A Simpler Proof Theory for Nominal Logic

James Cheney

Abstract

Nominal logic is a variant of first-order logic which 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 paper 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 conservativity with respect to Pitts' axiomatization of nominal logic; these proofs are considerably simpler for our system.

1 Introduction

Nominal logic [8] 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 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 [6].

One of the most interesting features of nominal logic is the presence of a novel form of quantification: quantification over *fresh names*. The formula $\mathsf{N}\mathsf{a}.\varphi$ 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 N-quantifier is the fact that in the presence of infinitely many names, a fresh name can be chosen for any finitely-supported value, whereas equally-fresh names are indistinguishable. As a result, a property $\varphi(a)$ holds for a fresh name a if and only if it holds for all fresh names; in either case, we say that $\mathsf{N}a.\varphi$ holds.

Several formalizations of nominal logic have been investigated. Pitts introduced nominal logic as a Hilbert-style axiomatic system. Gabbay [4] proposed Fresh Logic (FL), an intuitionistic Gentzen-style natural deduction system. Gabbay and Cheney [5] 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 [9].

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 N. 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 report we present a new and simpler sequent calculus for nominal logic (developed in the course of the author's dissertation research [3]). Its main novelty is the use of a *freshness context* to manage freshness information needed in reasoning about N-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 report will be used as the basis of additional results, including an improved proof-theoretic semantics of nominal logic programming and the development of a sound and complete embedding of $FO\lambda^{\nabla}$ (another logic with a self-dual "freshness" quantifier) into nominal logic.

Swapping

(CS_1) $\quad \forall a{:}\nu, x{:}\tau.\ (a\ a) \cdot x \approx x$

(CS_2) $\quad \forall a, a'{:}\nu, x{:}\tau.\ (a\ a') \cdot (a\ a') \cdot x \approx x$

(CS_3) $\quad \forall a, a'{:}\nu.\ (a\ a') \cdot a \approx a'$

Equivariance

(CE_1) $\quad \forall a, a'{:}\nu, b, b'{:}\nu', x{:}\tau.\ (a\ a') \cdot (b\ b') \cdot x \approx ((a\ a') \cdot b\ (a\ a') \cdot b') \cdot (a\ a') \cdot x$

(CE_2) $\quad \forall a, a'{:}\nu, b{:}\nu', x{:}\tau.\ b \# x \supset (a\ a') \cdot b \# (a\ a') \cdot x$

(CE_3) $\quad \forall a, a'{:}\nu, \overline{x}{:}\overline{\tau}.\ (a\ a') \cdot f(\overline{x}) \approx f((a\ a') \cdot \overline{x})$

(CE_4) $\quad \forall a, a'{:}\nu, \overline{x}{:}\overline{\tau}.\ p(x) \supset p((a\ a') \cdot \overline{x})$

(CE_5) $\quad \forall b, b'{:}\nu', a{:}\nu, x{:}\tau.\ (b\ b') \cdot (\langle a \rangle x) \approx \langle (b\ b') \cdot a \rangle ((b\ b') \cdot x)$

Freshness

(CF_1) $\quad \forall a, a'{:}\nu, x{:}\tau.\ a \# x \wedge a' \# x \supset (a\ a') \cdot x \approx x$

(CF_2) $\quad \forall a, a'{:}\nu.\ a \# a' \iff a \not\approx a'$

(CF_3) $\quad \forall a{:}\nu, a'{:}\nu'.\ a \# a'$

(CF_4) $\quad \forall \overline{x}{:}\overline{\tau}.\ \exists a{:}\nu.\ a \# \overline{x}$

И-quantifier

(CQ) $\quad \forall \overline{x}.(И a{:}\nu.\ \varphi) \iff (\exists a{:}\nu.\ a \# \overline{x} \wedge \varphi)$

where $FV(И a.\varphi) \subseteq \{\overline{x}\}$

Abstraction

(CA_1) $\quad \forall a, a'{:}\nu, x, x'{:}\tau.\ \langle a \rangle x \approx \langle a' \rangle x' \iff (a \approx a' \wedge x \approx x') \vee (a' \# x \wedge x' \approx (a\ a') \cdot x)$

(CA_2) $\quad \forall y{:}\langle \nu \rangle \tau . \exists a{:}\nu, x{:}\tau.\ y \approx \langle a \rangle x$

Figure 1: Axioms of Classical Nominal Logic

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 $fresh_{\nu\tau} : \nu \times \tau$ for each name type ν and type τ. In addition, nominal logic includes two special function symbols $swap_{\nu\tau} : \nu \times \nu \times \tau \to \tau$ and $abs_{\nu\tau} : \nu \times \tau \to \langle \nu \rangle \tau$, 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 И. A И-quantified formula $И x{:}\nu.\varphi$ 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 (CS_1), swapping is involutive (CS_2), and swapping exchanges names (CS_3).
- *Equivariance axioms (CE)*: prescribe the *equivariance* property, namely that all relations are preserved by and all function symbols commute with swapping. In particular, (CE_1) says that the swapping function symbol itself is equivariant; (CE_2) says that freshness is equivariant, (CE_3) 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 (often in concert 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.

- *И-quantifier axiom* (CQ): Pitts' original formalization introduced no new inference rules for И. Instead, И was defined using the axiom scheme Q, which asserts $\forall \overline{x}.(И a.\varphi \iff \exists a.a \# \overline{x} \wedge \varphi)$, where $FV(\varphi) \subseteq \{a, \overline{x}\}$.

- *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 argued that this is a natural generalization of α-equivalence in, for example, the lambda-calculus [6]; 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 И have been considered in previous work. Pitts [8] proposed sequent and natural deduction rules for И based on the observation that

$$\forall a.(a \# \overline{x} \supset \varphi(a, \overline{x})) \supset И a.\varphi(a, \overline{x}) \supset \exists a.(a \# \overline{x} \wedge \varphi(a, \overline{x})) \ .$$

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

Gabbay [4] 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 [5] presented a similar sequent calculus called FL_{Seq}. Both FL and FL_{Seq} had complex rules for И. In FL, Gabbay introduced a technical device called *slices* for obtaining rules that are closed under substitution Technically, a slice $\varphi[a\#\overline{u}]$ of a formula φ is a decomposition of the formula as $\varphi(a, \overline{x})[\overline{u}/\overline{x}]$ for fresh variables $\overline{x}$, such that a does not appear in any of the $\overline{u}$. Slices were used in both FL and FL_{Seq} to deal with И (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 $И L$ was similar to $\exists L$ and $И R$ 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 αProlog, a logic programming language based on nominal logic [1], suggests a much simpler reading of И as a proof-search operation than that implied by the FL-style rules. In αProlog, when a И-quantifier is encountered (either in a goal or program clause), proof search proceeds by generating a fresh name a to be used for the И-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 search for a suitable slice of the И-quantified formula. This reading seems much less deterministic than that employed in αProlog.

In this paper we present a simplified sequent calculus for nominal logic, called $NL^{\Rightarrow}$, in which slices are not needed in the rules for И (or anywhere else). Following Urban, Pitts, and Gabbay [11, 4], we employ a new syntactic class of *name-symbols* $\mathsf{a}, \mathsf{b}, \ldots$. Like variables, such name-symbols may be bound (by И), but unlike variables, two distinct name-symbols are always regarded as denoting distinct name values. In

$$\frac{\Gamma, a \# \overline{x} \Rightarrow \varphi, \Delta \quad (\dagger)}{\Gamma \Rightarrow \mathsf{И}a.\varphi, \Delta}\ \mathsf{И}R \qquad \frac{\Gamma, a \# \overline{x}, \varphi \Rightarrow \Delta \quad (\dagger)}{\Gamma, \mathsf{И}a.\varphi \Rightarrow \Delta}\ \mathsf{И}L \qquad (NL)$$

$$\frac{\Gamma \vdash u \# \overline{t} \quad \Gamma \vdash \varphi[u/\mathsf{a}] \quad (*)}{\Gamma \vdash \mathsf{И}\mathsf{a}.\varphi}\ \mathsf{И}I \qquad \frac{\Gamma \vdash \mathsf{И}\mathsf{a}.\varphi \qquad \Gamma \vdash u \# \overline{t} \quad \Gamma, \varphi[u/\mathsf{a}] \vdash \psi \quad (*)}{\Gamma \Rightarrow \psi}\ \mathsf{И}E \qquad (FL)$$

$$\frac{\Gamma, u \# \overline{t} \Rightarrow \varphi[u/a] \quad (*)}{\Gamma, u \# \overline{t} \Rightarrow \mathsf{И}a.\varphi}\ \mathsf{И}R \qquad \frac{\Gamma, u \# \overline{t}, \varphi[u/a] \Rightarrow \psi \quad (*)}{\Gamma, u \# \overline{t}, \mathsf{И}a.\varphi \Rightarrow \psi}\ \mathsf{И}L \qquad (FL_{Seq})$$

$$\frac{\Gamma, a \# \overline{t} \Rightarrow \varphi \quad (*), (**)}{\Gamma \Rightarrow \mathsf{И}a.\varphi}\ \mathsf{И}R \qquad \frac{\Gamma, a \# \overline{t}, \varphi \Rightarrow \psi \quad (*), (**)}{\Gamma, \mathsf{И}a.\varphi \Rightarrow \psi}\ \mathsf{И}L \qquad (FL'_{Seq})$$

$$\frac{\Sigma \# \mathsf{a} : \Gamma \Rightarrow \varphi \quad (\mathsf{a} \notin \Sigma)}{\Sigma : \Gamma \Rightarrow \mathsf{И}\mathsf{a}.\varphi}\ \mathsf{И}R \qquad \frac{\Sigma \# \mathsf{a} : \Gamma, \varphi \Rightarrow \psi \quad (\mathsf{a} \notin \Sigma)}{\Sigma : \Gamma, \mathsf{И}\mathsf{a}.\varphi \Rightarrow \psi}\ \mathsf{И}L \qquad (NL^{\Rightarrow})$$

$$(\dagger)\ \overline{x} = FV(\Gamma, \mathsf{И}a.\varphi, \Delta) \quad (*)\ \varphi = \varphi[a\#\overline{t}] \quad (**)\ a \notin FV(\Gamma, \psi)$$

Figure 2: Evolution of rules for И

place of slices, we introduce contexts that encode information about freshness as well as identifying the types of variables and name-symbols. Specifically, contexts $\Sigma\#\mathsf{a}{:}\nu$ 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 И (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 paper 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 of conservativity of $NL^{\Rightarrow}$ relative to NL. Section 5 concludes.

The sequent calculus in Section 3 is (except for minor changes) the one presented in Chapter 4 of the author's dissertation [3].

3 Sequent Calculus

3.1 Syntax

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

$$\begin{array}{rcl} \tau & ::= & o \mid \delta \mid \nu \mid \tau \to \tau' \mid \tau \times \tau' \mid \langle\nu\rangle\tau \\ t, u & ::= & c \mid x \mid \mathsf{a} \mid \lambda x{:}\tau.t \mid t\ u \mid \pi_i(t) \mid \langle t, u\rangle \mid (a\ b) \cdot t \mid \langle a\rangle t \\ \varphi, \psi & ::= & \top \mid \bot \mid t \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \varphi \supset \psi \mid \forall x.\varphi \mid \exists x.\varphi \mid \mathsf{И}\mathsf{a}.\varphi \end{array}$$

The base types are the type o of propositions, datatypes δ and name-types ν; additional types are formed using the function and abstraction type constructors. Variables x, y are drawn from a countably infinite set V; also, name-symbols a, b are drawn from a countably infinite set A disjoint form V. The letters a, b are typically used for terms of some name-type ν. Note that λ-terms with surjective pairing are included in this language and are handled in a traditional fashion. In particular, terms are considered equal up to $\alpha\beta\eta$-equivalence in the conventional sense. Negation and logical equivalence are defined as follows:

$$\neg\varphi = (\varphi \supset \bot) \qquad \varphi \iff \psi = (\varphi \supset \psi) \wedge (\psi \supset \varphi)$$

The base type o is used for formulas. However, quantification is limited to types not mentioning o. We assume given a signature that maps constant symbols c to types τ, and containing at least the following

$$
\begin{array}{rcl}
FV(x) & = & \{x\} \\
FV(\mathsf{a}) & = & \varnothing \\
FV(Qx.t) & = & FV(t) - \{x\} \quad (Q \in \{\lambda, \forall, \exists\}) \\
FV(\mathit{И}\mathsf{a}.\varphi) & = & FV(t)
\end{array}
$$

$$
\begin{array}{rcl}
FN(x) & = & \varnothing \\
FN(\mathsf{a}) & = & \{\mathsf{a}\} \\
FN(Qx.t) & = & FN(t) \quad (Q \in \{\lambda, \forall, \exists\}) \\
FN(\mathit{И}\mathsf{a}.\varphi) & = & FN(t) - \{\mathsf{a}\}
\end{array}
$$

$$
\begin{array}{rcl}
F\alpha(c) = F\alpha(\top) = F\alpha(\bot) & = & \varnothing \\
F\alpha(t\ u) = F\alpha(\langle t, u\rangle) = F\alpha(t \circ u) & = & F\alpha(t) \cup F\alpha(u) \quad (\circ \in \{\wedge, \vee, \supset\}) \\
F\alpha(\pi_i(t)) & = & F\alpha(t) \\
F\alpha((a\ b) \cdot t) & = & F\alpha(a) \cup F\alpha(b) \cup F\alpha(t)
\end{array}
$$

$$
FVN(t) = FV(t) \cup FN(t)
$$

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

declarations:

$$
\begin{array}{rcl@{\qquad}rcl}
eq_\tau & : & \tau \times \tau \to o & fresh_{\nu\tau} & : & \nu \times \tau \to o \\
swap_{\nu\tau} & : & \nu \times \nu \times \tau \to \tau & abs_{\nu\tau} & : & \nu \times \tau \to \langle\nu\rangle\tau
\end{array}
$$

for name-types ν and types τ. The subscripts are dropped when clear from context. The notations $t \approx u$, $t \# u$, $(a\ b) \cdot t$, and $\langle t\rangle u$ are syntactic sugar for the terms $eq(t,u)$, $fresh(t,u)$, $swap(a,b,t)$, and $abs(t,u)$, respectively. We write ω for a term that may be either a name-symbol a or a variable x. 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).

Remark 1. The inclusion of λ-terms and identification of terms and formulas with bound names up to α-equivalence may be objectionable because it appears that we are circularly attempting to define binding in terms of binding. This is not the case. A key contribution of Gabbay and Pitts' approach is that it shows how one can formally justify an informal (and traditional) approach to binding syntax by constructing syntax trees modulo α-equivalence as simple mathematical objects in a particularly clever way [6][3, Ch. 3–4]. We assume that this or some other standard technique for dealing with binding is acting behind the scenes.

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

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

We often abbreviate $\cdot, x{:}\tau$ and $\cdot\#\mathsf{a}{:}\nu$ to $x{:}\tau$ and $\mathsf{a}{:}\nu$ respectively. We write $\omega{:}\tau \in \Sigma$ if the binding $\omega{:}\tau$ is present in Σ. We write $\Sigma; \Sigma'$ for the result of concatenating two contexts such that $FVN(\Sigma) \cap FVN(\Sigma') = \varnothing$.

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 irrelevant. There are only two nonstandard rules for typechecking; the remaining rules (shown in Figure 4) are standard. Terms viewed as formulas must, as usual, be of type o. Quantification using $\forall$ and $\exists$ is only allowed over types not mentioning o; $\mathit{И}$-quantification is only allowed over name-types.

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:

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

$$\frac{c : \tau}{\Sigma \vdash c : \tau} \qquad \frac{\Sigma \vdash t : \sigma_1 \quad \Sigma \vdash u : \sigma_2}{\Sigma \vdash \langle t, u\rangle : \sigma_1 \times \sigma_2} \qquad \frac{\Sigma \vdash t : \tau_1 \times \tau_2}{\Sigma \vdash \pi_i t : \tau_i} \qquad \frac{\omega : \tau \in \Sigma}{\Sigma \vdash \omega : \tau}$$

$$\frac{\Sigma, x{:}\tau \vdash t : \sigma}{\Sigma \vdash \lambda x.t : \tau \to \sigma} \qquad \frac{\Sigma \vdash t : \tau \to \sigma \quad \Sigma \vdash u : \tau}{\Sigma \vdash t\ u : \sigma} \qquad \frac{}{\Sigma \vdash \top, \bot : o} \qquad \frac{\Sigma \vdash \varphi, \psi : o \quad (\circ \in \{\wedge, \vee, \supset\})}{\Sigma \vdash \varphi \circ \psi : o}$$

$$\frac{\Sigma, x{:}\tau \vdash \varphi : o}{\Sigma \vdash \forall x{:}\tau.\varphi, \exists x{:}\tau.\varphi : o} \qquad \frac{\Sigma \# \mathsf{a} \vdash \varphi : o}{\Sigma \vdash И\mathsf{a}{:}\nu.\varphi : o}$$

Figure 4: Well-formedness rules

For example, $\mathsf{a} \# x, \mathsf{b} \# \mathsf{a}$, and $\mathsf{b} \# f\ x\ y \in |x\#\mathsf{a}, y\#\mathsf{b}|$. We say that Σ' is stronger than Σ ($\Sigma \leq \Sigma'$) if $Tm_\Sigma \subseteq Tm_{\Sigma'}$ and $|\Sigma| \subseteq |\Sigma'|$. For example, $\mathsf{a}, x \leq x\#\mathsf{a}, y$. The following routine properties hold:

Lemma 2 (Term Weakening). *If $\Sigma \vdash t : \tau$ and $\Sigma \leq \Sigma'$ then $\Sigma' \vdash t : \tau$.*

Lemma 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 freshness 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 5), whereas *Intuitionistic* $NL^{\Rightarrow}$ ($INL^{\Rightarrow}$) is based on the multiple-conclusion intuitionistic calculus **G3im** (see Figure 6). Both versions include two additional *logical rules*, $И L$ and $И R$, shown in Figure 2($NL^{\Rightarrow}$). In addition, $NL^{\Rightarrow}$ includes several *nonlogical rules* (Figure 8) defining the properties of swapping, equality, freshness and abstraction.

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 without affecting cut-elimination [7]. 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. 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.

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 $И L$ and $И R$.

Figure 9 lists some rules whose admissibility in $NL^{\Rightarrow}$ will be shown in the next section.

3.3 Structural Properties

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 4 (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 5 (Context Weakening). *If $\vdash_n \Sigma : \Gamma \Rightarrow \Delta$ and $\Sigma \leq \Sigma'$ then $\vdash_n \Sigma' : \Gamma \Rightarrow \Delta$.*

Lemma 6 (Substitution). *If $\vdash_n \Sigma \vdash t : \tau$ and $\Sigma, x{:}\tau; \Sigma' : \Gamma \Rightarrow \Delta$ then $\vdash_n \Sigma; \Sigma' : \Gamma[t/x] \Rightarrow \Delta[t/x]$.*

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

For F we have a derivation

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

$$\frac{}{\Sigma : \Gamma, p\,\bar{t} \Rightarrow p\,\bar{t}, \Delta}\ hyp$$

$$\frac{}{\Sigma : \Gamma \Rightarrow \top, \Delta}\ \top R \qquad \frac{}{\Sigma : \Gamma, \bot \Rightarrow \Delta}\ \bot L$$

$$\frac{\Sigma : \Gamma \Rightarrow \varphi, \Delta \quad \Sigma : \Gamma \Rightarrow \psi, \Delta}{\Sigma : \Gamma \Rightarrow \varphi \wedge \psi, \Delta}\ \wedge R \qquad \frac{\Sigma : \Gamma, \varphi_1, \varphi_2 \Rightarrow \Delta}{\Sigma : \Gamma, \varphi_1 \wedge \varphi_2 \Rightarrow \Delta}\ \wedge L$$

$$\frac{\Sigma : \Gamma \Rightarrow \varphi_1, \varphi_2, \Delta}{\Sigma : \Gamma \Rightarrow \varphi_1 \vee \varphi_2, \Delta}\ \vee R \qquad \frac{\Sigma : \Gamma, \varphi \Rightarrow \Delta \quad \Gamma, \psi \Rightarrow \Delta}{\Sigma : \Gamma, \varphi \vee \psi \Rightarrow \Delta}\ \vee L$$

$$\frac{\Sigma : \Gamma, \varphi \Rightarrow \psi, \Delta}{\Sigma : \Gamma \Rightarrow \varphi \supset \psi, \Delta}\ \supset R \qquad \frac{\Sigma : \Gamma \Rightarrow \varphi, \Delta \quad \Sigma : \Gamma, \psi \Rightarrow \Delta}{\Sigma : \Gamma, \varphi \supset \psi \Rightarrow \Delta}\ \supset L$$

$$\frac{\Sigma, x : \Gamma \Rightarrow \varphi, \Delta \quad (x \notin \Sigma)}{\Sigma : \Gamma \Rightarrow \forall x.\varphi, \Delta}\ \forall R \qquad \frac{\Sigma \vdash t : \sigma \quad \Sigma : \Gamma, \forall x{:}\sigma.\varphi, \varphi\{t/x\} \Rightarrow \Delta}{\Sigma : \Gamma, \forall x{:}\sigma.\varphi \Rightarrow \Delta}\ \forall L$$

$$\frac{\Sigma \vdash t : \sigma \quad \Sigma : \Gamma \Rightarrow \exists x{:}\sigma.\varphi, \varphi\{t/x\}, \Delta}{\Sigma : \Gamma \Rightarrow \exists x{:}\sigma.\varphi, \Delta}\ \exists R \qquad \frac{\Sigma, x : \Gamma, \varphi \Rightarrow \Delta \quad (x \notin \Sigma)}{\Sigma : \Gamma, \exists x.\varphi \Rightarrow \Delta}\ \exists L$$

$$\frac{\Sigma : \Gamma, t \approx t \Rightarrow \Delta}{\Sigma : \Gamma \Rightarrow \Delta}\ \approx R \qquad \frac{\Sigma : \Gamma, t \approx u, P(t), P(u) \Rightarrow \Delta}{\Sigma : \Gamma, t \approx u, P(t) \Rightarrow \Delta}\ \approx S$$

Figure 5: Classical first-order equational sequent calculus (**G3c**)

$$\frac{\Sigma : \Gamma, \varphi \Rightarrow \psi}{\Sigma : \Gamma \Rightarrow \varphi \supset \psi, \Delta}\ \supset R \qquad \frac{\Sigma : \Gamma, \varphi \supset \psi \Rightarrow \varphi \quad \Sigma : \Gamma, \psi \Rightarrow \Delta}{\Sigma : \Gamma, \varphi \supset \psi \Rightarrow \Delta}\ \supset L$$

$$\frac{\Sigma, x : \Gamma \Rightarrow \varphi \quad (x \notin \Sigma)}{\Sigma : \Gamma \Rightarrow \forall x.\varphi, \Delta}\ \forall R \qquad \frac{\Sigma \vdash t : \sigma \quad \Sigma : \Gamma, \forall x{:}\sigma.\varphi, \varphi\{t/x\} \Rightarrow \Delta}{\Sigma : \Gamma, \forall x{:}\sigma.\varphi \Rightarrow \Delta}\ \forall L$$

$$\frac{\Sigma \vdash t : \sigma \quad \Sigma : \Gamma \Rightarrow \exists x{:}\sigma.\varphi, \varphi\{t/x\}, \Delta}{\Sigma : \Gamma \Rightarrow \exists x{:}\sigma.\varphi, \Delta}\ \exists R \qquad \frac{\Sigma, x : \Gamma, \varphi \Rightarrow \Delta \quad (x \notin \Sigma)}{\Sigma : \Gamma, \exists x.\varphi \Rightarrow \Delta}\ \exists L$$

Figure 6: Variant rules for the intuitionistic multiple-conclusion calculus (**G3im**)

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

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

Now we consider some nontrivial structural properties.

Lemma 8 (Admissibility of EVL, EVR). *The EVL and EVR rules*

$$\frac{\Sigma : \Gamma, (a\ b) \cdot \varphi \Rightarrow \Delta}{\Sigma : \Gamma, \varphi \Rightarrow \Delta}\ EVL \qquad \frac{\Sigma : \Gamma \Rightarrow (a\ b) \cdot \varphi, \Delta}{\Sigma : \Gamma \Rightarrow \varphi, \Delta}\ EVR$$

(S_1)	$(a\ a) \cdot x \approx x$	(E_4)	$(a\ b) \cdot \lambda x.e[x] \approx \lambda x.(a\ b) \cdot e[(a\ b) \cdot x]$
(S_2)	$(a\ b) \cdot (a\ b) \cdot x \approx x$	(F_1)	$a \# x \wedge b \# x \supset (a\ b) \cdot x \approx x$
(S_3)	$(a\ b) \cdot a \approx b$	(F_2)	$a \# b \quad (a : \nu, b : \nu', \nu \not\equiv \nu')$
(E_1)	$(a\ b) \cdot c \approx c$	(F_3)	$a \# a \supset \bot$
(E_2)	$(a\ b) \cdot (t\ u) \approx ((a\ b) \cdot t)\ ((a\ b) \cdot u)$	(F_4)	$a \# b \vee a \approx b$
(E_3)	$p(x) \supset p((a\ b) \cdot x)$	(A_1)	$a \# y \wedge x \approx (a\ b) \cdot y \supset \langle a \rangle x \approx \langle b \rangle y$

Figure 7: Equational and freshness axioms

$$\frac{\Sigma : \Gamma, P, Q_1 \Rightarrow \Delta \quad \cdots \quad \Sigma : \Gamma, P, Q_n \Rightarrow \Delta}{\Sigma : \Gamma, P \Rightarrow \Delta}\ Ax \qquad \bigwedge \overline{P} \supset \bigvee \overline{Q} \text{ an axiom instance}$$

$$\frac{\Sigma : \Gamma, \langle a\rangle t \approx \langle b\rangle u, a \approx b, t \approx u \Rightarrow \Delta \quad \Sigma : \Gamma, \langle a\rangle t \approx \langle b\rangle u, a \# u, t \approx (a\ b)\cdot u \Rightarrow \Delta}{\Sigma : \Gamma, \langle a\rangle t \approx \langle b\rangle u \Rightarrow \Delta}\ A_2$$

$$\frac{\Sigma \vdash t : \langle \nu\rangle\sigma \quad \Sigma, a{:}\nu, x{:}\sigma : \Gamma, t \approx \langle a\rangle x \Rightarrow \Delta \quad (a, x \notin \Sigma)}{\Sigma : \Gamma \Rightarrow \Delta}\ A_3$$

$$\frac{\Sigma\#\mathsf{a} : \Gamma \Rightarrow \Delta \quad (\mathsf{a} \notin \Sigma)}{\Sigma : \Gamma \Rightarrow \Delta}\ F \qquad \frac{\Sigma : \Gamma, t \# u \Rightarrow \Delta \quad (t \# u \in |\Sigma|)}{\Sigma : \Gamma \Rightarrow \Delta}\ \Sigma\#$$

Figure 8: Nonlogical rules

$$\frac{\Sigma : \Gamma \Rightarrow \Delta}{\Sigma : \Gamma, \varphi \Rightarrow \Delta}\ W \qquad \frac{}{\Sigma : \Gamma, \varphi \Rightarrow \varphi, \Delta}\ hyp^* \qquad \frac{\Sigma : \Gamma \Rightarrow \varphi, \Delta \quad \Sigma : \Gamma', \varphi \Rightarrow \Delta'}{\Sigma : \Gamma, \Gamma' \Rightarrow \Delta, \Delta'}\ cut$$

$$\frac{\Sigma : \Gamma, \varphi, \varphi \Rightarrow \Delta}{\Sigma : \Gamma, \varphi \Rightarrow \Delta}\ C \qquad \frac{\Sigma : \Gamma, (a\ b)\cdot\varphi \Rightarrow \Delta}{\Sigma : \Gamma, \varphi \Rightarrow \Delta}\ EVL \qquad \frac{\Sigma : \Gamma \Rightarrow (a\ b)\cdot\varphi, \Delta}{\Sigma : \Gamma \Rightarrow \Delta, \varphi}\ EVR$$

Figure 9: Some admissible rules of $NL^{\Rightarrow}$

where φ is an arbitrary formula, are admissible.

Proof. We proceed by induction 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

$$\frac{}{\Gamma, (a\ b)\cdot A \Rightarrow (a\ b)\cdot A, \Delta}$$

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

$$\frac{\dfrac{}{\Sigma : \Gamma, (a\ b)\cdot A \Rightarrow (a\ b)\cdot A, \Delta}\ E_p}{\Sigma : \Gamma, A \Rightarrow (a\ b)\cdot A, \Delta}$$

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

If the derivation is of the form

$$\frac{\Sigma : \Gamma, (a\ b)\cdot P \supset (a\ b)\cdot Q \Rightarrow (a\ b)\cdot P, \Delta \quad \Sigma : \Gamma, (a\ b)\cdot Q \Rightarrow \Delta}{\Sigma : \Gamma, (a\ b)\cdot P \supset (a\ b)\cdot Q \Rightarrow \Delta}\ \supset L$$

then using the admissibility of EVR and EVL on the left and EVR on the right we obtain

$$\frac{\dfrac{\Sigma : \Gamma, (a\ b)\cdot P \supset (a\ b)\cdot Q \Rightarrow (a\ b)\cdot P, \Delta}{\Sigma : \Gamma, P \supset Q \Rightarrow P, \Delta}\ EVL, EVR \quad \dfrac{\Sigma : \Gamma, (a\ b)\cdot Q \Rightarrow \Delta}{\Sigma : \Gamma, Q \Rightarrow \Delta}\ EVL}{\Sigma : \Gamma, P \supset Q \Rightarrow \Delta}\ \supset L$$

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

$$\frac{}{\Gamma, (a\ b)\cdot A \Rightarrow (a\ b)\cdot A, \Delta}$$

Then we can derive

$$\frac{\dfrac{}{\Gamma, (a\ b)\cdot(a\ b)\cdot A \Rightarrow A, \Delta}\ \approx, hyp}{\Gamma, (a\ b)\cdot A \Rightarrow A, \Delta}\ E_p$$

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

If the derivation is of the form

$$\frac{\Gamma, (a\ b)\cdot P \Rightarrow (a\ b)\cdot Q, \Delta}{\Gamma \Rightarrow (a\ b)\cdot P \supset (a\ b)\cdot Q, \Delta}\ \supset R$$

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

$$\frac{\dfrac{\Gamma, (a\ b)\cdot P \Rightarrow (a\ b)\cdot Q, \Delta}{\Gamma, P \Rightarrow Q, \Delta}\ EVL, EVR}{\Gamma \Rightarrow P \supset Q, \Delta}\ \supset R$$

This transformation is obviously logical height-preserving by induction. □

Lemma 9 (Swapping Fresh Names). *Suppose* $\Sigma\#\mathsf{a} \vdash \varphi(\mathsf{a}) : o$*. Then the rule*

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

is admissible using nonlogical axioms only.

Proof. Let $\overline{X} = FV(\Sigma)$. The derivation is roughly as follows:

$$\frac{\dfrac{\dfrac{\Sigma\#\mathsf{a}\#\mathsf{b} : \Gamma, \mathsf{a}\ \#\ \overline{x}, \mathsf{b}\ \#\ \overline{x}, \varphi(\mathsf{b}) \Rightarrow \Delta}{\Sigma\#\mathsf{a}\#\mathsf{b} : \Gamma, \mathsf{a}\ \#\ \overline{x}, \mathsf{b}\ \#\ \overline{x}, (\mathsf{a}\ \mathsf{b})\cdot\varphi(\mathsf{a}) \Rightarrow \Delta}\ Ax}{\Sigma\#\mathsf{a}\#\mathsf{b} : \Gamma, \mathsf{a}\ \#\ \overline{x}, \mathsf{b}\ \#\ \overline{x}, \varphi(\mathsf{a}) \Rightarrow \Delta}\ EVL}{\Sigma\#\mathsf{a}\#\mathsf{b} : \Gamma, \varphi(\mathsf{a}) \Rightarrow \Delta}\ \Sigma\#$$

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

Lemma 10 (Admissibility of hyp^*). *The hyp^* rule*

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

where φ is an arbitrary formula, 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 $\varphi = И\mathsf{a}.P$ is as follows. By induction, we may assume that $\Sigma\#\mathsf{a}\#\mathsf{b} : \Gamma, \varphi(\mathsf{b}) \Rightarrow \varphi(\mathsf{b}), \Delta$ is derivable. We derive

$$\frac{\dfrac{\dfrac{\Sigma\#\mathsf{a}\#\mathsf{b} : \Gamma, \varphi(\mathsf{b}) \Rightarrow \varphi(\mathsf{b}), \Delta}{\Sigma\#\mathsf{a}\#\mathsf{b} : \Gamma, \varphi(\mathsf{a}) \Rightarrow \varphi(\mathsf{b}), \Delta}\ \text{Lemma 9}}{\Sigma\#\mathsf{a} : \Gamma, \varphi(\mathsf{a}) \Rightarrow И\mathsf{a}.P, \Delta}\ И R}{\Sigma : \Gamma, И\mathsf{a}.P \Rightarrow И\mathsf{a}.P, \Delta}\ И L$$

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

Lemma 11 (Inversion). *The $\wedge L$, $\vee L$, $\supset L$, $\exists L$, $\forall R$, $И L$, and $И R$ rules are invertible; that is,*

1. *If* $\vdash^l_n \Sigma : \Gamma, \varphi \wedge \psi \Rightarrow \Delta$ *then* $\vdash^l_n \Sigma : \Gamma, \varphi, \psi \Rightarrow \Delta$.
2. *If* $\vdash^l_n \Sigma : \Gamma, \varphi \vee \psi \Rightarrow \Delta$ *then* $\vdash^l_n \Sigma : \Gamma, \varphi \Rightarrow \Delta$ *or* $\vdash^l_n \Sigma : \Gamma, \psi \Rightarrow \Delta$.
3. *If* $\vdash^l_n \Sigma : \Gamma, \varphi \supset \psi \Rightarrow \Delta$ *then* $\vdash^l_n \Sigma : \Gamma, \psi \Rightarrow \Delta$.
4. *If* $\vdash^l_n \Sigma : \Gamma, \exists x.\varphi \Rightarrow \Delta$ *then* $\vdash^l_n \Sigma, y : \Gamma, \varphi[y/x] \Rightarrow \Delta$.
5. *If* $\vdash^l_n \Sigma : \Gamma \Rightarrow \Delta, \forall x.\varphi$ *then* $\vdash^l_n \Sigma, y : \Gamma\Delta, \varphi[y/x]$.

6. *If* $\vdash^l_n \Sigma : \Gamma, И\mathsf{a}.\varphi \Rightarrow \Delta$ *then* $\vdash^l_n \Sigma\#\mathsf{a} : \Gamma, \varphi \Rightarrow \Delta$ *for fresh* a.

7. *If* $\vdash^l_n \Sigma : \Gamma \Rightarrow \Delta, И\mathsf{a}.\varphi$ *then* $\vdash^l_n \Sigma\#\mathsf{a} : \Gamma \Rightarrow \Delta, \varphi$ *for fresh* a.

Proof. The proofs for the rules $\wedge L$, $\vee L$, $\supset L$, $\exists L$, $\forall R$ are similar to those for the systems $\mathbf{G3c}$ and $\mathbf{G3im}$ [7].

For $ИL$, 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

$$\frac{\Sigma, x : \Gamma, И\mathsf{a}.\varphi \Rightarrow \psi}{\Sigma : \Gamma, И\mathsf{a}.\varphi \Rightarrow \forall x.\psi}$$

then using the induction hypothesis, we have $\Sigma, x\#\mathsf{b} : \Gamma, \varphi(\mathsf{b}) \Rightarrow \psi$. Using structural weakening we have $\Sigma\#\mathsf{a}, x\#\mathsf{b} : \Gamma, \varphi(\mathsf{b}) \Rightarrow \psi$. Using equivariance and equational reasoning (and the fact that $x \notin FV(\varphi)$), we can derive $\Sigma\#\mathsf{a}, x\#\mathsf{b} : \Gamma, \varphi(\mathsf{a}) \Rightarrow \psi$. Now b is not mentioned in the sequent so using F and $\forall R$ we can derive $\Sigma\#\mathsf{a} : \Gamma, \varphi(\mathsf{b}) \Rightarrow \forall x.\psi$, as desired.

The proof for the invertibility of $ИR$ is symmetric. □

Lemma 12 (Contraction). *If* $\vdash^l_n \Sigma : \Gamma, \varphi, \varphi \Rightarrow \Delta$ *then so is* $\vdash^l_n \Sigma : \Gamma, \varphi \Rightarrow \Delta$. *Similarly, if* $\vdash^l_n \Sigma : \Gamma \Rightarrow \Delta, \varphi, \varphi$ *then* $\vdash^l_n \Sigma : \Gamma \Rightarrow \Delta, \varphi$.

Proof. The proof is by induction on the logical height and secondary induction on the 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 $И\mathsf{a}.\varphi$. 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 $ИL$:

$$\frac{\Sigma\#\mathsf{a} : \Gamma, \varphi(\mathsf{a}), И\mathsf{b}.\varphi(\mathsf{b}) \Rightarrow \Delta}{\Sigma : \Gamma, И\mathsf{a}.\varphi(\mathsf{a}), И\mathsf{b}.\varphi(\mathsf{b}) \Rightarrow \Delta}\ ИL$$

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

$$\dfrac{\dfrac{\Sigma\#\mathsf{a}\#\mathsf{b} : \Gamma, \varphi(\mathsf{a}) \Rightarrow \Delta}{\Sigma\#\mathsf{a} : \Gamma, \varphi(\mathsf{a}) \Rightarrow \Delta}\ F}{\Sigma : \Gamma, И\mathsf{a}.\varphi(\mathsf{a}) \Rightarrow \Delta}\ ИL$$

The proof for right-contraction is symmetric, using the invertibility of $ИR$. □

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 13 (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 [7], 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. [7] or [10]). In addition, Negri and von Plato [7] 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 $\forall$-quantifier rules.

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

  In the first case, we have

  $$\dfrac{\begin{matrix}\Pi\\ \Sigma\#\mathsf{a} : \Gamma, \psi \Rightarrow \Delta, \varphi\end{matrix}}{\Sigma : \Gamma, \forall \mathsf{a}.\psi \Rightarrow \Delta, \varphi}\ \forall L$$

  where $\mathsf{a} \notin \Sigma$. We can weaken Π' to derive $W(\Pi')$ deriving $\Sigma\#\mathsf{a} : \Gamma', \varphi \Rightarrow \Delta'$, and by induction, we have $\Sigma\#\mathsf{a} : \Gamma, \psi, \Gamma' \Rightarrow \Delta, \Delta'$. Then we may derive $\Sigma : \Gamma, \forall \mathsf{a}.\psi, \Gamma' \Rightarrow \Delta, \Delta'$ using $\forall L$.

  In the second case, we have

  $$\dfrac{\begin{matrix}\Pi\\ \Sigma\#\mathsf{a} : \Gamma \Rightarrow \Delta, \psi, \varphi\end{matrix}}{\Sigma : \Gamma \Rightarrow \Delta, \forall \mathsf{a}.\psi, \varphi}\ \forall R$$

  where $\mathsf{a} \notin \Sigma$. We can weaken Π' to get $W(\Pi')$ deriving $\Sigma\#\mathsf{a} : \Gamma' \Rightarrow \Delta'$ and then by induction obtain $\Sigma\#\mathsf{a} : \Gamma', \Gamma \Rightarrow \Delta, \Delta', \psi$. Using $\forall R$ we can derive $\Sigma : \Gamma', \Gamma \Rightarrow \Delta, \Delta', \forall \mathsf{a}.\psi$.
- Right-commuting cases. These cases are exactly symmetric to the left-commuting cases.

  In the first case, we have

  $$\dfrac{\begin{matrix}\Pi'\\ \Sigma\#\mathsf{a} : \Gamma', \varphi, \psi \Rightarrow \Delta'\end{matrix}}{\Sigma : \Gamma', \varphi, \forall \mathsf{a}.\psi \Rightarrow \Delta'}\ \forall L$$

  where $\mathsf{a} \notin \Sigma$. We can weaken Π to derive $W(\Pi)$ deriving $\Sigma\#\mathsf{a} : \Gamma \Rightarrow \Delta, \varphi$, and by induction, we have $\Sigma\#\mathsf{a} : \Gamma, \psi, \Gamma' \Rightarrow \Delta, \Delta'$. Then we may derive $\Sigma : \Gamma, \forall \mathsf{a}.\psi, \Gamma' \Rightarrow \Delta, \Delta'$ using $\forall L$.

  In the second case, we have

  $$\dfrac{\begin{matrix}\Pi'\\ \Sigma\#\mathsf{a} : \Gamma', \varphi \Rightarrow \Delta', \psi\end{matrix}}{\Sigma : \Gamma', \varphi \Rightarrow \Delta', \forall \mathsf{a}.\psi}\ \forall R$$

  where $\mathsf{a} \notin \Sigma$. We can weaken Π to get $W(\Pi)$ deriving $\Sigma\#\mathsf{a} : \Gamma \Rightarrow \Delta, \varphi$ and then by induction obtain $\Sigma\#\mathsf{a} : \Gamma', \Gamma \Rightarrow \Delta, \Delta', \psi$. Using $\forall R$ we can derive $\Sigma : \Gamma', \Gamma \Rightarrow \Delta, \Delta', \forall \mathsf{a}.\psi$.
- Principal cases. In this case, both Π and Π' decompose the cut formula. The only new rule for decomposing formulas on the right is $\forall R$, so the only new principal cut case is when we have

  $$\dfrac{\begin{matrix}\Pi\\ \Sigma\#\mathsf{a} : \Gamma \Rightarrow \Delta, \varphi\end{matrix}}{\Sigma : \Gamma \Rightarrow \Delta, \forall \mathsf{a}.\varphi}\ \forall R \quad \dfrac{\begin{matrix}\Pi'\\ \Sigma\#\mathsf{a} : \Gamma', \varphi \Rightarrow \Delta'\end{matrix}}{\Sigma : \Gamma', \forall \mathsf{a}.\varphi \Rightarrow \Delta'}\ \forall L$$

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

This completes the proof. □

Theorem 14. *Any derivable 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. □

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 $\Rightarrow \bot$. 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 of it contains only constraints.

Proposition 15. *If $\Rightarrow \bot$ has a derivation, then it has one using only nonlogical rules, in which each sequent is of the form $\Gamma \Rightarrow \bot$, where Γ is a constraint set.*

The proof is immediate by observing that only nonlogical rules are applicable to a derivation of $\Gamma \Rightarrow \bot$ where Γ is a constraint set.

This means that nominal logic is consistent if and only if the nonlogical rules are consistent. To prove the consistency of the nonlogical rules, it is necessary to exhibit a model. An appropriate semantics can be defined in terms of the syntax of nominal terms.

Definition 16 (Syntactic Swapping, Equality and Freshness). *Let Tm be the set of swapping-free nominal terms generated by the grammar*

$$t ::= \mathsf{a} \mid \langle\rangle \mid \langle t, u\rangle \mid \langle \mathsf{a}\rangle t \mid f(t)$$

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

$$\begin{array}{rcl}
(\mathsf{a}\ \mathsf{b}) \cdot \mathsf{a} & = & \mathsf{b} \\
(\mathsf{a}\ \mathsf{b}) \cdot \mathsf{b} & = & \mathsf{a} \\
(\mathsf{a}\ \mathsf{b}) \cdot \mathsf{c} & = & \mathsf{c} \quad (\mathsf{a}, \mathsf{b} \neq \mathsf{c}) \\
(\mathsf{a}\ \mathsf{b}) \cdot \langle\rangle & = & \langle\rangle \\
(\mathsf{a}\ \mathsf{b}) \cdot \langle t, u\rangle & = & \langle (\mathsf{a}\ \mathsf{b}) \cdot t, (\mathsf{a}\ \mathsf{b}) \cdot u\rangle \\
(\mathsf{a}\ \mathsf{b}) \cdot f(t) & = & f((\mathsf{a}\ \mathsf{b}) \cdot t) \\
(\mathsf{a}\ \mathsf{b}) \cdot \langle \mathsf{c}\rangle t & = & \langle (\mathsf{a}\ \mathsf{b}) \cdot \mathsf{c}, (\mathsf{a}\ \mathsf{b}) \cdot t\rangle
\end{array}$$

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

$$\frac{(\mathsf{a} \neq \mathsf{b})}{\mathsf{a} \# \mathsf{b}} \qquad \frac{}{\mathsf{a} \# \langle\rangle} \qquad \frac{\mathsf{a} \# t}{\mathsf{a} \# f(t)} \qquad \frac{\mathsf{a} \# t \quad \mathsf{a} \# u}{\mathsf{a} \# \langle t, u\rangle} \qquad \frac{}{\mathsf{a} \# \langle \mathsf{a}\rangle t} \qquad \frac{\mathsf{a} \# t \quad (\mathsf{a} \neq \mathsf{b})}{\mathsf{a} \# \langle \mathsf{b}\rangle t}$$

The nominal equality *relation is defined as follows:*

$$\frac{}{\mathsf{a} \approx \mathsf{a}} \qquad \frac{}{\langle\rangle \approx \langle\rangle} \qquad \frac{t_1 \approx u_1 \quad t_2 \approx u_2}{\langle t_1, t_2\rangle \approx \langle u_1, u_2\rangle} \qquad \frac{t \approx u}{f(t) \approx f(u)} \qquad \frac{t \approx u}{\langle \mathsf{a}\rangle t \approx \langle \mathsf{a}\rangle u} \qquad \frac{t \approx (\mathsf{a}\ \mathsf{b}) \cdot u \quad \mathsf{a} \# u \quad (\mathsf{a} \neq \mathsf{b})}{\langle \mathsf{a}\rangle t \approx \langle \mathsf{b}\rangle u}$$

Proposition 17. *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 18. *Let $\theta : V \to NTm$ be a substitution of ground nominal terms for variables, called an* interpretation. *We lift θ to a function $\theta : NTm \to NTm$ as follows:*

$$\begin{aligned}
\theta(\mathsf{a}) &= \mathsf{a} \\
\theta(\langle\rangle) &= \langle\rangle \\
\theta(\langle t_1, t_2\rangle) &= \langle\theta(t_1), \theta(t_2)\rangle \\
\theta(f(t)) &= f(\theta(t)) \\
\theta((a\ b)\cdot t) &= (\theta(a)\ \theta(b))\cdot\theta(t) \\
\theta(\langle a\rangle t) &= \langle\theta(a)\rangle\theta(t)
\end{aligned}$$

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

We write $\theta \vDash t \approx u$ or $\theta \vDash a \# t$ to indicate that $\theta(t) \approx \theta(u)$ or $\theta(a) \# \theta(t)$ respectively. Similarly, $\theta \vDash \Gamma$ indicates that $\theta \vDash A$ 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 $\theta \vDash A$ ($\theta \vDash \Gamma$) holds in NTm.*

Proposition 19. *The axioms listed in Figure 7 are valid for NTm, in the sense that for each axiom $\bigwedge P \supset \bigvee Q$, if $\theta \vDash P$ then $\theta \vDash Q_i$ for some $Q_i \in Q$.*

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 unit, pairing, abstraction, and other function symbols besides swapping itself are equivariant. In addition, it is not difficult to show that

$$(\mathsf{a}\ \mathsf{a}')\cdot(\mathsf{b}\ \mathsf{b}')\cdot x = ((\mathsf{a}\ \mathsf{a}')\cdot\mathsf{b}\ (\mathsf{a}\ \mathsf{a}')\cdot\mathsf{b}')\cdot(\mathsf{a}\ \mathsf{a}')\cdot x$$

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

$$x \approx y \supset (\mathsf{a}\ \mathsf{b})\cdot x \approx (\mathsf{a}\ \mathsf{b})\cdot y$$

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

$$\mathsf{a} \# x \supset (\mathsf{b}\ \mathsf{b}')\cdot\mathsf{a} \# (\mathsf{b}\ \mathsf{b}')\cdot x$$

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

For the axiom F_1, we must show that if $\mathsf{a} \# x$ and $\mathsf{b} \# x$ then $(\mathsf{a}\ \mathsf{b})\cdot x \approx x$. The proof is by induction on the structure of x. For $x = \langle\rangle$ the result is immediate; similarly, for $x = f(y)$ or $x = \langle y_1, y_2\rangle$ the induction step is straightforward. For $x = \mathsf{c}$, we have $\mathsf{a}, \mathsf{b} \neq \mathsf{c}$ so $(\mathsf{a}\ \mathsf{b})\cdot\mathsf{c} = \mathsf{c} \approx \mathsf{c}$. For $x = \langle\mathsf{c}\rangle y$, there are two cases. If $\mathsf{a}, \mathsf{b} \neq \mathsf{c}$ then we have $\mathsf{a}, \mathsf{b} \# y$ and

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

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

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

For F_3, we need to show that $\mathsf{a} \# \mathsf{a}$ is underivable. This is immediate from the definition of the freshness relation.

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

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

□

Proposition 20. *If $\theta \vDash \langle a\rangle x \approx \langle b\rangle y$ then either $\theta \vDash a \approx b, x \approx y$ or $\theta \vDash a \# y, x \approx (a\ b) \cdot y$.*

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. □

Proposition 21. *If $\theta : \Sigma$ then $\theta \vDash \mathsf{a} \# t$ for each $\mathsf{a} \# t \in |\Sigma|$.*

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 $\mathsf{a} \# \theta(x)$ for each $\mathsf{a} \# x \in |\Sigma|$. □

Theorem 22. *Let Γ be a set of freshness and equality formulas. If $\Sigma : \Gamma \Rightarrow \bot$ is derivable then Γ is unsatisfiable.*

Proof. Proof is by induction on the structure of the derivation. Note that the only applicable rules are nonlogical 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, so the cases in which these axioms are used are straightforward. For example, for a derivation of the form

$$\frac{}{\Sigma : \Gamma, a \# a \Rightarrow \bot}\ F_3$$

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

For a derivation of the form

$$\frac{\Sigma : \Gamma, a \# b \Rightarrow \bot \quad \Sigma : \Gamma, a \approx b \Rightarrow \bot}{\Sigma : \Gamma \Rightarrow \bot}\ F_4$$

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,

$$\frac{\Sigma\#\mathsf{a} : \Gamma \Rightarrow \bot}{\Sigma : \Gamma \Rightarrow \bot}\ F$$

if $\theta : \Sigma$, then without loss of generality we can assume $\mathsf{a} \# \theta$ so that $\theta : \Sigma\#\mathsf{a}$ and so $\theta \nvDash \Gamma$ by induction.

For

$$\frac{\Sigma : \Gamma, \mathsf{a} \# t \Rightarrow \bot}{\Sigma : \Gamma \Rightarrow \bot}\ \Sigma\#$$

if $\theta : \Sigma$ then $\theta \vDash \mathsf{a} \# t$ for any $\mathsf{a} \# t \in |\Sigma|$, by Proposition 21. Consequently $\theta \nvDash \Gamma$.

For A_2,

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

suppose $\theta : \Sigma$. By induction $\theta \nvDash \Gamma, a \approx b, x \approx y$ and $\theta \nvDash, a \# y, x \approx (a\ b) \cdot y$. There are three cases. If $\theta(a) \approx \theta(b)$ and $\theta(x) \approx \theta(y)$, then $\theta \nvDash \Gamma$. Similarly, if $\theta(a) \# \theta(y)$ and $\theta(x) \approx (\theta(a)\ \theta(b)) \cdot \theta(y)$ then $\theta \nvDash \Gamma$. Otherwise, by the contrapositive of Proposition 20, $\theta \nvDash \langle a\rangle x \approx \langle b\rangle y$. In any case, $\theta \nvDash \Gamma, \langle a\rangle x \approx \langle b\rangle y$.

For A_3

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

if $\theta : \Sigma$ then $\theta(t) = \langle \mathsf{a}\rangle v$ for some $\mathsf{a} : \nu$ and $v : \tau$, so let $\theta' = \theta[a \mapsto \mathsf{a}, x \mapsto v]$. Clearly $\theta' : \Sigma, a, x$ and $\theta' \vDash t \approx \langle a\rangle x$ so by induction $\theta \nvDash \Gamma$. □

Corollary 23 (Syntactic consistency). *There is no derivation of $\Rightarrow \bot$.*

Proof. This follows from Proposition 15 and Theorem 22, since $\varnothing$ is a satisfiable constraint set. □

4.2 Separation

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

Theorem 24 (Separation). *Suppose $\Sigma : \Gamma \Rightarrow \Delta$ and Γ, Δ have no subterms of the form $\langle a \rangle t$ (or $\lambda x.t$ or $\langle t, u \rangle$). Then there is a derivation of $\Sigma : \Gamma \Rightarrow \Delta$ that does not use any nonlogical rules involving abstraction (or λ or pairing).*

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 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 $\langle a \rangle t \approx \langle b \rangle y$; 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 $\langle a \rangle x \approx \langle b \rangle y \in \Gamma'$, we have either $\vdash^{-A} \Sigma : \Gamma \Rightarrow a \approx b$ and $x \approx y$ or $\vdash^{-A} \Sigma : \Gamma \Rightarrow a \# y$ and $x \approx (a\ b) \cdot y$.

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 $\varnothing$, so the separation 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 $\Gamma, a \# x, x \approx (a\ b) \cdot y, \langle a \rangle x$. 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 Γ is redundant relative to $\Gamma', \langle a \rangle x \approx \langle b \rangle y$ 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

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

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

The proofs of separation for λ and pairing are similar, but considerably simpler because there are no branching rules for either. □

4.3 Conservativity

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 $\Sigma : \Gamma, \Gamma' \Rightarrow \Delta$ 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 $\mathsf{И}$ and appearing in freshness contexts, whereas in Pitts' system $\mathsf{И}$ quantifies ordinary variables. To bridge this gap, we translate NL formulas to $NL^{\Rightarrow}$ formulas by replacing $\mathsf{И}$-bound variables with fresh name-symbols. For example, the NL formula $\mathsf{И}a.\mathsf{И}b.p(a,b)$ translates to the $NL^{\Rightarrow}$ formula $\mathsf{И}\mathsf{a}.\mathsf{И}\mathsf{b}.p(\mathsf{a},\mathsf{b})$. We write φ^* for the translation of φ, which is defined as follows:

$$\begin{array}{rcl} A^* & = & A \\ \bot^* & = & \bot \\ (\varphi \supset \psi)^* & = & \varphi^* \supset \psi^* \\ (\forall x.\varphi)^* & = & \forall x.\varphi^* \\ (\mathsf{И}a.\varphi)^* & = & \mathsf{И}\mathsf{a}.(\varphi^*[\mathsf{a}/a]) \quad (\mathsf{a} \notin N(\varphi^*)) \end{array}$$

We write $\mathsf{a} \notin N(\varphi)$ to indicate that the name a does not appear *at all* in φ (bound or free). 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 25. *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 **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 $\forall \overline{x}. \bigwedge \overline{P} \supset \bigvee \overline{Q}$ are clearly derivable from the corresponding nonlogical rules as follows:

$$\dfrac{\dfrac{\dfrac{\overline{x} : \overline{P}, Q_1 \Rightarrow \bigvee \overline{Q} \quad \cdots \quad \overline{x} : \overline{P}, Q_n \Rightarrow \bigvee \overline{Q}}{\overline{x} : \overline{P} \Rightarrow \bigvee \overline{Q}}\, Ax}{\overline{x} \Rightarrow \bigwedge \overline{P} \supset \bigvee \overline{Q}}\, \supset R, \wedge R}{\Rightarrow \forall \overline{x}. \bigwedge \overline{P} \supset \bigvee \overline{Q}}\, \forall R$$

with the topsequents all derivable using $\vee R$ 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 $\forall\overline{x}.\exists a.a \# \overline{x}$ of CF_4, the derivation is of the form

$$\dfrac{\dfrac{\dfrac{}{\overline{x}\#\mathsf{a} : \mathsf{a} \# \overline{x} \Rightarrow \mathsf{a} \# \overline{x}}\ \exists R, \Sigma\#}{\overline{x}\#\mathsf{a} \Rightarrow \exists a.a \# \overline{x}}\ F}{\overline{x} \Rightarrow \exists a.a \# \overline{x}}\ \forall R}{\Rightarrow \forall\overline{x}.\exists a.a \# \overline{x}}$$

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

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\overline{x}\#\mathsf{a} : \mathsf{a} \# \overline{x} \Rightarrow \mathsf{a} \# \overline{x}}\ \Sigma\#^n}{\overline{x}\#\mathsf{a} \Rightarrow \mathsf{a} \# \overline{x}} \quad \dfrac{}{\overline{x}\#\mathsf{a} : \varphi(\mathsf{a},\overline{x}) \Rightarrow \varphi(\mathsf{a},\overline{x})}}{\overline{x}\#\mathsf{a} : \varphi(\mathsf{a},\overline{x}) \Rightarrow \mathsf{a} \# \overline{x} \wedge \varphi(\mathsf{a},\overline{x})}\ \wedge R}{\overline{x} : \text{И}\mathsf{a}.\varphi(a,\overline{x}) \Rightarrow \exists a.a \# \overline{x} \wedge \varphi(a,\overline{x})}\ \text{И}L, \exists R}{\Rightarrow \forall\overline{x}.(\text{И}\mathsf{a}.\varphi(a,\overline{x}) \supset \exists a.a \# \overline{x} \wedge \varphi(a,\overline{x}))}\ \forall R^n, \supset R$$

For the reverse direction, we need to show $\forall\overline{x}.\exists a.a \# \overline{x} \wedge \varphi(a,\overline{x}) \supset \text{И}\mathsf{a}.\varphi(\mathsf{a},\overline{x})$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\overline{x}, a\#\mathsf{b} : (a\ \mathsf{b}) \cdot \varphi(\mathsf{b},\overline{x}) \Rightarrow \varphi(\mathsf{b},\overline{x})}}{\overline{x}, a\#\mathsf{b} : a \# \overline{x}, \mathsf{b} \# \overline{x}, (a\ \mathsf{b}) \cdot \varphi(a,\overline{x}) \Rightarrow \varphi(\mathsf{b},\overline{x})}\ Ax^*}{\overline{x}, a\#\mathsf{b} : a \# \overline{x}, \varphi(a,\overline{x}) \Rightarrow \varphi(\mathsf{b},\overline{x})}\ \Sigma\#^*, EVL}{\overline{x}, a\#\mathsf{b} : a \# \overline{x}, \varphi(a,\overline{x}) \Rightarrow \text{И}\mathsf{a}.\varphi(\mathsf{a},\overline{x})}\ \text{И}R}{\overline{x}, a : a \# \overline{x}, \varphi(a,\overline{x}) \Rightarrow \text{И}\mathsf{a}.\varphi(\mathsf{a},\overline{x})}\ \exists L, \wedge L}{\overline{x} : \exists a.a \# \overline{x} \wedge \varphi(a,\overline{x}) \Rightarrow \text{И}\mathsf{a}.\varphi(\mathsf{a},\overline{x})}\ \forall R, \supset R}{\Rightarrow \forall\overline{x}.(\exists a.a \# \overline{x} \wedge \varphi(a,\overline{x}) \supset \text{И}\mathsf{a}.\varphi(\mathsf{a},\overline{x}))}$$

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

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 $\vdash_{NL^{\Rightarrow}} \Sigma : \Gamma \Rightarrow \Delta$. □

Now we consider the problem of conservativity: 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.

Lemma 26. *Suppose* $\Sigma\#\mathsf{a} : \Gamma[\mathsf{a}] \Rightarrow \Delta[\mathsf{a}]$. *Then* $\Sigma, a : \Gamma[a], a \# \Sigma \Rightarrow \Delta[a]$, *where* $a \# \Sigma$ *is an abbreviation for* $\{a \# \omega \mid \omega \in \Sigma\}$.

Proof. We prove the stronger induction hypothesis: "If Σ mentions only variables and $\vdash^l_n \Sigma\#\mathsf{a}; \Sigma' : \Gamma[\mathsf{a}] \Rightarrow \Delta[\mathsf{a}]$, then $\vdash^l_n \Sigma, a; \Sigma' : \Gamma[a], a \# \Sigma \Rightarrow \Delta[a]$".

The proof is by induction on the structure of the derivation of $\Sigma\#\mathsf{a}; \Sigma' : \Gamma[\mathsf{a}] \Rightarrow \Delta[\mathsf{a}]$. Almost all cases are straightforward. The only exception is the case for $\Sigma\#$. In this case we have

$$\dfrac{\Sigma\#\mathsf{a}; \Sigma', t[\mathsf{a}] \# u[\mathsf{a}] : \Gamma[\mathsf{a}] \Rightarrow \Delta[\mathsf{a}]}{\Sigma\#\mathsf{a}; \Sigma' : \Gamma[\mathsf{a}] \Rightarrow \Delta[\mathsf{a}]}\ \Sigma\#$$

for some $t[\mathsf{a}] \# u[\mathsf{a}] \in |\Sigma\#\mathsf{a}; \Sigma'|$. By induction, we have $\Sigma, a; \Sigma', t[a] \# u[a] : \Gamma[a] \Rightarrow \Delta[a]$. Note that $t[\mathsf{a}]$ must be a name-symbol. If $t[\mathsf{a}] \neq \mathsf{a}$, then $t[\mathsf{a}]$ is some name-symbol $\mathsf{b} \neq \mathsf{a}$. Since b must be in Σ or Σ', it is easy to see that $\mathsf{b} \# u[\mathsf{a}] \in |\Sigma\#\mathsf{a}; \Sigma'|$ implies $\mathsf{b} \# u[a] \in |\Sigma, a; \Sigma'|$, so the induction step is straightforward. Otherwise, the constraint is of the form $\mathsf{a} \# u[\mathsf{a}]$, where $\Sigma \vdash u[\mathsf{a}]$. Obviously, $u = u[\mathsf{a}]$ cannot depend on a. Moreover, from $a \# \Sigma$ it is possible to derive $a \# u$. Thus, using nonlogical rules only we can derive

$$\dfrac{\Sigma, a; \Sigma' : \Gamma[a], a \# \Sigma, a \# u \Rightarrow \Delta[a]}{\Sigma, a; \Sigma' : \Gamma[a], a \# \Sigma \Rightarrow \Delta[a]}\ Ax$$

from the derivation obtained using the induction hypothesis. □

With this fact in hand, we can show the desired result.

Theorem 27. *If the translated NL sequent $\Sigma : \Gamma^* \Rightarrow \Delta^*$ is derivable in $NL^{\Rightarrow}$ then $\Sigma : \Gamma \Rightarrow \Delta$ is derivable in NL.*

Proof. By the separation property, the rules involving pairing and λ-terms can be removed from a derivation of $\Sigma : \Gamma^* \Rightarrow \Delta^*$. The proof is by induction on the structure of this derivation. 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 $\forall \overline{x}. \bigwedge \overline{P} \supset \bigvee \overline{Q}$, suppose that we have derivations of the form

$$\frac{\Sigma : \Gamma, \overline{P}, Q_1 \Rightarrow \Delta \quad \Sigma : \Gamma, \overline{P}, Q_1 \Rightarrow \Delta}{\Sigma : \Gamma, \overline{P} \Rightarrow \Delta}$$

Then by induction, we have NL derivations of the NL sequents $\Sigma : \Gamma, \overline{P}, Q_1 \Rightarrow \Delta$. Now

$$\dfrac{\Sigma : \Gamma \Rightarrow \forall \overline{x}. \bigwedge \overline{P} \supset \bigvee \overline{Q} \qquad \dfrac{\dfrac{\Sigma : \overline{P} \Rightarrow \bigwedge \overline{P} \qquad \dfrac{\Sigma : \overline{B}, Q_1 \Rightarrow \Delta \quad \cdots \quad \Sigma : \overline{B}, Q_n \Rightarrow \Delta}{\Sigma : \overline{P}, \bigvee \overline{Q} \Rightarrow \Delta}}{\Sigma : \overline{P}, \bigwedge \overline{P} \supset \bigvee \overline{Q} \Rightarrow \Delta}}{\Sigma : \forall \overline{x}. \bigwedge \overline{P} \supset \bigvee \overline{Q} \Rightarrow \Delta}}{\Sigma : \overline{P} \Rightarrow \Delta}$$

The cases for $F_3, F_4, F, A_2, A_3, \Sigma\#, И L, И R$ remain.

For F_3, we have a derivation

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

In NL we can derive $\Sigma : \Gamma, a \# a \Rightarrow \Delta$ 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

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

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

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

The upper sequent is not a NL sequent because $\Sigma\#\mathsf{a}$ mentions a name-symbol, but by Lemma 26, it is equivalent to the NL sequent $\Sigma, a : \Gamma, a \# \overline{x} \Rightarrow \Delta$, where $\{\overline{x}\} = FV(\Sigma)$. By induction this has a derivation in NL. We can derive

$$\dfrac{\Sigma : \cdot \Rightarrow \forall \overline{x}. \exists a. a \# \overline{x} \quad \dfrac{\Sigma, a : \Gamma, a \# \overline{x} \Rightarrow \Delta}{\Sigma : \Gamma, \forall \overline{x}. \exists a. a \# \overline{x} \Rightarrow \Delta} \; \forall R, \exists R}{\Sigma : \Gamma \Rightarrow \Delta} \; cut$$

It is easy to derive rules A_2, A_3 from axioms $(CA_1), (CA_2)$ of NL. The rule $\Sigma\#$ cannot apply because if no name-symbols appear in Σ, then $|\Sigma| = \varnothing$.

Finally, we consider the cases for $И L$ and $И R$. For $И L$, we have

$$\frac{\Sigma\#\mathsf{a} : \Gamma, \varphi(\mathsf{a}, \overline{x}) \Rightarrow \Delta}{\Sigma : \Gamma, И\mathsf{a}.\varphi(\mathsf{a}, \overline{x}) \Rightarrow \Delta} \; И L$$

From the upper derivation we can (by Lemma 26) obtain a derivation of the sequent $\Sigma, a : \Gamma, a \# \Sigma, \varphi(a, \overline{x}) \Rightarrow \Delta$ of no greater complexity. By induction we have a NL derivation of the same sequent. Now we can also

Swapping

(IS_1) $\quad \forall a{:}\nu, x{:}\tau.\ (a\ a) \cdot x \approx x$

(IS_2) $\quad \forall a, a'{:}\nu, x{:}\tau.\ (a\ a') \cdot (a\ a') \cdot x \approx x$

(IS_3) $\quad \forall a, a'{:}\nu.\ (a\ a') \cdot a \approx a'$

Equivariance

(IE_1) $\quad \forall a, a'{:}\nu, b, b'{:}\nu', x{:}\tau.\ (a\ a') \cdot (b\ b') \cdot x \approx ((a\ a') \cdot b\ (a\ a') \cdot b') \cdot (a\ a') \cdot x$

(IE_2) $\quad \forall a, a'{:}\nu, b{:}\nu', x{:}\tau.\ b \# x \supset (a\ a') \cdot b \# (a\ a') \cdot x$

(IE_3) $\quad \forall a, a'{:}\nu, \overline{x} : \overline{S}.\ (a\ a') \cdot f(\overline{x}) \approx f((a\ a') \cdot \overline{x})$

(IE_4) $\quad \forall a, a'{:}\nu, \overline{x} : \overline{S}.\ p(x) \supset p((a\ a') \cdot \overline{x})$

(IE_5) $\quad \forall b, b'{:}\nu', a{:}\nu, x{:}\tau.\ (b\ b') \cdot (\langle a \rangle x) \approx \langle (b\ b') \cdot a \rangle ((b\ b') \cdot x)$

Freshness

(IF_1) $\quad \forall a, a'{:}\nu, x{:}\tau.\ a \# x \wedge a' \# x \supset (a\ a') \cdot x \approx x$

(IF_2) $\quad \forall a{:}\nu.\ \neg(a \# a)$

(IF_3) $\quad \forall a, a'{:}\nu.\ a \# a' \vee a \approx a'$

(IF_4) $\quad \forall a{:}\nu, a'{:}\nu'.\ a \# a'$

(IF_5) $\quad \forall \overline{x} : \overline{S}.\ \exists a{:}\nu.\ a \# \overline{x}$

И-quantifier

(IQ) $\quad \forall \overline{x}.(И a{:}\nu.\ \varphi) \iff (\exists a{:}\nu.\ a \# \overline{x} \wedge \varphi)$

where $FV(И a.\varphi) \subseteq \{\overline{x}\}$

Abstraction

(IA_1) $\quad \forall a, a'{:}\nu, x, x'{:}\tau.\ \langle a \rangle x \approx \langle a' \rangle x' \iff (a \approx a' \wedge x \approx x') \vee (a' \# x \wedge x' \approx (a\ a') \cdot x)$

(IA_2) $\quad \forall y : \langle \nu \rangle S. \exists a{:}\nu, x{:}\tau.\ y \approx \langle a \rangle x$

Figure 10: Axioms of Intuitionistic Nominal Logic

derive $\Sigma : \Gamma, \forall \overline{x}. \exists a. a \# \overline{x} \wedge \varphi(a, \overline{x}) \Rightarrow \Delta$ using $\exists L$ and $\forall L$. Finally, we can cut against the axiom instance $\forall \overline{x}. \exists a. a \# \overline{x} \wedge \varphi(a, \overline{x}) \iff И a. \varphi(a, \overline{x})$ to prove that $\Sigma : \Gamma, И \mathsf{a}. \varphi(\mathsf{a}, \overline{x}) \Rightarrow \Delta$.

For $И R$, we have

$$\frac{\Sigma \# \mathsf{a} : \Gamma \Rightarrow \varphi(\mathsf{a}, \overline{x}), \Delta}{\Sigma : \Gamma \Rightarrow И \mathsf{a}. \varphi(\mathsf{a}, \overline{x}), \Delta}\ И R$$

Again by Lemma 26 and the induction hypothesis, we can derive $\Sigma, a : \Gamma, a \# \Sigma \Rightarrow \varphi(a, \overline{x}), \Delta$ in NL. In NL it is not difficult to prove that

$$\vdash_{NL} \forall \overline{y}. \forall a. a \# \overline{y} \supset \psi \supset И \mathsf{a}. \psi$$

whenever $FV(И a.\varphi) \subseteq \overline{y}$, so taking $\overline{y} = FV(\Sigma) \supseteq FV(И a.\varphi)$ and $\psi = \varphi(a, \overline{y})$, we can derive $\vdash_{NL} \Sigma : \forall a. a \# \Sigma \supset \varphi(a, \overline{x}) \Rightarrow И \mathsf{a}. \varphi(\mathsf{a}, \overline{x})$. Using cut, we can obtain $\Sigma : \Gamma \Rightarrow \Delta, И \mathsf{a}. \varphi(\mathsf{a}, \overline{x})$. □

4.3.2 Intuitionistic Case

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. 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 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 27, 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 from Pitts' axioms. Moreover, in INL, equality and freshness among names are both decidable.

For this reason, we introduce an alternative axiomatization INL, shown in Figure 10, differing in the replacement of (CF_2) with two axioms (IF_2) and (IF_3). This decision may be defended on the following grounds:

1. Though stronger in intuitionistic logic, INL is equivalent to NL in classical logic.
2. Every intuitionistic theorem of NL is an intuitionistic theorem of INL, and the inclusion is strict.
3. Every intuitionistic theorem of INL is a classical theorem of NL, and the inclusion is strict.
4. From a constructivist point of view, it is usually acceptable for equality to be decidable at base types (such as name-types).
5. Fresh Logic, another intuitionistic form of nominal logic, also included rules similar to F_3 and F_4.

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 28. $\vdash_{INL} \Sigma : \Gamma \Rightarrow \Delta$ *is derivable if and only if* $\vdash_{INL^{\Rightarrow}} \Sigma : \Gamma^* \Rightarrow \Delta^*$.

5 Conclusions

Nominal logic is a recently developed logic that is of considerable interest because it provides powerful techniques for reasoning about fresh names and name-binding. One of the most interesting features of nominal logic is the N-quantifier. However, the techniques used for reasoning with N offered by previous formalizations of nominal logic are highly (but unnecessarily) complex.

In this paper we have introduced a new sequent calculus for nominal logic which uses *freshness contexts* to deal with the new-quantifier. Its rules for N are symmetric and rationalize a proof-search semantics for N that seems natural and intuitive (inspired by the treatment of 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.

Two applications of $NL^{\Rightarrow}$ are to be addressed by future work:

- $NL^{\Rightarrow}$ provides a proof-search reading of N which is much closer to the approach taken in the αProlog nominal logic programming language [1]. While Gabbay and Cheney gave a proof-theoretic semantics of nominal logic programming based on FL_{Seq}, this analysis does not seem relevant to αProlog because it suggests a radically different (and much more computationally intensive) proof-search technique for N-quantified formulas. In contrast, $NL^{\Rightarrow}$ seems to provide a rationale for αProlog's existing search technique. We plan to study the proof-theoretic semantics of nominal logic programming in $NL^{\Rightarrow}$ in order to establish the correctness of (or fix any bugs in) αProlog's search technique.
- Gabbay and Cheney showed that $FO\lambda^{\nabla}$, another logic possessing a self-dual "fresh value" quantifier, can be soundly interpreted in nominal logic via a proof-theoretic translation. However, the translation they developed was incomplete, and the possibility of finding a faithful translation was left open. Using $NL^{\Rightarrow}$, it appears to be possible (though still not easy) to prove the completeness of this translation [2].

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 equality and freshness theory that necessitates the many nonlogical rules in $NL^{\Rightarrow}$ could be dealt with implicitly via rewriting and normalization, leading to an even simpler proof theory for nominal logic. However, work in this direction by Schöpp and Stark [9] indicates that there may be significant obstacles to this approach; the system introduced in this paper may be viewed as a well-behaved fragment of their system. Further improvement to the proof theory of nominal logic seems possible and desirable.

References

[1] J. Cheney and C. Urban. Alpha-Prolog: A logic programming language with names, binding and alpha-equivalence. In *Proc. 20th Int. Conf. on Logic Programming (ICLP 2004)*, number 3132 in LNCS, pages 269–283, 2004.

[2] James Cheney. A simpler proof theory for nominal logic. In *Proceedings of the 2005 Conference on Foundations of Software Science and Computation Structures (FOSSACS 2005)*, 2005. To appear.

[3] James R. Cheney. *Nominal Logic Programming*. PhD thesis, Cornell University, Ithaca, NY, August 2004.

[4] M. J. Gabbay. Fresh logic: A logic of FM, 2003. Submitted.

[5] M. J. Gabbay and J. Cheney. A proof theory for nominal logic. In *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science (LICS 2004)*, pages 139–148, Turku, Finland, 2004.

[6] M. J. Gabbay and A. M. Pitts. A new approach to abstract syntax with variable binding. *Formal Aspects of Computing*, 13:341–363, 2002.

[7] Sara Negri and Jan von Plato. *Structural Proof Theory*. Cambridge University Press, 2001.

[8] A. M. Pitts. Nominal logic, a first order theory of names and binding. *Information and Computation*, 183:165–193, 2003.

[9] Ulrich Schöpp and Ian Stark. A dependent type theory with names and binding. In *Proceedings of the 2004 Computer Science Logic Conference*, number 3210 in Lecture notes in Computer Science, pages 235–249, Karpacz, Poland, 2004.

[10] A. S. Troelstra and H. Schwichtenberg. *Basic Proof Theory*. Number 43 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, second edition, 2000.

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