Nominal Lambda Calculus

Thesis submitted for the degree of

Doctor of Philosophy

at the University of Leicester

by

Frank Nebel MSc (Amsterdam)

Department of Computer Science

University of Leicester

December 2014

Abstract

Nominal Lambda Calculus

Frank Nebel

Since their introduction, nominal techniques have been widely applied in computer science to reason about syntax of formal systems involving name-binding operators. The work in this thesis is in the area of "nominal" type theory, or more precisely the study of "nominal" simple types.

We take Nominal Equational Logic (NEL), which augments equational logic with freshness judgements, as our starting point to introduce the Nominal Lambda Calculus (NLC), a typed lambda calculus that provides a simple form of name-dependent function types. This is a key feature of NLC, which allows us to encode freshness in a novel way.

We establish meta-theoretic properties of NLC and introduce a sound model-theoretic semantics. Further, we introduce NLC[$\mathbb{A}$], an extension of NLC that captures name abstraction and concretion, and provide pure NLC[$\mathbb{A}$] with a strongly normalising and confluent $\beta\eta$-reduction system.

A property that has not yet been studied for "nominal" typed lambda calculi is completeness of $\beta\eta$-conversion for a nominal analogue of full set-theoretic hierarchies. Aiming towards such a result, we analyse known proof techniques and identify various issues. As an interesting precursor, we introduce full nominal hierarchies and demonstrate that completeness holds for $\beta\eta$-conversion of the *ordinary* typed lambda calculus.

The notion of FM-categories was developed by Ranald Clouston to demonstrate that FM-categories correspond precisely to NEL-theories. We augment FM-categories with equivariant exponentials and show that they soundly model NLC-theories. We then outline why NLC is not complete for such categories, and discuss in detail an approach towards extending NLC which yields a promising framework from which we aim to develop a future (sound and complete) categorical semantics and a categorical type theory correspondence.

Moreover, in pursuit of a categorical conservative extension result, we study (enriched/internal) Yoneda isomorphisms for "nominal" categories and some form of "nominal" gluing.

Acknowledgments

First and foremost, I would like to express my sincere gratitude to my supervisor Dr. Roy L. Crole for his continuous support, invaluable guidance and constant encouragement during my PhD studies.

I gratefully acknowledge the members of my PhD committee, Professor Maribel Fernández and Dr. Fer-Jan de Vries, for their time and valuable feedback.

I thank all the members of the Computer Science Department for providing such a vibrant research environment and an almost familiar atmosphere. A special thanks goes to Dr. Alexander Kurz, Dr. Daniela Petrişan and Dr. Paula G. Severi for many interesting and fruitful discussions.

Lastly, I would like to thank my family for always being there for me, most importantly my parents, Jutta and Ludwig, to whom I am eternally grateful for their unconditional love, support and encouragement. I also wish to thank Daniel for being such a good friend. The last words of acknowledgement I have reserved for Carmen, for her unwavering love, incredible patience and understanding, as well as for her infinite support over all these years.

Contents

List of Tables

Chapter 1

Introduction

In semantics of programming languages the notion of abstract syntax is a basic tool that is employed in meta reasoning. It is used to abstract away from the unnecessary syntactic details of concrete syntax and to concentrate on the essential constructs of a language. The key idea is that abstract syntax trees are introduced by structural induction, which allows one to define operations by structural recursion on abstract syntax trees and therefore to prove properties of such operations by structural induction. It is this structural approach that considerably simplifies meta reasoning as can be observed in the context of algebraic systems.

For languages with name-binding operators however it has to be considered that one does not formally deal with abstract syntax trees anymore, but with α-equivalence classes of abstract syntax trees, which are not inherently inductively defined. In informal reasoning this is merely a technical issue, which is handled by working with representatives of the α-equivalence classes and renaming of bound variables if a clash of names occurs. This informal approach, with the necessary care, is sufficient in the context of informal reasoning, but if one is interested in formal (mechanised) proofs a formal treatment of abstract syntax for languages with name-binding is required.

The nominal set model[1], introduced by Gabbay and Pitts [32, 33], provides a math-

[1] originally introduced as the FM-set model.

ematical foundation to compute and reason about syntax of formal systems involving name-binding operators. While the original application of the nominal set model was to formally reason about abstract syntax with variables and binders, it is now widely applied to different areas of computer science and logic, where name-binding operators are used. The underlying ideas of the nominal set model are often referred to as nominal techniques. Examples of where nominal techniques have been applied are nominal logics (equational logic [18, 36] and first-order logic [53, 31]), game semantics [1, 69], domain theory [62, 68], rewriting systems [27], theorem provers [70], logical programming [11] and functional programming [61]. One of the key features of nominal techniques is that the corresponding formalities are close to informal reasoning ("pen and paper mathematics").

The work conducted in this thesis is in the area of "nominal" type theory, which in recent years has become an active area of research, with contributions ranging over simple types [9, 55, 23], dependent types [10, 26, 60] and polymorphic types [25].

Our focus is on "nominal" simple types and the introduction of a "nominal" typed lambda calculus. The typed lambda calculus ($\lambda^{\rightarrow}$) and its equational proof system is a fundamental formal framework in mathematical logic and computer science to investigate logical consequences of a set of axioms. It was introduced by Church [12] and is a direct extension of algebraic (first-order) equational reasoning. We recall that in the context of algebraic equational reasoning two nominal approaches have already been introduced: Nominal Equational Logic (NEL) by Clouston and Pitts [18] and Nominal Algebra (NA) by Gabbay and Mathijssen [36]. In this thesis, we have borrowed key ideas and structures from NEL [13] to introduce a "nominal" (higher-order) functional type theory, referred to as the Nominal Lambda Calculus (NLC), and to study its syntax and semantics. Note that the initial driving factor for choosing NEL as our starting point was our goal to extend the categorical type theory correspondence for NEL [15]. To provide an overview we give a more detailed description of our contributions and goals:

- We introduce NLC and provide a detailed account of the meta-theory of raw terms (Section 3.1), as well as the properties of the type and equation system (Section 3.2).

- We introduce a model-theoretic and categorical semantics for NLC and demonstrate that both semantics are sound (Section 3.4 and Section 5.2).

- We prove that NLC is a conservative extension of NEL and $\lambda^{\to}$ (Section 3.5).

- We capture the notion of name abstraction and concretion in NLC. The extended calculus is referred to as NLC[$\mathbb{A}$] (Section 4.1). Moreover, we introduce $\mathcal{U}$NLC, which extends NLC[$\mathbb{A}$] with local fresh atomic names (Section 4.3). For both calculi, we extend the model-theoretic semantics and prove soundness.

- It is well known that functional theories over $\lambda^{\to}$ correspond to cartesian closed categories [20, 42]. In the light of the categorical type theory correspondence for NEL, which uses FM-categories, a natural question to ask is whether NLC corresponds to some form of "cartesian closed FM-category". This will be discussed in Section 5.3 where we make some important observations and introduce a promising framework towards a future (sound and complete) categorical semantics and a categorical type theory correspondence for an extension of NLC.

Moreover, we analyse properties of pure NLC (without constants) for the pure $\beta\eta$-theory (without axioms). Work in this direction, but with different foundations, has been carried out by Cheney [9] and Pitts [55].

- We introduce a pure $\beta\eta$-reduction system for NLC[$\mathbb{A}$] and prove it to be strongly normalising and confluent. Further, using these results, we demonstrate that provable equality is decidable (Section 4.2).

- We introduce the notion of full nominal hierarchies, a nominal analogue of full set-theoretic hierarchies, and prove $\beta\eta$-conversion to be complete for full nomi-

nal hierarchies in $\lambda^{\rightarrow}$. This result can be seen as a precursor for completeness theorems of "nominal" typed lambda calculi.

In connection to our work towards a categorical type theory correspondence result for NLC, we also aimed to prove an additional conservative extension property (based on ground types) for NLC using a categorical proof argument that we sketched in Section B.1. This proof argument motivated a careful investigation of enriched and internal Yoneda isomorphisms for "nominal" categories (Chapter 7), as well as an extension of the well known gluing lemma for cartesian closed categories (Section 5.1.3) such that the properties of an FM-category are also inherited by the glued category (Freyd scone). This work lead us to various interesting observations and technical results, which we thought worthwhile to be included in this thesis.

Note that the content of Section 3.1 and 3.2, as well as Chapter 5 has been published in [23], and was written with Roy Crole.

1.1 Thesis Overview

We first provide an overview of the content of each individual chapter and later clarify how the chapters relate to one another from a motivational perspective.

In Chapter 2 we recall the concept of abstract syntax with name-binding operators and its formalisation using the three main approaches. We give a detailed account of the nominal set model with its key notions of finite support and freshness, as well as name abstraction, concretion and local fresh atomic names. In addition, we sketch the FM-set model, which uses FM-set theory as its foundation and is the precursor of the nominal set model. Based on these mathematical models, we introduce three categories, *Nom*, *FMSet* and *FMNom*, also referred to as nominal/FM categories, and prove various categorical properties of these categories.

In Chapter 3 we introduce NLC. We establish several meta-theoretic properties of its raw terms and study its mutual inductively defined type and equation system.

We then introduce a model-theoretic semantics (in *FMSet*) and prove the type and equation system to be sound. Further, using a semantic model construction, we prove that NLC is a conservative extension of NEL and $\lambda^{\to}$.

In Chapter 4 we introduce NLC[$\mathbb{A}$], an extension of NLC, and demonstrate that it captures name abstraction and concretion. We extend the model-theoretic semantics of NLC and prove it to be sound. Further, we define a pure $\beta\eta$-reduction system for NLC[$\mathbb{A}$] and prove it to be strongly normalising and confluent. Using these results, we deduce that provable equality is decidable, and moreover that type checking and type inference are decidable as well. In addition, we demonstrate that NLC can be used to capture local fresh atomic names. The respective extension of NLC[$\mathbb{A}$] is referred to as ИNLC. We conclude by demonstrating on various concrete examples how ИNLC can be used to capture various constructions in the FM-set model that involve name abstraction, concretion and local fresh atomic names.

In Chapter 5 we recall the notion of an FM-category and extend it with equivariant exponentials. The new construction is called an FM-cartesian closed category (FM-ccc). We investigate if certain categorical constructions, which are required in a categorical proof argument (see B.1), lift various properties of an FM-ccc. Further, based on the notion of an FM-ccc we define a sound categorical semantics for NLC. We then provide evidence that suggests that NLC is not expressive enough to construct a syntactically generated classifying FM-ccc (term quotient category), or more precisely, the construction of an exponential in such a category is not possible for NLC. As a consequence, a categorical type theory correspondence for NLC, as well as a "least-model" completeness result cannot be obtained in the usual way. As a step towards resolving this issue, we outline two approaches, which aim to extend NLC such that a syntactically generated classifying category can be constructed (with a generic model). The approach we have ultimately chosen to pursue is semantically motivated by properties observed in the category of FM-sets and FM-functions (*FMSet*). Based on these observations, we enhance NLC which ultimately allows us to construct an

equivariant exponential in a syntactically generated category. This constitutes an important step towards a syntactically generated classifying category, and ultimately a complete categorical semantics and a categorical type theory correspondence result.

In Chapter 6 we introduce the notion of a full nominal hierarchy, which is a nominal analogue of full set-theoretic hierarchies, and demonstrate that it is an ordinary environment model (Henkin model). We pursue the question if the pure $\beta\eta$-theory can be proven complete for full nominal hierarchies in pure $\lambda^{\to}$. We argue that this intermediate goal is not only interesting in its own right, but it also serves as a precursor towards completeness theorems of "nominal" lambda calculi in the future. We then provide an overview of the standard proof techniques used to prove completeness of the pure $\beta\eta$-theory for full set-theoretic hierarchies in pure $\lambda^{\to}$ and observe that the logical relation based proof argument is problematic in the context of full nominal models. This is due to the fact that the argument relies on the axiom of choice. To circumvent the related issues we apply a stronger precondition, which leads to a rather restricted completeness theorem. To obtain a stronger completeness theorem we pursued an alternative route towards completeness using Statman's 1-section theorem, which provides us with a necessary and sufficient condition to determine if the pure $\beta\eta$-theory is complete for a particular Henkin model in $\lambda^{\to}$. We demonstrate that this condition is satisfied for full nominal hierarchies and due to the fact that full nominal hierarchies are Henkin models, the 1-section theorem can directly be applied to prove completeness for full nominal hierarchies.

In Chapter 7 we determine if the nominal/FM-categories that we introduced in Chapter 2 have certain categorical properties, like cartesian closure or (co)completeness, and recall that these results have direct consequences if one wishes to obtain an enriched or internal Yoneda isomorphism. We discuss these consequences in more detail and unravel variants of enriched Yoneda isomorphism. Moreover, we provide a barehands version of such an enriched Yoneda isomorphism, which reveals interesting technical details. Further, we prove an internal Yoenda isomorphism for *FMSet* and

introduce the notion of a finitely supported (co)limit for *FMSet*, for which *FMSet* is (co)complete.

In Chapter 8 we provide a summary of our contributions, comment on related work and propose future lines of research.

As previously indicated, we will now relate the different content chapters from a motivational perspective. For this reason we provide two mind maps (see Figure 1.1 and Figure 1.2) and the following explanatory text:

At first, we want to emphasise that our initial goal, namely to extend the categorical type theory correspondence for NEL, had a major impact on the research carried out in this thesis, because it significantly influenced the definition of NLC, which is the core of this thesis (Chapter 3). Taking NLC as our starting point, we pursued the following goals:

- Based on the observation that NLC can be used to capture name abstraction and concretion, we introduced NLC[$\mathbb{A}$] as an extension of NLC. We studied its properties, extended its denotational (model-theoretic) semantics and furthermore introduced an operational semantics (Section 4.1 and 4.2).

- Due to the difficulties that we encountered, regarding the construction of a syntactically generated classifying category for NLC, we considered various possibilities of how to extend NLC. We ultimately decided to pursue a semantically motivated approach. This approach required various additional structures in NLC: There is name abstraction and concretion, something that we had already considered in a different context (Section 4.1). In addition, we had to introduce local fresh atomic names, which lead to the introduction of $И$NLC (Section 4.3). However, to ultimately construct equivariant exponentials, we introduced [$И$]NLC (Chapter 5), a semantically motivated variant of $И$NLC, that mimics an adjoint equivalence property of *FMSet*.

- The work carried out in Chapter 7 and Section 5.1.3 was directly motivated by

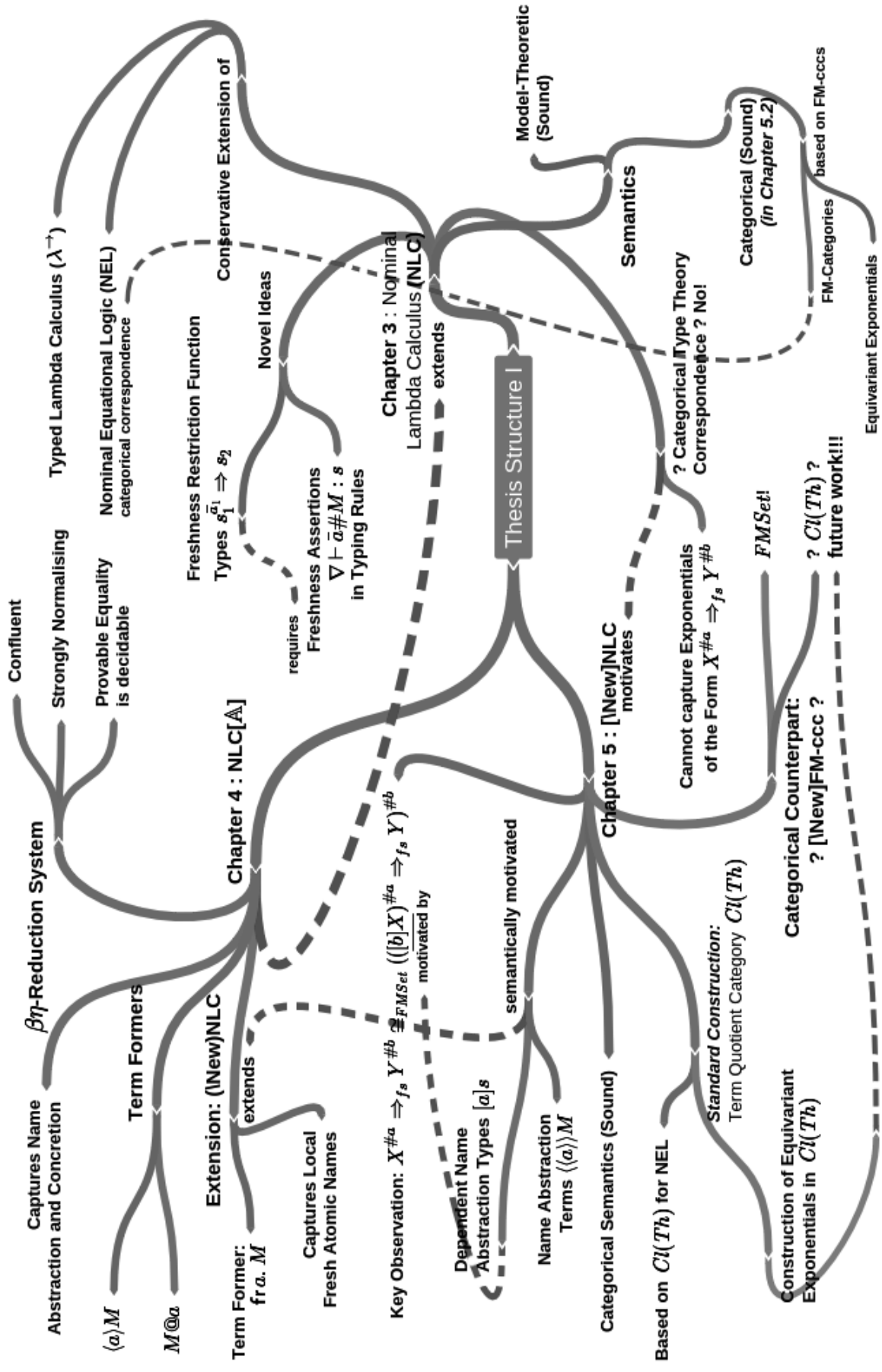


Figure 1.1: Thesis Structure I

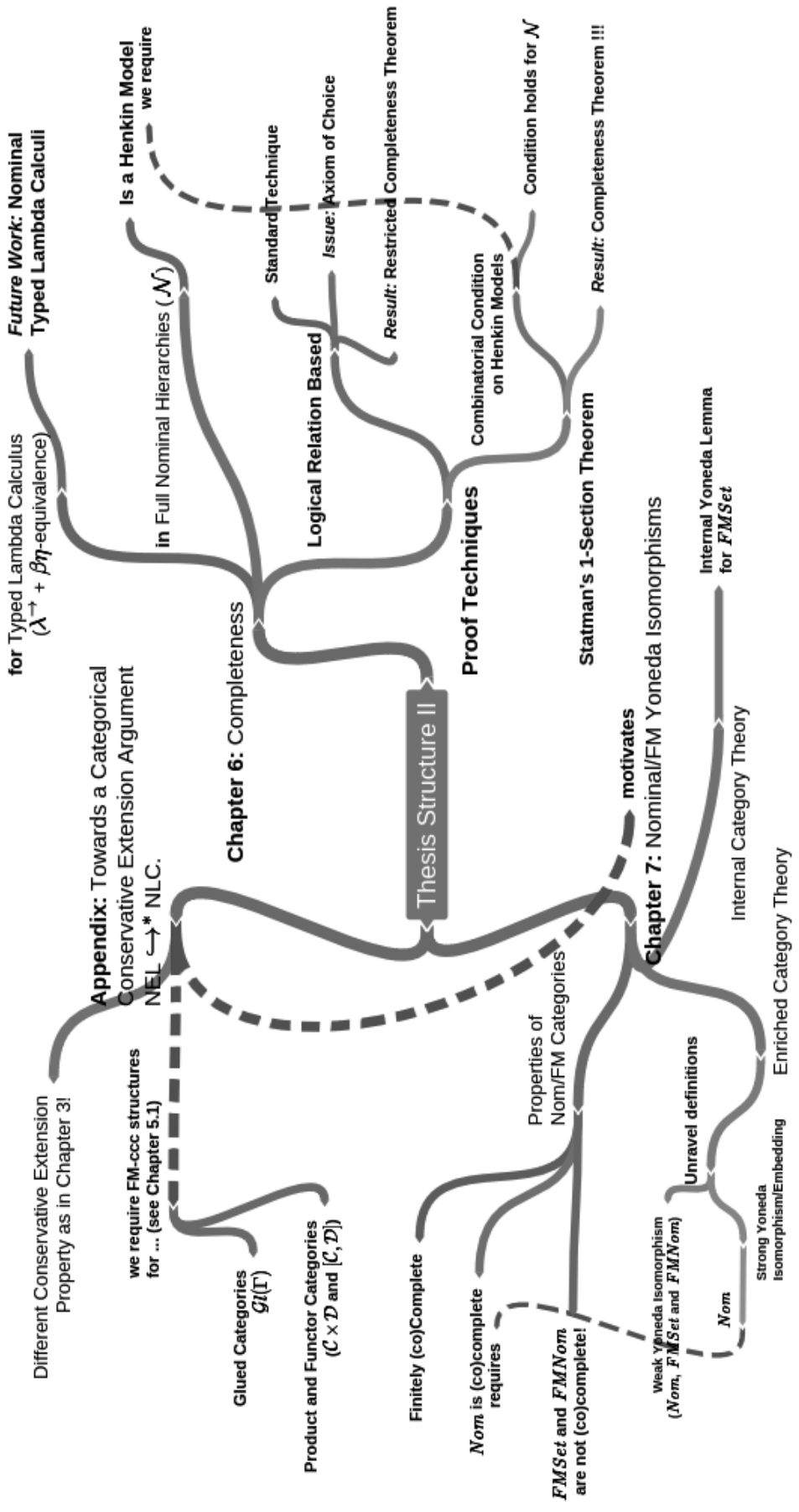


Figure 1.2: Thesis Structure II

a categorical proof argument that we intended to utilise to prove an additional conservative extension property for NEL and NLC (on base types).

- Another interesting line of work that we pursued was to study completeness for "nominal" typed lambda calculi. We initially focused on NLC, but later turned towards a more restricted problem domain, which allowed us to obtain an intermediate result that we believe may serve as a precursor towards completeness theorems for "nominal" typed lambda calculi (Chapter 6).

Chapter 2

Nominal sets and FM-sets

In this chapter we provide a comprehensive introduction in nominal techniques and their underlying mathematical model.

We begin by recalling the concept of abstract syntax with name-binding operators and how it can be formalised. This includes a reminder of the three main approaches and their typical characteristics. We then turn towards the mathematical model that underpins nominal techniques. It is the FM-set model, introduced by Gabbay and Pitts [32, 33], which was later refined by Pitts [53] to the nominal set model. A certain advantage of the nominal set model, regarding applicability, is that it does not rely on non-standard set-theory (FM-set theory), which makes it slightly more intuitive. We provide a detailed account of the nominal set model, which includes the basic notions of finite support support and freshness, as well as three key concepts of nominal techniques, namely name abstraction, concretion and local fresh atomic names. Up to this point, the corresponding definitions and properties are, apart from minor deviations, almost identical for the FM-set model. So, we decided to only provide a quick overview of the FM-set model. This, of course, does not mean that all their properties are identical, but for the tasks we have in mind, it will suffice to point out differences in due course. We conclude this chapter by introducing nominal/FM categories and by proving various categorical properties for these categories.

2.1 Abstract Syntax with Name-Binding

We remind the reader that an atomic name, often just referred to as a name or an atom, is an identifier without an intrinsic meaning, which has one basic property, namely that it can be compared for identity with other such atomic names. The association of an object with an atomic name is referred to as atomic name-binding.

Two prototypical examples, in logic and computer science, for formal systems with name binders (variable binders) in their object language are first-order logic and the simply typed lambda calculus. Note that in the context of name binders, we often require the ability to choose a sufficiently fresh atomic name (variable). This is for example necessary to define a notion of capture-avoiding substitution (in both systems) or to express the following logical identity and equational rule:

- First Order Logic: $P \wedge \exists x(Q(x)) \Longleftrightarrow \exists x(P \wedge Q(x))$ for $x \notin fv(P)$
- Typed Lambda Calculus: $\Gamma \vdash \lambda x : \sigma.(M\ x) = M : \sigma \Rightarrow \tau$ for $x \notin fv(M)$

A fundamental property arising in the context of name-binding is α-equivalence. Informally, two terms in a language are α-equivalent if they differ only in the choice of bound names (variables). Note that it is not unreasonable to expect that any semantics for a language assigns an equal meaning to α-equivalent terms of a language. As a consequence, many operations on syntax need to respect α-equivalence and therefore it is meaningful to define operations on α-equivalence classes of abstract syntax trees in the first place when name binders are involved.

In informal practice α-equivalence is dealt with informally by using ordinary abstract syntax trees (with explicitly chosen bound names). This means that there is no formal distinction between an α-equivalence class and a chosen representatives of this α-equivalence class. In the case that a particular representative of an α-equivalence class is involved in a clash of bound names, another representative with a fresh name can simply be chosen to avoid such a clash. In the literature this is usually indicated

by the phrase "we identify terms up to α-equivalence", also referred to as the *Barendregt Variable Convention*. There are various approaches towards formalising names and binders. We will now give an overview of the three main approaches and their key characteristics:

An early and rather technical approach is based on de Bruijn indices [7], which propagate a "nameless" presentation of languages with name-binding. The key idea behind these indices is that the specific name of a bound name does not matter, but only the fact that a name is bound by a particular binder. So, for example, the S combinator $\lambda x \lambda y \lambda z.(xz)(yz)$ is represented by $\lambda\lambda\lambda(31)(21)$. An immediate consequence of this approach is that α-equivalence is reduced to syntactic equality. Moreover, operations on syntax are rather arithmetical in nature, which makes this approach very well suited for mechanised proofs, but rather challenging for humans to read and manipulate ("coding gap").

Another approach to represent abstract syntax trees of languages with name-binding is Higher Order Abstract Syntax (HOAS) [51]. The core idea is to use a typed lambda calculus as a meta-language in which an object language is embedded. Thus, questions of binding and α-equivalence are dealt with at a meta-level rather than as a first class aspect of the object language.

A more recent approach, which was introduced by Gabbay and Pitts [32], is the theory of nominal sets. Contrary to de Bruijn indices or HOAS, where names are either avoided or handled as second class citizens, nominal techniques grant names a first-class status. Its key characteristic is that it fully formalises the *informal practice* of name-binding, which we have previously sketched, by using notions of support, freshness and the novel concept of name abstractions.

To work with names as first class citizen, the concept of a permutation model has been chosen as the mathematical foundation. It allows to implicitly expresses the dependency of names (support) and independency of names (freshness). Such permutation models and their corresponding notion of support already existed in the

literature, but have been "rediscovered" in the context of names and binding. It was originally introduced by Fraenkel [29] and refined by Mostowski [49] to prove the independence of the axiom of choice from the axioms of ZFA (ZF with atoms).

A key novelty of the nominal set model is the introduction of name abstractions, which allow one to *explicitly* express the dependency of a structure on names. Thus, name-binding can now be directly captured in the nominal set model. In combination with cartesian products and disjoint union, name abstraction can also be used to represent abstract syntax modulo α-equivalence as an inductive date type in the nominal set model with the corresponding notions of α-structural induction and α-structural recursion. An instructive example for the untyped lambda calculus can be found in [33].

For discussions regarding the advantages and disadvantages of the nominal set model, compared to the earlier approaches, we refer to [4, 8, 19]. In general, in the wider research community, the nominal set model is by now well accepted and widely applied as an alternative approach to formalise names and binding.

2.2 Nominal Set Model

We now recall the basic definitions and properties of the nominal set model, as well as various related constructions that will be used in this thesis. For full details we refer to [33, 56].

Support and Nominal Sets

Let $\mathbb{A}$ be the countable infinite set of names and $Perm(\mathbb{A})$ the set of finite permutations of $\mathbb{A}$, which are bijections $\pi : \mathbb{A} \to \mathbb{A}$ such that $\{a \in \mathbb{A} \mid \pi(a) \neq a\}$ is finite. $Perm(\mathbb{A})$ has a group structure with function composition, denoted by $\circ$, acting as group multiplication, the identity permutation, denoted by ι, as group identity and the inverse function, denoted by π^{-1}, as group inverse. We write $\overline{a}$ for finite subsets

of $\mathbb{A}$ and $(a\,a')$ for a **transposition**, also referred to as **swapping** between names.

The notion of a $Perm(\mathbb{A})$**-set** is defined to be a set X, which is equipped with a **permutation action** ($Perm(\mathbb{A}) \times X \to X$), denoted as $(\pi, x) \mapsto \pi \cdot_X x$, such that the following two properties hold:

$$\iota \cdot x = x \qquad\qquad \pi \cdot (\pi' \cdot x) = (\pi \circ \pi') \cdot x$$

If it is clear from context we usually just write $\pi \cdot x$. A function $f : X \to Y$ between $Perm(\mathbb{A})$-sets is called **equivariant** if for all $\pi \in Perm(\mathbb{A})$ we have that

$$\pi \cdot f(x) = f(\pi \cdot x)$$

A subset $S \subseteq X$ of a $Perm(\mathbb{A})$-set X is an **equivariant subset** if is closed under the permutation action of X, i.e. for all $\pi \in Perm(\mathbb{A})$ and $x \in X$ we have that if $x \in S$ then $\pi \cdot x \in S$. For a $Perm(\mathbb{A})$-Set X, we say that $S \subseteq \mathbb{A}$ **supports** $x \in X$ if for all $a', a'' \in \mathbb{A} \setminus S$ we have that $(a'\,a'') \cdot x = x$. If S is a finite subset, we say that x is **finitely supported** by S. A $Perm(\mathbb{A})$-Set X with the finite support property, which requires that every $x \in X$ is finitely supported, is called a **nominal set**. We now recall some basic properties of nominal sets. For proof details we refer to [33].

Lemma 2.2.1 *Let X be a nominal set and $x \in X$. Then there exists a least* finite *set, written as $supp(x)$, which supports x. Moreover, $supp(x)$ can be explicitly expressed as follows:*

$$a \in supp(x) \iff \{a' \in \mathbb{A} \mid (a\,a') \cdot x \neq x\} \text{ is not finite.}$$

Lemma 2.2.2 *Let X be a nominal set and $x \in X$. Then $\pi \cdot supp(x) = supp(\pi \cdot x)$.*

Lemma 2.2.3 *Let $f : X \to Y$ be an equivariant function between $Perm(\mathbb{A})$-sets X and Y. Then the following properties hold:*

(i) if $S \subseteq \mathbb{A}$ supports $x \in X$, then it also supports $f(x) \in Y$. Moreover, if X and Y are nominal sets, then $supp(f(x)) \subseteq supp(x)$.

(ii) *if f is injective and Y is a nominal set, then X is a nominal set with $supp(x) = supp(f(x))$.*

(iii) *if f is surjective and X is a nominal set, then Y is a nominal set.*

We continue by recalling several examples of nominal sets. For further details we refer to [33].

Example 2.2.4

(i) ***Discrete Sets:*** *Any set X, equipped with the trivial permutation action $\pi \cdot x \stackrel{\text{def}}{=} x$, is a nominal set with $supp(x) = \emptyset$. X is called a **discrete** nominal set.*

(ii) ***Set of Names:*** *The set of names $\mathbb{A}$, equipped with the permutation action $\pi \cdot a \stackrel{\text{def}}{=} \pi(a)$, is a nominal set with $supp(a) = \{a\}$.*

(iii) ***Finite Permutations:*** *The set $Perm(\mathbb{A})$ of finite permutations of $\mathbb{A}$, equipped with the conjugational permutation action $\pi \cdot \tau \stackrel{\text{def}}{=} \pi\tau\pi^{-1}$, is a nominal set with $supp(\tau) = \{a \in \mathbb{A} \mid \tau(a) \neq a\}$. Note that the map $\pi \cdot \tau \stackrel{\text{def}}{=} \pi\tau$ is also a permutation action, but $Perm(\mathbb{A})$ is not a nominal set under this permutation action.*

(iv) ***Binary Cartesian Product:*** *Let X and Y be nominal sets. The binary cartesian product $X \times Y \stackrel{\text{def}}{=} \{(x, y) \mid x \in X \wedge y \in Y\}$, equipped with the pointwise permutation action $\pi \cdot_{X\times Y} (x, y) \stackrel{\text{def}}{=} (\pi \cdot_X x, \pi \cdot_Y y)$, is a nominal set with $supp((x, y)) = supp(x) \cup supp(y)$.*

(v) ***Disjoint Union:*** *Let A and B be nominal sets. The disjoint union $A \oplus B \stackrel{\text{def}}{=} \{(1, x) \mid x \in A\} \cup \{(2, y) \mid y \in B\}$, equipped with the permutation action*

$$\pi \cdot z \stackrel{\text{def}}{=} \begin{cases} (1, \pi \cdot_X x) & z = (1, x) \\ (2, \pi \cdot_Y y) & z = (2, y) \end{cases}$$

is a nominal set with $supp((i, z)) = supp(z)$ $(i = 1, 2)$.

(vi) ***Nominal Powerset:*** *Let X be a Perm($\mathbb{A}$)-set. The ordinary powerset $\mathcal{P}(X)$, equipped with the permutation action $\pi \cdot_{\mathcal{P}(X)} S = \{\pi \cdot_X x \mid x \in S\}$ for $S \subseteq X$, is a Perm($\mathbb{A}$)-set. Note that if X is a nominal set, it does not generally hold that $\mathcal{P}(X)$ is a nominal set. A counter example is $\{a_1, a_3, a_5, ...\} \subset \mathbb{A}$. However, for any Perm($\mathbb{A}$)-set X, the* ***nominal powerset*** *$\mathcal{P}_{fs}(X)$ is a nominal set by construction.*

$$\mathcal{P}_{fs}(X) \stackrel{\text{def}}{=} \{S \subseteq X \mid S \text{ is finitely supported w.r.t. } \cdot_{\mathcal{P}(X)}\}$$

(vii) ***Linear Order:*** *Let $\mathcal{X}$ be a set of nominal sets, which is linearly ordered by the equivariant subset relation. Then the union $\bigcup \mathcal{X} = \{x \mid \exists X \in \mathcal{X} \text{ such that } x \in X\}$, equipped with the permutation action $\pi \cdot x \stackrel{\text{def}}{=} \pi \cdot_X x$ (if $x \in X \in \mathcal{X}$), is a nominal set with x being supported by the finite set of names that support it in any $X \in \mathcal{X}$ where $x \in X$.*

(viii) ***Quotient Structure:*** *Let X be a Perm($\mathbb{A}$)-set and $R \subseteq X \times X$ an equivariant equivalence relation. Then the quotient set ${}^{X}/_{R}$ of equivalence classes $[x]_R$, equipped with the permutation action $\pi \cdot [x]_R \stackrel{\text{def}}{=} [\pi \cdot x]_R$, is a Perm($\mathbb{A}$)-set and the quotient function $x \in X \mapsto [x]_R \in {}^{X}/_{R}$ is a surjective equivariant function. Moreover, if X is a nominal set, then ${}^{X}/_{R}$ is a nominal set with $supp(q) = \bigcap\{supp(x) \mid x \in q\}$ for $q \in {}^{X}/_{R}$.*

(ix) Let X be a nominal set, then the set of emptily supported elements of X, denoted by $(X)_{es} \stackrel{\text{def}}{=} \{x \in X \mid (\forall x)\ \pi \cdot x = x\}$, is an equivariant subset of X. Given that X is a nominal set, we have that $(X)_{es}$ is a nominal set as well.

The **disagreement set** between two permutations, denoted by $ds(\pi, \pi')$, is defined by $\{a \in \mathbb{A} \mid \pi(a) \neq \pi'(a)\}$. It follows directly that $ds(\pi, \pi') = supp(\pi^{-1}\pi')$.

Freshness

Definition 2.2.5 *The notion of **freshness**, which is denoted by $a \# x$, is complementary to the notion of support, i.e.*

$$a \# x \iff a \notin supp(x)$$

The set of confinite sets of names is denoted by $\text{И} \stackrel{\text{def}}{=} \{S \subseteq \mathbb{A} \mid \mathbb{A} \setminus S \text{ is finite}\}$. If P is a property of names, then the **freshness quantifier** $(\text{И}a)\ P$ asserts that the set $\{a \in \mathbb{A} \mid P(a)\}$ is in И. So, the freshness quantifier $(\text{И}a)P$ expresses that P holds for all but finitely many names. The freshness quantifier can now be used to given an alternative definition of freshness:

Lemma 2.2.6 *Let X be a nominal set and $a \in \mathbb{A}$. Then*

$$a \# x \iff (\text{И}b)\ (a\,b) \cdot x = x$$

As demonstrated in the following theorem, the freshness quantifier can under certain conditions be read as "for some/any fresh name a" $P(a)$ holds.

Theorem 2.2.7 (Some/Any Theorem) *If X is a nominal set and $R \subseteq \mathbb{A} \times X$ is an equivariant subset, then for any $x \in X$, the following conditions are equivalent*

(i) $(\exists a \in \mathbb{A})\ a \# x \wedge (a, x) \in R$

(ii) $(\forall a \in \mathbb{A})\ a \# x \Longrightarrow (a, x) \in R$

(iii) $(\text{И}a)(a, x) \in R$

Note that the previous definitions and results can easily be generalised to finite sets of names. In this case we often write $\overline{a} \# x$ to mean that for all $a \in \overline{a}$ we have that $a \# x$ holds.

Name Abstraction, Concretion and Local Fresh Atomic Names

We now recall the notion of name abstraction, concretion and local fresh atomic names. For a detailed account, including the corresponding proofs, we refer to [33].

Name Abstraction and Concretion

Definition 2.2.8 *Let X be a nominal set. The generalised α-equivalence relation $(a, x) \sim (a', x')$ on $\mathbb{A} \times X$ is defined as follows:*

$$(a, x) \sim (a', x') \iff (\mathit{И}b)^1\ (a\ b) \cdot x = (a'\ b) \cdot x'$$

Lemma 2.2.9 *The generalised α-equivalence relation $\sim$ is an equivariant equivalence relation.*

Using the above Lemma, the $\mathit{И}$-quantifier in Definition 2.2.8 can now be justified by Theorem 2.2.7.

Definition 2.2.10 *The **set of name abstractions on a nominal** X, denoted by $[\mathbb{A}]X$, is defined to be the quotient set $\mathbb{A} \times X/_{\sim}$. The equivalence class of a pair $(a, x) \in \mathbb{A} \times X$ is called the **name abstraction of** a **in** x and is denoted by $\langle a \rangle x$.*

Lemma 2.2.11 *Let X be a nominal set and $\langle a \rangle x \in [\mathbb{A}]X$. Then*

$$(\mathit{И}b)^2\ \langle a \rangle x = \langle b \rangle ((b\ a) \cdot x)$$

As shown in Example 2.2.4 (vii), the set of name abstractions $[\mathbb{A}]X \stackrel{\text{def}}{=} \{\langle a \rangle x \mid a \in \mathbb{A} \wedge x \in X\}$, equipped with the permutation action $\pi \cdot \langle a \rangle x \stackrel{\text{def}}{=} \langle \pi \cdot a \rangle \pi \cdot x$ is a nominal set and $supp(\langle a \rangle x) \subseteq supp(x) \cup \{a\}$. As expressed in the following Lemma, this can be further strengthened.

[1] $b \# (a, a', x, x')$
[2] $b \# (a, x)$

Lemma 2.2.12 *Let X be a nominal set, $a \in \mathbb{A}$ and $x \in X$. Then*

$$supp(\langle a \rangle x) = supp(x) \setminus \{a\}$$

Let X be a nominal set. It can be shown that any name abstraction $F \in [\mathbb{A}]X$ is a subset of $\mathbb{A} \times X$ and for any $a \in \mathbb{A}$ and $x_1, x_2 \in X$ the following property holds: if $(a, x_1) \sim (a, x_2)$, then $x_1 = x_2$. Moreover, $Dom(F) = \{a \in \mathbb{A} \mid a \# F\} = \mathbb{A} \setminus supp(F)$, and as previously demonstrated, F is finitely supported. Thus, F is a finitely supported partial function from $\mathbb{A}$ to X:

$$[\mathbb{A}]X \subseteq \mathbb{A} \rightharpoonup_{fs} X$$

Further, there exists an operation, referred to as **concretion**, which applies a name abstraction $F \in [\mathbb{A}]X$ to a name $a \in Dom(F)$. It is denoted by $F \mathbin{@} a$ and is called the concretion of F at a. In particular, the partial concretion function $@ : [\mathbb{A}]X \otimes \mathbb{A} \to X$ with $\otimes$ being the tensor product, is defined as follows:

$$\langle a \rangle x \mathbin{@} b \stackrel{\text{def}}{=} \begin{cases} x & \text{if } a = b \\ (a\,b) \cdot x & \text{if } a \neq b \wedge b \# x \\ \uparrow & \text{o/w} \end{cases}$$

It follows by standard computations that the function is equivariant, and therefore $supp(F \mathbin{@} a) \subseteq supp(F) \cup \{a\}$.

Lemma 2.2.13 *If $F \in [\mathbb{A}]X$ and $a \# F$, then $F = \langle a \rangle (F \mathbin{@} a)$.*

Analogous to ordinary function spaces, an extensionality principle for name abstractions can be obtained.

Lemma 2.2.14 *For all $F, F' \in [\mathbb{A}]X$ [$(\mathsf{M}b)$ $F \mathbin{@} b = F' \mathbin{@} b \iff F = F'$]*

Local Fresh Atomic Names

When working with the nominal set model, we often need to choose fresh names to define certain constructions. To ultimately show that a construction is well defined, it has to be verified that the construction is independent of the name chosen. The following theorem, referred to as the "Freshness Theorem", provides a simple condition for such independence proofs (see Pitts [56]).

Theorem 2.2.15 (Freshness Theorem) *Let X be a nominal set. If a finitely supported partial function $F \in \mathbb{A} \rightharpoonup_{fs} X$ satisfies*

$$(\mathsf{N}a)\ (\exists x)[a \# x \wedge F(a) \equiv x]\ (\diamond 1)$$

then there exists a unique element $\mathrm{fresh}_X F \in X$ *satisfying*

$$(\mathsf{N}a)\ [F(a) \equiv \mathrm{fresh}_X F]\ (\diamond 2)$$

Let $\Lambda a \in \mathbb{A} \to \varphi(a)$ be the description of a partial function in $\mathbb{A} \rightharpoonup_{fs} X$ which satisfies $(\diamond 1)$. We then choose to write $\mathrm{fresh}_X(\Lambda a \in \mathbb{A} \to \varphi(a)) \in X$ as fresh a in $\varphi(a)$.

Note that fresh a in $\varphi(a)$ is merely a way to express that the independence property holds. However, if we use this form of locally scoped name in the informal meta-theory of nominal sets, we usually pick a sufficiently fresh name a and directly work with $\varphi(a)$.

2.3 FM-set Model

Due to their kinship, as previously pointed out, we only provide a brief overview of the FM-set model. For an extensive account of the FM-set model we refer to Gabbay [35]. We begin by recalling the definition of the **von Neumann cummulative hierarchy** of sets and the corresponding **von Neumann universe** [63]. The cumulative

hierarchy is the collection of sets $\mathcal{V}_\alpha$, indexed by the class of ordinal numbers, which is defined by transfinite recursion using the empty set and the powerset operation $\mathcal{P}$:

$$\begin{aligned} \mathcal{V}_0 &\stackrel{\text{def}}{=} \emptyset \\ \mathcal{V}_{\alpha+1} &\stackrel{\text{def}}{=} \mathcal{P}(\mathcal{V}_\alpha) \\ \mathcal{V}_\lambda &\stackrel{\text{def}}{=} \bigcup_{\alpha<\lambda} \mathcal{V}_\alpha \qquad \text{(with } \lambda \text{ a limit ordinal)} \end{aligned}$$

The **von Neumann universe** is a proper class, which is defined to be the union of $\mathcal{V}_\alpha$ over the ordinals α.

$$\mathcal{V} \stackrel{\text{def}}{=} \bigcup_{\alpha} \mathcal{V}_\alpha$$

The **Fraenkel-Mostowski cummulative hierarchy**, also referred to as the FM-hierarchy, extends the von Neumann cummulative hierarchy with names $\mathbb{A}$ (urelements) and furthermore restricts to the nominal powerset operation of finitely supported subsets $\mathcal{P}_{fs}$ to generate the collection of hereditary finitely supported sets:

$$\begin{aligned} \mathcal{FM}_0 &= \emptyset \\ \mathcal{FM}_{\alpha+1} &= \mathbb{A} + \mathcal{P}_{fs}(\mathcal{FM}_\alpha) \\ \mathcal{FM}_\lambda &= \bigcup_{\alpha<\lambda} \mathcal{FM}_\alpha \qquad \text{(with } \lambda \text{ a limit ordinal)} \end{aligned}$$

We recall that $\emptyset$ and $\mathbb{A}$ are nominal sets. Further, we have that $\mathcal{P}_{fs}(\mathcal{FM}_\alpha)$ is a nominal set, as well as $\mathbb{A} + \mathcal{P}_{fs}(\mathcal{FM}_\alpha)$. Hence, we have that $\mathcal{FM}_{\alpha+1}$ is a a nominal set as well. For limit ordinals λ, we recall that each $\mathcal{FM}_\alpha$ is an equivariant subset of $\mathcal{FM}_{\alpha+1}$, and therefore the union over all $\alpha < \lambda$ is also a nominal set with $x \in \mathcal{FM}_\lambda$ being finitely supported as an element of some $\mathcal{FM}_\alpha$ ($\alpha < \lambda$). Hence, we have a hierarchy of nominal sets.

The **Fraenkel-Mostowski universe**, also referred to as the FM-universe, is defined to be the union over all $\mathcal{FM}_\alpha$ (as α ranges over the ordinals):

$$\mathcal{FM} = \bigcup_{\alpha} \mathcal{FM}_\alpha$$

$\mathcal{FM}$ is a proper class such that $\mathbb{A} + \mathcal{P}_{fs}(\mathcal{FM}) = \mathcal{FM}$. Hence, every element of $\mathcal{FM}$ is either a name or a set, which is formally denoted as $(0, a)$ for $a \in \mathbb{A}$ and $(1, X)$ for X being a finitely supported subset of elements of $\mathcal{FM}$. We call $(1, X)$ an **FM-set**. For brevity we usually just write X. Based on the permutation action for $\mathbb{A}$ and $\mathcal{P}_{fs}(-)$, the permutation action for $\mathcal{FM}$ is recursively defined as follows:

$$\pi \cdot a \stackrel{\text{def}}{=} \pi(a) \qquad\qquad \pi \cdot X \stackrel{\text{def}}{=} \{\pi \cdot x \mid x \in X\}$$

Hence, $\mathcal{FM}$ is actually a nominal class. Moreover, we have that the FM-set model is sufficiently rich to encode many of the usual set-theoretic structures for FM-sets. As an example we present cartesian products and function spaces with their respective permutation actions and support:

Definition 2.3.1 *An **ordered pair** is represented, using the Kuratowski implementation, by* $(x, y) \stackrel{\text{def}}{=} \{\{x\}, \{x, y\}\}$. *The **cartesian product** of FM-sets* X *and* Y *is an FM-set, which is defined as* $X \times Y \stackrel{\text{def}}{=} \{(x, y) \mid x \in X \wedge y \in Y\}$. *The deduced permutation action is* $\pi \cdot (x, y) = (\pi \cdot x, \pi \cdot y)$ *with* $supp((x, y)) = supp(x) \cup supp(y)$.

Definition 2.3.2 *Let* X *and* Y *be FM-sets. An **FM-function** (finitely supported function)* f *between FM-sets* X *and* Y *is an FM-set of ordered pairs from* $X \times Y$, *where each* $x \in X$ *occurs exactly once as the first component of a pair. The deduced permutation action is the conjugation action:*

$$(\pi \cdot f)(x) = \pi \cdot f(\pi^{-1} \cdot x)$$

This is deduced as follows: Let $z \in \pi \cdot X$. *We then have* $(\pi \cdot f)(z) = z'$ *for some* $z' \in \pi \cdot Y$ *and* $(z, z') \in \pi \cdot f$. *Note that* $z = \pi \cdot x$ *for some* $x \in X$, $z' = \pi \cdot y$ *for some* $y \in Y$ *and* $(x, y) \in f$. *Next, we can compute* $\pi \cdot f(\pi^{-1} \cdot z) = \pi \cdot f(\pi^{-1} \cdot \pi \cdot x) = \pi \cdot f(x) = \pi \cdot y = z' = (\pi \cdot f)(z)$. *Moreover, we can deduce that* $\pi \cdot f(x) = (\pi \cdot f)(\pi \cdot x)$

We call an FM-function with empty support an equivariant function and an equivariant FM-set a nominal set. Further, similar to ordinary set-theory, we usually refer

to an FM-set without specifying its transfinite recursive foundations. As indicated before, the notions of name abstraction, concretion and local fresh atomic names also exist in the FM-set model (with minor adaption) (see [33]). In addition, we require the notion of freshness restricted sets:

Definition 2.3.3 *For any FM-Set X and finite set of names $\overline{a}$ such that $\overline{a} \# X$, the* ***freshness restricted set*** $X^{\#\overline{a}} \stackrel{\text{def}}{=} \{x \in X \mid \overline{a} \# x\}$*, equipped with the permutation action of X, is an FM-Set with $supp(X^{\#\overline{a}}) = supp(X) \cup \overline{a}$.*

Lemma 2.3.4 $(\_)^{\#(\_)} : ob(FMSet) \otimes \mathcal{P}_{fin}(\mathbb{A}) \to ob(FMSet)$ *is equivariant.*

Proof Let X be an FM-Set and $\overline{a}$ a finite set of names such that $\overline{a} \# X$. Then $X^{\#\overline{a}}$ is a freshness restricted set. We can then deduce that $\pi \cdot X$ is an FM-Set and by equivariance of freshness that $\pi \cdot \overline{a} \# \pi \cdot X$. Hence, we have that $\pi \cdot X^{\#\pi\cdot\overline{a}}$ is a freshness restricted set as well. What remains to be shown is that $\pi \cdot X^{\#\overline{a}} = \pi \cdot X^{\#\pi\cdot\overline{a}}$. We recall that $\pi \cdot X^{\#\overline{a}} \stackrel{\text{def}}{=} \pi \cdot \{x \in X \mid \overline{a} \# x\} \stackrel{\text{def}}{=} \{\pi \cdot x \mid x \in X \wedge \overline{a} \# x\}$. Let's take any $\pi \cdot x$ with $x \in X$ and $\overline{a} \# x$. Then, by equivariance, we have that $\pi \cdot x \in \pi \cdot X$ and $\pi \cdot \overline{a} \# \pi \cdot x$, and therefore by definition we have that $\pi \cdot x \in (\pi \cdot X)^{\#\pi\cdot\overline{a}}$. Hence, $\subseteq$ holds. The converse follows similarly. □

2.4 Nominal/FM Categories

We use the following standard notation for categories [43]: For a category $\mathcal{C}$ the class of objects and morphisms is denoted by $ob\,\mathcal{C}$ and $mor\,\mathcal{C}$, respectively. Each morphism f in $\mathcal{C}$ has a unique domain object A and codomain object B, which is denoted by $f : A \to B$ or $f \in \mathcal{C}(A, B)$, where $\mathcal{C}(A, B)$ denotes the hom-class of all morphisms from A to B. The composition of morphisms $f : A \to B$ and $g : B \to C$ is written as $g \circ f : A \to C$ and the identity morphism for an object A is written as id_A.

Based on the previously introduced notions of the nominal and FM-set model, we introduce three categories, also referred to as nominal/FM categories:

- *Nom* - the category of nominal sets and equivariant functions.
- *FMNom* - the category of nominal sets and finitely supported functions
- *FMSet* - the category of FM-Sets and FM-functions

We can immediately observe the following inclusions

$$Nom \hookrightarrow FMNom \hookrightarrow FMSet$$

where the first is a lluf subcategory and the second is a full subcategory. For any nominal/FM category X, we have the obvious forgetful functor to *Set*, namely $(|X|, \cdot) \mapsto |X|$.

Next, we present various concrete limits and colimits in nominal/FM categories, which will be used on various occasions in this thesis.

Lemma 2.4.1 *Nom, FMNom and FMSet have*

(i) Terminal objects

(ii) Binary Products

(iii) Equaliser

Proof The constructions follow similarly to *Set*. So, we only need to demonstrate that the respective nominal/FM properties hold:

(i) A terminal object, 1, is a singleton set, equipped with the trivial permutation action. It follows immediately that it is a nominal set (FM-set). For any object X, the unique function $f : X \to 1$, which maps every element to the element of the singleton set, is clearly equivariant (emptily supported).

(ii) For any objects X and Y the triple $(X \times Y, pr_X, pr_Y)$ is a binary product with the cartesian product $X \times Y \stackrel{\text{def}}{=} \{(x, y) \mid x \in X \wedge y \in Y\}$ and the usual projection maps pr_X and pr_Y. As previously recalled, $X \times Y$ is a nominal set (FM-Set). Further, it follows by standard computations that the projection maps are equivariant (emptily supported). For the universal property, it can directly be deduced that for any object Z and functions $f : Z \to X$ and $g : Z \to Y$, the mediating function $\langle f, g\rangle : Z \to X \times Y$, which is defined as $\langle f, g\rangle(z) \stackrel{\text{def}}{=} (f(z), g(z))$, is equivariant (finitely supported by $supp(f) \cup supp(g)$).

(iii) Let X and Y be two objects and $f, g : X \to Y$ two parallel functions. An equaliser is provided by the object $E \stackrel{\text{def}}{=} \{x \in X \mid f(x) = g(x)\}$ with the inclusion function $e : E \subseteq X$. It can easily be deduced that e is equivariant (finitely supported by $supp(f) \cup supp(g) \cup supp(X)$). Given that e is injective, we can directly deduce, by Lemma 2.2.3 (ii), that E is a nominal set. It follows by routine computations that E is an FM-Set (finitely supported by $supp(f) \cup supp(g) \cup supp(X)$). For the universal property, we have that for any object Z and morphism h, the mediating function is h; thus the mediating function is equivariance (finitely support) by construction.

□

Lemma 2.4.2 *Nom, FMNom and FMSet have*

(i) Initial Objects

(ii) Binary CoProducts

(iii) CoEqualiser

Proof The basic constructions follow similarly to *Set*. So, we only need to demonstrate that the respective nominal/FM properties hold:

(i) An initial object, $\emptyset$, is the empty set, which is trivially a nominal set (FM-Set) via the empty permutation action. For any object X, there exists a unique morphism, namely the empty function, which is trivially equivariant (emptily supported).

(ii) A binary coproduct for objects A and B is the triple $(A \oplus B, \mathit{incl}_A, \mathit{incl}_B)$, where $A \oplus B \stackrel{\text{def}}{=} \{(1, x) \mid x \in X\} \cup \{(2, y) \mid y \in Y\}$ is the disjoint union of A and B and incl_A and incl_B are the usual inclusion maps. As previously recalled, $A \oplus B$ is a nominal set (FM-Set). Further, it follows by standard computations that the inclusion maps are equivariant (finitely supported). For any object Z and functions $f : X \to Z$ and $g : Y \to Z$, the mediating function $[f, g] : X \oplus Z \to Z$, defined as $[f, g]((1, x)) \stackrel{\text{def}}{=} f(x)$ and $[f, g]((2, y)) \stackrel{\text{def}}{=} g(y)$, is equivariant (finitely supported with $\mathit{supp}([f, g]) = \mathit{supp}(f) \cup \mathit{supp}(g)$).

(iii) Let X and Y be two objects and $f, g : X \to Y$ parallel morphisms. We define the relation $R \stackrel{\text{def}}{=} \{(f(x), g(x)) \mid x \in X\} \subseteq Y \times Y$ and then form the reflexive, symmetric and transitive closure to obtain the minimal equivalence relation containing R, which we refer to as $\sim \subseteq Y \times Y$. We first define the reflexive and symmetric closure as $R_1 \stackrel{\text{def}}{=} R \cup \{(y, y) \mid y \in Y\} \cup \{(y, x) \mid (x, y) \in R\}$. The transitive closure is given by $\sim \stackrel{\text{def}}{=} \bigcup_{i \in \omega} R_i$, where R_i is inductively defined with R_1 as base case and for $i > 1$,

$$R_{i+1} \stackrel{\text{def}}{=} R_1 \circ R_i \stackrel{\text{def}}{=} \{(x, z) \in Y \times Y \mid \exists y \in Y.(x, y) \in R_1 \wedge (y, z) \in R_i\}$$

We then prove by induction on R_i that $R_i(y, y')$ implies $R_i(\pi \cdot y, \pi \cdot y')$. From this we can directly determine that $y \sim y'$ implies $\pi \cdot y \sim \pi \cdot y'$ and therefore $\sim$ is an equivariant subset of $Y \times Y$. A coequaliser is defined as $Q \stackrel{\text{def}}{=} {}^{Y}\!/_{\sim}$ with morphism $q : Y \to {}^{Y}\!/_{\sim}$ being defined as $y \mapsto [y]_{\sim}$. By Example 2.2.4 (viii) we have that Q is a nominal set (FM-Set) and q is an equivariant (emptily supported) function.

For object P and function $p : Y \to P$ such that $p \circ f = p \circ f$, the mediating morphism m is defined as $m([y]_\sim) \stackrel{\text{def}}{=} p(y)$. The fact that m is equivariant (finitely supported) follows immediately.

□

Let's now consider the notion of a **global element** for an object X in nominal/FM categories, which is defined as $p : 1 \to X$. In the case of *Nom* we have that any global element p is equivariant, and therefore the image $p(*)$ is emptily supported:

$$\pi \cdot p(*) = p(\pi \cdot *) \stackrel{\text{def}}{=} p(*)$$

Hence, we have that for any nominal set X

$$Nom(1_{Nom}, X) \cong_{Set} |(X)_{es}|$$

Using the fact that the image of a global element is emptily supported, we can easily construct parallel functions $f, g : X \to Y$ such that $f \circ p = g \circ p$ for all global elements $p : 1 \to X$, but $f(x) \neq g(x)$ for non-emptily supported elements $x \in X$. Thus, *Nom* is not well-pointed.

For *FMNom* and *FMSet* we have that any global element is finitely supported and therefore

$$FMSet(1_{FMSet}, X) \cong_{Set} |X|$$
$$FMNom(1_{FMNom}, X) \cong_{Set} |X|$$

Moreover, contrary to *Nom*, it can easily be deduced that *FMSet* and *FMNom* are well-pointed. We conclude by showing that all three categories are cartesian closed.

Proposition 2.4.3 *The categories Nom, FMNom and FMSet are cartesian closed.*

Proof It is well known that *Nom* is cartesian closed (see Section 3.2 in [54]). The fact that *FMSet* is cartesian closed follows with minor modifications, as will now be

demonstrated. We omit the case for *FMNom* as it is a combination of *Nom* and *FMSet*. As we have already demonstrated in Lemma 2.4.1, *FMSet* has cartesian products. We can now proceed with exponentials:

As in *Nom*, the evaluation function $ev : (B \Rightarrow C) \times B \to C$ is defined as $ev(f, x) \stackrel{\text{def}}{=} f(x)$ and given any objects B and C the exponential $B \Rightarrow_{fs} C$ is defined to be the set of finitely supported functions with respect to the permutation action:

$$(\pi \cdot_{\Rightarrow_{fs}} f)(x) \stackrel{\text{def}}{=} \pi \cdot_C f(\pi^{-1} \cdot_B x)$$

Hence, by construction we have that $B \Rightarrow_{fs} C$ is an FM-Set with $supp(B \Rightarrow_{fs} C) = supp(B) \cup supp(C)$. It follows by routine computations that ev is emptily supported. We continue with the universal property: Let $f : A \times B \to C$. The exponential mate $\lambda(f) : A \to B \Rightarrow_{fs} C$ is defined as $\lambda(f)(x) \stackrel{\text{def}}{=} \lambda y.f(x, y)$ for each $x \in X$. $\lambda(f)(x)$ is finitely supported by $supp(f) \cup supp(x)$ and therefore $\lambda(f)(x) \in B \Rightarrow_{fs} C$. Hence, we have that $\lambda(f)$ is well defined. Further, $\lambda(f)$ is finitely supported by $supp(f)$. □

Chapter 3

Nominal Lambda Calculus

In this chapter we introduce the Nominal Lambda Calculus, referred to as NLC, which is an extension of the ordinary typed lambda calculus ($\lambda^{\rightarrow}$) that incorporates ideas and concepts of Nominal Equational Logic (NEL) [18, 13, 17]. The decision to choose NEL as our foundation is a direct consequence of our initial goal to extend the categorical type theory correspondence of NEL. Due to the influence of NEL, regarding the definition of NLC, we will recall details of NEL on several occasions to justify and explain various design decisions in the context of NLC and its model-theoretic semantics. For a full account of NEL, its model-theoretic semantics and properties we refer the reader to [13].

Before we introduce NLC, we provide a quick overview of NEL and its basic ideas and features. We begin by recalling that NEL is itself a direct extension of equational logic (EL) and EL provides a formal framework for reasoning about equations in context of the form $x_i : s_i \vdash^{\text{EL}} M = M' : s'$ (see for example [46, 48]). Further, a model-theoretic semantics of EL in *Set* interprets variables and terms as elements of sets with the interpretation of terms depending on a semantic environment for variables [48, 71]. Towards introducing NEL, the fundamental idea is that variables are now interpreted as elements of FM-sets. Thus, if one seeks a notion of sound and complete equational theories with respect to a model-theoretic semantics in *FMSet*,

one expects NEL to capture the notion of a permutation action and the corresponding notions of finite support and freshness. This means that judgements $\overline{a} \# M$ are provided, which assert that a name a is fresh for M, apart from the usual equations $M = M'$. This would require that we are able to assert hypotheses $\overline{a} \# x$ about variables x that may occur (freely) in M. Indeed in NEL one sees freshness contexts $\nabla \stackrel{\text{def}}{=} \overline{a}_1 \# x_1 : s_1, ..., \overline{a}_n \# x_n : s_n$ and freshness assertions $\nabla \vdash^{\text{NEL}} \overline{a} \# M : s$ capturing the intuition that if sets of names $\overline{a}_i$ are fresh for (the interpretation of) the x_i, then $\overline{a}$ is fresh for (the interpretation of) M. Given that freshness can be defined equationally in *FMSet* (see Lemma 2.2.6) as

$$\overline{a} \# x \Longleftrightarrow (И\overline{b})\ (\overline{b}\,\overline{a}) \cdot x = x$$

one can mimic this characterisation of freshness and work with purely equational judgements instead. Hence, provability of $\nabla \vdash^{\text{NEL}} \overline{a} \# M : s$ can be expressed by provability of a certain kind of equation [16]:

$$Th \rhd \nabla \vdash^{\text{NEL}} \overline{a} \# M : s \stackrel{\text{def}}{=} (И\overline{b})\ Th \rhd \nabla^{\#\overline{b}} \vdash^{\text{NEL}} (\overline{b}\,\overline{a}) * M \approx M : s$$

where $\nabla^{\#\overline{b}} \stackrel{\text{def}}{=} (\overline{a}_1 \cup \overline{b} \# x_1 : s_1, ..., \overline{a}_n \cup \overline{b} \# x_n : s_n)$ and $- * -$ denotes a syntactic permutation action on raw terms. Clouston further introduced a sound model-theoretic semantics for NEL in *FMSet*, interpreting variables as elements of FM-sets, and showed that deductive completeness holds.

Note that in this chapter we will lift various results of NEL and $\lambda^{\rightarrow}$ to NLC. At a conceptual level, we prove analogue results of NEL and $\lambda^{\rightarrow}$, however the technical proof details are quite different, because in formulating NLC we work with *name-dependent simple types*.

3.1 The Meta-Theory of NLC Raw Terms

We first introduce signatures, types and raw terms for NLC, as well as permutation actions, α-equivalence and capture-avoiding substitution on raw terms. Then we

provide a collection of meta-theoretic properties of NLC raw terms, properties which will be applied throughout this thesis.

3.1.1 Signatures, Types and Raw Terms

We recall that in NEL one works with a nominal set of *types*[1]. In NLC we work with a nominal set of ground types and generate from it the nominal set of types using domain (freshness) restricted function types that take the form $s^{\overline{a}} \Rightarrow s'$. This particular form of function types in NLC is motivated as follows: NLC extends NEL, as one would expect, by providing term formers for lambda abstractions and applications. Let's now focus on $\overline{a} \# x : s \vdash^{\text{NEL}} M : s'$, a term in context in NEL. If we want to capture the "mapping" $x \mapsto M$ as an abstraction, we could consider $\lambda x : s.\, M$. However, if we apply $\lambda x : s.\, M$ to a term $N : s$ we also need to ensure that $\overline{a} \# N$ holds. For this reason, to make the freshness restriction visible, we decided to codify the finite set $\overline{a}$ in the abstraction. Hence, an abstraction in NLC naturally takes the form $\lambda^{\overline{a}} x : s.\, M$.

Moreover, we recall that in NEL the model-theoretic semantics of a context element $\overline{a} \# x : s$ (in *FMSet*) is specified by requiring that $[\![x]\!] \in [\![s]\!]^{\#\overline{a}} \stackrel{\text{def}}{=} \{e \in [\![s]\!] \mid \overline{a} \# e\}$. This strongly suggests the use of $s^{\overline{a}}$ as the domain of a function type for abstractions in NLC with a compositional semantics $[\![s^{\overline{a}}]\!] \stackrel{\text{def}}{=} [\![s]\!]^{\#\overline{a}}$. Considering the definition of $\overline{a} \# x : s$ in NEL, we additionally require that $\overline{a} \# s$ for $s^{\overline{a}}$.

Remark 3.1.1 *Instead of labelling λ with a finite set of names $\overline{a}$, we could have introduced freshness restriction types of the form $s^{\overline{a}}$. Compared to $\lambda^{\overline{a}} x : s.\, M$, this would have provide us with a more "elegant" way to codify freshness restrictions for lambda abstractions as $\lambda x : s^{\overline{a}}.\, M$. However, the introduction of freshness restriction types of the form $s^{\overline{a}}$ would require us to introduce a notion of subtyping, an additional complication that we wanted to avoid.*

[1] In [13] "types" are called sorts. We use the word type since it better matches general usage in computer science, and categorical type theory

Following the motivation given above, we now formally introduce signatures, types and raw terms of NLC. We start with an NLC-signature Sg, which is composed of

(i) Gnd_{Sg}, a nominal **set of ground types**. The **set of types** $Type_{Sg}$ is then generated by the BNF grammar $s ::= \gamma \mid s^{\overline{a}} \Rightarrow s$ where γ is any ground type and $\overline{a}$ is a finite set of names such that $\overline{a} \# s$. Since each type s is a finite tree and Gnd_{Sg} is a nominal set, each type s is finitely supported by the following permutation action:

$$\pi \cdot \gamma \stackrel{\text{def}}{=} \pi \cdot_{Gnd_{Sg}} \gamma \qquad \pi \cdot (s^{\overline{a}} \Rightarrow s') \stackrel{\text{def}}{=} (\pi \cdot s)^{\pi \cdot \overline{a}} \Rightarrow (\pi \cdot s')$$

and hence $Type_{Sg}$ is a nominal set of types.

(ii) A nominal **set of (higher order functions) constant symbols** Fun_{Sg}.

(iii) An equivariant typing function $Fun_{Sg} \to Type_{Sg}$, which assigns to each constant symbol c a type s. We usually write this as a **constant typing** $c : s$.

The NLC-calculus for an empty NLC-signature, without constant symbols, and only one base type γ, is called **pure**.

Note that we "inherit" from NEL an alternative approach of how to present name-binding operators. Instead of an explicit notion of name-binding, as it is used in nominal signatures, binding axioms can be used. For more details we refer to [14].

Fixing a set $Var \stackrel{\text{def}}{=} \{\mathsf{V}_1, \mathsf{V}_2, \mathsf{V}_3, \ldots\}$ of (ordered) **variables**, the raw NLC-terms are specified by $M ::= \pi x \mid c \mid \lambda^{\overline{a}} x : s.\, M \mid M\ M$ where πx is called a **suspension** [18, 17] of any variable $x \in Var$ and permutation $\pi \in Perm(\mathbb{A})$. We refer to the set of raw terms for signature Sg by $Term_{Sg}$.

Each occurrence of a variable in a term is either **free** or **bound** (where all occurrences of x in any "subterm" $\lambda^{\overline{a}} x : s.\, M$ are bound). The set of all variables that occur in M, called $var(M)$, and the set of all free variables of M, called $fv(M)$, are recursively defined on M. A term M with $fv(M) = \emptyset$ is called a **closed** term. Furthermore, the set of all names in M, referred to as $name(M)$, is defined recursively

on M. Given that many of our proofs are by induction on the size of raw terms, we also introduce the **size** of a raw term M, which is denoted by $|M|$ and recursively defined on the structure of M. The precise definitions are given in Table 4.6.

– $\lvert\pi x\rvert \stackrel{\text{def}}{=} 1$	– $fv(\pi x) \stackrel{\text{def}}{=} \{x\}$
– $\lvert c\rvert \stackrel{\text{def}}{=} 1$	– $fv(c) \stackrel{\text{def}}{=} \emptyset$
– $\lvert M\ N\rvert \stackrel{\text{def}}{=} \lvert M\rvert + \lvert N\rvert$	– $fv(N\ N') \stackrel{\text{def}}{=} fv(N) \cup fv(N')$
– $\lvert\lambda^{\overline{a}} x : s.\, M\rvert \stackrel{\text{def}}{=} \lvert M\rvert + 1$	– $fv(\lambda^{\overline{a}} x : s.\, N) \stackrel{\text{def}}{=} fv(N) \setminus \{x\}$
	– $name(\pi x) \stackrel{\text{def}}{=} supp(\pi)$
– $var(\pi x) \stackrel{\text{def}}{=} \{x\}$	– $name(c) \stackrel{\text{def}}{=} supp(c)$
– $var(c) \stackrel{\text{def}}{=} \emptyset$	– $name(N\ N') \stackrel{\text{def}}{=} name(N) \cup name(N')$
– $var(N\ N') \stackrel{\text{def}}{=} var(N) \cup var(N')$	– $name(\lambda^{\overline{a}} x : s.\, N) \stackrel{\text{def}}{=} name(N) \cup \overline{a} \cup supp(s)$
– $var(\lambda^{\overline{a}} x : s.\, N) \stackrel{\text{def}}{=} var(N) \cup \{x\}$	

Table 3.1: $|M|$, $fv(M)$, $var(M)$ and $name(M)$ for NLC

3.1.2 Permutation Actions for Raw Terms

We introduce three permutation actions on raw terms: We begin with $\pi \bullet (-)$ ($\pi \in Perm(Var)$), which permutes any occurrence of variables in raw terms [33]. This permutation action will play an important role in defining α-equivalence.

We continue with two standard permutation actions on $Perm(\mathbb{A})$ that we recalled in Example 2.2.4 (iii), namely conjugation (which makes $Perm(\mathbb{A})$ into a nominal set) and left multiplication (which does not). Clouston & Pitts [18] and Gabbay &

Mathijssen [36] defined two permutation actions on terms, referred to as meta-level $\pi \cdot M$ and object-level $\pi * M$ [18, 17], which are syntactic analogues of the actions on $Perm(\mathbb{A})$. We now extend these permutation actions for NLC by defining additional clauses for lambda abstractions and applications. In principle, we simply mimic the permutation action of exponential objects in *FMSet*, which is a (form of) *conjugation action*, as well as the related permutation action on function application. However, there is one important difference. In (categorical) type theory one often works with terms in context. As such, a term M with a free variable x can be regarded as a "morphism" $x \mapsto M$. The meta-level permutation action $\pi \cdot M$ does now also apply the conjugation action to all free variables, whereas this is not the case for the object-level permutation action. Further, as in the case of $Perm(\mathbb{A})$, we will show that the set of raw terms is a nominal set under the meta-level permutation action, but not under the object-level permutation action. However, from a technical point of view the object-level permutation action turns out to be rather useful. This will become clearer, once we introduced the model-theoretic semantics and have proven various auxiliary results involving the object-level permutation action.

Note that the intention of the object-level and meta-level permutation action is to mimic and ultimately capture the permutation actions for exponential objects and function application in *FMSet*. In contrast, variable swapping is introduced to handle variable-binding for lambda abstractions in the calculus. To increase readability we use π and τ for permutations in $Perm(\mathbb{A})$ and μ for permutations in $Perm(Var)$.

The precise recursive definition of such mappings $(\mu \in Perm(Var), M) \mapsto \mu \bullet M$, $(\pi \in Perm(\mathbb{A}), M) \mapsto \pi * M$ and $(\pi \in Perm(\mathbb{A}), M) \mapsto \pi \cdot M$ for NLC is given in Table 3.2. Note that in order to define the object-level permutation action we first define a basic form of substitution $M[\pi^{-1}x/x]$ on raw terms M. We call this a **suspension-substitution**. Informally, free occurrences of x in M are replaced by $\pi^{-1}x$. Formally, the recursive definition is the expected one, where on suspensions we define $(\pi' y)[\pi^{-1}x/x] \stackrel{\text{def}}{=} \pi' y$ if $x \neq y$ and otherwise $(\pi' x)[\pi^{-1}x/x] \stackrel{\text{def}}{=} (\pi'\pi^{-1})x$.

– $\mu \bullet \pi z \stackrel{\text{def}}{=} \pi(\mu(z))$

– $\mu \bullet c \stackrel{\text{def}}{=} c$

– $\mu\bullet(\lambda^{\overline{a}} z : s.\, M) \stackrel{\text{def}}{=} \lambda^{\overline{a}} \mu(z) : s.\, (\mu \bullet M)$

– $\mu \bullet (M\ M') \stackrel{\text{def}}{=} (\mu \bullet M)\ (\mu \bullet M)$

Variable Swapping	
– $\pi \cdot \pi' x \stackrel{\text{def}}{=} (\pi\pi'\pi^{-1})x$	– $\pi * \pi' x \stackrel{\text{def}}{=} (\pi\pi')x$
– $\pi \cdot c \stackrel{\text{def}}{=} \pi \cdot_{Fun_{Sg}} c$	– $\pi * c \stackrel{\text{def}}{=} \pi \cdot_{Fun_{Sg}} c$
– $\pi \cdot (\lambda^{\overline{a}} x : s.M) \stackrel{\text{def}}{=} \lambda^{\pi\cdot\overline{a}} x : \pi \cdot s.\pi \cdot M$	– $\pi * (\lambda^{\overline{a}} x : s.M) \stackrel{\text{def}}{=} \lambda^{\pi\cdot\overline{a}} x : \pi \cdot s.(\pi * (M[\pi^{-1}x/x]))$
– $\pi \cdot (M\ N) \stackrel{\text{def}}{=} (\pi \cdot M)\ (\pi \cdot N)$	– $\pi * (MN) \stackrel{\text{def}}{=} (\pi * M)(\pi * N)$
Meta-Level	Object-Level

Table 3.2: Permutation Actions for NLC

Remark 3.1.2 *Note that a more general definition of substitution is not necessary at this point, because the notion of α-equivalence can be defined using variable swapping instead of capture avoiding substitution, as will be shown later.*

The following technical lemma is used in various proofs by induction on the size of raw terms:

Lemma 3.1.3 *The size of a raw term is not affected by suspension-substitution, variable swapping, the object-level permutation action and the meta-level permutation action, i.e.*

(i) $|M| = |(x\,y) \bullet M|$

(ii) $|M| = |M[\pi^{-1}x/x]|$

(iii) $|M| = |\pi * M|$

(iv) $|M| = |\pi \cdot M|$

Proof Follows by induction on the structure of M. In the case of (iii), we also need to apply (ii).

Before we demonstrate that the mappings in Table 3.2 are indeed permutation actions, we require the following lemma for the object-level permutation action.

Lemma 3.1.4 *For any raw term M, we have $(\pi * M)[\pi^{-1}x/x] = \pi * (M[\pi^{-1}x/x])$.*

Proof Proof by induction on the size of raw term M.

SUSP: (M is τx)

$$\begin{aligned}(\pi * \tau x)[\pi^{-1}x/x] &\stackrel{\text{def}}{=} \pi\tau x[\pi^{-1}x/x] \\ &\stackrel{\text{def}}{=} \pi\tau\pi^{-1}x \\ &\stackrel{\text{def}}{=} \pi * \tau\pi^{-1}x \\ &\stackrel{\text{def}}{=} \pi * \tau x[\pi^{-1}x/x]\end{aligned}$$

CONST: (M is c)

$$(\pi * c)[\pi^{-1}x/x] \stackrel{\text{def}}{=} (\pi \cdot c)[\pi^{-1}x/x] \stackrel{\text{def}}{=} \pi \cdot c \stackrel{\text{def}}{=} \pi * c \stackrel{\text{def}}{=} \pi * c[\pi^{-1}x/x]$$

LAM-ABS: (M is $\lambda^{\overline{a}}y : s.\, N$). We consider the case $x \neq y$ (the other case follows immediately).

$$\begin{aligned}(\pi * \lambda^{\overline{a}}y : s.N)[\pi^{-1}x/x] &\stackrel{\text{def}}{=} (\lambda^{\pi\cdot\overline{a}}y : \pi \cdot s.\pi * N[\pi^{-1}y/y])[\pi^{-1}x/x] \\ &\stackrel{\text{def}}{=} \lambda^{\pi\cdot\overline{a}}y : \pi \cdot s.(\pi * N[\pi^{-1}y/y])[\pi^{-1}x/x] \\ &= \lambda^{\pi\cdot\overline{a}}y : \pi \cdot s.\pi * (N[\pi^{-1}y/y][\pi^{-1}x/x]) \qquad \text{(induction)} \\ &= \lambda^{\pi\cdot\overline{a}}y : \pi \cdot s.\pi * ((N[\pi^{-1}x/x])[\pi^{-1}y/y]) \qquad (x \neq y) \\ &\stackrel{\text{def}}{=} \pi * (\lambda^{\overline{a}}y : s.N[\pi^{-1}x/x]) \\ &\stackrel{\text{def}}{=} \pi * (\lambda^{\overline{a}}y : s.N)[\pi^{-1}x/x]\end{aligned}$$

APP: (M is $P\ Q$)

$$\begin{aligned}(\pi * P\ Q)[\pi^{-1}x/x] &\stackrel{\text{def}}{=} (\pi * P)[\pi^{-1}x/x]\ (\pi * Q)[\pi^{-1}x/x] \\ &= \pi * (P[\pi^{-1}x/x])\ \pi * (Q[\pi^{-1}x/x]) \qquad \text{(induction)} \\ &\stackrel{\text{def}}{=} \pi * (P\ Q[\pi^{-1}x/x])\end{aligned}$$

□

Proposition 3.1.5 (Permutation Action Definitions)

- *The mapping* $(\mu \in Perm(Var), M) \mapsto \mu \bullet M$ *is a permutation action; we call it* ***variable swapping****. A raw term* M *is finitely supported by all the variables occurring in it. So,* $Term_{Sg}$ *is a nominal set with* $supp(M) = var(M)$ *(see [33]). From now onwards we may write* $z \# M$ *instead of* $z \notin var(M)$*.*

- *The mapping* $(\pi \in Perm(\mathbb{A}), M) \mapsto \pi \cdot M$ *is a permutation action; we call it the* ***meta-level*** *permutation action. A raw term* M *is finitely supported by all the atomic names occurring in it. So,* $Term_{Sg}$ *is a nominal set with* $supp(M) = name(M)$*.*

- *The mapping* $(\pi \in Perm(\mathbb{A}), M) \mapsto \pi * M$ *is a permutation action; we call it the* ***object-level*** *permutation action.*

Proof It is easy to see that variable swapping and the meta-level (conjugation) mapping is a permutation action, with the corresponding finite support set. To show that the object-level (left multiplication) mapping is a permutation action, we need to induct on the size of M and appeal to Lemma 3.1.4 in the abstraction case.

We introduce another technical lemma that demonstrates how variable swapping interacts with suspension-substitutions, as well as the object-level and meta-level permutation action:

Lemma 3.1.6 *The following properties hold:*

(i) $(x\,y) \bullet (M[\pi^{-1}z/z]) = ((x\,y) \bullet M)[\pi^{-1}(x\,y)(z)/(x\,y)(z)]$

(ii) $(x\,y) \bullet (\pi * M) = \pi * ((x\,y) \bullet M)$

(iii) $(x\,y) \bullet (\pi \cdot M) = \pi \cdot ((x\,y) \bullet M)$

Proof Follows by induction on the structure of raw term M. In the case of (ii), we need to apply (i).

Lemma 3.1.7 *For any closed raw term M, we have that $\pi * M = \pi \cdot M$*

Proof Follows by induction on the size of raw term M.

3.1.3 α-Equivalence and Capture Avoiding Substitution

So far we have used structural equality on raw terms $M = N$ (also denoted by $M \equiv N$). Since we wish to work with capture avoiding substitution, which makes use of variable renaming, we have to replace $=$ with α-equivalence. This is also necessary to define a confluent reduction system for NLC at a later point.

We first recall the standard definition, as for example in [38], where α-equivalence is defined using the notion of capture avoiding substitution. More precisely, the α-equivalence relation $\sim_\alpha$ is defined to be the smallest equivalence relation closed under the congruence rules (for application and abstraction terms) and the axiom $\lambda^{\overline{a}}x : s.\, M \sim_\alpha \lambda^{\overline{a}}x' : s.\, M\{x'/x\}$ where $x' \notin var(M)$. The notion of capture avoiding substitution is then subsequently lifted to α-equivalent terms.

An alternative way to define α-equivalence is terms of variable swapping [22, 33], where we define a structural congruence relation, $\equiv_\alpha \subset Term_{Sg} \times Term_{Sg}$, using the rules in Table 3.3. Note that both definitions generate the same equivalence relation, which follows analogously to [33] (Proposition 2.2). Hence, α-equivalence via the relation $\equiv_\alpha$ is defined without recourse to capture avoiding substitution. For this thesis, we use $\equiv_\alpha$ as our definition of α-equivalence.

$$\frac{}{\pi x \equiv_\alpha \pi x}\,(x \in \mathit{Var} \quad \pi \in \mathit{Perm}(\mathbb{A})) \qquad \frac{M_1 \equiv_\alpha M_1' \qquad M_2 \equiv_\alpha M_2'}{M_1 M_2 \equiv_\alpha M_1' M_2'}$$

$$\frac{(z\,x) \bullet M_1 \equiv_\alpha (z\,y) \bullet M_2}{\lambda^{\overline{a}} x : s.\, M_1 \equiv_\alpha \lambda^{\overline{a}} y : s.\, M_2}\,(z \# (M_1, M_2, x, y))$$

Table 3.3: Alpha Equivalence by Variable Swapping

We denote the set of α-equivalence classes of raw terms by $\mathit{Term}_{Sg}/_{\equiv_\alpha}$ and call its elements $[M]_\alpha$ **expressions**. We next have to show that various operations, previously defined on raw terms, preserve α-equivalence and can therefore be lifted to expressions.

Lemma 3.1.8 *Suspension-substitution, variable swapping, as well as the object-level and meta-level permutation actions preserve α-equivalence, i.e.*

(i) $M \equiv_\alpha M' \Longrightarrow \mu \bullet M \equiv_\alpha \mu \bullet M'$

(ii) $M \equiv_\alpha M' \Longrightarrow M[\pi^{-1}x/x] \equiv_\alpha M'[\pi^{-1}x/x]$

(iii) $M \equiv_\alpha M' \Longrightarrow \pi * M \equiv_\alpha \pi * M'$

(iv) $M \equiv_\alpha M' \Longrightarrow \pi \cdot M \equiv_\alpha \pi \cdot M'$

Proof All four properties are proved by rule-based induction over $M \equiv_\alpha M'$. We only provide full details for (iii). The other properties follow similarly: Let $M, M' \in \mathit{Term}_{Sg}$ and $\pi \in \mathit{Perm}(\mathbb{A})$.

SUSP: Suppose that $\tau x \equiv_\alpha \tau x$. By the suspension axiom we have $\pi\tau x \equiv_\alpha \pi\tau x$. Given that $\pi * \tau x \stackrel{\text{def}}{=} \pi\tau x$, we immediately get $\pi * \tau x \equiv_\alpha \pi * \tau x$.

LAM: Suppose $\lambda^{\overline{a}} x : s.M_1 \equiv_\alpha \lambda^{\overline{a}} y : s.M_2$. We need to show that $\pi * (\lambda^{\overline{a}} x : s.M_1) \equiv_\alpha \pi * (\lambda^{\overline{a}} y : s.M_2)$, or by definition of $\pi * -$ that $\lambda^{\pi\cdot\overline{a}} x : \pi \cdot s.\pi * M_1[\pi^{-1}x/x] \equiv_\alpha \lambda^{\pi\cdot\overline{a}} y : \pi \cdot s.\pi * M_2[\pi^{-1}y/y]$ holds. We pick a variable z such that $z \# (M_1, M_2, x, y)$.

It follows immediately that $z \# (\pi * M_1[\pi^{-1}x/x], \pi * M_2[\pi^{-1}y/y], x, y)$ is satisfied. We then have to show that $(z\,x) \bullet (\pi * M_1[\pi^{-1}x/x]) \equiv_\alpha (z\,y) \bullet (\pi * M_2[\pi^{-1}y/y])$ holds. Given that $z \# (M_1, M_2, x, y)$ we obtain by assumption and the α-equivalence rule for lambda abstraction that $(z\,x) \bullet M_1 \equiv_\alpha (z\,y) \bullet M_2$. So, by induction we have $\pi * ((z\,x) \bullet M_1) \equiv_\alpha \pi * ((z\,y) \bullet M_2)$. To complete this case, we apply property (ii) from above, as well as Lemma 3.1.4 and Lemma 3.1.6.

APP: Suppose $M_1\ M_2 \equiv_\alpha M_1'\ M_2'$. By the application rule we have $M_1 \equiv_\alpha M_1'$ and $M_2 \equiv_\alpha M_2'$. Then, by induction we have $\pi * M_1 \equiv_\alpha \pi * M_1'$ and $\pi * M_2 \equiv_\alpha \pi * M_2'$. By the application rule we obtain $(\pi * M_1)(\pi * M_2) \equiv_\alpha (\pi * M_1')(\pi * M_2')$ and finally by definition of the object-level permutation action on application we have $\pi * (M_1\ M_2) \equiv_\alpha \pi * (M_1'\ M_2')$. □

Based on the previous equivariance results and Example 2.2.4 (viii), the following well-defined permutation actions on expressions are induced:

$$\mu \bullet [M]_\alpha \stackrel{\text{def}}{=} [\mu \bullet M]_\alpha$$
$$\pi \cdot [M]_\alpha \stackrel{\text{def}}{=} [\pi \cdot M]_\alpha$$
$$\pi * [M]_\alpha \stackrel{\text{def}}{=} [\pi * M]_\alpha$$

Furthermore, the set of α-equivalence classes, $Term_{Sg}/_{\equiv_\alpha}$, is a nominal set under variable swapping, as well as the meta-level permutation action, with $supp([M]_\alpha) = fv(M)$ and $supp([M]_\alpha) = name(M)$, respectively.

Remark 3.1.9 *Having taken great care in defining expressions $[M]_\alpha$, we adopt the usual convention of just writing M. This includes using $=$ instead of $\equiv_\alpha$.* However, our proofs deal correctly with the intricacies that arise from variable re-naming to avoid capture (see for example [53] (page 169) and [47]). *A full example in the context of* NLC *is provided in Lemma 3.2.4 (page 54).*

We continue by extending the recursive definition of capture avoiding substitution for "ordinary" λ-terms as for example presented in [38, 48]. The substitution

operation acts on suspended variables, analogue to NEL, by applying the object-level permutation action. In addition, we use variable swapping to rename variables.

Definition 3.1.10 *Substituting raw term N for every free occurrences of variable x in the raw term M yields another raw term, which we denote by $M\{N/x\}$. The definition follows by recursion on the size of M as follows:*

$$
\begin{array}{rll}
(\pi y)\{N/x\} =_{def} \pi y & & \textit{if } x \neq y \\
(\pi y)\{N/x\} =_{def} \pi * N & & \textit{if } x = y \\
c\{N/x\} =_{def} c & & \\
M\ M'\{N/x\} \stackrel{\text{def}}{=} M\{N/x\}\ M'\{N/x\} & & \\
\lambda^{\overline{a}} y : s.\, M\{N/x\} \stackrel{\text{def}}{=} \lambda^{\overline{a}} y : s.\, M & & \textit{if } x = y \\
\lambda^{\overline{a}} y : s.\, M\{N/x\} \stackrel{\text{def}}{=} \lambda^{\overline{a}} y : s.\, M\{N/x\} & & \textit{if } x \neq y,\ y \# N \\
\lambda^{\overline{a}} y : s.\, M\{N/x\} \stackrel{\text{def}}{=} \lambda^{\overline{a}} z : s.\, ((z\, y) \bullet M)\{N/x\} & & \textit{if } x \neq y,\ \neg(y \# N), \\
 & & \textit{first variable } z \# (M, N)
\end{array}
$$

Note that we assume that the variables are enumerated. The operation of capture avoiding substitution can then be lifted to the level of expressions as follows:

$$[M]_\alpha\{[N]_\alpha/x\} \stackrel{\text{def}}{=} [M\{N/x\}]_\alpha$$

To show that capture avoiding substitution is well-defined, it has to be demonstrated that α-equivalence is preserved, i.e. if $M \equiv_\alpha M'$ and $N \equiv_\alpha N'$, then $M\{N/x\} \equiv_\alpha M'\{N'/x\}$. This follows directly, with minor modifications, from the proof in [24] (Theorem 2 (a), page 95 - 103). This is also the case for the following two identities: Let x and y be two distinct variables.

$$
\begin{array}{ll}
(M\{P/x\})\{Q/x\} = M\{P\{Q/x\}/x\} & \\
(M\{P/y\})\{Q/x\} = (M\{Q/x\})\{(P\{Q/x\})/y\} & \text{if } y \neq x \text{ and } y \# Q
\end{array}
$$

In addition, we require simultaneous capture avoiding substitution of expressions, which will be crucial for defining composition of morphisms in a classifying category– see Section 5.3.5. It can be generalised from capture avoiding substitution: Substituting expressions $N_1, \ldots, N_n$ for free occurrences of the distinct variables $x_1, \ldots, x_n$ in the expression M yields another expression, which we denote by $M\{\vec{N_i}/\vec{x_i}\}$ or $M\{N_1, \ldots, N_n/x_1, \ldots, x_n\}$.

The following technical lemma is implicitly used in many inductive proofs:

Lemma 3.1.11 *For any expression M and variable x, $M[\pi^{-1}x/x] = M\{\pi^{-1}x/x\}$*

Proof Proof by induction on the structure of expression M.

SUSP: (M is τy) We consider the case ($x = y$). The other case follows immediately.

$$(\tau y)[\pi^{-1}x/x] \stackrel{\text{def}}{=} \tau\pi^{-1}x \stackrel{\text{def}}{=} \tau * (\pi^{-1}x) \stackrel{\text{def}}{=} (\tau y)\{\pi^{-1}x/x\}$$

CONST: (M is c)

$$c[\pi^{-1}x/x] \stackrel{\text{def}}{=} c \stackrel{\text{def}}{=} c\{\pi^{-1}x/x\}$$

LAM-ABS: (M is $\lambda^{\overline{a}}y : s.\, N$) We consider the case $x \neq y$. The other case follows immediately. Due to the restricted form of substitution, we do not need to consider the binding case.

$$\begin{aligned}(\lambda^{\overline{a}}y : s.N)[\pi^{-1}x/x] &\stackrel{\text{def}}{=} \lambda^{\overline{a}}y : s.N[\pi^{-1}x/x] \\ &= \lambda^{\overline{a}}y : s.N\{\pi^{-1}x/x\} && \text{(induction)} \\ &\stackrel{\text{def}}{=} (\lambda^{\overline{a}}y : s.N)\{\pi^{-1}x/x\}\end{aligned}$$

APP: (M is $P\ Q$)

$$\begin{aligned}(P\ Q)[\pi^{-1}x/x] &\stackrel{\text{def}}{=} P[\pi^{-1}x/x]\ Q[\pi^{-1}x/x] \\ &= P\{\pi^{-1}x/x\}\ Q\{\pi^{-1}x/x\} && \text{(induction)} \\ &\stackrel{\text{def}}{=} (P\ Q)\{\pi^{-1}x/x\}\end{aligned}$$

□

The next lemma expresses the meta-level permutation action in terms of the object-level permutation action. It will be used on various occasions to prove properties of both NLC and its semantics.

Lemma 3.1.12 ($\cdot$ in terms of $*$) *For any expression M and $\{x_1, ..., x_n\} \subseteq Var$ with $fv(M) \subseteq \{x_1, ..., x_n\}$ we have $\pi \cdot M = (\pi * M)\{\pi^{-1}x_1/x_1, ..., \pi^{-1}x_n/x_n\}$.*

Proof Proof by induction on the structure of expression M of

$$(\forall\pi)(\forall\{x_1, \ldots, x_n\})((fv(M) \subseteq \{x_1 \ldots x_n\} \\ \Longrightarrow \pi \cdot M = (\pi * M)\{\pi^{-1}x_1/x_1, ..., \pi^{-1}x_n/x_n\})$$

SUSP: When M is τx_i the result follows immediately by the definition of substitution and the fact that both are permutation actions.

CONST: Follows immediately.

LAM-ABS: Case M is $\lambda^{\overline{a}}x : s.\, M'$ where $fv(\lambda^{\overline{a}}x : s.\, M') \stackrel{\text{def}}{=} fv(M') \setminus \{x\} \subseteq \{x_1 \ldots x_n\}$. We examine the case when x is not an x_i; if x is an x_i the details are not too dissimilar. Due to the restricted form of substitution, we do not need to consider the binding case. So for the induction step $fv(M') \subseteq \{x, x_1 \ldots, x_n\}$.

$$\begin{aligned}
&\pi \cdot (\lambda^{\overline{a}}x : s.M') \\
&\quad \stackrel{\text{def}}{=} \lambda^{\pi\cdot\overline{a}}x : \pi \cdot s.\pi \cdot M' \\
&\quad = \lambda^{\pi\cdot\overline{a}}x : \pi \cdot s.(\pi * M')\{\pi^{-1}x/x, \pi^{-\vec{1}}x_i/\vec{x_i}\} && \text{(induction)} \\
&\quad = \lambda^{\pi\cdot\overline{a}}x : \pi \cdot s.((\pi * M')\{\pi^{-1}x/x\})\{\pi^{-\vec{1}}x_i/\vec{x_i}\} && (x \neq x_i) \\
&\quad = \lambda^{\pi\cdot\overline{a}}x : \pi \cdot s.(\pi * (M'\{\pi^{-1}x/x\}))\{\pi^{-\vec{1}}x_i/\vec{x_i}\} && \text{(Lemma 3.1.4)} \\
&\quad = (\lambda^{\pi\cdot\overline{a}}x : \pi \cdot s.\pi * (M'\{\pi^{-1}x/x\}))\{\pi^{-\vec{1}}x_i/\vec{x_i}\} && (x \neq x_i, \text{ no capture}) \\
&\quad = (\lambda^{\pi\cdot\overline{a}}x : \pi \cdot s.\pi * (M'[\pi^{-1}x/x]))\{\pi^{-\vec{1}}x_i/\vec{x_i}\} && \text{(Lemma 3.1.11)} \\
&\quad \stackrel{\text{def}}{=} (\pi * (\lambda^{\overline{a}}x : s.M'))\{\pi^{-\vec{1}}x_i/\vec{x_i}\}
\end{aligned}$$

APP: Case M is $N\ N'$.

$$\begin{aligned}
&\pi \cdot (N\ N') \\
&\quad \stackrel{\text{def}}{=} (\pi \cdot N)\ (\pi \cdot N')) \\
&\quad = ((\pi * N)\{\pi^{-\vec{1}} x_i/\vec{x_i}\})\ ((\pi * N')\{\pi^{-\vec{1}} x_i/\vec{x_i}\}) \qquad \text{(induction)} \\
&\quad \stackrel{\text{def}}{=} ((\pi * N)\ (\pi * N'))\{\pi^{-\vec{1}} x_i/\vec{x_i}\} \\
&\quad \stackrel{\text{def}}{=} (\pi * (N\ N'))\{\pi^{-\vec{1}} x_i/\vec{x_i}\}
\end{aligned}$$

□

For brevity, we use $M\{\pi^{-1}\_\}$ as syntactic sugar for $M\{\pi^{-\vec{1}} x_i/\vec{x_i}\}$ with $fv(M) = \{x_1, ..., x_n\}$. We next demonstrate that $\bullet$ distributes over $\{/\}$.

Proposition 3.1.13 $\mu \bullet (M\{N/z\}) = (\mu \bullet M)\{(\mu \bullet N)/\mu \bullet z\}$

Proof Follows directly by induction on the size of M.

The next propositions are crucial for many of the following results, namely that $*$ associates with $\{/\}$ and $\cdot$ distributes over $\{/\}$.

Proposition 3.1.14 $(\pi * M)\{N/x\} = \pi * (M\{N/x\})$

Proof We prove it by induction on the size of expression M.

SUSP: (M is τy) We consider the case ($x = y$). The other case follows immediately.

$$(\pi * (\tau y))\{N/x\} \stackrel{\text{def}}{=} (\pi\tau y)\{N/x\} \stackrel{\text{def}}{=} \pi\tau * N \stackrel{\text{def}}{=} \pi * (\tau * N) \stackrel{\text{def}}{=} \pi * (\tau y)\{N/x\}$$

CONST: (M is c)

$$(\pi * c)\{N/x\} \stackrel{\text{def}}{=} (\pi \cdot c)\{N/x\} \stackrel{\text{def}}{=} \pi \cdot c \stackrel{\text{def}}{=} \pi * c \stackrel{\text{def}}{=} \pi * (c\{N/x\})$$

LAM-ABS: (M is $\lambda^{\overline{a}} y : s.\, M$) We consider case ($x \neq y$) and $y \in fv(N)$. The other cases follow similarly.

$$
\begin{aligned}
&(\pi * (\lambda^{\overline{a}} y : s.M))\{N/x\} \\
&\stackrel{\text{def}}{=} (\lambda^{\pi\cdot\overline{a}} y : \pi \cdot s : \pi * M[\pi^{-1}y/y])\{N/x\} \\
&\stackrel{\text{def}}{=} \lambda^{\pi\cdot\overline{a}} z : \pi \cdot s.\, ((z\, y) \bullet (\pi * M[\pi^{-1}y/y]))\{N/x\} && \text{(for } z \# (M, N)) \\
&= \lambda^{\pi\cdot\overline{a}} z : \pi \cdot s.\, (\pi * (((z\, y) \bullet M)[\pi^{-1}z/z])\{N/x\} && \text{(Lemma 3.1.6, Proposition 3.1.13)} \\
&= \lambda^{\pi\cdot\overline{a}} z : \pi \cdot s.\, \pi * (((z\, y) \bullet M)[\pi^{-1}z/z])\{N/x\} && \text{(induction)} \\
&= \lambda^{\pi\cdot\overline{a}} z : \pi \cdot s.\, \pi * (((z\, y) \bullet M)\{\pi^{-1}z/z\})\{N/x\} && \text{(Lemma 3.1.11)} \\
&= \lambda^{\pi\cdot\overline{a}} z : \pi \cdot s.\, \pi * (((z\, y) \bullet M)\{N/x\})\{\pi^{-1}z/z\} && (z \# (M, N)) \\
&= \lambda^{\pi\cdot\overline{a}} z : \pi \cdot s.\, \pi * (((z\, y) \bullet M)\{N/x\})[\pi^{-1}z/z] && \text{(Lemma 3.1.11)} \\
&\stackrel{\text{def}}{=} \pi * (\lambda^{\overline{a}} z : s.\, ((z\, y) \bullet M)\{N/x\}) \\
&\stackrel{\text{def}}{=} \pi * ((\lambda^{\overline{a}} x : s.M)\{N/x\})
\end{aligned}
$$

APP: (M is $M\ M'$)

$$
\begin{aligned}
(\pi * (M\ M'))\{N/x\} &\stackrel{\text{def}}{=} ((\pi * M)(\pi * M'))\{N/x\} \\
&\stackrel{\text{def}}{=} (\pi * M)\{N/x\}\ (\pi * M')\{N/x\} \\
&= (\pi * M\{N/x\})\ (\pi * M'\{N/x\}) && \text{(induction)} \\
&\stackrel{\text{def}}{=} \pi * ((M\{N/x\})\ (M'\{N/x\})) \\
&\stackrel{\text{def}}{=} \pi * ((M\ M')\{N/x\})
\end{aligned}
$$

□

Proposition 3.1.15 $\pi \cdot (M\{N/x\}) = (\pi \cdot M)\{(\pi \cdot N)/x\}$

Proof This proof is by induction on the size of expression M.

SUSP: (M is $\pi' y$) We consider the case $y = x$. The other case follows immediately.

$$\begin{aligned}
\pi \cdot ((\pi' x)\{N/x\}) &\stackrel{\text{def}}{=} \pi \cdot (\pi' * N) \\
&= (\pi * (\pi' * N))\{\pi^{-1}_\} && \text{(Lemma 3.1.12)} \\
&\stackrel{\text{def}}{=} (\pi\pi' * N)\{\pi^{-1}_\} \\
&= \pi\pi' * (N\{\pi^{-1}_\}) && \text{(Lemma 3.1.14)} \\
&\stackrel{\text{def}}{=} (\pi\pi'\pi^{-1}\pi) * (N\{\pi^{-1}_\}) \\
&\stackrel{\text{def}}{=} (\pi\pi'\pi^{-1}) * (\pi * (N\{\pi^{-1}_\})) \\
&= (\pi\pi'\pi^{-1}) * ((\pi * N)\{\pi^{-1}_\}) && \text{(Lemma 3.1.14)} \\
&= (\pi\pi'\pi^{-1}) * (\pi \cdot N) && \text{(Lemma 3.1.12)} \\
&\stackrel{\text{def}}{=} (\pi\pi'\pi^{-1}) * (\pi \cdot N) \\
&\stackrel{\text{def}}{=} (\pi\pi'\pi^{-1} x)\{\pi \cdot N/x\} \\
&\stackrel{\text{def}}{=} (\pi \cdot (\pi' x))\{\pi \cdot N/x\}
\end{aligned}$$

CONST: (M is c)

$$\pi \cdot (c\{N/x\}) \stackrel{\text{def}}{=} \pi \cdot c \stackrel{\text{def}}{=} (\pi \cdot c)\{\pi \cdot N/x\}$$

LAM-ABS: (M is $\lambda^{\overline{a}} y : s.M$) We consider the case $x \neq y$ and $y \in fv(N)$. The other cases follow similarly.

$$\begin{aligned}
\pi \cdot ((\lambda^{\overline{a}} y : s.M)\{N/x\}) &\stackrel{\text{def}}{=} \pi \cdot (\lambda^{\overline{a}} z : s.\, ((z\, y) \bullet M)\{N/x\}) \\
&\stackrel{\text{def}}{=} \lambda^{\pi \cdot \overline{a}} z : \pi \cdot s.\, \pi \cdot ((z\, y) \bullet M)\{N/x\} \\
&= \lambda^{\pi \cdot \overline{a}} z : \pi \cdot s.\, (\pi \cdot ((z\, y) \bullet M))\{\pi \cdot N/x\} && \text{(induction)} \\
&= \lambda^{\pi \cdot \overline{a}} z : \pi \cdot s.\, ((z\, y) \bullet (\pi \cdot M))\{\pi \cdot N/x\} && \text{(Lemma 3.1.6)} \\
&\stackrel{\text{def}}{=} (\lambda^{\pi \cdot \overline{a}} y : \pi \cdot s.\, \pi \cdot M)\{\pi \cdot N/x\} && (z \# \pi \cdot N) \\
&\stackrel{\text{def}}{=} (\pi \cdot (\lambda^{\overline{a}} y : s.M))\{\pi \cdot N/x\}
\end{aligned}$$

APP: (M is $(M\ M')$)

$$\begin{aligned}
\pi \cdot ((M\ M')\{N/x\}) &\stackrel{\text{def}}{=} \pi \cdot (M\{N/x\}\ M'\{N/x\}) \\
&\stackrel{\text{def}}{=} (\pi \cdot M\{N/x\}\ \pi \cdot M'\{N/x\}) \\
&= ((\pi \cdot M)\{\pi \cdot N/x\}\ (\pi \cdot M')\{\pi \cdot N/x\}) \qquad \text{(induction)} \\
&\stackrel{\text{def}}{=} (((\pi \cdot M)\ (\pi \cdot M))\{\pi \cdot N/x\}) \\
&\stackrel{\text{def}}{=} (\pi \cdot (M\ M'))\{\pi \cdot N/x\}
\end{aligned}$$

□

Both of the previous lemmas can be generalised for simultaneous capture-avoiding substitution:

Proposition 3.1.16 *For expressions M,* distinct *variables $x_1, \ldots, x_n$, and expressions $N_1, \ldots, N_n$ we have*

$$\begin{aligned}
(\pi * M)\{\vec{N_i}/\vec{x_i}\} &= \pi * (M\{\vec{N_i}/\vec{x_i}\}) \\
\pi \cdot (M\{\vec{N_i}/\vec{x_i}\}) &= (\pi \cdot M)\{(\pi \cdot \vec{N_i})/\vec{x_i}\}
\end{aligned}$$

3.2 Typed Expressions and Equational Theories

As our next step, we specify the type and equation system of NLC. The intuitions of NEL and NLC are again rather similar, but technicalities are quite different due to the introduction of name-dependent simple types. We recall that in NEL [13], raw terms are typed using ordinary typing contexts $\Gamma \stackrel{\text{def}}{=} x_1 : s_1, \ldots, x_n : s_n$ with typing judgements being of the form $\Gamma \vdash^{\text{NEL}} M : s$ and equational judgements are given as $\nabla \vdash^{\text{NEL}} M \approx M' : s$ where $\nabla = \overline{a}_1 \# x_1 : s_1, ..., \overline{a}_n \# x_n : s_n$ records assumptions about freshness and types. NEL judgements of the form $\nabla \vdash^{\text{NEL}} M : s$ are simply sugar for reflexive equations. This means that the type system is entirely separate from the freshness/equation system. This is in contrast to NLC, where we

cannot separate the type system in this way, since *the types of abstractions depend directly on freshness assertions*. Thus, the contexts we use in the type system *must* already encode freshness assertions (and cannot be of the form Γ). Further, our typing judgements $\nabla \vdash^{\text{NLC}} M : s$ are not abbreviations for reflexive equations, but *first class citizens* of the calculus.

We now formally introduce the type and equation system for NLC: A **freshness context**, or just **context**, is analogously to NEL a finite partial function $\nabla : \mathit{Var} \to \mathcal{P}_{fin}(\mathbb{A}) \otimes \mathit{Type}_{Sg}$ with $\otimes$ being the tensor product. By definition, it maps each variable $x \in \mathit{dom}(\nabla)$ to a pair $(\overline{a}, s)$ where $\overline{a}$ is a finite set of names, $s \in \mathit{Type}_{Sg}$ and $\overline{a} \# s$. The set of contexts Ctxt_{Sg}, equipped with the permutation action

$$(\pi \cdot \nabla)(x) \stackrel{\text{def}}{=} (\pi \cdot \overline{a}, \pi \cdot s)$$

is a nominal set. We often write a context ∇ for $\overline{a}_1 \# x_1 : s_1, ..., \overline{a}_n \# x_n : s_n$. For contexts ∇ and ∇', we write $\nabla \leq \nabla'$ if $\mathit{dom}(\nabla) \subseteq \mathit{dom}(\nabla')$ and for all $x \in \mathit{dom}(\nabla)$ such that $\nabla(x) = (\overline{a}, s)$ and $\nabla'(x) = (\overline{a}', s')$ we have $\overline{a} \subseteq \overline{a}'$ and $s = s'$. We define expressions and equations in context as follows:

- An **expression-in-context** is a judgement of the form $\nabla \vdash^{\text{NLC}} M : s$ where ∇ is a freshness environment, M is an expression and s is a type.
- An **equation-in-context** is a judgement of the form $\nabla \vdash^{\text{NLC}} M \approx M' : s$ where $\nabla \vdash^{\text{NLC}} M : s$ and $\nabla \vdash^{\text{NLC}} M' : s$ are expressions-in-context.

An NLC-theory Th is a pair (Sg, Ax), where Sg is an NLC-signature and Ax is a collection of equations-in-context. We shall use Th to inductively define a subset of expressions-in-context and equations-in-context. Any expression-in-context $\nabla \vdash^{\text{NLC}} M : s$ that has a derivation is called a **typed expression**; and any equation-in-context $\nabla \vdash^{\text{NLC}} M \approx M' : s$ that has a derivation is called a **theorem**. The set of typed expressions and theorems of a NLC-theory Th is defined to be the least set of judgements containing the axioms of Th and being closed under the rules in Table 3.4

and Table 3.5. We refer to the set of derived expressions and the set of derived equations of type s and freshness context ∇ (for signature Sg) as $Exp_{Sg}(\nabla, s)$ and $Equ_{Sg}(\nabla, s)$, respectively. We formally indicate that a judgement J has a derivation in theory Th by writing $Th \rhd J$. For brevity, we usually omit this notation in our proofs. The NLC-theory, which is composed of the empty signature Sg (no constants and only one ground type) and the empty set Ax, is called the **pure $\beta\eta$-theory**.

Let's now consider the rule (AP) in more detail. Since F has type $s^{\overline{a}} \Rightarrow s'$ we require that $\overline{a}$ is fresh for the argument A in context ∇, which we encoded as $\nabla \vdash^{\text{NLC}} \overline{a} \# A : s$. Note that in defining the rules of NLC, we write $\nabla \vdash^{\text{NLC}} \overline{a} \# A : s$ in the hypothesis of rule (AP) as syntactic sugar for $\nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * A \approx M : s$ and the additional side condition $\overline{c} \# (\nabla, \overline{a}, A, s)$. Hence, the rule (AP) would formally be presented as a rule schema of the following form:

$$\frac{\nabla \vdash^{\text{NLC}} F : s^{\overline{a}} \Rightarrow s' \qquad \nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * A \approx A : s}{\nabla \vdash^{\text{NLC}} F\ A : s'}\ (\overline{c} \# (\nabla, \overline{a}, A, s))$$

The role that $\nabla \vdash^{\text{NLC}} \overline{a} \# M : s$, or more precisely the underlying equations (used to express freshness), plays in the definition of the typing system of NLC, leads to a crucial difference between NEL and NLC. *The type system rules have equations-in-context as hypotheses, and the equation rules have expressions-in-context as hypotheses. Thus theorems and typed expressions are mutually inductively defined.* Obviously this complicates our proofs, at least in comparison to NEL, and leads to some subtleties which we explain in due course.

(SP) $\dfrac{}{\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} \pi x : \pi \cdot s}$ (C) $\dfrac{}{\nabla \vdash^{\text{NLC}} c : s}$ ($c \in Fun_{Sg}$ and $c : s$ in Sg)

(ABS) $\dfrac{\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} M : s'}{\nabla \vdash^{\text{NLC}} \lambda^{\overline{a}} x : s.\, M : s^{\overline{a}} \Rightarrow s'}$ (AP) $\dfrac{\nabla \vdash^{\text{NLC}} F : s^{\overline{a}} \Rightarrow s' \qquad \nabla \vdash^{\text{NLC}} \overline{a} \# A : s}{\nabla \vdash^{\text{NLC}} F\ A : s'}$

Table 3.4: NLC Typing Rules

$$(\mathsf{AXIOM})\ \frac{\nabla \vdash^{\mathrm{NLC}} M : s \qquad \nabla \vdash^{\mathrm{NLC}} M' : s}{\nabla \vdash^{\mathrm{NLC}} M \approx M' : s}\ (\nabla \vdash^{\mathrm{NLC}} M \approx M' : s \in Ax)$$

$$(\mathsf{REF})\ \frac{\nabla \vdash^{\mathrm{NLC}} M : s}{\nabla \vdash^{\mathrm{NLC}} M \approx M : s} \qquad (\mathsf{SYM})\ \frac{\nabla \vdash^{\mathrm{NLC}} M \approx M' : s}{\nabla \vdash^{\mathrm{NLC}} M' \approx M : s}$$

$$(\mathsf{TRANS})\ \frac{\nabla \vdash^{\mathrm{NLC}} M \approx M' : s \qquad \nabla \vdash^{\mathrm{NLC}} M' \approx M'' : s}{\nabla \vdash^{\mathrm{NLC}} M \approx M'' : s}$$

$$(\mathsf{WEAK})\ \frac{\nabla \vdash^{\mathrm{NLC}} M \approx M' : s}{\nabla' \vdash^{\mathrm{NLC}} M \approx M' : s}\ (\nabla \leq \nabla')$$

$$(\mathsf{AE})\ \frac{\nabla^{\#\overline{a}} \vdash^{\mathrm{NLC}} M \approx M' : s}{\nabla \vdash^{\mathrm{NLC}} M \approx M' : s}\ (\overline{a} \# (\nabla, M, M'))$$

$$(\mathsf{SUSP})\ \frac{}{ds(\pi, \pi') \# x : s \vdash^{\mathrm{NLC}} \pi x \approx \pi' x : \pi \cdot s}$$

$$(\mathsf{B})\ \frac{\nabla, \overline{a} \# x : s \vdash^{\mathrm{NLC}} M : s' \qquad \nabla \vdash^{\mathrm{NLC}} \overline{a} \# N : s}{\nabla \vdash^{\mathrm{NLC}} (\lambda^{\overline{a}} x : s.\, M)\ N \approx M\{N/x\} : s'}$$

$$(\mathsf{E})\ \frac{\nabla \vdash^{\mathrm{NLC}} M : s^{\overline{a}} \Rightarrow s'}{\nabla \vdash^{\mathrm{NLC}} \lambda^{\overline{a}} x : s.\, (M\ x) \approx M : s^{\overline{a}} \Rightarrow s'}\ (x \# M)$$

$$(\mathsf{CL})\ \frac{\nabla, \overline{a} \# x : s \vdash^{\mathrm{NLC}} M \approx M' : s'}{\nabla \vdash^{\mathrm{NLC}} \lambda^{\overline{a}} x : s.\, M \approx \lambda^{\overline{a}} x : s.\, M' : s^{\overline{a}} \Rightarrow s'}$$

$$(\mathsf{CA})\ \frac{\nabla \vdash^{\mathrm{NLC}} \overline{a} \# A_i : s \quad \nabla \vdash^{\mathrm{NLC}} F_1 \approx F_2 : s^{\overline{a}} \Rightarrow s' \quad \nabla \vdash^{\mathrm{NLC}} A_1 \approx A_2 : s}{\nabla \vdash^{\mathrm{NLC}} F_1\ A_1 \approx F_2\ A_2 : s'}\ (\text{i=1,2})$$

Table 3.5: NLC Equation Rules

Lemma 3.2.1 *The rules (SUBST), (SUB) and (SSUB) (in Table 3.6) are derivable.*

Proof

(i) (SUBST): Suppose $\nabla, \overline{a} \# x : s \vdash^{\mathrm{NLC}} M_1 \approx M_2 : s'$, $\nabla \vdash^{\mathrm{NLC}} N_1 \approx N_2 : s'$, $\nabla \vdash^{\mathrm{NLC}} \overline{a} \# N_1 : s$ and $\nabla \vdash^{\mathrm{NLC}} \overline{a} \# N_2 : s$. We first apply (CL) on $\nabla, \overline{a} \#$

$x : s \vdash^{\text{NLC}} M_1 \approx M_2 : s'$ to obtain $\nabla \vdash^{\text{NLC}} \lambda^{\overline{a}} x : s.\, M_1 \approx \lambda^{\overline{a}} x : s.\, M_2 : s^{\overline{a}} \Rightarrow s'$ and then (CA) to obtain $\nabla \vdash^{\text{NLC}} (\lambda^{\overline{a}} x : s.\, M_1)\ N_1 \approx (\lambda^{\overline{a}} x : s.\, M_2)\ N_2 : s'$. The conclusion follows by application of rules (B) and (TRANS).

(ii) (SUB): Follows similarly to (SUBST) (using (WEAK)).

(iii) (SSUB): Suppose $\nabla \vdash^{\text{NLC}} M \approx M' : s'$, $\nabla' \vdash^{\text{NLC}} N_i \approx N_i' : s'$, $\nabla' \vdash^{\text{NLC}} \overline{a} \# N_i : s$ and $\nabla' \vdash^{\text{NLC}} \overline{a} \# N_i' : s$. We can now apply (CL) multiple times on $\nabla \vdash^{\text{NLC}} M \approx M' : s'$ to obtain $\emptyset \vdash^{\text{NLC}} \lambda^{\overline{a}_1} x_1 : s_1. ... \lambda^{\overline{a}_n} x_n : s_n.\, M \approx \lambda^{\overline{a}_1} x_1 : s_1. ... \lambda^{\overline{a}_n} x_n : s_n.\, M' : {s_i}^{\overline{a}_i} \Rightarrow s'$. Since we work with expressions and therefore can rename bound variables, we have that

$$\begin{aligned} \emptyset \vdash^{\text{NLC}} & \lambda^{\overline{a}_1} y_1 : s_1. ... \lambda^{\overline{a}_n} y_n : s_n.\, (x_i\ y_i) \bullet M \\ \approx & \lambda^{\overline{a}_1} y_1 : s_1. ... \lambda^{\overline{a}_n} y_n : s_n.\, (x_i\ y_i) \bullet M' : {s_i}^{\overline{a}_i} \Rightarrow s' \end{aligned}$$

for $y_i \# (N_i, N_i')$. Next, we apply the rules (WEAK) and (CA) to deduce the following

$$\begin{aligned} \nabla' \vdash^{\text{NLC}} & \lambda^{\overline{a}_1} y_1 : s_1. ... \lambda^{\overline{a}_n} y_n : s_n.\, ((x_i\ y_i) \bullet M)\ N_1 ... N_n \\ \approx & \lambda^{\overline{a}_1} y_1 : s_1. ... \lambda^{\overline{a}_n} y_n : s_n.\, ((x_i\ y_i) \bullet M')\ N_1' ... N_n' : s' \end{aligned}$$

We then apply rules (B) and (TRANS) to get $\nabla' \vdash^{\text{NLC}} ((x_i\ y_i) \bullet M)\{N_i/y_i\} \approx ((x_i\ y_i) \bullet M')\{N_i'/y_i\} : s'$. Given that $((x_i\ y_i) \bullet M)\{N_i/y_i\} = M\{N_i/x_i\}$ and $((x_i\ y_i) \bullet M')\{N_i'/y_i\} = M'\{N_i'/x_i\}$, we are done.

□

The "inversion lemma" is a technical lemma that describes how the subexpressions of a well-typed expression are typed. It is used implicitly in proofs by induction on the structure of expressions. In the case of NLC it follows immediately by definition and the fact that we have syntax-directed rules.

$$(\mathsf{SUBST})\quad \frac{\nabla, \overline{a} \# x : s \vdash^{\mathrm{NLC}} M_1 \approx M_2 : s' \qquad \nabla \vdash^{\mathrm{NLC}} \overline{a} \# N_i : s \qquad \nabla \vdash^{\mathrm{NLC}} N_1 \approx N_2 : s}{\nabla \vdash^{\mathrm{NLC}} M_1\{N_1/x\} \approx M_2\{N_2/x\} : s'}\ (i = 1, 2)$$

$$(\mathsf{SUB})\quad \frac{\nabla, \overline{a} \# x : s \vdash^{\mathrm{NLC}} M_1 \approx M_2 : s' \qquad \nabla' \vdash^{\mathrm{NLC}} \overline{a} \# N_i : s \qquad \nabla' \vdash^{\mathrm{NLC}} N_1 \approx N_2 : s}{\nabla, \nabla' \vdash^{\mathrm{NLC}} M_1\{N_1/x\} \approx M_2\{N_2/x\} : s'}$$

$$(\mathsf{SSUB})\quad \frac{\nabla \vdash^{\mathrm{NLC}} M \approx M' : s' \qquad \nabla' \vdash^{\mathrm{NLC}} N_i \approx N_i' : s_i \qquad \nabla' \vdash^{\mathrm{NLC}} \overline{a_i} \# N_i : s_i \qquad \nabla' \vdash^{\mathrm{NLC}} \overline{a_i} \# N_i' : s_i}{\nabla' \vdash^{\mathrm{NLC}} M\{\vec{N_i}/\vec{x_i}\} \approx M'\{\vec{N_i'}/\vec{x_i}\} : s'}$$

Table 3.6: NLC Derivable Rules

Lemma 3.2.2 (Typing Rule Inversion) *Suppose* $Th \rhd \nabla \vdash^{\mathrm{NLC}} M : s$

- *If* M *is a suspension* πx*, then* $\nabla = \nabla', \overline{a} \# x : s_1$ *for some context* ∇'*, type* s_1 *and set of finite names* $\overline{a}$*; thus* $s = \pi \cdot s_1$
- *If* M *is* $N\ N'$ *then* $\nabla \vdash^{\mathrm{NLC}} N : {s_1}^{\overline{a}} \Rightarrow s$ *and* $\nabla \vdash^{\mathrm{NLC}} \overline{a} \# N' : s_1$ *for some type* s_1 *and finite set of names* $\overline{a}$*.*
- *If* M *is* c*, then* $c : s$ *is the constant typing.*
- *If* M *is* $\lambda^{\overline{a}} x : s_1 . M'$*, then* s *is of the form* ${s_1}^{\overline{a}} \Rightarrow s_2$ *for some type* s_2 *and* $\nabla, \overline{a} \# x : s_1 \vdash^{\mathrm{NLC}} M' : s_2$*.*

We have yet another technical lemma which is crucial for proving some important facts about NLC. Lemma 3.2.3 is used in induction steps of lambda abstraction, or more precisely in the case when the binding variable also occurs in the environment. For an example induction we refer to the proof of Lemma 3.2.4, where we illustrate the particular care we take over dealing with lambda abstraction in proofs. A detailed explanation of the problem can also be found in [53] (page 169). For brevity, we will usually omit this particular case in proofs, and assume that binding variables do not occur in the context.

Lemma 3.2.3 (Variable Equivariance of Judgements) *All well-typed expressions and theorems (hence freshness assertions too) are equivariant under variable swapping. More precisely, for any $\mu \in Perm(Var)$ we have that*

$$Th \rhd \nabla \vdash^{\text{NLC}} M : s \Longrightarrow Th \rhd \mu \bullet \nabla \vdash^{\text{NLC}} \mu \bullet M : s$$
$$Th \rhd \nabla \vdash^{\text{NLC}} M \approx M' : s \Longrightarrow Th \rhd \mu \bullet \nabla \vdash^{\text{NLC}} \mu \bullet M \approx \mu \bullet M' : s$$

Proof The proof is by mutual induction.

Lemma 3.2.4 *$Th \rhd \nabla, \overline{a} \# x : s \vdash^{\text{NLC}} M : s'$ if and only if $Th \rhd \nabla, \pi \cdot \overline{a} \# x : \pi \cdot s \vdash^{\text{NLC}} M\{\pi^{-1}x/x\} : s'$ and similarly for equations.*

Proof Since permutations are isomorphisms we only need to prove one direction of the implication. We have to prove, by (mutual) induction over the rules in Table 3.4 and 3.5,

$$(\forall Th \rhd \nabla' \vdash^{\text{NLC}} [M]_\alpha : s') \quad [$$
$$(\forall\, \nabla, \overline{a}, \pi, x, s) \quad (\nabla' \equiv \nabla, \overline{a} \# x : s$$
$$\Longrightarrow Th \rhd \nabla, \pi \cdot \overline{a} \# x : \pi \cdot s \vdash^{\text{NLC}} [M\{\pi^{-1}x/x\}]_\alpha : s')) \quad]$$

$$(\forall Th \rhd \nabla' \vdash^{\text{NLC}} [M]_\alpha \approx [M']_\alpha : s') \quad [$$
$$(\forall\, \nabla, \overline{a}, \pi, x, s) \quad (\nabla' \equiv \nabla, \overline{a} \# x : s$$
$$\Longrightarrow Th \rhd \nabla, \pi \cdot \overline{a} \# x : \pi \cdot s \vdash^{\text{NLC}} [M\{\pi^{-1}x/x\}]_\alpha \approx [M'\{\pi^{-1}x/x\}]_\alpha : s')) \quad]$$

The case of rules (SP) and (C) follows immediately. We now concentrate on the interesting cases for rules (ABS) and (AP):

(ABS): To save any confusion over variable names we consider the specific instance of the rule

$$\frac{\nabla', \overline{b} \# y : t \vdash^{\text{NLC}} [N]_\alpha : t'}{\nabla' \vdash^{\text{NLC}} [\lambda^{\overline{b}}\boxed{y} : t.\, N]_\alpha : t^{\overline{b}} \Rightarrow t'}\,(\mathsf{ABS})$$

in which y is now in local scope. Consider the conclusion of the rule, and the local instantiation of $(\forall\, \nabla, \overline{a}, \pi, x, s)$ when $x \stackrel{\text{def}}{=} y$ (and the other names remain the same).

Thus we have $\nabla' \equiv \nabla, \overline{a} \# \boxed{y} : s$. For Induction Property Closure we have to prove

$$\nabla, \pi \cdot \overline{a} \# y : \pi \cdot s \vdash^{\text{NLC}} [(\lambda^{\overline{b}} y : t.\, N)\{\pi^{-1} y / y\}]_\alpha = [\lambda^{\overline{b}} y : t.\, N]_\alpha : t' \quad (\diamond)$$

We cannot immediately invert ABS since the binding $\boxed{y}$ occurs in ∇'. Choosing distinct y' we have that $[\lambda^{\overline{b}} y : t.\, N]_\alpha = [\lambda^{\overline{b}} y' : t.\, (y'\, y) \bullet N]_\alpha$; so we may now invert (ABS) to get

$$\nabla, \overline{a} \# y : s, \overline{b} \# y' : t \vdash^{\text{NLC}} [(y'\, y) \bullet N]_\alpha : t'$$

and hence by the variable equivariance of judgements, Lemma 3.2.3,

$$\nabla, \overline{a} \# y' : s, \overline{b} \# y : t \vdash^{\text{NLC}} [N]_\alpha : t'$$

Therefore, by induction with $(\forall\ \nabla, \overline{a}, \pi, x, s)$ locally instantiated as follows $\nabla, \overline{b} \# y : t, \overline{a}, \pi, y', s$, we have

$$\nabla, \pi \cdot \overline{a} \# y' : \pi \cdot s, \overline{b} \# y : t \vdash^{\text{NLC}} [N\{\pi^{-1} y' / y'\}]_\alpha = [N]_\alpha : t'$$

since $y' \notin var(N)$. Hence, by Lemma 3.2.3, we can obtain $(\diamond)$ from

$$\nabla, \pi \cdot \overline{a} \# y : \pi \cdot s, \overline{b} \# y' : t \vdash^{\text{NLC}} [(y'\, y) \bullet N]_\alpha : t'$$

followed by an instance of (ABS).

(AP): Contrary to the previous case, we do not explicitly mention α-equivalence classes. Suppose $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} P\ Q : s''$. From this, we can deduce that $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} P : s'^{\overline{b}} \Rightarrow s''$ and $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} \overline{b} \# Q : s'$. Then, by induction, we obtain that $\nabla, \pi \cdot \overline{a} \# x : \pi \cdot s \vdash^{\text{NLC}} P\{\pi^{-1} x / x\} : s'^{\overline{b}} \Rightarrow s''$. To apply an instance of rule (AP) we have to show that $\nabla, \pi \cdot \overline{a} \# x : \pi \cdot s \vdash^{\text{NLC}} \overline{b} \# Q\{\pi^{-1} x / x\} : s'$ holds.

We pick $\overline{c} \# (\nabla, \overline{a}, \overline{b}, Q, \pi, s, s')$, which clearly satisfies $\overline{c} \# ((\nabla, \pi \cdot \overline{a} \# x : \pi \cdot s), \overline{b}, Q\{\pi^{-1} x / x\}, s')$. So, by definition, we have to show that $\nabla^{\#\overline{c}}, \overline{c} \cup \pi \cdot \overline{a} \# x : \pi \cdot s \vdash^{\text{NLC}} (\overline{c}\, \overline{b}) * Q\{\pi^{-1} x / x\} \approx Q\{\pi^{-1} x / x\} : s'$ holds.

Given that $\overline{c} \# ((\nabla, \overline{a} \# x : s), \overline{b}, Q, s')$ and by assumption $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} \overline{b} \# Q : s'$, we have by definition $\nabla^{\#\overline{c}}, \overline{c} \cup \overline{a} \# x : s \vdash^{\text{NLC}} (\overline{c}\, \overline{b}) * Q \approx Q : s'$. Then, by

induction, we obtain $\nabla^{\#\overline{c}}, \pi\cdot\overline{c}\cup\pi\cdot\overline{a} \# x : \pi\cdot s \vdash^{\text{NLC}} ((\overline{c}\,\overline{b})*Q)\{\pi^{-1}x/x\} \approx Q\{\pi^{-1}x/x\} : s'$. Given that $\overline{c} \# \pi$ and by applying Proposition 3.1.14 we have completed this case.

The induction closure property for equations is obtained as follows: Suppose $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} M \approx M' : s'$. By an instance of rule (SP) we have that $\pi\cdot\overline{a} \# x : \pi\cdot s \vdash^{\text{NLC}} \pi^{-1}x : s$. We can now apply an instance of rule (REF) to obtain $\pi\cdot\overline{a} \# x : \pi\cdot s \vdash^{\text{NLC}} \pi^{-1}x \approx \pi^{-1}x : s$. Before we can apply the derived rule (SUB) to complete the argument, we have to show that $\pi\cdot\overline{a} \# x : \pi\cdot s \vdash^{\text{NLC}} \overline{a} \# \pi^{-1}x : \pi\cdot s$. We pick $\overline{c} \# (\pi, \overline{a}, s)$; thus $\overline{c} \# (\pi\cdot\overline{a} \# x : \pi\cdot s, \overline{a}, \pi^{-1}x, \pi\cdot s)$. Then, by definition, we have to show that $\overline{c}\cup\pi\cdot\overline{a} \# x : \pi\cdot s \vdash^{\text{NLC}} (\overline{c}\,\overline{a})\pi^{-1}x \approx \pi^{-1}x : \pi\cdot s$. We can now compute the following:

$$ds((\overline{c}\,\overline{a})\circ\pi^{-1}, \pi^{-1}) = supp(\pi\circ(\overline{c}\,\overline{a})\circ\pi^{-1}) = supp((\overline{c}\,\pi\cdot\overline{a})) = \overline{c}\cup\pi\cdot\overline{a}$$

and then apply an instance of rule (SUSP) to deduce the freshness assertion. □

In order to define the model-theoretic semantics in *FMSet* and the categorical semantics we require Proposition 3.2.5 and Proposition 3.2.6.

Proposition 3.2.5 ($*$ preserves Typed Expressions and Equalities) *Given a theory Th,*

$$Th \rhd \nabla \vdash^{\text{NLC}} M : s \text{ implies } Th \rhd \nabla \vdash^{\text{NLC}} \pi * M : \pi\cdot s$$
$$Th \rhd \nabla \vdash^{\text{NLC}} M \approx M' : s \text{ implies } Th \rhd \nabla \vdash^{\text{NLC}} \pi * M \approx \pi * M' : \pi\cdot s$$

Proof We prove by mutual induction over the rules in Figure 3.4 and Figure 3.5 the following statements:

$$(\forall Th \rhd \nabla \vdash^{\text{NLC}} M : s) \quad (Th \rhd \nabla \vdash^{\text{NLC}} \pi * M : \pi\cdot s)$$
$$(\forall Th \rhd \nabla \vdash^{\text{NLC}} M \approx M' : s) \quad (Th \rhd \nabla \vdash^{\text{NLC}} \pi * M \approx \pi * M' : \pi\cdot s)$$

We begin with the Induction Property Closure for typed expressions. Note that the cases for rule (SP) and (C) follow as in the case of NEL.

(ABS): Suppose that $\nabla \vdash^{\text{NLC}} \lambda^{\overline{a}}x : s.\, M : s^{\overline{a}} \Rightarrow s'$. We consider the case where x does not occur in the context. We can now directly deduce that $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} M : s'$ holds. So, by induction we obtain $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} \pi * M : \pi \cdot s'$. We then apply Lemma 3.2.4 to get $\nabla, \pi \cdot \overline{a} \# x : \pi \cdot s \vdash^{\text{NLC}} (\pi * M)\{\pi^{-1}x/x\} : \pi \cdot s$. Using an instance of rule (ABS), we obtain $\nabla \vdash^{\text{NLC}} \lambda^{\pi \cdot \overline{a}}x : \pi \cdot s.\, (\pi * M)\{\pi^{-1}x/x\} : (\pi \cdot s)^{\pi \cdot \overline{a}} \Rightarrow \pi \cdot s'$. Next, to complete this case, we apply Proposition 3.1.14 and Lemma 3.1.11.

(AP): Suppose that $\nabla \vdash^{\text{NLC}} F\ A : s'$. From this we can directly deduce $\nabla \vdash^{\text{NLC}} F : s^{\overline{a}} \Rightarrow s'$ and $\nabla \vdash^{\text{NLC}} \overline{a} \# A : s$. So, by definition we have that $\nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * A \approx A : s$ for some $\overline{c} \# (\nabla, \overline{a}, A, s)$. We now have to show that $\nabla \vdash^{\text{NLC}} \pi * (F\ A) : \pi \cdot s'$ holds, or equally $\nabla \vdash^{\text{NLC}} (\pi * F)\ (\pi * A) : \pi \cdot s'$ by definition of the object-level permutation action. By an instance of rule (AP), this can be deduced by demonstrating that $\nabla \vdash^{\text{NLC}} \pi * F : \pi \cdot s^{\pi \cdot \overline{a}} \Rightarrow \pi \cdot s'$ and $\nabla \vdash^{\text{NLC}} \pi \cdot \overline{a} \# \pi * A : \pi \cdot s$ hold. The former follows immediately by induction on $\nabla \vdash^{\text{NLC}} F : s^{\overline{a}} \Rightarrow s'$. For the latter, applying Lemma 3.2.4 and a couple of applications of Proposition 3.1.14, we can equally demonstrate that $\pi \cdot \nabla \vdash^{\text{NLC}} \pi \cdot \overline{a} \# \pi * A\{\pi^{-1}\_\} : \pi \cdot s$ holds. By Lemma 3.1.12 we have that $\pi * A\{\pi^{-1}\_\} = \pi \cdot A$. Then, by equivariance of freshness we obtain $\pi \cdot \overline{c} \# (\pi \cdot \nabla, \pi \cdot \overline{a}, \pi * A\{\pi^{-1}\_\}, \pi \cdot s)$. So, by definition it remains to be shown that

$$(\pi \cdot \nabla)^{\#\pi \cdot \overline{c}} \vdash^{\text{NLC}} (\pi \cdot \overline{c}\,\pi \cdot \overline{a}) * (\pi * A\{\pi^{-1}\_\}) \approx \pi * A\{\pi^{-1}\_\} : \pi \cdot s \quad (\diamond)$$

By induction on $\nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * A \approx A : s$ we obtain that $\nabla^{\#\overline{c}} \vdash^{\text{NLC}} \pi * (\overline{c}\,\overline{a}) * A \approx \pi * A : \pi \cdot s$. Given that $\pi * -$ is a permutation action we have that $\nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\pi \cdot \overline{c}\,\pi \cdot \overline{a}) * \pi * A \approx \pi * A : \pi \cdot s$. By applying Lemma 3.2.4 and a couple of applications of Proposition 3.1.14, we obtain $(\diamond)$ and the argument is complete.

The induction closure property for equations is obtained as follows: Suppose that $\nabla \vdash^{\text{NLC}} M \approx M' : s$. By (SUSP) we have that $\emptyset \# x : s \vdash^{\text{NLC}} \pi x \approx \pi x : \pi \cdot s$. We can now apply (SUBST) to directly obtain $\nabla \vdash^{\text{NLC}} \pi * M \approx \pi * M' : \pi \cdot s$. This completes the argument. □

Corollary 3.2.6 (Name Equivariance of Judgements) *Given a theory Th,*

$$Th \rhd \nabla \vdash^{\text{NLC}} M : s \text{ implies } Th \rhd \pi \cdot \nabla \vdash^{\text{NLC}} \pi \cdot M : \pi \cdot s$$
$$Th \rhd \nabla \vdash^{\text{NLC}} M \approx M' : s \text{ implies } Th \rhd \pi \cdot \nabla \vdash^{\text{NLC}} \pi \cdot M \approx \pi \cdot M' : \pi \cdot s$$

Proof Suppose $Th \rhd \nabla \vdash^{\text{NLC}} M : s$. We then apply Proposition 3.2.5 to obtain $Th \rhd \nabla \vdash^{\text{NLC}} \pi * M : \pi \cdot s$. Note that we clearly have $\pi * M = (\pi * M)\{\pi^{-1}\_\}\{\pi\_\}$. By Lemma 3.2.4 we obtain $Th \rhd \pi \cdot \nabla \vdash^{\text{NLC}} (\pi * M)\{\pi^{-1}\_\} : \pi \cdot s$ and by Proposition 3.1.14 we have $\pi \cdot \nabla \vdash^{\text{NLC}} \pi \cdot M : \pi \cdot s$. The case for theorems is proven similarly. □

Hence, we have that the set of derived terms and the set of derived equations are equivariant subsets of $Ctxt_{Sg} \times Term_{Sg}/\!\sim_\alpha \times Type_{Sg}$ and $Ctxt_{Sg} \times Term_{Sg}/\!\sim_\alpha \times Term_{Sg}/\!\sim_\alpha \times Type_{Sg}$, respectively. Hence, they are nominal sets. Further, we can deduce that $Exp_{Sg}(\nabla, s)$ and $Equ_{Sg}(\nabla, s)$ are FM-sets (with support $supp(\nabla) \cup supp(s)$). Analogous to Clouston [17, 16], we use the following definition for derivability of $\nabla \vdash^{\text{NLC}} \overline{a} \# M : s$:

Definition 3.2.7 *Let Th be an* NLC*-theory and* $\overline{a} \# s$

$$Th \rhd \nabla \vdash^{\text{NLC}} \overline{a} \# M : s \quad \stackrel{\text{def}}{=} \quad (И\overline{c}) \quad Th \rhd \nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{a}\,\overline{c}) * M \approx M : s$$

where $\nabla^{\#\overline{c}} \stackrel{\text{def}}{=} \overline{a}_1 \cup \overline{c} \# x_1 : s_1, ..., \overline{a}_n \cup \overline{c} \# x_n : s_n$ *and the transposition,* $\overline{a} \in \mathbb{A}^n$ *is sugar for a tuple of the atoms in the set* $\overline{a}$*. Note that* $\overline{c} \in \mathbb{A}^n$ *is* any/some fresh *tuple of the same size such that* $\overline{c} \# (\nabla, \overline{a}, M)$*. If* $Th \rhd \nabla \vdash^{\text{NLC}} \overline{a} \# M : s$ *then we may legitimately call the judgement a* theorem*, but we will usually call it a* ***freshness assertion****.*

What remains to be shown is that the И-quantifier is correctly applied in the previous definition, i.e. we have to demonstrate that Theorem 2.2.7 (Some/Any Theorem) can be applied. This is done in the following Lemma.

Lemma 3.2.8 *$X \stackrel{\text{def}}{=} Ctxt_{Sg} \times \mathcal{P}_{fin}(\mathbb{A}) \times Term_{Sg}/\!\sim_\alpha \times Type_{Sg}$ is a nominal set and the relation*

$$R \stackrel{\text{def}}{=} \{(\overline{c}, (\nabla, \overline{a}, M, s)) \mid Th \rhd \nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{a}\,\overline{c}) * M \approx M : s\} \subseteq \mathcal{P}_{fin}(\mathbb{A}) \times X$$

is an equivariant subset of $\mathcal{P}_{fin}(\mathbb{A}) \times X$.

Proof Given that all the sets involved in the definition of X are nominal sets, it follows immediately that X is a nominal set. Suppose that $Th \rhd \nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{a}\,\overline{c}) * M \approx M : s$ and $\pi \in Perm(\mathbb{A})$. We then obtain, by Lemma 3.2.6, that $Th \rhd (\pi \cdot \nabla)^{\#\pi\cdot\overline{c}} \vdash^{\text{NLC}} \pi \cdot (\overline{a}\,\overline{c}) * M \approx \pi \cdot M : \pi \cdot s$ holds. Further, we can deduce the following

$$\begin{aligned}
\pi \cdot (\overline{a}\,\overline{c}) * M &= \pi * ((\overline{a}\,\overline{c}) * M)\{\pi^{-1}_\} && \text{(Lemma 3.1.12)}\\
&= \pi * (\overline{a}\,\overline{c}) * M\{\pi^{-1}_\} && \text{(Lemma 3.1.14)}\\
&= (\pi(\overline{a})\,\pi(\overline{c})) * \pi * M\{\pi^{-1}_\}\\
&= (\pi(\overline{a})\,\pi(\overline{c})) * \pi \cdot M && \text{(Lemma 3.1.12)}
\end{aligned}$$

So, we have $Th \rhd (\pi \cdot \nabla)^{\#\pi\cdot\overline{c}} \vdash^{\text{NLC}} (\pi(\overline{a})\,\pi(\overline{c})) * \pi \cdot M \approx \pi \cdot M : \pi \cdot s$ and therefore $(\pi \cdot \overline{c}, (\pi \cdot \nabla, \pi \cdot \overline{a}, \pi \cdot M, \pi \cdot s)) = \pi \cdot (\overline{c}, (\nabla, \overline{a}, M, s)) \in R$. □

Remark 3.2.9 *Considering Definition 3.2.7, we can now express $\nabla \vdash^{\text{NLC}} \overline{a} \# A : s$, which is originally defined as syntactic sugar for $\nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * M \approx M : s$ with the side condition $\overline{c} \# (\nabla, \overline{a}, A, s)$, using a more nominal vernacular, as $(\mathsf{N}\,\overline{c})\ \nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * M \approx M : s$. Hence, $\boxed{\nabla \vdash^{\text{NLC}} \overline{a} \# A : s}$ closely coincides with the characterisation of freshness in FMSet and the "un-sugared" rule (AP) can now be read as*

$$\frac{\nabla \vdash^{\text{NLC}} F : s^{\overline{a}} \Rightarrow s' \qquad \boxed{(\mathsf{N}\,\overline{c}) \quad \nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * A \approx A : s}}{\nabla \vdash^{\text{NLC}} F\ A : s'}$$

It follows immediately that the previously obtained equivariance results carry over to freshness assertions:

Corollary 3.2.10 *Let $\pi \in Perm(\mathbb{A})$. We have the following two properties:*

(i) *if* $Th \rhd \nabla \vdash^{\text{NLC}} \overline{a} \# M : s$, *then* $Th \rhd \nabla \vdash^{\text{NLC}} \pi \cdot \overline{a} \# \pi * M : \pi \cdot s$

(ii) *if* $Th \rhd \nabla \vdash^{\text{NLC}} \overline{a} \# M : s$, *then* $Th \rhd \pi \cdot \nabla \vdash^{\text{NLC}} \pi \cdot \overline{a} \# \pi \cdot M : \pi \cdot s$

We now present two obvious non-properties with respect to freshness assertions:

Lemma 3.2.11 *For any well typed expression* $\nabla \vdash^{\text{NLC}} M : s$, *the following two implications do not hold in general:*

(i) $Th \rhd \nabla \vdash^{\text{NLC}} \overline{a} \# M : s \Longrightarrow \overline{a} \# (\nabla, M)$

(ii) $\overline{a} \# (\nabla, M) \Longrightarrow Th \rhd \nabla \vdash^{\text{NLC}} \overline{a} \# M : s$

Proof For the first implication, it follows immediately that $b \# x : s \vdash^{\text{NLC}} a \# (a\,b)x : s$ holds, but $a \# (a\,b)x$ clearly does not hold.

For the second implication we reason semantically by using the model-theoretic semantics in *FMSet* for NLC (in Section 3.4). We pick a name $a \in \mathbb{A}$ and choose a signature Sg with a single base type $Name$ (with empty support); thus $a \# Name$ and $\emptyset \# x : Name$ is a well defined context. Then, by an instance of rule (SP) we obtain that $\emptyset \# x : Name \vdash^{\text{NLC}} x : Name$ holds. Further, we have that $a \# ((\emptyset \# x : Name), x)$. We can now define an NLC-structure for Sg with $[\![Name]\!] \stackrel{\text{def}}{=} \mathbb{A}$ and provide an environment $\eta(x) \stackrel{\text{def}}{=} a$. It follows immediately that $\eta \models (\emptyset \# x : Name)$. Thus, we have that $[\![x]\!]_\eta \stackrel{\text{def}}{=} \eta(x) \stackrel{\text{def}}{=} a$ and therefore $a \# [\![x]\!]_\eta$ does not hold. Using the contrapositive of Corollary 3.4.15 (Soundness lemma for freshness assertions), we can deduce that $\emptyset \# x : Name \vdash^{\text{NLC}} a \# x : Name$ cannot be derived. □

We continue by proving various standard properties of the typing system. Note that these properties are usually proved by induction on typing derivations, but can also be proved by induction on the structure of expressions. Given that the type and equation system of NLC are defined using mutual induction, we use inductive proofs over the structure of expressions whenever possible to avoid proof overhead.

Lemma 3.2.12 (Uniqueness of Types) *if* $Th \rhd \nabla \vdash^{\text{NLC}} M : s$ *and* $Th \rhd \nabla \vdash^{\text{NLC}} M : s'$*, then* $s = s'$*.*

Proof Proof by induction on the structure of expression M:

$$\forall(\nabla, s, s') \quad [Th \rhd \nabla \vdash^{\text{NLC}} M : s \ \wedge \ Th \rhd \nabla \vdash^{\text{NLC}} M : s' \Longrightarrow s = s']$$

SUSP: (M is πx). Suppose that $\nabla \vdash^{\text{NLC}} \pi x : t$ and $\nabla \vdash^{\text{NLC}} \pi x : t'$. From this we can deduce that $\nabla \equiv \nabla', \overline{a} \# x : s$, $t \equiv \pi \cdot s$ and $t' \equiv \pi \cdot s$. Hence, $t = t'$.

CONST: Follows immediately.

LAM-ABS: (M is $\lambda^{\overline{a}} x : s.\, N$) Suppose that $\nabla \vdash^{\text{NLC}} \lambda^{\overline{a}} x : s.\, N : \sigma$ and $Th \rhd \nabla \vdash^{\text{NLC}} \lambda^{\overline{a}} x : s.\, N : \tau$. By type inversion we obtain $\sigma = s^{\overline{a}} \Rightarrow t$ and $\tau = s^{\overline{a}} \Rightarrow t'$ for some types t and t'. Then, by rule (ABS), and the induction hypothesis, we can deduce that $t = t'$ and therefore $\sigma = \tau$.

APP: (M is $F\ A$). Suppose that $\nabla \vdash^{\text{NLC}} F\ A : t$ and $\nabla \vdash^{\text{NLC}} F\ A : t'$. From this we can deduce by type inversion that $\nabla \vdash^{\text{NLC}} F : {s_1}^{\overline{a}_1} \Rightarrow t$ and $\nabla \vdash^{\text{NLC}} F : {s_2}^{\overline{a}_2} \Rightarrow t'$. Then, by the induction hypothesis on F, we obtain that ${s_1}^{\overline{a}_1} \Rightarrow t = {s_2}^{\overline{a}_2} \Rightarrow t'$, and therefore $t = t'$ holds. □

This means that we have a partial typing function from $Ctxt_{Sg} \times Term_{Sg}/\!\sim_\alpha$ to $Type_{Sg}$, which is also equivariant by Proposition 3.2.6. We can now deduce the following useful property: if $Th \rhd \nabla \vdash^{\text{NLC}} M : s$ and $\overline{a} \# (\nabla, M)$, then $\overline{a} \# s$ holds. This property allows us to simplify various side conditions for type and equation rules.

Lemma 3.2.13 (Context Permutation Lemma) *If* $\nabla \vdash^{\text{NLC}} M : s$ *and* ∇' *is a permutation of* ∇*, then* $\nabla' \vdash^{\text{NLC}} M : s$*.*

Proof Proof by induction on the structure of expression M.

Lemma 3.2.14 (Weakening) *If* $Th \rhd \nabla \vdash^{\text{NLC}} M : s$ *and* $\nabla \leq \nabla'$*, then* $Th \rhd \nabla' \vdash^{\text{NLC}} M : s$*.*

Proof Proof by induction on the structure of M

$$\forall(\nabla, \nabla', s) \quad [Th \rhd \nabla \vdash^{\text{NLC}} M : s \ \wedge \ \nabla \leq \nabla' \Longrightarrow Th \rhd \nabla' \vdash^{\text{NLC}} M : s]$$

SUSP: (M is πx). Suppose that $\nabla \vdash^{\text{NLC}} \pi x : t$ and $\nabla \leq \nabla'$. From this we can deduce that $\nabla \equiv \nabla_1, \overline{a} \# x : s$ and $t \equiv \pi \cdot s$. Given that $\nabla \leq \nabla'$ we have that $x \in dom(\nabla')$ and $\nabla'(x) = (\overline{b}, s)$ with $\overline{a} \subseteq \overline{b}$. By an instance of rule (SP) we obtain that $\nabla' \vdash^{\text{NLC}} \pi x : \pi \cdot s$, and therefore $\nabla' \vdash^{\text{NLC}} \pi x : t$.

CONST: Follows immediately.

LAM-ABS: (M is $\lambda^{\overline{a}} x : s.\, N$) Suppose that $\nabla \vdash^{\text{NLC}} \lambda^{\overline{a}} x : s.\, N : s^{\overline{a}} \Rightarrow s'$ and $\nabla \leq \nabla'$. We consider the case where x does not occur in the both contexts. From this we can deduce that $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} N : s'$ holds. We then apply the induction hypothesis to obtain $\nabla', \overline{a} \# x : s \vdash^{\text{NLC}} N : s'$, and by an instance of rule (ABS), we have $\nabla' \vdash^{\text{NLC}} \lambda^{\overline{a}} x : s.\, N : s^{\overline{a}} \Rightarrow s'$

APP: (M is $F\ A$) Suppose that $\nabla \vdash^{\text{NLC}} F\ A : s'$ and $\nabla \leq \nabla'$. From this we can deduce that $\nabla \vdash^{\text{NLC}} F : s^{\overline{a}} \Rightarrow s'$ and $\nabla \vdash^{\text{NLC}} \overline{a} \# A : s$. By induction we obtain that $\nabla' \vdash^{\text{NLC}} F : s^{\overline{a}} \Rightarrow s'$. To apply an instance of rule (AP) and to complete the argument we need to show that $\nabla' \vdash^{\text{NLC}} \overline{a} \# N : s$ holds.

Pick $\overline{c} \# (\nabla, \nabla', \overline{a}, N)$. We need to show that $(\nabla')^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * A \approx A : s$. Given that $\nabla \leq \nabla'$ and $\overline{c} \# (\nabla, \nabla')$ we have that $\nabla^{\#\overline{c}} \leq \nabla'^{\#\overline{c}}$. Further, given that $\overline{c} \# (\nabla, A, \overline{a})$ we get $\nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * A \approx A : s$, and by using (WEAK) we are done. □

Lemma 3.2.15 (Atom Elimination for Typing) *If* $Th \rhd \nabla^{\#\overline{a}} \vdash^{\text{NLC}} M : s$ *and* $\overline{a} \# (\nabla, M)$, *then* $Th \rhd \nabla \vdash^{\text{NLC}} M : s$.

Proof Proof by induction on the structure of expression M that

$$\forall(\nabla, \overline{a}, s) \quad [Th \rhd \nabla^{\#\overline{a}} \vdash^{\text{NLC}} M : s \ \wedge \ \overline{a} \# (\nabla, M) \Longrightarrow Th \rhd \nabla \vdash^{\text{NLC}} M : s]$$

SUSP: (M is πx) Suppose that $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} \pi x : t$ and $\overline{a} \# (\nabla, \pi)$. From this we can deduce that $\nabla^{\#\overline{a}} = (\nabla', \overline{b} \# x : s)^{\#\overline{a}} = \nabla'^{\#\overline{a}}, \overline{a} \cup \overline{b} \# x : s$ and $t = \pi \cdot s$. We can then apply an instance of rule (SP) to obtain $\nabla', \overline{b} \# x : s \vdash^{\text{NLC}} \pi x : \pi \cdot s$.

CONST: Follows immediately.

LAM-ABS: (M is $\lambda^{\overline{b}}x : s.\, N$) Suppose that $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} \lambda^{\overline{b}}x : s.\, N : s^{\overline{b}} \Rightarrow s'$ and $\overline{a} \# (\nabla, M)$. So, by definition we have that $\overline{a} \# (\overline{b}, s, N)$. We consider the case where x does not occur in the context. We can then deduce that $\nabla^{\#\overline{a}}, \overline{b} \# x : s \vdash^{\text{NLC}} N : s'$. Next, we apply Lemma 3.2.14 to obtain $\nabla^{\#\overline{a}}, \overline{a} \cup \overline{b} \# x : s \vdash^{\text{NLC}} N : s'$. Given that $\nabla^{\#\overline{a}}, \overline{a} \cup \overline{b} \# x : s = (\nabla, \overline{b} \# x : s)^{\#\overline{a}}$ and $\overline{a} \# (\nabla, \overline{b}, s, N)$ we obtain, by induction, that $\nabla, \overline{b} \# x : s \vdash^{\text{NLC}} N : s'$. We complete this case by applying rule (ABS).

APP: (M is $P\ Q$) Suppose that $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} P\ Q : s'$ and $\overline{a} \# (\nabla, P\ Q)$. So, by definition of the meta-level permutation action, we have that $\overline{a} \# (P, Q)$. Further, we can deduce that $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} P : s^{\overline{b}} \Rightarrow s'$ and $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} \overline{b} \# Q : s$. Then, by induction on the type derivation we obtain that $\nabla \vdash^{\text{NLC}} P : s^{\overline{b}} \Rightarrow s'$. To apply an instance of rule (AP) and complete the argument, we need to show that $\nabla \vdash^{\text{NLC}} \overline{b} \# Q : s$ holds. We pick $\overline{c} \# (\nabla, \overline{b}, Q, \overline{a})$ and demonstrate that $\nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{b}) * Q \approx Q : s$.

From $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} \overline{b} \# Q : s$ and $\overline{c} \# (\overline{a}, \nabla, \overline{b}, Q)$ we obtain by definition that $(\nabla^{\#\overline{a}})^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{b}) * Q \approx Q : s$. Given that $(\nabla^{\#\overline{a}})^{\#\overline{c}} = (\nabla^{\#\overline{c}})^{\#\overline{a}}$, we have that $(\nabla^{\#\overline{c}})^{\#\overline{a}} \vdash^{\text{NLC}} (\overline{c}\,\overline{b}) * Q \approx Q : s$. Note that from $\nabla \vdash^{\text{NLC}} P : s^{\overline{b}} \Rightarrow s'$ and $\overline{a} \# (\nabla, P)$ we can deduce that $\overline{a} \# s^{\overline{b}} \Rightarrow s'$; thus $\overline{a} \# \overline{b}$. We have that $\overline{a} \# (\nabla^{\#\overline{c}}, (\overline{c}\,\overline{b}) * Q, Q)$ and therefore an instance of rule (AE) can be applied to complete this argument. □

Lemma 3.2.16 *if* $Th \rhd \nabla \vdash^{\text{NLC}} M \approx M' : s$, *then* $Th \rhd \nabla \vdash^{\text{NLC}} M : s$ *and* $Th \rhd \nabla \vdash^{\text{NLC}} M : s$.

Proof Follows immediately by mutual induction on $Th \rhd \nabla \vdash^{\text{NLC}} M \approx M' : s$.

Lemma 3.2.17 *if* $Th \rhd \nabla \vdash^{\text{NLC}} M : s$, *then* $fv(M) \subseteq dom(\nabla)$.

Proof Proof by induction on the structure of M:

$$(\forall M)\ [(\forall\ \nabla, s)\ (Th \rhd \nabla \vdash^{\text{NLC}} M : s \Longrightarrow fv(M) \subseteq \nabla)\ ]$$

SUSP: (M is πx). Suppose that $\nabla \vdash^{\text{NLC}} \pi x : t$. From this we can deduce that $\nabla = \nabla', \overline{a} \# x : s$ and $t = \pi \cdot s$. Hence, we immediately get that $fv(\pi x) = \{x\} \subseteq dom(\nabla)$.

CONST: Follows immediately.

LAM-ABS: (M is $\lambda^{\overline{a}} x : s.\, N$) Suppose that $\nabla \vdash^{\text{NLC}} \lambda^{\overline{a}} x : s.\, N : s^{\overline{a}} \Rightarrow s'$. We consider the case where x does not occur in the context. By an instance of the rule (ABS) we obtain $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} N : s'$. Next, we apply the induction hypothesis to get $fv(N) \subseteq dom(\nabla, \overline{a} \# x : s)$. So, we have that $fv(N) \setminus \{x\} \subseteq dom(\nabla)$ and therefore $fv(\lambda^{\overline{a}} x : s.\, N) \stackrel{\text{def}}{=} fv(N) \setminus \{x\} \subseteq dom(\nabla)$.

APP:(M is $P\ Q$): Suppose $\nabla \vdash^{\text{NLC}} P\ Q : s'$. From this we can deduce that $\nabla \vdash^{\text{NLC}} P : s^{\overline{a}} \Rightarrow s'$ and $\nabla \vdash^{\text{NLC}} \overline{a} \# Q : s$, as well as $\nabla \vdash^{\text{NLC}} Q : s$ by Lemma 3.2.16. By induction we have $fv(P) \subseteq dom(\nabla)$ and $fv(Q) \subseteq dom(\nabla)$. Hence, we have $fv(P\ Q) \subseteq dom(\nabla)$. □

Lemma 3.2.18 *If* $Th \rhd \nabla \vdash^{\text{NLC}} \overline{a} \# M : s$ *and* $Th \rhd \nabla \vdash^{\text{NLC}} M \approx M' : s$, *then* $Th \rhd \nabla \vdash^{\text{NLC}} \overline{a} \# M' : s$.

Proof Suppose that $\nabla \vdash^{\text{NLC}} \overline{a} \# M : s$ ($\diamond 1$) and $\nabla \vdash^{\text{NLC}} M \approx M' : s$. We pick $\overline{c} \# (\nabla, \overline{a}, M', M, s)$ and demonstrate that $\nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * M' \approx M' : s$ holds: We apply an instance of rule (WEAK) on $\nabla \vdash^{\text{NLC}} M \approx M' : s$ to obtain $\nabla^{\#\overline{c}} \vdash^{\text{NLC}} M \approx M' : s$. Then, by Proposition 3.2.5, we get $\nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * M \approx (\overline{c}\,\overline{a}) * M' : (\overline{c}\,\overline{a}) \cdot s$ ($\diamond 2$). Note that $(\overline{c}\,\overline{a}) \cdot s = s$. Further, by definition, we have that $\nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * M \approx M : s$ ($\diamond 3$). The equation then follows directly from multiple applications of rule (TRANS) for ($\diamond 1$), ($\diamond 2$) and ($\diamond 3$). □

Lemma 3.2.19 *if* $Th \rhd \nabla^{\#\overline{a}} \vdash^{\text{NLC}} M : s$ *and* $\overline{a} \# (\nabla, M, s)$, *then* $Th \rhd \nabla^{\#\overline{a}} \vdash^{\text{NLC}} \overline{a} \# M : s$.

Proof Proof by induction on the structure of M:

$$[(\forall\, \nabla, \overline{a}, s)\ (Th \rhd \nabla^{\#\overline{a}} \vdash^{\text{NLC}} M : s\ \wedge\ \overline{a} \# (\nabla, M, s) \Longrightarrow Th \rhd \nabla^{\#\overline{a}} \vdash^{\text{NLC}} \overline{a} \# M : s\ ]$$

SUSP: (M is πx). Suppose that $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} \pi x : t$ and $\overline{a} \# (\nabla, \pi, t)$. From this we can deduce that $\nabla^{\#\overline{a}} = \nabla'^{\#\overline{a}}, \overline{a} \cup \overline{b} \# x : s$ and $t = \pi \cdot s$; so $\overline{a} \# (\overline{b}, s)$. Next, we pick $\overline{c} \# (\pi, \overline{a}, \nabla)$ and demonstrate that $\nabla'^{\#\overline{a}\cup\overline{c}}, \overline{c} \cup \overline{a} \cup \overline{b} \# x : s \vdash^{\text{NLC}} (\overline{c}\,\overline{a})\pi x \approx \pi x : \pi \cdot s$.

By applying an instance of rule (SUSP) we obtain that $ds((\overline{a}\,\overline{c})\pi, \pi) \# x : s \vdash^{\text{NLC}} (\overline{a}\,\overline{c})\pi x \approx \pi x : \pi \cdot s$. Note that $ds((\overline{a}\,\overline{c})\pi, \pi) = supp(\pi^{-1}(\overline{a}\,\overline{c})\pi) = \overline{a} \cup \overline{c}$. By an instance of rule (WEAK) we obtain $\nabla'^{\#\overline{a}}, \overline{c} \cup \overline{a} \cup \overline{b} \# x : s \vdash^{\text{NLC}} (\overline{a}\,\overline{c})\pi x \approx \pi x : \pi \cdot s$.

CONST: Follows immediately.

LAM-ABS: (M is $\lambda^{\overline{b}} x : s.\, N$): Suppose that $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} \lambda^{\overline{b}} x : s.\, N : s^{\overline{b}} \Rightarrow s'$ and $\overline{a} \# (\nabla, M, s^{\overline{b}} \Rightarrow s')$. From this we can directly deduce that $\overline{a} \# (\overline{b}, s, s', N)$. We now have to show that $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} \overline{a} \# \lambda^{\overline{b}} x : s.\, N : s^{\overline{b}} \Rightarrow s'$. We pick $\overline{c} \# (\nabla, \overline{a}, \overline{b}, s, s', N)$ and demonstrate that $\nabla^{\#\overline{a}\cup\overline{c}} \vdash^{\text{NLC}} (\overline{a}\,\overline{c}) * \lambda^{\overline{b}} x : s.\, N \approx \lambda^{\overline{b}} x : s.\, N : s^{\overline{b}} \Rightarrow s'$. Note that by definition of the object-level permutation action and an instance of rule (CL) we can equally show that $\nabla^{\#\overline{a}\cup\overline{c}}, \overline{b} \# x : s \vdash^{\text{NLC}} (\overline{a}\,\overline{c}) * N\{(\overline{a}\,\overline{c})x/x\} \approx N : s'$ holds.

Due to the fact that $\overline{a}, \overline{c} \# N$, the left-hand and right-hand side of the equation we wish to prove only differ in their suspended variables. More precisely, we have that for any $y \in dom(\nabla^{\#\overline{a},\overline{c}})$ the suspensions for y can be shown equal by using an instance of rule (SUSP). In the case of x this is not necessary, because the suspensions are identical (using $\overline{a}, \overline{c} \# N$). Hence, by applying instances of the rules (SUSP), (REF) and the various congruence rules this judgement can directly be deduced.

APP:(M is $P\ Q$) Suppose $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} P\ Q : s$ and $\overline{a} \# (\nabla, P\ Q, s)$. From this we can deduce that $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} P : s^{\overline{b}} \Rightarrow s'$ and $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} \overline{b} \# Q : s$. Given that $\overline{a} \# (\nabla, P)$ and by Lemma 3.2.15 we obtain that $\nabla \vdash^{\text{NLC}} P : s^{\overline{b}} \Rightarrow s'$. Further, we have that $\overline{a} \# s^{\overline{b}} \Rightarrow s'$, and therefore $\overline{a} \# (s, s', \overline{b})$. By induction, we obtain that $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} \overline{a} \# P : s^{\overline{b}} \Rightarrow s'$. To apply an instance of rule (AP), we have to show that $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} \overline{a} \# P\ Q : s'$, i.e. we pick $\overline{c} \# (\nabla, \overline{a}, P, Q, \overline{b})$ and show that $\nabla^{\#\overline{a},\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * P\ Q \approx P\ Q : s'$ holds.

From $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} \overline{a} \# P : s^{\overline{b}} \Rightarrow s'$ we obtain by definition that $\nabla^{\#\overline{a},\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * P \approx P : s^{\overline{b}} \Rightarrow s'$ ($\diamond 1$). From $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} \overline{b} \# Q : s$ we can deduce by Lemma 3.2.16 that

$\nabla^{\#\overline{a}} \vdash^{\text{NLC}} Q : s$ and by induction that $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} \overline{a} \# Q : s$. Hence, by definition, we have that $\nabla^{\#\overline{a},\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * Q \approx Q : s$ ($\diamond 2$). We then apply an instance of rule (WEAK), which also holds for freshness assertions, on $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} \overline{b} \# Q : s$ to obtain $\nabla^{\#\overline{a},\overline{c}} \vdash^{\text{NLC}} \overline{b} \# Q : s$ ($\diamond 3$). Next, we apply Corollary 3.2.10 on ($\diamond 3$) to obtain $\nabla^{\#\overline{a},\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) \cdot \overline{b} \# (\overline{c}\,\overline{a}) * Q : (\overline{c}\,\overline{a}) \cdot s$. Given that $\overline{a}, \overline{c} \# (\overline{b}, s)$, we have $\nabla^{\#\overline{a},\overline{c}} \vdash^{\text{NLC}} \overline{b} \# (\overline{c}\,\overline{a}) * Q : s$ ($\diamond 4$). Using ($\diamond 1$), ($\diamond 2$), ($\diamond 3$) and ($\diamond 4$), we can apply rule CA. The result then follows immediately by the definition of the meta-level permutation action. □

Lemma 3.2.20 (Preservation of Types under Substitution) *If* $Th \rhd \nabla, \overline{a} \# x : s \vdash^{\text{NLC}} M : s'$ *and* $Th \rhd \nabla' \vdash^{\text{NLC}} \overline{a} \# N : s$ *with* ∇, ∇' *well defined, then* $Th \rhd \nabla, \nabla' \vdash^{\text{NLC}} M\{N/x\} : s'$.

Proof Proof by induction on the structure of expression M

SUSP: (M is πx). Suppose that $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} \pi x : \pi \cdot s$ and $\nabla' \vdash^{\text{NLC}} \overline{a} \# N : s$. By Lemma 3.2.16 we have that $\nabla' \vdash^{\text{NLC}} N : s$ holds and by Proposition 3.2.5 $\nabla' \vdash^{\text{NLC}} \pi * N : \pi \cdot s$. Then, by Lemma 3.2.14 and the definition of capture avoiding substitution we obtain that $\nabla, \nabla' \vdash^{\text{NLC}} (\pi x)\{N/x\} : \pi \cdot s$.

CONST: Follows immediately.

LAM-ABS: (M is $\lambda^{\overline{b}} y : s'.\, P$) Suppose that $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} \lambda^{\overline{b}} y : s.\, P : {s'}^{\overline{b}} \Rightarrow s''$ and $\nabla' \vdash^{\text{NLC}} \overline{a} \# N : s$. We consider the case where y does not occur in both contexts. We can now directly deduce that $\nabla, \overline{a} \# x : s, \overline{b} \# y : s \vdash^{\text{NLC}} P : s''$. Next, we apply the induction hypothesis to obtain $\nabla, \nabla', \overline{b} \# y : s' \vdash^{\text{NLC}} P\{N/x\} : s''$. By an instance of rule (ABS) and the definition of capture avoiding substitution we obtain $\nabla, \nabla' \vdash^{\text{NLC}} (\lambda^{\overline{b}} y : s'.\, P)\{N/x\} : {s'}^{\overline{b}} \Rightarrow s''$.

APP: (M is $F\ A$) Suppose that $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} P\ Q : s'$. From this we can deduce that $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} P : (s'')^{\overline{b}} \Rightarrow s'$ and $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} \overline{b} \# Q : s''$. Then, by induction, we obtain $\nabla, \nabla' \vdash^{\text{NLC}} P\{N/x\} : (s'')^{\overline{b}} \Rightarrow s'$. To apply an instance of rule (AP), which completes the argument, we need to show that $\nabla, \nabla' \vdash^{\text{NLC}} \overline{b} \# Q\{N/x\} :$

s''. We pick $\overline{c} \# (\nabla, \nabla', \overline{b}, Q, N, \overline{a}, s)$. Note that by Proposition 3.1.15 we have that $\overline{c} \# Q\{N/x\}$ holds. We then demonstrate that $(\nabla, \nabla')^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{b}) * Q\{N/x\} \approx Q\{N/X\} : s''$ holds.

From $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} \overline{b} \# Q : s''$, we obtain by definition that $\nabla^{\#\overline{c}}, \overline{c} \cup \overline{a} \# x : s \vdash^{\text{NLC}} (\overline{c}\,\overline{b}) * Q \approx Q : s''$. Given that $\nabla' \vdash^{\text{NLC}} \overline{a} \# N : s$ we can apply Lemma 3.2.16 to obtain $\nabla' \vdash^{\text{NLC}} N : s$. We then apply instances of rules (REF) and (WEAK) to get $\nabla'^{\#\overline{c}} \vdash^{\text{NLC}} N \approx N : s$. Using Lemma 3.2.19, we can deduce from $\nabla'^{\#\overline{c}} \vdash^{\text{NLC}} N : s$ and $\overline{c} \# (\nabla', N, s)$ that $\nabla'^{\#\overline{c}} \vdash^{\text{NLC}} \overline{c} \# N : s$ holds. Further, by assumption and an application of (WEAK) we have that $\nabla'^{\#\overline{c}} \vdash^{\text{NLC}} \overline{a} \# N : s$ and therefore $\nabla'^{\#\overline{c}} \vdash^{\text{NLC}} \overline{a} \cup \overline{c} \# N : s$. Hence, we can apply the derived rule (SUBST) to obtain $\nabla^{\#\overline{c}}, \nabla'^{\#\overline{c}} \vdash^{\text{NLC}} ((\overline{c}\,\overline{b}) * Q)\{N/x\} \approx Q\{N/x\} : s''$. By Proposition 3.1.14 and $\nabla^{\#\overline{c}}, \nabla'^{\#\overline{c}} = (\nabla, \nabla')^{\#\overline{c}}$ we can complete the argument. □

The preservation lemma for types under simultaneous capture avoiding substitution follows likewise:

Lemma 3.2.21 (Preservation of Types under Sim. Substitution) *Let* $\nabla \stackrel{\text{def}}{=} \overline{a}_1 \# x_1 : s_1, ..., \overline{a}_n \# x_n : s_n$ *and* $1 \leq i \leq n$. *If* $Th \rhd \nabla \vdash^{\text{NLC}} M : s'$ *and* $Th \rhd \nabla' \vdash^{\text{NLC}} \overline{a_i} \# N_i : s_i$, *then* $Th \rhd \nabla' \vdash^{\text{NLC}} M\{N_1, \ldots, N_n/x_1, \ldots, x_n\} : s'$.

As already observed by Clouston [16], the following Lemma can be obtained.

Lemma 3.2.22 (Permutation Equation) *if* $Th \rhd \nabla \vdash^{\text{NLC}} M : s$ *and* $ds(\pi, \pi') \# (\nabla, M)$, *then* $Th \rhd \nabla^{\#ds(\pi,\pi')} \vdash^{\text{NLC}} \pi * M \approx \pi' * M : \pi \cdot s$.

Proof Suppose $\nabla \vdash^{\text{NLC}} M : s$ and $ds(\pi, \pi') \# (\nabla, M)$. We can deduce that $ds(\pi, \pi') \# s$. Next, we apply Lemma 3.2.14 to obtain $\nabla^{\#ds(\pi,\pi')} \vdash^{\text{NLC}} M : s$. Then, by Lemma 3.2.19, we get $\nabla^{\#ds(\pi,\pi')} \vdash^{\text{NLC}} ds(\pi, \pi') \# M : s$. By instances of the rules (REF) and (SUSP) we obtain $\nabla^{\#ds(\pi,\pi')} \vdash^{\text{NLC}} M \approx M : s$ and $ds(\pi, \pi') \# x : s \vdash^{\text{NLC}} \pi x \approx \pi' x : s$, respectively. The result then follows immediately by the derived rule (SUBST). □

3.3 Examples of NLC Expressions

We now present various interesting examples of expressions and their types in NLC. We begin with the derived expression

$$\emptyset \# y : s_2 \vdash^{\text{NLC}} \lambda^{a_1} x : s_1 . y : {s_1}^{a_1} \Rightarrow s_2$$

which trivially follows by (ABS) and (SP). Note that by augmenting the freshness context to $a_2 \# y : s_2$ we can still only deduce

$$a_2 \# y : s_2 \vdash^{\text{NLC}} \lambda^{a_1} x : s_1 . y : {s_1}^{a_1} \Rightarrow s_2$$

and not ${s_1}^{a_1} \Rightarrow {s_2}^{a_2}$, as one might initially expect, since this type is not supported in NLC. Another example is the identity function $id_{X^{\#a}} : X^{\#a} \to X^{\#a}$ in *FMSet*, which can be expressed by $\lambda^a x : s . x$. Again, we can still only deduce

$$\emptyset \vdash^{\text{NLC}} \lambda^a x : s . x : s^a \Rightarrow s$$

for type $s^a \Rightarrow s$ instead of the more precise type $s^a \Rightarrow s^a$. This indicates that NLC cannot *directly* capture all elements in an exponential object of the form $X_1^{\#\overline{a}_1} \Rightarrow_{fs} X_2^{\#\overline{a}_2}$ (in *FMSet*), at least not with their accurate type.

We continue with the following two examples, which demonstrate that the type system of NLC correctly enforces the freshness restrictions. We first show introduce an expression-in-context that cannot be derived since it violates a freshness restriction.

$$\emptyset \# y : s \not\vdash^{\text{NLC}} (\lambda^a x : s . x)\ y : s$$

This can easily be shown by using the model-theoretic semantics that we will introduce in the following section and by applying the contrapositive of Lemma 3.4.13 (Soundness Lemma). However by extending the freshness context to $a \# y : s$, the expression can clearly be derived using rules (AP), (ABS) and (SP):

$$a \# y : s \vdash^{\text{NLC}} (\lambda^a x : s . x)\ y : s$$

Interestingly, as will be shown next, the freshness restriction on the target of an identity function is indirectly captured by the following freshness assertion, which is derived by using rules (B), (SUSP) and (TRANS), as well as Proposition 3.2.5

$$a \# y : s \vdash^{\text{NLC}} a \# (\lambda^a x : s.\, x)\; y : s$$

Hence, using freshness assertions, we can determine if expressions have additional freshness restrictions, beyond the explicitly provided types in NLC. However, this does not suffice to resolve certain issues as we will discuss in Chapter 5.

3.4 A Sound Model-theoretic Semantics

In this section we define the model-theoretic semantics of NLC in *FMSet*. We first introduce the notion of a *Sg*-structure and an environment in *FMSet*, and then define the interpretation of NLC expressions for environments in such a structure. We continue by proving various auxiliary results for the interpretation function and conclude by demonstrating that type and equational soundness hold.

3.4.1 Structures, Environments and Interpretations in *FMSet*

Let *Sg* be a NLC-signature. A ***Sg*-structure** $\mathcal{M}$ in *FMSet* is specified by a pair of equivariant maps:

(i) $\mathcal{M}[\![\_]\!] : Gnd_{Sg} \to ob(FMSet)$, which extends to the map $\mathcal{M}[\![\_]\!] : Type_{Sg} \to ob(FMSet)$ via structural recursion on types:

$$\mathcal{M}[\![\gamma]\!] \stackrel{\text{def}}{=} \mathcal{M}[\![\gamma]\!]$$
$$\mathcal{M}[\![s^{\overline{a}} \Rightarrow s']\!] \stackrel{\text{def}}{=} \mathcal{M}[\![s]\!]^{\#\overline{a}} \Rightarrow_{fs} \mathcal{M}[\![s']\!]$$

The disjoint union of type interpretations in $\mathcal{M}$, denoted by $\mathcal{U} \stackrel{\text{def}}{=} \biguplus_{s \in Type_{Sg}} \mathcal{M}[\![s]\!]$, is called the **universe of** $\mathcal{M}$.

(ii) $\mathcal{M}[\![\_]\!] : \mathit{Fun}_{Sg} \to \mathcal{U}$ such that if $f : s$, then $\mathcal{M}[\![f]\!] \in \mathcal{M}[\![s]\!]$.

Lemma 3.4.1 *The function $\mathcal{M}[\![\_]\!] : \mathit{Type}_{Sg} \to ob(\mathit{FMSet})$ on types is equivariant. Moreover, the universe $\mathcal{U}$ is a nominal set.*

Proof Proof by induction on type s. The case for base types follows immediately by definition of a structure $\mathcal{M}$. For the inductive case (s is $s'^{\overline{a}} \Rightarrow s''$), it can be deduced as follows:

$$\begin{aligned}
\pi \cdot [\![s'^{\overline{a}} \Rightarrow s'']\!] &\stackrel{\text{def}}{=} \pi \cdot ([\![s']\!]^{\#\overline{a}} \Rightarrow_{fs} [\![s'']\!]) \\
&\stackrel{\text{def}}{=} \pi \cdot ([\![s']\!]^{\#\overline{a}}) \Rightarrow_{fs} \pi \cdot [\![s'']\!] \\
&= (\pi \cdot [\![s']\!])^{\#\pi\cdot\overline{a}} \Rightarrow_{fs} \pi \cdot [\![s'']\!] && \text{(Lemma 2.3.4)} \\
&= [\![\pi \cdot s']\!]^{\#\pi\cdot\overline{a}} \Rightarrow_{fs} [\![\pi \cdot s'']\!] && \text{(induction)} \\
&\stackrel{\text{def}}{=} [\![(\pi \cdot s')^{\pi\cdot\overline{a}} \Rightarrow \pi \cdot s'']\!] \\
&\stackrel{\text{def}}{=} [\![\pi \cdot (s'^{\overline{a}} \Rightarrow s'')]\!]
\end{aligned}$$

Given that $\mathcal{M}[\![\_]\!] : \mathit{Type}_{Sg} \to ob(\mathit{FMSet})$ is an equivariant map, it follows immediately that the universe $\mathcal{U}$ is closed under permutations and therefore it is a nominal set by construction. □

We define an **environment** for a structure $\mathcal{M}$ to be a finite function from Var to the universe $\mathcal{U}$ of $\mathcal{M}$. The set of environments, Env_{Sg}, equipped with the point-wise permutation action

$$(\pi \cdot \eta)(x) \stackrel{\text{def}}{=} \pi \cdot \eta(x)$$

is an FM-Set (finite product of FM-Sets) such that for every $\eta \in \mathit{Env}_{Sg}$ we have

$$\mathit{supp}(\eta) = \bigcup_{x \in dom(\eta)} \mathit{supp}(\eta(x))$$

If $\eta \in \mathit{Env}_{Sg}$, $x \in \mathit{Var}$ and $d \in \mathcal{U}$, then the **updated environment** $\eta[x \mapsto d]$ is mapping x to d, and otherwise acts like η.

We now define the interpretation of an NLC expression M under an environment η in a Sg-structure $\mathcal{M}$. At this point we have to be cautious, because NLC is dependently typed: in particular the type system and equation system are mutual inductively defined. This means that we *cannot* just introduce a recursive definition of a *total* interpretation function over NLC expressions and environments, and prove it sound. Instead, we proceed by using a proof technique, which is usually applied to prove soundness in the context of dependent types (see for example [66, 52]). This involves the definition of a *partial* interpretation function for expressions and environments, which is later shown to be total for well-typed expressions. More precisely, the application of the interpretation function relies on certain condition; conditions which directly follow from the derivation of well typed expressions in the soundness proof. Hence, the proof of "totality" for the interpretation function will merely be postponed, until we prove type and equational soundness.

As an example, we can take the interpretation of a lambda application $M\ N$ (see Table 3.7), which relies on the condition that the interpretation of M is a finitely supported function and that the interpretation of N is in its domain; thus the freshness condition is meet. When we now consider rule (AP) in the soundness proof (see page 83), we can immediately see that both conditions hold. Hence, the interpretation of $M\ N$ exists for well typed terms.

We define the **interpretation** of an NLC-expression M for an environment η (in an Sg-structure $\mathcal{M}$), written as $\mathcal{M}[\![M]\!]_\eta$, by induction on the structure of M (see Table 3.7). To indicate if an interpretation is defined, we write $\mathcal{M}[\![M]\!]_\eta\Downarrow$. Hence, we have defined a *partial* **interpretation function**:

$$\mathcal{M}[\![-]\!]_- : Term_{Sg}/{\sim_\alpha} \times Env_{Sg} \to \mathcal{U}$$

Note that if it is clear from context, we usually do not explicitly mention the underlying Sg-structure.

$\mathcal{M}[\![\pi x]\!]_\eta$	$\stackrel{\text{def}}{=} \pi \cdot \eta(x)$	if $x \in dom(\eta)$
$\mathcal{M}[\![c]\!]_\eta$	$\stackrel{\text{def}}{=} \mathcal{M}[\![c]\!]$	
$\mathcal{M}[\![\lambda^{\overline{a}} x : s.\, M]\!]_\eta$	$\stackrel{\text{def}}{=} \Lambda d \in \mathcal{M}[\![s]\!]^{\#\overline{a}}.\mathcal{M}[\![M]\!]_{\eta[x \mapsto d]}$	if $\mathcal{M}[\![M]\!]_{\eta[x\mapsto d]}\Downarrow$ for any $d \in \mathcal{M}[\![s]\!]^{\#\overline{a}}$
$\mathcal{M}[\![M\ N]\!]_\eta$	$\stackrel{\text{def}}{=} \mathcal{M}[\![M]\!]_\eta(\mathcal{M}[\![N]\!]_\eta)$	if $\mathcal{M}[\![M]\!]_\eta\Downarrow$, $\mathcal{M}[\![N]\!]_\eta\Downarrow$, $\mathcal{M}[\![M]\!]_\eta$ is a fs-function and $\mathcal{M}[\![N]\!]_\eta$ in its domain

Table 3.7: Partial Interpretation Function for NLC

3.4.2 Properties of the Interpretation Function

To prove soundness, we require various auxiliary results for the previously defined partial interpretation function. Due to the fact that the partial interpretation function is defined by recursion on the structure of expressions, we can prove the following properties by induction on the structure of expressions. Taking into account that the interpretation function is *partial*, we need to introduce the notion of *Kleene equality*: We shall write $L \asymp R$ to mean that $L\Downarrow \iff R\Downarrow$ and that $L = R$.

Lemma 3.4.2 *Let η be an environment, M an expression and x, x' two distinct variables. Then*

$$[\![(x\,x') \bullet M]\!]_\eta \asymp [\![M]\!]_{\eta\circ(x\,x')}$$

Proof Proof by induction on the structure of expression M:

SUSP: (M is πy) ($y \neq x, x'$):

$$\begin{aligned}
[\![(x\,x') \bullet \pi y]\!]_\eta &\asymp [\![\pi y]\!]_\eta \\
&\asymp \pi \cdot \eta(y) \\
&\asymp \pi \cdot ((\eta \circ (x\,x'))(y)) \\
&\asymp [\![\pi y]\!]_{\eta\circ(x\,x')}
\end{aligned}$$

M is πx: (similarly for x')

$$\begin{aligned}\llbracket (x\,x') \bullet \pi x \rrbracket_\eta &\asymp \llbracket \pi x' \rrbracket_\eta \\ &\asymp \pi \cdot \eta(x') \\ &\asymp \pi \cdot ((\eta \circ (x\,x'))(x)) \\ &\asymp \llbracket \pi x \rrbracket_{\eta \circ (x\,x')}\end{aligned}$$

LAM-ABS: (M is $\lambda^{\overline{a}} y : s.\, N$): ($y \neq x, x'$) The cases for ($y = x$) and ($y = x'$) follow similarly. Suppose $\llbracket (x\,x') \bullet (\lambda^{\overline{a}} y : s.\, N) \rrbracket_\eta \Downarrow$. By definition we have $\llbracket \lambda^{\overline{a}} y : s.\, (x\,x') \bullet N \rrbracket_\eta \Downarrow$, and it is equal to $\Lambda d \in \llbracket s \rrbracket^{\#\overline{a}}. \llbracket (x\,x') \bullet N \rrbracket_{\eta[y \mapsto d]}$ with $\llbracket (x\,x') \bullet N \rrbracket_{\eta[y \mapsto d]} \Downarrow$ for all $d \in \llbracket s \rrbracket^{\#\overline{a}}$. By induction we have that for all $d \in \llbracket s \rrbracket^{\#\overline{a}}$, $\llbracket N \rrbracket_{(\eta[y \mapsto d]) \circ (x\,x')} \Downarrow$ and $\llbracket (x\,x') \bullet N \rrbracket_{\eta[y \mapsto d]} = \llbracket N \rrbracket_{(\eta[y \mapsto d]) \circ (x\,x')}$. Given that $y \neq x, x'$ we have $\llbracket N \rrbracket_{(\eta[y \mapsto d]) \circ (x\,x')} = \llbracket N \rrbracket_{\eta \circ (x\,x')[y \mapsto d]}$ ($\diamond 1$), and by definition of partial interpretations we have $\llbracket \lambda^{\overline{a}} y : s.\, N \rrbracket_{\eta \circ (x\,x')} \Downarrow$. The converse follows similarly. We now continue with the equational part. Let $d' \in \llbracket s \rrbracket^{\#\overline{a}}$.

$$\begin{aligned}\llbracket (x\,x') \bullet (\lambda^{\overline{a}} y : s.\, N) \rrbracket_\eta (d') &\stackrel{\text{def}}{=} \llbracket \lambda^{\overline{a}} y : s.\, (x\,x') \bullet N \rrbracket_\eta (d') \\ &\stackrel{\text{def}}{=} (\Lambda d \in \llbracket s \rrbracket^{\#\overline{a}}. \llbracket (x\,x') \bullet N \rrbracket_{\eta[y \mapsto d]})(d') \\ &\stackrel{\text{def}}{=} \llbracket (x\,x') \bullet N \rrbracket_{\eta[y \mapsto d']} \\ &= \llbracket N \rrbracket_{(\eta[y \mapsto d']) \circ (x\,x')} && \text{(induction)} \\ &= \llbracket N \rrbracket_{\eta \circ (x\,x')[y \mapsto d']} && (\diamond 1) \\ &\stackrel{\text{def}}{=} (\Lambda d \in \llbracket s \rrbracket^{\#\overline{a}}. \llbracket N \rrbracket_{\eta \circ (x\,x')[y \mapsto d]})(d') \\ &\stackrel{\text{def}}{=} \llbracket \lambda^{\overline{a}} y : s.\, N \rrbracket_{\eta \circ (x\,x')}\end{aligned}$$

APP: (M is $N\ N'$): Suppose that $\llbracket (x\,x') \bullet (N\ N') \rrbracket_\eta \Downarrow$. By definition of variable swapping we have $\llbracket ((x\,x') \bullet N)\ ((x\,x') \bullet N') \rrbracket_\eta \Downarrow$, and it is equal to $\llbracket (x\,x') \bullet N \rrbracket_\eta (\llbracket (x\,x') \bullet N' \rrbracket_\eta)$ with $\llbracket (x\,x') \bullet N \rrbracket_\eta \Downarrow$ and $\llbracket (x\,x') \bullet N' \rrbracket_\eta \Downarrow$. By induction we obtain

$$\begin{aligned}&\llbracket N \rrbracket_{\eta \circ (x\,x')} \Downarrow && \llbracket (x\,x') \bullet N \rrbracket_\eta = \llbracket N \rrbracket_{\eta \circ (x\,x')} \\ &\llbracket N' \rrbracket_{\eta \circ (x\,x')} \Downarrow && \llbracket (x\,x') \bullet N' \rrbracket_\eta = \llbracket N' \rrbracket_{\eta \circ (x\,x')}\end{aligned}$$

From this we can directly deduce that $[\![N\ N']\!]_{\eta\circ(x\,x')}\Downarrow$. The equation is deduced as follows:

$$\begin{aligned}
[\![(x\,x')\bullet(N\ N')]\!]_\eta &\stackrel{\text{def}}{=} [\![((x\,x')\bullet N)\ ((x\,x')\bullet N')]\!]_\eta \\
&\stackrel{\text{def}}{=} [\![(x\,x')\bullet N]\!]_\eta([\![(x\,x')\bullet N']\!]_\eta) \\
&= [\![N]\!]_{\eta\circ(x\,x')}([\![N']\!]_{\eta\circ(x\,x')}) \qquad \text{(induction)} \\
&\stackrel{\text{def}}{=} [\![(N\ N')]\!]_{\eta\circ(x\,x')}
\end{aligned}$$

□

The following lemma shows that the interpretation of an expression M for an environment η only depends on $\eta(x)$ if $x \in fv(M)$.

Lemma 3.4.3 *Let η and η' be environments and M an expression. If $\eta(x) = \eta'(x)$ for all $x \in fv(M)$, then $[\![M]\!]_\eta \asymp [\![M]\!]_{\eta'}$.*

Proof Proof by induction on the structure of expression M.

SUSP: (M is πx):

$$[\![\pi x]\!]_\eta \asymp \pi\cdot\eta(x) \asymp \pi'\cdot\eta(x) \asymp [\![\pi x]\!]_{\eta'}$$

LAM-ABS: (M is $\lambda^{\overline{a}}y : s.\,N$). Suppose that for all $x \in fv(M) = fv(N)\setminus\{y\}$ we have that $\eta(x) = \eta'(x)$. We now assume that $[\![\lambda^{\overline{a}}y : s.\,N]\!]_\eta\Downarrow$, which is equal by definition to $\Lambda d \in [\![s]\!]^{\#\overline{a}}.[\![N]\!]_{\eta[y\mapsto d]}$ with $[\![N]\!]_{\eta[y\mapsto d]}\Downarrow$ for all $d \in [\![s]\!]^{\#\overline{a}}$. We recall that $\eta[y\mapsto d] = \eta'[y\mapsto d]$ for all $x \in fv(N)$. So, by induction, we obtain that $[\![N]\!]_{\eta'[y\mapsto d]}\Downarrow$ and $[\![N]\!]_{\eta[y\mapsto d]} = [\![N]\!]_{\eta'[y\mapsto d]}$ for all $d \in [\![s]\!]^{\#\overline{a}}$. From this we can directly deduce that $[\![\lambda^{\overline{a}}y : s.\,N]\!]_{\eta'}\Downarrow$. The equation is deduced as follows: Let $d' \in [\![s]\!]^{\#\overline{a}}$.

$$\begin{aligned}
[\![\lambda^{\overline{a}}y : s.\,N]\!]_\eta(d') &\stackrel{\text{def}}{=} [\![N]\!]_{\eta[y\mapsto d']} \\
&= [\![N]\!]_{\eta'[y\mapsto d']} \qquad \text{(induction)} \\
&\stackrel{\text{def}}{=} [\![\lambda^{\overline{a}}y : s.\,N]\!]_{\eta'}(d')
\end{aligned}$$

APP: (M is $N\ N'$): The existence part follows immediately by induction, as well as the equation:

$$\begin{aligned}\llbracket N\ N' \rrbracket_\eta &\stackrel{\text{def}}{=} \llbracket N \rrbracket_\eta(\llbracket N' \rrbracket_\eta) \\ &= \llbracket N \rrbracket_{\eta'}(\llbracket N' \rrbracket_{\eta'}) && \text{(induction)} \\ &\stackrel{\text{def}}{=} \llbracket N\ N' \rrbracket_{\eta'}\end{aligned}$$

□

To show that the partial interpretation function is well defined we need to demonstrate that it preserves α-equivalence.

Lemma 3.4.4 *Let η be an environment and M, M' two raw terms. If $M \equiv_\alpha M'$, then $\llbracket M \rrbracket_\eta \asymp \llbracket M' \rrbracket_\eta$.*

Proof Proof by rule-based induction on $\equiv_\alpha$:

SUSP: For $\pi x \equiv_\alpha \pi x$ it follows immediately.

ABS: Suppose that $\lambda^{\overline{a}} x : s.\, N \equiv_\alpha \lambda^{\overline{a}} x' : s.\, N'$. From this we can deduce that for some name $z \# (N, N', x, x')$ we have $(z\,x) \bullet N \equiv_\alpha (z\,x') \bullet N'$. Given that α-equivalent terms share the same set of free variables, we have that $S \stackrel{\text{def}}{=} fv(\lambda^{\overline{a}} x : s.\, N) = fv(\lambda^{\overline{a}} x' : s.\, N')$. It follows immediately that $x, x', z \notin S$.

Suppose that $\llbracket \lambda^{\overline{a}} x : s.\, N \rrbracket_\eta \Downarrow$. We take η' to be the restriction of η to S. Using Lemma 3.4.3 we obtain $\llbracket \lambda^{\overline{a}} x : s.\, N \rrbracket_{\eta'} \Downarrow$ and

$$\llbracket \lambda^{\overline{a}} x : s.\, N \rrbracket_\eta = \llbracket \lambda^{\overline{a}} x : s.\, N \rrbracket_{\eta'} \ (\diamond 1)$$

By definition, we have that $\llbracket \lambda^{\overline{a}} x : s.\, N \rrbracket_{\eta'}$ is equal to $\Lambda d \in \llbracket s \rrbracket^{\#\overline{a}}.\llbracket N \rrbracket_{\eta'[x \mapsto d]}$ and $\llbracket N \rrbracket_{\eta'[x \mapsto d]} \Downarrow$ for all $d \in \llbracket s \rrbracket^{\#\overline{a}}$. Given that $x, z \notin dom(\eta')$, we have $\llbracket N \rrbracket_{\eta'[x \mapsto d]} = \llbracket N \rrbracket_{(\eta'[z \mapsto d]) \circ (z\,x)}$ and by applying Lemma 3.4.2 we obtain $\llbracket (z\,x) \bullet N \rrbracket_{\eta'[z \mapsto d]} \Downarrow$ and

$$\llbracket N \rrbracket_{(\eta'[z \mapsto d]) \circ (z\,x)} = \llbracket (z\,x) \bullet N \rrbracket_{\eta'[z \mapsto d]} \ (\diamond 2)$$

Then, by induction, we have that $[\![(z\,x') \bullet N']\!]_{\eta'[z\mapsto d]}\Downarrow$ and

$$[\![(z\,x) \bullet N]\!]_{\eta'[z\mapsto d]} = [\![(z\,x') \bullet N']\!]_{\eta'[z\mapsto d]} \ (\diamond 3)$$

We now repeat the same argument in the reverse order. We apply Lemma 3.4.2 to obtain $[\![N']\!]_{(\eta'[z\mapsto d])\circ(z\,x')}\Downarrow$ and

$$[\![(z\,x') \bullet N']\!]_{\eta'[z\mapsto d]} = [\![N']\!]_{(\eta'[z\mapsto d])\circ(z\,x')} \ (\diamond 4)$$

Given that $x', z \notin dom(\eta')$, we have $[\![N']\!]_{(\eta'[z\mapsto d])\circ(z\,x')} = [\![N']\!]_{\eta'[x'\mapsto d]}$. So, we have that for all $d \in [\![s]\!]^{\#\overline{a}}$, $[\![N']\!]_{\eta'[x'\mapsto d]}\Downarrow$ and therefore $[\![\lambda^{\overline{a}}x' : s.\, N']\!]_{\eta'}$. By applying Lemma 3.4.3 we obtain that $[\![\lambda^{\overline{a}}x' : s.\, N']\!]_{\eta}\Downarrow$ and

$$[\![\lambda^{\overline{a}}x' : s.\, N']\!]_{\eta'} = [\![\lambda^{\overline{a}}x' : s.\, N']\!]_{\eta} \ (\diamond 5)$$

The converse direction of the existence parts follows similarly. We continue with the equational part: Let $d \in [\![s]\!]^{\#\overline{a}}$.

$$\begin{aligned}
[\![\lambda^{\overline{a}}x : s.\, N]\!]_{\eta}(d) &= [\![\lambda^{\overline{a}}x : s.\, N]\!]_{\eta'}(d) && (\diamond 1)\\
&\stackrel{\text{def}}{=} [\![N]\!]_{\eta'[x\mapsto d]} &&\\
&= [\![N]\!]_{(\eta'[z\mapsto d])\circ(z\,x)} && (x, z \notin dom(\eta'))\\
&= [\![(z\,x) \bullet N]\!]_{\eta'[z\mapsto d]} && (\diamond 2)\\
&= [\![(z\,x') \bullet N']\!]_{\eta'[z\mapsto d]} && (\diamond 3)\\
&= [\![N']\!]_{(\eta'[z\mapsto d])\circ(z\,x')} && (\diamond 4)\\
&= [\![N']\!]_{\eta'[x'\mapsto d]} && (x', z \notin dom(\eta'))\\
&\stackrel{\text{def}}{=} [\![\lambda^{\overline{a}}x' : s.\, N']\!]_{\eta'}(d) &&\\
&= [\![\lambda^{\overline{a}}x' : s.\, N']\!]_{\eta}(d) && (\diamond 5)
\end{aligned}$$

APP: Suppose that $M\ N \equiv_\alpha M'\ N'$. From this we can deduce that $M \equiv_\alpha M'$ and $N \equiv_\alpha N'$. For the existence part we suppose that $[\![M\ N]\!]_\eta\Downarrow$. By definition we have $[\![M]\!]_\eta\Downarrow$ and $[\![N]\!]_\eta\Downarrow$. Then, by induction, we have that $[\![M']\!]_\eta\Downarrow$, $[\![N']\!]_\eta$, $[\![M]\!]_\eta = [\![M']\!]_\eta$

and $[\![N]\!]_\eta = [\![N']\!]_\eta$. Hence, we have that $[\![M'\ N']\!]_\eta\Downarrow$ and both interpretations can be shown equal as follows:

$$\begin{aligned}[\![M\ N]\!]_\eta &\stackrel{\text{def}}{=} [\![M]\!]_\eta([\![N]\!]_\eta) \\ &= [\![M']\!]_\eta([\![N']\!]_\eta) && \text{(induction)} \\ &\stackrel{\text{def}}{=} [\![M'\ N']\!]_\eta\end{aligned}$$

□

We continue by proving various properties involving the object-level permutation action, meta-level permutation action and capture-avoiding substitution in the context of the partial interpretation function. The existence part of the Kleene equality follows rather straightforwardly. So, for brevity, we only provide details for the equality part.

Lemma 3.4.5 *Let M be an expression and $\eta[x \mapsto d]$ an environment. Then*

$$[\![M]\!]_{\eta[x\mapsto\pi^{-1}\cdot d]} \asymp [\![M[\pi^{-1}x/x]]\!]_{\eta[x\mapsto d]}$$

Proof Proof by induction on the structure of expression M

SUSP: (M is τy) We consider the case $y = x$. The other case follows similarly.

$$\begin{aligned}[\![\tau x]\!]_{\eta[x\mapsto\pi^{-1}\cdot d]} &\stackrel{\text{def}}{=} \tau\cdot(\pi^{-1}\cdot d) \\ &\stackrel{\text{def}}{=} \tau\pi^{-1}\cdot d \\ &\stackrel{\text{def}}{=} [\![\tau\pi^{-1}x]\!]_{\eta[x\mapsto d]} \\ &\stackrel{\text{def}}{=} [\![(\tau x)[\pi^{-1}x/x]]\!]_{\eta[x\mapsto d]}\end{aligned}$$

LAM-APP: (M is $\lambda^{\overline{a}}y : s.\, N$). We consider the case ($y \neq x$). The other case

follows immediately. Let $d' \in [\![s]\!]^{\#\overline{a}}$.

$$\begin{aligned}
[\![\lambda^{\overline{a}} y : s.\, N]\!]_{\eta[x \mapsto \pi^{-1} \cdot d]}(d') &\stackrel{\text{def}}{=} [\![N]\!]_{\eta[x \mapsto \pi^{-1} \cdot d][y \mapsto d']} \\
&= [\![N[\pi^{-1} x/x]]\!]_{\eta[x \mapsto d][y \mapsto d']} && \text{(induction)} \\
&\stackrel{\text{def}}{=} [\![\lambda^{\overline{a}} y : s.\, N[\pi^{-1} x/x]]\!]_{\eta[x \mapsto d]} \\
&\stackrel{\text{def}}{=} [\![(\lambda^{\overline{a}} y : s.\, N)[\pi^{-1} x/x]]\!]_{\eta[x \mapsto d]}
\end{aligned}$$

APP: (M is $N\ N'$).

$$\begin{aligned}
[\![N\ N']\!]_{\eta[x \mapsto \pi^{-1} \cdot d]} &\stackrel{\text{def}}{=} [\![N]\!]_{\eta[x \mapsto \pi^{-1} \cdot d]}([\![N']\!]_{\eta[x \mapsto \pi^{-1} \cdot d]}) \\
&= [\![N[\pi^{-1} x/x]]\!]_{\eta[x \mapsto d]}([\![N'[\pi^{-1} x/x]]\!]_{\eta[x \mapsto d]}) && \text{(induction)} \\
&\stackrel{\text{def}}{=} [\![(N[\pi^{-1} x/x])\ (N'[\pi^{-1} x/x])]\!]_{\eta[x \mapsto d]} \\
&\stackrel{\text{def}}{=} [\![(N\ N')[\pi^{-1} x/x]]\!]_{\eta[x \mapsto d]}
\end{aligned}$$

□

Lemma 3.4.6 *Let M be an expression and η an environment. Then*

$$\pi \cdot [\![M]\!]_\eta \asymp [\![\pi * M]\!]_\eta$$

Proof Proof by induction on the structure of expression M.

SUSP: (M is τx)

$$\begin{aligned}
\pi \cdot [\![\tau x]\!]_\eta &\asymp \pi \cdot \tau \cdot \eta(x) \\
&\asymp (\pi \circ \tau) \cdot \eta(x) \\
&\asymp [\![\pi \tau x]\!]_\eta \\
&\asymp [\![\pi * \tau x]\!]_\eta
\end{aligned}$$

LAM-ABS: (M is $\lambda^{\overline{a}}x : s.\, N$). Let $d \in [\![\pi \cdot s]\!]^{\#\pi\cdot\overline{a}}$.

$$\begin{aligned}
(\pi \cdot [\![\lambda^{\overline{a}}x : s.\, N]\!]_\eta)(d) &\stackrel{\text{def}}{=} \pi \cdot [\![\lambda^{\overline{a}}x : s.\, N]\!]_\eta(\pi^{-1} \cdot d) \\
&\stackrel{\text{def}}{=} \pi \cdot [\![N]\!]_{\eta[x\mapsto\pi^{-1}\cdot d]} \\
&= [\![\pi * N]\!]_{\eta[x\mapsto\pi^{-1}\cdot d]} && \text{(induction)} \\
&= [\![(\pi * N)\{\pi^{-1}x/x\}]\!]_{\eta[x\mapsto d]} && \text{(Lemma 3.4.5)} \\
&= [\![(\pi * N)[\pi^{-1}x/x]]\!]_{\eta[x\mapsto d]} && \text{(Lemma 3.1.11)} \\
&= [\![\pi * N[\pi^{-1}x/x]]\!]_{\eta[x\mapsto d]} && \text{(Lemma 3.1.4)} \\
&\stackrel{\text{def}}{=} [\![\lambda^{\pi\cdot\overline{a}}x : \pi \cdot s.\, \pi * N[\pi^{-1}x/x]]\!]_\eta(d) \\
&\stackrel{\text{def}}{=} [\![\pi * \lambda^{\overline{a}}x : s.\, N]\!]_\eta(d)
\end{aligned}$$

APP: (M is $N\ N'$).

$$\begin{aligned}
\pi \cdot [\![N\ N']\!]_\eta &\stackrel{\text{def}}{=} \pi \cdot [\![N]\!]_\eta([\![N']\!]_\eta) \\
&\stackrel{\text{def}}{=} (\pi \cdot [\![N]\!]_\eta)\ (\pi \cdot [\![N']\!]_\eta) \\
&= [\![\pi * N]\!]_\eta([\![\pi * N']\!]_\eta) && \text{(induction)} \\
&\stackrel{\text{def}}{=} [\![(\pi * N)\ (\pi * N')]\!]_\eta \\
&\stackrel{\text{def}}{=} [\![\pi * N\ N']\!]_\eta
\end{aligned}$$

□

Lemma 3.4.7 *Let M be an expression and η an environment. Then*

$$\pi \cdot [\![M]\!]_\eta \asymp [\![\pi \cdot M]\!]_{\pi\cdot\eta}$$

Proof Proof by induction on the structure of expression M.

SUSP: (M is τx).

$$\begin{aligned}
\pi \cdot [\![\tau x]\!]_\eta &\stackrel{\text{def}}{=} \pi \cdot (\tau \cdot \eta(x)) \\
&\stackrel{\text{def}}{=} \pi\tau \cdot \eta(x) \\
&\stackrel{\text{def}}{=} \pi\tau\pi^{-1}\pi \cdot \eta(x) \\
&\stackrel{\text{def}}{=} \pi\tau\pi^{-1} \cdot ((\pi \cdot \eta)(x)) \\
&\stackrel{\text{def}}{=} [\![\pi\tau\pi^{-1}x]\!]_{\pi\cdot\eta} \\
&\stackrel{\text{def}}{=} [\![\pi \cdot (\tau x)]\!]_{\pi\cdot\eta}
\end{aligned}$$

LAM-ABS: (M is $\lambda^{\overline{a}}x : s.\, N$). Let $d \in [\![\pi \cdot s]\!]^{\#\pi\cdot\overline{a}}$.

$$\begin{aligned}
(\pi \cdot [\![\lambda^{\overline{a}}x : s.\, N]\!]_\eta)(d) &\stackrel{\text{def}}{=} \pi \cdot [\![\lambda^{\overline{a}}x : s.\, N]\!]_\eta(\pi^{-1} \cdot d) \\
&\stackrel{\text{def}}{=} \pi \cdot [\![N]\!]_{\eta[x\mapsto\pi^{-1}\cdot d]} \\
&= [\![\pi \cdot N]\!]_{(\pi\cdot\eta)[x\mapsto d]} && \text{(induction)} \\
&\stackrel{\text{def}}{=} [\![\lambda^{\pi\cdot\overline{a}}x : \pi \cdot s.\, \pi \cdot N]\!]_{\pi\cdot\eta}(d) \\
&\stackrel{\text{def}}{=} [\![\pi \cdot (\lambda^{\overline{a}}x : s.\, N)]\!]_{\pi\cdot\eta}(d)
\end{aligned}$$

APP: (M is $N\ N'$).

$$\begin{aligned}
\pi \cdot [\![N\ N']\!]_\eta &\stackrel{\text{def}}{=} \pi \cdot [\![N]\!]_\eta([\![N']\!]_\eta) \\
&\stackrel{\text{def}}{=} (\pi \cdot [\![N]\!]_\eta)\ (\pi \cdot [\![N']\!]_\eta) \\
&= [\![\pi \cdot N]\!]_{\pi\cdot\eta}([\![\pi \cdot N']\!]_{\pi\cdot\eta}) && \text{(induction)} \\
&\stackrel{\text{def}}{=} [\![(\pi \cdot N)\ (\pi \cdot N')]\!]_{\pi\cdot\eta} \\
&\stackrel{\text{def}}{=} [\![\pi \cdot N\ N']\!]_{\pi\cdot\eta}
\end{aligned}$$

□

Lemma 3.4.8 (Substitution Lemma) *Let M and N be expressions and η an environment.*

$$[\![M\{N/x\}]\!]_\eta \asymp [\![M]\!]_{\eta[x\mapsto[\![N]\!]_\eta]}$$

Proof Induction on the size of expression M.

SUSP: (M is πy): Case: ($y = x$)

$$\begin{aligned}
[\![(\pi x)\{N/x\}]\!]_\eta &\stackrel{\text{def}}{=} [\![\pi * N]\!]_\eta \\
&= \pi \cdot [\![N]\!]_\eta && \text{(Lemma 3.4.6)} \\
&\stackrel{\text{def}}{=} [\![\pi x]\!]_{\eta[x \mapsto [\![N]\!]_\eta]}
\end{aligned}$$

Case: ($y \neq x$)

$$\begin{aligned}
[\![(\pi y)\{N/x\}]\!]_\eta &\stackrel{\text{def}}{=} [\![\pi y]\!]_\eta \\
&\stackrel{\text{def}}{=} \pi \cdot \eta(y) \\
&\stackrel{\text{def}}{=} [\![\pi y]\!]_{\eta[x \mapsto [\![N]\!]_\eta]}
\end{aligned}$$

LAM-ABS: (M is $\lambda^{\overline{a}} y : s.\, M'$) ($y \neq x$). Let η be an environment and $d \in [\![s]\!]^{\#\overline{a}}$. We deduce by Lemma 3.4.3 that $[\![\lambda^{\overline{a}} y : s.\, M']\!]_{\eta[x \mapsto [\![N]\!]_\eta]} = [\![\lambda^{\overline{a}} y : s.\, M']\!]_{\eta'[x \mapsto [\![N]\!]_\eta]}$ ($\diamond 1$), where η' is the restriction of η to $fv(M)$. So, given that $y \notin fv(M)$ we have $y \notin dom(\eta')$.

$$\begin{aligned}
& [\![(\lambda^{\overline{a}} y : s.\, M')\{N/x\}]\!]_\eta(d) \\
\stackrel{\text{def}}{=}\ & [\![\lambda^{\overline{a}} z : s.\, ((z\, y) \bullet M')\{N/x\}]\!]_\eta(d) && \text{(for } z \# (M', N)) \\
\stackrel{\text{def}}{=}\ & [\![((z\, y) \bullet M')\{N/x\}]\!]_{\eta[z \mapsto d]} \\
=\ & [\![(z\, y) \bullet M']\!]_{\eta[z \mapsto d][x \mapsto [\![N]\!]_{\eta[z \mapsto d]}]} && \text{(induction)} \\
=\ & [\![(z\, y) \bullet M']\!]_{\eta'[z \mapsto d][x \mapsto [\![N]\!]_{\eta[z \mapsto d]}]} && \text{(Lemma 3.4.3, } z \# M') \\
=\ & [\![M']\!]_{(\eta'[z \mapsto d][x \mapsto [\![N]\!]_{\eta[z \mapsto d]}]) \circ (z\, y)} && \text{(Lemma 3.4.2)} \\
=\ & [\![M']\!]_{\eta'[y \mapsto d][x \mapsto [\![N]\!]_{\eta[z \mapsto d]}]} && ((z, y \neq x);\ (z, y \notin dom(\eta'))) \\
=\ & [\![M']\!]_{\eta'[y \mapsto d][x \mapsto [\![N]\!]_\eta]} && \text{(Lemma 3.4.3; } z \# N) \\
\stackrel{\text{def}}{=}\ & [\![\lambda^{\overline{a}} y : s.\, M']\!]_{\eta'[x \mapsto [\![N]\!]_\eta]}(d) \\
=\ & [\![\lambda^{\overline{a}} y : s.\, M']\!]_{\eta[x \mapsto [\![N]\!]_\eta]}(d) && \text{(Lemma 3.4.3}
\end{aligned}$$

APP: (M is $P\ P'$):

$$\begin{aligned}
[\![(P\ P')\{N/x\}]\!]_\eta &\stackrel{\text{def}}{=} [\![P\{N/x\}\ P'\{N/x\}]\!]_\eta \\
&\stackrel{\text{def}}{=} [\![P\{N/x\}]\!]_\eta([\![P'\{N/x\}]\!]_\eta) \\
&= [\![P]\!]_{\eta[x\mapsto[\![N]\!]_\eta]}([\![P']\!]_{\eta[x\mapsto[\![N]\!]_\eta]}) \qquad \text{(induction)} \\
&\stackrel{\text{def}}{=} [\![P\ P']\!]_{\eta[x\mapsto[\![N]\!]_\eta]}
\end{aligned}$$

□

3.4.3 Type and Equational Soundness

Before we prove soundness by mutual induction on the type and equation system, we introduce the notion of satisfaction for environments, terms-in-context and equations-in-context.

Definition 3.4.9 *An environment η **satisfies** a context ∇, written as $\eta \models \nabla$, if $\eta(x) \in [\![s]\!]^{\#\overline{a}}$ for every $\overline{a} \# x : s \in \nabla$.*

Lemma 3.4.10 *$\models \subseteq Env_{Sg} \times Ctxt_{Sg}$ is equivariant.*

Proof Let ∇ be a context and η an environment such that $\eta \models \nabla$. We demonstrate that $\pi \cdot \eta \models \pi \cdot \nabla$ for any $\pi \in Perm(\mathbb{A})$, i.e. for any $\overline{a} \# x : s \in \pi \cdot \nabla$ we need to show that $(\pi \cdot \eta)(x) \in [\![s]\!]^{\#\overline{a}}$

We first deduce from $\overline{a} \# x : s \in \pi \cdot \nabla$ that $\pi^{-1} \cdot \overline{a} \# x : \pi^{-1} \cdot s \in \nabla$. Then, given that $\eta \models \nabla$ holds by assumption, we have that $\eta(x) \in [\![\pi^{-1} \cdot s]\!]^{\#\pi^{-1}\cdot\overline{a}} = \pi^{-1} \cdot [\![s]\!]^{\#\overline{a}}$. We can now deduce that $\pi \cdot \eta(x) \in [\![s]\!]^{\#\overline{a}}$. Due to $(\pi \cdot \eta)(x) \stackrel{\text{def}}{=} \pi \cdot \eta(x)$, we have completed the argument. □

Definition 3.4.11 *The notion of satisfaction for terms-in-context and equations-in-context is defined as follows:*

- *Given* $\nabla \vdash^{\text{NLC}} M : s$ *we say that* $\mathcal{M}$ ***satisfies*** *the typing judgement if for any* $\eta \models \nabla$ *we have that* $\mathcal{M}[\![M]\!]_\eta \Downarrow$, *and* $\mathcal{M}[\![M]\!]_\eta \in \mathcal{M}[\![s]\!]$.

- *Given* $\nabla \vdash^{\text{NLC}} M \approx M' : s$ *we say that* $\mathcal{M}$ ***satisfies*** *the equation judgement if for any* $\eta \models \nabla$, *we have that* $\mathcal{M}[\![M]\!]_\eta \Downarrow$, $\mathcal{M}[\![M']\!]_\eta \Downarrow$ *and* $\mathcal{M}[\![M]\!]_\eta = \mathcal{M}[\![M']\!]_\eta$

Definition 3.4.12 *A Sg-structure* $\mathcal{M}$ *is a* ***model*** *of an* NLC*-theory* $Th = (Sg, Ax)$ *if* $\mathcal{M}$ *satisfies all of the equations-in-context in* Ax.

Theorem 3.4.13 (Type and Equational Soundness) *Let* Th *be a* NLC *theory and* $\mathcal{M}$ *a model. Then every well typed expression* $Th \rhd \nabla \vdash^{\text{NLC}} M : s$ *and theorem* $Th \rhd \nabla \vdash^{\text{NLC}} M \approx M' : s$ *is satisfied in* $\mathcal{M}$.

Proof The proof does proceed by mutual induction. Induction Property Closure for the rules in Table 3.4 and Table 3.5 is proved as follows:

(SP): Suppose $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} \pi x : \pi \cdot s$ and $\eta \models \nabla, \overline{a} \# x : s$. So, by definition, we have that $\eta(x) \in [\![s]\!]^{\#\overline{a}}$. From this we can deduce that $\pi \cdot \eta(x) \in [\![\pi \cdot s]\!]^{\#\pi\cdot\overline{a}} \subseteq [\![\pi \cdot s]\!]$. Further, we can deduce that $x \in dom(\eta)$ and therefore $[\![\pi x]\!]_\eta \Downarrow$. It then follows immediately that

$$[\![\pi x]\!]_\eta \stackrel{\text{def}}{=} \pi \cdot \eta(x) \in \pi \cdot [\![s]\!] = [\![\pi \cdot s]\!]$$

(C): It directly follows by the definition of a model, or more precisely the underlying structure.

(ABS): Suppose we have $\nabla \vdash^{\text{NLC}} \lambda^{\overline{a}} x : s.\, M : s^{\overline{a}} \Rightarrow s'$. We consider the case where x does not occur in the context ∇. From this we can deduce that $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} M : s'$. Let $\eta \models \nabla$. We have to show that $[\![\lambda^{\overline{a}} x : s.\, M]\!]_\eta \Downarrow$ and $[\![\lambda^{\overline{a}} x : s.\, M]\!]_\eta \in [\![s^{\overline{a}} \Rightarrow s']\!]$. Pick $d \in [\![s]\!]^{\#\overline{a}}$. It follows immediately that $\eta[x \mapsto d] \models \nabla, \overline{a} \# x : s$ and by induction that $[\![M]\!]_{\eta[x\mapsto d]} \Downarrow$ and $[\![M]\!]_{\eta[x\mapsto d]} \in [\![s']\!]$. Hence, we have that $[\![\lambda^{\overline{a}} x : s.\, M]\!]_\eta \Downarrow$, with interpretation $\Lambda d \in [\![s]\!]^{\#\overline{a}}.[\![M]\!]_{\eta[x\mapsto d]}$. This means that $[\![M]\!]_{\eta[x\mapsto d]}$ is a function from $[\![s]\!]^{\#\overline{a}}$ to $[\![s']\!]$. Next, we have to demonstrate that the function is finitely supported.

Let $c, c' \notin supp((s, \overline{a}, M, \eta))$.

$$
\begin{aligned}
& (c\,c') \cdot [\![\lambda^{\overline{a}} x : s.\, M]\!]_\eta \\
\stackrel{\text{def}}{=}\ & (c\,c') \cdot (\Lambda d \in [\![s]\!]^{\#\overline{a}}.[\![M]\!]_{\eta[x \mapsto d]}) \\
\stackrel{\text{def}}{=}\ & \Lambda d \in (c\,c') \cdot [\![s]\!]^{\#\overline{a}}.(c\,c') \cdot [\![M]\!]_{\eta[x \mapsto (c\,c') \cdot d]} \\
\stackrel{\text{def}}{=}\ & \Lambda d \in [\![(c\,c') \cdot s]\!]^{\#(c\,c') \cdot \overline{a}}.(c\,c') \cdot [\![M]\!]_{\eta[x \mapsto (c\,c') \cdot d]} \\
=\ & \Lambda d \in [\![(c\,c') \cdot s]\!]^{\#(c\,c') \cdot \overline{a}}.[\![(c\,c') \cdot M]\!]_{((c\,c') \cdot \eta)[x \mapsto d]} && \text{(Lemma 3.4.7)} \\
=\ & \Lambda d \in [\![s]\!]^{\#\overline{a}}.[\![M]\!]_{\eta[x \mapsto d]} && (c, c' \,\#\, (s, \overline{a}, M, \eta)) \\
\stackrel{\text{def}}{=}\ & [\![\lambda^{\overline{a}} x : s.\, M]\!]_\eta
\end{aligned}
$$

Hence, we have that $[\![\lambda^{\overline{a}} x : s.\, M]\!]_\eta \in [\![s]\!]^{\#\overline{a}} \Rightarrow_{fs} [\![s']\!] \stackrel{\text{def}}{=} [\![s^{\overline{a}} \Rightarrow s']\!]$.

(AP): Suppose $\nabla \vdash^{\text{NLC}} F\ A : s'$. From this we can deduce that $\nabla \vdash^{\text{NLC}} F : s^{\overline{a}} \Rightarrow s'$ and $\nabla \vdash^{\text{NLC}} \overline{a} \,\#\, A : s$. Let $\eta \models \nabla$. We need to demonstrate that $[\![F\ A]\!]_\eta \Downarrow$ and $[\![F\ A]\!]_\eta \in \mathcal{M}[\![s']\!]$ holds.

By induction we obtain that $[\![F]\!]_\eta$ is defined and $[\![F]\!]_\eta \in [\![s^{\overline{a}} \Rightarrow s']\!] \stackrel{\text{def}}{=} [\![s]\!]^{\#\overline{a}} \Rightarrow_{fs} [\![s']\!]$. Hence, we have that $[\![F]\!]_\eta$ is a finitely supported function from $[\![s]\!]^{\#\overline{a}}$ to $[\![s']\!]$. What remains to be proven, considering the interpretation of an application, is that $[\![A]\!]_\eta$ exists, $[\![A]\!]_\eta \in [\![s]\!]$ and $\overline{a} \,\#\, [\![A]\!]_\eta$.

Before we prove these conditions, we recall that satisfaction of $\nabla \vdash^{\text{NLC}} \overline{a} \,\#\, A : s$ is defined to be the satisfaction of particular equations $(\mathit{И}\,\overline{c}')$ $(\nabla^{\#\overline{c}'} \vdash^{\text{NLC}} A \approx (\overline{a}\,\overline{c}') * A : s)$ for $\overline{c}' \,\#\, (\nabla, \overline{a}, A, s)$ (see Remark 3.2.9). Then, by induction, we obtain that for all environments $\eta' \models \nabla^{\#\overline{c}'}$, we have that $[\![A]\!]_{\eta'} \Downarrow$, $[\![(\overline{a}\,\overline{c}') * A]\!]_{\eta'} \Downarrow$ and $[\![A]\!]_{\eta'} = [\![(\overline{a}\,\overline{c}') * A]\!]_{\eta'}$.

We pick $\overline{c} \,\#\, (\nabla, A, \overline{a}, \eta, s)$. From $\overline{c} \,\#\, \eta$ and $\eta \models \nabla$ we can directly deduce that $\eta \models \nabla^{\#\overline{c}}$. Given that $\overline{c} \,\#\, (\nabla, A, \overline{a})$, and $\eta \models \nabla^{\#\overline{c}}$ we can deduce by induction, as shown above, that $[\![A]\!]_\eta \Downarrow$, $[\![A]\!]_\eta \in [\![s]\!]$ and $[\![A]\!]_\eta = [\![(\overline{a}\,\overline{c}) * A]\!]_\eta$. Then, by Lemma 3.4.6, we get $[\![(\overline{a}\,\overline{c}) * A]\!]_\eta = (\overline{a}\,\overline{c}) \cdot [\![A]\!]_\eta$. Thus, we have $[\![A]\!]_\eta = (\overline{a}\,\overline{c}) \cdot [\![A]\!]_\eta$ and therefore $\overline{a} \,\#\, [\![A]\!]_\eta$ holds. Hence, we have completed this case.

The Property closure for the equation rules in Table 3.5 is trivial for **(REF)**, **(SYM)**, **(TRANS)** and **(WEAK)**. The existence part for the congruence rules **(CL)** and **(CA)**

follows similarly to **(ABS)** and **(AP)** and the equational part follows by standard computations.

(B): Suppose $\nabla \vdash^{\text{NLC}} (\lambda^{\overline{a}} x : s.\, M)\ N \approx M\{N/x\} : s'$. We consider the case where x does not occur in context ∇. From this we can deduce that $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} M : s'$ and $\nabla \vdash^{\text{NLC}} \overline{a} \# N : s$. Let $\eta \models \nabla$. Then, by induction, reasoning similar to (ABS), we obtain $[\![\lambda^{\overline{a}} x : s.\, M]\!]_\eta \Downarrow$ and $[\![\lambda^{\overline{a}} x : s.\, M]\!]_\eta \in [\![s^{\overline{a}} \Rightarrow s']\!] = [\![s]\!]^{\#\overline{a}} \Rightarrow_{fs} [\![s']\!]$. Further, similar to (AP), we can deduce by induction on $\nabla \vdash^{\text{NLC}} \overline{a} \# N : s$ that $[\![N]\!]_\eta \Downarrow$, $[\![N]\!]_\eta \in [\![s]\!]$ and $\overline{a} \# [\![N]\!]_\eta$, and therefore $[\![N]\!]_\eta \in [\![s]\!]^{\#\overline{a}}$. From this, it immediately follows that $[\![(\lambda^{\overline{a}} x : s.\, M)\ N]\!]_\eta \Downarrow$, which is $[\![\lambda^{\overline{a}} x : s.\, M]\!]_\eta\ ([\![N]\!]_\eta)$.

From $[\![\lambda^{\overline{a}} x : s.\, M]\!]_\eta \Downarrow$, we obtain by definition that for all $d \in [\![s]\!]^{\#\overline{a}}$, it is the case that $[\![M]\!]_{\eta[x \mapsto d]} \Downarrow$. Due to the fact that $[\![N]\!]_\eta \in [\![s]\!]^{\#\overline{a}}$, we obtain $[\![M]\!]_{\eta[x \mapsto [\![N]\!]_\eta]}$ and by applying Lemma 3.4.8 we get $[\![M\{N/x\}]\!]_\eta \Downarrow$ and $[\![M]\!]_{\eta[x \mapsto [\![N]\!]_\eta]} = [\![M\{N/x\}]\!]_\eta$ ($\diamond 1$). The equation now follows immediately:

$$\begin{aligned} [\![(\lambda^{\overline{a}} x : s.\, M)\ N]\!]_\eta &\stackrel{\text{def}}{=} [\![\lambda^{\overline{a}} x : s.\, M]\!]_\eta\ ([\![N]\!]_\eta) \\ &\stackrel{\text{def}}{=} [\![M]\!]_{\eta[x \mapsto [\![N]\!]_\eta]} \\ &= [\![M\{N/x\}]\!]_\eta \qquad (\diamond 1) \end{aligned}$$

(E): Suppose $\nabla \vdash^{\text{NLC}} \lambda^{\overline{a}} x : s.\, (M\ x) \approx M : s^{\overline{a}} \Rightarrow s'$. We consider the case $x \notin dom(\nabla)$. From this we can deduce that $\nabla \vdash^{\text{NLC}} M : s^{\overline{a}} \Rightarrow s'$. Let $\eta \models \nabla$. Then, by induction, we obtain $[\![M]\!]_\eta \Downarrow$ and $[\![M]\!]_\eta \in [\![s^{\overline{a}} \Rightarrow s']\!] = [\![s]\!]^{\#\overline{a}} \Rightarrow_{fs} [\![s']\!]$. We now need to demonstrate that $[\![\lambda^{\overline{a}} x : s.\, (M\ x)]\!]_\eta \Downarrow$. Let $d \in [\![s]\!]^{\#\overline{a}}$. We have to show that $[\![M\ x]\!]_{\eta[x \mapsto d]} \Downarrow$. Given that $x \notin fv(M)$ and $[\![M]\!]_\eta \Downarrow$, we can apply Lemma 3.4.3 to obtain that $[\![M]\!]_{\eta[x \mapsto d]} \Downarrow$ and

$$[\![M]\!]_\eta = [\![M]\!]_{\eta[x \mapsto d]} \ (\diamond 2)$$

We directly obtain $[\![x]\!]_{\eta[x \mapsto d]} \Downarrow$, it is d. Given that $d \in [\![s]\!]^{\#\overline{a}}$ and $[\![M]\!]_\eta \in [\![s]\!]^{\#\overline{a}} \Rightarrow_{fs}$

$[\![s']\!]$, $[\![M\ x]\!]_{\eta[x\mapsto d]}\Downarrow$ follows by definition. The equation can be deduced as follows:

$$\begin{aligned}[\![\lambda^{\overline{a}}x : s.\,(M\ x)]\!]_\eta(d) &\stackrel{\text{def}}{=} [\![M\ x]\!]_{\eta[x\mapsto d]} \\ &\stackrel{\text{def}}{=} [\![M]\!]_{\eta[x\mapsto d]}\ ([\![x]\!]_{\eta[x\mapsto d]}) \\ &= [\![M]\!]_\eta\ (d) \qquad\qquad (\diamond 2)\end{aligned}$$

(AE): Suppose $\nabla \vdash^{\text{NLC}} M \approx M' : s$ and $\overline{a} \# (\nabla, M, M')$. From this we can deduce that $\nabla^{\#\overline{a}} \vdash^{\text{NLC}} M \approx M' : s$. Let $\eta \models \nabla$. We have to show that $[\![M]\!]_\eta\Downarrow$, $[\![M']\!]_\eta\Downarrow$ and $[\![M]\!]_\eta = [\![M']\!]_\eta$.

Pick $\overline{c} \# (\overline{a}, \eta, \nabla, M, M')$. We can now deduce that $\overline{a} \# (\overline{c}\,\overline{a})\cdot\eta$. So, by induction, we have that $[\![M]\!]_{(\overline{c}\,\overline{a})\cdot\eta}\Downarrow$, $[\![M']\!]_{(\overline{c}\,\overline{a})\cdot\eta}\Downarrow$ and $[\![M]\!]_{(\overline{c}\,\overline{a})\cdot\eta} = [\![M']\!]_{(\overline{c}\,\overline{a})\cdot\eta}$ $(\diamond)$.

$$\begin{aligned}[\![M]\!]_\eta &= [\![(\overline{c}\,\overline{a})\cdot M]\!]_\eta && (\overline{c}, \overline{a} \# M) \\ &= (\overline{c}\,\overline{a})\cdot[\![M]\!]_{(\overline{c}\,\overline{a})\cdot\eta} && \text{(Lemma 3.4.7)} \\ &= (\overline{c}\,\overline{a})\cdot[\![M']\!]_{(\overline{c}\,\overline{a})\cdot\eta} && ((\diamond) \text{ permuted by } (\overline{c}\,\overline{a})) \\ &= [\![(\overline{c}\,\overline{a})\cdot M']\!]_\eta && \text{(Lemma 3.4.7)} \\ &= [\![M']\!]_\eta && (\overline{c}, \overline{a} \# M')\end{aligned}$$

(SUSP): Suppose $ds(\pi,\pi') \# x : s \vdash^{\text{NLC}} \pi x \approx \pi' x : \pi\cdot s$ and $\eta \models (ds(\pi,\pi') \# x : s)$. We can now directly deduce that $[\![\pi x]\!]_\eta\Downarrow$ and $[\![\pi' x]\!]_\eta\Downarrow$, with the respective interpretations $\pi\cdot\eta(x)$ and $\pi'\cdot\eta(x)$. We recall that $ds(\pi,\pi') \stackrel{\text{def}}{=} \{a \in \mathbb{A} \mid \pi(a) \neq \pi'(a)\}$ and $ds(\pi,\pi') = supp(\pi^{-1}\pi')$. By definition of $\eta \models (ds(\pi,\pi') \# x : s)$ we have $ds(\pi,\pi') \# \eta(x)$, and therefore $supp(\pi^{-1}\pi') \# \eta(x)$. From this we can directly deduce that $\pi^{-1}\pi'\cdot\eta(x) = \eta(x)$, and furthermore by applying π on both sides that $\pi'\cdot\eta(x) = \pi\cdot\eta(x)$. □

Definition 3.4.14 *Given $\nabla \vdash^{\text{NLC}} \overline{a} \# M : s$ we say that $\mathcal{M}$ **satisfies** the freshness assertion if $\mathcal{M}[\![M]\!]_\eta\Downarrow$ and $\mathcal{M}[\![M]\!]_\eta \in \mathcal{M}[\![s]\!]^{\#\overline{a}}$.*

Corollary 3.4.15 (Freshness Soundness) *Let Th be an* NLC*-theory and $\mathcal{M}$ a model. Then every freshness assertion $Th \rhd \nabla \vdash^{\text{NLC}} \overline{a} \# M : s$ is satisfied in $\mathcal{M}$.*

Proof Suppose $\nabla \vdash^{\text{NLC}} \overline{a} \# M : s$. Then, by definition, we have that $(\mathsf{N}\,\overline{c})\ \nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * M \approx M : s$. Let $\eta \models \nabla$. We have to show that $[\![M]\!]_\eta \Downarrow$ and $[\![M]\!]_\eta \in [\![s]\!]^{\#\overline{a}}$. By definition, the latter can be proved by showing that $[\![M]\!]_\eta \in [\![s]\!]$ and $\overline{a} \# [\![M]\!]_\eta$.

We pick $\overline{c} \# (\eta, \nabla, \overline{a}, M)$. Given that $\overline{c} \# (\eta, \nabla)$, we can deduce that $\eta \models \nabla^{\#\overline{c}}$. Then, due to $\overline{c} \# (\nabla, \overline{a}, M)$, we have that $\nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{c}\,\overline{a}) * M \approx M : s$ and by Theorem 3.4.13 we obtain $[\![(\overline{c}\,\overline{a}) * M]\!]_\eta \Downarrow$, $[\![M]\!]_\eta \Downarrow$, $[\![M]\!]_\eta \in [\![s]\!]$ and $[\![(\overline{c}\,\overline{a}) * M]\!]_\eta = [\![M]\!]_\eta$. By Lemma 3.4.6 we have $[\![(\overline{c}\,\overline{a}) * M]\!]_\eta = (\overline{c}\,\overline{a}) \cdot [\![M]\!]_\eta$, and therefore $(\overline{c}\,\overline{a}) \cdot [\![M]\!]_\eta = [\![M]\!]_\eta$. Hence, $\overline{a} \# [\![M]\!]_\eta$ and given that $[\![M]\!]_\eta \in [\![s]\!]$, we have completed the argument. □

3.5 Conservative Extension Results

Having defined NLC with NEL and $\lambda^{\to}$ in mind, we now demonstrate that NLC is actually a conservative extension of NEL and $\lambda^{\to}$. We recall that there are two standard routes towards a conservative extension result, namely either by using confluence of the $\beta\eta$-reduction system or by a semantic model construction. Given that we have already obtained all the results required for a semantic model construction, we will pursue this route to prove conservativity of NLC with respect to NEL. The proof for $\lambda^{\to}$ is not shown in detail, but follows analogously to NEL.

Proposition 3.5.1 (Conservative Extension of NEL**)** *Let $Th = (Sg, Ax)$ be a* NEL *theory. Then for any freshness context ∇, type (sort) $s \in Sort_{Sg}$ and* NEL *terms M, M' we have that*

$$Sg \rhd \nabla^{:} \vdash^{NEL} M : s \iff Sg' \rhd \nabla \vdash^{\text{NLC}} M : s$$

$$Th \rhd \nabla \vdash^{NEL} M \approx M' : s \iff Th' \rhd \nabla \vdash^{\text{NLC}} M \approx M' : s$$

where $Th' \stackrel{\text{def}}{=} (Sg', Ax')$ is the direct extension of Th to an NLC*-theory.*

Proof Let Sg be a NEL-signature. The direct extension of Sg to an NLC-signature Sg' is defined as follows: $Gnd_{Sg'} \stackrel{\text{def}}{=} Sort_{Sg}$ and $Fun_{Sg'} \stackrel{\text{def}}{=} Op_{Sg}$. Then, for any

$f \in Fun_{Sg'}$ we have by definition that $f \in Op_{Sg}$. Suppose that $f : s_1 \times ... \times s_n \to s$ is the typing assigned to f using the typing function of Sg. So, the typing function of Sg' is defined to map f to the typing $f : s_1 \Rightarrow ... \Rightarrow s_n \Rightarrow s$. It follows by definition that $Gnd_{Sg'}$ and $Fun_{Sg'}$ are nominal sets and the typing function of Sg' is equivariant. Hence, we have that Sg' is an NLC-signature. Moreover, given that an equation-in-context of Ax is also an equation-in-context in Ax', we take $Ax' \stackrel{\text{def}}{=} Ax$, and therefore Th' is an NLC-theory.

The left-to-right direction for well typed expressions, i.e. $Sg \rhd \nabla^{:} \vdash^{\text{NEL}} M : s$ implies $Sg' \rhd \nabla \vdash^{\text{NLC}} M : s$, follows immediately by induction on $\nabla^{:} \vdash^{\text{NEL}} M : s$. Further, the left-to-right direction for equations is proved by induction on the derivation of $Th \rhd \nabla \vdash^{\text{NEL}} M \approx M' : s$:

For rule (REF) of NEL, we suppose that $\nabla \vdash^{\text{NEL}} M \approx M : s$. This only holds if $\nabla^{:} \vdash^{\text{NEL}} M : s$. Applying the previous result, we have that $Sg' \rhd \nabla \vdash^{\text{NLC}} M : s$ and by rule (REF) of NLC we obtain that $Th' \rhd \nabla \vdash^{\text{NLC}} M \approx M : s$ holds. For all the remaining rules, apart from (SSUB) of NEL we have that NLC has identical rules. Due to the fact that we can derive the rule (SSUB) in NLC, we have completed the left-to-right direction.

To prove the converse directly we use a semantic model construction: We first demonstrate that every Sg-structure $\mathcal{A}$ can be extended to a Sg'-structure $\mathcal{M}_{\mathcal{A}}$ such that for any $\nabla^{:} \vdash^{\text{NEL}} M : s$ and $\eta \models \nabla$ we have that if $\mathcal{M}_{\mathcal{A}}[\![M]\!]_{\eta}\Downarrow$, then $\mathcal{A}[\![M]\!]_{\eta} = \mathcal{M}_{\mathcal{A}}[\![M]\!]_{\eta}$ ($\diamond$). The extended Sg'-structure $\mathcal{M}_{\mathcal{A}}$ is defined as follows:

$$\mathcal{M}_{\mathcal{A}}[\![\gamma]\!] \stackrel{\text{def}}{=} \mathcal{A}[\![\gamma]\!]$$
$$\mathcal{M}_{\mathcal{A}}[\![f]\!] \stackrel{\text{def}}{=} curry(\mathcal{A}[\![f]\!])$$

For any base type γ, we have that $\mathcal{M}_{\mathcal{A}}[\![\gamma]\!]$ is an FM-Set, because $\mathcal{A}[\![\gamma]\!]$ is an FM-Set. Next, we show that the condition for the function map holds: Suppose that $f \in Fun_{Sg'}$ with $f : s_1 \Rightarrow ... \Rightarrow s_n \Rightarrow s$. By definition of Sg' we have that $f \in Op_{Sg}$ with $f : s_1 \times ... \times s_n \to s$. Hence, we have that $\mathcal{A}[\![f]\!] : \mathcal{A}[\![s_1]\!] \times ... \times \mathcal{A}[\![s_n]\!] \to \mathcal{A}[\![s]\!]$.

Next, by definition, we have that $\mathcal{M}_{\mathcal{A}}[\![f]\!] = curry(\mathcal{A}[\![f]\!])$ and therefore $\mathcal{M}_{\mathcal{A}}[\![f]\!] \in \mathcal{A}[\![s_1]\!] \Rightarrow ... \Rightarrow \mathcal{A}[\![s_n]\!] \Rightarrow \mathcal{A}[\![s]\!]$. Further, by definition we obtain that $\mathcal{M}_{\mathcal{A}}[\![f]\!] \in \mathcal{M}_{\mathcal{A}}[\![s_1 \Rightarrow ... \Rightarrow s_n \Rightarrow s]\!]$. The equivariants of both maps follows directly from the equivariants of the corresponding maps in $\mathcal{A}$ and therefore we have shown that $\mathcal{M}_{\mathcal{A}}$ is an Sg'-structure in *FMSet*.

We now prove that property ($\diamond$) holds: Let $Sg \rhd \nabla \vdash^{\text{NEL}} M : s$ and $\eta \models \nabla$. Under the assumption that $\mathcal{M}_{\mathcal{A}}[\![M]\!]_\eta\Downarrow$, it can be shown by induction on the structure of M that $\mathcal{A}[\![M]\!]_\eta = \mathcal{M}_{\mathcal{A}}[\![M]\!]_\eta$:

SUSP: (M is πx)

$$\mathcal{A}[\![\pi x]\!]_\eta \stackrel{\text{def}}{=} \pi \cdot x \stackrel{\text{def}}{=} \mathcal{M}_{\mathcal{A}}[\![\pi x]\!]_\eta$$

OP: (M is $f\ M_1\ ...\ M_n$)

$$\begin{aligned}
\mathcal{A}[\![fM_1...M_n]\!]_\eta &\stackrel{\text{def}}{=} \mathcal{A}[\![f]\!](\mathcal{A}[\![M_1]\!]_\eta, ..., \mathcal{A}[\![M_n]\!]_\eta) \\
&\stackrel{\text{def}}{=} curry(\mathcal{A}[\![f]\!])\ \mathcal{A}[\![M_1]\!]_\eta, ..., \mathcal{A}[\![M_n]\!]_\eta \\
&= curry(\mathcal{A}[\![f]\!])\ \mathcal{M}_{\mathcal{A}}[\![M_1]\!]_\eta, ..., \mathcal{M}_{\mathcal{A}}[\![M_n]\!]_\eta \qquad \text{(induction)} \\
&\stackrel{\text{def}}{=} \mathcal{M}_{\mathcal{A}}[\![f]\!]\ \mathcal{M}_{\mathcal{A}}[\![M_1]\!]_\eta, ..., \mathcal{M}_{\mathcal{A}}[\![M_n]\!]_\eta \\
&\stackrel{\text{def}}{=} \mathcal{M}_{\mathcal{A}}[\![fM_1...M_n]\!]_\eta
\end{aligned}$$

Suppose that $\mathcal{A}$ is a Th-algebra ($\mathcal{A}$ satisfies all axioms in Ax). We now demonstrate that $\mathcal{M}_{\mathcal{A}}$ is a model of Th' ($\mathcal{M}_{\mathcal{A}}$ satisfies all axioms in Ax'). Suppose that $\nabla \vdash^{\text{NLC}} M \approx M' : s$ is an axiom in Ax' and $\eta \models \nabla$. By definition we have that $\nabla \vdash^{\text{NEL}} M \approx M' : s$ is an axiom in Ax. Due to the fact that $\mathcal{A}$ is a Th-algebra we have that $\mathcal{A}[\![M]\!]_\eta = \mathcal{A}[\![M']\!]_\eta$. Recalling that we are limited to NEL terms, we clearly have that $\mathcal{M}_{\mathcal{A}}[\![M]\!]_\eta\Downarrow$ and $\mathcal{M}_{\mathcal{A}}[\![M']\!]_\eta\Downarrow$. We can now apply property ($\diamond$) to deduce that $\mathcal{M}_{\mathcal{A}}$ satisfies $\nabla \vdash^{\text{NLC}} M \approx M' : s$.

We continue by proving the right-to-left direction for equations derived in NLC. We suppose that $Th' \rhd \nabla \vdash^{\text{NLC}} M \approx M' : s$ holds. To show that $Th \rhd \nabla \vdash^{\text{NEL}}$

$M \approx M' : s$ holds we apply the deductive completeness result for NEL ([13] Theorem 5.4.7 and Theorem 5.5.8): In particular, we prove that for any Th-Algebra $\mathcal{A}$ and environment $\eta \models \nabla$ we have that $\mathcal{A}[\![M]\!]_\eta = \mathcal{A}[\![M']\!]_\eta$ holds.

We can deduce, as shown above, that there exists a model $\mathcal{M}_\mathcal{A}$ for Th' such that $\mathcal{A}[\![M]\!]_\eta = \mathcal{M}_\mathcal{A}[\![M]\!]_\eta$ ($\diamond 1$) and $\mathcal{A}[\![M']\!]_\eta = \mathcal{M}_\mathcal{A}[\![M']\!]_\eta$ ($\diamond 2$). Moreover, by soundness of NLC, we have that $\mathcal{M}_\mathcal{A}[\![M]\!]_\eta = \mathcal{M}_\mathcal{A}[\![M']\!]_\eta$ ($\diamond 3$). We can then deduce that

$$\begin{aligned} \mathcal{A}[\![M]\!]_\eta &= \mathcal{M}_\mathcal{A}[\![M]\!]_\eta && (\diamond 1) \\ &= \mathcal{M}_\mathcal{A}[\![M']\!]_\eta && (\diamond 3) \\ &= \mathcal{A}[\![M']\!]_\eta && (\diamond 2) \end{aligned}$$

This concludes the argument for equations. For derived expressions in NLC we suppose that $Sg' \rhd \nabla \vdash^{\text{NLC}} M : s$. By (REF) we have that $Th' \rhd \nabla \vdash^{\text{NLC}} M \approx M : s$ and therefore by the argument above we directly obtain that $Th \rhd \nabla \vdash^{\text{NEL}} M \approx M : s$. Hence, by definition of (REF) for NEL we can deduce that $Sg \rhd \nabla^{:} \vdash^{\text{NEL}} M : s$ holds. □

Proposition 3.5.2 (Conservative Extension of $\lambda^\to$) *Let $Th = (Sg, Ax)$ be a $\lambda^\to$ theory. Then for any typing context Γ, type $s \in Type_{Sg}$ and $\lambda^\to$ expressions M, M' we have that*

$$Sg \rhd \Gamma \vdash^{LC} M : s \iff Sg' \rhd \Gamma^* \vdash^{\text{NLC}} M : s$$
$$Th \rhd \Gamma \vdash^{LC} M \approx M' : s \iff Th' \rhd \Gamma^* \vdash^{\text{NLC}} M \approx M' : s$$

where $Th' \stackrel{\text{def}}{=} (Sg', Ax')$ is the extension of Th to an NLC*-theory and Γ^* trivially extends Γ to a freshness context.*

Proof The prove follows analogue to the argument above, but here we rely on the fact that $\lambda^\to$ is complete with respect to cartesian closed categories and the categorical soundness argument for NLC. □

There is yet another conservative extension property that we are interested in, namely to demonstrate that NLC provides a more expressive syntax, but if one is limited to ground types, this does not increase the computational power of the original theory *Th* in NEL.

Conjecture 3.5.3 (Conservative Extension) *Let* $Th = (Sg, Ax)$ *be a* NEL *theory and* $Th' \stackrel{\text{def}}{=} (Sg', Ax')$ *the direct extension of Th to an* NLC*-theory with* $Ax' \stackrel{\text{def}}{=} Ax$ *and* Sg' *being the direct extension of the* NEL*-signature Th. Then for any freshness context* $\nabla \stackrel{\text{def}}{=} \overline{a}_1 \# x_1 : \gamma_1, ..., \overline{a}_n \# x_n : \gamma_n$ *(with* $\gamma_i \in Gnd_{Sg}$*), ground type* $\gamma \in Gnd_{Sg}$ *and* NLC*-expression E such that* $Th' \rhd \nabla \vdash^{\text{NLC}} E : \gamma$ *we can deduce the following:*

(i) there exists a NEL*-term M for which* $Th \rhd \nabla \vdash^{NEL} M : \gamma$ *and* $Th' \rhd \nabla \vdash^{\text{NLC}} E \approx M : \gamma$ *(informally: E is* $\beta\eta$*-reducible to M).*

(ii) if there exists another NEL*-term* M' *for which* $Th \rhd \nabla \vdash^{NEL} M' : \gamma$ *and* $Th' \rhd \nabla \vdash^{\text{NLC}} E \approx M' : \gamma$ *then* $Th' \rhd \nabla \vdash^{NEL} M \approx M' : \gamma$.

Our original aim was to prove this result using a "nominal" version of categorical gluing as presented by Crole [21, 20]) for EL and $\lambda^{\rightarrow}$. For more details about such a result and first intermediate results we refer to Appendix B.1.

Chapter 4

NLC: Name Abstraction, Concretion and Local Fresh Atomic Names

In this chapter we extend NLC to capture the notions of name abstraction and concretion, as well as local fresh atomic names. These are standard constructions in the FM-set model, which we recalled in Section 2.2. The content of this chapter is organised as follows:

- The notion of concretion in the FM-set model is a partial function, which is restricted in its domain by a freshness condition. This coincides precisely with the notion of domain (freshness) restricted function types in NLC. In combination with the ability to use freshness assertions in typing rules, analogue to rule (AP), we can extend NLC to NLC[$\mathbb{A}$], a calculus which captures name abstraction and concretion as first class citizens.

- We introduce a small-step operational semantics for pure NLC[$\mathbb{A}$] and prove it to be sound, strongly normalising and confluent. This allows us to demonstrate that provable equality is decidable in NLC[$\mathbb{A}$].

- We introduce ИNLC, an extension of NLC[$\mathbb{A}$] that additionally captures the notion of local fresh atomic names. It is the precursor of a calculus that we will introduce in Chapter 5.

Note that the properties studied for NLC (up to soundness) can easily be extended to NLC[$\mathbb{A}$] and ИNLC. The proofs of these properties, mostly by induction, extend in a routine way. So, we decided to refrain from re-stating these properties or to extend all the respective proofs in this chapter. Instead we focus on the definition of the respective calculus and its model-theoretic semantics, as well as some of its key properties (e.g. soundness). Thus, if we reference auxiliary results of the previous chapter in the context of NLC[$\mathbb{A}$] or ИNLC, we implicitly refer to their extensions for the respective calculus. Other calculi that capture name abstraction and concretion are SNTT [9] and the $\lambda\alpha\nu$-calculus [55], which we will shortly discuss and compare to NLC[$\mathbb{A}$] in Chapter 8.

4.1 NLC[$\mathbb{A}$]: NLC with Name Abstraction and Concretion

We define NLC[$\mathbb{A}$] as an extension of NLC, which captures the notions of name abstraction and concretion. We remind the reader that in the FM-set model the set of name abstractions, $[\mathbb{A}]X$, is defined as a quotient structure using a generalised notion of α-equivalence. Further, it is a well known fact that name abstractions can be represented as finitely supported partial functions with names applied to them via concretion. Thus, name abstractions can be represented in two isomorphic ways, namely as equivalence classes or as partial functions.

The key aspect that has to be dealt with is that concretion is a partial function that can only be applied to arguments that meet certain freshness conditions. Considering that NLC provides domain (freshness) restricted function types and freshness

assertions in typing rules, it is not too surprising that concretion can be captured in NLC. Indeed, one might think that NEL would already be expressive enough to deal with such freshness conditions on the syntactic level, but as has been observed by Clouston [14], this is not the case: Suppose that $con_a : Name.s \to s$ is a concretion function with a name abstraction type $Name.s$. Then $con_a\, M$ would only be well typed if $a \# M$ (in context), but NEL does not support such partiality. However, by restricting to structures with total concretion, a total concretion theory can already be defined in NEL [14].

In NLC we can now exploit the name-dependent type system, which yields a solution to generally capture name abstraction and concretion. More precisely, the type system of NLC allows us to introduce name indexed constants con_a which are assigned the type $(Name.s)^a \Rightarrow s$. Thus, by extending the total concretion theory provided in [14] we obtain a general concretion theory for NLC.

We remind the reader that the notions of name abstraction and concretion, as defined in Section 2.2, were introduced in the nominal set model, whereas the model-theoretic semantics of NLC is using the FM-set model. However, this is not an issue, as previously pointed out, because these notions can equally be defined in the FM-set model, with minor changes that will be pointed out in due course. For a detailed account of the FM-set model we refer to [33, 35].

The Meta-Theory of NLC[$\mathbb{A}$] Raw Terms

We define an NLC[$\mathbb{A}$]-signature Sg to be an NLC-signature that is extended with a **name abstraction type**, which we denote by $Name.s$. The corresponding permutation action for $Type_{Sg}$ is then extended as follows:

$$\pi \cdot Name.s \stackrel{\text{def}}{=} Name.\pi \cdot s$$

For the newly introduced name abstraction type, we have an introduction form $\langle a \rangle M$ (non-binding) and an elimination form $M \mathbin{@} a$, referred to as **name abstrac-**

tion and **concretion**, respectively. To emphasise the direct correspondence between syntactic and semantic notions of name abstraction and concretion, we abuse the notation. The set of raw terms, $Term_{Sg}$, is extended as follows:

$$M ::= ... \mid \langle a \rangle M \mid M \mathbin{@} b$$

The recursively defined auxiliary functions $|M|$, $var(M)$, $fv(M)$ and $name(M)$ of NLC extend trivially. The permutation actions on raw terms for NLC extend as shown in Table 4.1.

(i) $\mu \bullet (\langle a \rangle M) \stackrel{\text{def}}{=} \langle a \rangle \mu \bullet M$	(i) $\pi \cdot (\langle a \rangle M) \stackrel{\text{def}}{=} \langle \pi \cdot a \rangle \pi \cdot M$
(ii) $\mu \bullet (M \mathbin{@} b) \stackrel{\text{def}}{=} \mu \bullet M \mathbin{@} b$	(ii) $\pi \cdot (M \mathbin{@} b) \stackrel{\text{def}}{=} \pi \cdot M \mathbin{@} \pi \cdot b$
Variable Swapping	Meta-Level
(i) $\pi * (\langle a \rangle M) \stackrel{\text{def}}{=} \langle \pi \cdot a \rangle \pi * M$	
(ii) $\pi * (M \mathbin{@} b) \stackrel{\text{def}}{=} \pi * M \mathbin{@} \pi \cdot b$	
Object-Level	

Table 4.1: Permutation Actions for NLC[$\mathbb{A}$]

Given that $\langle a \rangle M$ is non-binding the notion of α-equivalence directly extends over the newly added term formers. Further, as in the case of NLC, all three permutation actions can be shown to respect α-equivalence and can therefore be lifted to expressions:

$$\mu \bullet [M]_\alpha \stackrel{\text{def}}{=} [\mu \bullet M]_\alpha$$
$$\pi \cdot [M]_\alpha \stackrel{\text{def}}{=} [\pi \cdot M]_\alpha$$
$$\pi * [M]_\alpha \stackrel{\text{def}}{=} [\pi * M]_\alpha$$

with $supp([M]_\alpha) = fv(M)$ and $supp([M]_\alpha) = name(M)$. The definition of capture avoiding substitution is extended as follows:

$$(\langle a\rangle M)\{N/x\} \stackrel{\text{def}}{=} \langle a\rangle(M\{N/x\})$$
$$(M \mathbin{@} b)\{N/x\} \stackrel{\text{def}}{=} (M\{N/x\}) \mathbin{@} b$$

Typed Expressions and Equational Theories of NLC[$\mathbb{A}$]

We continue by introducing type and equation rules for NLC[$\mathbb{A}$], which allow us to capture the notions of name abstraction and concretion and their corresponding properties as presented in Section 2.2 (see Table 4.2 and Table 4.3). With the function space interpretation of $[\mathbb{A}]X$ in mind, it is not surprising that the rules for name abstraction and concretion are rather similar to lambda abstraction and application.

$$(\mathsf{NABS})\ \frac{\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s \qquad a \in \mathbb{A}}{\nabla \vdash^{\text{NLC}[\mathbb{A}]} \langle a\rangle M : Name.s}\,(a \# s) \qquad (\mathsf{CONC})\ \frac{\nabla \vdash^{\text{NLC}[\mathbb{A}]} b \# M : Name.s}{\nabla \vdash^{\text{NLC}[\mathbb{A}]} M \mathbin{@} b : s}$$

Table 4.2: NLC[$\mathbb{A}$] Typing Rules

Note that in the FM-set model the definition of a name abstraction set, $[\mathbb{A}]X$, additionally requires that for a name abstraction $\langle a\rangle x \in [\mathbb{A}]X$, the side condition $a \# X$ holds (see Definition 5.4 in [33]). This ensures that the definition of the concretion function is well defined. In rule (NABS), this condition is directly reflected in the side condition $a \# s$, which becomes obvious when we consider that the interpretation map on types is defined to be equivariant for Sg-structures of NLC[$\mathbb{A}$]. The key aspect that needs to be consider is the partiality of concretion, but this can easily be expressed by a freshness assertion. Hence, (CONC) is defined analogue to (AP).

We continue with the equation rules in Table 4.3. (BNABS) is a binding rule for name abstraction and directly captures Definition 2.2.8. The И-quantifier stands for (some/any) name b such that $b \# (a', a'', \nabla, M', M''))$. Further, (BN) captures the

definition of concretion and (EN) captures Lemma 2.2.13. Considering the analogy of lambda abstraction/application and name abstraction/concretion, it is not surprising that the rules (BN) and (EN) are rather similar to (B) and (E), respectively.

$$(\mathsf{BNABS})\ \frac{(\not\exists\, b) \qquad \nabla^{\#b} \vdash^{\mathrm{NLC}[\mathbb{A}]} (b\,a') * M' \approx (b\,a'') * M'' : s}{\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \langle a' \rangle M' \approx \langle a'' \rangle M'' : Name.s}\ (a', a'' \# s)$$

$$(\mathsf{BN})\ \frac{\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} b \# \langle a \rangle M : Name.s}{\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} (\langle a \rangle M) \mathbin{@} b \approx (a\,b) * M : s}$$

$$(\mathsf{EN})\ \frac{\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} a \# M : Name.s}{\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \langle a \rangle (M \mathbin{@} a) \approx M : Name.s}$$

$$(\mathsf{CC})\ \frac{\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M_1 \approx M_2 : Name.s \qquad \nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} b \# M_i : Name.s}{\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M_1 \mathbin{@} b \approx M_2 \mathbin{@} b : s}$$

Table 4.3: NLC[$\mathbb{A}$] Equation Rules

Lemma 4.1.1 *The following rule, referred to as (CNA), is admissible.*

$$\frac{\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M \approx M' : s}{\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \langle a \rangle M \approx \langle a \rangle M' : Name.s}\ (a \# s)$$

Proof Suppose $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M \approx M' : s$ and $a \# s$. We have to show that $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \langle a \rangle M \approx \langle a \rangle M' : Name.s$. Pick $c \# (a, M, M', \nabla, s)$. By applying rule (BNABS), we can equivalently show that $\nabla^{\#c} \vdash^{\mathrm{NLC}[\mathbb{A}]} (c\,a) * M \approx (c\,a) * M' : s$

We apply Proposition 3.2.5 on $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M \approx M' : s$ to obtain $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} (c\,a) * M \approx (c\,a) * M' : (c\,a) \cdot s$. Given that $a, c \# s$ and by an instance of rule (WEAK), we have completed the argument. □

Lemma 4.1.2 (Binding Axiom) *The following rule is admissible.*

$$\frac{\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \langle a \rangle M : Name.s}{\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} a \# \langle a \rangle M : Name.s}$$

Proof Suppose $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \langle a\rangle M : \mathit{Name}.s$. From this we can deduce that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s$ and $a \# s$. We have to show that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} a \# \langle a\rangle M : \mathit{Name}.s$, i.e. we pick a name $c \# (\nabla, a, M, s)$ and demonstrate that $\nabla^{\#c} \vdash^{\text{NLC}[\mathbb{A}]} (c\,a)*\langle a\rangle M \approx \langle a\rangle M : \mathit{Name}.s$ holds. Given that $(c\,a) * \langle a\rangle M \stackrel{\text{def}}{=} \langle c\rangle(c\,a) * M$, we can apply an instance of rule (BNABS). So, we pick a name $c' \# (c, a, M, \nabla, s)$ and show that $\nabla^{\#c,c'} \vdash^{\text{NLC}[\mathbb{A}]} (c'\,c) * (c\,a) * M \approx (c'\,a) * M : s$ holds.

Given that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s$ and $c, c' \# (\nabla, M)$ we can apply the admissible rule (PERM) to obtain that $\nabla^{\#c,c'} \vdash^{\text{NLC}[\mathbb{A}]} (c'\,c) * M \approx M : s$. We then apply Proposition 3.2.5 to get $\nabla^{\#c,c'} \vdash^{\text{NLC}[\mathbb{A}]} (c'\,a) * (c'\,c) * M \approx (c'\,a) * M : (c'\,a) \cdot s$. Given that $c', a \# s$ holds, we have completed the argument. □

A Sound Model-theoretic Semantics of NLC[$\mathbb{A}$]

We extend the definition of a Sg-structure $\mathcal{M}$ for NLC by providing an interpretation for the name abstraction type:

$$\mathcal{M}[\![\mathit{Name}.s]\!] \stackrel{\text{def}}{=} [\mathbb{A}]\mathcal{M}[\![s]\!]$$

The partial interpretation of an NLC[$\mathbb{A}$] expressions M for an environment η (in a Sg-structure) is extended as follows:

$$\begin{aligned} \mathcal{M}[\![\langle a\rangle M]\!]_\eta &\stackrel{\text{def}}{=} \langle a\rangle \mathcal{M}[\![M]\!]_\eta && (\text{if } \mathcal{M}[\![M]\!]_\eta\Downarrow) \\ \mathcal{M}[\![M \mathbin{@} b]\!]_\eta &\stackrel{\text{def}}{=} \mathcal{M}[\![M]\!]_\eta \mathbin{@} b && (\text{if } \mathcal{M}[\![M]\!]_\eta\Downarrow \text{ and } b \# \mathcal{M}[\![M]\!]_\eta) \end{aligned}$$

Note that the interpretation of a concretion expression $M \mathbin{@} b$ is defined analogue to the interpretation of an application expression, namely by using a freshness side condition. The soundness proof is extended as follows:

Soundness Theorem for NLC[$\mathbb{A}$]**:** The proof by mutual induction for NLC is extended with the new type and equation rules:

(NABS): Suppose $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \langle a\rangle M : \mathit{Name}.s$. From this we can deduce that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s$ and $a \# s$. Let $\eta \models \nabla$. By induction we obtain that $[\![M]\!]_\eta\Downarrow$ and $[\![M]\!]_\eta \in [\![s]\!]$.

Then, by definition, we have that $[\![\langle a\rangle M]\!]_\eta \stackrel{\text{def}}{=} \langle a\rangle[\![M]\!]_\eta$. Hence, we directly obtain that $[\![\langle a\rangle M]\!]_\eta\Downarrow$ and $[\![\langle a\rangle M]\!]_\eta \in [\mathbb{A}][\![s]\!] \stackrel{\text{def}}{=} [\![Name.s]\!]$.

(CONC): Suppose $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M \mathbin{@} b : s$. From this we can deduce that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} b \# M : Name.s$. Let $\eta \models \nabla$. By induction, analogous to AP, we obtain that $[\![M]\!]_\eta\Downarrow$ and $[\![M]\!]_\eta \in [\![Name.s]\!]^{\#b}$. By definition we have $b \# [\![M]\!]_\eta$ and therefore the conditions of the partial interpretation function is meet. So, we have that $[\![M \mathbin{@} b]\!]_\eta \stackrel{\text{def}}{=} [\![M]\!]_\eta \mathbin{@} b$ exists and $[\![M \mathbin{@} b]\!]_\eta \in [\![s]\!]$.

(BNABS): Suppose $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \langle a'\rangle M' \approx \langle a''\rangle M'' : Name.s$. From this we can deduce that $a', a'' \# s$ and for (some/any) $b \# (a', a'', \nabla, M', M''))$ we have that $\nabla^{\#b} \vdash^{\text{NLC}[\mathbb{A}]} (b\,a') * M' \approx (b\,a'') * M'' : s$ holds. Let $\eta \models \nabla$. We have to show that $[\![\langle a'\rangle M']\!]_\eta\Downarrow$, $[\![\langle a'\rangle M']\!]_\eta\Downarrow$ and that both are equal.

Pick $b \# (a', a'', \nabla, M', M'', \eta))$. We can now deduce that $\eta \models \nabla^{\#b}$ and by induction on $\nabla^{\#b} \vdash^{\text{NLC}[\mathbb{A}]} (b\,a') * M' \approx (b\,a'') * M'' : s$ we obtain $[\![(b\,a') * M']\!]_\eta\Downarrow$, $[\![(b\,a'') * M'']\!]_\eta\Downarrow$ and $[\![(b\,a') * M']\!]_\eta = [\![(b\,a'') * M'']\!]_\eta$. Then, using Lemma 3.4.6, we get $[\![M']\!]_\eta\Downarrow$, $[\![M'']\!]_\eta\Downarrow$, $[\![(b\,a') * M']\!]_\eta = (b\,a') \cdot [\![M']\!]_\eta$ and $[\![(b\,a'') * M'']\!]_\eta = (b\,a'') \cdot [\![M'']\!]_\eta$. Hence, by definition, we have that $[\![\langle a'\rangle M']\!]_\eta\Downarrow$, $[\![\langle a'\rangle M']\!]_\eta\Downarrow$ and furthermore $(b\,a') \cdot [\![M']\!]_\eta = (b\,a'') \cdot [\![M'']\!]_\eta$. From $b \# (M', M'', \eta)$ we can deduce, using Lemma 3.4.7, that $b \# ([\![M']\!]_\eta, [\![M'']\!]_\eta)$. So, by definition 2.2.8, we have that $\langle a'\rangle[\![M']\!]_\eta = \langle a''\rangle[\![M'']\!]_\eta$, which is used to prove the following equation:

$$[\![\langle a'\rangle M']\!]_\eta \stackrel{\text{def}}{=} \langle a'\rangle[\![M']\!]_\eta = \langle a''\rangle[\![M'']\!]_\eta \stackrel{\text{def}}{=} [\![\langle a''\rangle M'']\!]_\eta$$

(BN): Suppose $\nabla \vdash^{\text{NLC}[\mathbb{A}]} (\langle a\rangle M) \mathbin{@} b \approx (a\,b) * M : s$. From this we can directly deduce that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} b \# \langle a\rangle M : Name.s$ and $a, b \# s$. Let $\eta \models \nabla$. By induction, similar to AP, we have $[\![\langle a\rangle M]\!]_\eta\Downarrow$ and $[\![\langle a\rangle M]\!]_\eta \in [\![Name.s]\!]^{\#b}$. So, by definition, we have $b \# [\![\langle a\rangle M]\!]_\eta$ ($\diamond 1$), $[\![M]\!]_\eta\Downarrow$ and $[\![M]\!]_\eta \in [\![s]\!]$. Hence, we have $(a\,b) \cdot [\![M]\!]_\eta\Downarrow$ and $(a\,b) \cdot [\![M]\!]_\eta \in (a\,b) \cdot [\![s]\!]$. By applying Lemma 3.2.5 and recalling that $a, b \# s$, we obtain $[\![(a\,b) * M]\!]_\eta\Downarrow$ and $[\![(a\,b) * M]\!]_\eta \in (a\,b) \cdot [\![s]\!] \stackrel{\text{def}}{=} [\![(a\,b) \cdot s]\!] = [\![s]\!]$. We can now

prove the equality as follows:

$$\begin{aligned}
[\![(\langle a\rangle M) \mathbin{@} b]\!]_\eta &\stackrel{\text{def}}{=} [\![\langle a\rangle M]\!]_\eta \mathbin{@} b \\
&\stackrel{\text{def}}{=} (\langle a\rangle [\![M]\!]_\eta) \mathbin{@} b \\
&\stackrel{\text{def}}{=} (a\,b) \cdot [\![M]\!]_\eta && (\diamond 1) \\
&= [\![(a\,b) * M]\!]_\eta && (\text{Lemma } 3.2.5)
\end{aligned}$$

(EN): Suppose $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \langle a\rangle(M \mathbin{@} a) \approx M : \mathit{Name}.s$. From this we can deduce that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} a \# M : \mathit{Name}.s$ and $a \# s$. Let $\eta \models \nabla$. By induction, similar to AP, we have $[\![M]\!]_\eta\Downarrow$ and $[\![M]\!]_\eta \in [\![\mathit{Name}.s]\!]^{\#a}$. So, by definition, we have $a \# [\![M]\!]_\eta$ $(\diamond 2)$. Next, using the definition of the interpretation function we have $[\![M \mathbin{@} a]\!]_\eta \stackrel{\text{def}}{=} [\![M]\!]_\eta \mathbin{@} a$ and furthermore $[\![\langle a\rangle(M \mathbin{@} a)]\!]_\eta \stackrel{\text{def}}{=} \langle a\rangle(M \mathbin{@} a)$. Hence, $[\![\langle a\rangle(M \mathbin{@} a)]\!]_\eta\Downarrow$ and the equality is obtained using $(\diamond 2)$ and Lemma 2.2.13:

$$[\![\langle a\rangle(M \mathbin{@} a)]\!]_\eta \stackrel{\text{def}}{=} \langle a\rangle [\![M \mathbin{@} a]\!]_\eta \stackrel{\text{def}}{=} \langle a\rangle [\![M]\!]_\eta \mathbin{@} a = [\![M]\!]_\eta$$

(CC): Suppose $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M_1 \mathbin{@} b \approx M_2 \mathbin{@} b : s$. Let $\eta \models \nabla$. The existence part follows similarly to CONC and the equations is proved as follows:

$$\begin{aligned}
[\![M_1 \mathbin{@} b]\!]_\eta &\stackrel{\text{def}}{=} [\![M_1]\!]_\eta \mathbin{@} b \\
&= [\![M_2]\!]_\eta \mathbin{@} b && (\text{induction}) \\
&\stackrel{\text{def}}{=} [\![M_2 \mathbin{@} b]\!]_\eta
\end{aligned}$$

□

4.2 Small-step Operational Semantics for NLC[$\mathbb{A}$]

We introduce a small-step operational semantics for pure NLC[$\mathbb{A}$] (without constants and axioms), or more precisely a pure $\beta\eta$-reduction system. We demonstrate that the reduction system is sound, strongly normalising and confluent. From these properties we can deduce that provable equality of NLC[$\mathbb{A}$] is decidable, and by reusing

and slightly adapting the algorithms for type checking and type inference of $\lambda^{\rightarrow}$ it can additionally be shown that both are decidable as well.

A $\beta\eta$-reduction system for NLC[$\mathbb{A}$] can be defined by giving directionality to most of the equation rules, apart from (SUSP), (BNABS) and (WEAK), where the α-equivalence relation $\equiv_\alpha$ needs to be extended instead. Due to the fact that (SUSP) requires a context, the structural equivalence relation will now be written as $\equiv_\alpha^\nabla$ (see Table 4.5). So, the corresponding one-step $\beta\eta$-reduction system would formally be defined on terms modulo $\equiv_\alpha^\nabla$. A notion of $\beta\eta$-conversion can be defined using the reflexive, symmetric and transitive closure of the one-step $\beta\eta$-reduction system. Once it has been demonstrated that the $\beta\eta$-reduction system is strongly normalising and confluent, $\beta\eta$-conversion can be shown to be decidable. In addition, given that $\beta\eta$-conversion coincides with provable equality in NLC[$\mathbb{A}$], it directly follows that provable equality is decidable as well.

Since we are mostly interested in decidability of provable equality, we favour a more direct proof argument without the detour over $\beta\eta$-conversion. This does not only present us with a shorter proof, but it also provides us with a clearer picture of what is required to prove strong normalisation and confluence for the $\beta\eta$-reduction system of NLC[$\mathbb{A}$]. So, for example, we observe that it suffices to define $\beta\eta$-reduction on raw terms modulo $\equiv_\alpha$ (expressions) to prove strong normalisation. However, to prove confluence we additionally require the structural equivalence relation $\equiv_\alpha^\nabla$.

To sum it up, we first introduce a pure $\beta\eta$-reduction system on expressions and prove it to be strongly normalising. Next, we introduce the notion of a structural relation $\equiv_\alpha^\nabla$ and demonstrate that the reduction system is confluent up to $\equiv_\alpha^\nabla$. We then conclude by using both properties to demonstrate that the notion of provable equality is decidable.

4.2.1 $\beta\eta$-Reduction System

We now formally define a $\beta\eta$-reduction system for the pure NLC[$\mathbb{A}$] calculus with a strong evaluation strategy (reductions may occur in any context). We begin with β-contraction, which is defined on expressions as follows:

- (R-BAPP): $(\lambda^{\overline{a}} x : s.\, M)\ N \to_\beta M\{N/x\}$
- (R-BCONC): $(\langle a \rangle M) \mathbin{@} b \to_\beta (a\, b) * M$

Lemma 4.2.1 *The contraction axioms (R-BAPP) and (R-BCONC) preserve types.*

Proof (R-BAPP) Suppose $\nabla \vdash^{\text{NLC}[\mathbb{A}]} (\lambda^{\overline{a}} x : s.\, M)\ N : s'$. From this we can deduce $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \lambda^{\overline{a}} x : s.\, M : s^{\overline{a}} \Rightarrow s'$ and $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \overline{a} \# N : s$, as well as $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}[\mathbb{A}]} M : s'$. We then apply Lemma 3.2.20 to obtain $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M\{N/x\} : s'$.

(R-BCONC) Suppose $\nabla \vdash^{\text{NLC}[\mathbb{A}]} (\langle a \rangle M) \mathbin{@} b : s$. From this we can deduce $\nabla \vdash^{\text{NLC}[\mathbb{A}]} b \# \langle a \rangle M : \mathit{Name}.s$. So, we have $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \langle a \rangle M : \mathit{Name}.s$ and $b \# s$. Further, we can deduce $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s$ and $a \# s$. We then apply Lemma 3.2.5 to obtain $\nabla \vdash^{\text{NLC}[\mathbb{A}]} (a\, b) * M : (a\, b) \cdot s$. Given that $a, b \# s$, we have completed this case. □

We continue with η-contraction, where we first recall that for pure $\lambda^{\to}$ the type preservation property follows directly from the strengthening property of pure $\lambda^{\to}$: If $\Gamma, x : \sigma \vdash M : \tau$ and $x \notin fv(M)$, then $\Gamma \vdash M : \tau$. Note that the strengthening property still holds for pure NLC[$\mathbb{A}$], but we now have to consider an additional complication regarding type preservation of η-contraction: Let $\overline{b} \subset \overline{a}$. We can deduce

$$\emptyset \vdash^{\text{NLC}[\mathbb{A}]} \lambda^{\overline{a}} x : s.\, ((\lambda^{\overline{b}} y : s.\, y)\ x) : s^{\overline{a}} \Rightarrow s$$
$$\emptyset \vdash^{\text{NLC}[\mathbb{A}]} \lambda^{\overline{b}} y : s.\, y : s^{\overline{b}} \Rightarrow s$$

However, due to the fact that $\overline{a} \neq \overline{b}$, we have that $s^{\overline{a}} \Rightarrow s \neq s^{\overline{b}} \Rightarrow s$ and therefore the usual η-contraction axiom does not preserve types anymore. To resolve this issue we introduce a typing assertion as an additional side condition, which enforces the

preservation of types for η-contraction. To emphasise this, we "decorate" the η-contraction with $\vdash$ and a context ∇. In contrast, despite their many similarities, this is not required for name abstractions.

- (R-EABS) $\nabla \vdash \lambda^{\overline{a}} x : s.\,(M\ x) \to_\eta M$ if $x \# M$ and $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M : s^{\overline{a}} \Rightarrow \tau$ for some type τ
- (R-ENABS): $\langle a \rangle (M \mathbin{@} a) \to_\eta M$

Remark 4.2.2 *The side condition of (R-EABS) is similar to a side condition used for pure $\beta\eta$-reduction systems of $\lambda^{\to}$ with subtyping [58]. Of course,* NLC[$\mathbb{A}$] *does not have an explicit subtyping relation, but it has some flavour of subtyping due its domain (freshness) restricted function types and the (AP) typing rule. Thus, in retrospect it is not too surprising that an additional side condition for η-contraction is required.*

Lemma 4.2.3 *The contraction axioms (R-EABS) and (R-ENABS) preserve types.*

Proof For (R-EABS) type preservation follows immediately by definition of the axiom. In the case of (R-ENABS), we suppose that $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \langle a \rangle (M \mathbin{@} a) : \mathit{Name}.s$. From this we can deduce $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M \mathbin{@} a : s$ and $a \# s$. Further, we have that $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} a \# M : \mathit{Name}.s$ and therefore $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M : \mathit{Name}.s$. □

We refer to expressions of the form $(\lambda^{\overline{a}} x : s.\,M)\ N$ or $(\langle a \rangle M) \mathbin{@} b$ (in context ∇) as a **β-redex** and expressions of the form $\lambda^{\overline{a}} x : s.\,(M\ x)$ ($x \# M$ and $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M : s^{\overline{a}} \Rightarrow \tau$ for some type τ) or $\langle a \rangle (M \mathbin{@} a)$ (in context ∇) as an **η-redex**.

We say that M **$\beta\eta$-reduces to** N (in context ∇) in one step, written as $\nabla \vdash M \to_{\beta\eta} N$, if N can be obtained by applying a β-contraction or η-contraction on some $\beta\eta$-redex in M (in context ∇). The relation $\to_{\beta\eta}$ (in context ∇) is formally defined as the smallest binary relation on expressions that is closed under the rules in Table 4.4. It follows by standard computations that the relation preserves α-equivalence and is therefore well defined. The reflexive and transitive closure of $\to_{\beta\eta}$ is denoted

by $\twoheadrightarrow_{\beta\eta}$ and an expression M is in $\beta\eta$**-normal form** (for context ∇) if there is no N such that $\nabla \vdash M \rightarrow_{\beta\eta} N$.

$$\frac{\nabla \vdash M_1 \rightarrow_{\beta\eta} M_1'}{\nabla \vdash M_1\ M_2 \rightarrow_{\beta\eta} M_1'\ M_2}\ (\textsf{R-APP-1})$$

$$\frac{\nabla \vdash M_2 \rightarrow_{\beta\eta} M_2'}{\nabla \vdash M_1\ M_2 \rightarrow_{\beta\eta} M_1\ M_2'}\ (\textsf{R-APP-2})$$

$$\frac{\nabla, \overline{a} \# x : s \vdash M_1 \rightarrow_{\beta\eta} M_1'}{\nabla \vdash \lambda^{\overline{a}} x : s.\, M_1 \rightarrow_{\beta\eta} \lambda^{\overline{a}} x : s.\, M_1'}\ (\textsf{R-ABS})$$

$$\frac{\nabla \vdash M_1 \rightarrow_{\beta\eta} M_1'}{\nabla \vdash M_1 \mathbin{@} b \rightarrow_{\beta\eta} M_1' \mathbin{@} b}\ (\textsf{R-CONC})$$

$$\frac{\nabla \vdash M_1 \rightarrow_{\beta\eta} M_1'}{\nabla \vdash \langle a \rangle M_1 \rightarrow_{\beta\eta} \langle a \rangle M_1'}\ (\textsf{R-NABS})$$

$$\frac{}{\nabla \vdash (\lambda^{\overline{a}} x : s.\, M)\ N \rightarrow_{\beta\eta} M\{N/x\}}\ (\textsf{R-BAPP})$$

$$\frac{}{\nabla \vdash (\langle a \rangle M) \mathbin{@} b \rightarrow_{\beta\eta} (a\, b) * M}\ (\textsf{R-BCONC})$$

$$\frac{x \# M \qquad \nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M : s^{\overline{a}} \Rightarrow \tau}{\nabla \vdash \lambda^{\overline{a}} x : s.\, (M\ x) \rightarrow_{\beta\eta} M}\ (\textsf{R-EABS})$$

$$\frac{}{\nabla \vdash \langle a \rangle (M \mathbin{@} a) \rightarrow_{\beta\eta} M}\ (\textsf{R-ENABS})$$

Table 4.4: $\beta\eta$-Reduction System for NLC[$\mathbb{A}$]

An immediate consequence of the definition of the $\beta\eta$-reduction system is the following technical lemma:

Lemma 4.2.4 *If* $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M : s$ *and* $\nabla \vdash M \twoheadrightarrow_{\beta\eta} N$*, then* $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M \approx N : s$.

Proof Suppose $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M : s$ and $\nabla \vdash M \twoheadrightarrow_{\beta\eta} N$. By definition, it is either the

case that $M \equiv_\alpha N$ or there exists $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M' : s$ such that $\nabla \vdash M \to_{\beta\eta} M'$ and $\nabla \vdash M' \twoheadrightarrow_{\beta\eta} N$. The first case follows immediately by rule (REF) on $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s$. The second case is proved by induction on the number of reduction steps:

Base Case: We have to show by rule-based induction that for any one step reduction $\nabla \vdash M \to_{\beta\eta} M'$, we can deduce $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M \approx M' : s$. But this follows immediately by the congruence rules and the axioms (B), (E), (BN) and (EN), as well as the derived expression $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s$.

Inductive Case: Suppose $\nabla \vdash M \to_{\beta\eta} M'$ and $\nabla \vdash M' \twoheadrightarrow_{\beta\eta} N$. We obtain $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M \approx M' : s$ using the base case and $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M' \approx N : s$ by the induction hypothesis. By applying (TRANS) we obtain that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M \approx N : s$ holds. □

We prove via subject reduction that the reduction system is sound.

Lemma 4.2.5 (Subject Reduction) *If* $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s$ *and* $\nabla \vdash M \twoheadrightarrow_{\beta\eta} M'$, *then* $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M' : s$.

Proof We first prove subject reduction for the one-step $\beta\eta$-reduction system by induction on the structure of M. The case for $\twoheadrightarrow_{\beta\eta}$ then follows immediately by induction on the number of reduction steps.

For suspended variables there are no reduction rules to apply. We now have to consider the various reduction rules for application and abstraction. Given that we have already shown it for the axiom cases, we only need to consider the congruence reduction rules:

Case 1: (AP) Suppose that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M\ N : s'$. From this we can deduce that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s^{\overline{a}} \Rightarrow s'$ and $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \overline{a} \# N : s$. Further, we have that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N : s$ and $\overline{a} \# s$. We now consider the two congruence rules for applications:

Case 1.1: (R-APP-1) Suppose that $\nabla \vdash M\ N \to_{\beta\eta} M'\ N$. From this we can deduce $\nabla \vdash M \to_{\beta\eta} M'$. Further, given that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s^{\overline{a}} \Rightarrow s'$, we obtain $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M' : s^{\overline{a}} \Rightarrow s'$ by induction. Then, by rule AP, we have $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M'\ N : s'$.

Case 1.2: (R-APP-2) Suppose that $\nabla \vdash M\ N \to_{\beta\eta} M\ N'$. From this we can deduce $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N : s$ and by induction $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N' : s$. Moreover, applying Lemma 4.2.4 on $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N : s$ and $\nabla \vdash N \to_{\beta\eta} N'$ we obtain $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N \approx N' : s$. Then, by Lemma 3.2.18, we can deduce that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \overline{a} \# N' : s$ and by applying rule (AP) we obtain $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M\ N' : s$.

Case 2: (ABS) Suppose that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \lambda^{\overline{a}} x : s.\, N : s^{\overline{a}} \Rightarrow s'$. From this we can deduce $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}[\mathbb{A}]} N : s'$. There is one reduction rule that can be applied.

Case 2.1: (R-ABS) Suppose that $\nabla \vdash \lambda^{\overline{a}} x : s.\, N \to_{\beta\eta} \lambda^{\overline{a}} x : s.\, N'$. From this we can deduce $\nabla, \overline{a} \# x : s \vdash N \to_{\beta\eta} N'$ and given that $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}[\mathbb{A}]} N : s'$, we obtain $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}[\mathbb{A}]} N' : s'$ by induction. Then, by rule ABS, we have $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \lambda^{\overline{a}} x : s.\, N' : s^{\overline{a}} \Rightarrow s'$.

Case 3: (CONC) Suppose that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M\ @\ b : s$. From this we can deduce $\nabla \vdash^{\text{NLC}[\mathbb{A}]} b \# M : \mathit{Name}.s$, and moreover $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : \mathit{Name}.s$ and $\overline{b} \# s$. There is one reduction rule that can be applied.

Case 3.1: (R-CONC) Suppose that $\nabla \vdash M\ @\ b \to_{\beta\eta} M'\ @\ b$. From this we can deduce $\nabla \vdash M \to_{\beta\eta} M'$ and given that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : \mathit{Name}.s$ holds, we obtain $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M' : \mathit{Name}.s$ by induction. Moreover, by applying Lemma 4.2.4 on $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : \mathit{Name}.s$ and $\nabla \vdash M \to_{\beta\eta} M'$ we obtain $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M \approx M' : \mathit{Name}.s$. Next, we apply Lemma 3.2.18 to deduce that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \overline{a} \# M' : \mathit{Name}.s$. Then, by rule (CONC), we have $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M\ N' : s$.

Case 4: (NABS) Suppose that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \langle a \rangle N : \mathit{Name}.s$. From this we can deduce that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N : s$ and $a \# s$. There is one reduction rule that can be applied.

Case 4.1: (R-NABS) Suppose that $\nabla \vdash \langle a \rangle N \to_{\beta\eta} \langle a \rangle N'$. From this we can deduce $\nabla \vdash N \to_{\beta\eta} N'$ and given that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N : s$ holds, we obtain $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N' : s$ by induction. Then, by rule (NABS), we have $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \langle a \rangle N' : \mathit{Name}.s$. □

Note that due to the subject reduction lemma we may now write $\nabla \vdash M \twoheadrightarrow_{\beta\eta} N : s$ for $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s$ and $\nabla \vdash M \twoheadrightarrow_{\beta\eta} N$. We continue by proving various technical lemmas which will be used to prove strong normalisation, confluence and decidability

of provable equality for pure NLC[$\mathbb{A}$].

Lemma 4.2.6 *if* $\nabla, \overline{a} \# x : s \vdash P \rightarrow_{\beta\eta} P' : s'$*, then for any* $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \overline{a} \# M : s$*, we have* $\nabla \vdash P\{M/x\} \rightarrow_{\beta\eta} P'\{M/x\} : s'$.

Proof Proof by induction on the structure of P. In the case of suspended variables there are no reductions to apply. For application and abstraction we have to consider the various reduction cases. Here, we only consider the reduction axioms, the congruence reduction rules follow by routine computations using the induction hypothesis. We consider the cases where x is a free variable in the corresponding redex, otherwise the property would follow immediately.

Case 1: (R-BAPP) Suppose $\nabla, \overline{a} \# x : s \vdash (\lambda^{\overline{a}} y : s.\, N)\ Q \rightarrow_{\beta\eta} N\{Q/y\} : s'$. We consider the case where $y \# (x, dom(\nabla), Q, M)$. We apply Lemma 3.2.20 to obtain $\nabla \vdash^{\text{NLC}[\mathbb{A}]} ((\lambda^{\overline{b}} y : s.\, N)\ Q)\{M/x\} : s'$. Then, by definition of the object-level permutation action we have

$$((\lambda^{\overline{b}} y : s.\, N)\ Q)\{M/x\} \equiv_\alpha (\lambda^{\overline{b}} y : s.\, N\{M/x\})\ Q\{M/x\}$$

We can then apply the reduction rule (R-BAPP) to obtain

$$\nabla \vdash (\lambda^{\overline{b}} y : s.\, N\{M/x\})\ Q\{M/x\} \rightarrow_{\beta\eta} (N\{M/x\})\{Q\{M/x\}/y\} : s'$$

Using the substitution identity we have

$$(N\{M/x\})\{Q\{M/x\}/y\} \equiv_\alpha (N\{Q/y\})\{M/x\}$$

Hence, we have deduced

$$\nabla \vdash ((\lambda^{\overline{b}} y : s.\, N)\ Q)\{M/x\} \rightarrow_{\beta\eta} (N\{Q/y\})\{M/x\} : s'$$

Case 2: (R-BCONC) Suppose $\nabla, \overline{a} \# x : s \vdash (\langle a \rangle N)\ @\ b \rightarrow_{\beta\eta} (a\, b) * N : s'$. By Lemma 3.2.20 we have that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} ((\langle a \rangle N)\ @\ b)\{M/x\} : s'$. Then by definition of the object-level permutation action we have

$$((\langle a \rangle N)\ @\ b)\{M/x\} \equiv_\alpha (\langle a \rangle N\{M/x\})\ @\ b$$

We can then apply the reduction rule (R-BCONC) to obtain

$$\nabla \vdash (\langle a \rangle N\{M/x\}) \mathbin{@} b \to_{\beta\eta} (a\, b) * N\{M/x\} : s'$$

Applying Lemma 3.1.14 we obtain

$$(a\, b) * N\{M/x\} \equiv_\alpha ((a\, b) * N)\{M/x\}$$

Hence, we have deduced

$$\nabla \vdash ((\langle a \rangle N) \mathbin{@} b)\{M/x\} \to_{\beta\eta} ((a\, b) * N)\{M/x\} : s'$$

Case 3: (R-EABS) Suppose $\nabla, \overline{a} \# x : s \vdash \lambda^{\overline{b}} y : s'.(N\ y) \to_{\beta\eta} N : s'^{\overline{b}} \Rightarrow s''$ $(y \# N)$. We consider the case $y \# (x, dom(\nabla), M)$. By Lemma 3.2.20 we have that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} (\lambda^{\overline{b}} y : s'.(N\ y))\{M/x\} : s'^{\overline{b}} \Rightarrow s''$ and $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N\{M/x\} : s'^{\overline{b}} \Rightarrow s''$. Then by definition of the object-level permutation action and $y \# M$ we have

$$(\lambda^{\overline{b}} y : s'.(N\ y))\{M/x\} \equiv_\alpha \lambda^{\overline{b}} y : s'.(N\{M/x\}\ y)$$

From $y \# (M, N)$ we can deduce that $y \# N\{M/x\}$. So, we have an η-redex and therefore the reduction rule R-EABS can be applied to obtain

$$\nabla \vdash \lambda^{\overline{b}} y : s'.(N\{M/x\}\ y) \to_{\beta\eta} N\{M/x\} : s'^{\overline{b}} \Rightarrow s''$$

Hence, we have deduced

$$\nabla \vdash (\lambda^{\overline{b}} y : s'.(N\ y))\{M/x\} \to_{\beta\eta} N\{M/x\} : s'^{\overline{b}} \Rightarrow s''$$

Case 4: (R-ENABS) Suppose $\nabla, \overline{a} \# x : s \vdash \langle a \rangle (N \mathbin{@} a) \to_{\beta\eta} N : Name.s$. Applying Lemma 3.2.20 we have that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} (\langle a \rangle (N \mathbin{@} a))\{M/x\} : Name.s$. Then, by definition of the object-level permutation action we obtain

$$(\langle a \rangle (N \mathbin{@} a))\{M/x\} \equiv_\alpha \langle a \rangle (N\{M/x\} \mathbin{@} a)$$

Hence, we have an η-redex and can therefore apply the reduction rule (R-ENABS) to obtain

$$\nabla \vdash \langle a \rangle (N\{M/x\} \mathbin{@} a) \to_{\beta\eta} N\{M/x\} : Name.s$$

Hence, we have deduced

$$\nabla \vdash (\langle a \rangle (N \mathbin{@} a))\{M/x\} \rightarrow_{\beta\eta} N\{M/x\} : Name.s$$

□

Lemma 4.2.7 *if* $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \overline{a} \# M, M' : s$ *and* $\nabla \vdash M \rightarrow_{\beta\eta} M' : s$*, then for any* $\nabla, \overline{a} \# x : s \vdash^{\mathrm{NLC}[\mathbb{A}]} P : s'$*, we have* $\nabla \vdash P\{M/x\} \twoheadrightarrow_{\beta\eta} P\{M'/x\} : s'$*.*

Proof Using Lemma 3.2.20 we obtain $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} P\{M/x\} : s'$ and $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} P\{M'/x\} : s'$. The proof immediately follows by induction on the number of occurrences of x in P, where the base case is analogue to the proof of Lemma 4.2.6 □

Lemma 4.2.8 *if* $\nabla, \overline{a} \# x : s \vdash P \twoheadrightarrow_{\beta\eta} P' : s'$*,* $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \overline{a} \# M, M' : s$ *and* $\nabla \vdash M \twoheadrightarrow_{\beta\eta} M' : s$*, then we have* $\nabla \vdash P\{M/x\} \twoheadrightarrow_{\beta\eta} P'\{M'/x\} : s'$*.*

Proof By induction on the amount of reduction steps of $\nabla, \overline{a} \# x : s \vdash P \twoheadrightarrow_{\beta\eta} P' : s'$ we obtain $\nabla \vdash P\{M/x\} \twoheadrightarrow_{\beta\eta} P'\{M/x\}$ and by induction on the amount of reduction steps of $\nabla \vdash M \twoheadrightarrow_{\beta\eta} M' : s$, we obtain that $\nabla \vdash P'\{M/x\} \twoheadrightarrow_{\beta\eta} P'\{M'/x\} : s'$. The result then follows by transitivity. □

Lemma 4.2.9 (Equivariance of Reduction) $\rightarrow_{\beta\eta}$ *is equivariant with respect to* $\pi * -$*, i.e. if* $\nabla \vdash M \rightarrow_{\beta\eta} M' : s$*, then* $\nabla \vdash \pi * M \rightarrow_{\beta\eta} \pi * M' : \pi \cdot s$*. Moreover,* $\twoheadrightarrow_{\beta\eta}$ *is equivariant with respect to* $\pi * -$*.*

Proof Proof by induction on the structure of M. In the case of suspended variables there are no reductions to apply. For application and abstraction we have to consider the various reduction cases. Here, we only consider the reduction axioms, the congruence reduction rules follow by routine computations using the induction hypothesis.

Case 1: (R-BAPP) Suppose $\nabla \vdash (\lambda^{\overline{a}} x : s.\, P)\ Q \rightarrow_{\beta\eta} P\{Q/x\} : s$. By Lemma 3.2.5 we have that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \pi * ((\lambda^{\overline{a}} x : s.\, P)\ Q) : \pi \cdot s$. Then, by definition of the object-level permutation action, we have that

$$\pi * ((\lambda^{\overline{a}} x : s.\, P)\ Q) \stackrel{\text{def}}{=} ((\lambda^{\pi \cdot \overline{a}} x : \pi \cdot s.\, \pi * P[\pi^{-1} x/x])\ (\pi * Q))$$

Next, we apply the reduction rule (R-BAPP) to obtain

$$\nabla \vdash ((\lambda^{\pi \cdot \overline{a}} x : \pi \cdot s.\, \pi * P[\pi^{-1} x/x])\ (\pi * Q)) \rightarrow_{\beta\eta} (\pi * P[\pi^{-1} x/x])\{\pi * Q/x\} : \pi \cdot s$$

Using Lemma 3.1.11 and a standard substitution identity we obtain

$$(\pi * P[\pi^{-1} x/x])\{\pi * Q/x\} \equiv_{\alpha} \pi * P\{\pi^{-1} x/x\}\{\pi * Q/x\} \equiv_{\alpha} \pi * P\{Q/x\}$$

Hence, we have that

$$\nabla \vdash \pi * ((\lambda^{\overline{a}} x : s.\, P)\ Q) \rightarrow_{\beta\eta} \pi * (P\{Q/x\}) : \pi \cdot s$$

Case 2: (R-BCONC) Suppose $\nabla \vdash (\langle a \rangle N) \mathbin{@} b \rightarrow_{\beta\eta} (a\, b) * N : s$. By Lemma 3.2.5 we have that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \pi * ((\langle a \rangle N) \mathbin{@} b) : \pi \cdot s$. Then, by definition of the object-level permutation action, we have

$$\pi * ((\langle a \rangle N) \mathbin{@} b) \stackrel{\text{def}}{=} (\langle \pi \cdot a \rangle \pi * N) \mathbin{@} \pi \cdot b$$

Next, we apply the reduction rule (R-BCONC) to obtain

$$\nabla \vdash (\langle \pi \cdot a \rangle \pi * N) \mathbin{@} \pi \cdot b \rightarrow_{\beta\eta} (\pi \cdot a\, \pi \cdot b) * \pi * N : \pi \cdot s$$

By the property of permutation actions we obtain

$$(\pi \cdot a\, \pi \cdot b) * \pi * N \equiv \pi * (a\, b) * N$$

Hence, we have that

$$\nabla \vdash \pi * ((\langle a \rangle N) \mathbin{@} b) \rightarrow_{\beta\eta} \pi * ((a\, b) * N) : \pi \cdot s$$

Case 3: (R-EABS) Suppose $\nabla \vdash \lambda^{\overline{a}} x : s.\,(M\ x) \to_{\beta\eta} M : s^{\overline{a}} \Rightarrow s'\ (x \# M)$. By Lemma 3.2.5 we have that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \pi * \lambda^{\overline{a}} x : s.\,(M\ x) : \pi \cdot (s^{\overline{a}} \Rightarrow s')$ and $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \pi * M : \pi \cdot (s^{\overline{a}} \Rightarrow s')$. Then by definition of the object-level permutation action and $x \# M$ we have

$$\pi * (\lambda^{\overline{a}} x : s.\,(M\ x)) \stackrel{\text{def}}{=} \lambda^{\pi \cdot \overline{a}} x : \pi \cdot s.\,\pi * (M\ \pi^{-1} x) \equiv \lambda^{\pi \cdot \overline{a}} x : \pi \cdot s.\,((\pi * M)\ x)$$

Given that $x \# \pi * M$ and $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \pi * M : \pi \cdot (s^{\overline{a}} \Rightarrow s')$, it is an η-redex and therefore we can apply the reduction rule (R-EABS) to obtain

$$\nabla \vdash \lambda^{\pi \cdot \overline{a}} x : \pi \cdot s.\,((\pi * M)\ x) \to_{\beta\eta} \pi * M : \pi \cdot (s^{\overline{a}} \Rightarrow s')$$

Hence, we have that

$$\nabla \vdash \pi * (\lambda^{\overline{a}} x : s.\,(M\ x)) \to_{\beta\eta} \pi * M : \pi \cdot (s^{\overline{a}} \Rightarrow s')$$

Case 4: (R-ENABS) Suppose $\nabla \vdash \langle a \rangle (M\ @\ a) \to_{\beta\eta} M : \mathit{Name}.s$. By Lemma 3.2.5 we have that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \pi * \langle a \rangle (M\ @\ a) : \pi \cdot \mathit{Name}.s$. Then, by definition of the object-level permutation action, we have

$$\pi * \langle a \rangle (M\ @\ a) \stackrel{\text{def}}{=} \langle \pi \cdot a \rangle (\pi * M\ @\ \pi \cdot a)$$

This is an eta-redex and therefore we can apply the reduction rule (R-EABS) to obtain

$$\nabla \vdash \langle \pi \cdot a \rangle (\pi * M\ @\ \pi \cdot a) \to_{\beta\eta} \pi * M : \pi \cdot \mathit{Name}.s$$

Hence, we have that

$$\nabla \vdash \pi * \langle a \rangle (M\ @\ a) \to_{\beta\eta} \pi * M : \pi \cdot \mathit{Name}.s$$

The fact that $\twoheadrightarrow_{\beta\eta}$ is equivariant follows immediately by induction on the number of reduction steps. □

Lemma 4.2.10 *if M (in context ∇) is in $\beta\eta$-normal form, then $\pi * M$ (in context ∇) is in $\beta\eta$-normal form.*

Proof The contrapositive of this property follows directly using a proof by cases for the different $\beta\eta$-redexes and Lemma 4.2.9. □

4.2.2 Strong Normalisation

We prove a strong normalisation theorem for the pure $\beta\eta$-reduction system of NLC[$\mathbb{A}$]. To achieve this goal we adapt the proof argument used in [38], which is based on Tait's logical relations argument [67].

Definition 4.2.11 *An expression M (in context ∇) is **strongly normalising**, denoted by $\mathrm{SN}_{\nabla}(M)$, if there are no infinite reduction sequences originating from it. Consequently, every reduction sequence must end in a normal form after a finite number of reduction steps.*

Definition 4.2.12 *The **logical predicate** $\mathcal{R} := \{\mathrm{R}^{s}_{\nabla} \mid s \in \mathit{Type}, \nabla \in \mathit{Ctxt}\}$, over well typed expressions, is defined as a type-indexed family of relations, which are recursively defined as follows:*

(i) $\mathrm{R}^{s}_{\nabla} \subseteq \mathit{Exp}(\nabla, s)$

(ii) $\mathrm{R}^{\gamma}_{\nabla}(M) \Longleftrightarrow \mathrm{SN}_{\nabla}(M)$

(iii) $\mathrm{R}^{s\overline{a}\Rightarrow s'}_{\nabla}(M) \Longleftrightarrow (\forall N \in \mathit{Exp}(\nabla, s))\,.\,[\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \overline{a} \# N : s \wedge \mathrm{R}^{s}_{\nabla}(N) \Longrightarrow \mathrm{R}^{s'}_{\nabla}(M\ N)]$

(iv) $\mathrm{R}^{\mathit{Name}.s}_{\nabla}(M) \Longleftrightarrow (\forall b \in \mathbb{A})\,.\,[\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} b \# M : \mathit{Name}.s \Longrightarrow \mathrm{R}^{s}_{\nabla}(M \mathbin{@} b)]$

Lemma 4.2.13 *For any context ∇ and type τ*

(i) if $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} [[...[x\ X_1]\ ...]\ X_n] : \tau$ and $\mathrm{SN}_{\nabla}(X_i)$, then $\mathrm{R}^{\tau}_{\nabla}([[...[x\ X_1]\ ...]\ X_n])$, where $[-\ -]$ represents either an application $(-\ -)$ or a concretion $(-\ @\ -)$, and X_i are respectively well typed expressions or names.

(ii) If $\mathrm{R}^{\tau}_{\nabla}(M)$ then $\mathrm{SN}_{\nabla}(M)$

Note that property (i) is an auxiliary result, which is used to demonstrate that each $\mathrm{R}^{\sigma}_{\nabla}$ is non-empty. This is required to prove property (ii).

Proof Both properties are proved simultaneous by induction on type τ.

Case (τ is γ):

(i) Suppose $\nabla \vdash^{\text{NLC}[\mathbb{A}]} [[...[x\ X_1]\ ...]\ X_n] : \gamma$ and $\text{SN}_\nabla(X_i)$. By definition of $\mathcal{R}$, for base types, we need to show that $\text{SN}_\nabla([[...[x\ X_1]\ ...]\ X_n])$, which follows immediately because the only redexes are in the X_i's and we have that $\text{SN}_\nabla(X_i)$.

(ii) Follows immediately by definition of $\text{R}^\gamma_\nabla(M)$.

Case (τ is $s^{\overline{a}} \Rightarrow s'$):

(i) Suppose $\nabla \vdash^{\text{NLC}[\mathbb{A}]} [[...[x\ X_1]\ ...]\ X_n] : s^{\overline{a}} \Rightarrow s'$ and $\text{SN}_\nabla(X_i)$. Let $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \overline{a}\ \#\ N : s$ and $\text{R}^s_\nabla(N)$. By the induction hypothesis for (ii) on type s, we obtain $\text{SN}_\nabla(N)$. Moreover, using rule (AP), we can deduce that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} [[...[x\ X_1]\ ...]\ X_n]\ N : s'$. Then, by the induction hypothesis for (i) on type s', we obtain $\text{R}^{s'}_\nabla([[...[x\ X_1]\ ...]\ X_n]\ N)$. Hence, by definition of $\mathcal{R}$, we have that $\text{R}^{s^{\overline{a}} \Rightarrow s'}_\nabla([[...[x\ X_1]\ ...]\ X_n])$.

(ii) Suppose $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s^{\overline{a}} \Rightarrow s'$ and $\text{R}^{s^{\overline{a}} \Rightarrow s'}_\nabla(M)$. We pick a variable x such that $x\ \#\ M$. It is clearly the case that $\nabla, \overline{a}\ \#\ x : s \vdash^{\text{NLC}[\mathbb{A}]} \overline{a}\ \#\ x : s$ and by the induction hypothesis for (i) on type s, we obtain that $\text{R}^s_{\nabla, \overline{a}\#x:s}(x)$. From $\text{R}^{s^{\overline{a}} \Rightarrow s'}_\nabla(M)$ we can deduce that $\text{R}^{s^{\overline{a}} \Rightarrow s'}_{\nabla, \overline{a}\#x:s}(M)$. By definition of $\mathcal{R}$, we obtain that $\text{R}^{s'}_{\nabla, \overline{a}\#x:s}(M\ x)$, and by the induction hypothesis for (ii) on type s' that $\text{SN}_{\nabla, \overline{a}\#x:s}(M\ x)$. Given that any subterm of a strongly normalising term is strongly normalising, we have $\text{SN}_\nabla(M)$.

Case (τ is $Name.s$):

(i) Suppose $\nabla \vdash^{\text{NLC}[\mathbb{A}]} [[...[x\ X_1]\ ...]\ X_n] : Name.s$ and $\text{SN}_\nabla(X_i)$. Let $b \in \mathbb{A}$ and $\nabla \vdash^{\text{NLC}[\mathbb{A}]} b\ \#\ [[...[x\ X_1]\ ...]\ X_n] : Name.s$. From this we can deduce by rule (CONC) that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} [[...[x\ X_1]\ ...]\ X_n]\ @\ b : s$. Then, by the induction hypothesis for (i) on type s, we obtain $\text{R}^s_\nabla([[...[x\ X_1]\ ...]\ X_n]\ @\ b)$. Hence, by definition of $\mathcal{R}$, we have that $\text{R}^{Name.s}_\nabla([[...[x\ X_1]\ ...]\ X_n])$ holds.

(ii) Suppose that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : Name.s$ and $\text{R}^{Name.s}_\nabla(M)$. We pick a name $b \in \mathbb{A}$ such that $b\ \#\ (\nabla, M, s)$. By applying Lemma 3.2.14 (Weakening), we obtain $\nabla^{\#b} \vdash^{\text{NLC}[\mathbb{A}]} M : Name.s$. Then, by Lemma 3.2.19, we can deduce that $\nabla^{\#b} \vdash^{\text{NLC}[\mathbb{A}]} b\ \#\ M : Name.s$ holds. From $\text{R}^{Name.s}_\nabla(M)$ it can be deduced that $\text{R}^{Name.s}_{\nabla^{\#b}}(M)$ holds. So,

by definition of $\mathcal{R}$, we obtain that $\mathrm{R}^{s}_{\nabla\#b}(M \mathbin{@} b)$ and by the induction hypothesis for (ii) on type s that $\mathrm{SN}_{\nabla\#b}(M \mathbin{@} b)$. Given that any subterm of a strongly normalising term is strongly normalising, we have $\mathrm{SN}_{\nabla}(M)$. □

Lemma 4.2.14 (Equivariance of $\mathcal{R}$) *if* $\mathrm{R}^{\tau}_{\nabla}(M)$ *then* $\mathrm{R}^{\pi\cdot\tau}_{\nabla}(\pi * M)$

Proof We prove it by induction on type τ:

Case (τ is γ): Suppose that $\mathrm{R}^{\gamma}_{\nabla}(M)$. It follows by definition that $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M : \gamma$ and $\mathrm{SN}_{\nabla}(M)$. We can apply Lemma 3.2.5 to obtain that $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \pi * M : \pi \cdot \gamma$ holds. We now suppose, for a contradiction, that $\pi * M$ is not strongly normalising. So, by definition, there exists an infinite reduction sequence originating from $\pi * M$. By applying Lemma 4.2.9 we can deduce that there also exists an infinite reduction sequence, which originates from M. But this contradicts our assumption that $\mathrm{SN}_{\nabla}(M)$. Hence, we have $\mathrm{SN}_{\nabla}(\pi * M)$ and by definition of $\mathcal{R}$ we have $\mathrm{R}^{\pi\cdot\gamma}_{\nabla}(\pi * M)$.

Case (τ is $s^{\overline{a}} \Rightarrow s'$): Suppose $\mathrm{R}^{s^{\overline{a}}\Rightarrow s'}_{\nabla}(M)$. We have to show that $\mathrm{R}^{\pi\cdot(s^{\overline{a}}\Rightarrow s')}_{\nabla}(\pi * M)$. By definition of $\mathcal{R}$, we can equally demonstrate that $\mathrm{R}^{\pi\cdot s'}_{\nabla}(\pi * M)\ N$ under the assumption that $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \pi \cdot \overline{a} \# N : \pi \cdot s$ and $\mathrm{R}^{\pi\cdot s}_{\nabla}(N)$.

By applying Lemma 3.2.5 we obtain that $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \overline{a} \# \pi^{-1} * N : s$ and by induction on s we get $\mathrm{R}^{s}_{\nabla}(\pi^{-1} * N)$. Applying this to $\mathrm{R}^{s^{\overline{a}}\Rightarrow s'}_{\nabla}(M)$, we obtain by definition that $\mathrm{R}^{s'}_{\nabla}(M\ (\pi^{-1} * N))$. Then, by induction on s', we get that $\mathrm{R}^{\pi\cdot s'}_{\nabla}(\pi * M\ (\pi^{-1} * N))$, and by definition of the object-level permutation action we have that $\mathrm{R}^{\pi\cdot s'}_{\nabla}(\pi * M\ N)$.

Case (τ is $Name.s$): Suppose $\mathrm{R}^{Name.s}_{\nabla}(M)$. We have to show that $\mathrm{R}^{\pi\cdot(Name.s)}_{\nabla}(\pi * M)$. By definition of $\mathcal{R}$ and the permutation action on types, we can equally demonstrate that $\mathrm{R}^{\pi\cdot s}_{\nabla}((\pi * M) \mathbin{@} b)$ under the assumption that $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} b \# \pi * M : \pi \cdot Name.s$.

We can apply Lemma 3.2.5 to obtain $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \pi^{-1} \cdot b \# M : Name.s$, and by definition of $\mathcal{R}$ for $\mathrm{R}^{Name.s}_{\nabla}(M)$ we have $\mathrm{R}^{s}_{\nabla}(M \mathbin{@} \pi^{-1}\cdot b)$. Next, we apply the induction hypothesis for type s to obtain that $\mathrm{R}^{\pi\cdot s}_{\nabla}(\pi * (M \mathbin{@} \pi^{-1} \cdot b))$. This case is completed by definition of the object-level permutation action. □

Lemma 4.2.15

(i) For any $\nabla, \overline{a} \# x : s \vdash^{\text{NLC}[\mathbb{A}]} M : \sigma$ *and* $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N : s$ *such that* $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \overline{a} \# N : s$ *we have that*

$$\mathrm{R}^{\sigma}_{\nabla}(M\{N/x\}) \Longrightarrow \mathrm{R}^{\sigma}_{\nabla}((\lambda^{\overline{a}} x : s.\, M)\ N)$$

provided that $\mathrm{R}^{s}_{\nabla}(N)$ *if* $x \notin fv(M)$.

(ii) For any $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \langle a \rangle M : Name.s$ *such that* $\nabla \vdash^{\text{NLC}[\mathbb{A}]} b \# \langle a \rangle M : Name.s$

$$\mathrm{R}^{\sigma}_{\nabla}((a\, b) * M) \Longrightarrow \mathrm{R}^{\sigma}_{\nabla}((\langle a \rangle M) \mathbin{@} b)$$

Proof

(i) Suppose $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N : s$ such that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \overline{a} \# N : s$, as well as $\mathrm{R}^{\sigma}_{\nabla}(M\{N/x\})$. As can easily be observed, any type τ can uniquely be written in the form

$$\langle\ -\ \ -\ \langle\ ...\ \langle\ -\ \ -\ \langle\ -\ \ -\ \ \gamma\ \rangle\ \rangle\ ...\ \rangle\ \rangle$$

where γ is a base type, $\langle\ s\ \overline{a}\ s'\ \rangle \stackrel{\text{def}}{=} s^{\overline{a}} \Rightarrow s'$ and $\langle\ *\ \ *\ \ s\ \rangle \stackrel{\text{def}}{=} Name.s$. We take σ to be presented in this form. Suppose that σ has a nesting depth of n. We now demonstrate that $\mathrm{R}^{\sigma}_{\nabla}((\lambda^{\overline{a}} x : s.\, M)\ N)$ holds. Let $X_1, ..., X_n$ be well typed expressions or names that satisfy the type and freshness conditions for type σ, as well as $\mathrm{R}(X_i)$ if X_i is part of a lambda application. Then, by definition of $\mathrm{R}^{\sigma}_{\nabla}$ we can equally demonstrate that

$$\mathrm{R}^{\gamma}_{\nabla}([[...[((\lambda^{\overline{a}} x : s.\, M)\ N)\ X_1]\ ...]X_n])$$

Note that $[-\ -]$ was defined in Lemma 4.2.13. Given that $\mathrm{R}^{\sigma}_{\nabla}(M\{N/x\})$ holds by assumption, we can deduce by definition of $\mathcal{R}$ that

$$\mathrm{R}^{\gamma}_{\nabla}([[...[(M\{N/x\})\ X_1]\ ...]X_n])$$

Hence, by definition of $\mathcal{R}$ for base types, we can equally prove that

$$\mathrm{SN}_{\nabla}([[...[(M\{N/x\})\ X_1]\ ...]X_n]) \Longrightarrow \mathrm{SN}_{\nabla}([[...[((\lambda^{\overline{a}} x : s.\, M)\ N)\ X_1]\ ...]X_n])$$

Note that by hypothesis we have that $\mathrm{R}^{s}_{\nabla}(N)$ if N does not occur in $M\{N/x\}$. By applying Lemma 4.2.8, the remainder of the argument follows as in the case of the pure $\lambda^{\rightarrow}$.

(ii) The argument follows similarly to (i). To demonstrate that the implication of strongly normalising terms holds, we have to apply Lemma 4.2.9. □

Lemma 4.2.16 *If* $\overline{a}_1 \# x_1 : s_1, ..., \overline{a}_n \# x_n : s_n \vdash^{\mathrm{NLC}[\mathbb{A}]} M : s$, $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \overline{a}_i \# N_i : s_i$ *(*$1 \leq i \leq n$*) and* $\mathrm{R}^{s_i}_{\nabla}(N_i)$, *then* $\mathrm{R}^{s}_{\nabla}(M\{N_1/x_1, ..., N_n/x_n\})$.

Proof Proof by induction on the structure of M. We write ∇ for $\overline{a}_1 \# x_1 : s_1, ..., \overline{a}_n \# x_n : s_n$ and θ for $\{N_1/x_1, ..., N_n/x_n\}$.

SUSP: (M is πx_i) Suppose $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \pi x_i : \pi \cdot s_i$. Applying Lemma 4.2.14 we can deduce from $\mathrm{R}^{s_i}_{\nabla}(N_i)$ that $\mathrm{R}^{\pi \cdot s_i}_{\nabla}(\pi * N_i)$ holds. Given that $(\pi x_i)\theta \stackrel{\mathrm{def}}{=} \pi * N_i$, we obtain that $\mathrm{R}^{\pi \cdot s_i}_{\nabla}((\pi x_i)\theta)$ holds.

ABS: (M is $\lambda^{\overline{a}} x : s.\, P$) Suppose $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \lambda^{\overline{a}} x : s.\, P : s^{\overline{a}} \Rightarrow s'$. We consider the case where $x \# (x_1, ..., x_n, N_1, ..., N_n)$. Applying Lemma 3.2.20 we obtain that $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} (\lambda^{\overline{a}} x : s.\, P)\theta : s^{\overline{a}} \Rightarrow s'$ holds. Considering the choice of x we have that $(\lambda^{\overline{a}} x : s.\, P)\theta \equiv_{\alpha} \lambda^{\overline{a}} x : s.\, (P\theta)$ and therefore $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \lambda^{\overline{a}} x : s.\, (P\theta) : s^{\overline{a}} \Rightarrow s'$. We now need to demonstrate that $\mathrm{R}^{s^{\overline{a}} \Rightarrow s'}_{\nabla}(\lambda^{\overline{a}} x : s.\, (P\theta))$. Let $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} N : s$ such that $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \overline{a} \# N : s$ and $\mathrm{R}^{s}_{\nabla}(N)$. By definition of $\mathcal{R}$ we can equally show that $\mathrm{R}^{s'}_{\nabla}((\lambda^{\overline{a}} x : s.\, (P\theta))\ N)$ holds. By applying rule (AP) we get $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} (\lambda^{\overline{a}} x : s.\, (P\theta))\ N : s'$. Then, by induction on $\nabla, \overline{a} \# x : s \vdash^{\mathrm{NLC}[\mathbb{A}]} P : s'$ with $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \overline{a}_i \# N_i : s_i$, $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} \overline{a} \# N : s$, $\mathrm{R}^{s_i}_{\nabla}(N_i)$ and $\mathrm{R}^{s}_{\nabla}(N)$, we obtain that $\mathrm{R}^{s'}_{\nabla}((P\{N_1/x_1, ..., N_n/x_n, N/x\}))$ holds. Further, by the choice of variable x, we can compute

$$P\{N_1/x_1, ..., N_n/x_n, N/x\} \equiv_{\alpha} (P\{N_1/x_1, ..., N_n/x_n\})\{N/x\} \equiv (P\theta)\{N/x\}$$

Hence, we have that $\mathrm{R}^{s'}_{\nabla}((P\theta)\{N/x\})$ holds and by applying Lemma 4.2.15 (i) we can deduce that $\mathrm{R}^{s'}_{\nabla}((\lambda^{\overline{a}} x : s.\, (P\theta))\ N)$ holds.

APP: (M is $N\ N'$) Suppose that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N\ N' : s'$. From this we can deduce that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N : s^{\overline{a}} \Rightarrow s'$ and $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \overline{a} \# N' : s$, as well as $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N' : s$. Considering that $(N\ N')\theta \stackrel{\text{def}}{=} (N\theta)\ (N'\theta)$, we have to demonstrate that $\text{R}^{s'}_{\nabla}((N\theta)\ (N'\theta))$ holds.

By induction on $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N : s^{\overline{a}} \Rightarrow s'$ and $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N' : s$ we obtain $\text{R}^{s^{\overline{a}} \Rightarrow s'}_{\nabla}(N\theta)$ and $\text{R}^{s}_{\nabla}(N'\theta)$. Then, by Lemma 3.2.20, we can deduce that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \overline{a} \# N'\theta : s$ holds. Next, we can deduce, by definition of $\text{R}^{s^{\overline{a}} \Rightarrow s'}_{\nabla}(N\theta)$, that $\text{R}^{s'}_{\nabla}((N\theta)(N'\theta))$ holds.

NABS: (M is $\langle a \rangle N$) Suppose that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \langle a \rangle N : \mathit{Name}.s$. From this we can deduce that $a \# s$ and $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N : s$. Then, by induction on $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N : s$ we obtain $\text{R}^{s}_{\nabla}(N\theta)$. We now need to show that $\text{R}^{\mathit{Name}.s}_{\nabla}((\langle a \rangle N)\theta)$, or equally by definition of substitution that $\text{R}^{\mathit{Name}.s}_{\nabla}(\langle a \rangle (N\theta))$ holds. Let $b \in \mathbb{A}$ and $\nabla \vdash^{\text{NLC}[\mathbb{A}]} b \# \langle a \rangle N\theta : \mathit{Name}.s$. By definition of $\text{R}^{\mathit{Name}.s}_{\nabla}$ we can equally show that $\text{R}^{s}_{\nabla}((\langle a \rangle (N\theta))\ @\ b)$ holds.

We apply Lemma 4.2.14 on $\text{R}^{s}_{\nabla}(N\theta)$ to obtain that $\text{R}^{\pi \cdot s}_{\nabla}((a\,b) * (N\theta))$. We complete this case by applying Lemma 4.2.15 (ii) to obtain $\text{R}^{s}_{\nabla}((\langle a \rangle (N\theta))\ @\ b)$.

CONC: (M is $N\ @\ b$) Suppose that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N\ @\ b : s$. From this we can deduce that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} b \# N : \mathit{Name}.s$, as well as $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N : \mathit{Name}.s$ and $b \# \mathit{Name}.s$. Applying capture avoiding substitution, we need to show that $\text{R}^{s}_{\nabla}((N\theta)\ @\ b)$ holds.

By induction on $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N : \mathit{Name}.s$ we obtain $\text{R}^{\mathit{Name}.s}_{\nabla}(N\theta)$, and by Lemma 3.2.20 we get that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} b \# N\theta : \mathit{Name}.s$. We can then directly deduce by definition of $\text{R}^{\mathit{Name}.s}_{\nabla}(N\theta)$ that $\text{R}^{s}_{\nabla}((N\theta)\ @\ b)$ holds. □

Theorem 4.2.17 (Strong Normalisation) *If* $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s$, *then* $\text{SN}_{\nabla}(M)$.

Proof Suppose $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s$. We take the identity substitution $N_i \equiv x_i$. It is clearly the case that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \overline{a}_i \# x_i : s_i$. So, by Lemma 4.2.13 (i), we have that $\text{R}^{s_i}_{\nabla}(x_i)$. Next, we apply Lemma 4.2.16 to deduce that $\text{R}^{s}_{\nabla}(M)$ and complete the argument by applying Lemma 4.2.13 (ii) to obtain that $\text{SN}_{\nabla}(M)$ holds. □

4.2.3 Confluence

Given that pure NLC[$\mathbb{A}$] is strongly normalising it suffices to prove weak confluence since confluence can directly be deduced using Newman's lemma [50]. As previously indicated, we require a structural congruence relation $\equiv_\alpha^\nabla$ on raw terms in $\beta\eta$-normal form (see Table 4.5) to prove weak confluence. It is a direct extension of the α-equivalence relation $\equiv_\alpha$, as presented in Table 3.3, where we additionally incorporate the rules (SUSP) and (BNABS) of NLC[$\mathbb{A}$].

$$\frac{M_1 \equiv_\alpha^\nabla M_1' \qquad M_2 \equiv_\alpha^\nabla M_2'}{M_1\ M_2 \equiv_\alpha^\nabla M_1'\ M_2'} \qquad \frac{M \equiv_\alpha^\nabla M'}{M\ @\ b \equiv_\alpha^\nabla M'\ @\ b}$$

$$\frac{(z\,x) \bullet M \equiv_\alpha^{\nabla, \overline{a}\#z:s} (z\,y) \bullet M'}{\lambda^{\overline{a}} x : s.\, M \equiv_\alpha^\nabla \lambda^{\overline{a}} y : s.\, M'}\ (z \# (x, y, M, M'))$$

$$\frac{(b\,a_1) * M_1 \equiv_\alpha^{\nabla^{\#b}} (b\,a_2) * M_2}{\langle a_1 \rangle M_1 \equiv_\alpha^\nabla \langle a_2 \rangle M_2}\ (b \# (\nabla, a_1, a_2, M_1, M_2))$$

$$\frac{}{\pi x \equiv_\alpha^\nabla \pi' x}\ (ds(\pi, \pi') \subseteq \overline{a} \text{ for } \nabla(x) = (\overline{a}, s))$$

Table 4.5: Structural Congruence Relation for NLC[$\mathbb{A}$]

The following technical lemma is a straightforward extension of the corresponding results for α-equivalence as presented in [56] (Section 4.1).

Lemma 4.2.18 *The operations suspension-substitution, variable swapping, object-level and meta-level permutation actions (on well typed terms) preserve the structural congruence relation, i.e.*

(i) $M \equiv_\alpha^\nabla M' \Longrightarrow \mu \bullet M \equiv_\alpha^{\mu(\nabla)} \mu \bullet M'$

(ii) $M \equiv_\alpha^\nabla M' \Longrightarrow M[\pi^{-1}x/x] \equiv_\alpha^\nabla M'[\pi^{-1}x/x]$

(iii) $M \equiv_\alpha^\nabla M' \Longrightarrow \pi * M \equiv_\alpha^\nabla \pi * M'$

(iv) $M \equiv_\alpha^\nabla M' \Longrightarrow \pi \cdot M \equiv_\alpha^{\pi\cdot\nabla} \pi \cdot M'$

Lemma 4.2.19 $\equiv_\alpha^\nabla$ *is an equivalence relation.*

Proof We extend the proof that $\equiv_\alpha$ is an equivalence relation as presented in [56] (Section 4.1) such that it holds for $\equiv_\alpha^\nabla$. Given that the proofs for reflexivity and symmetry are straightforward, we will only provide details for transitivity. Here, we follow the argument as provided in [56] (Example 7.8). We first recall that the transitivity property

$$\forall M_1, M_2, M_3 \, [\, (M_1 \equiv_\alpha^\nabla M_2 \wedge M_2 \equiv_\alpha^\nabla M_3) \Longrightarrow M_1 \equiv_\alpha^\nabla M_2 \,]$$

which can be transformed into

$$\forall M_1, M_2 \, [\, M_1 \equiv_\alpha^\nabla M_2 \Longrightarrow (\forall M_3 \,.\, M_2 \equiv_\alpha^\nabla M_3 \Longrightarrow M_1 \equiv_\alpha^\nabla M_3) \,]$$

Hence, the property can be proved by rule-based induction on $M_1 \equiv_\alpha^\nabla M_2$. Let $Q \stackrel{\text{def}}{=} \{(M_1, M_2) \mid (\forall M_3 \,.\, M_2 \equiv_\alpha^\nabla M_3 \Longrightarrow M_1 \equiv_\alpha^\nabla M_3)\}$. Note that from the fact that $\equiv_\alpha^\nabla$ is equivariant, we can deduce that Q is equivariant as well.

Given that the rules for application and abstraction follow, with minor modifications, from the original proof argument, we only provide details for the additional rules with respect to suspension, name abstraction and concretion.

Concretion: Suppose $M \mathbin{@} b \equiv_\alpha^\nabla M' \mathbin{@} b$. From this we can deduce that $M \equiv_\alpha^\nabla M'$. Then, by induction we have that $Q(M, M')$ holds. We need to show that $Q(M \mathbin{@} b, M' \mathbin{@} b)$. Let N be a well typed expression. By definition of property Q we have to demonstrate that $M' \mathbin{@} b \equiv_\alpha^\nabla N \Longrightarrow M \mathbin{@} b \equiv_\alpha^\nabla N$. We suppose that $M' \mathbin{@} b \equiv_\alpha^\nabla N$ holds and demonstrate that $M \mathbin{@} b \equiv_\alpha^\nabla N$.

Applying the rule inversion principle on $M' \mathbin{@} b \equiv_\alpha^\nabla N$ we deduce that $N \equiv N' \mathbin{@} b$ for some well typed term N' with $M' \equiv_\alpha^\nabla N'$. By definition of $Q(M, M')$ we can

then deduce that $M \equiv_\alpha^\nabla N'$ and by applying the rule schema for concretion that $M \mathbin{@} b \equiv_\alpha^\nabla N' \mathbin{@} b$.

Name Abstraction: Suppose $\langle a_1\rangle M_1 \equiv_\alpha^\nabla \langle a_2\rangle M_2$. From this we can deduce that $(b\,a_1) * M_1 \equiv_\alpha^{\nabla^{\#b}} (b\,a_2) * M_2$ for some $b \# (\nabla, a_1, a_2, M_1, M_2)$. Then, by induction we have that $Q((b\,a_1) * M_1, (b\,a_2) * M_2)$. We need to show that $Q(\langle a_1\rangle M_1, \langle a_2\rangle M_2)$ holds. Let N be a well typed expression. By definition of property Q we have to demonstrate that $\langle a_2\rangle M_2 \equiv_\alpha^\nabla N \Longrightarrow \langle a_1\rangle M_1 \equiv_\alpha^\nabla N$. We suppose that $\langle a_2\rangle M_2 \equiv_\alpha^\nabla N$ and demonstrate that $\langle a_1\rangle M_1 \equiv_\alpha^\nabla N$.

Applying the rule inversion principle on $\langle a_2\rangle M_2 \equiv_\alpha^\nabla N$ we can deduce that $N \equiv \langle a_3\rangle M_3$ for some name a_3 and well typed expression M_3, and there exists a name $b' \# (a_2, M_2, a_3, M_3, \nabla)$ with $(a_2\,b') * M_2 \equiv_\alpha^{\nabla^{\#b'}} (a_3\,b') * M_3$.

Pick $b'' \# (a_1, M_1, a_2, M_2, a_3, M_3, b, b', \nabla)$. Given that $\equiv_\alpha^\nabla$ is equivariant we can apply $(b'\,b'') \cdot -$ on both sides to obtain

$$(b'\,b'') \cdot ((a_2\,b') * M_2) \equiv_\alpha^{(b'\,b'')\cdot(\nabla^{\#b'})} (b'\,b'') \cdot ((a_3\,b') * M_3)$$

We now compute the following:

$$\begin{aligned}
(b'\,b'') \cdot (\nabla^{\#b'}) &= ((b'\,b'') \cdot \nabla)^{\#(b'\,b'')\cdot b'} && \text{(equivariance)}\\
&= \nabla^{\#b''} && (b', b'' \# \nabla)
\end{aligned}$$

$$\begin{aligned}
(b'\,b'') \cdot ((a_2\,b') * M_2) &= (b'\,b'') * ((a_2\,b') * M_2)\{(b'\,b'')\!-\!\} && \text{(Lemma 3.1.12)}\\
&= (b'\,b'') * (a_2\,b') * M_2\{(b'\,b'')\!-\!\} && \text{(Lemma 3.1.4)}\\
&= (a_2\,b'') * (b'\,b'') * M_2\{(b'\,b'')\!-\!\} && (b', b'' \# a_2)\\
&= (a_2\,b'') * ((b'\,b'') \cdot M_2) && \text{(Lemma 3.1.12)}\\
&= (a_2\,b'') * M_2 && (b', b'' \# M_2)
\end{aligned}$$

we can equally compute that

$$(b'\,b'') \cdot ((a_3\,b') * M_3) = (a_3\,b'') * M_3$$

Hence, we have deduced that

$$(a_2\, b'') * M_2 \equiv_\alpha^{\nabla \# b''} (a_3\, b'') * M_3 \quad (\diamond 1)$$

Further, given that the property Q is equivariant we can apply $(b\, b'') \cdot -$ on $Q((b\, a_1) * M_1, (b\, a_2) * M_2)$, i.e. for any well typed expression P

$$(b\, b'') \cdot ((a_2\, b) * M_2) \equiv_\alpha^{(b\, b'') \cdot \nabla \# b} P \Longrightarrow (b\, b'') \cdot ((a_1\, b) * M_1) \equiv_\alpha^{(b\, b'') \cdot (\nabla \# b)} P$$

We can now compute the following:

$$\begin{aligned}
(b\, b'') \cdot (\nabla^{\# b}) &= ((b\, b'') \cdot \nabla)^{\# (b\, b'') \cdot b} && \text{(equivariance)} \\
&= \nabla^{\# b''} && (b, b'' \# \nabla)
\end{aligned}$$

$$\begin{aligned}
(b\, b'') \cdot ((a_2\, b) * M_2) &= (b\, b'') * ((a_2\, b) * M_2)\{(b\, b'') \cdot -\} && \text{(Lemma 3.1.12)} \\
&= (b\, b'') * (a_2\, b) * M_2\{(b\, b'') \cdot -\} && \text{(Lemma 3.1.4)} \\
&= (a_2\, b'') * (b\, b'') * M_2\{(b\, b'') \cdot -\} && (b, b'' \# a_2) \\
&= (a_2\, b'') * ((b\, b'') \cdot M_2) && \text{(Lemma 3.1.12)} \\
&= (a_2\, b'') * M_2 && (b, b'' \# M_2)
\end{aligned}$$

we can equally compute

$$(b\, b'') \cdot ((a_1\, b) * M_1) = (a_1\, b'') * M_1$$

Hence, we have deduced that

$$(a_2\, b'') * M_2 \equiv_\alpha^{\nabla \# b''} P \Longrightarrow (a_1\, b'') * M_1 \equiv_\alpha^{\nabla \# b''} P \quad (\diamond 2)$$

For P is $(a_3\, b'') * M_3$ we can now deduce from $(\diamond 1)$ and $(\diamond 2)$ that

$$(a_1\, b'') * M_1 \equiv_\alpha^{\nabla \# b''} (a_3\, b'') * M_3$$

Given that $b'' \# (a_1, M_1, a_3, M_3)$ we can apply the rule schema for concretion and obtain that $\langle a_1 \rangle M_1 \equiv_\alpha^\nabla \langle a_3 \rangle M_3$ holds.

Suspension: Suppose $\pi x \equiv_\alpha^\nabla \pi' x$ with $ds(\pi, \pi') \subseteq \overline{a}$ for $\nabla(x) = (\overline{a}, s)$. Let N be a well typed expression. We need to show that $Q(\pi x, \pi' x)$. Suppose that $\pi' x \equiv_\alpha^\nabla N$ holds. We need to show that $\pi x \equiv_\alpha^\nabla N$. By the rule inversion principle on $\pi' x \equiv_\alpha^\nabla N$ we have that $N \equiv \tau x$ with $ds(\pi', \tau) \subseteq \overline{a}$ for $\nabla(x) = (\overline{a}, s)$.

Suppose towards a contradiction that $c \notin \overline{a}$ and $\pi(c) \neq \tau(c)$. We then have that $pi'(c) = \tau(c)$ and $\pi(c) = \pi'(c)$; thus $\pi(c) = \tau(c)$. Hence, $ds(\pi, \tau) \subseteq \overline{a}$ and by the rule schema we have $\pi x \equiv_\alpha^\nabla \tau x$. □

We continue by proving a technical lemma, which is an immediate consequence of the definition of $\equiv_\alpha^\nabla$.

Lemma 4.2.20 *Suppose M, M' (in context ∇) are in $\beta\eta$-normal form. Then*

$$\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M \approx M' : s \iff M \equiv_\alpha^\nabla M'$$

Proof ($\Rightarrow$:) Proof by rule-based induction on equations. Note that by definition of $\equiv_\alpha^\nabla$ all cases apart from (AE) follow immediately. Hence, we only need to demonstrate closure under the rule schema (AE): Suppose that $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M \approx M' : s$ holds. From this we can deduce that $\nabla^{\#\overline{a}} \vdash^{\mathrm{NLC}[\mathbb{A}]} M \approx M' : s$ with $\overline{a} \# (\nabla, M, M')$. We then apply the induction hypothesis to obtain $M \equiv_\alpha^{\nabla^{\#\overline{a}}} M'$. Given that $\overline{a} \# (M, M')$, we can directly deduce that $\overline{a}$ does not occur in any disagreement set of suspended variables in M and M', and therefore it is not required to prove that the structural congruence relation holds; thus we have that $M \equiv_\alpha^\nabla M'$.

($\Leftarrow$): It follows immediately by a rule-based induction on $\equiv_\alpha^\nabla$. □

Lemma 4.2.21 (Weak Church-Rosser) *Let $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M : s$. If $\nabla \vdash M \to_{\beta\eta} M'$ and $\nabla \vdash M \to_{\beta\eta} M''$, then there exists a well typed expression $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} N : s$ such that $\nabla \vdash M' \twoheadrightarrow_{\beta\eta} N$ and $\nabla \vdash M'' \twoheadrightarrow_{\beta\eta} N$ (up to $\equiv_\alpha^\nabla$ equivalence).*

Proof This proof proceeds by cases of $\beta\eta$-redexes. Suppose that $\nabla \vdash M \to_{\beta\eta} M'$ and $\nabla \vdash M \to_{\beta\eta} M''$, with non-nesting redexes, then M''' is the expression with both

redexes reduced and M' and M'' directly reduce to M'''. We continue with the various nesting cases. We use the notation $\mathcal{C}[R]$ to indicate that R is a redex in context $\mathcal{C}$.

Case 1: We begin with the nesting cases of β-redexes for applications.

Case 1.1: Suppose that $M \equiv \mathcal{C}[(\lambda^{\overline{a}} x : s.\, N)\ P]$, $M' \equiv \mathcal{C}[N\{P/x\}]$, $M'' \equiv \mathcal{C}[(\lambda^{\overline{a}} x : s.\, N)\ P']$ and $\nabla \vdash P \to_{\beta\eta} P'$. Then $M''' \equiv \mathcal{C}[N\{P'/x\}]$ with $\nabla \vdash M' \twoheadrightarrow_{\beta\eta} M'''$ in as many steps as there are occurrences of the variable x in term N and $\nabla \vdash M'' \to_{\beta} M'''$.

Case 1.2: Suppose that $M \equiv \mathcal{C}[(\lambda^{\overline{a}} x : s.\, N)\ P]$, $M' \equiv \mathcal{C}[N\{P/x\}]$, $M'' \equiv \mathcal{C}[(\lambda^{\overline{a}} x : s.\, N')\ P]$ and $\nabla \vdash N \to_{\beta\eta} N'$. We take $M''' \equiv \mathcal{C}[N'\{P/x\}]$. We then apply Lemma 4.2.8 on $\nabla \vdash N \to_{\beta\eta} N'$ to obtain $\nabla \vdash N\{P/x\} \to_{\beta\eta} N'\{P/x\}$; thus $\nabla \vdash M' \to_{\beta\eta} M'''$ and $\nabla \vdash M'' \to_{\beta} M'''$.

Case 2: We consider the nesting cases of η-redexes for lambda abstractions.

Case 2.1: Suppose $M \equiv \mathcal{C}[\lambda^{\overline{a}} x : s.\, (N\ x)]$, $M' \equiv \mathcal{C}[N]$, $M'' \equiv \mathcal{C}[\lambda^{\overline{a}} x : s.\, (N'\ x)]$ and $\nabla \vdash N \to_{\beta\eta} N'$. From $x \# N$ we can directly deduce that $x \# N'$. Further, the typing condition still holds and therefore $\lambda^{\overline{a}} x : s.\, (N'\ x)$ is also an η-redex. Take $M''' \equiv \mathcal{C}[N']$. So, by rule (R-EABS) we have $\nabla \vdash M'' \to_{\eta} M'''$. $\nabla \vdash M' \to_{\beta\eta} M'''$ follows directly from $\nabla \vdash N \to_{\beta\eta} N'$.

Case 2.2: Suppose $M \equiv \mathcal{C}[\lambda^{\overline{a}} x : s.\, (N\ x)]$, $M' \equiv \mathcal{C}[N]$, $M'' \equiv \mathcal{C}[\lambda^{\overline{a}} x : s.\, P\{x/y\}]$ where $N \equiv \lambda^{\overline{b}} y : s'.\, P$ and $\nabla \vdash N\ x \to_{\beta\eta} P\{x/y\}$. From $x \# N$ we can deduce that $x \# P$. We can then deduce from $\lambda^{\overline{a}} x : s.\, ((\lambda^{\overline{b}} y : s'.\, P)\ x)$ that $s = s'$ and $\overline{b} \subseteq \overline{a}$. Further, by definition of η-contraction we can actually deduce that $\overline{a} = \overline{b}$. Given that $x \# P$, we have $\lambda^{\overline{b}} y : s'.\, P \equiv_{\alpha} \lambda^{\overline{a}} x : s.\, P\{x/y\}$. Hence, we have that $M' \equiv_{\alpha} M''$.

Case 3: We consider the nesting case of β-redexes for concretion. Suppose that $M \equiv \mathcal{C}[(\langle a \rangle N) \mathbin{@} b]$, $M' \equiv \mathcal{C}[(a\, b) * N]$, $M'' \equiv \mathcal{C}[(\langle a \rangle N') \mathbin{@} b]$ and $\nabla \vdash N \to_{\beta\eta} N'$. We take $M''' \equiv \mathcal{C}[(a\, b) * N']$. We then apply Lemma 4.2.9 on $\nabla \vdash N \to_{\beta\eta} N'$ to obtain $\nabla \vdash (a\, b) * N \to_{\beta\eta} (a\, b) * N'$; thus $\nabla \vdash M' \to_{\beta\eta} M'''$. $\nabla \vdash M'' \to_{\beta} M'''$ follows by rule (R-BCONC).

Case 4: We consider the nesting cases of η-redexes for name abstractions:

Case 4.1: Suppose that $M \equiv \mathcal{C}[\langle a\rangle(N \mathbin{@} a)]$, $M' \equiv \mathcal{C}[N]$, $M'' \equiv \mathcal{C}[\langle a\rangle(N' \mathbin{@} a)]$ and $\nabla \vdash N \to_{\beta\eta} N'$. Take $M''' \equiv \mathcal{C}[N']$. By rule (R-EABS) we have that $\nabla \vdash M'' \to_{\eta} M'''$ and $\nabla \vdash M' \to_{\beta\eta} M'''$ follows by $\nabla \vdash N \to_{\beta\eta} N'$.

Case 4.2: Suppose that $M \equiv \mathcal{C}[\langle a\rangle(N \mathbin{@} a)]$, $M' \equiv \mathcal{C}[N]$ and $M'' \equiv \mathcal{C}[\langle a\rangle(a\,b) * P]$ where $N \equiv \langle b\rangle P$. From $\nabla \vdash^{\text{NLC}[\mathbb{A}]} \langle a\rangle(\langle b\rangle P \mathbin{@} a) : \mathit{Name}.s$ we can deduce that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} a \# \langle b\rangle P : \mathit{Name}.s$ holds. The case for $a = b$ follows trivially. In the case of $a \neq b$, we can directly deduce that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} a \# P : s$ holds. Due to the fact that the reduction system is strongly normalising, we can reduce M' and M'' to normal form, using the same reduction strategy on $\mathcal{C}$ and P to obtain $\mathcal{C}'$ and P'. Hence, we have $M''' \equiv \mathcal{C}'[\langle b\rangle P']$ and $M'''' \equiv \mathcal{C}'[\langle a\rangle(a\,b) * P']$ with $\nabla \vdash M' \twoheadrightarrow_{\beta\eta} M'''$ and $\nabla \vdash M'' \twoheadrightarrow_{\beta\eta} M''''$. We now need to show that $M''' \equiv_{\alpha}^{\nabla} M''''$. So, what remains to be proven is that

$$\langle b\rangle P' \equiv_{\alpha}^{\nabla} \langle a\rangle(a\,b) * P'$$

By definition of $\equiv_{\alpha}^{\nabla}$, we choose any $c \# (\nabla, P', a, b)$ and show that $(b\,c) * P' \equiv_{\alpha}^{\nabla^{\#c}} (a\,c) * (a\,b) * P$ holds:

From $\nabla \vdash P \twoheadrightarrow_{\beta\eta} P'$ we can deduce by Lemma 4.2.4 that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} P \approx P' : s$. Given that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} a \# P : s$ we can deduce by Lemma 3.2.18 that $\nabla \vdash^{\text{NLC}[\mathbb{A}]} a \# P' : s$. Then, by definition of freshness assertions and $c \# (\nabla, a, P')$ we have that $\nabla^{\#c} \vdash^{\text{NLC}[\mathbb{A}]} (a\,c) * P' \approx P' : s$ holds. We can now apply Lemma 3.2.5 to obtain $\nabla^{\#c} \vdash^{\text{NLC}[\mathbb{A}]} (b\,c) * (a\,c) * P' \approx (b\,c) * P' : s$. By Lemma 4.2.10 we can deduce that both sides of the equation are in $\beta\eta$-normal form, and therefore it follows immediately by Lemma 4.2.20 that $(b\,c) * (a\,c) * P' \equiv_{\alpha}^{\nabla^{\#c}} (b\,c) * P'$. We complete this case by observing that $(b\,c) * (a\,c) * P' \equiv (a\,c) * (a\,b) * P'$ holds. □

Theorem 4.2.22 (Church Rosser) *Let $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s$. If $\nabla \vdash M \twoheadrightarrow_{\beta\eta} M'$ and $\nabla \vdash M \twoheadrightarrow_{\beta\eta} M''$, then there exists a well typed expression $\nabla \vdash^{\text{NLC}[\mathbb{A}]} N : s$ such that $\nabla \vdash M' \twoheadrightarrow_{\beta\eta} N$ and $\nabla \vdash M'' \twoheadrightarrow_{\beta\eta} N$ (up to $\equiv_{\alpha}^{\nabla}$ equivalence).*

Proof From strong normalisation and weak confluence for the $\beta\eta$-reduction system,

we can deduce that $\beta\eta$-reduction system is confluent (up to $\equiv_\alpha^\nabla$ equivalence) using Newman's lemma [50]. □

There are two immediate corollary of strong normalisation and confluence.

Corollary 4.2.23 *For any well typed expression $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M : s$, there exists a unique $\beta\eta$-normal form (up to $\equiv_\alpha^\nabla$-equivalence), written as $NF(M)$.*

Proof We can deduce from Theorem 4.2.17 (strong normalisation) that there exists a $\beta\eta$-normal form for M and that by confluence any such normal form is unique (up to $\equiv_\alpha^\nabla$-equivalence) □

Corollary 4.2.24 (Consistency) *The pure $\beta\eta$-theory in* NLC[$\mathbb{A}$] *is consistent.*

Proof This follows as in the case of pure $\lambda^\to$. We pick two distinct normal forms $\emptyset \# y : s \vdash^{\text{NLC}[\mathbb{A}]} \lambda x : s.\, x : s \Rightarrow s$ and $\emptyset \# y : s \vdash^{\text{NLC}[\mathbb{A}]} \lambda x : s.\, y : s \Rightarrow s$. Using Lemma 4.2.20 we can immediately deduce that NLC[$\mathbb{A}$] is consistent. □

4.2.4 Decidability of Provable Equality

Using strong normalisation and confluence of the $\beta\eta$-reduction system we can deduce that the notion of provable equality for NLC[$\mathbb{A}$] is decidable. We first extend Lemma 4.2.20 to hold for any well typed NLC[$\mathbb{A}$]-expressions.

Lemma 4.2.25 *For any well typed expressions $\nabla \vdash^{\text{NLC}[\mathbb{A}]} M, M' : s$*

$$\nabla \vdash^{\text{NLC}[\mathbb{A}]} M \approx M' : s \iff NF(M) \equiv_\alpha^\nabla NF(M')$$

Proof ($\Rightarrow$:) Proof by rule-based induction. The cases (SUSP), (BNABS) and (AE) follow analogue to Lemma 4.2.20, apart from (AE), where we additionally need to observe that we can deduce from $\overline{a} \# M$ that $\overline{a} \# NF(M)$ holds. The other cases follow, using confluence, as for $\lambda^\to$.

($\Leftarrow$): Suppose $NF(M) \equiv_\alpha^\nabla NF(M')$. From $\nabla \vdash M \twoheadrightarrow_{\beta\eta} NF(M)$ and $\nabla \vdash M' \twoheadrightarrow_{\beta\eta} NF(M')$ we can deduce, applying Lemma 4.2.4, that $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M \approx NF(M) : s$ and $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} M' \approx NF(M') : s$. Further, from $NF(M) \equiv_\alpha^\nabla NF(M')$ we obtain, using Lemma 4.2.20, that $\nabla \vdash^{\mathrm{NLC}[\mathbb{A}]} NF(M) \approx NF(M') : s$. We conclude by applying rule (TRANS). □

Theorem 4.2.26 (Decidability) *Provable equality for* NLC[$\mathbb{A}$] *is decidable.*

Proof We know by Corollary 4.2.23 that there exists a unique normal form (up to $\equiv_\alpha^\nabla$ equivalence), which can directly be computed by the "normal order" reduction strategy. This is clearly decidable. Given that the structural congruence relation $\equiv_\alpha^\nabla$ is also decidable, we can deduce from Lemma 4.2.25 that provable equality is decidable as well. □

Remark 4.2.27 *We can easily observe that type checking and type inference in* NLC[$\mathbb{A}$] *are essentially syntax-directed. This is analogue to $\lambda^\to$, where both are decidable. To demonstrate that this also holds for* NLC[$\mathbb{A}$]*, we need to deal with freshness assertions, which are a particular kind of equations. However, after we have shown that provable equality is decidable, it follows by minor enhancements of the algorithms for $\lambda^\to$ that type checking and type inference are decidable as well.*

4.3 ИNLC: NLC[$\mathbb{A}$] with Local Fresh Atomic Names

In this section we introduce ИNLC, an extension of NLC[$\mathbb{A}$], which captures the notion of local fresh atomic names.

Meta-Theory for ИNLC Raw Terms

We extend NLC[$\mathbb{A}$] by introducing a term former $\mathsf{fr}\, a.\, M$, which is a binder on names. Hence, each occurrence of a name is now either **free** or **bound** (where all

occurrences of a in any "subterm" $\mathsf{fr}\, a.\, M$ are bound). The recursively defined auxiliary functions $|M|$, $var(M)$, $name(M)$ and $fv(M)$ can trivially be extended. Further, the recursive definition of $fn(M)$, which determines the free names in M, is given in Table 4.6. Using the term formers $\mathsf{fr}\, a.\, M$ and $\langle a \rangle M$, we can furthermore introduce a binding variant of name abstraction as syntactic sugar

$$\alpha a.M \stackrel{\text{def}}{=} \mathsf{fr}\, a.\, \langle a \rangle M$$

– $fn(\pi x) \stackrel{\text{def}}{=} supp(\pi)$	
– $fn(c) \stackrel{\text{def}}{=} supp(c)$	
– $fn(M\ N) \stackrel{\text{def}}{=} fn(M) \cup fn(N)$	– $fn(\mathsf{fr}\, a.\, M) \stackrel{\text{def}}{=} fn(M) \setminus \{a\}$
– $fn(\lambda^{\overline{a}} x : s.\, M) \stackrel{\text{def}}{=} \overline{a} \cup supp(s) \cup supp(M)$	– $fn(M \mathbin{@} b) \stackrel{\text{def}}{=} fn(M) \cup \{b\}$
	– $fn(\langle a \rangle M) \stackrel{\text{def}}{=} fn(M) \cup \{a\}$

Table 4.6: $fn(M)$ for ИNLC

Remark 4.3.1 *The binding variant of name abstraction, $\alpha a.M$, directly corresponds to the notions of name abstraction introduced for* SNTT *(denoted by $\langle a \rangle M$) and the $\lambda\alpha\nu$-calculus (denoted by $\alpha a.M$). In contrast to* ИNLC*, the $\lambda\alpha\nu$-calculus introduces the binding variant, $\alpha a.M$, as a first class citizen, while the non-binding variant, denoted by $\langle a \rangle M$, is introduced as syntactic sugar using term formers $\alpha a.M$ and $(a//a')$ (a syntactic notion of name swapping).* SNTT *does not express both notions of name abstraction.*

We extend variable swapping and the meta-level permutation action of NLC[$\mathbb{A}$] as follows: Let $\mu \in Perm(Var)$ and $\pi \in Perm(\mathbb{A})$.

$$\mu \bullet (\mathsf{fr}\, a.\, M) \stackrel{\text{def}}{=} \mathsf{fr}\, a.\, \mu \bullet M$$

$$\pi \cdot (\mathsf{fr}\, a.\, M) \stackrel{\text{def}}{=} \mathsf{fr}\, \pi \cdot a.\, \pi \cdot M$$

Using the previously defined meta-level permutation action, we extend the notion of α-equivalence as follows:

$$\frac{(\mathsf{M}\, b) \quad (b\, a) \cdot M \equiv_\alpha (b\, a') \cdot M'}{\mathsf{fr}\, a.\, M \equiv_\alpha \mathsf{fr}\, a'.\, M'}$$

In particular, the rule schema is defined for (some/any) name b such that $b \# (M, M', a, a')$. Analogue to NLC[$\mathbb{A}$], both permutation actions preserve α-equivalence and can therefore be lifted to expressions:

$$\mu \bullet [M]_\alpha \stackrel{\text{def}}{=} [\mu \bullet M]_\alpha$$
$$\pi \cdot [M]_\alpha \stackrel{\text{def}}{=} [\pi \cdot M]_\alpha$$

with $supp([M]_\alpha) = fv(M)$ and $supp([M]_\alpha) = fn(M)$. Next, we extend the object-level permutation action $- * - : Perm(\mathbb{A}) \times Term_{Sg}/{\sim_\alpha} \to Term_{Sg}/{\sim_\alpha}$.

$$\pi * \mathsf{fr}\, a.\, M \stackrel{\text{def}}{=} \mathsf{fr}\, a'.\, \pi * ((a\, a') \cdot M) \quad (a' \# (M, \pi))$$

Note that this definition differs from NLC[$\mathbb{A}$], where we defined the object-level permutation action on raw terms. Further, the definition of capture avoiding substitution is extended as follows:

$$(\mathsf{fr}\, a.\, M)\{N/x\} \stackrel{\text{def}}{=} \mathsf{fr}\, a'.\, ((a\, a') \cdot M)\{N/x\} \quad (a' \# (M, N))$$

It follows by routine computations that the object-level permutation action and capture avoiding substitution are well defined, i.e. the choice of free names does not matter and both operations preserve α-equivalence. We will now provide some proof details for selected auxiliary results:

Extension of Lemma 3.1.12 Proof by induction on the size of M:

(M is $\mathsf{fr}\, a.\, N$): We have by definition that

$$\pi \cdot (\mathsf{fr}\, a.\, N) \stackrel{\text{def}}{=} \mathsf{fr}\, \pi(a).\, \pi \cdot N$$

Further, we can deduce the following:

$$
\begin{aligned}
& \pi * (\mathsf{fr}\, a.\, N)\{\pi^{-1}\_\} \\
\stackrel{\text{def}}{=}\ & \pi * (\mathsf{fr}\, a'.\, ((a'\ a) \cdot N)\{\pi^{-1}\_\}) && (\text{for } a' \# (\pi, N)) \\
\stackrel{\text{def}}{=}\ & \mathsf{fr}\, a''.\, \pi * [(a''\ a') \cdot ((a'\ a) \cdot N)\{\pi^{-1}\_\}] && (\text{for } a'' \# (a, a', N, \pi)) \\
=\ & \mathsf{fr}\, a''.\, \pi * [(a''\ a') * ((a'\ a) \cdot N)\{\pi^{-1}\_\}\{(a''\ a')\_\}] && (\text{induction}) \\
=\ & \mathsf{fr}\, a''.\, \pi \circ (a''\ a') * ((a'\ a) \cdot N)\{\pi^{-1} \circ (a''\ a')\_\} \\
=\ & \mathsf{fr}\, a''.\, ((a''\ a') \circ \pi) * ((a'\ a) \cdot N)\{((a''\ a') \circ \pi)^{-1}\_\} && (a', a'' \# \pi) \\
=\ & \mathsf{fr}\, a''.\, ((a''\ a') \circ \pi) \cdot ((a'\ a) \cdot N) && (\text{induction}) \\
=\ & \mathsf{fr}\, a''.\, ((a''\ a') \circ \pi \circ (a'\ a)) \cdot N \\
=\ & \mathsf{fr}\, a''.\, (\pi \circ (a''\ a)) \cdot N
\end{aligned}
$$

What remains to be shown is that $\mathsf{fr}\, \pi(a).\, \pi \cdot N \equiv_\alpha \mathsf{fr}\, a''.\, (\pi \circ (a''\ a)) \cdot N$ holds, which follows immediately by definition, using basic permutation operations and the freshness assumption for a''. Hence, we have that both α-equivalence classes are equal and therefore

$$\pi \cdot (\mathsf{fr}\, a.\, N) = \pi * (\mathsf{fr}\, a.\, N)\{\pi^{-1}\_\}$$

□

Extension of Proposition 3.1.14 Proof by induction on the size of M:

(M is $\mathsf{fr}\, a.\, P$):

$$
\begin{aligned}
(\pi * \mathsf{fr}\, a.\, P)\{N/x\} \stackrel{\text{def}}{=}\ & \mathsf{fr}\, a'.\, (\pi * ((a'\ a) \cdot P))\{N/x\} && (a' \# (a, \pi, P, N)) \\
=\ & \mathsf{fr}\, a'.\, \pi * ((a'\ a) \cdot P)\{N/x\} && (\text{induction}) \\
\stackrel{\text{def}}{=}\ & \pi * (\mathsf{fr}\, a.\, P)\{N/x\} && (a' \# (a, \pi, P, N))
\end{aligned}
$$

□

Extension of Proposition 3.1.15 Proof by induction on the size of M:

(M is $\mathsf{fr}\, a.\, P$):

$$\begin{aligned}\pi \cdot (\mathsf{fr}\, a.\, P)\{N/x\} &\stackrel{\text{def}}{=} \mathsf{fr}\, \pi \cdot a.\, \pi \cdot P\{N/x\} \\ &= \mathsf{fr}\, \pi \cdot a.\, (\pi \cdot P)\{\pi \cdot N/x\} \qquad \text{(induction)} \\ &\stackrel{\text{def}}{=} (\pi \cdot \mathsf{fr}\, a.\, P)\{\pi \cdot N/x\}\end{aligned}$$

□

Typed Expressions and Equational Theories of ИNLC

To capture the notion of local fresh atomic names and its properties, as presented in Section 2.2, we extend NLC[$\mathbb{A}$] with new type and equation rules (see Table 4.7).

We begin with type rule (LFAN), which is directly motivated by Theorem 2.2.15 (Freshness Theorem), or more precisely the corresponding freshness condition

$$(И a)\ [F(a)\downarrow\ \wedge\ a \#\ F(a)]$$

which is captured in the rule by $\nabla^{\#a} \vdash^{\text{ИNLC}} a \#\ M : s$. The rule expresses that once we know that the name a is not in the element denoted by M (given that a is not in the support of any parameter upon which M depends), we can then form the expression $\mathsf{fr}\, a.\, M$. As explained in Section 2.2, we take this expression to stand for "$M(a)$ for some/any name a". Moreover, in computations we want to be able to replace $\mathsf{fr}\, a.\, M$ by M provided that a is sufficiently fresh for parameters currently in context (up to α-equivalence of $\mathsf{fr}\, a.\, M$). The corresponding property of the Freshness theorem, namely

$$(И a)\ [F(a) \equiv \text{fresh}_X F]$$

is captured by the equation rule (LFANR). In addition, we introduce equation rules (LFANS) and (LFANL), which capture standard properties of a local scoping operator.

$$(\mathsf{LFAN})\quad \frac{\nabla^{\#a} \vdash^{\text{ИNLC}} a \# M : s}{\nabla \vdash^{\text{ИNLC}} \mathsf{fr}\, a.\, M : s}\,(a \# \nabla)$$

$$(\mathsf{LFANR})\quad \frac{\nabla^{\#a} \vdash^{\text{ИNLC}} a \# M : s}{\nabla^{\#a} \vdash^{\text{ИNLC}} \mathsf{fr}\, a.\, M \approx M : s}\,(a \# \nabla)$$

$$(\mathsf{LFANS})\quad \frac{\nabla^{\#a,a'} \vdash^{\text{ИNLC}} a, a' \# M : s}{\nabla \vdash^{\text{ИNLC}} \mathsf{fr}\, a.\, \mathsf{fr}\, a'.\, M \approx \mathsf{fr}\, a'.\, \mathsf{fr}\, a.\, M : s}\,(a \neq a')$$

$$(\mathsf{LFANL})\quad \frac{\nabla^{\#a} \vdash^{\text{ИNLC}} a \# M : s}{\nabla \vdash^{\text{ИNLC}} \mathsf{fr}\, a'.\, \lambda^{\overline{a}} x : s.\, M \approx \lambda^{\overline{a}} x : s.\, \mathsf{fr}\, a'.\, M : s^{\overline{a}} \Rightarrow s'}\,(a' \notin \overline{a})$$

Table 4.7: ИNLC Equation Rules

A Sound Model-theoretic Semantics in *FMSet*

We begin with the interpretation of $\mathsf{fr}\, a.\, M$, which can be defined as follows:

$$\mathcal{M}[\![\mathsf{fr}\, a.\, M]\!]_\eta \stackrel{\text{def}}{=} \mathcal{M}[\![(a'\, a) \cdot M]\!]_\eta \qquad \text{for } a' \# (\eta, a, M)$$

To demonstrate that the interpretation of $\mathsf{fr}\, a.\, M$ is well defined, it has to be shown that the interpretation is independent of the choice of a'. At this point, we recall that the independence property can directly be deduced using Theorem 2.2.15 (Freshness Theorem). Hence, the partial interpretation of NLC[$\mathbb{A}$] can also be defined as follows:

$$\mathcal{M}[\![\mathsf{fr}\, a.\, M]\!]_\eta \stackrel{\text{def}}{=} \text{fresh } a' \text{ in } \mathcal{M}[\![(a'\, a) \cdot M]\!]_\eta$$

Note that the interpretation of $\mathsf{fr}\, a.\, M$ is partial since the side condition of the Freshness theorem needs to be satisfied. However, this is not an issue, because for well typed expressions, analogous to (AP) and (CONC), the side condition is satisfied by rule (LFAN) and therefore the interpretation is total for well typed expressions.

Before we extend the soundness theorem we provide proof details for two auxiliary results:

Extension of Lemma 3.4.7 The proof is by induction on the size of M: We only concentrate on the equational part of the Kleene equality.

(M is $\mathsf{fr}\, a.\, N$)

$$\begin{aligned}
\pi \cdot [\![\mathsf{fr}\, a.\, N]\!]_\eta &\stackrel{\text{def}}{=} \pi \cdot [\![(a'\, a) \cdot N]\!]_\eta && (\text{for } a' \# (a, N, \eta)) \\
&= [\![\pi \cdot (a'\, a) \cdot N]\!]_{\pi\cdot\eta} && (\text{induction}) \\
&= [\![(\pi(a')\, \pi(a)) \cdot \pi \cdot N]\!]_{\pi\cdot\eta} \\
&\stackrel{\text{def}}{=} [\![\mathsf{fr}\, \pi(a).\, \pi \cdot N]\!]_{\pi\cdot\eta} && (\text{for } \pi \cdot a' \# (\pi(a), \pi \cdot N, \pi \cdot \eta)) \\
&\stackrel{\text{def}}{=} [\![\pi \cdot (\mathsf{fr}\, a.\, N)]\!]_{\pi\cdot\eta}
\end{aligned}$$

Extension of Lemma 3.4.6 Proof by induction on the size of M: We only concentrate on the equational part of the Kleene equality.

(M is $\mathsf{fr}\, a.\, N$)

$$\begin{aligned}
[\![\pi * (\mathsf{fr}\, a.\, N)]\!]_\eta &\stackrel{\text{def}}{=} [\![\mathsf{fr}\, a''.\, \pi * ((a\, a'') \cdot N)]\!]_\eta && (\text{for } a'' \# (N, \pi, a)) \\
&\stackrel{\text{def}}{=} [\![(a''\, a') \cdot (\pi * ((a\, a'') \cdot N))]\!]_\eta && (\text{for } a' \# (\eta, N, \pi, a'', a)) \\
&= (a''\, a') \cdot [\![\pi * ((a\, a'') \cdot N)]\!]_{(a''\, a')\cdot\eta} && (\text{Lemma 3.4.7}) \\
&= (a''\, a') \cdot \pi \cdot [\![(a\, a'') \cdot N]\!]_{(a''\, a')\cdot\eta} && (\text{induction}) \\
&= \pi \cdot (a''\, a') \cdot [\![(a\, a'') \cdot N]\!]_{(a''\, a')\cdot\eta} && (a', a'' \# \pi) \\
&= \pi \cdot [\![(a''\, a') \cdot (a\, a'') \cdot N]\!]_\eta && (\text{Lemma 3.4.7}) \\
&= \pi \cdot [\![(a'\, a) \cdot (a''\, a') \cdot N]\!]_\eta \\
&= \pi \cdot [\![(a'\, a) \cdot M]\!]_\eta && (a'', a' \# N) \\
&\stackrel{\text{def}}{=} \pi \cdot [\![\mathsf{fr}\, a.\, N]\!]_\eta && (a' \# (a, M, \eta))
\end{aligned}$$

Extension of Soundness Theorem for ИNLC: The proof by mutual induction for NLC[$\mathbb{A}$] is extended with cases for the new type and equation rules:

(LFAN): Suppose that $\nabla \vdash^{\text{ИNLC}} \mathsf{fr}\, a.\, M : s$. From this we can deduce that $\nabla^{\#a} \vdash^{\text{ИNLC}} a \# M : s$ and $a \# (\nabla, s)$. Let $\eta \models \nabla$. We have to demonstrate that $[\![\mathsf{fr}\, a.\, M]\!]_\eta{\Downarrow}$

and $[\![\mathsf{fr}\, a.\, M]\!]_\eta \in [\![s]\!]$. By definition of the interpretation function we have to show that fresh a' in $[\![(a'\, a) \cdot M]\!]_\eta$ exists. Thus, we have to show that the precondition of Theorem 2.2.15 (Freshness Theorem) holds:

We begin by demonstrating that the function $F \stackrel{\text{def}}{=} \Lambda a' \in \mathbb{A}.[\![(a'\, a) \cdot M]\!]_\eta$ is finitely supported. Let $c, c' \notin \{a\} \cup \mathit{supp}(M) \cup \mathit{supp}(\eta)$ and $b \in \mathbb{A}$.

$$\begin{aligned}
((c\, c') \cdot F)(b) &\stackrel{\text{def}}{=} (c\, c') \cdot F((c\, c')(b)) \\
&\stackrel{\text{def}}{=} (c\, c') \cdot [\![((c\, c')(b)\, a) \cdot M]\!]_\eta \\
&= [\![(c\, c') \cdot ((c\, c')(b)\, a) \cdot M]\!]_{(c\, c') \cdot \eta} && \text{(Lemma 3.4.7)} \\
&= [\![(b\, a) \cdot (c\, c') \cdot M]\!]_\eta && (c, c' \# (a, \eta)) \\
&= [\![(b\, a) \cdot M]\!]_\eta && (c, c' \# (M)) \\
&\stackrel{\text{def}}{=} F(b)
\end{aligned}$$

We continue by showing that there exists a name $a' \in \mathbb{A}$ such that $a' \# F$ and $a' \# F(a')$. We pick $a' \# (a, \nabla, M, s, \eta)$. We can now directly deduce that $a' \# F$. What remains to be shown is that $a' \# F(a')$. Given that $\eta \models \nabla$ and $a' \# (\nabla, \eta)$, we have that $\eta \models \nabla^{\#a'}$. Then, by Lemma 3.4.10 and $a, a' \# \nabla$, we deduce that

$$(a'\, a) \cdot \eta \models (a'\, a) \cdot \nabla^{\#a'} = ((a'\, a) \cdot \nabla)^{\#a} = \nabla^{\#a}$$

Next, by induction on $\nabla^{\#a} \vdash^{\text{ИNLC}} a \# M : s$ we obtain $[\![M]\!]_{(a'\, a) \cdot \eta}\Downarrow$ and $[\![M]\!]_{(a'\, a) \cdot \eta} \in [\![s]\!]^{\#a}$. Further, given that $a', a \# s$ and by applying Lemma 3.4.7, we have that

$$F(a') \stackrel{\text{def}}{=} [\![(a'\, a) \cdot M]\!]_\eta = (a'\, a) \cdot [\![M]\!]_{(a'\, a) \cdot \eta} \in (a'\, a) \cdot [\![s]\!]^{\#a} \stackrel{\text{def}}{=} [\![(a'\, a) \cdot s]\!]^{\#a'} = [\![s]\!]^{\#a'}$$

So, we have that $F(a')$ exists and $F(a') \in [\![s]\!]^{\#a'}$. Then, by definition, we have that $a' \# F(a')$ holds and the second condition is satisfied.

(LFANR): Suppose that $\nabla^{\#a} \vdash^{\text{ИNLC}} \mathsf{fr}\, a.\, M \approx M : s$. From this we can deduce that $\nabla^{\#a} \vdash^{\text{ИNLC}} a \# M : s$ and $a \# (\nabla, s)$. Let $\eta \models \nabla^{\#a}$. So, by definition, we have that $a \# \eta$ holds. The existence part of the Kleene equality follows similarly to (LFAN). Hence, we have that $[\![\mathsf{fr}\, a.\, M]\!]_\eta\Downarrow$ and $[\![\mathsf{fr}\, a.\, M]\!]_\eta \in [\![s]\!]^{\#a}$.

By definition of the partial interpretation function we have that $[\![\mathsf{fr}\, a.\, M]\!]_\eta \stackrel{\text{def}}{=}$ fresh a' in $F(a')$ ($\diamond 1$) . We pick $a'' \# (a, M, \eta)$. Hence, we have that $a'' \# F$ holds and by Lemma 3.4.7 (Equivariance of $[\![(-)]\!]_{(-)}$) we get that $a'' \# [\![M]\!]_\eta$ holds. Then, by Theorem 2.2.15 (Freshness Theorem) we obtain that $F(a'') =$ fresh a' in $F(a')$ ($\diamond 2$). Moreover, we can deduce ($\diamond 3$) that

$$\begin{aligned} F(a'') &\stackrel{\text{def}}{=} [\![(a''\, a) \cdot M]\!]_\eta \\ &= (a''\, a) \cdot [\![M]\!]_{(a''\, a)\cdot\eta} && \text{(Lemma 3.4.7)} \\ &= (a''\, a) \cdot [\![M]\!]_\eta && (a, a'' \# \eta) \\ &= [\![M]\!]_\eta && (a, a'' \# [\![M]\!]_\eta) \end{aligned}$$

Using ($\diamond 1$), ($\diamond 2$) and ($\diamond 3$), we can complete this case:

$$[\![\mathsf{fr}\, a.\, M]\!]_\eta = \text{fresh } a' \text{ in } F(a') = F(a'') = [\![M]\!]_\eta$$

(LFANS): Suppose that $\nabla \vdash^{\text{NLC}} \mathsf{fr}\, a.\, \mathsf{fr}\, a'.\, M \approx \mathsf{fr}\, a'.\, \mathsf{fr}\, a.\, M : s$. From this we can deduce that $\nabla^{\#a,a'} \vdash^{\text{NLC}} \{a, a'\} \# M : s$ and $a \neq a'$. Let $\eta \models \nabla$. The existence part of the Kleene equality follows similarly to (LFAN). We continue with the equational part:

$$\begin{aligned} & [\![\mathsf{fr}\, a.\, \mathsf{fr}\, a'.\, M]\!]_\eta \\ &\stackrel{\text{def}}{=} \text{fresh } b \text{ in } (b\, a) \cdot [\![\mathsf{fr}\, a'.\, M]\!]_{(b\, a)\cdot\eta} && \text{(Lemma 3.4.7)} \\ &\stackrel{\text{def}}{=} \text{fresh } b \text{ in } (b\, a) \cdot [\text{fresh } b' \text{ in } (b'\, a') \cdot [\![M]\!]_{(b'\, a')\cdot(b\, a)\cdot\eta}] && \text{(Lemma 3.4.7)} \\ &= (c\, a) \cdot [(c'\, a') \cdot [\![M]\!]_{(c'\, a')\cdot(c\, a)\cdot\eta}] && (c, c' \# (a, a', M, \eta);\ c \neq c') \\ &= (c'\, a') \cdot [(c\, a) \cdot [\![M]\!]_{(c\, a)\cdot(c'\, a')\cdot\eta}] && (a, a', c, c' \text{ are distinct}) \\ &= \text{fresh } b' \text{ in } (b'\, a') \cdot [\text{fresh } b \text{ in } (b\, a) \cdot [\![M]\!]_{(b\, a)\cdot(b'\, a')\cdot\eta}] && (c, c' \# (a, a', M, \eta);\ c \neq c') \\ &\stackrel{\text{def}}{=} \text{fresh } b' \text{ in } (b'\, a') \cdot [\![\mathsf{fr}\, a.\, M]\!]_{(b'\, a')\cdot\eta} && \text{(Lemma 3.4.7)} \\ &\stackrel{\text{def}}{=} [\![\mathsf{fr}\, a'.\, \mathsf{fr}\, a.\, M]\!]_\eta && \text{(Lemma 3.4.7)} \end{aligned}$$

(LFANL): Suppose that $\nabla \vdash^{\text{NLC}} \mathsf{fr}\, a'.\, \lambda^{\overline{a}} x : s.\, M \approx \lambda^{\overline{a}} x : s.\, \mathsf{fr}\, a'.\, M : s^{\overline{a}} \Rightarrow s'$. From this we can deduce that $\nabla^{\#a'}, a' \cup \overline{a} \# x : s \vdash^{\text{NLC}} a' \# M : s'$ and $a' \notin \overline{a}$. Let $\eta \models \nabla$

and $d \in [\![s]\!]^{\#\overline{a}}$. The existence part of the Kleene equality follows similarly to (LFAN). The equational part is deduced as follows:

$$\begin{aligned}
[\![\mathsf{fr}\, a'.\, \lambda^{\overline{a}} x : s.\, M]\!]_\eta(d) &\stackrel{\text{def}}{=} [\text{fresh } b \text{ in } (a'\, b) \cdot [\![\lambda^{\overline{a}} x : s.\, M]\!]_{(b\, a')\cdot\eta}](d) && \text{(Lemma 3.4.7)} \\
&= [(c\, a') \cdot [\![\lambda^{\overline{a}} x : s.\, M]\!]_{(c\, a')\cdot\eta}](d) && (c \,\#\, (a', \overline{a}, s, M, \eta)) \\
&\stackrel{\text{def}}{=} (c\, a') \cdot [\![\lambda^{\overline{a}} x : s.\, M]\!]_{(c\, a')\cdot\eta}((c\, a') \cdot d) \\
&\stackrel{\text{def}}{=} (c\, a') \cdot [\![M]\!]_{((c\, a')\cdot\eta)[x \mapsto (c\, a')\cdot\eta]} \\
&= (c\, a') \cdot [\![M]\!]_{(c\, a')\cdot(\eta[x \mapsto (c\, a')\cdot\eta])} \\
&\stackrel{\text{def}}{=} [\![\mathsf{fr}\, a'.\, M]\!]_{\eta[x \mapsto d]} && \text{(Lemma 3.4.7)} \\
&\stackrel{\text{def}}{=} [\![\lambda^{\overline{a}} x : s.\, \mathsf{fr}\, a'.\, M]\!]_\eta(d)
\end{aligned}$$

Hence, we have that $[\![\mathsf{fr}\, a'.\, \lambda^{\overline{a}} x : s.\, M]\!]_\eta = [\![\lambda^{\overline{a}} x : s.\, \mathsf{fr}\, a'.\, M]\!]_\eta$. □

4.4 Examples of NLC[𝔸] and ⧸NLC Expressions

We begin by providing some basic examples of NLC[𝔸] expressions-in-context:

$$\emptyset \vdash^{\text{NLC}[\mathbb{A}]} \lambda x : s.\, \langle a \rangle x : s \Rightarrow \mathit{Name}.s \tag{4.1}$$

$$\emptyset \,\#\, x : \mathit{Name}.s \not\vdash^{\text{NLC}[\mathbb{A}]} x \mathbin{@} a : s \tag{4.2}$$

$$a \,\#\, x : \mathit{Name}.s \vdash^{\text{NLC}[\mathbb{A}]} x \mathbin{@} a : s \tag{4.3}$$

$$\emptyset \,\#\, x : s \vdash^{\text{NLC}[\mathbb{A}]} \langle a \rangle x \mathbin{@} a : s \tag{4.4}$$

$$b \,\#\, x : s \vdash^{\text{NLC}[\mathbb{A}]} \langle a \rangle x \mathbin{@} b : s \tag{4.5}$$

By applying rule (ABS), (NABS) and (SP) we can easily deduce (4.1). The fact that (4.2) cannot be derived follows by a semantic argument in combination with applying the contrapositive of the soundness lemma for NLC[𝔸]. The next expression-in-context is derived by applying rules (CONC), (SP) and (SUSP). Further, by additionally applying rule (NABS), we can also derive (4.4) and (4.5).

We continue by introducing three constructions of the FM-set model that require name abstraction, concretion and local fresh atomic names to be defined and then demonstrate how ИNLC can be used to capture these constructions.

Example 4.4.1 *The morphism component of the name abstraction functor* $[\mathbb{A}]- : FMSet \to FMSet$ *is defined as follows (see Lemma 4.10 in [56]):*

$$H : (X \Rightarrow_{fs} Y) \to ([\mathbb{A}]X \Rightarrow_{fs} [\mathbb{A}]Y)$$
$$H \stackrel{\text{def}}{=} \Lambda F \in X \Rightarrow_{fs} Y \to \Lambda d \in [\mathbb{A}]X \to \mathit{fresh}\ a\ \mathit{in}\ \langle a\rangle(F(d \mathbin{@} a))$$

Example 4.4.2 *The isomorphism* $[\mathbb{A}](X \Rightarrow_{fs} Y) \cong [\mathbb{A}]X \Rightarrow_{fs} [\mathbb{A}]Y$ *([34] (Corollary 9.6.9)) is observed by* I *and* J*, which are defined as follows:*

$$I : [\mathbb{A}](X \Rightarrow_{fs} Y) \to ([\mathbb{A}]X \Rightarrow_{fs} [\mathbb{A}]Y)$$
$$I \stackrel{\text{def}}{=} \Lambda u \in [\mathbb{A}](X \Rightarrow_{fs} Y) \to \Lambda z \in [\mathbb{A}]X \to \mathit{fresh}\ a\ \mathit{in}\ \langle a\rangle x((u \mathbin{@} a)(z \mathbin{@} a))$$
$$J : ([\mathbb{A}]X \Rightarrow_{fs} [\mathbb{A}]Y) \to [\mathbb{A}](X \Rightarrow_{fs} Y)$$
$$J \stackrel{\text{def}}{=} \Lambda F \in [\mathbb{A}]X \Rightarrow_{fs} [\mathbb{A}]Y \to \mathit{fresh}\ a\ \mathit{in}\ \langle a\rangle(\Lambda x \in X \to F(\langle a\rangle x) \mathbin{@} a)$$

We propose expressions h, i and j as the respective counter parts of H, I and J:

$$h : (s \Rightarrow s') \Rightarrow Name.s \Rightarrow Name.s'$$
$$h \stackrel{\text{def}}{=} \lambda f : s \Rightarrow s'.\, \lambda x : Name.s.\, \alpha a.(f(x \mathbin{@} a))$$
$$i : (Name.s \Rightarrow s') \Rightarrow Name.s \Rightarrow Name.s'$$
$$i \stackrel{\text{def}}{=} \lambda x : Name.s \Rightarrow s'.\, \lambda y : Name.s.\, \alpha a.((x \mathbin{@} a)(y \mathbin{@} a))$$
$$j : (Name.s \Rightarrow Name.s') \Rightarrow Name.(s \Rightarrow s')$$
$$j \stackrel{\text{def}}{=} \lambda f : Name.s \Rightarrow Name.s'.\, \mathsf{fr}\, a.\, \langle a\rangle(\lambda x : s.\, f(\langle a\rangle x) \mathbin{@} a)$$

and then demonstrate that all expressions are well typed in ИNLC and that their interpretation, using the model-theoretic semantics in *FMSet*, correspond to H, I and J.

- h **is well defined and captures** H**:** By rule (ABS) we have to show that $\emptyset \# f : s \Rightarrow s', \emptyset \# x : Name.s \vdash^{\text{ИNLC}} \alpha a.(f\ (x \mathbin{@} a)) : Name.s'$ holds. Recall that $\alpha a.(f\ (x \mathbin{@} a)) \stackrel{\text{def}}{=} \mathsf{fr}\, a.\, \langle a \rangle (f\ (x \mathbin{@} a))$. We can assume, up to α-equivalence, that $a \# (\emptyset \# f : s \Rightarrow s', \emptyset \# x : Name.s, Name.s')$. So, by (LFAN) we need to show that $a \# f : s \Rightarrow s', a \# x : Name.s \vdash^{\text{ИNLC}} a \# \langle a \rangle (f\ (x \mathbin{@} a)) : Name.s'$. The freshness assertion follows immediately from Lemma 4.1.2 once we have shown that $a \# f : s \Rightarrow s', a \# x : Name.s \vdash^{\text{ИNLC}} \langle a \rangle (f\ (x \mathbin{@} a)) : Name.s'$. We now apply (NABS) and show that $a \# f : s \Rightarrow s', a \# x : Name.s \vdash^{\text{ИNLC}} f\ (x \mathbin{@} a) : Name.s'$ holds. Then we apply (AP). The left hand side of the rule follows immediately by (SP) and for the right hand side we apply (CONC). The corresponding typed expressions and freshness assertion follows directly from (SP) and (SUSP).

$$
\begin{aligned}
&\llbracket \lambda f : s \Rightarrow s'.\, \lambda x : Name.s.\, \alpha a.(f(x \mathbin{@} a)) \rrbracket_{\epsilon} \\
&\stackrel{\text{def}}{=} \llbracket \lambda f : s \Rightarrow s'.\, \lambda x : Name.s.\, \mathsf{fr}\, a.\, \langle a \rangle (f(x \mathbin{@} a)) \rrbracket_{\epsilon} \\
&\stackrel{\text{def}}{=} \Lambda F \in \llbracket s \rrbracket \Rightarrow_{fs} \llbracket s' \rrbracket \to \Lambda d \in [\mathbb{A}]\llbracket s \rrbracket \to \llbracket \mathsf{fr}\, a.\, \langle a \rangle (f\ (x \mathbin{@} a)) \rrbracket_{[f \mapsto F][x \mapsto d]} \\
&\stackrel{\text{def}}{=} \Lambda F \to \Lambda d \to \text{fresh } a' \text{ in } \llbracket (a'\ a) \cdot \langle a \rangle (f\ (x \mathbin{@} a)) \rrbracket_{[f \mapsto F], [x \mapsto d]} \\
&\stackrel{\text{def}}{=} \Lambda F \to \Lambda d \to \text{fresh } a' \text{ in } \llbracket \langle a' \rangle (f\ (x \mathbin{@} a')) \rrbracket_{[f \mapsto F], [x \mapsto d]} \\
&\stackrel{\text{def}}{=} \Lambda F \in \llbracket s \rrbracket \Rightarrow_{fs} \llbracket s' \rrbracket \to \Lambda d \in [\mathbb{A}]\llbracket s \rrbracket \to \text{fresh } a' \text{ in } \langle a' \rangle F(d \mathbin{@} a') \\
&= H
\end{aligned}
$$

- i **is well defined and captures** I**:** By rule (ABS) we have to show that $\emptyset \# x : Name.s \Rightarrow s', \emptyset \# y : Name.s \vdash^{\text{ИNLC}} \alpha a.((x \mathbin{@} a)\ (y \mathbin{@} a)) : Name.s'$ holds. Recall that $\alpha a.((x \mathbin{@} a)\ (y \mathbin{@} a)) \stackrel{\text{def}}{=} \mathsf{fr}\, a.\, \langle a \rangle ((x \mathbin{@} a)\ (y \mathbin{@} a))$. We can assume, up to α-equivalence, that $a \# (\emptyset \# x : Name.s \Rightarrow s', \emptyset \# y : Name.s)$. So, by (LFAN), we have to show that $a \# x : Name.s \Rightarrow s', a \# y : Name.s \vdash^{\text{ИNLC}} a \# \langle a \rangle ((x \mathbin{@} a)\ (y \mathbin{@} a)) : Name.s'$ holds. The freshness assertion follows immediately by Lemma 4.1.2 once we have show that $a \# x : Name.s \Rightarrow s', a \#$

$y : Name.s \vdash^{\text{NLC}} \langle a\rangle((x \mathbin{@} a)\ (y \mathbin{@} a)) : Name.s'$ holds. Next, we apply (NABS), and show that $a \# x : Name.s \Rightarrow s', a \# y : Name.s \vdash^{\text{NLC}} (x \mathbin{@} a)\ (y \mathbin{@} a) : s'$ holds. We first apply (AP) and then apply (CONC). The corresponding typed expressions and freshness assertion follow directly from (SP) and (SUSP).

$$
\begin{aligned}
&[\![\lambda x : Name.s \Rightarrow s'.\, \lambda y : Name.s.\, \alpha a.((x \mathbin{@} a)(y \mathbin{@} a))]\!]_\epsilon \\
&\stackrel{\text{def}}{=} [\![\lambda x : Name.s \Rightarrow s'.\, \lambda y : Name.s.\, \mathsf{fr}\, a.\, \langle a\rangle((x \mathbin{@} a)(y \mathbin{@} a))]\!]_\epsilon \\
&\stackrel{\text{def}}{=} \Lambda d \in [\mathbb{A}]([\![s]\!] \Rightarrow_{fs} [\![s']\!]) \to \Lambda d' \in [\mathbb{A}][\![s]\!] \to [\![\mathsf{fr}\, a.\, \langle a\rangle((x \mathbin{@} a)(y \mathbin{@} a))]\!]_{[x\mapsto d][y\mapsto d']} \\
&\stackrel{\text{def}}{=} \Lambda d \to \Lambda d' \to \text{fresh } a' \text{ in } [\![(a'\, a) \cdot \langle a\rangle((x \mathbin{@} a)(y \mathbin{@} a))]\!]_{[x\mapsto d][y\mapsto d']} \\
&\stackrel{\text{def}}{=} \Lambda d \to \Lambda d' \to \text{fresh } a' \text{ in } [\![\langle a\rangle((x \mathbin{@} a')(y \mathbin{@} a'))]\!]_{[x\mapsto d][y\mapsto d']} \\
&\stackrel{\text{def}}{=} \Lambda d \in [\mathbb{A}]([\![s]\!] \Rightarrow_{fs} [\![s']\!]) \to \Lambda d' \in [\mathbb{A}][\![s]\!] \to \text{fresh } a' \text{ in } \langle a'\rangle((d \mathbin{@} a')(d' \mathbin{@} a')) \\
&= I
\end{aligned}
$$

- j **is well defined and captures** J**:** By rule (ABS) we have to show that $\emptyset \# f : Name.s \Rightarrow Name.s' \vdash^{\text{NLC}} \mathsf{fr}\, a.\, \langle a\rangle(\lambda x : s.\, f(\langle a\rangle x) \mathbin{@} a) : Name.s \Rightarrow s'$ holds. Note that for the definition of j we implicitly assumed that $a \notin supp((s, s'))$. By (LFAN) we have to show that $a \# f : Name.s \Rightarrow Name.s' \vdash^{\text{NLC}} a \# \langle a\rangle(\lambda x : s.\, f(\langle a\rangle x) \mathbin{@} a) : Name.s \Rightarrow s'$ holds. The freshness assertion follows immediately from Lemma 4.1.2 once we have shown that $a \# f : Name.s \Rightarrow Name.s' \vdash^{\text{NLC}} \langle a\rangle(\lambda x : s.\, f(\langle a\rangle x) \mathbin{@} a) : Name.s \Rightarrow s'$. Next, we apply (NABS) and demonstrate that $a \# f : Name.s \Rightarrow Name.s' \vdash^{\text{NLC}} \lambda x : s.\, f(\langle a\rangle x) \mathbin{@} a : s \Rightarrow s'$ holds. Then, by (ABS), we have to show that $a \# f : Name.s \Rightarrow Name.s', \emptyset \# x : s \vdash^{\text{NLC}} f(\langle a\rangle x) \mathbin{@} a : s'$. We continue by applying (CONC) and show that $a \# f : Name.s \Rightarrow Name.s', \emptyset \# x : s \vdash^{\text{NLC}} a \# f(\langle a\rangle x) : Name.s'$. So, by definition of freshness assertions, we have to prove that $\{a, c\} \# f : Name.s \Rightarrow Name.s', c \# x : s \vdash^{\text{NLC}} (a\, c) * f\ (\langle a\rangle x) \approx f\ (\langle a\rangle x) : s'$ holds. This

follows directly by application of the rules (CA), (SUSP) and (BNABS).

$$
\begin{aligned}
&[\![\lambda f : \mathit{Name}.s \Rightarrow \mathit{Name}.s'.\, \mathsf{fr}\, a.\, \langle a\rangle(\lambda x : s.\, f(\langle a\rangle x) \mathbin{@} a)]\!]_{\epsilon} \\
&\stackrel{\text{def}}{=} \Lambda F \in [\mathbb{A}][\![s]\!] \Rightarrow_{fs} [\mathbb{A}][\![s']\!] \to [\![\mathsf{fr}\, a.\, \langle a\rangle(\lambda x : s.\, f\ (\langle a\rangle x) \mathbin{@} a)]\!]_{[f \mapsto F]} \\
&\stackrel{\text{def}}{=} \Lambda F \in [\mathbb{A}][\![s]\!] \Rightarrow_{fs} [\mathbb{A}][\![s']\!] \to \text{fresh } a' \text{ in } [\![(a'\, a) \cdot \langle a\rangle(\lambda x : s.\, f\ (\langle a\rangle x) \mathbin{@} a)]\!]_{[f \mapsto F]} \\
&\stackrel{\text{def}}{=} \Lambda F \in [\mathbb{A}][\![s]\!] \Rightarrow_{fs} [\mathbb{A}][\![s']\!] \to \text{fresh } a' \text{ in } \langle a'\rangle(\Lambda d \in [\![s]\!].F(\langle a'\rangle d) \mathbin{@} a') \\
&= J
\end{aligned}
$$

Chapter 5

Towards a Categorical Type Theory Correspondence for NLC

The notion of FM-categories (which generalises *FMSet*) has been introduced by Clouston [13, 17] to prove a categorical type theory correspondence for NEL. The corresponding proof argument includes the definition of a sound categorical semantics for NEL and a classifying FM-category for any NEL-theory (with a generic model). This means that the categorical semantics can also be proved complete. For a general introduction to categorical logic, we refer the reader to [42, 20, 52]. An immediate question, which will only be partially answered in this chapter, is if these results can be extended to hold for NLC. Our aim in this chapter is to identify and approach the central issues involved in the construction of a classifying category for NLC. The chapter is organised as follows:

- We recall the notion of an FM-category and FM-functor, as well as some auxiliary results. We continue by extending FM-categories with equivariant exponentials, referred to as FM-cccs, and prove various auxiliary results.
- We analyse if certain categorical constructions "lift" some of the previously defined properties of an FM-ccc. These constructions play a key role in a

categorical proof argument (see Appendix B.1), which we intended to use to prove Conjecture 3.5.3 (conservative extension property).

- The notion of an FM-ccc will underpin our categorical semantics for NLC, which we demonstrate to be sound. We then approach the construction of a syntactically generated (classifying) category $Cl(Th)$ (for any NLC-theory Th), which is defined to be a term quotient structure using Th. We observe that the construction of exponentials in such a category constitutes a key challenge, explain why it is problematic in the case of NLC and propose two approaches to extend NLC such that exponentials can be constructed in $Cl(Th)$. The approach we ultimately pursue is semantically motivated by properties of *FMSet* and leads to an extension of NLC, referred to as [И]NLC. This has already been published in [23]. In this chapter, we provide a full verification argument to prove that the proposed construction is indeed an equivariant exponential in $Cl(Th)$. This is an important step towards constructing a syntactically generated classifying category $Cl(Th)$ for any theory Th of the semantically motivated calculus [И]NLC. However, there is more work required, which will not be part of this chapter. This will be worked out in full detail, as part of a journal version of [23], at a later point.

5.1 Preliminaries

5.1.1 FM-Categories and FM-Functors

We give an overview of FM-categories and FM-Functors, as well as some auxiliary results, which will be required in this chapter. For full details we refer to [13, 17]:

Definition 5.1.1 *A category $\mathcal{C}$ has an **internal permutation action** if for each $\pi \in Perm(\mathbb{A})$ and $C \in ob\,\mathcal{C}$ there is a $\mathcal{C}$-arrow π_C with domain C such that*

(i) ι_C *is the identity* id_C

(ii) $(\pi' \circ \pi)_C = \pi'_{\pi \cdot C} \circ \pi_C$, *where* $\pi \cdot C$ *is defined to be the codomain of* π_C.

Lemma 5.1.2 *Suppose* $\mathcal{C}$ *is a small category with an internal permutation action* $(\pi, C) \mapsto \pi_C$. *Then*

(i) The map $(\pi, C) \mapsto \pi \cdot C$, *with* $\pi \cdot C$ *being the codomain of* π_C, *defines a permutation action on* $ob\mathcal{C}$

(ii) Given a morphism $f : C \to D$ *in* $\mathcal{C}$, $\pi \cdot f : \pi \cdot C \to \pi \cdot D$ *is defined as*

$$\pi \cdot f =_{def} \pi_D \circ f \circ (\pi^{-1})_{\pi \cdot C}$$

Then the map $(\pi, f) \mapsto \pi \cdot f$ *defines the conjugational permutation action on arrows in* $\mathcal{C}$.

(iii) If every arrow in $ar\mathcal{C}$ *is finitely supported with respect to the conjugational permutation action given by (ii), then every object in* $ob\mathcal{C}$ *is also finitely supported.*

Lemma 5.1.3 *The following results hold for the internal permutation action:*

(i) The map $(\pi, C) \mapsto \pi_C$ *is equivariant, i.e. we have* $\pi \cdot (\pi'_C) = (\pi\pi'\pi^{-1})_{\pi \cdot C}$

(ii) Each $\pi_C : C \to \pi \cdot C$ *is an isomorphism, with the corresponding inverse* $\pi^{-1}_{\pi \cdot C} : \pi \cdot C \to C$

(iii) The maps $\pi \cdot (-)$ *on* $ob\mathcal{C}$ *and* $ar\mathcal{C}$ *define an endofunctor* $\mathcal{C} \to \mathcal{C}$

(iv) The internal permutation actions $\pi_C : C \to \pi \cdot C$ *are components of a natural transformation* $\pi : id_{\mathcal{C}} \to \pi \cdot (-)$, *where* $id_{\mathcal{C}}$ *is the identity functor on* $\mathcal{C}$.

Definition 5.1.4 *An internal permutation action on a category* $\mathcal{C}$ *is* ***finitely supported*** *if every arrow in* $\mathcal{C}$ *is finitely supported with respect to the permutation action defined by Lemma 5.1.2 (ii). We call a category with a finitely supported permutation action a* ***perm-category***.

Definition 5.1.5 *Let $\mathcal{C}$ and $\mathcal{C}'$ be two perm-categories. A functor $F : \mathcal{C} \to \mathcal{C}'$ that strictly preserves internal permutation actions (i.e. $F(\pi_C) = \pi_{FC}$), is called a* ***perm-functor****.*

Lemma 5.1.6 *A small perm-category is an internal category in Nom.*

The converse does not hold, as has been explained in [17] (see Remark 4.6).

Lemma 5.1.7 *Let $F : \mathcal{C} \to \mathcal{C}'$ be a perm-functor between two small perm-categories $\mathcal{C}$ and $\mathcal{C}'$. Then the following holds:*

(i) F is a functor internal in Nom;

(ii) Given two such functors F, F' , any natural transformation $\phi : F \to F'$ is a natural transformation in Nom.

Thus, the category of small perm-categories and perm-functors, *PermCat*, is a subcategory of $Int(Nom)$ (the category of categories and functors internal in *Nom*).

Definition 5.1.8 *Take a category $\mathcal{C}$ with an internal permutation action and finite products defined by a terminal object 1, and for each pair of objects C_1, C_2, a product $C_1 \times C_2$ with projections $pr_i(C_1, C_2) : C_1 \times C_2 \to C_i$ (for $i = 1, 2$). Then $\mathcal{C}$ has* ***equivariant finite products*** *if for all $\pi \in Perm$, we have that $\pi \cdot 1 = 1$ and $\pi \cdot pr_i(C_1, C_2) = pr_i(\pi \cdot C_1, \pi \cdot C_2)$ (for $i = 1, 2$). As a direct consequence $\pi \cdot (C_1 \times C_2) = \pi \cdot C_1 \times \pi \cdot C_2$.*

Definition 5.1.9 *Let $\mathcal{C}$ have a finitely supported internal permutation action and equivariant finite products. Then $\mathcal{C}$ has* ***fresh inclusions*** *if for every finite set of names $\overline{a} \subseteq \mathbb{A}$ and $\mathcal{C}$-object C such that $\overline{a} \# C$ there exists an object $C^{\#\overline{a}}$ and a morphism $i^{\overline{a}}_C : C^{\#\overline{a}} \to C$ in $\mathcal{C}$ for which the following properties hold:*

(i) (Equivariance): $\pi \cdot i^{\overline{a}}_C = i^{\pi \cdot \overline{a}}_{\pi \cdot C}$;

(ii) (Sets of Names): $i^{\emptyset}_C = id_C$ *and* $i^{\overline{a}}_C \circ i^{\overline{a}'}_{C\#\overline{a}} = i^{\overline{a}\cup\overline{a}'}_C$.

(iii) (Products): $i^{\overline{a}}_{C_1\times C_2} = i^{\overline{a}}_{C_1} \times i^{\overline{a}}_{C_2}$;

(iv) (Internal permutation action): if $supp(\pi)\ \#\ C$ *then* $\pi_{C\#supp(\pi)}$ *is equal to the identity* $id_{C\#supp(\pi)}$;

(v) (Epi When Fresh): If we have parallel $\mathcal{C}$*-arrows* $f, g : C \to D$ *such that* $f \circ i^{\overline{a}}_C = g \circ i^{\overline{a}}_C$ *and* $\overline{a}\ \#\ (f, g)$*, then* $f = g$;

(vi) (Freshness): Let $f : C \to D$ *be a morphism in* $\mathcal{C}$ *and* $\overline{a}$ *a finite set of names such that* $\overline{a}\ \#\ D$*. Define condition* $\dagger(\overline{a}, f) \stackrel{\text{def}}{=} (\exists \overline{b}\ \#\ (\overline{a}, f))\ ((\overline{a}\,\overline{b})_D \circ f \circ i^{\overline{b}}_C = f \circ i^{\overline{b}}_C)$*. If* $\dagger(\overline{a}, f)$ *holds, then there exists a unique morphism* $f^* : C \to D^{\#\overline{a}}$ *in* $\mathcal{C}$*, called the* ***image restriction*** *of* f*, such that* $i^{\overline{a}}_D \circ f^* = f$.

Remark 5.1.10 *Each of the freshness properties of an FM-category has a clear intuition once we see it in the context of FMSet. As an example we consider (Freshness), which is the most involved property. Let* $f : X \to Y$ *be finitely supported in FMSet and* $a\ \#\ Y$*. By choosing* $b\ \#\ (a, f)$ *and* $x \in X$ *such that* $b\ \#\ x$ *we have* $(f \circ i^b_C)(x) = f(x)$ *and the condition amounts to* $(a\,b) \cdot f(x) = f(x)$*. Hence, by definition we have* $f(x) \in Y^{\#a}$ *and so* f *image restricts with* $f^* : x \mapsto f(x)$.

Lemma 5.1.11 *Let* $\mathcal{C}$ *be a category with fresh inclusions. Then*

(i) If $\overline{a}\ \#\ (f : C \to D)$*, there is a unique arrow* $f^{\#\overline{a}} : C^{\#\overline{a}} \to D^{\#\overline{a}}$ *such that* $f \circ i = i \circ f^{\#\overline{a}}$

(ii) The assignment is functorial, i.e. $(id_C)^{\#\overline{a}} = id_{C^{\#\overline{a}}}$ *and* $(g \circ f)^{\#\overline{a}} = g^{\#\overline{a}} \circ f^{\#\overline{a}}$

Lemma 5.1.12 *Let* $\mathcal{C}$ *be a category with fresh inclusions. Then all fresh inclusions* $i^{\overline{a}}_C$ *are monomorphisms.*

Definition 5.1.13 *A category with finitely supported internal permutation action, equivariant finite products and fresh inclusions is said to be an* ***FM-category****. A functor F that strictly preserves internal permutation actions ($F(\pi_C) = \pi_{FC}$), finite products ($F(1_C) = 1_{FC}$ and $F(pr_i(C_1, C_2)) = pr_i(FC_1, FC_2)$) and fresh inclusions ($F(i^{\overline{a}}_C) = i^{\overline{a}}_{FC}$) is called an* ***FM-functor****.*

The fact that *FMSet* is an FM-category follows straightforwardly.

5.1.2 Equivariant Exponentials and FM-ccc

We introduce the notion of equivariant exponentials and prove various auxiliary results which are required to establish a sound categorical semantics in FM-ccc.

Definition 5.1.14 *Let $\mathcal{C}$ be a perm-category with equivariant finite products and exponentials, i.e. for any any objects B and C there is an object $B \Rightarrow C$ and a morphism $ev_{B,C} : (B \Rightarrow C) \times B \to C$ such that there exists a unique morphism $\lambda(f) : A \to (B \Rightarrow C)$ such that $f = ev \circ (\lambda(f) \times id_B)$ for any $f : A \times B \to C$. The category $\mathcal{C}$ has* ***equivariant exponentials*** *if the evaluation map is preserved by the internal permutation action, i.e. $\pi \cdot ev_{B,C} = ev_{\pi\cdot B,\pi\cdot C}$. Consequently, $\pi \cdot (B \Rightarrow C) = (\pi \cdot B \Rightarrow \pi \cdot C)$. An FM-category with equivariant exponential is called* ***FM-ccc*** *and a functor which preserves equivariant exponentials, i.e. $\lambda(F(ev_{B,C}) \circ \cong) = id_{FB\Rightarrow FC}$, is called an* ***FM-ccc functor****.*

We know that *FMSet* is cartesian closed (see Lemma 2.4.3). Next, we deduce that the exponential is equivariant, and therefore *FMSet* is an FM-ccc.

$$\begin{aligned}(\pi \cdot ev_{A,B})(f, x) &\stackrel{\text{def}}{=} \pi \cdot ev_{A,B}(\pi^{-1} \cdot f, \pi^{-1} \cdot x)\\ &\stackrel{\text{def}}{=} \pi \cdot ((\pi^{-1} \cdot f)(\pi^{-1} \cdot x))\\ &\stackrel{\text{def}}{=} f(x)\\ &\stackrel{\text{def}}{=} ev_{\pi\cdot A,\pi\cdot B}(f, x)\end{aligned}$$

We recall that for any cartesian closed category $\mathcal{C}$, there is an exponential functor $(-) \Rightarrow (-) : \mathcal{C}^{op} \times \mathcal{C} \to \mathcal{C}$:

$$(A, B) \mapsto A \Rightarrow B$$
$$(f, g) \mapsto f \Rightarrow g \stackrel{\text{def}}{=} \lambda(g \circ ev \circ (id_{A \Rightarrow B} \times f))$$

Lemma 5.1.15 *For any FM-cartesian closed category $\mathcal{C}$ the following properties hold:*

(a) For any $f : A \times B \to C$ and $g : A' \to A$, we have $\lambda(f) \circ g = \lambda(f \circ (g \times id_B))$

(b) $\pi_{A_i} \circ pr_i(A_1, A_2) = pr_i(\pi \cdot A_1, \pi \cdot A_2) \circ \pi_{A_1 \times A_2}$ $(i = 1, 2)$

(c) $\pi_B \circ ev_{A,B} = ev_{\pi \cdot A, \pi \cdot B} \circ (\pi_{A \Rightarrow B} \times \pi_A)$.

(d) $\pi \cdot \lambda(f) = \lambda(\pi \cdot f)$

(e) $\pi \cdot (f \Rightarrow g) = \pi \cdot f \Rightarrow \pi \cdot g$

(f) $\pi_{A \times B} = \pi_A \times \pi_B$

(g) $\pi_{A \Rightarrow B} = \pi^{-1}_{\pi \cdot A} \Rightarrow \pi_B$

(h) For any $f : A \times B \to C$ we have $\pi_{B \Rightarrow C} \circ \lambda(f) = \lambda(\pi_C \circ f \circ (id \times \pi^{-1}_{\pi \cdot B}))$

Proof The proofs follow by routine computations in category theory.

(a): Given that $\mathcal{C}$ is cartesian closed, we have that $\lambda(f \circ (g \times id_B))$ is the unique morphism such that $f \circ (g \times id_B) = ev \circ (\lambda(f \circ (g \times id_B)) \times id_B)$ $(\diamond 1)$. Moreover, we have that $\lambda(f)$ is the unique morphism such that $f = ev \circ (\lambda(f) \times id_B)$. From the previous equation we can deduce that $f \circ (g \times id_B) = ev \circ (\lambda(f) \times id_B) \circ (g \times id_B) = ev \circ (\lambda(f) \circ g \times id_B)$. Hence, by the universal property of $(\diamond 1)$ we have that $\lambda(f) \circ g = \lambda(f \circ (g \times id_B))$

(b, c): Follows directly from the fact that internal permutation actions are natural transformations (Lemma 5.1.3 (iv))

(d): Given that $\mathcal{C}$ is cartesian closed we have that $\lambda(\pi \cdot f)$ is the unique morphism such that $\pi \cdot f = ev \circ (\lambda(\pi \cdot f) \times id_{\pi \cdot B})$ ($\diamond$1). Moreover, we have that $\lambda(f)$ is the unique morphism such that $f = ev \circ (\lambda(f) \times id_B)$. From the previous equation and the fact that $\pi \cdot (-)$ is a functor and ev is equivariant, we obtain that $\pi \cdot f = ev \circ (\pi \cdot \lambda(f) \times id_{\pi \cdot B})$. Hence, by the universal property for ($\diamond$1) we have that $\pi \cdot \lambda(f) = \lambda(\pi \cdot f)$.

(e): It is deduced as follows:

$$\begin{aligned}
\pi \cdot (f \Rightarrow g) &\stackrel{\text{def}}{=} \pi \cdot (\lambda(g \circ ev \circ (id \times f))) \\
&= \lambda(\pi \cdot (g \circ ev \circ (id \times f))) && \text{(by (d))} \\
&= \lambda((\pi \cdot g) \circ (\pi \cdot ev) \circ (id \times (\pi \cdot f))) && (\circ \text{ is equivariant}) \\
&= \lambda((\pi \cdot g) \circ ev \circ (id \times (\pi \cdot f))) && (ev \text{ is equivariant}) \\
&\stackrel{\text{def}}{=} (\pi \cdot f) \Rightarrow (\pi \cdot g)
\end{aligned}$$

(f): For internal permutation actions π_A and π_B, there exists a unique morphism $\pi_A \times \pi_B$ such that

$$\begin{aligned}
\pi_A \circ pr_A &= pr_{\pi \cdot A} \circ (\pi_A \times \pi_B) \\
\pi_B \circ pr_B &= pr_{\pi \cdot B} \circ (\pi_A \times \pi_B)
\end{aligned}$$

Using property (b) from above, we directly obtain that

$$\begin{aligned}
pr_{\pi \cdot A} \circ \pi_{A \times B} &= pr_{\pi \cdot A} \circ (\pi_A \times \pi_B) \\
pr_{\pi \cdot B} \circ \pi_{A \times B} &= pr_{\pi \cdot B} \circ (\pi_A \times \pi_B)
\end{aligned}$$

Hence, we have that $\pi_{A \times B} = \pi_A \times \pi_B$.

(g): We have $\pi^{-1}_{\pi \cdot A} \Rightarrow \pi_B \stackrel{\text{def}}{=} \lambda(g)$ where $g \stackrel{\text{def}}{=} \pi_B \circ ev \circ (id_{A \Rightarrow B} \times \pi^{-1}_{\pi \cdot A}) : (A \Rightarrow$

$B) \times \pi \cdot A \to \pi \cdot B$. We can now deduce the following:

$$\begin{aligned} g &\stackrel{\text{def}}{=} \pi_B \circ ev_{A,B} \circ (id_{A \Rightarrow B} \times \pi^{-1}_{\pi \cdot A}) \\ &= ev_{\pi \cdot A, \pi \cdot B} \circ (\pi_{A \Rightarrow B} \times \pi_A) \circ (id_{A \Rightarrow B} \times \pi^{-1}_{\pi \cdot A}) && \text{(by (c))} \\ &= ev_{\pi \cdot A, \pi \cdot B} \circ (\pi_{A \Rightarrow B} \times \pi_A \circ \pi^{-1}_{\pi \cdot A}) \\ &= ev_{\pi \cdot A, \pi \cdot B} \circ (\pi_{A \Rightarrow B} \times id_{\pi \cdot A}) && \text{(Lemma 5.1.3 (ii))} \end{aligned}$$

Hence, we have that $g = ev_{\pi \cdot A, \pi \cdot B} \circ (\pi_{A \Rightarrow B} \times id_{\pi \cdot A})$. Regarding the fact that $\lambda(g)$ is the unique map such that $g = ev \circ (\lambda(g) \times id_{\pi \cdot A})$, we have completed the argument.

(h):

$$\begin{aligned} \pi_{B \Rightarrow C} \circ \lambda(f) &= (\pi^{-1}_{\pi \cdot B} \Rightarrow \pi_C) \circ \lambda(f) && \text{(by (g))} \\ &\stackrel{\text{def}}{=} \lambda(\pi_C \circ ev \circ (id \times \pi^{-1}_{\pi \cdot B})) \circ \lambda(f) \\ &= \lambda(\pi_C \circ ev \circ (id \times \pi^{-1}_{\pi \cdot B}) \circ (\lambda(f) \times id)) && \text{(by (a))} \\ &= \lambda(\pi_C \circ ev \circ (\lambda(f) \times id) \circ (id \times \pi^{-1}_{\pi \cdot B})) \\ &\stackrel{\text{def}}{=} \lambda(\pi_C \circ f \circ (id \times \pi^{-1}_{\pi \cdot B})) \end{aligned}$$

□

5.1.3 Examples of FM-Constructions

We investigate if apart from *FMSet* there are other categorical constructions with the aforementioned properties of an FM-ccc. The constructions that we consider play a key role in our attempt to prove an additional conservative extension property for NEL and NLC via a categorical proof argument. An overview of the categorical proof argument can be found in Section B.1.

Freyd-style Glued Categories

We first recall the so called Freyd scone [42, 45, 21].

Theorem 5.1.16 *Suppose that $\mathcal{C}$ and $\mathcal{D}$ are cartesian closed categories, $\mathcal{C}$ has all pullbacks and $\Gamma : \mathcal{D} \to \mathcal{C}$ is a functor which strictly preserves finite products. Then the glued category $\mathcal{GL}(\Gamma) \stackrel{\text{def}}{=} (id_{\mathcal{C}} \downarrow \Gamma)$ with*

- *Objects: (C, f, D) where $C \in ob\,\mathcal{C}$, $D \in ob\,\mathcal{D}$ and $f : C \to \Gamma D$ is a morphism in $\mathcal{C}$.*
- *Morphisms: (h_1, h_2) where $h_1 : C \to C'$ and $h_2 : D \to D'$ are morphisms in $\mathcal{C}$ and $\mathcal{D}$ respectively such that $f' \circ h_1 = \Gamma(h_2) \circ f$.*
- *$id_{(C,f,D)} \stackrel{\text{def}}{=} (id_C, id_D,)$ and $(h_1, h_2) \circ (h'_1, h'_2) \stackrel{\text{def}}{=} (h_1 \circ h'_1, h_2 \circ h'_2)$.*

is cartesian closed with

- *terminal object: $(1_{\mathcal{C}}, \cong, 1_{\mathcal{D}})$.*
- *cartesian product: $(C, f, D) \times (C', f', D') =_{def} (C \times C', \cong \circ (f \times f'), D \times D')$ where $\cong$ is standing for $\Gamma D \times \Gamma D' \cong \Gamma(D \times D')$ and projection maps $pr_{(C,f,D)} : (C, f, D) \times (C', f', D') \to (C, f, D)$ as $pr_{(C,f,D)} \stackrel{\text{def}}{=} (pr_C, pr_D)$ and $pr_{(C',f',D')} \stackrel{\text{def}}{=} (pr_{C'}, pr_{D'})$.*
- *exponential object*

$$(C, f, D) \Rightarrow (C', f', D') =_{def} ((C \Rightarrow C') \times \Gamma(D \Rightarrow D'), \pi_2, D \Rightarrow D')$$

is defined by the following pullback

$$\begin{array}{ccc} (C \Rightarrow C') \times \Gamma(D \Rightarrow D') & \xrightarrow{\pi_2} & \Gamma(D \Rightarrow D') \\ \downarrow{\scriptstyle \pi_1} & & \downarrow{\scriptstyle (f \Rightarrow id_{\Gamma D'}) \circ \cong} \\ C \Rightarrow C' & \xrightarrow[id_D \Rightarrow f']{} & C \Rightarrow \Gamma D' \end{array}$$

with the evaluation map $ev_{(C,f,D),(C',f',D')} \stackrel{\text{def}}{=} (ev_{C,C'} \circ (pr_{C \Rightarrow C'} \times id_C), ev_{D,D'})$

The category $\mathcal{GL}(\Gamma)$ *is called a* ***glued category*** *to indicate that* $\mathcal{D}$ *has been glued to* $\mathcal{C}$ *along the functor* Γ*. Moreover, the projection functor* $P_2 : \mathcal{GL}(\Gamma) \to \mathcal{D}$ *(*$(C, f, D) \mapsto D$ *and* $(g, h) \mapsto h$*), is cartesian closed, i.e. preserves finite products and exponentials.*

Next, we demonstrate that under the assumption that Γ strictly preserves internal permutation actions, the glued category $\mathcal{GL}(\Gamma)$ also inherits the structure of an FM-category.

Theorem 5.1.17 *Let* $\Gamma : D \to C$ *be a functor between the FM-categories* $\mathcal{C}$ *and* $\mathcal{D}$ *such that* Γ *has the following properties:*

- *strictly preserves finite products*
- *strictly preserves internal permutation actions, i.e.* $\Gamma(\pi_D) = \pi_{\Gamma D}$

Let $\mathcal{GL}(\Gamma)$ *be the comma category* $(id_{\mathcal{C}} \downarrow \Gamma)$*. Then* $\mathcal{GL}(\Gamma)$ *is an FM-category, and* $P_2 : \mathcal{GL}(\Gamma) \to \mathcal{D}$ *is a morphism of FM-categories.*

Proof see Appendix B.2.

By combining both results we can immediately deduce that the gluing lemma holds for FM-cccs as well.

Corollary 5.1.18 *Let* $\Gamma : D \to C$ *be a functor between the FM-cccs* $\mathcal{C}$ *and* $\mathcal{D}$ *such that* Γ *strictly preserves finite products and internal permutation actions, and* $\mathcal{GL}(\Gamma)$ *is the comma category* $(id_{\mathcal{C}} \downarrow \Gamma)$*. Then* $\mathcal{GL}(\Gamma)$ *is an FM-ccc, and* $P_2 : \mathcal{GL}(\Gamma) \to \mathcal{D}$ *is an FM-ccc functor.*

Proof We apply Theorem 5.1.16 and Theorem 5.1.17 to obtain that $\mathcal{GL}(\Gamma)$ is an FM-ccc and $P_2 : \mathcal{GL}(\Gamma) \to \mathcal{D}$ an FM-ccc functor. What remains to be shown is that

the exponential is equivariant:

$$\begin{aligned}
&\pi \cdot ev_{(C,f,D),(C',f',D')} \\
&\stackrel{\text{def}}{=} \pi \cdot (ev_{C,C'} \circ (pr_{C \Rightarrow C'} \times id_C), ev_{D,D'}) \\
&= (\pi \cdot (ev_{C,C'} \circ (pr_{C \Rightarrow C'} \times id_C)), \pi \cdot ev_{D,D'}) && \text{(Lemma B.2.2)} \\
&= (ev_{\pi \cdot C, \pi \cdot C'} \circ (pr_{\pi \cdot C \Rightarrow \pi \cdot C'} \times id_{\pi \cdot C}), ev_{\pi \cdot D, \pi \cdot D'}) && (\mathcal{C}, \mathcal{D} \text{ are FM-cccs}) \\
&\stackrel{\text{def}}{=} ev_{(\pi \cdot C, \pi \cdot f, \pi \cdot D),(\pi \cdot C', \pi \cdot f', \pi \cdot D')} \\
&\stackrel{\text{def}}{=} ev_{\pi \cdot (C,f,D), \pi \cdot (C',f',D')}
\end{aligned}$$

□

Product and Functor Categories

We demonstrate that the binary product category of FM-categories is an FM-category, which follows straightforwardly.

Lemma 5.1.19 *Let $\mathcal{C}$ and $\mathcal{D}$ be FM-categories. Then the product category $\mathcal{C} \times \mathcal{D}$ with the usual definition*

- $ob\,(\mathcal{C} \times \mathcal{D}) \stackrel{\text{def}}{=} ob\,(\mathcal{C}) \times ob\,(\mathcal{D})$
- $mor\,(\mathcal{C} \times \mathcal{D}) \stackrel{\text{def}}{=} mor\,(\mathcal{C}) \times mor\,(\mathcal{D})$
- $(f_2, g_2) \circ (f_1, g_1) \stackrel{\text{def}}{=} (f_2 \circ f_1, g_2 \circ g_1)$
- $id_{(C,D)} \stackrel{\text{def}}{=} (id_C, id_D)$

is an FM-category with the following point-wise structures:

- $1_{\mathcal{C} \times \mathcal{D}} \stackrel{\text{def}}{=} (1_{\mathcal{C}}, 1_{\mathcal{D}})$
- $(C, D) \times (C', D') \stackrel{\text{def}}{=} (C \times C', D \times D')$ *with projection morphisms* $pr_{(C,D)} \stackrel{\text{def}}{=} (pr_C, pr_D) : (C, D) \times (C', D') \to (C, D)$ *and* $pr_{(C',D')} \stackrel{\text{def}}{=} (pr_{C'}, pr_{D'}) : (C, D) \times (C', D') \to (C', D')$

- $\pi_{(C,D)} \stackrel{\text{def}}{=} (\pi_C, \pi_D) : (C, D) \to (\pi{\cdot}C, \pi{\cdot}D)$. *Consequently, we have that* $\pi{\cdot}(f, g) = (\pi \cdot f, \pi \cdot g)$
- $i^{\overline{a}}_{(C,D)} \stackrel{\text{def}}{=} (i^{\overline{a}}_C, i^{\overline{a}}_D) : (C^{\#\overline{a}}, D^{\#\overline{a}}) \to (C, D)$

Moreover, we can show that the category of small FM-categories and FM-functors has finite products.

Proof see Appendix B.3

This result is not too surprising. A more interesting question is if for FM-categories $\mathcal{C}$ and $\mathcal{D}$, the functor category $[\mathcal{C}, \mathcal{D}]$ is also an FM-category.

We recall that for functor categories, properties can often be lifted point-wise. This is for example the case for (co)limits, where (co)limits in $\mathcal{D}$ can be lifted point-wise to (co)limits in $[\mathcal{C}, \mathcal{D}]$. This also applies for equivariant finite products (see Lemma 5.1.23), but not for fresh inclusions as will be shown later. Further, with respect to internal permutation actions of $[\mathcal{C}, \mathcal{D}]$ we require a definition that is analogue to the permutation action used for exponentials in *FMSet*. Thus, we define an internal permutation action that mimics the conjugational permutation action of *FMSet*.

Lemma 5.1.20 *Let $\mathcal{C}$ and $\mathcal{D}$ be categories with internal permutation actions. An internal permutation action for the functor category $[\mathcal{C}, \mathcal{D}]$ can be defined as follows: For a functor $F : \mathcal{C} \to \mathcal{D}$, $\pi_F : F \Rightarrow \pi \cdot F$ is defined to be a natural transformation with the following components (in $\mathcal{D}$) for every $X \in ob\,(\mathcal{C})$*

$$(\pi_F)_X : FX \to (\pi \cdot F)(X)$$
$$(\pi_F)_X \stackrel{\text{def}}{=} \pi_{F(\pi^{-1} \cdot X)} \circ F(\pi_X^{-1})$$

where the functor $\pi \cdot F : \mathcal{C} \to \mathcal{D}$ is defined on objects by $(\pi \cdot F)X \stackrel{\text{def}}{=} \pi \cdot F(\pi^{-1} \cdot X)$ and on morphisms by $(\pi \cdot F)f \stackrel{\text{def}}{=} \pi \cdot F(\pi^{-1} \cdot f)$.

Proof $\pi \cdot F$ is clearly a functor. We now demonstrate that $\pi_F : F \Rightarrow \pi \cdot F$ is a natural transformation. Let $X \in ob\,\mathcal{C}$. Given that $\mathcal{C}$ has internal permutation actions, we have that π_X^{-1} is a morphism in $\mathcal{C}$ and by F being a functor, we have that $F(\pi_X^{-1})$ is a morphism in $\mathcal{D}$. Further, given that $\mathcal{D}$ has internal permutation actions, we have that $\pi_{F(\pi^{-1} \cdot X)}$ is a morphism in $\mathcal{D}$. Then, by composition, $(\pi_F)_X$ is a morphism in $\mathcal{D}$. Next, we demonstrate that π_F satisfies the naturality condition: Let $f : X \to Y$ in $\mathcal{C}$

$$
\begin{aligned}
& (\pi \cdot F)(f) \circ (\pi_F)_X \\
\stackrel{\text{def}}{=}\ & (\pi \cdot F(\pi^{-1} \cdot f)) \circ (\pi_F)_X \\
\stackrel{\text{def}}{=}\ & (\pi \cdot F(\pi^{-1} \cdot f)) \circ \pi_{F(\pi^{-1} \cdot X)} \circ F(\pi_X^{-1}) \\
=\ & \pi_{F(\pi^{-1} \cdot Y)} \circ F(\pi^{-1} \cdot f) \circ F(\pi_X^{-1}) && \text{(Lemma 5.1.3 (iv) for } \pi_{\_} \text{ in } \mathcal{D}) \\
=\ & \pi_{F(\pi^{-1} \cdot Y)} \circ F(\pi^{-1} \cdot f \circ \pi_X^{-1}) && (F \text{ is a functor}) \\
=\ & \pi_{F(\pi^{-1} \cdot Y)} \circ F(\pi_Y^{-1} \circ f) && \text{(Lemma 5.1.3 (iv) for } \pi_{\_}^{-1} \text{ in } \mathcal{C}) \\
=\ & \pi_{F(\pi^{-1} \cdot Y)} \circ F(\pi_Y^{-1}) \circ F(f) && (F \text{ is a functor}) \\
\stackrel{\text{def}}{=}\ & (\pi_F)_Y \circ F(f)
\end{aligned}
$$

Hence, we have that π_F is a morphism in $[\mathcal{C}, \mathcal{D}]$. What remains to be shown is that the conditions of an internal permutation action are satisfied. The first condition, $\iota_F = id_F$, follows directly from the fact that $\mathcal{C}$ and $\mathcal{D}$ have internal permutation actions and F being a functor. This is also the case for the second condition, $\pi' \circ \pi_F = \pi'_{\pi \cdot F} \circ \pi_F$, where we additionally apply that $\pi_{(\_)}$ is a natural transformation in $\mathcal{D}$. □

The permutation action on morphisms of $[\mathcal{C}, \mathcal{D}]$ can be expressed in a more "natural" way as follows:

Lemma 5.1.21 *Let $F, G \in ob\,([\mathcal{C}, \mathcal{D}])$ and $\alpha \in mor\,([\mathcal{C}, D])$ from F to G. Then the permutation action $\pi \cdot \alpha$ can be represented as follows:*

$$(\pi \cdot \alpha)_X = \pi \cdot \alpha_{\pi^{-1} \cdot X}$$

Proof

$$
\begin{aligned}
&(\pi \cdot \alpha)_X \\
&\stackrel{\text{def}}{=} (\pi_G \circ \alpha \circ \pi^{-1}_{\pi \cdot F})_X \\
&\stackrel{\text{def}}{=} (\pi_G)_X \circ \alpha_X \circ (\pi^{-1}_{\pi \cdot F})_X \\
&\stackrel{\text{def}}{=} \pi_{G(\pi^{-1} \cdot X)} \circ G(\pi_{\pi^{-1} \cdot X}) \circ \alpha_X \circ \pi^{-1}_{(\pi \cdot F)(\pi \cdot X)} \circ (\pi \cdot F)(\pi^{-1}_{\pi \cdot X}) \\
&\stackrel{\text{def}}{=} \pi_{G(\pi^{-1} \cdot X)} \circ G(\pi_{\pi^{-1} \cdot X}) \circ \alpha_X \circ \pi^{-1}_{\pi \cdot (FX)} \circ \pi \cdot F(\pi^{-1} \cdot \pi^{-1}_{\pi \cdot X}) \\
&= \pi_{G(\pi^{-1} \cdot X)} \circ G(\pi_{\pi^{-1} \cdot X}) \circ \alpha_X \circ \pi^{-1}_{\pi \cdot (FX)} \circ \pi \cdot F(\pi^{-1}_X) && \text{(Lemma 5.1.3 (i))} \\
&\stackrel{\text{def}}{=} \pi_{G(\pi^{-1} \cdot X)} \circ G(\pi_{\pi^{-1} \cdot X}) \circ \alpha_X \circ \pi^{-1}_{\pi \cdot (FX)} \circ \pi_{FX} \circ F(\pi^{-1}_X) \circ \pi^{-1}_{(\pi \cdot F)X} \\
&\stackrel{\text{def}}{=} \pi_{G(\pi^{-1} \cdot X)} \circ G(\pi_{\pi^{-1} \cdot X}) \circ \alpha_X \circ F(\pi^{-1}_X) \circ \pi^{-1}_{(\pi \cdot F)X} \\
&= \pi_{G(\pi^{-1} \cdot X)} \circ G(\pi_{\pi^{-1} \cdot X}) \circ G(\pi^{-1}_X) \circ \alpha_{\pi^{-1} \cdot X} \circ \pi^{-1}_{(\pi \cdot F)X} && \text{(naturality of } \alpha\text{)} \\
&= \pi_{G(\pi^{-1} \cdot X)} \circ G(\pi_{\pi^{-1} \cdot X} \circ \pi^{-1}_X) \circ \alpha_{\pi^{-1} \cdot X} \circ \pi^{-1}_{(\pi \cdot F)X} && (G \text{ is a functor}) \\
&\stackrel{\text{def}}{=} \pi_{G(\pi^{-1} \cdot X)} \circ G(id_X) \circ \alpha_{\pi^{-1} \cdot X} \circ \pi^{-1}_{(\pi \cdot F)X} \\
&= \pi_{G(\pi^{-1} \cdot X)} \circ \pi^{-1}_{G(\pi^{-1} \cdot X)} \circ \pi \cdot \alpha_{\pi^{-1} \cdot X} && \text{(Lemma 5.1.3 (iv))} \\
&= \pi \cdot \alpha_{\pi^{-1} \cdot X} && \text{(Lemma 5.1.3 (ii))}
\end{aligned}
$$

□

Note that for perm-categories $\mathcal{C}$ and $\mathcal{D}$ the internal permutation action for $[\mathcal{C}, \mathcal{D}]$ may not be finitely supported and therefore $[\mathcal{C}, \mathcal{D}]$ would not be a perm-category. This is analogue to *Nom*, where the exponential is defined as the set of finitely supported morphisms. Hence, we introduce the functor category of finitely supported natural transformations, $[\mathcal{C}, \mathcal{D}]_{fs} \subseteq [\mathcal{C}, \mathcal{D}]$ (luff subcategory), which is by definition a perm-category. We can now prove the following result:

Lemma 5.1.22 *The category PermCat of small perm-categories and perm-functors is cartesian closed.*

Proof see Appendix B.3

Lemma 5.1.23 *Let $\mathcal{C}$ be a category and $\mathcal{D}$ a perm-category with equivariant finite products. The point-wise definition of finite products of $[\mathcal{C}, \mathcal{D}]$ is equivariant.*

Proof see Appendix B.3

We conclude by showing that a point-wise definition for fresh inclusions in $[\mathcal{C}, \mathcal{D}]$ does not suffice: Suppose $\overline{a} \subseteq \mathbb{A}$, $F : \mathcal{C} \to \mathcal{D}$ and $\overline{a} \# F$. A point-wise fresh inclusion would be defined as a natural transformation $i_F^{\overline{a}} : F^{\#\overline{a}} \to F$ with components

$$(i_F^{\overline{a}})_X \stackrel{\text{def}}{=} i_{FX}^{\overline{a}} : FX^{\#\overline{a}} \to FX$$

and $F^{\#\overline{a}}(X) \stackrel{\text{def}}{=} (FX)^{\#\overline{a}}$. Note that this definition is only well defined if we can show that the fresh inclusion $i_{FX}^{\overline{a}}$ actually exists. This means that we would have to show that $\overline{a} \# FX$ holds, which cannot be guaranteed, because $supp(FX) \subseteq supp(F) \cup supp(X)$ and we only have $\overline{a} \# F$.

5.2 A Sound Categorical Semantics for NLC

In this section we introduce a categorical semantics that will interpret typed expressions $Th \rhd \nabla \vdash^{\text{NLC}} M : s$ as morphisms $[\![\nabla \vdash^{\text{NLC}} M : s]\!] : [\![\nabla]\!] \longrightarrow [\![s]\!]$ in an FM-ccc $\mathcal{C}$. Similar to the model-theoretic semantics in *FMSet*, we need to consider that NLC is dependently typed: in particular the type system and equation system are mutually inductively defined. So, following [52, 66] we introduce a *partial* interpretation function, which is defined only when certain equations are themselves satisfied by $[\![-]\!]$.

We formally define the categorical semantics for NLC as follows: Let $\mathcal{C}$ be an FM-ccc and Sg a NLC-signature. Then a **Sg-structure $\mathcal{M}$** in $\mathcal{C}$ is specified by giving:

- An equivariant map $[\![-]\!] : Gnd_{Sg} \longrightarrow ob\,\mathcal{C}$ that extends to the map $[\![-]\!] : Type_{Sg} \longrightarrow ob\,\mathcal{C}$ via structural recursion ($[\![s^{\overline{a}} \Rightarrow s']\!] \stackrel{\text{def}}{=} [\![s]\!]^{\#\overline{a}} \Rightarrow [\![s']\!]$) and can easily be shown to be equivariant as well, since $\mathcal{C}$ has equivariant structure.

- An equivariant map $[\![-]\!] : Fun_{Sg} \longrightarrow ob\,\mathcal{C}$ where for each higher order function constant $c : s$ we have $[\![c]\!] : 1 \longrightarrow [\![s]\!]$ (recall that $\mathcal{C}$ has finite products—hence an equivariant terminal object).

Let $\nabla = \overline{a}_1 \# x_1 : s_1, ..., \overline{a}_n \# x_n : s_n \in Ctxt_{Sg}$ be a freshness context. We define the $\mathcal{C}$-object $[\![\nabla]\!]$ by $[\![\nabla]\!] \stackrel{\text{def}}{=} [\![s_1]\!]^{\#\overline{a}_1} \times ... \times [\![s_n]\!]^{\#\overline{a}_n}$. Let $\mathcal{M}$ be a structure for a NLC-signature in an FM-ccc $\mathcal{C}$ and consider the binary relation $\blacktriangleright$ in Table 5.1.

$$\frac{}{[\![\nabla, \overline{a} \# x : s \vdash \pi x : \pi \cdot s]\!] \blacktriangleright \pi_{[\![s]\!]} \circ i^{\overline{a}}_{[\![s]\!]} \circ pr : [\![\nabla, \overline{a} \# x : s]\!] \longrightarrow \pi \cdot [\![s]\!]}$$

$$\frac{}{[\![\nabla \vdash c : s]\!] \blacktriangleright [\![c]\!] \circ ! : [\![\nabla]\!] \to 1 \to [\![s]\!]}$$

$$\frac{[\![\nabla, \overline{a} \# x : s \vdash M : s']\!] \blacktriangleright m : [\![\nabla]\!] \times [\![s]\!]^{\#\overline{a}} \to [\![s']\!]}{[\![\nabla \vdash \lambda^{\overline{a}} x : s.M : s^{\overline{a}} \Rightarrow s']\!] \blacktriangleright \lambda(m) : [\![\nabla]\!] \to ([\![s]\!]^{\#\overline{a}} \Rightarrow [\![s']\!])}$$

$$\frac{[\![\nabla \vdash F : s^{\overline{a}} \Rightarrow s']\!] \blacktriangleright f : [\![\nabla]\!] \to ([\![s]\!]^{\#\overline{a}} \Rightarrow [\![s']\!]) \quad [\![\nabla \vdash \overline{a} \# A : s]\!] \blacktriangleright \theta : [\![\nabla]\!] \to [\![s]\!]^{\#\overline{a}}}{[\![\nabla \vdash F\ A : s']\!] \blacktriangleright ev \circ \langle f, \theta \rangle : [\![\nabla]\!] \to ([\![s]\!]^{\#\overline{a}} \Rightarrow [\![s']\!]) \times [\![s]\!]^{\#\overline{a}} \to [\![s']\!]}$$

$$\frac{[\![\nabla \vdash^{\text{NLC}} M : s]\!] \blacktriangleright m : [\![\nabla]\!] \to [\![s]\!]}{[\![\nabla \vdash^{\text{NLC}} \overline{a} \# M : s]\!] \blacktriangleright m^* : [\![\nabla]\!] \to [\![s]\!]^{\#\overline{a}}} \qquad \dagger(\overline{a}, m)$$

Table 5.1: Categorical Semantics for NLC in an FM-ccc

Following [17] we *make use of the freshness axiom of an FM-ccc*, which is given by Definition 5.1.9 (vi). Based on this freshness axiom we can directly define the semantics of $\nabla \vdash^{\text{NLC}} \overline{a} \# A : s$ as $[\![\nabla \vdash^{\text{NLC}} \overline{a} \# A : s]\!] \stackrel{\text{def}}{=} [\![\nabla \vdash^{\text{NLC}} A : s]\!]^*$ under the assumption that the condition $\dagger(\overline{a}, [\![\nabla \vdash^{\text{NLC}} A : s]\!])$ holds. Note that the categorical semantics slightly differs from the model-theoretic semantics (in *FMSet*), where the partial interpretation function does not provide an additional clause for freshness assertions.

It can easily be seen that Table 5.1 actually specifies a partial function $J \mapsto [\![J]\!]$ from judgements to morphisms $[\![J]\!]$ in $\mathcal{C}$. We define a notion of satisfaction for both expressions-in-context and equations-in-context. Given $\nabla \vdash^{\text{NLC}} M : s$ or $\nabla \vdash^{\text{NLC}} \overline{a} \#$

$M : s$ we say that $\mathcal{M}$ **satisfies** the judgement if the morphism $[\![\nabla \vdash^{\text{NLC}} M : s]\!] : [\![\nabla]\!] \longrightarrow [\![s]\!]$ or $[\![\nabla \vdash^{\text{NLC}} \overline{a} \# M : s]\!] : [\![\nabla]\!] \longrightarrow [\![s]\!]^{\#\overline{a}}$ in $\mathcal{C}$ is *defined* (that is, the partial function $J \mapsto [\![J]\!]$ is defined). If so we write $[\![\nabla \vdash^{\text{NLC}} M : s]\!]\Downarrow$ or $[\![\nabla \vdash^{\text{NLC}} \overline{a} \# M : s]\!]\Downarrow$. Generally, $[\![J]\!]\Downarrow \stackrel{\text{def}}{=} (\exists j)([\![J]\!] \blacktriangleright j)$. We may write $[\![J]\!]$ or even $[\![J]\!]\Downarrow$ for morphism j. Given $\nabla \vdash^{\text{NLC}} M \approx M' : s$ we say that $\mathcal{M}$ **satisfies** it if both $[\![\nabla \vdash^{\text{NLC}} M : s]\!]\Downarrow$ and $[\![\nabla \vdash^{\text{NLC}} M' : s]\!]\Downarrow$ and they are equal morphisms in $\mathcal{C}$. We say that $\mathcal{M}$ is a **model** of a NLC theory $Th = (Sg, Ax)$ if $\mathcal{M}$ satisfies all of the equations-in-context in Ax. With this, we can specify our soundness theorem:

Theorem 5.2.1 (Soundness) *Let Th be a* NLC *theory and $\mathcal{M}$ a model of Th in an FM-ccc. Then every typed expression $Th \rhd \nabla \vdash^{\text{NLC}} M : s$ and theorem $Th \rhd \nabla \vdash^{\text{NLC}} M \approx M' : s$ is satisfied by $\mathcal{M}$. As a consequence, every freshness assertion $Th \rhd \nabla \vdash^{\text{NLC}} \overline{a} \# M : s$ is satisfied by $\mathcal{M}$ as well.*

Note that despite some minor technical differences the overall proof technique used to deal with name-dependent types and the mutual inductive definition of types and equations is analogous to the model-theoretic semantics in *FMSet*. For this reason, we only present full proof details for selected results.

We require the following auxiliary results to prove the soundness theorem. Note that we appeal to Propositions 3.2.5 and Proposition 3.2.6 to ensure that the NLC judgements mentioned below are properly defined. We shall write $L \asymp R$ to mean that $L\Downarrow \iff R\Downarrow$ and that $L = R$ (Kleene equality).

Lemma 5.2.2 *Let $\nabla = \overline{a}_1 \# x_1 : s_1, ..., \overline{a}_n \# x_n : s_n$ be a freshness context. Then*

(i) $id_{[\![\nabla]\!]} \asymp \langle [\![\nabla \vdash \overline{a}_1 \# x_1 : s_1]\!], ..., [\![\nabla \vdash \overline{a}_n \# x_n : s_n]\!] \rangle$

(ii) $i^{\overline{a}}_{[\![\nabla]\!]} \asymp \langle [\![\nabla^{\#\overline{a}} \vdash \overline{a}_1 \# x_1 : s_1]\!], ..., [\![\nabla^{\#\overline{a}} \vdash \overline{a}_n \# x_n : s_n]\!] \rangle$

(iii) $\pi_{[\![\nabla]\!]} \asymp \langle [\![\nabla \vdash \pi \cdot \overline{a}_1 \# \pi x_1 : \pi \cdot s_1]\!], ..., [\![\nabla \vdash \pi \cdot \overline{a}_n \# \pi x_n : \pi \cdot s_n]\!] \rangle$

(iv) *Take* $\nabla_1, \nabla_2 \in Ctxt_{Sg}$ *with disjoint domains (suppose* $\nabla_j = \nabla$ *for* $j = 1$ *and* 2*). Then* $pr_{[\![\nabla_j]\!]} : [\![\nabla_1]\!] \times [\![\nabla_2]\!] \to [\![\nabla_j]\!] \asymp \langle [\![\nabla_1 \cup \nabla_2 \vdash^{\text{NLC}} \overline{a}_1 \# x_1 : s_1]\!], ..., [\![\nabla_1 \cup \nabla_2 \vdash^{\text{NLC}} \overline{a}_n \# x_n : s_n]\!] \rangle$.

Lemma 5.2.3 (Useful Semantic Factorisations "$[\![\xi]\!] = [\![\xi]\!] \circ m$")

(i) *The map* $\nabla \mapsto [\![\nabla]\!]$ *is equivariant.*

(ii) $\pi \cdot [\![\nabla \vdash^{\text{NLC}} M : s]\!] \asymp [\![\pi \cdot \nabla \vdash^{\text{NLC}} \pi \cdot M : \pi \cdot s]\!]$

(iii) $[\![\nabla \vdash \pi * M : \pi \cdot s]\!] \asymp \pi_{[\![s]\!]} \circ [\![\nabla \vdash M : s]\!]$

(iv)

$$[\![\nabla, \pi \cdot \overline{a} \# x : \pi \cdot s \vdash^{\text{NLC}} M\{\pi^{-1}x/x\} : s']\!] \asymp [\![\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} M : s']\!] \circ (id_{[\![\nabla]\!]} \times \pi^{-1}_{[\![\pi \cdot s]\!]^{\# \pi \cdot \overline{a}}})$$

(v) *Given* $\nabla \leq \nabla'$ *there exists an arrow* $weak : [\![\nabla']\!] \to [\![\nabla]\!]$ *such that for any well-typed expression* $\nabla \vdash^{\text{NLC}} M : s$, $[\![\nabla' \vdash^{\text{NLC}} M : s]\!] \asymp [\![\nabla \vdash^{\text{NLC}} M : s]\!] \circ weak$.

(vi) $[\![\nabla^{\# \overline{a}} \vdash^{\text{NLC}} M : s]\!] \asymp [\![\nabla \vdash^{\text{NLC}} M : s]\!] \circ i^{\overline{a}}$ *where* $\overline{a} \# \nabla$.

The proofs of Lemmas 5.2.2 and 5.2.3 require a combination of direct calculations and inductions over the structure of expressions.

Proof of Lemma 5.2.3 (i) and (ii)

(i) It follows immediately from the definition of a perm-category and the definition

of a categorical semantics that

$$\begin{aligned}
\pi \cdot [\![\nabla]\!] &\stackrel{\text{def}}{=} \pi \cdot ([\![s_1]\!]^{\#\overline{a}_1} \times ... \times [\![s_n]\!]^{\#\overline{a}_n}) \\
&\stackrel{\text{def}}{=} (\pi \cdot [\![s_1]\!]^{\#\overline{a}_1} \times ... \times \pi \cdot [\![s_n]\!]^{\#\overline{a}_n}) \\
&\stackrel{\text{def}}{=} ((\pi \cdot [\![s_1]\!])^{\#\pi\cdot\overline{a}_1} \times ... \times (\pi \cdot [\![s_n]\!])^{\#\pi\cdot\overline{a}_n}) \\
&\stackrel{\text{def}}{=} ([\![\pi \cdot s_1]\!]^{\#\pi\cdot\overline{a}_1} \times ... \times [\![\pi \cdot s_n]\!]^{\#\pi\cdot\overline{a}_n}) \\
&\stackrel{\text{def}}{=} [\![(\pi \cdot \overline{a}_1 \# x_1 : \pi \cdot s_1, ..., \pi \cdot \overline{a}_n \# x_n : \pi \cdot s_n)]\!] \\
&\stackrel{\text{def}}{=} [\![\pi \cdot \nabla]\!]
\end{aligned}$$

(ii) Proof by induction on the structure of M

$$(\forall M) \quad [\quad (\forall\, \nabla, \pi, s) \quad (\pi \cdot [\![\nabla \vdash^{\text{NLC}} M : s]\!] \asymp [\![\pi \cdot \nabla \vdash^{\text{NLC}} \pi \cdot M : \pi \cdot s]\!])) \quad]$$

SUSP: (M is $\pi' x$) It directly follows from the categorical semantics that

$$[\![\pi \cdot \nabla', \pi \cdot \overline{a} \# x : \pi \cdot s \vdash^{\text{NLC}} \pi \cdot \pi' x : \pi \cdot \pi' \cdot s]\!]\Downarrow$$

and $[\![\nabla', \overline{a} \# x : s \vdash^{\text{NLC}} \pi' x : \pi' \cdot s]\!]\Downarrow$. The equality follows by basic properties of FM-cccs. **CONST:** (M is c) It is immediate that $[\![\nabla \vdash^{\text{NLC}} c : s]\!]\Downarrow$ and $[\![\pi\cdot\nabla \vdash^{\text{NLC}} \pi\cdot c : \pi\cdot s]\!]\Downarrow$. The equality follows from the fact that $[\![-]\!] : Fun_\Sigma \to ob\,\mathcal{C}$ is equivariant.

LAM-ABS: (M is $\lambda^{\overline{a}} x : s.\, N$) Suppose $[\![\pi\cdot\nabla \vdash^{\text{NLC}} \pi\cdot(\lambda^{\overline{a}} x : s.\, N) : \pi\cdot(s^{\overline{a}} \Rightarrow s')]\!]\Downarrow$ and it is equal to f_π. By the definition of the meta-level permutation action and the inductively defined semantics we get

$$[\![\pi \cdot \nabla \vdash^{\text{NLC}} \lambda^{\pi\cdot\overline{a}} x : \pi \cdot s.\, \pi \cdot N : (\pi \cdot s)^{\pi\cdot\overline{a}} \Rightarrow \pi \cdot s']\!] \blacktriangleright \lambda(n_\pi)$$

for some n_π where $[\![\pi\cdot\nabla, \pi\cdot\overline{a} \# x : \pi\cdot s \vdash^{\text{NLC}} \pi\cdot N : \pi\cdot s']\!] \blacktriangleright n_\pi$. By induction we deduce that $[\![\nabla, \overline{a} \# x : s \vdash^{\text{NLC}} N : s']\!] \blacktriangleright n$ such that $\pi \cdot n = n_\pi$. We then apply the rule for semantics of abstraction to obtain $[\![\nabla \vdash^{\text{NLC}} \lambda^{\overline{a}} x : s.\, N : s^{\overline{a}} \Rightarrow s']\!] \blacktriangleright \lambda(n)$, that is, $[\![\nabla \vdash^{\text{NLC}} \lambda^{\overline{a}} x : s.\, N : s^{\overline{a}} \Rightarrow s']\!]\Downarrow$. The definitional existence proof in

the converse direction follows by similar reasoning. We now need to show that $f_\pi = \pi \cdot \lambda(n)$.

$$\begin{aligned} f_\pi &\stackrel{\text{def}}{=} \lambda(n_\pi) \\ &= \lambda(\pi \cdot n) && \text{(induction)} \\ &= \pi \cdot \lambda(n) && \text{(Lemma 5.1.15 (d))} \end{aligned}$$

APP: (M is $F\ A$) Suppose $[\![\pi \cdot \nabla \vdash^{\text{NLC}} \pi \cdot (F\ A) : \pi \cdot s']\!]\Downarrow$ and it is equal to t_π. By the definition of the meta-level permutation action and the rule for semantics of applications

$$[\![\pi \cdot \nabla \vdash^{\text{NLC}} (\pi \cdot F)\ (\pi \cdot A) : \pi \cdot s']\!] \blacktriangleright ev \circ \langle f_\pi, \theta_\pi \rangle$$

for

$$[\![\pi \cdot \nabla \vdash^{\text{NLC}} \pi \cdot F : (\pi \cdot s)^{\pi \cdot \overline{a}} \Rightarrow \pi \cdot s']\!] \blacktriangleright f_\pi$$

and

$$[\![\pi \cdot \nabla \vdash^{\text{NLC}} \pi \cdot \overline{a} \# \pi \cdot A : \pi \cdot s]\!] \blacktriangleright \theta_\pi.$$

We have $[\![\pi \cdot \nabla \vdash^{\text{NLC}} \pi \cdot \overline{a} \# \pi \cdot A : \pi \cdot s]\!] \blacktriangleright {\alpha_\pi}^*$ by the rule for freshness assertion semantics where $[\![\pi \cdot \nabla \vdash^{\text{NLC}} \pi \cdot A : \pi \cdot s]\!] \blacktriangleright \alpha_\pi$ such that $\dagger(\pi \cdot \overline{a}, \alpha_\pi)$. Given that $\blacktriangleright$ is a partial function, we have that $\theta_\pi = {\alpha_\pi}^*$. By induction we get $[\![\nabla \vdash^{\text{NLC}} F : s^{\overline{a}} \Rightarrow s']\!] \blacktriangleright f$ and $[\![\nabla \vdash^{\text{NLC}} A : s]\!] \blacktriangleright \alpha$ such that $f_\pi = \pi \cdot f$ and $\alpha_\pi = \pi \cdot \alpha$. We now deduce from $\dagger(\pi \cdot \overline{a}, \alpha_\pi)$ that $\dagger(\overline{a}, \alpha)$ holds: Let $\overline{c} \# (\overline{a}, \alpha)$. It follows immediately that $\pi \cdot \overline{c} \# (\pi \cdot \overline{a}, \pi \cdot \alpha)$ and hence from $\dagger(\pi \cdot \overline{a}, \pi \cdot \alpha)$ we obtain equation ($\diamond 1$). In the equations below, we write internal permutation actions τ_C as τ_- since the source-target data does not play a significant role in

our reasoning, and indeed is probably obfuscating:

$$(\pi \cdot \overline{c}\,\pi \cdot \overline{a})_{\_} \circ (\pi \cdot \alpha) \circ i = (\pi \cdot \alpha) \circ i \qquad (\diamond 1)$$

$$(\pi \cdot \overline{c}\,\pi \cdot \overline{a})_{\_} \circ \pi_{\_} \circ \alpha \circ \pi_{\_}^{-1} \circ i = \pi_{\_} \circ \alpha \circ \pi_{\_}^{-1} \circ i \qquad (\diamond 2)$$

$$\pi_{\_} \circ (\overline{c}\,\overline{a})_{\_} \circ \alpha \circ \pi_{\_}^{-1} \circ i = \pi_{\_} \circ \alpha \circ \pi_{\_}^{-1} \circ i \qquad (\diamond 3)$$

$$(\overline{c}\,\overline{a})_{\_} \circ \alpha \circ i \circ \pi_{\_}^{-1} = \alpha \circ i \circ \pi_{\_}^{-1} \qquad (\diamond 4)$$

$$(\overline{c}\,\overline{a})_{\_} \circ \alpha \circ i = \alpha \circ i \qquad (\diamond 5)$$

By definition of the FM-ccc permutation action on morphisms we obtain equation ($\diamond 2$). The transposition notation $(\overline{c}\,\overline{a})$ is short for $(c_1\,a_1) \circ \ldots \circ (c_k\,a_k)$. Since in $Perm(\mathbb{A})$, $\pi \circ (c\,d) = (\pi(c)\,\pi(d)) \circ \pi$ holds generally for single transpositions $(c\,d)$, and since permutation actions satisfy $(\tau' \circ \tau)_C = \tau'_{\tau \cdot C} \circ \pi_C$ we have

$$\pi_{\_} \circ (\overline{c}\,\overline{a})_{\_} = (\pi \circ (\overline{c}\,\overline{a}))_{\_} = ((\pi \cdot \overline{c}\,\pi \cdot \overline{a}) \circ \pi)_{\_} = (\pi \cdot \overline{c}\,\pi \cdot \overline{a})_{\_} \circ \pi_{\_}$$

This gives us equation ($\diamond 3$). Any internal permutation action $(\tau_C : C \to \tau \cdot C \mid C \in ob\,\mathcal{C})$ is a natural transformation $Id \to \tau \cdot -$ and in particular so is $\pi_{\_}^{-1}$. Since also $\pi_{\_}$ is iso, equation ($\diamond 4$) holds. Finally since $\pi_{\_}^{-1}$ is iso we obtain ($\diamond 5$). Hence, $\dagger(\overline{a}, \alpha)$ holds.

We can now apply the rule for freshness assertion semantics to obtain $[\![\nabla \vdash^{\text{NLC}} \overline{a} \,\#\, A : s]\!] \blacktriangleright \alpha^*$, followed by the rule for application semantics to get $[\![\nabla \vdash^{\text{NLC}} F\ A : s']\!] \blacktriangleright ev \circ \langle f, \alpha^* \rangle$. Hence, we have $[\![\nabla \vdash^{\text{NLC}} F\ A : s']\!]{\Downarrow}$. The definitional existence proof in the converse direction follows by similar reasoning.

We now show that $t_\pi = \pi \cdot (ev \circ \langle f, \alpha^* \rangle)$. Note that $(\pi \cdot \alpha)^* = \pi \cdot \alpha^*$ ($\diamond$) holds: This follows immediately from the universal property of inclusion image

restriction and the definition of $\pi \cdot (-)$. Hence

$$\begin{aligned}
t_\pi &\stackrel{\text{def}}{=} ev_{(\llbracket \pi \cdot s \rrbracket^{\#\pi\cdot\overline{a}}, \llbracket \pi \cdot s' \rrbracket)} \circ \langle f_\pi, \theta_\pi \rangle \\
&= ev_{(\llbracket \pi \cdot s \rrbracket^{\#\pi\cdot\overline{a}}, \llbracket \pi \cdot s' \rrbracket)} \circ \langle f_\pi, {\alpha_\pi}^* \rangle && (\theta_\pi = {\alpha_\pi}^*) \\
&= ev_{(\pi\cdot(\llbracket s \rrbracket^{\#\overline{a}}), \pi\cdot\llbracket s' \rrbracket)} \circ \langle \pi \cdot f, (\pi \cdot \alpha)^* \rangle && (\text{induction}) \\
&= ev_{(\pi\cdot(\llbracket s \rrbracket^{\#\overline{a}}), \pi\cdot\llbracket s' \rrbracket)} \circ \langle \pi \cdot f, \pi \cdot \alpha^* \rangle && (\diamond) \\
&= ev_{(\pi\cdot(\llbracket s \rrbracket^{\#\overline{a}}), \pi\cdot\llbracket s' \rrbracket)} \circ (\pi \cdot \langle f, \alpha^* \rangle) && (\text{equivariant products}) \\
&= (\pi \cdot ev_{(\llbracket s \rrbracket^{\#\overline{a}}, \llbracket s' \rrbracket)}) \circ (\pi \cdot \langle f, \alpha^* \rangle) && (\text{equivariant exponentials}) \\
&= \pi \cdot (ev_{(\llbracket s \rrbracket^{\#\overline{a}}, \llbracket s' \rrbracket)} \circ \langle f, \alpha^* \rangle) && (\text{equivariance of } \circ)
\end{aligned}$$

□

Proposition 5.2.4 (Compositional Semantics) *Let* $\nabla \stackrel{\text{def}}{=} \overline{a}_1 \# x_1 : s_1, ..., \overline{a}_n \# x_n : s_n$. *Suppose, for theory Th, we have the typed expression* $\nabla \vdash^{\text{NLC}} M : s$ *and freshness assertions* $\nabla' \vdash^{\text{NLC}} \overline{a}_i \# N_i : s_i$ *for each* $i \leq n$. *Then we have* $\nabla' \vdash^{\text{NLC}} M\{\vec{N_i}/\vec{x_i}\} : s$. *Moreover, if* $\llbracket \nabla \vdash^{\text{NLC}} M : s \rrbracket\Downarrow$ *and* $\llbracket \nabla' \vdash^{\text{NLC}} \overline{a}_i \# N_i : s_i \rrbracket\Downarrow$ *for each* i *then we have* $\llbracket \nabla' \vdash^{\text{NLC}} M\{\vec{N_i}/\vec{x_i}\} : s \rrbracket\Downarrow$ *and further*

$$\begin{aligned}
\llbracket \nabla' \vdash^{\text{NLC}} M\{\vec{N_i}/\vec{x_i}\} : s \rrbracket = &\llbracket \nabla \vdash^{\text{NLC}} M : s \rrbracket \circ \\
&\langle \llbracket \nabla' \vdash^{\text{NLC}} \overline{a}_1 \# N_1 : s_1 \rrbracket, ..., \llbracket \nabla' \vdash^{\text{NLC}} \overline{a}_n \# N_n : s_n \rrbracket \rangle
\end{aligned}$$

The proof of Proposition 5.2.4 is by induction over the structure of M and does *not* require a complicated statement that is provable by mutual induction. The intuition is that, as one would expect, the semantics of expressions is derivation independent. We can now prove Theorem 5.2.1.

Proof (Soundness) This proof does proceed by a mutual induction establishing the satisfaction of all judgement forms. Induction Property Closure for all the rules in Table 3.4 and Table 3.5 is similar to our example:

(AP): We need to show that $[\![\nabla \vdash^{\text{NLC}} F\ A : s']\!]\Downarrow \quad (\diamond)$. By induction we have $[\![\nabla \vdash^{\text{NLC}} F : s^{\overline{a}} \Rightarrow s']\!] \blacktriangleright f \quad (\diamond 1)$. Recalling that satisfaction of the freshness assertion is the satisfaction of an equation

$$\nabla \vdash^{\text{NLC}} \overline{a} \# A : s \quad \stackrel{\text{def}}{=} \quad (\mathcal{N}\,\overline{c}) \quad (\nabla^{\#\overline{c}} \vdash^{\text{NLC}} A \approx (\overline{a}\,\overline{c}) * A : s)$$

we have $[\![\nabla^{\#\overline{c}} \vdash^{\text{NLC}} A : s]\!] \blacktriangleright \theta$ and $[\![\nabla^{\#\overline{c}} \vdash^{\text{NLC}} (\overline{a}\,\overline{c}) * A : s]\!] \blacktriangleright \theta'$ with $\theta = \theta'$. Hence, by Lemma 5.2.2 (vi), we have that $\theta = \alpha \circ i$ where $[\![\nabla \vdash^{\text{NLC}} A : s]\!] \blacktriangleright \alpha$ and by Lemma 5.2.2 (iii) and 5.2.2 (vi) we have that $\theta' = (\overline{a}\,\overline{c})_{[\![s]\!]} \circ \alpha \circ i$. From the mono property of freshness inclusions we have $\alpha = (\overline{a}\,\overline{c})_{[\![s]\!]} \circ \alpha$, that is $\dagger(\alpha, \overline{a})$. Hence, by definition, we get $[\![\nabla \vdash^{\text{NLC}} \overline{a} \# A : s]\!] \blacktriangleright \alpha^* \quad (\diamond 2)$. From $(\diamond 1)$ and $(\diamond 2)$ we obtain $(\diamond)$, with the definition $ev \circ \langle f, \alpha^* \rangle$.

Property Closure for the rules in Table 3.5 is trivial for (REF) (SYM) (TRANS). (WEAK) uses Lemma 5.2.2 (v). (AE) uses Lemma 5.2.3 (vi), Lemma 5.2.3 (ii) and Definition 5.1.9 (v). (SUSP) follows by standard nominal computations. The existence argument for (B) and (E) follows similarly to (AP). The equational part for (B) follows from Proposition 5.2.4 □

5.3 Construction of Exponentials in a Classifying Category

We remind the reader that the key step in proving completeness and ultimately a categorical type theory correspondence for NLC-theories, with respect to FM-cccs, is that for any NLC-theory, a syntactically generated classifying category, referred to as $Cl(Th)$, can be constructed, which is an FM-ccc build from the syntax of an NLC-theory Th together with a generic model $\mathcal{G}$. The notion of a classifying category is a standard one in category theory [42, 20]. For clarity, we decompose the required proof argument in four separate steps and recall it in the context of EL and $\lambda^{\rightarrow}$.

(i) For any theory Th, a term quotient category, referred to as $Cl(Th)$[1], is constructed, which is analogue to a term quotient structure in model theory (used to prove least-model completeness).

(ii) It has to be demonstrated that various categorical structures can be constructed in $Cl(Th)$. For example, in the case of algebraic theories it has to be shown that $Cl(Th)$ has finite products and in the case of functional theories it additionally has to be shown that $Cl(Th)$ is cartesian closed.

(iii) To prove (least-model) completeness for a categorical semantics, a generic model $\mathcal{G}$ for $Cl(Th)$ and an auxiliary lemma is required.

(iv) Finally, it has to be shown that $Cl(Th)$ satisfies the corresponding categorical property of a classifying category.

Note that our focus in this chapter will be on the first two steps, while the remaining two steps will be future work. So, for this reason, we refrain from introducing the precise definition of a classifying category and just refer to [42, 20].

We recall that the classifying category for a NEL-theory, as presented in ([17]), is an extension of the classifying category for an algebraic theory, with only minor modifications. Further, given that the construction of classifying categories for algebraic and functional theories are essentially the same, we have chose the classifying categories for NEL-theories as our starting point for NLC. As previously indicated, we now have to demonstrate that $Cl(Th)$ is an FM-ccc for any NLC-theory Th. Due to the fact that $Cl(Th)$, for any NEL-theory, is already an FM-category, it only remains to be shown that $Cl(Th)$ has equivariant exponentials. However, this turns out to be more problematic than initially expected.

As a first step, one may mimic the construction of exponentials in a classifying category for $\lambda^{\rightarrow}$ and adapt it to NLC. But this does not suffice, as we will now

[1]Note that we abuse notation at this point, because $Cl(Th)$ is not yet proven to be a classifying category. This is delayed until step (iv).

discuss in more detail. We make use of the following typical objects in the intended classifying category of NLC:

$$\nabla \stackrel{\text{def}}{=} (\overline{a}_1 \# x_1 : s_1, ..., \overline{a}_n \# x_n : s_n)$$
$$\nabla' \stackrel{\text{def}}{=} (\overline{a}'_1 \# x'_1 : s'_1, ..., \overline{a}'_m \# x'_m : s'_m)$$
$$\nabla'' \stackrel{\text{def}}{=} (\overline{a}''_1 \# x''_1 : s''_1, ..., \overline{a}''_k \# x''_k : s''_k)$$

For objects (freshness contexts) ∇ and ∇', an exponential, motivated by the classifying category of $\lambda^{\rightarrow}$, would be defined as follows

$$\nabla \Rightarrow \nabla' \stackrel{\text{def}}{=} (\overline{a}'_1 \# f_1 : \vec{s_i}^{\vec{a_i}} \Rightarrow s'_1, ..., \overline{a}'_m \# f_m : \vec{s_i}^{\vec{a_i}} \Rightarrow s'_m)$$

using the freshness restricted function types of NLC, where for each $j = 1, ..., m$, $\vec{s_i}^{\vec{a_i}} \Rightarrow s'_j \stackrel{\text{def}}{=} {s_1}^{\overline{a}_1} \Rightarrow ({s_2}^{\overline{a}_2} \Rightarrow ...({s_n}^{\overline{a}_n} \Rightarrow s'_j))$. The corresponding evaluation map $ev : (\nabla \Rightarrow \nabla') \times \nabla \rightarrow \nabla'$ would then be defined by $([f_1\ \vec{x_i}]_{\approx}, \ldots, [f_m\ \vec{x_i}]_{\approx})$ where $f_j\ \vec{x_i}$ is shorthand for the NLC-application $(...((f_j\ x_1)\ x_2)...)\ x_n$ and $\times$ is defined by list concatination.

While the definition of an exponential $\nabla \Rightarrow \nabla'$ is clearly an object in $Cl(Th)$ (a freshness context), it can immediately be observed that the evaluation map ev is not a morphism in $Cl(Th)$. We recall that to demonstrate that ev is a morphism in $Cl(Th)$, we have to show that $\nabla \Rightarrow \nabla' \times \nabla \vdash^{\text{NLC}} \overline{a}'_j \# f_j\ \vec{x_i} : s'_j$ holds for any $(1 \leq j \leq m)$. However, this is not generally the case, because $\overline{a}'_j$ is not necessarily included in the freshness context of $\vec{x_i}$. Hence, the freshness assertion cannot be derived.

Once we take a semantic viewpoint, the core problem regarding the construction of exponentials in $Cl(Th)$ for NLC can directly be identified. What proves to be problematic is the existence of exponentials (in $FMSet$) of the form $X^{\#\overline{a}} \Rightarrow_{fs} Y^{\#\overline{b}}$. To express exponentials of this form, one would expect types of the form ${s_1}^{\overline{a}_1} \Rightarrow {s_2}^{\overline{a}_2}$, but such types are not part of the NLC typing system.

An immediate question that arises in this context is if such types can be mimicked using NLC. We do not have a definitive answer, but believe that it is rather unlikely. To provide a solution to this problem, we now discuss two routes to extend NLC

such that exponentials can be constructed in a syntactically generated (classifying) category $Cl(Th)$:

- Earlier in this thesis we pointed out the possibility of introducing a variant of NLC that uses freshness restriction types of the form $s^{\overline{a}}$. Apart from being an interesting variant in its own right, it would now serve a more immediate purpose, because it would allow us to directly capture exponentials of the form $X^{\#\overline{a}} \Rightarrow_{fs} Y^{\#\overline{b}}$. More precisely, using such a variant of NLC, exponentials could then be constructed, with only minor deviations, by following the definition of a classifying category for functional theories: For objects ∇ and ∇', using the more general type ${s_1}^{\overline{a}_1} \Rightarrow {s_2}^{\overline{a}_2}$, we would have

$$\nabla \Rightarrow \nabla' \stackrel{\text{def}}{=} (\emptyset \# f_1 : \vec{s_i}^{\vec{a_i}} \Rightarrow {s'_1}^{\overline{a}'_1}, ..., \emptyset \# f_m : \vec{s_i}^{\vec{a_i}} \Rightarrow {s'_m}^{\overline{a}'_m})$$

where for each $j = 1, ..., m$, $\vec{s_i}^{\vec{a_i}} \Rightarrow s'_j \stackrel{\text{def}}{=} {s_1}^{\overline{a}_1} \Rightarrow ({s_2}^{\overline{a}_2} \Rightarrow ...({s_n}^{\overline{a}_n} \Rightarrow s'_j))$. The evaluation map $ev : (\nabla \Rightarrow \nabla') \times \nabla \to \nabla'$ could then be defined as

$$([f_1\ \vec{x_i}]_{\approx}, \ldots, [f_m\ \vec{x_i}]_{\approx})$$

where $f_j\ \vec{x_i}$ is shorthand for $(...((f_j\ x_1)\ x_2)...)\ x_n$. Further, given the morphisms $\delta \stackrel{\text{def}}{=} ([M_1]_{\approx}, \ldots, [M_m]_{\approx}) : \nabla'' \times \nabla \to \nabla'$ the exponential mate $\lambda(\delta) : \nabla'' \to (\nabla \Rightarrow \nabla')$ could be defined as

$$([\lambda^{\overline{a}_1} y_1 : s_1. ...\lambda^{\overline{a}_n} y_n : s_n. M_1\{y_1...y_n/x_1...x_n\}]_{\approx}, \ldots,$$
$$[\lambda^{\overline{a}_1} y_1 : s_1. ...\lambda^{\overline{a}_n} y_n : s_n. M_m\{y_1...y_n/x_1...x_n\}]_{\approx})$$

Hence, this approach is quite appealing when it comes to constructing exponentials in $Cl(Th)$. However, as previously indicated, we would have to introduce some form of subtyping and therefore the whole type and equation system of NLC would have to be modified, which may result in further complications.

- Another approach is motivated by the following isomorphisms in *FMSet*:

$$X^{\#\overline{a}} \Rightarrow_{fs} Y^{\#\overline{a}} \cong_{FMSet} (X \Rightarrow_{fs} Y)^{\#\overline{a}}$$
$$X^{\#\overline{a}} \Rightarrow_{fs} Y^{\#\overline{b}} \cong_{FMSet} (([\overline{b}]X)^{\#\overline{a}} \Rightarrow_{fs} Y)^{\#\overline{b}} \qquad (\overline{a} \cap \overline{b} = \emptyset)$$

  which provide us with an "alternative" presentation of exponentials in *FMSet*, a presentation that is much closer to what NLC can already express since domain (freshness) restricted function types $s^{\overline{a}} \Rightarrow s'$ and freshness assertions are already supported by NLC.

  The key idea of this approach is to extend NLC such that we can mimic the "alternative" presentation of an exponential (in *FMSet*) in a syntactically generated (classifying) category $Cl(Th)$. This, for example, includes the extension of NLC such that name abstraction (using a binding term former) and concretion, as well as local fresh atomic names can be captured. Note that the required extensions are rather extensive, but compared to the first approach the core of NLC as presented in Chapter 3 would remain unchanged.

While the first approach is appealing to resolve this issue, we believe that the introduction of subtyping is rather unpredictable. So, after some careful contemplation, we decided to pursue the second approach. Another strong argument in favour of the second approach is that we have already demonstrated how name abstraction, concretion and local fresh atomic names can be captured in NLC (see Chapter 4).

Due to the key role of the "alternative" presentation of exponentials in *FMSet* for the chosen approach, we will now sketch various properties that are required to obtain the aforementioned isomorphisms. We are grateful to Andrew Pitts for bringing these properties of *FMSet* to our attention, properties which have independently been observed and proven by Gabbay [35] (Lemma 8.12 and Lemma 8.18).

Tailored to fit our purpose, we introduce a slightly modified variant of name abstraction and also slightly reorganise the original proof arguments. Furthermore, we need to generalise the properties to hold for finite sets of names.

We recall that for each finite set of names $\overline{a}$, there exists a **freshness restricted category**, denoted by $FMSet^{\#\overline{a}}$, which is defined by

$$ob\ FMSet^{\#\overline{a}} \stackrel{\text{def}}{=} \{X \in ob\ FMSet \mid \overline{a} \# X\}$$
$$mor\ FMSet^{\#\overline{a}} \stackrel{\text{def}}{=} \{f \in mor\ FMSet \mid \overline{a} \# f\}$$

Moreover, there exists a **freshness restricted functor** $(-)^{\#\overline{a}} : FMSet^{\#\overline{a}} \to FMSet$ which is defined by

$$(X)^{\#\overline{a}} \stackrel{\text{def}}{=} \{x \in X \mid \overline{a} \# x\}$$
$$(f)^{\#\overline{a}} \stackrel{\text{def}}{=} f$$

The key property we are interested in is that $(-)^{\#\overline{a}}$ *has a right adjoint* $[\overline{a}](-)$*, which is moreover an adjoint equivalence between* $FMSet^{\#\overline{a}}$ *and* $FMSet$ *(*$FMSet^{\#\overline{a}} \simeq FMSet$*).* Note that from $((-)^{\#\overline{a}}, [\overline{a}](-), \eta, \varepsilon)$ being an adjoint equivalence, it follows by a routine argument that $([\overline{a}](-), (-)^{\#\overline{a}}, \varepsilon^{-1}, \eta^{-1})$ is also an adjoint equivalence. Hence, for any finite set of names $\overline{b}$ we also have that $[\overline{b}](-) \dashv (-)^{\#\overline{b}}$, which can be represented as a natural isomorphism $(\diamond 1)$ $FMSet^{\#\overline{b}}([\overline{b}]X, Y) \cong_{Set} FMSet(X, Y^{\#\overline{b}})$ for $X \in FMSet$ and $Y \in FMSet^{\#\overline{b}}$.

Further, it can be shown that for any $X \in FMSet^{\#\overline{b}}$ and $\overline{a} \cap \overline{b} = \emptyset$ we have that $(\diamond 2)$ $([\overline{a}]X)^{\#\overline{b}} \cong_{Set} [\overline{a}]X^{\#\overline{b}}$. By applying property $(\diamond 1)$ (for $X^{\#\overline{a}} \in FMSet$) and property $(\diamond 2)$ we can deduce the following:

$$FMSet(X^{\#\overline{a}}, Y^{\#\overline{b}}) \cong_{Set} FMSet^{\#\overline{b}}([\overline{b}]X^{\#\overline{a}}, Y) \cong_{Set} FMSet^{\#\overline{b}}(([\overline{b}]X)^{\#\overline{a}}, Y)$$

which by definition of $FMSet$ and $FMSet^{\#\overline{b}}$ can be presented as

$$X^{\#\overline{a}} \Rightarrow_{fs} Y^{\#\overline{b}} \cong_{Set} (([\overline{b}]X)^{\#\overline{a}} \Rightarrow_{fs} Y)^{\#\overline{b}}$$

It is not hard to see that this can be lifted to an isomorphism in $FMSet$, i.e. the left and right hand side are FM-sets and the morphisms observing the isomorphism in Set are finitely supported.

To sum it up. We first observed that NLC cannot capture exponentials of the form $X^{\#\overline{a}} \Rightarrow_{fs} Y^{\#\overline{b}}$. We then pointed out that there is an "alternative" notion of exponentials in *FMSet*, which is rather close to what NLC can already express. Hence, to construct an exponential in $Cl(Th)$ we intend to mimic this "alternative" presentation in $Cl(Th)$. But to achieve this we need to extend NLC with name abstraction, concretion and local fresh atomic names. As we have shown in Chapter 4 these concepts can already be captured in NLC (see ИNLC). However, our ultimate goal is to modify the definition of ИNLC such that the adjoint equivalence in *FMSet* is also taken into account. Hence, we wish to introduce a variant of ИNLC, referred to as [И]NLC, which is semantically motivated by the adjoint equivalence we have just introduced.

Given that the corresponding results of [И]NLC (up to soundness) only involve minor modifications of the corresponding results for ИNLC, we refrain from re-stating those results in this chapter. Moreover, due to the routine nature of the proof extensions for auxiliary results, we only provide proof details for important key lemmas and soundness. Thus, if we reference results of ИNLC in Chapter 4 in the context of [И]NLC we actually refer to their respective extensions for [И]NLC.

5.3.1 "Alternative" Exponentials in *FMSet*

We begin by introducing the isomorphism

$$X^{\#\overline{a}} \Rightarrow_{fs} Y^{\#\overline{a}} \cong_{FMSet} (X \Rightarrow_{fs} Y)^{\#\overline{a}},$$

which is an immediate extension of [35] (Lemma 8.12). For the second isomorphism ($\overline{a} \cap \overline{b} = \emptyset$) we wish to obtain:

$$X^{\#\overline{a}} \Rightarrow_{fs} Y^{\#\overline{b}} \cong_{FMSet} (([\overline{b}]X)^{\#\overline{a}} \Rightarrow_{fs} Y)^{\#\overline{b}}$$

This is achieved by proving two intermediate results. We first prove the existence

of an adjoint equivalence

$$FMSet^{\#\overline{b},a} \simeq FMSet^{\#\overline{b}} \quad (a \notin \overline{b}) \quad (\diamond)$$

which is a variant of the property proved in [35] (Lemma 8.18). From this, we can now directly deduce $FMSet^{\#\overline{a}} \simeq FMSet$ by using functors $(-)^{\#\overline{a}}$ and $[\overline{a}](-)$, which represent multiple applications of the respective functors of the adjoint equivalence $(\diamond)$. We now recall the functor $(-)^{\#a} : FMSet^{\#\overline{b},a} \to FMSet^{\#\overline{b}}$:

- $X \in FMSet^{\#\overline{b},a}$ is mapped to $X^{\#a} \stackrel{\text{def}}{=} \{x \in X \mid a \# x\} \in FMSet^{\#\overline{b}}$ and
- $f : X \to Y$ in $FMSet^{\#\overline{b},a}$ is mapped to $f^{\#a} : X^{\#a} \to Y^{\#a}$ in $FMSet^{\#\overline{b}}$, which is defined by $f^{\#a}(x \in X^{\#a}) \stackrel{\text{def}}{=} f(x) \in Y^{\#a}$.

As we have already seen, $X^{\#a}$ is an FM-set with $supp(X^{\#a}) = supp(X) \cup \{a\}$. Given that $\overline{b} \# X$ and $a \notin \overline{b}$, we have that $\overline{b} \# X^{\#a}$ and therefore $X^{\#a} \in FMSet^{\#\overline{b}}$.

From $a \# f$ and $a \# x$ we can directly deduce $a \# f(x)$. Hence, $f^{\#a}$ is well defined. Further, we have that $supp(f^{\#a}) = supp(f) \cup \{a\}$ and from f being finitely supported, we can deduce that $f^{\#a}$ is finitely supported as well. Due to the fact that $\overline{b} \# f$ and $a \notin \overline{b}$, we have that $\overline{b} \# f^{\#a}$ and therefore $f^{\#a} \in FMSet^{\#\overline{b}}$. The functor properties follow by routine computations. We continue by introducing the functor $[a](-) : FMSet^{\#\overline{b}} \to FMSet^{\#\overline{b},a}$, which is defined as follows:

- $X \in FMSet^{\#\overline{b}}$ is mapped to $[a]X \stackrel{\text{def}}{=} \{\langle a' \rangle\, x \mid a' \# X \wedge x \in (a\, a') \cdot X\} \in FMSet^{\#\overline{b},a}$, where $\langle a' \rangle\, x$ is the ordinary notion of name abstraction.
- $f : X \to Y$ in $FMSet^{\#\overline{b}}$ is mapped to $[a]f : [a]X \to [a]Y$ in $FMSet^{\#\overline{b},a}$, which is defined by $[a]f(z \in [a]X) \stackrel{\text{def}}{=} \mathsf{fresh}\ b\ \mathsf{in}\ \langle b \rangle((a\, b) \cdot f)(z\ @\ b)$

It is not hard to see that $supp([a]X) = supp(X) \setminus \{a\}$ and therefore $[a]X$, equipped with the permutation action $\pi \cdot \langle a' \rangle x \stackrel{\text{def}}{=} \langle \pi \cdot a' \rangle \pi \cdot x$, is an FM-set with $supp(\langle a' \rangle x) = supp(x) \setminus \{a'\}$. Given that $\overline{b} \# X$ and $a \notin \overline{b}$, we have that $\overline{b} \# [a]X$. Hence, we have that $[a]X \in FMSet^{\#\overline{b},a}$.

Due to the fact that $[a]f$ is defined using the freshness theorem, we need to demonstrate that the condition of the freshness theorem is met. We immediately obtain that $h \stackrel{\text{def}}{=} \Lambda b \in \mathbb{A} \to \langle b \rangle(((a\,b) \cdot f)(z \mathbin{@} b))$ is finitely supported by $supp(f) \cup supp(z) \cup \{a\}$. We then choose any name a' such that $a' \# h$. We now have $h(a') \stackrel{\text{def}}{=} \langle a' \rangle(((a\,a') \cdot f)(z \mathbin{@} a'))$ and by Lemma 2.2.12 we obtain that $a' \# h(a')$. Hence, $[a]f$ is well defined. Further, we have that $supp([a]f) = supp(f) \cup \backslash\{a\}$ and therefore $[a]f$ is an FM-function with $a \# [a]f$. Moreover, given that $\overline{b} \# f$ and $a \notin \overline{b}$, we have that $\overline{b} \# [a]f$. Thus, $[a]f \in FMSet^{\#\overline{b},a}$. The functor properties follow by routine computations. After we have defined both functors, we can now prove the key part, namely the adjoint equivalence.

Proposition 5.3.1 $(-)^{\#a} \dashv [a](-)$ *and moreover* $FMSet^{\#\overline{b},a} \simeq FMSet^{\#\overline{b}}$

Proof We prove that $(-)^{\#a} \dashv [a](-)$ is an adjunction via the unit and counit characterisation. We have chosen this characterisation because we directly use the counit in the definition of our categorical semantics. Further, we prove that $FMSet^{\#\overline{b},a} \simeq FMSet^{\#\overline{b}}$ is an adjoint equivalence by demonstrating that the unit and counit are natural isomorphisms. The unit and counit, as well as their respective inverses are defined as follows:

- The unit $\eta_a : 1_{FMSet^{\#\overline{b},a}} \Rightarrow [a](-) \circ (-)^{\#a}$ is a natural transformation with the components $\eta_{a,X} : X \to [a](X^{\#a})$ in $FMSet^{\#\overline{b},a}$ being defined by $\eta_{a,X}(x) \stackrel{\text{def}}{=}$ fresh a' in $\langle a' \rangle x$. The inverse $\eta^{-1}_{a,X} : [a](X^{\#a}) \to X$ in $FMSet^{\#\overline{b},a}$ is defined by $\eta^{-1}_{a,X}(\langle a' \rangle x) \stackrel{\text{def}}{=} x$.

- The counit $\varepsilon_a : (-)^{\#a} \circ [a](-) \Rightarrow 1_{FMSet^{\#\overline{b}}}$ is a natural transformation with the components $\varepsilon_{a,X} : ([a]X)^{\#a} \to X$ in $FMSet^{\#\overline{b}}$ being defined by $\varepsilon_{a,X}(\langle a' \rangle y) \stackrel{\text{def}}{=} (a\,a') \cdot y$. The inverse $\varepsilon^{-1}_{a,X} : X \to ([a]X)^{\#a}$ in $FMSet^{\#\overline{b}}$ is defined by $\varepsilon^{-1}_{a,X}(x) \stackrel{\text{def}}{=}$ fresh a' in $\langle a' \rangle((a\,a') \cdot x)$.

Let $x \in X$. In the case of $\eta_{a,X}$ we have to show that the condition of the freshness theorem holds. We can immediately observe that the function $\Lambda a' \in \mathbb{A} \to \langle a' \rangle x$ is finitely supported by $supp(x) \cup supp(X) \cup \{a\}$. Then, for any $b \# (x, X, a)$ we directly obtain that $b \# \langle b \rangle x$. Let $a' \# (a, X, x)$. We next have to show that

$$\langle a' \rangle x \in [a](X^{\#a}) \stackrel{\text{def}}{=} \{\langle a' \rangle x \mid a' \# X^{\#a}, x \in (a\, a') \cdot X^{\#a}\}$$

From $a' \# X$ and $a' \neq a$ we can deduce that $a' \# X^{\#a}$ holds. Further, from $x \in X$, $a' \# (x, X)$ and $a \# X$, we can deduce that $x \in X^{\#a'} = ((a\, a') \cdot X)^{\#(a\, a') \cdot a} = (a\, a') \cdot X^{\#a}$. Hence, $\eta_{a,X}$ is well defined and it can easily be seen that $\eta_{a,X}$ is finitely supported by $supp(X)$. Hence, given that $\overline{b}, a \# X$ we immediately get that $\overline{b}, a \# \eta_{a,X}$, and therefore $\eta_{a,X} \in FMSet^{\#\overline{b},a}$. The inverse map $\eta^{-1}_{a,X}$ is well defined and finitely supported by $supp(X)$. So, we have $\eta^{-1}_{a,X} \in FMSet^{\#\overline{b},a}$. The argument for the counit follows analogously.

We continue by demonstrating that η_a and ε_a are natural transformations: Let $f : X \to Y$ in $FMSet^{\#\overline{b},a}$ $(\overline{b}, a \# f)$ and $x \in X$.

$$\begin{aligned}
([a]f^{\#a})(\eta_{a,X}(x)) &\stackrel{\text{def}}{=} [a]f^{\#a}(\langle a' \rangle x) && \text{(fresh name } a') \\
&\stackrel{\text{def}}{=} \langle a'' \rangle(((a''\, a) \cdot f^{\#a})(\langle a' \rangle x \mathbin{@} a'')) && \text{(fresh name } a'') \\
&\stackrel{\text{def}}{=} \langle a'' \rangle(f^{\#a''}(a'\, a'') \cdot x) && (a, a'' \# f;\ a'' \# x) \\
&\stackrel{\text{def}}{=} \langle a'' \rangle(f((a'\, a'') \cdot x)) \\
&= \langle a'' \rangle(f(x)) && (a', a'' \# x) \\
&\stackrel{\text{def}}{=} \eta_{a,Y}(f(x))
\end{aligned}$$

Let $f : X \to Y$ in $\mathcal{C}^{\#\overline{b}}$ and $\langle a' \rangle x \in ([a]X)^{\#a}$. So, by definition we have that

$a \# \langle a' \rangle x$. We consider the case where $a \neq a'$. Hence, we have that $a \# x$.

$$\begin{aligned}
\varepsilon_{a,Y}(([a]f)^{\#a}(\langle a' \rangle x)) &\stackrel{\text{def}}{=} \varepsilon_{a,Y}([a]f(\langle a' \rangle x)) \\
&\stackrel{\text{def}}{=} \varepsilon_{a,Y}(\langle a'' \rangle(((a\,a'') \cdot f)(\langle a' \rangle x \mathbin{@} a''))) && \text{(fresh name } a'') \\
&\stackrel{\text{def}}{=} (a\,a'') \cdot (((a\,a'') \cdot f)((a'\,a'') \cdot x)) \\
&= f((a\,a'')(a'\,a'') \cdot x) \\
&= f((a'\,a)(a\,a'') \cdot x) \\
&= f((a'\,a) \cdot x) && (a, a'' \# x) \\
&\stackrel{\text{def}}{=} f(\varepsilon_{a,X}(\langle a' \rangle x))
\end{aligned}$$

Next, we prove that the previously defined inverse maps for η_a and ε_a are indeed the respective inverses, and therefore the components of η_a and ε_a are natural isomorphisms. Let $x \in X$ $(\overline{b}, a \# X)$

$$\begin{aligned}
\eta^{-1}_{a,X}(\eta_{a,X}(x)) &\stackrel{\text{def}}{=} \eta^{-1}_{a,X}(\langle a' \rangle x) && \text{(fresh name } a') \\
&\stackrel{\text{def}}{=} x
\end{aligned}$$

Let $\langle a' \rangle y \in [a](X^{\#a})$. So, by definition, we have $a' \# X^{\#a}$ and $y \in (a\,a') \cdot X^{\#a}$. We can now deduce that $a' \# X$, $a' \neq a$ and $(a\,a') \cdot X^{\#a} = (a\,a') \cdot X^{\#a'} = X^{\#a'}$. Hence, we have that $a' \# y$.

$$\begin{aligned}
\eta_{a,X}(\eta^{-1}_{a,X}(\langle a' \rangle y)) &\stackrel{\text{def}}{=} \eta_{a,X}(y) \\
&\stackrel{\text{def}}{=} \langle a'' \rangle y && \text{(fresh name } a'') \\
&= \langle a'' \rangle (a''\,a') \cdot y && (a', a'' \# y) \\
&\stackrel{\text{def}}{=} \langle a' \rangle y && (a'' \# a', y)
\end{aligned}$$

We continue with the counit ε_a: Let $x \in X$

$$\begin{aligned}
\varepsilon_{a,X}(\varepsilon^{-1}_{a,X}(x)) &\stackrel{\text{def}}{=} \varepsilon_{a,X}(\langle a' \rangle ((a\,a') \cdot x)) && \text{(fresh name } a') \\
&\stackrel{\text{def}}{=} (a\,a') \cdot (a\,a') \cdot x \\
&= x
\end{aligned}$$

Let $\langle a'\rangle y \in [a]X$ and $a \# \langle a'\rangle y$. We only consider the case $a \neq a'$. So, $a \# y$.

$$\begin{aligned}
\varepsilon^{-1}_{a,X}(\varepsilon_{a,X}(\langle a'\rangle y)) &\stackrel{\text{def}}{=} \varepsilon^{-1}_{a,X}((a\,a')\cdot y) \\
&\stackrel{\text{def}}{=} \langle a''\rangle((a\,a'')(a\,a')\cdot y) && \text{(fresh name } a'') \\
&\stackrel{\text{def}}{=} \langle a''\rangle((a''\,a')(a\,a'')\cdot y) \\
&\stackrel{\text{def}}{=} \langle a''\rangle((a''\,a')\cdot y) && (a, a'' \# y) \\
&\stackrel{\text{def}}{=} \langle a'\rangle y && (a'' \# (a, y))
\end{aligned}$$

We complete this proof by checking the triangle equalities: Let $\langle a'\rangle y \in [a]Y$. We consider the case $a' \neq a$.

$$\begin{aligned}
[a]\varepsilon_{a,Y}(\eta_{a,[a]Y}(\langle a'\rangle y)) &\stackrel{\text{def}}{=} [a]\varepsilon_{a,Y}(\langle a''\rangle\langle a'\rangle y) && \text{(fresh name } a'') \\
&\stackrel{\text{def}}{=} \langle a'''\rangle(((a\,a''')\cdot\varepsilon_{a,Y})(\langle a''\rangle\langle a'\rangle y \mathbin{@} a''')) && \text{(fresh name } a''') \\
&= \langle a'''\rangle((a\,a''')\cdot\varepsilon_{a,Y}((a\,a''')\cdot(\langle a''\rangle\langle a'\rangle y \mathbin{@} a'''))) \\
&= \langle a'''\rangle((a\,a''')\cdot\varepsilon_{a,Y}((a\,a''')\cdot((a''\,a''')\cdot\langle a'\rangle y))) \\
&\stackrel{\text{def}}{=} \langle a'''\rangle((a\,a''')\cdot\varepsilon_{a,Y}(\langle a'\rangle(a\,a''')(a''\,a''')\cdot y)) \\
&\stackrel{\text{def}}{=} \langle a'''\rangle((a\,a''')(a\,a')(a\,a''')(a''\,a''')\cdot y) \\
&= \langle a'''\rangle((a\,a''')(a\,a''')(a'''\,a')(a''\,a''')\cdot y) \\
&= \langle a'''\rangle((a'''\,a')\cdot y) && (a'', a''' \# y) \\
&= \langle a'\rangle y && (a''' \# (a', y))
\end{aligned}$$

Let $x \in X^{\#a}$ $(a \# x)$

$$\begin{aligned}
\varepsilon_{a,X^{\#a}}((\eta_{a,X})^{\#a}(x)) &\stackrel{\text{def}}{=} \varepsilon_{a,X^{\#a}}(\eta_{a,x}(x)) \\
&\stackrel{\text{def}}{=} \varepsilon_{a,X^{\#a}}(\langle a'\rangle x) && \text{(fresh name } a') \\
&\stackrel{\text{def}}{=} (a\,a')\cdot x \\
&= x && (a, a' \# x)
\end{aligned}$$

□

To ultimately obtain the second isomorphism we require a second property, which is a variant of [35] (Lemma 8.13). Here, we prove it for single names, but it extends easily to finite sets of names.

Lemma 5.3.2 *If $X \in FMSet^{\#b}$ and $a \neq b$, then $[a]X^{\#b} = ([a]X)^{\#b}$.*

Proof Let $X \in FMSet^{\#b}$ and $a \neq b$.

($\subseteq$): Take $\langle a' \rangle x' \in [a]X^{\#b}$. We have by definition that $a' \# X^{\#b}$ and $x' \in (a\,a') \cdot (X^{\#b})$. Given that $supp(X^{\#b}) = supp(X) \cup \{b\}$, we can deduce that $a' \# X$ and $a' \neq b$. Further, we have that $x' \in (a\,a') \cdot (X^{\#b}) \subseteq (a\,a') \cdot X$. Hence, we have that $\langle a' \rangle x' \in [a]X$. What remains to be shown is that $b \# \langle a' \rangle x'$. From $x' \in (a\,a') \cdot (X^{\#b})$ we can deduce that $(a\,a') \cdot x' \in X^{\#b}$. So, we have that $b \# (a\,a') \cdot x'$. Given that $b \# (a, a')$ and equivariance we have that $b \# x'$, and therefore $b \# \langle a' \rangle x'$.

($\supseteq$): Take $\langle a' \rangle x' \in ([a]X)^{\#b}$. So, by definition we have that $b \# \langle a' \rangle x'$, $a' \# X$ and $x' \in (a\,a') \cdot X$. We consider the case ($a' \neq b$). The other case follows by choosing a representative for $\langle a' \rangle x'$ which is fresh for b. We can now deduce that $a' \# X^{\#b}$ and $b \# x'$. Considering the assumption $x' \in (a\,a') \cdot X$ and $b \# x'$ we obtain that $x' \in ((a\,a') \cdot X)^{\#b} = (a\,a') \cdot (X^{\#b})$. □

Remark 5.3.3 *We are not yet finished, because to support our claim that exponentials in FMSet can be presented in an "alternative" way, we also have to consider the case $\overline{a} \cap \overline{b} \neq \emptyset$. So, let $\overline{c} = \overline{a} \cap \overline{b}$. We can now use the previously introduced generalised isomorphism to deduce the following:*

$$\begin{aligned} X^{\#\overline{a}} \Rightarrow_{fs} Y^{\#\overline{b}} &\cong_{FMSet} (X^{\#\overline{a}\setminus\overline{c}} \Rightarrow_{fs} Y^{\#\overline{b}\setminus\overline{c}})^{\#\overline{c}} \\ &\cong_{FMSet} (((\overline{b} \setminus \overline{c}]X)^{\#\overline{a}\setminus\overline{c}} \Rightarrow_{fs} Y)^{\#\overline{b}\setminus\overline{c}})^{\#\overline{c}} \\ &= ((\overline{b} \setminus \overline{c}]X)^{\#\overline{a}\setminus\overline{c}} \Rightarrow_{fs} Y)^{\#\overline{b}} \end{aligned}$$

5.3.2 [И]NLC: ИNLC with Dependent Name Abstraction Types

Based on the previously observed properties for *FMSet*, we introduce a new calculus, which is denoted by [И]NLC. We take ИNLC, which was introduced in Chapter 4, as our starting point. Recall that the aim of this semantically motivated extension of NLC is to mimic the "alternative" presentation of an exponential (in *FMSet*) in a syntactically generated (classifying) category $Cl(Th)$.

We augment the type system with a **dependent name abstraction type** $[a]s$, instead of $Name.s$, which is a binder on names. The permutation action on types is extended as follows:

$$\pi \cdot [a]s \stackrel{\text{def}}{=} [\pi \cdot a]\pi \cdot s$$

with $supp([a]s) = \{a\} \cup supp(s)$. The set of free names of types, referred to as $fnt(s)$, is recursively defined as follows:

$$\begin{aligned} fnt(\gamma) &\stackrel{\text{def}}{=} supp(\gamma) \\ fnt({s_1}^{\overline{a}} \Rightarrow s_2) &\stackrel{\text{def}}{=} supp(s_1) \cup supp(s_2) \cup \overline{a} \\ fnt([a]s) &\stackrel{\text{def}}{=} supp(s) \setminus a \end{aligned}$$

Next, we introduce α-equivalence for types with the following key rule:

$$\frac{(И b)\ (b\, a_1) \cdot s_1 \equiv_\alpha (b\, a_2) \cdot s_2}{[a_1]s_1 \equiv_\alpha [a_2]s_2}$$

The permutation action can be shown to preserve α-equivalence, and therefore the permutation action can be lifted on α-equivalence classes of types with $supp([s]_\alpha) = fnt(s)$.

We extend the collection of raw terms with **name abstraction** and **concretion**, respectively denoted by $\langle\langle a \rangle\rangle M$ and $M \mathbin{@} a$, as well as a term former for **local fresh atomic names** $\mathsf{fr}\, a.\, M$. Analogous to $\langle a \rangle M$, occurrences of a in M are *not* bound in $\langle\langle a \rangle\rangle M$. The permutation actions on the resulting raw terms (and expressions) are defined as for ИNLC.

The set of typed expressions and theorems of an [И]NLC-theory Th is defined to be the least set of judgements containing the axioms of Th and being closed under the rules in Table 5.2 and Table 5.3.

$$(\mathsf{NABSD})\quad \frac{\nabla^{\#a} \vdash^{[И]\mathrm{NLC}} M : (a\,a') \cdot s}{\nabla \vdash^{[И]\mathrm{NLC}} \langle\langle a' \rangle\rangle M : [a]s}\,(a \# (\nabla, M), a' \# s)$$

$$(\mathsf{CONCD})\quad \frac{\nabla \vdash^{[И]\mathrm{NLC}} b \# F : [a]s}{\nabla^{\#a} \vdash^{[И]\mathrm{NLC}} F \mathbin{@} b : (a\,b) \cdot s}\,(a \# \nabla)$$

$$(\mathsf{LFAN})\quad \frac{\nabla^{\#a} \vdash^{[И]\mathrm{NLC}} a \# M : s}{\nabla \vdash^{[И]\mathrm{NLC}} \mathsf{fr}\, a.\, M : s}\,(a \# \nabla)$$

Table 5.2: [И]NLC Typing Rules

While the rule (LFAN) stays unchanged, compared to ИNLC, we have minor modifications for rules (NABSD) and (CONCD), which are both semantically motivated by the adjunction $(-)^{\#a} \dashv [a](-)$. The final shape of both rules becomes obvious when we present the adjunction as follows:

$$\frac{\begin{array}{ll} \mathit{FMSet} & \boxed{D^{\#a}} \to X \cong \boxed{(a\,a') \cdot X} \end{array}}{\begin{array}{ll} \mathit{FMSet}^{\#a} & \boxed{D} \to \boxed{[a]X} \end{array}}$$

5.3.3 A Sound Categorical Semantics for [И]NLC in *FMSet*

We introduce a categorical semantics for [И]NLC in *FMSet*, which relies on the additional structures that we have just introduced for *FMSet* (see Table 5.4). Note that the partial interpretation function is not specifically defined for *FMSet*, but for any category with such structures (later referred to as [И]FM-cccs). This is necessary for a future categorical type theory correspondence for [И]NLC. However, as part of the soundness proof we unravel these definitions (for *FMSet*) and observe that it is

$$(\mathsf{BND})\ \frac{\nabla^{\#a} \vdash^{[И]\mathrm{NLC}} M : (a\,a') \cdot s \qquad \nabla \vdash^{[И]\mathrm{NLC}} b \mathbin{\#} \langle\langle a' \rangle\rangle M : [a]s}{\nabla^{\#a} \vdash^{[И]\mathrm{NLC}} (\langle\langle a' \rangle\rangle M) \mathbin{@} b \approx (a'\,b) * M : (a\,b) \cdot s}\ (a \mathbin{\#} \nabla, M)$$

$$(\mathsf{END})\ \frac{\nabla \vdash^{[И]\mathrm{NLC}} b \mathbin{\#} F : [a]s}{\nabla \vdash^{[И]\mathrm{NLC}} \langle\langle b \rangle\rangle (F \mathbin{@} b) \approx F : [a]s}$$

$$(\mathsf{BNABSD})\ \frac{(И\,b)\ \nabla^{\#a,b} \vdash^{[И]\mathrm{NLC}} (b\,a') * M \approx (b\,a'') * M' : (a\,a') \cdot s}{\nabla \vdash^{[И]\mathrm{NLC}} \langle\langle a' \rangle\rangle M \approx \langle\langle a'' \rangle\rangle M' : [a]s}\quad \begin{array}{l}(a \mathbin{\#} (\nabla, M, M');\\ a', a'' \mathbin{\#} s)\end{array}$$

$$(\mathsf{CCD})\ \frac{\nabla \vdash^{[И]\mathrm{NLC}} b \mathbin{\#} F : [a]s \qquad \nabla \vdash^{[И]\mathrm{NLC}} F \approx F' : [a]s}{\nabla^{\#a} \vdash^{[И]\mathrm{NLC}} F \mathbin{@} b \approx F' \mathbin{@} b : (a\,b) \cdot s}\ (a \mathbin{\#} \nabla)$$

$$(\mathsf{LFANR})\ \frac{\nabla^{\#a} \vdash^{[И]\mathrm{NLC}} a \mathbin{\#} M : s}{\nabla^{\#a} \vdash^{[И]\mathrm{NLC}} \mathsf{fr}\, a.\, M \approx M : s}\ (a \mathbin{\#} \nabla)$$

$$(\mathsf{LFANS})\ \frac{\nabla^{\#a,a'} \vdash^{[И]\mathrm{NLC}} a, a' \mathbin{\#} M : s}{\nabla \vdash^{[И]\mathrm{NLC}} \mathsf{fr}\, a.\, \mathsf{fr}\, a'.\, M \approx \mathsf{fr}\, a'.\, \mathsf{fr}\, a.\, M : s}\ (a \neq a)$$

$$(\mathsf{LFANL})\ \frac{\nabla^{\#a'}, a', \overline{a} \mathbin{\#} x : s \vdash^{[И]\mathrm{NLC}} a' \mathbin{\#} M : s'}{\nabla \vdash^{[И]\mathrm{NLC}} \mathsf{fr}\, a'.\, \lambda^{\overline{a}} x : s.\, M \approx \lambda^{\overline{a}} x : s.\, \mathsf{fr}\, a'.\, M : s^{\overline{a}} \Rightarrow s'}\ (a' \notin \overline{a})$$

Table 5.3: [И]NLC Equation Rules

close to the to the partial interpretation function introduced for the model-theoretic semantics for ИNLC (in *FMSet*) that we introduced in Chapter 4.

$$\frac{\dfrac{[\![\nabla^{\#a} \vdash^{[И]\mathrm{NLC}} M : (a\,a') \cdot s]\!] \blacktriangleright m : [\![\nabla]\!]^{\#a} \to (a\,a') \cdot [\![s]\!]}{[\![\nabla \vdash^{[И]\mathrm{NLC}} \langle\langle a' \rangle\rangle M : [a]s]\!] \blacktriangleright [a]((a\,a')_{(a\,a') \cdot [\![s]\!]} \circ m) \circ \varepsilon_{a,[\![\nabla]\!]} : [\![\nabla]\!] \to [a][\![\nabla]\!]^{\#a} \to [a][\![s]\!]} \qquad [\![\nabla \vdash^{[И]\mathrm{NLC}} b \mathbin{\#} F : [a]s]\!] \blacktriangleright f : [\![\nabla]\!] \to ([a][\![s]\!])^{\#b}}{[\![\nabla^{\#a} \vdash^{[И]\mathrm{NLC}} F \mathbin{@} b : (a\,b) \cdot s]\!] \blacktriangleright conc_b \circ f \circ i^a_{[\![\nabla]\!]} : [\![\nabla]\!]^{\#a} \to [\![\nabla]\!] \to ([a][\![s]\!])^{\#b} \to (a\,b) \cdot [\![s]\!]}$$

$$\frac{[\![\nabla^{\#a} \vdash^{[И]\mathrm{NLC}} a \mathbin{\#} M : s]\!] \blacktriangleright \theta : [\![\nabla]\!]^{\#a} \to [\![s]\!]^{\#a}}{[\![\nabla \vdash^{[И]\mathrm{NLC}} \mathsf{fr}\, a.\, M : s]\!] \blacktriangleright \varepsilon^{-1}_{a,[\![s]\!]} \circ [a]\theta \circ \varepsilon_{a,[\![\nabla]\!]} : [\![\nabla]\!] \to [a][\![\nabla]\!]^{\#a} \to [a][\![s]\!]^{\#a} \to [\![s]\!]}$$

Table 5.4: Categorical Semantics for [И]NLC in *FMSet*

Extension of Soundness Theorem for $[\mathsf{И}]$NLC

We recall that in *FMSet*, i^a is an inclusion map. We observe that for $a \# (\nabla, M, \tau)$ and $\nabla^{\#a} \vdash^{[\mathsf{И}]\mathrm{NLC}} M : \tau$, we can deduce that $\nabla^{\#a} \vdash^{[\mathsf{И}]\mathrm{NLC}} a \# M : \tau$. Moreover, it follows that $[\![\nabla^{\#a} \vdash^{[\mathsf{И}]\mathrm{NLC}} M : \tau]\!] \blacktriangleright m$ and the condition $\dagger(a, m)$ holds. This is analogous to Lemma 3.2.19 for the model-theoretic semantics. Let $c \# (\nabla, a, M, \tau)$ and $d \in [\![\nabla]\!]^{\#a,c}$. So, $a, c \# d$

$$
\begin{aligned}
& ((a\,c)_{[\![\tau]\!]} \circ [\![\nabla^{\#a} \vdash^{[\mathsf{И}]\mathrm{NLC}} M : \tau]\!] \circ i^c_{[\![\nabla]\!]^{\#a,c}})(d) && \\
&\stackrel{\text{def}}{=} (a\,c)_{[\![\tau]\!]}([\![\nabla^{\#a} \vdash^{[\mathsf{И}]\mathrm{NLC}} M : \tau]\!](d)) && (\text{in } \mathit{FMSet}) \\
&= (a\,c)_{[\![\tau]\!]}([\![\nabla^{\#a,c} \vdash^{[\mathsf{И}]\mathrm{NLC}} M : \tau]\!](d)) && (\text{in } \mathit{FMSet},\ c \# d) \\
&\stackrel{\text{def}}{=} (a\,c) \cdot [\![\nabla^{\#a,c} \vdash^{[\mathsf{И}]\mathrm{NLC}} M : \tau]\!](d) && (\text{in } \mathit{FMSet}) \\
&= ((a\,c) \cdot [\![\nabla^{\#a,c} \vdash^{[\mathsf{И}]\mathrm{NLC}} M : \tau]\!])((a\,c) \cdot d) && \\
&= ([\![(a\,c) \cdot \nabla^{\#a,c} \vdash^{[\mathsf{И}]\mathrm{NLC}} (a\,c) \cdot M : (a\,c) \cdot \tau]\!])((a\,c) \cdot d) && (\text{Lemma 5.2.3 (ii)}) \\
&= ([\![\nabla^{\#a,c} \vdash^{[\mathsf{И}]\mathrm{NLC}} M : \tau]\!])(d) && (a, c \# (\nabla, M, \tau, d)) \\
&= ([\![\nabla^{\#a} \vdash^{[\mathsf{И}]\mathrm{NLC}} M : \tau]\!])(d) && (\text{in } \mathit{FMSet},\ c \# d)
\end{aligned}
$$

Hence, by definition of our semantics we have that $[\![\nabla^{\#a} \vdash^{[\mathsf{И}]\mathrm{NLC}} a \# M : \tau]\!] \blacktriangleright m^*$ with $m = i^a \circ m^*$ ($\diamond 1$). We begin with typing rules and their denotation for *FMSet*:

(NABSD): Suppose that $\nabla \vdash^{[\mathsf{И}]\mathrm{NLC}} \langle\langle a' \rangle\rangle M : [a]s$. From this we can deduce that $\nabla^{\#a} \vdash^{[\mathsf{И}]\mathrm{NLC}} M : (a\,a') \cdot s$, $a \# (\nabla, M)$ and $a' \# s$. We now need to show that $[\![\nabla \vdash^{[\mathsf{И}]\mathrm{NLC}} \langle\langle a' \rangle\rangle M : [a]s]\!]\Downarrow$. By induction we have $[\![\nabla^{\#a} \vdash^{[\mathsf{И}]\mathrm{NLC}} M : (a\,a') \cdot s]\!] \blacktriangleright m$. So, the existence follows immediately by definition of the semantics. Next, we demonstrate

how the interpretation can be simplified in the case of *FMSet*. Let $d \in [\![\nabla]\!]$

$$
\begin{aligned}
& [\![\nabla \vdash^{[\mathcal{N}]\mathrm{NLC}} \langle\langle a' \rangle\rangle M : [a]s]\!](d) \\
\stackrel{\text{def}}{=}\ & [a]((a\,a')_{(a\,a')\cdot[\![s]\!]} \circ m)(\varepsilon_{a,[\![\nabla]\!]}(d)) \\
\stackrel{\text{def}}{=}\ & [a]((a\,a')_{(a\,a')\cdot[\![s]\!]} \circ m)(\langle a'' \rangle d) && \text{(fresh name } a'') \\
\stackrel{\text{def}}{=}\ & \langle a''' \rangle(((a\,a''') \cdot [(a\,a')_{(a\,a')\cdot[\![s]\!]} \circ m])(d)) \\
\stackrel{\text{def}}{=}\ & \langle a''' \rangle(a\,a''') \cdot [(a\,a')_{(a\,a')\cdot[\![s]\!]} \circ m])((a\,a''') \cdot d) \\
\stackrel{\text{def}}{=}\ & \langle a''' \rangle(a\,a''') \cdot [(a\,a')_{(a\,a')\cdot[\![s]\!]}(m((a\,a''') \cdot d))]) \\
\stackrel{\text{def}}{=}\ & \langle a''' \rangle(a\,a''') \cdot [(a\,a') \cdot (m((a\,a''') \cdot d))]) \\
=\ & \langle a''' \rangle(a'\,a''') \cdot (a\,a''') \cdot (m((a\,a''') \cdot d)) \\
=\ & \langle a' \rangle(a\,a''') \cdot [m((a\,a''') \cdot d)] && (a''' \,\#\, (a\,a''') \cdot (m((a\,a''') \cdot d)) \text{ using } (\diamond 1)) \\
=\ & \langle a' \rangle((a\,a''') \cdot m)(d)
\end{aligned}
$$

Hence, for some fresh name b we have the following interpretation

$$[\![\nabla \vdash^{[\mathcal{N}]\mathrm{NLC}} \langle\langle a' \rangle\rangle M : [a]s]\!](d) = \langle a' \rangle((a\,b) \cdot m)(d)$$

(CONCD): This case follows similarly to (AP) for NLC. The interpretation can be simplified as follows:

$$
\begin{aligned}
[\![\nabla^{\#a} \vdash^{[\mathcal{N}]\mathrm{NLC}} F \mathbin{@} b : (a\,b) \cdot s]\!](d) &\stackrel{\text{def}}{=} (conc_b \circ f \circ i^a_{[\![\nabla]\!]})(d) \\
&\stackrel{\text{def}}{=} conc_b(f(d)) \\
&\stackrel{\text{def}}{=} f(d) \mathbin{@} b \\
&\stackrel{\text{def}}{=} ([\![\nabla \vdash^{[\mathcal{N}]\mathrm{NLC}} b \,\#\, F : [a]s]\!])(d) \mathbin{@} b
\end{aligned}
$$

(LFAN): This case follows similarly to (AP) for NLC. The interpretation can be

simplified as follows: Let $[\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} a \# M : s]\!] \blacktriangleright \theta$ and $d \in [\![\nabla]\!]$.

$$\begin{aligned}
[\![\nabla \vdash^{[\mathcal{M}]\mathrm{NLC}} \mathsf{fr}\, a.\, M : s]\!](d) &\stackrel{\mathrm{def}}{=} \varepsilon^{-1}_{a,[\![\nabla]\!]}([a]\theta(\varepsilon_{a,[\![s]\!]}(d))) \\
&\stackrel{\mathrm{def}}{=} \varepsilon^{-1}_{a,[\![\nabla]\!]}([a]\theta(\langle a''\rangle d)) && \text{(fresh name } a'') \\
&\stackrel{\mathrm{def}}{=} \varepsilon^{-1}_{a,[\![\nabla]\!]}(\langle a'\rangle(((a\,a')\cdot\theta)(\langle a''\rangle d \mathbin{@} a'))) && \text{(fresh name } a') \\
&= \varepsilon^{-1}_{a,[\![\nabla]\!]}(\langle a'\rangle(((a\,a')\cdot\theta)(d))) && (a', a'' \# d) \\
&\stackrel{\mathrm{def}}{=} ((a\,a')\cdot\theta)(d)
\end{aligned}$$

Hence, for some fresh name a' we have that

$$[\![\nabla \vdash^{[\mathcal{M}]\mathrm{NLC}} \mathsf{fr}\, a.\, M : s]\!](d) = ((a\,a')\cdot[\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} a \# M : s]\!])(d)$$

The justification that we can use fresh a' in $(-)$ instead follows similarly to the model-theoretic semantics in *FMSet*. We continue with the equational rules. The existence part of the Kleene equality follows analogue to the typing cases. So, we only concentrate on the equational part:

(BND): Suppose that $\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} (\langle\langle a'\rangle\rangle M) \mathbin{@} b \approx (a'\,b) * M : (a\,b)\cdot s$. From this we can deduce that $\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} M : (a\,a')\cdot s$, $\nabla \vdash^{[\mathcal{M}]\mathrm{NLC}} b \# \langle\langle a'\rangle\rangle M : [a]s$ and $a \# (\nabla, M)$. Further, it follows by definition that $a' \# s$ and $b \# [a]s$. We consider the case where all names are distinct. Hence, we have that $b \# s$. The equation is deduced as follows: Let $d \in [\![\nabla]\!]^{\#a}$.

$$\begin{aligned}
&[\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} (\langle\langle a'\rangle\rangle M) \mathbin{@} b : (a\,b)\cdot s]\!](d) \\
&\stackrel{\mathrm{def}}{=} [\![\nabla \vdash^{[\mathcal{M}]\mathrm{NLC}} b \# \langle\langle a'\rangle\rangle M : [a]s]\!](d) \mathbin{@} b \\
&= [\![\nabla \vdash^{[\mathcal{M}]\mathrm{NLC}} \langle\langle a'\rangle\rangle M : [a]s]\!](d) \mathbin{@} b && \text{(in } \mathit{FMSet}) \\
&\stackrel{\mathrm{def}}{=} [\langle a'\rangle((a\,a'')\cdot[\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} M : (a\,a')\cdot s]\!])(d)] \mathbin{@} b && \text{(fresh name } a'') \\
&\stackrel{\mathrm{def}}{=} (a'\,b)\cdot((a\,a'')\cdot[\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} M : (a\,a')\cdot s]\!])(d) \\
&\stackrel{\mathrm{def}}{=} (a'\,b)\cdot(a\,a'')\cdot[\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} M : (a\,a')\cdot s]\!]((a\,a'')\cdot d) \\
&= (a'\,b)\cdot(a\,a'')\cdot[\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} M : (a\,a')\cdot s]\!](d) && (a, a'' \# d)
\end{aligned}$$

$$
\begin{aligned}
&= (a'\,b)\cdot[\![\nabla^{\#a}\vdash^{[\mathcal{M}]\mathrm{NLC}} M : (a\,a')\cdot s]\!](d) && \text{(see } (\diamond 2)\text{)}\\
&= ((a'\,b)_{(-)}\circ[\![\nabla^{\#a}\vdash^{[\mathcal{M}]\mathrm{NLC}} M : (a\,a')\cdot s]\!])(d) && \text{(in } \mathit{FMSet}\text{)}\\
&= [\![\nabla^{\#a}\vdash^{[\mathcal{M}]\mathrm{NLC}} (a'\,b)*M : (a'\,b)\cdot(a\,a')\cdot s]\!](d) && \text{(Lemma 5.2.3 (iii))}\\
&= [\![\nabla^{\#a}\vdash^{[\mathcal{M}]\mathrm{NLC}} (a'\,b)*M : (a\,b)\cdot(a'\,b)\cdot s]\!](d)\\
&= [\![\nabla^{\#a}\vdash^{[\mathcal{M}]\mathrm{NLC}} (a'\,b)*M : (a\,b)\cdot s]\!](d) && (a', b \# s)
\end{aligned}
$$

For $(\diamond 2)$ we observe that $a \# [\![\nabla^{\#a}\vdash^{[\mathcal{M}]\mathrm{NLC}} M : (a\,a')\cdot s]\!](d)$, which follows from $(\diamond 1)$. Further, we have $a'' \# [\![\nabla^{\#a}\vdash^{[\mathcal{M}]\mathrm{NLC}} M : (a\,a')\cdot s]\!](d)$, because a'' is freshly chosen.

(END): The equation is deduced as follows: Let $d \in [\![\nabla]\!]$

$$
\begin{aligned}
&[\![\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} \langle\langle b\rangle\rangle(F\mathbin{@} b) : [a]s]\!](d)\\
&\overset{\mathrm{def}}{=} \langle b\rangle(a\,a'')\cdot[\![\nabla^{\#a}\vdash^{[\mathcal{M}]\mathrm{NLC}} F\mathbin{@} b : (a\,b)\cdot s]\!]((a\,a'')\cdot d) && \text{(fresh name } a''\text{)}\\
&\overset{\mathrm{def}}{=} \langle b\rangle(a\,a'')\cdot[[\![\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} b \# F : [a]s]\!]((a\,a'')\cdot d)]\mathbin{@} b\\
&\overset{\mathrm{def}}{=} \langle b\rangle(a\,a'')\cdot[[\![\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} F : [a]s]\!]^*((a\,a'')\cdot d)]\mathbin{@} b\\
&= \langle b\rangle[(a\,a'')\cdot[\![\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} F : [a]s]\!]^*((a\,a'')\cdot d)]\mathbin{@} b\\
&= \langle b\rangle[[\![(a\,a'')\cdot\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} (a\,a'')\cdot F : (a\,a'')\cdot[a]s]\!]^*(d)]\mathbin{@} b\\
&= \langle b\rangle[\![\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} F : [a]s]\!]^*(d)\mathbin{@} b && a \# (\nabla, F, [a]s)\\
&= [\![\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} F : [a]s]\!]^*(d) && (\diamond)\\
&= i^b\circ[\![\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} F : [a]s]\!](d)\\
&= [\![\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} F : [a]s]\!](d) && \text{(in } \mathit{FMSet}\text{)}
\end{aligned}
$$

with $(\diamond)$ being $b \# [\![\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} F : [a]s]\!]^*(d)$ and Lemma 2.2.13.

(BNABSD): Suppose $\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}}\langle\langle a'\rangle\rangle M' \approx \langle\langle a''\rangle\rangle M'' : [a]s$. From this we can deduce that $\nabla^{\#a,b}\vdash^{[\mathcal{M}]\mathrm{NLC}} (b\,a')*M' \approx (b\,a'')*M'' : (a\,a')\cdot s$ for some/any fresh b, $a \# (\nabla, M', M'')$ and $a', a'' \# s$. Let $[\![\nabla^{\#a}\vdash^{[\mathcal{M}]\mathrm{NLC}} M' : (a\,a')\cdot s]\!] \blacktriangleright m'$, $[\![\nabla^{\#a}\vdash^{[\mathcal{M}]\mathrm{NLC}} M'' : (a\,a')\cdot s]\!] \blacktriangleright m''$ and $d \in [\![\nabla]\!]$. Then, by definition, we have that for some fresh c

$$
\begin{aligned}
[\![\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}}\langle\langle a'\rangle\rangle M' : [a]s]\!] &\overset{\mathrm{def}}{=} \langle a'\rangle((a\,c)\cdot m')(d)\\
[\![\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}}\langle\langle a''\rangle\rangle M'' : [a]s]\!] &\overset{\mathrm{def}}{=} \langle a''\rangle((a\,c)\cdot m'')(d)
\end{aligned}
$$

Next, by definition, we choose a fresh name b and demonstrate that

$$(a'\,b)\cdot((a\,c)\cdot m')(d) = (a''\,b)\cdot((a\,c)\cdot m'')(d)$$

$$\begin{aligned}
&(a'\,b)\cdot((a\,c)\cdot m')(d)\\
&= (a'\,b)\cdot(a\,c)\cdot m'((a\,c)\cdot d)\\
&= (a\,c)\cdot(a'\,b)\cdot m'((a\,c)\cdot d)\\
&= (a\,c)\cdot(a'\,b)\cdot[\![\nabla^{\#a}\vdash^{[\mathcal{M}]\mathrm{NLC}} M' : (a\,a')\cdot s]\!]((a\,c)\cdot d)\\
&= (a\,c)\cdot(a'\,b)\cdot[\![\nabla^{\#a,b}\vdash^{[\mathcal{M}]\mathrm{NLC}} M' : (a\,a')\cdot s]\!]((a\,c)\cdot d) && (\text{in } \mathit{FMSet},\ b\,\#\,(a\,c)\cdot d)\\
&= (a\,c)\cdot((a'\,b)_{(-)}\circ[\![\nabla^{\#a,b}\vdash^{[\mathcal{M}]\mathrm{NLC}} M' : (a\,a')\cdot s]\!])((a\,c)\cdot d) && (\text{in } \mathit{FMSet})\\
&= (a\,c)\cdot[\![\nabla^{\#a,b}\vdash^{[\mathcal{M}]\mathrm{NLC}} (a'\,b) * M' : (a'\,b)\cdot(a\,a')\cdot s]\!]((a\,c)\cdot d) && (\text{Lemma 5.2.3 (iii)})\\
&= (a\,c)\cdot[\![\nabla^{\#a,b}\vdash^{[\mathcal{M}]\mathrm{NLC}} (a'\,b) * M' : (a\,b)\cdot(a'\,b)\cdot s]\!]((a\,c)\cdot d)\\
&= (a\,c)\cdot[\![\nabla^{\#a,b}\vdash^{[\mathcal{M}]\mathrm{NLC}} (a'\,b) * M' : (a\,b)\cdot s]\!]((a\,c)\cdot d) && (a', b\,\#\,s)\\
&= (a\,c)\cdot[\![\nabla^{\#a,b}\vdash^{[\mathcal{M}]\mathrm{NLC}} (a''\,b) * M'' : (a\,b)\cdot s]\!]((a\,c)\cdot d) && (\text{induction})\\
&= \ldots && (\text{reverse reasoning})\\
&= (a''\,b)\cdot((a\,c)\cdot m'')(d)
\end{aligned}$$

(CCD): Suppose $\nabla^{\#a}\vdash^{[\mathcal{M}]\mathrm{NLC}} F\ @\ b \approx F'\ @\ b : (a\,b)\cdot s$. From this we can deduce that $\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} b\,\#\,F : [a]s$, $\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} F\approx F' : [a]s$ and $a\,\#\,\nabla$. The equational part is deduced as follows: Let $d\in[\![\nabla]\!]^{\#a}$.

$$\begin{aligned}
[\![\nabla^{\#a}\vdash^{[\mathcal{M}]\mathrm{NLC}} F\ @\ b : (a\,b)\cdot s]\!](d) &\overset{\mathrm{def}}{=} [\![\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} b\,\#\,F : [a]s]\!](d)\ @\ b\\
&\overset{\mathrm{def}}{=} [\![\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} F : [a]s]\!](d)\ @\ b && (\text{in } \mathit{FMSet})\\
&= [\![\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} F' : [a]s]\!](d)\ @\ b && (\text{induction})\\
&\overset{\mathrm{def}}{=} [\![\nabla\vdash^{[\mathcal{M}]\mathrm{NLC}} b\,\#\,F' : [a]s]\!](d)\ @\ b && (\text{in } \mathit{FMSet})\\
&\overset{\mathrm{def}}{=} [\![\nabla^{\#a}\vdash^{[\mathcal{M}]\mathrm{NLC}} F'\ @\ b : (a\,b)\cdot s]\!](d)
\end{aligned}$$

(LFANR): The equational part is deduced as follows: Let $d\in[\![\nabla]\!]^{\#a}$. So, by

definition, we have that $a \# d$.

$$
\begin{aligned}
&[\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} \mathsf{fr}\, a.\, M : s]\!](d) \\
&\stackrel{\text{def}}{=} ((a\, a') \cdot [\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} a \# M : s]\!])(d) && \text{(fresh name } a') \\
&\stackrel{\text{def}}{=} (a\, a') \cdot [[\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} a \# M : s]\!]((a\, a') \cdot d)] \\
&\stackrel{\text{def}}{=} (a\, a') \cdot [[\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} a \# M : s]\!](d)] && (a, a' \# d) \\
&= [\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} a \# M : s]\!](d) && (a, a' \# [\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} a \# M : s]\!](d)) \\
&\stackrel{\text{def}}{=} [\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} M : s]\!](d)
\end{aligned}
$$

(LFANS): Suppose $\nabla \vdash^{[\mathcal{M}]\mathrm{NLC}} \mathsf{fr}\, a.\, \mathsf{fr}\, a'.\, M \approx \mathsf{fr}\, a'.\, \mathsf{fr}\, a.\, M : s$. From this we can deduce that $\nabla^{\#a,a'} \vdash^{[\mathcal{M}]\mathrm{NLC}} \{a, a'\} \# M : s$ and $a, a' \# \nabla$. We now prove the equational part: Let $d \in [\![\nabla]\!]$

$$
\begin{aligned}
&[\![\nabla \vdash^{[\mathcal{M}]\mathrm{NLC}} \mathsf{fr}\, a.\, \mathsf{fr}\, a'.\, M : s]\!](d) \\
&\stackrel{\text{def}}{=} ((a\, a'') \cdot [\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} a \# \mathsf{fr}\, a'.\, M : s]\!])(d) && \text{(fresh name } a'') \\
&= (a\, a'') \cdot [\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} a \# \mathsf{fr}\, a'.\, M : s]\!]((a\, a'') \cdot d) \\
&\stackrel{\text{def}}{=} (a\, a'') \cdot [\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} \mathsf{fr}\, a'.\, M : s]\!]^*((a\, a'') \cdot d) \\
&\stackrel{\text{def}}{=} (a\, a'') \cdot [\![\nabla^{\#a} \vdash^{[\mathcal{M}]\mathrm{NLC}} \mathsf{fr}\, a'.\, M : s]\!]((a\, a'') \cdot d) && \text{(in } \mathit{FMSet}) \\
&\stackrel{\text{def}}{=} (a\, a'') \cdot [((a'\, a''') \cdot [\![\nabla^{\#a,a'} \vdash^{[\mathcal{M}]\mathrm{NLC}} a' \# M : s]\!])((a\, a'') \cdot d)] \\
&= (a\, a'') \cdot (a'\, a''') \cdot [\![\nabla^{\#a,a'} \vdash^{[\mathcal{M}]\mathrm{NLC}} a' \# M : s]\!]((a'\, a''') \cdot (a\, a'') \cdot d) \\
&\stackrel{\text{def}}{=} (a\, a'') \cdot (a'\, a''') \cdot [\![\nabla^{\#a,a'} \vdash^{[\mathcal{M}]\mathrm{NLC}} M : s]\!]^*((a'\, a''') \cdot (a\, a'') \cdot d) \\
&= (a\, a'') \cdot (a'\, a''') \cdot [\![\nabla^{\#a,a'} \vdash^{[\mathcal{M}]\mathrm{NLC}} M : s]\!]((a'\, a''') \cdot (a\, a'') \cdot d) && \text{(in } \mathit{FMSet}) \\
&= (a'\, a''') \cdot (a\, a'') \cdot [\![\nabla^{\#a',a} \vdash^{[\mathcal{M}]\mathrm{NLC}} M : s]\!]((a\, a'') \cdot (a'\, a''') \cdot d) && \text{(distinct } a, a', a'', a''') \\
&= \ldots \\
&= [\![\nabla \vdash^{[\mathcal{M}]\mathrm{NLC}} \mathsf{fr}\, a'.\, \mathsf{fr}\, a.\, M : s]\!](d)
\end{aligned}
$$

(LFANL): Suppose that $\nabla \vdash^{[\mathcal{M}]\mathrm{NLC}} \mathsf{fr}\, a'.\, \lambda^{\overline{a}} x : s.\, M \approx \lambda^{\overline{a}} x : s.\, \mathsf{fr}\, a'.\, M : s^{\overline{a}} \Rightarrow s'$. From this we can deduce that $\nabla^{\#a'}, \{a'\} \cup \overline{a} \# x : s \vdash^{[\mathcal{M}]\mathrm{NLC}} a' \# M : s'$ and $a' \notin \overline{a}$. We prove

the equational part: Let $d \in [\![\nabla]\!]$ and $d' \in [\![\nabla]\!]^{\#\overline{a}}$.

$$
\begin{aligned}
&((([\![\nabla \vdash^{[\mathsf{И}]\mathrm{NLC}} \mathsf{fr}\, a'.\, \lambda^{\overline{a}} x : s.\, M : s^{\overline{a}} \Rightarrow s']\!])(d))(d') \\
&\stackrel{\text{def}}{=} ([(a'\,b) \cdot [\![\nabla^{\#a'} \vdash^{[\mathsf{И}]\mathrm{NLC}} a' \,\#\, \lambda^{\overline{a}} x : s.\, M : s^{\overline{a}} \Rightarrow s']\!]](d))(d') && \text{(fresh name } b\text{)} \\
&= ((a'\,b) \cdot [[\![\nabla^{\#a'} \vdash^{[\mathsf{И}]\mathrm{NLC}} a' \,\#\, \lambda^{\overline{a}} x : s.\, M : s^{\overline{a}} \Rightarrow s']\!]((a'\,b) \cdot d)](d') \\
&= (a'\,b) \cdot ([[\![\nabla^{\#a'} \vdash^{[\mathsf{И}]\mathrm{NLC}} a' \,\#\, \lambda^{\overline{a}} x : s.\, M : s^{\overline{a}} \Rightarrow s']\!]((a'\,b) \cdot d)]((a'\,b) \cdot d')) \\
&\stackrel{\text{def}}{=} (a'\,b) \cdot ([[\![\nabla^{\#a'} \vdash^{[\mathsf{И}]\mathrm{NLC}} \lambda^{\overline{a}} x : s.\, M : s^{\overline{a}} \Rightarrow s']\!]((a'\,b) \cdot d)]((a'\,b) \cdot d')) && \text{(in } \mathit{FMSet}\text{)} \\
&\stackrel{\text{def}}{=} (a'\,b) \cdot ([\lambda([\![\nabla^{\#a'}, \overline{a} \,\#\, x : s \vdash^{[\mathsf{И}]\mathrm{NLC}} M : s']\!])((a'\,b) \cdot d)]((a'\,b) \cdot d')) \\
&\stackrel{\text{def}}{=} (a'\,b) \cdot ([\![\nabla^{\#a'}, \overline{a} \,\#\, x : s \vdash^{[\mathsf{И}]\mathrm{NLC}} M : s']\!]((a'\,b) \cdot d, (a'\,b) \cdot d')] && \text{(in } \mathit{FMSet}\text{)} \\
&= (a'\,b) \cdot [[\![\nabla^{\#a'}, \overline{a} \,\#\, x : s \vdash^{[\mathsf{И}]\mathrm{NLC}} M : s']\!]((a'\,b) \cdot d, i^{a'}_{[\![s]\!]}((a'\,b) \cdot d'))] \\
&= (a'\,b) \cdot [[\![\nabla^{\#a'}, \overline{a} \cup \{a'\} \,\#\, x : s \vdash^{[\mathsf{И}]\mathrm{NLC}} M : s']\!]((a'\,b) \cdot d, (a'\,b) \cdot d')] \\
&= (a'\,b) \cdot [[\![\nabla^{\#a'}, \overline{a} \cup \{a'\} \,\#\, x : s \vdash^{[\mathsf{И}]\mathrm{NLC}} a' \,\#\, M : s']\!]((a'\,b) \cdot d, (a'\,b) \cdot d')] \\
&= ((a'\,b) \cdot [\![\nabla^{\#a'}, \overline{a} \cup \{a'\} \,\#\, x : s \vdash^{[\mathsf{И}]\mathrm{NLC}} a' \,\#\, M : s']\!])(d, d') \\
&\stackrel{\text{def}}{=} [\![\nabla, \overline{a} \,\#\, x : s \vdash^{[\mathsf{И}]\mathrm{NLC}} \mathsf{fr}\, a'.\, M : s']\!](d, d') \\
&= (\lambda([\![\nabla, \overline{a} \,\#\, x : s \vdash^{[\mathsf{И}]\mathrm{NLC}} \mathsf{fr}\, a'.\, M : s']\!])(d))(d') && \text{(in } \mathit{FMSet}\text{)} \\
&\stackrel{\text{def}}{=} ((([\![\nabla \vdash^{[\mathsf{И}]\mathrm{NLC}} \lambda^{\overline{a}} x : s.\, \mathsf{fr}\, a'.\, M : s^{\overline{a}} \Rightarrow s']\!])(d))(d')
\end{aligned}
$$

□

5.3.4 [И]FM-cartesian Closed Categories

We have shown, as an intermediate step, that our categorical semantics for [И]NLC (in *FMSet*) is sound and moreover it closely corresponds to the model-theoretic semantics (in *FMSet*). The essential question is now, if the categorical semantics that we have just introduced is also complete and if a categorical type theory correspondence for [И]NLC can be obtained. As a step towards answering this question we have to abstract away from *FMSet* and introduce a category with the following structures:

(i) For any FM-category $\mathcal{C}$, there is a family of categories $(\mathcal{C}^{\#\overline{a}} \mid \overline{a} \subseteq_{fin} \mathbb{A})$ where

$ob\,\mathcal{C}^{\#\overline{a}}$ consists of those $C \in ob\,\mathcal{C}$ for which $\overline{a} \# C$. Given such $C, C' \in ob\,\mathcal{C}^{\#\overline{a}}$, $f : C \to C' \in mor\,\mathcal{C}$ is a morphism in $\mathcal{C}^{\#\overline{a}}$ just in case $\overline{a} \# f$. The properties of fresh inclusions ensure that each $\mathcal{C}^{\#\overline{a}}$ is indeed a category, and moreover that for each $a \in \mathbb{A}$ and $\overline{b} \subseteq_{fin} \mathbb{A}$ $(a \notin \overline{b})$ there is a functor $(-)^{\#a} : \mathcal{C}^{\#\overline{b},a} \to \mathcal{C}^{\#\overline{b}}$ as shown by Clouston [13] (see Lemma 7.2.8).

(ii) We then require that each such functor $(-)^{\#a} : \mathcal{C}^{\#\overline{b},a} \to \mathcal{C}^{\#\overline{b}}$ has a right adjoint $[a](-) : \mathcal{C}^{\#\overline{b}} \to \mathcal{C}^{\#\overline{b},a}$, which is moreover an adjoint equivalence. Further, $[a](-)$ needs to preserve finite products and $[a]X^{\#b} \cong [a](X^{\#b})$ for $a \neq b$.

(iii) Further, there is a family of morphisms $conc_b : ([a]C)^{\#b} \to (a\,b) \cdot C$.

(iv) We also require that these structures satisfy the commutativity properties which are needed in order to soundly model the equations (see Table 5.2) for the categorical semantics (see Table 5.4).

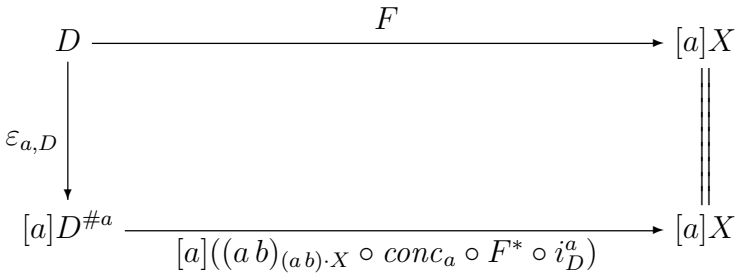
For now, we have decided to incorporate soundness in the definition of the structure, until a more "elegant" structure can be provided. Thus, there is one commutative diagram for each equational rule and its determined by unravelling the categorical semantic for the left and right side of the equations of [$\mathcal{N}$]NLC. For *example*, we have for every $D \in ob\,\mathcal{C}^{\#a}$, $X \in ob\,\mathcal{C}$, and $a', b \# X$, where $\varepsilon_{a,D} : D \to [a]D^{\#a}$ is the counit of the adjunction, the following commutative diagrams for (BND) and (END).

$$\begin{array}{ccccc} D^{\#a} & \xrightarrow{m} & (a\,a') \cdot X & \xrightarrow{(a'\,b)_{(a\,a')\cdot X}} & (a\,b) \cdot X \\ \Big\downarrow{\scriptstyle i} & & & & \Big\uparrow{\scriptstyle conc_b} \\ D & & \xrightarrow[F^*]{} & & ([a]X)^{\#b} \end{array}$$

with F being the morphism

$$D \xrightarrow{\varepsilon_{a,D}} [a]D^{\#a} \xrightarrow{[a]m} [a](a\,a') \cdot X \xrightarrow{[a](a\,a')_{(a\,a')\cdot X}} [a]X$$

Further



where

$$D^{\#a} \xrightarrow{i^a_D} D \xrightarrow{F^*} ([a]X)^{\#b} \xrightarrow{conc_b} (a\,b)\cdot X \xrightarrow{(a\,b)_{(a\,b)\cdot X}} X$$

The other commutative diagrams follow analogously. We call an FM-cccs with these additional structures a **[И]FM-cccs**. Hence, as shown in the previous section, we have that *FMSet* is an [И]FM-ccc.

5.3.5 Construction of Exponentials via [И]NLC

We recall that the starting point of this section was the observation that NLC is not expressive enough to construct an exponential for a syntactically generated (classifying) category $Cl(Th)$. This lead to the introduction of [И]NLC, which is semantically motivated by an "alternative" exponential in *FMSet*. To close the circle, we will now demonstrate that [И]NLC is expressive enough to construct an equivariant exponential in $Cl(Th)$. This is an important step towards obtaining a syntactically generated classifying [И]FMccc. However, to ultimately obtain a classifying category, the remaining properties of a [И]FM-ccc still have to be proven, which is work in progress that will not be part of this thesis.

We take the syntactic classifying category (term quotient structure) that was introduced by Clouston [17] as our starting point:

Objects: $ob\ Cl(Th)$ is defined to be the set of freshness contexts. We make use of

the following typical objects

$$\nabla \stackrel{\text{def}}{=} (\overline{a}_1 \# x_1 : s_1, ..., \overline{a}_n \# x_n : s_n)$$
$$\nabla' \stackrel{\text{def}}{=} (\overline{a}'_1 \# x'_1 : s'_1, ..., \overline{a}'_m \# x'_m : s'_m)$$
$$\nabla'' \stackrel{\text{def}}{=} (\overline{a}''_1 \# x''_1 : s''_1, ..., \overline{a}''_k \# x''_k : s''_k)$$

Morphisms: *mor* $Cl(Th)$ is defined to be the set of morphisms

$$\delta \stackrel{\text{def}}{=} ([M_1]_\approx, \ldots, [M_m]_\approx) : \nabla \to \nabla',$$

which are lists of typed expressions such that for $1 \leq i \leq m$ we have $Th \rhd \nabla \vdash^{[\mathcal{M}]\text{NLC}} \overline{a}'_i \# M_i : s'_i$, and $[M_i]_\approx$ is the equivalence class of those T such that $Th \rhd \nabla \vdash^{[\mathcal{M}]\text{NLC}} M_i \approx T : s'_i$.

Identity: For ∇, as defined above, the identity morphism is given by $([x_1]_\approx, \ldots, [x_n]_\approx)$.

Composition: Let $\delta : \nabla \to \nabla'$ be as defined above and let $\delta' \stackrel{\text{def}}{=} ([N_1]_\approx \ldots [N_k]_\approx) : \nabla' \to \nabla''$. Then

$$\delta' \circ \delta \stackrel{\text{def}}{=} ([N_1\{\vec{M}_i/\vec{x}_i\}]_\approx, \ldots, [N_k\{\vec{M}_i/\vec{x}_i\}]_\approx)$$

Finitely supported internal permutation action: For ∇ the internal permutation action π_∇ is given by $([\pi x_1]_\approx, \ldots, [\pi x_n]_\approx) : \nabla \to \pi \cdot \nabla$. By applying the conjugation permutation action on a morphism δ (see above) we obtain

$$\pi \cdot ([M_1]_\approx, \ldots, [M_m]_\approx) = ([\pi \cdot M_1]_\approx, \ldots, [\pi \cdot M_m]_\approx) : \pi \cdot \nabla \to \pi \cdot \nabla'$$

Equivariant finite products: The terminal object in $Cl(Th)$ is the empty freshness context. For ∇ and ∇', the product is defined by concatination of environments:

$$\nabla \times \nabla' \stackrel{\text{def}}{=} (\overline{a}_1 \# x_1 : s_1, ..., \overline{a}_n \# x_n : s_n, \overline{a}'_1 \# x'_1 : s'_1, ..., \overline{a}'_m \# x'_m : s'_m)$$

and the projection maps are defined as:

$$pr_1 \stackrel{\text{def}}{=} ([x_1]_\approx, \ldots, [x_n]_\approx) : \nabla \times \nabla' \to \nabla$$
$$pr_2 \stackrel{\text{def}}{=} ([x'_1]_\approx, \ldots, [x'_m]_\approx) : \nabla \times \nabla' \to \nabla'$$

Fresh-inclusion: For ∇ the fresh inclusion is defined as follows:

$$i_{\nabla}^{\overline{a}} \stackrel{\text{def}}{=} ([x_1]_{\approx}, \ldots, [x_n]_{\approx}) : \nabla^{\#\overline{a}} \to \nabla$$

Exponential: Due to the fact that the computations involved in the verification of the exponential are rather tedious and very lengthy, we believe that it is more instructive to first verify it for the case $\nabla_i \stackrel{\text{def}}{=} (a_i \# x_i : s_i)$ $(i = 1, 2, 3)$. This argument can later be generalised without problems. So, let's consider $\nabla_1 \Rightarrow \nabla_2$. As previously discussed, one would imagine that an exponential has the following form:

$$(\emptyset \# f : {s_1}^{a_1} \Rightarrow {s_2}^{a_2})$$

which is not a legitimate type in [И]NLC. In order to construct an exponential, using the typing system of [И]NLC, we mimic the isomorphisms observed in *FMSet*:

$$\nabla_1 \Rightarrow \nabla_2 \stackrel{\text{def}}{=} (a_2 \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2)$$

The evaluation map $ev : (\nabla_1 \Rightarrow \nabla_2) \times \nabla_1 \to \nabla_2$ is defined to be

$$ev \stackrel{\text{def}}{=} ([\mathsf{fr}\, b.\, (b\, a_2) * f(\langle\langle b\rangle\rangle (b\, a_2) x_1)]_{\approx})$$

Further, for any $\delta \stackrel{\text{def}}{=} ([M]_{\approx}) : \nabla_3 \times \nabla_1 \to \nabla_2$ the exponential mate $\lambda(\delta) : \nabla_3 \to \nabla_1 \Rightarrow \nabla_2$ is given by

$$\lambda(\delta) \stackrel{\text{def}}{=} ([\lambda^{a_1} z : [a_2]s_1.\, \mathsf{fr}\, b.\, (b\, a_2) * M\{(b\, a_2) * (z \mathbin{@} b)/x_1\}]_{\approx})$$

We now demonstrate that the previously defined structures constitute an exponential in $Cl(Th)$.

Verification of the Exponential Object

For the construction of an exponential object for $(a_1 \# x_1 : s_1)$ and $(a_2 \# x_2 : s_2)$, we may assume without loss of generality that $a_2 \# s_1$. If this were not the case,

we could choose a fresh name b and permute $(a_1 \# x_1 : s_1)$ with $(b\,a_2)$ to obtain $(a_1 \# x_1 : (b\,a_2) \cdot s_1)$, which is isomorphic to $(a_1 \# x_1 : s_1)$ via the morphism $([(b\,a_2)]_\approx) : \nabla_1 \to (a_1 \# x_1 : (b\,a_2) \cdot s_1)$ in $Cl(Th)$. So, as previously defined, we have $\nabla_1 \Rightarrow \nabla_2 \stackrel{\text{def}}{=} (a_2 \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2)$, which is clearly a freshness context and therefore an object of $Cl(Th)$.

Verification of the Evaluation Map

We next demonstrate that the evaluation map $ev : (\nabla_1 \Rightarrow \nabla_2) \times \nabla_1 \to \nabla_2$, which is defined as $ev \stackrel{\text{def}}{=} ([\mathsf{fr}\, b.\, (b\,a_2) * f(\langle\langle b\rangle\rangle(b\,a_2)x_1)]_\approx)$, is a morphism in $Cl(Th)$. Hence, we have to show that

$$(\nabla_1 \Rightarrow \nabla_2) \times \nabla_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} a_2 \# \mathsf{fr}\, b.\, (b\,a_2) * f(\langle\langle b\rangle\rangle(b\,a_2)x_1) : s_2$$

holds. We first check if the typing assertion is correct and later prove the freshness assertion, referred to as ($\diamond$1). So, we need to show that

$$\{a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{a_1\} \# x_1 : s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} \mathsf{fr}\, b.\, (b\,a_2) * f(\langle\langle b\rangle\rangle(b\,a_2)x_1) : s_2$$

To apply rule (LFAN), we need $b \# (\{a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{a_1\} \# x_1 : s_1)$, which is satisfied by using α-equivalence. Hence, we have to demonstrate that

$$\{b, a_2\} \# f : [a_2]{s_1}^{a_1} \Rightarrow s_2, \{b, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} b \# (b\,a_2) * f(\langle\langle b\rangle\rangle(b\,a_2)x_1) : s_2$$

Again, we first check if the typing assertion is correct and later prove the freshness assertion, referred to as ($\diamond$2). By applying Lemma 3.2.5 and considering that $b, a_2 \# s_2$, we can equally prove that

$$\{b, a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{b, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} f(\langle\langle b\rangle\rangle(b\,a_2)x_1) : s_2$$

By rule (SP) we immediately get $\{b, a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{b, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} f : ([a_2]s_1)^{a_1} \Rightarrow s_2$. Note that the previous typing assertion follows directly by (AP) once we have shown that

$$\{b, a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{b, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} a_1 \# \langle\langle b\rangle\rangle(b\,a_2)x_1 : [a_2]s_1$$

Yet again, we first check if the typing assertion is correct and later prove the freshness assertion, referred to as ($\diamond$3). Given that $a_2 \# s_1$, we can choose a fresh name a_2' and α-convert $[a_2]s_1$ to $[a_2']s_1$. Hence, we can equally show that

$$\{b, a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{b, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{N}]\text{NLC}} \langle\langle b\rangle\rangle(b\,a_2)x_1 : [a_2']s_1$$

Given that $b \# [a_2']s_1$ and $a_2' \# (\{b, a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{b, a_1\} \# x_1 : s_1, (b\,a_2)x_1)$, we can apply rule (NABSD) and show that

$$\{a_2', b, a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{a_2', b, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{N}]\text{NLC}} (b\,a_2)x_1 : (a_2'\,b) \cdot s_1$$

which follows immediately by (SP) and the fact that $b, a_2 \# s_1$. What remains to be shown are the three freshness conditions ($\diamond$1), ($\diamond$2) and ($\diamond$3). We begin with freshness assertion ($\diamond$3):

$$\{b, a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{b, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{N}]\text{NLC}} a_1 \# \langle\langle b\rangle\rangle(b\,a_2)x_1 : [a_2]s_1$$

By definition, we choose a fresh name c and prove that

$$\{c, b, a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{c, b, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{N}]\text{NLC}} (a_1\,c) * \langle\langle b\rangle\rangle(b\,a_2)x_1 \approx \langle\langle b\rangle\rangle(b\,a_2)x_1 : [a_2]s_1$$

Given that $b \# (a_1, c)$ and by the derived rule (CNA) we can equally show that

$$\{c, b, a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{c, b, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{N}]\text{NLC}} (a_1\,c)(b\,a_2)x_1 \approx (b\,a_2) : s_1$$

We can now compute the following

$$ds((b\,a_2), (a_1\,c)(b\,a_2)) = supp((b\,a_2)(a_1\,c)(b\,a_2))$$
$$= supp((a_1\,c)(b\,a_2)(b\,a_2)) = supp((a_1\,c)) = \{a_1, c\}$$

and therefore (SUSP) can be applied to complete the freshness assertion argument ($\diamond$3). We continue with the freshness assertion ($\diamond$2):

$$\{b, a_2\} \# f : [a_2]{s_1}^{a_1} \Rightarrow s_2, \{b, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{N}]\text{NLC}} b \# (b\,a_2) * f(\langle\langle b\rangle\rangle(b\,a_2)x_1) : s_2$$

Note that by Corollary 3.2.10 we can equally show that

$$\{b, a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{b, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{N}]\mathrm{NLC}} a_2 \# f(\langle\langle b\rangle\rangle(b\,a_2)x_1) : s_2$$

Given that $\{b, a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{b, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{N}]\mathrm{NLC}} a_2 \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2$ trivially follows, it suffices to show that $\{b, a_2\} \# f : [a_2]{s_1}^{a_1} \Rightarrow s_2, \{b, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{N}]\mathrm{NLC}} a_2 \# \langle\langle b\rangle\rangle(b\,a_2)x_1 : [a_2]s_1$. So, by definition, we need to demonstrate that for a fresh name c we can obtain

$$\{c, b, a_2\} \# f : [a_2]{s_1}^{a_1} \Rightarrow s_2, \{c, b, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{N}]\mathrm{NLC}} (a_2\,c) * \langle\langle b\rangle\rangle(b\,a_2)x_1 \approx \langle\langle b\rangle\rangle(b\,a_2)x_1 : [a_2]s_1$$

Given that $b \# (a_2, c)$ and by the derived rule (SNA) we can equally show

$$\{b, a_2\} \# f : [a_2]{s_1}^{a_1} \Rightarrow s_2, \{b, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{N}]\mathrm{NLC}} (a_2\,c)(b\,a_2)x_1 \approx (b\,a_2)x_1 : [a_2]s_1$$

which follows directly by (SUSP) given that $ds((b\,a_2), (a_2\,c)(b\,a_2)) = supp((b\,c)) = \{b, c\}$ and b, c are in the freshness context for x_1. We conclude by showing that the freshness assertion ($\diamond 1$) holds:

$$\{a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{a_1\} \# x_1 : s_1 \vdash^{[\mathcal{N}]\mathrm{NLC}} a_2 \# \mathsf{fr}\, b.\, (b\,a_2) * f(\langle\langle b\rangle\rangle(b\,a_2)x_1) : s_2$$

So, by definition, we have to show that for some fresh name c

$$\{c, a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{c, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{N}]\mathrm{NLC}} (c\,a_2) * \mathsf{fr}\, b.\, (b\,a_2) * f(\langle\langle b\rangle\rangle(b\,a_2)x_1) \approx \mathsf{fr}\, b.\, (b\,a_2) * f(\langle\langle b\rangle\rangle(b\,a_2)x_1) : s_2$$

Then, by definition of the object level permutation action on $\mathsf{fr}\, b.\, (-)$ and rule (LFAN), we need to show that

$$\{b, c, a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2, \{b, c, a_1\} \# x_1 : s_1 \vdash^{[\mathcal{N}]\mathrm{NLC}} (c\,a_2) * (b\,a_2) * f(\langle\langle b\rangle\rangle(b\,a_2)x_1) \approx (b\,a_2) * f(\langle\langle b\rangle\rangle(b\,a_2)x_1) : s_2$$

By definition of the object level permutation action we have

$$(c\,a_2) * (b\,a_2) * f\ (\langle\langle b\rangle\rangle(b\,a_2)x_1) \stackrel{\text{def}}{=} ((c\,a_2)(b\,a_2)f)\ (\langle\langle c\rangle\rangle(c\,a_2)x_1)$$
$$(b\,a_2) * f\ (\langle\langle b\rangle\rangle(b\,a_2)x_1) \stackrel{\text{def}}{=} ((b\,a_2)f)\ (\langle\langle a_2\rangle\rangle x_1)$$

Further, we can compute

$$ds(((b\,a_2),(c\,a_2)(b\,a_2)) = supp((b\,a_2)(c\,a_2)(b\,a_2)) = supp((b\,c)) = \{b,c\}$$

and given that $\{b,c\}$ is in the freshness context for variable f, we can use rule (SUSP) to obtain

$$\{b,c,a_2\} \# f : ([a_2]s_1)^{a_1} \Rightarrow s_2 \vdash^{[\mathcal{M}]\text{NLC}} (c\,a_2)(b\,a_2)f \approx (b\,a_2)f : ([b]s_1)^{a_1} \Rightarrow s_2$$

Next, we demonstrate that

$$\{b,c,a_1\} \# x_1 : s_1 \vdash^{[\mathcal{M}]\text{NLC}} \langle\langle c\rangle\rangle(c\,a_2)x_1 \approx \langle\langle a_2\rangle\rangle x_1 : [b]s_1$$

Then, by rule (BNABSD), we have to show that for a fresh name c' we have

$$\{c',b,c,a_1\} \# x_1 : s_1 \vdash^{[\mathcal{M}]\text{NLC}} (c'\,c)(c\,a_2)x_1 \approx (c'\,a_2)x_1 : s_1$$

We now compute that

$$ds((c'\,a_2),(c\,c')(c\,a_2)) = supp((c'\,a_2)(c\,c')(c\,a_2)) = supp((c\,c')) = \{c,c'\}$$

Given that $\{c,c'\}$ is in the freshness context for x_1, the equational judgement follows directly by (SUSP). To complete the freshness assertion argument, we apply (WEAK) and (CA). Thus, we have shown that ev is a morphism in $Cl(Th)$.

Verification of the Exponential Mate

We turn towards demonstrating that the exponential mate is a morphism in $Cl(Th)$. Suppose $\delta \stackrel{\text{def}}{=} ([M]_{\approx}) : \nabla_3 \times \nabla_1 \to \nabla_2$. We recall that the exponential mate is defined as follows:

$$\lambda(\delta) \stackrel{\text{def}}{=} ([\lambda^{a_1} z : [a_2]s_1.\,\mathsf{fr}\,b.\,(b\,a_2) * M\{(b\,a_2) * (z \mathbin{@} b)/x_1\}]_{\approx}) : \nabla_3 \to (\nabla_1 \Rightarrow \nabla_2)$$

Hence, we have to demonstrate that

$$\nabla_3 \vdash^{[\mathcal{M}]\mathrm{NLC}} a_2 \# \lambda^{a_1} z : [a_2]s_1.\, \mathsf{fr}\, b.\, (b\, a_2) * M\{(b\, a_2) * (z \mathbin{@} b)/x_1\} : ([a_2]s_1)^{a_1} \Rightarrow s_2$$

holds. We first check if the typing is correct and later prove the freshness assertion, referred to as ($\diamond 4$). By rule (ABS) we have to show that

$$\{a_3\} \# x_3 : s_3, \{a_1\} \# z : [a_2]s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} \mathsf{fr}\, b.\, (b\, a_2) * M\{(b\, a_2) * (z \mathbin{@} b)/x_1\} : s_2$$

Given that $\mathsf{fr}\, b.\, (-)$ is a binder, we have that $b \# (\{a_3\} \# x_3 : s_3, \{a_1\} \# z : [a_2]s_1)$ (up to α-equivalence). Hence, we can apply rule (LFAN) and show that

$$\{b, a_3\} \# x_3 : s_3, \{b, a_1\} \# z : [a_2]s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} b \# (b\, a_2) * M\{(b\, a_2) * (z \mathbin{@} b)/x_1\} : s_2$$

We postpone the freshness assertion, referred to as ($\diamond 5$), and continue with the typing assertion. Using Lemma 3.2.5 and considering that $b, a_2 \# s_2$, we can equally show that

$$\{b, a_3\} \# x_3 : s_3, \{b, a_1\} \# z : [a_2]s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} M\{(b\, a_2) * (z \mathbin{@} b)/x_1\} : s_2$$

By assumption and the application of rule (WEAK) we obtain that

$$\{b, a_3\} \# x_3 : s_3, \{b, a_1\} \# z : [a_2]s_1, \{a_1\} \# x_1 : s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} M : s_2$$

To be able to apply Lemma 3.2.20 and complete the typing assertion, it remains to be proven that

$$\{a_1, b\} \# z : [a_2]s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} a_1 \# (b\, a_2) * (z \mathbin{@} b) : s_1$$

Again, we postpone the freshness assertion, referred to as ($\diamond 6$), until later. Using Lemma 3.2.5 and considering that $b, a_2 \# s_1$, we can equally demonstrate that

$$\{a_1, b\} \# z : [a_2]s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} z \mathbin{@} b : s_1$$

Given that $a_2 \# (z \mathbin{@} b, a_1, b, [a_2]s_1)$, we can use the rule (AE) and show that

$$\{a_2, a_1, b\} \# z : [a_2]s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} z \mathbin{@} b : s_1$$

holds instead. Using rule (CONCD) and considering that $(a_2\, b) \cdot s_1 = s_1$ we have to show that

$$\{a_1, b\} \# z : [a_2]s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} b \# z : [a_2]s_1$$

which follows directly by (SUSP). We now turn towards the freshness assertions ($\diamond$4), ($\diamond$5) and ($\diamond$6). We begin with freshness assertion ($\diamond$6)

$$\{a_1, b\} \# z : [a_2]s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} a_1 \# (b\, a_2) * (z \mathbin{@} b) : s_1$$

By definition of freshness assertions (with a fresh name c) and the definition of the object level permutation action we have to show that

$$\{c, a_1, b\} \# z : [a_2]s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} (a_1\, c)(b\, a_2)z \mathbin{@} a_2 \approx (b\, a_2)z \mathbin{@} \ : a_2$$

which follows directly from (CCD) and (SUSP), taking into account that

$$ds((b\, a_2), (a_1\, c)(b\, a_2)) = supp((a_1\, c)) = \{a_1, c\}$$

and both names are in the freshness context for variable z. We continue with freshness assertion ($\diamond$5):

$$\{a_3, b\} \# x_3 : s_3, \{a_1, b\} \# z : [a_2]s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} b \# (b\, a_2) * M\{(b\, a_2) * (z \mathbin{@} b)/x_1\} : s_2$$

We have by assumption that

$$\{a_3\} \# x_3 : s_3, \{a_1\} \# x_1 : s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} a_2 \# M : s_2$$

From this and the fact that ($\diamond$6) holds, we can deduce by the derived rule (SUBST)

$$\{a_3\} \# x_3 : s_3 \vdash^{[\mathcal{M}]\mathrm{NLC}} a_2 \# M\{(b\, a_2) * (z \mathbin{@} b)/x_1\} : s_2$$

Then, by rule (WEAK) and Corollary 3.2.10 (using permutation $(b\, a_2)$), we are done. What remains to be shown is freshness assertion ($\diamond$4):

$$\nabla_3 \vdash^{[\mathcal{M}]\mathrm{NLC}} a_2 \# \lambda^{a_1} z : [a_2]s_1.\, \mathsf{fr}\, b.\, (b\, a_2) * M\{(b\, a_2) * (z \mathbin{@} b)/x_1\} : ([a_2]s_1)^{a_1} \Rightarrow s_2$$

By definition we have to show that for a fresh name c we have

$$\{c, a_3\} \# x_3 : s_3 \vdash^{[\mathcal{N}]\mathrm{NLC}} (c\,a_2) * \lambda^{a_1} z : [a_2]s_1.\,\mathsf{fr}\,b.\,(b\,a_2) * M\{(b\,a_2) * (z \mathbin{@} b)/x_1\}$$
$$\approx \lambda^{a_1} z : [a_2]s_1.\,\mathsf{fr}\,b.\,(b\,a_2) * M\{(b\,a_2) * (z \mathbin{@} b)/x_1\} : ([a_2]s_1)^{a_1} \Rightarrow s_2$$

We first apply the object level permutation action

$$(c\,a_2) * \lambda^{a_1} z : [a_2]s_1.\,\mathsf{fr}\,b.\,(b\,a_2) * M\{(b\,a_2) * (z \mathbin{@} b)/x_1\}$$
$$\stackrel{\text{def}}{=} \lambda^{a_1} z : [a_2]s_1.\,\mathsf{fr}\,b.\,(c\,a_2) * (b\,a_2) * M\{(b\,a_2) * (z \mathbin{@} b)/x_1\}\{(c\,a_2)z/z\}$$

before we use rules (CL) and (LFAN). Hence, we need to show that

$$\{b, c, a_3\} \# x_3 : s_3, \{b, a_1\} \# z : [a_2]s_1 \vdash^{[\mathcal{N}]\mathrm{NLC}}$$
$$(c\,a_2) * (b\,a_2) * M\{(b\,a_2) * (z \mathbin{@} b)/x_1\}\{(c\,a_2)z/z\}$$
$$\approx (b\,a_2) * M\{(b\,a_2) * (z \mathbin{@} b)/x_1\} : s_2$$

We can now deduce the following

$$\begin{aligned}
&(c\,a_2) * (b\,a_2) * M\{(b\,a_2) * (z \mathbin{@} b)/x_1\}\{(c\,a_2)z/z\} \\
&= (c\,a_2) * (b\,a_2) * M\{(c\,a_2)z/z\}\{(b\,a_2) * ((c\,a_2)z \mathbin{@} b)/x_1\} \\
&= (c\,a_2) * (b\,a_2) * M\{(b\,a_2) * ((c\,a_2)z \mathbin{@} b)/x_1\} && (z \notin fv(M)) \\
&= (c\,a_2) * (b\,a_2) * M\{(b\,a_2)(c\,a_2)z \mathbin{@} a_2/x_1\} \\
&= (c\,a_2)(b\,a_2) * M\{((c\,a_2)(b\,a_2))^{-1} z \mathbin{@} a_2/x_1\} \\
&= (c\,a_2)(b\,a_2) * (M\{z \mathbin{@} a_2/x_1\})\{((c\,a_2)(b\,a_2))^{-1} z/z\} \\
&= (c\,a_2)(b\,a_2) * (M\{z \mathbin{@} a_2/x_1\})\{((c\,a_2)(b\,a_2))^{-1} z/z\} \\
&\quad \{((c\,a_2)(b\,a_2))x_3/x_3\}\{((c\,a_2)(b\,a_2))^{-1} x_3/x_3\} \\
&= (c\,a_2)(b\,a_2) * (M\{z \mathbin{@} a_2/x_1\}\{((c\,a_2)(b\,a_2))x_3/x_3\}) \\
&\quad \{((c\,a_2)(b\,a_2))^{-1} z/z\}\{((c\,a_2)(b\,a_2))^{-1} x_3/x_3\} \\
&= (c\,a_2)(b\,a_2) \cdot (M\{z \mathbin{@} a_2/x_1\}\{((c\,a_2)(b\,a_2))x_3/x_3\}) && (\text{Lemma } 3.1.12) \\
&= (b\,a_2)(c\,b) \cdot (M\{z \mathbin{@} a_2/x_1\}\{((c\,a_2)(b\,a_2))x_3/x_3\})
\end{aligned}$$

$$
\begin{aligned}
&= (b\,a_2) \cdot (((c\,b) \cdot M)\{z \mathbin{@} a_2/x_1\}\{(c\,b) \cdot ((c\,a_2)(b\,a_2))x_3/x_3\}) && \text{(Proposition 3.1.15)}\\
&= (b\,a_2) \cdot (M\{z \mathbin{@} a_2/x_1\}\{(b\,a_2)(c\,a_2)x_3/x_3\}) && (b, c \# M)\\
&= (b\,a_2) * (M\{z \mathbin{@} a_2/x_1\}\{(b\,a_2)(c\,a_2)x_3/x_3\})\\
&\quad \{(b\,a_2)z/z\}\{(b\,a_2)x_3/x_3\} && \text{(Lemma 3.1.12)}\\
&= (b\,a_2) * M\{(b\,a_2)z \mathbin{@} a_2/x_1\}\{(b\,a_2)(c\,a_2)(b\,a_2)x_3/x_3\}\\
&= (b\,a_2) * M\{(b\,a_2) * (z \mathbin{@} b)/x_1\}\{(b\,c)x_3/x_3\}
\end{aligned}
$$

Let's suppose that the suspension of x_3 in M is τ. We then have that $ds(\tau, \tau(b\,c)) = \{b, c\}$ $(b, c \# \tau)$ and given that $\{b, c\}$ are in the freshness context for variable x_3, we are done by rule (SUSP). Hence, we have shown that the exponential mate is a morphism in $Cl(Th)$.

Verification of the Universal Property

The uniqueness of the exponential mate follows immediately by construction of the classifying category as a term quotient structure. What remains to be shown is that the following equation holds:

$$ev \circ (\lambda(([M]_{\approx})) \times id_{\nabla_1}) = ([M]_{\approx})$$

We recall the definitions of ev and $\lambda(-)$:

$$
\begin{aligned}
ev &\stackrel{\text{def}}{=} ([\mathsf{fr}\, b_1.\, (b_1\,a_2) * f(\langle\langle b_1\rangle\rangle(b_1\,a_2)x_1)]_{\approx})\\
\lambda(([M]_{\approx})) &\stackrel{\text{def}}{=} ([\lambda^{a_1} z : [a_2]s_1.\, \mathsf{fr}\, b.\, (b\,a_2) * M\{(b\,a_2) * (z \mathbin{@} b)/x_1\}]_{\approx})
\end{aligned}
$$

For brevity, we omit to explicitly mention the congruence rules used to obtain

various of the following equations.

$$
\begin{aligned}
& ev \circ (\lambda([M]_{\approx})) \times id_{\nabla_1} \\
\stackrel{\text{def}}{=} & ([\mathsf{fr}\, b.\, (b\, a_2) * f(\langle\langle b\rangle\rangle(b\, a_2)x_1)]_{\approx}) \circ \\
& (([\lambda^{a_1} z : [a_2]s_1.\, \mathsf{fr}\, b'.\, (b'\, a_2) * M\{(b'\, a_2) * (z \mathbin{@} b')/x_1\}]_{\approx}) \times ([x_1]_{\approx})) \\
\stackrel{\text{def}}{=} & ([\mathsf{fr}\, b.\, (b\, a_2) * [\lambda^{a_1} z : [a_2]s_1.\, \mathsf{fr}\, b'.\, (b'\, a_2) * M\{(b'\, a_2) * (z \mathbin{@} b')/x_1\}] \\
& (\langle\langle b\rangle\rangle(b\, a_2)x_1)]_{\approx}) \\
\stackrel{\text{def}}{=} & ([\mathsf{fr}\, b.\, (b\, a_2) * [\mathsf{fr}\, b'.\, (b'\, a_2) * M\{(b'\, a_2) * ((\langle\langle b\rangle\rangle(b\, a_2)x_1) \mathbin{@} b')/x_1\}]]_{\approx}) && (\mathsf{B},\ z \notin fv(M)) \\
= & ([\mathsf{fr}\, b.\, (b\, a_2) * [\mathsf{fr}\, b'.\, (b'\, a_2) * M\{(b'\, a_2) * ((b\, b') * (b\, a_2)x_1)/x_1\}]]_{\approx}) && (\mathsf{BND}) \\
= & ([\mathsf{fr}\, b.\, (b\, a_2) * [\mathsf{fr}\, b'.\, (b'\, a_2) * M\{(b\, b')x_1/x_1\}]]_{\approx}) \\
= & ([\mathsf{fr}\, b.\, (b\, a_2) * [\mathsf{fr}\, b'.\, (b'\, a_2) * M]]_{\approx}) && (\mathsf{SUSP}, \text{ for } x_1) \\
\stackrel{\text{def}}{=} & ([\mathsf{fr}\, b.\, \mathsf{fr}\, b'.\, (b\, a_2) * (b'\, a_2) * M]_{\approx}) \\
= & ([\mathsf{fr}\, b.\, \mathsf{fr}\, b'.\, (b'\, a_2) * (b\, b') * M]_{\approx}) \\
= & ([\mathsf{fr}\, b.\, \mathsf{fr}\, b'.\, (b'\, a_2) * M]_{\approx}) && (\mathsf{PERM}) \\
= & ([\mathsf{fr}\, b.\, \mathsf{fr}\, b'.\, M]_{\approx}) && (\diamond) \\
= & ([M]_{\approx}) && (\text{rule } \mathsf{LFANR})
\end{aligned}
$$

The equation $(\diamond)$ follows from our assumption that

$$a_3 \# x_3 : s_3, a_1 \# x_1 : s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} a_2 \# M : s_2$$

which by definition gives us the equation

$$\{b', a_3\} \# x_3 : s_3, \{b', a_1\} \# x_1 : s_1 \vdash^{[\mathcal{M}]\mathrm{NLC}} (b'\, a_2) * M \approx M : s_2$$

Hence, we have shown that the classifying category $Cl(Th)$ is cartesian closed.

Verification of Equivariant Exponential

Given the definition of the evaluation map ev

$$ev_{\nabla_1,\nabla_2} : \nabla_1 \Rightarrow \nabla_2 \times \nabla_1 \to \nabla_2$$
$$ev_{\nabla_1,\nabla_2} \stackrel{\text{def}}{=} ([\mathsf{fr}\, b_1.\, (b_1\, a_2) * f(\langle\langle b_1\rangle\rangle(b_1\, a_2)x_1)]_{\approx})$$

we have to demonstrate the following:

$$\pi \cdot ev_{\nabla_1,\nabla_2} = ev_{\pi\cdot\nabla_1,\pi\cdot\nabla_2}$$

Given that $Cl(Th)$ has internal permutation actions we have

$$\pi \cdot ev_{\nabla_1,\nabla_2} : \pi \cdot (\nabla_1 \Rightarrow \nabla_2 \times \nabla_1) \to \pi \cdot \nabla_2 = \pi \cdot \nabla_1 \Rightarrow \pi \cdot \nabla_2 \times \pi \cdot \nabla_1 \to \pi \cdot \nabla_2$$
$$\pi \cdot ev_{\nabla_1,\nabla_2} = ([\pi \cdot (\mathsf{fr}\, b_1.\, (b_1\, a_2) * f(\langle\langle b_1\rangle\rangle(b_1\, a_2)x_1))]_{\approx})$$

We can then deduce by definition that

$$ev_{\pi\cdot\nabla_1,\pi\cdot\nabla_2} : \pi \cdot \nabla_1 \Rightarrow \pi \cdot \nabla_2 \times \pi \cdot \nabla_1 \to \pi \cdot \nabla_2$$
$$ev_{\pi\cdot\nabla_1,\pi\cdot\nabla_2} \stackrel{\text{def}}{=} ([\mathsf{fr}\, b_1.\, (b_1\, \pi(a_2)) * f(\langle\langle b_1\rangle\rangle(b_1\, \pi(a_2))x_1)]_{\approx})$$

and demonstrate that both morphisms are equal:

$$\begin{aligned}
&\pi \cdot ev_{\nabla_1,\nabla_2} \\
&= ([\pi \cdot (\mathsf{fr}\, b_1.\, (b_1\, a_2) * f(\langle\langle b_1\rangle\rangle(b_1\, a_2)x_1))]_{\approx}) \\
&\stackrel{\text{def}}{=} ([\mathsf{fr}\, \pi(b_1).\, (\pi(b_1)\, \pi(a_2)) * f(\langle\langle \pi(b_1)\rangle\rangle(\pi(b_1)\, \pi(a_2))x_1)]_{\approx}) \\
&= ([\mathsf{fr}\, b_1.\, (b_1\, \pi(a_2)) * f(\langle\langle b_1\rangle\rangle(b_1\, \pi(a_2))x_1)]_{\approx}) \qquad (\alpha\text{-equivalent}) \\
&= ev_{\pi\cdot\nabla_1,\pi\cdot\nabla_2}
\end{aligned}$$

Remark 5.3.4 *We have shown that $Cl(Th)$ has equivariant exponentials and therefore $Cl(Th)$ is an FM-ccc. What remains to be proven is that $Cl(Th)$ is a $[\mathsf{M}]$FM-ccc, which is not part of this thesis. A detailed account of those constructions and the corresponding verifications is work in progress that we aim to publish as a journal version of [23].*

Chapter 6

Completeness of $\beta\eta$-conversion for Full Nominal Hierarchies in $\lambda^{\rightarrow}$

Our interest in this chapter is on completeness theorems for typed lambda calculi in the context of the nominal set model. To our knowledge, a "nominal" environment model (Henkin model) or a completeness theorem for a nominal analogue of full set-theoretic hierarchies has not yet been studied for such calculi. An interesting attempt in this direction is the work of Gabbay and Mulligan [37], but their notion of a nominal Henkin-style model is not an environment model in the usual sense, because function types are not interpreted as function spaces. Moreover, their completeness result only refers to the existence of a model that satisfies β-conversion (a term quotient model).

We recall that completeness of $\beta\eta$-conversion in the ordinary simply typed lambda calculus ($\lambda^{\rightarrow}$) has been proven for full set-theoretic hierarchies [30], full continuous hierarchies and full recursive hierarchies [57, 48]. All these results have originally been obtained using a logical relation based proof technique. An alternative proof technique is used in Statman's 1-Section Theorem [64, 65], which provides a necessary and sufficient condition on the combinatorial structure of Henkin models to prove completeness of $\beta\eta$-conversion on terms typable in $\lambda^{\rightarrow}$.

A logical relation based proof technique is usually the first choice in proving a

completeness theorem. This is due to the fact that the proof argument is rather flexible and can conveniently be adapted to a new calculus. At first sight, this seems like the best approach to prove completeness theorems in a nominal context, but we need to be careful, because the proof argument relies on the axiom of choice which may result in complications. Hence, to proceed in this direction, we would have to find a way to circumvent these complications. In contrast, the 1-Section theorem and its underlying proof argument is rarely applied in the literature to prove completeness results, which is related to the more involved and inflexible proof argument. Still, it has an immediate advantage when it comes to nominal techniques, because the argument does not involve the axiom of choice.

Having introduced NLC in this thesis, it would be a natural starting point to investigate if $\beta\eta$-conversion is complete for nominal models in NLC, but as we have previously observed, NLC can be quite unwieldy at times due to the intertwined type and equation system. Furthermore, taking into consideration that SNTT and $\lambda\alpha\nu-$calculus use ordinary function types, we ultimately decided to first turn towards a more general question, namely if $\beta\eta$-conversion is complete for *nominal models in* $\lambda^{\to}$. Once this question is answered, it can be seen as a stepping stone towards completeness theorems for NLC, SNTT or the $\lambda\alpha\nu-$calculus in the future.

To sum it up, we introduce full nominal hierarchies, which are full hierarchies of finitely supported functions over nominal sets, and analyse if $\beta\eta$-conversion on terms typable in pure $\lambda^{\to}$ is complete for such models. At first, this may seem somewhat cautious, but it isolates rather well the main problem that we will encounter when we adapt the logical relation based proof technique. An immediate technical advantage of this intermediate goal is that we can directly use various preliminary results of $\lambda^{\to}$ which allows us to concentrate on the core problems of the completeness argument. Moreover, similar to the study of full recursive hierarchies or full continuous hierarchies, it is interesting in its own right to analyse if $\beta\eta$-conversion captures equality in full nominal hierarchies. We now clarify that this is not merely an immediate

consequence of the completeness result for full set-theoretic hierarchies:

We consider pure typed lambda terms M and N of type $(\gamma \Rightarrow \gamma) \Rightarrow \gamma$, where γ is a base type. We recall that the interpretations of M and N, which are total functions, can be shown to be equal in a full set-theoretic hierarchy by applying the extensionality property. Note that this also holds in case of a full nominal hierarchy. It may now be the case that the interpretations of M and N map precisely one element of the function space $[\![\gamma]\!] \Rightarrow [\![\gamma]\!]$ to different elements in $[\![\gamma]\!]$, and therefore by extensionality the interpretations of M and N are not equal. If we now take into account that the full nominal hierarchy is restricted to finitely supported functions, the domain for the interpretations of M and N, $[\![\gamma]\!] \Rightarrow_{fs} [\![\gamma]\!]$, may be smaller. Hence, the element in the domain that observed the inequality of the interpretation of M and N in the full set-theoretic model may not be an element of $[\![\gamma]\!] \Rightarrow_{fs} [\![\gamma]\!]$, and therefore the interpretations of M and N would actually be equal in the full nominal hierarchy. Thus, there may be additional equations which hold in full nominal hierarchies, but are not necessarily captured by $\beta\eta$-conversion.

The chapter is structured as follows. We first recall the ordinary typed lambda calculus and the notion of an environment model (Henkin model). We then introduce full nominal hierarchies and demonstrate that they are Henkin models. We continue by discussing how to adapt the logical relation based proof argument such that it can be applied to full nominal hierarchies and point out issues involving the axiom of choice. To circumvent this issue, we propose a modification of the proof argument, which relies on a stronger assumption. This provides us with a rather restricted completeness theorem, which would need to be further investigated in the future. Independently of this, we pursue an alternative route towards completeness by applying Statman's 1-Section theorem. This involves proving that the necessary and sufficient condition of the 1-section theorem holds for full nominal hierarchies. Given that full nominal hierarchies are Henkin models, it directly follows that $\beta\eta$-conversion is complete for full nominal hierarchies in $\lambda^{\rightarrow}$.

6.1 Preliminaries

We recall the basic definitions of the ordinary simply typed lambda calculus, as well as the notions of a typed applicative structure and an environment model (Henkin model). For full details we refer to [3] and [48], respectively.

Simply Typed Lambda Calculus ($\lambda^{\to}$)

Definition 6.1.1 (Syntax of λ-terms) *We introduce a λ-**signature**, $Sg = (\mathsf{B}, \mathsf{C})$, which consists of the following data:*

- *A set of base types B. The set of types T is generated by the BNF grammar $\sigma ::= \gamma \mid \sigma \Rightarrow \sigma$, where γ is a base type.*
- *A set C of typed constant symbols.*

*For an empty signature (C is empty), the corresponding simply typed lambda calculus is called **pure**. The set $Term_{Sg}$ of raw terms over a λ-signature Sg is generated as follows:*

$$M ::= x \mid c \mid \lambda x{:}\sigma.M \mid MM$$

where x ranges over the set V of variables and c ranges over a set C of typed constants. We denote the set of free variables of a term M by $\mathsf{fv}(M)$ and the set of constants of a term M as $\mathsf{const}(M)$. Terms are identified up to renaming of bound variables (Barendregt's Variable Convention). Capture avoiding substitution is denoted by $N[x := M]$.

Definition 6.1.2 ($\beta\eta$-reduction System) *The β and η rules are defined as follows:*

$$(\lambda x{:}\sigma.N)M \longrightarrow N[x := M] \quad (\beta\text{-rule})$$
$$\lambda x{:}\sigma.Mx \longrightarrow M, \text{ if } x \notin \mathsf{fv}(M) \quad (\eta\text{-rule})$$

Terms of the form $(\lambda x{:}\sigma.N)M$ are referred to as a ***β-redex*** *and terms of the form $\lambda x{:}\sigma.Mx$ as a* ***η-redex****. We say that M* ***$\beta\eta$-reduces to*** *N in one step, written as $\to_{\beta\eta}$, if N can be obtained by applying the β or η rule to a corresponding redex in M. Formally, the relation $\to_{\beta\eta}$ is defined as the smallest relation on P that is closed under contexts and the β and η-rules. The reflexive and transitive closure is denoted by $\twoheadrightarrow_{\beta\eta}$. The notion of reduction gives rise to a natural definition of equality modulo reduction. Two terms M and N are called* ***$\beta\eta$-convertible****, denoted by $M =_{\beta\eta} N$, if $M \equiv_\alpha N$, or there exists a term P with $P =_{\beta\eta} N$ and either $M \twoheadrightarrow_{\beta\eta} P$ or $P \twoheadrightarrow_{\beta\eta} M$. The* ***$\beta\eta$-normal form*** *of a term M is denoted by $NF(M)$.*

Due to the fact that pure $\lambda^{\to}$ is confluent and strongly normalising the following decision procedure for $\beta\eta$-conversion can be used:

$$M =_{\beta\eta} N \Longleftrightarrow NF(M) \equiv_\alpha NF(N)$$

Definition 6.1.3 (Typing System) *A* ***context*** *is a finite list of variable type pairs, usually written as $\Gamma = [x_1 : \sigma_1, ..., x_n : \sigma_n]$, where the variables are required to be distinct. A* ***term-in-context*** *is judgement of the form $\Gamma \vdash M{:}\sigma$ where Γ is a context, M is term and σ is a type. A term-in-context is called a* ***derived (proved) term****, if it is generated by the rules:*

$$\frac{x:\sigma \in \Gamma}{\Gamma \vdash x:\sigma}\ (\mathsf{var}) \qquad \frac{c:\sigma \in \mathsf{C}}{\Gamma \vdash c:\sigma}\ (\mathsf{const})$$

$$\frac{\Gamma, x:\sigma \vdash N{:}\tau}{\Gamma \vdash \lambda x{:}\sigma.N : \sigma \Rightarrow \tau}\ (\to I) \qquad \frac{\Gamma \vdash M{:}\sigma \Rightarrow \tau \qquad \Gamma \vdash N{:}\sigma}{\Gamma \vdash MN : \tau}\ (\to E)$$

Applicative Structures and Henkin Models

Definition 6.1.4 (Applicative Structure) *A* ***typed applicative structure*** *over a λ-signature Sg is a pair*

$$\mathcal{A} ::= \langle \{A^\sigma\}, \{\mathrm{app}^{\sigma,\tau}\} \rangle$$

where A^σ is a type indexed set and $\mathrm{app}^{\sigma,\tau}$ is a type indexed map $A^{\sigma \Rightarrow \tau} \to A^\sigma \to A^\tau$.

Definition 6.1.5 (Extensional Applicative Structure) *We say that a typed applicative structure is* ***extensional*** *if it satisfies the following property:*

$$\forall f, g \in A^{\sigma \Rightarrow \tau} \,.\, [(\forall d \in A^{\sigma} \,.\, \mathrm{app}^{\sigma,\tau}\ f\ d = \mathrm{app}^{\sigma,\tau}\ g\ d) \Longrightarrow f = g]$$

To state the completeness theorem we use a structure that already assumes extensionality, namely a typed frame.

Definition 6.1.6 *A* ***typed frame*** *is a typed applicative structure*

$$\mathcal{A} ::= \langle \{A^{\sigma}\}, \{\mathrm{app}^{\sigma,\tau}\} \rangle$$

such that $A^{\sigma \Rightarrow \tau} \subseteq A^{\sigma} \Rightarrow A^{\tau}$ *and* $\mathrm{app}^{\sigma,\tau}\ f\ d = f(d)$ *for any* $f \in A^{\sigma \Rightarrow \tau}$ *and* $d \in A^{\sigma}$.

Definition 6.1.7 *For a* λ*-signature* Sg*, typed applicative structure* $\mathcal{A}$ *and context* Γ*, we have the following definitions:*

- *A* ***constant assignment*** κ *is a partial function such that for* $\mathsf{c} \in \mathsf{C}$ *and* $\mathsf{c} : \sigma \in Sg$*, we have that* $\kappa(\mathsf{c}) \in A^{\sigma}$ *holds.*
- *An* ***environment*** η *is a partial function from the set of variables* V *to the disjoint union of all* A^{σ}*. For* $d \in A^{\sigma}$*, the* ***updated environment*** $\eta[x \mapsto d]$ *is the environment mapping* x *to* d *and all other variables* $y \neq x$ *to* $\eta(y)$*.*
- *We say that* η ***satisfies*** Γ*, denoted by* $\eta \models \Gamma$*, if* $\eta(x) \in A^{\sigma}$ *for every* $x : \sigma \in \Gamma$*.*
- *A* ***total interpretation function*** *for a typed applicative structure* $\mathcal{A}$ *maps a derivation* $\Gamma \vdash M : \sigma$*, constant assignment* κ *and environment* $\eta \models \Gamma$ *to an element* $[\![\Gamma \vdash M : \sigma]\!]^{\mathcal{A}}_{\kappa,\eta}$ *in* A^{σ}*.*

Definition 6.1.8 (Henkin models) *A pair* $\langle \mathcal{A}, \kappa \rangle$ *is called a* ***Henkin model*** *if* $\mathcal{A}$ *is an extensional typed applicative structure,* κ *is a constant assignment and there exists a total interpretation function satisfying the following condition, referred to as the* ***environment model condition****:*

(i) $[\![\Gamma \vdash c : \sigma]\!]^{\mathcal{A}}_{\kappa,\eta} = \kappa(c)$

(ii) $[\![\Gamma \vdash x : \sigma]\!]^{\mathcal{A}}_{\kappa,\eta} = \eta(x)$

(iii) $[\![\Gamma \vdash M\ N : \sigma]\!]^{\mathcal{A}}_{\kappa,\eta} = \mathrm{app}^{\sigma,\tau}\ [\![\Gamma \vdash M : \sigma \Rightarrow \tau]\!]^{\mathcal{A}}_{\kappa,\eta}\ [\![\Gamma \vdash N : \sigma]\!]^{\mathcal{A}}_{\kappa,\eta}$

(iv) $[\![\Gamma \vdash \lambda^{x:\sigma} M.\ : \sigma \Rightarrow \tau]\!]^{\mathcal{A}}_{\kappa,\eta} = f$, *where* f *is the unique* $f \in A^{\sigma\Rightarrow\tau}$ *such that* $\forall d \in A^{\sigma}.\mathrm{app}^{\sigma,\tau}\ f\ d = [\![\Gamma, x : \sigma \vdash M : \tau]\!]^{\mathcal{A}}_{\kappa,\eta[x\mapsto d]}$

We say that a Henkin model $\langle \mathcal{A}, \kappa \rangle$ **satisfies** an equation, denoted by $\langle \mathcal{A}, \kappa \rangle \models \Gamma \vdash M = N{:}\sigma$, if $\Gamma \vdash M{:}\sigma$, $\Gamma \vdash N{:}\sigma$ and $[\![\Gamma \vdash M : \sigma]\!]^{\mathcal{A}}_{\kappa,\eta} = [\![\Gamma \vdash N{:}\sigma]\!]^{\mathcal{A}}_{\kappa,\eta}$ for all η with $\eta \models \Gamma$. If Γ is empty then we omit the context and write $\langle \mathcal{A}, \kappa \rangle \models M = N{:}\sigma$. Similarly, if the signature is empty (no constants) we omit κ. We denote the set of equations satisfied in a Henkin model $\mathcal{A}$ by $Th(\mathcal{A})$.

The following technical lemma shows that the interpretation of a term M only depends on the elements assigned to the free variables of M in the environment and the constants that occur in M.

Lemma 6.1.9 *Let* $\langle \mathcal{A}, \kappa \rangle$ *be a Henkin model. If* $\Gamma \vdash M : \sigma$ *and* $\eta \models \Gamma$, *then*

$$[\![\Gamma \vdash M : \sigma]\!]^{\mathcal{A}}_{\kappa,\eta} = [\![\Gamma \vdash M : \sigma]\!]^{\mathcal{A}}_{\kappa^*,\eta^*}$$

where η^* *and* κ^* *are the maps* η *and* κ, *which are restricted to the set* $\mathsf{fv}(M)$ *and* $\mathsf{const}(M)$, *respectively.*

Lemma 6.1.10 (Soundness for Henkin Models) *Let* $\langle \mathcal{A}, \kappa \rangle$ *be a Henkin model.*

1. ***Type Soundness***. *If* $\Gamma \vdash M : \sigma$, *then* $[\![\Gamma \vdash M : \sigma]\!]^{\mathcal{A}}_{\kappa,\eta} \in A^{\sigma}$ *for all* $\eta \models \Gamma$.
2. ***Soundness of*** $\beta\eta$***-conversion***. *Let* $\Gamma \vdash M : \sigma$ *and* $\Gamma \vdash N : \sigma$. *If* $M =_{\beta\eta} N$ *then* $\langle \mathcal{A}, \kappa \rangle \models \Gamma \vdash M = N : \sigma$.

We now state the completeness theorem of $\beta\eta$-conversion for Henkin models in the pure $\lambda^{\to}$, which is formally defined as follows:

Definition 6.1.11 (Completeness for Henkin Models) *A Henkin model $\mathcal{A}$ is called **complete** for the pure theory of $\beta\eta$-conversion if for any context Γ, type σ and pure lambda terms M, M' of type σ, the following holds:*

$$\mathcal{A} \models \Gamma \vdash M = M' : \sigma \Longrightarrow M =_{\beta\eta} M'$$

Note that without loss of generality the definition of the completeness theorem can be restricted to closed terms. Further, an example of such a complete model for $\beta\eta$-conversion is the syntactically generated term quotient model $\mathcal{Q}$.

Theorem 6.1.12 (Completeness of the Term Quotient Model) *There exists a Henkin model (without empty sorts), referred to as $\mathcal{Q}$, that satisfies precisely the pure theory of $\beta\eta$-conversion.*

6.2 Full Nominal Hierarchy $\mathcal{N}$

We now formally introduce the full hierarchy of finitely supported functions over nominal sets and demonstrate that it is a Henkin model.

Definition 6.2.1 (Full Nominal Hierarchy) *Let Sg be a λ-signature and κ a constant assignment. The **full nominal hierarchy** over infinite nominal sets $\{X^\gamma \mid \gamma \in \mathsf{B}\}$, denoted by $\langle \mathcal{N}, \kappa \rangle$, is defined to be the typed frame*

$$N^\gamma \stackrel{\text{def}}{=} X^\gamma$$

$$N^{\sigma \Rightarrow \tau} \stackrel{\text{def}}{=} N^\sigma \Rightarrow_{fs} N^\tau$$

that is equipped with the following interpretation function, which is recursively defined on typing derivations for environments η and the constant assignment κ:

$$[\![\Gamma \vdash c : \sigma]\!]^{\mathcal{N}}_{\kappa,\eta} = \kappa(c)$$

$$[\![\Gamma \vdash x : \sigma]\!]^{\mathcal{N}}_{\kappa,\eta} = \eta(x)$$

$$[\![\Gamma \vdash M\ N : \sigma]\!]^{\mathcal{N}}_{\kappa,\eta} = \mathrm{app}^{\sigma,\tau}\ [\![\Gamma \vdash M : \sigma \Rightarrow \tau]\!]^{\mathcal{N}}_{\kappa,\eta}\ [\![\Gamma \vdash N : \sigma]\!]^{\mathcal{N}}_{\kappa,\eta}$$

$$[\![\Gamma \vdash \lambda x : \sigma . M : \sigma \Rightarrow \tau]\!]^{\mathcal{N}}_{\kappa,\eta} = \{(d, [\![\Gamma, x : \sigma \vdash M : \tau]\!]^{\mathcal{N}}_{\kappa,\eta[x \mapsto d]}) \mid d \in N^\sigma\}$$

Note that typed frames are defined to be families of *sets* and *not* families of nominal sets. However, this is not a problem, because the permutation actions are only used to construct the hierarchy and do not interfere with the actual functions contained in the sets. Hence, $\mathcal{N}$ is constructed using the nominal set structure, but the actual typed frame is defined on the underlying sets.

For the following auxiliary result, we equip the set of constant assignments and environments with a point-wise permutation action. Hence, both sets are $Perm(\mathbb{A})$-sets with $(\pi \cdot \kappa)(c) \stackrel{\text{def}}{=} \pi \cdot \kappa(c)$ and $(\pi \cdot \epsilon)(x) \stackrel{\text{def}}{=} \pi \cdot \epsilon(x)$.

Lemma 6.2.2 *Suppose that* $\Gamma \vdash M : \sigma$.

(i) $\pi \cdot [\![\Gamma \vdash M : \sigma]\!]^{\mathcal{N}}_{\kappa,\eta} = [\![\Gamma \vdash M : \sigma]\!]^{\mathcal{N}}_{\pi\cdot\kappa,\pi\cdot\eta}$.

(ii) $supp([\![\Gamma \vdash M : \sigma]\!]^{\mathcal{N}}_{\kappa,\eta}) \subseteq \bigcup_{c\in \mathsf{const}(M)} supp(\kappa(c)) \cup \bigcup_{x\in \mathsf{fv}(M)} supp(\eta(x))$

Proof (i) The first property is proved by induction on the structure of M:

- Case (M is x)

$$\begin{aligned} \pi \cdot [\![\Gamma, x : \sigma \vdash x : \sigma]\!]^{\mathcal{N}}_{\kappa,\eta} &\stackrel{\text{def}}{=} \pi \cdot \eta(x) \\ &\stackrel{\text{def}}{=} (\pi \cdot \eta)(x) \\ &\stackrel{\text{def}}{=} [\![\Gamma, x : \sigma \vdash M : \sigma]\!]^{\mathcal{N}}_{\pi\cdot\kappa,\pi\cdot\eta} \end{aligned}$$

- Case (M is c):

$$\begin{aligned} \pi \cdot [\![\Gamma \vdash c : \sigma]\!]^{\mathcal{N}}_{\kappa,\eta} &\stackrel{\text{def}}{=} \pi \cdot \kappa(c) \\ &\stackrel{\text{def}}{=} (\pi \cdot \kappa)(c) \\ &\stackrel{\text{def}}{=} [\![\Gamma \vdash c : \sigma]\!]^{\mathcal{N}}_{\pi\cdot\kappa,\pi\cdot\eta} \end{aligned}$$

- Case (M is $N\ N'$):

$$\begin{aligned}
&\pi \cdot [\![\Gamma \vdash N\ N' : \tau]\!]^{\mathcal{N}}_{\kappa,\eta} \\
&\stackrel{\text{def}}{=} \pi \cdot (\text{app}\ [\![\Gamma \vdash N : \sigma \Rightarrow \tau]\!]^{\mathcal{N}}_{\kappa,\eta}\ [\![\Gamma \vdash N' : \sigma]\!]^{\mathcal{N}}_{\kappa,\eta}) \\
&\stackrel{\text{def}}{=} \pi \cdot ((([\![\Gamma \vdash N : \sigma \Rightarrow \tau]\!]^{\mathcal{N}}_{\kappa,\eta})[\![\Gamma \vdash N' : \sigma]\!]^{\mathcal{N}}_{\kappa,\eta}) \\
&\stackrel{\text{def}}{=} (\pi \cdot [\![\Gamma \vdash N : \sigma \Rightarrow \tau]\!]^{\mathcal{N}}_{\kappa,\eta})\ \pi \cdot [\![\Gamma \vdash N' : \sigma]\!]^{\mathcal{N}}_{\kappa,\eta} \\
&\stackrel{\text{def}}{=} \text{app}\ \pi \cdot [\![\Gamma \vdash N : \sigma \Rightarrow \tau]\!]^{\mathcal{N}}_{\kappa,\eta}\ \pi \cdot [\![\Gamma \vdash N' : \sigma]\!]^{\mathcal{N}}_{\kappa,\eta} \\
&= \text{app}\ [\![\Gamma \vdash N : \sigma \Rightarrow \tau]\!]^{\mathcal{N}}_{\pi\cdot\kappa,\pi\cdot\eta}\ [\![\Gamma \vdash N' : \sigma]\!]^{\mathcal{N}}_{\kappa,\pi\cdot\eta} \qquad \text{(induction)} \\
&\stackrel{\text{def}}{=} [\![\Gamma \vdash N\ N' : \tau]\!]^{\mathcal{N}}_{\pi\cdot\kappa,\pi\cdot\eta}
\end{aligned}$$

- Case (M is $\lambda x : \sigma.\, N$): Let $d \in N^{\sigma}$. We can now deduce the following:

$$\begin{aligned}
&(\pi \cdot [\![\Gamma \vdash \lambda x : \sigma.\, N : \sigma \Rightarrow \tau]\!]^{\mathcal{N}}_{\kappa,\eta})(d) \\
&\stackrel{\text{def}}{=} \pi \cdot [\![\Gamma, x : \sigma \vdash N : \tau]\!]^{\mathcal{N}}_{\kappa,\eta[x\mapsto\pi^{-1}\cdot d]} \\
&= [\![\Gamma, x : \sigma \vdash N : \tau]\!]^{\mathcal{N}}_{\pi\cdot\kappa,\pi\cdot\eta[x\mapsto\pi\cdot\pi^{-1}\cdot d]} \qquad \text{(induction)} \\
&= [\![\Gamma, x : \sigma \vdash N : \tau]\!]^{\mathcal{N}}_{\pi\cdot\kappa,\pi\cdot\eta[x\mapsto d]} \\
&\stackrel{\text{def}}{=} [\![\Gamma \vdash \lambda x : \sigma.N : \sigma \Rightarrow \tau]\!]^{\mathcal{N}}_{\pi\cdot\kappa,\pi\cdot\eta}(d)
\end{aligned}$$

(ii) For the second property we reason as follows: By Lemma 6.1.9, we have that

$$supp([\![\Gamma \vdash M : \sigma]\!]^{\mathcal{N}}_{\kappa,\eta}) = supp([\![\Gamma \vdash M : \sigma]\!]^{\mathcal{N}}_{\kappa^*,\eta^*}) \quad (\diamond 1)$$

where η^* and κ^* are the maps η and κ restricted to the set $\mathsf{fv}(M)$ and $\mathsf{const}(M)$ respectively. From property (i) above, we know that $[\![\Gamma \vdash M : \sigma]\!]^{\mathcal{N}}_{(-,-)}$ is equivariant and therefore by Lemma 2.2.3 (i) we have that

$$supp([\![\Gamma \vdash M : \sigma]\!]^{\mathcal{N}}_{\kappa^*,\eta^*}) \subseteq supp(\kappa^*) \cup supp(\eta^*) \quad (\diamond 2)$$

Due to the fact that the set of environments and assignments are $Perm(\mathbb{A})$-sets with a point-wise permutation action, we have that:

$$supp(\kappa^*) = \bigcup_{c\in\mathsf{const}(M)} supp(\kappa^*(x)) = \bigcup_{c\in\mathsf{const}(M)} supp(\kappa(x)) \quad (\diamond 3)$$

$$supp(\eta^*) = \bigcup_{x \in fv(M)} supp(\eta^*(x)) = \bigcup_{x \in fv(M)} supp(\eta(x)) \quad (\diamond 4)$$

From ($\diamond$1), ($\diamond$2), ($\diamond$3) and ($\diamond$4), we get the following:

$supp([\![\Gamma \vdash M : \sigma]\!]^{\mathcal{N}}_{\kappa,\eta}) \subseteq \bigcup_{x \in fv(M)} supp(\eta(x)) \cup \bigcup_{c \in \mathsf{const}(M)} supp(\kappa(c))$

□

The previous lemma can now be used to demonstrate that $\mathcal{N}$ is a total interpretation function.

Lemma 6.2.3 *Let κ be a constant assignment and η an environment such that $\eta \models \Gamma$. If $\Gamma \vdash M : \sigma$ then $[\![\Gamma \vdash M : \sigma]\!]^{\mathcal{N}}_{\kappa,\eta} \in N^{\sigma}$.*

Proof The lemma follows by induction on derivations of $\Gamma \vdash M : \sigma$. We only consider the case for lambda abstraction. The other cases follow immediately. Suppose $\Gamma \vdash \lambda x : \sigma.M : \sigma \Rightarrow \tau$ and $\eta \models \Gamma$. From this we can deduce that $\Gamma, x : \sigma \vdash M : \tau$. Let $d \in N^{\sigma}$. We directly obtain that $\eta[x \mapsto d] \models \Gamma, x : \sigma$ and by induction $[\![\Gamma, x : \sigma \vdash M : \tau]\!]^{\mathcal{N}}_{\kappa,\eta} \in N^{\tau}$. Hence, we clearly have that

$$\{(d, [\![\Gamma, x : \sigma \vdash M : \tau]\!]^{\mathcal{N}}_{\kappa,\eta[x \mapsto d]}) \mid d \in N^{\sigma}\}$$

is a function from N^{σ} to N^{τ}. Given that $N^{\sigma \Rightarrow \tau} \stackrel{\text{def}}{=} N^{\sigma} \Rightarrow_{fs} N^{\tau}$, we have to show that it is finitely supported as well. This follows immediately from Lemma 6.2.2 (ii). Thus, we have that $[\![\Gamma \vdash \lambda x : \sigma.M : \sigma \to \tau]\!]^{\mathcal{N}}_{\kappa,\eta} \in N^{\sigma \Rightarrow \tau}$ □

Due to the fact that the interpretation function of $\mathcal{N}$ trivially satisfies the environment model condition, we can directly deduce that $\langle \mathcal{N}, \kappa \rangle$ is a Henkin model:

Theorem 6.2.4 *The full nominal hierarchy $\langle \mathcal{N}, \kappa \rangle$ is a Henkin model.*

6.3 Routes Towards Completeness for $\mathcal{N}$

In this section we pursue two routes towards a completeness theorem for $\mathcal{N}$. Note that in both cases we rely on the fact that $\mathcal{N}$ is a Henkin model. We further recall that the completeness theorem that we pursue is for pure $\lambda^{\rightarrow}$ (without constants) and to simplify matters, we further restrict to only one base type. Hence, we fix an empty signature Sg with one base type γ.

6.3.1 Completeness via Logical Relations

We begin with a well known proof technique, based on logical relations, which was used by Friedman [30][1] and Plotkin [57] to prove that the pure theory of $\beta\eta$-conversion is complete for full set-theoretic hierarchies. We recall that logical relations are a powerful tool in logic and semantics, which are used to establish links between syntax and semantics of a calculus. Here we are interested in how the proof technique can be used to prove completeness for full nominal hierarchies.

Definition 6.3.1 (Logical Relations) *For typed applicative structures $\mathcal{A}$ and $\mathcal{B}$, a* ***logical relation*** $\mathcal{R} = \{\mathrm{R}^{\sigma}\}$ *over $\mathcal{A}$ and $\mathcal{B}$ is defined as a family of type indexed relations which is recursively defined as follows:*

(i) $\mathrm{R}^{\sigma} \subseteq A^{\sigma} \times B^{\sigma}$

(ii) $\mathrm{R}^{\sigma\Rightarrow\tau}(f,g) \Longleftrightarrow \forall x \in A^{\sigma} \,.\, \forall y \in B^{\sigma} \,.\, [\mathrm{R}^{\sigma}(x,y) \Longrightarrow \mathrm{R}^{\tau}(\mathrm{app}^{\sigma,\tau}\ f\ x,\ \mathrm{app}^{\sigma,\tau}\ g\ y)]$

We present two auxiliary results that are important building blocks for the proof of completeness via logical relations:

Lemma 6.3.2 *[see Lemma 8.2.17 in [48]] If $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{B}$ is a logical partial function from Henkin model $\mathcal{A}$ to $\mathcal{B}$, then $Th(\mathcal{A}) \subseteq Th(\mathcal{B})$.*

[1] not yet called logical relations at that time.

Corollary 6.3.3 *Let $\mathcal{A}$ be a Henkin model and $\mathcal{B}$ a complete Henkin model of $\beta\eta$-conversion. If $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{B}$ is a logical partial function, then $Th(\mathcal{A}) = Th(\mathcal{B})$*

Proof Due to the fact that $\mathcal{B}$ is complete, we have that $Th(\mathcal{B})$ is precisely the theory of $\beta\eta$-conversion. By applying the soundness lemma on Henkin model $\mathcal{A}$ we deduce that $Th(\mathcal{B}) \subseteq Th(\mathcal{A})$ and therefore $Th(\mathcal{A}) = Th(\mathcal{B})$ by applying Lemma 6.3.2. □

This result provides us with a way to determine if a particular Henkin model is complete. We will now focus on the following two properties, namely if there exists a complete model $\mathcal{B}$ and a logical partial function $\mathcal{R}$.

We begin with the construction of a logical partial function $\mathcal{R}$ between two Henkin models, which is required to apply Lemma 6.3.2. We consider this as the main hurdle towards a completeness result for $\mathcal{N}$.

We first recall that for pure $\lambda^{\to}$ (without constants) a logical relation can directly be constructed by choosing an arbitrary relation $\mathrm{R}^{\gamma} \subseteq A^{\gamma} \times B^{\gamma}$ for base type γ and by extending it inductively to function types. To construct a logical relation $\mathcal{R}$ that is also a partial function, we can proceed as follows: We suppose that R^{γ} is a partial function and prove by induction that this property can be lifted to all type-indexed relations in $\mathcal{R}$, which is informally expressed as $\mathcal{R}$ is a partial function.

Note that to prove this property, applying the original proof argument for full set-theoretic hierarchies, we additionally require that $\mathcal{R}$ is surjective. Hence, we also need to assert that R^{γ} is surjective and prove that surjectivity can be lifted to all type-indexed relations in $\mathcal{R}$. The corresponding property is therefore sometimes called the "Surjective Lemma", and it is informally expressed as: if $\mathcal{B}$ is a typed frame and R^{γ} is a partial surjection, then $\mathcal{R} \subseteq \mathcal{N} \times \mathcal{B}$ is a logical partial surjection.

Moreover, as will become obvious later, we need to further restrict to typed frames $\mathcal{B}$ of finitely supported functions over a nominal set, also referred to as **nominal typed frames**, which are formally defined as typed frames over a nominal set with

$$A^{\sigma \Rightarrow \tau} \subseteq A^{\sigma} \Rightarrow_{fs} A^{\tau}$$

A first attempt of a "Surjective Lemma" for $\mathcal{N}$ can be stated as follows:

Attempt 6.3.4 (Surjective Lemma I) *Let $\mathcal{R} \subseteq \mathcal{N} \times \mathcal{B}$ be a logical relation over the full nominal hierarchy $\mathcal{N}$ and a nominal type frame $\mathcal{B}$. If $\mathrm{R}^{\gamma} \subseteq N^{\gamma} \times B^{\gamma}$ is a partial surjection, then $\mathcal{R}$ is a partial surjection.*

We recall that in the original proof argument, as pointed out above, it is required that $\mathcal{R}$ is surjective to ultimately show that $\mathcal{R}$ is a partial function. But to show that $\mathcal{R}$ is surjective in the first place, the axiom of choice is required, which is problematic in the context of $\mathcal{N}$. In particular, to prove that $\mathcal{R}$ surjective it has to be demonstrated (in the function type case of the inductive proof) that for any $g \in B^{\sigma\Rightarrow\tau}$ there exists a function $f \in N^{\sigma\Rightarrow\tau}$ such that $\mathrm{R}^{\sigma\Rightarrow\tau}(f, g)$.

Writing R^{σ} and R^{τ} as functions, it can immediately be observed that f can be any function such that for all $x \in \mathit{Dom}(\mathrm{R}^{\sigma})$ we have that $f(x)$ is an element of the non-empty set $(\mathrm{R}^{\tau})^{-1}(g(\mathrm{R}^{\sigma}x))$. Such functions obviously exists in the full set-theoretic hierarchy, but in the case of $\mathcal{N}$ we also have to demonstrate that such a function is finitely supported. Due to the fact that g is directly involved in the definition of f, this justifies why we have restricted the "Surjective Lemma" to nominal typed frames.

A direct way to prove that a function is finitely supported in the nominal set model would be to apply the "Finite Support Principle" ([56]): *Any function or relation that is defined from finitely supported functions and relations using higher-order classical logic without choice principles, is itself finitely supported.*

However, due to the fact that the construction of f involves the axiom of choice, this principle is not applicable in this case. So far, we did not succeed in proving or disproving that there exits such a finitely supported f for the provided version of a "Surjective Lemma".

An obvious way to proceed at this point is to further restrict the "Surjective Lemma". More precisely, we address this particular problem by adapting the "Sur-

jective Lemma" similarly to Mitchell's proof for full recursive models [48] (see Section 8.4.2). The key idea is to apply a stronger condition on R^{γ}, which allows us to directly construct the finitely supported functions that are required to prove surjectivity.

Before we can state the new "Surjective Lemma", we need to introduce some auxiliary notions: Let A and B be nominal sets. We call $\langle f, g\rangle$ a **finitely supported embedding** from B into A if $f : B \to A$ and $g : A \to B$ are finitely supported functions and $g \circ f = id_B$. Analogous to the proof for full recursive models, we use the concept of realizability in connection with a logical relation $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{B}$

- f^{σ} **realizes that** R^{σ} **is surjective** by computing $f^{\sigma}(b) \in A^{\sigma}$ with $\mathrm{R}^{s}(f^{\sigma}(b), b)$ for every element $b \in B^{\sigma}$.
- g^{σ} **realizes that** R^{σ} **is a partial function** by computing the unique $g^{\sigma}(a) = b \in B^{\sigma}$ with $\mathrm{R}^{\sigma}(a, b)$ for any $a \in dom(\mathrm{R}^{\sigma})$

We modify the notion of a logical partial surjection by equipping it with the notion of realizing functions: Let $\mathcal{A}$ and $\mathcal{B}$ be nominal type frames. We say that a family $\{\mathrm{R}^{\sigma}, f^{\sigma}, g^{\sigma}\}$ of relations and functions is a **finitely supported logical partial surjection** from $\mathcal{A}$ to $\mathcal{B}$ if $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{B}$ is a logical partial surjection, and functions $f^{\sigma} : B^{\sigma} \to A^{\sigma}$ and $g^{\sigma} : A^{\sigma} \to B^{\sigma}$ are finitely supported functions with f^{σ} realizing that R^{σ} is surjective and g^{σ} realizing that R^{σ} is a partial function.

Lemma 6.3.5 (Surjective Lemma II) *Let $\mathcal{B}$ be nominal type frames. If B^{γ} is finitely supported embedded in N^{γ}, then there exists a finitely supported partial surjection $\{\mathrm{R}^{\sigma}, f^{\sigma}, g^{\sigma}\}$ from $\mathcal{N}$ to $\mathcal{B}$.*

Proof Using the finitely supported embedding (f^{γ}, g^{γ}), with the finitely supported functions $f^{\gamma} : B^{\gamma} \to N^{\gamma}$ and $g^{\gamma} : N^{\gamma} \to B^{\gamma}$ for base type γ, we can define R^{γ} as follows:

$$\mathrm{R}^{\gamma}(x, y) \Longleftrightarrow g^{\gamma}(x) = y$$

Due to the fact that we are restricted to pure $\lambda^{\rightarrow}$, we can construct a logical relation $\mathcal{R} \subseteq \mathcal{N} \times \mathcal{B}$ by extending R^{γ} inductively to function types. We then demonstrate that by using the recursively defined realizing functions

$$f^{\sigma \Rightarrow \tau} \stackrel{\text{def}}{=} f^{\tau} \circ \_ \circ g^{\sigma}$$
$$g^{\sigma \Rightarrow \tau} \stackrel{\text{def}}{=} g^{\tau} \circ \_ \circ f^{\sigma}$$

we can prove that $\{\mathrm{R}^{\sigma}, f^{\sigma}, g^{\sigma}\}$ is a finitely supported logical partial surjection. We now prove by induction on types that the functions are indeed finitely supported realizing functions. From this we can then directly deduce that $\mathcal{R}$ is a partial surjection.

Base Case:

(i) Let $y \in B^{\gamma}$. We need to show that $\mathrm{R}^{\gamma}(f^{\gamma}(y), y)$, or equally, by definition of R^{γ} that $g^{\gamma}(f^{\gamma}(y)) = y$. Given that (f^{γ}, g^{γ}) is a finitely supported embedding, we have that $g^{\gamma} \circ f^{\gamma} = id$; thus we are done.

(ii) The fact that g^{γ} realizes that R^{γ} is a partial function follows directly by definition of R^{γ}.

Both realizing maps are finitely supported by definition.

Inductive Case:

(i) Suppose $h \in B^{\sigma \Rightarrow \tau}$. We have to demonstrate that $f^{\sigma \Rightarrow \tau}(h) \in N^{\sigma \Rightarrow \tau}$ and $\mathrm{R}^{\sigma \Rightarrow \tau}(f^{\sigma \Rightarrow \tau}(h), h)$. We clearly have that $f^{\tau} \circ h \circ g^{\sigma}$ is finitely supported and given that $\mathcal{N}$ contains all finitely supported functions, the first condition holds. For the second condition, we take into account that $\mathcal{R}$ is a logical relation. Hence, we can equally show that for all $y, y' \in N^{\sigma}$ such that $\mathrm{R}^{\sigma}(y, y')$ we can deduce that $\mathrm{R}^{\tau}((f^{\sigma \Rightarrow \tau}(h))(y), h(y'))$.

Suppose $y, y' \in N^\sigma$ and $\mathrm{R}^\sigma(y, y')$. By induction we have that g^σ realizes that R^σ is a partial function. So, given that $\mathrm{R}^\sigma(y, y')$ we can directly deduce that $g^\sigma(y) = y'$. Again, by induction, we know that f^τ realizes that R^τ is surjective. So, given that that $h(y') \in B^\tau$ we get $\mathrm{R}^\tau(f^\tau(h(y')), h(y'))$. Further, using the fact that $g^\sigma(y) = y'$, we obtain $\mathrm{R}^\tau(f^\tau(h(g^\sigma(y))), h(y'))$ and by definition of $f^{\sigma\Rightarrow\tau}$ we have $\mathrm{R}^\tau((f^{\sigma\Rightarrow\tau}(h))(y), h(y'))$.

(ii) Let $h \in dom(\mathrm{R}^{\sigma\Rightarrow\tau})$. We take any $h' \in B^{\sigma\Rightarrow\tau}$ such that $\mathrm{R}^{\sigma\Rightarrow\tau}(h, h')$ and demonstrate that $h' = g^{\sigma\Rightarrow\tau}(h)$. Let $y \in B^\sigma$. By induction f^σ realizes that R^σ is surjective. So, we have that $\mathrm{R}^\sigma(f^\sigma(y), y)$ and given that $\mathcal{R}$ is a logical relation we can deduce that $\mathrm{R}^\tau(h(f^\sigma(y)), h'(y))$. Further, by induction, g^τ realizes that R^τ is a partial function. So, we have that $\mathrm{R}^\tau(h(f^\sigma(y)), g^\tau(h(f^\sigma(y))))$ and by uniqueness we get $g^\tau(h(f^\sigma(y))) = h'(y)$. Given that y is any element, we have that $h' = g^{\sigma\Rightarrow\tau}(h)$.

By construction, we have that both realizing functions, $f^{\sigma\Rightarrow\tau}$ and $g^{\sigma\Rightarrow\tau}$, are finitely supported. This concludes the proof. □

Based on Lemma 6.3.5 we can now prove a completeness theorem for $\mathcal{N}$:

Theorem 6.3.6 *If $\mathcal{B}$ is a complete nominal typed frame and B^γ is finitely supported embedded in N^γ, then $\mathcal{N}$ is complete with respect to the theory of $\beta\eta$-conversion.*

Proof By Theorem 6.2.4 we have that $\mathcal{N}$ is a Henkin model and by Lemma 6.3.5 that $\mathcal{R} \subseteq \mathcal{N} \times \mathcal{B}$ is a logical partial function. Given that $\mathcal{B}$ is a complete Henkin model, we can apply Corollary 6.3.3 to obtain $Th(\mathcal{N}) = Th(\mathcal{B})$. □

What remains to be shown to apply the completeness theorem in a meaningful way is the existence of a complete nominal typed frame $\mathcal{B}$. The prime candidate would be the syntactically generated term quotient model $\mathcal{Q}$. For the the pure $\lambda^{\rightarrow}$, we can apply Theorem 6.1.12 to obtain that $\mathcal{Q}$ is a Henkin model. Next, we have to

show that $\mathcal{Q}$ can be viewed as a nominal type frame, i.e. there exits a nominal type frame $\mathcal{B}$ such that $\mathcal{B} \cong \mathcal{Q}$. In the case that we use the trivial permutation action, this follows similarly to the case for *Set*.

We can now directly deduce that $\mathcal{N}$ over $\mathbb{N}$ (equipped with the trivial permutation action) is complete. Given that B^{γ} is countable infinite, there exists a bijective map between B^{γ} and $\mathbb{N}$. Taking into account that both sets are equipped with a trivial permutation action, we can directly deduce that the map is equivariant and therefore we have a finitely supported embedding. This is a trivial example, which holds for all full nominal hierarchies $\mathcal{N}$ over an infinite discrete nominal set.

Considering more interesting examples, the limitations of the theorem become rather obvious. For example, we did not succeed in proving that B^{γ} is finitely supported embedded in $\mathbb{A}$. Thus, it is either the case that the trivial permutation action for $\mathcal{Q}$ is too weak or the finite support restriction for the "Surjective Lemma" is too strong to begin with. Further work will be necessary to answer this question in a satisfactory way.

6.3.2 Completeness via Statman's 1-Section Theorem

An alternative route towards a completeness theorem for full nominal hierarchies is based on Statman's 1-Section theorem [64]. The 1-Section theorem provides a necessary and sufficient condition on the combinatorial structure of Henkin models to prove completeness of $\beta\eta$-conversion. Informally, the condition is checking if a Henkin model has enough elements of a particular type to distinguish the interpretations of lambda terms that are not $\beta\eta$-equivalent.

We begin by recalling the 1-section theorem and show how it is applied to prove completeness for ordinary full set-theoretic hierarchies. We then turn towards full nominal hierarchies and demonstrate that the condition of the 1-section theorem holds. Given that the 1-section theorem can be applied on full nominal hierarchies over an infinite nominal set, it is a Henkin model by Lemma 6.2.4, we ultimately

obtained a completeness theorem of $\beta\eta$-conversion for full nominal hierarchies in $\lambda^{\rightarrow}$.

The 1-Section Theorem and its Applications

We introduce the 1-Section theorem as stated in [2] (for Henkin models instead of classes of Henkin models). For a more extensive account of the 1-section theorem we refer to [59].

Theorem 6.3.7 (1-Section for Models) *For any Henkin model $\mathcal{A}$ over the empty signature Sg, the following properties are equivalent:*

(i) For any type σ and any closed pure typed lambda terms M, N of type σ:

$$\mathcal{A}[\![M]\!] = \mathcal{A}[\![N]\!] \Rightarrow M =_{\beta\eta} N$$

(ii) For any closed pure typed lambda terms M, N of type $(\gamma \Rightarrow \gamma \Rightarrow \gamma) \Rightarrow \gamma \Rightarrow \gamma$

$$\mathcal{A}[\![M]\!] = \mathcal{A}[\![N]\!] \Rightarrow M =_{\beta\eta} N$$

Due to the fact that pure $\lambda^{\rightarrow}$ is confluent and strongly normalising, we have that $M =_{\beta\eta} N \iff NF(M) \equiv_{\alpha} NF(N)$. Further, given that $\mathcal{M}$ is a Henkin model, we can deduce from $M =_{\beta\eta} NF(M)$ and $N =_{\beta\eta} NF(N)$ that $\mathcal{A}[\![M]\!] = \mathcal{A}[\![NF(M)]\!]$ and $\mathcal{A}[\![N]\!] = \mathcal{A}[\![NF(N)]\!]$ by soundness. Thus, we can rephrase the condition of the 1-section theorem as follows: for any closed pure typed lambda terms M, N of type $(\gamma \Rightarrow \gamma \Rightarrow \gamma) \Rightarrow \gamma \Rightarrow \gamma$

$$\mathcal{A}[\![NF(M)]\!] = \mathcal{A}[\![NF(N)]\!] \Rightarrow NF(M) \equiv_{\alpha} NF(N)$$

We can immediately observe that for an empty signature, without constants, the closed terms of this type, in normal form, encode binary trees

$$t ::= \mathsf{c} \mid \mathsf{f}\ t\ t$$

which are constructed over a signature $\{\mathsf{f}, \mathsf{c}\}$, where $\mathsf{c} : \gamma$ and $\mathsf{f} : \gamma \Rightarrow \gamma \Rightarrow \gamma$. This observation is useful once we start to prove completeness via the 1-Section theorem, because the condition can be further simplified. We ultimately aim to prove the following condition (contrapositive):

$$(\forall t)(\forall t') \,.\, [t \not\equiv t' \implies (\exists \kappa)\ [\![t]\!]^{\mathcal{A}}_{\kappa,\emptyset} \neq [\![t']\!]^{\mathcal{A}}_{\kappa,\emptyset}]$$

where κ assigns elements to c and f. An immediate consequence of Statman's 1-Section theorem is that the full set-theoretic hierarchy over the natural numbers $\mathbb{N}$ is complete for $\beta\eta$-conversion. A result, which was originally proven by Friedman [30].

Corollary 6.3.8 *The full set-theoretic hierarchy $\mathcal{P}$ over the set of natural numbers $\mathbb{N}$ is complete for the pure theory of $\beta\eta$-conversion.*

Proof Given that $\mathcal{P}$ is a Henkin model, the 1-section theorem can be applied to determine if $\mathcal{P}$ over $\mathbb{N}$ is complete. As pointed out above, to prove completeness via the 1-section theorem, it suffices to show that for any binary trees t, t' such that $t \not\equiv t'$ there exist an interpretation for c and f such that the interpretations of t and t' are distinct.

In the case of $\mathcal{P}$ over $\mathbb{N}$ we can even prove a stronger property, namely that there exists a interpretation for c and f such that for any binary trees t, t' with $t \not\equiv t'$ we have that the interpretation of t and t' are distinct. From this, the above property follows immediately.

We choose an injective pairing function $p : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ such that 0 is not in the image of p. We can now take the curried version of p, denoted by $\lambda(p)$, as the interpretation of f and 0 as the interpretation of c. So, we clearly have that $\lambda(p)$ is an element of $\mathbb{N} \Rightarrow \mathbb{N} \Rightarrow \mathbb{N}$ and $0 \in \mathbb{N}$. Hence, $\kappa \stackrel{\text{def}}{=} \{(\mathsf{c}, 0), (\mathsf{f}, \lambda(p))\}$ is well defined and we can now deduce by induction on the structure of binary tree t that the interpretation of t and t' are distinct.

- Take $\mathsf{c} \not\equiv \mathsf{f}\ t_1\ t_2$. Given that 0 is not in the image of p, we can directly deduce that $[\![\mathsf{c}]\!]^{\mathcal{P}}_{\kappa,\emptyset} \stackrel{\text{def}}{=} 0 \neq p([\![t_1]\!]^{\mathcal{P}}_{\kappa,\emptyset}, [\![t_2]\!]^{\mathcal{P}}_{\kappa,\emptyset}) \stackrel{\text{def}}{=} (\lambda(p)([\![t_1]\!]^{\mathcal{P}}_{\kappa,}))([\![t_2]\!]^{\mathcal{P}}_{\kappa,\emptyset}) \stackrel{\text{def}}{=} [\![\mathsf{f}\ t_1\ t_2]\!]^{\mathcal{P}}_{\kappa,\emptyset}$.

- Suppose $\mathsf{f}\ t_1\ t_2 \not\equiv \mathsf{f}\ t'_1\ t'_2$. W.l.o.g. we assume that $t_1 \not\equiv t'_1$. By induction and injectivity of p, we can deduce

$$\begin{aligned}
[\![\mathsf{f}\ t_1\ t_2]\!]^{\mathcal{P}}_{\kappa,\emptyset} &\stackrel{\text{def}}{=} ([\![\mathsf{f}]\!]^{\mathcal{P}}_{\kappa,\emptyset}([\![t_1]\!]^{\mathcal{P}}_{\kappa,\emptyset}))([\![t_2]\!]^{\mathcal{P}}_{\kappa,\emptyset}) \\
&\stackrel{\text{def}}{=} (\lambda(p)([\![t_1]\!]^{\mathcal{P}}_{\kappa,\emptyset}))([\![t_2]\!]^{\mathcal{P}}_{\kappa,\emptyset}) \\
&\stackrel{\text{def}}{=} p([\![t_1]\!]^{\mathcal{P}}_{\kappa,\emptyset}, [\![t_2]\!]^{\mathcal{P}}_{\kappa,\emptyset}) \\
&\neq p([\![t'_1]\!]^{\mathcal{P}}_{\kappa,\emptyset}, [\![t'_2]\!]^{\mathcal{P}}_{\kappa,\emptyset}) \qquad \text{(Induction and } p \text{ is injective)} \\
&\stackrel{\text{def}}{=} [\![f\ t'_1\ t'_2]\!]^{\mathcal{P}}_{\kappa,\emptyset}
\end{aligned}$$

□

Due to the fact that for any infinite set X there exists a countable infinite subset $S \subseteq X$, the above result can be generalised to hold for any infinite set X. The pairing function can then be applied on the enumeration of S to obtain a completeness result.

Corollary 6.3.9 *The full set-theoretic hierarchy $\mathcal{P}$ over any infinite set X is complete with respect to the theory of $\beta\eta$-conversion.*

Completeness Theorem for $\mathcal{N}$

We now demonstrate that the 1-section theorem can be used to prove that $\beta\eta$-conversion is complete for full nominal hierarchies over an infinite nominal set X.

A trivial example is $\mathcal{N}$ over $\mathbb{N}$, where the set of natural numbers is a nominal set via the trivial permutation action. In this case, completeness follows trivially using the same argument as for the full set-theoretic hierarchy over natural numbers and by observing that under the trivial permutation action the pairing function is equivariant. Note that this holds for any $\mathcal{N}$ over a discrete nominal set.

For nominal sets in general, it can easily be observed that the argument for the full set-theoretic hierarchy cannot be directly applied. This is due to the fact that a pairing function is not necessarily finitely supported. So, we need to provide another argument to demonstrate that the condition of the 1-section theorem is satisfied for $\mathcal{N}$. To be able to focus on the key reasoning steps, we restrict the completeness theorem to countable infinite nominal sets. The more general case follows without problems by observing that for any infinite nominal set X, there exists a countable infinite subset $S \subseteq X$. Note that for the construction of S the axiom of dependent choice is used and therefore the set is not necessarily a nominal set, but this is not an issue, because we only require that the elements are finitely supported.

Corollary 6.3.10 *The full nominal hierarchy $\mathcal{N}$ over a countable infinite nominal set X is complete for the pure theory of $\beta\eta$-conversion.*

Proof By Theorem 6.2.4 we have that $\mathcal{N}$ over an infinite nominal set is a Henkin model. Hence, we can apply the 1-section theorem to determine if $\mathcal{N}$ is complete.

In the case of $\mathcal{N}$ we demonstrate that for any binary tree t there exists a interpretation for c and f such that for any binary tree t' we have that if $t \not\equiv t'$ then the interpretation of t and t' are distinct. This is formally expressed as:

$$(\forall t)(\exists \kappa)(\forall t')\ .\ [t \not\equiv t' \implies [\![t]\!]^{\mathcal{N}}_{\kappa,\emptyset} \neq [\![t']\!]^{\mathcal{N}}_{\kappa,\emptyset}] \quad (\diamond)$$

From this, the property of the 1-section theorem can directly be deduced. We now choose an enumeration on the set of binary trees with the following basic properties:

$$t_0 \equiv c \qquad \text{and} \qquad (\forall t_i)\,(\forall t_j)\,[\,(t_j \text{ is a subtree of } t_i) \Rightarrow j \leq i\,]$$

Given that X is a countable infinite set we also have an enumeration for X. We now use both enumerations to construct for any binary tree t_i a corresponding function $g_i \in X \times X \Rightarrow_{fs} X$

$$g_i\ x_l\ x_m \stackrel{\text{def}}{=} \begin{cases} x_k & \text{if } ((k \leq i) \text{ where } (\mathsf{f}\ t_l\ t_m) \equiv t_k) \\ x_{i+1} & \text{o/w} \end{cases}$$

Next, we demonstrate that g_i is finitely supported by $A := \mathit{supp}(x_0) \cup ... \cup \mathit{supp}(x_{i+1})$. Let $c, c' \in \mathbb{A} \setminus A$ and $x_l, x_m \in X$.

Case 1: $(l > i+1)$ or $(m > i+i)$. We assume w.l.o.g that $l > i+1$. Note that for any binary tree t_j with $j > i+1$ we can deduce from the enumeration of binary trees that the index of $\mathsf{f}\ t_j\ t$ for any binary tree t is greater than j. So, we have by definition of g_i that $g_i(x_j, \_) \stackrel{\text{def}}{=} x_{i+1}$ ($\diamond$1).

We now assume, for a contradiction, that $(c\,c') \cdot x_l = x_k$ for $k \leq i+1$. By applying the permutation $(c\,c')$ on both sides of the equation, we obtain that $x_l = (c\,c') \cdot x_k$. Given that $k \leq i+1$ and $c, c' \# x_k$, we have that $x_l = x_k$ and therefore $l = k$. Hence, we have reached a contradiction, because $l > i+1$ and $k \leq i+1$, and therefore $k > i+1$ ($\diamond$2). We can now deduce the following equation:

$$\begin{aligned}
((c\,c') \cdot g_i)(x_l, x_m) &\stackrel{\text{def}}{=} (c\,c') \cdot (g_i((c\,c') \cdot x_l, (c\,c') \cdot x_m)) \\
&\stackrel{\text{def}}{=} (c\,c') \cdot x_{i+1} && (\diamond 1, \diamond 2) \\
&= x_{i+1} && (c, c' \# x_{i+1}) \\
&\stackrel{\text{def}}{=} g_i(x_l, x_m) && (\diamond 1)
\end{aligned}$$

Case 2: $(l, m \leq i+1)$

Case 2.1: $(l, m$ such that $f\ t_l\ t_m \equiv t_k$ and $k > i)$

$$\begin{aligned}
((c\,c') \cdot g_i)(x_l, x_m) &\stackrel{\text{def}}{=} (c\,c') \cdot (g_i((c\,c') \cdot x_l, (c\,c') \cdot x_m)) \\
&= (c\,c') \cdot (g_i(x_l, x_m)) && (c, c' \# (x_l, x_m)) \\
&\stackrel{\text{def}}{=} (c\,c') \cdot x_{i+1} \\
&= x_{i+1} && (c, c' \# x_{i+1}) \\
&\stackrel{\text{def}}{=} g_i(x_l, x_m)
\end{aligned}$$

Case 2.1: (l, m such that $f\ t_l\ t_m \equiv t_k$ and $k \leq i$)

$$\begin{aligned}((c\,c') \cdot g_i)(x_l, x_m) &\stackrel{\text{def}}{=} (c\,c') \cdot (g_i((c\,c') \cdot x_l, (c\,c') \cdot x_m)) \\ &= (c\,c') \cdot (g_i(x_l, x_m)) && (c, c' \,\#\, (x_l, x_m)) \\ &\stackrel{\text{def}}{=} (c\,c') \cdot x_k \\ &= x_k && (c, c' \,\#\, x_k) \\ &\stackrel{\text{def}}{=} g_i(x_l, x_m)\end{aligned}$$

Hence, we have that $\lambda(g_i) \in X \Rightarrow_{fs} X \Rightarrow_{fs} X$ and therefore it is an element of the full nominal hierarchy $\mathcal{N}$ over X.

Take $\kappa \stackrel{\text{def}}{=} \{(\mathsf{c}, x_0), (\mathsf{f}, \lambda(g_i))\}$. Then, by construction of g_i we can prove by induction that for all $j \leq i$, $[\![t_j]\!]^{\mathcal{N}}_{\kappa,\emptyset} = x_j$ and x_{i+1} otherwise:

Clearly, $[\![t_0]\!]^{\mathcal{N}}_{\kappa,} \stackrel{\text{def}}{=} [\![\mathsf{c}]\!]^{\mathcal{N}}_{\kappa,\emptyset} \stackrel{\text{def}}{=} x_0$. In the case of $t_m \equiv \mathsf{f}\ t_k\ t_l$ $(k, l < m \leq i)$ with $[\![t_k]\!]^{\mathcal{N}}_{\kappa,\emptyset} = x_k$ and $[\![t_l]\!]^{\mathcal{N}}_{\kappa,\emptyset} = x_l$, we have

$$[\![t_m]\!]^{\mathcal{N}}_{\kappa,\emptyset} = [\![\mathsf{f}\ t_k\ t_l]\!]^{\mathcal{N}}_{\kappa,\emptyset} \stackrel{\text{def}}{=} g_i([\![t_k]\!]^{\mathcal{N}}_{\kappa,\emptyset}, [\![t_l]\!]^{\mathcal{N}}_{\kappa,\emptyset}) = g_i(x_k, x_l) \stackrel{\text{def}}{=} x_m$$

The default case follows similarly. We can now prove property ($\diamond$): Suppose t is a binary tree with index k. We then take $\kappa \stackrel{\text{def}}{=} \{(\mathsf{c}, x_0), (\mathsf{f}, \lambda(g_k))\}$. Suppose that t' is a binary tree such that $t \not\equiv t'$. It follows immediately that $[\![t]\!]^{\mathcal{N}}_{\kappa,\emptyset} \neq [\![t']\!]^{\mathcal{N}}_{\kappa,\emptyset}$. The 1-section theorem can then be applied to obtain the completeness theorem. $\square$

Chapter 7

Yoneda isomorphisms for Nom/FM categories

In this chapter we study enriched and internal Yoneda isomorphism for nominal/FM categories. An overview of basic definitions and results of enriched and internal category theory will be provided in Appendix A. Our interest in this particular topic stems from our efforts towards proving that NLC is a conservative extension of NEL via a categorical proof argument, where such Yoneda isomorphisms play an important role (see Appendix B.1). The computations that we need to perform to obtain such a categorical conservative extension result are very "fine grained", why we have to use and understand the Nom/FM Yoneda lemmas not only abstractly, but concretely in the form presented in this Chapter. This allows us to compute explicitly with the permutation actions involved. However, a careful investigation of this topic is not only means to an end, but also interesting in its own right.

We begin by providing an overview of categorical properties, which are required to hold for nominal/FM categories such that an enriched or internal Yoneda isomorphisms can be obtained:

- For internal category theory we require that any category we internalise into

has pullbacks (see Definition A.2.1). In addition, it is often required to have finite completeness.

- For enriched category theory we require that the categories we enrich over are closed monoidal categories (see Definition A.1.2). Further, to obtain an enriched Yoneda isomorphism (embedding), an additional property has to satisfied, namely completeness (see Theorem A.1.10).

As demonstrated in Section 2.4, nominal/FM categories have terminal objects, initial objects, binary (co)products and (co)equaliser. Hence, by standard results [5], all three categories are finitely (co)complete as well as internally (co)complete taking into consideration that all three categories are also cartesian closed. From cartesian closure it follows immediately that all three categories are closed monoidal categories. What remains to be shown is that *Nom*, *FMSet* and *FMNom* are also complete. In addition, out of general interest, we also consider cocompleteness.

To sum it up. We first analyse if nominal/FM categories are (co)complete. We then unravel various weak and strong enriched Yoneda isomorphism, prove subsidiary results and point out some interesting observations. Next, we present an internal Yoneda isomorphism for *FMSet*. We conclude by introducing a notion of fs-(co)limits and prove that *FMSet* is complete with respect to this notion of fs-(co)limits.

7.1 (Co)Completeness of Nominal/FM Categories

We begin by recalling a well known fact, namely that in *Nom* each family of objects has a (co)product. In addition *Nom* has (co)equaliser, as shown in Lemma 2.4.1 and 2.4.2; thus we can directly deduce that *Nom* is (co)complete.

Lemma 7.1.1 *Let I be an index set and $(X_i \mid i \in I)$ a family of objects in Nom. A product of that family is defined as $(P, (pr_i \mid i \in I))$, where $P \subseteq \Pi_{i \in I} X_i$ is the set of finitely supported tuples with respect to the permutation action*

$$\pi \cdot (x_1, x_2, ..., x_i, ...) \stackrel{\text{def}}{=} (\pi \cdot x_1, \pi \cdot x_2, ..., \pi \cdot x_i, ...)$$

and $pr_j : \Pi_{i \in I} X_i \to X_j$ *are projection maps.*

Proof P is a nominal set by definition and equivariance of the projection maps follows by standard computations. For the universal property we suppose that $(D, (f_i \mid i \in I))$ is a cone. Then, analogous to products in *Set*, there exists a unique morphism $h : D \to P$, which is defined as $h(d) \stackrel{\text{def}}{=} (f_i(d) \mid i \in I)$ for any $d \in D$. We now need to show that h is well defined, i.e. $h(d)$ is an element of P. This follows immediately, because d is finitely supported (an element of the nominal set D) and the morphisms f_i are equivariant. Hence, we have that $supp(f_i(d)) \subseteq supp(d)$ and therefore $\bigcup_{i \in I} supp(f_i(d)) \subseteq supp(d)$. Thus, $h(d)$ is a finitely supported tuple and the fact that h is equivariant is deduced as follows:

$$\begin{aligned}
\pi \cdot h(d) &\stackrel{\text{def}}{=} \pi \cdot (f_i(d) \mid i \in I) \\
&\stackrel{\text{def}}{=} (\pi \cdot f_i(d) \mid i \in I) \\
&= (f_i(\pi \cdot d) \mid i \in I) && (f_i \text{ is equivariant}) \\
&\stackrel{\text{def}}{=} h(\pi \cdot d)
\end{aligned}$$

The commuting condition follows as in the case of *Set*. □

Lemma 7.1.2 *Let* I *be an index set and* $(X_i \mid i \in I)$ *a family of objects in Nom. A coproduct of that family is defined as* $(\coprod_{i \in I} X_i, (incl_i \mid i \in I))$*, where* $\coprod_{i \in I} X_i \stackrel{\text{def}}{=} \{(i, x) \mid i \in I \wedge x \in X_i\}$ *is the disjoint union over* X_i*, equipped with the permutation action*

$$\pi \cdot (i, x) \stackrel{\text{def}}{=} (\pi \cdot i, \pi \cdot x)$$

where $\pi \cdot i$ *is the trivial permutation action and* $\pi \cdot x$ *is the permutation action of* X_i *with* $supp(i, x) = supp(x)$*, and* $incl_j : X_j \to \coprod_{i \in I} X_i$ *are inclusion maps.*

Proof It follows directly that $\coprod_{i\in I} X_i$ is a nominal set and the inclusion maps are equivariant. For the universal property we suppose that $(C, (f_i \mid i \in I))$ is a cocone. The mediating morphism $h : \coprod_{i\in I} X_i \to C$, which is defined as $h((i,z)) \stackrel{\text{def}}{=} f_i(z)$, is well defined by construction. Further, h can be proven equivariant as follows:

$$\pi \cdot h(i,x) \stackrel{\text{def}}{=} \pi \cdot f_i(x) = f_i(\pi \cdot x) \stackrel{\text{def}}{=} h(i, \pi \cdot x) \stackrel{\text{def}}{=} h(\pi \cdot i, \pi \cdot x)$$

□

Remark 7.1.3 *The fact that Nom is (co)complete can also be directly deduced from Nom being equivalent to Set^Ip.b.p (Schanuel topos)*[1]*, where I is the category of finite names and injective functions, and the fact that Set^Ip.b.p is a full reflective subcategory of Set^I:*

$$Nom \simeq Set^I p.b.p \hookrightarrow Set^I$$

We recall that a reflective subcategory is closed under limits. This is not the case for colimits, where the reflective functor can be used to transform colimits in Set^I into colimits of Set^Ip.b.p. Note that Set^I is (co)complete (via a point-wise definition). We can now directly deduce that Set^Ip.b.p is (co)complete with (co)limits constructed as indicated above. Using the equivalence of categories, we then directly obtain that Nom is (co)complete with the corresponding (co)limits.

A natural question to ask is if *FMNom* and *FMSet* can also be proven (co)complete. So, let's first reconsider the construction of a mediating function for products and coproducts in *Nom*:

- $h(d) \stackrel{\text{def}}{=} (f_i(d) \mid i \in I)$ and
- $h((i,z)) \stackrel{\text{def}}{=} f_i(z)$

As can immediately be observed, the mediating function for products would not be well defined anymore, because for all i we have that $supp(f_i(d)) \subseteq supp(f_i) \cup supp(d)$,

[1] we refer to [56] for full details.

but the family of f_i's does not necessarily have a common finite support. Hence, the tuple would not be finitely supported. For the same reason, the mediating function for coproducts is not finitely supported anymore.

Now, of course, there might be an alternative construction of products and coproducts, but this is not the case, as we will show with the following proofs by contradiction.

Proposition 7.1.4 *FMNom is not complete.*

Proof We assume, for a contradiction, that *FMNom* is complete. Let I be an infinite index set, indeed just ω, and $(\mathbb{A} \mid i \in I)$ a family of infinite copies of $\mathbb{A}$. By assumption, a product $\Pi_{i \in I}\mathbb{A}$ (referred to as P) exists in *FMNom* with projection maps $pr_i : P \to \mathbb{A}$. Note that by definition we have that all projection maps, pr_i, are finitely supported. We now recursively select names from $\mathbb{A}$ such that $a_1 \notin supp(pr_1)$ and for $i > 1$ we pick

$$a_i \notin \bigcup_{j \leq i} supp(pr_j) \cup \{a_j \mid j < i\}$$

Hence, we have infinitely many distinct names a_i, which satisfy $\forall u \in \mathbb{N}. \forall k \in \mathbb{N}. k \geq u \Rightarrow a_k \# pr_u$. We then introduce morphisms $f_i : 1 \to \mathbb{A}$, which are defined by $f_i(*) \stackrel{\text{def}}{=} a_i$. The functions f_i are finitely supported by their image $\{a_i\}$. Thus $(1, \{f_i \mid i \in I\})$ is a cone for $(\mathbb{A} \mid i \in I)$ in *FMNom*.

Applying the universal property for products, there exists a finitely supported morphism $h : 1 \to P$ in *FMNom*. Given that the support of h is finite, there exist $u \in \mathbb{N}$ such that for all $k \geq u$ we have $a_k \# h$. Let $n \in \mathbb{N}$ be such a natural number. We can now pick the two names a_{n+1} and a_{n+2}, which are distinct by construction,

and deduce the following contradiction:

$$\begin{aligned}
a_{n+2} &= (a_{n+2}\, a_{n+1}) \cdot a_{n+1} \\
&\stackrel{\text{def}}{=} (a_{n+2}\, a_{n+1}) \cdot f_{n+1}(*) \\
&= (a_{n+2}\, a_{n+1}) \cdot (pr_{n+1}(h(*))) && \text{(universal property)} \\
&= pr_{n+1}((a_{n+2}\, a_{n+1}) \cdot h(*)) && (a_{n+1}, a_{n+2} \# p_{n+1}) \\
&= pr_{n+1}(h((a_{n+2}\, a_{n+1}) \cdot *)) && (a_{n+2}, a_{n+1} \# h) \\
&= pr_{n+1}(h(*)) && \text{(trivial permutation action)} \\
&= f_{n+1}(*) && \text{(universal property)} \\
&\stackrel{\text{def}}{=} a_{n+1}
\end{aligned}$$

□

Proposition 7.1.5 *FMNom is not cocomplete.*

Proof We assume, for a contradiction, that *FMNom* is cocomplete. Let I be an infinite index set, indeed just ω, and $(\mathbb{A} \mid i \in I)$ a family of infinite copies of $\mathbb{A}$. By assumption, a coproduct $\Sigma_{i\in I}\mathbb{A}$ (referred to as C) exists in *FMNom* with inclusion maps $incl_i : \mathbb{A} \to C$. Note that, by definition, all inclusion maps $incl_i$ are finitely supported. We now recursively select names from $\mathbb{A}$ such that $a_1 \notin supp(incl_1)$ and for $i > 1$ we pick

$$a_i \notin \bigcup_{j\leq i} supp(incl_j) \cup \{a_j \mid j < i\}$$

Hence, we have infinitely many distinct names a_i, which satisfy $\forall u \in \mathbb{N}.\, \forall k \in \mathbb{N}.\, k \geq u \Rightarrow a_k \# incl_u$. We then introduce constant functions $g_i : \mathbb{A} \to \mathbb{A}$, which are defined by $f_i(a) \stackrel{\text{def}}{=} a_i$ for any $a \in \mathbb{A}$. The functions f_i are finitely supported by their image $\{a_i\}$.

Thus $(\mathbb{A}, \{g_i \mid i \in I\})$ is a cone for the diagram F in *FMNom*. Using the universal property for coproducts, there exists a finitely supported morphism $h : C \to \mathbb{A}$ in *FMNom*. Given that the support of h is finite, there exist $u \in \mathbb{N}$ such that for all

$k \geq u$ we have $a_k \# h$. Let $n \in \mathbb{N}$ be such a natural number. We can now pick two names a_{n+1} and a_{n+2}, which are distinct by definition, and deduce the following contradiction:

$$\begin{aligned}
a_{n+2} &= (a_{n+2}\, a_{n+1}) \cdot a_{n+1} \\
&\stackrel{\text{def}}{=} (a_{n+2}\, a_{n+1}) \cdot g_{n+1}(a) \\
&= (a_{n+2}\, a_{n+1}) \cdot (h(incl_{n+1}(a))) && \text{(univeral property)} \\
&= h((a_{n+2}\, a_{n+1}) \cdot incl_{n+1}(a)) && (a_{n+2}, a_{n+1} \# h) \\
&= h(incl_{n+1}((a_{n+2}\, a_{n+1}) \cdot a)) && (a_{n+1}, a_{n+2} \# incl_{n+1}) \\
&= g_{n+1}((a_{n+2}\, a_{n+1}) \cdot a) && \text{(univeral property)} \\
&\stackrel{\text{def}}{=} a_{n+1}
\end{aligned}$$

□

Corollary 7.1.6 *FMSet is not (co)complete.*

Proof We have that 1 and $\mathbb{A}$ are FM-Sets. Given that *FMNom* is a full subcategory or *FMSet*, the same constructions in the proof by contradiction for *FMNom* carry over to *FMSet*. □

Remark 7.1.7 *Considering the rich structure of Set and Nom, the results for FMNom and FMSet seem, at first, slightly surprising, but in retrospect it is natural to expect that for an infinite collection of finitely supported functions, a finitely supported factorisation does not exist. Note that this result is not directly mentioned in the nominal/FM literature, but can be seen as a Folk's theorem in the context of permutation models, which we now recall in the context of nominal/FM categories.*

7.2 Enriched Yoneda Isomorphisms

In this section we discuss enriched Yoneda isomorphisms for nominal/FM categories. We start by unravelling the weak Yoneda lemma for *Nom*, *FMNom* and *FMSet*

and the strong Yoneda lemma for *Nom*. This allows us to present the respective theorems in a more "natural" way, which will pay dividends when working towards a categorical conservative extension result. Thereby we also prove some interesting subsidiary results.

The fact that *FMNom* and *FMSet* are not complete leads to some immediate complications when one is interested in working with enriched functor categories and enriched Yoneda isomorphisms (embeddings). We briefly discuss these complications and point out a way to circumvent them via universe enlargement.

7.2.1 Weak and Strong Yoneda Lemma

Before we unravel and discuss the enriched Yoneda isomorphisms for different nominal/FM categories, we recall the basic notions leading to an enriched Yoneda lemma in Appendix A.1. Moreover, we recall that *Nom*, *FMNom* and *FMSet* are cartesian closed categories with the exponential for objects X and Y denoted by $X \Rightarrow_{fs} Y$. We can now deduce that every cartesian closed category is a symmetric monoidal closed category:

- The tensor product $\otimes$ and monoidal unit I are respectively given by the cartesian product $\times$ and the terminal object 1.
- The "associativity" isomorphism $\alpha_{A,B,C} : (A \times B) \times C \to A \times (B \times C)$ is defined as $\langle pr_1 \circ pr_1, \langle pr_2 \circ pr_1, pr_2 \rangle \rangle$. The inverse is defined analogously.
- The "left unit" natural isomorphism $l_A : 1 \times A \to A$ is defined as $l_A \stackrel{\text{def}}{=} pr_2$ and the "right unit" natural isomorphism $r_A : A \times 1 \to A$ as $r_A \stackrel{\text{def}}{=} pr_1$. The inverses are defined analogously.
- The closure property follows immediately with the internal homset being defined as $[A, B] \stackrel{\text{def}}{=} A \Rightarrow_{fs} B$.

- The "symmetric" natural isomorphism $s_{A,B} : A \times B \to B \times A$ is defined as $s_{A,B} \stackrel{\text{def}}{=} \langle \pi_2, \pi_1 \rangle$. The inverse is defined analogously.

Thus, each nominal/FM category $\mathcal{V}$ is trivially symmetric monoidal closed and therefore $\mathcal{V}$ can be enriched over itself, which is notationally expressed as $\mathcal{V}^{\text{er}}$ with the hom object

$$\mathcal{V}^{\text{er}}(X, Y) \stackrel{\text{def}}{=} [X, Y] \stackrel{\text{def}}{=} X \Rightarrow_{fs} Y$$

Let $\mathbb{C}$ be a $\mathcal{V}$-enriched category and $F, G : \mathbb{C} \to \mathcal{V}^{\text{er}}$ are $\mathcal{V}$-valued $\mathcal{V}$-functors. We now introduce the notion of "natural families" of morphism, such as those that arise as the components of natural transformations between functors (in ordinary category theory). For brevity, we write $\text{NC}(\alpha) = \text{NC}(\alpha, \text{F}, \text{G}, \text{C}, \text{C}', \text{f})$ for the naturaliy condition: $G(f) \circ \alpha_C = \alpha_{C'} \circ F(f)$ for all $f \in \mathbb{C}(C, C')$.

$$\begin{aligned}
\text{Nat}(F, G) &\stackrel{\text{def}}{=} \{\alpha \in \Pi_{C \in ob(\mathbb{C})} |FC| \Rightarrow |GC| \mid \text{NC}(\alpha)\} \\
\text{Nat}_{es}(F, G) &\stackrel{\text{def}}{=} \{\alpha \in \Pi_{C \in ob(\mathbb{C})} |(FC \Rightarrow_{fs} GC)_{es}| \mid \text{NC}(\alpha)\} \\
\text{Nat}_{fs}(F, G) &\stackrel{\text{def}}{=} \{\alpha \in \Pi_{C \in ob(\mathbb{C})} |FC \Rightarrow_{fs} GC| \mid \text{NC}(\alpha)\}
\end{aligned}$$

where the products of underlying function spaces are in *Set*. The elements of these sets are called **ordinary natural families**.

Weak Enriched Yoneda Lemma

We now apply the weak Yoneda lemmas on all three nominal/FM categories and further show how the collection of $\mathcal{V}$-natural transformation between $\mathcal{V}$-valued $\mathcal{V}$-enriched functors $\mathbb{C}^A$ and F, written as $\mathcal{V}\text{–Nat}(\mathbb{C}^A, F)$, relates to the previously introduced sets of ordinary natural families. This provides us with a more "natural" way to present the weak Yoneda lemmas for the respective categories.

Corollary 7.2.1 (Weak Yoneda Lemma for *Nom***)** *Let $\mathbb{C}$ be a small Nom-category. For every Nom-functor $F : \mathbb{C} \to Nom^{\text{er}}$ and every object $A \in ob(\mathbb{C})$, there exists a*

bijection

$$|(FA)_{es}| \cong_{Set} Nom\text{–}\mathrm{Nat}(\mathbb{C}^A, F) \cong_{Set} \mathrm{Nat}_{es}(\mathbb{C}^A, F)$$

Proof For global elements of FA (in Nom) we immediately obtain that

$$|(FA)_{es}| \cong_{Set} Nom(1_{Nom}, FA)$$

Next, we observe that $Nom\text{–}\mathrm{Nat}(\mathbb{C}^A, F) \cong_{Set} \mathrm{Nat}_{es}(\mathbb{C}^A, F)$: We recall that a Nom-enriched natural transformation $\alpha : \mathbb{C}^A \Rightarrow F$ is defined to be a family of morphisms (global elements) $(\alpha_B : 1_{Nom} \to Nom^{\mathrm{er}}(\mathbb{C}(A,B), F(B)) \stackrel{\mathrm{def}}{=} \mathbb{C}(A,B) \Rightarrow_{fs} FB \mid B \in ob(\mathbb{C}))$ such that for all objects $A, A' \in ob(\mathbb{C})$ we have that

$$c_{FA,GA,GA'} \circ (\alpha_A \otimes F_{A,A'}) \circ l^{-1}_{\mathbb{A}(A,A')} = c_{FA,FA',GA'} \circ (\mathbb{C}^A_{A,A'} \otimes \alpha_{A'}) \circ r^{-1}_{\mathbb{A}(A,A')}$$

Again, for global elements of $\mathbb{C}(A,B) \Rightarrow_{fs} FB$ we immediately obtain that

$$|(\mathbb{C}(A,B) \Rightarrow_{fs} FB)_{es}| \cong_{Set} Nom(1_{Nom}, \mathbb{C}(A,B) \Rightarrow_{fs} FB)$$

and therefore the components correspond with the ordinary natural families. What remains to be shown is that the commuting condition relate as well. By definition of Nom^{er}, the "composition" morphism is defined as

$$c_{A,B,C} \stackrel{\mathrm{def}}{=} \lambda((ev_{B,C} \circ s_{B,B\Rightarrow_{fs}C} \circ (ev_{A,B} \otimes id_{B\Rightarrow_{fs}C}) \circ s_{B\Rightarrow_{fs}C,A})$$

Considering the definition of the exponential mate $\lambda(-)$, the evaluation map ev and the symmetry map s for Nom, we obtain that the "composition" morphism is the "usual" composition $\circ$ of Nom. With the definition of l^{-1} and r^{-1}, as previously presented, we can now compute each side of the equation for $\theta \in \mathbb{C}(A, A')$:

$$\begin{aligned}(c_{FA,GA,GA'} \circ (\alpha_A \times F_{A,A'}) \circ l^{-1}_{\mathbb{C}(A,A')})(\theta) &\stackrel{\mathrm{def}}{=} (c_{FA,GA,GA'} \circ (\alpha_A \times F_{A,A'}))(*, \theta) \\ &\stackrel{\mathrm{def}}{=} c_{FA,GA,GA'}(\alpha_A(*), F_{A,A'}(\theta)) \\ &\stackrel{\mathrm{def}}{=} F_{A,A'}(\theta) \circ \alpha_A(*) \\ &\stackrel{\mathrm{def}}{=} F_{A,A'}(\theta) \circ \overline{\alpha}_A\end{aligned}$$

$$\begin{aligned}(c_{FA,FA',GA'} \circ (\mathbb{C}^A_{A,A'} \times \alpha_{A'}) \circ r^{-1}_{\mathbb{C}(A,A')})(\theta) &\stackrel{\text{def}}{=} (c_{FA,FA',GA'} \circ (\mathbb{C}^A_{A,A'} \times \alpha_{A'}))(\theta, *) \\ &\stackrel{\text{def}}{=} c_{FA,FA',GA'}(\mathbb{C}^A_{A,A'}(\theta), \alpha_{A'}(*)) \\ &\stackrel{\text{def}}{=} \alpha_{A'}(*) \circ \mathbb{C}^A_{A,A'}(\theta) \\ &\stackrel{\text{def}}{=} \overline{\alpha}_{A'} \circ \mathbb{C}^A_{A,A'}(\theta)\end{aligned}$$

Hence, we have that $F_{A,A'}(\theta) \circ \overline{\alpha}_A = \overline{\alpha}_{A'} \circ \mathbb{C}^A_{A,A'}(\theta)$ (NC(α) holds) and therefore

$$Nom\text{–}\mathrm{Nat}(\mathbb{C}^A, F) \cong_{Set} \mathrm{Nat}_{es}(\mathbb{C}^A, F)$$

Recall that the enriched weak Yoneda lemma for *Nom* states that

$$Nom(1_{Nom}, FA) \cong_{Set} Nom\text{–}\mathrm{Nat}(\mathbb{C}^A, F)$$

Thus, we have

$$|(FA)_{es}| \cong_{Set} Nom(1_{Nom}, FA) \cong_{Set} Nom\text{–}\mathrm{Nat}(\mathbb{C}^A, F) \cong_{Set} \mathrm{Nat}_{es}(\mathbb{C}^A, F)$$

□

Corollary 7.2.2 (Weak Yoneda Lemma for *FMNom* and *FMSet*) *Let $\mathbb{C}$ be a small $\mathcal{V}$-category. For every $\mathcal{V}$-functor $F : \mathbb{C} \to \mathbb{V}$ and every object $A \in ob(\mathbb{C})$, there exists a bijection*

$$|FA| \cong_{Set} \mathcal{V}\text{–}\mathrm{Nat}(\mathbb{C}^A, F) \cong_{Set} \mathrm{Nat}_{fs}(\mathbb{C}^A, F)$$

Proof It follows analogously to the previous corollary.

Strong Enriched Yoneda Lemma

We continue by providing a more "natural" presentation of the strong Yoneda lemma for *Nom*:

Corollary 7.2.3 (Strong Yoneda Lemma for *Nom*) *Let* $\mathbb{C}$ *be a small Nom-category. For every Nom-functor* $F : \mathbb{C} \to Nom^{\mathrm{er}}$ *and every object* $A \in ob(\mathbb{C})$*, there exists an isomorphism*

$$FA \cong_{Nom} [\mathbb{C}, Nom^{\mathrm{er}}](\mathbb{C}^A, F) = (\mathrm{Nat}_{fs}(\mathbb{C}^A, F), \cdot_{\int})$$

where the permutation action for $\alpha \in \mathrm{Nat}_{fs}(\mathbb{C}^A, F)$ *is given by*

$$(\pi \cdot_{\int} \alpha)_C \stackrel{\mathrm{def}}{=} \pi \cdot_{\Rightarrow_{fs}} \alpha_C$$

Proof The isomorphism follows directly from the Strong Yoenda lemma. Following Borceux [6] (Proposition 6.3.1), we unravel $[\mathbb{C}, Nom^{\mathrm{er}}](\mathbb{C}^A, F)$, which is defined as the equaliser $EQR(u, v)$ over the pair of morphisms

$$u, v : \Pi_{A \in ob(\mathbb{C})}(\mathbb{C}(A, A) \Rightarrow_{fs} FA) \to \Pi_{A', A'' \in ob(\mathbb{C})}(\mathbb{C}(A', A'') \Rightarrow_{fs} (\mathbb{C}(A, A') \Rightarrow_{fs} FA''))$$

which are defined as follows:

- u is the "mediating" product morphism, written as $\langle u_{A',A''} \mid A', A'' \in ob(\mathbb{C})\rangle$, for the family of morphisms

$$(u_{A',A''} : \Pi_{A \in ob(\mathbb{C})}(\mathbb{C}(A, A) \Rightarrow_{fs} F(A)) \to \mathbb{C}(A', A'') \Rightarrow_{fs} (\mathbb{C}(A, A') \Rightarrow_{fs} FA'') \mid A', A'' \in ob(\mathbb{C}))$$

  where $u_{A',A''} \stackrel{\mathrm{def}}{=} \lambda(c_{\mathbb{C}(A,A'),FA',FA''} \circ (pr_{A'} \times F_{A',A''}))$.

- v is the "mediating" product morphism, written as $\langle v_{A',A''} \mid A', A'' \in ob(\mathbb{C})\rangle$, for the family of morphisms

$$(v_{A',A''} : \Pi_{A \in ob(\mathbb{C})}(\mathbb{C}(A, A) \Rightarrow_{fs} F(A)) \to \mathbb{C}(A', A'') \Rightarrow_{fs} (\mathbb{C}(A, A') \Rightarrow_{fs} FA'') \mid A', A'' \in ob(\mathbb{C}))$$

  where $v_{A',A''} \stackrel{\mathrm{def}}{=} \lambda(c_{\mathbb{C}(A,A'),\mathbb{C}(A,A''),FA''} \circ (\mathbb{C}^A_{A',A''} \times pr_{A''}) \circ s)$.

Given how equalisers in *Nom* are defined we have that

$$E \stackrel{\text{def}}{=} \{\alpha \in \Pi_{B \in ob(\mathbb{C})}(\mathbb{C}(A,B) \Rightarrow_{fs} FB) \mid u(\alpha) = v(\alpha)\}$$

Let $\alpha \in \Pi_{B \in ob(\mathbb{C})}(\mathbb{C}(A,B) \Rightarrow_{fs} FB)$, $A', A'' \in ob(\mathbb{C})$ and $f \in \mathbb{C}(A', A'')$. We now unravel the property $u(\alpha) = v(\alpha)$:

$$\begin{aligned}
&(pr_{A',A''} \circ u)(\alpha)(f) \\
&= u_{A',A''}(\alpha)(f) && \text{(mediating morphisms)} \\
&\stackrel{\text{def}}{=} \lambda(c_{\mathbb{C}(A,A'),FA',FA''} \circ (pr_{A'} \times F_{A',A''}))(\alpha)(f) \\
&\stackrel{\text{def}}{=} c_{\mathbb{C}(A,A'),FA',FA''} \circ (pr_{A'} \times F_{A',A''})(\alpha, f) \\
&\stackrel{\text{def}}{=} c_{\mathbb{C}(A,A'),FA',FA''}(pr_{A'}(\alpha), F_{A',A''}(f)) \\
&\stackrel{\text{def}}{=} F_{A',A''}(f) \circ \alpha_{A'}
\end{aligned}$$

$$\begin{aligned}
&(pr_{A',A''} \circ v)(\alpha)(f) \\
&= v_{A',A''}(\alpha)(f) && \text{(mediating morphisms)} \\
&\stackrel{\text{def}}{=} \lambda(c_{\mathbb{C}(A,A'),\mathbb{C}(A,A''),FA''} \circ (\mathbb{C}^A_{A',A''} \times pr_{A''}) \circ s)(\alpha)(f) \\
&\stackrel{\text{def}}{=} c_{\mathbb{C}(A,A'),\mathbb{C}(A,A''),FA''} \circ (\mathbb{C}^A_{A',A''} \times pr_{A''}) \circ s(\alpha, f) \\
&\stackrel{\text{def}}{=} c_{\mathbb{C}(A,A'),\mathbb{C}(A,A''),FA''} \circ (\mathbb{C}^A_{A',A''} \times pr_{A''})(f, \alpha) \\
&\stackrel{\text{def}}{=} c_{\mathbb{C}(A,A'),\mathbb{C}(A,A''),FA''}(\mathbb{C}^A_{A',A''}(f), pr_{A''}(\alpha)) \\
&\stackrel{\text{def}}{=} \alpha_{A''} \circ \mathbb{C}^A_{A',A''}(f)
\end{aligned}$$

Hence, we have that $F_{A',A''}(f) \circ \alpha_{A'} = \alpha_{A''} \circ \mathbb{C}^A_{A',A''}(f)$ and therefore $\mathrm{NC}(\alpha)$ holds. Recalling how products in *Nom* are defined, we have that $\Pi_{B \in ob(\mathbb{C})}(\mathbb{C}(A,B) \Rightarrow_{fs} FB)$ is given as

$$\{\alpha = (\alpha_B \in \mathbb{C}(A,B) \Rightarrow_{fs} FB \mid B \in ob(\mathbb{C})) \mid supp(\alpha) < \infty\}$$

where the permutation action is $\pi \cdot \alpha \stackrel{\text{def}}{=} \pi \cdot_{\Rightarrow_{fs}} \alpha_B$ with $supp(\alpha) = \bigcup_{B \in ob(\mathbb{C})} supp(\alpha_B)$. To sum it up, we have that

$$[\mathbb{C}, Nom^{\mathrm{er}}](\mathbb{C}^A, F) = EQR(u,v) \in ob\, Nom$$

is the nominal set

$$\{(\alpha_B \in \mathbb{C}(A,B) \Rightarrow_{fs} FB \mid B \in ob(\mathbb{C})) \mid \bigcup_{B \in ob(\mathbb{C})} supp(\alpha_B) < \infty \wedge \mathrm{NC}(\alpha)\}$$

with $\pi \cdot_{\int} \alpha \stackrel{\text{def}}{=} \pi \cdot_{\Rightarrow_{fs}} \alpha_B$. We can then immediately deduce that

$$[\mathbb{C}, Nom^{\mathrm{er}}](\mathbb{C}^A, F) \subseteq \mathrm{Nat}_{fs}(\mathbb{C}^A, F)$$

For the converse we suppose that $\alpha \in \mathrm{Nat}_{fs}(\mathbb{C}^A, F)$. To deduce that $\alpha \in [\mathbb{C}, Nom^{\mathrm{er}}](\mathbb{C}^A, F)$, we have to show that $supp(\alpha)$ is finite. By the definition of $\mathrm{NC}(\alpha)$, or more precisely the instance $\mathrm{NC}(\alpha, \mathrm{A}, \mathrm{B}, \mathrm{f})$ for $f \in \mathbb{C}(A,B)$ we can deduce $(\diamond)$

$$\begin{aligned}
\alpha_B(f) &= \alpha_B(f \circ id_A) \\
&\stackrel{\text{def}}{=} (\alpha_B \circ \mathbb{C}(A,f))(id_A) \\
&= ((Ff) \circ \alpha_A)(id_A) && (\mathrm{NC}((\alpha, \mathrm{A}, \mathrm{B}, \mathrm{f}))) \\
&= (Ff)(\alpha_A(id_A))
\end{aligned}$$

Based on this observation, we can now demonstrate that $supp(\alpha_B) \subseteq supp(\alpha_A(id_A))$. Let $\pi \# \alpha_A(id_A)$ and $f \in \mathbb{C}(A,B)$. Noting that $\pi^{-1} \cdot f \in \mathbb{C}(A,B)$ we can deduce

$$\begin{aligned}
(\pi \cdot_{\Rightarrow_{fs}} \alpha_B)(f) &\stackrel{\text{def}}{=} \pi \cdot (\alpha_B(\pi^{-1} \cdot f)) \\
&= \pi \cdot ((F_{A,B}(\pi^{-1} \cdot f))(\alpha_A(id_A)) && (\diamond) \\
&= \pi \cdot ((\pi^{-1} \cdot (F_{A,B}f))(\alpha_A(id_A))) && (F_{A,B} \text{ is equivariant}) \\
&= (F_{A,B}f)(\pi \cdot (\alpha_A(id_A))) \\
&= (F_{A,B}f)(\alpha_A(id_A)) && (\pi \# \alpha_A(id_A)) \\
&= \alpha_B(f) && (\diamond)
\end{aligned}$$

Given that $supp(\alpha) = \bigcup_{B \in ob(\mathbb{C})} supp(\alpha_B) = supp(\alpha_A(id_A)) < \infty$, the argument is complete. $\square$

Remark 7.2.4 *We have seen in the proof of Theorem 7.2.3 that the naturality condition* $\mathrm{NC}(\alpha)$ *implies that for any* $C \in ob(\mathbb{C})$ *we have* $supp(\alpha_C) = supp(\alpha_A(id_A))$. *We used this to show that* $supp(\alpha) = \bigcup_{C \in ob(\mathbb{C})} supp(\alpha_C)$ *is finite. But there is more to it, because it shows that naturality is itself a sufficient condition to ensure that each component* α_C *is indeed finitely supported (by* $supp(\alpha_A(id_A))$*. Thus finite support is a theorem for free and ordinary natural families coincide with finitely supported natural families.*

This is a simple, but in some ways slightly surprising property of natural transformations in the nominal setting. More importantly, it allows us to present the strong Yoneda lemma in a particularly pleasing way, which extends the weak Yoneda lemma very directly.

Corollary 7.2.5 *Let* $\mathbb{C}$ *be enriched over Nom and* $F : \mathbb{C} \to Nom^{\mathrm{er}}$ *be an enriched functor.*

$$FA \cong_{Nom} (\mathrm{Nat}(\mathbb{C}^A, F), \cdot_{\int})$$

Moreover, we have $(\mathrm{Nat}(\mathbb{C}^A, F), \cdot_{\int}) \cong_{Nom} (\mathrm{Nat}_{fs}(\mathbb{C}^A, F), \cdot_{\int})$ *(via the identity)*

Before we provide a direct proof of the strong Yoneda lemma for *Nom*, we prove the following "lifting" lemma:

Lemma 7.2.6 *Suppose that* $M = (|M|, \cdot)$ *is a nominal set and* S *is a set such that* $\Phi : |M| \cong_{Set} S : \Psi$ *is an isomorphism of sets. Then* S *is a nominal set via the "canonical" permutation action:* $\pi * s \stackrel{\mathrm{def}}{=} \Phi(\pi \cdot_M \Psi(s))$ *with* $supp(s) = supp(\Psi(s))$ *and moreover* $\Phi : M \cong_{Nom} (S, *)$.

Proof The map $(\pi, x) \mapsto \pi * x$ is well defined by construction. It follows immediately that it is a permutation action:

$$\iota * x \stackrel{\mathrm{def}}{=} \Phi(\iota \cdot_M \Psi(x)) = \Phi(\Psi(x)) = x$$

$$\begin{aligned}
\pi' * (\pi * x) &\stackrel{\text{def}}{=} \pi' * \Phi(\pi \cdot_M \Psi(x)) \\
&\stackrel{\text{def}}{=} \Phi(\pi' \cdot_M \Psi(\Phi(\pi \cdot_M \Psi(x)))) \\
&= \Phi(\pi' \cdot_M (\pi \cdot_M \Psi(x))) && \text{(isomorphism)} \\
&= \Phi((\pi' \circ \pi) \cdot_M \Psi(x)) && \text{(permutation action)} \\
&\stackrel{\text{def}}{=} (\pi' \circ \pi) * x
\end{aligned}$$

Next, we demonstrate that each element of $|S|$ is supported by $supp(\Psi(x))$. Let $x \in |S|$ and $a, a' \# \Psi(x)$. We can then deduce that

$$(a\, a') * x \stackrel{\text{def}}{=} \Phi((a\, a') \cdot_M \Psi(x)) = \Phi(\Psi(x)) = x$$

Given that $\Psi(x) \in |M|$ and M is a nominal set, we have that $\Psi(x)$ is finitely supported and therefore x finitely supported as well. Hence, $(|S|, *)$ is a nominal set. Finally we show that Φ and Ψ are equivariant.

$$\pi * \Phi(x) \stackrel{\text{def}}{=} \Phi(\pi \cdot_M \Psi(\Phi(x))) = \Phi(\pi \cdot_M x)$$

$$\pi \cdot_M \Psi(\alpha) = \Psi(\Phi(\pi \cdot_M \Psi(\alpha))) \stackrel{\text{def}}{=} \Psi(\pi * \alpha)$$

□

Theorem 7.2.7 (Strong Yoneda Lemma: Bare Hands Version) *Let $\mathbb{C}$ be a small Nom-category and $F : \mathbb{C} \to Nom^{\text{er}}$ a Nom-functor. Then there is an isomorphism in Set*

$$\Phi : |FA| \cong_{Set} \text{Nat}_{fs}(\mathbb{C}^A, F) : \Psi$$

given by $\Phi(x) \stackrel{\text{def}}{=} \overline{x}$ with $\overline{x}_C(\theta) \stackrel{\text{def}}{=} (F\theta)(x)$ and $\Psi(\alpha) \stackrel{\text{def}}{=} \alpha_A(id_A)$ (ordinary Yoneda lemma). Further, we have that $\text{Nat}_{fs}(\mathbb{C}^A, F)$ is a nominal set via the permutation action $*$, which is defined by passing the permutation action of FA across the bijection: $\pi * \alpha \stackrel{\text{def}}{=} \Phi(\pi \cdot_{FA} \Psi(\alpha))$. Moreover, we can show that this permutation action coincides with the point-wise permutation from Corollary 7.2.3, i.e. we have that $\pi * \alpha = \pi \cdot_{\int} \alpha$. This all leads to an isomorphism in Nom which is natural in A and F.

$$\Phi : FA \cong_{Nom} (\text{Nat}_{fs}(\mathbb{C}^A, F), *) = (\text{Nat}_{fs}(\mathbb{C}^A, F), \cdot_{\int}) : \Psi$$

Proof We begin by demonstrating that Φ is well defined. By definition of Φ, as given in the ordinary Yoneda lemma, we have that $\overline{x}$ is a natural transformation with components $\overline{x}_C : \mathbb{C}(A, C) \to FC$. So, we have that NC($\overline{\mathrm{x}}$). What remains to be shown is that for each $C \in ob(\mathbb{C})$ we have that $\overline{x}_C$ is finitely supported under $\cdot_{\Rightarrow_{fs}}$. Let $\pi \# x \in FA$ and $\theta \in \mathbb{C}(A, C)$.

$$\begin{aligned}
(\pi \cdot_{\Rightarrow_{fs}} \overline{x}_C)(\theta) &\stackrel{\text{def}}{=} \pi \cdot_{FC} (\overline{x}_C(\pi^{-1} \cdot_{\mathbb{C}(A,C)} \theta)) \\
&\stackrel{\text{def}}{=} \pi \cdot_{FC} (F_{A,C}(\pi^{-1} \cdot_{\mathbb{C}(A,C)} \theta))(x) \\
&= \pi \cdot_{FC} (\pi^{-1} \cdot_{\mathbb{C}(A,C)} F_{A,C}(\theta))(x) && (F_{A,C} \text{ is equivariant}) \\
&\stackrel{\text{def}}{=} F_{A,C}(\theta)(\pi \cdot x) \\
&= F_{A,C}(\theta)(x) && (\pi \# x) \\
&\stackrel{\text{def}}{=} \overline{x}_C(\theta)
\end{aligned}$$

Ψ is trivially well defined, because α_A is a map from $\mathbb{C}(A, A)$ to FA and therefore $\alpha_A(id_A) \in FA$. The fact that Φ and Ψ are mutually inverse follows exactly as with the ordinary Yoneda lemma.

$$\Phi : |FA| \cong_{Set} \mathrm{Nat}_{fs}(\mathbb{C}^A, F) : \Psi$$

Now, since FA is by definition a nominal set, Lemma 7.2.6 ("lifting lemma") provides us with a permutation action $*$ on $\mathrm{Nat}_{fs}(\mathbb{C}^A, F)$ such that Φ and Ψ yield a nominal isomorphism.

$$\Phi : FA \cong_{Nom} (\mathrm{Nat}_{fs}(\mathbb{C}^A, F), *) : \Psi$$

What remains to be shown it that $\pi \cdot_{\int} \alpha = \pi * \alpha$

$$
\begin{aligned}
(\pi * \alpha)_C(\theta) &\stackrel{\text{def}}{=} \Phi(\pi \cdot_{FA} \Psi(\alpha))(\theta) \\
&\stackrel{\text{def}}{=} \overline{(\pi \cdot_{FA} \alpha_A(id_A))}_C(\theta) \\
&\stackrel{\text{def}}{=} (F\theta)(\pi \cdot_{FA} \alpha_A(id_A)) \\
&= \pi \cdot_{FC} \pi^{-1} \cdot_{FC} (F\theta)(\pi \cdot_{FA} \alpha_A(id_A)) \\
&\stackrel{\text{def}}{=} \pi \cdot_{FC} ((\pi^{-1} \cdot_{FC} (F\theta))(\alpha_A(id_A))) \\
&= \pi \cdot_{FC} (F(\pi^{-1} \cdot_{\mathbb{C}(A,C)} \theta))(\alpha_A(id_A))) \qquad (F \text{ is equivariant}) \\
&\stackrel{\text{def}}{=} \pi \cdot_{FC} \alpha_C(\pi^{-1} \cdot_{\mathbb{C}(A,C)} \theta) \\
&\stackrel{\text{def}}{=} (\pi \cdot_{\Rightarrow_{fs}} \alpha_C)(\theta) \\
&\stackrel{\text{def}}{=} (\pi \cdot_{\int} \alpha)_C(\theta)
\end{aligned}
$$

□

We have seen that there is an enriched Yoneda isomorphism for *Nom*, but what about *FMNom* and *FMSet*? Given that *FMNom* and *FMSet* are both not complete, we cannot directly use the machinery of enriched category theory (as in the case of *Nom*) to obtain an enriched functor category and enriched Yoneda isomorphism. This, of course, does not mean that an enriched Yoneda isomorphism or embedding does not exist, because there might be other ways to enrich the functor category and Yoneda embedding. Independent of an answer to this question, there is an alternative way to deal with the absence of completeness in categories we want to enrich over:

It is possible to enlarge the universe, while preserving limits and colimits, such that the Yoneda embedding is enriched in the enlarged universe. In general, universe enlargement is applied in situations where functor categories are too big to exist as $\mathcal{V}$-categories. It allows one to interpret them as $\mathcal{V}'$-categories for a suitable enlargement $\mathcal{V}'$ of $\mathcal{V}$. Full details and more general results regarding functor categories are given in Kelly [41] (Section 3.11 and 3.12). Note that the particular universe enlargement

we would be interested in is given as a presheaf category

$$\mathcal{V}' = [\mathcal{V}^{op}, Set]$$

which is (point-wise) complete.

7.2.2 Cartesian Closure via the Enriched Yoneda Lemma

Let $\mathcal{C}$ be a small category. To prove that the functor category $[\mathcal{C}, Nom]$ is cartesian closed, the enriched Yoneda lemma for *Nom* plays an important role. This result is one of the building blocks towards a categorical conservative extension proof that we sketched in Section B.1.

Proposition 7.2.8 *The functor category* $[\mathcal{C}, Nom]$ *is cartesian closed.*

Proof We adapt the proof in ([44], Proposition 1 (p.46)) for *Nom*. For any functors $P, Q : \mathcal{C} \to Nom$, we have that $P \Rightarrow Q$ is a functor, which is defined as follows:

- for any object $C \in ob\, C$, $(P \Rightarrow Q)(C) \stackrel{\text{def}}{=} (\mathrm{Nat}(\mathcal{C}^C \times P, Q), \cdot_{\int})$[2] is a nominal set, equipped with the permutation action $(\pi \cdot_{\int} \alpha)_B \stackrel{\text{def}}{=} \pi \cdot_{\Rightarrow_{fs}} \alpha_B$.
- for any morphism $F : C \to C'$ we have a morphism $(P \Rightarrow Q)(f) : (\mathrm{Nat}(\mathcal{C}^C \times P, Q), \cdot) \to (\mathrm{Nat}(\mathcal{C}^{C'} \times P, Q), \cdot)$, which is defined as $([(P \Rightarrow Q)(f)](\alpha))_B \stackrel{\text{def}}{=} \alpha_B \circ (\mathcal{C}(C', f) \times id_{PB})$ for $\alpha \in \mathrm{Nat}(\mathcal{C}^C \times P, Q)$ and $B \in ob\, C$.

The fact that $(\mathrm{Nat}(\mathcal{C}^C \times P, Q), \cdot)$ is a nominal with $supp(\alpha) = supp(\alpha_C)$ follows analogously to the argument in Corollary 7.2.3. What remains to be shown is that

[2] The hom-functor $\mathcal{C}^C$ is trivially enriched.

the morphism is equivariant.

$$\begin{aligned}(\pi \cdot [(P \Rightarrow Q)(f)](\alpha))_B &\stackrel{\text{def}}{=} \pi \cdot ([(P \Rightarrow Q)(f)](\alpha))_B \\ &\stackrel{\text{def}}{=} \pi \cdot (\alpha_B \circ (\mathcal{C}(C', f) \times id_{PB})) \\ &= (\pi \cdot \alpha_B) \circ (\pi \cdot \mathcal{C}(C', f) \times \pi \cdot id_{PB}) \\ &= (\pi \cdot_{\int} \alpha)_B \circ (\mathcal{C}(C', f) \times id_{PB}) \qquad \text{(equivariance)} \\ &\stackrel{\text{def}}{=} ([(P \Rightarrow Q)(f)](\pi \cdot_{\int} \alpha))_B\end{aligned}$$

Next, we introduce the evaluation map $ev : (P \Rightarrow Q) \times P \Rightarrow Q$, which is a natural transformation with components $ev_C : (\mathrm{Nat}(\mathcal{C}^C \times P, Q), \cdot) \times PC \to QC$

$$ev_C(\alpha, y) \stackrel{\text{def}}{=} \alpha_C(id_C, y) \in QC$$

We have to show that the components are in *Nom*; thus equivariant.

$$\begin{aligned}\pi \cdot ev_C(\alpha, y) &\stackrel{\text{def}}{=} \pi \cdot \alpha_C(id_C, y) \\ &= (\pi \cdot \alpha_C)(\pi \cdot id_C, \pi \cdot y) \\ &\stackrel{\text{def}}{=} (\pi \cdot \alpha)_C(id_C, \pi \cdot y) \\ &\stackrel{\text{def}}{=} ev_C(\pi \cdot \alpha, \pi \cdot y)\end{aligned}$$

We continue by checking the universal property. Let $Z : \mathcal{C} \to Nom$ be an object and $\phi : Z \times P \to Q$ a morphism in $[\mathcal{C}, Nom]$. The exponential mate $\lambda(\phi) : Z \to P \Rightarrow Q$ is a natural transformation with components $\lambda(\phi)_C : ZC \to (\mathrm{Nat}(\mathcal{C}^C \times P, Q), \cdot)$, which map any $u \in ZC$ to a natural transformation with $(\lambda(\phi)_C(u))_D : \mathcal{C}(C, D) \times PD \to QD$

$$((\lambda(\phi)_C)(u))_D(f, y) \stackrel{\text{def}}{=} \phi_D(Z(f)(u), y)$$

for $D \in ob\, C$, $f : C \to D$ and $y \in PD$. The fact that $\lambda(\phi)_C(u)$ is indeed a natural transformation follows as in the case of *Set*. Hence, we have that $\lambda(\phi)_C(u) \in \mathrm{Nat}(\mathcal{C}^C \times P, Q)$. The same is the case for $\lambda(\phi)$. So, to show that $\lambda(\phi)$ is well defined, it remains to show that the components of $\lambda(\phi)$ are morphisms in *Nom* (equivariant

functions).

$$
\begin{aligned}
& [\pi \cdot (\lambda(\phi)_C(u))]_D(f,y) \\
\stackrel{\text{def}}{=}\ & (\pi \cdot [\lambda(\phi)_C(u)]_D)(f,y) \\
\stackrel{\text{def}}{=}\ & \pi \cdot ([\lambda(\phi)_C(u)]_D)(\pi^{-1} \cdot f, \pi^{-1} \cdot y) \\
\stackrel{\text{def}}{=}\ & \pi \cdot \phi_D(Z(\pi^{-1} \cdot f)(u), \pi^{-1} \cdot y) \\
=\ & \phi_D(\pi \cdot Z(\pi^{-1} \cdot f)(u), y) && (\phi_D \text{ is equivariant}) \\
=\ & \phi_D(Z(\pi^{-1} \cdot f)(\pi \cdot u), y) && (Z(\pi \cdot f) \text{ is equivariant}) \\
=\ & \phi_D(Z(f)(\pi \cdot u), y) && (\mathcal{C}(C,D) \text{ is trivially enriched}) \\
\stackrel{\text{def}}{=}\ & (\lambda(\phi)_C(\pi \cdot u))_D(f,y)
\end{aligned}
$$

□

7.3 Internal Yoneda isomorphisms in *FMSet*

As we have pointed out in the previous section, there is no enriched Yoneda isomorphism for *FMNom* and *FMSet*, which is due to the fact that both categories are not complete. Motivated by the fact that both categories are internally (co)complete, we got interested in determining if there is an internal variant of a Yoneda isomorphism for *FMSet* and *FMNom*.

The basic notions of internal category theory are recalled in Appendix A.2. Let $\mathbf{C}$ be an internal category in *FMSet* with structure $(\mathbf{C}_0, \mathbf{C}_1, d_0, d_1, i, c)$. For any object $A \in \mathbf{C}_0$, the internal *FMSet*-valued hom functor $\mathbf{H}^A : \mathbf{C} \to \mathit{FMSet}$ is specified by the tuple $(d_0^{-1}(A), \hat{d}_1, \hat{c})$ where

$$
\begin{aligned}
\hat{d}_1 &\stackrel{\text{def}}{=} d_1 \mid_{d_0^{-1}(A)} : d_0^{-1}(A) \to \mathbf{C}_0 \\
\hat{c} &\stackrel{\text{def}}{=} c \mid_{\mathbf{C}_1 \times_{\mathbf{C}_0} d_0^{-1}(A)} : \mathbf{C}_1 \times_{\mathbf{C}_0} d_0^{-1}(A) \to d_0^{-1}(A)
\end{aligned}
$$

In particular, we have

- $\hat{d}_1(f) \stackrel{\text{def}}{=} X$ for $f : A \to X \in d_0^{-1}(A)$
- $\hat{c}(f, g) \stackrel{\text{def}}{=} f \circ g$ for $f : X \to Y \in \mathbf{C}_1$ and $g : A \to X \in d_0^{-1}(A)$

We need to show that $\mathbf{H}^A$ is well defined. By definition we have that $d_0^{-1}(A) \subseteq \mathbf{C}_1$. Then, given that $\mathbf{C}_1$ is an FM-Set, we have that all elements of $d_0^{-1}(A)$ are finitely supported. It can easily be seen that $d_0^{-1}(A)$ is finitely supported by $supp(\mathbf{C}_0) \cup supp(A)$. Hence, we have that $d_0^{-1}(A)$ is an FM-Set. Further, the morphisms are finitely supported by $supp(d_1) \cup supp(d_0^{-1}(A))$ and $supp(c) \cup supp(d_0^{-1}(A))$. The axioms follow by trivial computations. Thus, we have that $\mathbf{H}^A$ is an internal *FMSet*-valued functor.

Given that *FMSet* has pullbacks, it can be deduced that the functor category $[\mathbf{C}, FMSet]$ with internal *FMSet*-valued functors as objects and internal natural transformations as arrows is a category, but it is not internal in *FMSet*. Due to the fact that *FMSet* is not small, the functor category cannot be small either and therefore cannot be internalised into *FMSet*. Thus, we clearly do not have an internal Yoneda embedding, but there is an internal Yoneda lemma in *FMSet*, as we will now demonstrate. This also holds for *FMNom* (with minor modifications).

Lemma 7.3.1 (Internal Yoneda Lemma in *FMSet*) *Let $\mathbf{C}$ be an internal category in FMSet, $F : \mathbf{C} \to FMSet$ be an FMSet-valued internal functor which is represented by (P, p_0, p_1) and $A \in \mathbf{C}_0$. Then for $FA \stackrel{\text{def}}{=} p_0^{-1}(A)$ there exists an isomorphism*

$$\Phi : FA \cong_{FMSet} [\mathbf{C}, FMSet](\mathbf{H}^A, F) : \Psi$$

where Φ and Ψ are defined as follows:

- $x \in FA \mapsto \Phi(x) : d_0^{-1}(A) \to P$ *such that for any* $f \in d_0^{-1}(A)$, $\Phi(x)(f) \stackrel{\text{def}}{=} p_1(f, x)$
- $\alpha \in [\mathbf{C}, FMSet](\mathbf{H}^A, F) \mapsto \Psi(\alpha) \stackrel{\text{def}}{=} \alpha(i(A)) \in FA$

Moreover, we have that the isomorphism is natural in A and F.

Proof Given that $FA \subseteq P$ and P is an FM-set, we have that all elements in FA are finitely supported. We further have that $supp(FA) = supp(p_0) \cup supp(A)$. So, FA is an FM-Set. The elements of the set $[\mathbf{C}, FMSet](\mathbf{H}^A, F)$ are finitely supported by definition and its support is $supp(\mathbf{H}^A) \cup supp(F)$.

We continue by showing that Φ and Ψ are well defined maps. Let $x \in FA$. We demonstrate that $\Phi(x)$ is an internal natural transformations between $\mathbf{H}^A$ and F:

Suppose $f : A \to X \in d_0^{-1}(A)$ for some $X \in \mathbf{C}_0$. Recall that $p_1 : \mathbf{C}_1 \times_{\mathbf{C}_0} P \to P$. We have that $d_0^{-1}(A) \subseteq \mathbf{C}_1$ and therefore $f \in \mathbf{C}_1$, $x \in FA \stackrel{\text{def}}{=} p_0^{-1}(A) \subseteq P$ and $\Phi(x)(f) \in P$. So, we have that $\Phi(x) : d_0^{-1}(A) \to P$ is a well defined morphism. Next, we demonstrate that $\Phi(x)$ is finitely supported by $supp(p_1) \cup supp(x)$: Let $\pi \# (p_1, x)$.

$$\begin{aligned}
(\pi \cdot_{\Rightarrow_{fs}} \Phi(x))(f) &\stackrel{\text{def}}{=} \pi \cdot_P \Phi(x)(\pi^{-1} \cdot_{d_0^{-1}(A)} f) \\
&\stackrel{\text{def}}{=} \pi \cdot_P p_1(\pi^{-1} \cdot_{d_0^{-1}(A)} f, x) \\
&= (\pi \cdot p_1)(f, \pi \cdot_{p_0^{-1}(A)} x) \\
&= p_1(f, x) && (\pi \# (p_1, x)) \\
&\stackrel{\text{def}}{=} \Phi(x)(f)
\end{aligned}$$

Hence, we have that $\Phi(x)$ is a morphism in *FMSet*. We continue by showing that $\Phi(x)$ satisfies the axioms for an internal natural transformation, which follows similarly to *Set*. Hence, we have that Φ is well defined.

Let $\alpha \in [\mathbf{C}, FMSet](\mathbf{H}^A, F)$. So, we have that $\alpha : d_0^{-1}(A) \to P$ is a morphism in *FMSet*. Further, we have $i : \mathbf{C}_0 \to \mathbf{C}_1$ with $i(A) : A \to A$ being the identity on A. We have that $i(A) \in d_0^{-1}(A)$ and by axiom (i) of the internal natural transformation α we have that $\alpha(i(A)) \in p_0^{-1}(A) \stackrel{\text{def}}{=} FA$. Hence, Ψ is well defined as well.

We now demonstrate by standard computations that Φ and Ψ are both finitely supported morphisms. Let $\pi \# (p_1, FA, [\mathbf{C}, FMSet](\mathbf{H}^A, F))$, $x \in FA = p_0^{-1}(A)$ and

$f : A \to X \in d_0^{-1}(A)$.

$$\begin{aligned}((\pi \cdot \Phi)(x))(f) &\stackrel{\text{def}}{=} (\pi \cdot \Phi(\pi^{-1} \cdot x))(f) \\ &\stackrel{\text{def}}{=} \pi \cdot \Phi(\pi^{-1} \cdot x)(\pi^{-1} \cdot f) \\ &\stackrel{\text{def}}{=} \pi \cdot p_1(\pi^{-1} \cdot x, \pi^{-1} \cdot f) \\ &= (\pi \cdot p_1)(x, f) \\ &= p_1(x, f) && (\pi \# p_1) \\ &\stackrel{\text{def}}{=} \Phi(x)(f)\end{aligned}$$

Let $\pi \# (i, A, [\mathbf{C}, \mathit{FMSet}](\mathbf{H}^A, F))$ and $\alpha \in [\mathbf{C}, \mathit{FMSet}](\mathbf{H}^A, F)$. Then we have

$$\begin{aligned}(\pi \cdot \Psi)(\alpha) &\stackrel{\text{def}}{=} \pi \cdot \Psi(\pi^{-1} \cdot \alpha) \\ &\stackrel{\text{def}}{=} (\pi^{-1} \cdot \alpha)(i(A)) \\ &= \alpha(\pi \cdot i(A)) \\ &= \alpha((\pi \cdot i)(\pi \cdot A)) \\ &= \alpha(i(A)) && (\pi \# (i, A)) \\ &\stackrel{\text{def}}{=} \Psi(\alpha)\end{aligned}$$

Hence, both are morphism in *FMSet*. The fact that Φ and Ψ are mutually inverse and the isomorphism is natural in F and A follows as in the ordinary Yoneda lemma. □

7.4 Finitely supported (co)limits

We have shown that *FMSet* and *FMNom* are neither complete nor cocomplete. In the respective counter examples we constructed an infinite family of morphisms $(f_i \mid i \in \omega)$ with an infinite support $(\bigcup_{i\in\omega} \mathit{supp}(f_i) = \bigcup_{i\in\omega}\{a_i\} = \mathbb{A})$. In this section we introduce notions of "finitely supported" (co)cones and (co)limit for which *FMNom* and *FMSet* are (co)complete. This was developed in connection with the question if *FMSet* and *FMNom* are (co)complete.

Definition 7.4.1 *Let $\mathcal{A}$ be a small category. A **fs-diagram** is a functor $F : \mathcal{A} \to FMSet$ such that*

(i) $\bigcup_{I \in ob\,\mathcal{A}} supp(FI)$ *is finite*

(ii) $\bigcup_{f \in mor\,\mathcal{A}} supp(Ff)$ *is finite*

Definition 7.4.2 *Given a diagram $F : \mathcal{A} \to FMSet$, a **fs-cone** on F consists of*

(i) an object $C \in FMSet$

(ii) for every object $I \in \mathcal{A}$, a morphism $p_I : C \to FI$ in FMSet such that for every morphism $u : I \to I'$ in $\mathcal{A}$, $p_{I'} = Fu \circ p_I$.

(iii) $\bigcup_{I \in ob\,\mathcal{A}} supp(p_I)$ *is finite.*

Definition 7.4.3 *Given a fs-diagram $F : \mathcal{A} \to FMSet$, a **fs-cocone** on F consists of*

(i) an object $C \in FMSet$

(ii) for every object $I \in ob\,\mathcal{A}$, a morphism $s_I : FI \to C$ in FMSet such that for every morphism $u : I' \to I$ in $\mathcal{A}$, $s_{I'} = s_I \circ Fu$.

(iii) $\bigcup_{I \in ob\,\mathcal{A}} supp(s_I)$ *is finite.*

Definition 7.4.4 *Given a fs-diagram $F : \mathcal{A} \to FMSet$, a **fs-limit** of F is a fs-cone $(L, (p_I)_{I \in ob\,\mathcal{A}})$ on F such that for every fs-cone $(M, (q_I)_{I \in ob\,\mathcal{A}})$ on F, there exists a unique morphism $h : M \to L$ in FMSet such that for every object $I \in ob\,\mathcal{A}$, $q_I = p_I \circ h$.*

Definition 7.4.5 *Given a fs-diagram $F : \mathcal{A} \to FMSet$, a **fs-colimit** of F is a fs-cocone $(L, (s_I)_{I \in \mathcal{A}})$ on F such that for every fs-cocone $(M, (t_I)_{I \in ob\,\mathcal{A}})$ on F, there exists a unique morphism $h : L \to M$ in FMSet such that for every object $I \in ob\,\mathcal{A}$, $t_I = h \circ s_I$.*

We now demonstrate that *FMNom* and *FMSet* are fs-complete and fs-cocomplete.

Theorem 7.4.6 *FMSet and FMNom have all fs-limits.*

Proof We enhance the proof argument that was originally used to demonstrate that *Set* is complete with "nominal" reasoning steps. Let $F : \mathcal{A} \to FMSet$ be a fs-diagram[3]. We define $L \subseteq \prod_{I \in ob\,\mathcal{A}} FI$ to be the set of finitely supported tuples $(x_I)_{I \in ob\,\mathcal{A}}$ with respect to the point-wise permutation action

$$\pi \cdot (x_{I_1}, x_{I_2}, ...) \stackrel{\text{def}}{=} (\pi \cdot x_{I_1}, \pi \cdot x_{I_2}, ...)$$

which satisfy the condition

$$(\forall u : I \to I') \quad Fu(x_I) = x_{I'} \qquad (\diamond)$$

Let p_I be the usual projection maps, which are clearly equivariant. We now demonstrate that $(L, p_I : L \to FI)_{I \in ob\,\mathcal{A}}$ is the fs-limit of F in *FMSet*:

We begin by demonstrating that it is an fs-cone. The fact that L is an FM-set can directly be deduced from condition (i) of an fs-diagram and the fact that L contains only finitely supported tuples. The commuting condition follows directly from condition $(\diamond)$ of L. Given that the projection maps are equivariant we have an fs-cone.

For the universal property we suppose that $(Z, q_I : Z \to FI)_{I \in ob\,\mathcal{A}}$ is a fs-cone. The "mediating" function $h : Z \to L$ is defined as follows:

$$h(y) \stackrel{\text{def}}{=} (q_I(y))_{I \in ob\,\mathcal{A}}$$

We have to show that h is well defined, i.e. $h(y) \in L$. It follows by construction that $h(y)$ is an element of $\prod_{I \in ob\,\mathcal{A}} FI$ and the condition $(\diamond)$ follows directly from the fact that $(Z, q_I : Z \to FI)_{I \in ob\,\mathcal{A}}$ is a fs-cone (commuting condition). Further, we know that $supp(q_I(y)) \subseteq supp(q_I) \cup supp(y)$ and that all the maps q_I's share a common support. Hence, $h(y)$ is finitely supported tuple and therefore $h(y) \in L$

[3]Note that we do not require condition (ii) of an fs-diagram in this proof

What remains to be shown is that h is finitely supported map. Let $z, z' \# (Z, L, \bigcup_{I \in ob\,\mathcal{A}} supp(q_I))$. Given that $(Z, q_I : Z \to FI)_{I \in ob\,\mathcal{A}}$ is a fs-cone we have that $\bigcup_{I \in ob\,\mathcal{A}} supp(q_I)$ is finite.

$$\begin{aligned}
((z\,z') \cdot h)(y) &\stackrel{\text{def}}{=} (z\,z') \cdot h((z\,z') \cdot y) \\
&\stackrel{\text{def}}{=} (z\,z') \cdot (q_I((z\,z') \cdot y))_{I \in ob\,\mathcal{A}} \\
&\stackrel{\text{def}}{=} ((z\,z') \cdot q_I((z\,z') \cdot y))_{I \in ob\,\mathcal{A}} \\
&\stackrel{\text{def}}{=} (((z\,z') \cdot q_I)(y))_{I \in ob\,\mathcal{A}} \\
&= (q_I(y))_{I \in ob\,\mathcal{A}} \qquad\qquad (z, z' \# \bigcup_{I \in ob\,\mathcal{A}} supp(q_I)) \\
&\stackrel{\text{def}}{=} h(y)
\end{aligned}$$

The argument for *FMNom* follows analogously. But we clearly do not require condition (i) of an fs-diagram. □

Theorem 7.4.7 *FMSet and FMNom has all fs-colimits.*

Proof We enhance the proof argument that was originally used to demonstrate that *Set* is cocomplete with nominal reasoning steps. Let $F : \mathcal{A} \to FMSet$ be a fs-diagram. We define $C \stackrel{\text{def}}{=} \coprod_{I \in ob\,\mathcal{A}} FI \Big/ R$ where $\coprod_{I \in ob\,\mathcal{A}} FI$ denotes a disjoint union and R is an equivalence relation defined to be the reflexive, symmetric and transitive closure of $\sim \subseteq \coprod_{I \in ob\,\mathcal{A}} FI \times \coprod_{I \in ob\,\mathcal{A}} FI$, which is defined as:

$$(I', x') \sim (I, x) \iff x' = F(u)(x) \text{ for } u : I \to I'$$

Further, we define functions $p_I : FI \to C$:

$$p_I(x) \stackrel{\text{def}}{=} [incl_I(x)]_R$$

where $incl_I(x) \stackrel{\text{def}}{=} (I, x)$ is the inclusion function of $\coprod_{I \in ob\,\mathcal{A}} FI$. We first demonstrate that $(C, p_I : FI \to C)$ is fs-cocone: The infinite disjoint union $\coprod_{I \in ob\,\mathcal{A}} FI$, equipped with the permutation action $\pi \cdot (I, y) \stackrel{\text{def}}{=} (\pi \cdot I, \pi \cdot_{FI} y)$, contains only finitely

supported elements. Further, under condition (i) of an fs-diagram, we have that $\coprod_{I \in ob\,\mathcal{A}} FI$ is finitely supported and therefore $\coprod_{I \in ob\,indexcat} FI$ is an FM-set.

Next, we demonstrate that R is a finitely supported subset of $\coprod_{I \in ob\,\mathcal{A}} FI \times \coprod_{I \in ob\,\mathcal{A}} FI$: Let S_0 and S_1 be the finite sets corresponding to conditions (i) and (ii) of the fs-diagram F. Suppose $(y, y') \in R$ and $c, c' \# S_0, S_1$. We have to show that $((c\,c') \cdot y, (c\,c') \cdot y') \in R$. We only consider the base case of R, namely $\sim$. Suppose, we have $y = (I, x)$, $y' = (I', x')$ and $u : I \to I'$ such that $x' = Fu(x)$. From this, we can directly deduce, using the fact that $c, c' \# S_0$, that $(c\,c') \cdot x \in (c\,c') \cdot FI = FI$ and $(c\,c') \cdot x' \in (c\,c') \cdot FI' = FI'$. So, we have that $(I, (c\,c') \cdot x), (I', (c\,c') \cdot x') \in \coprod_{I \in ob\,\mathcal{A}} FI$. Further, using the fact that $c, c' \# S_1$ and $x' = Fu(x)$, we can deduce that

$$(c\,c') \cdot x' = (c\,c') \cdot (Fu)(x) = ((c\,c') \cdot (Fu))((c\,c') \cdot x) = F(u)((c\,c') \cdot x)$$

and therefore we have that

$$(c\,c') \cdot (I, x) \stackrel{\text{def}}{=} (I, (c\,c') \cdot x) \sim (I', (c\,c') \cdot x') \stackrel{\text{def}}{=} (c\,c') \cdot (I', x')$$

From the fact that R is a finitely supported equivalence relation and $\coprod_{I \in ob\,\mathcal{A}} FI$ is an FM-set, we can deduce that the quotient C is an FM-Set with the permutation action $\pi \cdot [x]_R \stackrel{\text{def}}{=} [\pi \cdot x]_{\pi \cdot R}$ with $supp([x]_R) \subseteq supp(x) \cup supp(R)$. C itself is finitely supported by $supp(\coprod_{I \in A_0} FI) \cup supp(R)$. Next, we have to demonstrate that p_I is finitely supported. Let $c, c' \# S_0, S_1$. Then

$$\begin{aligned}
((c\,c') \cdot p_I)(x) &\stackrel{\text{def}}{=} (c\,c') \cdot (p_I((c\,c') \cdot x)) \\
&\stackrel{\text{def}}{=} (c\,c') \cdot [incl_I((c\,c') \cdot x)]_R \\
&\stackrel{\text{def}}{=} [(c\,c') \cdot incl_I((c\,c') \cdot x)]_{(c\,c') \cdot R} \\
&= [(c\,c') \cdot incl_I((c\,c') \cdot x)]_R && (\pi \# S_0, S_1) \\
&\stackrel{\text{def}}{=} [(\pi \cdot incl_I)(x)]_R \\
&= [incl_I(x)]_R && (c, c' \# S_0) \\
&\stackrel{\text{def}}{=} p_I(x)
\end{aligned}$$

Due to the fact that we have $supp(p_I) \subseteq S_0 \cup S_1$ for all $I \in ob\,\mathcal{A}$, they all share a common support. The commuting condition follows as in the case for *Set*. Hence, we have that $(C, p_I : FI \to C)$ is fs-cocone.

For the universal property we suppose that $(Z, q_I : FI \to Z)_{I \in ob\,\mathcal{A}}$ is a cocone. We define $h : C \to Z$ as follows: $h([x]_R) \stackrel{\text{def}}{=} [q_{I_1}, q_{I_2}, ...](x)$. We have to show that h is well defined, i.e. it is independent of the choice of the representative of the equivalence class. This follows immediately from the definition of R and the fact that $(Z, q_I : FI \to Z)_{I \in ob\,\mathcal{A}}$ is a fs-cocone. What remains to be proved is that h is finitely supported. Let $z, z' \,\#\, (R, \bigcup_{I \in ob\,\mathcal{A}}(supp(q_I))$.

$$
\begin{aligned}
((z\,z') \cdot h)([(I,x)]_R) &\stackrel{\text{def}}{=} (z\,z') \cdot h((z\,z') \cdot [(I,x)]_R) \\
&\stackrel{\text{def}}{=} (z\,z') \cdot h([(z\,z') \cdot (I,x)]_{(z\,z')\cdot R}) \\
&\stackrel{\text{def}}{=} (z\,z') \cdot h([(I,(z\,z') \cdot x)]_{(z\,z')\cdot R}) \\
&= (z\,z') \cdot h([(I,(z\,z') \cdot x)]_R) && (z, z' \,\#\, R) \\
&\stackrel{\text{def}}{=} (z\,z') \cdot q_I((z\,z') \cdot x) \\
&\stackrel{\text{def}}{=} ((z\,z') \cdot q_I)(x) \\
&= q_I(x) && (z, z' \,\#\, q_I) \\
&\stackrel{\text{def}}{=} h([(I,x)]_R)
\end{aligned}
$$

The argument for *FMNom* follows analogously, but we do not need condition (i) of an fs-diagram. □

Chapter 8

Conclusions

In this chapter we provide a summary of the contributions made in this thesis and discuss future lines of research and related work.

NLC: A Nominal Lambda Calculus

We have introduced NLC as an extension of the typed lambda calculus which incorporates ideas and concepts from NEL. A key novelty of NLC is the introduction of name-dependent function types, which are directly motivated by NEL with the initial goal in mind to extend the categorical type theory correspondence of NEL. As we have seen, these types have a major impact on the definition of NLC, most importantly the type and equation system, and ultimately the model-theoretic semantics.

With respect to the meta-theory of NLC raw terms, we had to extend various standard results of NEL and $\lambda^{\to}$. Note that unlike NEL, we introduced the object-level and meta-level permutation action on raw terms instead of well typed terms. This is a subtle change, which provides us with a slightly neater presentation.

An interesting aspect of NLC is its type and equation system, which we believe to be intuitive regarding the origin of the individual rules. However, there is one important aspect to consider, namely that both systems are mutual inductively defined.

This is due to the fact that name-dependent function types are used. So, the type system with its freshness judgements is actually "forced" upon us since the typing of domain (freshness) restricted lambda abstractions is mutually tied to hypotheses of freshness assertions. We have found working with a dependently typed system interesting and at times challenging to ensure that our definitions lead to a consistent system with a cleanly defined semantics. In the end, we have succeeded in recovering many of the standard properties of $\lambda^{\rightarrow}$ for NLC. Furthermore, we demonstrated that NLC is actually a conservative extension of NEL and $\lambda^{\rightarrow}$.

An appealing future line of research might be to study a variant of NLC that uses "proper" freshness restriction types of the form $s^{\overline{a}}$. However, an important aspect that would need to be considered is that while NLC already has some "flavour" of subtyping, considering the rule (AP), we would need a proper subtyping relation, like

$$s^{\overline{a}} \preceq s^{\overline{b}} \quad \text{if} \quad \overline{b} \subseteq \overline{a}$$

Further, for a more immediate extension of NLC, other simple types, like unit or product types could be considered.

On a more procedural note we have taken great care over our proofs, attempting to ensure that we do not fall into any of the typical traps related to renaming. In the construction of NLC we initially followed the nominal approach and introduced a variable swapping permutation action, which allowed us to define α-equivalence for raw terms without a recourse to capture avoiding substitution. We believe that this improves the presentation of various definitions and results. But to ultimately prove that operations on syntax, like permutation actions and capture avoiding substitution are independent of the choice of variables (names), we deviated from the formal framework of nominal sets and used a more traditional approach. In retrospect, it would have been more elegant to use the notions of α-structural induction and α-structural recursion of nominal techniques [54], but formalisation of results was not the prime objective of this thesis. Nevertheless, we believe that a formalisation of

these results in an automated theorem prover, using a nominal framework, may be interesting in its own right.

NLC[$\mathbb{A}$]: Name Abstraction and Concretion in NLC

The starting point was the observation that concretion can be captured in NLC by using name-dependent function types and freshness assertions in typing rules. We then introduced NLC[$\mathbb{A}$] as an extension of NLC which captures name abstraction and concretion as first class citizen and proved various properties. It is interesting to consider that our approach is in some sense contrary to an approach applied for nominal rewriting systems in [28], where name abstraction is defined first and is later utilised to model higher-order functions involving local state.

Based on pure NLC[$\mathbb{A}$] (without constants) we introduced a $\beta\eta$-reduction system and proved it to be strongly normalising and confluent. This is not only interesting in its own right, but it also allowed us to demonstrate that provable equality for NLC[$\mathbb{A}$] (NLC) is decidable, as well as type checking and type inference. Moreover, regarding future work, a confluent $\beta\eta$-reduction system for NLC might be used to prove Conjecture 3.5.3 (conservative extension property).

Based on the possible extensions of NLC that we have previously proposed, we will now provide various observations on possible effects such extensions would have on the $\beta\eta$-reduction system:

- There is the proposal to introduce "proper" freshness restriction types of the form $s^{\overline{a}}$. As previously pointed out, this would require some notion of subtyping. In this context, we recall that in [58] two classes of subtyping relations were studied with respect to $\beta\eta$-reduction systems for $\lambda^{\rightarrow}$. The idea would be to apply these results and observations on the suggested subtyping relation for NLC; and indeed the proposed subtyping relation does fall into one of the classes that were analysed in [58]. So, this would suggest the following: While

the $\beta\eta$-reduction system would remain strongly normalising, it would not be confluent anymore. However, one would still obtain decidability for provable equality as remarked in [58].

- An obvious extension is the introduction of other simple types, like product or unit types. While product types follow immediately, we would have to be careful with unit types, because η-expansion needs to be used instead of η-contraction. As can easily be seen, the results of strong normalisation and confluence for NLC[$\mathbb{A}$] would follow, with only minor modification, from the proof arguments provided in [39].

To conclude this section, we discuss how NLC[$\mathbb{A}$] relates to SNTT [9] and the $\lambda\alpha\nu$-calculus [55], two alternative extensions of the typed lambda calculus that capture name abstraction and concretion. What can immediately be observed is that the properties that were studied for all three calculi are very much in line. However, the techniques that were used to capture name abstraction and concretion, and the respective proof details are quite different. The general aim of NLC and SNTT is to internalise freshness side conditions. In SNTT this is achieved by using a notion of bunched contexts, a concept which originates from bunched logic. In bunched contexts, apart from variable typing, additional information about object-level freshness is provided. This is similar to NLC[$\mathbb{A}$], where we use freshness contexts. However, the presentation of freshness contexts and how object-level freshness is used in NLC[$\mathbb{A}$] is quite different to SNTT. A close similarity of both calculi is that concretion is captured as a partial function and well definedness is guaranteed by an internalised freshness condition. A direct consequence of internalised freshness conditions is that SNTT and NLC[$\mathbb{A}$] need to modify the ordinary typed lambda calculus quite extensively. In comparison, the $\lambda\alpha\nu-$calculus is less syntactically "tedious", because it has a conventional meta-theory of raw terms and type system. This is sufficient to capture name abstraction and concretion, because concretion is total due to the use

of a local scoping operator. At this point, it can of course be argued that SNTT and NLC[$\mathbb{A}$] do not require a local scoping operator to capture concretion, which would justify the extra effort. Moreover, the introduction of local scoping adds some extra level of complexity that also has to be dealt with.

Towards Completeness and a Categorical Type Theory Correspondence for NLC

With the aim of obtaining a sound and complete categorical semantics and a categorical type theory correspondence for NLC, we extended the notion of an FM-category with equivariant exponentials. This notion does indeed underpin our categorical semantics of NLC, which we have proven sound. We then emphasised that the key step in obtaining completeness and a correspondence result for NLC is the construction of a syntactically generated classifying FM-ccc, $Cl(Th)$, which is usually defined as a quotient construction using the syntax of NLC. Next, we observed that NLC is not expressive enough to construct exponentials in $Cl(Th)$. We attributed this to the limited type system of NLC. To circumvent this issue, we suggested two approaches towards extending NLC such that exponentials can be constructed in a syntactically generated classifying category $Cl(Th)$ for such an extended calculus.

The approach that we have ultimately pursued is motivated by the observation that exponentials in *FMSet* can be presented in a way that is much closer to what NLC can already express. The underlying idea was to extend NLC such that we can mimic this presentation of an exponential (in *FMSet*) in a syntactically generated classifying category $Cl(Th)$ (for any theory Th of the extended calculus). Hence, we introduced [И]NLC, which is a semantically motivated extension of ИNLC and defined a sound categorical semantics in *FMSet*. Note that ИNLC was introduced beforehand to independently investigate how local fresh atomic names can be captured in NLC. By using the additional syntactic structures of [И]NLC we succeeded in constructing an

equivariant exponential in $Cl(Th)$ for any [M]NLC-theory Th. This is an important step, but there is more work required, which is not part of this thesis. It will be pursued as future work. Thus, the ultimate goal is to use the new "framework" to show that for every [M]NLC-theorem Th, there exits a classifying category $Cl(Th)$ that is a [M]FM-ccc.

Completeness of $\beta\eta$-conversion for $\mathcal{N}$ in $\lambda^{\to}$

In recent years various "nominal" typed lambda calculi were introduced and analysed in the literature. A property of such a calculus that to our knowledge has not yet been studied is completeness of $\beta\eta$-conversion for a nominal analogue of full set-theoretic hierarchies. We believe that our contributions in Chapter 6 constitute a first step in this direction.

We introduced the notion of a full nominal hierarchy $\mathcal{N}$ and demonstrated that it is a Henkin model. To investigate completeness of $\beta\eta$-conversion for such models we have chosen a simplified problem statement as our starting point. More precisely, we have asked the question if $\beta\eta$-conversion is complete for full nominal hierarchies in $\lambda^{\to}$. We justified that this is a valid problem to be pursued, and moreover, it is not just a means to an end, but an interesting problem in its own right. Apart from this, it isolates rather well a problem of completeness proofs via logical relations that one encounters in the context of nominal sets.

We first provided an overview of the logical relation based proof argument and identified the "Surjective Lemma" as a key hurdle towards a completeness theorem for full nominal hierarchies. We then observed that the argument relies on the axiom of choice to prove that $\mathcal{R}$ is a partial logical function. To circumvent the related issues, we introduced finitely supported embeddings and inductively generated finitely supported function to prove that $\mathcal{R}$ is a partial logical function. But there is a "price" to be paid, because the condition is rather restrictive and impairs the usability of such

a completeness theorem. Nevertheless, despite its limited applicability, the result is useful to shed some light on logical relation based completeness theorems for full nominal models. We believe that it would be interesting to further investigate if there are non-trivial nominal sets that can be proven complete or if there is a way to strengthen the result.

We proposed Statman's 1-section theorem as an alternative route towards completeness for full nominal hierarchies. An immediate advantage of the 1-section theorem is its rather straightforward combinatorial condition on Henkin models. Due to the fact that we have restricted ourselves to pure $\lambda^{\rightarrow}$ and full nominal hierarchies are Henkin models, we were able to directly apply the 1-section theorem. We then showed that the condition of the 1-section theorem is satisfied for full nominal hierarchies, and therefore succeeded in proving completeness of $\beta\eta$-conversion for full nominal hierarchies in $\lambda^{\rightarrow}$.

Considering the next step, namely to obtain a completeness theorem for full nominal hierarchies in a nominal lambda calculus, like NLC, we believe that the most promising route, as of now, would be to adapt Statman's 1-Section theorem.

Enriched/Internal Yoneda Isomorphisms

In connection to our goal to obtain a categorical type theory correspondence for NLC, we pursed another goal, namely to prove an additional conservative extension property (Conjecture 3.5.3) using a categorical proof argument (see Appendix B). As we have recalled, an important component of the original categorical proof argument for EL and $\lambda^{\rightarrow}$, as presented in Crole [20] (Section 4.10), is the Yoneda isomorphism. Hence, to extend the proof argument to hold for NEL and NLC (or an extension of NLC) we had to study enriched/internal Yoneda isomorphisms in more detail.

We reminded the reader that *Nom* is (co)complete and demonstrated that *FMSet* and *FMNom* are finitely (co)complete, but not (co)complete. Based on this, we

obtained weak enriched Yoneda lemmas for all three nominal/FM categories and a strong enriched Yoneda isomorphism (embedding) for *Nom*. Due to the very "fine grained" computations that need to be performed in the categorical proof argument, we had to unravel the rather abstract results to be able to directly work with the permutation actions involved. This provided us with some interesting insights.

Moreover, we proved an internal Yoneda lemma for *FMSet* and showed that *FMSet* is (co)complete for a newly introduced notion of finitely supported (co)limits.

Appendix A

Enriched and Internal Category Theory

A.1 Enriched Category Theory

We recall the basic notions of enriched category theory, which are necessary to discuss enriched Yoneda isomorphisms, as presented in [41, 6].

Definition A.1.1 *A **monoidal category** is a category $\mathcal{V}$ equipped with*

(i) a bifunctor $\otimes : \mathcal{V} \times \mathcal{V} \to \mathcal{V}$, called the tensor product. The image of the pair (A, B) under $\otimes$ is written as $A \otimes B$.

(ii) an object $I \in \mathcal{V}$, called the unit

(iii) an "associativity" natural isomorphism α with components $\alpha_{A,B,C} : (A \otimes B) \otimes C \cong A \otimes (B \otimes C)$

(iv) "left unit" and "right unit" natural isomorphism l and r with components $l_A : I \otimes A \cong A$ and $r_A : A \otimes I \cong A$ respectively.

(v) the "associativity" coherence condition for any objects A, B, C, D:

$$(id_A \otimes \alpha_{A,B,C}) \circ \alpha_{A,B\otimes C,D} \circ (\alpha_{A,B,C} \otimes id_D) = \alpha_{A,B,C\otimes D} \circ \alpha_{A\otimes B,C,D}$$

(vi) and the unit coherence condition for any objects A, B:

$$(id_B \otimes l_B) \circ \alpha_{A,I,B} = r_A \otimes id_B$$

We assume from here onwards that all monoidal categories $\mathcal{V}$ are locally small. By fixing a monoidal category $\mathcal{V}$, the notions of category, functor and natural transformation can be enriched over $\mathcal{V}$ as follows:

Definition A.1.2 *Let $\mathcal{V}$ be a monoidal category. A **$\mathcal{V}$-enriched category** $\mathbb{C}$ consists of*

(i) a class $ob(\mathbb{C})$ of "objects" of $\mathbb{C}$

(ii) for every pair $A, B \in ob(\mathbb{C})$ an object $\mathbb{C}(A, B) \in \mathcal{V}$

(iii) for any objects $A, B, C \in ob(\mathbb{C})$, a "composition" morphism $c_{A,B,C} : \mathbb{C}(A, B) \otimes \mathbb{C}(B, C) \to \mathbb{C}(A, C)$ (in $\mathcal{V}$)

(iv) for any object $A \in ob(\mathbb{C})$, an "identity" morphism in $\mathcal{V}$, $j_A : I \to \mathbb{C}(A, A)$

(v) the "associativity" coherence condition for any objects A, B, C, D, E:

$$c_{A,C,D} \circ (c_{A,B,C} \otimes id_{\mathbb{C}(C,D)}) = c_{A,B,D} \circ (id_{\mathbb{C}(A,B)} \otimes c_{B,C,D}) \circ \alpha_{\mathbb{C}(A,B),\mathbb{C}(B,C),\mathbb{C}(C,D)}$$

(vi) the "unit" coherence condition for any objects A, B:

$$id_{\mathbb{C}(A,B)} l_{\mathbb{C}(A,B)} = c_{A,A,B} \circ (j_A \otimes id_{\mathbb{C}(A,B)})$$
$$c_{A,B,B} \circ (id_{\mathbb{C}(A,B)}) \otimes j_B = id_{\mathbb{C}(A,B)} \circ r_{\mathbb{C}(A,B)}$$

A $\mathcal{V}$-enriched category $\mathbb{A}$ is called small if $ob(\mathbb{A})$ is a set.

Definition A.1.3 *Let $\mathcal{V}$ be a monoidal category. Given $\mathcal{V}$-enriched categories $\mathbb{C}$ and $\mathbb{D}$, a **$\mathcal{V}$-enriched functor** $F : \mathbb{C} \to \mathbb{D}$ consists of*

(i) a function $F : ob(\mathbb{C}) \to ob(\mathbb{D})$

(ii) for every pair $C, C' \in ob(\mathbb{C})$, a morphism $F_{A,A'} : \mathbb{C}(C, C') \to \mathbb{D}(FC, FC')$

(iii) the "composition" axiom for any objects C, C', C'':

$$F_{C,C''} \circ c_{C,C',C''} = c_{FC,FC',FC''} \circ (F_{C,C'} \otimes F_{C',C''})$$

(iv) the "identity" axiom for any object C: $F_{C,C} \circ j_C = j_{FC}$

Definition A.1.4 *Let $\mathcal{V}$ be a monoidal category. Let $\mathbb{C}$ and $\mathbb{D}$ be $\mathcal{V}$-enriched categories and $F, G : \mathbb{C} \to \mathbb{D}$ two $\mathcal{V}$-enriched functors. A **$\mathcal{V}$-enriched natural transformation** $\alpha : F \to G$ is a family of morphisms $\alpha_C : I \to \mathbb{D}(FC, GC)$ for every object $C \in ob(\mathbb{C})$ (in $\mathcal{V}$) such that for all $C, C' \in ob(\mathbb{C})$*

$$c_{FC,GC,GC'} \circ (\alpha_C \otimes G_{C,C'}) \circ l^{-1}_{\mathbb{C}(C,C')} = c_{FC,FC',GC'} \circ (F_{C,C'} \otimes \alpha_{C'}) \circ r^{-1}_{\mathbb{C}(C,C')}$$

We further recall some additional structures on monoidal closed categories, which are necessary to obtain an enriched Yoneda lemma.

Definition A.1.5 *A monoidal category $\mathcal{V}$ is **symmetric** if for every pair $A, B \in \mathcal{V}$, there is an isomorphism $s_{A,B} : A \otimes B \cong B \otimes A$ that is natural in A and B such that the following diagrams commute:*

(i) the "associativity" coherence condition for every triple A, B, C

$$(id_B \otimes s_{A,C}) \circ \alpha_{B,A,C} \circ (s_{A,B} \otimes id_C) = \alpha_{B,C,A} \circ s_{A,B\otimes C} \circ \alpha_{A,B,C}$$

(ii) the "unit" coherence condition for every object A: $l_A \circ s_{A,I} = r_A$

(iii) the "symmetry axiom" for every pair A, B*:* $s_{B,A} \circ s_{A,B} = id_{A \otimes B}$

Definition A.1.6 *A monoidal category* $\mathcal{V}$ *is* ***closed*** *if for each object* $B \in \mathcal{V}$*, the functor* $- \otimes B : \mathcal{V} \to \mathcal{V}$ *has a right adjoint* $[B, -] : \mathcal{V} \to \mathcal{V}$*.*

Due to the fact that $\mathcal{V}$ is locally small, it can equally be expressed as follows: for any object B, there are bijections, often referred to as currying, between the homsets (natural in both A and C)

$$\lambda : \mathcal{V}(A \otimes B, C) \cong_{Set} \mathcal{V}(A, [B, C]) : \mu$$

Lemma A.1.7 *Any symmetric monoidal closed category* $\mathcal{V}$ *can be enriched over itself, written as* $\mathcal{V}^{\mathrm{er}}$*.*

Proof $\mathcal{V}^{\mathrm{er}}$ is an enriched category with the following structure:

(i) $ob(\mathcal{V}^{\mathrm{er}}) \stackrel{\mathrm{def}}{=} ob\,\mathcal{V}$

(ii) $\mathcal{V}^{\mathrm{er}}(A, B) \stackrel{\mathrm{def}}{=} [A, B]$

(iii) the "composition" morphism $c_{A,B,C} : [A, B] \otimes [B, C] \to [A, C]$ is defined as

$$c_{A,B,C} \stackrel{\mathrm{def}}{=} \lambda(ev_{B,C} \circ s_{B,[B,C]} \circ (ev_{A,B} \otimes id_{[B,C]}) \circ s_{[B,C],A})$$

(iv) the "identity" morphism $j_B : I \to [B, B]$ is defined as $j_B \stackrel{\mathrm{def}}{=} \lambda(l_B)$.

The coherence condition follow by standard computations. □

Lemma A.1.8 *Let* $\mathcal{V}$ *be a symmetric monoidal closed category and* $\mathbb{C}$ *a* $\mathcal{V}$*-category. Every object* $A \in ob(\mathbb{C})$ *induces two* ***representable*** $\mathcal{V}$***-enriched functors*** $\mathbb{C}^A : \mathbb{C} \to \mathcal{V}^{\mathrm{er}}$ *and* $\mathbb{C}_A : \mathbb{C}^{op} \to \mathcal{V}^{\mathrm{er}}$*, which are defined as follows:*

- *Any object* $B \in ob(\mathbb{C})$ *is respectively mapped to the objects* $\mathbb{C}(A, B), \mathbb{C}(B, A) \in ob\,\mathcal{V}$*.*

- *Given objects B and B' in $ob(\mathbb{C})$, the morphism*

$$\mathbb{C}^A_{B,B'} : \mathbb{C}(B, B') \to \mathcal{V}^{\mathrm{er}}(\mathbb{C}(A, B), \mathbb{C}(A, B')) \stackrel{\mathrm{def}}{=} [\mathbb{C}(A, B), \mathbb{C}(A, B')]$$

is defined as $\mathbb{C}^A_{B,B'} \stackrel{\mathrm{def}}{=} \lambda(c_{A,B,C} \circ s_{\mathbb{C}(A,B),\mathbb{C}(A,B')})$, and respectively for $\mathbb{C}^{B,B'}_A$.

The notion of a weak and strong Yoneda isomorphism are defined as follows:

Theorem A.1.9 (Weak Yoneda Lemma (Kelly [41])) *Let $\mathcal{V}$ be a symmetric monoidal closed category and $\mathbb{C}$ a small $\mathcal{V}$-category. For every $\mathcal{V}$-valued $\mathcal{V}$-functor $F : \mathbb{C} \to \mathcal{V}^{\mathrm{er}}$ and every object $A \in ob(\mathbb{C})$, there exists a bijection*

$$\mathcal{V}(I, FA) \cong_{Set} \mathcal{V}\text{-}\mathrm{Nat}(\mathbb{C}^A, F)$$

Under the additional assumption that $\mathcal{V}$ is complete, functor categories can be enriched over $\mathcal{V}$ and we obtain the Strong Yoneda Lemma.

Theorem A.1.10 (Strong Yoneda Lemma (Kelly [41])) *Let $\mathcal{V}$ be a complete symmetric monoidal closed category and $\mathbb{C}$ a small $\mathcal{V}$-enriched category. For every object $A \in ob(\mathbb{C})$ and every $\mathcal{V}$-valued $\mathcal{V}$-functor $F : \mathbb{C} \to \mathcal{V}^{\mathrm{er}}$ there exists an isomorphism in $\mathcal{V}$*

$$FA \cong_{\mathcal{V}} [\mathbb{C}, \mathcal{V}^{\mathrm{er}}](\mathbb{C}^A, F)$$

Corollary A.1.11 *Let $\mathcal{V}$ be a complete symmetric monoidal closed category. For every small $\mathcal{V}$-enriched category $\mathbb{C}$, the mapping $Y : \mathbb{C}^{op} \to [\mathbb{C}, \mathcal{V}^{\mathrm{er}}]$ with $A \mapsto \mathbb{C}^A$ can be extended to a $\mathcal{V}$-functor, referred to as the $\mathcal{V}$-Yoneda embedding.*

A.2 Internal Category Theory

We introduce some basic concepts of internal category theory. For a complete introduction we refer the reader to [5, 40].

Definition A.2.1 *Let $\mathcal{C}$ be a category with pullbacks. An* ***internal category*** $\mathbf{A}$ *in $\mathcal{C}$ has the following structure*

(1) an object $\mathbf{A}_0 \in ob\,\mathcal{C}$ (object of objects)

(2) an object $\mathbf{A}_1 \in ob\,\mathcal{C}$ (object of arrows)

(3) two morphisms $d_0, d_1 \in \mathbf{A}_1 \to \mathbf{A}_0$ in $mor\,\mathcal{C}$ (source and target)

(4) a morphism $i : \mathbf{A}_0 \to \mathbf{A}_1$ in $mor\,\mathcal{C}$ (identity)

(5) a morphism $c : \mathbf{A}_1 \times_{\mathbf{A}_0} \mathbf{A}_1 \to \mathbf{A}_1$ in $mor\,\mathcal{C}$, where $(\mathbf{A}_1 \times_{\mathbf{A}_0} \mathbf{A}_1, pr_0, pr_1)$ is the pullback for d_0 and d_1. (composition)

and must satisfy the following axioms:

(i) $d_0 \circ i = id_{\mathbf{A}_0} = d_1 \circ i$

(ii) $d_1 \circ pr_1 = d_1 \circ c$ *and* $d_0 \circ pr_0 = d_0 \circ c$.

(iii) $c \circ h_0 = id_{\mathbf{A}_1} = c \circ h_1$, *where h_0 and h_1 are factorisations through the pullback $(\mathbf{A}_1 \times_{\mathbf{A}_0} \mathbf{A}_1, pr_0, pr_1)$ for cones $(\mathbf{A}_1, 1_{\mathbf{A}_1}, id \circ d_0)$ and $(\mathbf{A}_1, i \circ d_1, id_{\mathbf{A}_1})$.*

(iv) $c \circ (id_{\mathbf{A}_1} \times_{\mathbf{A}_0} c) = c \circ (c \times_{\mathbf{A}_0} 1_{\mathbf{A}_1})$, *where*

$$(1_{\mathbf{A}_1} \times_{\mathbf{A}_0} c) : \mathbf{A}_1 \times_{\mathbf{A}_0} (\mathbf{A}_1 \times_{\mathbf{A}_0} \mathbf{A}_1) \to \mathbf{A}_1 \times_{\mathbf{A}_0} \mathbf{A}_1$$
$$(c \times_{\mathbf{A}_0} id_{\mathbf{A}_1}) : (\mathbf{A}_1 \times_{\mathbf{A}_0} \mathbf{A}_1) \times_{\mathbf{A}_0} \mathbf{A}_1) \to \mathbf{A}_1 \times_{\mathbf{A}_0} \mathbf{A}_1$$

are factorisations through the pullbacks.

Definition A.2.2 *Let $\mathcal{C}$ be a category with pullbacks. Given two internal categories $\mathbf{A}$ and $\mathbf{B}$, an* ***internal functor*** $\mathbf{F} : \mathbf{A} \to \mathbf{B}$ *is a pair of morphisms in $mor\,\mathcal{C}$*

$$\mathbf{F}_0 : \mathbf{A}_0 \to \mathbf{B}_0 \quad \mathbf{F}_1 : \mathbf{A}_1 \to \mathbf{B}_1$$

which satisfies the following conditions:

(1) $d_0 \circ \mathbf{F}_1 = \mathbf{F}_0 \circ d_0$, $d_1 \circ \mathbf{F}_1 = \mathbf{F}_0 \circ d_1$

(2) $\mathbf{F}_1 \circ i = i \circ \mathbf{F}_0$

(3) $\mathbf{F}_1 \circ c = c \circ (\mathbf{F}_1 \times_{\mathbf{F}_0} \mathbf{F}_1)$, *where*

$$(\mathbf{F}_1 \times_{\mathbf{F}_0} \mathbf{F}_1) : \mathbf{A}_1 \times_{\mathbf{A}_0} \mathbf{A}_1 \to \mathbf{B}_1 \times_{\mathbf{B}_0} \mathbf{B}_1$$

is the unique morphism such that $pr_0 \circ (\mathbf{F}_1 \times_{\mathbf{F}_0} \mathbf{F}_1) = \mathbf{F}_1 \circ pr_0$ *and* $pr_1 \circ (\mathbf{F}_1 \times_{\mathbf{F}_0} \mathbf{F}_1) = \mathbf{F}_1 \circ pr_1$

Definition A.2.3 *Let* $\mathcal{C}$ *be a category with pullbacks and let* $\mathbf{A}$ *be an internal category in* $\mathcal{C}$ *and* $\mathcal{P}, \mathcal{Q} : \mathcal{A} \to \mathcal{C}$ *two internal functors, written explicitly as* $\mathbb{P} = (P, p_0, p_1)$ *and* $\mathbb{Q} = (Q, q_0, q_1)$. *An* ***internal natural transformation*** $\alpha : \mathbb{P} \to \mathbb{Q}$ *is an arrow* $\alpha : P \to Q$ *such that*

- $q_0 \circ \alpha = p_0$
- $\alpha \circ p_1 = q_1 \circ (id_{A_1} \times_{A_0} \alpha)$

Definition A.2.4 *Let* $\mathcal{C}$ *be a category with pullbacks and let* $\mathbf{A}$ *be an internal category in* $\mathcal{C}$. *An* ***internal*** $\mathcal{C}$***-valued functor*** $\mathbb{P} : \mathcal{A} \to \mathcal{C}$ *is a triple* (P, p_0, p_1) *where:*

- $P \in ob\,\mathcal{C}$
- $p_0 : P \to A_0 \in mor\,\mathcal{C}$
- $p_1 : A_1 \times_{A_0} P \to P \in mor\,\mathcal{C}$, *where* $(A_1 \times_{A_0} P,\ \mathrm{pr}_{A_1}, \mathrm{pr}_P)$ *is the pullback of* d_0 *(domain map) and* p_0.

such that the following axioms hold

(i) $p_0 \circ p_1 = d_1 \circ \pi_{A_1}$

(ii) $p_1 \circ (i \circ p_0, id_P) = id_P$

(iii) $p_1 \circ (id_{A_1} \times_{A_0} p_1) = p_1 \circ (c \times_{A_0} id_P)$

Appendix B

Towards a Conservative Extension Result

In connection with the syntactically generated classifying category for [И]NLC that we aimed to construct in Chapter 5, we pursued another goal, namely to prove a categorical conservative extension result, which requires such classifying categories for NEL and [И]NLC.

We now sketch the categorical proof argument used and highlight the required auxiliary results. This motivates why certain properties like enriched and internal Yoneda isomorphisms, a notion of "nominal" gluing and FM-properties for functor categories have been studied in the first place.

B.1 Outline of the Categorical Proof Argument

We recall the third conservative extension property that we introduced in Section 3.5. We originally intended to prove this property for NEL and NLC, however the categorical proof argument requires classifying categories for both, NEL and NLC. But this is not possible for NLC, as we have explained in Chapter 5. So, instead, the property would have to be extended to [И]NLC.

Conjecture 3.5.3 (for [И]NLC**):** Let $Th = (Sg, Ax)$ be a NEL-theory. Let $Th' \stackrel{\text{def}}{=} (Sg', Ax')$ be the extension of Th to an [И]NLC-theory with $Ax' \stackrel{\text{def}}{=} Ax$ and Sg' is the extension of the NEL-signature Th. Then for any freshness context $\nabla \stackrel{\text{def}}{=} \overline{a}_1 \# x_1 : \gamma_1, ..., \overline{a}_n \# x_n : \gamma_n$ (with $\gamma_i \in Gnd_{Sg}$), ground type $\gamma \in Gnd_{Sg}$ and [И]NLC-expression E (without local scoping) such that $Th' \rhd \nabla \vdash^{[И]\text{NLC}} E : \gamma$ we can deduce the following:

(i) there exists a NEL-expression M for which $Th \rhd \nabla \vdash^{\text{NEL}} M : \gamma$ and $Th' \rhd \nabla \vdash^{[И]\text{NLC}} E \approx M : \gamma$ (informally: E is $\beta\eta$-reducible to M).

(ii) if there exists another NEL-expression M' for which $Th \rhd \nabla \vdash^{\text{NEL}} M' : \gamma$ and $Th' \rhd \nabla \vdash^{[И]\text{NLC}} E \approx M' : \gamma$ then $Th' \rhd \nabla \vdash^{\text{NEL}} M \approx M' : \gamma$.

Note that a key step of the proof argument is to translate the conservative extension property into a categorical setting using the classifying categories (categorical type theory correspondence) for NEL and [И]NLC. For the arguments sake, we suppose that a correspondence result for [И]NLC can be obtained. We remind the reader that for classifying categories $Cl(Th)$ and $Cl(Th')$ we have the following definitions:

$$Cl(Th)(\nabla, \emptyset \# x : s) \stackrel{\text{def}}{=} \{([M]^{\text{NEL}}_{\approx}) \mid Th \rhd \nabla \vdash^{\text{NEL}} M : s\}$$
$$Cl(Th')(\nabla, \emptyset \# x : s) \stackrel{\text{def}}{=} \{([E]^{[И]\text{NLC}}_{\approx}) \mid Th' \rhd \nabla \vdash^{[И]\text{NLC}} E : s\}$$

where the quotients $[M]^{\text{NEL}}_{\approx}$ and $[E]^{[И]\text{NLC}}_{\approx}$ are defined using provable equality of NEL and [И]NLC, respectively. Thus, the definition of a conservative extension, as given above, now amounts to demonstrating that the inclusion function

$$\{[M]^{\text{NEL}}_{\approx} \mid Th \rhd \nabla \vdash^{\text{NEL}} M : \gamma\} \to \{[E]^{[И]\text{NLC}}_{\approx} \mid Th' \rhd \nabla \vdash^{[И]\text{NLC}} E : \gamma\}$$

is surjective and injective. This corresponds categorically to an inclusion functor I, $I(\nabla) = \nabla$ and $I([M]^{\text{NEL}}_{\approx}) = [M]^{\text{NLC}}_{\approx}$, with $Cl(Th)(\nabla, \emptyset \# x : s) \to Cl(Th')(\nabla, \emptyset \# x : s)$ being bijective. Thus, to prove the conservative extension result, we could

equally show that there exists a full and faithfull inclusion functor I. Analogue to Crole [20] (Section 4.10), the existence of such an I depends, among other things, on demonstrating that for some category $\mathcal{D}$ and cartesian closed $\mathcal{V}$ such as *Nom*, *FMSet* and *FMNom*) we have

(i) $\mathcal{D}(A, B) \cong [\mathcal{D}, \mathcal{V}](YA, YB)$ (Yoneda isomorphism)

(ii) The construction of a glued category $\mathcal{GL}(\Gamma)$, where Γ is constructed using the Yoneda isomorphism.

(iii) The functor category $[\mathcal{D}, \mathcal{V}]$ is cartesian closed

This motivated our study of enriched and internal Yoneda isomorphisms for $\mathcal{V}$ being *Nom*, *FMNom* or *FMSet* in Chapter 7.

Further, given that we work with FM-categories in the context of NEL, we also need to augment the categorical gluing lemma to hold for FM-categories. A first step in this direction is the gluing lemma in Section 5.1.3. Furthermore, to be able to apply the gluing lemma in the categorical proof argument, we would have to show that the functor category $[\mathcal{D}, \mathcal{V}]$ is a cartesian closed category. In Section 7.2.2, we provided a first result for $\mathcal{V}$ being *Nom*. In addition, we would have to show that the functor category is an FM-category. We discussed this in more detail in Section 5.1.3.

The following sections contain the proofs of some of the aforementioned properties, which were stated in Chapter 5.

B.2 Freyd-style Gluing Proof

We decompose Theorem 5.1.17 into various lemmas. We start by introducing an internal permutation action for $\mathcal{GL}(\Gamma)$.

Lemma B.2.1 *An internal permutation action on $\mathcal{GL}(\Gamma)$ is defined as follows: for each object (C, f, D) and permutation π we specify a morphism $\mathcal{GL}(\Gamma)$*

$$\pi_{(C,f,D)} : (C, f, D) \to \pi \cdot (C, f, D) = (\pi \cdot C, \pi \cdot f, \pi \cdot D)$$

which is point-wise defined by $\pi_{(C,f,D)} \stackrel{\text{def}}{=} (\pi_C, \pi_D)$, where $\pi_C : C \to \pi \cdot C$ and $\pi_D : D \to \pi \cdot D$ are the respective internal permutation actions of $\mathcal{C}$ and $\mathcal{D}$, and $\pi \cdot f$ is the internal permutation action on morphisms of $\mathcal{C}$.

Proof It is clearly the case that $(\pi \cdot C, \pi \cdot f, \pi \cdot D)$ is an element of $\mathcal{GL}(\Gamma)$. To show that $\pi_{(C,f,D)}$ is a $\mathcal{GL}(\Gamma)$-arrow we need to check if the commuting condition $(\pi \cdot f) \circ \pi_C = \Gamma(\pi_D) \circ f$ holds:

$$\begin{aligned}
(\pi \cdot f) \circ \pi_C &\stackrel{\text{def}}{=} (\pi_{\Gamma D} \circ f \circ (\pi^{-1})_{\pi \cdot C}) \circ \pi_C \\
&\stackrel{\text{def}}{=} \pi_{\Gamma D} \circ f \circ (\pi^{-1} \circ \pi)_C \\
&= \pi_{\Gamma D} \circ f \circ \iota_C \\
&\stackrel{\text{def}}{=} \pi_{\Gamma D} \circ f \circ id_C \\
&= \pi_{\Gamma D} \circ f \\
&= \Gamma(\pi_D) \circ f && (\Gamma \text{ preserves } \pi_D)
\end{aligned}$$

Next we have to demonstrate that the properties of an internal permutation action hold. Given that $\mathcal{C}$ and $\mathcal{D}$ are FM-categories and the internal permutation action is defined point-wise, the result directly follows:

(i) We have $\iota_{(C,f,D)} \stackrel{\text{def}}{=} (\iota_C, \iota_D)$, $\iota_C = id_C$ and $\iota_D = id_D$. Hence, we can deduce that $\iota_{(C,f,D)} \stackrel{\text{def}}{=} (\iota_C, \iota_D) = (id_C, id_D) \stackrel{\text{def}}{=} id_{(C,f,D)}$.

(ii)

$$\begin{aligned}
(\pi' \circ \pi)_{(C,f,D)} &\stackrel{\text{def}}{=} ((\pi' \circ \pi)_C, (\pi' \circ \pi)_D) \\
&\stackrel{\text{def}}{=} (\pi'_{\pi \cdot C} \circ \pi_C, \pi'_{\pi \cdot D} \circ \pi_D) \\
&\stackrel{\text{def}}{=} (\pi'_{\pi \cdot C}, \pi'_{\pi \cdot D}) \circ (\pi_C, \pi_D) \\
&\stackrel{\text{def}}{=} \pi'_{(\pi \cdot C, \pi \cdot f, \pi \cdot D)} \circ \pi_{(C,f,D)}
\end{aligned}$$

□

We now demonstrate that there is a more "natural" way to apply the permutation action on morphisms of $\mathcal{GL}(\Gamma)$:

Lemma B.2.2 *For any morphism (h_1, h_2) in $\mathcal{GL}(\Gamma)$, we have that $\pi \cdot (h_1, h_2) = (\pi \cdot h_1, \pi \cdot h_2)$. Consequently, (h_1, h_2) is finitely supported by $supp(h_1) \cup supp(h_2)$ and the internal permutation action on $\mathcal{GL}(\Gamma)$ is finitely supported.*

Proof Take any $\mathcal{GL}(\Gamma)$-arrow $h = (h_1, h_2) : (C, f, D) \to (C', f', D')$. We then deduce the following:

$$\begin{aligned}
\pi \cdot h &\stackrel{\text{def}}{=} \pi_{(C',f',D')} \circ h \circ (\pi^{-1})_{\pi\cdot(C,f,D)} \\
&\stackrel{\text{def}}{=} (\pi_{C'}, \pi_{D'}) \circ (h_1, h_2) \circ (\pi^{-1})_{(\pi\cdot C, \pi\cdot f, \pi\cdot D)} \\
&\stackrel{\text{def}}{=} (\pi_{C'}, \pi_{D'}) \circ (h_1, h_2) \circ ((\pi^{-1})_{\pi\cdot C}, (\pi^{-1})_{\pi\cdot D}) \\
&\stackrel{\text{def}}{=} (\pi_{C'} \circ h_1 \circ (\pi^{-1})_{\pi\cdot C}, \pi_{D'} \circ h_2 \circ (\pi^{-1})_{\pi\cdot D}) \\
&\stackrel{\text{def}}{=} (\pi \cdot h_1, \pi \cdot h_2)
\end{aligned}$$

Given that both $\mathcal{C}$ and $\mathcal{D}$ are FM-categories, and h_1 and h_2 are respectively $\mathcal{C}$-arrows and $\mathcal{D}$-arrows, we have that $supp(h_1)$ and $supp(h_2)$ are finite sets and therefore h is finitely supported by $supp(h_1) \cup supp(h_2)$. Thus, by Definition 5.1.4, we have that the internal permutation action is finitely supported.

□

Lemma B.2.3 $\mathcal{GL}(\Gamma)$ *has equivariant finite products.*

Proof The terminal object $(1_\mathcal{C}, id_{1_\mathcal{C}}, 1_\mathcal{D})$ of $\mathcal{GL}(\Gamma)$, denoted as 1, is composed of the terminal objects $1_\mathcal{C}$ and $1_\mathcal{D}$ and the identify function $id_{1_\mathcal{C}}$. Given that Γ strictly preserves finite products we have $\Gamma 1_\mathcal{D} = 1_\mathcal{C}$ and therefore 1 is clearly an object of $\mathcal{GL}(\Gamma)$. Due to the fact that $\mathcal{C}$ has equivariant finite products, we have that $\pi \cdot 1_\mathcal{C} = 1_\mathcal{C}$.

Further, given that $1_{\mathcal{C}}$ is a terminal object, there is one unique morphism and therefore $\pi \cdot id_{1_{\mathcal{C}}} = id_{1_{\mathcal{C}}}$. We can now deduce the following:

$$\begin{aligned}
\pi \cdot 1 &\stackrel{\text{def}}{=} \pi \cdot (1_{\mathcal{C}}, id_{1_{\mathcal{C}}}, 1_{\mathcal{D}}) \\
&\stackrel{\text{def}}{=} (\pi \cdot 1_{\mathcal{C}}, \pi \cdot id_{1_{\mathcal{C}}}, \pi \cdot 1_{\mathcal{D}}) \\
&\stackrel{\text{def}}{=} (1_{\mathcal{C}}, \pi \cdot id_{1_{\mathcal{C}}}, 1_{\mathcal{D}}) \\
&\stackrel{\text{def}}{=} (1_{\mathcal{C}}, id_{1_{\mathcal{C}}}, 1_{\mathcal{D}}) \\
&\stackrel{\text{def}}{=} 1
\end{aligned}$$

Next, we have to show that for any pair of objects $(C, f, D), (C', f', D') \in \mathcal{GL}(\Gamma)$ with a product $(C, f, D) \times (C', f', D')$ and projections $pr_i : (C, f, D) \times (C', f', D') \to (C, f, D)$ for $i = 1, 2$, the following property holds:

$$\pi \cdot pr_i((C, f, D), (C', f', D')) = pr_i(\pi \cdot (C, f, D), \pi \cdot (C', f', D')) \text{ for } i = 1, 2$$

Note that by applying the definition of projections pr_i for $\mathcal{GL}(\Gamma)$, we can equally show that

$$\pi \cdot (pr_i(C, C'), (pr_i(D, D')) = (pr_i(\pi \cdot C, \pi \cdot C'), pr_i(\pi \cdot D, \pi \cdot D')) \text{ for } i = 1, 2$$

But this follows immediately by Lemma B.2.2 and the fact that $\mathcal{C}$ and $\mathcal{D}$ have finite equivariant products.

$$\begin{aligned}
\pi \cdot (pr_i(C, C'), (pr_i(D, D')) &= (\pi \cdot pr_i(C, C'), \pi \cdot pr_1(D, D)) && \text{(Lemma B.2.2)} \\
&= (pr_i(\pi \cdot C, \pi \cdot C'), pr_1(\pi \cdot D, \pi \cdot D'))
\end{aligned}$$

□

Lemma B.2.4 *$\mathcal{GL}(\Gamma)$ has fresh inclusions, which are defined point-wise as follows: $i^{\overline{a}}_{(C,f,D)} =_{def} (i^{\overline{a}}_{C}, i^{\overline{a}}_{D}) : (C^{\#\overline{a}}, f \uparrow C^{\#\overline{a}}, D^{\#\overline{a}}) \to (C, f, D)$ such that $f \circ i^{\overline{a}}_{C} = \Gamma i^{\overline{a}}_{D} \circ (f \uparrow C^{\#\overline{a}})$.*

Proof It is clearly the case that $i^{\overline{a}}_{(C,f,D)}$ is an $\mathcal{GL}(\Gamma)$-arrow. We prove the various properties step by step:

(**i**) We first have to prove equivariance: $\pi \cdot i^{\overline{a}}_{(C,f,D)} = i^{\pi\cdot\overline{a}}_{\pi\cdot(C,f,D)}$:

$$\begin{aligned}
\pi \cdot i^{\overline{a}}_{(C,f,D)} &\stackrel{\text{def}}{=} \pi \cdot (i^{\overline{a}}_{C}, i^{\overline{a}}_{D}) \\
&= (\pi \cdot i^{\overline{a}}_{C}, \pi \cdot i^{\overline{a}}_{D}) && \text{(Lemma B.2.2)} \\
&= (i^{\pi\cdot\overline{a}}_{\pi\cdot C}, i^{\pi\cdot\overline{a}}_{\pi\cdot D}) && \text{(equivariance)}
\end{aligned}$$

(**ii**) Next, we have to show that $i^{\emptyset}_{(C,f,D)} = id_{(C,f,D)}$ and $i^{\overline{a}}_{(C,f,D)} \circ i^{\overline{a}'}_{(C^{\#\overline{a}}, f{\restriction}C^{\#\overline{a}}, D^{\#\overline{a}})} = i^{\overline{a}\cup\overline{a}'}_{(C,f,D)}$.

$$\begin{aligned}
i^{\emptyset}_{(C,f,D)} &\stackrel{\text{def}}{=} (i^{\emptyset}_{C}, i^{\emptyset}_{D}) \\
&= (id_C, id_D) && \text{(fresh inclusions)} \\
&\stackrel{\text{def}}{=} id_{(C,f,D)}
\end{aligned}$$

$$\begin{aligned}
i^{\overline{a}}_{(C,f,D)} \circ i^{\overline{a}'}_{(C^{\#\overline{a}}, f{\restriction}C^{\#\overline{a}}, D^{\#\overline{a}})} &\stackrel{\text{def}}{=} (i^{\overline{a}}_{C}, i^{\overline{a}}_{D}) \circ (i^{\overline{a}'}_{C^{\#\overline{a}}}, i^{\overline{a}'}_{D^{\#\overline{a}}}) \\
&\stackrel{\text{def}}{=} (i^{\overline{a}}_{C} \circ i^{\overline{a}'}_{C^{\#\overline{a}}}, i^{\overline{a}}_{D} \circ i^{\overline{a}'}_{D^{\#\overline{a}}}) \\
&= (i^{\overline{a}\cup\overline{a}'}_{C}, i^{\overline{a}\cup\overline{a}'}_{D}) && \text{(fresh inclusions)}
\end{aligned}$$

(**iii**) We have to show that $i^{\overline{a}}_{(C,f,D)\times(C',f',D')} = i^{\overline{a}}_{(C,f,D)} \times i^{\overline{a}}_{(C',f',D')}$ holds.

$$\begin{aligned}
i^{\overline{a}}_{(C,f,D)\times(C',f',D')} &\stackrel{\text{def}}{=} i^{\overline{a}}_{(C\times C', \cong\circ(f\times f'), D\times D')} \\
&\stackrel{\text{def}}{=} (i^{\overline{a}}_{C\times C'}, i^{\overline{a}}_{D\times D'}) \\
&= (i^{\overline{a}}_{C} \times i^{\overline{a}}_{C'}, i^{\overline{a}}_{D} \times i^{\overline{a}}_{D'}) && \text{(fresh inclusions)} \\
&\stackrel{\text{def}}{=} (i^{\overline{a}}_{C}, i^{\overline{a}}_{D}) \times (i^{\overline{a}}_{C'}, i^{\overline{a}}_{D'}) \\
&\stackrel{\text{def}}{=} i^{\overline{a}}_{(C,f,D)} \times i^{\overline{a}}_{(C',f',D')}
\end{aligned}$$

(**iv**) Suppose that $supp(\pi)$ # (C, f, D). We have to show that $\pi_{(C,f,D)^{\#supp(\pi)}} =$

$id_{(C,f,D)^{\#supp(\pi)}}$ holds.

$$\begin{aligned}\pi_{(C,f,D)^{\#supp(\pi)}} &\stackrel{\text{def}}{=} \pi_{(C^{\#supp(\pi)}, f\upharpoonright C^{\#supp(\pi)}, D^{\#supp(\pi)})}\\ &\stackrel{\text{def}}{=} (\pi_{C^{\#supp(\pi)}}, \pi_{D^{\#supp(\pi)}})\\ &= (id_{C^{\#supp(\pi)}}, id_{D^{\#supp(\pi)}})\\ &\stackrel{\text{def}}{=} id_{(C,f,D)^{\#supp(\pi)}}\end{aligned}$$

(**v**) Let $f, g : (C, f, D) \to (C', f', D')$ be two parallel $\mathcal{C}$-arrows such that $f \circ i^{\overline{a}}_{(C,f,D)} = g \circ i^{\overline{a}}_{(C,f,D)}$ and $\overline{a} \# (f, g)$. We have to show that $f = g$. Let's refer to the components of f and g as (f_1, f_2) and (g_1, g_2) ,respectively. By enfolding the equation we get:

$$\begin{aligned}f \circ i^{\overline{a}}_{(C,f,D)} &\stackrel{\text{def}}{=} (f_1, f_2) \circ (i^{\overline{a}}_C, i^{\overline{a}}_D)\\ &\stackrel{\text{def}}{=} (f_1 \circ i^{\overline{a}}_C, f_2 \circ i^{\overline{a}}_D)\\ &\stackrel{\text{def}}{=} (g_1 \circ i^{\overline{a}}_C, g_2 \circ i^{\overline{a}}_D)\\ &\stackrel{\text{def}}{=} (g_1, g_2) \circ (i^{\overline{a}}_C, i^{\overline{a}}_D)\\ &= g \circ i^{\overline{a}}_{(C,f,D)}\end{aligned}$$

Hence, we have $f_1 \circ i^{\overline{a}}_C = g_1 \circ i^{\overline{a}}_C$ and $f_2 \circ i^{\overline{a}}_D = g_2 \circ i^{\overline{a}}_D$. We can furthermore deduce from $\overline{a} \# (f, g)$ that $\overline{a} \# f_1, f_2, g_1, g_2$. So, given that $\mathcal{C}$ and $\mathcal{D}$ are FM-categories we have that $f_1 = g_1$ and $f_2 = g_2$; thus $f = g$.

(**vi**) Let $h : (C, f, D) \to (C', f', D')$ be a morphism in $\mathcal{GL}(\Gamma)$ and $\overline{a}$ a finite set of names such that $\overline{a} \# (C', f', D')$. We pick a finite set of names $\overline{b}$ such that $\overline{b} \# (\overline{a}, h)$ and $(\overline{a}\,\overline{b})_{(C',f',D')} \circ h \circ i^{\overline{b}}_{(C,f,D)} = h \circ i^{\overline{b}}_{(C,f,D)}$. We have to prove that there exists a unique $\hat{h} : (C, f, D) \to (C', f', D')^{\#\overline{a}}$ such that $i^{\overline{a}}_{(C',f',D')} \circ \hat{h} = h$ holds. Let $h \stackrel{\text{def}}{=} (h_1, h_2)$. By

enfolding the equation we obtain:

$$\begin{aligned}(\overline{a}\,\overline{b})_{(C',f',D')} \circ h \circ i^{\overline{b}}_{(C,f,D)} &\stackrel{\text{def}}{=} ((\overline{a}\,\overline{b})_{C'}, (\overline{a}\,\overline{b})_{D'}) \circ (h_1, h_2) \circ (i^{\overline{b}}_C, i^{\overline{b}}_D) \\ &\stackrel{\text{def}}{=} ((\overline{a}\,\overline{b})_{C'} \circ h_1 \circ i^{\overline{b}}_C, (\overline{a}\,\overline{b})_{D'} \circ h_2 \circ i^{\overline{b}}_D) \\ &= (h_1 \circ i^{\overline{b}}_C, h_2 \circ i^{\overline{b}}_D) \\ &\stackrel{\text{def}}{=} (h_1, h_2) \circ (i^{\overline{b}}_C, i^{\overline{b}}_D) \\ &\stackrel{\text{def}}{=} h \circ i^{\overline{b}}_{(C,f,D)}\end{aligned}$$

Hence, we have that $(\overline{a}\,\overline{b})_{C'} \circ h_1 \circ i^{\overline{b}}_C = h_1 \circ i^{\overline{b}}_C$ and $(\overline{a}\,\overline{b})_{D'} \circ h_2 \circ i^{\overline{b}}_D = h_2 \circ i^{\overline{a}}_D$. From $\overline{a} \# (C', f', D')$ we can deduce that $\overline{a} \# C'$ and from $\overline{b} \# (\overline{a}, h)$ that $\overline{b} \# (\overline{a}, h_1)$. So, there exists $\hat{h}_1 : C \to C'^{\#\overline{a}}$ and $\hat{h}_2 : D \to D'^{\#\overline{a}}$ such that $h_1 = i^{\overline{a}}_{C'} \circ \hat{h}_1$ and $h_2 = i^{\overline{a}}_{D'} \circ \hat{h}_2$ ($\diamond$) We can now take $\hat{h} =_{\text{def}} (\hat{h}_1, \hat{h}_2)$. The fact that $h = i^{\overline{a}}_{(C',f',D')} \circ \hat{h}$ is deduced as follows:

$$\begin{aligned}i^{\overline{a}}_{(C',f',D')} \circ \hat{h} &\stackrel{\text{def}}{=} i^{\overline{a}}_{(C',f',D')} \circ (\hat{h}_1, \hat{h}_2) \\ &\stackrel{\text{def}}{=} (i^{\overline{a}}_{C'}, i^{\overline{a}}_{D'}) \circ (\hat{h}_1, \hat{h}_2) \\ &\stackrel{\text{def}}{=} (i^{\overline{a}}_{C'} \circ \hat{h}_1, i^{\overline{a}}_{D'} \circ \hat{h}_2) \\ &= (h_1, h_2) \qquad (\diamond) \\ &\stackrel{\text{def}}{=} h\end{aligned}$$

We can deduce from the fact that $\hat{h}_1$ and $\hat{h}_2$ are unique that $\hat{h}$ is unique with respect to the commuting property. □

Lemma B.2.5 *$P_2 : \mathcal{GL}(\Gamma) \to \mathcal{D}$ is an FM-functor, i.e. it strictly preserves internal permutation actions, products and fresh inclusions*

Proof

(i) for internal permutations we can deduce:

$$P_2(\pi_{(C,f,D)}) \stackrel{\text{def}}{=} P_2((\pi_C, \pi_D)) \stackrel{\text{def}}{=} \pi_D \stackrel{\text{def}}{=} \pi_{P_2((C,f,D))}$$

(ii) for products we can deduce:

$$P_2(1_{\mathcal{C}}, id, 1_{\mathcal{D}}) \stackrel{\text{def}}{=} 1_{\mathcal{D}} \stackrel{\text{def}}{=} P_2((1_{\mathcal{C}}, id, 1_{\mathcal{D}}))$$

$$P_2(pr_{(C,f,D)}) \stackrel{\text{def}}{=} P_2((pr_C, pr_D)) \stackrel{\text{def}}{=} pr_D \stackrel{\text{def}}{=} pr_{P_2((C,f,D))}$$

(iii) for fresh inclusions we can deduce:

$$P_2(i^{\overline{a}}_{(C,f,D)}) \stackrel{\text{def}}{=} P_2((i^{\overline{a}}_C, i^{\overline{a}}_D)) \stackrel{\text{def}}{=} i^{\overline{a}}_D \stackrel{\text{def}}{=} i^{\overline{a}}_{P_2((C,f,D))}$$

□

B.3 Product and Functor Category Proofs

Proof of Lemma 5.1.19 It follows by routine computations that the product category $\mathcal{C} \times \mathcal{D}$ is an FM-category with the provided structures. We now demonstrate that the category of small FM-categories and FM-functors has finite products:

As it is well known, any category with a single object A and morphism (id_A) is a terminal object in the category of small categories. This can trivially be extended to an FM-category. As shown in Lemma 5.1.19, we have that $\mathcal{C} \times \mathcal{D}$ is an FM-category and if $\mathcal{C}$ and $\mathcal{D}$ are small, we have that $\mathcal{C} \times \mathcal{D}$ is small as well. The corresponding projection maps $pr_i : \mathcal{C}_1 \times \mathcal{C}_2 \to \mathcal{C}_i$ (for $i = 1, 2$) are functors defined as follows: $pr_i((C_1, C_2)) \stackrel{\text{def}}{=} C_i$ and $pr_i((f_1, f_2)) \stackrel{\text{def}}{=} f_i$. We continue with the universal property. For any product cone $(\mathcal{B}, F_1, F_2)$, the mediating functor $\langle F_1, F_2\rangle : \mathcal{B} \to \mathcal{C}_1 \times \mathcal{C}_2$ is defined as follows: $\langle F_1, F_2\rangle(B) \stackrel{\text{def}}{=} \langle F_1 B, F_2 B\rangle$ and $\langle F_1, F_2\rangle(f) \stackrel{\text{def}}{=} \langle F_1 f, F_2 f\rangle$. The proof that the commuting condition holds and that the projection functors and the mediating functions are FM-functors follows by routine computations. □

Proof of Lemma 5.1.22 We enhance the proof argument that shows that the category of small categories and functors is cartesian closed with "nominal" computations.

As we have shown in Lemma 5.1.19, the category has finite products. So, we only have to demonstrate that the category has exponentials: Let $\mathcal{C}$ and $\mathcal{D}$ be small perm-categories. Then $[\mathcal{C},\mathcal{D}]_{fs} \subseteq [\mathcal{C},\mathcal{D}]$ (luff subcategory) is the exponential of $\mathcal{C}$ and $\mathcal{D}$. Given that $\mathcal{C}$ and $\mathcal{D}$ are small, we have that $[\mathcal{C},\mathcal{D}]$ is small and therefore $[\mathcal{C},\mathcal{D}]_{fs}$ is small as well. By Lemma 5.1.20 we have that there exists an internal permutation action for $[\mathcal{C},\mathcal{D}]_{fs}$. Then, by definition, we have that the morphisms are finitely supported and therefore $[\mathcal{C},\mathcal{D}]_{fs}$ is a perm-category. The evaluation map $ev : [\mathcal{C},\mathcal{D}]_{fs} \times \mathcal{C} \to \mathcal{D}$ is defined as follows:

$$ev((F,C)) \stackrel{\text{def}}{=} FC$$
$$ev(\eta, f) \stackrel{\text{def}}{=} \eta_{C'} \circ Ff = Gf \circ \eta_C \text{ for } \eta : F \Rightarrow G \text{ and } f : C \to C'$$

It can easily be demonstrated that ev is a functor. In addition, we have that show that ev preserves internal permutation actions:

$$\begin{aligned}
ev(\pi_{(F,C)}) &\stackrel{\text{def}}{=} ev(\pi_F, \pi_C) \\
&\stackrel{\text{def}}{=} (\pi_F)_{\pi\cdot C} \circ F(\pi_C) \\
&\stackrel{\text{def}}{=} \pi_{F(\pi^{-1}\cdot\pi\cdot C)} \circ F(\pi^{-1}_{\pi\cdot C}) \circ F(\pi_C) \\
&= \pi_{FC} \circ F(\pi^{-1}_{\pi\cdot C} \circ \pi_C) && (F \text{ is a functor}) \\
&= \pi_{FC} \circ F(id_C) && (\text{Lemma 5.1.3 (ii)}) \\
&= \pi_{FC} \circ id_{FC} && (F \text{ is a functor}) \\
&= \pi_{FC} \\
&\stackrel{\text{def}}{=} \pi_{ev((F,C))}
\end{aligned}$$

For any perm-functor $F : \mathcal{A} \times \mathcal{C} \to \mathcal{D}$, there exists a unique exponential mate $\lambda(F) : \mathcal{A} \to [\mathcal{C},\mathcal{D}]_{fs}$, which is defined as follows:

Objects: $\lambda(F)(A)$: is a functor defined by:

$$\lambda(F)(A)(C) \stackrel{\text{def}}{=} F((A, C))$$
$$\lambda(F)(A)(f) \stackrel{\text{def}}{=} F((id_A, f)) \text{ for } f : C \to C'$$

Morphisms: For $g : A \to A'$, $\lambda(F)g : \lambda(F)(A) \to \lambda(F)(A')$ is a natural transformation with components

$$(\lambda(F)(g))_C \stackrel{\text{def}}{=} F((g, id_C)) : F((A, C)) \to F((A', C))$$

The fact that $\lambda(F)(A)$ is a functor and $\lambda(F)(g)$ is a natural transformation follows as in the original argument. What remains to be shown is that $\lambda(F)(g)$ is finitely supported. Let $c, c' \# g$ (g is finitely supported). Given that F is a perm-functor, we have that $F(\pi_C) = \pi_{FC}$ and therefore $\pi \cdot FC = F(\pi \cdot C)$.

$$\begin{aligned}
((c\,c') \cdot \lambda(F)(g))_C &= (c\,c') \cdot (\lambda(F)(g))_{(c\,c')\cdot C} && \text{(Lemma 5.1.21)} \\
&\stackrel{\text{def}}{=} (c\,c') \cdot F((g, id_{(c\,c')\cdot C})) \\
&= F((c\,c') \cdot (g, id_{(c\,c')\cdot C})) && (F \text{ is a perm-functor}) \\
&= F(((c\,c') \cdot g, (c\,c') \cdot id_{(c\,c')\cdot C})) \\
&= F((g, id_C)) && (c, c' \# g) \\
&\stackrel{\text{def}}{=} (\lambda(F)(g))_C
\end{aligned}$$

The fact that $\lambda(F)$ is a functor follows as in the original argument. Next, we have

to show that $\lambda(F)$ preserves internal permutation actions. Let π_A and $C \in \mathcal{C}$.

$$\begin{aligned}
(\pi_{\lambda(F)(A)})_C &\stackrel{\text{def}}{=} \pi_{\lambda(F)(A)(\pi^{-1}\cdot C)} \circ \lambda(F)(A)(\pi_C^{-1}) \\
&\stackrel{\text{def}}{=} \pi_{F((A,\pi^{-1}\cdot C))} \circ F((id_A, \pi_C^{-1})) \\
&= F(\pi_{(A,\pi^{-1}\cdot C)}) \circ F((id_A, \pi_C^{-1})) && (F \text{ preserves int. perm.}) \\
&\stackrel{\text{def}}{=} F((\pi_A, \pi_{\pi^{-1}\cdot C})) \circ F((id_A, \pi_C^{-1})) \\
&= F((\pi_A, \pi_{\pi^{-1}\cdot C}) \circ (id_A, \pi_C^{-1})) \\
&\stackrel{\text{def}}{=} F((\pi_A \circ id_A, \pi_{\pi^{-1}\cdot C} \circ \pi_C^{-1})) \\
&= F((\pi_A, id_C)) && (\text{inverse}) \\
&\stackrel{\text{def}}{=} (\lambda(F)(\pi_A))_C
\end{aligned}$$

The fact that $\lambda(F)$ is unique and the commuting condition holds follows as in the case of small categories and functors. This concludes the argument. □

Proof of Lemma 5.1.23 We recall that finite products in a functor category are constructed point-wise. We now only have to show that the conditions of an equivariant finite product hold:

$$\begin{aligned}
(\pi \cdot 1_{[\mathcal{C},\mathcal{D}]})(X) &\stackrel{\text{def}}{=} \pi \cdot 1_{[\mathcal{C},\mathcal{D}]}(\pi^{-1} \cdot X) \\
&\stackrel{\text{def}}{=} \pi \cdot 1_{\mathcal{D}} && (\text{point-wise}) \\
&= 1_{\mathcal{D}} && (\mathcal{D} \text{ has equiv. products}) \\
&\stackrel{\text{def}}{=} 1_{[\mathcal{C},\mathcal{D}]}(X)
\end{aligned}$$

The argument for morphisms follows similarly. We now continue with the equiv-

ariance of the projection maps:

$$
\begin{aligned}
(\pi \cdot pr_i(F,G))_X &= \pi \cdot (pr_i(F,G))_{\pi^{-1}\cdot X} && \text{(Lemma 5.1.21)}\\
&\stackrel{\text{def}}{=} \pi \cdot pr_i(F(\pi^{-1}\cdot X), G(\pi^{-1}\cdot X)) && \text{(point-wise)}\\
&= pr_i(\pi \cdot F(\pi^{-1}\cdot X), \pi \cdot G(\pi^{-1}\cdot X)) && (\mathcal{D} \text{ has equiv. products})\\
&\stackrel{\text{def}}{=} pr_i((\pi \cdot F)(X), (\pi \cdot G)(X))\\
&= (pr_i(\pi \cdot F, \pi \cdot G))_X && \text{(point-wise)}
\end{aligned}
$$

□

Bibliography

[1] S. Abramsky, D. Ghica, A. Murawski, L. Ong, and I. Stark. Nominal Games and Full Abstraction for the Nu-Calculus. In *Proceedings of the Nineteenth Annual IEEE Symposium on Logic in Computer Science*, pages 150–159. IEEE Computer Society Press, 2004.

[2] R. M. Amadio and P.-L. Curien. *Domains and Lambda-Calculi*. Cambridge University Press, New York, NY, USA, 1998.

[3] H. Barendregt, S. Abramsky, D. M. Gabbay, T. S. E. Maibaum, and H. P. Barendregt. Lambda Calculi with Types. In *Handbook of Logic in Computer Science*, pages 117–309. Oxford University Press, 1992.

[4] S. Berghofer and C. Urban. A Head-to-Head Comparison of de Bruijn Indices and Names. In *In Proceedings of the International Workshop on Logical Frameworks and Meta Languages: Theory and Practice*, pages 46–59, 2006.

[5] F. Borceux. *Handbook of Categorical Algebra: Volume 1, Basic Category Theory*. Cambridge Textbooks in Linguistics. Cambridge University Press, 1994.

[6] F. Borceux. *Handbook of Categorical Algebra: Volume 2, Categories and Structures*. Cambridge Studies in Philosophy. Cambridge University Press, 1994.

[7] N. G. D. Bruijn. Lambda Calculus Notation with Nameless Dummies, a Tool

for Automatic Formula Manipulation, with Application to the Church-Rosser Theorem. *INDAG. MATH*, 34:381–392, 1972.

[8] J. Cheney. Logic Column 14: Nominal Logic and Abstract Syntax. *ACM SIGACT News*, 36(4):47–69, December 2005.

[9] J. Cheney. A Simple Nominal Type Theory. *Electronic Notes in Theoretical Computer Science*, 228:37–52, Jan. 2009.

[10] J. Cheney. A Dependent Nominal Type Theory. *Logical Methods in Computer Science*, 8(1), 2012.

[11] J. Cheney and C. Urban. Nominal Logic Programming. *ACM Transactions on Programming Languages and Systems*, 30(5), 2008.

[12] A. Church. A Formulation of the Simple Theory of Types. *Jurnal of Symbolic Logic*, 5(2):56–68, June 1940.

[13] R. Clouston. *Equational Logic for Names and Binding*. PhD thesis, University of Cambridge, Cambridge, United Kingdom, June 2009.

[14] R. Clouston. Binding in Nominal Equational Logic. *Electronic Notes in Theoretical Computer Science*, 265:259 – 276, 2010.

[15] R. Clouston. Nominal Lawvere Theories. In *WoLLIC'11*, pages 67–83, 2011.

[16] R. Clouston. Nominal Logic with Equations Only. In H. Geuvers and G. Nadathur, editors, *Proceedings of the Sixth International Workshop on Logical Frameworks and Meta-languages: Theory and Practice (LFMTP 2011)*, volume 71 of *Electronic Proceedings in Theoretical Computer Science*, pages 44–57, 2011.

[17] R. Clouston. Nominal Lawvere Theories: A Category Theoretic Account of Equational Theories with Names. *Journal of Computer and System Sciences*, 2013.

[18] R. Clouston and A. M. Pitts. Nominal Equational Logic. *Electronic Notes in Theoretical Computer Science*, 172:223–257, 2007.

[19] K. Crary and R. Harper. Logic Column 16: Higher-Order Abstract Syntax: Setting the Record Straight. *ACM SIGACT News*, 37(3):93–96, September 2006.

[20] R. L. Crole. *Categories for Types*. Cambridge University Press, 1993.

[21] R. L. Crole. On Fixpoint Objects and Gluing Constructions. *Applied Categorical Structures*, 4(2-3):251–281, 1996.

[22] R. L. Crole. α-Equivalence Equalities. *Theoretical Computer Science*, 433:1–19, May 2012.

[23] R. L. Crole and F. Nebel. An Internal Language for FM-Cartesian Closed Categories (The Nominal Lambda Calculus). In M. Mislove, editor, *Proceedings of Mathematical Foundations of Programming Semantics (MFPS), New Orleans, LA*, 2013. To Appear.

[24] H. B. Curry and R. Feys. *Combinatory Logic, Volume I*. North-Holland, 1958. Second printing 1968.

[25] E. Fairweather, M. Fernández, and M. J. Gabbay. Principal Types for Nominal Theories. In *Proceedings of the 18th International Conference on Fundamentals of Computation Theory*, FCT'11, pages 160–172, Berlin, Heidelberg, 2011. Springer-Verlag.

[26] E. Fairweather, M. Fernández, N. Szasz, and A. Tasistro. Dependent Types for a Nominal Logical Framework, 2012.

[27] M. Fernández and M. Gabbay. Nominal Rewriting. *Information and Computation*, 205:917–965, 2007.

[28] M. Fernández and M. J. Gabbay. Nominal Rewriting with Name Generation: Abstraction vs. Locality. In *Proceedings of the 7th ACM SIGPLAN International Symposium on Principles and Practice of Declarative Programming (PPDP 2005)*, page 4758. ACM Press, July 2005.

[29] A. Fraenkel. *Der Begriff "definit" und die Unabhängigkeit des Auswahlaxioms.* Sitzungsberichte der Preussischen Akademie der Wissenschaften. Physikalisch-mathematische Klasse. 1922.

[30] H. Friedman. Equality between Functionals. In *Logic Colloquium*, volume 453 of *Springer Lecture Notes*, pages 22–37, 1975.

[31] M. Gabbay and J. Cheney. A Sequent Calculus for Nominal Logic. In *LICS*, pages 139–148. IEEE Computer Society, 2004.

[32] M. Gabbay and A. Pitts. A New Approach to Abstract Syntax Involving Binders. In G. Longo, editor, *Proceedings of the Fourteenth Annual IEEE Symp. on Logic in Computer Science, LICS 1999*, pages 214–224. IEEE Computer Society Press, July 1999.

[33] M. Gabbay and A. M. Pitts. A New Approach to Abstract Syntax with Variable Binding. *Formal Asp. Comput.*, 13(3-5):341–363, 2002.

[34] M. J. Gabbay. *A Theory of Inductive Definitions with α-Equivalence.* PhD thesis, University of Cambridge, UK, March 2001.

[35] M. J. Gabbay. Foundations of Nominal Techniques: Logic and Semantics of Variables in Abstract Syntax. *Bulletin of Symbolic Logic*, 17(2):161–229, 2011.

[36] M. J. Gabbay and A. Mathijssen. Nominal Universal Algebra: Equational Logic with Names and Binding. *Journal of Logic and Computation*, 19(6):1455–1508, December 2009.

[37] M. J. Gabbay and D. P. Mulligan. Nominal Henkin Semantics: Simply-Typed Lambda-Calculus Models in Nominal Sets. In *LFMTP*, pages 58–75, 2011.

[38] J. R. Hindley and J. P. Seldin. *Lambda-Calculus and Combinators: An Introduction*. Cambridge University Press, New York, NY, USA, 2 edition, 2008.

[39] C. B. Jay and N. Ghani. The Virtues of Eta-Expansion. *Journal of Functional Programming*, 5(2):135–154, 1995.

[40] P. Johnstone. *Topos Theory*. Number 10 in London Mathematical Society Monographs. Academic Press, 1977.

[41] G. Kelly. *Basic Concepts of Enriched Category Theory*. London Mathematical Society lecture note series. Cambridge University Press, 1982.

[42] J. Lambek and P. Scott. *Introduction to Higher Order Categorical Logic*. Cambridge University Press, 1986.

[43] S. Lane. *Categories for the Working Mathematician*. Graduate Texts in Mathematics. Springer, 1998.

[44] S. Lane and I. Moerdijk. *Sheaves in Geometry and Logic: A First Introduction to Topos Theory*. Mathematical Sciences Research Institute Publications. U.S. Government Printing Office, 1992.

[45] Q. Ma and J. Reynolds. Types, abstraction, and parametric polymorphism, part 2. In S. Brookes, M. Main, A. Melton, M. Mislove, and D. Schmidt, editors, *Mathematical Foundations of Programming Semantics*, volume 598 of *Lecture Notes in Computer Science*, pages 1–40. Springer Berlin Heidelberg, 1992.

[46] E. Manes. *Algebraic Theories*, volume 26 of *Graduate Texts in Mathematics*. SV, 1976.

[47] J. McKinna and R. Pollack. Some Lambda Calculus and Type Theory Formalized. *JAR*, 1998.

[48] J. C. Mitchell. *Foundations for Programming Languages*. Foundation of computing series. MIT Press, 1996.

[49] A. Mostowski. Über den Begriff einer endlichen Menge. 31:13–20, 1938.

[50] M. Newman. On Theories with a Combinatorial Definition of "Equivalence". *Annals of Mathematics*, 43(2):223–243, 1942.

[51] F. Pfenning and C. Elliot. Higher-order Abstract Syntax. *SIGPLAN Notices*, 23(7):199–208, June 1988.

[52] A. M. Pitts. Categorical Logic. In *Handbook of Logic in Computer Science: Volume 5: Logic and Algebraic Methods*, pages 39–123, Oxford, UK, 2000. Oxford University Press.

[53] A. M. Pitts. Nominal Logic, A First Order Theory of Names and Binding. *Information and Computation*, 186:165–193, 2003.

[54] A. M. Pitts. Alpha-Structural Recursion and Induction. *Journal of the ACM*, 53:459–506, 2006.

[55] A. M. Pitts. Structural Recursion with Locally Scoped Names. *Journal of Functional Programming*, 21(3):235–286, 2011.

[56] A. M. Pitts. *Nominal Sets: Names and Symmetry in Computer Science*, volume 57 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2013.

[57] G. Plotkin. Notes on Completeness of the Full Continuous Hierarchy. Unpublished Manuscript, 1982.

[58] Z. Qian and T. Nipkow. Reduction and Unification in Lambda Calculi with a General Notion of Subtype. *Journal of Automated Reasoning*, 12(3):389–406, 1994.

[59] J. G. Riecke. Statman's 1-Section Theorem. *Journal of Information and Computation*, 116(2):294–303, 1995.

[60] U. Schöpp and I. Stark. A Dependent Type Theory with Names and Binding. In *In Proceedings of the 2004 Computer Science Logic Conference, number 3210 in Lecture notes in Computer Science*, pages 235–249. Springer-Verlag, 2004.

[61] M. R. Shinwell. *The Fresh Approach: Functional Programming with Names and Binders*. PhD thesis, University of Cambridge, February 2005.

[62] M. R. Shinwell and A. M. Pitts. On a Monadic Semantics for Freshness. *Theoretical Computer Science*, 342:28–55, 2005.

[63] J. R. Shoenfield. Axioms of Set Theory. In J. Barwise, editor, *Handbook of Mathematical Logic*, pages 321–344. North-Holland, Amsterdam, 1977.

[64] R. Statman. Completeness, Invariance and λ-definability. *Journal of Symbolic Logic*, 1982.

[65] R. Statman. Equality between Functionals, revisted. In *Harvey Friedman's Research on the Foundations of Mathematics*, pages 331–338. 1985.

[66] T. Streicher. *Semantics of Type Theory: Correctness, Completeness, and Independence Results*, volume XII of *Progress in Theoretical Computer Science*. Basel: Birkhäuser Verlag, 1991.

[67] W. W. Tait. Intensional Interpretations of Functionals of Finite Type I. *Journal of Symbolic Logic*, 32(2):198–212, 06 1967.

[68] D. Turner and G. Winskel. Nominal Domain Theory for Concurrency. In *CSL*, pages 546–560, 2009.

[69] N. Tzevelekos. *Nominal Game Semantics*. PhD thesis, University of Oxford, 2008.

[70] C. Urban. Nominal Techniques in Isabelle/HOL. *Journal of Automated Reasoning*, 40(4):327–356, May 2008.

[71] G. Winskel. *The Formal Semantics of Programming Languages*. Foundations of Computing. The MIT Press, Cambridge, Massachusetts, 1993.