Logical Relations for Dynamic Name Creation *

Yu Zhang and David Nowak

LSV, CNRS UMR 8643, ENS de Cachan
94235 Cachan Cedex, France
{zhang,nowak}@lsv.ens-cachan.fr

Abstract. Pitts and Stark's nu-calculus is a typed lambda-calculus which forms a basis for the study of interaction between higher-order functions and dynamically created names. A similar approach has received renewed attention recently through Sumii and Pierce's cryptographic lambda-calculus, which deals with security protocols. *Logical relations* are a powerful tool to prove properties of such a calculus, notably observational equivalence. While Pitts and Stark construct a logical relation for the nu-calculus, it rests heavily on operational aspects of the calculus and is hard to be extended. We propose an alternative Kripke logical relation for the nu-calculus, which is derived naturally from the categorical model of the nu-calculus and the general notion of Kripke logical relation. This is also related to the Kripke logical relation for the name creation monad by Goubault-Larrecq et al. (CSL'2002), which the authors claimed had similarities with Pitts and Stark's logical relation. We show that their Kripke logical relation for names is strictly weaker than Pitts and Stark's. We also show that our Kripke logical relation, which extends the definition of Goubault-Larrecq et al., is equivalent to Pitts and Stark's up to first-order types; our definition rests on purely semantic constituents, and dispenses with the detours through operational semantics that Pitts and Stark use.
Keywords: Kripke logical relation, name creation, nu-calculus, categorical models of lambda calculi

1 Introduction

The *nu-calculus* of Pitts and Stark is a typed lambda-calculus, extended with state in the form of dynamically created names. Through the interaction between names and functions, the language can capture notions of scope, visibility and sharing. It has also connections to local declarations in general; to the mobile processes of the π-calculus; to security protocols in the spi-calculus; and particularly, to the cryptographic lambda-calculus of Sumii and Pierce [15]. When we regard dynamically created names as dynamically created keys, the nu-calculus is indeed a syntactical subset of the cryptographic lambda-calculus. It contains a primitive for key generation but lacks those for encryption and decryption.

* Partially supported by the RNTL project EVA, the ACI jeunes chercheurs "Sécurité informatique, protocoles cryptographiques et détection d'intrusions" and the ACI Cryptologie "PSI-Robuste".

Logical relations are a powerful technique to prove properties of typed lambda calculi such like the nu-calculus and the cryptographic lambda-calculus. In [11, 12], Pitts and Stark define an operational logical relation to establish the observational equivalence between nu-calculus expressions. They also prove that this logical relation is complete for first-order types. But the operational logical relation is a quite syntactic one and it is defined based on some operational semantics of the nu-calculus, which makes it hard to be extended.

We propose another logical relation for the nu-calculus in a natural way. This logical relation follows the general notion of Kripke logical relations [5] and is derived from a denotational model of the nu-calculus, which is a functor category with a strong monad. It is also related to the Kripke logical relation for the name creation monad by Goubault-Larrecq, Lasota and Nowak [3]. The latter claims to have similarities with Pitts and Stark's operational logical relation, while we shall show that it is indeed strictly weaker than Pitts and Stark's. We extend Goubault-Larrecq, Lasota and Nowak's definition so as to derive naturally our Kripke logical relation in a categorical style. We also show that, this logical relation is equivalent to Pitts and Stark's operational logical relation up to first-order types.

As preliminaries, Section 2 gives a brief review of the nu-calculus and the operational logical relation, more details of which can be found in [11, 12, 14]. Section 3 introduces a categorical model $\boldsymbol{Set}^{\mathcal{I}}$ for the nu-calculus [12, 13] to provide a basis for our Kripke logical relation. We give in Section 4 the definition of the Kripke logical relation, while before that, we derive another one from Goubault-Larrecq, Lasota and Nowak's definition for name creation monad in [3] and show that this one is strictly weaker than Pitts and Stark's operational logical relation. At the end of Section 4, we show that ours is equivalent to Pitts and Stark's up to first-order types.

2 The nu-calculus and the operational logical relation

2.1 The nu-calculus

The nu-calculus is a small language designed to show the interaction between dynamically created names and higher-order functions [11, 12]. It is a typed call-by-value lambda-calculus extended with the notion of a *name*; names have their own type ν, they can be created fresh, passed around and tested for equality. Higher-order functions and booleans are also available, but to ensure termination functions cannot be defined recursively. New names are created in expressions of the form $\nu n.M$, which binds a fresh name to the identifier n and then evaluates M. The use of a call-by-value semantics means that although only the expression M can refer to n explicitly, the new name itself may escape from this scope. For example, $(\lambda x^{\nu}.x = x)(\nu n.n)$ evaluates to $true$, with x bound to a name rather than to $\nu n.n$ itself.

The syntax of the nu-calculus is based on the simply-typed lambda-calculus. Types take the form

$$\sigma ::= o \mid \nu \mid \sigma \to \sigma$$

where o and ν are two ground types denoting *booleans* and *names* respectively. Terms take the form

$$
\begin{array}{rll}
M ::= & x & \text{variable} \\
| & n & \text{name} \\
| & true \mid false & \text{boolean values} \\
| & \mathit{if}\ M\ \mathit{then}\ M\ \mathit{else}\ M & \text{conditional} \\
| & M = M & \text{equality of names} \\
| & \nu n.M & \text{local name declaration} \\
| & \lambda x^{\sigma}.M & \text{function abstraction} \\
| & MM & \text{function application}
\end{array}
$$

where $x \in Var$, an infinite set whose elements are called *variables*, and $n \in Nme$, an infinite set (disjoint from Var) whose elements are called *names*. Function abstraction is a variable-binding construct (occurrences of x in M are bound in $\lambda x^{\sigma}.M$), whereas local name declaration is a name-binding construct (occurrences of n in M are bound in $\nu n.M$). We implicitly identify expressions which only differ in their choice of bound variables and names (α-conversion).

An expression is *closed* if it has no free variables; a closed expression may still have free names. We denote by $M[M'/x]$ (respectively $M[M'/n]$) the result of substituting the expression M' for free occurrences of the variable x (respectively, the name n) in the expression M. The substitution is *capture avoiding*. This can always be arranged by α-converting M.

Expressions will be assigned types via typing assertions of the form

$$s, \Gamma \vdash M : \sigma$$

where s is a finite subset of Nme, Γ is a finite set of typed variables, σ is a type, and M is an expression (more precisely, an α-equivalence class of expressions) with free names in s and free variables in Γ.

An expression is in *canonical form* if it is either a name, a variable, one of the boolean constants $true$ or $false$, or a function abstraction. These are the *values* of the nu-calculus. As what Pitts and Stark do in [11, 12], we define the sets

$$
\begin{aligned}
\mathrm{Exp}_{\sigma}(s, \Gamma) &= \{M \mid s, \Gamma \vdash M : \sigma\} \\
\mathrm{Can}_{\sigma}(s, \Gamma) &= \{C \mid C \in \mathrm{Exp}_{\sigma}(s, \Gamma), C \text{ canonical}\} \\
\mathrm{Exp}_{\sigma}(s) &= \mathrm{Exp}_{\sigma}(s, \emptyset) \\
\mathrm{Can}_{\sigma}(s) &= \mathrm{Can}_{\sigma}(s, \emptyset)
\end{aligned}
$$

of expressions and canonical expressions at any type σ and for any finite sets s, Γ of names and typed variables.

The operational semantics of the nu-calculus is specified in [11, 12] by an inductively defined evaluation relation. Elements of the relation take the form

$$s \vdash M \Downarrow_{\sigma}^{s'} C$$

where s and s' are disjoint, $M \in \mathrm{Exp}_\sigma(s)$ and $C \in \mathrm{Can}_\sigma(s \oplus s')$. This is intended to mean that in the presence of the names s, expression M of type σ evaluates to canonical form C while creating the fresh names in s' in the process. The sets s and $s \oplus s'$ can be seen as initial and final *states* of the computation. See [11, 12] for details of the evaluation relation.

2.2 The operational logical relation

An operational logical relation between nu-calculus expressions is given by Pitts and Stark in [11, 12]. It rests on the operational semantics of the nu-calculus and a certain kind of relation between sets of names.

Definition 1 (Spans). If s_1 and s_2 are sets of names, then a *span* $R : s_1 \rightleftharpoons s_2$ is an injective partial map from s_1 to s_2. That is, the graph $R \subseteq s_1 \times s_2$ satisfies

$$(n_1 \ R \ n_2) \ \& \ (n_1' \ R \ n_2') \Longrightarrow (n_1 = n_1' \Longleftrightarrow n_2 = n_2').$$

A span can also be represented as a pair of injections $s_1 \leftarrowtail R \rightarrowtail s_2$.

If $R' : s_1' \rightleftharpoons s_2'$ is another span, with s_1' and s_2' disjoint from s_1 and s_2 respectively, then the disjoint union of R and R' is also a span:

$$R \oplus R' : s_1 \oplus s_1' \rightleftharpoons s_2 \oplus s_2'.$$

The domain and co-domain of definition for $R : s_1 \rightleftharpoons s_2$ are defined by :

$$\begin{aligned} \mathrm{dom}(R) &= \{n_1 \in s_1 \mid \exists n_2 \in s_2 \ (n_1 \ R \ n_2)\} \\ \mathrm{cod}(R) &= \{n_2 \in s_2 \mid \exists n_1 \in s_1 \ (n_1 \ R \ n_2)\} \end{aligned}$$

The intuition behind spans is that they should capture the use that nu-calculus expressions make of public and private names. So if $R : s_1 \rightleftharpoons s_2$, then the bijection between $\mathrm{dom}(R) \subseteq s_1$ and $\mathrm{cod}(R) \subseteq s_2$ represents matching use of "visible" names, while the remaining elements not in the graph of R are "unseen" names.

Definition 2 (Operational logical relation). If $R : s_1 \rightleftharpoons s_2$ is a span then the relations

$$\begin{aligned} R_\sigma^{can} &\subseteq \mathrm{Can}_\sigma(s_1) \times \mathrm{Can}_\sigma(s_2) \\ R_\sigma^{exp} &\subseteq \mathrm{Exp}_\sigma(s_1) \times \mathrm{Exp}_\sigma(s_2) \end{aligned}$$

are defined by induction over the structure of the type σ, according to:

$$\begin{aligned} b_1 \ R_o^{can} \ b_2 &\Longleftrightarrow b_1 = b_2 \\ n_1 \ R_\nu^{can} \ n_2 &\Longleftrightarrow n_1 \ R \ n_2 \end{aligned}$$

$$\begin{aligned} &(\lambda x^\sigma.M_1) \ R_{\sigma \to \sigma'}^{can} \ (\lambda x^\sigma.M_2) \Longleftrightarrow \\ &\qquad \forall R' : s_1' \rightleftharpoons s_2', C_1 \in Can_\sigma(s_1 \oplus s_1'), C_2 \in Can_\sigma(s_2 \oplus s_2') \end{aligned}$$

$$(C_1\ (R \oplus R')^{can}_{\sigma}\ C_2 \Longrightarrow M_1[C_1/x]\ (R \oplus R')^{exp}_{\sigma'}\ M_2[C_2/x])$$

$$\begin{aligned}
M_1\ R^{exp}_{\sigma}\ M_2 \iff& \\
&\exists R' : s'_1 \rightleftharpoons s'_2, C_1 \in Can_{\sigma}(s_1 \oplus s'_1), C_2 \in Can_{\sigma}(s_2 \oplus s'_2) \\
&(s_1 \vdash M_1 \Downarrow^{s'_1}_{\sigma} C_1\ \&\ s_2 \vdash M_2 \Downarrow^{s'_2}_{\sigma} C_2\ \&\ C_1\ (R \oplus R')^{can}_{\sigma}\ C_2)
\end{aligned}$$

The relation R^{can}_{σ} and R^{exp}_{σ} coincide on canonical forms, and we may write them as R_{σ} indiscriminately.

This operational logical relation can be used to establish observational equivalences in the nu-calculus. Consider the two nu-calculus expressions of type $\nu \to o$: $\nu n.\lambda x^{\nu}.(x = n)$ and $\lambda x^{\nu}.false$. According to the evaluation relation in [11, 12],

$$s, \Gamma \vdash \nu n.\lambda x^{\nu}.(x = n) \Downarrow^{\{n\}}_{\sigma} \lambda x^{\nu}.(x = n)$$

and

$$s, \Gamma \vdash \lambda x^{\nu}.false \Downarrow^{\emptyset}_{\sigma} \lambda x^{\nu}.false.$$

The first term creats a new name, which is definitely private to the function, and then compare it with the argument. Since no external context can supply this name, the function always returns the value $false$, so it is observationally equivalent to the second term. The only possible span for new names is the empty relation $R' : \{n\} \rightleftharpoons \emptyset$. Whatever the span $R : s_1 \rightleftharpoons s_2$ is, we have

$$\lambda x^{\nu}.(x = n)\ (R \oplus R')^{can}_{\nu \to o}\ \lambda x^{\nu}.false$$

and consequently, the above two expressions are related.

3 A categorical model for the nu-calculus

As the definition of R^{exp}_{σ} shows, the operational logical relation relies heavily on the operational semantics of the nu-calculus. Moreover this definition silently assumes that each term has a canonical form — it is indeed the case in nu-calculus. This prevents one from extending this relation to non-terminating calculi with names. This section introduces the categorical model $\boldsymbol{Set}^{\mathcal{I}}$ defined by Stark in [12, 13] to give the denotational semantics for the nu-calculus. This model is a functor category equipped with a strong monad and it provides us not only a denotational model for the behavior of nu-calculus expressions, but also a sound basis for a logical relation to be defined next.

3.1 Categorical model

Let $\mathcal{I}$ be the category of finite sets and injective functions. Let $\boldsymbol{C}$ be the category $\boldsymbol{Set}^{\mathcal{I}}$ of functors from $\mathcal{I}$ to $\boldsymbol{Set}$ and natural transformations. For short, we write $\boldsymbol{T}As$ for $\boldsymbol{T}(A)(s)$ and similarly for other notations. Let $+$ denote disjoint union in $\mathcal{I}$.[1]

We consider the strong monad $(\boldsymbol{T}, \boldsymbol{\eta}, \boldsymbol{\mu}, \boldsymbol{t})$ on $\boldsymbol{C}$ defined in [12, 13, 3]:

[1] Note that $+$ is not a coproduct in $\mathcal{I}$ which in fact does not have a coproduct. However $+$ is functorial in both components, associative, and has a neutral element.

- $\boldsymbol{T}A = \mathrm{colim}_{s'} A(\_ + s') : \mathcal{I} \to \boldsymbol{Set}$. Explicitly:
  - $\boldsymbol{T}As$ is the set of equivalence classes of pairs (s', a), where $s' \in \mathcal{I}$ and $a \in A(s+s')$, modulo the smallest equivalence relation $\equiv_{\boldsymbol{C}}$ such that $(s', a) \equiv_{\boldsymbol{C}} (s'', A(\mathrm{id}_s + i)a)$ for each morphism $i : s' \to s''$ in $\mathcal{I}$. We write $[s, a]$ for the equivalence class of (s, a).
  - On morphism i, $\boldsymbol{T}Ai$ maps the equivalence class of (s', a) to the equivalence class of $(s', A(i + \mathrm{id}_{s'})a)$.
- For any $f : F \to G$ in $\boldsymbol{C}$, $\boldsymbol{T}fs : \boldsymbol{T}Fs \to \boldsymbol{T}Gs$ is defined by $\boldsymbol{T}fs[s', a] = [s', f(s+s')a]$. This is compatible with $\equiv_{\boldsymbol{C}}$ because f is natural.
- $\boldsymbol{\eta}_A s : As \to \boldsymbol{T}As$ is defined by $\boldsymbol{\eta}_A sa = [\emptyset, a]$.
- $\boldsymbol{\mu}_A s : \boldsymbol{T}^2 As \to \boldsymbol{T}As$ is defined by $\boldsymbol{\mu}_A s[s', [s'', a]] = [s' + s'', a]$
- $\boldsymbol{t}_{A_1,A_2} s : A_1 s \times \boldsymbol{T}A_2 s \to \boldsymbol{T}(A_1 \times A_2)s$ is defined by $\boldsymbol{t}_{A_1,A_2} s(a, [s', b]) = [s', (A_1 i_{s,s'} a, b)]$ where $i_{s,s'} : s \to s + s'$ is the canonical injection.

It is standard that this category is cartesian closed. If $F, G : \mathcal{I} \to \boldsymbol{Set}$ are functors then their exponent is defined by the standard construction in covariant presheaves [4]:

$$G^F s = \boldsymbol{Set}^{\mathcal{I}}(\mathcal{I}(s, -) \times F, G)$$
$$G^F ifs''\langle j, a''\rangle = fs''\langle j \circ i, a''\rangle,$$

where $s, s', s'', i : s \to s', j : s' \to s'' \in \mathcal{I}$ and $f \in G^F s, a'' \in Fs''$.

3.2 The interpretation of the nu-calculus

The interpretation of the nu-calculus is based on a computational metalanguage [12, 13], which stems from Moggi's computational lambda calculus [8–10]. In this particular metalanguage, besides the usual boolean type, name type and function types, we have a unary type constructor T: if A is a type, then elements of TA are *computations* of type A. The difference between a value and a computation is that a computation may generate new names before returning a value. A nu-calculus expression is regarded as a computation in the metalanguage which generates a set of names and then returns a value, and an expression in canonical form simply returns a value without generating any name.

The category $\boldsymbol{Set}^{\mathcal{I}}$ is qualified to be a categorical model of this metalanguage [12, 13]. In this category, objects of $\mathcal{I}$ represent *stages of computation*, i.e., what names have been created. For a functor $A : \mathcal{I} \to \boldsymbol{Set}$, the set As is composed of values defined over the names in s. Morphisms in $\mathcal{I}$ and their images in $\boldsymbol{Set}$ correspond to name substitutions.

We begin by defining two functors B and N from $\mathcal{I}$ to $\boldsymbol{Set}$. Functor B is defined by:

$$\forall s \in \mathcal{I}, Bs = Bool \ \ \& \ \ \forall i : s \to s' \in \mathcal{I}, Bi = \mathrm{id}_{Bool},$$

and functor N:

$$\forall s \in \mathcal{I}, Ns = s \ \ \& \ \ \forall i : s \to s' \in \mathcal{I}, Ni = i,$$

where *Bool* is a set containing only two elements — *tt* and *ff*. Types in the nu-calculus are interpreted as objects of $\boldsymbol{Set}^{\mathcal{I}}$:

$$\begin{aligned} [\![o]\!] &= B \\ [\![\nu]\!] &= N \\ [\![\sigma \to \sigma']\!] &= (\boldsymbol{T}[\![\sigma']\!])^{[\![\sigma]\!]}. \end{aligned}$$

According to what Stark does in [12, 13], for each valid type assertion $s, \Gamma \vdash M : \sigma$ in the nu-calculus, the semantics of M in the categorical model is given with an index s:

$$[\![M]\!]_{\Gamma,s} \in (\boldsymbol{T}[\![\sigma]\!])^{[\![\Gamma]\!]}s \quad \text{where} \quad [\![\Gamma]\!] = \prod_{x_i:\sigma_i \in \Gamma} [\![\sigma_i]\!].$$

If M is a closed term, i.e., $M \in \mathrm{Exp}_\sigma(s)$, then $[\![M]\!]_s \in \boldsymbol{T}[\![\sigma]\!]s$, identifying A^1 with A, where 1 is the terminal object in $\boldsymbol{Set}^{\mathcal{I}}$ and $1s$ is a singleton for all s. The semantics of a closed canonical expression $[\![C]\!]_s$ is always of the form $[\emptyset, |C|_s]$, an element of the set $\boldsymbol{T}[\![\sigma]\!]s$. Here $|C|_s \in [\![\sigma]\!]s$ and $|-|_s$ is the canonical interpretation defined for each closed canonical form. If $s \vdash M \Downarrow_\sigma^{s'} C$, where $M \in \mathrm{Exp}_\sigma(s)$ and $C \in \mathrm{Can}_\sigma(s)$, the semantics of the expression M can also be given by the canonical interpretation of the corresponding canonical form C:

$$[\![M]\!]_s = [\![\nu s'.C]\!]_s = [s', |C|_{s+s'}] \;\; \in \boldsymbol{T}[\![\sigma]\!]_s. \tag{1}$$

A definability result for the model $\boldsymbol{Set}^{\mathcal{I}}$ is also given in [12].

Lemma 1 (Definability). *Suppose that σ is a ground or first-order type of the nu-calculus and s is some set of names. If $a \in [\![\sigma]\!]s$ and $e \in \boldsymbol{T}[\![\sigma]\!]s$, then there are expressions $C \in \mathrm{Can}_\sigma(s)$ and $M \in \mathrm{Exp}_\sigma(s)$ such that $a = |C|_s$ and $e = [\![M]\!]_s$.* □

4 The Kripke logical relation

4.1 A weak logical relation

We try to define a logical relation on the categorical model $\boldsymbol{Set}^{\mathcal{I}}$ so that it does not depend on any syntactical aspect of the nu-calculus. The general Kripke logical relation [5] gives a natural way to construct such a logical relation. A Kripke logical relation is based on a Kripke lambda models [5, 6] which includes a partially-ordered set of *possible worlds* and have a set of elements of each type at each possible world. Informally, the relationship between elements of type σ at different worlds is that every element of σ at w will continue to be an element of σ at every possible $w' \geq w$.

Strictly speaking, a Kripke logical relation is indexed by elements of a poset. It has a relation for each type and a *possible world* instead of just having a relation for each type [5]. This can be naturally extended to be indexed by objects of a category as argued in [1]. The category $\boldsymbol{Set}^{\mathcal{I}}$ in the last section just defines a Kripke model. We take the sets in $\mathcal{I}$ as the possible worlds and set inclusion as the partial order between them.

In [7], a categorical definition of logical relations based on subsconing was proposed. This definition was extended in [3] in order to define a notion of logical relation for monadic types. According to this extended logical relation, we define a Kripke logical relation $\mathbb{R}^s_A \subseteq As \times As$ in the category $\boldsymbol{Set}^{\mathcal{I}}$ by induction over the structure of the corresponding type of A in a metalanguage for $\boldsymbol{Set}^{\mathcal{I}}$ [8–10] (here we do not distinguish on notation between an object of a category and the corresponding type in the metalanguage). Notably, we specify the types corresponding to the objects B and N as two constant types:

$$\begin{aligned}
b_1 \ \mathbb{R}^s_B \ b_2 &\Longleftrightarrow b_1 = b_2, \\
n_1 \ \mathbb{R}^s_N \ n_2 &\Longleftrightarrow n_1 = n_2, \\
f_1 \ \mathbb{R}^s_{G^F} \ f_2 &\Longleftrightarrow \forall s', i : s \to s' \in \mathcal{I}, a_1, a_2 \in Fs' \\
&\qquad (a_1 \ \mathbb{R}^{s'}_F \ a_2 \Longrightarrow f_1 s'\langle i, a_1\rangle \ \mathbb{R}^{s'}_G \ f_2 s'\langle i, a_2\rangle), \\
[s_1, a_1] \ \mathbb{R}^s_{\boldsymbol{T}A} \ [s_2, a_2] &\Longleftrightarrow \exists s_0, i_1 : s_1 \to s_0, i_2 : s_2 \to s_0 \in \mathcal{I} \\
&\qquad A(\mathrm{id}_s + i_1)a_1 \ \mathbb{R}^{s+s_0}_A \ A(\mathrm{id}_s + i_2)a_2.
\end{aligned}$$

To define the above logical relation, we do not need any operational semantics for the nu-calculus such as Pitts and Stark define in [11, 12]. And since it is defined on the pure categorical model $\boldsymbol{Set}^{\mathcal{I}}$, it can be easily generalized to study other typed lambda-calculi which $\boldsymbol{Set}^{\mathcal{I}}$ is suitable to model.

However, there is a small flaw in this logical relation. With the *span* in Pitts and Stark's operational logical relation, we can distinguish between public and private names while we cannot in the above Kripke logical relation. In fact, the above logical relation is strictly weaker than the operational one. Consider the two expressions which we have already shown to be related by the operational logical relation:

$$\nu n.\lambda x^\nu.(x = n) \ R^{exp}_{\nu \to o} \ \lambda x^\nu.false.$$

By (1), their semantics are

$$\begin{aligned}
[\![\nu n.\lambda x^\nu.(x = n)]\!]_s &= [\{n\}, |\lambda x^\nu.(x = n)|_s] \in \boldsymbol{T}((\boldsymbol{T}B)^N)s \\
[\![\lambda x^\nu.false]\!]_s &= [\emptyset, |\lambda x^\nu.false|_s] \in \boldsymbol{T}((\boldsymbol{T}B)^N)s.
\end{aligned}$$

If these two terms can be related by the above Kripke logical relation, then by the definition of $\mathbb{R}^s_{\boldsymbol{T}A}$ with $A = (\boldsymbol{T}B)^N$ we get

$$(\boldsymbol{T}B)^N(\mathrm{id}_s + i_1)|\lambda x^\nu.(x = n)|_{s+s_0} \ \mathbb{R}^{s+s_0}_{N \to \boldsymbol{T}B} \ (\boldsymbol{T}B)^N(\mathrm{id}_s + i_2)|\lambda x^\nu.false|_{s+s_0}$$

for some $s_0 \in \mathcal{I}$ and $i_1 : \{n\} \to s_0, i_2 : \emptyset \to s_0 \in \mathcal{I}$. Without losing the universality, we just let $s_0 = \{n\}, i_1 = \mathrm{id}_{\{n\}}$, and then get

$$|\lambda x^\nu.(x = n)|_{s+\{n\}} \ \mathbb{R}^{s+\{n\}}_{N \to \boldsymbol{T}B} \ |\lambda x^\nu.false|_{s+\{n\}}.$$

Now applying n to both functions, we get consequently $tt \ \mathbb{R}^{s+\{n\}}_B \ ff$, which is a contradiction to the definition of the relation for boolean type.

A direct way to represent this identification of public names is to provide the set of public names in the logical relation. We modify the structure of possible worlds by adding a name set w so as to make the pair $\langle w, s\rangle$ instead of s as the possible world. Intuitively, the new set w stands for the set of public names of two nu-calculus expressions, so we require $w \subseteq s$. The partial order on the modified possible worlds is given by

$$\langle w_1, s_1\rangle \leq \langle w_2, s_2\rangle \iff w_1 \subseteq w_2 \;\;\&\;\; s_1 \subseteq s_2,$$

and we change the relation for name type:

$$n_1 \; \mathbb{R}_N^{\langle w,s\rangle} \; n_2 \iff n_1 = n_2 \;\;\&\;\; n_1, n_2 \in w.$$

The intuition behind this relation is that two names are related if and only if they are equal and they are both public names.

4.2 Observation category

Whereas in [3] the chosen observation category was the categorical model $\boldsymbol{Set}^{\mathcal{I}}$ itself, here we refine the observation category in order to be able to derive a stronger logical relation.

Consider the comma category $\mathcal{I}^{\rightarrow}$, whose objects are tuples $\langle w, i, s\rangle$ with $i : w \to s$ in $\mathcal{I}$, and whose morphisms are pairs $\langle j, k\rangle : \langle w, i, s\rangle \to \langle w', i', s'\rangle$ where $j : w \to w'$ in $\mathcal{I}$ and $k : s \to s'$ in $\mathcal{I}$ such that the following diagram commutes:

$$\begin{tikzcd} w \arrow[r, "i"] \arrow[d, "j"'] & s \arrow[d, "k"] \\ w' \arrow[r, "i'"'] & s' \end{tikzcd}$$

We write i for $\langle w, i, s\rangle$ when the domain w and the codomain s of i are clear from context.

Let $\mathbb{C}$ be the category $\boldsymbol{Set}^{\mathcal{I}^{\rightarrow}}$ of functors from $\mathcal{I}^{\rightarrow}$ to $\boldsymbol{Set}$ and natural transformations. We define a strong monad (T, η, μ, t) on $\mathbb{C}$ by:

- $TA = \mathrm{colim}_{i'}\, A(\_ + i') : \mathcal{I}^{\rightarrow} \to \boldsymbol{Set}$. Explicitly:
  - TAi is the set of equivalence classes of pairs (i', a), where $i' : w' \to s'$ in $\mathcal{I}$ and $a \in A(i + i')$, modulo the smallest equivalence relation $\equiv_{\mathbb{C}}$ such that $(i', a) \equiv_{\mathbb{C}} (i'', A(\mathrm{id}_i + \langle j, k\rangle)a)$ for each morphism $\langle j, k\rangle : \langle w', i', s'\rangle \to \langle w'', i'', s''\rangle$ in $\mathcal{I}^{\rightarrow}$. We write $[i, a]$ for the equivalence class of (i, a).
  - On morphism $\langle j, k\rangle$, $TA\langle j, k\rangle$ maps the equivalence class of (i', a) to the equivalence class of $(i', A(\langle j, k\rangle + \mathrm{id}_{i'})a)$.
- For any $f : A \to B$ in $\mathbb{C}$, $Tfi : TAi \to TBi$ is defined by $Tfi[i', a] = [i', f(i + i')a]$. This is compatible with $\equiv_{\mathbb{C}}$ because f is natural.
- $\eta_A i : Ai \to TAi$ is defined by $\eta_A ia = [\emptyset, a]$.
- $\mu_A i : T^2Ai \to TAi$ is defined by $\mu_A i[i', [i'', a]] = [i' + i'', a]$
- $t_{A,B}i : Ai \times TBi \to T(A \times B)i$ is defined by $t_{A,B}i(a, [i', b]) = [i', (Ai^{\rightarrow}_{i,i'}a, b)]$ where $i^{\rightarrow}_{i,i'} : i \to i + i'$ is the canonical injection.

4.3 Natural derivation of the logical relation

Let $U : \mathcal{I}^{\to} \to \mathcal{I}$ be the forgetful functor which maps an object $\langle w, i, s\rangle$ to s and a morphism $\langle j, k\rangle$ to k. Let $|\_| : \boldsymbol{C} \to \mathbb{C}$ be the functor $\mathrm{id}_{\boldsymbol{Set}}{}^{U}$ (where $\boldsymbol{C} = \boldsymbol{Set}^{\mathcal{I}}$ and $\mathbb{C} = \boldsymbol{Set}^{\mathcal{I}^{\to}}$). On an object A, $|A|$ is equal to $A \circ U$. On a morphism f (which is a natural transformation), for each $\langle w, i, s\rangle$ in $\mathcal{I}^{\to}$, the component $|f|_{\langle w,i,s\rangle}$ is equal to $(f * U)_{\langle w,i,s\rangle}$, that is to say f_s. Let $\sigma : T|\_| \to |\boldsymbol{T}\_|$ be the monad morphism[2] defined by

$$\sigma_A i[\langle w', i', s'\rangle, a] = [s', a]$$

This is well defined as $|A|(i + i') = A(s + s')$. We can then define $\sigma_{\langle A_1, A_2\rangle} : T|A_1 \times A_2| \to |\boldsymbol{T}A_1| \times |\boldsymbol{T}A_2|$ by

$$\sigma_{\langle A_1,A_2\rangle} i = \langle \sigma_{A_1} i \circ T\pi_1, \sigma_{A_2} i \circ T\pi_2\rangle$$

That is to say

$$\sigma_{\langle A_1,A_2\rangle} i[\langle w', i', s'\rangle, (a_1, a_2)] = ([s', a_1], [s', a_2])$$

$\mathbb{C}$ has a mono factorization system consisting of point-wise surjections and point-wise injections. And it is clear that T and finite product preserve point-wise surjections. The conditions required in [3] are therefore satisfied. We thus define the monadic type component of our Kripke logical relation. Each injection $f : S\langle w, i, s\rangle \hookrightarrow |A_1 \times A_2|\langle w, i, s\rangle = A_1 s \times A_2 s$ is a representation of a relation $S\langle w, i, s\rangle \subseteq A_1 s \times A_2 s$ up to isomorphism. The logical relation, denoted $\widetilde{S}$, is given by mono factorization of the composition of Tf with $\sigma_{\langle A_1,A_2\rangle}$:

$$\begin{tikzcd}
TS \arrow[r, "Tf"] \arrow[d, two heads] & T|A_1 \times A_2| \arrow[d, "\sigma_{\langle A_1,A_2\rangle}"] \\
\widetilde{S} \arrow[r, hook] & |\boldsymbol{T}A_1| \times |\boldsymbol{T}A_2|
\end{tikzcd}$$

Explicitly, $\widetilde{S}i \hookrightarrow |\boldsymbol{T}A_1|i \times |\boldsymbol{T}A_2|i$ is given by:

$$\begin{aligned}
&[i_1, a_1] \quad \widetilde{S}i \quad [i_2, a_2] \quad \Longleftrightarrow \\
&\quad \exists i_0 \in \mathcal{I}^{\to} \cdot \exists \langle j_1, k_1\rangle : i_1 \to i_0 \in \mathcal{I}^{\to} \cdot \exists \langle j_2, k_2\rangle : i_2 \to i_0 \in \mathcal{I}^{\to} \cdot \\
&\quad |A_1|(\mathrm{id}_i + \langle j_1, k_1\rangle) \quad S(i + i_0) \quad |A_2|(\mathrm{id}_i + \langle j_2, k_2\rangle)
\end{aligned}$$

It can be simplified in

$$\begin{aligned}
&[i_1, a_1] \quad \widetilde{S}i \quad [i_2, a_2] \quad \Longleftrightarrow \\
&\quad \exists i_0 \in \mathcal{I}^{\to} \cdot \exists \langle j_1, k_1\rangle : i_1 \to i_0 \in \mathcal{I}^{\to} \cdot \exists \langle j_2, k_2\rangle : i_2 \to i_0 \in \mathcal{I}^{\to} \cdot \\
&\quad A_1(\mathrm{id}_s + k_1) \quad S(i + i_0) \quad A_2(\mathrm{id}_s + k_2)
\end{aligned}$$

where s is the codomain of i.

[2] More precisely, the pair $(|\_|, \sigma)$ is the monad morphism from $\boldsymbol{T}$ to T where σ is a natural transformation. σ was named 'distributivity law' in [3]. It turned out to be a bad choice as it could be confused with 'distributive law' which denotes a close but different concept by Beck [2]. Also, while 'distributive laws' tend to be rare, monad morphisms abound.

Lemma 2. *The full subcategory $\mathcal{I}^{\subseteq}$ of $\mathcal{I}^{\rightarrow}$ consisting only of inclusions is equivalent to the whole category $\mathcal{I}^{\rightarrow}$.*

Proof. Let $F : \mathcal{I}^{\subseteq} \rightarrow \mathcal{I}^{\rightarrow}$ be the inclusion functor. Let $G : \mathcal{I}^{\rightarrow} \rightarrow \mathcal{I}^{\subseteq}$ be the functor which maps $\langle w, i, s\rangle$ to $\langle i(w), s\rangle$, and $\langle j, k\rangle : \langle w, i, s\rangle \rightarrow \langle w', i', s'\rangle$ to $i' \circ j \circ i^{-1} : \langle i(w), s\rangle \rightarrow \langle i'(w'), s'\rangle$. Then clearly, $G \circ F$ is identity, and $F \circ G$ maps $\langle w, i, s\rangle$ to $\langle i(w), s\rangle$ which are isomorphic through $\langle i, \mathrm{id}_s\rangle$ and $\langle i^{-1} \upharpoonright_{i(w)}, \mathrm{id}_s\rangle$. So (F, G) is an equivalence of categories. $\square$

Thus the definition of $\widetilde{S}i$ is equivalent to

$$\begin{array}{l} [\langle w_1, s_1\rangle, a_1] \quad \widetilde{S}\langle w, s\rangle \quad [\langle w_2, s_2\rangle, a_2] \quad \iff \\ \quad \exists s_0 \in \mathcal{I} \cdot \exists w_0 \subseteq s_0 \cdot \exists k_1 : s_1 \rightarrow s_0 \in \mathcal{I} \cdot \exists k_2 : s_2 \rightarrow s_0 \in \mathcal{I} \cdot \\ \quad A_1(\mathrm{id}_s + k_1) \quad S(w + w_0 \subseteq s + s_0) \quad A_2(\mathrm{id}_s + k_2). \end{array}$$

Note that a Kripke logical relation indexed by elements of a poset can be naturally extended to be indexed by objects of a category (in this case $\mathcal{I}^{\rightarrow}$) as argued in [1]. We therefore arrive at:

Definition 3 (Kripke logical relation). *The relations $\mathbb{R}_A^{\langle w,s\rangle} \subseteq As \times As$, where $w \subseteq s$ and $A \in \boldsymbol{Set}^{\mathcal{I}}$, are defined by induction over the structure of the corresponding type in the metalanguage, according to:*

$$b_1 \ \mathbb{R}_B^{\langle w,s\rangle} \ b_2 \iff b_1 = b_2$$

$$n_1 \ \mathbb{R}_N^{\langle w,s\rangle} \ n_2 \iff n_1 = n_2 \ \wedge \ n_1, n_2 \in w$$

$$\begin{array}{l} f_1 \ \mathbb{R}_{G^F}^{\langle w,s\rangle} \ f_2 \iff \\ \quad \forall s', \forall k : s \rightarrow s' \in \mathcal{I}, \forall w' \ \textit{s.t.} \ k(w) \subseteq w' \subseteq s', \forall a_1, a_2 \in Fs', \\ \qquad (a_1 \ \mathbb{R}_F^{\langle w',s'\rangle} \ a_2 \Longrightarrow f_1 s'\langle k, a_1\rangle \ \mathbb{R}_G^{\langle w',s'\rangle} \ f_2 s'\langle k, a_2\rangle) \end{array}$$

$$\begin{array}{l} [s_1, a_1] \ \mathbb{R}_{TA}^{\langle w,s\rangle} \ [s_2, a_2] \iff \\ \quad \exists s_0, w_0 \subseteq s_0, i_1 : s_1 \rightarrow s_0, i_2 : s_2 \rightarrow s_0 \in \mathcal{I}, \\ \qquad A(id_s + i_1)a_1 \ \mathbb{R}_A^{\langle w+w_0, s+s_0\rangle} \ A(id_s + i_2)a_2. \quad \square \end{array}$$

Lemma 3. *For all canonical forms $C_1, C_2 \in \mathrm{Can}_\sigma(s)$,*

$$[\![C_1]\!]_s \ \mathbb{R}_{\boldsymbol{T}[\![\sigma]\!]}^{\langle w,s\rangle} \ [\![C_2]\!]_s \iff |C_1|_s \ \mathbb{R}_{[\![\sigma]\!]}^{\langle w,s\rangle} \ |C_2|_s$$

Proof. The interpretation of nu-calculus canonical forms into category $\boldsymbol{Set}^{\mathcal{I}}$ provides us $[\![C_k]\!]_s = [\emptyset, |C_k|_s]$ $(k = 1, 2)$. According to the definition of the Kripke logical relation,

$$[\emptyset, |C_1|_s] \ \mathbb{R}_{\boldsymbol{T}[\![\sigma]\!]}^{\langle w,s\rangle} \ [\emptyset, |C_2|_s] \iff [\![\sigma]\!](\mathrm{id}_s + i_1)|C_1|_s \ \mathbb{R}_{[\![\sigma]\!]}^{\langle w+w_0, s+s_0\rangle} \ [\![\sigma]\!](\mathrm{id}_s + i_2)|C_2|_s$$

for some $s_0, w_0 \subseteq s_0$ and $i_1, i_2 : \emptyset \rightarrow s_0 \in \mathcal{I}$. Since whatever s_0 and w_0 are, i_1 and i_2 are always the empty injection, we simply let s_0, w_0 be the empty set and i_1, i_2 be $\mathrm{id}_\emptyset$. Then we get $[\![\sigma]\!](\mathrm{id}_s + i_k)|C_k|_s = |C_k|_s$, $(k = 1, 2)$, and consequently,

$$[\![C_1]\!]_s \ \mathbb{R}_{\boldsymbol{T}[\![\sigma]\!]}^{\langle w,s\rangle} \ [\![C_2]\!]_s \iff |C_1|_s \ \mathbb{R}_{[\![\sigma]\!]}^{\langle w,s\rangle} \ |C_2|_s. \qquad \square$$

4.4 Equivalence between operational and Kripke logical relations

Through this Kripke logical relation, we can relate the two nu-calculus expressions in Section 4.1 again. To prove that, for $\mathbb{R}_{\boldsymbol{T}A}^{\langle w,s\rangle}$ with $A = (\boldsymbol{T}B)^N$, we let $w_0 = \emptyset$ and by the definition of $\mathbb{R}_{\boldsymbol{T}A}^{\langle w,s\rangle}$, we need to prove

$$|\lambda x^\nu.(x = n)|_{s+\{n\}}\ \mathbb{R}_{(\boldsymbol{T}B)^N}^{\langle w,s+\{n\}\rangle}\ |\lambda x^\nu.false|_{s+\{n\}}.$$

By the definition of $\mathbb{R}_N^{\langle w,s+\{n\}\rangle}$, related names must be in set w and are definitely different from n, so applying any related name to the function $|\lambda x^\nu.(x = n)|_{s+\{n\}}$ must return the value tt (by (1)), which is same as the result of applying it to the second function. Then, by the definition of $\mathbb{R}_{N\to\boldsymbol{T}B}^{\langle w,s+\{n\}\rangle}$ and $\mathbb{R}_B^{\langle w,s+\{n\}\rangle}$, the above two functions can be related.

Theorem 1. *If $R : s_1 \rightleftharpoons s_2$ is a span and $s_1 \vdash M_1 : \sigma$, $s_2 \vdash M_2 : \sigma$ where σ is a ground type or first-order type, then*

$$\begin{aligned} M_1\ R_\sigma\ M_2 \iff &\exists s, w \subseteq s, i_1 : s_1 \to s, i_2 : s_2 \to s \in \mathcal{I}, \\ &\boldsymbol{T}[\![\sigma]\!]i_1([\![M_1]\!]_{s_1})\ \mathbb{R}_{\boldsymbol{T}[\![\sigma]\!]}^{\langle w,s\rangle}\ \boldsymbol{T}[\![\sigma]\!]i_2([\![M_2]\!]_{s_2}). \end{aligned}$$

Proof. If M_1 and M_2 are two expressions in canonical form, by the definition of the category $\boldsymbol{Set}^{\mathcal{I}}$ in Section 3.1,

$$\boldsymbol{T}[\![\sigma]\!]i_k[\![M_1]\!]_{s_k} = \boldsymbol{T}[\![\sigma]\!]i_k[\emptyset, |M_1|_{s_k}] = [\emptyset, [\![\sigma]\!]i_k|M_1|_{s_k}] \qquad (k = 1, 2).$$

Then by Lemma 3, the theorem is equivalent to

$$M_1\ R_\sigma^{can}\ M_2 \iff [\![\sigma]\!]i_1(|M_1|_{s_1})\ \mathbb{R}_{[\![\sigma]\!]}^{\langle w,s\rangle}\ [\![\sigma]\!]i_2(|M_2|_{s_2})$$

for some $w \subseteq s$. So we can made the proof by induction on the structure of types in the metalanguage.

- The case for boolean type is obvious.
- For name type, we should define a proper w and the injections i_1, i_2 to represent the span R in the operational logical relation:

  $$\forall n_1 \in s_1, n_2 \in s_2\ (i_1(n_1) = i_2(n_2) \iff n_1\ R\ n_2)$$

  and w is the image of the domain of R under i_1, equivalently the image of the codomain of R under i_2. Because $Ni_k = i_k$, we have

  $$n_1\ R_\nu^{can}\ n_2 \iff Ni_1(n_1) = Ni_2(n_2) \iff Ni_1(n_1)\ \mathbb{R}_N^{\langle w,s\rangle}\ Ni_2(n_2).$$

- For first-order function types in the metalanguage ($\sigma \equiv \tau \to \tau'$), the only possibility is the nu-calculus canonical expressions of the form $\lambda x^\tau.M$. Suppose the type of M is τ', then the canonical interpretation $|\lambda x^\tau.M|$ is of type $[\![\tau]\!] \to \boldsymbol{T}[\![\tau']\!]$. The definability result of the model $\boldsymbol{Set}^{\mathcal{I}}$ (Lemma 1) ensure that for each $f \in (\boldsymbol{T}[\![\tau']\!])^{[\![\tau]\!]}s$, we can find a canonical function expression

$\lambda x^\tau.M'$, where M' is of type τ', such that $|\lambda x^\tau.M'|_s = f$. Then by the induction hypothesis and the definitions of $R^{can}_{\tau\to\tau'}$ and $\mathbb{R}^{\langle w,s\rangle}_{(\boldsymbol{T}[\![\tau']\!])^{[\![\tau]\!]}}$, we get

$$\lambda x^\tau.M_1' \; R^{can}_{\tau\to\tau'} \; \lambda x^\tau.M_2' \iff$$
$$(\boldsymbol{T}[\![\tau']\!])^{[\![\tau]\!]} i_1(|\lambda x^\tau.M_1'|_s) \; \mathbb{R}^{\langle w,s\rangle}_{(\boldsymbol{T}[\![\tau']\!])^{[\![\tau]\!]}} \; (\boldsymbol{T}[\![\tau']\!])^{[\![\tau]\!]} i_2(|\lambda x^\tau.M_2'|_s)$$

for some $w \subseteq s$ and $i_1 : s_1 \to s, i_2 : s_2 \to s \in \mathcal{I}$.

- The case for the monadic types in the metalanguage corresponds to R^{exp}_σ in the operational logical relation. If $s_1 \vdash M_1 \Downarrow^{s_1'}_\sigma C_1$ and $s_2 \vdash M_2 \Downarrow^{s_2'}_\sigma C_2$, then by the definition of R^{exp}_σ we can find some $R' : s_1' \rightleftharpoons s_2'$ such that $C_1 \; (R \oplus R')^{can}_\sigma \; C_2$, and consequently by the induction hypothesis, we get

$$[\![\sigma]\!] j_1(|C_1|_{s_1+s_1'}) \; \mathbb{R}^{\langle w',s'\rangle}_{[\![\sigma]\!]} \; [\![\sigma]\!] j_2(|C_2|_{s_2+s_2'}),$$

for some $w' \subseteq s'$ and $j_1 : s_1 + s_1' \to s', j_2 : s_2 + s_2' \to s' \in \mathcal{I}$. On the other hand, by (1) and the definition of $\boldsymbol{Set}^{\mathcal{I}}$, we have

$$\boldsymbol{T}[\![\sigma]\!] i_k([\![M_k]\!]_{s_k}) = [s_k', [\![\sigma]\!](i_k + \mathrm{id}_{s_k'})|C_k|_{s_k+s_k'}] \qquad (k = 1, 2),$$

and according to the definition of $\mathbb{R}^{\langle w,s\rangle}_{\boldsymbol{T}[\![\sigma]\!]}$,

$$\boldsymbol{T}[\![\sigma]\!] i_1([\![M_1]\!]_{s_1}) \; \mathbb{R}^{\langle w,s\rangle}_{\boldsymbol{T}[\![\sigma]\!]} \; \boldsymbol{T}[\![\sigma]\!] i_2([\![M_2]\!]_{s_2}) \iff$$
$$\exists s_0, w_0 \subseteq s_0, i_1' : s_1' \to s_0, i_2' : s_2' \to s_0 \in \mathcal{I},$$
$$[\![\sigma]\!](i_1 + i_1')|C_1|_{s_1+s_1'} \; \mathbb{R}^{\langle w+w_0, s+s_0\rangle}_{[\![\sigma]\!]} \; [\![\sigma]\!](i_2 + i_2')|C_2|_{s_2+s_2'}.$$

Let i_k, i_k' $(k = 1, 2)$ be defined as

$$\forall n \in s_k, i_k(n) = j_k(n) \qquad \text{and} \qquad \forall n' \in s_k', i_k'(n') = j_k(n'),$$

and let $s + s_0 = s'$, $w + w_0 = w'$, then we get

$$M_1 \; R^{exp}_\sigma \; M_2 \iff \boldsymbol{T}[\![\sigma]\!] i_1([\![M_1]\!]_{s_1}) \; \mathbb{R}^{\langle w,s\rangle}_{\boldsymbol{T}[\![\sigma]\!]} \; \boldsymbol{T}[\![\sigma]\!] i_2([\![M_2]\!]_{s_2}). \qquad \square$$

We only claim the equivalence between these two logical relations for ground types and first-order function types. The failure of the equivalence for high-order function types is due to the non-fully abstract categorical model, while a similar Kripke logical relation defined directly on the computational metalanguage has been shown to be equivalent to the operational logical relation for any type [16].

5 Conclusion

We construct a Kripke logical relation for Pitts and Stark's nu-calculus in this paper and we show that it is equivalent to their operational logical relation up to first-order types. Because our Kripke logical relation is defined on a categorical model, it does not rely on any operational aspect of the nu-calculus, hence it is easier to be generalized. Our future work is to use a similar method to derive logical relations for the cryptographic lambda calculus, which is syntactically an extension of the nu-calculus (Sumii and Pierce define a logical relation for this language in [15], but it is still very operational).

Acknowledgements

We are grateful to Jean Goubault-Larrecq, John Power and Ian Stark for useful discussions and to anonymous reviewers for their comments.

References

1. Moez Alimohamed. A characterization of lambda definability in categorical models of implicit polymorphism. *Theoretical Computer Science*, 146(1–2):5–23, 1995.
2. Jon Beck. Distributive laws. In *Seminar on Triples and Categorical Homology Theory*, volume 80 of *Lecture Notes in Mathematics*, pages 119–140. Springer-Verlag, 1969.
3. Jean Goubault-Larrecq, Slawomir Lasota, and David Nowak. Logical relations for monadic types. In *CSL'02*, volume 2471 of *LNCS*, pages 553–568. Springer-Verlag, 2002.
4. Joachim Lambek and Philip J. Scott. *Introduction to Higher Order Categorical Logic*. Cambridge studies in advanced mathematics. Cambridge University Press, 1986.
5. John C. Mitchell. *Foundations of Programming Languages*. MIT Press, 1996.
6. John C. Mitchell and Eugenio Moggi. Kripke-style models for typed lambda calculus. In *LICS'87*, pages 303–314. The Computer Society of the IEEE, 1987.
7. John C. Mitchell and Andre Scedrov. Notes on sconing and relators. In *CSL'92*, volume 702 of *LNCS*, pages 352–378. Springer Verlag, 1993.
8. Eugenio Moggi. Computational lambda-calculus and monads. In *LICS'89*, pages 14–23. IEEE Computer Society Press, 1989.
9. Eugenio Moggi. An abstract view of programming languages. Technical Report ECS-LFCS-90-113, LFCS, Department of Computer Science, University of Edinburgh, 1990.
10. Eugenio Moggi. Notions of computation and monads. *Information and Computation*, 93:55–92, 1991.
11. Andrew Pitts and Ian Stark. Observable properties of higher order functions that dynamically create local names, or: What's *new*? In *MFCS'93*, volume 711 of *LNCS*, pages 122–141. Springer-Verlag, 1993.
12. Ian Stark. *Names and Higher-Order Functions*. PhD thesis, University of Cambridge, 1994.
13. Ian Stark. Categorical models for local names. *Lisp and Symbolic Computation*, 9(1):77–107, 1996.
14. Ian Stark. Names, equations, relations: Practical ways to reason about *new*. *Fundamenta Informaticae*, 33(4):369–396, 1998.
15. Eijiro Sumii and Benjamin Pierce. Logical relations for encryption. In *CSFW'01*, pages 256–272. IEEE, 2001.
16. Yu Zhang. Logical relations for names. Master's thesis, University of Paris 7, 2002. *http://www.lsv.ens-cachan.fr/Publis/PAPERS/ZY-dea02.ps.*