Permissive nominal terms and their unification

Gilles Dowek Murdoch J. Gabbay Dominic Mulligan

Abstract. We introduce permissive nominal terms. Nominal terms extend first-order terms with binding. They lack properties of first- and higher-order terms: Terms must be reasoned on in a context of 'freshness assumptions'; it is not always possible to 'choose a fresh variable symbol' for a nominal term; and it is not always possible to 'alpha-convert a bound variable symbol'. Permissive nominal terms recover these 'always fresh' and 'always alpha-rename' properties. Freshness contexts are elided and the notion of unifier is based on substitution alone rather than on nominal terms' notion of substitution plus freshness conditions. We prove that all expressivity of nominal unification is retained.

1 Introduction

Many formal languages feature *variable binding*: examples include λ-abstraction, quantification, and sets comprehension $\{x \mid \phi(x)\}$. Binding is ubiquitous, because variables are there to be bound or substituted.

Variables cannot be bound in first-order terms (constructed from variables and term-formers). This motivates extensions of first-order syntax; logics where variables can be bound by function or predicate symbols [DHK02], rewriting on terms with binders [KvOvR93,MN98,FG07], and languages with datatypes with binders [PE88,Mil91,SPG03,HHP87,Pau90,CU04].

We will study nominal terms [UPG04]. These preserve the flavour of first-order terms and extend them to represent informal statements like

> "If $y \notin fv(t)$ then $\lambda x.t$ is α-equivalent with $\lambda y.[y/x]t$" and
> "How can we choose t and u to make $\lambda x.\lambda y.(y\ t)$ equal to $\lambda x.\lambda x.(x\ u)$?".

(We write $fv(t)$ for the free variables of t; e.g. $fv(\lambda x.(xy)) = \{y\}$.)

Nominal terms use a characteristic combination of features: two levels of variable (*atoms* and *unknowns*), *freshness* conditions, *permutations*, and a distinctive theory of α-equivalence (see Section 6 or [UPG04] for details). The first statement above is rendered in nominal terms as the equality judgement $b\#X \vdash [a]X = [b](b\ a) \cdot X$. Here a and b denote atoms, which represent the 'x' and 'y'; X denotes an unknown, it represents the 't'; $b\#X$ is a *freshness side-condition*, it represents the '$y \notin fv(t)$'; $(b\ a)$ is a *permutation* meaning 'map a to b and b to a', it represents the '$[y/x]$' (we assumed $y \notin fv(t)$, so this is possible).

Yet nominal terms from [UPG04] have some less attractive properties too:

– Freshness contexts are not fixed so we must often prove properties of *terms-in-freshness context*. This is harder than reasoning just about terms.

– We cannot always pick a fresh variable symbol and α-rename a bound variable. For X in the empty freshness context, there is — by definition of the empty freshness context — no a such that $a\#X$.[1]

We propose *permissive nominal terms*. An unknown takes the form X^S where S is a permission sort (Definition 3). Permission sorts are sets of atoms that are both infinite and co-infinite (see Definition 2; $S \subseteq X$ is co-finite when $X \setminus S$ is infinite). Thus, we can always choose a fresh atom, always α-convert (Definition 8, Lemma 12), α-equivalence is independent of a freshness context (there is no freshness context), and the notion of unifier is based just on substitution rather than (as for nominal terms) a freshness context and a substitution.

We illustrate this with two examples; definitions are in the paper. *comb* is an infinite, co-infinite set of atoms. Suppose $a, b \in \mathit{comb}$. Assume a binary term-former (function-symbol) for application. Then:

– α-equivalence is 'if $y \notin \mathit{fv}(u)$ then $\lambda x.u = \lambda y.(u[x/x])$'.
In nominal terms this becomes $b\#Z \vdash [a]Z = [b](b\ a)\cdot Z$.[2] In permissive nominal terms it becomes $[a]Z^{\mathit{comb}\setminus\{b\}} = [b](b\ a)\cdot Z^{\mathit{comb}\setminus\{b\}}$.
It is not possible to α-convert a in the nominal term $\varnothing \vdash [a]Z$; to do this, we must enrich the freshness context $\varnothing$.
In permissive nominal terms, it is always possible to α-convert a in $[a]Z^{\mathit{comb}\setminus\{b\}}$; we can convert to any of the atoms not in $\mathit{comb} \setminus \{b\}$.

– 'Unify $\lambda x.\lambda y.(yt)$ with $\lambda x.\lambda x.(xu)$' becomes in nominal terms '$[a][b](bX) \stackrel{?}{=} [a][a](a\ Y)$' and has solution $(b\#Y,\ [X{:=}(b\ a)\cdot Z,\ Y{:=}Z])$.
In permissive nominal terms it becomes '$[a][b](bX^{\mathit{comb}}) \stackrel{?}{=} [a][a](aY^{\mathit{comb}})$' and has solution $[X^{\mathit{comb}}{:=}Z^{\mathit{comb}\setminus\{b\}},\ Y^{\mathit{comb}}{:=}Z^{\mathit{comb}\setminus\{b\}}]$.

Unification is the basis of rewriting and logic programming. Nominal terms have served as the basis of both [FG07,CU04]. Therefore, and following that work, in this paper we explore permissive nominal unification.

2 Permissive nominal terms

Definition 1 Fix a countably infinite set $\mathbb{A}$ of **atoms**. We use a **permutative convention** that $a, b, c, \ldots$ will range over *distinct* atoms. Fix a set of **term-formers**. $\mathsf{f}, \mathsf{g}, \mathsf{h}$ will range over distinct term-formers.

Examples of term-formers are symbols like lam, app, and forall. In the presence of rewrites or equational axioms we can make these term-formers 'do' something; in this paper, they are just ways of building terms with suggestive names.

[1] 'Freshness contexts' sounds like 'typing contexts', but the terminology misleads. Extending a typing context makes more terms typable and, intuitively, extends the universe of discourse with extra typable terms. Extending a freshness context makes more terms *equal*. This is a more drastic event because it is more likely to change the behaviour of existing elements; they may now be equal to other elements with quite different behaviour.

[2] In the body of the paper we put dots on atoms and unknowns in nominal terms, to emphasise the difference from permissive nominal terms. Here, we do not bother.

Definition 2 Call $S \subseteq \mathbb{A}$ co-infinite when $\mathbb{A} \setminus S$ is infinite. Fix an infinite, co-infinite set $comb \subseteq \mathbb{A}$. A **permission sort** has the form $(comb \cup A_1) \setminus A_2$ for finite sets $A_1 \subseteq \mathbb{A}$ and $A_2 \subseteq \mathbb{A}$. S, S', T will range over permission sorts.

If S and T are permission sorts, so are $S \cup T$ and $S \cap T$, and both S and $\mathbb{A} \setminus S$ are infinite.

Definition 3 For each S fix a disjoint countably infinite set of **unknowns** of sort S. X^S, Y^S, Z^S, will range over distinct unknowns of sort S. If $S \neq S'$ then there is no particular connection between X^S and $X^{S'}$. $\mathcal{V}$ will range over finite sets of unknowns (we use this from Section 4 onwards).

Definition 4 A **permutation** is a bijection on atoms such that $\{a \mid \pi(a) \neq a\}$ is finite. π and π' will range over permutations (not necessarily distinct).

Write id for the **identity** permutation such that $id(a) = a$ always. Write $(a\ b)$ for the **swapping** which maps a to b, b to a, and all other c to themselves.

Definition 5 Define **(permissive nominal) terms** by:

$$r, s, t, \ldots ::= a \mid \mathsf{f}(r_1, \ldots, r_n) \mid [a]r \mid \pi \cdot X^S$$

We write $\equiv$ for syntactic identity; $r \equiv s$ when r and s denote identical terms.

Definition 6 Define a **permutation action** by:

$$\pi{\cdot}a{\equiv}\pi(a)\ \ \pi{\cdot}(\mathsf{f}(r_1,\ldots,r_n)){\equiv}\mathsf{f}(\pi{\cdot}r_1,\ldots,\pi{\cdot}r_n)\ \ \pi{\cdot}[a]r{\equiv}[\pi(a)](\pi{\cdot}r)\ \ \pi{\cdot}(\pi'{\cdot}X^S){\equiv}(\pi{\circ}\pi'){\cdot}X^S$$

Definition 7 If $S \subseteq \mathbb{A}$, define the **pointwise** action by: $\pi \cdot S = \{\pi(a) \mid a \in S\}$. Define **free atoms** $fa(r)$ and **unknowns** $fV(r)$ by:

$$fa(a){=}\{a\}\quad fa(\mathsf{f}(r_1,\ldots,r_n)) = fa(r_1) \cup \ldots \cup fa(r_n)\quad fa([a]r){=}fa(r){\setminus}\{a\}\quad fa(\pi{\cdot}X^S){=}\pi{\cdot}S$$

$$fV(a){=}\varnothing\quad fV(\mathsf{f}(r_1,\ldots,r_n)) = fV(r_1) \cup \ldots \cup fV(r_n)\quad fV([a]r){=}fV(r)\quad fV(\pi{\cdot}X^S){=}\{X^S\}$$

An intuition for $fa(r)$ is 'possible free atoms after instantiation'.

Definition 8 Let $\pi|_S$ denote the partial function 'π restricted to S'. Define **α-equivalence** $=_\alpha$ inductively by:

$$\frac{}{a =_\alpha a}\ (=_\alpha\mathbf{aa}) \qquad \frac{r_1 =_\alpha s_1 \quad \cdots \quad r_n =_\alpha s_n}{\mathsf{f}(r_1,\ldots,r_n) =_\alpha \mathsf{f}(s_1,\ldots,s_n)}\ (=_\alpha\mathsf{f}) \qquad \frac{r =_\alpha s}{[a]r =_\alpha [a]s}\ (=_\alpha[\mathbf{a}])$$

$$\frac{(b\ a)\cdot r =_\alpha s \qquad (b \notin fa(r))}{[a]r =_\alpha [b]s}\ (=_\alpha[\mathbf{b}]) \qquad \frac{(\pi|_S = \pi'|_S)}{\pi \cdot X^S =_\alpha \pi' \cdot X^S}\ (=_\alpha\mathbf{X})$$

Lemma 9 *$id \cdot r \equiv r$ and $\pi' \cdot (\pi \cdot r) \equiv (\pi' \circ \pi) \cdot r$.*

Lemma 10 *$\pi \cdot fa(r) = fa(\pi \cdot r)$.*

Theorem 11 *$=_\alpha$ is transitive, reflexive, and symmetric.*

Lemma 12 *For any r, there exist infinitely many b such that $b \notin fa(r)$. As a corollary, for any r and a there exists infinitely many fresh b (so $b \notin fa(r)$) such that for some s, $[a]r =_\alpha [b]s$.*

Proof. The first part is by an easy induction on r. We consider only the case $r \equiv \pi \cdot X^S$: $fa(\pi \cdot X^S) = \pi \cdot S$, and infinitely many b are such that $b \notin \pi \cdot S$. The corollary follows using $(=_\alpha[\mathbf{b}])$.

3 Substitutions

Definition 13 A **substitution** θ is a function from unknowns to terms such that $fa(\theta(X^S)) \subseteq S$ always (so S in X^S describes the 'permission' we have to instantiate X^S, namely to terms with free atoms in S). θ, θ', θ_1, θ_2, will range over substitutions.[3]

Write id for the **identity** substitution mapping X^S to $id \cdot X^S$ always. It will always be clear whether id means the identity substitution or permutation.

Suppose $fa(t) \subseteq S$. Write $[X^S{:=}t]$ for the substitution such that $[X^S{:=}t](X^S) \equiv t$ and $[X^S{:=}t](Y^T) \equiv id \cdot Y^T$ for all other Y^T.

Definition 14 Define a **substitution action** of substitutions on terms by:

$$a\theta{\equiv}a \quad \mathsf{f}(r_1,\ldots,r_n)\theta{\equiv}\mathsf{f}(r_1\theta,\ldots,r_n\theta) \quad ([a]r)\theta{\equiv}[a](r\theta) \quad (\pi{\cdot}X^S)\theta{\equiv}\pi{\cdot}\theta(X^S)$$

Theorem 15 $fa(r\theta) \subseteq fa(r)$.

Proof. By induction on r. We consider the case $r \equiv \pi \cdot X^S$. By definition $fa(\pi \cdot X^S) = \pi \cdot S$. By assumption in Definition 13, $fa(\theta(X^S)) \subseteq S$. Using Lemma 10, it follows that $fa(\pi \cdot \theta(X^S)) \subseteq \pi \cdot S$.

Lemma 16 $\pi \cdot (r\theta) \equiv (\pi \cdot r)\theta$.

Proof. By a routine induction on r using the definitions and Lemma 9.

Theorem 17 *If* $\theta(X^S) =_\alpha \theta'(X^S)$ *for all* $X^S \in fV(r)$, *then* $r\theta =_\alpha r\theta'$.

Definition 18 Define **composition** $\theta{\circ}\theta'$ by $(\theta{\circ}\theta')(X^S) \equiv (\theta(X^S))\theta'$.

Theorem 19 $(r\theta)\theta' \equiv r(\theta{\circ}\theta')$.

Proof. By induction on r. We consider the case of $\pi \cdot X^S$, and use Lemma 16: $(\pi \cdot X^S)(\theta{\circ}\theta') \equiv \pi \cdot (\theta{\circ}\theta')(X^S) \equiv \pi \cdot (\theta(X^S)\theta') \equiv (\pi \cdot \theta(X^S))\theta' \equiv ((\pi \cdot X^S)\theta)\theta'$

4 Support inclusion problems

Recall from Definition 13 that $fa(\theta(X^S)) \subseteq fa(X^S) = S$, and from Theorem 15 that instantiation must reduce the set of free atoms. We will exhibit an algorithm which, intuitively, solves the problem "please make $fa(r\theta) \subseteq T$ true" (Definition 27 and Lemma 25). In fact the algorithm calculates solutions that are most general, in a sense made formal in Theorem 32.

Definition 20 A **support inclusion** is a pair $r \sqsubseteq T$ of a term and a permission sort. A **support inclusion problem** is a finite multiset of support inclusions; Inc will range over support inclusion problems. Call θ a **solution** to Inc when $fa(r\theta) \subseteq T$ for every $r \sqsubseteq T \in Inc$. Write $Sol(Inc)$ for the solutions of Inc.

[3] '$fa(\theta(X^S)) \subseteq S$' looks absent in nominal terms theory ([UPG04, Definition 2.13], [FG07, Definition 4]), yet it is there: see the conditions '$\nabla' \vdash \theta(\nabla)$' in Lemma 2.14, and '$\nabla \vdash a\#\theta(t)$' in Definition 3.1 of [UPG04]. More on this in Section 6.

Definition 21 Define a **simplification** rewrite relation by:

$$
\begin{array}{lrll}
(\sqsubseteq\mathbf{a}) & a \sqsubseteq T, Inc & \Longrightarrow Inc & (a \in T) \\
(\sqsubseteq\mathsf{f}) & \mathsf{f}(r_1,\ldots,r_n) \sqsubseteq T, Inc & \Longrightarrow r_1 \sqsubseteq T, \ldots, r_n \sqsubseteq T, Inc & \\
(\sqsubseteq[]) & [a]r \sqsubseteq T, Inc & \Longrightarrow r \sqsubseteq T \cup \{a\},\ Inc & \\
(\sqsubseteq\mathbf{X}) & \pi \cdot X^S \sqsubseteq T, Inc & \Longrightarrow id \cdot X^S \sqsubseteq \pi^{-1} \cdot T,\ Inc & (S \not\subseteq \pi^{-1} \cdot T,\ \pi \neq id) \\
(\sqsubseteq\mathbf{X}') & \pi \cdot X^S \sqsubseteq T, Inc & \Longrightarrow Inc & (S \subseteq \pi^{-1} \cdot T)
\end{array}
$$

Lemma 22 *If $Inc \Longrightarrow Inc'$ then $Sol(Inc) = Sol(Inc')$.*

Proof. We consider $(\sqsubseteq\mathbf{X})$. Suppose $S \not\subseteq \pi^{-1} \cdot T$ and $\pi \neq id$. Suppose $\theta \in Sol(\pi \cdot X^S \sqsubseteq T, Inc)$. Then $fa((\pi \cdot X^S)\theta) \subseteq T$. By Lemma 16 $(\pi \cdot X^S)\theta \equiv \pi \cdot (X^S\theta)$. By Lemma 10 $fa(\pi \cdot (X^S\theta)) = \pi \cdot fa(X^S\theta)$. So $\pi \cdot fa(X^S\theta) \subseteq T$. π is a bijection, so $fa(X^S\theta) \subseteq \pi^{-1} \cdot T$ and $\theta \in Sol(X^S \sqsubseteq \pi^{-1} \cdot T, Inc)$. Conversely, if $\theta \in Sol(X^S \sqsubseteq \pi^{-1} \cdot T, Inc)$ then $\theta \in Sol(\pi \cdot X^S \sqsubseteq T, Inc)$ follows by a similar argument.

Lemma 23 *Support inclusion problem simplification is confluent and terminating. Write $nf(Inc)$ for the unique $\Longrightarrow$-normal form of Inc.*

Proof.(Sketch) Support inclusion problem simplification is strongly confluent (see, for instance [BN98]), hence it is confluent. It is not hard to define a notion of size on problems such that rewrite rules strictly reduce the size of the problem they act on; it follows that simplification is terminating.

Definition 24 Call Inc **consistent** when $a \sqsubseteq T \notin nf(Inc)$ for all atoms a and permission sorts T. Call Inc **solvable** when $Sol(Inc) \neq \varnothing$. Call Inc **non-trivial** when $nf(Inc) \neq \varnothing$.

Lemma 25 *If Inc is consistent then all $inc \in nf(Inc)$ have the form $X^S \sqsubseteq T$ where $S \not\subseteq T$.*

Definition 26 Define $fV(Inc)$ by $fV(Inc) = \bigcup\{fV(r) \mid \exists T.r \sqsubseteq T \in Inc\}$.

Definition 27 Let $\mathcal{V}$ range over finite sets of unknowns.

Suppose Inc is consistent. For every $X^S \in \mathcal{V}$ make a fixed but arbitrary choice of $X'^{S'}$ such that $X'^{S'} \notin \mathcal{V}$ and $S' = S \cap \bigcap\{T \mid X^S \sqsubseteq T \in nf(Inc)\}$.

We make our choice injectively; for distinct $X^S \in fV(Inc)$ and $Y^T \in fV(Inc)$, we choose $X'^{S'}$ and $Y'^{T'}$ distinct. It will be convenient to write $\mathcal{V}'^{\mathcal{V}}_{Inc}$ for the set of our choices $\{X'^{S'} \mid X^S \in \mathcal{V}\}$. Define a substitution $\rho^{\mathcal{V}}_{Inc}$ by:

$$\rho^{\mathcal{V}}_{Inc}(X^S) \equiv id \cdot X'^{S'} \text{ if } X^S \in \mathcal{V} \qquad \rho^{\mathcal{V}}_{Inc}(Y^T) \equiv id \cdot Y^T \text{ otherwise.}$$

Lemma 28 *If Inc is consistent then $\rho^{\mathcal{V}}_{Inc} \in Sol(Inc)$. ('$\rho^{\mathcal{V}}_{Inc}$ solves Inc'.)*

Proof. Suppose Inc is a $\Longrightarrow$-normal form. If $X^S \sqsubseteq T \in Inc$ then $\rho^{\mathcal{V}}_{Inc}(X) = id \cdot X'^{S'}$ for an S' which, by construction, satisfies $S' \subseteq T$. The result follows.

More generally, if Inc is not a $\Longrightarrow$-normal form then note that by Lemma 22 $Sol(Inc) = Sol(nf(Inc))$; we use the previous paragraph.

Theorem 29 *Inc is consistent if and only if Inc is solvable.*

Proof. By Lemma 22 $Sol(Inc) = Sol(nf(Inc))$, so it suffices to show the result for the special case that Inc is a $\Longrightarrow$-normal form.

Suppose Inc is inconsistent, so $nf(Inc)$ contains some support inclusions of the form $a \sqsubseteq T$ where $a \notin T$. $a\theta \equiv a$ always, so there is no substitution θ such that $a\theta \subseteq T$. Conversely, if Inc is consistent the result follows by Lemma 28.

Definition 30 Suppose that Inc is consistent, $fV(Inc) \subseteq \mathcal{V}$, and $\theta \in Sol(Inc)$. Define a substitution $\theta\text{-}\rho^{\mathcal{V}}_{Inc}$ by:

- $(\theta\text{-}\rho^{\mathcal{V}}_{Inc})(X'^{S'}) \equiv \theta(X^S)$ if $X^S \in \mathcal{V}$ and $\rho^{\mathcal{V}}_{Inc}(X^S) \equiv id \cdot X'^{S'}$.
- $(\theta\text{-}\rho^{\mathcal{V}}_{Inc})(X^S) \equiv \theta(X^S)$ if $X^S \notin \mathcal{V}$.

Lemma 31 *Suppose $\theta \in Sol(Inc)$ and $fV(Inc) \subseteq \mathcal{V}$. Then $\rho^{\mathcal{V}}_{Inc}$ exists, and $\theta\text{-}\rho^{\mathcal{V}}_{Inc}$ is a substitution.*

Proof. If $\theta \in Sol(Inc)$ then Inc is solvable (Definition 20). By Theorem 29, Inc is consistent. Therefore $\rho^{\mathcal{V}}_{Inc}$ from Definition 27 exists. We now show that $fa((\theta\text{-}\rho^{\mathcal{V}}_{Inc})(X'^{S'})) \subseteq S$ always. We reason by cases:

– The case that $id \cdot X'^{S'} \equiv \rho^{\mathcal{V}}_{Inc}(X^S)$ for $X^S \in \mathcal{V}$.
It is not hard to prove that $fV(nf(Inc)) \subseteq fV(Inc)$ always, so $fV(nf(Inc)) \subseteq \mathcal{V}$.
There are two sub-cases:
– The case $X^S \notin fV(nf(Inc))$. Then $S = S'$ and $(\theta\text{-}\rho^{\mathcal{V}}_{Inc})(X'^S) = \theta(X^S)$. By assumption $fa(\theta(X^S)) \subseteq S$.
– The case $X^S \in fV(nf(Inc))$. $\theta \in Sol(Inc)$ so by Lemma 22 $\theta \in Sol(nf(Inc))$ and so $fa(\theta(X^S)) \subseteq T$ for every T such that $X^S \sqsubseteq T \in nf(Inc)$. By definition $S' = \bigcap\{T \mid X^S \sqsubseteq T \in nf(Inc)\}$ and the result follows.
– Otherwise, $(\theta\text{-}\rho^{\mathcal{V}}_{Inc})(X^S) \equiv \theta(X^S)$. By assumption $fa(\theta(X^S)) \subseteq S$.

Theorem 32 *Suppose $\theta \in Sol(Inc)$ and suppose $fV(Inc) \subseteq \mathcal{V}$. Then $\theta(X^S) \equiv (\rho^{\mathcal{V}}_{Inc} \circ (\theta\text{-}\rho^{\mathcal{V}}_{Inc}))(X^S)$ for every $X^S \in \mathcal{V}$.*

Proof. $\rho(X^S) \equiv id \cdot X'^S$ for some fresh $X'^S \notin \mathcal{V}$, and $(\theta\text{-}\rho^{\mathcal{V}}_{Inc})(X'^S) \equiv \theta(X^S)$. The result follows by Lemma 9.

5 Permissive nominal unification problems

5.1 Problems, solutions, the unification algorithm

Definition 33 An **equality (problem)** is a pair $r \stackrel{?}{=} s$. A **problem** Pr is a finite multiset of equalities. Define $Pr\theta$ by $Pr\theta = \{r\theta \stackrel{?}{=} s\theta \mid r \stackrel{?}{=} s \in Pr\}$.

Definition 34 θ **solves** Pr when $r \stackrel{?}{=} s \in Pr$ implies $r\theta =_\alpha s\theta$. Write $Sol(Pr)$ for the set of solutions to Pr. Call Pr **solvable** when $Sol(Pr)$ is non-empty.

A solution to Pr 'makes the equalities valid', as for first- and higher-order unification. This simplifies the nominal unification notion of solution (Definition 66 or [UPG04, Definition 3.1]) based on 'a substitution + a freshness context'.

Lemma 35 *$\theta \circ \theta' \in Sol(Pr)$ if and only if $\theta' \in Sol(Pr\theta)$.*

Proof. Suppose $\theta \circ \theta' \in Sol(Pr)$ and $r \stackrel{?}{=} s \in Pr$. We reason using Theorem 19: $(r\theta)\theta' \equiv r(\theta \circ \theta') =_\alpha s(\theta \circ \theta') \equiv (s\theta)\theta'$. The reverse implication is similar.

Definition 36 If Pr is a problem, define a support inclusion problem $Pr_\sqsubseteq$ by: $Pr_\sqsubseteq = \{r \sqsubseteq fa(s),\ s \sqsubseteq fa(r) \mid r \stackrel{?}{=} s \in Pr\}$.

Definition 37 Define a **simplification** rewrite relation $\mathcal{V}; Pr \Longrightarrow \mathcal{V}'; Pr'$ by:

($\stackrel{?}{=}$**a**)	$\mathcal{V}; a \stackrel{?}{=} a,\ Pr$	$\Longrightarrow$	$\mathcal{V}; Pr$
($\stackrel{?}{=}$**f**)	$\mathcal{V}; \mathsf{f}(r_1, \ldots) \stackrel{?}{=} \mathsf{f}(s_1, \ldots),\ Pr$	$\Longrightarrow$	$\mathcal{V}; r_1 \stackrel{?}{=} s_1, \ldots, Pr$
($\stackrel{?}{=}$[**a**])	$\mathcal{V}; [a]r \stackrel{?}{=} [a]s,\ Pr$	$\Longrightarrow$	$\mathcal{V}; r \stackrel{?}{=} s,\ Pr$
($\stackrel{?}{=}$[**b**])	$\mathcal{V}; [a]r \stackrel{?}{=} [b]s,\ Pr$	$\Longrightarrow$	$\mathcal{V}; (b\ a) \cdot r \stackrel{?}{=} s,\ Pr$ $(b \notin fa(r))$
($\stackrel{?}{=}$**X**)	$\mathcal{V}; \pi \cdot X^S \stackrel{?}{=} \pi \cdot X^S,\ Pr$	$\Longrightarrow$	$\mathcal{V}; Pr$
(**I1**)	$\mathcal{V}; \pi \cdot X^S \stackrel{?}{=} s,\ Pr$	$\stackrel{[X^S := \pi^{-1} \cdot s]}{\Longrightarrow}$	$\mathcal{V}; Pr[X^S := \pi^{-1} \cdot s]$ $(X^S \notin fV(s),\ fa(s) \subseteq \pi \cdot S)$
(**I2**)	$\mathcal{V}; r \stackrel{?}{=} \pi \cdot X^S,\ Pr$	$\stackrel{[X^S := \pi^{-1} \cdot r]}{\Longrightarrow}$	$\mathcal{V}; Pr[X^S := \pi^{-1} \cdot r]$ $(X^S \notin fV(r),\ fa(r) \subseteq \pi \cdot S)$
(**I3**)	$\mathcal{V}; Pr$	$\stackrel{\rho^{\mathcal{V}}_{Pr_\sqsubseteq}}{\Longrightarrow}$	$\mathcal{V} \cup \mathcal{V}'^{\mathcal{V}}_{Pr_\sqsubseteq}; Pr\ \rho^{\mathcal{V}}_{Pr_\sqsubseteq}$ ($Pr_\sqsubseteq$ consistent and non-trivial)

Call ($\stackrel{?}{=}$**a**), ($\stackrel{?}{=}$**f**), ($\stackrel{?}{=}$[**a**]), ($\stackrel{?}{=}$[**b**]), and ($\stackrel{?}{=}$**X**) **non-instantiating rules**.
Call (**I1**), (**I2**), and (**I3**) **instantiating rules**.

We insist $Pr_\sqsubseteq$ is non-trivial (Definition 20) to avoid indefinite rewrites. We insist $Pr_\sqsubseteq$ is consistent so $\rho^{\mathcal{V}}_{Pr_\sqsubseteq}$ exists. $\rho^{\mathcal{V}}_{Pr_\sqsubseteq}$ and $\mathcal{V}'^{\mathcal{V}}_{Pr_\sqsubseteq}$ are defined in Definition 27.

Lemma 38 *If $Pr \Longrightarrow Pr'$ by a non-instantiating rule then $Sol(Pr) = Sol(Pr')$.*

Proof. The interesting case is ($\stackrel{?}{=}$[**b**]). Suppose that $Pr = [a]r \stackrel{?}{=} [b]s, Pr''$ and $b \notin fa(r)$ and so $Pr \Longrightarrow (b\ a) \cdot r \stackrel{?}{=} s, Pr''$ with ($\stackrel{?}{=}$[**b**]). Then:
– Suppose $([a]r)\theta =_\alpha ([b]s)\theta$. By Definition 14 $[a](r\theta) =_\alpha [b](s\theta)$. By the structure of the rules in Definition 8, $(b\ a)\cdot(r\theta) =_\alpha s\theta$. By Lemma 16 and Theorem 11, $((b\ a) \cdot r)\theta =_\alpha s\theta$.
– Suppose $((b\ a)\cdot r)\theta =_\alpha s\theta$. By Lemma 16 and Theorem 11, $(b\ a)\cdot(r\theta) =_\alpha s\theta$. By Theorem 15 $b \notin fa(r\theta)$. Therefore by ($=_\alpha$[**b**]) $[a](r\theta) =_\alpha [b](s\theta)$. By Definition 14 $[a](r\theta) =_\alpha [b](s\theta)$, as required.

Definition 39 Define $fV(Pr) = \bigcup\{fV(r) \cup fV(s) \mid r \stackrel{?}{=} s \in Pr\}$.

Definition 40 Suppose $\mathcal{V}$ is a set of unknowns. Define $\theta|_{\mathcal{V}}$ by:[4]

$$\theta|_{\mathcal{V}}(X) \equiv \theta(X) \text{ if } X \in \mathcal{V} \qquad \theta|_{\mathcal{V}}(X) \equiv id \cdot X \text{ otherwise.}$$

[4] We overload $|$, for technical convenience: $\pi|_S$ (Definition 8) is partial and $\theta|_{\mathcal{V}}$ is total.

Definition 41 If Pr is a problem, define a **unification algorithm** by:

1. Rewrite $fV(Pr); Pr$ using the rules of Definition 37 as much as possible, with top-down precedence (so apply $(\stackrel{?}{=}\mathbf{a})$ before $(\stackrel{?}{=}\mathsf{f})$, and so on down to (**I3**)).
2. If we reduce to $\mathcal{V}'; \varnothing$, we succeed and return $\theta|_{\mathcal{V}}$ where θ is the functional composition of all the substitutions labelling rewrites (we take $\theta = id$ if there are none). Otherwise, we fail.

Lemma 42 *The algorithm of Definition 41 always terminates.*

Proof. By an easy argument using a notion of the size of a unification problem.

Lemma 43 *Suppose $\theta(X^S) =_\alpha \theta'(X^S)$ for all $X^S \in fV(Pr)$. Then $\theta \in Sol(Pr)$ if and only if $\theta' \in Sol(Pr)$.*

Proof. Unpacking Definition 34 it suffices to show that $r\theta =_\alpha s\theta$ if and only if $r\theta' =_\alpha s\theta'$, for every $r \stackrel{?}{=} s \in Pr$. This is easy using Theorem 17 and the fact by construction (Definition 39) that $fV(r) \subseteq fV(Pr)$ and $fV(s) \subseteq fV(Pr)$.

Definition 44 Write $\theta\text{-}X^S$ for the substitution such that:
$(\theta\text{-}X^S)(X^S) \equiv id \cdot X^S$ and $(\theta\text{-}X^S)(Y^T) \equiv \theta(Y^T)$ for all other Y^T

Theorem 45 *Suppose $(id \cdot X^S)\theta =_\alpha s\theta$ and $X^S \notin fV(s)$. Then:*
$\theta(X^S) =_\alpha ([X^S{:=}s] \circ (\theta\text{-}X^S))(X^S)$ *and* $\theta(Y^T) =_\alpha ([X^S{:=}s] \circ (\theta\text{-}X^S))(Y^T)$

Proof. We reason as follows:

$([X^S{:=}s] \circ (\theta\text{-}X^S))(X^S)$	$\equiv$	$s(\theta\text{-}X^S)$	Definition 14
	$\equiv$	$s\theta$	$X^S \notin fV(s)$, Theorem 17
	$=_\alpha$	$\theta(X^S)$	Assumption
$([X^S{:=}s] \circ (\theta\text{-}X^S))(Y^T)$	$\equiv$	$(\theta\text{-}X^S)(Y^T)$	Definition 18
	$\equiv$	$\theta(Y^T)$	Definition 44

5.2 Simplification rewrites calculate principal solutions

Definition 46 Write $\theta_1 \leq \theta_2$ when there exists θ' such that $X^S\theta_2 =_\alpha X^S(\theta_1 \circ \theta')$ always. We call $\leq$ the **instantiation ordering**.

Definition 47 A **principal** (or **most general**) solution to a problem Pr is a solution $\theta \in Sol(Pr)$ such that $\theta \leq \theta'$ for all other $\theta' \in Sol(Pr)$.

Our main results are Theorems 48 — the unification algorithm from Definition 41 calculates a solution — and 53 — the solution it calculates, is principal.

Theorem 48 *Suppose $fV(Pr) \subseteq \mathcal{V}$. If $\mathcal{V}; Pr \stackrel{\theta}{\Longrightarrow} \mathcal{V}'; \varnothing$ then $\theta|_{\mathcal{V}} \in Sol(Pr)$.*

Proof. By induction on the length of the path in $\stackrel{\theta}{\Longrightarrow}$. If it has length 0 then $Pr = \varnothing$ and $\theta \equiv id$ and the result follows. Otherwise, there are three cases:

– *The non-instantiating case.* Suppose $\mathcal{V}; Pr \Longrightarrow \mathcal{V}; Pr'' \overset{\theta}{\Longrightarrow}{}^{*} \mathcal{V}'; \varnothing$. By easy calculations $fV(Pr'') \subseteq \mathcal{V}$. By inductive hypothesis $\theta|_{\mathcal{V}} \in Sol(Pr'')$. By Lemma 38, $\theta|_{\mathcal{V}} \in Sol(Pr)$.

– *The case of* (**I1**) *or* (**I2**). Suppose $\mathcal{V}; Pr \overset{\chi}{\Longrightarrow} \mathcal{V}; Pr\chi \overset{\theta'}{\Longrightarrow}{}^{*} \mathcal{V}'; \varnothing$. By easy calculations $fV(Pr\chi) \subseteq \mathcal{V}$. By inductive hypothesis $\theta'|_{\mathcal{V}} \in Sol(Pr\chi)$. It is a fact that $(\chi \circ \theta')|_{\mathcal{V}} = \chi \circ (\theta'|_{\mathcal{V}})$. By Lemma 35, $(\chi \circ \theta')|_{\mathcal{V}} \in Sol(Pr)$.

– *The case of* (**I3**). Suppose $\mathcal{V}; Pr \overset{\rho}{\Longrightarrow} \mathcal{V}'; Pr\rho \overset{\theta'}{\Longrightarrow}{}^{*} \mathcal{V}''; \varnothing$. By easy calculations $fV(Pr\rho) \subseteq \mathcal{V}'$. By inductive hypothesis $\theta'|_{\mathcal{V}'} \in Sol(Pr\rho)$. By Lemma 35, $\rho \circ (\theta'|_{\mathcal{V}'}) \in Sol(Pr)$. It is a fact that $\rho \circ (\theta'|_{\mathcal{V}'}) = (\rho \circ \theta')|_{\mathcal{V}'}$. By Lemma 43, $(\rho \circ \theta')|_{\mathcal{V}} \in Sol(Pr)$.

Lemma 49 *If $\theta_1 \leq \theta_2$ then $\theta \circ \theta_1 \leq \theta \circ \theta_2$.*

Proof. By Definition 46 θ' exists such that $X^S\theta_2 =_\alpha X^S(\theta_1 \circ \theta')$ always. We use Theorems 19 and 17: $X^S(\theta \circ \theta_2) \equiv (X^S\theta)\theta_2 =_\alpha (X^S\theta)(\theta_1 \circ \theta') \equiv X^S((\theta \circ \theta_1) \circ \theta')$

Lemma 50 *Suppose $X^S\theta_2 =_\alpha X^S\theta_2'$ always. Then $\theta_1 \leq \theta_2$ implies $\theta_1 \leq \theta_2'$.*

Proof. By a routine calculation unpacking Definition 46 and using Theorem 11.

Lemma 51 *If $r =_\alpha s$ then $fa(r) = fa(s)$.*

Lemma 52 *If $\theta \in Sol(Pr)$ (Definition 34) then $\theta \in Sol(Pr_{\sqsubset})$ (Definition 20).*

Proof. Using Lemma 51.

Theorem 53 *Suppose $fV(Pr) \subseteq \mathcal{V}$. If $\mathcal{V}; Pr \overset{\theta}{\Longrightarrow}{}^{*} \mathcal{V}'; \varnothing$ then $\theta|_{\mathcal{V}}$ is a principal solution to Pr.*

Proof. By Theorem 48 $\theta|_{\mathcal{V}} \in Sol(Pr)$. We prove that $\theta|_{\mathcal{V}}$ is principal by induction on the path length of $\mathcal{V}; Pr \overset{\theta}{\Longrightarrow}{}^{*} \mathcal{V}'; \varnothing$.

– *Length* 0. So $Pr = \varnothing$ and $\theta = id|_{\mathcal{V}}$. $id|_{\mathcal{V}} \leq \theta'|_{\mathcal{V}}$ is a fact of Definition 46.

– *Length* $k+1$. We consider the rules in Definition 37.

– *The non-instantiating case.* Suppose $\mathcal{V}; Pr \Longrightarrow \mathcal{V}; Pr' \overset{\theta}{\Longrightarrow}{}^{*} \mathcal{V}'; \varnothing$ where $\mathcal{V}; Pr \Longrightarrow \mathcal{V}; Pr'$ is non-instantiating. By inductive hypothesis, $\theta|_{\mathcal{V}}$ is a principal solution of Pr'. By Lemma 38 $\theta|_{\mathcal{V}}$ is also a principal solution of Pr.

– The case (**I1**). Suppose $fa(s) \subseteq \pi \cdot S$ and $X^S \notin fV(s)$. Write $\chi = [X^S := \pi^{-1} \cdot s]$. Suppose $Pr = \pi \cdot X^S \stackrel{?}{=} s,\ Pr''$ so that $\mathcal{V}; \pi \cdot X^S \stackrel{?}{=} s,\ Pr'' \overset{\chi}{\Longrightarrow} \mathcal{V}; Pr''\chi$ and suppose $\mathcal{V}; Pr''\chi \overset{\theta''}{\Longrightarrow}{}^{*} \mathcal{V}'; \varnothing$ and $\theta'|_{\mathcal{V}} \in Sol(Pr)$.
By Theorem 48 $\theta''|_{\mathcal{V}} \in Sol(Pr''\chi)$. It is routine to check that $fV(Pr''\chi) \subseteq \mathcal{V}$. By Theorem 45 and Lemma 43, $\chi \circ (\theta'|_{\mathcal{V}}\text{-}X^S) \in Sol(Pr)$. It is a fact that $(\theta'|_{\mathcal{V}}\text{-}X^S) = (\theta'\text{-}X^S)|_{\mathcal{V}}$ so by Lemma 35, $(\theta' - X^S)|_{\mathcal{V}} \in Sol(Pr''\chi)$.
By inductive hypothesis $\theta''|_{\mathcal{V}} \leq (\theta'\text{-}X^S)|_{\mathcal{V}}$. By Lemma 49 we have $\chi \circ (\theta''|_{\mathcal{V}}) \leq \chi \circ (\theta'\text{-}X^S)|_{\mathcal{V}}$. Now by assumption $fV(s) \subseteq \mathcal{V}$ and $X \in \mathcal{V}$, and it follows that $\chi \circ (\theta''|_{\mathcal{V}}) = (\chi \circ \theta'')|_{\mathcal{V}}$. Also, it is a fact that $(\theta' - X^S)|_{\mathcal{V}} = \theta'|_{\mathcal{V}}\text{-}X^S$. By Theorem 45 and Lemma 50, $(\chi \circ \theta'')|_{\mathcal{V}} \leq \theta'|_{\mathcal{V}}$ as required.

– The case (**I2**) is similar to the case of (**I1**).

– The case (**I3**). Suppose $Pr_{\sqsubseteq}$ is consistent and non-trivial. Write $\rho = \rho^{\mathcal{V}}_{Pr_{\sqsubseteq}}$, so that $\mathcal{V}; Pr \stackrel{\rho}{\Longrightarrow} \mathcal{V}''; Pr\rho$ and $\mathcal{V}''; Pr\rho \stackrel{\theta''}{\Longrightarrow}{}^{*} \mathcal{V}'; \varnothing$ and suppose $\theta'|_{\mathcal{V}} \in Sol(Pr)$. By Theorem 48 $\theta''|_{\mathcal{V}''} \in Sol(Pr\rho)$. $\mathcal{V}'' = \mathcal{V} \cup \mathcal{V}'^{\mathcal{V}}_{Pr_{\sqsubseteq}}$, so $fV(Pr\rho) \subseteq \mathcal{V}''$. By Lemma 52 $\theta'|_{\mathcal{V}} \in Sol(Pr_{\sqsubseteq})$. By Theorem 32 and Lemma 43 it follows that $\rho \circ (\theta'|_{\mathcal{V}}\text{-}\rho) \in Sol(Pr)$. By Lemma 35, $\theta'|_{\mathcal{V}}\text{-}\rho \in Sol(Pr\rho)$.
By inductive hypothesis $\theta''|_{\mathcal{V}} \leq \theta'|_{\mathcal{V}}\text{-}\rho$. By Lemma 49 $\rho \circ \theta''|_{\mathcal{V}} \leq \rho \circ (\theta'|_{\mathcal{V}}\text{-}\rho)$. It is a fact that $\rho \circ (\theta''|_{\mathcal{V}}) = (\rho \circ \theta'')|_{\mathcal{V}}$. By Theorem 32 and Lemma 50, $(\rho \circ \theta'')|_{\mathcal{V}} \leq \theta'|_{\mathcal{V}}$ as required.

Lemma 54 *If* $\mathcal{V}; Pr \stackrel{\chi}{\Longrightarrow} \mathcal{V}; Pr'$ *with* (**I1**) *or* (**I2**) *then* $\theta \in Sol(Pr)$ *implies* $\theta\text{-}\chi \in Sol(Pr')$. *Similarly, if* $\mathcal{V}; Pr \stackrel{\rho}{\Longrightarrow} \mathcal{V}'; Pr'$ *with* (**I3**) *then* $\theta \in Sol(Pr)$ *implies* $\theta\text{-}\rho \in Sol(Pr')$.

Proof. Suppose $fa(s) \subseteq \pi \cdot S$ and $X^S \notin fV(s)$. Write $\chi = [X^S := \pi^{-1} \cdot s]$. Suppose $Pr = \pi \cdot X^S \stackrel{?}{=} s,\ Pr''$ so that $\mathcal{V}; \pi \cdot X^S \stackrel{?}{=} s,\ Pr'' \stackrel{\chi}{\Longrightarrow} \mathcal{V}; Pr''\chi$. Now suppose $\theta \in Sol(Pr)$. By Lemma 43 and Theorem 45, $\chi \circ (\theta\text{-}X^S)) \in Sol(Pr)$. By Lemma 35, $\theta\text{-}X^S \in Sol(Pr\chi)$. It follows that $\theta\text{-}X^S \in Sol(Pr''\chi)$ as required.

Suppose $Pr_{\sqsubseteq}$ is consistent and non-trivial. Write $\rho = \rho^{\mathcal{V}}_{Pr_{\sqsubseteq}}$, so that $\mathcal{V}; Pr \stackrel{\rho}{\Longrightarrow} \mathcal{V}''; Pr\rho$. Now suppose $\theta \in Sol(Pr)$. By Lemma 43 and Theorem 32, $\rho \circ (\theta\text{-}\rho) \in Sol(Pr)$. By Lemma 35, $\theta\text{-}\rho \in Sol(Pr\rho)$ as required.

Theorem 55 *Given a problem* Pr, *if the algorithm of Definition 41 succeeds then it returns a principal solution; if it fails then there is no solution.*

Proof. If the algorithm succeeds we use Theorem 53. Otherwise, the algorithm generates an element of the form $\mathsf{f}(r_1, \ldots, r_n) \stackrel{?}{=} \mathsf{f}(r'_1, \ldots, r'_{n'})$ where $n \neq n'$, $\mathsf{f}(\ldots) \stackrel{?}{=} \mathsf{g}(\ldots)$, $\mathsf{f}(\ldots) \stackrel{?}{=} [a]s$, $\mathsf{f}(\ldots) \stackrel{?}{=} a$, $[a]r =_\alpha a$, $[a]r =_\alpha b$, $a \stackrel{?}{=} b$, a Pr such that $Pr_{\sqsubseteq}$ is inconsistent, or $\pi \cdot X^S \stackrel{?}{=} r$ or $r \stackrel{?}{=} \pi \cdot X^S$ where $X^S \in fV(r)$. By arguments on syntax and size of syntax, no solution to the reduced problem exists. It follows by Lemma 54 that no solution to Pr exists.

6 Relation to nominal terms

In permissive nominal terms, freshness information is fixed once and for all. This is mentioned already as a design alternative in [UPG04, Remark 2.6], but there, we would obtain 'permission sorts' A such that A is infinite and $\mathbb{A} \setminus A$ is finite. Permission sorts of *co-infinite* sets of atoms are new, as far as we know.

We will now develop the intuitively clear connection with [UPG04] into a precise mathematical correspondence between nominal unification and permissive nominal unification. Definition 59 translates from 'nominal' to 'permissive'. Theorems 62 and 63 express how this translation is sound and complete for respective notions of α-equivalence. Theorem 71 then shows the most interesting fact: that furthermore, solutions to unification problems are also preserved across the translation.

A feature of Definition 59 is that it maps nominal terms to permissive nominal terms with free atoms in *comb*. In nominal terms we may need to enrich the

freshness context (see [GM07, Figure 2, axiom (**fr**)] and [GM09b, e.g. Lemma 25 and Theorem 33]). One way to view the interpretation of Definition 59 is therefore this: *comb* is 'the atoms we had available so far' (any other permission sort would do as well) and $\mathbb{A} \setminus comb$ is 'the atoms with which we will extend the freshness context, in the future'. Both these sets are countably infinite, and syntax is finite, so it is not absolutely necessary to explicitly separate them: permissive nominal terms do this, for each fixed permission sort S; nominal terms do not.

Definition 56 Fix a countably infinite set of **nominal atoms**, $\dot{\mathbb{A}}$. $\dot{a}, \dot{b}, \dot{c}, \ldots$ will range over distinct nominal atoms. Fix a bijection ι between $\dot{\mathbb{A}}$ and *comb* (Definition 2). Fix a countably infinite set of **nominal unknowns**. $\dot{X}, \dot{Y}, \dot{Z}, \ldots$ will range over distinct nominal unknowns. A **nominal permutation** is a bijection $\dot{\pi}$ on $\dot{\mathbb{A}}$ such that $\{\dot{a} \mid \dot{\pi}(\dot{a}) \neq \dot{a}\}$ is finite. $\dot{\pi}, \dot{\pi}', \dot{\pi}'', \ldots$ will range over permutations.

Write $\dot{\pi}^{-1}$ for the inverse of $\dot{\pi}$, $\dot{id}$ for the identity permutation, and $\dot{\pi} \circ \dot{\pi}'$ for function composition, as is standard. For example, $(\dot{\pi} \circ \dot{\pi}')(\dot{a}) = \dot{\pi}(\dot{\pi}'(\dot{a}))$

Definition 57 Define **nominal terms** by $\dot{r}, \dot{s}, \dot{t} ::= \dot{a} \mid \dot{\pi} \cdot \dot{X} \mid [\dot{a}]\dot{r} \mid \mathsf{f}(\dot{r}, \ldots, \dot{r})$, with permutation action

$$\dot{\pi}\cdot\dot{a} \equiv \dot{\pi}(\dot{a}) \quad \dot{\pi}\cdot\mathsf{f}(\dot{r}_1, \ldots) \equiv \mathsf{f}(\dot{\pi}\cdot\dot{r}_1, \ldots) \quad \dot{\pi}\cdot[\dot{a}]\dot{r} \equiv [\dot{\pi}(\dot{a})](\dot{\pi}\cdot\dot{r}) \quad \dot{\pi}\cdot(\dot{\pi}'\cdot\dot{X}) \equiv (\dot{\pi}\circ\dot{\pi}')\cdot\dot{X}$$

Write $\equiv$ for syntactic identity. f ranges over term-formers (Definition 1).

Definition 58 A **freshness** is a pair $\dot{a}\#\dot{r}$. An **equality** is a pair $\dot{r} = \dot{s}$. A **freshness context** is a finite set of freshnesses of the form $\dot{a}\#\dot{X}$. Define **derivable freshness** and **derivable equality** by:

$$\frac{}{\Delta \vdash \dot{a}\#\dot{b}}\,(\#\dot{\mathbf{b}}) \qquad \frac{\Delta \vdash \dot{a}\#\dot{r}_i \quad (1 \leq i \leq n)}{\Delta \vdash \dot{a}\#\mathsf{f}(\dot{r}_1, \ldots, \dot{r}_n)}\,(\#\mathsf{f}) \qquad \frac{}{\Delta \vdash \dot{a}\#[\dot{a}]\dot{r}}\,(\#[\dot{\mathbf{a}}])$$

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

$$\frac{}{\Delta \vdash \dot{a} = \dot{a}}\,(=\dot{\mathbf{a}}) \qquad \frac{\Delta \vdash \dot{r}_i = \dot{s}_i \quad (1 \leq i \leq n)}{\Delta \vdash \mathsf{f}(\dot{r}_1, \ldots, \dot{r}_n) = \mathsf{f}(\dot{s}_1, \ldots, \dot{s}_n)}\,(=\mathsf{f}) \qquad \frac{\Delta \vdash \dot{r} = \dot{s}}{\Delta \vdash [\dot{a}]\dot{r} = [\dot{a}]\dot{s}}\,(=[\dot{\mathbf{a}}])$$

$$\frac{\Delta \vdash (\dot{b}\ \dot{a})\cdot\dot{r} = \dot{s} \quad \Delta \vdash \dot{b}\#\dot{r}}{\Delta \vdash [\dot{a}]\dot{r} = [\dot{b}]\dot{s}}\,(=[\dot{\mathbf{b}}]) \qquad \frac{(\dot{a}\#\dot{X} \in \Delta \text{ for every } \dot{\pi}(\dot{a}) \neq \dot{\pi}'(\dot{a}))}{\Delta \vdash \dot{\pi}\cdot\dot{X} = \dot{\pi}'\cdot\dot{X}}\,(=\dot{\mathbf{X}})$$

Definition 58 repeats [UPG04, Figure 2], up to differences in presentation. A full discussion of nominal terms is in [UPG04]. $\Delta \vdash \dot{a}\#\dot{r}$ corresponds with '$a \notin fa(r)$' (made formal in Lemma 61), and $\Delta \vdash \dot{r} = \dot{s}$ corresponds with '$r =_\alpha s$' (see Theorems 62 and 63). Δ plays the role of permissions sorts, but is part of the judgement-form.

Definition 59 Define a mapping $[\![\dot\pi]\!]$ from nominal permutations to permissive nominal permutations by $[\![\dot\pi]\!](\iota(\dot a)) = \iota(\dot\pi(\dot a))$ and $[\![\dot\pi]\!](c) = c$ for all $c \in \mathbb{A} \setminus comb$. Define an **interpretation** $[\![\dot r]\!]_\Delta$ by:

$$[\![\dot a]\!]_\Delta \equiv \iota(\dot a) \qquad [\![\mathsf{f}(\dot r_1,\ldots,\dot r_n)]\!]_\Delta \equiv \mathsf{f}([\![\dot r_1]\!]_\Delta,\ldots,[\![\dot r_n]\!]_\Delta) \qquad [\![[\dot a]\dot r]\!]_\Delta \equiv [\iota(\dot a)][\![\dot r]\!]_\Delta$$
$$[\![\dot\pi \cdot \dot X]\!]_\Delta \equiv [\![\dot\pi]\!] \cdot X^S \quad \text{where } S = comb \setminus \{\iota(\dot a) \mid \dot a \# \dot X \in \Delta\}$$

Here, we make a fixed but arbitrary choice of X^S for each $\dot X$, injectively so that $[\![\dot X]\!]_\Delta$ and $[\![\dot Y]\!]_\Delta$ are always distinct.

Lemma 60 $[\![\dot\pi]\!] \cdot [\![\dot r]\!]_\Delta \equiv [\![\dot\pi \cdot \dot r]\!]_\Delta$

Lemma 61 *$\iota(\dot a) \notin fa([\![\dot r]\!]_\Delta)$ if and only if $\Delta \vdash \dot a \# \dot r$. $b \notin fa([\![\dot r]\!]_\Delta)$ if $b \in \mathbb{A} \setminus comb$.*

Theorem 62 *$[\![\dot r]\!]_\Delta =_\alpha [\![\dot s]\!]_\Delta$ implies $\Delta \vdash \dot r = \dot s$.*

Proof. Induction on the derivation of $[\![\dot r]\!]_\Delta =_\alpha [\![\dot s]\!]_\Delta$. We consider some cases:
– The case $(=_\alpha[\mathbf{b}])$. Suppose $(\iota(\dot b)\ \iota(\dot a)) \cdot [\![\dot r]\!]_\Delta =_\alpha [\![\dot s]\!]_\Delta$ and $\iota(\dot b) \notin fa([\![\dot r]\!]_\Delta)$. By Lemmas 60 and 61 $[\![(\dot b\ \dot a) \cdot \dot r]\!]_\Delta =_\alpha [\![\dot s]\!]_\Delta$ and $\Delta \vdash \dot b \# \dot r$. By inductive hypothesis $\Delta \vdash (\dot b\ \dot a) \cdot \dot r = \dot s$. We use $(=[\dot{\mathbf{b}}])$.
– The case $(=_\alpha \mathbf{X})$. Suppose $[\![\dot\pi]\!]|_S = [\![\dot\pi']\!]|_S$ where $S = comb \setminus \{\iota(\dot a) \mid \dot a \# \dot X \in \Delta\}$. ι is injective, so $a \# \dot X \in \Delta$ for all $\dot a$ such that $\dot\pi(\dot a) \neq \dot\pi'(\dot a)$. We use $(=\dot{\mathbf{X}})$.

Theorem 63 *If $\Delta \vdash \dot r = \dot s$ then $[\![\dot r]\!]_\Delta =_\alpha [\![\dot s]\!]_\Delta$.*

Proof. Induction on the derivation of $\Delta \vdash \dot r = \dot s$. We consider some cases:
– The case $(=[\dot{\mathbf{b}}])$. Suppose $\Delta \vdash (\dot b\ \dot a) \cdot \dot r = \dot s$ and $\Delta \vdash \dot b \# \dot r$. By inductive hypothesis and Lemma 60, $(\dot b\ \dot a) \cdot [\![\dot r]\!]_\Delta =_\alpha [\![\dot s]\!]_\Delta$. By Lemma 61 $\iota(\dot b) \notin fa([\![\dot r]\!]_\Delta)$. We use $(\stackrel{?}{=}[\mathbf{b}])$.
– The case $(=\dot{\mathbf{X}})$. Recall that $[\![\dot\pi \cdot \dot X]\!]_\Delta = [\![\dot\pi]\!] \cdot X^S$ and $[\![\dot\pi' \cdot \dot X]\!]_\Delta = [\![\dot\pi']\!] \cdot X^S$ where $S = comb \setminus \{\iota(\dot a) \mid \dot a \# \dot X \in \Delta\}$. Suppose $\dot\pi(\dot a) \neq \dot\pi'(\dot a)$ implies $\Delta \vdash \dot a \# \dot X$. Using Lemma 61, $[\![\dot\pi]\!](\iota(\dot a)) \neq [\![\dot\pi']\!](\iota(\dot a))$ implies $\iota(\dot a) \notin S$. We use $(\stackrel{?}{=}\mathbf{X})$.

Definition 64 A **substitution** $\dot\theta$ is a function from nominal unknowns to nominal terms such that $\{\dot X \mid \dot\theta(\dot X) \not\equiv \dot{id} \cdot \dot X\}$ is finite. $\dot\theta, \dot\theta', \dot\theta'', \ldots$ will range over nominal substitutions. Write $\dot{id}$ for the **identity**, mapping $\dot X$ to $\dot{id} \cdot \dot X$.

Definition 65 Define a **substitution action** on nominal terms by:

$$\dot a\dot\theta \equiv \dot a \quad \mathsf{f}(\dot r_1,\ldots,\dot r_n)\dot\theta \equiv \mathsf{f}(\dot r_1\dot\theta,\ldots,\dot r_n\dot\theta) \quad ([\dot a]\dot r)\dot\theta \equiv [\dot a](\dot r\dot\theta) \quad (\dot\pi \cdot \dot X)\dot\theta \equiv \dot\pi \cdot \dot\theta(\dot X)$$

Definition 66 Following [UPG04, Definition 3.1], a **unification problem** $\dot Pr$ is a finite multiset of freshnesses and equalities. A **solution** to $\dot Pr$ is a pair $(\Delta, \dot\theta)$ such that $\Delta \vdash \dot a \# \dot r\dot\theta$ for every $\dot a \# \dot r \in \dot Pr$, and $\Delta \vdash \dot r\theta = \dot s\theta$ for every $\dot r = \dot s \in \dot Pr$.

Definition 67 We extend the interpretation of Definition 59 to solutions by:

$$[\![(\Delta, \dot\theta)]\!](X^S) \equiv [\![\dot\theta(X)]\!]_\Delta \text{ if } id \cdot X^S \equiv [\![X]\!]_\Delta \quad [\![(\Delta, \dot\theta)]\!](Y^T) \equiv id \cdot Y^T \text{ otherwise}$$

Lemma 68 $[\![\dot{r}]\!]_\Delta[\![(\Delta,\dot{\theta})]\!] \equiv [\![\dot{r}\dot{\theta}]\!]_\Delta$.

Proof. By inductions on $\dot{r}$. We consider the case $\dot{\pi}\cdot\dot{X}$, and reason using Lemma 60: $[\![(\dot{\pi}\cdot\dot{X})\dot{\theta}]\!]_\Delta \equiv [\![\dot{\pi}\cdot\dot{\theta}(\dot{X})]\!]_\Delta \equiv [\![\dot{\pi}]\!]\cdot[\![\dot{\theta}(\dot{X})]\!]_\Delta \equiv [\![\dot{\pi}]\!]\cdot[\![\dot{\theta}]\!]([\![\dot{X}]\!]_\Delta)$

Definition 69 Define $[\![\dot{P}r]\!]_\Delta$ by mapping $\dot{r}=\dot{s}$ to $[\![\dot{r}]\!]_\Delta \stackrel{?}{=} [\![\dot{s}]\!]_\Delta$ and mapping $\dot{a}\#\dot{r}$ to $(b\ \iota(\dot{a}))\cdot[\![\dot{r}]\!]_\Delta \stackrel{?}{=} [\![\dot{r}]\!]_\Delta$, for some choice of fresh b (so $b\not\in fa([\![\dot{r}]\!]_\Delta)$; in fact, it suffices to choose some $b\not\in comb$).

Lemma 70 *Suppose $b\not\in fa(r)$. Then $a\not\in fa(r)$ if and only if $(b\ a)\cdot r =_\alpha r$.*

Theorem 71 *$(\Delta,\dot{\theta})$ solves $\dot{P}r$ if and only if $[\![(\Delta,\dot{\theta})]\!]$ solves $[\![\dot{P}r]\!]_\Delta$.*

Proof. We sketch the necessary reasoning.

Suppose $\Delta\vdash\dot{r}\dot{\theta}=\dot{s}\dot{\theta}$. By Lemma 68 and Theorem 63, $[\![\dot{r}]\!]_\Delta[\![(\Delta,\dot{\theta})]\!] =_\alpha [\![\dot{s}]\!]_\Delta[\![(\Delta,\dot{\theta})]\!]$. The reverse direction uses Theorem 62.

Suppose $\Delta\vdash a\#\dot{r}\dot{\theta}$. By Lemmas 61, 70, 68, and 16 $((b\ \iota(\dot{a}))\cdot[\![\dot{r}]\!]_\Delta)[\![(\Delta,\dot{\theta})]\!] =_\alpha [\![\dot{r}]\!]_\Delta[\![(\Delta,\dot{\theta})]\!]$. Here, we use the b chosen fresh in Definition 69. The reverse direction uses the same results.

7 Conclusions

Nominal contrasted with permissive nominal terms. It can be problematic that in nominal terms [GM09b,GM08] we often want to enrich the freshness context, e.g. to α-convert or in solving a unification problem. Also, it is unfortunate that we cannot 'just quotient terms by α-conversion' since we cannot apply nominal abstract syntax [GP01] ... to nominal terms. Permissive nominal terms let us do all these things, *and* they behave more like first-order terms; we recover Lemma 12, α-equivalence is a property of terms, and the notions of unification problem and solution involve just equality, not equality-and-freshness-context.

The step from permissive nominal terms to nominal terms makes a remarkably big difference. This has implications for developing nominal techniques further; we believe that in many situations, permissive nominal terms may be easier to work with and better-behaved. By Definition 59 and Theorem 71, there is no loss in expressivity.

Permissive nominal terms do not necessarily obsolete nominal terms; if we really do want to discuss 'an arbitrary term', then the nominal terms unknown $\dot{X}$ from Section 6 may be more simply and directly useful than X^{comb}. One possibly interesting extension of permissive nominal terms, is with variables for permission sorts.

Note that the unification algorithm in [UPG04] solves freshnesses concurrently with equalities. We have factored the algorithm differently, so that problems to do with free atoms (Section 4) are solved separately from problems to do with equalities (Section 5). The link is rule (**I3**) in Definition 37.

Concerning implementation, permissive nominal terms are implementable. Sorts are infinite sets of atoms, but differ finitely from *comb* and so may be finitely represented in an implementation.

Related work not based on nominal techniques. Permissive nominal terms resemble first- and higher-order terms more than do nominal terms, but they are a special case of neither. Higher-order patterns also have good properties of first-order terms [Mil91]. A significant difference is that the notion of unification is based on capture-avoiding substitution rather than the (permissive) nominal term capturing substitution; this gives the system a different flavour. The reader is referred to a recent paper [LV08], which goes some way to teasing out formal connections between these two approaches.

Permission sorts can be compared with the types used in [DHK00] and fully developed in [NPP07] that indicate the atoms that may appear in a term.

Hamana's β_0 unification of λ-terms with holes adds a capturing substitution [Ham01]. Level 2 variables (which are instantiated) are annotated with level 1 variable symbols that *may* appear in them; permissive nominal terms move in this direction in the sense that permission sorts also describe which level 1 variable symbols (we call them atoms in this paper) may appear in them, though with our permission sorts there are infinitely many that may, and infinitely many that may not. The treatment of α-equivalence in Hamana's system is not nominal (not based on permutations) and Hamana's system does not have most general unifiers. Similarly Qu-Prolog [NR96] adds level 2 variables, but does not manage α-conversion in nominal style and also, for better or for worse, it is more ambitious in what it expresses and loses mathematical properties (unification is semi-decidable, most general unifiers need not exist).

Future work. We have investigated how the instantiation ordering interacts with the interpretation of Section 6. Most general solutions of nominal unification problems in the sense of [UPG04, Definition 3.1] do translate to most general solutions of the translated problem, in the sense of Definition 46. It is also possible to leverage permissive nominal terms to improve on the correspondence between nominal unification and Miller's pattern unification [Mil92,Mil91] reported on in [LV08]; in [LV08] only *solvability* is considered, but permissive nominal terms' good properties can be applied to give a particularly neat translation of *solutions* between permissive nominal unification and higher-order pattern unification. These results with full proofs will be reported in a journal version of this paper.

We hypothesise that 'permissive' versions of nominal rewriting, logic programming, and algebra, should be possible, following previous work using nominal terms [FG07,CU03,GM07,GM09a]. On this topic, addressing a point by Fernández, suppose $c \notin \mathit{comb}$; then the rewrite $X^{\mathit{comb}} \to X^{\mathit{comb}}$ cannot trigger the rewrite $U^{\mathit{comb} \cup \{c\}} \to U^{\mathit{comb} \cup \{c\}}$ because there is no π such that $\pi \cdot \mathit{comb} = \mathit{comb} \cup \{c\}$, and likewise there is no substitution σ, because of the side-condition that $\mathit{fa}(\sigma(X^{\mathit{comb}})) \subseteq \mathit{comb}$. It may therefore be desirable to consider only permissions sorts of the form $\mathit{comb} \setminus A$ for $A \subseteq \mathbb{A}$ finite. This is not a problem for unification and the proofs in this paper because they do not 'add' atoms to permissions sorts (we only ever need to 'subtract' them, as in Section 4). Alternatively, as mentioned above, we could consider permissive nominal terms syntax with variables ranging over permissions sorts.

References

BN98. Franz Baader and Tobias Nipkow. *Term rewriting and all that*. Cambridge University Press, Great Britain, 1998.

CU03. J. Cheney and C. Urban. System description: Alpha-Prolog, a fresh approach to logic programming modulo alpha-equivalence. In *UNIF'03*, pages 15–19. Universidad Politecnica de Valencia, 2003.

CU04. James Cheney and Christian Urban. Alpha-prolog: A logic programming language with names, binding and alpha-equivalence. In Bart Demoen and Vladimir Lifschitz, editors, *Proc. of the 20th Int'l Conf. on Logic Programming (ICLP 2004)*, number 3132 in Lecture Notes in Computer Science, pages 269–283. Springer, 2004.

DHK00. G. Dowek, Th. Hardin, and C. Kirchner. Higher-order unification via explicit substitutions. *Information and Computation*, 157, 2000.

DHK02. Gilles Dowek, Thérèse Hardin, and Claude Kirchner. Binding logic: Proofs and models. In *LPAR '02: Proceedings of the 9th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning*, pages 130–144, London, UK, 2002. Springer.

FG07. Maribel Fernández and Murdoch J. Gabbay. Nominal rewriting (journal version). *Information and Computation*, 205(6):917–965, 2007.

GM07. Murdoch J. Gabbay and Aad Mathijssen. A formal calculus for informal equality with binding. In *Proceedings of* 14^{th} *Workshop on Logic, Language and Information in Computation (WoLLIC 2007)*, volume 4576 of *Lecture Notes in Computer Science*, pages 162–176, 2007.

GM08. Murdoch J. Gabbay and Dominic P. Mulligan. One-and-a-halfth Order Terms: Curry-Howard for Incomplete Derivations. In *Proceedings of* 15^{th} *Workshop on Logic, Language and Information in Computation (WoLLIC 2008)*, volume 5110 of *Lecture Notes in Artificial Intelligence*, pages 180–194, 2008.

GM09a. Murdoch J. Gabbay and Aad Mathijssen. Nominal (universal) algebra: equational logic with names and binders. *Journal of Logic and Computation*, 2009. Accepted subject to revision.

GM09b. Murdoch J. Gabbay and Dominic P. Mulligan. Two-and-a-halfth Order Lambda-calculus. *Electronic Notes in Theoretical Computer Science*, 2009. To appear.

GP01. Murdoch J. Gabbay and A. M. Pitts. A New Approach to Abstract Syntax with Variable Binding (journal version). *Formal Aspects of Computing*, 13(3–5):341–363, 2001.

Ham01. Makoto Hamana. A logic programming language based on binding algebras. In *TACS'01*, volume 2215 of *Lecture Notes in Computer Science*, pages 243–262. Springer, 2001.

HHP87. Robert Harper, Furio Honsell, and Gordon Plotkin. A framework for defining logics. In *Proc. 2nd Annual IEEE Symposium on Logic in Computer Science, LICS'87*, pages 194–204. IEEE Computer Society Press, 1987.

KvOvR93. J.-W. Klop, V. van Oostrom, and F. van Raamsdonk. Combinatory reduction systems. *Theoretical Computer Science*, 121:279–308, 1993.

LV08. Jordi Levy and Mateu Villaret. Nominal unification from a higher-order perspective. In *Proceedings of RTA'08*, volume 5117 of *Lecture Notes in Computer Science*, pages 246–260. Springer, 2008.

Mil91. Dale Miller. A logic programming language with lambda-abstraction, function variables, and simple unification. *Journal of Logic and Computation*, 1(4):497–536, 1991.

Mil92. Dale Miller. Unification under a mixed prefix. *Journal of Symbolic Computation*, 14(4):321–358, 1992.

MN98. Richard Mayr and Tobias Nipkow. Higher-order rewrite systems and their confluence. *Theoretical Computer Science*, 192:3–29, 1998.

NPP07. Aleksandar Nanevski, Frank Pfenning, and Brigitte Pientka. Contextual modal type theory. *Transactions on Computational Logic*, 2007.

NR96. Peter Nickolas and Peter J. Robinson. The Qu-Prolog unification algorithm: formalisation and correctness. *Theoretical Computer Science*, 169(1):81–112, 1996.

Pau90. Lawrence C. Paulson. Isabelle: the next 700 theorem provers. In P. Odifreddi, editor, *Logic and Computer Science*, pages 361–386. Academic Press, 1990.

PE88. F. Pfenning and C. Elliot. Higher-order abstract syntax. In *PLDI (Programming Language design and Implementation)*, pages 199–208. ACM Press, 1988.

SPG03. M. R. Shinwell, A. M. Pitts, and Murdoch J. Gabbay. FreshML: Programming with Binders Made Simple. In *ICFP'03*, volume 38, pages 263–274. ACM Press, 2003.

UPG04. Christian Urban, Andrew M. Pitts, and Murdoch J. Gabbay. Nominal Unification. *Theoretical Computer Science*, 323(1–3):473–497, 2004.

Acknowledgements. Supported by grant RYC-2006-002131 at the Polytechnic University of Madrid.