



THE UNIVERSITY *of* EDINBURGH

Edinburgh Research Explorer

A simple sequent calculus for nominal logic

Citation for published version:

Cheney, J 2016, 'A simple sequent calculus for nominal logic', *Journal of Logic and Computation*, vol. 26, no. 4, pp. 699-726. <https://doi.org/10.1093/logcom/exu024>

Digital Object Identifier (DOI):

[10.1093/logcom/exu024](https://doi.org/10.1093/logcom/exu024)

Link:

[Link to publication record in Edinburgh Research Explorer](#)

Document Version:

Peer reviewed version

Published In:

Journal of Logic and Computation

Publisher Rights Statement:

This is an electronic version of an article published in Journal of Logic and Computation ©: 2014

General rights

Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy

The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.



A Simple Sequent Calculus for Nominal Logic

James Cheney

December 18, 2013

Abstract

Nominal logic is a variant of first-order logic that provides support for reasoning about bound names in abstract syntax. A key feature of nominal logic is the new-quantifier, which quantifies over *fresh names* (names not appearing in any values considered so far). Previous attempts have been made to develop convenient rules for reasoning with the new-quantifier, but we argue that none of these attempts is completely satisfactory.

In this article we develop a new sequent calculus for nominal logic in which the rules for the new-quantifier are much simpler than in previous attempts. We also prove several structural and metatheoretic properties, including cut-elimination, consistency, and equivalence to Pitts’ axiomatization of nominal logic.

1 Introduction

Nominal logic [15] is a variant of first-order logic with additional constructs for dealing with *names* and *binding* (or *name-abstraction*) based on the primitive notions of bijective renaming (*swapping*) and name-independence (*freshness*). It was introduced by Pitts [15] as a first-order and reasonably well-behaved fragment of *Fraenkel-Mostowski set theory*, the setting for Gabbay and Pitts’ earlier foundational work on formalizing names, freshness, and binding using swapping [8].

One of the most interesting features of nominal logic is the presence of a novel form of quantification over *fresh names*. The formula $\forall a.\varphi$ means, intuitively, “for fresh names a , φ holds”. The intended semantics of nominal logic interprets expressions as values in *finitely-supported nominal sets*, or sets acted upon by name-swapping and such that each value depends on at most finitely many names. The inspiration for the \forall -quantifier is the fact that in the presence of infinitely many names, a fresh name can be chosen for any finitely-supported value, and equally-fresh names are indistinguishable. As a result, a property $\varphi(a)$ holds for *some* fresh name a if and only if it holds for *all* fresh names; in either case, we say that $\forall a.\varphi$ holds.

Several formalizations of nominal logic have been investigated. Pitts introduced nominal logic as a Hilbert-style axiomatic system. Gabbay [9] proposed Fresh Logic (FL), an intuitionistic Gentzen-style natural deduction system. Gabbay and Cheney [7] presented FL_{Seq} , a sequent calculus version of Fresh Logic. Schöpp and Stark have developed a dependent type theory of names and binding that contains nominal logic as a special case [17].

However, none of these formalizations is ideal. Hilbert systems have well-known deficiencies for computer science applications. FL and FL_{Seq} rely on a complicated technical device called *slices* for the rules involving \forall . Schöpp and Stark’s system is much more powerful than seems necessary for many applications of nominal logic, and there are many unresolved issues, such as proof normalization and the decidability of the equality and typechecking judgments.

In this article we present a new and simpler sequent calculus for nominal logic. Its main novelty is the use of freshness information in typing contexts needed in reasoning about \forall -quantified formulas, rather than the technically more cumbersome *slices* used in FL and FL_{Seq} . We prove basic proof-theoretic results such as cut-elimination, establishing that this calculus is proof-theoretically sensible. In addition, we prove that NL^{\Rightarrow} is consistent and equivalent to Pitts’ original axiomatization of nominal logic.

This article incorporates some revised material from a previous conference publication [1], extended with detailed proofs and additional results concerning conservativity. That paper also gave a sound and complete embedding of Miller and Tiu’s $FO\lambda^{\nabla}$ [13] in NL^{\Rightarrow} , extending an earlier result by Gabbay and Cheney [7]

Swapping	
(CS ₁)	$\forall a:\nu, x:\tau. (a a) \cdot x \approx x$
(CS ₂)	$\forall a, a':\nu, x:\tau. (a a') \cdot (a a') \cdot x \approx x$
(CS ₃)	$\forall a, a':\nu. (a a') \cdot a \approx a'$
Equivariance	
(CE ₁)	$\forall a, a':\nu, b, b':\nu', x:\tau. (a a') \cdot (b b') \cdot x \approx ((a a') \cdot b (a a') \cdot b') \cdot (a a') \cdot x$
(CE ₂)	$\forall a, a':\nu, b:\nu', x:\tau. b \# x \supset (a a') \cdot b \# (a a') \cdot x$
(CE ₃)	$\forall a, a':\nu, \bar{x}:\bar{\tau}. (a a') \cdot f(\bar{x}) \approx f((a a') \cdot \bar{x})$
(CE ₄)	$\forall a, a':\nu, \bar{x}:\bar{\tau}. p(\bar{x}) \supset p((a a') \cdot \bar{x})$
(CE ₅)	$\forall b, b':\nu', a:\nu, x:\tau. (b b') \cdot \langle a \rangle x \approx \langle (b b') \cdot a \rangle ((b b') \cdot x)$
Freshness	
(CF ₁)	$\forall a, a':\nu, x:\tau. a \# x \wedge a' \# x \supset (a a') \cdot x \approx x$
(CF ₂)	$\forall a, a':\nu. a \# a' \iff a \not\approx a'$
(CF ₃)	$\forall a:\nu, a':\nu'. a \# a'$
(CF ₄)	$\forall \bar{x}:\bar{\tau}. \exists a:\nu. a \# \bar{x}$
\mathbb{I}-quantifier	
(CQ)	$\forall \bar{x}. (\mathbb{I}a:\nu. \varphi) \iff (\exists a:\nu. a \# \bar{x} \wedge \varphi)$
where $FV(\mathbb{I}a.\varphi) \subseteq \{\bar{x}\}$	
Abstraction	
(CA ₁)	$\forall a, a':\nu, x, x':\tau. \langle a \rangle x \approx \langle a' \rangle x' \iff \begin{array}{l} (a \approx a' \wedge x \approx x') \\ \vee \\ (a' \# x \wedge x' \approx (a a') \cdot x) \end{array}$
(CA ₂)	$\forall y:\langle \nu \rangle \tau. \exists a:\nu, x:\tau. y \approx \langle a \rangle x$

Figure 1: Axioms of Classical Nominal Logic

which gave a sound, but nonconservative translation from $FO\lambda^\nabla$ to FL_{Seq} . These results are not presented in this article.

2 Background

2.1 Pitts' axiomatization

As presented by Pitts, nominal logic consists of typed first-order logic with equality and with a number of special types, type constructors, and function and relation symbols formalized by a collection of axioms. In particular, the basic sort symbols of nominal logic are divided into *data types* δ, δ' and *atom types* ν, ν' (which we shall also preferentially call *name types*). In addition, whenever ν is a name type and τ is a type, there exists another type $\langle \nu \rangle \tau$ called the *abstraction* of τ by ν .

Besides possessing equality at every type, nominal logic includes a binary *freshness* relation symbol $fresh_{\nu\tau} : \nu, \tau \rightarrow o$ for each name type ν and type τ . In addition, nominal logic includes two special function symbols $swap_{\nu\tau} : \nu, \nu, \tau \rightarrow \tau$ and $abs_{\nu\tau} : \nu, \tau \rightarrow \langle \nu \rangle \tau$, called *swapping* and *abstraction* respectively. When there is no risk of confusion, we abbreviate formulas of the form $fresh_{\nu\tau}(a, t)$ as $a \# t$, and terms of the form $swap_{\nu\tau}(a, b, t)$ and $abs_{\nu\tau}(a, t)$ as $(a b) \cdot t$ and $\langle a \rangle t$ respectively. In addition, besides the ordinary \forall and \exists quantifiers, nominal logic possesses a third quantifier, called the *fresh-name quantifier* and written \mathbb{I} . A \mathbb{I} -quantified formula $\mathbb{I}x:\nu.\varphi$ may be constructed for any name-type ν .

Pitts presented a Hilbert-style axiom system for nominal logic shown in Figure 1. The axioms are divided into five groups:

- *Swapping axioms (CS)*: describe the behavior of the swapping operation: swapping a name for itself has no effect (CS₁), swapping is involutive (CS₂), and swapping exchanges names (CS₃).
- *Equivariance axioms (CE)*: prescribe the *equivariance* property, namely that all relations are preserved by and all function symbols commute with swapping. In particular, (CE₁) says that the swapping function symbol itself is equivariant; (CE₂) says that freshness is equivariant, (CE₃) says that all other

function symbols are equivariant, and (CE_4) says that all other relation symbols are equivariant. Also, (CE_5) says that abstraction is equivariant.

- *Freshness axioms (CF)*: describe the behavior of the freshness relation (and its interaction with swapping). (CF_1) says that two names fresh for a value can be exchanged without affecting the value. (CF_2) says that freshness coincides with inequality for names. (CF_3) says that distinct name-types are disjoint. Finally, (CF_4) expresses the *freshness principle*, namely, that for any finite collection of values, a name fresh for all the values simultaneously may be chosen.
- *\mathcal{N} -quantifier axiom scheme (CQ)*: Pitts’ original formalization introduced no new inference rules for \mathcal{N} . Instead, \mathcal{N} was defined using the axiom scheme Q , which asserts $\forall \bar{x}. (\mathcal{N}a.\varphi \iff \exists a.a \# \bar{x} \wedge \varphi)$, where $FV(\varphi) \subseteq \{a, \bar{x}\}$.
- *Abstraction axioms (CA)*: These define special properties of the abstraction function symbol. Specifically, (CA_1) defines equality on abstractions as either structural equality or equality up to “safe” renaming of bound names. Gabbay and Pitts showed that this generalizes α -equivalence in, for example, the lambda-calculus [8]; we shall not repeat the argument here. Axiom (CA_2) states a surjectivity property for abstraction: any value of abstraction type $\langle \nu \rangle \tau$ can be written as $\langle a \rangle x$ for some name $a : \nu$ and value $x : \tau$.

2.2 Gentzen systems

While admirable from a reductionist point of view, Hilbert systems have well-known deficiencies: Hilbert-style proofs can be highly nonintuitive and circuitous. Instead, Gentzen-style *natural deduction* and *sequent* systems provide a more intuitive approach to formal reasoning in which logical connectives are explained as *proof-search* operations. Gentzen systems are especially useful for computational applications, such as automated deduction and logic programming. Such systems are also convenient for relating logics by proof-theoretic translations.

Gentzen-style rules for \mathcal{N} have been considered in previous work. Pitts [15] proposed sequent and natural deduction rules for \mathcal{N} based on the observation that

$$\forall a:\nu.(a \# \bar{x} \supset \varphi(a, \bar{x})) \supset \mathcal{N}a:\nu.\varphi(a, \bar{x}) \supset \exists a:\nu.(a \# \bar{x} \wedge \varphi(a, \bar{x})) .$$

These rules (see Figure 2(NL)) are symmetric, emphasizing \mathcal{N} ’s self-duality. However, they are not closed under substitution, which complicates proofs of cut-elimination or proof-normalization properties.

Gabbay [9] introduced an intuitionistic natural deduction calculus called Fresh Logic (FL) and studied semantic issues including soundness and completeness as well proving proof-normalization. Gabbay and Cheney [7] presented a similar sequent calculus called FL_{Seq} . Both FL and FL_{Seq} had complex rules for \mathcal{N} . In FL , Gabbay introduced a technical device called *slices* for obtaining rules that are closed under substitution. (For the purpose of this discussion, it is not necessary to go into the details of what slices are, since we will show that we can do without them.) Technically, a slice $\varphi[a\#\bar{u}]$ of a formula φ is a decomposition of the formula as $\varphi(a, \bar{x})[\bar{u}/\bar{x}]$ for fresh variables \bar{x} , such that a does not appear in any of the \bar{u} . Slices were used in both FL and FL_{Seq} to deal with \mathcal{N} (see Figure 2(FL, FL_{Seq})). The slice-based rules shown in Figure 2(FL_{Seq}) are closed under substitution, so proving cut-elimination for these rules is relatively straightforward once several technical lemmas involving slices have been proved. Noting that the FL_{Seq} rules are structurally similar to $\forall L$ and $\exists R$, respectively, Gabbay and Cheney observed that alternate rules in which $\mathcal{N}L$ was similar to $\exists L$ and $\mathcal{N}R$ similar to $\forall R$ were possible (see Figure 2(FL'_{Seq})). These rules seem simpler and more deterministic; however, they still involve slices.

Experience gained in the process of implementing α Prolog, a logic programming language based on nominal logic [3], suggests a much simpler reading of \mathcal{N} as a proof-search operation than that implied by the FL -style rules. In α Prolog, when a \mathcal{N} -quantifier is encountered (either in a goal or program clause), proof search proceeds by generating a fresh name a to be used for the \mathcal{N} -quantified name. Besides satisfying a syntactic freshness requirement (like eigenvariables in \forall -introduction or \exists -elimination rules), the fresh name is also required to be *semantically fresh*, that is, fresh for all values appearing in the derivation up to the point at which it is generated. In contrast, the proof-search interpretation suggested by FL -style rules is to

$\frac{\Gamma, a \# \bar{x} \Rightarrow \varphi, \Delta \quad (\bar{x} = FV(\Gamma, \mathbf{I}a:\nu.\varphi, \Delta))}{\Gamma \Rightarrow \mathbf{I}a:\nu.\varphi, \Delta} \mathbf{IR}$	$\frac{\Gamma, a \# \bar{x}, \varphi \Rightarrow \Delta \quad (\bar{x} = FV(\Gamma, \mathbf{I}a:\nu.\varphi, \Delta))}{\Gamma, \mathbf{I}a:\nu.\varphi \Rightarrow \Delta} \mathbf{IL} \quad (NL)$
$\frac{\Gamma \vdash u \# \bar{t} \quad \Gamma \vdash \varphi[a\#\bar{t}][u/a]}{\Gamma \vdash \mathbf{I}a:\nu.\varphi[a\#\bar{t}]} \mathbf{II}$	$\frac{\Gamma \vdash \mathbf{I}a:\nu.\varphi[a\#\bar{t}] \quad \Gamma \vdash u \# \bar{t} \quad \Gamma, \varphi[u/a] \vdash \psi}{\Gamma \Rightarrow \psi} \mathbf{IE} \quad (FL)$
$\frac{\Gamma, u \# \bar{t} \Rightarrow \varphi[u/a]}{\Gamma, u \# \bar{t} \Rightarrow \mathbf{I}a:\nu.\varphi[a\#\bar{t}]} \mathbf{IR}$	$\frac{\Gamma, u \# \bar{t}, \varphi[u/a] \Rightarrow \psi}{\Gamma, u \# \bar{t}, \mathbf{I}a:\nu.\varphi[a\#\bar{t}] \Rightarrow \psi} \mathbf{IL} \quad (FL_{Seq})$
$\frac{\Gamma, a \# \bar{t} \Rightarrow \varphi \quad (a \notin FV(\Gamma, \psi))}{\Gamma \Rightarrow \mathbf{I}a:\nu.\varphi[a\#\bar{t}]} \mathbf{IR}$	$\frac{\Gamma, a \# \bar{t}, \varphi \Rightarrow \psi \quad (a \notin FV(\Gamma, \psi))}{\Gamma, \mathbf{I}a:\nu.\varphi[a\#\bar{t}] \Rightarrow \psi} \mathbf{IL} \quad (FL'_{Seq})$
$\frac{\Sigma \# a:\nu; \Gamma \Rightarrow \varphi \quad (a \notin \Sigma)}{\Sigma; \Gamma \Rightarrow \mathbf{I}a:\nu.\varphi} \mathbf{IR}$	$\frac{\Sigma \# a:\nu; \Gamma, \varphi \Rightarrow \psi \quad (a \notin \Sigma)}{\Sigma; \Gamma, \mathbf{I}a:\nu.\varphi \Rightarrow \psi} \mathbf{IL} \quad (NL \Rightarrow)$

Figure 2: Evolution of rules for \mathbf{I}

search for a suitable slice of the \mathbf{I} -quantified formula. This reading seems much less deterministic than that employed in α Prolog.

In this article we present a simplified sequent calculus for nominal logic, called NL^{\Rightarrow} , in which slices are not needed in the rules for \mathbf{I} (or anywhere else). Following Urban, Pitts, and Gabbay [19, 9], and our prior work [2], we employ a new syntactic class of *name-symbols* a, b, \dots different from ordinary variables x, y, z, \dots . Like variables, such name-symbols may be bound (by \mathbf{I}), but unlike variables, two distinct name-symbols always denote distinct name values. As explained in our previous paper [2], name-symbols can be used to construct ground terms, which is convenient from the perspective of studying Herbrand models and consistency. In place of slices, we introduce contexts that encode information about freshness as well as identifying the types of variables and name-symbols. Specifically, contexts $\Sigma \# a:\nu$ may be formed by adjoining a *fresh name-symbol* a which is also assumed to be semantically fresh for any value mentioned in Σ . Our rules for \mathbf{I} (Figure 2(NL^{\Rightarrow})) are in the spirit of the original rules and are very simple.

Besides presenting the sequent calculus and proving structural properties such as cut-elimination, we verify that NL^{\Rightarrow} and Pitts' axiomatization NL are equivalent. We also present a syntactic proof of the consistency of the nonlogical rules, which together with cut-elimination implies consistency of the whole system.

The structure of this article is as follows: Section 3 presents the sequent calculus NL^{\Rightarrow} along with proofs of structural properties. Section 4 discusses several applications, including proofs of consistency and equivalence of NL^{\Rightarrow} to NL . Section 5 concludes.

This article builds upon prior work by Gabbay and Cheney [7] and Gabbay [9], which introduced sequent and natural-deduction calculi for nominal logic, based on slices. The closest-related prior publication is Cheney [1], which introduced a single-conclusion, intuitionistic version of NL^{\Rightarrow} with the simpler rules for \mathbf{I} -quantifiers shown above. This article generalizes the approach taken there and provides detailed proofs of the main results, along with proofs of new results including equivalence to classical nominal logic.

3 Sequent Calculus

3.1 Syntax

The types τ , terms t , and formulas φ of NL^{\Rightarrow} are generated by the following grammar:

$$\begin{array}{ll}
\tau, \sigma & ::= \delta \mid \nu \mid \langle \nu \rangle \tau \\
t, u & ::= x \mid a \mid c \mid f(\bar{t}) \quad \parallel (ab) \cdot t \mid \langle a \rangle t \\
\varphi, \psi & ::= \top \mid \perp \mid p(\bar{t}) \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \varphi \supset \psi \mid \forall x:\tau.\varphi \mid \exists x:\tau.\varphi \mid \mathbf{I}a:\nu.\varphi \quad \parallel t \approx u \mid t \# u
\end{array}$$

The constructs to the right of \parallel are syntactic sugar that are definable in terms of the core language as explained below; we list them in the grammar for ease of reference. The base types are datatypes δ and name-types ν ;

$$\begin{aligned}
FV(x) &= \{x\} \\
FV(\mathbf{a}) &= \emptyset \\
FV(Qx:\sigma.\varphi) &= FV(\varphi) - \{x\} & (Q \in \{\forall, \exists\}) \\
FV(\mathbf{Ia}:\nu.\varphi) &= FV(\varphi) \\
\\
FN(x) &= \emptyset \\
FN(\mathbf{a}) &= \{\mathbf{a}\} \\
FN(Qx:\sigma.\varphi) &= FN(\varphi) & (Q \in \{\forall, \exists\}) \\
FN(\mathbf{Ia}:\nu.\varphi) &= FN(\varphi) - \{\mathbf{a}\} \\
\\
F\alpha(c) = F\alpha(\top) = F\alpha(\perp) &= \emptyset \\
F\alpha(f(\bar{t})) = F\alpha(p(\bar{t})) &= \bigcup F\alpha(t_i) \\
F\alpha(\varphi \circ \psi) &= F\alpha(\varphi) \cup F\alpha(\psi) & (\circ \in \{\wedge, \vee, \supset\}) \\
F\alpha((a\ b) \cdot t) &= F\alpha(a) \cup F\alpha(b) \cup F\alpha(t) \\
F\alpha(\langle a \rangle t) &= F\alpha(a) \cup F\alpha(t) \\
\\
FVN(t) &= FV(t) \cup FN(t)
\end{aligned}$$

Figure 3: Free variables and names (note $F\alpha$ stands for either FV or FN)

$$\begin{aligned}
(a\ b) \cdot \varphi &= \varphi & (\varphi \in \{\top, \perp\}) \\
(a\ b) \cdot p(\bar{t}) &= p((a\ b) \cdot \bar{t}) \\
(a\ b) \cdot \varphi \circ \psi &= (a\ b) \cdot \varphi \circ (a\ b) \cdot \psi & (\circ \in \{\wedge, \vee, \supset\}) \\
(a\ b) \cdot Qx:\sigma.\varphi &= Qx:\sigma.(a\ b) \cdot \varphi & (Q \in \{\forall, \exists\}, x \notin FV(a) \cup FV(b)) \\
(a\ b) \cdot \mathbf{Ia}:\nu.\varphi &= \mathbf{Ia}:\nu.(a\ b) \cdot \varphi & (\mathbf{a} \notin FN(a) \cup FN(b))
\end{aligned}$$

Figure 4: Swapping for formulas

additional types are formed using the abstraction type constructor. Terms are first-order, with variables x, y are drawn from a countably infinite set \mathbb{V} ; also, name-symbols \mathbf{a}, \mathbf{b} are drawn from a countably infinite set \mathbb{A} disjoint from \mathbb{V} . The letters a, b are typically used for terms of some name-type ν . Negation and logical equivalence are defined as follows:

$$\neg\varphi = (\varphi \supset \perp) \quad \varphi \iff \psi = (\varphi \supset \psi) \wedge (\psi \supset \varphi)$$

We assume given a signature that maps constant symbols c to types δ , function symbols f to sorts $\tau_1, \dots, \tau_n \rightarrow \delta$, and relation symbols to sorts $\tau_1, \dots, \tau_n \rightarrow o$, and containing at least the following declarations:

$$\begin{aligned}
\text{swap}_{\nu\tau} &: \nu, \nu, \tau \rightarrow \tau & \text{abs}_{\nu\tau} &: \nu, \tau \rightarrow \langle \nu \rangle \tau \\
\text{eq}_{\tau} &: \tau, \tau \rightarrow o & \text{fresh}_{\nu, \tau} &: \nu, \tau \rightarrow o
\end{aligned}$$

for name-types ν and types τ . The subscripts are dropped when clear from context. The notations $(a\ b) \cdot t$ and $\langle t \rangle u$ are syntactic sugar for the terms $\text{swap}(a, b, t)$ and $\text{abs}(t, u)$, respectively. Likewise, $t \approx u$ and $t \# u$ are syntactic sugar for $\text{eq}(t, u)$ and $\text{fresh}(t, u)$, respectively. The functions $FV(\cdot)$, $FN(\cdot)$, $FVN(\cdot)$ calculate the sets of free variables, name-symbols, or both variables and name-symbols of a term or formula (see Figure 3). We lift the swapping operation to formulas as shown in Figure 4.

The *typing contexts* used in NL^{\Rightarrow} are generated by the grammar:

$$\Sigma ::= \cdot \mid \Sigma, x:\tau \mid \Sigma \# \mathbf{a}:\nu$$

We often write $\cdot, x:\tau$ and $\cdot \# \mathbf{a}:\nu$ to $x:\tau$ and $\mathbf{a}:\nu$ respectively. We write ω for a term that may be either a name-symbol \mathbf{a} or a variable x . The $\Sigma \# \mathbf{a}:\nu$ binding indicates that \mathbf{a} is a name of type ν and is assumed to be fresh with respect to all names and variables in Σ . We write $\omega:\tau \in \Sigma$ if the binding $\omega:\tau$ is present in Σ . We write Σ, Σ' for the result of concatenating two contexts such that $FVN(\Sigma) \cap FVN(\Sigma') = \emptyset$.

We write $\Sigma \vdash t : \tau$ or $\Sigma \vdash \varphi : o$ to indicate that t is a well-formed term of type τ or φ is a well-formed formula. From the point of view of typechecking, the additional freshness information in the context is

$$\begin{array}{c}
\frac{c : \delta}{\Sigma \vdash c : \delta} \quad \frac{f : \tau_1, \dots, \tau_n \rightarrow \delta \quad \Sigma \vdash t_i : \tau_i}{\Sigma \vdash f(\vec{t}) : \delta} \quad \frac{\omega : \tau \in \Sigma}{\Sigma \vdash \omega : \tau} \quad \frac{}{\Sigma \vdash \top : o} \\
\frac{}{\Sigma \vdash \perp : o} \quad \frac{\Sigma \vdash \varphi, \psi : o \quad (o \in \{\wedge, \vee, \supset\})}{\Sigma \vdash \varphi \circ \psi : o} \quad \frac{\Sigma \vdash a : \nu \quad \Sigma \vdash t : \tau}{\Sigma \vdash a \# t : o} \quad \frac{\Sigma \vdash t, u : \tau}{\Sigma \vdash t \approx u : o} \\
\frac{\Sigma, x : \tau \vdash \varphi : o}{\Sigma \vdash \forall x : \tau. \varphi : o} \quad \frac{\Sigma, x : \tau \vdash \varphi : o}{\Sigma \vdash \exists x : \tau. \varphi : o} \quad \frac{\Sigma \# a : \nu \vdash \varphi : o}{\Sigma \vdash \mathbb{I} a : \nu. \varphi : o}
\end{array}$$

Figure 5: Well-formedness rules

irrelevant. The rules for typechecking (shown in Figure 5) are standard, except for the rules for freshness and the \mathbb{I} -quantifier. Quantification using \forall and \exists is only allowed over types not mentioning o ; \mathbb{I} -quantification is only allowed over name-types.

Definition 3.1. Let $Tm_\Sigma = \{t \mid \Sigma \vdash t : \tau\}$ be the set of well-formed terms in context Σ .

- We associate a set of freshness formulas $|\Sigma|$ to each context Σ as follows:

$$|\cdot| = \emptyset \quad |\Sigma, x : \tau| = |\Sigma| \quad |\Sigma \# a : \nu| = |\Sigma| \cup \{a \# t \mid t \in Tm_\Sigma\}$$

For example, $a \# x$, $b \# a$ and $b \# f(x, y)$ are in $|x : \tau \# a : \nu, y : \sigma \# b : \nu'|$.

- We say that Σ' is stronger than Σ ($\Sigma \leq \Sigma'$) if $Tm_\Sigma \subseteq Tm_{\Sigma'}$ and $|\Sigma| \subseteq |\Sigma'|$. For example, $a : \nu, x : \tau \leq x : \tau \# a : \nu, y : \sigma$.
- We say that $a : \nu \in \Sigma$ if $\Sigma = \Sigma' \# a : \nu, \Sigma''$ for some contexts Σ', Σ'' and similarly $x : \tau \in \Sigma$ means that $\Sigma = \Sigma', x : \tau, \Sigma''$ for some contexts Σ', Σ'' .
- We say that a is fresh for Σ if a is not among the names appearing in Σ ; we write $a \notin \Sigma$ to indicate that this is the case. Similarly, we write $x \notin \Sigma$ to indicate that variable x does not appear in Σ .

The following routine properties hold:

Lemma 3.2 (Term Weakening). *If $\Sigma \vdash t : \tau$ and $\Sigma \leq \Sigma'$ then $\Sigma' \vdash t : \tau$.*

Lemma 3.3 (Term Substitution). *If $\Sigma \vdash t : \tau$ and $\Sigma, x : \tau, \Sigma' \vdash u : \tau'$ then $\Sigma, \Sigma' \vdash u[t/x] : \tau'$.*

3.2 The Rules

Judgments are of the form $\Sigma; \Gamma \Rightarrow \Delta$, where Σ is a typing context and Γ, Δ are multisets of formulas. We define classical and intuitionistic versions of NL^{\Rightarrow} . *Classical* NL^{\Rightarrow} is based on the classical sequent calculus **G3c** (see Figure 6). The new rules defining NL^{\Rightarrow} are defined in Figures 7 and 8. NL^{\Rightarrow} includes two additional *logical rules*, $\mathbb{I}L$ and $\mathbb{I}R$, as already shown in Figure 2. In addition, NL^{\Rightarrow} includes several new *nonlogical rules* defining the properties of swapping, equality, freshness and abstraction. (The standard rules involving equality in Figure 6 are also considered nonlogical rules.)

Many of the nonlogical rules correspond to first-order universal axioms of nominal logic (Figure 7), which may be incorporated into sequent systems in a uniform fashion using the Ax rule schema without affecting cut-elimination [14]. Here, we write an axiom of the form $P_1 \wedge \dots \wedge P_n \supset Q_1 \vee \dots \vee Q_m$ as $\bigwedge \overline{P} \supset \bigvee \overline{Q}$. To illustrate, the instances of this scheme for axioms F_3 and F_4 are:

$$\frac{}{\Sigma; \Gamma, a \# a \Rightarrow \Delta} F_3 \quad \frac{\Sigma; \Gamma, a \# b \Rightarrow \Delta \quad \Sigma; \Gamma, a \approx b \Rightarrow \Delta}{\Sigma; \Gamma \Rightarrow \Delta} F_4$$

The key point of this treatment of nonlogical rules is that they act only on the hypothesis set Γ , so they do not introduce new principal cut cases in the proof of cut-elimination.

$$\begin{array}{c}
\frac{}{\Sigma; \Gamma, p(\bar{t}) \Rightarrow p(\bar{t}), \Delta} \text{hyp} \\
\frac{}{\Sigma; \Gamma \Rightarrow \top, \Delta} \top R \\
\frac{\Sigma; \Gamma \Rightarrow \varphi, \Delta \quad \Sigma; \Gamma \Rightarrow \psi, \Delta}{\Sigma; \Gamma \Rightarrow \varphi \wedge \psi, \Delta} \wedge R \\
\frac{\Sigma; \Gamma \Rightarrow \varphi_1, \varphi_2, \Delta}{\Sigma; \Gamma \Rightarrow \varphi_1 \vee \varphi_2, \Delta} \vee R \\
\frac{\Sigma; \Gamma, \varphi \Rightarrow \psi, \Delta}{\Sigma; \Gamma \Rightarrow \varphi \supset \psi, \Delta} \supset R \\
\frac{\Sigma, x:\sigma; \Gamma \Rightarrow \varphi, \Delta \quad (x \notin \Sigma)}{\Sigma; \Gamma \Rightarrow \forall x:\sigma.\varphi, \Delta} \forall R \\
\frac{\Sigma \vdash t : \sigma \quad \Sigma; \Gamma \Rightarrow \exists x:\sigma.\varphi, \varphi\{t/x\}, \Delta}{\Sigma; \Gamma \Rightarrow \exists x:\sigma.\varphi, \Delta} \exists R \\
\frac{\Sigma; \Gamma, t \approx t \Rightarrow \Delta}{\Sigma; \Gamma \Rightarrow \Delta} \approx R \\
\frac{}{\Sigma; \Gamma, \perp \Rightarrow \Delta} \perp L \\
\frac{\Sigma; \Gamma, \varphi_1, \varphi_2 \Rightarrow \Delta}{\Sigma; \Gamma, \varphi_1 \wedge \varphi_2 \Rightarrow \Delta} \wedge L \\
\frac{\Sigma; \Gamma, \varphi \Rightarrow \Delta \quad \Gamma, \psi \Rightarrow \Delta}{\Sigma; \Gamma, \varphi \vee \psi \Rightarrow \Delta} \vee L \\
\frac{\Sigma; \Gamma \Rightarrow \varphi, \Delta \quad \Sigma; \Gamma, \psi \Rightarrow \Delta}{\Sigma; \Gamma, \varphi \supset \psi \Rightarrow \Delta} \supset L \\
\frac{\Sigma \vdash t : \sigma \quad \Sigma; \Gamma, \forall x:\sigma.\varphi, \varphi\{t/x\} \Rightarrow \Delta}{\Sigma; \Gamma, \forall x:\sigma.\varphi \Rightarrow \Delta} \forall L \\
\frac{\Sigma, x:\sigma; \Gamma, \varphi \Rightarrow \Delta \quad (x \notin \Sigma)}{\Sigma; \Gamma, \exists x:\sigma.\varphi \Rightarrow \Delta} \exists L \\
\frac{\Sigma; \Gamma, t \approx u, P(t), P(u) \Rightarrow \Delta}{\Sigma; \Gamma, t \approx u, P(t) \Rightarrow \Delta} \approx S
\end{array}$$

Figure 6: Classical first-order equational sequent calculus (**G3c**)

The remaining nonlogical rules are as follows. Rule A_2 expresses an invertibility property for abstractions: two abstractions are equal only if they are structurally equal or equal by virtue of A_1 . A_3 says that all values of abstraction type are formed using the abstraction function symbol. The F rule expresses the freshness principle: that a name fresh for a given context may always be chosen. It is important to note that the fresh name chosen in F may be of any name type ν , and thus, all name types are inhabited; however, base data types δ could be empty, and an abstraction type $\langle \nu \rangle \tau$ is inhabited if and only if τ is. Finally, the $\Sigma\#$ rule allows freshness information to be extracted from the context Σ . It states that in context Σ , any constraint in $|\Sigma|$ is valid.

Remark 3.4. Although we have motivated some choices in NL^{\Rightarrow} in terms of proof-search behavior based on experience with α Prolog, some rules, such as A_3 and $\Sigma\#$, do not have particularly pleasant proof-search properties. It is fair to say that NL^{\Rightarrow} addresses only the proof search complexity arising from the \forall -quantifier and (to some extent) freshness but does not help very much with the complexity arising from equational/freshness reasoning. In α Prolog, special cases of these problems are dealt with using nominal unification and freshness constraint solving; in this paper we aim to deal with full nominal logic.

The naming of the nonlogical rule groups corresponds to that used by Pitts: the axioms are divided into groups for swapping (S), equivariance (E), freshness (F), and abstraction (A). The (Q) axiom is replaced by the logical rules $\forall L$ and $\exists R$.

3.3 Structural Properties

Figure 9 lists some additional rules, including weakening, contraction, general form of hypothesis and equivariance rules, and cut. We will now prove their admissibility. Note that these rules are not part of the definition of NL^{\Rightarrow} , and so in proving admissibility, it suffices to consider only derivations using the core rules introduced in Section 3.2.

We now list some routinely-verified syntactic properties of NL^{\Rightarrow} . We write $\vdash_n J$ to indicate that judgment J has a derivation of height at most n .

Lemma 3.5 (Weakening). *If $\vdash_n \Sigma; \Gamma \Rightarrow \Delta$ is derivable then so is $\vdash_n \Sigma; \Gamma, \varphi \Rightarrow \Delta$. Similarly, $\vdash_n \Sigma; \Gamma \Rightarrow \Delta, \varphi$.*

Lemma 3.6 (Context Weakening). *If $\vdash_n \Sigma; \Gamma \Rightarrow \Delta$ and $\Sigma \leq \Sigma'$ then $\vdash_n \Sigma'; \Gamma \Rightarrow \Delta$.*

Lemma 3.7 (Substitution). *If $\vdash_n \Sigma \vdash t : \tau$ and $\Sigma, x:\tau, \Sigma'; \Gamma \Rightarrow \Delta$ then $\vdash_n \Sigma, \Sigma'; \Gamma[t/x] \Rightarrow \Delta[t/x]$.*

(S_1)	$(a a) \cdot x \approx x$	(F_1)	$a \# x \wedge b \# x \supset (a b) \cdot x \approx x$
(S_2)	$(a b) \cdot (a b) \cdot x \approx x$	(F_2)	$a \# b \quad (a : \nu, b : \nu', \nu \neq \nu')$
(S_3)	$(a b) \cdot a \approx b$	(F_3)	$a \# a \supset \perp$
(E_1)	$(a b) \cdot c \approx c$	(F_4)	$a \# b \vee a \approx b$
(E_2)	$(a b) \cdot f(\bar{t}) \approx f((a b) \cdot \bar{t})$	(A_1)	$a \# y \wedge x \approx (a b) \cdot y \supset \langle a \rangle x \approx \langle b \rangle y$
(E_3)	$p(\bar{t}) \supset p((a b) \cdot \bar{t})$		

Figure 7: Equational and freshness axioms

$\frac{\Sigma; \Gamma, \bigwedge \bar{P}, Q_1 \Rightarrow \Delta \quad \dots \quad \Sigma; \Gamma, \bigwedge \bar{P}, Q_n \Rightarrow \Delta}{\Sigma; \Gamma, \bigwedge \bar{P} \Rightarrow \Delta} Ax$	$\bigwedge \bar{P} \supset \bigvee \bar{Q} \text{ an axiom instance in Figure 7}$
$\frac{\Sigma; \Gamma, \langle a \rangle t \approx \langle b \rangle u, a \approx b, t \approx u \Rightarrow \Delta \quad \Sigma; \Gamma, \langle a \rangle t \approx \langle b \rangle u, a \# u, t \approx (a b) \cdot u \Rightarrow \Delta}{\Sigma; \Gamma, \langle a \rangle t \approx \langle b \rangle u \Rightarrow \Delta} A_2$	
$\frac{\Sigma \vdash t : \langle \nu \rangle \sigma \quad \Sigma, a : \nu, x : \sigma; \Gamma, t \approx \langle a \rangle x \Rightarrow \Delta \quad (a, x \notin \Sigma)}{\Sigma; \Gamma \Rightarrow \Delta} A_3$	
$\frac{\Sigma \# a : \nu; \Gamma \Rightarrow \Delta \quad (a \notin \Sigma)}{\Sigma; \Gamma \Rightarrow \Delta} F$	$\frac{\Sigma; \Gamma, a \# t \Rightarrow \Delta \quad (a \# t \in \Sigma)}{\Sigma; \Gamma \Rightarrow \Delta} \Sigma \#$
$\frac{\Sigma \# a : \nu; \Gamma \Rightarrow \varphi, \Delta \quad (a \notin \Sigma)}{\Sigma; \Gamma \Rightarrow \mathcal{I}a : \nu. \varphi, \Delta} \mathcal{I}R$	$\frac{\Sigma \# a : \nu; \Gamma, \varphi \Rightarrow \Delta \quad (a \notin \Sigma)}{\Sigma; \Gamma, \mathcal{I}a : \nu. \varphi \Rightarrow \Delta} \mathcal{I}L$

Figure 8: Nonlogical and \mathcal{I} -quantifier rules

$\frac{\Sigma; \Gamma \Rightarrow \Delta}{\Sigma; \Gamma, \varphi \Rightarrow \Delta} W$	$\frac{}{\Sigma; \Gamma, \varphi \Rightarrow \varphi, \Delta} hyp^*$	$\frac{\Sigma; \Gamma \Rightarrow \varphi, \Delta \quad \Sigma; \Gamma', \varphi \Rightarrow \Delta'}{\Sigma; \Gamma, \Gamma' \Rightarrow \Delta, \Delta'} cut$
$\frac{\Sigma; \Gamma, \varphi, \varphi \Rightarrow \Delta}{\Sigma; \Gamma, \varphi \Rightarrow \Delta} C$	$\frac{\Sigma; \Gamma, (a b) \cdot \varphi \Rightarrow \Delta}{\Sigma; \Gamma, \varphi \Rightarrow \Delta} EVL$	$\frac{\Sigma; \Gamma \Rightarrow (a b) \cdot \varphi, \Delta}{\Sigma; \Gamma \Rightarrow \Delta, \varphi} EVR$

Figure 9: Some admissible rules of $NL \Rightarrow$

Proof. The interesting cases are for the new rules, specifically, nonlogical rules, $\mathcal{I}L$, and $\mathcal{I}R$. All of the nonlogical rules are closed under substitution; in particular, for $\Sigma \#$ we have $a \# u \in |\Sigma, x, \Sigma'|$ then $a \# u[t/x] \in |\Sigma, \Sigma'|$.

For F we have a derivation

$$\frac{\Sigma, x : \tau, \Sigma' \# a : \nu; \Gamma \Rightarrow \Delta}{\Sigma, x : \tau, \Sigma'; \Gamma \Rightarrow \Delta} F$$

By induction we have $\Sigma, \Sigma' \# a : \nu; \Gamma[t/x] \Rightarrow \Delta[t/x]$, so we can use F again to derive $\Sigma, \Sigma'; \Gamma[t/x] \Rightarrow \Delta[t/x]$. This requires the observation that since $a \notin \Sigma$ and $\Sigma \vdash t : \tau$, we must have $a \notin FN(t)$. The proofs for $\mathcal{I}L$ and $\mathcal{I}R$ are similar, requiring the additional observation that $(\mathcal{I}a : \nu. \varphi)[t/x] = \mathcal{I}a : \nu. (\varphi[t/x])$ since $a \notin FN(t)$. \square

The remaining structural transformations do not preserve the height of derivations. However, they do preserve the logical height of the derivation, which is defined as follows.

Definition 3.8. The *logical height* of a derivation is the maximum number of logical rules in any branch of the derivation. We write $\vdash_n^l J$ to indicate that J has a derivation of logical height $\leq n$.

Now we consider some structural properties specific to $NL \Rightarrow$. In the following, recall the definition of $(a b) \cdot \varphi$ given in Figure 4.

Lemma 3.9 (Admissibility of EVL , EVR). *The EVL and EVR rules*

$$\frac{\Sigma; \Gamma, (a b) \cdot \varphi \Rightarrow \Delta}{\Sigma; \Gamma, \varphi \Rightarrow \Delta} \text{EVL} \quad \frac{\Sigma; \Gamma \Rightarrow (a b) \cdot \varphi, \Delta}{\Sigma; \Gamma \Rightarrow \varphi, \Delta} \text{EVR}$$

where φ is an arbitrary formula in Tm_{Σ} , are admissible; if the antecedent of EVL or EVR is derivable, then the respective conclusion has a derivation of the same logical height.

Proof. We proceed by induction on the lexicographic product of logical height and total height to show that if the hypothesis of an instance of EVL or EVR has a derivation then the conclusion of the respective rule has a derivation of the same logical height.

We first consider EVL . The only interesting cases are when $(a b) \cdot \varphi$ is principal on the left, otherwise the induction step is straightforward. Furthermore, only the cases for hyp and $\supset L$ are nontrivial.

If the derivation is of the form

$$\frac{}{\Sigma; \Gamma, (a b) \cdot A \Rightarrow (a b) \cdot A, \Delta} \text{hyp}$$

then we may derive $\Gamma, A \Rightarrow (a b) \cdot A, \Delta$ as follows:

$$\frac{\frac{}{\Sigma; \Gamma, (a b) \cdot A \Rightarrow (a b) \cdot A, \Delta} \approx, \text{hyp}}{\Sigma; \Gamma, A \Rightarrow (a b) \cdot A, \Delta} E_3$$

This derivation has the same logical height, 1, as the first.

If the derivation is of the form

$$\frac{\Sigma; \Gamma, (a b) \cdot \varphi \supset (a b) \cdot \psi \Rightarrow (a b) \cdot \varphi, \Delta \quad \Sigma; \Gamma, (a b) \cdot \psi \Rightarrow \Delta}{\Sigma; \Gamma, (a b) \cdot \varphi \supset (a b) \cdot \psi \Rightarrow \Delta} \supset L$$

then using the admissibility of EVR and EVL on the left and EVR on the right (on derivations of smaller logical height) we obtain

$$\frac{\frac{\Sigma; \Gamma, (a b) \cdot \varphi \supset (a b) \cdot \psi \Rightarrow (a b) \cdot \varphi, \Delta}{\Sigma; \Gamma, \varphi \supset \psi \Rightarrow \varphi, \Delta} \text{EVL, EVR} \quad \frac{\Sigma; \Gamma, (a b) \cdot \psi \Rightarrow \Delta}{\Sigma; \Gamma, \psi \Rightarrow \Delta} \text{EVL}}{\Sigma; \Gamma, \varphi \supset \psi \Rightarrow \Delta} \supset L$$

This transformation is obviously logical height-preserving by induction.

For EVR , the interesting cases are those for hyp and $\supset R$ where $(a b) \cdot \varphi$ is principal on the right. Suppose the derivation is of the form

$$\frac{}{\Sigma; \Gamma, (a b) \cdot A \Rightarrow (a b) \cdot A, \Delta} \text{hyp}$$

Then we can derive

$$\frac{\frac{}{\Sigma; \Gamma, (a b) \cdot (a b) \cdot A \Rightarrow A, \Delta} \approx, \text{hyp}}{\Sigma; \Gamma, (a b) \cdot A \Rightarrow A, \Delta} E_3$$

This derivation has the same logical height, 1, as the first.

If the derivation is of the form

$$\frac{\Sigma; \Gamma, (a b) \cdot \varphi \Rightarrow (a b) \cdot \psi, \Delta}{\Sigma; \Gamma \Rightarrow (a b) \cdot \varphi \supset (a b) \cdot \psi, \Delta} \supset R$$

then since EVL and EVR are admissible for all subderivations of this derivation, by induction we can derive

$$\frac{\frac{\Sigma; \Gamma, (a b) \cdot \varphi \Rightarrow (a b) \cdot \psi, \Delta}{\Sigma; \Gamma, \varphi \Rightarrow \psi, \Delta} \text{EVL, EVR}}{\Sigma; \Gamma \Rightarrow \varphi \supset \psi, \Delta} \supset R$$

This transformation is obviously logical height-preserving by induction. \square

Lemma 3.10 (Swapping Fresh Names). *Suppose $\Sigma \# a : \nu \vdash \varphi(a) : o$ and $b \notin FN(\Sigma \# a : \nu)$. Then the rule*

$$\frac{\Sigma \# a : \nu \# b : \nu ; \Gamma, \varphi(b) \Rightarrow \Delta}{\Sigma \# a : \nu \# b : \nu ; \Gamma, \varphi(a) \Rightarrow \Delta}$$

is admissible using nonlogical axioms only.

Proof. Let $\bar{x} = FV(\Sigma)$. The derivation is as follows:

$$\frac{\frac{\frac{\Sigma \# a : \nu \# b : \nu ; \Gamma, a \# \bar{x}, b \# \bar{x}, \varphi(b) \Rightarrow \Delta}{\Sigma \# a : \nu \# b : \nu ; \Gamma, a \# \bar{x}, b \# \bar{x}, (a \ b) \cdot \varphi(a) \Rightarrow \Delta} Ax}{\Sigma \# a : \nu \# b : \nu ; \Gamma, a \# \bar{x}, b \# \bar{x}, \varphi(a) \Rightarrow \Delta} EVL}{\Sigma \# a : \nu \# b : \nu ; \Gamma, \varphi(a) \Rightarrow \Delta} \Sigma \#$$

where F_1 and equational reasoning is used repeatedly to show that $(a \ b) \cdot \varphi(a) \supset \varphi(b)$. \square

Lemma 3.11 (Admissibility of hyp^*). *The hyp^* rule*

$$\frac{}{\Sigma ; \Gamma, \varphi \Rightarrow \varphi, \Delta} hyp^*$$

where φ is an arbitrary formula in Tm_Σ , is admissible.

Proof. The proof is by induction on the construction of φ . The cases for the ordinary connectives of first-order logic are standard. The case for $\varphi = \mathbf{I}a : \nu. \varphi'$ is as follows. By induction, we may assume that $\Sigma \# a : \nu \# b : \nu ; \Gamma, \varphi(b) \Rightarrow \varphi(b), \Delta$ is derivable. We derive

$$\frac{\frac{\frac{\Sigma \# a : \nu \# b : \nu ; \Gamma, \varphi(b) \Rightarrow \varphi(b), \Delta}{\Sigma \# a : \nu \# b : \nu ; \Gamma, \varphi(a) \Rightarrow \varphi(b), \Delta} \text{Lemma 3.10}}{\Sigma \# a : \nu ; \Gamma, \varphi(a) \Rightarrow \mathbf{I}a : \nu. \varphi', \Delta} \mathbf{I}R}{\Sigma ; \Gamma, \mathbf{I}a : \nu. \varphi' \Rightarrow \mathbf{I}a : \nu. \varphi', \Delta} \mathbf{I}L$$

Using the induction hypothesis, the judgment $\Sigma \# a : \nu \# b : \nu ; \Gamma, \varphi(b) \Rightarrow \varphi(b), \Delta$ is derivable, since it is an instance of hyp^* with a smaller principal formula. \square

Lemma 3.12 (Inversion). *The $\wedge L, \vee L, \supset L, \exists L, \forall R, \mathbf{I}L$, and $\mathbf{I}R$ rules are invertible; that is,*

1. *If $\vdash_n^l \Sigma ; \Gamma, \varphi \wedge \psi \Rightarrow \Delta$ then $\vdash_n^l \Sigma ; \Gamma, \varphi, \psi \Rightarrow \Delta$.*
2. *If $\vdash_n^l \Sigma ; \Gamma, \varphi \vee \psi \Rightarrow \Delta$ then $\vdash_n^l \Sigma ; \Gamma, \varphi \Rightarrow \Delta$ and $\vdash_n^l \Sigma ; \Gamma, \psi \Rightarrow \Delta$.*
3. *If $\vdash_n^l \Sigma ; \Gamma, \varphi \supset \psi \Rightarrow \Delta$ then $\vdash_n^l \Sigma ; \Gamma, \psi \Rightarrow \Delta$.*
4. *If $\vdash_n^l \Sigma ; \Gamma, \exists x. \varphi \Rightarrow \Delta$ then $\vdash_n^l \Sigma, y ; \Gamma, \varphi[y/x] \Rightarrow \Delta$.*
5. *If $\vdash_n^l \Sigma ; \Gamma \Rightarrow \Delta, \forall x. \varphi$ then $\vdash_n^l \Sigma, y ; \Gamma \Rightarrow \Delta, \varphi[y/x]$.*
6. *If $\vdash_n^l \Sigma ; \Gamma, \mathbf{I}a : \nu. \varphi \Rightarrow \Delta$ then $\vdash_n^l \Sigma \# a : \nu ; \Gamma, \varphi \Rightarrow \Delta$ for fresh $a \notin \Sigma$.*
7. *If $\vdash_n^l \Sigma ; \Gamma \Rightarrow \Delta, \mathbf{I}a : \nu. \varphi$ then $\vdash_n^l \Sigma \# a : \nu ; \Gamma \Rightarrow \Delta, \varphi$ for fresh $a \notin \Sigma$.*

Proof. The proofs for the rules $\wedge L, \vee L, \supset L, \exists L, \forall R$ are similar to those for the systems **G3c** and **G3im** [14].

For $\mathbf{I}L$, the proof is by induction on the height of the derivation. Most cases are straightforward. Only cases such as $\forall R, \exists L, A_3, F$ that introduce variables or name-symbols into Σ are exceptions. We show the reasoning for $\forall R$.

If the derivation is of the form

$$\frac{\Sigma, x : \tau ; \Gamma, \mathbf{I}a : \nu. \varphi \Rightarrow \psi}{\Sigma ; \Gamma, \mathbf{I}a : \nu. \varphi \Rightarrow \forall x : \tau. \psi}$$

then using the induction hypothesis, we have $\Sigma, x : \tau \# b : \nu ; \Gamma, \varphi(b) \Rightarrow \psi$. Using structural weakening we have $\Sigma \# a : \nu, x : \tau \# b : \nu ; \Gamma, \varphi(b) \Rightarrow \psi$. Since a and b are fresh with respect to all terms in Tm_Σ , it is

straightforward to show that $\Sigma \# a : \nu, x : \tau \# b : \nu : \Gamma, (a \ b) \cdot \varphi(a) \Rightarrow \psi$. Thus, by equivariance, we can derive $\Sigma \# a : \nu, x : \tau \# b : \nu; \Gamma, \varphi(a) \Rightarrow \psi$. Now b is not mentioned in the sequent so using F we can derive $\Sigma \# a : \nu, x : \tau; \Gamma, \varphi(a) \Rightarrow \psi$, and using $\forall R$ we can derive $\Sigma \# a : \nu; \Gamma, \varphi(a) \Rightarrow \forall x : \tau. \psi$, as desired.

The proof for the invertibility of \mathcal{IR} is symmetric. \square

Lemma 3.13 (Contraction). *If $\vdash_n^l \Sigma; \Gamma, \varphi, \varphi \Rightarrow \Delta$ is derivable then so is $\vdash_n^l \Sigma; \Gamma, \varphi \Rightarrow \Delta$. Similarly, if $\vdash_n^l \Sigma; \Gamma \Rightarrow \Delta, \varphi, \varphi$ is derivable then $\vdash_n^l \Sigma; \Gamma \Rightarrow \Delta, \varphi$.*

Proof. The proof is by induction on the lexicographic product of logical height and total height. That is, the induction hypothesis applies to all derivations of smaller logical height and to all derivations of equal logical height but smaller total height. Most cases are similar to any standard proof. The only new cases involve nonlogical rules and $\mathcal{I}a : \nu. \varphi$. For the nonlogical rules it suffices to show that for each nonlogical rule that has a contractable instance, there is a nonlogical rule corresponding to the contraction. The only such rule is F_1 . If the derivation is of the form

$$\frac{\Sigma; \Gamma, a \# x, a \# x, (a \ a) \cdot x \approx x \Rightarrow \Delta}{\Sigma; \Gamma, a \# x, a \# x \Rightarrow \Delta} F_1$$

then we can transform the derivation to

$$\frac{\Sigma; \Gamma, a \# x, (a \ a) \cdot x \approx x \Rightarrow \Delta}{\Sigma; \Gamma, a \# x \Rightarrow \Delta} S_1$$

Most of the remaining cases are standard. The only interesting new case is when the contracted formula is derived using \mathcal{IL} :

$$\frac{\Sigma \# a : \nu; \Gamma, \varphi(a), \mathcal{I}b : \nu. \varphi(b) \Rightarrow \Delta}{\Sigma; \Gamma, \mathcal{I}a : \nu. \varphi(a), \mathcal{I}b : \nu. \varphi(b) \Rightarrow \Delta} \mathcal{IL}$$

Then using inversion we have $\vdash_{n-1}^l \Sigma \# a : \nu \# b : \nu : \Gamma, \varphi(a), \varphi(b) \Rightarrow \Delta$. Now using nonlogical rules we can derive $\vdash_{n-1}^l \Sigma \# a : \nu \# b : \nu; \Gamma, \varphi(a), \varphi(a) \Rightarrow \Delta$. Then using the induction hypothesis we have $\vdash_{n-1}^l \Sigma \# a : \nu \# b : \nu; \Gamma, \varphi(a) \Rightarrow \Delta$. Finally we can derive

$$\frac{\frac{\Sigma \# a : \nu \# b : \nu; \Gamma, \varphi(a) \Rightarrow \Delta}{\Sigma \# a : \nu; \Gamma, \varphi(a) \Rightarrow \Delta} F}{\Sigma; \Gamma, \mathcal{I}a : \nu. \varphi(a) \Rightarrow \Delta} \mathcal{IL}$$

The proof for right-contraction is symmetric, using the invertibility of \mathcal{IR} . \square

3.4 Cut-Elimination

As usual for sequent systems, the most important property to check to verify that the system is sensible is cut-elimination.

Lemma 3.14 (Admissibility of Cut). *If $\vdash \Sigma; \Gamma \Rightarrow \Delta, \varphi$ and $\vdash \Sigma'; \Gamma', \varphi \Rightarrow \Delta$ then $\vdash \Sigma; \Gamma, \Gamma' \Rightarrow \Delta, \Delta'$.*

Proof. Following the proof of cut-elimination for similar systems such as **G3c** or **G3im** of [14], we prove the lemma by induction on the structure of the cut-formula φ and then by a sub-induction on the sizes of the subderivations Π of $\Sigma; \Gamma \Rightarrow \Delta, \varphi$ and Π' of $\Sigma'; \Gamma', \varphi \Rightarrow \Delta$. Thus, for the induction hypothesis, we may assume that the lemma holds for any instances with a less complex cut-formula or for all instances with the same cut-formula but with a smaller derivation of one or the other of Π, Π' .

As in other proofs of cut-elimination for similar systems, there are four categories of cases:

- Base cases in which Π or Π' is an axiom or initial sequent.
- Left-commuting cases in which Π starts with a rule in which φ is not principal.
- Right-commuting cases in which Π' starts with a rule in which φ is not principal.
- Principal cases in which Π and Π' both start with a rule in which φ is principal.

All cases involving first-order rules exclusively are standard, and are shown in any standard proof of cut-elimination (e.g. [14] or [18]); their proofs rely upon the properties established in the previous section, including weakening, admissibility of hyp^* , contraction, and inversion. In addition, Negri and von Plato [14] showed that nonlogical rules of the form we consider can be added to sequent systems like **G3c** or **G3im** without damaging cut-elimination. Hence, it will suffice to consider only the new cases involving the \mathcal{U} -quantifier rules.

- Base cases: There are no new base cases.
- Left-commuting cases: There are two new cases in which Π begins with $\mathcal{U}R$ or $\mathcal{U}L$.

In the first case, we have

$$\frac{\Pi}{\frac{\Sigma\#a:\nu; \Gamma, \psi \Rightarrow \Delta, \varphi}{\Sigma; \Gamma, \mathcal{U}a:\nu.\psi \Rightarrow \Delta, \varphi} \mathcal{U}L}$$

where $a \notin \Sigma$. We can weaken Π' to obtain a derivation $W(\Pi')$ of $\Sigma\#a:\nu; \Gamma', \varphi \Rightarrow \Delta'$, and by induction, we have $\Sigma\#a:\nu; \Gamma, \psi, \Gamma' \Rightarrow \Delta, \Delta'$. Then we may derive $\Sigma; \Gamma, \mathcal{U}a:\nu.\psi, \Gamma' \Rightarrow \Delta, \Delta'$ using $\mathcal{U}L$.

In the second case, we have

$$\frac{\Pi}{\frac{\Sigma\#a:\nu; \Gamma \Rightarrow \Delta, \psi, \varphi}{\Sigma; \Gamma \Rightarrow \Delta, \mathcal{U}a:\nu.\psi, \varphi} \mathcal{U}R}$$

where $a \notin \Sigma$. We can weaken Π' to get $W(\Pi')$ deriving $\Sigma\#a:\nu; \Gamma', \varphi \Rightarrow \Delta'$ and then by induction obtain $\Sigma\#a:\nu; \Gamma', \Gamma \Rightarrow \Delta, \Delta', \psi$. Using $\mathcal{U}R$ we can derive $\Sigma : \Gamma', \Gamma \Rightarrow \Delta, \Delta', \mathcal{U}a:\nu.\psi$.

- Right-commuting cases. These cases are exactly symmetric to the left-commuting cases.

In the first case, we have

$$\frac{\Pi'}{\frac{\Sigma\#a:\nu; \Gamma', \varphi, \psi \Rightarrow \Delta'}{\Sigma; \Gamma', \varphi, \mathcal{U}a:\nu.\psi \Rightarrow \Delta'} \mathcal{U}L}$$

where $a \notin \Sigma$. We can weaken Π to obtain a derivation $W(\Pi)$ of $\Sigma\#a:\nu; \Gamma \Rightarrow \Delta, \varphi$, and by induction, we have $\Sigma\#a:\nu; \Gamma, \psi, \Gamma' \Rightarrow \Delta, \Delta'$. Then we may derive $\Sigma; \Gamma, \mathcal{U}a:\nu.\psi, \Gamma' \Rightarrow \Delta, \Delta'$ using $\mathcal{U}L$.

In the second case, we have

$$\frac{\Pi'}{\frac{\Sigma\#a:\nu; \Gamma', \varphi \Rightarrow \Delta', \psi}{\Sigma; \Gamma', \varphi \Rightarrow \Delta', \mathcal{U}a:\nu.\psi} \mathcal{U}R}$$

where $a \notin \Sigma$. We can weaken Π to obtain a derivation $W(\Pi)$ of $\Sigma\#a:\nu; \Gamma \Rightarrow \Delta, \varphi$ and then by induction obtain $\Sigma\#a:\nu; \Gamma', \Gamma \Rightarrow \Delta, \Delta', \psi$. Using $\mathcal{U}R$ we can derive $\Sigma; \Gamma', \Gamma \Rightarrow \Delta, \Delta', \mathcal{U}a:\nu.\psi$.

- Principal cases. In this case, both Π and Π' decompose the cut formula. The only new rule for decomposing formulas on the right is $\mathcal{U}R$, so the only new principal cut case is when we have

$$\frac{\frac{\Pi}{\Sigma\#a:\nu; \Gamma \Rightarrow \Delta, \varphi} \mathcal{U}R}{\Sigma; \Gamma \Rightarrow \Delta, \mathcal{U}a:\nu.\varphi} \mathcal{U}R \quad \frac{\frac{\Pi'}{\Sigma\#a:\nu; \Gamma', \varphi \Rightarrow \Delta'} \mathcal{U}L}{\Sigma; \Gamma', \mathcal{U}a:\nu.\varphi \Rightarrow \Delta'} \mathcal{U}L$$

for some $a \notin \Sigma$. By induction we have $\Sigma\#a:\nu; \Gamma, \Gamma' \Rightarrow \Delta, \Delta'$, and we may conclude $\Sigma; \Gamma, \Gamma' \Rightarrow \Delta, \Delta'$ by an application of the freshness rule.

This completes the proof. □

Theorem 3.15. *Any derivable NL^{\Rightarrow} sequent has a cut-free derivation; there is an algorithm for producing such derivations.*

Proof. Proof by induction on the number of cuts. Given a derivation using cut, we can always find an uppermost use of cut in the derivation tree and remove it. This reduces the number of cuts by one. □

$$\begin{array}{c}
\frac{\Sigma; \Gamma, \varphi \Rightarrow \psi}{\Sigma; \Gamma \Rightarrow \varphi \supset \psi, \Delta} \supset R \qquad \frac{\Sigma; \Gamma, \varphi \supset \psi \Rightarrow \varphi \quad \Sigma; \Gamma, \psi \Rightarrow \Delta}{\Sigma; \Gamma, \varphi \supset \psi \Rightarrow \Delta} \supset L \\
\frac{\Sigma, x:\sigma; \Gamma \Rightarrow \varphi \quad (x \notin \Sigma)}{\Sigma; \Gamma \Rightarrow \forall x:\sigma.\varphi, \Delta} \forall R \qquad \frac{\Sigma \vdash t : \sigma \quad \Sigma; \Gamma, \forall x:\sigma.\varphi, \varphi\{t/x\} \Rightarrow \Delta}{\Sigma; \Gamma, \forall x:\sigma.\varphi \Rightarrow \Delta} \forall L \\
\frac{\Sigma \vdash t : \sigma \quad \Sigma; \Gamma \Rightarrow \exists x:\sigma.\varphi, \varphi\{t/x\}, \Delta}{\Sigma; \Gamma \Rightarrow \exists x:\sigma.\varphi, \Delta} \exists R \qquad \frac{\Sigma, x:\sigma; \Gamma, \varphi \Rightarrow \Delta \quad (x \notin \Sigma)}{\Sigma; \Gamma, \exists x:\sigma.\varphi \Rightarrow \Delta} \exists L
\end{array}$$

Figure 10: Variant rules for the intuitionistic multiple-conclusion calculus (**G3im**)

3.5 Intuitionistic calculus

Intuitionistic NL^{\Rightarrow} (INL^{\Rightarrow}) is based on the multiple-conclusion intuitionistic calculus **G3im** [14], in which certain rules are restricted to discard alternative conclusions (see Figure 10). It is straightforward to show that all of the structural properties including cut-elimination hold for INL^{\Rightarrow} ; the same arguments as given above in the classical case apply. We will show in Section 4.3.2 that INL^{\Rightarrow} corresponds to a theory of first-order intuitionistic logic that is equivalent to Pitts' axiomatization in classical NL .

Theorem 3.16. *In INL^{\Rightarrow} , if $\Sigma; \Gamma \Rightarrow \Delta$ holds then there is a cut-free derivation of $\Sigma; \Gamma \Rightarrow \Delta$.*

It is also straightforward to show that INL^{\Rightarrow} is equivalent to a single-conclusion intuitionistic calculus, since the nonlogical and \mathcal{I} -quantifier rules preserve the single-conclusion property.

Theorem 3.17. *If $\Sigma; \Gamma \Rightarrow \Delta$ holds in INL^{\Rightarrow} then $\Sigma; \Gamma \Rightarrow \bigvee \Delta$ holds in the single-conclusion variant of INL^{\Rightarrow} .*

Proof. Most cases of the proof are analogous to the usual proof relating **G3i** and **G3im** [14]. The additional cases involve the nonlogical and \mathcal{I} -quantifier rules. Of these, the nonlogical rules are straightforward because nothing changes on the right-hand side of the sequent in these rules. The case for $\mathcal{I}L$ is also straightforward for the same reason.

We show the case for $\mathcal{I}R$. Suppose the derivation is of the form:

$$\frac{\Sigma \# a:\nu; \Gamma \Rightarrow \varphi, \Delta}{\Sigma; \Gamma \Rightarrow \mathcal{I}a:\nu.\varphi, \Delta} \mathcal{I}R$$

By induction on the subderivation we know that $\Sigma \# a:\nu; \Gamma \Rightarrow \varphi \vee \bigvee \Delta$. We reason as follows:

$$\frac{\frac{\frac{\Sigma \# a:\nu \# b:\nu; \Gamma, \varphi \Rightarrow \varphi[b/a]}{\Sigma \# a:\nu; \Gamma, \varphi \Rightarrow \mathcal{I}a:\nu.\varphi} \mathcal{I}R \quad \frac{\Sigma \# a:\nu; \Gamma, \bigvee \Delta \Rightarrow \bigvee \Delta}{\Sigma \# a:\nu; \Gamma, \bigvee \Delta \Rightarrow \mathcal{I}a:\nu.\varphi \vee \bigvee \Delta} hyp^*}{\Sigma \# a:\nu; \Gamma, \varphi \Rightarrow \mathcal{I}a:\nu.\varphi \vee \bigvee \Delta} \vee R_1 \quad \frac{\Sigma \# a:\nu; \Gamma, \bigvee \Delta \Rightarrow \bigvee \Delta}{\Sigma \# a:\nu; \Gamma, \bigvee \Delta \Rightarrow \mathcal{I}a:\nu.\varphi \vee \bigvee \Delta} \vee R_2}{\Sigma \# a:\nu; \Gamma \Rightarrow \varphi \vee \bigvee \Delta \quad \Sigma \# a:\nu; \Gamma, \varphi \vee \bigvee \Delta \Rightarrow \mathcal{I}a:\nu.\varphi \vee \bigvee \Delta} cut \quad \vee L}{\frac{\Sigma \# a:\nu; \Gamma \Rightarrow \varphi \vee \bigvee \Delta}{\Sigma; \Gamma \Rightarrow \mathcal{I}a:\nu.\varphi \vee \bigvee \Delta} F}$$

We can use the intuitionistic (single-conclusion) variant of Lemma 3.10 to conclude $\Sigma \# a:\nu \# b:\nu; \Gamma, \varphi \Rightarrow \varphi[b/a]$. \square

4 Applications

4.1 Syntactic Consistency

For pure first-order logic, cut-elimination immediately implies consistency, since by inspection of the rules there can be no shortest proof of $\cdot; \cdot \Rightarrow \perp$. However, in the presence of general nonlogical rules, only a weaker result holds. We say that an atomic formula is a *constraint* if it is an equality or freshness formula, and Γ is a constraint set if it contains only constraints.

Proposition 4.1. *If $\cdot; \cdot \Rightarrow \perp$ has a cut-free derivation, then it has one using only nonlogical rules, in which each sequent is of the form $\cdot; \Gamma \Rightarrow \perp$, where Γ is a constraint set.*

The proof is immediate by observing that only nonlogical rules are applicable to a derivation of $\cdot; \Gamma \Rightarrow \perp$ where Γ is a constraint set. In particular, note that the instance of the Ax rule scheme for $a \# a \supset \perp$ (axiom F_3) has no hypotheses:

$$\frac{}{\Sigma; \Gamma, a \# a \Rightarrow \Delta} F_3$$

so it is not necessary to allow \perp as a constraint (though this would not do any harm either).

This means that nominal logic is consistent if and only if the nonlogical rules are consistent. We know that classical nominal logic is consistent with respect to the semantics given by Pitts using nominal sets [15], and we will show in the next section that the two systems are equivalent, however, here we would like to give a direct syntactic proof that applies to both classical and intuitionistic variants of NL^\Rightarrow . To prove the consistency of the nonlogical rules, it is necessary to exhibit a model. We review how to define a Herbrand-style semantics in terms of the syntax of nominal terms (see e.g. Cheney [2] for more details).

Definition 4.2 (Syntactic Swapping, Equality and Freshness). Let Tm be the set of swapping-free nominal terms generated by the grammar

$$t ::= a \mid c \mid f(\bar{t}) \mid \langle a \rangle t$$

We define the *swapping function* on such terms as follows:

$$\begin{aligned} (a \ b) \cdot a &= b \\ (a \ b) \cdot b &= a \\ (a \ b) \cdot c &= c \quad (a, b \neq c) \\ (a \ b) \cdot c &= c \\ (a \ b) \cdot f(\bar{t}) &= f((a \ b) \cdot \bar{t}) \\ (a \ b) \cdot \langle c \rangle t &= \langle (a \ b) \cdot c, (a \ b) \cdot t \rangle \end{aligned}$$

We define the *freshness* relation on ground terms using the rules:

$$\frac{(a \neq b)}{a \# b} \quad \frac{}{a \# c} \quad \frac{a \# t_1 \ \dots \ a \# t_n}{a \# f(\bar{t})} \quad \frac{}{a \# \langle a \rangle t} \quad \frac{a \# t \ (a \neq b)}{a \# \langle b \rangle t}$$

The *nominal equality* relation is defined as follows:

$$\frac{}{a \approx a} \quad \frac{}{c \approx c} \quad \frac{t_1 \approx u_1 \ \dots \ t_n \approx u_n}{f(\bar{t}) \approx f(\bar{u})} \quad \frac{t \approx u}{\langle a \rangle t \approx \langle a \rangle u} \quad \frac{t \approx (a \ b) \cdot u \quad a \# u \quad (a \neq b)}{\langle a \rangle t \approx \langle b \rangle u}$$

The following properties of syntactic freshness and equality are a special case of more general properties established elsewhere, e.g. by Urban et al. [19]:

Proposition 4.3. *The nominal equality relation \approx is an equivalence relation. Hence, $NTm = Tm/\approx$ is well-defined. Moreover, both \approx and $\#$ are equivariant relations on Tm .*

We now show how to interpret arbitrary nominal terms in NTm .

Definition 4.4. Let $\theta : V \rightarrow NTm$ be a substitution of ground nominal terms for variables, called an *interpretation*. We lift θ to a function from arbitrary terms to NTm as follows:

$$\begin{aligned} \theta(a) &= a \\ \theta(c) &= c \\ \theta(f(\bar{t})) &= f(\theta(t_1), \dots, \theta(t_n)) \\ \theta((a \ b) \cdot t) &= (\theta(a) \ \theta(b)) \cdot \theta(t) \\ \theta(\langle a \rangle t) &= \langle \theta(a) \rangle \theta(t) \end{aligned}$$

We say that $\theta : FV(\Sigma) \rightarrow NTm$ satisfies Σ (written $\theta : \Sigma$) if $\theta(x) : \Sigma(x)$ for each x and $a \# \theta(x)$ for each constraint $a \# x \in |\Sigma|$.

We write $\theta \models t \approx u$ or $\theta \models a \# t$ to indicate that $\theta(t) \approx \theta(u)$ or $\theta(a) \# \theta(t)$ respectively. Similarly, $\theta \models \Gamma$ indicates that $\theta \models A$ for each constraint A in constraint set Γ . We say that a constraint A (or constraint set Γ) is *satisfiable* if there is an interpretation $\theta : \Sigma$ such that $\theta \models A$ (respectively, $\theta \models \Gamma$) holds in NTm .

Proposition 4.5. *The axioms listed in Figure 7 are valid for NTm , in the sense that for each axiom $\bigwedge P \supset \bigvee Q$, if $\theta \models \bigwedge P$ then $\theta \models Q_i$ for some $Q_i \in \bigvee Q$.*

Proof. For S_1 and S_2 , the proof is by induction on the definition of swapping for ground terms. The validity of S_3 is immediate.

For the equivariance axioms, the definition of swapping makes plain that abstraction and other function symbols besides swapping itself are equivariant. In addition, it is not difficult to show that

$$(a \ a') \cdot (b \ b') \cdot x = ((a \ a') \cdot b \ (a \ a') \cdot b') \cdot (a \ a') \cdot x$$

that is, that the syntactic swapping function is equivariant. For the equivariance axioms for formulas, we only need to consider E_{\approx} and $E_{\#}$. But clearly equality is equivariant since

$$x \approx y \supset (a \ b) \cdot x \approx (a \ b) \cdot y$$

can be shown by induction on the derivation of $x \approx y$; similarly,

$$a \# x \supset (b \ b') \cdot a \# (b \ b') \cdot x$$

can be shown valid by induction on the derivation of $a \# x$.

For the axiom F_1 , we must show that if $a \# x$ and $b \# x$ then $(a \ b) \cdot x \approx x$. The proof is by induction on the structure of x . For $x = c$ the result is immediate; similarly, for $x = f(\bar{y}t)$ the induction step is straightforward. For $x = c$, we have $a, b \neq c$ so $(a \ b) \cdot c = c \approx c$. For $x = \langle c \rangle y$, there are two cases. If $a, b \neq c$ then we have $a, b \# y$ and

$$(a \ b) \cdot \langle c \rangle y = \langle (a \ b) \cdot c \rangle (a \ b) \cdot y \approx \langle c \rangle y$$

since by induction $(a \ b) \cdot y \approx y$. Otherwise, without loss of generality suppose $b = c$ (the case where $a = c$ is symmetric). We need to show that $(a \ b) \cdot \langle b \rangle y \approx \langle b \rangle y$, or equivalently that $\langle a \rangle (a \ b) \cdot y \approx \langle b \rangle y$. If $a = b$, this is trivial. Otherwise, it is sufficient to show that $(a \ b) \cdot y \approx (a \ b) \cdot y$ (which is immediate) and $a \# y$. But since $a \# \langle b \rangle y$ and $a \neq b$, we know that $a \# y$ holds.

For F_2 , clearly any two name symbols $a:\nu$ and $b:\nu'$ of different sorts are distinct, so $a \# b$.

For F_3 , we need to show that $a \# a$ is not derivable. This is immediate from the definition of the freshness relation.

For F_4 , we need to show that either $a \# b$ or $a \approx b$ is derivable. If $a = b$ then $a \approx b$ is derivable; otherwise $a \neq b$ so $a \# b$ is derivable.

Finally, for A_1 we need to show that if $a \# y$ and $x \approx (a \ b) \cdot y$ then $\langle a \rangle x \approx \langle b \rangle y$. There are two cases. If $a \neq b$ then the last rule in the definition of nominal equality applies to show $\langle a \rangle x \approx \langle b \rangle y$. Otherwise, $a = b$ so $x \approx (a \ b) \cdot y = y$ and so $\langle a \rangle x \approx \langle b \rangle y$. \square

Proposition 4.6. *If $\theta \models \langle a \rangle x \approx \langle b \rangle y$ then either $\theta \models a \approx b, x \approx y$ or $\theta \models a \# y, x \approx (a \ b) \cdot y$.*

Proof. The proof is by case analysis of the possible derivations of $\theta(\langle a \rangle x) \approx \theta(\langle b \rangle y)$. There are only two cases, corresponding to the last two rules in the definition of structural equality. The result is immediate. \square

Proposition 4.7. *If $\theta : \Sigma$ then $\theta \models a \# t$ for each $a \# t \in |\Sigma|$.*

Proof. The proof is by induction on the structure of t . The critical case is for t a variable; in this case, we need to use the fact that $\theta : \Sigma$ only if $a \# \theta(x)$ for each $a \# x \in |\Sigma|$. \square

Theorem 4.8. *Let Γ be a set of freshness and equality formulas. If $\Sigma; \Gamma \Rightarrow \perp$ is derivable then Γ is unsatisfiable.*

Proof. Proof is by induction on the structure of the derivation. Note that the only applicable rules are non-logical rules. There is one case for each nonlogical rule. Most cases are straightforward. We present some interesting cases.

All of the axioms in Figure 7 hold in NTm , by Proposition 4.5, so the cases in which these axioms are used are straightforward. For example, for a derivation of the form

$$\frac{}{\Sigma; \Gamma, a \# a \Rightarrow \perp} F_3$$

clearly $\Gamma, a \# a$ is unsatisfiable.

For a derivation of the form

$$\frac{\Sigma; \Gamma, a \# b \Rightarrow \perp \quad \Sigma; \Gamma, a \approx b \Rightarrow \perp}{\Sigma; \Gamma \Rightarrow \perp} F_4$$

we have $\Gamma, a \approx b$ and $\Gamma, a \# b$ unsatisfiable. If $\theta : \Sigma$ then either $\theta(a) \approx \theta(b)$ or $\theta(a) \neq \theta(b)$, in which case $\theta(a) \# \theta(b)$. In either case, θ cannot satisfy Γ .

For a derivation ending with F ,

$$\frac{\Sigma \# a : \nu; \Gamma \Rightarrow \perp}{\Sigma; \Gamma \Rightarrow \perp} F$$

if $\theta : \Sigma$, then without loss of generality we can assume $a \# \theta$ so that $\theta : \Sigma \# a : \nu$ and so $\theta \not\vdash \Gamma$ by induction.

For $\Sigma \#$:

$$\frac{\Sigma; \Gamma, a \# t \Rightarrow \perp \quad (a \# t \in |\Sigma|)}{\Sigma; \Gamma \Rightarrow \perp} \Sigma \#$$

if $\theta : \Sigma$ then $\theta \models a \# t$ for any $a \# t \in |\Sigma|$, by Proposition 4.7. Consequently $\theta \not\vdash \Gamma$.

For A_2 ,

$$\frac{\Sigma; \Gamma, a \approx b, x \approx y \Rightarrow \perp \quad \Sigma; \Gamma, a \# y, x \approx (ab) \cdot y \Rightarrow \perp}{\Sigma; \Gamma, \langle a \rangle x \approx \langle b \rangle y \Rightarrow \perp} A_2$$

suppose $\theta : \Sigma$. By induction $\theta \not\vdash \Gamma, a \approx b, x \approx y$ and $\theta \not\vdash \Gamma, a \# y, x \approx (ab) \cdot y$. There are three cases. If $\theta(a) \approx \theta(b)$ and $\theta(x) \approx \theta(y)$, then $\theta \not\vdash \Gamma$. Similarly, if $\theta(a) \# \theta(y)$ and $\theta(x) \approx (\theta(a) \theta(b)) \cdot \theta(y)$ then $\theta \not\vdash \Gamma$. Otherwise, by the contrapositive of Proposition 4.6, $\theta \not\vdash \langle a \rangle x \approx \langle b \rangle y$. In any case, $\theta \not\vdash \Gamma, \langle a \rangle x \approx \langle b \rangle y$.

For A_3 ,

$$\frac{\Sigma \vdash t : \langle \nu \rangle \tau \quad \Sigma, a : \nu, x : \tau; \Gamma, t \approx \langle a \rangle x \Rightarrow \perp}{\Sigma; \Gamma \Rightarrow \perp} A_3$$

if $\theta : \Sigma$ then $\theta(t) = \langle a \rangle v$ for some $a : \nu$ and $t : \tau$, so let $\theta' = \theta[a \mapsto a, x \mapsto t]$. Clearly $\theta' : \Sigma, a : \nu, x : \tau$ and $\theta' \models t \approx \langle a \rangle x$ so by induction $\theta' \not\vdash \Gamma$. Since Γ does not mention a or x , we can conclude $\theta \not\vdash \Gamma$. \square

Corollary 4.9 (Syntactic consistency). *There is no derivation of $\cdot; \cdot \Rightarrow \perp$.*

Proof. This follows from Proposition 4.1 and Theorem 4.8, since \emptyset is a satisfiable constraint set. \square

4.2 Orthogonality of abstraction

Using cut-elimination, we can also show that some parts of the equational theory are “orthogonal extensions”, that is, derivable sequents not mentioning abstraction can be derived without using the special properties of these symbols.

Theorem 4.10 (Conservativity). *Suppose Σ has no variables mentioning abstraction and $\Sigma; \Gamma \Rightarrow \Delta$ and Γ, Δ have no subterms of the form $\langle a \rangle t$. Then there is a derivation of $\Sigma; \Gamma \Rightarrow \Delta$ that does not use any nonlogical rules involving abstraction.*

Proof. We say that a context, formula, formula multiset, or sequent is abstraction-free if the abstraction function symbol and type constructor do not appear in it. A derivation is abstraction-free if the rules A_1, A_2, A_3 do not appear in it. We write \vdash^{-A} for abstraction-free derivability.

The proof is by induction on the structure of cut-free derivations. We need a stronger induction hypothesis. We say Γ is *good* if abstraction is only mentioned in equations and freshness formulas. Note that if Σ is

abstraction-free and there are no constants whose types mention abstraction then the only well-formed closed terms of type $\langle \nu \rangle \tau$ are of the form $\langle a \rangle t$. Hence, any equations among abstraction-typed terms are of the form $\langle a \rangle t \approx \langle b \rangle u$; we call such formulas abstraction equations. Any context can be partitioned into Γ, Γ' such that Γ' contains all the abstraction equations. We say that Γ' is *redundant* relative to Γ if whenever $\langle a \rangle t \approx \langle b \rangle u \in \Gamma'$, we have either $\vdash^{-A} \Sigma; \Gamma \Rightarrow a \approx b$ and $t \approx u$ or $\vdash^{-A} \Sigma; \Gamma \Rightarrow a \# u$ and $t \approx (a b) \cdot u$.

We will show that if Σ, Δ are abstraction-free and Γ, Γ' is good and Γ' is redundant relative to Γ , then if $\vdash \Sigma; \Gamma, \Gamma' \Rightarrow \Delta$ then $\vdash^{-A} \Sigma; \Gamma \Rightarrow \Delta$. An abstraction-free Γ is obviously good and redundant relative to \emptyset , so the main theorem is a special case.

The proof is by structural induction on the derivation. The cases involving left or right rules are straightforward because such rules act only on Γ and do not affect goodness and redundancy. The case for *hyp* is easy since the hypothesis cannot be in Γ' .

For A_1 , we have

$$\frac{\Sigma; \Gamma, a \# x, x \approx (a b) \cdot y, \Gamma', \langle a \rangle x \approx \langle b \rangle y \Rightarrow \Delta}{\Sigma; \Gamma, a \# x, x \approx (a b) \cdot y, \Gamma' \Rightarrow \Delta} A_1$$

Clearly, $\Gamma', \langle a \rangle x \approx \langle b \rangle y$ is redundant relative to $\Gamma, a \# x, x \approx (a b) \cdot y$. Also, goodness is preserved. So by induction we have $\Sigma; \Gamma, a \# x, x \approx (a b) \cdot y \Rightarrow \Delta$, as desired.

For A_2 , we have

$$\frac{\Sigma; \Gamma, \Gamma', \langle a \rangle x \approx \langle b \rangle y, a \approx b, x \approx y \Rightarrow \Delta \quad \Sigma; \Gamma, \Gamma', \langle a \rangle x \approx \langle b \rangle y, a \# y, x \approx (a b) \cdot y \Rightarrow \Delta}{\Sigma; \Gamma, \Gamma', \langle a \rangle x \approx \langle b \rangle y \Rightarrow \Delta} A_2$$

Since $\Gamma', \langle a \rangle x \approx \langle b \rangle y$ is redundant relative to Γ , there are two cases. If $\Sigma; \Gamma \Rightarrow a \approx b$ and $x \approx y$, then by induction we have a derivation of $\Sigma; \Gamma, a \approx b, x \approx y \Rightarrow \Delta$, and using cut we can derive $\Sigma; \Gamma \Rightarrow \Delta$ as desired. Otherwise, if $\Sigma; \Gamma \Rightarrow a \# y$ and $x \approx (a b) \cdot y$, then by induction we have a derivation of $\Sigma; \Gamma, a \# y, x \approx (a b) \cdot y \Rightarrow \Delta$, and using cut we can derive $\Sigma; \Gamma \Rightarrow \Delta$ as desired. Cut-elimination does not introduce uses of the abstraction rules, so the resulting derivations are abstraction-free.

For A_3 , we have

$$\frac{\Sigma \vdash t : \langle \nu \rangle \tau \quad \Sigma, a : \nu, x : \tau; \Gamma, t \approx \langle a \rangle x, \Gamma' \Rightarrow \Delta}{\Sigma; \Gamma, \Gamma' \Rightarrow \Delta} A_3$$

Since Σ has no variables of abstraction type, we must have $t = \langle u \rangle v$ for some terms $\Sigma \vdash u : \nu$ and $\Sigma \vdash v : \tau$. Therefore, we can substitute into the derivation $\Sigma, a : \nu, x : \tau; \Gamma, \Gamma', t \approx \langle a \rangle x \Rightarrow \Delta$ to get $\Sigma; \Gamma, \Gamma', \langle u \rangle v \approx \langle a \rangle x \Rightarrow \Delta$. Clearly $\Sigma; \Gamma \Rightarrow u \approx u$ and $v \approx v$, and $\Gamma', \langle u \rangle v \approx \langle a \rangle x$ is redundant relative to Γ , so by induction, we have a derivation of $\Sigma; \Gamma \Rightarrow \Delta$.

For the reflexivity rule $\approx R$, we have

$$\frac{\Sigma; \Gamma, \Gamma', t \approx t \Rightarrow \Delta}{\Sigma; \Gamma, \Gamma' \Rightarrow \Delta} \approx R$$

If $t = \langle a \rangle x$, then clearly $\Gamma \Rightarrow a \approx a$ and $x \approx x$, so $\Gamma', \langle a \rangle x \approx \langle a \rangle x$ is redundant relative to Γ , and we have $\Sigma; \Gamma \Rightarrow \Delta$ by induction. Otherwise, $\Gamma, \Gamma', t \approx t$ is obviously still good and Γ' redundant with respect to $\Gamma, t \approx t$, so we can again conclude $\Sigma; \Gamma \Rightarrow \Delta$ by induction.

For $\approx S$ -derivations, we have

$$\frac{\Sigma; \Gamma, t \approx u, P(t), P(u) \Rightarrow \Delta}{\Sigma; \Gamma, \Gamma', t \approx u, P(t) \Rightarrow \Delta} \approx S$$

If $P(u)$ is not an equation among abstraction-typed terms then the induction step is easy. There are many cases depending on the structure of $P(x)$, but in each case we can show that $P(u)$ is also redundant relative to $\Gamma, t \approx u$ (if $t \approx u$ is not an abstraction equation) or Γ (if $t \approx u$ is an abstraction equation).

The remaining nonlogical rules do not involve formulas of the form $\langle a \rangle x \approx \langle b \rangle y$, so the induction step is immediate for these rules. □

4.3 Equivalence to Nominal Logic

In this section we discuss the relationship between the sequent calculi NL^{\Rightarrow} and INL^{\Rightarrow} and classical and intuitionistic variants of Nominal Logic respectively. We aim to show that, modulo a straightforward syntactic translation, formulas are provable in one system if and only if they are provable in the other. This in turn suggests that they are equally expressive in a model-theoretic sense (provided models for NL^{\Rightarrow} are defined in an appropriate way for its slightly different syntax, as done for example for FL [9]); however, in this article we will not pursue the model theory of NL^{\Rightarrow} .

4.3.1 Classical Nominal Logic

We first consider the classical case. We write NL for the set of all axioms of Pitts' axiomatization of nominal logic, as reviewed in Section 2.1. For ordinary variable contexts Σ and NL -formula multisets Γ, Δ , we write $\vdash_{NL} \Sigma; \Gamma \Rightarrow \Delta$ to indicate that $\Sigma; \Gamma, \Gamma' \Rightarrow_{\mathbf{G3c}} \Delta$ for some $\Gamma' \subseteq NL$. Without loss of generality, a finite Γ' can always be used. We write $\vdash_{NL^{\Rightarrow}}$ for derivability in NL^{\Rightarrow} .

There is one technical point to address. Our system contains explicit name-constants quantified by \mathbf{I} and appearing in typing contexts, whereas in Pitts' system \mathbf{I} quantifies ordinary variables. To bridge this gap, we translate NL formulas to NL^{\Rightarrow} formulas by replacing \mathbf{I} -bound variables with fresh name-symbols. For example, the NL formula $\mathbf{I}a:\nu.\mathbf{I}b:\nu'.p(a, b)$ translates to the NL^{\Rightarrow} formula $\mathbf{I}a:\nu.\mathbf{I}b:\nu'.p(a, b)$. We write φ^* for the translation of φ , which is defined as follows:

$$\begin{aligned} A^* &= A \\ \perp^* &= \perp \\ (\varphi \supset \psi)^* &= \varphi^* \supset \psi^* \\ (\forall x:\tau.\varphi)^* &= \forall x:\tau.\varphi^* \\ (\mathbf{I}a:\nu.\varphi)^* &= \mathbf{I}a:\nu.(\varphi^*[a/a]) \quad (a = \iota(a)) \end{aligned}$$

Technically, we translate sequents or derivations mentioning variables in $\mathbb{V} \cup \mathbb{A}'$, to sequents or derivations mentioning variables in $\mathbb{V} \cup \mathbb{A}$, where \mathbb{A}' is an isomorphic copy of the set of names \mathbb{A} . We assume that before translation, formulas are renamed so that \mathbf{I} -bound variables are in \mathbb{A}' , and we fix an isomorphism $\iota : \mathbb{A}' \rightarrow \mathbb{A}$. In what follows, we will sometimes leave ι implicit and assume that $\iota(a) = a$ whenever we encounter a \mathbf{I} -quantifier or context of the form $\Sigma \# a:\nu$.

The omitted cases for $\top, \wedge, \vee, \exists$ are derivable via de Morgan identities. The translation of a judgment $\Sigma; \Gamma \Rightarrow \Delta$ is $\Sigma; \Gamma^* \Rightarrow \Delta^*$, where Γ^*, Δ^* is the result of translating each element of Γ, Δ respectively.

We first show that every theorem of NL translates to a theorem of NL^{\Rightarrow} .

Theorem 4.11. *If $\vdash_{NL} \Sigma; \Gamma \Rightarrow \Delta$ then $\vdash_{NL^{\Rightarrow}} \Sigma; \Gamma^* \Rightarrow \Delta^*$.*

Proof. We defined $\vdash_{NL} \Sigma; \Gamma \Rightarrow \Delta$ to mean $\vdash_{\mathbf{G3c}} \Sigma; \Gamma, \Gamma' \Rightarrow \Delta$ for some finite subset $\Gamma' \subseteq NL$. Any $\mathbf{G3c}$ derivation is an NL^{\Rightarrow} derivation, so we just need to show that in NL^{\Rightarrow} , all of the uses of NL axioms are redundant. We will show that each axiom $\varphi \in NL$ is derivable in NL^{\Rightarrow} . Thus, using *cut* finitely many times, we can derive $\Sigma; \Gamma \Rightarrow \Delta$ in NL^{\Rightarrow} .

For most of the axioms, this is straightforward. All of the axioms of the form $\forall \bar{x}. \bigwedge \bar{P} \supset \bigvee \bar{Q}$ are clearly derivable from the corresponding nonlogical rules as follows:

$$\frac{\frac{\frac{\bar{x}:\bar{\tau}; \bar{P}, Q_1 \Rightarrow \bigvee \bar{Q} \quad \dots \quad \bar{x}:\bar{\tau}; \bar{P}, Q_n \Rightarrow \bigvee \bar{Q}}{\bar{x}:\bar{\tau}; \bar{P} \Rightarrow \bigvee \bar{Q}} \text{Ax}}{\bar{x}:\bar{\tau}; \cdot \Rightarrow \bigwedge \bar{P} \supset \bigvee \bar{Q}} \supset R, \wedge R}{\cdot; \cdot \Rightarrow \forall \bar{x}:\bar{\tau}. \bigwedge \bar{P} \supset \bigvee \bar{Q}} \forall R$$

with the topsequents all derivable using $\forall R$ and *hyp*.

This leaves axioms not fitting this pattern, including (CF_2) , (CF_4) , (CA_1) , (CA_2) , and (CQ) . (CA_1) and (CA_2) can be derived using the nonlogical rules $A_1, A_2, A_3, \approx S$ of NL^{\Rightarrow} , and (CF_2) using F_3 and F_4 of NL^{\Rightarrow} . We will show the cases for (CF_4) and both directions of (CQ) in detail.

For an instance $\forall \bar{x}. \exists a. a \# \bar{x}$ of CF_4 , the derivation is of the form

$$\frac{\frac{\frac{\overline{\bar{x}:\bar{\tau}\#a:\nu : a \# \bar{x} \Rightarrow a \# \bar{x}}}{\overline{\bar{x}:\bar{\tau}\#a:\nu ; \cdot \Rightarrow \exists a:\nu.a \# \bar{x}}} \exists R, \Sigma\#}{\overline{\bar{x}:\bar{\tau}; \cdot \Rightarrow \exists a:\nu.a \# \bar{x}}} F}{\overline{\cdot ; \cdot \Rightarrow \forall \bar{x}:\bar{\tau}.\exists a:\nu.a \# \bar{x}}} \forall R$$

For a translated instance of (CQ) of the form $\forall \bar{x}. (\mathbf{I}a:\nu.\varphi(a, \bar{x}) \iff \exists a:\nu.a \# \bar{x} \wedge \varphi(a, \bar{x}))$, we will prove the two directions individually. For the forward direction, after some syntax-directed applications of right-rules we have

$$\frac{\frac{\frac{\overline{\bar{x}:\bar{\tau}\#a:\nu ; \varphi(a, \bar{x}), a \# \bar{x} \Rightarrow a \# \bar{x}}}{\overline{\bar{x}:\bar{\tau}\#a:\nu ; \varphi(a, \bar{x}) \Rightarrow a \# \bar{x}}} hyp}{\overline{\bar{x}:\bar{\tau}\#a:\nu ; \varphi(a, \bar{x}) \Rightarrow \varphi(a, \bar{x)}} \Sigma\#^n}{\overline{\bar{x}:\bar{\tau}\#a:\nu ; \varphi(a, \bar{x}) \Rightarrow \varphi(a, \bar{x})} \wedge R}{\frac{\frac{\overline{\bar{x}:\bar{\tau}\#a:\nu ; \varphi(a, \bar{x}) \Rightarrow a \# \bar{x} \wedge \varphi(a, \bar{x})}{\overline{\bar{x}:\bar{\tau}; \mathbf{I}a:\nu.\varphi(a, \bar{x}) \Rightarrow \exists a:\nu.a \# \bar{x} \wedge \varphi(a, \bar{x})} \mathbf{I}L, \exists R}{\overline{\cdot ; \cdot \Rightarrow \forall \bar{x}:\bar{\tau}.(\mathbf{I}a:\nu.\varphi(a, \bar{x}) \supset \exists a:\nu.a \# \bar{x} \wedge \varphi(a, \bar{x}))} \forall R^n, \supset R}$$

For the reverse direction, we need to show $\forall \bar{x}.\exists a:\nu.a \# \bar{x} \wedge \varphi(a, \bar{x}) \supset \mathbf{I}a:\nu.\varphi(a, \bar{x})$.

$$\frac{\frac{\frac{\overline{\bar{x}:\bar{\tau}, a:\nu\#b:\nu ; \varphi(b, \bar{x}) \Rightarrow \varphi(b, \bar{x})}{\overline{\bar{x}:\bar{\tau}, a:\nu\#b:\nu ; a \# \bar{x}, b \# \bar{x}, (a \ b) \cdot \varphi(a, \bar{x}) \Rightarrow \varphi(b, \bar{x})} Ax^*}{\overline{\bar{x}:\bar{\tau}, a:\nu\#b:\nu ; a \# \bar{x}, \varphi(a, \bar{x}) \Rightarrow \varphi(b, \bar{x})} \Sigma\#^*, EVL}{\overline{\bar{x}:\bar{\tau}, a:\nu ; a \# \bar{x}, \varphi(a, \bar{x}) \Rightarrow \mathbf{I}a:\nu.\varphi(a, \bar{x})} \mathbf{I}R}{\overline{\bar{x}:\bar{\tau}; \exists a:\nu.a \# \bar{x} \wedge \varphi(a, \bar{x}) \Rightarrow \mathbf{I}a:\nu.\varphi(a, \bar{x})} \exists L, \wedge L}{\overline{\cdot ; \cdot \Rightarrow \forall \bar{x}:\bar{\tau}.(\exists a:\nu.a \# \bar{x} \wedge \varphi(a, \bar{x}) \supset \mathbf{I}a:\nu.\varphi(a, \bar{x}))} \forall R, \supset R}$$

Since both a and b are fresh for all the other free variables of φ , we have $\varphi(a, \bar{x}) \iff \varphi((b \ a) \cdot a, (b \ a) \cdot \bar{x}) \iff \varphi(b, \bar{x})$ using equivariance and the fact that $a \# x \wedge b \# x \supset (a \ b) \cdot x \approx x$.

Consequently, all the translations of axioms of NL can be derived in NL^\Rightarrow . As a result, if $\Gamma \subset NL$ is a finite set of axioms such that $\vdash_{NL^\Rightarrow} \Sigma; \Gamma, \Gamma' \Rightarrow \Delta$, then using the derivations of the axioms and finitely many instances of cut , we can obtain a derivation of $\vdash_{NL^\Rightarrow} \Sigma; \Gamma \Rightarrow \Delta$. \square

Observe that this means that any closed theorem of NL can be derived in NL^\Rightarrow . For example, from Pitts [15, Prop. 3 and 4] we can show:

Proposition 4.12. • If $FV(t) \subseteq \bar{x}$ and $FN(t) = \emptyset$ then we can derive $\Sigma; \Gamma \Rightarrow \forall a:\nu.\forall \bar{x}:\bar{\tau}. a \# x_1 \wedge \dots \wedge a \# x_n \supset a \# t$.

• If $FV(\varphi) \subseteq \{a, \bar{x}\}$ then we can derive $\Sigma; \Gamma \Rightarrow \exists a:\nu.a \# \bar{x} \wedge \varphi(a, \bar{x}) \iff \forall a:\nu.a \# \bar{x} \supset \varphi(a, \bar{x})$

Now we consider the converse: showing that there are no “new theorems”, that any NL sequent derivable in NL^\Rightarrow is also derivable in NL . This is not as straightforward because subderivations of translated NL judgments may involve name-symbols. However, we can show that such name-symbols can always be removed.

We also introduce a converse translation mapping NL^\Rightarrow formulas to NL formulas:

$$\begin{aligned} A^\dagger &= A \\ \perp^\dagger &= \perp \\ (\varphi \supset \psi)^\dagger &= \varphi^\dagger \supset \psi^\dagger \\ (\forall x:\tau.\varphi)^\dagger &= \forall x:\tau.\varphi^\dagger \\ (\mathbf{I}a:\nu.\varphi)^\dagger &= \mathbf{I}a:\nu.(\varphi^\dagger[a/a]) \quad (\iota(a) = a) \end{aligned}$$

Technically, we translate NL formulas over variables \mathbb{V} to NL^\Rightarrow formulas over $\mathbb{V} \cup \mathbb{A}'$, again using the bijection ι between name-variables \mathbb{A}' and names in \mathbb{A} . Note that (up to α -equivalence) the $(-)^*$ -translation and $(-)^{\dagger}$ -translation are inverses. We also define the set $\|\Sigma\|$ as follows:

$$\|\Sigma\| = \{a \# x \mid \iota(a) \# x \in |\Sigma|\} \cup \{a \# b \mid \iota(a) \# \iota(b) \in |\Sigma|\}$$

that is, $\|\Sigma\|$ is the finite subset of $|\Sigma|$ consisting of constraints whose right-hand sides are variables or names, but with names replaced by the corresponding name-variables according to the bijection ι .

We can now show the desired result.

Theorem 4.13. *If $\Sigma; \Gamma \Rightarrow \Delta$ is derivable in NL^\Rightarrow then $\Sigma^\dagger; \Gamma^\dagger, \|\Sigma\| \Rightarrow \Delta^\dagger$ is derivable in NL .*

Proof. The proof is by induction on the logical height of this derivation, with secondary induction on the total height. For the cases corresponding to first-order/equational proof rules, the induction step is straightforward.

For the cases corresponding to nonlogical rules corresponding to universal axioms $\forall \bar{x}. \bigwedge \bar{P} \supset \bigvee \bar{Q}$, suppose that we have derivations of the form

$$\frac{\Sigma; \Gamma, \bar{P}, Q_1 \Rightarrow \Delta \quad \Sigma; \Gamma, \bar{P}, Q_n \Rightarrow \Delta}{\Sigma; \Gamma, \bar{P} \Rightarrow \Delta} Ax$$

Then by induction, we have NL derivations of the NL sequents $\Sigma^\dagger; \Gamma^\dagger, \bar{P}, Q_i, \|\Sigma\| \Rightarrow \Delta^\dagger$ for $i \in \{1, \dots, n\}$. It is straightforward to show that each of the axioms in Figure 7 is provable in NL , hence we can cut against each axiom instance:

$$\frac{\Sigma^\dagger; \Gamma^\dagger, \bar{P}, \|\Sigma\| \Rightarrow \bigwedge \bar{P} \quad \frac{\Sigma^\dagger; \Gamma^\dagger, \bar{P}, Q_1, \|\Sigma\| \Rightarrow \Delta^\dagger \quad \dots \quad \Sigma^\dagger; \Gamma^\dagger, \bar{P}, Q_n, \|\Sigma\| \Rightarrow \Delta^\dagger}{\Sigma^\dagger; \Gamma^\dagger, \bar{P}, \bigvee \bar{Q}, \|\Sigma\| \Rightarrow \Delta^\dagger} \vee L^n}{\frac{\Sigma^\dagger; \Gamma^\dagger, \bar{P}, \bigwedge \bar{P} \supset \bigvee \bar{Q}, \|\Sigma\| \Rightarrow \Delta^\dagger}{\Sigma^\dagger; \Gamma^\dagger, \bar{P}, \forall \bar{x}. \bigwedge \bar{P} \supset \bigvee \bar{Q}, \|\Sigma\| \Rightarrow \Delta^\dagger} \forall R}{\Sigma^\dagger; \Gamma^\dagger, \bar{P}, \|\Sigma\| \Rightarrow \Delta^\dagger} cut} \Sigma^\dagger; \cdot \Rightarrow \forall \bar{x}. \bigwedge \bar{P} \supset \bigvee \bar{Q}$$

The cases for $F_3, F_4, F, A_2, A_3, \Sigma\#, \mathbf{NL}, \mathbf{NR}$ remain.

For F_3 , we have a derivation

$$\frac{}{\Sigma; \Gamma, a \# a \Rightarrow \Delta} F_3$$

In NL we can derive $\Sigma^\dagger; \Gamma^\dagger, a \# a, \|\Sigma\| \Rightarrow \Delta^\dagger$ using the $a \# b \supset a \not\approx b$ direction of (CF_2) since $a \not\approx a$ is contradictory.

For F_4 , we have a derivation

$$\frac{\Sigma; \Gamma, a \approx b \Rightarrow \Delta \quad \Sigma; \Gamma, a \# b \Rightarrow \Delta}{\Sigma; \Gamma \Rightarrow \Delta} F_4$$

By induction, we have derivations of $\Sigma^\dagger; \Gamma^\dagger, a \approx b, \|\Sigma\| \Rightarrow \Delta^\dagger$ and $\Sigma^\dagger; \Gamma^\dagger, a \# b, \|\Sigma\| \Rightarrow \Delta^\dagger$. Since $a \# b \iff a \not\approx b$ and $a \approx b \vee a \not\approx b$ is a tautology in classical logic, $a \# b \vee a \not\approx b$ is also a tautology. We can cut against a derivation of this formula to derive $\Sigma; \Gamma \Rightarrow \Delta$ in NL .

For F , suppose we have a derivation of the form

$$\frac{\Sigma\#a:\nu; \Gamma \Rightarrow \Delta}{\Sigma; \Gamma \Rightarrow \Delta} F$$

By induction, we can derive the NL sequent $\Sigma^\dagger, a:\nu; \Gamma^\dagger, \|\Sigma\#a:\nu\| \Rightarrow \Delta^\dagger$. Note that $\|\Sigma\#a:\nu\| = \|\Sigma\|, a \# \bar{x}$ where $\bar{x} = FV(\Sigma^\dagger)$. Using the freshness axiom (CF_4) of NL , we can derive

$$\frac{\Sigma^\dagger, a:\nu; \Gamma^\dagger, \|\Sigma\|, a \# \bar{x} \Rightarrow \Delta^\dagger}{\Sigma^\dagger; \cdot \Rightarrow \forall \bar{x}. \bar{x}. \exists a:\nu. a \# \bar{x} \quad \Sigma^\dagger; \Gamma^\dagger, \forall \bar{x}. \exists a:\nu. a \# \bar{x}, \|\Sigma\| \Rightarrow \Delta^\dagger} \forall L, \exists L}{\Sigma^\dagger; \Gamma^\dagger, \|\Sigma\| \Rightarrow \Delta^\dagger} cut$$

It is likewise easy to derive rules A_2, A_3 from axioms $(CA_1), (CA_2)$ of NL using cut .
For $\Sigma\#$, suppose we have a derivation of the form:

$$\frac{\Sigma_1\#a:\nu, \Sigma_2; \Gamma, a \# t \Rightarrow \Delta \quad (a \# t \in |\Sigma_1|)}{\Sigma_1\#a:\nu, \Sigma_2; \Gamma \Rightarrow \Delta} \Sigma\#$$

By induction, we have $\Sigma_1^\dagger, a:\nu, \Sigma_2^\dagger; \Gamma^\dagger, a \# t, \|\Sigma_1\#a:\nu, \Sigma_2\| \Rightarrow \Delta^\dagger$. Observe that $a \# \bar{x} \subseteq \|\Sigma_1\#a:\nu, \Sigma_2\|$. Using Proposition 4.12(1), we can derive as follows:

$$\frac{\Sigma_1^\dagger, a:\nu, \Sigma_2^\dagger; \cdot \Rightarrow \forall a:\nu. \forall \bar{x}:\bar{\tau}. a \# \bar{x} \supset a \# t \quad \frac{\Sigma_1^\dagger, a:\nu, \Sigma_2^\dagger; \Gamma^\dagger, \|\Sigma_1\#a:\nu, \Sigma_2\|, a \# t \Rightarrow \Delta^\dagger}{\Sigma_1^\dagger, a:\nu, \Sigma_2^\dagger; \Gamma^\dagger, \|\Sigma_1\#a:\nu, \Sigma_2\|, \forall a:\nu. \forall \bar{x}:\bar{\tau}. a \# \bar{x} \supset a \# t \Rightarrow \Delta^\dagger} \forall L^*, \supset L^*}{\Sigma_1^\dagger, a:\nu, \Sigma_2^\dagger; \Gamma^\dagger, \|\Sigma_1\#a:\nu, \Sigma_2\| \Rightarrow \Delta^\dagger} cut$$

Finally, we consider the cases for \mathcal{NL} and \mathcal{NR} . For \mathcal{NL} , we have

$$\frac{\Sigma\#a:\nu; \Gamma, \varphi(a, \bar{x}) \Rightarrow \Delta}{\Sigma; \Gamma, \mathcal{N}a:\nu. \varphi(a, \bar{x}) \Rightarrow \Delta} \mathcal{NL}$$

From the upper derivation, by induction, we have a derivation of $\Sigma^\dagger, a:\nu; \Gamma^\dagger, \|\Sigma\#a:\nu\|, \varphi^\dagger(a, \bar{x}) \Rightarrow \Delta^\dagger$. Since $\|\Sigma\#a:\nu\| = \|\Sigma\|, a \# \bar{y}$ where $\bar{y} = FV(\Sigma^\dagger) \supseteq \bar{x}$, we can also derive $\Sigma^\dagger; \Gamma^\dagger, \|\Sigma\|, \exists a:\nu. a \# \bar{x} \wedge \varphi(a, \bar{x}) \Rightarrow \Delta$ using $\exists L$ and $\forall L$. Finally, we can cut against the axiom instance $\forall \bar{x}:\bar{\tau}. \exists a:\nu. a \# \bar{x} \wedge \varphi^\dagger(a, \bar{x}) \iff \mathcal{N}a:\nu. \varphi^\dagger(a, \bar{x})$ to prove that $\Sigma^\dagger; \Gamma^\dagger, \mathcal{N}a:\nu. \varphi^\dagger(a, \bar{x}) \Rightarrow \Delta^\dagger$.

For \mathcal{NR} , we have

$$\frac{\Sigma\#a:\nu; \Gamma \Rightarrow \varphi(a, \bar{x}), \Delta}{\Sigma; \Gamma \Rightarrow \mathcal{N}a:\nu. \varphi(a, \bar{x}), \Delta} \mathcal{NR}$$

The argument is similar to the previous case: by induction, we can derive $\Sigma^\dagger, a:\nu; \Gamma^\dagger, \|\Sigma\#a:\nu\| \Rightarrow \varphi^\dagger(a, \bar{x}), \Delta^\dagger$ in NL . Thus, since $\|\Sigma\#a:\nu\| = \|\Sigma\|, a \# \bar{y}$ where $\bar{y} = FV(\Sigma^\dagger)$, we can conclude $\Sigma^\dagger; \Gamma^\dagger, \|\Sigma\| \Rightarrow \forall a:\nu. a \# \bar{y} \supset \varphi(a, \bar{x}), \Delta^\dagger$. Using Proposition 4.12(2) and the axiom (CQ) defining \mathcal{N} in NL we can cut against the formula

$$\forall \bar{y}:\bar{\tau}. (\forall a:\nu. a \# \bar{y} \supset \varphi^\dagger(a, \bar{x})) \iff \mathcal{N}a:\nu. \varphi^\dagger(a, \bar{x})$$

where $\bar{y} \supseteq \bar{x}$. We can conclude that $\Sigma^\dagger; \Gamma^\dagger, \|\Sigma\| \Rightarrow \Delta^\dagger, \mathcal{N}a:\nu. \varphi^\dagger(a, \bar{x})$. \square

Corollary 4.14. *If Σ only contains variables and $\vdash_{NL} \Sigma; \Gamma^* \Rightarrow \Delta^*$ then $\Sigma; \Gamma \Rightarrow \Delta$ is derivable in NL^\Rightarrow .*

Proof. By Theorem 4.13, we know that $\Sigma^\dagger; (\Gamma^*)^\dagger, \|\Sigma\| \Rightarrow (\Delta^*)^\dagger$. By definition of the $(-)^*$ and $(-)^{\dagger}$ translations, we know that $(\Gamma^*)^\dagger = \Gamma$ and $(\Delta^*)^\dagger = \Delta$. Moreover, since Σ contains no name-symbols, by definition $\Sigma^\dagger = \Sigma$ and $\|\Sigma\| = \emptyset$. Hence, $\Sigma; \Gamma \Rightarrow \Delta$. \square

4.3.2 Intuitionistic Nominal Logic

We wish to argue that the intuitionistic calculus INL^\Rightarrow is really ‘‘intuitionistic nominal logic’’. However, Pitts only considered classical nominal logic. There is a subtlety having to do with Pitts’ axiom (CF_2) in the intuitionistic case.

Pitts’ original axiom (CF_2) stated that freshness among names is the same as inequality:

$$(CF_2) \quad \forall a, a':\nu. a \# a' \iff \neg(a \approx a')$$

However, this axiom does not fit the scheme for nonlogical rules given by Negri and von Plato [14]. Instead, in INL^\Rightarrow we use two nonlogical rules F_3 and F_4 asserting that no name is fresh for itself and that two names (of the same type) are either equal or fresh. These two axioms are equivalent to (CF_2) in classical logic, but in intuitionistic logic, Pitts’ axiom is weaker, since $a \approx b \vee a \not\approx b$ does not follow from (CF_2) . (Recall that for the F_4 case of Theorem 4.13, we used excluded middle for name-equality).

We have modified Pitts’ axiomatization slightly by replacing the original axiom (CF_2) with two rules, (IF_2) asserting that no name is fresh for itself, and (IF_3) stating that two names are either fresh or equal. In classical logic, these are equivalent axiomatizations, whereas (IF_3) is not provable in intuitionistic logic

Swapping	
(IS ₁)	$\forall a:\nu, x:\tau. (a\ a) \cdot x \approx x$
(IS ₂)	$\forall a, a':\nu, x:\tau. (a\ a') \cdot (a\ a') \cdot x \approx x$
(IS ₃)	$\forall a, a':\nu. (a\ a') \cdot a \approx a'$
Equivariance	
(IE ₁)	$\forall a, a':\nu, b, b':\nu', x:\tau. (a\ a') \cdot (b\ b') \cdot x \approx ((a\ a') \cdot b\ (a\ a') \cdot b') \cdot (a\ a') \cdot x$
(IE ₂)	$\forall a, a':\nu, b:\nu', x:\tau. b \# x \supset (a\ a') \cdot b \# (a\ a') \cdot x$
(IE ₃)	$\forall a, a':\nu, \bar{x} : \bar{\tau}. (a\ a') \cdot f(\bar{x}) \approx f((a\ a') \cdot \bar{x})$
(IE ₄)	$\forall a, a':\nu, \bar{x} : \bar{\tau}. p(\bar{x}) \supset p((a\ a') \cdot \bar{x})$
(IE ₅)	$\forall b, b':\nu', a:\nu, x:\tau. (b\ b') \cdot \langle a \rangle x \approx \langle (b\ b') \cdot a \rangle ((b\ b') \cdot x)$
Freshness	
(IF ₁)	$\forall a, a':\nu, x:\tau. a \# x \wedge a' \# x \supset (a\ a') \cdot x \approx x$
(IF ₂)	$\forall a:\nu. \neg(a \# a)$
(IF ₃)	$\forall a, a':\nu. a \# a' \vee a \approx a'$
(IF ₄)	$\forall a:\nu, a':\nu'. a \# a'$
(IF ₅)	$\forall \bar{x} : \bar{\tau}. \exists a:\nu. a \# \bar{x}$
\mathbb{N}-quantifier	
(IQ)	$\forall \bar{x}. (\mathbb{N}a:\nu. \varphi) \iff (\exists a:\nu. a \# \bar{x} \wedge \varphi)$
where $FV(\mathbb{N}a.\varphi) \subseteq \{\bar{x}\}$	
Abstraction	
(IA ₁)	$\forall a, a':\nu, x, x':\tau. \langle a \rangle x \approx \langle a' \rangle x' \iff (a \approx a' \wedge x \approx x' \vee (a' \# x \wedge x' \approx (a\ a') \cdot x))$
(IA ₂)	$\forall y : \langle \nu \rangle \tau. \exists a:\nu, x:\tau. y \approx \langle a \rangle x$

Figure 11: Axioms of Intuitionistic Nominal Logic

from Pitts' axioms. Moreover, it is computationally plausible that equality and freshness among names are both decidable, since names are typically finite, discrete data structures.

For this reason, we introduce an alternative axiomatization INL , shown in Figure 11, differing in the replacement of (CF_2) with two axioms (IF_2) and (IF_3) . These axioms are equivalent in classical logic to (CF_2) , but better-behaved from a proof-theoretic perspective.

Let \vdash_{INL} indicate derivability in intuitionistic logic from the axioms in INL . Using essentially the same proof techniques as for the classical case, we have:

Theorem 4.15. *If Σ contains only variables, then $\vdash_{INL} \Sigma; \Gamma \Rightarrow \Delta$ is derivable if and only if $\vdash_{INL} \Sigma; \Gamma^* \Rightarrow \Delta^*$.*

5 Conclusions

Nominal logic provides powerful techniques for reasoning about fresh names and name-binding. One of the most interesting features of nominal logic is the \mathbb{N} -quantifier. However, the techniques used for reasoning with \mathbb{N} offered by previous formalizations of nominal logic are highly (but unnecessarily) complex.

In this article we have introduced a new sequent calculus NL^{\Rightarrow} for nominal logic which uses typing contexts extended with freshness information to deal with the \mathbb{N} -quantifier. Its rules for \mathbb{N} are symmetric and rationalize a proof-search semantics for \mathbb{N} that seems natural and intuitive (inspired by the treatment of \mathbb{N} in nominal logic programming). We proved cut-elimination in detail. In addition, we used NL^{\Rightarrow} to provide a syntactic proof of consistency and a detailed proof of equivalence to Pitts' axiomatization modulo ordinary first-order (classical/intuitionistic) logic. These results are the first of their kind to be shown in detail.

NL^{\Rightarrow} has also been used in other work:

- NL^{\Rightarrow} provides a proof-search reading of \mathbb{N} which is much closer to the approach taken in the α Prolog nominal logic programming language [3, 5]. While Gabbay and Cheney gave a proof-theoretic semantics of nominal logic programming based on FL_{Seq} , this analysis does not seem relevant to

α Prolog because it suggests a quite different (and, for typical programs, much more computationally intensive) proof-search technique for \mathcal{N} -quantified formulas. In contrast, NL^{\Rightarrow} seems to provide a proof-theoretic foundation for α Prolog’s existing search technique.

- Gabbay and Cheney [7] showed that $FO\lambda^{\nabla}$, another logic due to Miller and Tiu [13] possessing a self-dual “fresh value” quantifier, can be soundly interpreted in a higher-order variant of FL_{Seq} via a proof-theoretic translation. However, the translation they developed was incomplete, and the possibility of finding a faithful translation was left open. Cheney [1] showed how to translate to a higher-order variant of NL^{\Rightarrow} and proved a completeness result. In this paper we have focused on NL^{\Rightarrow} only over first-order terms. It would be interesting to further explore NL^{\Rightarrow} over higher-order terms and compare its expressiveness to more recent variations of Miller and Tiu’s approach, such as the “nominal abstraction” system of Gacek et al. [10].
- Miculan, Scagnetto and Honsell [12] have shown how to translate derivable judgments from (a natural-deduction variant of) NL^{\Rightarrow} to the Theory of Contexts [11], an extension of the Calculus of Inductive Constructions with a theory axiomatizing a type of names with decidable equality, freshness, and name-binding encoded as second-order function symbols. It may be interesting to consider the reverse direction, e.g. translating a first-order fragment of the Theory of Contexts to nominal logic.

Additional directions for future work include the development of natural deduction calculi and type theories using the ideas of NL^{\Rightarrow} . One particularly interesting direction is the possibility of developing a type system and confluent term rewriting system that could be used to decide equality of nominal terms and proof terms. In such a system, the explicit equality and freshness theory that necessitates the many nonlogical rules in NL^{\Rightarrow} could be dealt with implicitly via traditional rewriting and syntactic side-conditions, leading to an even simpler proof theory for nominal logic. However, work in this direction by Schöpp and Stark [17] indicates that there may be significant obstacles to this approach; the system introduced in this article may be viewed as a well-behaved fragment of their system. Further development of the proof theory and type theory of nominal logic (for example, building on nominal type theories by Pitts [16], Cheney [4], or Crole and Nebel [6]) seems possible and desirable.

References

- [1] J. Cheney. A simpler proof theory for nominal logic. In *FOSSACS 2005*, volume 3441 of *LNCS*, pages 379–394. Springer-Verlag, 2005.
- [2] J. Cheney. Completeness and Herbrand theorems for nominal logic. *Journal of Symbolic Logic*, 71(1):299–320, 2006.
- [3] J. Cheney and C. Urban. Alpha-Prolog: A logic programming language with names, binding and alpha-equivalence. In *Proceedings of the 20th International Conference on Logic Programming (ICLP 2004)*, number 3132 in *LNCS*, pages 269–283, St. Malo, France, 2004. Springer-Verlag.
- [4] James Cheney. A dependent nominal type theory. *Logical Methods in Computer Science*, 8(1), 2012.
- [5] James Cheney and Christian Urban. Nominal logic programming. *ACM Transactions on Programming Languages and Systems*, 30(5):26, August 2008.
- [6] Roy L. Crole and Frank Nebel. Nominal lambda calculus: An internal language for FM-cartesian closed categories. In *MFPS*, 2013. In press.
- [7] M. J. Gabbay and J. Cheney. A sequent calculus for nominal logic. In *LICS 2004*, pages 139–148. IEEE, 2004.
- [8] M. J. Gabbay and A. M. Pitts. A new approach to abstract syntax with variable binding. *Formal Aspects of Computing*, 13:341–363, 2002.
- [9] Murdoch Gabbay. Fresh logic: proof-theory and semantics for FM and nominal techniques. *J. Applied Logic*, 5(2):356–387, 2007.

- [10] Andrew Gacek, Dale Miller, and Gopalan Nadathur. Nominal abstraction. *Inf. Comput.*, 209(1):48–73, 2011.
- [11] Furio Honsell, Marino Miculan, and Ivan Scagnetto. The theory of contexts for first order and higher order abstract syntax. In *TOSCA*, volume 62 of *Electronic Notes on Theoretical Computer Science*, 2001.
- [12] M. Miculan, I. Scagnetto, and F. Honsell. Translating specifications from nominal logic to CIC with the theory of contexts. In R. Pollack, editor, *MERLIN*, pages 41–49, Tallinn, Estonia, September 2005. ACM Press.
- [13] Dale Miller and Alwen Tiu. A proof theory for generic judgments. *ACM Trans. Comput. Logic*, 6(4):749–783, 2005.
- [14] Sara Negri and Jan von Plato. *Structural Proof Theory*. Cambridge University Press, 2001.
- [15] A. M. Pitts. Nominal logic, a first order theory of names and binding. *Information and Computation*, 183:165–193, 2003.
- [16] A. M. Pitts. Structural recursion with locally scoped names. *Journal of Functional Programming*, 21(3):235–286, 2011.
- [17] Ulrich Schöpp and Ian Stark. A dependent type theory with names and binding. In *CSL 2004*, number 3210 in LNCS, pages 235–249, Karpacz, Poland, 2004.
- [18] A. S. Troelstra and H. Schwichtenberg. *Basic Proof Theory*. Number 43 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, second edition, 2000.
- [19] C. Urban, A. M. Pitts, and M. J. Gabbay. Nominal unification. *Theoretical Computer Science*, 323(1–3):473–497, 2004.