Nominal Narrowing*

Mauricio Ayala-Rincón[1], Maribel Fernández[2], and Daniele Nantes-Sobrinho[3]

1 **Departamentos de Ciência da Computação e Matemática, Universidade de Brasília, Brasília, Brazil**
`ayala@unb.br`
2 **Department of Informatics, King's College London, London, UK**
`maribel.fernandez@kcl.ac.uk`
3 **Departamentos de Ciência da Computação e Matemática, Universidade de Brasília, Brasília, Brazil**
`dnantes@mat.unb.br`

Abstract

Nominal unification is a generalisation of first-order unification that takes α-equivalence into account. In this paper, we study nominal unification in the context of equational theories. We introduce nominal narrowing and design a general nominal E-unification procedure, which is sound and complete for a wide class of equational theories. We give examples of application.

1998 ACM Subject Classification F.4.1 [Mathematical Logic] lambda calculus and related systems, D.3.3 [Language Constructs and Features] data types and structures, frameworks

Keywords and phrases Nominal Rewriting, Nominal Unification, Matching, Narrowing, Equational Theories

Digital Object Identifier 10.4230/LIPIcs.FSCD.2016.11

1 Introduction

This is a paper about nominal unification in the context of equational theories.

Nominal techniques [16] facilitate reasoning in systems with binding operators, where α-equivalence must be taken into account. In nominal syntax [11, 29], *atoms*, which are used to represent object-level variables in the intended applications, can be abstracted: $[a]t$ denotes the abstraction of the atom a in the term t. *Variables* in nominal terms represent unknown parts of terms and behave like first-order variables, but nominal variables may be decorated with atom permutations. Permutations act on terms, swapping atoms (e.g., $(a\ b) \cdot t$ means that a and b are swapped everywhere in t).

Nominal syntax has interesting properties. Nominal unification [29], that is, unification of nominal terms modulo α-equivalence, is decidable and unitary. Efficient nominal unification algorithms are available [3, 19]. Nominal matching, a key ingredient in the definition of nominal rewriting [11], is a particular case of nominal unification that can be solved in linear time [4]. Nominal rewriting [11] can be used to reason in nominal equational theories (see [12]; a completion procedure is described in [14]).

However, to our knowledge, the concept of *nominal* E-*unification*, i.e., nominal unification in the context of an equational theory E, has not been addressed in previous works. Nominal E-unification is needed to solve equations between nominal terms where the function symbols

* This work was partially supported by CAPES PVE 146/2012 and CNPq UNIVERSAL 476952/2013-1.

satisfy properties defined by an equational theory. Nominal E-unification has applications in, e.g., functional-logical programming languages and analysis of cryptographic protocols.

The main contributions of this paper are:

- We define nominal E-unification problems, and the *nominal narrowing* relation, and study the relationship between nominal rewriting and nominal narrowing.
- We show that Hullot's results [17] (with the corrections from [1, 24]) relating first-order narrowing derivations and first-order E-unifiers can be transferred to nominal systems. Thus, we obtain a nominal E-unification procedure that is sound and complete for the class of convergent closed equational theories. We give examples to illustrate these results.
- We define *basic nominal narrowing* and provide sufficient conditions for termination of nominal narrowing derivations, which can be used to prove the decidability of nominal E-unification for certain equational theories.

Related Work. Narrowing has traditionally been used to solve equations in initial and free algebras modulo a set of equations. It is well-known that narrowing is a programming feature that allows integration of functional and logical programming languages [8, 20]. Narrowing was originally introduced for theorem proving [17], but nowadays it is used in type inference [27] and verification of cryptographic protocols [23], amongst other areas. Narrowing gives rise to a complete E-unification procedure if E is defined by a convergent rewrite system, but it is generally inefficient. Several strategies have been designed to make narrowing-based E-unification procedures more efficient by reducing the search space (e.g., basic narrowing [17] and variant narrowing [9], the latter inspired by the notion of E-variant [6]) and sufficient conditions for termination have been obtained [17, 9, 1]. In this paper we develop basic nominal narrowing strategies and associated termination conditions, and leave the study of other complete strategies for future work.

Nominal unification is closely related to higher-order pattern unification [18] and there is previous work addressing higher-order pattern E-unification: Prehofer [26] introduced higher-order narrowing and some variants (such as lazy narrowing, conditional narrowing, pattern narrowing), and considered applications of narrowing as an inference rule in logic and functional programming. Nominal extensions of logic and functional programming languages are already available (see, e.g., [28, 5]), and nominal narrowing could play a similar role in the definition of a functional-logic programming language.

Overview of the paper: Section 2 recalls basic concepts in nominal unification and rewriting. Section 3 introduces the notion of *nominal narrowing*, presents results relating nominal narrowing and nominal equational unification, and gives examples of application. Section 4 introduces *basic nominal narrowing* and the results regarding the termination of narrowing. Section 5 contains the conclusions and directions for future work.

2 Nominal Rewriting

We recall below the definitions of nominal unification and nominal rewriting; for more details we refer the reader to [11, 29].

2.1 Nominal terms and α-equivalence

A *nominal signature* Σ is a set of *function symbols* $f, g, \ldots$, each with a fixed *arity* $n \geq 0$. Fix a countably infinite set $\mathcal{X}$ of *variables* $X, Y, Z, \ldots$; these represent meta-level unknowns. Also, fix a countably infinite set $\mathcal{A}$ of *atoms* $a, b, c, n, x, \ldots$; these represent object-level variables. We assume that Σ, $\mathcal{X}$ and $\mathcal{A}$ are pairwise disjoint.

Nominal terms are generated by the grammar: $t ::= a \mid \pi \cdot X \mid [a]t \mid f(t_1, \ldots, t_n)$. Terms are called respectively *atoms*, *suspensions*, *abstractions* and *function applications*. We write $V(t)$ for the set of variables occurring in t, $A(t)$ for the set of atoms mentioned in t, and $atm(t)$ for the set of atoms that occur as subterms in t. For example, $A([a]b) = \{a, b\}$, $b \in atm([a]b)$, $a \notin atm([a]b)$. *Ground terms* are terms without variables, they may still contain atoms. The occurrences of a in a term are said to be *bound* (or abstracted) if they occur in the scope of an abstraction, otherwise they are said to be *free* (or unabstracted).

A *permutation* π is a bijection on atoms, with finite domain. $\pi \circ \pi'$ denotes *functional composition* of permutations and π^{-1} denotes the *inverse* of π. A *permutation action* $\pi \cdot t$ is defined by induction: $\pi \cdot a \equiv \pi(a)$, $\pi \cdot [a]t \equiv [\pi(a)](\pi \cdot t)$, $\pi \cdot (\pi' \cdot X) \equiv (\pi \circ \pi') \cdot X$ and $\pi \cdot f(t_1, \ldots, t_n) \equiv f(\pi \cdot t_1, \ldots, \pi \cdot t_n)$. We write $(a\ b)$ for the *swapping* permutation that maps a to b, b to a and all other atoms c to themselves, and Id for the *identity permutation*, so $Id(a) = a$. Note that X is not a term, but $Id \cdot X$ is. We abbreviate $Id \cdot X$ as X when there is no ambiguity.

A *substitution* σ is a mapping from variables to terms, with a finite domain denoted by $dom(\sigma)$; the image is denoted $Im(\sigma)$. Henceforth, if $X \notin dom(\sigma)$ then $\sigma(X)$ denotes $Id \cdot X$. Substitutions are generated by the grammar: $\sigma := Id \mid \{X \mapsto s\}\sigma$, where Id denotes the substitution with $dom(Id) = \emptyset$. We use the same notation for the identity permutation and the identity substitution, as there will be no ambiguity. For every substitution σ, we define $\sigma|_V$ (the *restriction of σ to V*) as the substitution that maps X to $\sigma(X)$ if $X \in V$ and to $Id \cdot X$ otherwise. The *substitution action $t\sigma$* is defined as follows: $a\sigma \equiv a$, $([a]t)\sigma \equiv [a](t\sigma)$, $f(t_1, \ldots, t_n)\sigma \equiv f(t_1\sigma, \ldots, t_n\sigma)$ and $(\pi \cdot X)\sigma \equiv \pi \cdot \sigma(X)$. If σ and θ are substitutions, $\theta \circ \sigma$ is the substitution that maps each X to $(X\sigma)\theta$. Note that substitution allows capture of free atoms (it behaves like first-order substitution, except that when instantiating $\pi \cdot X$, π applies).

On nominal terms, α-equivalence is defined using swappings and a notion of freshness. A *freshness constraint* is a pair $a\#t$ (read "a fresh in t") of an atom a and a term t. Intuitively, $a\#t$ means that if a occurs in t then it must be abstracted. An *α-equality constraint* is a pair $s \approx_\alpha t$ of two terms s and t. A *freshness context* is a set of freshness constraints of the form $a\#X$. Δ, Γ and ∇ will range over freshness contexts. A *freshness judgement* is a tuple of the form $\Delta \vdash a\#t$ whereas an *α-equivalence judgement* is a tuple of the form $\Delta \vdash s \approx_\alpha t$. The *derivable* freshness and α-equivalence judgements are defined by the rules in Figure 1. A set Pr of constraints is called a *problem*. We write $\Delta \vdash Pr$ when proofs exist for each $P \in Pr$, using the derivation rules given in Figure 1. The minimal Δ such that $\Delta \vdash Pr$, denoted by $\langle Pr \rangle_{nf}$, can be obtained by using a system of simplification rules [11, 29], which, given Pr, outputs Δ or fails.

2.2 Unification, Matching and Nominal Rewriting

Unification is about finding a substitution that makes two terms equal. For nominal terms the notion of equality is $\approx_\alpha$, which is defined in a freshness context; nominal unification takes this into account.

▶ **Definition 1.** A *solution* for a problem Pr is a pair (Γ, σ) such that $\Gamma \vdash Pr\sigma$, where $Pr\sigma$ is the problem obtained by applying the substitution σ to the terms in Pr.

We follow [11], defining nominal matching/unification problems *in context*. A *term-in-context* is a pair $\Delta \vdash t$ of a freshness context and a term. We may write $\vdash t$ or simply t if $\Delta = \emptyset$.

$$\frac{}{\Delta \vdash a\#b}\,(\#\mathbf{ab}) \qquad \frac{}{\Delta \vdash a\#[a]t}\,(\#[\mathbf{a}]) \qquad \frac{(\pi^{-1}(a)\#X) \in \Delta}{\Delta \vdash a\#\pi \cdot X}\,(\#\mathbf{X})$$

$$\frac{\Delta \vdash a\#t}{\Delta \vdash a\#[b]t}\,(\#[\mathbf{b}]) \qquad \frac{\Delta \vdash a\#t_1 \;\cdots\; \Delta \vdash a\#t_n}{\Delta \vdash a\#\mathsf{f}(t_1,\ldots,t_n)}\,(\#\mathsf{f}) \qquad \frac{}{\Delta \vdash a \approx_\alpha a}\,(\approx_\alpha \mathbf{a})$$

$$\frac{\Delta \vdash b\#t \quad \Delta \vdash (b\ a)\cdot t \approx_\alpha u}{\Delta \vdash [a]t \approx_\alpha [b]u}\,(\approx_\alpha [\mathbf{b}]) \qquad \frac{(a\#X \in \Delta \text{ for all } a \text{ s.t. } \pi(a) \neq \pi'(a))}{\Delta \vdash \pi \cdot X \approx_\alpha \pi' \cdot X}\,(\approx_\alpha \mathbf{X})$$

$$\frac{\Delta \vdash t \approx_\alpha u}{\Delta \vdash [a]t \approx_\alpha [a]u}\,(\approx_\alpha [\mathbf{a}]) \qquad \frac{\Delta \vdash t_i \approx_\alpha u_i \quad (1 \leq i \leq n)}{\Delta \vdash \mathsf{f}(t_1,\ldots,t_n) \approx_\alpha \mathsf{f}(u_1,\ldots,u_n)}\,(\approx_\alpha \mathsf{f})$$

Figure 1 Freshness and α-equality.

The action of substitutions extends to freshness contexts, instantiating the variables in freshness constraints.

▶ **Definition 2.** A *unification problem (in context)* is a pair $(\nabla \vdash l)\ {}_?{\approx}_?\ (\Delta \vdash s)$ where Δ, ∇ are freshness contexts and l, s are nominal terms. The *solution* to this unification problem, if it exists, is a pair (Δ', θ) that solves the problem $\Delta, \nabla, l \approx_\alpha s$, that is, $\Delta' \vdash \Delta\theta, \nabla\theta, l\theta \approx_\alpha s\theta$.

A *matching problem (in context)* is a particular kind of unification problem, written $(\nabla \vdash l)\ {}_?{\approx}\ (\Delta \vdash s)$,[1] where s is ground, or contains variables not occurring in ∇, l. The solution (Δ', θ) is such that $X\theta \equiv X$ for $X \in V(\Delta, s)$ (i.e., θ can only instantiate variables in ∇ and l, therefore, $\Delta' \vdash \Delta, \nabla\theta$ and $\Delta' \vdash l\theta \approx_\alpha s$).

▶ **Example 3.** $(\vdash [a][b]X')\ {}_?{\approx}\ (\vdash [b][a]X)$ has solution $(\emptyset, \{X' \mapsto (a\,b)\cdot X\})$.

▶ **Definition 4.** Let Γ_1, Γ_2 be contexts, and σ_1, σ_2 substitutions. Then $(\Gamma_1, \sigma_1) \leq (\Gamma_2, \sigma_2)$ if there exists some σ' such that: $\forall X,\ \Gamma_2 \vdash X\sigma_1\sigma' \approx_\alpha X\sigma_2\ and\ \Gamma_2 \vdash \Gamma_1\sigma'$. If we want to be more specific, we may write $(\Gamma_1, \sigma_1) \leq_{\sigma'} (\Gamma_2, \sigma_2)$. The relation $\leq$ is a partial order

Nominal unification is decidable and unitary [29]: a solvable problem has a unique least solution according to $\leq$, called *principal solution* or *most general unifier*, denoted by $mgu(Pr)$.

Below we recall the definitions of nominal equational reasoning [15] and nominal rewriting [11] from [12], where a *position* C is defined as a pair $(s, \_)$ of a term and a distinguished variable $\_ \in \mathcal{X}$ that occurs precisely once in s, with permutation Id. C is also called a *context*. When there is no ambiguity, we equate C with s and write $C[t]$ for the result of applying the substitution $\{\_ \mapsto t\}$ to s[2]. $\mathcal{P}os(u)$ denotes the set of positions of the nominal term u, that is, all the positions C such that $u = C[t]$ for some t. $\overline{\mathcal{P}os}(u) = \{C \in \mathcal{P}os(u) |\, u = C[t] \text{ and } t \neq \pi \cdot X\}$ is the set of non-variable positions.

An *equality judgement* (resp. *rewrite judgement*) is a tuple $\Delta \vdash s = t$ (resp. $\Delta \vdash s \to t$) of a freshness context Δ and two nominal terms s, t. An *equational theory* $\mathsf{E} = (\Sigma, Ax)$ is a pair of a signature Σ and a possibly infinite set of equality judgements Ax in Σ; they are called *axioms*. A *rewrite theory* $\mathsf{R} = (\Sigma, Rw)$ is a pair of a signature Σ and a possibly infinite set of rewrite judgements Rw in Σ; they are called *rewrite rules*. Σ may be omitted, identifying E

[1] The ${}_?{\approx}$ indicates that the variables being instantiated occur in the left-hand side term.
[2] This definition of position is equivalent to the standard notion of a position as a path in a tree; here we exploit the fact that nominal substitution corresponds to the informal notion of replacement of a 'hole' in a context by a term.

$$
\begin{array}{rlll}
 & \vdash app(lam([a]X), X') & \to sub([a]X, X') & (Beta) \\
 & \vdash sub([a]a, X) & \to X & \\
a\#Y & \vdash sub([a]Y, X) & \to Y & \\
 & \vdash sub([a]app(X, X'), Y) & \to app(sub([a]X, Y), sub([a]X', Y)) & \\
b\#Y & \vdash sub([a]lam([b]X), Y) & \to lam([b]sub([a]X, Y)) &
\end{array}
$$

Figure 2 λ-calculus with names and explicit substitutions [13].

with Ax and R with Rw when the signature is clear from the context. See Figure 2 for an example of a rewrite theory for the λ-calculus.

▶ **Definition 5.**

- *Nominal rewriting:* The *one-step rewrite relation* $\Delta \vdash s \xrightarrow{\mathsf{R}}_{[C,R,\theta,\pi]} t$ is the least relation such that for any $R = (\nabla \vdash l \to r) \in \mathsf{R}$, position C, term s', permutation π, and substitution θ,

$$\frac{s \equiv C[s'] \quad \Delta \vdash \left(\nabla\theta, s' \approx_\alpha \pi \cdot (l\theta), C[\pi \cdot (r\theta)] \approx_\alpha t\right)}{\Delta \vdash s \xrightarrow{\mathsf{R}}_{[C,R,\theta,\pi]} t}$$

  We may omit subindices if they are clear from the context, writing simply $\Delta \vdash s \xrightarrow{\mathsf{R}} t$. The *rewrite relation* $\Delta \vdash_{\mathsf{R}} s \to t$ is the reflexive transitive closure of the one-step rewrite relation, that is, the least relation that includes the one-step rewrite relation and such that: for all Δ, s, s': $\Delta \vdash_{\mathsf{R}} s \to s'$ if $\Delta \vdash s \approx_\alpha s'$ (the native notion of equality of nominal terms is α-equality)[3]; for all Δ, s, t, u: $\Delta \vdash_{\mathsf{R}} s \to t$ and $\Delta \vdash_{\mathsf{R}} t \to u$ implies $\Delta \vdash_{\mathsf{R}} s \to u$. If $\Delta \vdash_{\mathsf{R}} s \to t$ holds, we say that s rewrites to t in the context Δ. A *normal form* is a term-in-context $\Delta \vdash s$ that does not rewrite, that is, there is no t such that $\Delta \vdash s \xrightarrow{\mathsf{R}} t$. A rewrite theory R is convergent if the rewrite relation is confluent and terminating.
- *(Nominal algebra) equality:* $\Delta \vdash_{\mathsf{E}} s = t$ is the least transitive reflexive symmetric relation such that for any $(\nabla \vdash l = r) \in \mathsf{E}$, position C, permutation π, substitution θ, and fresh Γ (so if $a\#X \in \Gamma$ then a is not mentioned in Δ, s, t),

$$\frac{\Delta, \Gamma \vdash \left(\nabla\theta, \quad s \approx_\alpha C[\pi \cdot (l\theta)], \quad C[\pi \cdot (r\theta)] \approx_\alpha t\right)}{\Delta \vdash_{\mathsf{E}} s = t}.$$

Given an equational theory E and a rewrite theory R, we say that R is a *presentation* of E if: $\nabla \vdash s = t \in \mathsf{E} \Leftrightarrow (\nabla \vdash s \to t \in \mathsf{R} \vee \nabla \vdash t \to s \in \mathsf{R})$.

Nominal rewriting is not complete for equational reasoning in general; however, *closed nominal rewriting* is complete for equational reasoning with *closed* axioms (see [12]). Intuitively, no free atom occurs in a closed term, and closed axioms do not allow abstracted atoms to become free (a natural assumption). Closedness of a term can be easily checked by matching the term with a freshened copy of itself. For example, the term $f(a)$ is not closed (it is not possible to match $f(a)$ with a freshened variant $f(a')$); however, $f([a]a)$ is closed ($f([a]a) \approx_\alpha f([a']a')$). If there are variables, freshness contexts have to be taken into account. We recall below the definitions of freshened variant, closed rewrite rule and closed rewriting relation from [12].

[3] As in the case of conditional rewriting modulo an equivalence theory (see [22]), reflexivity takes into account the underlying equivalence relation, here $\approx_\alpha$.

If t is a term, we say that $t^{\shortparallel}$ is a *freshened variant* of t when $t^{\shortparallel}$ has the same structure as t, except that the atoms and unknowns have been replaced by 'fresh' atoms and unknowns. Similarly, if ∇ is a freshness context then $\nabla^{\shortparallel}$ will denote a freshened variant of ∇ (so if $a\#X \in \nabla$ then $a^{\shortparallel}\#X^{\shortparallel} \in \nabla^{\shortparallel}$, where $a^{\shortparallel}$ and $X^{\shortparallel}$ are chosen fresh for the atoms and unknowns appearing in ∇). We may extend this to other syntax, like equality and rewrite judgements. For example, $[a^{\shortparallel}][b^{\shortparallel}]X^{\shortparallel}$ is a freshened variant of $[a][b]X$, $a^{\shortparallel}\#X^{\shortparallel}$ is a freshened variant of $a\#X$, and $\emptyset \vdash f([a^{\shortparallel}]X^{\shortparallel}) \to [a^{\shortparallel}]X^{\shortparallel}$ is a freshened variant of $\emptyset \vdash f([a]X) \to [a]X$.

▶ **Definition 6** (Closed terms and rules, closed rewriting)**.** A term-in-context $\nabla \vdash l$ is *closed* if there exists a solution for the matching problem $(\nabla^{\shortparallel} \vdash l^{\shortparallel}) \;\; {}_?{\approx} \;\; (\nabla, A(\nabla^{\shortparallel}, l^{\shortparallel})\#V(\nabla, l) \vdash l)$[4]. Call $R = (\nabla \vdash l \to r)$ and $Ax = (\nabla \vdash l = r)$ *closed* when $\nabla \vdash (l, r)$ is closed[5]. Given a rewrite rule $R = (\nabla \vdash l \to r)$ and a term-in-context $\Delta \vdash s$, write $\Delta \vdash s \to^c_R t$ when there is some $R^{\shortparallel}$ a freshened variant of R (so fresh for R, Δ, s, and t), position C and substitution θ such that $s \equiv C[s']$ and $\Delta, A(R^{\shortparallel}) \# V(\Delta, s, t) \vdash (\nabla^{\shortparallel}\theta,\ s' {\approx_\alpha} l^{\shortparallel}\theta,\ C[r^{\shortparallel}\theta] {\approx_\alpha} t)$. We call this (one-step) *closed rewriting*. The *closed-rewrite relation* $\Delta \vdash_{\mathsf{R}} s \to^c t$ is the reflexive transitive closure as in Definition 5.

All the rewrite rules in Figure 2 are closed. Closed rewriting is an efficient mechanism to generate rewriting steps for closed rules (closed-rewriting steps can be generated simply using nominal matching; it is not necessary to find a permutation π to apply a rule). We refer the reader to [11, 12] for examples.

3 Nominal E-Unification and Narrowing

We start by generalising the notion of solution.

▶ **Definition 7** (Nominal E-unification)**.** An E-solution, or E-unifier, of a problem Pr is a pair (Γ, σ) of a freshness context and a substitution such that

1. $\Gamma \vdash_{\mathsf{E}} Pr'\sigma$ where Pr' is obtained from Pr by replacing each $\approx_\alpha$ by $=$, and $\Gamma \vdash_{\mathsf{E}} a\#t$ coincides with $\Gamma \vdash a\#t$.
2. $X\sigma = X\sigma\sigma$ for all X (i.e., σ is *idempotent*).

If there is no such (Γ, σ) then Pr is *unsolvable*. $\mathcal{U}_{\mathsf{E}}(Pr)$ is the *set of* E-*solutions* of Pr.

The notion of E-unification extends to terms-in-context in the natural way.

▶ **Definition 8.** A *nominal* E-*unification problem (in context)* is a pair $(\nabla \vdash l) \,{}_?\overset{\mathsf{E}}{\approx}_?\, (\Delta \vdash s)$. The pair (Δ', σ) is an E-solution, or E-unifier, of $(\nabla \vdash l) \,{}_?\overset{\mathsf{E}}{\approx}_?\, (\Delta \vdash s)$ iff (Δ', σ) is an E-solution of the problem $\nabla, \Delta, l \approx_\alpha s$, that is, $\Delta' \vdash_{\mathsf{E}} \nabla\sigma, \Delta\sigma, l\sigma = s\sigma$.

$\mathcal{U}_{\mathsf{E}}(\nabla \vdash l, \Delta \vdash s)$ denotes the set of all the E-solutions of $(\nabla \vdash l) \,{}_?\overset{\mathsf{E}}{\approx}_?\, (\Delta \vdash s)$. If ∇ and Δ are empty we write $\mathcal{U}_{\mathsf{E}}(l, s)$ for the set of E-unifiers of l and s.

Nominal E-*matching problems in context* are defined similarly, except that s is a ground term (or, if it has variables, the solution cannot instantiate them). E-matching problems in context are written $(\nabla \vdash l) \,{}_?\overset{\mathsf{E}}{\approx}\, (\Delta \vdash s)$.

▶ **Definition 9.** The ordering $\leq_{\mathsf{E}}$ is the extension of $\leq$ with respect to E: $(\Gamma_1, \sigma_1) \leq_{\mathsf{E}} (\Gamma_2, \sigma_2)$ iff there exists a substitution ρ such that $\forall X,\ \Gamma_2 \vdash_{\mathsf{E}} X\sigma_2 = (X\sigma_1)\rho$ and $\Gamma_2 \vdash \Gamma_1\rho$. We write $\leq^V_{\mathsf{E}}$ for the restriction of $\leq_{\mathsf{E}}$ to the set V of variables.

---

[4] $A(\nabla^{\shortparallel}, l^{\shortparallel})\#V(\nabla, l) = \{a\#X \mid a \in A(\nabla^{\shortparallel}, l^{\shortparallel}), X \in V(\nabla, l)\}$.
[5] Here we use the pair constructor as a term former and apply the definition above.

▶ **Definition 10** (Complete set of E-solutions of Pr)**.** Let W be a finite set of variables containing $V = V(Pr)$. We say that $\mathcal{S} = \{(\Gamma_1, \theta_1), \ldots, (\Gamma_n, \theta_n)\}$ is a *complete set of* E*-solutions of* Pr *away from* W iff

1. $\forall(\Gamma, \theta) \in \mathcal{S}$, $dom(\theta) \subseteq V$ and $Im(\theta) \cap W = \emptyset$,
2. $\mathcal{S} \subseteq \mathcal{U}_{\mathsf{E}}(Pr)$ (correctness),
3. $\forall(\Gamma, \sigma) \in \mathcal{U}_{\mathsf{E}}(Pr)\ \exists(\Gamma_i, \theta_i) \in \mathcal{S}$, $(\Gamma_i, \theta_i) \leq^V_{\mathsf{E}} (\Gamma, \sigma)$ (completeness).

We are now ready to define the *nominal narrowing relation* generated by R. The definition of nominal narrowing is similar to nominal rewriting, but we need to solve unification problems instead of matching problems.

▶ **Definition 11** (Nominal Narrowing)**.** The *one-step narrowing relation* $(\Delta \vdash s) \rightsquigarrow_{[C,R,\theta,\pi]} (\Delta' \vdash t)$ is the least relation such that for any $R = (\nabla \vdash l \to r) \in \mathsf{R}$, position C, term s', permutation π, and substitution θ,

$$\frac{s \equiv C[s'] \quad \Delta' \vdash \big(\nabla\theta,\ \Delta\theta,\ s'\theta \approx_\alpha \pi \cdot (l\theta),\ (C[\pi \cdot r])\theta \approx_\alpha t\big)}{(\Delta \vdash s) \rightsquigarrow_{[C,R,\theta,\pi]} (\Delta' \vdash t)}\ (\Delta', \theta) = mgu(\nabla, \Delta, s' \approx_\alpha \pi \cdot l).$$

We may omit subindices if they are clear from the context.
The *narrowing relation* $(\Delta \vdash s) \rightsquigarrow_{\mathsf{R}} (\Delta' \vdash t)$ is the reflexive transitive closure of the one-step narrowing relation, that is, the least relation that includes the one-step narrowing relation and such that: for all Δ, s, s': $(\Delta \vdash s) \rightsquigarrow_{\mathsf{R}} (\Delta \vdash s')$ if $\Delta \vdash s \approx_\alpha s'$; for all Δ, Δ', Δ'', s, t, u: $(\Delta \vdash s) \rightsquigarrow_{\mathsf{R}} (\Delta' \vdash t)$ and $(\Delta' \vdash t) \rightsquigarrow_{\mathsf{R}} (\Delta'' \vdash u)$ implies $(\Delta \vdash s) \rightsquigarrow_{\mathsf{R}} (\Delta'' \vdash u)$.

The Lifting Theorem given below relates nominal narrowing and nominal rewriting. It is an extension of Hullot's Theorem 1 [17], taking into account freshness contexts and α-equivalence. The notions of normalised substitution-in-context and satisfiability of freshness contexts play a key role. A substitution σ is normalised in Δ w.r.t. a rewrite theory R if $\Delta \vdash X\sigma$ is a normal form in R for every X. A substitution σ satisfies the freshness context Δ if there exists a freshness context ∇ such that $\nabla \vdash a\#X\sigma$ for each $a\#X \in \Delta$; the minimal such ∇ is $\langle \Delta\sigma \rangle_{nf}$.

▶ **Theorem 12** (Lifting)**.** *Let* $\mathsf{R} = \{\nabla_i \vdash l_i \to r_i\}$ *be a convergent rewrite theory. Let* $\Delta_0 \vdash s_0$ *be a nominal term-in-context and* V_0 *a finite set of variables containing* $V = V(\Delta_0, s_0)$*. Let* η *be a substitution with* $dom(\eta) \subseteq V_0$ *and satisfying* Δ_0*, that is, there exists* Δ *such that* $\Delta \vdash \Delta_0\eta$*. Assume moreover that* η *is normalised in* Δ*. Consider a rewrite derivation:*

$$\Delta \vdash s_0\eta = t_0 \to_{[C_0,R_0]} \cdots \to_{[C_{n-1},R_{n-1}]} t_n \qquad (*)$$

There exists an associated nominal narrowing derivation:

$$(\Delta_0 \vdash s_0) \rightsquigarrow_{[C'_0,R_0,\sigma_0]} \cdots \rightsquigarrow_{[C'_{n-1},R_{n-1},\sigma_{n-1}]} (\Delta_n \vdash s_n) \qquad (**)$$

for each i*,* $0 \leq i \leq n$*, a substitution* η_i *and a finite set of variables* $V_i \supseteq V(s_i)$ *such that:*

1. $dom(\eta_i) \subseteq V_i$*,*
2. η_i *is normalised in* Δ*,*
3. $\Delta \vdash \eta|_V \approx_\alpha \theta_i\eta_i|_V$*,*
4. $\Delta \vdash s_i\eta_i \approx_\alpha t_i$*,*
5. $\Delta \vdash \Delta_i\eta_i$

where $\theta_0 = Id$ *and* $\theta_{i+1} = \theta_i\sigma_i$*.*

Conversely, to each nominal narrowing derivation of the form $(**)$ *and every* η *such that* $(\Delta_n, \theta_n) \leq^V (\Delta, \eta)$ *and* $\Delta \vdash s_i\eta_i \approx_\alpha t_i$ *we can associate a nominal rewriting derivation of the form* $(*)$*.*

$$
\begin{tikzcd}
\Delta \vdash s\eta = t_0 \arrow[r] & \ldots & t_i \arrow[r] & t_{i+1} \arrow[r] & \ldots \arrow[r] & t_n \\
\Delta_0 \vdash s_0 \arrow[r, rightsquigarrow] \arrow[u, dashed, "\eta_0"] & \ldots & (\Delta_i \vdash s_i) \arrow[r, rightsquigarrow] \arrow[u, dashed, "\eta_i"] & (\Delta_{i+1} \vdash s_{i+1}) \arrow[r, rightsquigarrow] \arrow[u, dashed, "\eta_{i+1}"] & \ldots \arrow[r, rightsquigarrow] & (\Delta_n \vdash s_n) \arrow[u, dashed, "\eta_n"]
\end{tikzcd}
$$

Figure 3 Corresponding Rewriting and Narrowing Steps.

Proof.

($\Longrightarrow$) The proof is by induction on the length of the derivation. Figure 3 illustrates the relation between the two derivations.

Base Case. For $n = 0$, take $\eta_0 = \eta$, $V_0 = V \cup dom(\eta)$. By assumption, $\Delta \vdash \Delta_0\eta_0$.

$$\Delta_0 \vdash s_0 \dashrightarrow_{\eta_0} \Delta \vdash s_0\eta = t_0$$

Induction Step. Assume conditions (1)-(5) hold for i, and $\Delta \vdash t_i \to_{[C_i,R_i]} t_{i+1}$ (see Figure 3). We have:

(a) $R_i = \nabla_i \vdash l_i \to r_i \in \mathsf{R}$, $V(R_i) \cap V(\Delta, t_i) = \emptyset$,
(b) $t_i \equiv C_i[t'_i]$ for some position $C_i[\_]$ and $\Delta \vdash \nabla_i\sigma, \pi \cdot (l_i\sigma) \approx_\alpha t'_i$.
(c) $\Delta \vdash C_i[\pi \cdot (r_i\sigma)] \approx_\alpha t_{i+1}$

Also, $dom(\sigma) \cap V_i = \emptyset$ since $V(R_i) \cap V(\Delta, t_i) = \emptyset$.

By IH, it follows from assumptions 2., 4. and 5. that $s_i \equiv C'_i[s'_i]$ where $C'_i[\_] \in \overline{\mathcal{P}os}(s_i)$ and $\Delta \vdash s'_i\eta_i \approx_\alpha t'_i \approx_\alpha \pi \cdot (l_i\sigma)$ (if $C'_i[\_]$ were a variable position the term s'_i would be a variable, from (4), $\Delta \vdash s'_i\eta_i \approx_\alpha t'_i \approx_\alpha \pi \cdot (l_i\sigma) \to_{R_i} \pi \cdot (r_i\sigma)$, contradicting that η_i is a normalised substitution).

Let us consider $\rho = \eta_i \cup \sigma$, we have $\Delta \vdash s'_i\rho \approx_\alpha \pi \cdot (l_i\rho)$. The pair (Δ, ρ) is a solution for $(\Delta_i \vdash s'_i)\ {}_?{\approx}_?\ (\nabla_i \vdash l_i)$:

(i) $\Delta \vdash \Delta_i\rho$, because, by hypothesis, $\Delta \vdash \Delta_i\eta_i$ and σ does not affect Δ_i ($dom(\sigma) \subseteq V(R_i)$).
(ii) $\Delta \vdash \nabla_i\rho$.
(iii) $\Delta \vdash s'_i\rho \approx_\alpha \pi \cdot (l_i\rho)$.

Now, take the principal solution (Δ_{i+1}, σ_i) of $(\Delta_i \vdash s'_i)\ {}_?{\approx}_?\ (\nabla_i \vdash \pi \cdot l_i)$. Then, $\Delta_{i+1} \vdash \Delta_i\sigma_i, \nabla_i\sigma_i, s'_i\sigma \approx_\alpha \pi \cdot (l_i\sigma_i)$. Let s_{i+1} be a nominal term such that $\Delta_{i+1} \vdash C'_i[\pi \cdot r_i]\sigma_i \approx_\alpha s_{i+1}$. Therefore, $(\Delta_i \vdash s_i) \rightsquigarrow_{[C_i,R_i,\sigma_i]} (\Delta_{i+1} \vdash s_{i+1})$.

Since (Δ_{i+1}, σ_i) is the least unifier of $(\Delta_i \vdash s'_i)\ {}_?{\approx}_?\ (\nabla_i \vdash \pi \cdot l_i)$, $(\Delta_{i+1}, \sigma_i) \leq (\Delta, \rho)$ and thus there exists a substitution η' such that *for all* X, $\Delta \vdash X\sigma_i\eta' \approx_\alpha X\rho$ *and* $\Delta \vdash \Delta_{i+1}\eta'$. That is, $\Delta \vdash \sigma_i\eta' \approx_\alpha \rho$. Since $\rho = \eta_i \cup \sigma$ and $dom(\sigma) \cap V_i = \emptyset$, η_i is such that $\Delta \vdash \eta_i \approx_\alpha \sigma_i\eta'|_{V_i}$.

Now let $V_{i+1} = (V_i \cup Im(\sigma_i)) - dom(\sigma_i)$ and let η_{i+1} be such that $\Delta \vdash \eta_{i+1} \approx_\alpha \eta'|_{V_{i+1}}$. We get condition 1., that is, $dom(\eta_{i+1}) \subseteq V_{i+1}$ and from 3.: $\Delta \vdash \eta_i \approx_\alpha (\sigma_i\eta_{i+1})|_{V_i}$ (1).

(By hypothesis, $\Delta \vdash \eta|_V \approx_\alpha \theta_i\eta_i$. To illustrate, take $i = 4$, then $\eta|_V = \theta_4\eta_4 = \theta_5\eta_5$. Using the definition of θ_i, it follows that $\theta_4 = \sigma_0\sigma_1 \ldots \sigma_3$ and $\theta_5 = \sigma_0\sigma_1 \ldots \sigma_4$. Thus, $\theta_4\eta_4 = \sigma_0\sigma_1 \ldots \sigma_3\eta_4 = \sigma_0\sigma_1 \ldots \sigma_3\sigma_4\eta_5$ and $\eta_4 = \sigma_4\eta_5$.) Recall that we impose $dom(\sigma_i) \cap Im(\sigma_i) = \emptyset$.

To prove 5. for $i + 1$, notice that from $\Delta \vdash \Delta_{i+1}\eta'$ it follows that $\Delta \vdash \Delta_{i+1}\eta_{i+1}$, since $\Delta \vdash \eta_{i+1} \approx_\alpha \eta'$.

To prove 2. for $i + 1$, let us consider $X \in V_{i+1}$. There are two cases:

(i') $X \in V_i - dom(\sigma_i)$ then $\Delta \vdash X\eta_i \approx_\alpha X\sigma_i\eta' \approx_\alpha X\eta' \approx_\alpha X\eta_{i+1}$. Since η_i is a normalised substitution, by hypothesis, it follows that η_{i+1} is also a normalised substitution.

(ii') $X \in V(Im(\sigma_i))$, then there exists $Y \in dom(\sigma_i)$ such that $X \in V(Y\sigma_i)$. Then, $X\eta_{i+1}$ is a subterm of $Y\eta_i$ since $\Delta \vdash X\eta_{i+1} \approx_\alpha X\eta', Y\sigma_i\eta' \approx_\alpha Y\eta_i$, and since, by hypothesis, η_i is a normalised substitution, it follows that η_{i+1} is also normalised.

This proves (2) for $i+1$.

We now prove 3. for $i+1$, assuming it for i, i.e., $\Delta \vdash \eta|_V \approx_\alpha \theta_i\eta_i|_V$.

From equation (1) we get $\Delta \vdash \theta_i\eta_i|_V \approx_\alpha \theta_i(\sigma_i\eta_{i+1}|_{V_i})|_V$ From the definition of θ_i we have $Im(\theta_i) \subseteq V_i$ and $V_0 \subseteq V_i \cup dom(\theta_i)$. Therefore, $\Delta \vdash \underbrace{\theta_i\sigma_i}_{\theta_{i+1}}\eta_{i+1}|_V \approx_\alpha \theta_i\eta_i|_V \approx_\alpha \eta|_V$ proving condition 3) for $i+1$. Notice that, by 3), θ_i is normalised.

Finally, on the one hand $\Delta \vdash t_{i+1} \approx_\alpha C_i[\pi \cdot r_i\sigma] \approx_\alpha C_i[\pi \cdot r_i\rho] \approx_\alpha C_i[\pi \cdot r_i(\sigma_i\eta')] \approx_\alpha C_i[\pi \cdot r_i\eta_i]$. On the other hand, $\Delta \vdash s_{i+1}\eta_{i+1} \approx_\alpha (C'_i[\pi \cdot r_i]\sigma_i)\eta_{i+1} \approx_\alpha (C'_i[\pi \cdot r_i]\sigma_i)\eta' \approx_\alpha (C'_i(\sigma_i\eta'))[\pi \cdot r_i(\sigma_i\eta')] \approx_\alpha (C_i\eta_i)[\pi \cdot r_i\eta_i] \approx_\alpha C_i[\pi \cdot r_i\eta_i]$. Therefore, $\Delta \vdash s_{i+1}\eta_{i+1} \approx_\alpha t_{i+1}$, proving (4).

($\Longleftarrow$) Conversely, let us consider a derivation (**): $(\Delta_0 \vdash s_0) \rightsquigarrow_{[C'_0,R_0,\sigma_0]} \cdots \rightsquigarrow_{[C'_{n-1},R_{n-1},\sigma_{n-1}]} (\Delta_n \vdash s_n)$, and a substitution η such that $(\Delta_n, \theta_n) \leq^V (\Delta, \eta)$, that is, there exists ρ such that $\Delta \vdash X\eta|_V \approx_\alpha (X\theta_n)\rho|_V$ and $\Delta \vdash \Delta_n\rho$. We define substitutions η_i for $0 \leq i \leq n-1$ by: $\Delta \vdash \eta_i \approx_\alpha \sigma_i \ldots \sigma_{n-1}\rho$ (2). and a normalised substitution $\eta_n \equiv \rho$. By hypothesis, $\Delta \vdash \Delta_n\rho$, and by definition of narrowing step, it follows that $\Delta_{i+1} \vdash \Delta_i\sigma_i$ $(0 \leq i \leq n-1)$. Hence $\Delta \vdash \Delta_i\eta_i$, and in particular $\Delta \vdash \Delta_0\eta$. We define $s_i\eta_i \equiv t_i$ for $0 \leq i \leq n$, and show, by induction on i, that: $\Delta \vdash s_0\eta = t_0 \to_{[C_0,R_0]} \cdots \to_{[C_{n-1},R_{n-1}]} t_n$.

Base Case. When $i = 0$: $\Delta \vdash s_0\eta_0 \approx_\alpha s(\theta_n\eta_n) \approx_\alpha s\eta$. By definition, $\eta_0 = \underbrace{\sigma_0\sigma_1 \ldots \sigma_{n-1}}_{\theta_n}\rho$.

Induction Step. Suppose that $(\Delta_i \vdash s_i) \rightsquigarrow_{[C'_i,R_i,\sigma_i]} (\Delta_{i+1} \vdash s_{i+1})$. By the definition of nominal narrowing we have

- $R_i = \nabla_i \vdash l_i \to r_i \in \mathsf{R}$, $V(R_i) \cap V(\Delta_i, s_i) = \emptyset$.
- $s_i \equiv C'_i[s'_i]$, for a non-variable position $C'_i[\_]$ of s_i, and such that (Δ_{i+1}, σ_i) is the least solution for $(\Delta_i \vdash s'_i)\ {}_?{\approx}_?\ (\nabla_i \vdash \pi \cdot l_i)$. That is, $\Delta_{i+1} \vdash s'_i\sigma_i \approx_\alpha \pi \cdot (l_i\sigma_i)$ and $\Delta_{i+1} \vdash \Delta_i\sigma_i, \nabla_i\sigma_i$.
- $\Delta_{i+1} \vdash C'_i[\pi \cdot r_i]\sigma_i \approx_\alpha s_{i+1}$.

By definition, $\Delta \vdash s_i\eta_i \approx_\alpha t_i$. Since $C_i[\_]$ is a non-variable position and η_i is a normalised substitution, we have that $\Delta \vdash s'_i\eta_i \approx_\alpha t'_i$. In addition, define $\eta' \equiv \sigma_{i+1} \ldots \sigma_{n-1}\rho$, by equation (2) $\Delta \vdash \eta_{i+1} \approx_\alpha \eta'|_{V_{i+1}}$. $\Delta \vdash t'_i \approx_\alpha s'_i\eta_i \approx_\alpha s'_i(\sigma_i\eta') \approx_\alpha (\pi \cdot l_i\sigma_i)\eta' \to_{R_i} (\pi \cdot r_i\sigma_i)\eta'$ Therefore, $\Delta \vdash t_i \equiv C_i[t'_i] \to_{R_i} C_i[\pi \cdot r_i\sigma_i\eta'] \approx_\alpha s_{i+1}\eta_{i+1} \approx_\alpha t_{i+1}$. ◀

In a similar way, we can associate closed nominal rewriting derivations (see Definition 6) with *closed nominal narrowing* derivations, where closed narrowing is defined as follows.

▶ **Definition 13** (Closed narrowing). Given a rewrite rule $R = (\nabla \vdash l \to r)$ and a term-in-context $\Delta \vdash s$, write $(\Delta \vdash s) \rightsquigarrow^c_R (\Delta' \vdash t)$ when there is some $R^{\text{И}}$ a freshened variant of R (so fresh for R, Δ, s, and t), position C and substitution θ such that $s \equiv C[s']$ and $\Delta', A(R^{\text{И}}) \# V(\Delta, s, t) \vdash (\nabla^{\text{И}}\theta, \Delta\theta,\ s'\theta \approx_\alpha l^{\text{И}}\theta,\ (C[r^{\text{И}}])\theta \approx_\alpha t)$. We call this (one-step) *closed narrowing*. The *closed narrowing relation* $\Delta \vdash_{\mathsf{R}} s \rightsquigarrow^c \Delta' \vdash_{\mathsf{R}} t$ is the reflexive transitive closure as in Definition 5.

See Example 17 in Section 3.1 for examples of closed narrowing steps.

▶ Remark. We can state a "closed lifting" theorem by replacing nominal rewriting/narrowing for closed rewriting/narrowing. The proof is similar.

In the following we consider a *closed* nominal equational theory E, presented by a convergent set R of closed rules.

Let us consider an E-unification problem $(\Delta \vdash s) \,{}_?\overset{\mathsf{E}}{\approx}_?\, (\nabla \vdash t)$. To find a solution, we will apply *closed narrowing* on $\Delta \vdash s$ and $\nabla \vdash t$ in parallel. It will simplify matters to narrow the single term $u = (s,t)$[6] under Δ, ∇.

▶ **Lemma 14** (Soundness). *Let $\Delta \vdash s$ and $\nabla \vdash t$ be two nominal terms-in-context and $\Delta, \nabla \vdash (s,t) = u_0 \leadsto^c \ldots \leadsto^c \Delta_n \vdash u_n = (s_n, t_n)$ a closed narrowing derivation such that $\Delta_n, s_n \approx_\alpha t_n$ has a solution, say (Γ, σ). Then $(\Gamma, \theta_n\sigma)$ is an E-solution of the problem $\Delta, \nabla, s \,{}_?\overset{\mathsf{E}}{\approx}_?\, t$, where θ_n is the composition of substitutions along the narrowing derivation, as defined in Theorem 12.*

Proof. Using the $(\Leftarrow)$ part of the previous theorem with $\eta = \theta_n$, we can associate this narrowing derivation with the following rewriting derivation:
$\Gamma \vdash u_0\theta_n = v_0 \to^c v_1 \to^c v_2 \to^c \ldots \to^c v_n = (v_n^s, v_n^t)$. Thus, $\Gamma \vdash_{\mathsf{R}} s\theta_n \to^c v_n^s$ and $\Gamma \vdash_{\mathsf{R}} t\theta_n \to^c v_n^t$. Moreover, since $\eta_n = Id$ (because $\eta = \eta_n\theta_n$) it follows that $\Gamma \vdash v_n^s \approx_\alpha s_n$ and $\Gamma \vdash v_n^t \approx_\alpha t_n$, thus: $\Gamma \vdash_{\mathsf{E}} s\theta_n\sigma = t\theta_n\sigma$ and therefore, $(\Gamma, \theta_n\sigma)$ is an E-solution for $\Delta, \nabla, s \,{}_?\overset{\mathsf{E}}{\approx}_?\, t$. ◀

▶ **Lemma 15** (Completeness). *Let $\Delta \vdash s$ and $\nabla \vdash t$ be two nominal terms-in-context, such that the problem $(\Delta \vdash s) \,{}_?\overset{\mathsf{E}}{\approx}_?\, (\nabla \vdash t)$ has an E-solution, (Δ', ρ), and let V be a finite set of variables containing $V(\Delta, \nabla, s, t)$. Then there exists a closed narrowing derivation: $\nabla, \Delta \vdash u = (s,t) \leadsto^c \ldots \leadsto^c \Gamma_n \vdash (s_n, t_n)$, such that $\Gamma_n, s_n \approx_\alpha t_n$ has a solution. Let $(\Gamma, \mu) = mgu(\Gamma_n, s_n \approx_\alpha t_n)$, and θ_n the composition of the narrowing substitutions. Then, $(\Gamma, \theta_n\mu) \leq_{\mathsf{E}}^V (\Delta', \rho)$. Moreover, we are allowed to restrict our attention to $\leadsto^c$-derivations such that: $\forall i, 0 \leq i \leq n$, $\theta_i|_V$ is normalised.*

Proof. By Definition 8, $\Delta' \vdash_{\mathsf{E}} s\rho = t\rho, \nabla\rho, \Delta\rho$. Take $\eta = \rho\downarrow$, that is, ρ's normal form in Δ': $\Delta' \vdash X\eta \approx_\alpha (X\rho)\downarrow$. It follows that $\Delta' \vdash_{\mathsf{E}} s\eta = t\eta, \nabla\eta, \Delta\eta$ since the rules are closed.

Since E is a closed nominal theory presented by a convergent rewrite system R, and since closed rewriting is complete for equational reasoning in this case, $s\eta$ and $t\eta$ have the same normal form in Δ', which we will call r. Then, $\Delta' \vdash u\eta = (s\eta, t\eta) = t'_0 \to^c \ldots \to^c t'_n = (r, r)$. By Theorem 12 there exists a corresponding $\leadsto$-derivation ending with $\Gamma_n \vdash (s_n, t_n)$ such that: $\Delta' \vdash (s_n\eta_n, t_n\eta_n) \approx_\alpha t'_n = (r, r)$ and $\Delta' \vdash \Gamma_n\eta_n$. Thus, (Δ', η_n) is a solution of $\Gamma_n, s_n \approx_\alpha t_n$.

Since (Γ, μ) is the least unifier, it follows that $(\Gamma, \mu) \leq (\Delta', \eta_n)$ and: $\exists \xi : \forall X$, $\Delta' \vdash X\mu\xi \approx_\alpha X\eta_n$ and $\Delta' \vdash \Gamma\xi$. Therefore, by Theorem 12, $\Delta' \vdash (\theta_n\mu\xi)|_V \approx_\alpha \theta_n\eta_n|_V \approx_\alpha \eta|_V$ and $\Delta' \vdash_{\mathsf{E}} \eta|_V = \rho|_V$ that is, $(\Gamma, \theta_n\mu) \leq_{\mathsf{E}}^V (\Delta', \rho)$. ◀

Now we can describe how to build a complete set of E-unifiers for two terms-in-context.

▶ **Theorem 16.** *Let E be a closed nominal equational theory and R be an equivalent convergent nominal rewrite theory. Let $\Delta \vdash s$ and $\nabla \vdash t$ be two terms-in-context, and V be a finite set of variables containing $V(\Delta, s, \nabla, t)$. Let $\mathcal{S}$ be the set of pairs (Γ, σ) such that there exists a $\leadsto^c$-derivation: $\Gamma_0 \vdash u = (s,t) = u_0 \leadsto^c \ldots \leadsto^c \Gamma_n \vdash u_n = (s_n, t_n)$, where $(\Gamma_0 \equiv \Delta, \nabla)$, $\Gamma_n, s_n \approx_\alpha t_n$ has a least solution (Γ, μ), $\sigma \equiv \theta_n\mu$, and θ_n is the normalised composition of the narrowing substitutions. Then $\mathcal{S}$ is a complete set of E-unifiers of $\Delta \vdash s$ and $\nabla \vdash t$ away from V.*

Proof. Consequence of Lemmas 14 and 15. ◀

[6] Here we use the pair constructor as a term former.

$$
\begin{array}{lll}
y\#F & \vdash & \mathit{diff}(\mathit{lam}([y]F), X) \to 0 \\
 & \vdash & \mathit{diff}(\mathit{lam}([y]y), X) \to 1 \\
 & \vdash & \mathit{diff}(\mathit{lam}([y]\mathit{sin}(F)), X) \to \mathit{mult}(\mathit{cos}(\mathit{sub}([y]F, X)), \mathit{diff}(\mathit{lam}([y]F), X)) \\
 & \vdash & \mathit{diff}(\mathit{lam}([y]\mathit{plus}(F, G)), X) \to \mathit{plus}(\mathit{diff}(\mathit{lam}([y]F), X), \mathit{diff}(\mathit{lam}([y]G), X)) \\
 & \vdash & \mathit{diff}(\mathit{lam}([y]\mathit{mult}(F, G)), X) \to \mathit{plus}(\mathit{mult}(\mathit{diff}(\mathit{lam}([y]F), X), \mathit{sub}([y]G, X)), \\
 & & \qquad\qquad \mathit{mult}(\mathit{diff}(\mathit{lam}([y]G), X), \mathit{sub}([y]F, X)))
\end{array}
$$

Figure 4 Rewrite rules for symbolic differentiation.

An E-unification procedure follows from the construction of Theorem 16: enumerate all elements of $\mathcal{S}$. The set $\mathcal{S}$ may be infinite, one can organise the enumeration in such a way that if two nominal terms $\Delta \vdash s$ and $\nabla \vdash t$ are E-unifiable, then an E-solution will be produced in a finite number of steps. Thus, assuming E is presented by a convergent rewrite theory R, we have a semi-decision procedure for nominal E-unification.

3.1 An example: symbolic differentiation

The rewrite rules in Figure 2 define a λ-calculus with names and explicit substitutions [13]; the extension with numbers and operations (*plus*, *mult*, *sin*, *cos*) is straightforward.

Consider now symbolic differentiation [26]: $\mathit{diff}(F, X)$ computes the differential of a function F (meta-level unknown that can be instantiated by a λ-term) at a point X, using the rewrite rules given in Figure 4.

▶ **Example 17.** Let E be the theory defined by rewrite rules in Figures 2 and 4 together with standard rules for arithmetic operations. This system is closed but not convergent (we can simulate the untyped λ-calculus, which is non-terminating) so narrowing is not necessarily complete; however, we can still obtain the E-solution $(\emptyset, \{F \mapsto y\})$ for the nominal E-unification problem $\mathit{lam}([z]\mathit{diff}(\mathit{lam}([y]\mathit{sin}(F)), z))\ {}_?\overset{\mathsf{E}}{\approx}_?\ \mathit{lam}([z]\mathit{cos}(z))$ as follows[7].

The first closed-narrowing step uses a freshened rule
$\vdash \mathit{diff}(\mathit{lam}([y']\mathit{sin}(F')), X') \to \mathit{mult}(\mathit{cos}(\mathit{sub}([y']F', X')), \mathit{diff}(\mathit{lam}([y']F'), X'))$
with the assumption $y'\#F$ (below the narrowed subterm is in bold, the substitution used is $\{F' \mapsto (y\ y') \cdot F, X' \mapsto z\}$):

$$
\begin{array}{l}
\mathit{lam}([z]\mathit{diff}(\mathbf{lam}([\mathbf{y}]\mathbf{sin}(\mathbf{F})), \mathbf{z}))\ {}_?\approx_?\ \mathit{lam}([z]\mathit{cos}(z)) \\
\rightsquigarrow \mathit{lam}([z]\mathit{mult}(\mathit{cos}(\mathit{sub}([y'](y\ y') \cdot F, z)), \mathit{diff}(\mathit{lam}([y'](y\ y') \cdot F), z)))\ {}_?\approx_?\ \mathit{lam}([z]\mathit{cos}(z))
\end{array}
$$

We now use the freshened rule $\vdash \mathit{diff}(\mathit{lam}([w]w), W) \to 1$ with substitution $\{F \mapsto y, W \mapsto z\}$ and assumption $w\#F$ to narrow the second argument of *mult*:

$$
\rightsquigarrow \mathit{lam}([z]\mathit{mult}(\mathit{cos}(\mathit{sub}([y']y', z)), 1))\ {}_?\approx_?\ \mathit{lam}([z]\mathit{cos}(z))
$$

Using now the rules for *sub*, we can rewrite (hence also narrow) to

$$
\mathit{lam}([z]\mathit{mult}(\mathit{cos}(z), 1))\ {}_?\approx_?\ \mathit{lam}([z]\mathit{cos}(z))
$$

and by rewriting with the usual rules for multiplication, we obtain two equal terms.

[7] Here we do not rely on *Beta*, *diff* uses just the substitution rules, which are terminating.

$$
\begin{array}{llllll}
(1) & \emptyset & \vdash & \pi_i(\langle X_1, X_2\rangle) & \to X_i & (i \in \{1,2\}) \\
(2) & \emptyset & \vdash & d(\{X\}_Y, Y^{-1}) & \to X & \\
(3) & \emptyset & \vdash & d(\{X\}_{Y^{-1}}, Y) & \to X & \\
(4) & \emptyset & \vdash & (X^{-1})^{-1} & \to X & \\
(5) & \emptyset & \vdash & subst_j^n([\overrightarrow{z}]z_k, \overrightarrow{X}) & \to X_k & (1 \leq k \leq j) \\
(6) & z\#Y & \vdash & subst_1^n([z]Y, X) & \to Y & \\
(7) & z_k\#Y & \vdash & subst_j^n([\overrightarrow{z}]Y, \overrightarrow{X}) & \to \underline{subst_{j-1}^n([\overrightarrow{z'}]Y, \overrightarrow{X'})} & (1 \leq k \leq j, j > 1) \\
(8) & \emptyset & \vdash & subst_j^n([\overrightarrow{z}]f(\overrightarrow{W}), \overrightarrow{X}) & \to f(\overrightarrow{subst_j^n([\overrightarrow{z}]W, \overrightarrow{X})}) &
\end{array}
$$

Figure 5 Rewrite theory DYT.

3.2 Application: Intruder Deduction Problem

In this section we present an application of nominal E-matching.

▶ **Definition 18** (Nominal Intruder Deduction Problem)**.** Given a finite set of ground messages in normal form $\Gamma = \{t_1, \ldots, t_n\}$, a ground message in normal form m (the secret), and private names $a_1, \ldots, a_k$, we model the Intruder Deduction Problem (IDP) as a nominal E-matching problem with one unknown: $(\Delta \vdash subst([\overrightarrow{z}]X, \overrightarrow{t}))\ {}_?\overset{\mathsf{E}}{\approx}\ m$. Here m is short for $\emptyset \vdash m$ and $\Delta = \{a_1\#X, \ldots, a_k\#X\}$ is a freshness context specifying that the names $a_1, \ldots, a_n$ are fresh in the unknown term X, *subst* is a term-former denoting the substitution of $z_1, \ldots, z_n$ (denoted by $\overrightarrow{z}$) by $t_1, \ldots, t_n$ (denoted by $\overrightarrow{t}$), $\overrightarrow{z}$ are abstracted in X, and $\overrightarrow{t}$ represent the messages in Γ.

To illustrate the results we consider a simple equational theory, namely the *Axiomatised Dolev-Yao Theory* (DYT). It is essentially the classical Dolev-Yao model with explicit destructors such as decryption and projections. It is well-known that IDP for this theory is decidable in polynomial time[8], the purpose here is to show how nominal narrowing could be used to solve this security problem.

The signature for DYT, Σ_{DYT}, includes function symbols $\langle\_,\_\rangle, \pi_1(\_)$, $\pi_2(\_), d(\_,\_)$, $\{\_\}_\_, (\_)^{-1}$ for *pairing*, *projections*, *decryption*, *encryption* and *inverse*, respectively, as well as a family of symbols $subst_j^n$ ($n \geq 1, j \in \{1, \ldots, n\}$) to perform substitution. Intuitively, projections are inverses of pairing and decrypting with k^{-1} a message encrypted with k gives back the plaintext.

The rewrite rules are given, in a schematic way, in Figure 5. The index j in $subst_j^n$ denotes the number of abstracted atoms in $[\overrightarrow{z}]$, for $j \in \{1, \ldots, n\}$. In rule schemes (5) and (7), z_k is a term in $\{z_1, \ldots, z_j\}$ and there is a rule for each k s.t. $1 \leq k \leq j$. In rule scheme (7), $j > 1$; in case $j = 1$ we use rule (6). In rules (7) and (8) we use the following abbreviations:

- $[\overrightarrow{z}] = [z_1, \ldots, z_j]$ and $[\overrightarrow{z'}] = [z_1 \ldots, z_{k-1}, z_{k+1}, \ldots, z_j]$;
- $\overrightarrow{X} = (X_1, \ldots, X_j)$ and $\overrightarrow{X'} = (X_1, \ldots, X_{k-1}, X_{k+1}, \ldots, X_{j-1})$;
- $f \in \Sigma_{\mathsf{DYT}}$ is an r-ary function symbol (there is a version of rule (8) for each $f \neq subst$), and $f(\overrightarrow{subst_j^n([\overrightarrow{z}]W, \overrightarrow{X})}) = f(subst_j^n([\overrightarrow{z}]W_1, \overrightarrow{X}), \ldots, subst_j^n([\overrightarrow{z}]W_r, \overrightarrow{X}))$.

▶ **Proposition 19.** DYT *is a closed and convergent nominal rewrite system.*

Proof. The termination is obtained by a simplification ordering. It is convergent because the critical pairs obtained are joinable [11]. ◀

[8] This result was obtained using another approach [7]

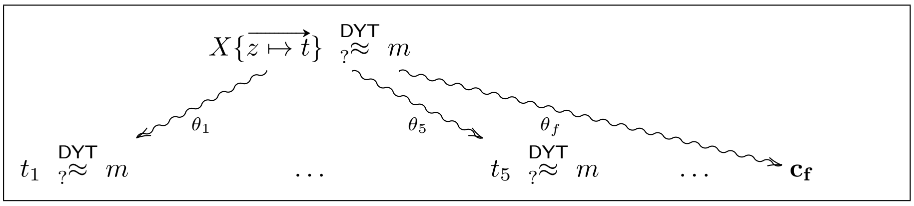


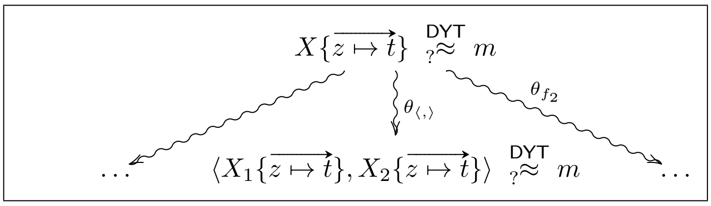
Figure 6 First level of the narrowing tree.

▶ Remark. Below, the notation $t\{z \mapsto t'\}$ is syntactic sugar for $subst([z]t, t')$.

▶ **Example 20.** Consider $\Gamma = \{\underbrace{\{\{m\}_b\}_c}_{t_1}, \underbrace{\{b^{-1}\}_k}_{t_2}, \underbrace{\{c^{-1}\}_r}_{t_3}, \underbrace{k^{-1}}_{t_4}, \underbrace{r^{-1}}_{t_5}\}$ and a secret m (a constant). Taking into account the theory DYT, this IDP can be stated as $X\{\overrightarrow{z \mapsto t}\} \;{}_?{\overset{\mathsf{DYT}}{\approx}}\; m$, where $\{\overrightarrow{z \mapsto t}\}$ denotes the substitution of t_i for z_i, $i = 1, \ldots, 5$. Figure 6 shows part of the first level of the narrowing tree for this problem.

The substitutions θ_i are $\{X \mapsto z_i\}$, $i = 1, \ldots, 5$ and the corresponding narrowing steps use rule (5). The result $t_i \;{}_?{\overset{\mathsf{DYT}}{\approx}}\; m$ is a ground problem, which can be decided by checking syntactic equality since each t_i and m are in normal form. The branch labelled with the substitution θ_f is an abbreviation for six branches, namely, one for each $f \in \Sigma_{\mathsf{DYT}}$ (except *subst*).

To illustrate, consider the case in which f is a constructor, for instance, $f = \langle\,,\,\rangle$:



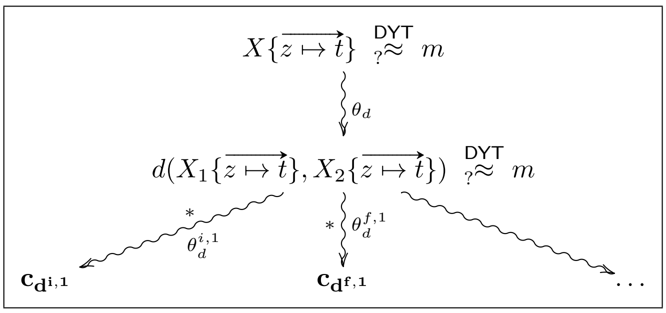
This branch is obtained via

- a version of rule (8): $\emptyset \vdash subst^5_j([\vec{w}]\langle W_1, W_2\rangle, \vec{Z}) \to \langle subst^5_j([\vec{w}]W_1, \vec{Z}), subst^5_j([\vec{w}]W_2, \vec{Z})\rangle$
- and substitution $\theta_{\langle,\rangle} = \{X \mapsto \langle X_1, X_2\rangle, W_1 \mapsto (w\ z)\cdot X_1, W_2 \mapsto (w\ z)\cdot X_2, \vec{Z} \mapsto \vec{T}\}$ with the assumption $w\#X$.

Consider the case in which f is a destructor, for instance, $f = d$. There is a narrowing step:

$$X\{\overrightarrow{z \mapsto t}\} \;{}_?{\overset{\mathsf{DYT}}{\approx}}\; m \rightsquigarrow_{\theta_d} d(X_1\{\overrightarrow{z \mapsto t}\}, X_2\{\overrightarrow{z \mapsto t}\}) \;{}_?{\overset{\mathsf{DYT}}{\approx}}\; m$$

obtained via substitution $\theta_d = \{X \mapsto d(X_1, X_2)\}$ and rule (8). From this node we can narrow with $\theta^{i,1}_d = \{X_1 \mapsto z_i\}$, or $\theta^{i,2}_d = \{X_2 \mapsto z_i\}$ $(i = 1, \ldots, 5)$, or $\theta^{f,1}_d = \{X_1 \mapsto f(X'_1)\}$ or $\theta^{f,2}_d = \{X_2 \mapsto f(X'_2)\}$ $(f \in \Sigma_{\mathsf{DYT}})$:



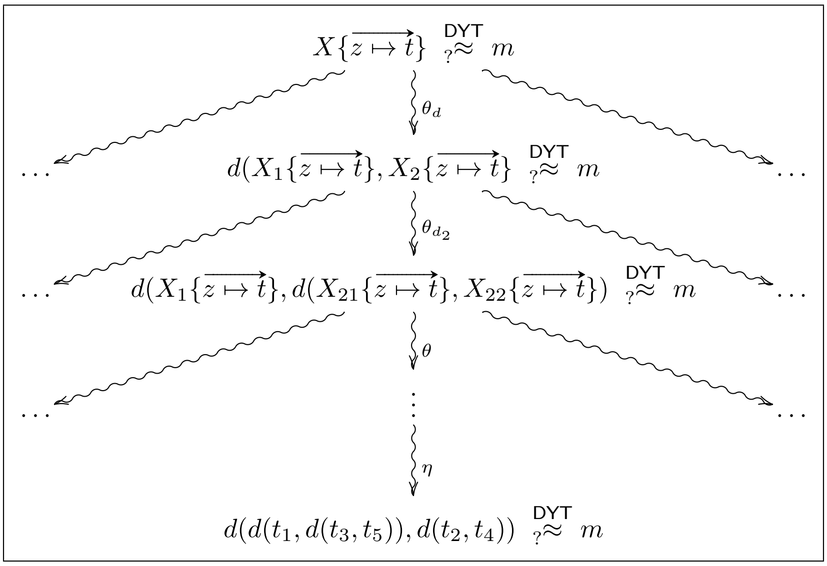


Figure 7 Narrowing Subtree for the Solution.

The left branch represents 5 narrowing branches, one for each i. After applying rule (5) one has $\mathbf{c}_{\mathbf{d}^{\mathbf{i}},\mathbf{1}} := d(t_i, X_2) \ {}_{?}\!\overset{\mathsf{DYT}}{\approx}\ m$. Similarly, $\mathbf{c}_{\mathbf{d}^{\mathbf{f}},\mathbf{1}}$ represents 6 other possible branches, one for each function symbol from Σ_{DYT}. Iterating this reasoning, we obtain the narrowing branch shown in Figure 7, which leads to a ground problem whose solution is positive.

The previous example illustrates the fact that a series of narrowing steps might be necessary in order to obtain a solution. Variables might need to be instantiated with constructors for two reasons:

- either the term m contains a sequence of constructors in its structure, therefore, the variables in the term being matched have to be instantiated with the same sequence of constructors, and rule (8) applies;
- or a sequence of constructors matches a sequence of the corresponding destructors in a term in Γ, enabling a rewriting rule to be applied.

As a consequence, the number of applications of DYT rules is bounded by $|\Gamma| + |m|$.

▶ **Theorem 21.** *If a narrowing derivation* $(\Delta_0 \vdash (subst([\vec{z}]X, \vec{t}), m) \rightsquigarrow_{\sigma_0} \dots \rightsquigarrow_{\sigma_{k-1}} (\Delta_k \vdash u_k)$ *has more than* $|\Gamma| + |m|$ *narrowing steps then* $height(subst([\vec{z}]X, \vec{t})\sigma_0\sigma_1 \dots \sigma_{k-1}) > height(m)$*. Therefore, it does not lead to a solution.*

Proof. Each application of a Dolev-Yao rule eliminates one symbol from the term. In the worst case, in all terms from Γ all the function symbols can be eliminated by a rule, before several steps of composition (with a constructor that has not just been eliminated) can be applied until one reaches the size of m.

Notice that for infinite branches of the form $\Pi_i\Pi_j\Pi_i\Pi_j \dots$ or $dddd\dots$ either the term m would have to be headed with the same sequence of functions or rewrite rules would be applied. By the Lifting Theorem, we can assume the compositions of substitutions are normalised, therefore, the only way to apply rewrite rules is when the terms in Γ contain, in the first case, a sequence of pairings $\langle\langle\langle \dots \rangle\rangle\rangle$ or, in the second case, a sequence of encryptions $\{\{\dots\}\}$. We cannot introduce a destructor followed by its corresponding constructor with a substitution, e.g, $\Pi_i\Pi_j\Pi_i\langle\langle\dots\rangle\rangle$, otherwise the substitution would not be normalised. Since all the terms in Γ are finite, only a finite number of destructive rewrite rules could be applied and the number of constructive rewrite rules that could be applied is bounded by the size of m. The same reasoning applies when we have interleaving of destructors $d\Pi_i d\Pi_j d\Pi_i d\Pi_j \dots$ or even constructors and destructors of the form $d\{\}d\{\}d\{\}$, when the encryption/decryption keys do not correspond. ◀

As a consequence, we obtain the decidability of the nominal IDP for DYT.

4 Basic Nominal Narrowing

Hullot [17] introduced *basic narrowing* to eliminate redundant narrowing derivations in order to give sufficient conditions for the termination of the narrowing process. Following [17], with the corrections made in [1, 24], we define *basic (closed) nominal narrowing*. In the rest of this section, $\mathsf{R} = \{R_k \equiv \nabla \vdash l_k \to r_k\}$ is a closed nominal rewrite theory.

▶ **Definition 22.** Consider a nominal term s and a set U of positions that are proper prefixes of s, that is, $U = \overline{\mathcal{P}os}(r)$, for some subterm r of s. We define by induction what it means for a nominal rewriting derivation $\Delta \vdash s = s_0 \to_{[C_0,R_0]} s_1 \to_{[C_1,R_1]} \to \dots \to_{[C_{n-1},R_{n-1}]} s_n$ to be *based on* U and construct sets of positions $U_i \subset \mathcal{P}os(s_i)$, $0 \le i \le n$, inductively: the empty derivation is based on U, and $U_0 = U$; if a derivation up to s_i is based on U, then the derivation obtained from it by adding one step $s_i \to_{[C_i,R_i]} s_{i+1}$ is based on U iff $C_i \in U_i$, and in this case we take: $U_{i+1} = (U_i - \{C \in U_i | C_i \le C\}) \cup \{C_i.C | C \in \overline{\mathcal{P}os}(r_i)\}$, where r_i denotes the right-hand side of the rule R_i in R[9].

A nominal rewrite step $\Delta \vdash C[s] \to C[s']$ at position C is *innermost* if for any C_i such that $C < C_i$ and $C[s] = C_i[s_i]$, there is no rewrite step $\Delta \vdash C_i[s_i] \to C_i[t]$ at position C_i. In other words, there is no rewrite step inside s. An innermost nominal rewrite derivation contains only innermost rewrite steps.

▶ **Lemma 23.** *Let $\Delta \vdash s \approx_\alpha s_0\eta$, with η normalised in Δ. Every innermost nominal rewrite derivation from $\Delta \vdash s$ is based on $\overline{\mathcal{P}os}(s_0)$.*

▶ **Definition 24.** A nominal narrowing derivation $(\Delta_0 \vdash s_0) \rightsquigarrow_{[C_0,R_0,\sigma_0]} \cdots \rightsquigarrow_{[C_{i-1},R_{i-1},\sigma_{i-1}]} (\Delta_i \vdash s_i)$, is *basic* if it is based on $\overline{\mathcal{P}os(s_0)}$ (in the same sense as in the previous definition for nominal rewriting derivation).

▶ **Theorem 25.** *The narrowing derivations constructed in Theorem 12 are all basic.*

Proof. Let $(\Delta_0 \vdash s_0) \rightsquigarrow_{[C'_0,R_0,\sigma_0]} \cdots \rightsquigarrow_{[C'_{n-1},R_{n-1},\sigma_{n-1}]} (\Delta_n \vdash s_n)$ be the nominal narrowing derivation associated by Theorem 12 with $\Delta \vdash s_0\eta = t_0 \to_{[C_0,R_0]} \cdots \to_{[C_{n-1},R_{n-1}]} t_n$, such that η is normalised. Since R is confluent we may assume that the nominal rewriting sequence from $\Delta \vdash s_0\eta$ is innermost. By Lemma 23, this nominal rewriting derivation is based on $\overline{\mathcal{P}os}(s_0)$, and since the sets U_i in the two derivations are equivalent, it follows that the considered nominal narrowing derivation is basic. ◀

▶ Remark. Definition 22, Lemma 23 can also be stated for closed narrowing. Theorem 16 holds also for closed basic narrowing.

The main interest of closed basic narrowing is that we can give a sufficient condition for the termination of the narrowing process when we consider only basic $\rightsquigarrow$-derivations and therefore for the termination of the corresponding nominal E-unification procedure.

[9] Given $C_i = (s_i, \_)$ and $C = (s, \_)$, it follows $C_i.C = (s_i\{\_ \mapsto s\}, \_)$ and $C_i \le C$ if $\exists t : s_i\{\_ \mapsto t\} = s$.

▶ **Proposition 26.** *Let* $\mathsf{R} = \{\nabla_k \vdash l_k \to r_k\}$ *be a convergent nominal rewriting system such that any basic* $\rightsquigarrow$*-derivation issuing from any of the right-hand sides* r_k *terminates. Then any basic* $\rightsquigarrow$*-derivation issuing from any nominal term terminates.*

The previous proposition also holds for basic closed narrowing.

▶ **Theorem 27.** *Basic closed nominal narrowing is complete for convergent closed nominal rewriting systems.*

Moreover, if R satisfies the hypothesis of Proposition 26, nominal basic narrowing leads to a complete and finite E-unification algorithm.

5 Conclusion and Future Work

We have introduced the *nominal narrowing* relation and designed a general nominal E-unification procedure, which is complete for a wide class of theories, namely, the theories defined by convergent closed nominal rewriting systems.

There is a lot of work to be done regarding nominal E-unification. A first step would be to study the relationship between nominal narrowing and pattern narrowing [26]. For the analysis of protocols, it would be interesting to study nominal unification modulo equational theories including associativity and commutativity axioms. From a practical point of view, narrowing strategies should be studied, such as lazy narrowing for nominal terms, and also more general versions of nominal narrowing such as conditional [26] and variant [10] narrowing, which have interesting applications [21, 23]. We would like to define conditions for termination of nominal narrowing similar to the *finite variant* and *boundedness* properties [6], to obtain an alternative way to study the security of protocols, via nominal narrowing.

Acknowledgements. We thank Santiago Escobar, Jesus Dominguez Alvarez and the FSCD reviewers for their valuable comments, which helped us improve the paper.

References

1 M. Alpuente, S. Escobar and J. Iborra. *Termination of Narrowing Revisited.* In *Theoretical Computer Science*, 410(46):4608–4625, 2009.

2 F. Baader and T. Nipkow. *Term Rewriting and All That.* Cambr. Univ. Press, 1998.

3 C. Calvès and M. Fernández. *The First-Order Nominal Link. Logic-Based Program Synthesis and Transformation - 20th International Symposium (LOPSTR 2010), Hagenberg, Austria, 2010, Revised Selected Papers*, vol. 6564 of *LNCS*, 234–248, 2011.

4 C. Calvès and M. Fernández. *Matching and alpha-equivalence check for nominal terms. Journal of Computer and System Sciences*, 76(5):283–301, 2009.

5 J. Cheney and C. Urban. *Nominal Logic Programming*, *ACM Trans. Program. Lang. Syst.* 30(5), Article 26, 2008.

6 H. Comon-Lundh and S. Delaune. *The finite Variant Property: How to get rid of some algebraic properties.* In *Proc. of Rewriting Techniques and Applications (RTA'05)*, vol. 3467 of *LNCS*, 294–307, 2005.

7 S. Delaune and F. Jacquemard. *A Decision Procedure for the Verification of Security Protocols with Explicit Destructors.* In *Proc. of 11th ACM Conference on Computer and Communications Security (CCS'04)*, pages 278–287, 2004.

8 N. Dershowitz and D.A. Plaisted. *Equational Programming. Machine Intelligence 11*, Clarendon Press, Oxford, Ch. 2, 21–56, 1988.

9 S. Escobar, J. Meseguer and R. Sasse. *Variant Narrowing and Equational Unification*. *Electr. Notes Theor. Comput. Sci.,* 238(3):103–119, 2009.

10 S. Escobar, R. Sasse and J. Meseguer *Folding Variant Narrowing and Optimal Variant Termination.* In *Proc. 8th International Workshop Rewriting Logic and Its Applications (WRLA 2010)*, vol. 6381 of *LNCS*, 52–68, 2010.

11 M. Fernández and M. J. Gabbay. Nominal rewriting. *Information and Computation*, 205(6):917–965, 2007.

12 M. Fernández and M. J. Gabbay. Closed Nominal Rewriting and Efficiently Computable Nominal Algebra Equality In *Proc. of 5th International Workshop on Logical Frameworks and Meta-languages: Theory and Practice (LFMTP)*, 37–51, 2010.

13 M. Fernández, M. J. Gabbay, and I. Mackie. *Nominal Rewriting Systems.* In *Proc. 6th Int.Conf. on Principles and Practice of Declarative Programming (PPDP 2004)*, 2004.

14 M. Fernández and A. Rubio. *Nominal Completion for Rewrite Systems with Binders.* In *Proc. of 39th International Colloquium Automata, Languages, and Programming (ICALP' 2012)* , vol. 7392 of *LNCS*, 201–213, 2012.

15 M. J. Gabbay and A. Mathijssen. *Nominal universal algebra: equational logic with names and binding. Journal of Logic and Computation*, 19(6):1455–1508, 2009.

16 M. J. Gabbay and A. M. Pitts. *A New Approach to Abstract Syntax with Variable Binding*. *Formal Aspects of Computing*, 13(3–5):341–363, 2001.

17 J-M. Hullot. *Canonical Forms and Unification* In *Proc. 5th Conference on Automated Deduction (CADE'80)*, vol. 87 of *LNCS*, 318–324, 1980.

18 J. Levy and M. Villaret. *Nominal unification from a higher-order perspective*. In Proc. Rewriting Techniques and Applications (RTA'08), vol. 5117 in *LNCS*, 2008.

19 J. Levy and M. Villaret. *An efficient nominal unification algorithm*. In Proc. of Rewriting Techniques and Applications (RTA'10), vol. 6 of *LIPIcs*, 209–226, 2010.

20 J. W. Lloyd. *Foundations of Logic Programming*. Springer-Verlag, 1987.

21 C. Meadows. *Using Narrowing in the Analysis of Key Management Protocols*. In Proc. (IEEE) Symposium on Security and Privacy, 138–147, 1989.

22 J. Meseguer. *Strict Coherence of Conditional Rewriting Modulo Axioms*. Technical Report, Computer Science Department, University of Illinois at Urbana-Champaign, August 2014. Available from `http://hdl.handle.net/2142/50288`

23 J. Meseguer and P. Thati. *Symbolic Reachability Analysis Using Narrowing and its Application to Verification of Cryptographic Protocols*. In Higher-Order and Symbolic Computation, 20(1):123–160, 2007.

24 A. Middeldorp and E. Hamoen. *Completeness Results for Basic Narrowing*. Applicable Algebra in Engineering, Communication and Computing, 5(3-4):213–253, 1994.

25 D. Miller *A Logic Programming Language with Lambda-Abstraction, Function Variables and Simple Unification*. Journal of Logic and Computation, 1(4):497–536, 1991.

26 C. Prehofer. *Solving Higher-Order Equations from Logic to Programming*. In Progress in Theoretical Computer Science, Birkhäuser, 1997.

27 T. Sheard. *Type-Level Computation Using Narrowing in Ωmega.* In Proc. of Programming Languages meets Program Verification (PLPV), *Electr. Notes Theor. Comput. Sci.,* 174(7):105–128, 2006.

28 M. R. Shinwell, A. M. Pitts and M. J. Gabbay. *FreshML: Programming with Binders Made Simple. SIGPLAN Notices* 38(9):263–274, 2003.

29 C. Urban, A. M. Pitts and M. J. Gabbay. *Nominal Unification.* Theoretical Computer Science, 323(1–3):473–497, 2004.