \mathcal{L}_{\leq n}$;
3) (Stability) If $u \in \mathcal{L}_{\leq i}$ then $t\{x^i \mapsto u\} \in \mathcal{L}_{\leq n}$;
4) (Beta) If $t \underset{\beta}{\longrightarrow} u$ then $u \in \mathcal{L}_{\leq n}$.

Proof.

0) By induction on m first, and secondly on the typing of terms of level n. For the base case, we show that $\mathcal{L}_o = \mathcal{T}_c$. This is because the only rules that apply are the first one with $n = 0$ and the last one with a layering declaration of the form $\mathbf{G} : 0 \to \ldots \to 0$). Other rules don't apply by definition of a layering assignment.
   For the induction step, we only need to show that terms well-layered in $\mathcal{L}_{n+1}$ can't have been assigned a strictly lower level. This is true with the first and last rules, since a symbol can't have two different declarations in a layering assignment. This true of the abstraction, whose level n is determined by its variable x^n. Finally this is true as well of applications: by induction hypothesis, this is true of t and u. If both are confined terms, then the application is in $\mathcal{L}_1$, otherwise in $\mathcal{L}_{max(n,k)}$.
1) By induction on t and definition of level-types. The result holds trivially in the base case $t = u$ and follows from the induction hypothesis in all other cases.
2) By induction on the size of t. If $t = x^i$, then $i = n$ and $t\{x^i \mapsto u\} = u \in \mathcal{L}_{\leq n}$. The other cases follow from the induction hypothesis.
3) By induction on the size of t. If the β-step occurs in a strict subterm we conclude by induction hypothesis. Otherwise, $t = (\lambda x^n:a.v\ \ w)$ and necessarily $u, v, w \in \mathcal{L}_{\leq n}$. If x^n doesn't occur in v, then $u = v \in \mathcal{L}_{\leq n}$. Otherwise $u = v\{w/x^n\}$ and we conclude by (Stability).

B. Example

Let us assume a single rule rewrite system $\mathcal{R} = \{\mathbf{F}(X,X) \to X\}$ and a layering assignement such that $\mathbf{F} : 0 \to 0 \to 1$.

Nested occurrences of $\mathbf{F}$ such as $\mathbf{F}(\mathbf{F}(t,t),\mathbf{F}(t,t))$ can't be type-checked by the layering type system even if t is a confined term. Indeed, $\mathbf{F}(t,t)$ will then inhabit $\mathcal{L}_1$ and the whole term becomes ill-layered.

In order to type terms with multiple occurrences of $\mathbf{F}$, $\mathbf{F}$ should instead be assigned a polymorphic level-type $\forall k : k \to k \to k+1$. Then $\mathbf{F}(t,t)$ will inhabit some layer $\mathcal{L}_k$ for some k depending upon which layer is inhabited by t, and the whole term will inhabit layer $k+1$.

Alternatively, it is possible to consider infinitely many overloaded versions, $\mathbf{F}_k$, of the $\mathbf{F}$ symbol, one for each natural number k, with the level-type declaration $k \to k \to k+1$. In our example: if t inhabits $\mathcal{L}_0$, then $\mathbf{F}^2(\mathbf{F}^1(t,t),\mathbf{F}^1(t,t))$ inhabits $\mathcal{L}_2$. Since these type declarations are all different, they satisfy the requirements of our layering type system.

In the latter system, the user needs to guess which layer an $\mathbf{F}$-headed term is going to belong to, while in the former, polymorphism does the job for us. Of course, we need to modify accordingly the last rule of the system by allowing for the polymorphic level-type declaration of $\mathbf{F}$.

As far as the properties of the layering system are concerned, both systems are of course equivalent. We can therefore prove them for the overloading system. First, properties of Lemma 16 hold for the overloaded layering system, which is just a particular case of the layering type system we have given, once symbols have been duplicated beforehand.

Lemma 17: The overloaded typing system defines layers $\{L_n\}$ that satisfy all properties **(H1)** to **(H7)**.

Proof. Note first that $\{\mathcal{L}_n\}_n$ is a partition of $\mathcal{L}$ by Lemma 16.

- **(H0)**: by Lemma 16(1).
- **(H1)**: by Lemma 16(2).
- **(H2)**: Holds for β by Lemma 16(4). It also holds for rewriting with the rule $\mathbf{F}(X,X) \to X$, since any instance $X\sigma$, assuming $\mathbf{F}^k(X,X)\sigma$ is well-layered, inhabits layer $k-1$ thanks to the level-type declarations.
- **(H3)**: by Lemma 16(3)
- **(H4)**: Assuming $\mathbf{F}^n(X,X')\sigma \in \mathcal{L}_{n+1}$, then $X\sigma \in \mathcal{L}_n$ thanks to the level-type declarations.
- **(H5)**, **(H6)** and **(H7)**: All easily checked since the only rule has no non-trivial subterm.

Theorem 3: Rewriting with the non-left-linear rule $\mathbf{F}(X,X) \to X$ preserves the confluence of β on well-layered terms generated by the level-type declaration $\mathbf{F} : \forall k : k \to k \to k+1$.

Proof. R_c is empty therefore confluent. There is no non-critical pair in $R_{\mathcal{U}}$. By Lemma 17, all assumptions **(H1)** to **(H7)** are satisfied. We conclude using Theorem 2.

To our knowledge, this is the first characterization of a set of terms containing all pure λ-terms for which the rule $\mathbf{F}(X,X) \to X$ preserves confluence of the β-rewrite rule.

We could of course consider all three rules of the introduction, even together, the result would still hold. The level-type declaration for the signature used in the last rule should be:

$$c, u : \forall k : k \to k+1 \to k+1$$

C. Extensions

The set of well-layered terms contains many terms that are not even typable. In particular all pure λ-terms are well-layered. This includes $\Delta_n := \lambda x^n.(x\ x)$, and even non-terminating terms such as the ill-typed $(\Delta\ \Delta)$ or fixpoint operators.

However some terms are excluded that are well-typed. Consider for instance the typing assignment $\mathbf{F} : A \to A \to A$. Then the term $\lambda x:A.\mathbf{F}(x,x)$ is easily typed with $A \to A$, yet no layer assignment for x allows it to be well-layered.

An easy way to extend the results of this paper to a wider set of terms would be to consider more relaxed layering inference rules. For instance, rather than n-arities, variables and symbols could be assigned arbitrary simple types whose base types

are natural numbers. The layering rules are then the typing rules of the simply typed λ-calculus which allow to derive $\lambda x\!:\!A.\mathbf{F}(x,x) : n \to n+1$. Layers can then be defined as the largest base type occurring in a simple type.

It is however critical to forbid *layer-decreasing* products from the considered set of simple types. Indeed, a term layer-typed by a layer-decreasing product, say $t : n+1 \to n$, would yield, when applied, a term $(t\ u)$ inhabiting $\mathcal{L}_n$, hence breaking **(H1)** since $u : n+1 > n$. Besides, in order to keep terms from the pure λ-calculus, such as Δ, this simple type system should allow arbitrary applications *within* a layer. This can be achieved by considering the following *equivalence* between simple types $n \equiv n \to n$ which allows to have $\Delta : n$.

In this extended setting, the term $\lambda z\!:\!A.\mathbf{F}(\mathbf{F}(z,z),\mathbf{F}(z,z))$ can be well-layered (at level 2) using two differently layered versions of $\mathbf{F}$. It is however worth noting that the following well-typed term t, which β-reduces to it, cannot be well-layered.

$$t = \lambda z\!:\!A.\,(\lambda y\!:\!(A\to A).(y\ (y\ z)\ (y\ z)) \quad \lambda x\!:\!A.\mathbf{F}(x,x))$$

Similarly the term $\lambda y\!:\!A\to A.\lambda z\!:\!A.\,(y\ \ \mathbf{F}(z,(y\ z)))$ is well-typed but no level-type can be assigned to the variable y that would make the whole term well-layered.

VI. Conclusion

Our confluence result is the first we know of which goes beyond Klop's unexpected observation that non-left-linear rules break the confluence property of the pure λ-calculus. As we have shown, it captures many existing others, including [1].

Besides being new, we believe that our result opens a research path sketched in Section V, whose goal is to accept more and more layered terms, until, possibly, all dependently-typed terms (hence including all closed instances of the left-hand sides of a given non-left-linear system) pass the test. Perhaps, this is not always possible, and it would then be important to know why.

On the practical side, it has been observed that higher-order non-left-linear rewrite rules are often hard to avoid when embedding rich logical systems into logical frameworks such as Isabelle or Dedukti. The case of polymorphic universes is a paradigmatic example in this respect. A preliminary, partial attempt to encode polymorphic universes was first considered in [1] using a confined level of first-order algebraic expressions encoding the universes which had non-left-linear rules, kept separated from the rest of the embedding system which used left-linear higher-order rules. The results of this paper extend this work by considering successive layers of higher-order terms and can therefore be used to show the confluence of more complex systems such as the ones introduced in [6].

It remains to be seen whether confluence on a restricted set of layered terms is enough to guarantee that a type system relying on rewrite rules is well-behaved. In an ideal world, all well-typed terms should be well-layered. In practice, however, having a restricted subset of layered terms is acceptable provided it is large enough to represent all expected objects, such as the image by a translation of a system we want to encode, for example, Coq with polymorphic universes.

Another research path would be to adapt to non-left-linear rules other confluence criteria developed for left-linear rules, for example assuming $\mathcal{R}$ is terminating [7], or allowing β-rewrite steps to be smaller than the other steps at some (not necessarily all) levels [8].

References

[1] Ali Assaf, Gilles Dowek, Jean-Pierre Jouannaud and Jiaxiang Liu. Untyped confluence in dependent type theories. draft hal-01515505, INRIA, january 2018. presented at HOR 2016, Porto. Available from http://dedukti.gforge.inria.fr/.

[2] Takahito Aoto, Junichi Yoshida, and Yoshihito Toyama. Proving confluence of term rewriting systems automatically. In Ralf Treinen, editor, *Rewriting Techniques and Applications, 20th International Conference, RTA 2009, Brasília, Brazil, June 29 - July 1, 2009, Proceedings*, volume 5595 of *Lecture Notes in Computer Science*, pages 93–102. Springer, 2009.

[3] Ali Assaf, Guillaume Burel, Raphaël Cauderlier, Gilles Dowek, Catherine Dubois, Frédéric Gilbert, Pierre Halmagrand, Olivier Hermant, and Ronan Saillard. Dedukti: a Logical Framework based on the lambda-pi-Calculus Modulo Theory. draft, INRIA, 2019.

[4] Frédéric Blanqui. Definitions by rewriting in the calculus of constructions. *Mathematical Structures in Computer Science*, 15(1):37–92, 2005.

[5] Jesper Cockx, Nicolas Tabareau, and Théo Winterhalter. The taming of the rew: a type theory with computational assumptions. *Proc. ACM Program. Lang.*, 5(POPL):1–29, 2021.

[6] G. Ferey. *Higher-Order Confluence and Universe Embedding in the Logical Framework*. PhD thesis, Université Paris-Saclay, Orsay, France, 2021. submitted.

[7] Gaspard Ferey and Jean-Pierre Jouannaud. Confluence in (Un)typed Higher-Order Type Theories I. draft. hal-, INRIA, march 2019. available from http://dedukti.gforge.inria.fr/.

[8] Gilles Dowek, Gaspard Ferey, Jean-Pierre Jouannaud and Jiaxiang Liu. Confluence in Untyped Higher-Order Theories by means of nested critical pairs. draft hal-, INRIA, january 2019. Full version of a work presented at HOR 2016.

[9] Healfdene Goguen. The metatheory of UTT. In Peter Dybjer, Bengt Nordström, and Jan M. Smith, editors, *Types for Proofs and Programs, International Workshop TYPES'94, Båstad, Sweden, June 6-10, 1994, Selected Papers*, volume 996 of *Lecture Notes in Computer Science*, pages 60–82. Springer, 1994.

[10] Jean-Pierre Jouannaud and Jiaxiang Liu. From diagrammatic confluence to modularity. *Theor. Comput. Sci.*, 464:20–34, 2012.

[11] Jan Willem Klop. *Combinatory reduction systems*. PhD thesis, CWI tracts, 1980.

[12] J.W. Klop. *Combinatory Reduction Systems*. Number 127 in Mathematical Centre Tracts. CWI, Amsterdam, The Netherlands, 1980. PhD Thesis.

[13] Jiaxiang Liu, Nachum Dershowitz, and Jean-Pierre Jouannaud. Confluence by critical pair analysis. In Gilles Dowek, editor, *Rewriting and Typed Lambda Calculi - Joint International Conference, RTA-TLCA 2014, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 14-17, 2014. Proceedings*, volume 8560 of *Lecture Notes in Computer Science*, pages 287–302. Springer, 2014.

[14] Jiaxiang Liu, Jean-Pierre Jouannaud, and Mizuhito Ogawa. Confluence of layered rewrite systems. In Stephan Kreutzer, editor, *24th EACSL Annual Conference on Computer Science Logic, CSL 2015, September 7-10, 2015, Berlin, Germany*, volume 41 of *LIPIcs*, pages 423–440. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.

[15] Richard Mayr and Tobias Nipkow. Higher-order rewrite systems and their confluence. *Theoretical Computer Science*, 192:3–29, 1998.

[16] Dale Miller. A logic programming language with lambda-abstraction, function variables, and simple unification. *Journal of Logic and Computation*, 1(4):497–536, 1991.

[17] Mitsuhiro Okada. Strong normalizability for the combined system of the typed lambda calculus and an arbitrary convergent term rewrite system. In Gaston H. Gonnet, editor, *Proceedings of the ACM-SIGSAM 1989 International Symposium on Symbolic and Algebraic Computation, ISSAC '89, Portland, Oregon, USA, July 17-19, 1989*, pages 357–363. ACM, 1989.

[18] Terese. Term rewriting systems. In *Cambridge Tracts in Theoretical Computer Science, Marc Bezem, Jan Willem Klop, and Roel de Vrijer editors*. Cambridge University Press, 2003.

[19] Vincent van Oostrom. Confluence by decreasing diagrams. *Theor. Comput. Sci.*, 126(2):259–280, 1994.

[20] Vincent van Oostrom. Confluence by decreasing diagrams converted. In Voronkov A., editor, *RTA*, volume 5117 of *Lecture Notes in Computer Science*, pages 306–320. Springer, 2008.

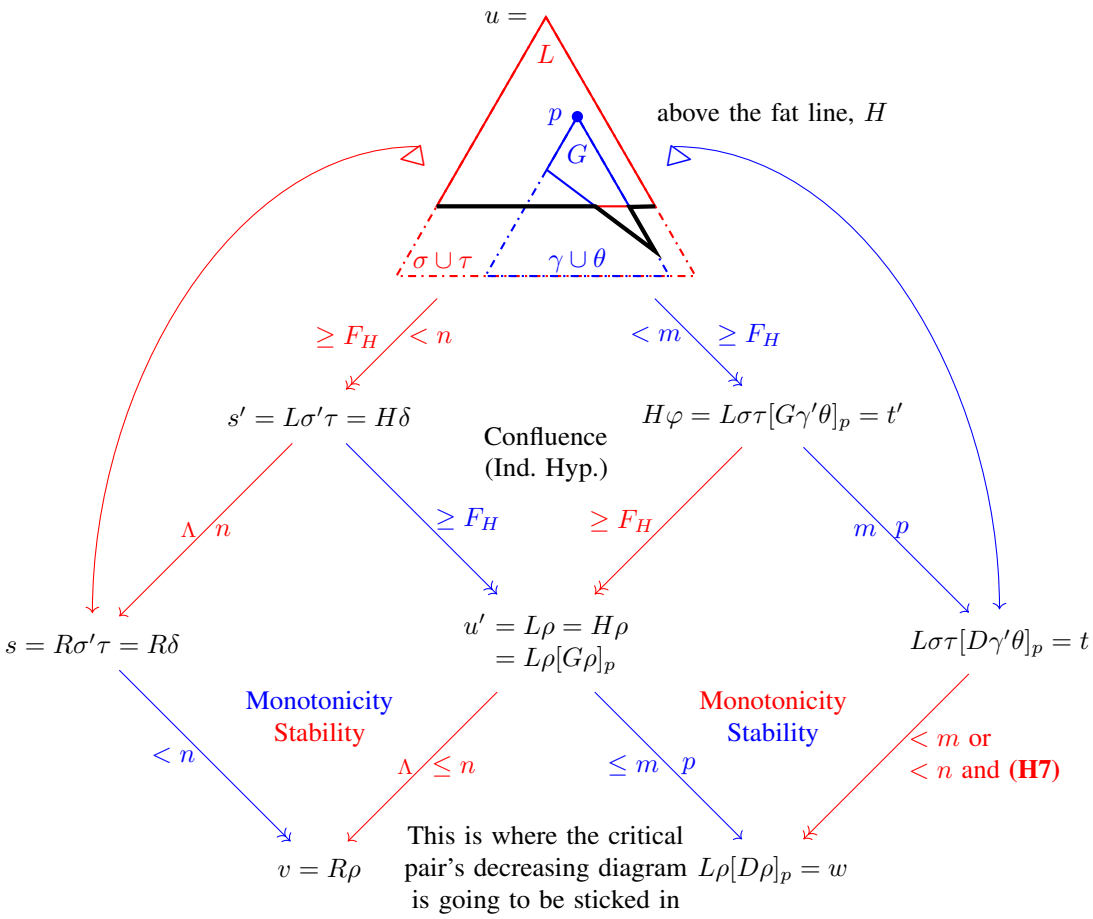

Fig. 2. Critical pair lemma: Lemma 9