Confluence of Orthogonal Nominal Rewriting Systems Revisited

Takaki Suzuki, Kentaro Kikuchi, Takahito Aoto, and Yoshihito Toyama

RIEC, Tohoku University,
2-1-1 Katahira, Aoba-ku, Sendai, Miyagi, 980-8577, Japan
{takaki, kentaro, aoto, toyama}@nue.riec.tohoku.ac.jp

Abstract

Nominal rewriting systems (Fernández, Gabbay & Mackie, 2004; Fernández & Gabbay, 2007) have been introduced as a new framework of higher-order rewriting systems based on the nominal approach (Gabbay & Pitts, 2002; Pitts, 2003), which deals with variable binding via permutations and freshness conditions on atoms. Confluence of orthogonal nominal rewriting systems has been shown in (Fernández & Gabbay, 2007). However, their definition of (non-trivial) critical pairs has a serious weakness so that the orthogonality does not actually hold for most of standard nominal rewriting systems in the presence of binders. To overcome this weakness, we divide the notion of overlaps into the self-rooted and proper ones, and introduce a notion of α-stability which guarantees α-equivalence of peaks from the self-rooted overlaps. Moreover, we give a sufficient criterion for uniformity and α-stability. The new definition of orthogonality and the criterion offer a novel confluence condition effectively applicable to many standard nominal rewriting systems. We also report on an implementation of a confluence prover for orthogonal nominal rewriting systems based on our framework.

1998 ACM Subject Classification F.4.1 Mathematical Logic

Keywords and phrases Nominal rewriting, Confluence, Orthogonality, Higher-order rewriting, α-equivalence

Digital Object Identifier 10.4230/LIPIcs.RTA.2015.301

1 Introduction

Expressive formal systems such as systems of predicate logics, λ-calculi, process calculi, etc. need variable binding. *Nominal rewriting* [5][3] is a framework that extends first-order term rewriting by a binding mechanism. Studies of nominal rewriting are preceded by extensive studies of a *nominal approach* to terms and unifications [6][13][17]. A distinctive feature of the nominal approach is that α-conversion and capture-avoiding substitution are not relegated to meta-level—they are explicitly dealt with at object-level. This makes nominal rewriting significantly different from classical frameworks of higher-order rewriting systems such as *Combinatory Reduction Systems* [8] and *Higher-Order Rewriting Systems* [9] based on 'higher-order syntax'.

Confluence is a fundamental property of rewriting systems. As expected, the first results on confluence of *nominal rewriting systems* (*NRSs* for short) are generalisations of two classical results on confluence, namely Rosen's criterion (orthogonal systems are confluent) and Knuth-Bendix's criterion (terminating and locally confluent systems are confluent) [3]. We notice, however, that the confluence criterion in [3] for orthogonal NRSs is not applicable to standard NRSs—as the orthogonality in [3] contains the emptiness of the root overlaps of equivariant rules obtained from the same rule (*self-rooted overlaps*), which does not hold if

26th International Conference on Rewriting Techniques and Applications (RTA'15).
Editor: Maribel Fernández; pp. 301–317

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

the system contains a rewrite rule with binders (cf. Remark at the end of Subsection 3.2). Moreover, in contrast to the first-order case, one cannot skip the joinability check of self-rooted overlaps (cf. Example 19)—thus, if one relaxes the definition of orthogonality to omit such overlaps, then confluence is not guaranteed.

The contributions of this paper are summarised as follows:

- Our rewrite relation does not allow α-equivalent terms on the result of rewriting. Accordingly, we come to study confluence properties of rewriting modulo α-equivalence, which enables us to perform fine-grained analysis where confluence modulo and Church-Rosser modulo are different properties. Such an approach was suggested in [19, page 220].
- We overcome the above-mentioned defect of the orthogonality in [3] by introducing a notion of α-stability, which guarantees α-equivalence of peaks from the self-rooted overlaps. We prove Church-Rosser modulo α-equivalence for the class of orthogonal nominal rewriting systems that are uniform and α-stable.
- We introduce a notion of abstract skeleton preserving (ASP) as a sufficient criterion for uniformity and α-stability. To show the α-stability of ASP rewrite rules, we prove some lemmas on the system of α-equivalence in the nominal setting, which seem to be new.
- We report on an implementation of a confluence prover for nominal rewriting systems based on our criterion, that is, orthogonality and ASP. To check the emptiness of the proper overlaps, we use equivariant unification [2] with one permutation variable.

While a rewrite system in [5][3] is defined as an infinite set of rewrite rules that is closed under equivariance, we define a rewriting system as a finite set of rewrite rules. Instead of appealing to the property of equivariance, we specify a permutation as a parameter in each rewrite relation. (The idea of specifying a permutation as a parameter is found also in [4].) This allows us to make a discussion on avoiding capture of a free atom (cf. the latter part of Example 10) without referring to the property of equivariance.

As regards related work, Vestergaard and Brotherston [18][19] study a confluence proof of λ-calculus with variable names, not in the nominal setting, where α-conversion is seriously taken into account. Their definition of confluence is that of the reflexive transitive closure of $\to_\alpha \cup \to_\beta$. Formalisation of a confluence proof of first-order orthogonal term rewriting systems has been studied, e.g. in [11]. Our proof of Church-Rosser modulo α-equivalence can be seen as an extension of an inductive confluence proof of first-order orthogonal term rewriting systems (e.g. [16, Section 4.7] and [7]).

The organisation of the paper is as follows. In Section 2, we explain basic notions and notations of nominal rewriting. In Section 3, we discuss problems on confluence in nominal rewriting, and prove confluence of a class of nominal rewriting systems. In Section 4, we give a sufficient criterion for the class, and conclude in Section 6.

2 Nominal rewriting

Nominal rewriting [5][3] is a framework that extends first-order term rewriting by a binding mechanism. In this section, we redefine nominal rewriting systems as finite sets of rewrite rules, and introduce a notion of rewrite relation that is related to but different from the rewrite relation defined in [5][3]. In the subsequent sections, we will study confluence properties on our notion of rewrite relation.

2.1 Nominal terms

First, we introduce notions and notations concerning nominal terms.

A *nominal signature* Σ is a set of fixed arity *function symbols* ranged over by $f, g, \ldots$. We fix a countably infinite set $\mathcal{X}$ of *variables* ranged over by $X, Y, Z, \ldots$, and a countably infinite set $\mathcal{A}$ of *atoms* ranged over by $a, b, c, \ldots$, and assume that Σ, $\mathcal{X}$, and $\mathcal{A}$ are pairwise disjoint. Unless otherwise stated, different meta-variables for objects in Σ, $\mathcal{X}$, or $\mathcal{A}$ denote different objects. A *swapping* is a pair of atoms, written $(a\ b)$. *Permutations* π are bijections on $\mathcal{A}$ such that the set of atoms for which $a \neq \pi(a)$ is finite. Permutations are represented by lists of swappings applied in the right-to-left order. For example, $((b\ c)(a\ b))(a) = c$, $((b\ c)(a\ b))(b) = a$, $((b\ c)(a\ b))(c) = b$. We write Id for the identity permutation, π^{-1} for the inverse of π, and $\pi \circ \pi'$ for the composition of π' and π.

Nominal terms, or simply *terms*, are generated by the grammar

$$t, s ::= a \mid \pi{\cdot}X \mid [a]t \mid f\ t \mid (t_1, \ldots, t_n)$$

and called, respectively, atoms, moderated variables, abstractions, function applications (which must respect the arity of the function symbol) and tuples. We abbreviate $Id{\cdot}X$ as X if there is no ambiguity. We write $f\ ()$ as simply f. An abstraction $[a]t$ is intended to represent t with a bound. The set of *free* atoms occurring in t, denoted by $FA(t)$, is defined as follows: $FA(a) = \{a\}$; $FA(\pi{\cdot}X) = \emptyset$; $FA([a]t) = FA(t) \setminus \{a\}$; $FA(f\ t) = FA(t)$; $FA((t_1, \ldots, t_n)) = \bigcup_i FA(t_i)$. We write $V(t)(\subseteq \mathcal{X})$ for the set of variables occurring in t. A *linear* term is a term in which any variable occurs at most once.

▶ **Example 1.** A nominal signature for the λ-calculus has two function symbols lam and app with arity 1 and 2, respectively. The nominal term $\mathtt{app}(\mathtt{lam}([a]\mathtt{lam}([b]\mathtt{app}(a, X))), a)$ represents the λ-term $(\lambda a.\lambda b.aX)a$ in the usual notation. Here X is a (meta-level) variable which can be instantiated by another term possibly with free atoms a and b. For this term t, we have $FA(t) = \{a\}$ and $V(t) = \{X\}$. ◀

Positions are finite sequences of positive integers. The empty sequence is denoted by ε. The set of positions in a term t, denoted by $Pos(t)$, is defined as follows: $Pos(a) = Pos(\pi{\cdot}X) = \{\varepsilon\}$; $Pos([a]t) = Pos(f\ t) = \{1p \mid p \in Pos(t)\} \cup \{\varepsilon\}$; $Pos((t_1, \ldots, t_n)) = \bigcup_i \{ip \mid p \in Pos(t_i)\} \cup \{\varepsilon\}$. The subterm of t at a position $p \in Pos(t)$ is written as $t|_p$. For each $X \in V(t)$, we define $Pos_X(t) = \{p \in Pos(t) \mid \exists \pi.\, t|_p = \pi{\cdot}X\}$, and the set of *variable positions* in t is defined by $Pos_{\mathcal{X}}(t) = \bigcup_{X \in V(t)} Pos_X(t)$. The set of *atom positions* in t is defined by $Pos_{\mathcal{A}}(t) = \{p \in Pos(t) \mid \exists a \in \mathcal{A}.\, t|_p = a\}$, and we define $Pos_{\mathcal{X}\mathcal{A}}(t) = Pos_{\mathcal{X}}(t) \cup Pos_{\mathcal{A}}(t)$.

A *context* is a term in which a distinguished function symbol $\Box$ with arity 0 occurs. The term obtained from a context $C[\,]$ by replacing each $\Box$ at positions p_i by terms t_i is written as $C[t_1, \ldots, t_n]_{p_1,\ldots,p_n}$ or simply $C[t_1, \ldots, t_n]$.

Next, we define two kinds of permutation actions, which operate on terms extending a permutation on atoms. These actions are used to define substitution, α-equivalence and rewrite relation for nominal rewriting systems. The first permutation action, written $\pi{\cdot}t$, is defined inductively by: $\pi{\cdot}a = \pi(a)$; $\pi{\cdot}(\pi'{\cdot}X) = (\pi \circ \pi'){\cdot}X$; $\pi{\cdot}(t_1, \ldots, t_n) = (\pi{\cdot}t_1, \ldots, \pi{\cdot}t_n)$; $\pi{\cdot}([a]t) = [\pi{\cdot}a](\pi{\cdot}t)$; $\pi{\cdot}(f\ t) = f\ \pi{\cdot}t$. The second permutation action, written t^π, is defined by: $a^\pi = \pi(a)$; $(\pi'{\cdot}X)^\pi = (\pi \circ \pi' \circ \pi^{-1}){\cdot}X$; $(t_1, \ldots, t_n)^\pi = (t_1^\pi, \ldots, t_n^\pi)$; $([a]t)^\pi = [a^\pi](t^\pi)$; $(f\ t)^\pi = f\ t^\pi$. The difference consists in the clause for moderated variables. In particular, when $\pi' = Id$, π is suspended before X in the first action as $\pi{\cdot}(Id{\cdot}X) = (\pi \circ Id){\cdot}X = \pi{\cdot}X$, while in the second action π has no effect as $(Id{\cdot}X)^\pi = (\pi \circ Id \circ \pi^{-1}){\cdot}X = Id{\cdot}X$.

A *substitution* is a map σ from variables to terms such that the set $\{X \in \mathcal{X} \mid \sigma(X) \neq X\}$ is finite. Substitutions act on variables, without avoiding capture of atoms. We write $t\sigma$ for

$$\frac{}{\nabla \vdash a\#b} \qquad \frac{\nabla \vdash a\#t}{\nabla \vdash a\#f\ t} \qquad \frac{\nabla \vdash a\#t_1 \quad \cdots \quad \nabla \vdash a\#t_n}{\nabla \vdash a\#(t_1,\ldots,t_n)}$$

$$\frac{}{\nabla \vdash a\#[a]t} \qquad \frac{\nabla \vdash a\#t}{\nabla \vdash a\#[b]t} \qquad \frac{\pi^{-1}\cdot a\#X \in \nabla}{\nabla \vdash a\#\pi\cdot X}$$

Figure 1 Rules for freshness constraints.

$$\frac{}{\nabla \vdash a \approx_\alpha a} \qquad \frac{\nabla \vdash t \approx_\alpha s}{\nabla \vdash f\ t \approx_\alpha f\ s} \qquad \frac{\nabla \vdash t_1 \approx_\alpha s_1 \quad \cdots \quad \nabla \vdash t_n \approx_\alpha s_n}{\nabla \vdash (t_1,\ldots,t_n) \approx_\alpha (s_1,\ldots,s_n)}$$

$$\frac{\nabla \vdash t \approx_\alpha s}{\nabla \vdash [a]t \approx_\alpha [a]s} \qquad \frac{\nabla \vdash (a\ b)\cdot t \approx_\alpha s \quad \nabla \vdash b\#t}{\nabla \vdash [a]t \approx_\alpha [b]s} \qquad \frac{\forall a \in ds(\pi,\pi').\ a\#X \in \nabla}{\nabla \vdash \pi\cdot X \approx_\alpha \pi'\cdot X}$$

Figure 2 Rules for α-equivalence.

the application of σ on t. Note here that by replacing X of a moderated variable $\pi\cdot X$ in t by $\sigma(X)$, a permutation action $\pi\cdot(\sigma(X))$ occurs. For a permutation π and a substitution σ, we define the substitution $\pi\cdot\sigma$ by $(\pi\cdot\sigma)(X) = \pi\cdot(\sigma(X))$.

2.2 α-equivalence and nominal rewriting systems

The distinctive feature of nominal rewriting is that it is equipped with a mechanism to avoid accidental capture of free atoms on the way of rewriting. This is partly achieved by α-conversion built in the matching process of the LHS of a rule and a redex involving also permutations (cf. Example 10).

In this subsection, we first recall the notion of α-equivalence in the nominal setting. This is different from α-equivalence in the traditional sense in that equivalence between terms is discussed under assumptions on the freshness of atoms in variables.

A pair $a\#t$ of an atom a and a term t is called a *freshness constraint*. Intuitively, this means that a does not occur as a free atom in t, including the cases where the variables in t are instantiated by other terms. A finite set $\nabla \subseteq \{a\#X \mid a \in \mathcal{A}, X \in \mathcal{X}\}$ is called a *freshness context*. For a freshness context ∇, we define $V(\nabla) = \{X \in \mathcal{X} \mid \exists a.\ a\#X \in \nabla\}$, $\nabla^\pi = \{a^\pi\#X \mid a\#X \in \nabla\}$, and $\nabla\sigma = \{a\#\sigma(X) \mid a\#X \in \nabla\}$.

The rules in Figure 1 defines the relation $\nabla \vdash a\#t$, which means that $a\#t$ is satisfied under the freshness context ∇. It can be seen that $a \notin FA(t)$ whenever $\nabla \vdash a\#t$. An example using the last rule is $\{c\#X\} \vdash a\#((a\ b)(b\ c))\cdot X$, since $((a\ b)(b\ c))^{-1}\cdot a = ((b\ c)(a\ b))(a) = c$.

The rules in Figure 2 defines the relation $\nabla \vdash t \approx_\alpha s$, which means that t is α-equivalent to s under the freshness context ∇. $ds(\pi,\pi')$ in the last rule denotes the set $\{a \in \mathcal{A} \mid \pi\cdot a \neq \pi'\cdot a\}$. For example, $ds((a\ b), Id) = \{a, b\}$.

▶ **Example 2.** Consider the nominal signature for the λ-calculus in Example 1, and suppose $\nabla = \{a\#X, b\#X\}$. Then we have the following derivation:

$$\dfrac{\dfrac{\dfrac{a\#X \in \nabla}{\nabla \vdash a\#X} \quad \dfrac{b\#X \in \nabla}{\nabla \vdash b\#X}}{\nabla \vdash (a\ b)\cdot X \approx_\alpha X} \quad \dfrac{b\#X \in \nabla}{\nabla \vdash b\#X}}{\nabla \vdash [a]X \approx_\alpha [b]X}}{\nabla \vdash \mathtt{lam}([a]X) \approx_\alpha \mathtt{lam}([b]X)}$$

◀

The following properties are shown in [3].

▶ **Proposition 3** ([3]).
1. $\nabla \vdash a\#t$ *if and only if* $\nabla \vdash \pi{\cdot}a\#\pi{\cdot}t$.
2. $\nabla \vdash t \approx_\alpha s$ *if and only if* $\nabla \vdash \pi{\cdot}t \approx_\alpha \pi{\cdot}s$.
3. *If* $\nabla \vdash a\#t$ *and* $\nabla \vdash t \approx_\alpha s$ *then* $\nabla \vdash a\#s$.
4. $\forall a \in ds(\pi, \pi').\, \nabla \vdash a\#t$ *if and only if* $\nabla \vdash \pi{\cdot}t \approx_\alpha \pi'{\cdot}t$.

▶ **Proposition 4** ([3]). *For any freshness context* ∇*, the binary relation* $\nabla \vdash - \approx_\alpha -$ *is a congruence (i.e. an equivalence relation that is closed under any context* $C[\,]$*).*

For terms with no variables, this relation coincides with usual α-equivalence (i.e. the relation reached by renamings of bound atoms) [6].

Now we define nominal rewrite rules and nominal rewriting systems.

▶ **Definition 5** (Nominal rewrite rule). A *nominal rewrite rule*, or simply *rewrite rule*, is a triple of a freshness context ∇ and terms l and r such that $V(\nabla) \cup V(r) \subseteq V(l)$. We write $\nabla \vdash l \to r$ for a rewrite rule. A rewrite rule $\nabla \vdash l \to r$ is *left-linear* if l is linear. We define $V(\nabla \vdash l \to r) = V(\nabla) \cup V(l) \cup V(r)$ and $(\nabla \vdash l \to r)^\pi = \nabla^\pi \vdash l^\pi \to r^\pi$.

▶ **Example 6.** Using the nominal signature for the λ-calculus in Example 1, the η-rule can be represented by the following rewrite rule (we omit the braces on the LHS of $\vdash$):

$$a\#X \quad \vdash \quad \mathtt{lam}([a]\mathtt{app}(X, a)) \quad \to \quad X \qquad (\mathsf{Eta})$$

This rule is left-linear. ◀

▶ **Definition 7** (Nominal rewriting system). A *nominal rewriting system*, or simply *rewriting system*, is a finite set of rewrite rules. A rewriting system is *left-linear* if so are all its rewrite rules.

▶ **Example 8.** We extend the signature in Example 1 by a function symbol $\mathtt{sub}$ with arity 2. By $\mathtt{sub}([a]t, s)$, we represent an explicit substitution $t\langle a := s\rangle$. Then, a nominal rewriting system to perform β-reduction is defined by the rule (Beta):

$$\vdash \quad \mathtt{app}(\mathtt{lam}([a]X), Y) \quad \to \quad \mathtt{sub}([a]X, Y) \qquad (\mathsf{Beta})$$

together with a rewriting system $\mathcal{R}_\sigma$ to execute substitution:

$$\mathcal{R}_\sigma = \left\{ \begin{array}{rcrcll} & \vdash & \mathtt{sub}([a]\mathtt{app}(X, Y), Z) & \to & \mathtt{app}(\mathtt{sub}([a]X, Z), \mathtt{sub}([a]Y, Z)) & (\sigma_{\mathtt{app}}) \\ & \vdash & \mathtt{sub}([a]a, X) & \to & X & (\sigma_{\mathtt{var}}) \\ & \vdash & \mathtt{sub}([a]b, X) & \to & b & (\sigma_{\mathtt{var}\epsilon}) \\ b\#Y & \vdash & \mathtt{sub}([a]\mathtt{lam}([b]X), Y) & \to & \mathtt{lam}([b]\mathtt{sub}([a]X, Y)) & (\sigma_{\mathtt{lam}}) \end{array} \right.$$

In a standard notation, the system $\mathcal{R}_\sigma$ is represented as follows:

$$\mathcal{R}_\sigma = \left\{ \begin{array}{rcrcll} & \vdash & (XY)\langle a := Z\rangle & \to & (X\langle a := Z\rangle)(Y\langle a := Z\rangle) & (\sigma_{\mathtt{app}}) \\ & \vdash & a\langle a := X\rangle & \to & X & (\sigma_{\mathtt{var}}) \\ & \vdash & b\langle a := X\rangle & \to & b & (\sigma_{\mathtt{var}\epsilon}) \\ b\#Y & \vdash & (\lambda b.X)\langle a := Y\rangle & \to & \lambda b.(X\langle a := Y\rangle) & (\sigma_{\mathtt{lam}}) \end{array} \right.$$

◀

In [5][3], nominal rewrite systems are defined as infinite sets of rewrite rules that are closed under equivariance, i.e., if R is a rule of a rewrite system $\mathcal{R}$ then so is R^π for any permutation π. In the present paper, we define rewriting systems as finite sets of rewrite rules that may not be closed under equivariance. Accordingly, our rewrite relation is defined with a permutation as a parameter unlike in the definition of rewrite relation in [5][3]. In the following, $\vdash$ is extended to mean to hold for every member of a set or a sequence on the RHS.

▶ **Definition 9** (Rewrite relation). Let $R = \nabla \vdash l \to r$ be a rewrite rule. For a freshness context Δ and terms s and t, the *rewrite relation* is defined by

$$\Delta \vdash s \to_{\langle R,\pi,p,\sigma\rangle} t \quad \overset{\text{def}}{\iff} \quad \Delta \vdash \nabla^\pi\sigma,\ s = C[s']_p,\ \Delta \vdash s' \approx_\alpha l^\pi\sigma,\ t = C[r^\pi\sigma]_p$$

where $V(l) \cap (V(\Delta) \cup V(s)) = \emptyset$. We write $\Delta \vdash s \to_{\langle R,\pi\rangle} t$ if there exist p and σ such that $\Delta \vdash s \to_{\langle R,\pi,p,\sigma\rangle} t$. We write $\Delta \vdash s \to_R t$ if there exists π such that $\Delta \vdash s \to_{\langle R,\pi\rangle} t$. For a rewriting system $\mathcal{R}$, we write $\Delta \vdash s \to_{\mathcal{R}} t$ if there exists $R \in \mathcal{R}$ such that $\Delta \vdash s \to_R t$.

▶ **Example 10.** Using the rule (Beta) in Example 8, we see that the term representing $(\lambda a.\lambda b.ba)b$ rewrites to $(\lambda b.ba)\langle a := b\rangle$, that is, we have

$$\vdash \mathtt{app}(\mathtt{lam}([a]\mathtt{lam}([b]\mathtt{app}(b,a))),b) \to_{\langle \mathsf{Beta},Id,\varepsilon,\sigma\rangle} \mathtt{sub}([a]\mathtt{lam}([b]\mathtt{app}(b,a)),b)$$

where σ is the substitution $[X := \mathtt{lam}([b]\mathtt{app}(b,a)), Y := b]$. The resulting term rewrites further to a normal form $\mathtt{lam}([c]\mathtt{app}(c,b))$ in four steps with rules of the system $\mathcal{R}_\sigma$. Here we give a detail of the first step with rule $(\sigma_{\mathtt{lam}})$ to see how capture of a free atom is avoided.

Let $s = \mathtt{sub}([a]\mathtt{lam}([b]\mathtt{app}(b,a)),b)$. Since the rule has a freshness context $\nabla = \{b\#Y\}$, to apply $(\sigma_{\mathtt{lam}})$ to s at the position $p = \varepsilon$, it is necessary to find a permutation π and a substitution σ that satisfy $\vdash \nabla^\pi\sigma$ and $\vdash s \approx_\alpha (\mathtt{sub}([a]\mathtt{lam}([b]X),Y))^\pi\sigma$. Here one cannot simply take $\pi = Id$, because then $\sigma(Y) = b$ from the condition for $\approx_\alpha$, which contradicts $\vdash \nabla^\pi\sigma$. So we take, e.g. $\pi = (b\ c)$ and $\sigma = [X := \mathtt{app}(c,a), Y := b]$ to satisfy the conditions, and get $(\mathtt{lam}([b]\mathtt{sub}([a]X,Y)))^\pi\sigma = \mathtt{lam}([c]\mathtt{sub}([a]\mathtt{app}(c,a),b))$ as the result of rewriting. ◀

In the following, a binary relation $\Delta \vdash - \bowtie -$ ($\bowtie$ is $\to_R$, $\approx_\alpha$, etc.) with a fixed freshness context Δ is called the relation $\bowtie$ under Δ, or simply the relation $\bowtie$ if there is no ambiguity. If a relation $\bowtie$ is written using $\to$ then the inverse is written using $\leftarrow$. Also, we write $\bowtie^=$ for the reflexive closure, and $\bowtie^*$ for the reflexive transitive closure. We use $\circ$ for the composition of relations.

▶ Remark. In [5, page 113][3, page 946], the rewrite relation, which we denote by $\Delta \vdash s \xrightarrow[FGM]{}_R t$, is defined in the following way. For a given rewrite rule $R = \nabla \vdash l \to r$, $\Delta \vdash s \xrightarrow[FGM]{}_R t$ holds if

1. $V(R) \cap (V(\Delta) \cup V(s)) = \emptyset$.
2. $s = C[s']$ for some context $C[\,]$ and term s', such that $\Delta \vdash \nabla\sigma,\ \Delta \vdash s' \approx_\alpha l\sigma$ for some σ.
3. $\Delta \vdash t \approx_\alpha C[r\sigma]$.

Hence, $\xrightarrow[FGM]{}_R$ differs from our $\to_R$ in the following two points. First, the rules of a rewrite system in [5][3] are closed under equivariance, so that the rewrite relation is defined without a permutation as a parameter. Secondly, α-equivalent terms are allowed on the result of rewriting. Consequently, under the same freshness context, we have $\xrightarrow[FGM]{}_R = \to_{\langle R,Id\rangle} \circ \approx_\alpha$.

3 Confluence of nominal rewriting systems

Having defined basic notions on nominal terms and nominal rewriting systems, we now set out to investigate confluence properties on the rewrite relations of nominal rewriting systems.

To be exact, we study confluence properties modulo the equivalence relation $\approx_\alpha$ in terms of abstract reduction systems [10].

▶ **Definition 11.** Let $\mathcal{R}$ be a nominal rewriting system.

1. $\to_\mathcal{R}$ is *confluent modulo* $\approx_\alpha$ if $\Delta \vdash s\ (\leftarrow_\mathcal{R}^* \circ \to_\mathcal{R}^*)\ t$ implies $\Delta \vdash s\ (\to_\mathcal{R}^* \circ \approx_\alpha \circ \leftarrow_\mathcal{R}^*)\ t$.
2. $\to_\mathcal{R}$ is *Church-Rosser modulo* $\approx_\alpha$ if $\Delta \vdash s\ (\leftarrow_\mathcal{R} \cup \to_\mathcal{R} \cup \approx_\alpha)^*\ t$ implies $\Delta \vdash s\ (\to_\mathcal{R}^* \circ \approx_\alpha \circ \leftarrow_\mathcal{R}^*)\ t$.
3. $\to_\mathcal{R}$ is *strongly compatible with* $\approx_\alpha$ if $\Delta \vdash s\ (\approx_\alpha \circ \to_\mathcal{R})\ t$ implies $\Delta \vdash s\ (\to_\mathcal{R}^= \circ \approx_\alpha)\ t$.

It is known that Church-Rosser modulo an equivalence relation $\sim$ is a stronger property than confluence modulo $\sim$ [10]. So we aim to prove Church-Rosser modulo $\approx_\alpha$ for some class of nominal rewriting systems. The strong compatibility with $\approx_\alpha$ also plays an important role in proving Church-Rosser modulo $\approx_\alpha$ through results in [10].

3.1 Self-rooted and proper overlaps

In the study of confluence, the notion of overlaps is important because they are useful for analysing how peaks $\Delta \vdash s \to_\mathcal{R} t$ and $\Delta \vdash s \to_\mathcal{R} t'$ occur. In this subsection, we introduce two kinds of overlaps and give some examples.

First, we define unification for nominal terms. Let P be a set of equations and freshness constraints $\{s_1 \approx t_1, \ldots, s_m \approx t_m, a_1\#u_1, \ldots, a_n\#u_n\}$ (where a_i and a_j may denote the same atom). Then, P is *unifiable* if there exist a freshness context Γ and a substitution θ such that $\Gamma \vdash s_1\theta \approx_\alpha t_1\theta, \ldots, s_m\theta \approx_\alpha t_m\theta, a_1\#u_1\theta, \ldots, a_n\#u_n\theta$; the pair $\langle \Gamma, \theta \rangle$ is called a *unifier* of P. It is known that the unification problem for nominal terms is decidable [17].

▶ **Example 12.** Consider the nominal signature for the λ-calculus in Example 1, and let $P = \{\mathtt{lam}([a]\mathtt{app}(X, a)) \approx \mathtt{lam}([a]Y), a\#X\}$. Then, $\langle \{a\#X\}, [Y := \mathtt{app}(X, a)] \rangle$ is a unifier of P. ◀

▶ **Definition 13** (Overlap)**.** Let $R_i = \nabla_i \vdash l_i \to r_i$ $(i = 1, 2)$ be rewrite rules. We assume without loss of generality that $V(R_1) \cap V(R_2) = \emptyset$. If $\nabla_1 \cup \nabla_2^{\pi_2} \cup \{l_1 \approx l_2^{\pi_2}|_p\}$ is unifiable for some permutation π_2 and a non-variable position p, then we say that R_1 *overlaps* on R_2, and the situation is called an *overlap* of R_1 on R_2. If R_1 and R_2 are identical modulo renaming of variables and $p = \varepsilon$, then the overlap is said to be *self-rooted*. An overlap that is not self-rooted is said to be *proper*.

▶ **Example 14.** Let R_1 and R_2 be the rules (Eta) $a\#X \vdash \mathtt{lam}([a]\mathtt{app}(X, a)) \to X$ and (Beta) $\vdash \mathtt{app}(\mathtt{lam}([a]Y), Z) \to \mathtt{sub}([a]Y, Z)$ from Examples 6 and 8, respectively. Then, R_1 overlaps on R_2, since $\{a\#X\} \cup \{\mathtt{lam}([a]\mathtt{app}(X, a)) \approx (\mathtt{app}(\mathtt{lam}([a]Y), Z))^{Id}|_{11} (= \mathtt{lam}([a]Y))\}$ is unifiable as seen in Example 12. This overlap is proper. ◀

▶ **Example 15.** There exist self-rooted overlaps of the rule (Beta) on its renamed variant, since $\{\mathtt{app}(\mathtt{lam}([a]Y), Z) \approx (\mathtt{app}(\mathtt{lam}([a]X), W))^\pi\}$ is unifiable for any permutation π. In the case of $\pi(a) = b$, we take $\langle \{a\#X, b\#X\}, [Y := X, Z := W] \rangle$ as a unifier (cf. Example 2). ◀

In first-order term rewriting, self-rooted overlaps do not matter, and only proper overlaps need to be analysed. However, in the case of nominal rewriting, that is not enough as seen in the next subsection.

3.2 Problems on confluence of nominal rewriting systems

In this subsection, we discuss problems on confluence in nominal rewriting that are not present in first-order term rewriting.

A standard confluence criterion in rewriting theory is the one by orthogonality.

▶ **Definition 16** (Orthogonality)**.** A nominal rewriting system $\mathcal{R}$ is *orthogonal* if it is left-linear and for any rules $R_1, R_2 \in \mathcal{R}$, there exists no proper overlap of R_1 on R_2.

This definition of orthogonality is different from the one in [3] (cf. Remark at the end of this subsection).

Unlike in first-order term rewriting, orthogonality is not enough to guarantee confluence of a nominal rewriting system.

▶ **Example 17.** Consider the nominal rewriting system $\mathcal{R}_{\text{uc-}\eta}$ with the only rewrite rule:

$$\vdash \quad \mathtt{lam}([a]\mathtt{app}(X,a)) \quad \to \quad X \qquad (\text{Uncond-eta})$$

The system $\mathcal{R}_{\text{uc-}\eta}$ is orthogonal, but is not Church-Rosser (even confluent) modulo $\approx_\alpha$, since $\vdash \mathtt{lam}([a]\mathtt{app}(a,a)) \to_{\langle \text{Uncond-eta}, Id\rangle} a$ and $\vdash \mathtt{lam}([a]\mathtt{app}(a,a)) \to_{\langle \text{Uncond-eta}, (a\ b)\rangle} b$. The latter follows from $\vdash \mathtt{lam}([a]\mathtt{app}(a,a)) \approx_\alpha \mathtt{lam}([b]\mathtt{app}(b,b)) = (\mathtt{lam}([a]\mathtt{app}(X,a)))^{(a\ b)}[X := b]$ (the third condition of rewrite relation in Definition 9). ◀

The above kind of rules can be excluded by the uniformity condition introduced in [3]. Intuitively, uniformity means that if an atom a is not free in s and s rewrites to t then a is not free in t. Here we employ the following definition of uniformity which is equivalent to the one in [3].

▶ **Definition 18** (Uniformity)**.** A rewrite rule $\nabla \vdash l \to r$ is *uniform* if for any atom a and any freshness context Δ, $\Delta \vdash \nabla$ and $\Delta \vdash a\#l$ imply $\Delta \vdash a\#r$. A rewriting system is *uniform* if so are all its rewrite rules.

The rule (Uncond-eta) in Example 17 is not uniform, since $\vdash a\#\mathtt{lam}([a]\mathtt{app}(X,a))$ but not $\vdash a\#X$. Uniform rewriting systems have many good properties, which we use in the proof of confluence in the next section.

Our definition of orthogonality together with uniformity does not guarantee confluence of a nominal rewriting system, as seen in the next example.

▶ **Example 19.** We extend the signature in Example 1 by a function symbol `uc-eta-exp` with arity 1. Consider the nominal rewriting system $\mathcal{R}_{\text{uc-}\eta\text{-exp}}$ with the only rewrite rule:

$$\vdash \quad \mathtt{uc\text{-}eta\text{-}exp}(X) \quad \to \quad \mathtt{lam}([a]\mathtt{app}(X,a)) \qquad (\text{Uncond-eta-exp})$$

The system $\mathcal{R}_{\text{uc-}\eta\text{-exp}}$ is orthogonal and uniform. Uniformity follows from the observation that for any atom a' and any freshness context Δ, if $\Delta \vdash a'\#\mathtt{uc\text{-}eta\text{-}exp}(X)$, which is equivalent to $\Delta \vdash a'\#X$, then $\Delta \vdash a'\#\mathtt{lam}([a]\mathtt{app}(X,a))$. We see, however, that $\mathcal{R}_{\text{uc-}\eta\text{-exp}}$ is not Church-Rosser (even confluent) modulo $\approx_\alpha$, since $\vdash \mathtt{uc\text{-}eta\text{-}exp}(a) \to_{\langle \text{Uncond-eta-exp}, Id\rangle} \mathtt{lam}([a]\mathtt{app}(a,a))$ and $\vdash \mathtt{uc\text{-}eta\text{-}exp}(a) \to_{\langle \text{Uncond-eta-exp}, (a\ b)\rangle} \mathtt{lam}([b]\mathtt{app}(a,b))$, where the resulting two terms are normal forms in $\mathcal{R}_{\text{uc-}\eta\text{-exp}}$ and not α-equivalent. ◀

So orthogonality together with uniformity is still not enough to guarantee confluence of a nominal rewriting system. In the next section, we consider another condition that excludes rewrite rules like the rule (Uncond-eta-exp) in Example 19.

▶ Remark. It is claimed in [3] that terminating uniform nominal rewrite systems are confluent if all non-trivial[1] critical pairs are joinable, and that orthogonal uniform nominal rewrite systems are confluent. However, the latter criterion is not applicable to standard nominal rewrite systems: In [3], a critical pair is said to be trivial if it is obtained from the root overlap of the same (renamed) rewrite rule or obtained from a variable overlap[2], where overlaps are considered without permutations unlike in our Definition 13, and a nominal rewrite system is said to be orthogonal if it is left-linear and has no non-trivial critical pair. Then, for example, a critical pair obtained from the root overlap of the two rules (Beta) and $(\mathsf{Beta})^\pi$ (cf. Example 15) is non-trivial, and so any rewrite system with the rule (Beta), which also has $(\mathsf{Beta})^\pi$ by equivariance, is not orthogonal in the sense of [3]. The same can be said of many other rewrite rules and systems.

3.3 Confluence proof of orthogonal nominal rewriting systems

In the previous subsection, we discussed problems on confluence that are peculiar to nominal rewriting. In this subsection, we prove confluence (Church-Rosser modulo $\approx_\alpha$) for a class of nominal rewriting systems with one more condition besides uniformity and orthogonality. The proof can be considered as an adaptation of inductive confluence proofs of first-order orthogonal term rewriting systems found, e.g. in [16, Section 4.7] and [7]. We omit some of (the details of) the proofs, which are available at [15].

First, we define a notion of parallel reduction using a particular kind of contexts.

▶ **Definition 20.** The *grammatical contexts*, ranged over by $G[\,]$, are the contexts defined by

$$G[\,] ::= a \mid \pi{\cdot}X \mid [a]\Box \mid f\ \Box \mid (\Box_1, \ldots, \Box_n)$$

Let $\mathcal{R}$ be a nominal rewriting system. For a given freshness context Δ, we define the relation $\Delta \vdash - \mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}} -$ inductively by the following rules:

$$\frac{\Delta \vdash s_1 \mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}} t_1 \quad \cdots \quad \Delta \vdash s_n \mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}} t_n}{\Delta \vdash G[s_1, \ldots, s_n] \mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}} G[t_1, \ldots, t_n]}\ (\mathsf{context}) \qquad \frac{\Delta \vdash s \to_{\langle R,\pi,\varepsilon,\sigma\rangle} t \quad R \in \mathcal{R}}{\Delta \vdash s \mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}} t}\ (\mathsf{head})$$

where $n\,(\geq 0)$ depends on the form of $G[\,]$. We define $\Delta \vdash \sigma \mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}} \delta$ by $\forall X.\, \Delta \vdash X\sigma \mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}} X\delta$.

▶ **Lemma 21.** *1.* $\Delta \vdash s \mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}} s$.
2. If $\Delta \vdash s \mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}} t$ *then* $\Delta \vdash C[s] \mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}} C[t]$.
3. If $\Delta \vdash s \to_{\langle R,\pi,p,\sigma\rangle} t$ *then* $\Delta \vdash s \mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}} t$.
4. If $\Delta \vdash s \mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}} t$ *then* $\Delta \vdash s \to_{\mathcal{R}}^* t$.

Instead of showing the diamond property of $\mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}}$ as in usual confluence proofs, we prove strong local confluence modulo $\approx_\alpha$ (Lemma 27), which together with strong compatibility with $\approx_\alpha$ (Lemma 22) yields Church-Rosser modulo $\approx_\alpha$ of $\mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}}$ (and hence of $\to_{\mathcal{R}}$).

▶ **Lemma 22** (Strong compatibility with $\approx_\alpha$)**.** *Let* $\mathcal{R}$ *be a uniform nominal rewriting system. If* $\Delta \vdash s' \approx_\alpha s \mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}} t$ *then there exists* t' *such that* $\Delta \vdash s' \mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}} t' \approx_\alpha t$.

Proof. By induction on the derivation of $\Delta \vdash s \mathrel{-\!\!\!\twoheadrightarrow_{\mathcal{R}}} t$. For the details, see [15]. ◀

[1] As mentioned soon, the definition of non-trivial here is different from the standard one.
[2] This definition is not very standard, but it is not the point here.

The key lemma to strong local confluence modulo $\approx_\alpha$ is Lemma 25, which corresponds to Lemma 4.7.7 of [16, page 122] in the first-order case. To show it, we first prove the following two technical lemmas.

▶ **Lemma 23.** *Let $\mathcal{R}$ be a nominal rewriting system, and let $R = \nabla \vdash l \to r \in \mathcal{R}$ and $\hat{R} = \hat{\nabla} \vdash \hat{l} \to \hat{r} \in \mathcal{R}$ (we assume $V(R) \cap V(\hat{R}) = \emptyset$). Suppose that l' is a proper subterm of l^π for some π where l' is not a moderated variable, and that there exist $\sigma, \hat{\pi}, \hat{\sigma}, \Delta, s$ that satisfy $\Delta \vdash \nabla^\pi\sigma$, $\Delta \vdash s \approx_\alpha l'\sigma$, $\Delta \vdash \hat{\nabla}^{\hat{\pi}}\hat{\sigma}$ and $\Delta \vdash s \approx_\alpha \hat{l}^{\hat{\pi}}\hat{\sigma}$. Then $\mathcal{R}$ is not orthogonal.*

Proof. Let p be the position of the subterm l' of l^π, i.e., $l^\pi|_p = l'$. Since $\hat{\nabla} \cup \nabla^{\hat{\pi}^{-1}\circ\pi} \cup \{\hat{l} \approx l^{\hat{\pi}^{-1}\circ\pi}|_p (= l'^{\hat{\pi}^{-1}})\}$ is unifiable with a unifier $\langle \Delta, \hat{\pi}^{-1}\cdot(\hat{\sigma} \cup \sigma)\rangle$, $\mathcal{R}$ is not orthogonal. ◀

▶ **Lemma 24.** *Let $\mathcal{R}$ be a uniform rewriting system. Then, if $\Delta \vdash \nabla\sigma$, $\Delta \vdash s \approx_\alpha \pi\cdot X\sigma$ and $\Delta \vdash s \mathrel{-\!\!\!\!\rightarrow\!\!\!\!\!\rightarrow}_{\mathcal{R}} t$ then there exists δ such that $\Delta \vdash \nabla\delta$, $\Delta \vdash t \approx_\alpha \pi\cdot X\delta$, $\Delta \vdash \sigma \mathrel{-\!\!\!\!\rightarrow\!\!\!\!\!\rightarrow}_{\mathcal{R}} \delta$ and for any $Y \neq X$, $Y\sigma = Y\delta$.*

Now we prove the announced lemma.

▶ **Lemma 25.** *Let $\mathcal{R}$ be an orthogonal uniform rewriting system, and let $\nabla \vdash l \to r \in \mathcal{R}$. Suppose that l' is a proper subterm of l^π for some π. Then, if $\Delta \vdash \nabla^\pi\sigma$, $\Delta \vdash s \approx_\alpha l'\sigma$ and $\Delta \vdash s \mathrel{-\!\!\!\!\rightarrow\!\!\!\!\!\rightarrow}_{\mathcal{R}} t$ then there exists δ such that $\Delta \vdash \nabla^\pi\delta$, $\Delta \vdash t \approx_\alpha l'\delta$, $\Delta \vdash \sigma \mathrel{-\!\!\!\!\rightarrow\!\!\!\!\!\rightarrow}_{\mathcal{R}} \delta$ and for any $X \notin V(l')$, $X\sigma = X\delta$.*

Proof. By induction on l'. The case where l' is a moderated variable $\pi'\cdot X$ follows from Lemma 24. For the other cases, we first show that the last rule used in the derivation of $\Delta \vdash s \mathrel{-\!\!\!\!\rightarrow\!\!\!\!\!\rightarrow}_{\mathcal{R}} t$ can not be (head). Suppose otherwise. Then by the definition of rewrite relation, we have $\Delta \vdash \hat{\nabla}^{\hat{\pi}}\hat{\sigma}$ and $\Delta \vdash s \approx_\alpha \hat{l}^{\hat{\pi}}\hat{\sigma}$ for some $\hat{\pi}$, $\hat{\sigma}$ and $\hat{\nabla} \vdash \hat{l} \to \hat{r} \in \mathcal{R}$. However, by Lemma 23, this contradicts the orthogonality of $\mathcal{R}$. Hence, the last rule used in the derivation of $\Delta \vdash s \mathrel{-\!\!\!\!\rightarrow\!\!\!\!\!\rightarrow}_{\mathcal{R}} t$ is (context). The rest of the proof is by case analysis according to the form of l'. For the details, see [15]. ◀

Now we introduce a notion of α-stability for proving Lemma 27. This notion as well as uniformity may be introduced independently from the study of confluence as a notion that yields well-behaved rewriting. Here we consider a version in which the redex position is ε.

▶ **Definition 26** (α-stability)**.** A rewrite rule $R = \nabla \vdash l \to r$ is *α-stable* if $\Delta \vdash s \approx_\alpha s'$, $\Delta \vdash s \to_{\langle R,\pi,\varepsilon,\sigma\rangle} t$ and $\Delta \vdash s' \to_{\langle R,\pi',\varepsilon,\sigma'\rangle} t'$ imply $\Delta \vdash t \approx_\alpha t'$. A rewriting system $\mathcal{R}$ is *α-stable* if so is every rewrite rule $R \in \mathcal{R}$.

The rule (Uncond-eta-exp) in Example 19 is not α-stable, since, as we saw,
$\vdash \mathtt{uc\text{-}eta\text{-}exp}(a) \to_{\langle \text{Uncond-eta-exp}, Id, \varepsilon, [] \rangle} \mathtt{lam}([a]\mathtt{app}(a,a))$ and
$\vdash \mathtt{uc\text{-}eta\text{-}exp}(a) \to_{\langle \text{Uncond-eta-exp}, (a\ b), \varepsilon, [] \rangle} \mathtt{lam}([b]\mathtt{app}(a,b))$, but not $\vdash \mathtt{lam}([a]\mathtt{app}(a,a)) \approx_\alpha \mathtt{lam}([b]\mathtt{app}(a,b))$. In the next section, we give a sufficient criterion for α-stability.

Now we show that $\mathrel{-\!\!\!\!\rightarrow\!\!\!\!\!\rightarrow}_{\mathcal{R}}$ is strongly locally confluent modulo $\approx_\alpha$ for a class of orthogonal nominal rewriting systems.

▶ **Lemma 27** (Strong local confluence modulo $\approx_\alpha$)**.** *Let $\mathcal{R}$ be an orthogonal rewriting system that is uniform and α-stable. If $\Delta \vdash s \mathrel{-\!\!\!\!\rightarrow\!\!\!\!\!\rightarrow}_{\mathcal{R}} t$ and $\Delta \vdash s \mathrel{-\!\!\!\!\rightarrow\!\!\!\!\!\rightarrow}_{\mathcal{R}} t'$ then there exist u and u' such that $\Delta \vdash t \mathrel{-\!\!\!\!\rightarrow\!\!\!\!\!\rightarrow}_{\mathcal{R}} u$, $\Delta \vdash t' \mathrel{-\!\!\!\!\rightarrow\!\!\!\!\!\rightarrow}_{\mathcal{R}} u'$ and $\Delta \vdash u \approx_\alpha u'$.*

Proof. By induction on s. We distinguish cases according to the last rules used in the derivations of $\Delta \vdash s \mathrel{-\!\!\!\!\rightarrow\!\!\!\!\!\rightarrow}_{\mathcal{R}} t$ and $\Delta \vdash s \mathrel{-\!\!\!\!\rightarrow\!\!\!\!\!\rightarrow}_{\mathcal{R}} t'$.

1. Both rules are (head). If they are by the same rewrite rule $R \in \mathcal{R}$, then we use the α-stability of $\mathcal{R}$. Otherwise, this case contradicts the orthogonality of $\mathcal{R}$.

2. Both rules are (context). The claim follows from the induction hypothesis.
3. One is (context) and the other is (head). It can be shown that the claim holds by using Lemma 24 or Lemma 25. For the details, see [15].

◀

We are now ready to show that $\to_{\mathcal{R}}$ is Church-Rosser modulo $\approx_\alpha$.

▶ **Theorem 28** (Church-Rosser modulo $\approx_\alpha$)**.** *Let $\mathcal{R}$ be an orthogonal nominal rewriting system that is uniform and α-stable. Then, $\to_{\mathcal{R}}$ is Church-Rosser modulo $\approx_\alpha$.*

Proof. By Lemma 22, $\twoheadrightarrow_{\mathcal{R}}$ is strongly compatible with $\approx_\alpha$, and by Lemma 27, $\twoheadrightarrow_{\mathcal{R}}$ is strongly locally confluent modulo $\approx_\alpha$. Hence by the results in [10], $\twoheadrightarrow_{\mathcal{R}}$ is Church-Rosser modulo $\approx_\alpha$. Since $\to_{\mathcal{R}} \subseteq \twoheadrightarrow_{\mathcal{R}} \subseteq \to_{\mathcal{R}}^*$ by Lemma 21, $\to_{\mathcal{R}}$ is Church-Rosser modulo $\approx_\alpha$. ◀

4 Criterion for uniformity and α-stability

In this section, we consider a sufficient criterion for uniform and α-stable nominal rewriting systems. The main reason why a rewrite rule R does not keep α-stability is that some free atom occurring in a term is bounded through a rewrite step by R. However, such irrelevant rewrite steps can be avoided by adding an appropriate constraint to the freshness context of R. We introduce the notion of abstract skeleton preserving for characterising this constraint and show it gives a sufficient criterion for uniformity and α-stability.

Throughout this section, different meta-variables for atoms may denote the same atom.

4.1 Abstract skeleton

The abstract skeleton of a nominal term is defined as a subterm abstracted with the binders occurring on the path from the root position ε to the position of the subterm.

▶ **Definition 29** (Abstract skeleton)**.** For a nominal term t and a position $p \in Pos(t)$, $skel(p,t)$ is defined as follows:

$$\begin{array}{rcl} skel(\varepsilon, s) & = & s \\ skel(1q, [a]s) & = & [a]skel(q,s) \\ skel(1q, f\ s) & = & skel(q,s) \\ skel(iq, (s_1,\dots,s_n)) & = & skel(q,s_i) \end{array}$$

$skel(p,t)$ is called an *abstract skeleton* at p of t. $skel(p,t) = [a_1]\dots[a_n]s$ is *non-duplicating* if $i \neq j$ implies $a_i \neq a_j$. We define $Skel(t) = \{skel(p,t) \mid p \in Pos(t)\}$.

▶ **Example 30.** Figure 3 shows the abstract skeletons at the leaf positions of the left and the right hand sides of the rule ($\sigma_{\mathtt{lam}}$) in Example 8. ◀

For each $X \in V(t)$, we define $Skel_X(t) = \{skel(p,t) \mid p \in Pos_X(t)\}$. We also define $Skel_{\mathcal{X}}(t) = \{skel(p,t) \mid p \in Pos_{\mathcal{X}}(t)\}$, $Skel_{\mathcal{A}}(t) = \{skel(p,t) \mid p \in Pos_{\mathcal{A}}(t)\}$, and $Skel_{\mathcal{XA}}(t) = Skel_{\mathcal{X}}(t) \cup Skel_{\mathcal{A}}(t)$.

The following lemmas are useful for discussing the freshness context and the α-equivalency of the term through the decomposed parts. (For the proofs, see [15].)

▶ **Lemma 31.** $\Delta \vdash a\#t\sigma \iff \forall u \in Skel_{\mathcal{XA}}(t).\ \Delta \vdash a\#u\sigma$

▶ **Lemma 32.** $\Delta \vdash t\sigma \approx_\alpha t^\pi\rho \iff \forall u \in Skel_{\mathcal{XA}}(t).\ \Delta \vdash u\sigma \approx_\alpha u^\pi\rho$

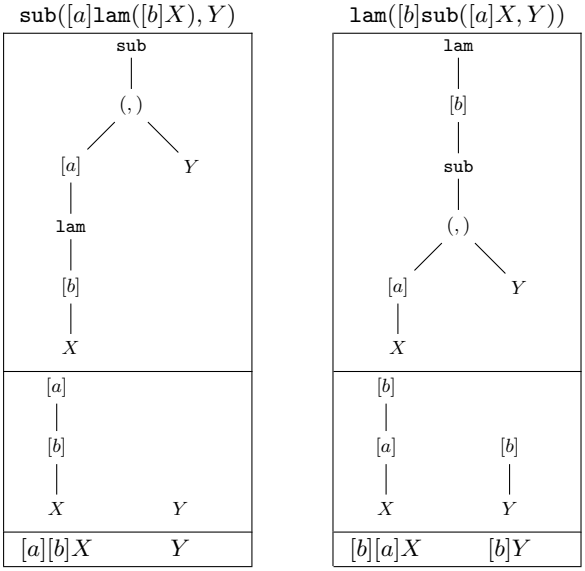

Figure 3 Abstract skeletons of $\mathtt{sub}([a]\mathtt{lam}([b]X), Y)$ and $\mathtt{lam}([b]\mathtt{sub}([a]X, Y))$.

4.2 Abstract skeleton preserving rewrite rules

In this subsection, we introduce the notions of abstract skeleton preserving rules and systems. First, we restrict rewrite rules to the following ones.

▶ **Definition 33** (Standard)**.** A nominal rewrite rule $\nabla \vdash l \to r$ is *standard* when:
(S1) For every moderated variable $\pi{\cdot}X$ appearing in l or r, $\pi = Id$,
(S2) $FA(r) \subseteq FA(l)$,
(S3) Every abstract skeleton $[a_1]\dots[a_n]t \in Skel(l) \cup Skel(r)$ is non-duplicating.
A nominal rewriting system $\mathcal{R}$ is *standard* if so is every rewrite rule $R \in \mathcal{R}$.

All examples of rewrite rules we treated so far are standard.

We now define the abstract skeleton preserving nominal rewriting systems.

▶ **Definition 34** (Abstract skeleton preseving)**.** A nominal rewrite rule $\nabla \vdash l \to r$ is *abstract skeleton preserving* (*ASP* for short) if it is standard and

$$\forall [a_1]\dots[a_m]X \in Skel_X(r).\exists [b_1]\dots[b_n]X \in Skel_X(l).\forall a \in ds(\{a_i\}_i, \{b_j\}_j).\ a\#X \in \nabla$$

where $ds(\{a_i\}_i, \{b_j\}_j)$ is the set of atoms such that $a \in \{a_1, \dots, a_m\}$ and $a \notin \{b_1, \dots, b_n\}$, or $a \notin \{a_1, \dots, a_m\}$ and $a \in \{b_1, \dots, b_n\}$. A nominal rewriting system $\mathcal{R}$ is *abstract skeleton preserving* (*ASP* for short) if so is every rewrite rule $R \in \mathcal{R}$.

It is easy to judge whether a standard rewrite rule is ASP or not. The rule (Uncond-eta) in Example 17 and the rule (Uncond-eta-exp) in Example 19 are not ASP. All the other rewrite rules we treated so far are ASP.

In the rest of this section, we show that the ASP property gives a sufficient criterion for the uniformity and the α-stability of nominal rewriting systems. First we prove the uniformity of ASP rewrite rules.

▶ **Lemma 35.** *An ASP rewrite rule is uniform.*

Proof. We show that for any ASP rewrite rule $R = \nabla \vdash l \to r$, if $\Delta \vdash \nabla$ and $\Delta \vdash a\#l$ then $\Delta \vdash a\#r$. Suppose $\Delta \vdash \nabla$ and $\Delta \vdash a\#l$. From the latter, we have $\forall u \in \mathit{Skel}_{\mathcal{XA}}(l).\ \Delta \vdash a\#u$ by Lemma 31. Similarly, the conclusion $\Delta \vdash a\#r$ is equivalent to $\forall u \in \mathit{Skel}_{\mathcal{XA}}(r).\ \Delta \vdash a\#u$. We show $\Delta \vdash a\#u$ for $u \in \mathit{Skel}_{\mathcal{A}}(r)$ and for $u \in \mathit{Skel}_{\mathcal{X}}(r)$, respectively.

1. $u \in \mathit{Skel}_{\mathcal{A}}(r)$. Then u has the form $[a_1]\dots[a_m]b$. If $a \in \{a_1, \dots, a_m\}$ or $b \neq a$, then $\Delta \vdash a\#[a_1]\dots[a_m]b$ holds. Otherwise, we have $a = b \in \mathit{FA}(r)$. Since R is standard, $a \in \mathit{FA}(l)$; contradicting $\Delta \vdash a\#l$.
2. $u \in \mathit{Skel}_{\mathcal{X}}(r)$. Then u has the form $[a_1]\dots[a_m]X$. If $a \in \{a_1, \dots, a_m\}$ then $\Delta \vdash a\#[a_1]\dots[a_m]X$ holds. Otherwise, it is enough to show $\Delta \vdash a\#X$. Since R is ASP, there exists $[b_1]\dots[b_n]X \in \mathit{Skel}_X(l)$ such that $\forall a \in ds(\{a_i\}_i, \{b_j\}_j).\ a\#X \in \nabla$. Since we have $\forall u \in \mathit{Skel}_{\mathcal{XA}}(l).\ \Delta \vdash a\#u$, it holds that $\Delta \vdash a\#[b_1]\dots[b_n]X$. Thus, if $a \notin \{b_1, \dots, b_n\}$ then $\Delta \vdash a\#X$. If $a \in \{b_1, \dots, b_n\}$, then we have $a \in ds(\{a_i\}_i, \{b_j\}_j)$ because now we discuss the case of $a \notin \{a_1, \dots, a_m\}$. Hence, $a\#X \in \nabla$ holds. Since we suppose $\Delta \vdash \nabla$, $\Delta \vdash a\#X$ is obtained.

◀

4.3 α-stability of abstract skeleton preserving rewrite rules

Next, we prove the α-stability of ASP rewrite rules. For this, we need to derive α-equivalence of respective reducts s' and t' of terms s, t, from α-equivalence of s and t. The idea is to use Lemma 32 and infer α-equivalence via abstract skeletons. Recall abstract skeletons of the rule ($\sigma_{\mathtt{lam}}$) in Example 30. Here, an abstract skeleton $[a][b]X$ in LHS changes to $[b][a]X$ in RHS. Thus, $[a][b]X \approx_\alpha [c][d]X'$ should imply $[b][a]X \approx_\alpha [d][c]X'$. But generally, this is not true; for example, we have $\vdash [a][a]a \approx_\alpha [a][b]b$ but $\not\vdash [a][a]a \approx_\alpha [b][a]b$. This can be guaranteed, however, for non-duplicating skeletons (cf. Lemma 36). Another abstract skeleton Y in LHS changes to $[b]Y$ in RHS in Example 30. Again, $\vdash Y \approx_\alpha Y'$ does not imply $\vdash [b]Y \approx_\alpha [b]Y'$ in general. Fortunately, the freshness constraint of the rule ($\sigma_{\mathtt{lam}}$) contains $b\#Y$. Thus, it suffices to guarantee $b\#Y, b\#Y' \vdash Y \approx_\alpha Y'$ implies $b\#Y, b\#Y' \vdash [b]Y \approx_\alpha [b]Y'$, which is indeed the case (cf. Lemma 37). The proofs of the following lemmas are found in [15].

▶ **Lemma 36.** *Let two terms $[a_1]\dots[a_n]s$ and $[b_1]\dots[b_n]t$ be both non-duplicating. Then,*

$$\begin{aligned} &\Delta \vdash [a_1]\dots[a_i]\dots[a_j]\dots[a_n]s \approx_\alpha [b_1]\dots[b_i]\dots[b_j]\dots[b_n]t \\ \Longrightarrow\ &\Delta \vdash [a_1]\dots[a_j]\dots[a_i]\dots[a_n]s \approx_\alpha [b_1]\dots[b_j]\dots[b_i]\dots[b_n]t \end{aligned}$$

▶ **Lemma 37.** *Let two terms $[a_1]\dots[a_n]s$ and $[b_1]\dots[b_n]t$ be both non-duplicating, and let $\Delta \vdash a_i\#s, b_i\#t$. Then,*

$$\begin{aligned} &\Delta \vdash [a_1]\dots[a_{i-1}][a_i][a_{i+1}]\dots[a_n]s \approx_\alpha [b_1]\dots[b_{i-1}][b_i][b_{i+1}]\dots[b_n]t \\ \Longleftrightarrow\ &\Delta \vdash [a_1]\dots[a_{i-1}][a_{i+1}]\dots[a_n]s \approx_\alpha [b_1]\dots[b_{i-1}][b_{i+1}]\dots[b_n]t \end{aligned}$$

▶ **Lemma 38.** *If $\Delta \vdash t\sigma \approx_\alpha t^\pi\rho$ then $\forall a \in \mathit{FA}(t).\ a = \pi{\cdot}a$.*

▶ **Theorem 39.** *ASP nominal rewriting systems are uniform and α-stable.*

Proof. We show that if $\mathcal{R}$ is ASP then every $R = \nabla \vdash l \to r \in \mathcal{R}$ is α-stable, that is,

$$\Delta \vdash s \approx_\alpha \hat{s} \wedge \Delta \vdash s \to_{\langle R,\pi,\varepsilon,\sigma\rangle} t \wedge \Delta \vdash \hat{s} \to_{\langle R,\hat{\pi},\varepsilon,\hat{\sigma}\rangle} \hat{t} \implies \Delta \vdash t \approx_\alpha \hat{t}.$$

Considering R^π as R and $\hat{\pi} \circ \pi^{-1}$ as $\hat{\pi}$, we can take $\pi = \mathit{Id}$ without loss of generality. (Note that if R is ASP then so is R^π.) Thus from here on we take Id as $\hat{\pi}$. From the definition of

the rewrite relation,

$$\Delta \vdash s \to_{\langle R,\pi,\epsilon,\sigma\rangle} t \iff \Delta \vdash \nabla^{\pi}\sigma,\ \Delta \vdash s \approx_{\alpha} l^{\pi}\sigma,\ t = r^{\pi}\sigma$$
$$\Delta \vdash \hat{s} \to_{\langle R,Id,\epsilon,\hat{\sigma}\rangle} \hat{t} \iff \Delta \vdash \nabla\hat{\sigma},\ \Delta \vdash \hat{s} \approx_{\alpha} l\hat{\sigma},\ \hat{t} = r\hat{\sigma}$$

From the assumption and the transitivity, we have $\Delta \vdash \nabla^{\pi}\sigma$, $\Delta \vdash \nabla\hat{\sigma}$ and $\Delta \vdash l\hat{\sigma} \approx_{\alpha} l^{\pi}\sigma$. Now our aim is to show $\Delta \vdash r\hat{\sigma} \approx_{\alpha} r^{\pi}\sigma$. Here, we have

$$\Delta \vdash l\hat{\sigma} \approx_{\alpha} l^{\pi}\sigma \iff \forall u \in \mathit{Skel}_{\mathcal{XA}}(l).\ \Delta \vdash u\hat{\sigma} \approx_{\alpha} u^{\pi}\sigma \text{ (from Lemma 32)} \quad (1)$$
$$\Delta \vdash r\hat{\sigma} \approx_{\alpha} r^{\pi}\sigma \iff \forall v \in \mathit{Skel}_{\mathcal{XA}}(r).\ \Delta \vdash v\hat{\sigma} \approx_{\alpha} v^{\pi}\sigma \text{ (from Lemma 32)}$$

We show $\Delta \vdash v\hat{\sigma} \approx_{\alpha} v^{\pi}\sigma$ for $v \in \mathit{Skel}_{\mathcal{A}}(r)$ and for $v \in \mathit{Skel}_{\mathcal{X}}(r)$, respectively.

1. $v \in \mathit{Skel}_{\mathcal{A}}(r)$. Then v has the form $[a_1]\dots[a_m]b$.
   - **a.** $b \in \{a_1,\dots,a_m\}$. First we show $\Delta \vdash [a_i]a_i \approx_{\alpha} [\pi\cdot a_i]\pi\cdot a_i$. It is clear when $a_i = \pi\cdot a_i$. When $a_i \neq \pi\cdot a_i$, from $\Delta \vdash \pi\cdot a_i \# a_i$ and $\Delta \vdash (a_i\ \pi\cdot a_i)\cdot a_i \approx_{\alpha} \pi\cdot a_i$ it follows. Moreover, for $a_j (\neq a_i)$, $\Delta \vdash a_j \# a_i$ and $\Delta \vdash \pi\cdot a_j \# \pi\cdot a_i$ hold. Since v and $\pi\cdot v$ are non-duplicating, applying Lemma 37 to $\Delta \vdash [a_i]a_i \approx_{\alpha} [\pi\cdot a_i]\pi\cdot a_i$ repeatedly, we obtain $\Delta \vdash [a_1]\dots[a_{i-1}][a_i][a_{i+1}]\dots[a_m]a_i \approx_{\alpha} [\pi\cdot a_1]\dots[\pi\cdot a_{i-1}][\pi\cdot a_i][\pi\cdot a_{i+1}]\dots[\pi\cdot a_m]\pi\cdot a_i$ Thus $\Delta \vdash v\hat{\sigma} \approx_{\alpha} v^{\pi}\sigma$ follows.
   - **b.** $b \notin \{a_1,\dots,a_m\}$. It is clear that $b \in \mathit{FA}(r)$. Since R is standard, $b \in \mathit{FA}(l)$. From $\Delta \vdash l\hat{\sigma} \approx_{\alpha} l^{\pi}\sigma$ and Lemma 38 it holds that $b = \pi\cdot b$. Thus, $\Delta \vdash b \approx_{\alpha} \pi\cdot b$. Moreover, $\Delta \vdash a_j \# b$ and $\Delta \vdash \pi\cdot a_j \# \pi\cdot b$ for every a_j. Since v and $\pi\cdot v$ are non-duplicating, applying Lemma 37 to $\Delta \vdash b \approx_{\alpha} \pi\cdot b$ repeatedly, we obtain $\Delta \vdash v\hat{\sigma} \approx_{\alpha} v^{\pi}\sigma$.
2. $v \in \mathit{Skel}_{\mathcal{X}}(r)$. Then v has the form $[a_1]\dots[a_m]X$. Since R is ASP, there exists $[b_1]\dots[b_n]X \in \mathit{Skel}_X(l)$ such that $\forall a \in ds(\{a_i\}_i, \{b_j\}_j).\ a\#X \in \nabla$. By (1), we have $\Delta \vdash ([b_1]\dots[b_n]X)\hat{\sigma} \approx_{\alpha} ([b_1]\dots[b_n]X)^{\pi}\sigma$, that is, $\Delta \vdash [b_1]\dots[b_n]X\hat{\sigma} \approx_{\alpha} [\pi\cdot b_1]\dots[\pi\cdot b_n]X\sigma$. Now, let $\{c_1,\dots,c_k\} = \{a_1,\dots,a_m\} \cap \{b_1,\dots,b_n\}$. Then $\forall a \in ds(\{b_j\}_j, \{c_h\}_h).a\#X \in \nabla$ and $\forall a \in ds(\{\pi\cdot b_j\}_j, \{\pi\cdot c_h\}_h).\ a\#X \in \nabla^{\pi}$. From $\Delta \vdash \nabla\hat{\sigma}$ and $\Delta \vdash \nabla^{\pi}\sigma$, we obtain $\forall a \in ds(\{b_j\}_j, \{c_h\}_h).\ \Delta \vdash a\#X\hat{\sigma}$ and $\forall a \in ds(\{\pi\cdot b_j\}_j, \{\pi\cdot c_h\}_h).\ \Delta \vdash a\#X\sigma$. Similarly, $\forall a \in ds(\{a_i\}_i, \{c_h\}_h).\ \Delta \vdash a\#X\hat{\sigma}$ and $\forall a \in ds(\{\pi\cdot a_i\}_i, \{\pi\cdot c_h\}_h).\ \Delta \vdash a\#X\sigma$. Therefore,

$$\begin{aligned}
&\Delta \vdash [b_1]\dots[b_n]X\hat{\sigma} \approx_{\alpha} [\pi\cdot b_1]\dots[\pi\cdot b_n]X\sigma \\
\iff\ &\Delta \vdash [c_1]\dots[c_k]X\hat{\sigma} \approx_{\alpha} [\pi\cdot c_1]\dots[\pi\cdot c_k]X\sigma \text{ (from Lemmas 36 and 37)} \\
\iff\ &\Delta \vdash [a_1]\dots[a_m]X\hat{\sigma} \approx_{\alpha} [\pi\cdot a_1]\dots[\pi\cdot a_m]X\sigma \text{ (from Lemmas 36 and 37)} \\
\iff\ &\Delta \vdash v\hat{\sigma} \approx_{\alpha} v^{\pi}\sigma
\end{aligned}$$

◀

By Theorems 28 and 39, we have the following corollary.

▶ **Corollary 40.** *Let $\mathcal{R}$ be an orthogonal nominal rewriting system that is ASP. Then, $\to_{\mathcal{R}}$ is Church-Rosser modulo $\approx_{\alpha}$.*

▶ **Example 41.** The rewriting system $\mathcal{R}_{\sigma}$ in Example 8 is left-linear and has no proper overlaps, and hence orthogonal. Moreover, all its rewrite rules are ASP. Hence, $\to_{\mathcal{R}_{\sigma}}$ is Church-Rosser modulo $\approx_{\alpha}$ by Corollary 40. ◀

5 Implementation and Experiments

We have implemented a confluence prover for NRSs proving that input NRSs are CR modulo $\approx_{\alpha}$, based on Corollary 40. We note that recently some confluence provers for TRSs and CTRSs are emerged (e.g. [1, 20, 14]) and the competition of confluence provers have been

Table 1 Summary of experiments.

	NRS	LL	non-PO	ASP	result	time (ms)
1	$\mathcal{R}_\sigma$	✓	✓	✓	CR	3
2	NNF of f.o.-formulas without DNE	✓	✓	✓	CR	<1
3	$\mathcal{R}_{\mathsf{uc}\text{-}\eta\text{-}\mathsf{exp}}$	✓	✓	×	Failure	<1
4	PNF of f.o.-formulas (Example 44 [3])	✓	×	✓	Failure	20
5	NNF of f.o.-formulas	✓	×	✓	Failure	<1
6	$\{\mathsf{Beta}\} \cup \mathcal{R}_\sigma$	✓	×	✓	Failure	8
7	$\{\mathsf{Beta}\} \cup \{\mathsf{Eta}\} \cup \mathcal{R}_\sigma$	✓	×	✓	Failure	3
8	β-reduction (Example 43 [3])	✓	×	✓	Failure	29
9	η-expansion (Introduction [3])	✓	✓	✓	CR	<1
10	structural substitution for $\lambda\mu$-term ([12])	✓	✓	✓	CR	17
11	fragment of ML ([3])	✓	×	✓	Failure	152
12	$\{a\#X \vdash f(X) \to [a]X\}$	✓	✓	✓	CR	<1
13	$\{\vdash f(X) \to [a]X\}$	✓	✓	×	Failure	<1
14	$\{a\#X \vdash X \to [a]X\}$ (Proof of Lemma 56 [3])	✓	✓	✓	CR	<1
15	Non-joinable trivial critical pair (Proof of Lemma 56 [3])	✓	✓	×	Failure	<1
16	PNF of f.o.-formulas with additional rules (Example 44 [3])	✓	×	✓	Failure	33
17	Substitution for λ-term (Example 43 [3])	✓	×	✓	Failure	27
18	$\{\mathsf{Eta}\}$	✓	✓	✓	CR	<1
19	$\mathcal{R}_{\mathsf{uc}\text{-}\eta}$	✓	✓	×	Failure	<1

held annually[3]. In contrast, no confluence provers for NRSs has been known previously, up to our knowledge.

In order to prove confluence of an NRS $\mathcal{R}$ based on Corollary 40, we have to show that (1) $\mathcal{R}$ is orthogonal and (2) $\mathcal{R}$ is abstract skeleton preserving (ASP). It is straightforward to check (2), as the standardness is just a syntactical restriction and $\nabla \vdash a\#X$ is easily checked for any freshness constraint ∇, $a \in \mathcal{A}$ and $X \in \mathcal{X}$. For (1), one has to check (1-a) left-linearity and that (1-b) there's no proper overlaps. The checking of (1-a) is easy. For (1-b), we have to check whether $\nabla_1 \cup \nabla_2^{\pi_2} \cup \{l_1 \approx l_2^{\pi_2}|_p\}$ is unifiable for some permutation π_2, for given $\nabla_1, \nabla_2, l_1, l_2|_p$—this problem is different from nominal unification problems as π_2 is not fixed. Fortunately, the problem can be directly reduced to a problem of *equivariant unification* [2], which has been known to be decidable. From the equivariant unification algorithm in [2], we obtain a constraint of π_2 for unifiability, if the problem is equivariantly unifiable. Our system reports concrete critical pairs generated from this constraint, if there is a proper overlap.

We have tested our confluence prover with 19 NRSs, collected from the literature, and constructed during our study. The summary of our experiments is shown in Table 1. The columns below 'NRS', 'LL', 'non-PO', 'ASP' , 'result' 'time (ms)' show the input NRS, left-linearity, non-existence of proper overlaps, ASP, the result of the confluence prover and

[3] Confluence Competition (CoCo) `http://coco.nue.riec.tohoku.ac.jp/`

execution time in millisecond, respectively. Here, PNF (NNF) denotes rules for computing prenex normal forms (resp. negation normal forms), and DNE denotes double negation elimination (`not (not X)` $\to$ `X`). The symbol '✓' denotes that the property holds, and the symbol '×' denotes that the property does not hold, which have been checked by the prover. Among 19 examples, our prover succeeded in proving confluence of 7 examples. All tests have been performed in a PC equipped with Intel Core i7-4600U processors of 2.1GHz and a memory of 8GB.

All details of the experiments are available on the webpage `http://www.nue.riec.tohoku.ac.jp/tools/experiments/rta15nrs/`.

6 Conclusion

Using our notion of rewrite relation with a permutation as a parameter, we have presented a proof of Church-Rosser modulo $\approx_\alpha$ for the class of orthogonal nominal rewriting systems that are uniform and α-stable. Moreover, we have introduced a notion of abstract skeleton preserving as a sufficient criterion for uniformity and α-stability. We have also implemented a confluence prover based on our result on Church-Rosser modulo $\approx_\alpha$ for abstract skeleton preserving rewriting systems.

As continuations of this work, we are going to study confluence of nominal rewriting systems with proper overlaps in both terminating and non-terminating cases. In such studies, it will be necessary to investigate joinability check of critical pairs with permutation variables. This is left as future work.

Acknowledgements. We would like to thank the anonymous referees for useful comments. This research was supported by JSPS KAKENHI Grant Numbers 25330004, 25280025 and 15K00003.

References

1 T. Aoto, Y. Yoshida, and Y. Toyama. Proving confluence of term rewriting systems automatically. In *Proceedings of RTA'09*, LNCS 5595, pages 93–102. Springer-Verlag, 2009.
2 J. Cheney. Equivariant unification. *Journal of Automated Reasoning*, 45:267–300, 2010.
3 M. Fernández and M. J. Gabbay. Nominal rewriting. *Information and Computation*, 205:917–965, 2007.
4 M. Fernández and M. J. Gabbay. Closed nominal rewriting and efficiently computable nominal algebra equality. In *Proceedings of LFMTP'10*, EPTCS 34, pages 37–51, 2010.
5 M. Fernández, M. J. Gabbay, and I. Mackie. Nominal rewriting systems. In *Proceedings of PPDP'04*, pages 108–119. ACM Press, 2004.
6 M. J. Gabbay and A. M. Pitts. A new approach to abstract syntax with variable binding. *Formal Aspects of Computing*, 13:341–363, 2002.
7 D. Kesner. Confluence. Course material, `http://www.pps.univ-paris-diderot.fr/~kesner/enseignement/master1/semantique/ConfluenceSP-4.pdf`.
8 J. W. Klop, V. van Oostrom, and F. van Raamsdonk. Combinatory reduction systems: introduction and survey. *Theoretical Computer Science*, 121:279–308, 1993.
9 R. Mayr and T. Nipkow. Higher-order rewrite systems and their confluence. *Theoretical Computer Science*, 192:3–29, 1998.
10 E. Ohlebusch. Church-Rosser theorems for abstract reduction modulo an equivalence relation. In *Proceedings of RTA'98*, LNCS 1379, pages 17–31. Springer-Verlag, 1998.
11 A. C. R. Oliveira and M. Ayala-Rincón. Formalizing the confluence of orthogonal rewriting systems. In *Proceedings of LSFA'12*, EPTCS 113, pages 145–152, 2012.

12 M. Parigot. $\lambda\mu$-calculus: an algorithmic interpretation of classical natural deduction. In *Proceedings of LPAR'92*, LNAI 624, pages 190–201. Springer-Verlag, 1992.

13 A. M. Pitts. Nominal logic, a first order theory of names and binding. *Information and Computation*, 186:165–193, 2003.

14 T. Sternagel and A. Middeldorp. Conditional confluence (system description). In *Proceedings of Joint RTA and TLCA'14*, LNCS 8560, pages 456–465. Springer-Verlag, 2014.

15 T. Suzuki, K. Kikuchi, T. Aoto, and Y. Toyama. Confluence of orthogonal nominal rewriting systems revisited. `http://www.nue.riec.tohoku.ac.jp/user/kentaro/cr-nominal/`.

16 Terese. *Term Rewriting Systems*. Cambridge University Press, 2003.

17 C. Urban, A. M. Pitts, and M. J. Gabbay. Nominal unification. *Theoretical Computer Science*, 323:473–497, 2004.

18 R. Vestergaard and J. Brotherston. A formalised first-order confluence proof for the λ-calculus using one-sorted variable names. In *Proceedings of RTA'01*, LNCS 2051, pages 306–321. Springer-Verlag, 2001.

19 R. Vestergaard and J. Brotherston. A formalised first-order confluence proof for the λ-calculus using one-sorted variable names. *Information and Computation*, 183:212–244, 2003.

20 H. Zankl, B. Felgenhauer, and A. Middeldorp. CSI – A confluence tool. In *Proceedings of CADE'11*, LNAI 6803, pages 499–505. Springer-Verlag, 2011.