Abstractness of the Nu-Calculus in Quasi-Borel Spaces at First-Order Types

Michael Wolman

Department of Mathematics and Statistics
McGill University, Montreal

August 2020

A thesis submitted to McGill University in partial fulfillment of the requirements of the degree of Master of Science in Mathematics

To Dov, Karen and Esther

Abstract

The nu-calculus is a simply typed, higher-order, call-by-value language that models fresh name generation [PS93]. In this language, names can be checked for equality and newly generated names are guaranteed to be distinct from all others.

In this thesis, we show that we can interpret names to be elements of a probability space, modelling fresh name generation as sampling names from a continuous measure. Specifically, we show that the nu-calculus can be soundly interpreted in the category of quasi-Borel spaces, a recent construction providing a model of higher-order probabilistic programming [Heu+17].

We then provide a novel analysis of higher-order functions in both the nu-calculus and the category of quasi-Borel spaces. In the nu-calculus, we construct a normal form for terms at first-order types eliminating the use of private names. We then analyze the structure of higher-order quasi-Borel spaces to prove that our semantics are abstract at first-order types.

Résumé

Le nu-calcul est un langage simplement typé, d'ordre supérieur, appel par valeur qui modélise la génération de noms frais [PS93]. Dans ce langage, l'égalité des noms peut être vérifié et les noms nouvellement générés sont garantis d'être distincts de tous les autres.

Dans cette thèse, nous montrons que nous pouvons interpréter les noms comme des éléments d'un espace de probabilité, modélisant la génération de noms frais comme échantillonnage de noms à partir d'une mesure continue. Plus précisément, nous montrons que le nu-calcul peut être correctement interprété dans la catégorie d'espaces quasi-boréliennes, une construction récente fournissant un modèle de programmation probabiliste d'ordre supérieure [Heu+17].

Nous fournissons ensuite une nouvelle analyse de fonctions d'ordre supérieur, à la fois dans le nu-calcul et dans la catégorie des espaces quasi-boréliennes. Dans le nu-calcul, nous construisons une forme normale pour les termes de types de premier ordre, éliminant l'utilisation de noms privés. Nous analysons ensuite la structure des espaces quasi-boréliennes d'ordre supérieur pour prouver que notre sémantique est abstraite aux types du premier ordre.

Acknowledgements

First, I would like to thank my supervisors Marcin and Prakash for supporting my research and the preparation of this thesis; Marcin, Prakash and McGill's Department of Mathematics and Statistics for their financial support; and the Fonds de Recherche du Québec for supporting me with a master's research scholarship. I would also like to thank Ohad Kammar for reviewing this thesis and for his detailed and insightful feedback and suggestions.

Marcin and Prakash, you have been incredible mentors throughout my studies and I cannot thank you enough for all of your time, patience, guidance and support.

Mom and dad, thank you for an endless supply of love and encouragement. You empower me to pursue my passions.

Esther, thank you for your love, energy and support, and for always being there for me when I need you.

Audrey, Harry, Amit, Aram, Jean — you guys are cool I guess.

Finally, thank you to my siblings, grandparents and the rest of my family, as well as all of the friends who have made my time at McGill special (you know who you are).

Contribution of Authors

The topic of this thesis was suggested by Sam Staton and Dario Stein, and this research has been conducted jointly with Marcin Sabok, Sam Staton and Dario Stein [Sab+20]. This work was done under the supervision of Prakash Panangaden and Marcin Sabok. The original material in this document consists of the work in Section 5.5 and Chapter 6, in particular the normal forms of Section 6.2.2 and our main result Theorem 6.31. The proof of Proposition 6.4 given in this document was suggested by Marcin Sabok, although a different proof was independently communicated to Sam Staton and Dario Stein by Alexander Kechris.

Contents

1 Introduction

The *nu-calculus* is a programming language constructed by Pitts and Stark to model *fresh name generation* [PS93]. A name is an atomic object, containing no information other than its identity. They can be compared for equality, and passed as arguments between functions. This language also includes the ability to generate *fresh names*, which are names that are guaranteed to be distinct from all other names. The desire to generate fresh names is not new, as can be seen for example in α-renaming for capture-avoiding substitution, and can be used to model memory allocation, for example, or the gensym metaprogramming macro in LISP.

The nu-calculus supports higher-order functions, an essential feature of functional programming. However, most of the challenges that arise in the analysis of fresh name generation occur at higher-order types. This is because it is possible to have terms with *private names*, names that are generated and used but do not affect the operational semantics of the term. It is therefore desirable to obtain models of the nu-calculus that identify terms with the same operational semantics. We call such models *abstract*.

There is also an analogy between the nu-calculus and probability. Fresh name generation behaves identically to sampling from continuous probability measures, for which the probability of sampling the same name twice is zero. There is therefore a desire to formalize this and construct a probabilistic model of the nu-calculus.

In order to construct a probabilistic model of the nu-calculus that allows us to reason about private names, we therefore need a category suitable for interpreting probability theory that supports the existence of function types. Unfortunately, a result of Aumann shows that the category of measurable spaces is not cartesian closed, so an alternative category must be considered [Aum61].

Quasi-Borel spaces, a recent development in the field of probabilistic programming, provides just what we want: a cartesian closed category with a suitable interpretation of probability theory [Heu+17]. In this work, we consider the category of quasi-Borel spaces and show that it soundly interprets the nu-calculus, letting names be elements of a standard Borel space and fresh name generation be sampling from a continuous measure.

We then analyze the higher-order features of both the nu-calculus and quasi-Borel spaces. In the nu-calculus, we construct a normal form for terms of first-order type, avoiding the difficulties presented by private names. We also use results of descriptive set theory to analyze the Borel structure of higher-order quasi-Borel spaces. Combining this analysis and the construction of our normal forms, we prove that quasi-Borel spaces are an abstract model of the nu-calculus at first-order types.

1.1 Organization of the Thesis

This thesis is split into five chapters, along with an introduction and a conclusion. Chapter 2 defines the nu-calculus and its operational semantics. It also defines logical relations, which will be featured heavily in the construction of our normal forms. Chapter 3 defines what it means to be a categorical model of the nu-calculus. Chapter 4 describes some basic results of descriptive set theory, along with an analysis of Borel on Borel sets and measures on the space $2^{\mathbb{R}}$ of Borel measurable functions $\mathbb{R} \to 2$. In Chapter 5, we define the category of quasi-Borel spaces, and prove that it is a sound categorical model of the nu-calculus. Finally, Chapter 6 contains our construction of nu-calculus normal forms for terms at first-order type, as well as our proof of abstractness.

2 Nu-Calculus

The nu-calculus is an extension of the simply-typed lambda-calculus constructed by Pitts and Stark to model fresh name generation [PS93]. In this chapter, we will define the syntax, specify big-step call-by-value operational semantics, and define observational equivalence for the nu-calculus. We will also define logical relations between terms, which will provide another, more convenient way to prove observational equivalences.

2.1 Syntax

The syntax of the nu-calculus is based off of the syntax of the simply-typed lambda-calculus and is given in Fig. 2.1. There are two *ground types*, the type B of Booleans and the type N of names, and the only type constructor is that of function type $\sigma \to \sigma$. The terms are an extension of the lambda-calculus, to which we add names and name generation, equality checks for names, truth values and conditionals.

Note that variables and names are different, and that $\lambda x : \sigma.M$ binds the variable x of type σ in M whereas $\nu n.M$ binds the name n in M. We define a *closed term* M to be one with no free variables, and note that a closed term may still have free names.

We will adopt the convention of denoting variables by x, y, z, names by n, m, and sets of names by s, t, u, v.

The term $\nu n.M$ is intended to denote a program that generates a "new", or "fresh", name n, and then evaluates and returns the expression M which may refer to this new name. We will refer to this as *fresh name generation*, and this will be captured in the semantics of Section 2.2 by taking disjoint unions of sets of names. The term $\nu n.n$, for example, is to be interpreted as an expression that generates a new name and returns it

Type	σ	$:= \mathsf{B} \mid \mathsf{N} \mid \sigma \to \sigma$	
Term	M	$:= x$	variables
		$\mid \lambda x : \sigma.M \mid MM$	function abstraction and application
		$\mid n$	names
		$\mid M = M$	equality checks for names
		$\mid \nu n.M$	name generation
		$\mid \mathsf{true} \mid \mathsf{false}$	truth values
		$\mid \mathsf{if}\ M\ \mathsf{then}\ M\ \mathsf{else}\ M$	conditionals

Figure 2.1: Syntax of the nu-calculus

$$\frac{x : \sigma \in \Gamma}{s, \Gamma \vdash x : \sigma} \qquad \frac{n \in s}{s, \Gamma \vdash n : \mathsf{N}} \qquad \frac{}{s, \Gamma \vdash \mathsf{true} : \mathsf{B}} \qquad \frac{}{s, \Gamma \vdash \mathsf{false} : \mathsf{B}}$$

$$\frac{s, \Gamma \sqcup \{x : \sigma\} \vdash M : \tau}{s, \Gamma \vdash \lambda x : \sigma . M : \sigma \to \tau} \qquad \frac{s, \Gamma \vdash M : \sigma \to \tau \quad s, \Gamma \vdash N : \sigma}{s, \Gamma \vdash MN : \tau}$$

$$\frac{s, \Gamma \vdash M : \mathsf{N} \quad s, \Gamma \vdash N : \mathsf{N}}{s, \Gamma \vdash M = N : \mathsf{B}} \qquad \frac{s \sqcup \{n\}, \Gamma \vdash M : \sigma}{s, \Gamma \vdash \nu n . M : \sigma}$$

$$\frac{s, \Gamma \vdash B : \mathsf{B} \quad s, \Gamma \vdash M_T : \sigma \quad s, \Gamma \vdash M_F : \sigma}{s, \Gamma \vdash \mathsf{if}\ B\ \mathsf{then}\ M_T\ \mathsf{else}\ M_F : \sigma}$$

Figure 2.2: Typing rules of the nu-calculus

immediately. This is an important term, and we abbreviate it new.

Notation 2.1. If $s = \{n_1, \ldots, n_k\}$ is a set of names and M is a term, we will let $\nu s.M$ denote $\nu n_1 \ldots \nu n_k . M$.

As usual, we identify terms up to *α-equivalence*, i.e. up to the renaming of bound variables. This is necessary in order to define substitution, but will also be necessary to define the semantics of fresh name generation. We let $M[N/x]$ and $M[N/n]$ denote the *capture-avoiding substitution* of N for the free variable x or free name n in M. Capture-avoiding substitution means that no free variable or name in N is bound in M, which is always possible as we can freely rename the bound variables in M.

We define the *order* of a type by induction on its structure: $\mathrm{Ord}(\sigma) = 0$ for ground types $\sigma \in \{\mathsf{B}, \mathsf{N}\}$, and otherwise

$$\mathrm{Ord}(\sigma \to \tau) = \max\{\mathrm{Ord}(\sigma) + 1, \mathrm{Ord}(\tau)\}.$$

In particular, we will be considering *first-order types*, which are the types of the form $\sigma_1 \to \cdots \to \sigma_n$ with each $\sigma_i \in \{\mathsf{B}, \mathsf{N}\}$ a ground type. Note that we adopt the convention that $\to$ is right-associative, so that $\sigma_1 \to \cdots \to \sigma_{n-1} \to \sigma_n$ means $\sigma_1 \to (\cdots \to (\sigma_{n-1} \to \sigma_n))$.

We interpret the *typing relation* $s, \Gamma \vdash M : \sigma$ to mean that the term M, with free variables in Γ and free names in s, has type σ. Here Γ is a finite context of typed variables, and s a finite set of names. The typing rules are given in Fig. 2.2.

Here, we let $\Gamma \sqcup \{x : \sigma\}$ and $s \sqcup \{n\}$ denote *disjoint unions* of contexts and names. It is always possible to do this by possibly renaming x or n to avoid taking variables or names that already exist in Γ and s.

Notation 2.2. We let

$$\mathrm{Exp}_\sigma(s, \Gamma) = \{M \mid s, \Gamma \vdash M : \sigma\}$$

denote the set of expressions of type σ with free variables and names in s, Γ. We say that a term is in *canonical form* if it is a name, a variable, a truth value or a lambda abstraction, and we denote these

$$\mathrm{Can}_\sigma(s, \Gamma) \subset \mathrm{Exp}_\sigma(s, \Gamma).$$

If Γ is empty, we may omit it and write instead $s \vdash M : \sigma$, $\mathrm{Exp}_\sigma(s)$ and $\mathrm{Can}_\sigma(s)$. Similarly, we may omit s if it is empty as well.

2.2 Operational Semantics

We define big-step operational semantics for the nu-calculus by the *evaluation relation*

$$s \vdash M \Downarrow_\sigma (t)C. \tag{2.1}$$

Here, s, t are disjoint sets of names, $M \in \mathrm{Exp}_\sigma(s)$, and $C \in \mathrm{Can}_\sigma(s \sqcup t)$. If t is empty, we also write $s \vdash M \Downarrow_\sigma C$. This is to be interpreted as the assertion that M, an expression with free names in s, evaluates to the canonical term C, and in doing so generates new, or *fresh*, names t. This relation is defined inductively on the structure of the term M in Fig. 2.3.

As mentioned in Section 2.1, the fact that s, t are disjoint sets of names in (2.1) is crucial in correctly interpreting fresh name generation. For example, recall that $\mathsf{new} = \nu n.n$ is supposed to generate a "new" name n and return it immediately. As such, we would expect that $\vdash (\nu n.n = \nu n.n) \Downarrow_\mathsf{B} (t)\,\mathsf{false}$ for some set of names t. This is enforced in the insistence on distinct names and disjoint unions in the semantics. Indeed, we may be tempted to write

$$n \vdash \nu n.n \Downarrow_\mathsf{N} (n)n$$

in order to conclude that $\vdash (\nu n.n = \nu n.n) \Downarrow_\mathsf{B} (n, n)\,\mathsf{true}$. However this derivation is invalid as we are forced to create distinct names in the LOCAL rule. Instead, we must rename one of the names to correctly derive

$$\frac{\vdash \nu n.n \Downarrow_\mathsf{N} (n)n \quad n \vdash \nu n'.n' \Downarrow_\mathsf{N} (n')n' \quad n \neq n'}{\vdash (\nu n.n = \nu n'.n') \Downarrow_\mathsf{B} (n, n')\,\mathsf{false}}\,.$$

Additionally, we note that these semantics are call-by-value, so that for example $\vdash (\lambda x : \mathsf{N}.x = x)\mathsf{new} \Downarrow_\mathsf{B} \mathsf{true}$, because we evaluate new before applying it to to $(\lambda x : \mathsf{N}.x = x)$:

$$\frac{\vdash \mathsf{new} \Downarrow_\mathsf{N} (n)n \quad n \vdash (n = n) \Downarrow_\mathsf{B} \mathsf{true}}{\vdash (\lambda x : \mathsf{N}.x = x)\mathsf{new} \Downarrow_\mathsf{N} \mathsf{true}}\ \text{APP}.$$

These operational semantics are nice, as it can be shown that evaluation always terminates:

Theorem 2.3 (Termination [PS93])**.** *For all $M \in \mathrm{Exp}_\sigma(s)$, there are some names t disjoint from s and a term $C \in \mathrm{Can}_\sigma(s \sqcup t)$ such that $s \vdash M \Downarrow_\sigma (t)C$. Moreover, this is unique up to renaming of bound variables and names in t.*

$$\frac{C \in \mathrm{Can}_\sigma(s)}{s \vdash C \Downarrow_\sigma C}\ \mathrm{CAN}$$

$$\frac{s \vdash B \Downarrow_{\mathsf{B}} (t_1)\,\mathsf{true} \quad s \sqcup t_1 \vdash M_T \Downarrow_\sigma (t_2)C}{s \vdash \mathsf{if}\ B\ \mathsf{then}\ M_T\ \mathsf{else}\ M_F \Downarrow_\sigma (t_1 \sqcup t_2)C}\ \mathrm{CONDT}$$

$$\frac{s \vdash B \Downarrow_{\mathsf{B}} (t_1)\,\mathsf{false} \quad s \sqcup t_1 \vdash M_F \Downarrow_\sigma (t_2)C}{s \vdash \mathsf{if}\ B\ \mathsf{then}\ M_T\ \mathsf{else}\ M_F \Downarrow_\sigma (t_1 \sqcup t_2)C}\ \mathrm{CONDF}$$

$$\frac{s \vdash M \Downarrow_{\mathsf{B}} (t_1)n \quad s \sqcup t_1 \vdash N \Downarrow_{\mathsf{B}} (t_2)n \quad n \in s}{s \vdash (M = N) \Downarrow_{\mathsf{B}} (t_1 \sqcup t_2)\,\mathsf{true}}\ \mathrm{EQ}$$

$$\frac{s \vdash M \Downarrow_{\mathsf{B}} (t_1)n \quad s \sqcup t_1 \vdash N \Downarrow_{\mathsf{B}} (t_2)m \quad m \neq n}{s \vdash (M = N) \Downarrow_{\mathsf{B}} (t_1 \sqcup t_2)\,\mathsf{false}}\ \mathrm{NEQ}$$

$$\frac{s \sqcup \{n\} \vdash M \Downarrow_\sigma (t)C \quad n \notin s \sqcup t}{s \vdash \nu n.M \Downarrow_\sigma (t \sqcup \{n\})C}\ \mathrm{LOCAL}$$

$$\frac{\begin{array}{c} s \vdash M \Downarrow_{\sigma\to\tau} (t_1)\lambda x : \sigma.M' \quad s \sqcup t_1 \vdash N \Downarrow_\tau (t_2)C' \\ s \sqcup t_1 \sqcup t_2 \vdash M'[C'/x] \Downarrow_\tau (t_3)C \end{array}}{s \vdash MN \Downarrow_\tau (t_1 \sqcup t_2 \sqcup t_3)C}\ \mathrm{APP}$$

Figure 2.3: Operational semantics of the nu-calculus

2.3 Observational Equivalence

Now that we know how programs are executed, we would like to identify expressions that behave the same way.

Example 2.4. Consider the term

$$\nu n.\lambda x : \mathsf{N}.x = n.$$

This term generates a fresh name n, and then returns the function $\lambda x : \mathsf{N}.x = n$. Since this name is fresh, we would expect that it cannot occur anywhere else in a program, and so we expect that it is always the case when calling this function that $x \neq n$. Thus, we would expect this function to behave identically to the constant function $\lambda x : \mathsf{N}.\ \mathsf{false}$.

To capture this idea, we define the notion of *observational equivalence* between terms. Here, we consider a *program context* to be an expression $P[-]$ with a hole in it, denoted by $-$, and we let $P[M]$ represent replacing all occurrences of the hole $-$ in P with the term M, possibly capturing its free variables and names.

Definition 2.5 (Observational Equivalence)**.** If $M_1, M_2 \in \mathrm{Exp}_\sigma(s)$, we say that M_1 and M_2 are observationally equivalent, written $s \vdash M_1 \approx_\sigma M_2$, if for all program contexts $P[-]$ and all $b \in \mathsf{B}$, we have

$$\exists t_1\,(s \vdash P[M_1] \Downarrow_{\mathsf{B}} (t_1)b) \iff \exists t_2\,(s \vdash P[M_2] \Downarrow_{\mathsf{B}} (t_2)b)$$

whenever this makes sense (i.e. whenever $P[M_i]$ is a well-formed expression of type B).

This formally captures the idea above, that we want to identify expressions that behave the same way, as we are saying that M_1, M_2 are equivalent if they result in the same behaviour when used in any program. For example, we will establish in the next section that

$$\lambda x : \mathsf{N}.\ \mathsf{false} \approx_{\mathsf{N}\to\mathsf{B}} \nu n.\lambda x : \mathsf{N}.x = n. \tag{2.2}$$

The Context Lemma gives a simpler formulation of observational equivalence. It states that in order to establish the observational equivalence of the closed terms M_1 and M_2, it suffices to check that they agree when applied to simple functions in $\mathrm{Can}_{\sigma\to\mathsf{B}}(s)$.

Lemma 2.6 (Context Lemma [Sta96])**.** *If $M_1, M_2 \in \mathrm{Exp}_\sigma(s)$, then $s \vdash M_1 \approx_\sigma M_2$ iff for all $b \in \mathsf{B}$ and all $\lambda x : \sigma.N \in \mathrm{Can}_{\sigma\to\mathsf{B}}(s)$,*

$$\exists t_1\,(s \vdash (\lambda x : \sigma.N)M_1 \Downarrow_{\mathsf{B}} (t_1)b) \iff \exists t_2\,(s \vdash (\lambda x : \sigma.N)M_2 \Downarrow_{\mathsf{B}} (t_2)b)\,.$$

Lemma 2.7 ([PS93, Corollary 6])**.** *We have the following facts about observational equivalence:*

1. *If $M \in \mathrm{Exp}_\sigma(s)$ and $n \notin s$, then $s \vdash \nu n.M \approx_\sigma M$.*
2. *If $M \in \mathrm{Exp}_\sigma(s \sqcup \{n, n'\})$ then $s \vdash \nu n.\nu n'.M \approx_\sigma \nu n'.\nu n.M$.*

3. *If* $s \vdash M \Downarrow_\sigma (t)C$, *then* $s \vdash M \approx_\sigma \nu t.C$.

4. *If* $s, x : \sigma \vdash M : \tau$ *and* $C \in \mathrm{Can}_\sigma(s)$, *then* $s \vdash (\lambda x : \sigma.M)C \approx_\tau M[C/x]$.

Proving observational equivalence between expressions of the nu-calculus is in general difficult, as it is hard to reason about the effects of fresh name generation and predicting where these names can end up.

Example 2.8. Stark provides examples in [Sta96] of some of the difficulties one can encounter reasoning about observational equivalence.

For example, in a call-by-value language line the nu-calculus, a function application $(\lambda x.M)N$ may not be observationally equivalent to the term $M[N/x]$, substituting x for N in M, if N is not a canonical term. This is exemplified by the non-equivalence

$$(\lambda x : \mathsf{N}.x = x)\mathsf{new} \not\approx_{\mathsf{B}} \mathsf{new} = \mathsf{new},$$

which holds because the left-hand-side evaluates to true whereas the right-hand-side evaluates to false.

Another challenge is that ν and λ don't commute. For example, the test function F given by $\lambda f : \mathsf{B} \to \mathsf{N}.(f\,\mathsf{true} = f\,\mathsf{true})$ witnesses that

$$\nu n.\lambda x : \mathsf{B}.n \not\approx_{\mathsf{B}\to\mathsf{N}} \lambda x : \mathsf{B}.\nu n.n.$$

The problem here is that in the first term the name n is generated once and bound to the closure of $\lambda x : \mathsf{B}.n$, while the second term generates a fresh name every time it is called. Thus, $F(\nu n.\lambda x : \mathsf{B}.n) \Downarrow_{\mathsf{B}} \mathsf{true}$, whereas $F(\lambda x : \mathsf{B}.\nu n.n) \Downarrow_{\mathsf{B}} \mathsf{false}$.

2.4 Logical Relations

In general it is difficult to reason about observational equivalence. We will now define logical relations, which provide us with a convenient way to prove observational equivalences such as (2.2).

Notation 2.9. Given sets of names s_1, s_2 and a relation $R \subseteq s_1 \times s_2$, we write $R : s_1 \rightleftharpoons s_2$ to mean that R is a partial bijection between s_1 and s_2. If $s \subseteq t$ are sets of names, we let $Id_s : t \rightleftharpoons t$ be the identity on s.

Definition 2.10 (Logical Relations [PS93])**.** For all types σ, sets of names s_1, s_2, and partial bijections $R : s_1 \rightleftharpoons s_2$, we will define the relations

$$R^{can}_\sigma \subseteq \mathrm{Can}_\sigma(s_1) \times \mathrm{Can}_\sigma(s_2),$$
$$R^{exp}_\sigma \subseteq \mathrm{Exp}_\sigma(s_1) \times \mathrm{Exp}_\sigma(s_2)$$

by induction on σ as follows:

$$\begin{aligned}
b_1\, R^{can}_{\mathsf{B}}\, b_2 &\iff b_1 = b_2 \\
n_1\, R^{can}_{\mathsf{N}}\, n_2 &\iff n_1\, R\, n_2 \\
(\lambda x : \sigma.M_1)\, R^{can}_{\sigma\to\tau}\, (\lambda x : \sigma.M_2) &\iff \forall R' : t_1 \rightleftharpoons t_2, C_i \in \mathrm{Can}_\sigma(s_i \sqcup t_i), C_1\, (R \sqcup R')^{can}_\sigma\, C_2 \\
&\qquad \text{we have } M_1[C_1/x]\, (R \sqcup R')^{exp}_\tau\, M_2[C_2/x] \\
M_1\, R^{exp}_\sigma M_2 &\iff \exists R' : t_1 \rightleftharpoons t_2, C_i \in \mathrm{Can}_\sigma(s_i \sqcup t_i), C_1\, (R \sqcup R')^{can}_\sigma\, C_2 \\
&\qquad \text{such that } s_1 \vdash M_1 \Downarrow_\sigma (t_1)C_1 \text{ and } s_2 \vdash M_2 \Downarrow_\sigma (t_2)C_2
\end{aligned}$$

Notation 2.11. Note that R^{exp}_σ coincides with R^{can}_σ when restricted to canonical terms. We will therefore write $M_1\, R_\sigma\, M_2$ unambiguously.

Remark 2.12. We note that if $M \in \mathrm{Exp}_\sigma(s)$ is any expression, then it is clear that $s \vdash M \approx_\sigma M$ and $M\, (Id_s)_\sigma\, M$.

The following proposition will be useful in establishing logical relations between terms:

Proposition 2.13 ([PS93, Proposition 21])**.** *Suppose $C_i \in \mathrm{Can}_{\sigma\to\tau}(s_i)$ and $R : s_1 \rightleftharpoons s_2$.*

1. *If $\sigma = \mathsf{B}$, then $C_1\, R_{\sigma\to\tau}\, C_2$ if and only if for $b = \mathsf{true}, \mathsf{false}$ we have $C_1 b\, R_\tau\, C_2 b$.*
2. *If $\sigma = \mathsf{N}$, then $C_1 R_{\sigma\to\tau} C_2$ if and only if for all $(m, n) \in R$ we have $C_1 m\, R_\tau\, C_2 n$ and for $n \notin s_1 \cup s_2$ we have $C_1 n\, (R \sqcup Id_{\{n\}})_\tau\, C_2 n$.*

We offer the following intuitive explanation of the logical relations. Suppose that $M_1 \in \mathrm{Exp}_\sigma(s_1), M_2 \in \mathrm{Exp}_\sigma(s_2), R : s_1 \rightleftharpoons s_2$ and $M_1\, R_\sigma\, M_2$. The names in $s_1 \setminus \mathrm{Dom}(R)$ represent names that are not leaked to the environment provided it does not already know about these names, and similarly for $s_2 \setminus \mathrm{Cod}(R)$. On the other hand, $R : \mathrm{Dom}(R) \to \mathrm{Cod}(R)$ represents a way to rename the names in M_1 to get a term that is observationally equivalent to M_2.

Thus, logical relations serve as witnesses to observational equivalence and provide a convenient way to prove equivalences such as (2.2). They do not in general capture observational equivalence, but they do at first-order.

Theorem 2.14 ([PS93, Theorems 14, 22])**.** *Suppose that $M_1, M_2 \in \mathrm{Exp}_\sigma(s)$. Then*

$$M_1\, (Id_s)_\sigma\, M_2 \implies s \vdash M_1 \approx_\sigma M_2.$$

Moreover, if σ is a first-order type, the converse holds:

$$s \vdash M_1 \approx_\sigma M_2 \implies M_1\, (Id_s)_\sigma\, M_2.$$

Example 2.15. We are now able to prove the equivalence in (2.2). To prove this, it suffices to show that

$$(\lambda x : \mathsf{N}.\, \mathsf{false})\, \emptyset_{\mathsf{N}\to\mathsf{B}}\, (\lambda x : \mathsf{N}.x = n),$$

where $\emptyset : \emptyset \rightleftharpoons \{n\}$ is the empty relation. This in turn follows from the assertion that

$$\mathsf{false}\, (Id_{\{m\}})_{\mathsf{B}}\, (m = n)$$

for any name $m \neq n$, which is clear as the right-hand-side evaluates to false.

3 Categorical Models of the Nu-Calculus

We would like to provide denotational semantics for the nu-calculus, as this will provide another way to reason about observational equivalence. In this chapter, we will outline Stark's framework for constructing categorical models of the nu-calculus [Sta96]. In order to construct these models, it is necessary to distinguish between expressions, which may generate fresh names, and canonical terms, which are the results of computations. To do this, we will explicitly distinguish between values of a type σ and computations that result in a value of type σ by first interpreting the nu-calculus in a metalanguage suitable for making this distinction. We will then interpret this metalanguage in a suitable category, following Moggi's monadic interpretation of computation [Mog91].

3.1 The Metalanguage

The metalanguage Stark introduced is similar to the nu-calculus and is given in Fig. 3.1 [Sta96]. We have the same basic types of Booleans and Names, as well as the function type constructor. However, we add a new type constructor T, so that if A is a type, TA is to be interpreted as the type of *computations of type* A.

Similarly, we add to the syntax the term constructor $[a]$, which is to be interpreted as the trivial computation returning the value a, and let-expressions to model sequential computation: $(\mathsf{let}\ x \Leftarrow a\ \mathsf{in}\ b)$ is interpreted as computing a, letting the value of this computation be x, and returning the computation $b(x)$. We also add the constant $\mathsf{new} : TName$. This will be interpreted as the computation returning a fresh name, corresponding to the term $\nu n.n$ in the nu-calculus.

The typing rules are given in Fig. 3.2, where we let $\Gamma, x : A$ denote the disjoint union $\Gamma \sqcup \{x : A\}$. Note that unlike for the nu-calculus, the metalanguage does not distinguish between variables and names, and this is reflected both in the syntax and typing rules, as the typing relation relies only on a context Γ and not on an additional set of free names.

We also introduce an equational logic to reason about the equality of terms. As we plan to interpret the metalanguage in a category, this will allow us to reason about the equality of objects in this category equationally, instead of through diagrams.

In this logic, we will make assertions of the form $\Gamma \vdash a = a'$ where $\Gamma \vdash a, a' : A$. More generally, if Φ is a set of equations in context Γ, we will reason about sequents of the form $\Gamma; \Phi \vdash a = a'$.

Type	A	$:= Bool \mid Name \mid A \to A$	
		$\mid TA$	computations
Term	a	$:= x$	variables
		$\mid \lambda x : A.a \mid aa$	function abstraction and application
		$\mid \mathsf{true} \mid \mathsf{false}$	truth values
		$\mid cond(a, a, a)$	conditionals
		$\mid eq(a, a)$	equality of names
		$\mid \mathsf{new}$	name generation
		$\mid [a]$	trivial computation
		$\mid \mathsf{let}\ x \Leftarrow a\ \mathsf{in}\ a$	sequential computation

Figure 3.1: Syntax of the metalanguage

$$\frac{x : A \in \Gamma}{\Gamma \vdash x : A} \qquad \frac{}{\Gamma \vdash \mathsf{new} : TName} \qquad \frac{}{\Gamma \vdash \mathsf{true} : Bool} \qquad \frac{}{\Gamma \vdash \mathsf{false} : Bool}$$

$$\frac{\Gamma \vdash n : Name \quad \Gamma \vdash m : Name}{\Gamma \vdash eq(n, m) : Bool} \qquad \frac{\Gamma \vdash b : Bool \quad \Gamma \vdash a : A \quad \Gamma \vdash a' : A}{\Gamma \vdash cond(b, a, a') : A}$$

$$\frac{\Gamma, x : A \vdash a : B}{\Gamma \vdash \lambda x : A.a : A \to B} \qquad \frac{\Gamma \vdash f : A \to B \quad \Gamma \vdash a : A}{\Gamma \vdash fa : B}$$

$$\frac{\Gamma \vdash a : A}{\Gamma \vdash [a] : TA} \qquad \frac{\Gamma \vdash e : TA \quad \Gamma, x : A \vdash e' : TB}{\Gamma \vdash \mathsf{let}\ x \Leftarrow e\ \mathsf{in}\ e' : TB}$$

Figure 3.2: Typing rules of the metalanguage

The axioms for this equational logic consist of the basic axioms

$$\frac{\phi \in \Phi}{\Gamma; \Phi \vdash \phi} \qquad \frac{\Gamma; \Phi \vdash \phi \quad \Gamma; \Psi, \phi \vdash \psi}{\Gamma; \Phi \cup \Psi \vdash \psi} \ ,$$

axioms asserting that equality is an equivalence relation, axioms asserting that equality is preserved by the term forming rules of Fig. 3.1, and the rules given in Fig. 3.3.

These rules consist of basic rules for Booleans, names, functions and computations, the MONO rule for ground types, and the rules DROP, SWAP and FRESH. The FRESH rule asserts that new names are distinct, and the DROP and SWAP rules assert, respectively, that generating unused names and swapping the order of generated names doesn't matter. Alternative formulations of these rules are discussed in [Sta96, p. 12]. Note that while FRESH asserts the freshness of a new name with respect to a single other name, it implies freshness of new names with respect to any finite set of other names (cf. [Sta94, Lemma 3.2]). The MONO rule asserts that the map $a \mapsto [a]$ is injective. While both Moggi and Stark require that this holds in general, a more modern approach — and the one we will take in this thesis — is to require injectivity only for terms at ground types.

3.2 Interpreting the Nu-Calculus in the Metalanguage

The translation of types, terms, and contexts from nu-calculus to the metalanguage is defined in Fig. 3.4. Here we define a function $C \mapsto |C|$ from canonical terms of the nu-calculus to the metalanguage, as well as functions $\sigma \mapsto [\![\sigma]\!]$, $M \mapsto [\![M]\!]$ and $s, \Gamma \mapsto [\![s, \Gamma]\!]$ mapping the types, expressions and contexts of the nu-calculus to the types, terms and contexts of the metalanguage, respectively. Note that viewed through this translation functions return computations. Additionally we see that for canonical terms C we have both the interpretation $|C|$ of C as a value and an interpretation $[\![C]\!] = [|C|]$ of C as a computation returning $|C|$.

Lemma 3.1 ([Sta96, lemma 1])**.** *For any nu-calculus canonical term C and expression M,*

$$\begin{aligned} s, \Gamma \vdash C : \sigma &\iff [\![s, \Gamma]\!] \vdash |C| : [\![\sigma]\!] \,, \\ s, \Gamma \vdash M : \sigma &\iff [\![s, \Gamma]\!] \vdash [\![M]\!] : T\,[\![\sigma]\!] \,. \end{aligned}$$

This interpretation of the nu-calculus in the metalanguage is *sound*, meaning that equality is preserved by the operational semantics of the nu-calculus.

Notation 3.2. If $s = \{n_1, \ldots, n_k\}$ is a set of distinct names, we let

$$(\neq s) = \{eq(n_i, n_j) = \mathsf{false} \mid i \neq j\}$$

be the set of formulas in the metalanguage asserting that the names in s are distinct, and we let $\mathsf{let}\ s \Leftarrow \mathsf{new}\ \mathsf{in}\ e$ be an abbreviation for the term

$$\mathsf{let}\ n_1 \Leftarrow \mathsf{new}\ \mathsf{in} \ldots \mathsf{let}\ n_k \Leftarrow \mathsf{new}\ \mathsf{in}\ e.$$

Functions:

$$\frac{\Gamma, x : A \vdash b : B \quad \Gamma \vdash a : A}{\Gamma \vdash (\lambda x : A.b)a = b[a/x]}\ \beta \qquad \frac{\Gamma \vdash f : A \to B}{\Gamma \vdash f = \lambda x : A.fx}\ \eta$$

Computations:

$$\frac{\Gamma \vdash e : TA}{\Gamma \vdash \mathsf{let}\ x \Leftarrow e\ \mathsf{in}\ [x] = e} \qquad \frac{\Gamma \vdash a, a' : A \quad A \in \{Name, Bool\}}{\Gamma; [a] = [a'] \vdash a = a'}\ \text{MONO}$$

$$\frac{\Gamma \vdash a : A \quad \Gamma, x : A \vdash e : TB}{\Gamma \vdash \mathsf{let}\ x \Leftarrow [a]\ \mathsf{in}\ e = e[a/x]}$$

$$\frac{\Gamma \vdash e : TA \quad \Gamma, x : A \vdash e' : TB \quad \Gamma, x' : B \vdash e'' : TC}{\Gamma \vdash \mathsf{let}\ x' \Leftarrow (\mathsf{let}\ x \Leftarrow e\ \mathsf{in}\ e')\ \mathsf{in}\ e'' = \mathsf{let}\ x \Leftarrow e\ \mathsf{in}\ (\mathsf{let}\ x' \Leftarrow e'\ \mathsf{in}\ e'')}$$

Booleans:

$$\frac{\Gamma; \Phi, b = \mathsf{true} \vdash \phi \quad \Gamma; \Phi, b = \mathsf{false} \vdash \phi}{\Gamma; \Phi \vdash \phi} \qquad \frac{\Gamma; \Phi \vdash \mathsf{true} = \mathsf{false}}{\Gamma; \Phi \vdash \phi}$$

$$\frac{\Gamma \vdash a, a' : A}{\Gamma \vdash cond(\mathsf{true}, a, a') = a} \qquad \frac{\Gamma \vdash a, a' : A}{\Gamma \vdash cond(\mathsf{false}, a, a') = a'}$$

Testing names:

$$\frac{\Gamma \vdash n : Name}{\Gamma \vdash eq(n, n) = \mathsf{true}} \qquad \frac{\Gamma \vdash n, n' : Name}{\Gamma; eq(n, n') = \mathsf{true} \vdash n = n'}$$

Generating names:

$$\frac{\Gamma \vdash e : TA \quad n : Name \notin \Gamma}{\Gamma \vdash e = \mathsf{let}\ n \Leftarrow \mathsf{new}\ \mathsf{in}\ e}\ \text{DROP}$$

$$\frac{\Gamma, n : Name, n' : Name \vdash e : TA}{\Gamma \vdash \mathsf{let}\ n \Leftarrow \mathsf{new}\ \mathsf{in}\ \mathsf{let}\ n' \Leftarrow \mathsf{new}\ \mathsf{in}\ e = \mathsf{let}\ n' \Leftarrow \mathsf{new}\ \mathsf{in}\ \mathsf{let}\ n \Leftarrow \mathsf{new}\ \mathsf{in}\ e}\ \text{SWAP}$$

$$\frac{\Gamma \vdash n : Name \quad \Gamma, n' : Name; \Phi, eq(n, n') = \mathsf{false} \vdash e = e'}{\Gamma; \Phi \vdash \mathsf{let}\ n' \Leftarrow \mathsf{new}\ \mathsf{in}\ e = \mathsf{let}\ n' \Leftarrow \mathsf{new}\ \mathsf{in}\ e'}\ \text{FRESH}$$

Figure 3.3: Logic of the metalanguage

Types:

$$\begin{aligned}
[\![\mathsf{B}]\!] &= Bool \\
[\![\mathsf{N}]\!] &= Name \\
[\![\sigma \to \tau]\!] &= [\![\sigma]\!] \to T\,[\![\tau]\!]
\end{aligned}$$

Canonical terms:

$$\begin{aligned}
|x| &= x \\
|n| &= n \\
|\,\mathsf{true}\,| &= \mathsf{true} \\
|\,\mathsf{false}\,| &= \mathsf{false} \\
|\lambda x : \sigma.M| &= \lambda x : [\![\sigma]\!]\,.\,[\![M]\!]
\end{aligned}$$

Expressions:

$$\begin{aligned}
[\![C]\!] &= [|C|] \quad C \text{ canonical} \\
[\![\mathsf{if}\ B\ \mathsf{then}\ M_T\ \mathsf{else}\ M_F]\!] &= \mathsf{let}\ b \Leftarrow [\![B]\!]\ \mathsf{in}\ cond(b, [\![M_T]\!], [\![M_F]\!]) \\
[\![M = N]\!] &= \mathsf{let}\ m \Leftarrow [\![M]\!]\ \mathsf{in}\ \mathsf{let}\ n \Leftarrow [\![N]\!]\ \mathsf{in}\ [eq(m,n)] \\
[\![\nu n.M]\!] &= \mathsf{let}\ n \Leftarrow \mathsf{new}\ \mathsf{in}\ [\![M]\!] \\
[\![FM]\!] &= \mathsf{let}\ f \Leftarrow [\![F]\!]\ \mathsf{in}\ \mathsf{let}\ m \Leftarrow [\![M]\!]\ \mathsf{in}\ fm
\end{aligned}$$

Contexts:

$$\begin{aligned}
[\![s, \Gamma]\!] &= n_1 : Name, \ldots, n_k : Name, x_1 : [\![\sigma_1]\!], \ldots, x_l : [\![\sigma_l]\!] \\
\text{where } s &= \{n_1, \ldots, n_k\} \\
\Gamma &= \{x_1 : \sigma_1, \ldots, x_l : \sigma_l\}
\end{aligned}$$

Figure 3.4: Translating nu-calculus to the metalanguage

Theorem 3.3 (Soundness [Sta96]). *Let* $M \in \mathrm{Exp}_\sigma(s)$. *If* $s \vdash M \Downarrow_\sigma (t)C$ *for* $C \in \mathrm{Can}_\sigma(s \sqcup t)$, *then*

$$[\![s]\!] ; (\neq s) \vdash [\![M]\!] = \mathsf{let}\ t \Leftarrow \mathsf{new}\ \mathsf{in}\ [\![C]\!]$$

is provable in the metalanguage.

We also have that the metalanguage is *adequate*, meaning that if two terms have equal denotations then they are observationally equivalent. It is also *abstract at ground types*, meaning that if two terms of ground type are observationally equivalent, then they have equal denotations.

Theorem 3.4 (Adequacy [Sta96]). *For all* $M_1, M_2 \in \mathrm{Exp}_\sigma(s)$, *if*

$$[\![s]\!] ; (\neq s) \vdash [\![M_1]\!] = [\![M_2]\!]$$

is provable in the metalanguage, then $s \vdash M_1 \approx_\sigma M_2$.

Theorem 3.5 (Abstractness at Ground Types [Sta96]). *Let* σ *be a ground type. For all* $M_1, M_2 \in \mathrm{Exp}_\sigma(s)$, *if* $s \vdash M_1 \approx_\sigma M_2$, *then*

$$[\![s]\!] ; (\neq s) \vdash [\![M_1]\!] = [\![M_2]\!]$$

is provable in the metalanguage.

3.3 Categorical Models

We will now give Stark's construction of categorical models of the nu-calculus [Sta96]. To do this, we interpret the metalanguage in a suitable category $\mathcal{C}$. The standard method of doing this is by interpreting types in the metalanguage as objects in $\mathcal{C}$, and expressions of the metalanguage as arrows in $\mathcal{C}$. Thus, we need, for each way of constructing types and terms in the metalanguage, a corresponding way to construct objects and arrows in $\mathcal{C}$.

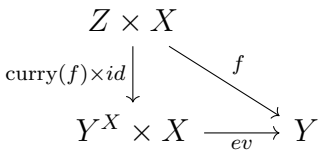
Specifically, we will need an object corresponding to the Boolean type B, an object corresponding to the name type N, and "object constructors" corresponding to the type constructors $\to$ and T. More precisely, if X, Y are objects of $\mathcal{C}$ corresponding to the types A, B, we need to be able to find objects Y^X and TX in $\mathcal{C}$ to correspond to the types $A \to B$ and TA respectively. Similarly, we will need to be able to construct maps corresponding to let-expressions, lambda abstraction, $\mathsf{true}, \mathsf{false}, \mathsf{new}, [\cdot], eq(\cdot,\cdot)$, and $cond(\cdot,\cdot,\cdot)$.

Of course, all of these objects and maps should be able to soundly interpret the metalanguage, including the equational reasoning described in Fig. 3.3. In order to do this, we will consider *cartesian closed categories*, which are categories with a suitable notion of function spaces, and *strong monads*, which can be used to soundly interpret computations in a category (cf. [Mog91]). Finally, we will list the additional conditions a category must satisfy to soundly interpret names, Booleans and new.

3.3.1 Cartesian Closed Categories and Strong Monads

We begin by defining function spaces in a category, in order to interpret function types.

Definition 3.6. A category $\mathcal{C}$ with finite products and a terminal object is *cartesian closed* if for all objects X, Y, there is an *exponential object* Y^X and an *evaluation map* $ev : Y^X \times X \to Y$ such that for any function $f : Z \times X \to Y$, there is a unique map $\text{curry}(f) : Z \to Y^X$ such that the following diagram commutes:



Example 3.7. The category of sets is cartesian closed, as we can take Y^X to be the set of set-theoretic functions $X \to Y$, ev the usual evaluation map, and let curry correspond to currying: $\text{curry}(f)(z)(x) = f(z, x)$.

Next, we want to define a structure on $\mathcal{C}$ that will allow us to interpret the types of computations TA present in the metalanguage. Therefore, for a given object X, we must have a corresponding object TX which should serve as an object of "computations" of object X.

In addition to this corresponding object TX, we need a map $X \to TX$ that should soundly interpret the term constructor $[\cdot]$ of the trivial computation.

We also need a way to encode let-expressions. A metalanguage expression of the form $x : A \vdash e' : TB$ should correspond to an arrow $X \to TY$ in $\mathcal{C}$. However, the expression $(\mathsf{let}\ x \Leftarrow e\ \mathsf{in}\ e')$ is a function taking $e : TA$ to a term of type TB, and so we need to be able to *lift* our arrow $X \to TY$ to an arrow $TX \to TY$.

Such a structure T on $\mathcal{C}$ is called a *monad*. The monad laws, which we present below, are motivated by the program logic (cf. Computations, Fig. 3.3) that we wish to model.

Definition 3.8. A *monad* on a category $\mathcal{C}$ is a functor $T : \mathcal{C} \to \mathcal{C}$, maps $\eta_X : X \to TX$ for all objects X, and a lift operation $(f : X \to TY) \mapsto (f^* : TX \to TY)$ satisfying

- $\eta_X^* = id_{TX}$,
- $f^* \circ \eta_X = f$ for $f : X \to TY$, and
- $g^* \circ f^* = (g^* \circ f)^*$ for $f : X \to TY$ and $g : Y \to TZ$.

Example 3.9. If we take a set X and let X^* denote the free monoid on X, then the map $X \mapsto X^*$ defines a monad on the category of sets, where $\eta(x) = x$, and if $f : X \to Y^*$, then $f^* : X^* \to Y^*$ is given by concatenation:

$$f^*(x_1 x_2 \cdots x_n) = f(x_1) f(x_2) \cdots f(x_n).$$

Given a monad, η will correspond to the term constructor $[\cdot]$ of the trivial computation and the lift will correspond to let-expressions.

It turns out that in order to properly interpret contexts, we need the stronger requirement that T is a *strong monad*, as explained in [Mog91, remark 3.1].

Notation 3.10. We use π_X, π_Y to denote the projections $X \times Y \to X$ and $X \times Y \to Y$, respectively. If $f : X \to X', g : Y \to Y'$, we let $f \times g$ denote the corresponding map $X \times Y \to X' \times Y'$. If $f : X \to Y, g : X \to Z$, we let $\langle f, g \rangle$ denote the corresponding map $X \to Y \times Z$.

Definition 3.11. A *strong monad* T on a cartesian closed category $\mathcal{C}$ is a functor $T : \mathcal{C} \to \mathcal{C}$, maps $\eta_X : X \to TX$ for all objects X, and a strengthened lift operation $(f : X \times Y \to TZ) \mapsto (f^* : X \times TY \to TZ)$ satisfying

- $(\eta_Y \circ \pi_Y)^* = \pi_{TY}$,
- $f^* \circ (id_X \times \eta_Y) = f$ for $f : X \times Y \to TZ$, and
- $g^* \circ \langle \pi_X, f^* \rangle = (g^* \circ \langle \pi_X, f \rangle)^*$ for $f : X \times Y \to TZ$ and $g : X \times Z \to TZ'$.

Example 3.12. The free monoid monad (cf. Example 3.9) is strong: given $f : X \times Y \to Z^*$, we define

$$f^*(x, y_1 y_2 \cdots y_n) = f(x, y_1) f(x, y_2) \cdots f(x, y_n).$$

More generally, every monad on the category of sets is strong, as the strengthened lift can be constructed from the normal lift $TZ^Y \to TZ^{TY}$ of the monad by currying.

We note that monads over an arbitrary category $\mathcal{C}$ may not always be strong, even if $\mathcal{C}$ is cartesian closed, as the lift $TZ^Y \to TZ^{TY}$ need not be a morphism in the category.

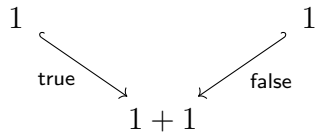
3.3.2 Categorical Models for the Nu-Calculus

We are now ready to show how to interpret the metalanguage in a category $\mathcal{C}$. We will give a translation from types in the metalanguage to objects in $\mathcal{C}$, and terms in the metalanguage to arrows in $\mathcal{C}$.

In order to soundly interpret function types $A \to B$ and types of computations TA we require that $\mathcal{C}$ is a cartesian closed category with a strong monad T, following the reasoning of Section 3.3.1. If A, B are types that map to objects X, Y in $\mathcal{C}$, we will map the type $A \to B$ to the object B^A, and the type TA to the object TX.

We interpret the types $Bool$ and $Name$ as follows. Let 1 denote the terminal object of $\mathcal{C}$. We ask that the coproduct $1 + 1$ of 1 with itself exists, and we map the type $Bool$ to this coproduct $1 + 1$. For $Name$, we will choose a distinguished object N in $\mathcal{C}$, and we will map the type $Name$ to N.

We now give the translation of terms of the metalanguage to arrows in $\mathcal{C}$. The terms true and false will denote the two inclusions



of the coproduct $1 + 1$. We choose a distinguished arrow $1 \to TN$ to correspond to the term new. Additionally, we ask that there is a distinguished function $eq : N \times N \to 1 + 1$, and we let this correspond to the term $eq(\cdot, \cdot)$.

Finally, for every object X in $\mathcal{C}$, we can define the function $cond_X : 2 \times X \times X \to X$ by the composition

$$(1+1) \times X \times X \xrightarrow{[\pi_1,\pi_2]\times id_{X\times X}} X^{(X\times X)} \times X \times X \xrightarrow{ev} X.$$

Here π_1, π_2 correspond respectively to the left and right projections $X \times X \to X$, so in particular $cond_X(\mathsf{true}) = \pi_1$ and $cond_X(\mathsf{false}) = \pi_2$. We will then map the term $cond(\cdot,\cdot,\cdot)$ to the arrows $cond_X$ in $\mathcal{C}$, where X depends on the type of the term containing $cond(\cdot,\cdot,\cdot)$.

The interpretations of lambda abstraction, function application, $[\cdot]$ and let-expressions in $\mathcal{C}$ are given by currying, composition of arrows and the η maps and lift operation of the monad, as described in Section 3.3.1.

Now that we have defined the translation of the basic terms and term constructors, we let the full translation of the metalanguage to $\mathcal{C}$ be given by induction on the structure of terms. The details of this translation are given in Fig. 3.5. In this figure, we take expressions of the form

$$\frac{\phi_1 \quad \dots \quad \phi_n}{\psi} \mapsto \frac{f_1 \quad \dots \quad f_n}{g}$$

to mean that if ψ is derived from the sequents $\phi_1, \dots, \phi_n$ in the metalanguage and $f_1, \dots, f_n$ are the arrows in $\mathcal{C}$ corresponding to $\phi_1, \dots, \phi_n$, which we have already constructed by induction, then g is the arrow corresponding to ψ.

The last thing we need in order to translate the metalanguage to the category $\mathcal{C}$ is an object corresponding to contexts of *distinct* names. That is, if s is a set of names and $N^{|s|} = N \times \cdots \times N$ the corresponding object in $\mathcal{C}$, we would like to construct the subobject $(\neq s)$ of $N^{|s|}$ corresponding to the $|s|$-tuples of *distinct* names in N. To do this, for any set s of names and $1 \le i < j \le |s|$, we let $\pi_i : N^{|s|} \to N$ be the projection onto the i-th coordinate and we take $eq_{i,j} : N^{|s|} \to 1 + 1$ be the map

$$N^{|s|} \xrightarrow{\langle \pi_i, \pi_j \rangle} N^2 \xrightarrow{eq} 1 + 1$$

checking if the i-th and j-th coordinates of vectors in $N^{|s|}$ are equal. We then ask that the limit $(\neq s) \to N^{|s|}$ of all of the maps

$$(\neq s) \longrightarrow N^{|s|} \underset{\mathsf{false}}{\overset{eq_{i,j}}{\rightrightarrows}} 1 + 1, \qquad i \neq j$$

exists, and we take $(\neq s)$ to be the object of distinct names in s.

We have now interpreted the nu-calculus in the metalanguage and the metalanguage in $\mathcal{C}$. Composing these, we obtain an interpretation of the nu-calculus in $\mathcal{C}$. Specifically, if σ is a type of the nu-calculus we get an object $[\![\sigma]\!]$ in $\mathcal{C}$, and if $C \in \mathrm{Can}_\sigma(s, \Gamma)$ and $M \in \mathrm{Exp}_\sigma(s, \Gamma)$ we get maps

$$|C| : N^{|s|} \times [\![\Gamma]\!] \to [\![\sigma]\!]$$
$$[\![M]\!] : N^{|s|} \times [\![\Gamma]\!] \to T\,[\![\sigma]\!]$$

$$\frac{x : A \in \Gamma}{\Gamma \vdash x : A} \quad \longmapsto \quad \overline{\pi_A : \Gamma \to A}$$

$$\overline{\Gamma \vdash \mathsf{new} : TName} \quad \longmapsto \quad \overline{\Gamma \to 1 \xrightarrow{new} TN}$$

$$\overline{\Gamma \vdash \mathsf{true} : Bool} \quad \longmapsto \quad \overline{\Gamma \to 1 \xrightarrow{\mathsf{true}} 2}$$

$$\overline{\Gamma \vdash \mathsf{false} : Bool} \quad \longmapsto \quad \overline{\Gamma \to 1 \xrightarrow{\mathsf{false}} 2}$$

$$\frac{\Gamma \vdash n : Name \quad \Gamma \vdash m : Name}{\Gamma \vdash eq(n, m) : Bool} \quad \longmapsto \quad \frac{n : \Gamma \to N \quad m : \Gamma \to N}{\Gamma \xrightarrow{\langle n,m \rangle} N \times N \xrightarrow{eq} 2}$$

$$\frac{\Gamma \vdash b : Bool \quad \Gamma \vdash a : A \quad \Gamma \vdash a' : A}{\Gamma \vdash cond(b, a, a') : A} \quad \longmapsto \quad \frac{b : \Gamma \to 2 \quad a : \Gamma \to A \quad a' : \Gamma \to A}{\Gamma \xrightarrow{\langle b,a,a' \rangle} 2 \times A \times A \xrightarrow{cond_A} A}$$

$$\frac{\Gamma, x : A \vdash a : B}{\Gamma \vdash \lambda x : A.a : A \to B} \quad \longmapsto \quad \frac{b : \Gamma \times A \to B}{\mathrm{curry}(b) : \Gamma \to B^A}$$

$$\frac{\Gamma \vdash f : A \to B \quad \Gamma \vdash a : A}{\Gamma \vdash fa : B} \quad \longmapsto \quad \frac{f : \Gamma \to B^A \quad a : \Gamma \to A}{\Gamma \xrightarrow{\langle f,a \rangle} B^A \times A \xrightarrow{ev} B}$$

$$\frac{\Gamma \vdash a : A}{\Gamma \vdash [a] : TA} \quad \longmapsto \quad \frac{a : \Gamma \to A}{\Gamma \xrightarrow{a} A \xrightarrow{\eta_A} TA}$$

$$\frac{\Gamma \vdash e : TA \quad \Gamma, x : A \vdash e' : TB}{\Gamma \vdash \mathsf{let}\ x \Leftarrow e\ \mathsf{in}\ e' : TB} \quad \longmapsto \quad \frac{e : \Gamma \to TA \quad e' : \Gamma \times A \to TB}{\Gamma \xrightarrow{\langle 1,e \rangle} \Gamma \times TA \xrightarrow{(e')^*} TB}$$

Figure 3.5: Translating the metalanguage to maps in $\mathcal{C}$

in $\mathcal{C}$. In order to soundly model the distinctness of the names in s, we restrict these maps to the object of distinct names $(\neq s)$ to get the maps

$$|C|_{\Gamma,\neq s} : (\neq s) \times \Gamma \longrightarrow N^{|s|} \times \Gamma \xrightarrow{|C|} [\![\sigma]\!]$$

$$[\![M]\!]_{\Gamma,\neq s} : (\neq s) \times \Gamma \longrightarrow N^{|s|} \times \Gamma \xrightarrow{[\![M]\!]} T\,[\![\sigma]\!]\,.$$

If Γ is empty we omit it and write $|C|_{\neq s}$ and $[\![M]\!]_{\neq s}$. If s is empty then the equalizer diagram is empty and $(\neq s) \cong 1$, so we omit it and write $|C|_\Gamma, [\![M]\!]_\Gamma$.

In order for $\mathcal{C}$ to be a sound and adequate model of the nu-calculus, that is in order for $\mathcal{C}$ to satisfy properties analogous to those of Theorems 3.3 and 3.4 of the metalanguage, Stark imposes additional requirements on the distinguished objects $1+1, N$, the monad T and the arrows $\mathsf{true}, \mathsf{false}, eq$ and new [Sta96]. These requirements correspond directly to the rules of the metalanguage that should be satisfied in $\mathcal{C}$.

The MONO rule of the metalanguage at ground types corresponds to the assertion that the maps η_N, η_2 of the monad are monomorphisms. To capture the Boolean rules that $\mathsf{true}, \mathsf{false}$ are the only truth values, we add some requirements to the coproduct $1+1$. In order to soundly interpret the equality of names, we add some conditions to N and eq. Finally, in order to model the DROP, SWAP and FRESH rules for fresh name generation, we add three additional requirements that must be satisfied by the distinguished map $\mathsf{new} : 1 \to TN$.

Definition 3.13. Let $\mathcal{C}$ be a category with terminal object 1 and initial object 0, and suppose that the coproduct $1+1$ exists in $\mathcal{C}$. Let $\mathsf{true}, \mathsf{false}$ be the left and right inclusion maps $1 \to 1+1$, as specified above. We say that $1+1$ is *disjoint* if the following diagram is a pullback.

$$\begin{array}{ccc} 0 & \longrightarrow & 1 \\ \downarrow & & \downarrow{\scriptstyle \mathsf{false}} \\ 1 & \xrightarrow[\mathsf{true}]{} & 1+1 \end{array}$$

Now suppose that N is an object in $\mathcal{C}$. We say that N is *decidable* if there is a map $eq : N \times N \to 1+1$ such that following diagram is a pullback. Here, $\Delta : N \to N \times N$ is the diagonal map.

$$\begin{array}{ccc} N & \xrightarrow{\Delta} & N \times N \\ \downarrow & & \downarrow{\scriptstyle eq} \\ 1 & \xrightarrow[\mathsf{true}]{} & 1+1 \end{array}$$

Definition 3.14. Let $\mathcal{C}$ be a cartesian closed category with a strong monad T. Suppose that $\mathcal{C}$ has initial and terminal objects $0, 1$, and that the coproduct $1+1$ exists and is disjoint. Let N be a distinguished object in $\mathcal{C}$ that is decidable with equality decided by the map $eq : N \times N \to 1+1$, and suppose that the equalizers $(\neq s) \to N^{|s|} \rightrightarrows 1+1$ exist

for all finite sets s, in the sense described above. Additionally, suppose that the maps η_N, η_{1+1} are monomorphisms. Finally, let $\mathsf{new} : 1 \to TN$ be a distinguished map in $\mathcal{C}$.

We say that $\mathcal{C}$, along with the data T, N, new, is a *categorical model of the nu-calculus* provided that the following properties are satisfied by new.

1. For any map $f : X \to TY$, we have $f(x) = (\lambda n.f(x))^*(\mathsf{new})$.
2. For any map $g : X \times N \times N \to TY$, we have
$$(\lambda n.(\lambda n'.g(x,n,n'))^*(\mathsf{new}))^*(\mathsf{new}) = (\lambda n'.(\lambda n.g(x,n,n'))^*(\mathsf{new}))^*(\mathsf{new}).$$
3. For any map $h : X \times (1+1) \times N \times N \to TY$, we have
$$(\lambda n'.h(x, eq(n,n'), n, n'))^*(\mathsf{new}) = (\lambda n'.h(x, \mathsf{false}, n, n'))^*(\mathsf{new}).$$

These conditions on new correspond respectively to the DROP, SWAP and FRESH rules of the metalanguage.

By construction, categorical models of the nu-calculus are exactly those that soundly model the metalanguage (cf. [Sta96]). In particular, we have the following:

Proposition 3.15. *Let $\mathcal{C}$ be a categorical model of the nu-calculus. If*
$$[\![s, \Gamma]\!] ; (\neq s) \vdash [\![M_1]\!] = [\![M_2]\!]$$
in the metalanguage, then $[\![M_1]\!]_{\neq s} = [\![M_2]\!]_{\neq s}$ in $\mathcal{C}$.

Proof. The conditions for a category to be a model of the nu-calculus correspond to the axioms of the metalanguage. We can therefore interpret any proof in the metalanguage soundly in our category. □

3.3.3 Adequacy and Abstractness of Categorical Models

Categorical models were created both to provide semantics for the nu-calculus and to reason about observational equivalence in the nu-calculus. As discussed in Section 2.3, it is difficult to reason about observational equivalences in general. However, a model that satisfies adequacy and abstractness properties analogous to those of the metalanguage (cf. Theorems 3.4 and 3.5) provides a new environment in which we can study observational equivalence.

In general, categorical models of the nu-calculus are sound, adequate and abstract at ground types (cf. Proposition 3.15).

Theorem 3.16 (Soundness [Sta96])**.** *Let $\mathcal{C}$ be a categorical model of the nu-calculus. Let $M \in \mathrm{Exp}_\sigma(s)$. If $s \vdash M \Downarrow_\sigma (t)C$ for $C \in \mathrm{Can}_\sigma(s \sqcup t)$, then*
$$[\![M]\!]_{\neq s} = [\![\nu t.C]\!]_{\neq s}\,.$$

Definition 3.17. We say a cartesian closed category $\mathcal{C}$ is *non-degenerate* if it contains two non-isomorphic objects.

Theorem 3.18 (Adequacy [Sta96]). *Let $\mathcal{C}$ be a categorical model of the nu-calculus. Suppose that $\mathcal{C}$ is non-degenerate. For all $M_1, M_2 \in \mathrm{Exp}_\sigma(s)$,*

$$[\![M_1]\!]_{\neq s} = [\![M_2]\!]_{\neq s} \implies s \vdash M_1 \approx_\sigma M_2.$$

Theorem 3.19 (Abstractness at Ground Types [Sta96]). *Let $\mathcal{C}$ be a categorical model of the nu-calculus. Let σ be a ground type. For all $M_1, M_2 \in \mathrm{Exp}_\sigma(s)$,*

$$s \vdash M_1 \approx_\sigma M_2 \implies [\![M_1]\!]_{\neq s} = [\![M_2]\!]_{\neq s}\,.$$

The adequacy of categorical models means we can use these models to prove observational equivalences. Conversely, abstractness means we can use these models to prove that terms are *not* observationally equivalent.

In general, categorical models do not satisfy abstractness for types more complex than ground types. We can therefore ask, given a model $\mathcal{C}$, at what types abstractness holds in $\mathcal{C}$. We say that a model is *fully abstract* if it is abstract for all types σ. More modestly, we say that a model is *abstract at first-order types* if it is abstract for all first-order types σ. In Chapter 6 we will prove that probabilistic semantics provide a model of the nu-calculus that is abstract at first-order types.

4 Descriptive Set Theory

In this chapter we will define measurable spaces, measures, probability spaces and spaces of measures, and we will show that the latter form a monad on the category of measurable spaces. We will also state a result of Aumann [Aum61] showing that the category of measurable spaces is not cartesian closed. This will motivate the construction of quasi-Borel spaces in Chapter 5, as we need a cartesian closed category in order to model the nu-calculus categorically.

We will then define the standard Borel spaces, a particularly nice class of measurable spaces. We will show that standard Borel spaces admit measure preserving group structures, which we will use in Chapter 6 to prove the abstractness of our model of the nu-calculus.

Finally we will define the Borel on Borel families of sets, and we will establish some basic properties of these families using the existence of Borel inseparable sets. This will be used in Chapter 6 to prove that the privacy equation holds in our model.

4.1 Measurable Spaces and Measures

Definition 4.1. A *measurable space* is a set X equipped with a σ-algebra Σ_X of subsets of X. We say a subset $B \subseteq X$ is *measurable* if $B \in \Sigma_X$. If (X, Σ_X) and (Y, Σ_Y) are measurable spaces, we say a function $f : X \to Y$ is a *measurable function* if for all $B \subseteq Y$ measurable in Y, $f^{-1}(B)$ is measurable in X.

The collection of measurable spaces and measurable functions forms the category **Meas**.

Example 4.2. Any set X can be turned into a measurable space by taking the σ-algebra Σ_X to be the collection of all subsets of X. We call this the *discrete σ-algebra* on X. With this σ-algebra, any function $f : X \to Y$ for a measurable space Y is measurable.

Example 4.3. If (X, Σ_X) is a measurable space and $Y \subseteq X$, then Y forms a measurable space under the *subset algebra* $\Sigma_Y = \{Y \cap B \mid B \in \Sigma_X\}$. With the subset algebra, the inclusion $Y \to X$ is measurable.

Example 4.4. If X is any topological space, we can define the *Borel σ-algebra* on X to be the smallest σ-algebra containing the open subsets of X. If X, Y are topological spaces, then any continuous function $f : X \to Y$ is measurable when X, Y are equipped with their Borel algebras.

Notation 4.5. If X is a topological space, we will always assume that it is a measurable space equipped with its Borel algebra, which we denote $\mathcal{B}(X)$. We call the elements of $\mathcal{B}(X)$ *Borel sets*.

Proposition 4.6. *The category of measurable spaces has products and equalizers, and is therefore complete:*

- *If $\{X_i\}_{i\in I}$ are measurable spaces, then the product of these spaces is given by (X, Σ_X), where $X = \prod_{i\in I} X_i$ and Σ_X is the smallest σ-algebra on X making the projections measurable.*
- *If $f, g : X \to Y$ are measurable functions, then the set $Z = \{x \in X \mid f(x) = g(x)\} \subseteq X$ with the subset algebra serves as the equalizer of f, g.*

However, the following result due to Aumann shows that **Meas** is not cartesian closed.

Notation 4.7. Let $\mathbb{R}$ be a measurable space with the Borel algebra $\mathcal{B}(\mathbb{R})$ and let 2 be a two-point space with the discrete algebra. We let $2^{\mathbb{R}}$ denote the set of measurable functions $\mathbb{R} \to 2$.

Remark 4.8. We can identify $\mathcal{B}(\mathbb{R})$ and $2^{\mathbb{R}}$ by letting a Borel set correspond to its characteristic function. When considering these sets as measurable spaces (with any σ-algebra), we will denote them by $2^{\mathbb{R}}$ (cf. Theorem 4.9 and Proposition 4.38). This notation is related to the function spaces we will define in Chapter 5.

Theorem 4.9 (Aumann [Aum61])**.** *There is no σ-algebra on the space $2^{\mathbb{R}}$ of measurable functions $\mathbb{R} \to 2$ such that the evaluation map $ev : 2^{\mathbb{R}} \times \mathbb{R} \to 2$ is measurable.*

Definition 4.10. A *measure* μ on a measurable space (X, Σ_X) is a function $\mu : \Sigma_X \to [0, \infty]$ such that $\mu(\emptyset) = 0$ and μ is *σ-additive*, meaning that if $\{B_n \mid n \in \omega\} \subseteq \Sigma_X$ is a countable disjoint family of measurable sets then

$$\mu\left(\bigcup_n B_n\right) = \sum_n \mu(B_n).$$

If $\mu(X) = 1$, we call μ a *probability measure*.

Example 4.11. Let $\mathbb{R}$ be the space of real numbers with the Borel algebra. The *Lebesgue measure* is the unique measure m on $\mathbb{R}$ such that $m((a, b)) = b - a$ for all intervals (a, b) and such that m is translation invariant: if $B \in \mathcal{B}(\mathbb{R})$ and $r \in \mathbb{R}$,

$$m(B) = m(B + r).$$

If we restrict this measure to the interval $[0, 1]$ then m is a probability measure.

Example 4.12. Let (X, Σ_X) be any measurable space. For any $x \in X$, we can take δ_x to be the map taking $B \in \Sigma_X$ to 1 if $x \in B$ and 0 otherwise. This is a probability measure on X called the *Dirac measure at x*.

Definition 4.13. If X is a measurable space and μ a probability measure on X, we call the pair (X, μ) a *probability space*.

Notation 4.14. If μ is a measure on X and $f : X \to [0, \infty]$, we let $\int_X f d\mu$ denote the usual Lebesgue integral of f with respect to μ.

Definition 4.15. Let (X, Σ_X) be a measurable space. We let $\mathcal{P}(X)$ be the set of probability measures on X, and we equip $\mathcal{P}(X)$ with the smallest σ-algebra such that the maps $\mu \mapsto \mu(B)$ are measurable maps $\mathcal{P}(X) \to [0,1]$ for all $B \in \Sigma_X$. We call this the *space of probability measures on* X.

Theorem 4.16 (The Giry Monad [Gir82])**.** *Let X, Y by measurable spaces, $f : X \to \mathcal{P}(Y)$ a measurable function and $\mu \in \mathcal{P}(X)$ a probability measure on X. For $B \subseteq Y$ measurable, let*

$$f^*(\mu)(B) = \int_X f(x)(B) d\mu(x).$$

This gives a probability measure $f^*(\mu) \in \mathcal{P}(Y)$, and the data $(\mathcal{P}, \delta, (-)^*)$ defines a strong monad on **Meas**. *This is called the* Giry monad.

4.2 Standard Borel Spaces

We are particularly interested in the standard Borel spaces, which are the measurable spaces that arise from the Borel algebras on Polish spaces.

Definition 4.17. A *Polish space* is a separable, completely metrizable topological space. A *standard Borel space* is a measurable space that is measurably isomorphic to a Polish space equipped with its Borel algebra.

Example 4.18. Any countable discrete space is Polish and forms a discrete measurable space. Thus, all countable discrete measurable spaces are standard Borel.

Example 4.19. The space $\mathbb{R}$ of real numbers is Polish so it is a standard Borel space when equipped with its Borel algebra. So are the *unit interval* $[0,1]$, the *Cantor space* $\mathcal{C} = 2^\omega$ and the *Baire space* $\mathcal{N} = \omega^\omega$ (cf. Proposition 4.21).

Remark 4.20. The space $\mathbb{R}$ is not a standard Borel space when equipped with the σ-algebra of Lebesgue measurable sets. We will always consider $\mathbb{R}$ to be equipped with the Borel algebra $\mathcal{B}(\mathbb{R})$. In general, if X is a measurable space and $f : (\mathbb{R}, \mathcal{B}(\mathbb{R})) \to X$ is measurable, we will say that f is *Borel measurable* to avoid ambiguity.

The collection of standard Borel spaces satisfies many convenient properties:

Proposition 4.21 ([Kec95, 12.B])**.** *The countable product of standard Borel spaces is standard Borel.*

Theorem 4.22 ([Kec95, 13.4])**.** *If X is standard Borel and $B \subseteq X$ is Borel, then B is standard Borel with its subset algebra.*

The following theorem is interesting in its own right, although we will not need it to prove our results.

Theorem 4.23 ([Kec95, 17.23])**.** *If X is a standard Borel space, so is $\mathcal{P}(X)$.*

Finally, we have the following structure theorem for standard Borel spaces.

Theorem 4.24 (Isomorphism Theorem for Standard Borel Spaces [Kec95, 15.6])**.** *If X, Y are standard Borel spaces with the same cardinality, then X, Y are isomorphic.*

In fact, every standard Borel space is either countable and discrete or isomorphic to $\mathbb{R}$.

4.3 Groups with Invariant Probability Measures

Let (X, μ) be a probability space. We would like to find a measure preserving group structure on X compatible with its measurable structure.

Example 4.25. The Lebesgue measure m is a translation invariant measure on $\mathbb{R}$. Restricting m to the unit interval $[0, 1]$ we get a probability measure. $[0, 1]$ forms a group under addition mod 1, and because m is translation invariant this is a measure preserving group structure. The group structure is compatible with the Borel structure of $[0, 1]$ because addition and subtraction are measurable operations.

More generally, the isomorphism theorem for standard Borel spaces (Theorem 4.24) can be extended to account for measures, so that if X is a standard Borel space and μ a continuous measure on X we can construct a measure preserving group structure on (X, μ) by transferring the structure of $[0, 1]$ onto X.

Definition 4.26. Let X be a measurable space such that for all $x \in X$, the singleton $\{x\}$ is measurable. A measure μ on X is *continuous* if $\mu(\{x\}) = 0$ for all $x \in X$.

Definition 4.27. Let $f : X \to Y$ be measurable and let μ be a measure on X. We define the measure $f_*\mu$ on Y by letting

$$f_*\mu(B) = \mu(f^{-1}(B))$$

for all $B \subseteq Y$ measurable. We call this the *pushforward measure* of μ along f.

Theorem 4.28 (The Isomorphism Theorem for Measures [Kec95, 17.41])**.** *Let (X, μ) be a standard Borel probability space and suppose μ is continuous. Let m be the Lebesgue measure on $[0, 1]$. There is a Borel measurable isomorphism $f : [0, 1] \to X$ such that $f_*m = \mu$.*

Definition 4.29. Let G be a measurable space and a group. We say G is a *measurable group*, or that the group structure is *compatible with the measurable structure of* G, if the map $(g, h) \mapsto g \cdot h^{-1}$ is a measurable map $G \times G \to G$.

Remark 4.30. Note that if G is a measurable group, $g \in G$ and $B \subseteq G$ is measurable, then $g \cdot B$ is measurable as well.

Definition 4.31. Let (G, μ) be a probability space. We say a group structure on G is *probability measure preserving* if it is compatible with the measurable structure of G and for all $B \subseteq G$ measurable and $g \in G$ we have $\mu(g \cdot B) = \mu(B)$.

Proposition 4.32. *Let (X, μ) be a standard Borel probability space. If μ is continuous, there is an abelian probability measure preserving group structure on X.*

Proof. By Theorem 4.28 there is a Borel measurable isomorphism $f : [0, 1] \to X$ such that $\mu = f_*m$. For $x, y \in X$, we define

$$x +_X y = f(f^{-1}(x) + f^{-1}(y)),$$

where $+$ is the usual addition mod 1 in $[0,1]$. Because f is a Borel isomorphism, this turns X into a measurable group. It is abelian because $[0,1]$ is abelian, and it is probability measure preserving because $+$ is m-preserving:

$$\mu(x +_X B) = m(f^{-1}(x) + f^{-1}(B)) = m(f^{-1}(B)) = \mu(B).$$

□

Measure preserving groups also behave nicely with respect to the lift of the Giry monad.

Proposition 4.33. *Let (G,μ) be a probability space with a probability measure preserving group structure. Let $f : G \to \mathcal{P}(Y)$ be a measurable map, and for $g \in G$, write $f_g(x) = f(g \cdot x)$. Then $f^*(\mu) = f_g^*(\mu)$ for all $g \in G$.*

Proof. Let $B \subseteq Y$ be measurable. Then

$$f_g^*(\mu)(B) = \int_X f(g \cdot x)(B) d\mu(x) = \int_X f(x)(B) d\mu(x) = f^*(\mu)(B).$$

□

4.4 Borel on Borel Sets

Consider the set $2^{\mathbb{R}}$ of measurable functions $\mathbb{R} \to 2$. We have seen in Theorem 4.9 that there is no σ-algebra on $2^{\mathbb{R}}$ such that the evaluation map is measurable. This space, however, will be important in our study of quasi-Borel spaces. Of particular interest will be the σ-algebra of Borel on Borel sets.

Notation 4.34. If X is a set and $B \subseteq X \times Y$, then for any $x \in X$, the x-section of B is the set $B_x = \{y \in Y \mid (x,y) \in B\}$.

Definition 4.35. Let X be a standard Borel space. A collection $\mathcal{F} \subseteq \mathcal{B}(X)$ of Borel subsets of X is *Borel on Borel* if for any standard Borel space Y and all Borel sets $B \subseteq Y \times X$, $\{y \in Y \mid B_y \in \mathcal{F}\}$ is Borel.

Remark 4.36. Because every standard Borel space is either discrete or isomorphic to $\mathbb{R}$, we only need to consider a single uncountable standard Borel space Y.

Notation 4.37. We let $\Sigma_{2^{\mathbb{R}}}$ be the collection of Borel on Borel subsets of $\mathcal{B}(\mathbb{R}) = 2^{\mathbb{R}}$.

Proposition 4.38. *The collection $\Sigma_{2^{\mathbb{R}}}$ is a σ-algebra on $2^{\mathbb{R}}$.*

Proof. This follows from the fact that $\mathcal{B}(\mathbb{R})$ is a σ-algebra. For example, if $\{\mathcal{F}_n\}_{n\in\omega}$ is a countable collection of Borel on Borel subsets of $\mathbb{R}$, then $\mathcal{F} = \bigcup_n \mathcal{F}_n$ is Borel on Borel, because for any $B \subseteq Y \times \mathbb{R}$ Borel, we have

$$\{y \in Y \mid B_y \in \mathcal{F}\} = \bigcup_n \{y \in Y \mid B_y \in \mathcal{F}_n\},$$

which is Borel as the countable union of Borel sets. □

We will now consider the probability measures on the measurable space $(2^{\mathbb{R}}, \Sigma_{2^{\mathbb{R}}})$.

Lemma 4.39. *Let $B \subseteq \mathbb{R} \times \mathbb{R}$ be Borel. Then the map $x \mapsto B_x$ is a Borel measurable map $\mathbb{R} \to 2^{\mathbb{R}}$.*

Proof. The preimage of a Borel on Borel set $\mathcal{F}$ under this map is

$$\{x \in \mathbb{R} \mid B_x \in \mathcal{F}\},$$

which is Borel because $\mathcal{F}$ is Borel on Borel. □

Let μ be any continuous probability measure on $\mathbb{R}$, and let $f : \mathbb{R} \to 2^{\mathbb{R}}$ be the Borel measurable map $f(x) = \{x\}$ (we see f is measurable by applying Lemma 4.39 to the diagonal). We are interested in the probability measures $\delta_\emptyset$ and $f_*\mu$ on $2^{\mathbb{R}}$. Here, $\delta_\emptyset$ is the Dirac measure at the empty set, and $f_*\mu$ measures "how many" singletons are in a Borel on Borel set $\mathcal{F}$:

$$f_*\mu(\mathcal{F}) = \mu\{x \mid \{x\} \in \mathcal{F}\}.$$

Surprisingly, it turns out that these describe the same measure. That is, in some sense, the Borel on Borel sets cannot distinguish the empty set from a random singleton.

Theorem 4.40. *Let $\mathcal{F} \subseteq 2^{\mathbb{R}}$ be Borel on Borel. Then $\emptyset \notin \mathcal{F}$ if and only if $\{x\} \notin \mathcal{F}$ for all but countably many $x \in \mathbb{R}$.*

To prove this, we use the existence of certain Borel inseparable sets.

Definition 4.41. Let X be a standard Borel space and let $A, A' \subseteq X$. We say that A, A' are *Borel separable* if there is a Borel set $B \in \mathcal{B}(X)$ such that $A \subseteq B$ and $A' \cap B = \emptyset$. If A, A' are not Borel separable, we say they are *Borel inseparable*.

Notation 4.42. Let X, Y be sets and $F \subseteq X \times Y$. For any $n \in \omega$, we let $F^n = \{x \in X \mid |F_x| = n\}$.

Theorem 4.43 ([Kec95, 35.2])**.** *Let $\mathcal{N} = \omega^\omega$ be the Baire space. There is a closed set $F \subseteq \mathcal{N} \times \mathcal{N}$ such that F^0 and F^1 are Borel inseparable.*

Corollary 4.44. *If X is a standard Borel space and $B \subseteq X$ is an uncountable Borel set in X, then there is a Borel set $F \subseteq X \times B$ such that F^0, F^1 are Borel inseparable.*

Proof. By Theorems 4.22 and 4.24 we can assume that $X = B = \mathcal{N}$, so this follows from Theorem 4.43. □

We can now prove Theorem 4.40.

Proof of Theorem 4.40. Suppose $\emptyset \notin \mathcal{F}$ and let $B = \{x \mid \{x\} \in \mathcal{F}\}$. Because $\mathcal{F}$ is Borel on Borel, B is Borel. Now for any $F \subseteq \mathbb{R} \times B$, $F^0 \subseteq \{x \mid F_x \notin \mathcal{F}\}$ and $F^1 \subseteq \{x \mid F_x \in \mathcal{F}\}$ so that F^0, F^1 are Borel separable. By Corollary 4.44, B must be countable.

The converse follows by replacing $\mathcal{F}$ with $\mathcal{F}^c$, which is also Borel on Borel. □

As a consequence, for any Borel on Borel set $\mathcal{F}$ we have that $\delta_\emptyset(\mathcal{F}) = 0$ if and only if $f_*\mu(\mathcal{F}) = 0$. By considering $\mathcal{F}^c$, which is also Borel on Borel, we have that $\delta_\emptyset(\mathcal{F}) = 1$ if and only if $f_*\mu(\mathcal{F}) = 1$. It follows that these measures are indeed equal.

5 Quasi-Borel Spaces

Probability theory provides a tempting framework in which to interpret the nu-calculus. For example, we may consider taking $\mathbb{R}$ to be our set of names and model fresh name generation as sampling from a continuous distribution, such as a Gaussian. Intuitively, this should adequately model fresh name generation as it is improbable to sample the same number twice.

Unfortunately, we are unable to interpret the nu-calculus in the category of measurable spaces, as we have seen that this category is not cartesian closed. In fact, we are not even able to restrict ourselves to spaces of interest, as we have seen in Theorem 4.9 that once we include the standard Borel spaces there is no exponential object $2^{\mathbb{R}}$. Thus, if one wants to interpret a programming language in a category suitable for probabilistic reasoning, one needs to find an alternative to the category of measurable spaces.

Recently, the category of quasi-Borel spaces has been constructed with the intention of providing a category suitable for both interpreting a typed lambda-calculus and for probabilistic reasoning [Heu+17]. Although the intention is to use quasi-Borel spaces to model probabilistic programming languages, we will use it to construct a model of the nu-calculus, letting fresh name generation correspond to sampling from a continuous distribution on $\mathbb{R}$.

In this chapter, we will define the category **QBS** of quasi-Borel spaces, a cartesian closed category that includes the standard Borel spaces. We will see that it supports probabilistic reasoning, as it has both a notion of integration and a probability monad.

We will then prove that quasi-Borel spaces form a categorical model of the nu-calculus, letting $\mathbb{R}$ be the space of names and fresh name generation correspond to sampling from a continuous distribution.

5.1 The Category QBS

The central object in the study of measurable spaces is the σ-algebra. We then define measurable functions in terms of σ-algebras on sets. Quasi-Borel spaces, on the other hand, take as primitive the measurable functions [Heu+17]. We therefore fix a sample space $\mathbb{R}$ and define a quasi-Borel structure on a set X to be a collection of functions $\mathbb{R} \to X$, which we interpret to be the measurable functions.

Definition 5.1. A *quasi-Borel space* X is a set X equipped with a collection M_X of functions $\mathbb{R} \to X$ satisfying the following:

- the constant functions are in M_X;
- if $\alpha \in M_X$ and $f : \mathbb{R} \to \mathbb{R}$ is Borel measurable then $\alpha \circ f \in M_X$;

- if $\{\alpha_n \mid n \in \omega\} \subseteq M_X$ and $\{B_n \mid n \in \omega\}$ is a countable disjoint cover of $\mathbb{R}$ by Borel sets, then $\bigcup_n \alpha_n \restriction_{B_n} \in M_X$.

Notation 5.2. If $x \in X$, let $\lambda r.x$ denote the constant function $\mathbb{R} \to X$ taking value x.

Definition 5.3. Given two quasi-Borel spaces $(X, M_X), (Y, M_Y)$, a function $f : X \to Y$ is a quasi-Borel map if for all $\alpha \in M_X$, we have $f \circ \alpha \in M_Y$.

These definitions turn the collection of quasi-Borel spaces and maps into a category, which we denote **QBS**. If X, Y are quasi-Borel spaces, we let $\mathbf{QBS}(X, Y)$ denote the set of all quasi-Borel maps $X \to Y$.

Example 5.4. Let (X, Σ_X) be a measurable space. We can turn this into a quasi-Borel space (X, M_{Σ_X}) by letting M_{Σ_X} be the set of Borel measurable functions $\mathbb{R} \to X$.

Similarly, given a quasi-Borel space (X, M_X), we can form the measurable space (X, Σ_{M_X}) by taking Σ_{M_X} to be the largest σ-algebra on X such that the functions in M_X are Borel measurable.

We shall take $\mathbb{R}$ to be the quasi-Borel space equipped with the quasi-Borel structure $M_{\mathbb{R}} = M_{\mathcal{B}(\mathbb{R})}$ consisting of the Borel measurable functions $\mathbb{R} \to \mathbb{R}$. This implies that if (X, M_X) is a quasi-Borel space, then $M_X = \mathbf{QBS}(\mathbb{R}, X)$ is exactly the collection of quasi-Borel maps $\mathbb{R} \to X$.

The construction of Example 5.4 provides a nice correspondence between the category **Meas** of measurable spaces and **QBS**.

Theorem 5.5 ([Heu+17, Proposition 15])**.** *Let (Y, Σ_Y) be a measurable space.*

- *If (X, M_X) is a quasi-Borel space, then $f : X \to Y$ is a measurable function $(X, \Sigma_{M_X}) \to (Y, \Sigma_Y)$ if and only if it is a quasi-Borel map $(X, M_X) \to (Y, M_{\Sigma_Y})$.*
- *If (X, Σ_X) is a standard Borel space, then $f : X \to Y$ is a quasi-Borel map $(X, M_{\Sigma_X}) \to (Y, M_{\Sigma_Y})$ if and only if it is a measurable map $(X, \Sigma_X) \to (Y, \Sigma_Y)$.*

Corollary 5.6. *The functor $(X, \Sigma_X) \mapsto (X, M_{\Sigma_X})$ is right adjoint to the functor $(X, M_X) \mapsto (X, \Sigma_{M_X})$. In particular, this functor preserves limits.*

Proof. The fact that this is an adjunction is exactly the first part of Theorem 5.5. This functor therefore preserves limits because right adjoints preserve limits. □

Corollary 5.7. *If (X, Σ_X) is a standard Borel space, then*

$$\Sigma_{M_{\Sigma_X}} = \Sigma_X.$$

In particular, $\Sigma_{M_{\mathbb{R}}} = \mathcal{B}(\mathbb{R})$ is the usual Borel algebra on $\mathbb{R}$.

We will therefore refer to standard Borel spaces such as $2, \mathbb{R}$ and $[0, \infty]$ unambiguously as both measurable spaces and quasi-Borel spaces.

5.2 Measures and Integration

As $\mathbb{R}$ was chosen to be the principal sample space of **QBS**, it is natural to choose $\mathbb{R}$ to be the source of randomness as well. Therefore, we take measures on $\mathbb{R}$ as fundamental, and define quasi-Borel measures in general to be the pushforwards of measures on $\mathbb{R}$ [Heu+17].

Definition 5.8. A *probability measure* on a quasi-Borel space (X, M_X) is a pair (α, μ), where μ is a probability measure on $\mathbb{R}$ and $\alpha \in M_X = \mathbf{QBS}(\mathbb{R}, X)$.

The pair (α, μ) of a probability measure on a quasi-Borel space X can be seen as the pushforward measure $\alpha_*\mu$. Indeed, by definition $\alpha : \mathbb{R} \to (X, \Sigma_{M_X})$ is Borel measurable, and so $\alpha_*\mu$ is a probability measure on (X, Σ_{M_X}).

More generally, if $f : X \to Y$ is a quasi-Borel map and (α, μ) a measure on X, then $(f \circ \alpha, \mu)$ is a measure on Y, and so we can push forward measures between arbitrary quasi-Borel spaces.

We can then reduce integration over general quasi-Borel spaces to integration over $\mathbb{R}$:

Definition 5.9. If $f : X \to [0, \infty]$ is a quasi-Borel map and (α, μ) is a measure on X, then we can define the *integral* of f with respect to (α, μ) to be

$$\int f d(\alpha, \mu) = \int_{\mathbb{R}} (f \circ \alpha) d\mu,$$

where the integral on the right-hand-side is the usual Lebesgue integral.

By Theorem 5.5 the map $f \circ \alpha : \mathbb{R} \to [0, \infty]$ is Borel measurable, so this definition makes sense.

5.3 Function Spaces

We will now provide the construction of product spaces and function spaces in **QBS**, showing that it is cartesian closed. Note that the one-point space 1 serves trivially as terminal object.

Proposition 5.10 (Products [Heu+17, proposition 16])**.** *Let $(X_i, M_{X_i})_{i \in I}$ be a collection of quasi-Borel spaces. Then (X, M_X) is a quasi-Borel space, where $X = \prod_I X_i$ is the set-theoretic product and*

$$M_X = \{f : \mathbb{R} \to X \mid \forall i \in I \, (\pi_i \circ f \in M_{X_i})\}.$$

The space X, together with the set-theoretic projections $\pi_i : X \to X_i$, form the categorical product of (X_i, M_{X_i}) in **QBS**.

Proposition 5.11 (Function Spaces [Heu+17, proposition 18])**.** *Let $(X, M_X), (Y, M_Y)$ be quasi-Borel spaces. Then (Y^X, M_{Y^X}) is a quasi-Borel space, where $Y^X = \mathbf{QBS}(X, Y)$ and*

$$M_{Y^X} = \{\alpha : \mathbb{R} \to Y^X \mid \mathrm{uncurry}(\alpha) \in \mathbf{QBS}(\mathbb{R} \times X, Y)\}.$$

Here, $\mathrm{uncurry}(\alpha)(r, x) = \alpha(r)(x)$*. This makes the evaluation map $Y^X \times X \to Y$ a quasi-Borel morphism, and exhibits* **QBS** *as cartesian closed.*

If X, Y are standard Borel spaces, then $\mathbf{QBS}(X, Y) = \mathbf{Meas}(X, Y)$ by Theorem 5.5. Therefore, the set Y^X is the set of measurable functions $X \to Y$. As we have seen in Theorem 4.9 this set does not admit a nice measurable structure making evaluation measurable. It does, however, admit a nice quasi-Borel structure.

Example 5.12. Consider the quasi-Borel space $2^{\mathbb{R}}$. We identify the set $2^{\mathbb{R}}$ with the collection $\mathcal{B}(\mathbb{R})$ of Borel subsets of $\mathbb{R}$. A function $f : \mathbb{R} \to 2^{\mathbb{R}}$ is quasi-Borel if and only if $\mathrm{uncurry}(f) : \mathbb{R} \times \mathbb{R} \to 2$ is Borel measurable. This means that there is a Borel set $B \subseteq \mathbb{R} \times \mathbb{R}$ such that $f(r) = B_r$ for all $r \in \mathbb{R}$.

Given such a B, we see that for $\mathcal{F} \subseteq 2^{\mathbb{R}}$ we have

$$f^{-1}(\mathcal{F}) = \{r \in \mathbb{R} \mid B_r \in \mathcal{F}\}.$$

Therefore, a set $\mathcal{F} \subseteq 2^{\mathbb{R}}$ is in the induced σ-algebra $\Sigma_{2^{\mathbb{R}}}$ if and only if for all $B \subseteq \mathbb{R} \times \mathbb{R}$ Borel, $\{r \in \mathbb{R} \mid B_r \in \mathcal{F}\}$ is Borel, and so $\Sigma_{2^{\mathbb{R}}}$ consists exactly of the Borel on Borel families of subsets of $\mathbb{R}$.

Remark 5.13. The Borel on Borel σ-algebra on $2^{\mathbb{R}}$ induced from the quasi-Borel structure on $2^{\mathbb{R}}$ does not contradict Theorem 4.9, as it does not make the evaluation map $2^{\mathbb{R}} \times \mathbb{R} \to 2$ measurable in **Meas**. This is because in general, the functor $\mathbf{QBS} \to \mathbf{Meas}$ does not preserve limits, and so the σ-algebra $\Sigma_{2^{\mathbb{R}} \times \mathbb{R}}$ on $2^{\mathbb{R}} \times \mathbb{R}$ induced by the quasi-Borel structure may be strictly larger than the product σ-algebra $\Sigma_{2^{\mathbb{R}}} \times \mathcal{B}(\mathbb{R})$ on $2^{\mathbb{R}} \times \mathbb{R}$.

5.4 Probability Spaces

Given a quasi-Borel space X we can construct the space of probability measures on X. This will form a monad on **QBS** just as it does in **Meas**.

As with the basic construction of quasi-Borel spaces and integration, the quasi-Borel structure on the space of measures will be derived from the structure of the Giry monad on the space $\mathcal{P}(\mathbb{R})$ of probability measures on $\mathbb{R}$.

Remark 5.14. At the moment, we have defined probability measures on a quasi-Borel space intensionally. For example, let $f : \mathbb{R} \to 2^{\mathbb{R}}$ be the map $f(x) = \{x\}$ and μ be a continuous probability measure on $\mathbb{R}$. We can then form the distinct quasi-Borel measures $(\lambda r.\emptyset, \mu)$ and (f, μ) on $2^{\mathbb{R}}$.

However, we have seen in Section 4.4 that these describe the same pushforward measures on $2^{\mathbb{R}}$. Therefore, despite having distinct descriptions, these measures behave identically. We will choose to identify measures that behave the same, and will define the space of probability measures extensionally.

Definition 5.15. Let X be a quasi-Borel space and $(\alpha, \mu), (\beta, \nu)$ be measures on X. We identify these two measures and write $(\alpha, \mu) \sim (\beta, \nu)$ if for all $f \in \mathbf{QBS}(X, [0, \infty])$,

$$\int f d(\alpha, \mu) = \int f d(\beta, \nu).$$

If (α, μ) is a measure on X, we denote its equivalence class by $[\alpha, \mu]_\sim$.

Remark 5.16. By Theorem 5.5, a function $f : X \to [0, \infty]$ is a quasi-Borel map $(X, M_X) \to [0, \infty]$ if and only if it is a measurable map $(X, \Sigma_{M_X}) \to [0, \infty]$. Therefore, if $(\alpha, \mu), (\beta, \nu)$ are measures on X, we have that $(\alpha, \mu) \sim (\beta, \nu)$ if and only if $\alpha_*\mu = \beta_*\nu$ as measures on (X, Σ_{M_X}).

Definition 5.17. Let (X, M_X) be a quasi-Borel space. We define the set of probability measures on X to be the quotient

$$P(X) = \{(\alpha, \mu) \mid \alpha \in M_X, \mu \in \mathcal{P}(\mathbb{R})\}/\sim .$$

We let $P(X)$ be the quasi-Borel space whose quasi-Borel structure is given by

$$M_{P(X)} = \{\beta : \mathbb{R} \to P(X) \mid \exists \alpha \in M_X, g : \mathbb{R} \to \mathcal{P}(\mathbb{R}) \text{ measurable s.t. } \beta(r) = [\alpha, g(r)]_\sim\}.$$

Note that in this definition, the map $g : \mathbb{R} \to \mathcal{P}(\mathbb{R})$ is a Borel measurable map $\mathbb{R} \to \mathcal{P}(\mathbb{R})$, which we have already defined independently of quasi-Borel spaces, and so this definition is not circular.

We will now define a probability monad for **QBS** analogous to the Giry monad.

Definition 5.18. Let X be a quasi-Borel space. For $x \in X$, we let $\delta_x = [\lambda r.x, \mu]_\sim \in P(X)$ be the *Dirac measure* at x on X, where μ is any probability measure on $\mathbb{R}$. This is a measure on X because the constant functions are always quasi-Borel maps.

Now suppose that $f : X \to P(Y)$ for quasi-Borel spaces X, Y. We would like to define a lift operator $f^* : P(X) \to P(Y)$. To do this, consider $[\alpha, \mu]_\sim \in P(X)$. The map $f \circ \alpha : \mathbb{R} \to P(Y)$ is a quasi-Borel map, and so $f \circ \alpha \in \mathbf{QBS}(\mathbb{R}, P(Y)) = M_{P(Y)}$. We can therefore find $\beta \in M_Y$ and $g : \mathbb{R} \to \mathcal{P}(\mathbb{R})$ Borel measurable such that $f \circ \alpha(r) = [\beta, g(r)]_\sim$. We then define

$$f^*([\alpha, \mu]_\sim) = [\beta, g^*(\mu)]_\sim,$$

where $g^*(\mu)$ is the lift of the Giry monad.

Theorem 5.19 ([Heu+17, Theorem 21, Proposition 22])**.** *The structure $(P, \delta, (-)^*)$ forms a strong monad on* **QBS***. Additionally, the monad P corresponds to the functor on* **QBS** *taking $f : X \to Y$ to $P(f) : P(X) \to P(Y)$, defined by*

$$P(f)([\alpha, \mu]_\sim) = [f \circ \alpha, \mu]_\sim.$$

The monad P is commutative*, meaning that if $p \in P(X), q \in P(Y)$ and $f : X \times Y \to P(Z)$, then*

$$(\lambda x.(\lambda y.f(x, y))^*(q))^*(p) = (\lambda y.(\lambda x.f(x, y))^*(p))^*(q).$$

If (X, Σ_X) is a standard Borel space and we construct the space $(P(X), M_{P(X)})$ by considering (X, M_{Σ_X}) to be a quasi-Borel space, then the measurable space $(P(X), \Sigma_{M_{P(X)}})$ we obtain from this is isomorphic to the standard Borel space $\mathcal{P}(X)$. In particular, we can identify $P(\mathbb{R})$ and $\mathcal{P}(\mathbb{R})$.

The commutativity of this monad corresponds to the commutativity of integration, i.e. Fubini's Theorem.

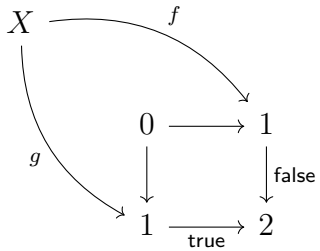
5.5 Interpreting the Nu-Calculus in QBS

Let ν be a continuous measure on $\mathbb{R}$. We will now show that by letting $\mathbb{R}$ be the set of names and new consist of sampling from ν, **QBS** is a categorical model of the nu-calculus,

Lemma 5.20. *The initial object* 0 *in* **QBS** *is the empty set. The coproduct* $1 + 1$ *in* **QBS** *is the standard Borel space* 2*, and this coproduct is disjoint.*

Proof. That the empty set is the initial object and the terminal object is the one-point set $1 = \{*\}$ is clear. Write $2 = \{T, F\}$ and let $\mathsf{true}, \mathsf{false} : 1 \to 2$ be the two inclusions. To see that this turns 2 into the coproduct $1 + 1$, we must show that if $x, y \in X$ for any quasi-Borel space X, then the map $T \mapsto x, F \mapsto y$ is a quasi-Borel map $2 \to X$, which is clear as we have chosen 2 to have the standard Borel structure.

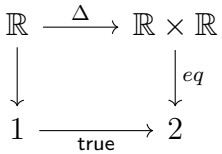
To see that this coproduct is disjoint, we must show that if we have the following commuting diagram



then f, g factor through a unique map $X \to 0$. But this is clear because if this diagram commutes then for all $x \in X$, $T = (\mathsf{true} \circ g)(x) = (\mathsf{false} \circ f)(x) = F$, so X must itself be empty. □

Lemma 5.21. *The space* $\mathbb{R}$ *is decidable.*

Proof. We first check this in **Meas**. The equality map $eq : \mathbb{R} \times \mathbb{R} \to 2$ is clearly Borel measurable. It remains to check that the square



is a pullback. To see this, we must show that if $g : X \to \mathbb{R} \times \mathbb{R}$ is a Borel measurable map and $g(X) \subseteq \Delta = \{(r, r) \mid r \in \mathbb{R}\}$, then there is a unique $h : X \to \mathbb{R}$ such that $g(x) = (h(x), h(x))$. But this is clear, as we are forced to take $h = \pi \circ g$.

Now products and pullbacks are limits, and the spaces $1, 2, \mathbb{R}, \mathbb{R} \times \mathbb{R}$ are all standard Borel spaces. Therefore by Theorem 5.5 and Corollary 5.6 this is a pullback in **QBS**. □

Lemma 5.22. *The equalizer* $\mathbb{R}^{\neq s} \to \mathbb{R}^{|s|}$ *of the maps*

$$\mathbb{R}^{|s|} \underset{\mathsf{false}}{\overset{eq \circ \langle \pi_i, \pi_j \rangle}{\rightrightarrows}} 2$$

for $1 \leq i < j \leq |s|$ *exists for all finite sets* s*.*

Proof. Again we first check this in **Meas**. In this case, this is clear as we can take $\mathbb{R}^{\neq s}$ to be the subset of $\mathbb{R}^{|s|}$ consisting of the $|s|$-tuples with distinct coordinates along with the inclusion map into $\mathbb{R}^{|s|}$. Now equalizers are limits and by Theorem 4.22 $\mathbb{R}^{\neq s}$ is standard Borel, so by Theorem 5.5 and Corollary 5.6 this is an equalizer in **QBS**. □

We have now seen that **QBS** is a cartesian closed category with a strong monad. The coproduct 2 is disjoint, $\mathbb{R}$ is decidable, and the appropriate equalizers exist. In order to show that **QBS** is indeed a categorical model of the nu-calculus, it remains to show that sampling from ν satisfies the DROP, SWAP and FRESH rules as specified in Definition 3.14.

That these rules hold under our probabilistic interpretation is not surprising. The DROP rule says that $\int_{\mathbb{R}} a d\nu = a$ for any constant a, which holds because we have chosen ν to be a probability measure. The SWAP rule is an application of the Fubini theorem, saying that we can swap the order of integration when integrating with respect to ν twice. The FRESH rule says that we can ignore singletons when integrating: for any $a \in \mathbb{R}$,

$$\int_{\mathbb{R}} f(r) d\nu(r) = \int_{\mathbb{R}\setminus\{a\}} f(r) d\nu(r).$$

This condition holds because we chose ν to be a continuous measure, so singletons have measure zero. This is why we chose ν to be a continuous probability measure on $\mathbb{R}$.

Lemma 5.23. *The measure ν satisfies the conditions on* new *of Definition 3.14:*

1. *For any map $f : X \to P(Y)$, we have $f(x) = (\lambda r.f(x))^*(\nu)$.*
2. *For any map $g : X \times \mathbb{R} \times \mathbb{R} \to P(Y)$, we have*

$$(\lambda r.(\lambda r'.g(x,r,r'))^*(\nu))^*(\nu) = (\lambda r'.(\lambda r.g(x,r,r'))^*(\nu))^*(\nu).$$

3. *For any map $h : X \times 2 \times \mathbb{R} \times \mathbb{R} \to P(Y)$, we have*

$$(\lambda r'.h(x, eq(r,r'), r, r'))^*(\nu) = (\lambda r'.h(x, \mathsf{false}, r, r'))^*(\nu).$$

Proof. 1. Let $f : X \to P(Y)$. For any $x \in X$, we can write $f(x) = [\alpha, \mu]_\sim \in P(Y)$. Now we can compute from the Giry monad that

$$(\lambda r.\mu)^*(\nu) = \int_{\mathbb{R}} \mu d\nu(r) = \mu$$

because μ is a constant. Therefore,

$$(\lambda r.[\alpha,\mu]_\sim)^*(\nu) = [\alpha, (\lambda r.\mu)^*(\nu)]_\sim = [\alpha,\mu]_\sim,$$

and so $(\lambda r.f(x))^*(\nu) = f(x)$.

2. This is exactly the commutativity of the monad P applied to ν (cf. Theorem 5.19).

3. Let $r \in \mathbb{R}$, let $f : \mathbb{R} \to P(\mathbb{R} \times 2)$ be the map $f(y) = \delta_{(y,r=y)}$ and let $g : \mathbb{R} \to P(\mathbb{R} \times 2)$ be the map $g(y) = \delta_{(y,\mathsf{false})}$. We claim that $f^*(\nu) = g^*(\nu)$.

   To see this, note that $\mathbb{R}, \mathbb{R} \times 2$ are standard Borel spaces, so by Corollary 5.6 and Theorem 5.19 we can consider f, g to be Borel measurable maps $\mathbb{R} \to \mathcal{P}(\mathbb{R} \times 2)$ and we must show that $f^*(\nu) = g^*(\nu)$ in terms of the Giry monad.

   Because ν is a continuous measure, $f = g$ almost everywhere, so $f^*(\nu) = g^*(\nu)$. Specifically, if $B \subseteq \mathbb{R} \times 2$ then

$$f^*(\nu)(B) = \int_{\mathbb{R}\setminus\{r\}} \delta_{(y,r=y)}(B) d\nu(y) = \int_{\mathbb{R}\setminus\{r\}} \delta_{(y,\mathsf{false})}(B) d\nu(y) = g^*(\nu)(B).$$

   Now let $h : X \times 2 \times \mathbb{R} \times \mathbb{R} \to P(Y)$ be a quasi-Borel map, and fix $x \in X, r \in \mathbb{R}$. Let $h_1(y) = h(x, r = y, r, y)$ and let $h_2(y) = h(x, \mathsf{false}, r, y)$. Then

$$h_1^*(\nu) = \int_{\mathbb{R}\times 2} h(x, b, r, y) df^*(\nu)(y, b) = \int_{\mathbb{R}\times 2} h(x, b, r, y) dg^*(\nu)(y, b) = h_2^*(\nu),$$

   as desired.

□

Theorem 5.24. QBS *with the probability monad and any continuous probability measure on* $\mathbb{R}$ *interpreting* new *is a categorical model of the nu-calculus.*

Proof. We have shown that **QBS** is a cartesian closed category with a strong monad. The maps $\delta_{\mathbb{R}} : \mathbb{R} \to P(\mathbb{R})$ and $\delta_2 : 2 \to P(2)$ are clearly injective (as they are injective in **Meas**), validating the MONO rule at ground types. Lemmas 5.20 to 5.23 show that **QBS** satisfies all of the remaining requirements to be a categorical model of the nu-calculus, letting $\mathbb{R}$ be the set of names and $\mathsf{new} = \nu$ be a continuous probability measure on $\mathbb{R}$. □

As **QBS** with the probability monad is a categorical model of the nu-calculus, it is sound, adequate and abstract at ground types (cf. Theorems 3.16, 3.18 and 3.19). Additionally, the metalanguage of the nu-calculus translates directly to terms in **QBS**. We include an explicit translation in Fig. 5.1 for convenience (cf. Fig. 3.5). We note in particular that the trivial computation $[a]$ corresponds to the Dirac measure δ_a at a, and that let expressions translate to averaging measures via the lift of the monad.

Remark 5.25. In general, there is nothing special about our choice of $\mathbb{R}$ as the fundamental sample space of **QBS**. By the isomorphism theorem for standard Borel spaces, we could instead have chosen any uncountable standard Borel space [Heu+17, Propositions 9, 13, 23].

Similarly, the proofs in this section that **QBS** is a categorical model of the nu-calculus do not depend on our choice of uncountable standard Borel space X and continuous probability measure ν on X.

$$\frac{x : A \in \Gamma}{\Gamma \vdash x : A} \quad \longmapsto \quad \overline{\pi_A : \Gamma \to A}$$

$$\overline{\Gamma \vdash \mathsf{new} : TName} \quad \longmapsto \quad \overline{\Gamma \to 1 \xrightarrow{\nu} P(\mathbb{R})}$$

$$\overline{\Gamma \vdash \mathsf{true} : Bool} \quad \longmapsto \quad \overline{\Gamma \to 1 \xrightarrow{\mathsf{true}} 2}$$

$$\overline{\Gamma \vdash \mathsf{false} : Bool} \quad \longmapsto \quad \overline{\Gamma \to 1 \xrightarrow{\mathsf{false}} 2}$$

$$\frac{\Gamma \vdash n : Name \quad \Gamma \vdash m : Name}{\Gamma \vdash eq(n,m) : Bool} \quad \longmapsto \quad \frac{n : \Gamma \to \mathbb{R} \quad m : \Gamma \to \mathbb{R}}{\Gamma \xrightarrow{\langle n,m \rangle} \mathbb{R} \times \mathbb{R} \xrightarrow{eq} 2}$$

$$\frac{\Gamma \vdash b : Bool \quad \Gamma \vdash a : A \quad \Gamma \vdash a' : A}{\Gamma \vdash cond(b,a,a') : A} \quad \longmapsto \quad \frac{b : \Gamma \to 2 \quad a : \Gamma \to A \quad a' : \Gamma \to A}{\Gamma \xrightarrow{\langle b,a,a' \rangle} 2 \times A \times A \xrightarrow{cond_A} A}$$

$$\frac{\Gamma, x : A \vdash a : B}{\Gamma \vdash \lambda x : A.a : A \to B} \quad \longmapsto \quad \frac{b : \Gamma \times A \to B}{\mathrm{curry}(b) : \Gamma \to B^A}$$

$$\frac{\Gamma \vdash f : A \to B \quad \Gamma \vdash a : A}{\Gamma \vdash fa : B} \quad \longmapsto \quad \frac{f : \Gamma \to B^A \quad a : \Gamma \to A}{\Gamma \xrightarrow{\langle f,a \rangle} B^A \times A \xrightarrow{ev} B}$$

$$\frac{\Gamma \vdash a : A}{\Gamma \vdash [a] : TA} \quad \longmapsto \quad \frac{a : \Gamma \to A}{\Gamma \xrightarrow{a} A \xrightarrow{\delta} P(A)}$$

$$\frac{\Gamma \vdash e : TA \quad \Gamma, x : A \vdash e' : TB}{\Gamma \vdash \mathsf{let}\ x \Leftarrow e\ \mathsf{in}\ e' : TB} \quad \longmapsto \quad \frac{e : \Gamma \to P(A) \quad e' : \Gamma \times A \to P(B)}{\Gamma \xrightarrow{\langle 1,e \rangle} \Gamma \times P(A) \xrightarrow{(e')^*} P(B)}$$

Figure 5.1: Translating the metalanguage to maps in **QBS**

6 Abstractness at First-Order Types

Fix a continuous probability measure ν on $\mathbb{R}$. We have shown that by interpreting new to be this measure, quasi-Borel spaces provide a categorical model of the nu-calculus. In this chapter we will prove that **QBS** is abstract at first-order types under this interpretation, meaning that if σ is a first-order type and $M_1, M_2 \in \mathrm{Exp}_\sigma(s)$, then

$$s \vdash M_1 \approx_\sigma M_2 \implies [\![M_1]\!]_{\neq s} = [\![M_2]\!]_{\neq s}\,.$$

In order to do this, we will first introduce the *privacy equation*, which will serve as a basic test for higher-order abstractness. We will show that **QBS** satisfies the privacy equation, introducing techniques that will be essential in our general proof of abstractness.

We will then construct a normal form for logical relations at first-order types. We will show that in order to prove abstractness at first-order types, it is both necessary and sufficient to show that passing to these normal forms preserves denotational equality.

Using the probability measure preserving group structure of Section 4.3 we will prove that normal forms preserve equality in **QBS**, completing our proof of abstractness. Our proof will use the ideas introduced in the proof of the privacy equation, along with an inductive construction on the structure of terms and their normal forms.

6.1 The Privacy Equation

The following well-known higher-order observational equivalence, established in Example 2.15 using the logical relations, is hard to model:

$$\lambda x : \mathsf{N}.\,\mathsf{false} \approx_{\mathsf{N}\to\mathsf{B}} \nu n.\lambda x : \mathsf{N}.x = n. \tag{6.1}$$

This equivalence states, essentially, that a fixed freshly generated name will never coincide with any other name. When proving higher-order abstractness of a categorical model of the nu-calculus, this equation, which we will call the *privacy equation*, is a good initial equivalence to consider.

In this section, we will show that **QBS** satisfies the privacy equation. This will serve both as an indication that **QBS** validates some amount of higher-order abstractness, as well as the base for our proof of abstractness in Section 6.3.

Remark 6.1. The equivalence of (6.1) states that a fixed freshly generated name will never coincide with any other name. This is similar to but distinct from the FRESH rule of the metalanguage, which asserts that any given name will not coincide with a fresh one. The difference is the order in which the name is given and the fresh name generated, and the distinction is related to the fact that ν and λ don't commute (cf. Example 2.8).

Specifically, the freshness criterion implies that

$$[\![\lambda x : \mathsf{N}.\nu n.x = n]\!] = [\![\lambda x : \mathsf{N}.\,\mathsf{false}]\!]\,,$$

should hold in an abstract categorical model. In fact, it is easy to prove this holds in any model by abstractness at ground types. On the other hand, the equivalence of (6.1) asserts that we should be able to prove

$$[\![\nu n.\lambda x : \mathsf{N}.x = n]\!] = [\![\lambda x : \mathsf{N}.\,\mathsf{false}]\!]\,.$$

We will show that this equality holds in **QBS**, although this need not be the case in general models.

Definition 6.2. Let $\mathcal{C}$ be a categorical model of the nu-calculus. We say that $\mathcal{C}$ satisfies the *privacy equation* if the following equality holds in $\mathcal{C}$:

$$[\![\nu n.\lambda x : \mathsf{N}.x = n]\!] = [\![\lambda x : \mathsf{N}.\,\mathsf{false}]\!]\,.$$

This equation is about the equality of elements of $T(N \to T(1+1))$. Specifically, if we translate this into an equation in our category (cf. Fig. 3.4), this asserts that

$$\mathsf{let}\ n \Leftarrow \mathsf{new}\ \mathsf{in}\ [\lambda x.[x = n]] = [\lambda x.[\mathsf{false}]].$$

In order to show that privacy holds in **QBS** we will first prove a similar — but distinct — equality in **QBS** (Proposition 6.4). This equality can be interpreted as the privacy equation of the metalanguage, and will be instrumental to our proofs of privacy and of abstractness at first-order.

Remark 6.3. Recall that we can identify the set $2^{\mathbb{R}}$ with the collection $\mathcal{B}(\mathbb{R})$ of Borel subsets of $\mathbb{R}$. In particular, this means that the evaluation map $ev : 2^{\mathbb{R}} \times \mathbb{R} \to 2$ is the "inclusion check" $(B, x) \mapsto x \in B$.

Proposition 6.4. *The following equality holds between elements of* $P(2^{\mathbb{R}})$ *in* **QBS***:*

$$\int [\{n\}]d\nu = [\emptyset].$$

Proof. These terms correspond to the measures $(\lambda r.\emptyset, \nu)$ and $(\lambda r.\{r\}, \delta_x)^*(\nu) = (\lambda r.\{r\}, \nu)$ on $2^{\mathbb{R}}$. We must therefore show that these are equal as pushforward measures on $2^{\mathbb{R}}$ equipped with the Borel on Borel algebra:

$$(\lambda r.\emptyset)_*(\nu) = (\lambda r.\{r\})_*(\nu).$$

This was established in Section 4.4 as a consequence of Theorem 4.40. □

We can now prove that privacy holds in **QBS**.

Theorem 6.5 (Privacy in **QBS**)**.** *The privacy equation holds in* **QBS***.*

Proof. Define a function $h : P(2^{\mathbb{R}}) \to P(P(2)^{\mathbb{R}})$ by

$$h(\mu) = \int_{2^{\mathbb{R}}} [\lambda x.[x \in B]] d\mu(B).$$

This is a quasi-Borel morphism as it is definable in the metalanguage. We verify using the rules of the metalanguage that

$$h([\emptyset]) = [\lambda x.[\mathsf{false}]],$$
$$h\left(\int \{n\} d\nu\right) = \int [\lambda x.[x = n]] d\nu.$$

By Proposition 6.4, we have that

$$[\![\lambda x : \mathsf{N}.\, \mathsf{false}]\!] = h([\emptyset]) = h\left(\int [\{n\}] d\nu\right) = [\![\nu n.\lambda x : \mathsf{N}.x = n]\!].$$

□

6.2 A Normal Form for Logical Relations

In this section we will construct a normal form that preserves logical relations, and therefore observational equivalence. We will then show that logically related terms have the same normal forms, providing a convenient way to reason about observational equivalence and abstractness.

Everything in this section will be done only for terms of first-order type.

6.2.1 η-Normal Forms

In order to construct the normal forms for logical relations, we need an η-normal form for canonical terms at first-order types. We will show that converting a term to η-normal form preserves both logical relations and equality in the metalanguage, so that we are able to assume all canonical terms are of this form, whether reasoning about observational equivalence or proving denotational equalities in categorical models (cf. Proposition 3.15).

Notation 6.6. Let $s = \{n_1, \ldots, n_k\}$ be a set of k distinct names. We let the nu-calculus term

$$\mathsf{case}\ x \in s\ \mathsf{then}\ M_x\ \mathsf{else}\ M_0$$

denote the expression

$$\mathsf{if}\ x = n_1\ \mathsf{then}\ M_{n_1}\ \mathsf{else}\ (\mathsf{if}\ x = n_2\ \mathsf{then}\ M_{n_2}\ \mathsf{else}\ (\cdots(\mathsf{if}\ x = n_k\ \mathsf{then}\ M_{n_k}\ \mathsf{else}\ M_0)\cdots)).$$

Definition 6.7 (η-Normal Forms)**.** Let σ be a first-order type and let $C \in \mathrm{Can}_\sigma(s)$. We construct the normal form $D \in \mathrm{Can}_\sigma(s)$ for C by induction on σ.

If σ is a ground type, we let $D = C$. Otherwise, we consider the cases where $\sigma = \mathsf{B} \to \tau$ and $\sigma = \mathsf{N} \to \tau$ separately.

Suppose $\sigma = \mathsf{B} \to \tau$ and write $C = \lambda x : \mathsf{B}.M$ for $M \in \mathrm{Exp}_\tau(s, \{x : \mathsf{B}\})$. By Termination (Theorem 2.3), we can find sets of names t_0, t_1 and canonical terms $C_i \in \mathrm{Can}_\tau(s \sqcup t_i)$ such that

$$s \vdash M[\mathsf{true}\,/x] \Downarrow_\tau (t_1)C_1 \quad \text{and} \quad s \vdash M[\mathsf{false}\,/x] \Downarrow_\tau (t_0)C_0.$$

As τ is a smaller type than σ, we have already constructed η-normal forms D_0, D_1 for C_0, C_1. We then let

$$D = \lambda x : \mathsf{B}.\mathsf{if}\ x = \mathsf{true}\ \mathsf{then}\ \nu t_1.D_1\ \mathsf{else}\ \nu t_0.D_0.$$

Now suppose that $\sigma = \mathsf{N} \to \tau$ and write $C = \lambda x : \mathsf{N}.M$ for $M \in \mathrm{Exp}_\tau(s, \{x : \mathsf{N}\})$. By Termination, for each $n \in s$ we can find a set of names t_n and a term $C_n \in \mathrm{Can}_\tau(s \sqcup t_n)$ such that $s \vdash M[n/x] \Downarrow_\tau (t_n)C_n$. Also by Termination we can find names t_0 and a term $C_0 \in \mathrm{Can}_\tau(s \sqcup \{x\} \sqcup t_0)$ such that $s \sqcup \{x\} \vdash M \Downarrow_\tau (t_0)C_0$. As τ is a smaller type than σ, we have already constructed η-normal forms D_n for C_n $(n \in s)$ and D_0 for C_0. We then let

$$D = \lambda x : \mathsf{N}.\mathsf{case}\ x \in s\ \mathsf{then}\ \nu t_x.D_x\ \mathsf{else}\ \nu t_0.D_0.$$

We will now show that passing to η-normal form preserves equality in the metalanguage, and hence that terms are logically related to their η-normal forms.

Lemma 6.8. *If $M \in \mathrm{Exp}_\sigma(s, \Gamma \sqcup \{x : \mathsf{N}\})$, then for all $n \in s$,*

$$[\![s, \Gamma \sqcup \{x : \mathsf{N}\}]\!]\,; (\neq s), x = n \vdash [\![M]\!] = [\![M[n/x]]\!]$$

is provable in the metalanguage. Similarly, if $M \in \mathrm{Exp}_\sigma(s, \Gamma \sqcup \{x : \mathsf{B}\})$, then for $b = \mathsf{true}, \mathsf{false}$ the metalanguage proves

$$[\![s, \Gamma \sqcup \{x : \mathsf{B}\}]\!]\,; (\neq s), x = b \vdash [\![M]\!] = [\![M[b/x]]\!]\,.$$

Proof. This is a simple proof by induction on the structure of M, using the fact that equality is preserved by the formation of terms. □

Lemma 6.9. *If the metalanguage proves*

$$[\![s, x : \mathsf{N}]\!]\,; (\neq s \sqcup \{x\}) \vdash a = b \quad \textit{and} \quad [\![s, x : \mathsf{N}]\!]\,; (\neq s), x = n \vdash a = b$$

for all $n \in s$, then

$$[\![s, x : \mathsf{N}]\!]\,; (\neq s) \vdash a = b$$

is also provable in the metalanguage.

Proof. Let $s = \{n_1, \ldots, n_k\}$ and let $\Phi_i = \{x \neq n_1, \ldots, x \neq n_i\}$ for $i \leq k$. We argue by induction on $(k - i)$ that $[\![s \sqcup \{x\}]\!]\,; (\neq s), \Phi_i \vdash a = b$. The case $i = k$ is given. Now suppose we have shown this for $i + 1$. Using the equational rule for Booleans, we see that

$$\frac{[\![s \sqcup \{x\}]\!]\,; (\neq s), \Phi_{i+1} \vdash a = b \quad [\![s \sqcup \{x\}]\!]\,; (\neq s), \Phi_i, x = n_{i+1} \vdash a = b}{[\![s \sqcup \{x\}]\!]\,; (\neq s), \Phi_i \vdash a = b}\,,$$

so it suffices to show that

$$[\![s \sqcup \{x\}]\!]\,; (\neq s), \Phi_i, x = n_{i+1} \vdash a = b,$$

which follows because $[\![s \sqcup \{x\}]\!]\,; (\neq s), x = n_{i+1} \vdash a = b$. □

Proposition 6.10. *Let σ be a first-order type and $C \in \mathrm{Can}_\sigma(s)$, and suppose that D is the η-normal form for C. Then $[\![s]\!]\,;(\neq s) \vdash |C| = |D|$ is provable in the metalanguage.*

In particular, the metalanguage proves $[\![s]\!]\,;(\neq s) \vdash [\![C]\!] = [\![D]\!]$, and $[\![C]\!]_{\neq s} = [\![D]\!]_{\neq s}$ in all categorical models of the nu-calculus.

Proof. We prove this by induction on σ. If σ is a ground type this is obvious. Now consider $C = \lambda x.M$.

Suppose that $\sigma = \mathsf{B} \to \tau$ and that we have already proven this for canonical terms of type τ. Then we have sets of names t_0, t_1 and canonical terms $C_i \in \mathrm{Can}_\tau(s \sqcup t_i)$ such that

$$s \vdash M[\mathsf{true}\,/x] \Downarrow_\tau (t_1)C_1 \quad \text{and} \quad s \vdash M[\mathsf{false}\,/x] \Downarrow_\tau (t_0)C_0.$$

Letting D_0, D_1 be the η-normal forms for C_0, C_1, we have that

$$D = \lambda x : \mathsf{B}.\mathsf{if}\ x = \mathsf{true}\ \mathsf{then}\ \nu t_1.D_1\ \mathsf{else}\ \nu t_0.D_0.$$

By our inductive hypothesis and the FRESH rule, $[\![s]\!]\,;(\neq s) \vdash [\![\nu t_i.C_i]\!] = [\![\nu t_i.D_i]\!]$. By Soundness (Theorem 3.3),

$$[\![s]\!]\,;(\neq s) \vdash [\![\nu t_1.C_1]\!] = [\![M[\mathsf{true}\,/x]]\!] \quad \text{and} \quad [\![s]\!]\,;(\neq s) \vdash [\![\nu t_0.C_0]\!] = [\![M[\mathsf{false}\,/x]]\!]\,.$$

Therefore, it suffices to show

$$[\![s, x : \mathsf{B}]\!]\,;(\neq s), x = \mathsf{true} \vdash [\![M]\!] = [\![M[\mathsf{true}\,/x]]\!]\,,$$
$$[\![s, x : \mathsf{B}]\!]\,;(\neq s), x = \mathsf{false} \vdash [\![M]\!] = [\![M[\mathsf{false}\,/x]]\!]\,,$$

which follow by Lemma 6.8.

Now suppose that $\sigma = \mathsf{N} \to \tau$ and that we have already proven this for canonical terms of type τ. We have, for $n \in s$, a set of names t_n and a canonical term $C_n \in \mathrm{Can}_\tau(s \sqcup t_n)$ such that $s \vdash M[n/x] \Downarrow_\tau (t_n)C_n$, and we have a set of names t_0 and a canonical term $M_0 \in \mathrm{Can}_\tau(s \sqcup \{x\} \sqcup t_0)$ such that $s \sqcup \{x\} \vdash M \Downarrow_\tau (t_0)C_0$. Letting D_n, D_0 be the corresponding η-normal forms, we have that

$$D = \lambda x : \mathsf{N}.\mathsf{case}\ x \in s\ \mathsf{then}\ \nu t_x.D_x\ \mathsf{else}\ \nu t_0.D_0.$$

By our inductive hypothesis and the FRESH rule,

$$[\![s]\!]\,;(\neq s) \vdash [\![\nu t_n.C_n]\!] = [\![\nu t_n.D_n]\!] \quad \text{and} \quad [\![s \sqcup \{x\}]\!]\,;(\neq s \sqcup x) \vdash [\![\nu t_0.C_0]\!] = [\![\nu t_0.D_0]\!]\,.$$

By Soundness,

$$[\![s]\!]\,;(\neq s) \vdash [\![\nu t_n.C_n]\!] = [\![M[n/x]]\!] \quad \text{and} \quad [\![s \sqcup \{x\}]\!]\,;(\neq s \sqcup \{x\}) \vdash [\![\nu t_0.C_0]\!] = [\![M]\!]\,.$$

If follows by Lemmas 6.8 and 6.9 that

$$[\![s, x : \mathsf{N}]\!]\,;(\neq s) \vdash [\![M]\!] = [\![\mathsf{case}\ x \in s\ \mathsf{then}\ \nu t_x.D_x\ \mathsf{else}\ \nu t_0.D_0]\!]\,,$$

so that $[\![s]\!]\,;(\neq s) \vdash |C| = |D|$.

This of course implies that $[\![s]\!]\,;(\neq s) \vdash [\![C]\!] = [\![D]\!]$, and so so by Proposition 3.15 the analogous statement holds in all categorical models of the nu-calculus. □

Proposition 6.11. *Let $C \in \mathrm{Can}_\sigma(s)$ for a first-order type σ and let D be the η-normal form for C. Then $C\,(Id_s)_\sigma\,D$.*

Proof. This follows by an easy induction on the structure of the terms. Alternatively, by Proposition 6.10 and Theorem 3.4 we have $s \vdash C \approx_\sigma D$, so by Theorem 2.14 we have $C\,(Id_s)_\sigma\,D$. □

By Proposition 6.10, when reasoning in categorical models of the nu-calculus — in particular **QBS** — we can always assume canonical terms are in η-normal form. Similarly, by Proposition 6.11, when reasoning about logical relations and observational equivalence we may always assume canonical terms are in η-normal form.

We also note that the η-normal form is well-defined:

Proposition 6.12. *Let σ be a first-order type and $C \in \mathrm{Can}_\sigma(s)$, and suppose that D is an η-normal form for C. Then D is well-defined up to the renaming of bound variables and names.*

Proof. This follows from the assertion in Theorem 2.3 that expressions of the nu-calculus evaluate to unique canonical terms. □

6.2.2 Construction of the Normal Form

We will now construct a normal form for terms of first-order type. To do this we will use logical relations, which coincide with observational equivalence at first-order types.

Example 6.13. Consider the term $\nu n.\lambda x : \mathsf{N}.x = n$. In Example 2.15 we showed using logical relations that it is observationally equivalent to the term $\lambda x : \mathsf{N}.\,\mathsf{false}$, which omits the name n.

Example 6.14. Consider the term

$$C = \lambda x : \mathsf{N}.\mathsf{if}\ x = a\ \mathsf{then}\ b\ \mathsf{else\ if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ x.$$

Given names a, b, this is a function that swaps a and b and is otherwise the identity.

If we let $s = \{a, b\}$, it is clear that $s \vdash C \approx_{\mathsf{N}\to\mathsf{N}} C$, so that $C\,(Id_s)_{\mathsf{N}\to\mathsf{N}}\,C$. It is also easy to verify that the relation $C\,\emptyset_{\mathsf{N}\to\mathsf{N}}\,C$ holds — intuitively, this is because for C to return a one must know b and to return b one must know a, and we are not given either of them.

We then eliminate a, b by defining $N = (\lambda x : \mathsf{N}.x)$. Because $C\,\emptyset_{\mathsf{N}\to\mathsf{N}}\,C$, it is clear that $C\,\emptyset_{\mathsf{N}\to\mathsf{N}}\,N$. Therefore we have $\nu a.\nu b.C\,\emptyset_{\mathsf{N}\to\mathsf{N}}\,N$, which implies that $\nu a.\nu b.C \approx_{\mathsf{N}\to\mathsf{N}} N$. Thus, N is a term that is equivalent to $\nu a.\nu b.C$, eliminating the names a, b.

As we can see in this example, the choice of partial bijection in the logical relations is not unique, leading to many possible choices of names to exclude. Therefore, in our construction of the normal form, we will consider the minimal partial bijections that satisfy the logical relation.

Lemma 6.15. *The logical relations are transitive at first-order types σ. This means that if $M_i \in \mathrm{Exp}_\sigma(s_i)$ for $i = 0, 1, 2$ and $R : s_0 \rightleftharpoons s_1, S : s_1 \rightleftharpoons s_2$ are partial bijections such that $M_0 \, R_\sigma \, M_1$ and $M_1 \, S_\sigma \, M_2$, then $M_0 \, (R \circ S)_\sigma \, M_2$. Here $R \circ S$ denotes composition of relations, meaning that $m \, (R \circ S) \, n$ iff there is some z such that $m \, R \, z$ and $z \, S \, n$.*

Proof. We prove this by induction on the structure of our terms. We first consider the case that $M_i = C_i$ are canonical. If σ is a ground type, then this is clear. Otherwise, $\sigma = \mathsf{B} \to \tau$ or $\sigma = \mathsf{N} \to \tau$ for some first-order type τ, and we assume by induction that we have already shown that logical relations are transitive for expressions of type τ.

If $\sigma = \mathsf{B} \to \tau$, then by Proposition 2.13 we need to show that for $b = \mathsf{true}, \mathsf{false}$ we have $C_0 b \, (R \circ S)_\tau \, C_2 b$. By assumption, $C_0 b \, R_\tau \, C_1 b$ and $C_1 b \, S_\tau \, C_2 b$, so this follows by transitivity at type τ.

If $\sigma = \mathsf{N} \to \tau$, then by Proposition 2.13 we need to show that for $(m, n) \in R \circ S$ we have $C_0 m \, (R \circ S)_\tau \, C_2 n$, and for $n \notin s_0 \cup s_2$ we have $C_0 n \, \left((R \circ S) \sqcup Id_{\{n\}}\right)_\tau \, C_2 n$. In the first case, there is some $z \in s_1$ such that $m \, R \, z$ and $z \, S \, n$. By assumption, $C_0 m \, R_\tau \, C_1 z$ and $C_1 z \, S_\tau \, C_2 n$, so this follows by transitivity at type τ. In the second case, we note that $(R \circ S) \sqcup Id_{\{n\}} = (R \sqcup Id_{\{n\}}) \circ (S \sqcup Id_{\{n\}})$, so this again follows by transitivity at type τ.

Now consider the case of expressions and suppose we have proven transitivity for canonical terms of type σ. Let $C_i \in \mathrm{Can}_\sigma(s_i \sqcup t_i)$ be chosen such that $s_i \vdash M_i \Downarrow_\sigma (t_i) C_i$. Because $M_0 \, R_\sigma \, M_1$ and $M_1 \, S_\sigma \, M_2$, there are partial bijections $R' : t_0 \rightleftharpoons t_1$ and $S' : t_1 \rightleftharpoons t_2$ such that $C_0 \, (R \sqcup R')_\sigma \, C_1$ and $C_1 \, (S \sqcup S')_\sigma \, C_2$. Note that $(R \sqcup R') \circ (S \sqcup S') = (R \circ S) \sqcup (R' \circ S')$, so by transitivity for canonical terms of type σ we have $C_0 \, ((R \circ S) \sqcup (R' \circ S'))_\sigma \, C_2$ and so $M_0 \, (R \circ S)_\sigma \, M_2$. □

Proposition 6.16. *Let σ be a first-order type and $M \in \mathrm{Exp}_\sigma(s \sqcup t)$. There is a unique minimal $s \subseteq u \subseteq s \sqcup t$ such that $M \, (Id_u)_\sigma \, M$.*

Proof. If $s \subseteq u_0, u_1 \subseteq s \sqcup t$, $M \, (Id_{u_0})_\sigma \, M$ and $M \, (Id_{u_1})_\sigma \, M$, then $Id_{u_0} \circ Id_{u_1} = Id_{(u_0 \cap u_1)}$ so by Lemma 6.15 we have $M \, (Id_{(u_0 \cap u_1)})_\sigma \, M$. We can therefore take u to be the intersection of all such sets. □

Remark 6.17. The minimal u in Proposition 6.16 depends on the choice of s. In general, if we were to partition the names differently, letting $s \sqcup t = s' \sqcup t'$, then the minimal $s' \subseteq u'$ such that $M \, (Id_{u'})_\sigma \, M$ need not be the same as u.

Corollary 6.18. *Let σ be a first-order type. Let $M_i \in \mathrm{Exp}_\sigma(s \sqcup t_i)$ and suppose there is some $R : t_1 \rightleftharpoons t_2$ such that $M_1 \, (Id_s \sqcup R)_\sigma \, M_2$. Let $u_i \subseteq t_i$ be the minimal set such that $M_i \, (Id_{s \sqcup u_i})_\sigma \, M_i$. Then after possibly renaming names in u_i we have $u_1 = u_2 = u$, $Id_u \subseteq R$ and $M_1 \, (Id_{s \sqcup u})_\sigma \, M_2$.*

Proof. We know that $R \circ R^{-1} = Id_{\mathrm{Dom}(R)}$, so $M_1 \, (Id_{s \sqcup \mathrm{Dom}(R)})_\sigma \, M_1$ by Lemma 6.15. By minimality, $u_1 \subseteq \mathrm{Dom}(R)$.

Now consider the restriction $R \restriction_{u_1}$ of R to u_1. Because $R \restriction_{u_1} = Id_{u_1} \circ R$, we have by Lemma 6.15 that $M_1 \, (Id_s \sqcup R \restriction_{u_1})_\sigma \, M_2$.

A symmetric argument shows that u_2 is contained in the range of $R \restriction_{u_1}$. By minimality of u_1, this must be a bijection. Therefore, after renaming names, we can assume that $u_1 = u_2 = u$ and $R \restriction_u = Id_u$. □

We are now ready to define the normal form for logical relations.

Definition 6.19 (Normal Form for Logical Relations)**.** Let σ be a first-order type. Let $M \in \mathrm{Exp}_\sigma(s \sqcup t)$ and suppose that $M\,(Id_s)_\sigma\,M$. We define the normal form $\langle M, s\rangle \in \mathrm{Exp}_\sigma(s)$; in the case that $M = C \in \mathrm{Can}_\sigma(s \sqcup t)$, the term $\langle C, s\rangle$ will be canonical as well. We construct this by induction on the type σ, and we break this up into four cases: canonical terms of ground type, canonical terms of type $\mathsf{B} \to \tau$, canonical terms of type $\mathsf{N} \to \tau$ and expressions.

Ground case: If σ is a ground type and C is canonical, then we let $\langle C, s\rangle = C$.

Function case $\mathsf{B} \to \tau$: Suppose C is a canonical term of type $\mathsf{B} \to \tau$ and that we have already constructed normal forms for expressions of type τ. Let D be the η-normal form of C, so that

$$D = \lambda x : \mathsf{B}.\mathsf{if}\ x = \mathsf{true}\ \mathsf{then}\ M_1\ \mathsf{else}\ M_0$$

for some $M_1, M_0 \in \mathrm{Exp}_\tau(s \sqcup t)$. By Proposition 6.11 we have $C\,(Id_s)_\sigma\,D$, so by Lemma 6.15 we have $D\,(Id_s)_\sigma\,D$. This implies that $M_1\,(Id_s)_\tau\,M_1$ and $M_0\,(Id_s)_\tau\,M_0$. We then define

$$\langle C, s\rangle = \lambda x : \mathsf{B}.\mathsf{if}\ x = \mathsf{true}\ \mathsf{then}\ \langle M_1, s\rangle\ \mathsf{else}\ \langle M_0, s\rangle.$$

Function case $\mathsf{N} \to \tau$: Suppose that C is a canonical term of type $\mathsf{N} \to \tau$ and that we have already constructed normal forms for expressions of type τ. Let D be the η-normal form of C, so that

$$D = \lambda x : \mathsf{N}.\mathsf{case}\ x \in s \sqcup t\ \mathsf{then}\ M_x\ \mathsf{else}\ M_0$$

for some $M_n \in \mathrm{Exp}_\tau(s \sqcup t)$ $(n \in s \sqcup t)$ and $M_0 \in \mathrm{Exp}_\tau(s \sqcup t \sqcup \{x\})$. By Proposition 6.11 we have $C\,(Id_s)_\sigma\,D$, so by Lemma 6.15 we have $D\,(Id_s)_\sigma\,D$. This implies that $M_0\,(Id_{s\oplus\{x\}})_\tau\,M_0$ and $M_n\,(Id_s)_\tau\,M_n$ for all $n \in s$. We then define

$$\langle C, s\rangle = \lambda x : \mathsf{N}.\mathsf{case}\ x \in s\ \mathsf{then}\ \langle M_x, s\rangle\ \mathsf{else}\ \langle M_0, s \sqcup \{x\}\rangle.$$

Expression case: Suppose that we have constructed normal forms for canonical terms of type σ. Because $M\,(Id_s)_\sigma\,M$, there is some $C \in \mathrm{Can}_\sigma(s \sqcup t \sqcup u \sqcup w)$ such that $s \sqcup t \vdash M \Downarrow_\sigma (u \sqcup w)C$ and $C\,(Id_{s\sqcup u})_\sigma\,C$. By Proposition 6.16, we can assume that u is the unique minimal subset of $u \oplus w$ such that this holds. We then define

$$\langle M, s\rangle = \nu u.\,\langle C, s \sqcup u\rangle\,.$$

Notation 6.20. If $s = \emptyset$, we will write $\langle M\rangle$ to denote the normal form of $(M, \emptyset)$.

Example 6.21. We can compute the following normal forms:

1. $\langle \nu n.\lambda x : \mathsf{N}.x = n\rangle = \lambda x : \mathsf{N}.\,\mathsf{false}$. To see this, note that

$$\nu n.\lambda x : \mathsf{N}.x = n \Downarrow_{\mathsf{N}\to\mathsf{B}} (n)\lambda x : \mathsf{N}.x = n \quad \text{and} \quad (\lambda x : \mathsf{N}.x = n)\,\emptyset_{\mathsf{N}\to\mathsf{B}}\,(\lambda x : \mathsf{N}.x = n).$$

The term $x = n$ is an abbreviation for $\mathsf{if}\ x \in \{n\}\ \mathsf{then}\ \mathsf{true}\ \mathsf{else}\ \mathsf{false}$, so that

$$\langle \nu n.\lambda x : \mathsf{N}.x = n\rangle = \langle \lambda x : \mathsf{N}.x = n\rangle = \lambda x : \mathsf{N}.\,\mathsf{false}\,.$$

2. $\langle \nu a.\nu b.\lambda x : \mathsf{N}.\mathsf{if}\ x = a\ \mathsf{then}\ b\ \mathsf{else}\ \mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ x\rangle = \lambda x : \mathsf{N}.x$. This follows as in the previous example (cf. Example 6.14).

3. We also have

$$\langle \nu a.\lambda x : \mathsf{N}.\nu b.\lambda y : \mathsf{N}.\mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ b\rangle = \lambda x : \mathsf{N}.\nu b.\lambda y : \mathsf{N}.b.$$

Intuitively, this is because b is generated after x is specified, so they can never be equal and we can ignore this case. To show this is the normal form, we first verify that

$$(\lambda x : \mathsf{N}.\nu b.\lambda y : \mathsf{N}.\mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ b) \\ \emptyset_{\mathsf{N}\to\mathsf{N}\to\mathsf{N}}\,(\lambda x : \mathsf{N}.\nu b.\lambda y : \mathsf{N}.\mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ b),$$

so that

$$\langle \nu a.\lambda x : \mathsf{N}.\nu b.\lambda y : \mathsf{N}.\mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ b\rangle \\ = \langle \lambda x : \mathsf{N}.\nu b.\lambda y : \mathsf{N}.\mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ b\rangle\,.$$

Next, we check that

$$(\lambda y : \mathsf{N}.\mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ b)\,(Id_{\{x\}})_{\mathsf{N}\to\mathsf{N}}\,(\lambda y : \mathsf{N}.\mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ b)$$

fails to hold, so that

$$\langle \nu b.\lambda y : \mathsf{N}.\mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ b, \{x\}\rangle = \nu b.\,\langle \lambda y : \mathsf{N}.\mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ b, \{x, b\}\rangle\,.$$

Finally, it is clear that $\{x, y, a, b\} \vdash \mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ b \Downarrow_{\mathsf{N}} b$, so that

$$\begin{aligned}\langle \nu a.\lambda x : \mathsf{N}.\nu b.\lambda y : \mathsf{N}.\mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ b\rangle \\ &= \langle \lambda x : \mathsf{N}.\nu b.\lambda y : \mathsf{N}.\mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ b\rangle \\ &= \lambda x : \mathsf{N}.\,\langle \nu b.\lambda y : \mathsf{N}.\mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ b, \{x\}\rangle \\ &= \lambda x : \mathsf{N}.\nu b.\,\langle \lambda y : \mathsf{N}.\mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ b, \{x, b\}\rangle \\ &= \lambda x : \mathsf{N}.\nu b.\lambda y : \mathsf{N}.\,\langle \mathsf{if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ b, \{x, y, b\}\rangle \\ &= \lambda x : \mathsf{N}.\nu b.\lambda y : \mathsf{N}.b.\end{aligned}$$

Remark 6.22. This construction differs from the η-normal forms in the case of expressions and canonical terms of type $\mathsf{N} \to \tau$. In the case of expressions, we restrict ν to range only over the minimal set u of freshly generated names instead of all generated names. In the case of canonical terms of type $\mathsf{N} \to \tau$, we allow x to range over only the names in s, while the η-normal form allows x to range over the names in both s and t.

The logical relation is needed to keep track of names we can remove without affecting the semantics of our terms. In the base case, when $C \in \mathrm{Can}_{\mathsf{N}}(s \sqcup t)$, the logical relation ensures that $C = \langle C, s\rangle \notin t$ so that we are in fact omitting the names in t. The recursive nature of the logical relations captures the interactions between names generated at different points in the term, and these ensure that we are able to consistently eliminate names in the inductive steps of our construction.

Proposition 6.23. *Let σ be a first-order type. Let $M \in \mathrm{Exp}_\sigma(s \sqcup t)$ and suppose that $M\,(Id_s)_\sigma\,M$. The normal form $\langle M, s\rangle$ is well-defined up to renaming bound variables and names.*

Proof. We argue by induction on our construction of the normal form $\langle M, s\rangle$. In the case of canonical terms, this follows by Proposition 6.12 and the inductive hypothesis. In the case of expressions, this follows because by Proposition 6.16 there is a canonical choice of minimal u in our construction. □

Proposition 6.24. *The normal forms preserve logical relations: if σ is a first-order type, $M \in \mathrm{Exp}_\sigma(s \sqcup t)$ and $M\,(Id_s)_\sigma\,M$, then $M\,(Id_s)_\sigma\,\langle M, s\rangle$.*

Proof. We argue by induction on our construction of the normal form $\langle M, s\rangle$. In the case of expressions, this follows directly from the inductive hypothesis. By Proposition 6.11 and Lemma 6.15, we only need to consider canonical terms in η-normal form. In the case that $\sigma = \mathsf{N} \to \tau$, we write

$$C = \lambda x : \mathsf{N}.\mathsf{case}\ x \in s \sqcup t\ \mathsf{then}\ M_x\ \mathsf{else}\ M_0.$$

By Proposition 2.13 we only need to verify that $M_0\,(Id_{s\sqcup\{x\}})_\tau\,\langle M_0, s \sqcup \{x\}\rangle$ and that $M_n\,(Id_s)_\tau\,\langle M_n, s\rangle$ for $n \in s$, which follows by the inductive hypothesis. The case that $\sigma = \mathsf{B} \to \tau$ is handled similarly. □

We can now equate the problem of checking if two terms are logically related to one of verifying the equality of their normal forms.

Theorem 6.25. *Let σ be a first-order type and let $M_i \in \mathrm{Exp}_\sigma(s \sqcup t_i)$ for $i = 1, 2$. The following are equivalent:*

1. *$M_1\,(Id_s)_\sigma\,M_2$.*
2. *$M_i\,(Id_s)_\sigma\,M_i$ and $\langle M_1, s\rangle = \langle M_2, s\rangle$ after possibly renaming bound variables and names.*

Proof. If $M_i\,(Id_s)_\sigma\,M_i$ and $\langle M_1, s\rangle = \langle M_2, s\rangle$, then $\langle M_1, s\rangle\,(Id_s)_\sigma\,\langle M_2, s\rangle$ and so by Proposition 6.24 and Lemma 6.15 we have $M_1\,(Id_s)_\sigma\,M_2$.

For the converse, suppose that $M_1\,(Id_s)_\sigma\,M_2$. Note that by Lemma 6.15, it is clear that $M_i\,(Id_s)_\sigma\,M_i$. To show that $\langle M_1, s\rangle = \langle M_2, s\rangle$, we argue by induction on the construction of the normal forms. The base case is clear. For the inductive step at canonical terms, by Proposition 6.11 and Lemma 6.15, we only need to consider canonical terms in η-normal form. In the case that $\sigma = \mathsf{N} \to \tau$, we write

$$C_i = \lambda x : \mathsf{N}.\mathsf{case}\ x \in s \sqcup t_i\ \mathsf{then}\ M_x^i\ \mathsf{else}\ M_0^i.$$

By definition of logical relations, because $C_1\,(Id_s)_\sigma\,C_2$, we have $M_0^1\,(Id_{s\sqcup\{x\}})_\sigma\,M_0^2$ and $M_n^1\,(Id_s)_\sigma\,M_n^2$ for $n \in s$. By our inductive hypothesis, this means that $\langle M_0^1, s \sqcup \{x\}\rangle = \langle M_0^2, s \sqcup \{x\}\rangle$ and $\langle M_n^1, s\rangle = \langle M_n^2, s\rangle$ for $n \in s$. It follows that $\langle C_1, s\rangle = \langle C_2, s\rangle$. The case that $\sigma = \mathsf{B} \to \tau$ is the same.

In the case of expressions, let $C_i \in \mathrm{Can}_\sigma(s \sqcup t_i \sqcup t_i')$ be the canonical terms such that $s \sqcup t_i \vdash M_i \Downarrow (t_i')C_i$. We have minimal $u_i \subseteq t_i'$ such that $C_i \, (Id_{s \sqcup u_i})_\sigma \, C_i$, and we have defined $\langle M_i, s\rangle = \nu u_i . \langle C_i, s \sqcup u_i\rangle$. We know that $M_1 \, (Id_s)_\sigma \, M_2$, so there is some $R : t_1' \rightleftharpoons t_2'$ such that $C_1 \, (Id_s \sqcup R)_\sigma \, C_2$. By Corollary 6.18, after possibly renaming names we have $u_1 = u_2 = u$ and $C_1 \, (Id_{s \sqcup u})_\sigma \, C_2$. We therefore have $\langle C_1, s \sqcup u\rangle = \langle C_2, s \sqcup u\rangle$ by our inductive hypothesis and so $\langle M_1, s\rangle = \langle M_2, s\rangle$. □

6.3 Abstractness at First-Order in QBS

We will now show that at first-order types, eliminating names is enough to prove abstractness. Specifically, we will show that if $\mathcal{C}$ is a categorical model of the nu-calculus and passing to normal forms preserves equality in $\mathcal{C}$, then $\mathcal{C}$ is abstract at first-order types.

Theorem 6.26. *Let $\mathcal{C}$ be a categorical model of the nu-calculus. $\mathcal{C}$ is abstract at first-order types if and only if for all first-order types σ and all $M \in \mathrm{Exp}_\sigma(s)$ we have*

$$[\![M]\!]_{\neq s} = [\![\langle M, s\rangle]\!]_{\neq s} .$$

Proof. By Proposition 6.24 and Theorem 2.14 it is clear that this is necessary. To see that it is sufficient, let σ be a first-order type and let $M_1, M_2 \in \mathrm{Exp}_\sigma(s)$. We need to show that

$$s \vdash M_1 \approx_\sigma M_2 \implies [\![M_1]\!]_{\neq s} = [\![M_2]\!]_{\neq s} .$$

So suppose that $s \vdash M_1 \approx_\sigma M_2$. By the completeness of logical relations at first-order types (Theorem 2.14), we know that $M_1 \, (Id_s)_\sigma \, M_2$. By Theorem 6.25, it follows that $\langle M_1, s\rangle = \langle M_2, s\rangle$. Using our assumption, we see that

$$[\![M_1]\!]_{\neq s} = [\![\langle M_1, s\rangle]\!]_{\neq s} = [\![\langle M_2, s\rangle]\!]_{\neq s} = [\![M_2]\!]_{\neq s} .$$

□

We will now show that passing to normal forms for logical relations preserves equality in **QBS**. In order to do this, we will we will argue as we did in the proof of Theorem 6.5, showing that equality between terms and their normal forms reduces to the statement of Proposition 6.4. By Theorem 6.26, this will complete our proof that quasi-Borel spaces are an abstract model of the nu-calculus at first-order types.

To build this reduction, we will use a probability measure preserving group structure on $\mathbb{R}$. By Proposition 4.32, there is an abelian probability measure preserving group structure $+$ on $\mathbb{R}$, which we will fix for the remainder of this section. Note that by Theorem 5.5 and Corollary 5.6 the map $\mathbb{R} \times \mathbb{R} \to \mathbb{R}, (x, y) \mapsto x - y$ is a quasi-Borel map, so this group structure is compatible with the quasi-Borel structure on $\mathbb{R}$ as well.

Example 6.27. Consider the term $M = \nu n.\lambda x : \mathsf{N}.x = n$. We have seen in Example 6.21 that its normal form is $\langle M\rangle = \lambda x : \mathsf{N}.\,\mathsf{false}$, and we proved in Theorem 6.5 using

Proposition 6.4 that $[\![M]\!] = [\![\langle M \rangle]\!]$. Unpacking this proof, we see that it consists of one key ingredient: a function $f : 2^{\mathbb{R}} \to P(2)^{\mathbb{R}}$ satisfying

$$f(\emptyset) = \lambda x.[\mathsf{false}] \quad \text{and} \quad f(\{n\}) = \lambda x.[x = n].$$

Given such a function, we can take $h(x) = [f(x)] : 2^{\mathbb{R}} \to P(P(2)^{\mathbb{R}})$ so that

$$h^*([\emptyset]) = [\lambda x.[\mathsf{false}]] = [\![\langle M \rangle]\!] \quad \text{and} \quad h^*\left(\int [\{n\}] d\nu\right) = \int [\lambda x.[x = n]] d\nu = [\![M]\!] .$$

Proposition 6.4 asserts that $[\emptyset] = \int [\{n\}] d\nu$, so that

$$[\![\langle M \rangle]\!] = h^*([\emptyset]) = h^*\left(\int [\{n\}] d\nu\right) = [\![M]\!] .$$

We note that we defined f by $f(B) = \lambda x.[x \in B]$, which is a quasi-Borel map as the inclusion check is quasi-Borel.

Example 6.28. Consider the term

$$M = \nu a.\nu b.\lambda x : \mathsf{N}.\mathsf{if}\ x = a\ \mathsf{then}\ b\ \mathsf{else\ if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ x.$$

We have seen in Example 6.21 that the normal form for this term is the identity map $\langle M \rangle = \lambda x : \mathsf{N}.x$. We would like to prove that $[\![M]\!] = [\![\langle M \rangle]\!]$, meaning that

$$[\lambda x.[x]] = \iint [\lambda x.[\mathsf{if}\ x = a\ \mathsf{then}\ b\ \mathsf{else\ if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ x]] d\nu(a) d\nu(b).$$

We define a function $f : 2^{\mathbb{R}} \times \mathbb{R} \times \mathbb{R} \to P(\mathbb{R})^{\mathbb{R}}$ by

$$f(B, a, b) = \lambda x : \mathbb{R}. \begin{cases} [(x - a) + b] & \text{if } x \in B + a, \\ [(x - b) + a] & \text{else if } x \in B + b, \\ [x] & \text{otherwise.} \end{cases}$$

When $B = \emptyset$, this gives the map $\lambda x.[x]$. On the other hand, when $B = \{n\}$ is a singleton, we have

$$f(\{n\}, a, b) = \lambda x.\mathsf{if}\ x = n + a\ \mathsf{then}\ n + b\ \mathsf{else\ if}\ x = n + b\ \mathsf{then}\ n + a\ \mathsf{else}\ x.$$

Because the group structure is compatible with the quasi-Borel structure on $\mathbb{R}$, the map f is quasi-Borel.

We now prove that $[\![M]\!] = [\![\langle M \rangle]\!]$. We define $g : 2^{\mathbb{R}} \to P(P(\mathbb{R})^{\mathbb{R}})$ by

$$g(B) = \iint [f(B, a, b)] d\nu(a) d\nu(b).$$

It follows that $g(\emptyset) = [\lambda x.[x]] = [\![\langle M \rangle]\!]$, and because we have chosen a ν preserving group structure we have

$$\begin{aligned} g(\{n\}) &= \iint [\lambda x.[\mathsf{if}\ x = n + a\ \mathsf{then}\ n + b\ \mathsf{else\ if}\ x = n + b\ \mathsf{then}\ n + a\ \mathsf{else}\ x]] d\nu(a) d\nu(b) \\ &= \iint [\lambda x.[\mathsf{if}\ x = a\ \mathsf{then}\ b\ \mathsf{else\ if}\ x = b\ \mathsf{then}\ a\ \mathsf{else}\ x]] d\nu(a) d\nu(b) = [\![M]\!] \end{aligned}$$

as in Proposition 4.33. Proposition 6.4 then implies that

$$[\![\langle M \rangle]\!] = g^*([\emptyset]) = g^*\left(\int [\{n\}] d\nu\right) = \int g(\{n\}) d\nu = [\![M]\!] .$$

Notation 6.29. If $r \in \mathbb{R}$ and $(n_1, \ldots, n_k) \in \mathbb{R}^k$, we let

$$r + (n_1, \ldots, n_k) = (r + n_1, \ldots, r + n_k).$$

Now suppose that σ is a first-order type and $M \in \mathrm{Exp}_\sigma(s)$. We will prove that $[\![M]\!] = [\![\langle M, s\rangle]\!]$ by constructing a function $f : 2^{\mathbb{R}} \times \mathbb{R}^{\neq s} \to P([\![\sigma]\!])$ satisfying

$$f(\emptyset, -) = [\![\langle M, s\rangle]\!]_{\neq s}(-) \quad \text{and} \quad f(\{n\}) = [\![M]\!]_{\neq s}(-),$$

as we did in the previous example, and applying Proposition 6.4.

We will construct this f inductively, parallel to the construction of the normal forms. In order to do this, we will provide a more general, parametrized version of this construction: given $M \in \mathrm{Exp}_\sigma(s \sqcup t)$ with $M\,(Id_s)_\sigma\,M$, we will construct a function $f : 2^{\mathbb{R}} \times \mathbb{R}^{\neq s \sqcup t} \to P([\![\sigma]\!])$ such that

$$f(\emptyset, -, \vec{t}) = [\![\langle M, s\rangle]\!]_{\neq s}(-) \quad \text{and} \quad f(\{n\}, -, \vec{t}) = [\![M]\!]_{\neq s}(-, n + \vec{t}).$$

We will use this parametrized version in the inductive step of our proof.

The construction of f itself is somewhat high-level, and is analogous to the difference between the η-normal form of a term and its normal form. It takes as arguments a set of name-permutations B, a sequence s of potentially leaked names, and a sequence of names t that are guaranteed to remain private. It then identifies the redundant parts of the η-normal form — where we compare against a private name t_i — and instead checks whether the name matches one of the names in $B + t_i$.

By selecting B to be a fresh permutation $\vec{t} \leftrightarrow \vec{t'}$, we recover the semantics of the η-normal form. On the other hand, by letting B be the empty set we skip redundant comparisons against private names, recovering the semantics of the normal form. We can then use the privacy equation to equate these two denotations, proving that each term is denotationally equivalent to its normal form.

Theorem 6.30. *Let σ be a first-order type and $M \in \mathrm{Exp}_\sigma(s \sqcup t)$. If $M\,(Id_s)_\sigma\,M$, then there is a quasi-Borel map*

$$f : 2^{\mathbb{R}} \times \mathbb{R}^{\neq s \sqcup t} \to P([\![\sigma]\!])$$

such that

$$f(\emptyset, \vec{s}, \vec{t}) = [\![\langle M, s\rangle]\!]_{\neq s}(\vec{s}) \quad \textit{and} \quad f(\{r\}, \vec{s}, \vec{t}) = [\![M]\!]_{\neq s \sqcup t}(\vec{s}, r + \vec{t})$$

whenever $(\vec{s}, r + \vec{t}) \in \mathbb{R}^{\neq s \sqcup t}$.

In the case that $M = C \in \mathrm{Can}_\sigma(s \sqcup t)$ is a canonical term, we can write $f(-) = [g(-)]$ for a quasi-Borel map

$$g : 2^{\mathbb{R}} \times \mathbb{R}^{\neq s \sqcup t} \to [\![\sigma]\!]$$

such that

$$g(\emptyset, \vec{s}, \vec{t}) = |\langle C, s\rangle|(\vec{s}) \quad \textit{and} \quad g(\{r\}, \vec{s}, \vec{t}) = |C|(\vec{s}, r + \vec{t})$$

whenever $(\vec{s}, r + \vec{t}) \in \mathbb{R}^{\neq s \sqcup t}$.

Abstractness at first-order types follows immediately:

Theorem 6.31. QBS *is abstract at first-order types. This means that for all first-order types σ and all $M_1, M_2 \in \mathrm{Exp}_\sigma(s)$,*

$$s \vdash M_1 \approx_\sigma M_2 \implies [\![M_1]\!]_{\neq s} = [\![M_2]\!]_{\neq s}\,.$$

Proof. We argue via Theorem 6.26 and show that normal forms preserve equality.

Let σ be a first-order type and let $M \in \mathrm{Exp}_\sigma(s)$. It is clear (cf. Remark 2.12) that $M\,(Id_s)_\sigma\,M$. By Theorem 6.30, there is a function

$$f : 2^{\mathbb{R}} \times \mathbb{R}^{\neq s} \to P([\![\sigma]\!])$$

such that

$$f(\emptyset, \vec{s}) = [\![\langle M, s\rangle]\!]_{\neq s}\,(\vec{s}) \;\text{ and }\; f(\{r\}, \vec{s}) = [\![M]\!]_{\neq s}\,(\vec{s})$$

for all $\vec{s} \in \mathbb{R}^{\neq s}$. For any fixed $\vec{s}$, we have

$$f^*([\emptyset], \vec{s}) = [\![\langle M, s\rangle]\!]_{\neq s}\,(\vec{s}) \;\text{ and }\; f^*\left(\int [\{n\}]d\nu, \vec{s}\right) = \int [\![M]\!]_{\neq s}\,(\vec{s})d\nu = [\![M]\!]_{\neq s}\,(\vec{s}).$$

Thus, by Proposition 6.4,

$$[\![\langle M, s\rangle]\!]_{\neq s}\,(\vec{s}) = f^*([\emptyset], \vec{s}) = f^*\left(\int [\{n\}]d\nu, \vec{s}\right) = [\![M]\!]_{\neq s}\,(\vec{s}).$$

□

Proof of Theorem 6.30. We construct f inductively, in parallel to the construction of the normal forms.

Ground case: If σ is a ground type and C is canonical, then $\langle C, s\rangle = C$ so we simply take $g = |C|$ and let $f = [g]$.

Function case $\mathsf{B} \to \tau$: Suppose that C is a canonical term of type $\mathsf{B} \to \tau$ and that we have already constructed these functions for expressions of type τ. By Proposition 6.10 we may assume without loss of generality that C is in η-normal form, so that

$$C = \lambda x : \mathsf{B}.\mathsf{if}\ x = \mathsf{true}\ \mathsf{then}\ M_1\ \mathsf{else}\ M_0.$$

By definition of logical relations and the normal form we have $M_1\,(Id_s)_\tau\,M_1$, $M_0\,(Id_s)_\tau\,M_0$ and

$$\langle C, s\rangle = \lambda x : \mathsf{B}.\mathsf{if}\ x = \mathsf{true}\ \mathsf{then}\ \langle M_1, s\rangle\ \mathsf{else}\ \langle M_0, s\rangle.$$

By our inductive hypothesis, we have functions

$$f_i : 2^{\mathbb{R}} \times \mathbb{R}^{\neq s \sqcup t} \to P([\![\tau]\!])$$

such that

$$f_i(\emptyset, \vec{s}, \vec{t}) = [\![\langle M_i, s\rangle]\!]_{\neq s}\,(\vec{s}) \;\text{ and }\; f_i(\{r\}, \vec{s}, \vec{t}) = [\![M_i]\!]_{\neq s \sqcup t}\,(\vec{s}, r + \vec{t})$$

whenever $(\vec{s}, r + \vec{t}) \in \mathbb{R}^{\neq s \sqcup t}$. We then define $g : 2^{\mathbb{R}} \times \mathbb{R}^{\neq s \sqcup t} \to [\![\sigma]\!]$ by

$$g(B, \vec{s}, \vec{t}) = \lambda x : \mathsf{B}.\mathsf{if}\ x = \mathsf{true}\ \mathsf{then}\ f_1(B, \vec{s}, \vec{t})\ \mathsf{else}\ f_0(B, \vec{s}, \vec{t}).$$

It is clear that $g(\emptyset, \vec{s}, \vec{t}) = |\langle C, s\rangle|\,(\vec{s})$ and $g(\{r\}, \vec{s}, \vec{t}) = |C|(\vec{s}, r + \vec{t})$, so $f = [g]$ is the function we want.

Function case $\mathsf{N} \to \tau$: Suppose that C is a canonical term of type $\mathsf{N} \to \tau$ and that we have already constructed these functions for expressions of type τ. By Proposition 6.10 we may assume without loss of generality that C is in η-normal form, so that

$$C = \lambda x : \mathsf{N}.\mathsf{case}\ x \in s \sqcup t\ \mathsf{then}\ M_x\ \mathsf{else}\ M_0.$$

By definition of logical relations and the normal form we have $M_n\,(Id_s)_\tau\,M_n$ for $n \in s$, $M_0\,(Id_{s \oplus \{x\}})_\tau\,M_0$ and

$$\langle C, s\rangle = \lambda x : \mathsf{N}.\mathsf{case}\ x \in s\ \mathsf{then}\ \langle M_x, s\rangle\ \mathsf{else}\ \langle M_0, s \sqcup \{x\}\rangle.$$

By our inductive hypothesis, we have functions

$$f_n : 2^{\mathbb{R}} \times \mathbb{R}^{\neq s \sqcup t} \to P([\![\tau]\!]), n \in s \quad \text{and} \quad f_0 : 2^{\mathbb{R}} \times \mathbb{R}^{\neq s \sqcup t \sqcup \{x\}} \to P([\![\tau]\!])$$

such that

$$\begin{aligned} f_n(\emptyset, \vec{s}, \vec{t}) &= [\![\langle M_n, s\rangle]\!]_{\neq s}\,(\vec{s}), & f_n(\{r\}, \vec{s}, \vec{t}) &= [\![M_n]\!]_{\neq s \sqcup t}\,(\vec{s}, r + \vec{t}), \\ f_0(\emptyset, \vec{s}, \vec{t}, x) &= [\![\langle M_0, s \sqcup \{x\}\rangle]\!]_{\neq s \sqcup \{x\}}\,(\vec{s}, x), & f_0(\{r\}, \vec{s}, \vec{t}, x) &= [\![M_0]\!]_{\neq s \sqcup t \sqcup \{x\}}\,(\vec{s}, r + \vec{t}, x) \end{aligned}$$

whenever $(\vec{s}, r + \vec{t}) \in \mathbb{R}^{\neq s \sqcup t}$ and $(\vec{s}, r + \vec{t}, x) \in \mathbb{R}^{\neq s \sqcup t \sqcup \{x\}}$. Writing $t = (t_1, \ldots, t_k)$, we define $g : 2^{\mathbb{R}} \times \mathbb{R}^{\neq s \sqcup t} \to [\![\sigma]\!]$ by

$$g(B, \vec{s}, \vec{t}) = \lambda x. \begin{cases} f_x(B, \vec{s}, \vec{t}) & \text{if } x \in \vec{s}, \\ [\![M_{t_1}]\!]\,(\vec{s}, (x - t_1) + \vec{t}) & \text{else if } x \in B + t_1, \\ \ldots & \\ [\![M_{t_k}]\!]\,(\vec{s}, (x - t_k) + \vec{t}) & \text{else if } x \in B + t_k, \\ f_0(B, \vec{s}, \vec{t}, x) & \text{otherwise.} \end{cases}$$

Note that g is quasi-Borel because the group structure is compatible with the quasi-Borel structure of $\mathbb{R}$.

We now verify that

$$g(\emptyset, \vec{s}, \vec{t}) = \lambda x. \left\{ \begin{array}{ll} [\![\langle M_x, s\rangle]\!]_{\neq s}\,(\vec{s}) & \text{if } x \in \vec{s}, \\ [\![\langle M_0, s \sqcup \{x\}\rangle]\!]_{\neq s \sqcup \{x\}}\,(\vec{s}, x) & \text{otherwise} \end{array} \right\} = |\langle C, s\rangle|\,(\vec{s})$$

and

$$g(\{r\}, \vec{s}, \vec{t}) = \lambda x. \left\{ \begin{array}{ll} [\![M_x]\!]_{\neq s \sqcup t}\,(\vec{s}, r + \vec{t}) & \text{if } x \in \vec{s}, \\ [\![M_x]\!]_{\neq s \sqcup t}\,(\vec{s}, r + \vec{t}) & \text{if } x \in r + \vec{t}, \\ [\![M_0]\!]_{\neq s \sqcup t \sqcup \{x\}}\,(\vec{s}, r + \vec{t}, x) & \text{otherwise} \end{array} \right\} = |C|(\vec{s}, r + \vec{t})$$

whenever $(\vec{s}, r + \vec{t}) \in \mathbb{R}^{\neq s \sqcup t}$. Therefore $f = [g]$ is the function we want.

Expression case: Suppose that we have constructed these reductions for canonical terms of type τ. We have $M\,(Id_s)_\sigma\,M$, so by definition of logical relations and the normal form there is some $C \in \mathrm{Can}_\tau(s \sqcup t \sqcup u \sqcup w)$ such that $s \sqcup t \vdash M \Downarrow_\tau (u \sqcup w)C$ and u is minimal such that $C\,(Id_{s \oplus u})_\tau\,C$, and we have defined $\langle M, s\rangle = \nu u.\,\langle C, s \sqcup u\rangle$.

By our inductive hypothesis, there is a function

$$f_C : 2^{\mathbb{R}} \times \mathbb{R}^{\neq s \sqcup t \sqcup u \sqcup w} \to P(\llbracket \sigma \rrbracket)$$

such that

$$\begin{aligned} f_C(\emptyset, \vec{s}, \vec{t}, \vec{u}, \vec{w}) &= \llbracket \langle C, s \sqcup u\rangle \rrbracket_{\neq s \sqcup u}\,(\vec{s}, \vec{u}), \\ f_C(\{r\}, \vec{s}, \vec{t}, \vec{u}, \vec{w}) &= \llbracket C \rrbracket_{\neq s \sqcup t \sqcup u \sqcup w}\,(\vec{s}, r + \vec{t}, \vec{u}, r + \vec{w}) \end{aligned}$$

whenever $(\vec{s}, r + \vec{t}, \vec{u}, r + \vec{w}) \in \mathbb{R}^{\neq s \sqcup t \sqcup u \sqcup w}$. We then define $f : 2^{\mathbb{R}} \times \mathbb{R}^{\neq s \sqcup t} \to P(\llbracket \sigma \rrbracket)$ by

$$f(B, \vec{s}, \vec{t}) = \iint f_C(B, \vec{s}, \vec{t}, \vec{u}, \vec{w}) d\nu(\vec{u}) d\nu(\vec{w}).$$

Because the group structure is measure preserving and by soundness (Proposition 4.33 and Theorem 3.16) we have

$$\begin{aligned} f(\{r\}, \vec{s}, \vec{t}) &= \iint \llbracket C \rrbracket_{\neq s \sqcup t \sqcup u \sqcup w}\,(\vec{s}, r + \vec{t}, \vec{u}, r + \vec{w}) d\nu(\vec{u}) d\nu(\vec{w}) \\ &= \iint \llbracket C \rrbracket_{\neq s \sqcup t \sqcup u \sqcup w}\,(\vec{s}, r + \vec{t}, \vec{u}, \vec{w}) d\nu(\vec{u}) d\nu(\vec{w}) \\ &= \llbracket \nu u.\nu w.C \rrbracket_{\neq s \sqcup t}\,(\vec{s}, r + \vec{t}) \\ &= \llbracket M \rrbracket_{\neq s \sqcup t}\,(\vec{s}, r + \vec{t}) \end{aligned}$$

whenever $(\vec{s}, r + \vec{t}) \in \mathbb{R}^{\neq s \sqcup t}$. Similarly, we have $f(\emptyset, \vec{s}, \vec{t}) = \llbracket \langle M, s\rangle \rrbracket_{\neq s}\,(\vec{s})$, as desired. □

7 Conclusion

We have shown that quasi-Borel spaces provide an abstract categorical model of the nu-calculus at first-order types. To do this, we have constructed a novel normal form for nu-calculus terms at first-order types. We also explored higher-order quasi-Borel spaces using results of descriptive set theory.

Our work suggests that there is a concrete connection between fresh name generation and higher-order probability theory. An interesting direction for further research would be to explore the implications of this connection, both for the nu-calculus and for probabilistic programming. For example, one may ask how a ν-invariant group structure on the set of names should be interpreted in terms of freshness and privacy. One may also ask if the nu-calculus can be soundly interpreted in all models of higher-order probabilistic programming featuring continuous measures, and at what levels of abstractness.

Our work also shows that there is a connection between quasi-Borel spaces and descriptive set theory, suggesting that a set-theoretic analysis of other models of higher-order probability theory extending the standard Borel spaces may be fruitful.

Bibliography

[Aum61] Robert J. Aumann. "Borel structures for function spaces". In: *Illinois Journal of Mathematics* 5.4 (Dec. 1961), pp. 614–630. DOI: 10.1215/ijm/1255631584 (cit. on pp. 1, 23, 24).

[Gir82] Michèle Giry. "A categorical approach to probability theory". In: *Lecture Notes in Mathematics*. Springer Berlin Heidelberg, 1982, pp. 68–85. DOI: 10.1007/bfb0092872 (cit. on p. 25).

[Heu+17] Chris Heunen et al. "A convenient category for higher-order probability theory". In: *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. IEEE, June 2017. DOI: 10.1109/lics.2017.8005137 (cit. on pp. iii, iv, 1, 29–31, 33, 36).

[Kec95] Alexander S. Kechris. *Classical Descriptive Set Theory*. Springer New York, 1995. DOI: 10.1007/978-1-4612-4190-4 (cit. on pp. 25, 26, 28).

[Mog91] Eugenio Moggi. "Notions of computation and monads". In: *Information and Computation* 93.1 (July 1991), pp. 55–92. DOI: 10.1016/0890-5401(91)90052-4 (cit. on pp. 10, 15, 16).

[PS93] Andrew M. Pitts and Ian Stark. "Observable Properties of Higher Order Functions that Dynamically Create Local Names, or: What's *new?*" In: *Mathematical Foundations of Computer Science: Proceedings of the 18th International Symposium MFCS '93*. Lecture Notes in Computer Science 711. Springer-Verlag, 1993, pp. 122–141 (cit. on pp. iii, iv, 1, 3, 5, 7–9).

[Sab+20] Marcin Sabok et al. *Probabilistic Programming Semantics for Name Generation*. 2020. arXiv: 2007.08638 [cs.PL] (cit. on p. vi).

[Sta94] Ian Stark. "Names and Higher-Order Functions". PhD thesis. University of Cambridge, Dec. 1994. DOI: 10/cgnm (cit. on p. 12).

[Sta96] Ian Stark. "Categorical Models for Local Names". In: *LISP and Symbolic Computation* 9.1 (Feb. 1996), pp. 77–107 (cit. on pp. 7, 8, 10, 12, 15, 20–22).