Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 203 (2008) 195–220
www.elsevier.com/locate/entcs

Predicate Liftings Versus Nabla Modalities

Raul Andres Leal[1,2]

ILLC
Univeriteit van Amsterdam
Amsterdam, The Netherlands

Abstract

We compare Moss' language and languages with predicate liftings. We prove that every monadic predicate lifting for a Kripke polynomial functor can be translated into Moss' language. We also prove that Moss' modality can always be translated into an appropriate language with predicate liftings.

Keywords: Coalgebra, modal logic, predicate lifting, singleton lifting, Moss' modality, logical translator.

1 Introduction

Modal logic is the logic of transition systems, which are coalgebras for the covariant power set functor. Coalgebraic (modal) languages generalize modal logic in that they describe more general kinds of dynamic systems i.e. coalgebras for different endofunctors. One important requirement for a coalgebraic language is that bisimilarity coincides with logical equivalence of states. A language with this property is said to be *expressive* with respect to bisimulation.

Lawrence S. Moss [4] defined a family of expressive [4,5,8] coalgebraic languages, which is parametric in T for any accessible and weak pullback preserving Set-endofunctor T. For any such functor, the associated Moss' language, denoted by $\mathcal{M}_T$ (Definition 2.2), has a unique modality, denoted by ∇_T, which is defined using relation liftings (see section 2.2). For Kripke frames, seen as coalgebras for the covariant power set functor $\mathcal{P}$, Moss' modality acts over sets of formulas and can be described in terms of the basic modal language as follows:

$$\nabla_{\mathcal{P}}\Psi = \Box \bigvee \Psi \wedge \bigwedge \Diamond \Psi.$$

[1] Thanks to Alessandra Palmigiano, Yde Venema, the anonymous referees, Alexander Kurz, Dirk Pattinson, and Clemens Kupke for all their valuable comments and suggestions. The research of this author has been made possible by VICI grant 639.073.501 of the Netherlands Organization for Scientific Research (NWO).

[2] Email: rlealrod@science.uva.nl

doi:10.1016/j.entcs.2008.05.026

Conversely, using $\nabla_{\mathcal{P}}$ as a primitive modality, the basic modalities can be expressed as follows:

$$\Box\varphi = \nabla_{\mathcal{P}}\emptyset \vee \nabla_{\mathcal{P}}\{\varphi\}; \quad \Diamond\varphi = \nabla_{\mathcal{P}}\{\varphi, \top\}.$$

However, in the case of other functors, it is not so easy to extract familiar modalities from Moss' modality.

A more direct approach to the definition of coalgebraic languages uses predicate liftings (see section 2.14). The underlying observation is that the modalities $\Box$ and $\Diamond$ are special predicate liftings. The idea of using (monadic) predicate liftings to define languages to describe coalgebras was first introduced by Dirk Pattinson [5]. Unlike Moss' modality, which is unique per functor, there are many different predicate liftings for a given functor T, which give rise to different T-languages, parametrized over sets of predicate liftings. Not all languages of (monadic) predicate liftings are expressive. Lutz Schröder [6] introduced the concept of polyadic predicate lifting and proved that every accessible functor that preserves monomorphisms admits a set of polyadic liftings Λ such that the associated language $\mathcal{L}_T(\Lambda)$ is expressive.

These two approaches define, in principle, two different expressive languages for any accessible functor that preserves weak pullbacks. In the case of Kripke frames, as we saw before, the two languages $\mathcal{M}_{\mathcal{P}}$ and $\mathcal{L}_{\mathcal{P}}(\{\Box, \Diamond\})$ can be mutually translated. Then a natural question is: *can this mutual translation be extended to other functors as well?* In this paper we partially answer this question by showing that every monadic predicate lifting for a Kripke polynomial functor can be translated into Moss' language [3]. In the converse direction we show that Moss' modality can always be translated into an appropriate language of polyadic predicate liftings. More precisely, we will show that, under appropriate conditions:

(i) For every Kripke polynomial functor T and every PL-language $\mathcal{L}_T(\Lambda)$ with only monadic predicate liftings, we can find a translation

$$t : \mathcal{L}_T(\Lambda) \longrightarrow \mathcal{M}_T,$$

where $\mathcal{M}_T$ is Moss' language (with negations) for T (see Theorems 4.9 and 4.10 on page 21).

(ii) For every accessible and weak pullback preserving functor T, there exists a cardinal number κ and a translation

$$t : \mathcal{M}_T \longrightarrow \mathcal{L}_T(\Lambda),$$

from Moss' language to the language $\mathcal{L}_T(\Lambda)$, where Λ is the set of all η-ary predicate liftings for all $\eta \in \kappa$ (see Theorem 5.6 on page 23).

The first result was proved by the author in his MSc thesis at the ILLC in Amsterdam [3]. The construction is based on the development of the concept of *logical translator*. The second result is joint work of the author together with Dirk Pattinson and Yde Venema. It is based on a representation theorem by Jiří Adámek and

[3] Here we should assume that the languages involved have enough conjunctions, see Remarks 2.29 and 3.2

Vera Trnková (Theorem 5.1 here), and an inductive presentation of Moss' language given by Yde Venema in [7].

The structure of the paper goes as follows: In the next section, we fix some notation, define the concept of translation and set our framework. In Section 3, we present the main conceptual contribution of this paper, which is the concept of *logical translator* (Definition 3.4). In Section 4, we illustrate two properties of logical translators: first we illustrate how to produce translations from logical translators (Theorem 4.2), then we show how to extend logical translators under the operations of product and coproduct between functors. We also show how to extend logical translators using the covariant power set functor (Theorem 4.6). As a corollary we will conclude that every monadic predicate lifting for a Kripke polynomial functor can be translated into Moss' language with disjunctions of appropriate arity. In Section 5 showing how to translate Moss' modality (Theorem 5.6). We finish with some conclusions and suggestions for further research.

2 Preliminaries

In this paper we only consider endofunctors in the category Set that are accessible and weak-pullback preserving. These properties are needed to show the existence of Moss' language [4,5], and to construct set of predicate liftings such that the language $\mathcal{L}_T(\Lambda)$ is expressive [6]. Let us recall the relevant definitions.

Definition 2.1 A cardinal number κ is regular if its equal to its cofinality.

Definition 2.2 [see [5]] Let κ be a regular cardinal. A functor $T : Set \longrightarrow Set$ is *κ-accessible* if for all sets X and all $x \in TX$ there is a subset $Y_x \subseteq X$ with $|Y_x| < \kappa$ such that $x \in T(i)(Y_x)$, where $i : Y_x \longrightarrow X$ is the inclusion; and the set $T(i)(Y_x)$ is the direct image of Y_x under the function $T(i)$.

A functor is *accessible* iff it is κ-accessible for some regular cardinal κ, and it is *finitary* iff it is ω-accessible.

Intuitively, a translation is a map between languages which preserves meaning, so formally:

Definition 2.3 Let T be a functor, let $\mathcal{L}_1$ and $\mathcal{L}_2$ be two languages that can be interpreted over T-coalgebras. A translation from $\mathcal{L}_1$ to $\mathcal{L}_2$ is a function $t : \mathcal{L}_1 \longrightarrow \mathcal{L}_2$ that preserves semantics, i.e. if $\varphi \in \mathcal{L}_1$ then $[\![\varphi]\!]_1 = [\![t(\varphi)]\!]_2$, where $[\![-]\!]_i$ is the satisfaction relation for $\mathcal{L}_i$ with $i = 1, 2$.

Definition 2.4 A functor T preserves weak pullbacks iff the diagram on the left

$$\begin{array}{ccc} TA & \xrightarrow{T(p)} & TB \\ {\scriptstyle T(q)}\downarrow & & \downarrow{\scriptstyle T(f)} \\ TC & \xrightarrow[T(g)]{} & TD \end{array} \qquad \begin{array}{ccc} A & \xrightarrow{p} & B \\ {\scriptstyle q}\downarrow & & \downarrow{\scriptstyle f} \\ C & \xrightarrow[g]{} & D \end{array}$$

is a weak pullback whenever the diagram on the right is a weak pullback.

2.1 Notation

We use T to denote functors and the greek letters λ, μ, τ to denote natural transformations.

Let $\check{\mathcal{P}}$ be the contravariant power set functor, $\mathcal{P}$ be the covariant power set functor, I be the identity functor. We write D for the constant functor with value D, and $Hom(D, -)$ for the (covariant) homomorphism functor. We use $\mathcal{D}$ for the finite distribution functor and $\mathcal{B}_{\mathbb{N}}$ for the finite multiset functor (bags).

For every cardinal number κ, we write $\mathcal{P}_\kappa$ for the functor mapping a set A to the collection of its subsets of cardinality strictly less than κ, and a function $f : A \longrightarrow B$ to its direct image.

The collection of *Kripke polynomial functors*, or KPFs for short, is inductively defined as follows:

$$K := I \mid D \mid K + K \mid K \times K \mid Hom(A, K) \mid \mathcal{P}K,$$

where A is a set and D is a constant functor with value D. Replacing $\mathcal{P}$ with the finite power set functor $\mathcal{P}_\omega$ we obtain the collection of *finitary Kripke polynomial functors*. Constant functors are used to introduce constants to our coalgebraic languages, because of that we will often refer to KPFs over a set D.

Given any other functor H, the collection of H-Kripke polynomial functors, or KPF + H for short, is inductively defined as follows:

$$K := I \mid D \mid H \mid K + K \mid K \times K \mid Hom(A, K) \mid \mathcal{P}K,$$

i.e. the functor H is added as a base case.

If $X \subseteq S$ and $U \subseteq \mathcal{P}S$, we denote the characteristic function of X by χ_X, and the complement of X in S by $\neg_S X$. We write $\bigwedge_S(U)$ for the intersection of the elements in U. The subindexes stress the fact that intersections and complements define natural transformations; the complement defines a natural transformation $\neg : \check{\mathcal{P}} \longrightarrow \check{\mathcal{P}}$, and intersections define a natural transformation $\bigwedge : \mathcal{P}\check{\mathcal{P}} \longrightarrow \check{\mathcal{P}}$. This will be relevant in the definition of Moss' language.

A T-coalgebra is a pair (S, σ) such that S is a set and σ is a function of the form $\sigma : S \longrightarrow TS$. The set S will be called the set of *states* and σ the *transition structure*, or transition map, of the coalgebra. We often identify a coalgebra with its transition structure σ (from which the set of states S can be determined as the domain of σ).

Moss' language and PL-languages are defined using certain natural transformations. Let us first introduce Moss' language.

2.2 Moss' Language

Moss' language uses a unique modality, which is defined using relation liftings.

Definition 2.5 [see [5]] For any functor $T : Set \longrightarrow Set$ and relation $R \subseteq X \times Y$, the relation lifting $\overline{T}(R) \subseteq TX \times TY$ of R, using T, is defined as follows:

$$\overline{T}(R) = \{(T(\pi_X)(u), T(\pi_Y)(u)) \,|\, u \in TR\}$$

Using relation liftings, we can characterize the property of weak pullback preservation, see [5] for more references and details.

Proposition 2.6 *A functor T preserves weak pullbacks if $\overline{T}(R \circ S) = \overline{T}R \circ \overline{T}S$ for all composable relations R, S.*

Example 2.7 If T be the covariant power set functor $\mathcal{P}$, we can easily show that the relation lifting of R is given by:

$$\overline{\mathcal{P}}(R) = \{(Q_0, Q_1) \,|\, (\forall q_0 \in Q_0)(\exists q_1 \in Q_1))((q_0, q_1) \in R) \\ \text{and } (\forall q_1 \in Q_1)(\exists q_0 \in Q_0))((q_0, q_1) \in R)\}\,.$$

In the case of KPFs, relation liftings can be described inductively, see [8] for more information and references:

Proposition 2.8 *Let S and S' be sets, and $R \subseteq S \times S'$ a binary relation. The following induction defines the relation lifting $\overline{K}(R) \subseteq KS \times KS'$, for each Kripke polynomial functor K.*

- $\overline{I}(R) = R$,
- $\overline{D}(R) = \Delta_D$,
- $\overline{K_0 \times K_1}(R) = \{((x_0, x_1), (x_0', x_1')) \,|\, (x_0, x_0') \in \overline{K_1}(R) \text{ and } (x_1, x_1') \in \overline{K_2}(R)\}$,
- $\overline{K_0 + K_1}(R) = \{(i_1(s), i_1(t)) \,|\, (s,t) \in \overline{K_1}(R)\} \cup \{(i_2(s), i_2(t)) \,|\, (s,t) \in \overline{K_2}(R)\}$,
- $\overline{Hom(D, K(R))} = \{(f, f') \,|\, (\forall d \in D)((f(d), f'(d)) \in \overline{K}(R))\}$,
- $\overline{\mathcal{P}K}(R) = \big\{(Q_0, Q_1) \,|\, (\forall q_0 \in Q_0)(\exists q_1 \in Q_1))((q_0, q_1) \in \overline{K}(R)) \\ \text{and } (\forall q_1 \in Q_1)(\exists q_0 \in Q_0))((q_0, q_1) \in \overline{K}(R))\big\}$,

Definition 2.9 [Moss' modality] Let T be a functor that preserves weak pullbacks. Moss' modality is the natural transformation $\nabla_T : T\check{\mathcal{P}} \longrightarrow \check{\mathcal{P}}T$ such that the S-component $(\nabla_T)_S : T\check{\mathcal{P}}S \longrightarrow \check{\mathcal{P}}TS$ maps an element $t \in T\check{\mathcal{P}}S$ to

$$(\nabla_T)_S(t) = \{s' \in TS \,|\, (s', t) \in \overline{T}(\in_S)\},$$

where $\in_S$ is the membership relation between elements in S and sets in $\check{\mathcal{P}}S$.

The hypothesis that T preserves weak pullbacks is needed to show that the previous functions are the components of a natural transformation, see [5].

Definition 2.10 [Moss' Language, see [5]] Let T be an accessible functor, and κ a regular cardinal which serves as a bound for conjunctions. Moss' language $\mathcal{M}_T^\kappa$ is the carrier of an initial algebra for the functor $L := \mathcal{P}_\kappa + T + I$. The elements of $\mathcal{M}_T^\kappa$ are the formulas of Moss' language.

The hypothesis of T being accessible is needed to prove that the initial algebra in the previous definition exists, see [1] or [9]. Recall that all initial algebras are isomorphic.

Notice that, by the universal property of coproducts, there exist three functions:

$$\bigwedge : \mathcal{P}_\kappa(\mathcal{M}_T^\kappa) \longrightarrow \mathcal{M}_T^\kappa; \quad \nabla : T\mathcal{M}_T^\kappa \longrightarrow \mathcal{M}_T^\kappa; \quad \neg : \mathcal{M}_T^\kappa \longrightarrow \mathcal{M}_T^\kappa.$$

These functions respectively define, conjunctions, Moss' modality, and negations in Moss' language.

Definition 2.11 [Complex algebra functor] Let T be an accessible functor that preserves weak pullbacks. The *complex algebra functor* is the functor

$$M_T : Coalg(T)^{op} \longrightarrow Alg(\mathcal{P}_\kappa + T + I).$$

this functor maps a T-coalgebra (S, σ) to the algebra

$$[\bigwedge_S, \sigma^{-1}(\nabla_T)_S, \neg_S] : \mathcal{P}_\kappa\check{\mathcal{P}}S + T\check{\mathcal{P}}S + \check{\mathcal{P}}S \longrightarrow \mathcal{P}S,$$

and a morphism $f : \sigma_1 \longrightarrow \sigma_2$ to

$$f^{-1} : (\mathcal{P}S_2, [\bigwedge_{S_2}, \sigma_2^{-1}(\nabla_T)_{S_2}, \neg_{S_2}]) \longrightarrow (\mathcal{P}S_1, [\bigwedge_{S_1}, \sigma_1^{-1}(\nabla_T)_{S_1}, \neg_{S_1}]).$$

The image of a coalgebra σ under the functor M is called the complex algebra of σ.

Remark 2.12 [Notational issues] We denote arbitrary formulas of Moss' language by ψ and use Ψ for special "formulas" in $T\mathcal{M}_T^\kappa$. This is because we want to stress that the objects in $T\mathcal{M}_T^\kappa$ act like formulas of a different nature.

The symbol ∇ has two uses for us, one is to denote the natural transformation defining Moss' modality (Definition 2.9), a second use is to denote the actual modality of Moss' language. We use the same symbol for both situations because they are tightly related. This we also want to simplify out notation.

Definition 2.13 [Moss' Semantics, see [5]] The semantics $[\![-]\!]^\sigma_{\mathcal{M}} : \mathcal{M}_T^\kappa \longrightarrow \check{\mathcal{P}}S$ of $\mathcal{M}_T^\kappa$ with respect to (S, σ) is the unique morphism of $Alg(\mathcal{P}_\kappa + T + I)$-algebras from $\mathcal{M}_T^\kappa$ to $M(S, \sigma)$. We also refer to $[\![-]\!]^\sigma_{\mathcal{M}}$ as the satisfaction relation of Moss' language.

Since the satisfaction relation defined above is a homomorphism of L-algebras, the following diagram

$$\begin{array}{ccc} T\mathcal{M}_T^\kappa & \xrightarrow{\nabla} & \mathcal{M}_T^\kappa \\ {\scriptstyle T([\![-]\!]^\sigma_{\mathcal{M}})}\downarrow & & \downarrow{\scriptstyle [\![-]\!]^\sigma_{\mathcal{M}}} \\ T\check{\mathcal{P}}S & \xrightarrow[\sigma^{-1}(\nabla_T)_S]{} & \check{\mathcal{P}}S \end{array}$$

commutes for every coalgebra σ. In other words, the extension of a formula $\nabla\Psi \in \mathcal{M}_T^\kappa$ can be computed as follows:

$$[\![\nabla\Psi]\!]^\sigma_{\mathcal{M}} = \sigma^{-1}(\nabla_T)_S T([\![-]\!]^\sigma_{\mathcal{M}})(\Psi).$$

This fact combined with Proposition 2.8 implies that in the case of KPFs, the satisfaction relation for Moss' language can be described inductively.

2.3 PL-languages

PL-languages have different modalities, which are obtained using predicate liftings.

Definition 2.14 [Predicate Liftings, see [6]] Let η be a cardinal number. An η-ary predicate lifting λ for T is a natural transformation

$$\lambda : (\check{\mathcal{P}})^\eta \longrightarrow \check{\mathcal{P}}T.$$

If $\eta = 1$, we say that λ is a monadic predicate lifting; or a predicate lifting for short. A polyadic predicate lifting is an η-ary predicate lifting for some cardinal η.

Example 2.15 For the covariant power set functor, the existential modality $\Diamond$ can be seen as the predicate lifting, the S-component of which is the function

$$\Diamond_S : \check{\mathcal{P}}S \longrightarrow \check{\mathcal{P}}\mathcal{P}S$$

mapping a set $X \subseteq S$ to

$$\Diamond_S(X) = \{U \subseteq S \,|\, U \cap X \neq \emptyset\}.$$

This shows that PL-languages generalize the basic modal language.

Definition 2.16 [see [6]] Let T be a functor, let Λ be some set of polyadic predicate liftings, and let κ be a regular cardinal that serves as a bound for conjunctions. The PL-language $\mathcal{L}_T^\kappa(\Lambda)$ associated with Λ is the least inductively defined set

$$\varphi := \bigwedge \Phi \,|\, \neg\varphi \,|\, \lambda(\varphi_i)_{i\in\eta};$$

such that $\Phi \subseteq \mathcal{L}_T^\kappa(\Lambda)$ with $|\Phi| < \kappa$, and λ is an η-ary lifting in Λ.

For every T-coalgebra σ, the semantics $[\![\varphi]\!]^\sigma_{\mathcal{L}} \subseteq S$ is given inductively by the following clauses:

(i) $[\![\bigwedge \Phi]\!]^\sigma_{\mathcal{L}} = \bigcap_{\varphi\in\Phi} [\![\varphi]\!]^\sigma_{\mathcal{L}}$.
(ii) $[\![\neg\varphi]\!]^\sigma_{\mathcal{L}} = \neg_S [\![\varphi]\!]^\sigma_{\mathcal{L}}$.
(iii) $[\![\lambda(\varphi_i)_{i\in\eta}]\!]^\sigma_{\mathcal{L}} = \sigma^{-1}\lambda_S([\![\varphi_i]\!]^\sigma_{\mathcal{L}})_{i\in\eta}$,

Remark 2.17 [Notational issues] We denote arbitrary formulas of PL-language by φ. Like we did for ∇, we use the symbol λ to denote predicate liftings (Defintion 2.14), and the modality associated with λ. This helps to keep the notation simple.

Lutz Schröder showed that the number of predicate liftings for T is closely related to the action of T on objects. this is stated in the following result.

Theorem 2.18 (see [6]) *The set of η-ary predicate liftings is in natural bijection with the subsets of $T(2^\eta)$.*

The following definition tell us how to obtain a predicate lifting from a set $C \subseteq T(2^\eta)$, it is motivated from the proof of the previous theorem.

Definition 2.19 Given a η-ary predicate lifting λ, we say that λ is associated with a set $C \subseteq T(2^\eta)$ if for every η-sequence $\overrightarrow{X}$ of subsets of S the following holds:

$$(\lambda_C)_S(\overrightarrow{X}) = \{t \in TS \mid T(\chi_{\overrightarrow{X}})(t) \in C\},$$

where $\chi_{\overrightarrow{X}} : X \longrightarrow 2^\eta$ is the product arrow of the characteristic functions of the sets in $\overrightarrow{X}$, i.e. if $\overrightarrow{X} = (X_i)_{i\in\eta}$ then $\chi_{\overrightarrow{X}}$ is the unique function such that $\pi_i \chi_{\overrightarrow{X}} = \chi_{X_i}$ for all $i \in \eta$.

As instances of the previous definition, we will present some examples of predicate liftings.

Example 2.20 If T is the covariant power set functor, and we represent $2 = \{\top, \bot\}$, the existential modality $\Diamond$ is associated with the set $\{\{\top\}, \{\top, \bot\}\}$. Analogously, the universal modality corresponds to the set $\{\emptyset, \{\top\}\}$.

Example 2.21 Given a cardinal number η, if $T(2^\eta)$ is nonempty, then T has at least two η-ary predicate liftings: one associated with the empty set, which will be denoted as $\lambda_\bot$ instead of $\lambda_\emptyset$; another one associated with the set $T(2^\eta)$, which will be denoted as $\lambda_\top$ instead of $\lambda_{T(2^\eta)}$. The S- components of these predicate liftings map an η-sequence $\overrightarrow{X}$ of subsets of S as follows: $(\lambda_\bot)_S(\overrightarrow{X}) = \emptyset$, and $(\lambda_\top)_S(\overrightarrow{X}) = TS$. Notice that for any η-sequence of formulas $(\varphi_i)_{i\in\eta} \subseteq \mathcal{L}_T^\kappa(\Lambda)$, the formula $\lambda_\bot(\varphi_i)_{i\in\eta}$ is never true, and the formula $\lambda_\top(\varphi_i)_{i\in\eta}$ is always true.

Among predicate liftings some are simpler than others, for example those related to singleton sets.

Definition 2.22 An η-ary predicate lifting λ is called a singleton predicate lifting, or a singleton lifting for short, if it is associated with an element $p \in T(2^\eta)$, i.e. if the following holds

$$\lambda_S(\overrightarrow{X}) = \{t \in TS \mid T(\chi_{\overrightarrow{X}})(t) = p\}. \tag{1}$$

If λ is a singleton lifting, we write it λ_p where p is the associated element of $T(2^\eta)$.

Example 2.23 If T is a constant functor with value D, then the singleton liftings for T are associated with elements $d \in D$. The S-component of a singleton lifting λ_d is the function $\lambda_d : \check{\mathcal{P}}S \longrightarrow \check{\mathcal{P}}D$ with constant value $\{d\}$.

Example 2.24 If T is the identity functor and we assume $2 = \{\top, \bot\}$, then there are two singleton liftings for I. The S-component of $\lambda_{\{\top\}}$ is the identity. Similarly,

the S- component of $\lambda_{\{\bot\}}$ is the function $(\lambda_{\{\bot\}})_S : \check{\mathcal{P}}S \longrightarrow \check{\mathcal{P}}S$ mapping a set $X \subseteq S$ to $\lambda_{\{\top\}}(X) = \neg_S X$.

Example 2.25 If T is the finite multiset functor, a singleton lifting is given by a pair of natural numbers (n, m). The S-component of $(\lambda_{(n,m)})_S$ is the function $(\lambda_{(n,m)})_S : \check{\mathcal{P}}S \longrightarrow \check{\mathcal{P}}\mathcal{B}_{\mathbb{N}}S$ mapping a set X to the set of bags over S with $n + m$ elements of whose n are in X and m are in its complement.

Example 2.26 If T is the finite distribution functor, a singleton lifting is given by a real number $q \in [0, 1]$. The S-component of λ_q is the function $(\lambda_q)_S : \check{\mathcal{P}}S \longrightarrow \check{\mathcal{P}}\mathcal{D}S$ mapping a set X to the set of probability distributions over S that give probability q to the set X.

One reason to pay special attention to singleton liftings is that in the case of KPFs they can be presented inductively over the complexity of the functor, as the next example illustrates.

Example 2.27 For every set $P \in \mathcal{P}T(2)$, the singleton lifting $\lambda_P : \check{\mathcal{P}} \longrightarrow \check{\mathcal{P}}\mathcal{P}T$ for $\mathcal{P}T$ can be presented using the singleton liftings for T associated with the elements of P as follows: for any η-sequence $\overrightarrow{X}$ of subsets of S,

$$(\lambda_P)_S(\overrightarrow{X}) = \{U \subseteq TS \,|\, U \subseteq \bigcup_{p \in P} (\lambda_p)_S(\overrightarrow{X}) \text{ and } (\forall p \in P)(U \cap (\lambda_p)_S(\overrightarrow{X}) \neq \emptyset)\}.$$

It is also possible to give inductive presentations like the previous one in the case of products and coproducts. This is especially useful in the case of products of functors. Singleton liftings are even more important because they generate all the other predicate liftings as it is shown in the next result.

Proposition 2.28 *If λ is an η-ary predicate lifting associated with a set $P \subseteq T(2^\eta)$, then for every set S and every η-sequence $\overrightarrow{X}$ of subsets of S we have*

$$\lambda_S(\overrightarrow{X}) = \bigcup_{p \in P} (\lambda_p)_S(\overrightarrow{X}).$$

In other words, every η-ary predicate lifting can be obtained as a join of singleton predicate liftings.

Proof. Since λ is associated with P its action over an η-sequence $\overrightarrow{X}$ can be described as follows

$$\begin{aligned}(\lambda_P)_S(\overrightarrow{X}) &= \{t \in TS \,|\, T(\chi_{\overrightarrow{X}})(t) \in P\} \\ &= \bigcup_{p \in P} \{t \in TS \,|\, T(\chi_{\overrightarrow{X}})(t) = p\} = \bigcup_{p \in P} (\lambda_p)_S(\overrightarrow{X}).\end{aligned}$$

This concludes the proof. □

Remark 2.29 [Concerning the arity of disjunctions] The previous proposition implies that for every language $\mathcal{L}^\kappa_T(\Lambda)$ with disjunction of at least size $|T(2^\eta|$, i.e.

$|T(2^\eta| < \kappa$, then every η-ary predicate lifting $\lambda \in \Lambda$ can be obtained as a disjunction of singleton liftings. This fact will play an important role in the translation of predicate liftings for Kripke polynomial functors, see Example 3.1. From now on we will make the following assumption: if $\mathcal{L}_T^\kappa(\Lambda)$ is a language of predicate liftings for a Kripke polynomial functor over a set D, then $\mathcal{L}_T^\kappa(\Lambda)$ has disjunctions of size at least $|D|$, i.e. $|D| < \kappa$.

3 Translations & Translators

Now that we fixed our framework, we can explain in detail the aim of this paper. Our first goal is to translate monadic predicate liftings. Our first question is:

> Can we translate every monadic predicate lifting λ for a KPF T into Moss' language for T?

If we only accept finitary formulas on the languages, the answer is no. Consider the following example.

Example 3.1 Let T be the constant functor with value $\mathbb{N}$, let $E \subseteq \mathbb{N}$ be the set of even numbers, and let λ_E be the predicate lifting associated with E. The predicate lifting λ_E can not be expressed in $\mathcal{M}_T^\omega$, i.e. Moss' language for T with finitary conjunctions and negation.

Consider the T coalgebra $N = (\mathbb{N}, 1_\mathbb{N})$, and the formula $\lambda_E\top$. The formula $\lambda_E\top$ defines the set of even numbers in the coalgebra N, i.e $[\![\lambda_E\top]\!]^\sigma_\mathcal{L} = E$. We will show that this set is not definable in Moss' language, i.e there exists no formula $\psi \in \mathcal{M}_T^\omega$ such that $[\![\psi]\!]^\sigma_\mathcal{M} = E$.

Clearly the formula $\top$ does not define E; now we will show that no finite formula involving $\nabla_\mathbb{N}$ does. Recall that Moss' modality for T can only be applied to elements of $\mathbb{N}$, and that a state s in a coalgebra (S, σ) satisfies a formula $\nabla_\mathbb{N} n$ iff $\sigma(s) = n$. From this we conclude that no finite conjunction of formulas with the shape $\nabla_\mathbb{N} n$ defines E. Furthermore notice that a formula $\nabla_\mathbb{N} n \wedge \nabla_\mathbb{N} m$, with $n \neq m$, defines the empty set. Now consider a formula $\neg\nabla_\mathbb{N} n$; a state s in a coalgebra (S, σ) satisfies $\neg\nabla_\mathbb{N} n$ iff $\sigma(s) \neq n$. From this we conclude that any finite conjunction of formulas with the shape $\neg\nabla_\mathbb{N} n$ defines a co-finite set in N. We conclude that there is no formula $\psi \in \mathcal{M}_T^\omega$ such that $[\![\psi]\!]^\sigma_\mathcal{M} = E$. This implies that the formula $\lambda_E\top$ can not be translated into Moss' language with only finitary conjunctions.

Remark 3.2 [Concerning the arity of disjunctions] The formula $\lambda_E\top$ in the previous example can be expressed into Moss' language if we allow infinite (countable) conjunctions in $\mathcal{M}_T^\kappa$, i.e. $\omega < \kappa$. Recall that constant functors are used to add constants to the language. Following the idea of Remark 2.29, from now on we will make the following assumption: Moss' language for a Kripke polynomial functor, over a set D, has disjunctions of at least size $|D|$. Theorems 4.9 and 4.10 present some conditions under which every monadic predicate lifting can be translated.

Fix a functor T, a PL-language $\mathcal{L}_T^\kappa(\Lambda)$, and a predicate lifting $\lambda \in \Lambda$. Since the functor T is fixed, we will write ∇ instead of ∇_T to simplify our notation. In order to

translate λ into Moss' language, we should show that for every formula $\varphi \in \mathcal{L}_T^\kappa(\Lambda)$ there exists a formula $\psi \in \mathcal{M}_T^\kappa$ such that $[\![\lambda\varphi]\!]_{\mathcal{L}}^\sigma = [\![\psi]\!]_{\mathcal{M}}^\sigma$. In other words, we would like to define a function $t : \mathcal{L}_T^\kappa(\Lambda) \longrightarrow \mathcal{M}_T^\kappa$ such that $[\![\lambda\varphi]\!]_{\mathcal{L}}^\sigma = [\![t(\lambda\varphi)]\!]_{\mathcal{M}}^\sigma$ for every T coalgebra. More precisely:

Definition 3.3 We say that a formula $\varphi \in \mathcal{L}_T^\kappa(\Lambda)$ can be translated into Moss' language if there exist a formula $\psi \in \mathcal{M}_T^\kappa$ such that $[\![\varphi]\!]_{\mathcal{L}}^\sigma = [\![\psi]\!]_{\mathcal{M}}^\sigma$ holds for every T-coalgebra.

Intuitively a predicate lifting λ describes some aspect of the one step evolution of coalgebras. Hence if a formula $\lambda\varphi$ can be translated into a formula $\psi \in \mathcal{M}_T^\kappa$, we would expect that Moss' modality takes part in the translation. Assume that we want our formula ψ to be the simplest. Namely we want $\psi = \nabla\Psi$, where $\Psi \in T\mathcal{M}_T^\kappa$. If we want the translation of λ to be of this form, we would like to define a function $t' : \mathcal{L}_T^\kappa(\Lambda) \longrightarrow T\mathcal{M}_T^\kappa$ such that

$$[\![\lambda\varphi]\!]_{\mathcal{L}}^\sigma = [\![\nabla t'(\lambda\varphi)]\!]_{\mathcal{M}}^\sigma.$$

Given t' we will define the translation to be $t(\lambda\varphi) = \nabla t'(\varphi)$. We will produce the function t' using *logical translators*; we will use the so called *signature of the translator*.

Now, we would like to stress an issue in the construction of a translation. We will first unravel the previous equation. By definition, we have that for any coalgebra σ, the following two equations hold:

$$[\![\lambda\varphi]\!]_{\mathcal{L}}^\sigma = \sigma^{-1}\lambda_S([\![\varphi]\!]_{\mathcal{L}}^\sigma) \text{ and } [\![\nabla t'(\lambda\varphi)]\!]_{\mathcal{M}}^\sigma = \sigma^{-1}\nabla_S T([\![-]\!]_{\mathcal{M}}^\sigma)t'(\varphi).$$

Putting these two equations together, we can see that we would like to inductively define a function t' such that

$$\sigma^{-1}\lambda_S[\![\varphi]\!]_{\mathcal{L}}^\sigma = \sigma^{-1}\nabla_S T([\![-]\!]_{\mathcal{M}}^\sigma)t'(\varphi).$$

This previous equation has three different kinds of components: the natural transformation associated with the modalities, the satisfaction relations, and the transition map σ. In the previous section we illustrated that the modalities and the satisfaction relations can be inductively presented over the complexity of T. Notice that transition maps can not be described inductively, for example, there is no natural way to describe a coalgebra for the functor $\mathcal{P}T$ in terms of coalgebras for the functor T. Therefore the dependence of the previous equation in the transition map σ might complicate our construction.

Since the presence of σ is problematic, we would like to avoid it. In other words, if we would like to construct a function t'; such that the equality

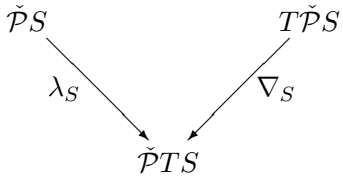
$$\lambda_S[\![\varphi]\!]_{\mathcal{L}}^\sigma = \nabla_S T([\![-]\!]_{\mathcal{M}}^\sigma)t'(\varphi) \tag{2}$$

holds for every set S. This might seem like a strong assumption, but one of the features of logical translators resides in the fact that they produce the function t'

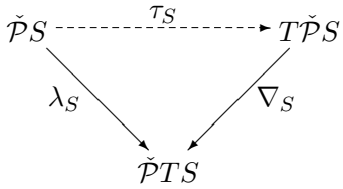
with exactly the property in the previous equation, see Remark 4.5 for a consequence of this. The intuition is that the formula $t'(\varphi)$ should be ∇-free. The development of logical translators was motivated by this previous intuition.

3.1 Logical Translators

Here we arrive at the key concept of this paper, this is the concept of *logical translator*. We will first give some intuitive motivations. Recall that the semantics of formulas $\lambda\varphi$ and $\nabla\Psi$ are defined using natural transformations with names λ (Definition 2.14) and ∇ (Definition 2.9). The S-component of Moss' modality is a function $\nabla_S : T\check{\mathcal{P}}S \longrightarrow \check{\mathcal{P}}TS$. Similarly, λ_S is a function $\lambda_S : \check{\mathcal{P}}S \longrightarrow \check{\mathcal{P}}TS$. The functions λ_S and ∇_S have the same codomain but different domains:



We have to relate the two domains. A first idea would be to complete the picture into a commutative diagram using a natural transformation $\tau : \check{\mathcal{P}} \longrightarrow T\check{\mathcal{P}}$.

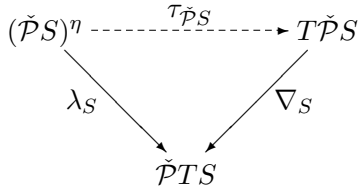


The intuition is that the natural transformation τ is a *semantic translation* of the predicate lifting λ. In other words, τ factors λ through ∇.

Unfortunately, we can not just use τ to define a translation. Since τ is a natural transformation just relating power sets, in principle it neither involves Moss' language nor the language $\mathcal{L}^{\kappa}_{T}(\Lambda)$. Recall that Moss' language $\mathcal{M}^{\kappa}_{T}$ is an algebra for the functor $L := \mathcal{P}_{\kappa} + T + I$. Empirical evidence, coming from examples, shows the main problem to define a translation is to transform formulas in $\mathcal{M}^{\kappa}_{T}$ to formulas in $T\mathcal{M}^{\kappa}_{T}$, i.e. to transform formulas for which we can not apply Moss' modality into formulas to which we can apply Moss modality. The examples also suggest that this transformation can be done using boolean combinations of formulas plus some kind of natural transformation. Hence, we do not want involve all L, but only the boolean part of L. In summary, the idea is not to use natural transformation $\tau : \check{\mathcal{P}} \longrightarrow T\check{\mathcal{P}}$, but a natural transformation $\tau : U \longrightarrow TU$, where U is the forgetful functor $U : Alg(\mathcal{P}_{\kappa} + I) \longrightarrow Set$. Notice that this argument still works for polyadic liftings. The above considerations lead us to the following definition.

Definition 3.4 [logical translators] Let λ be an η-ary predicate lifting for a functor T, and let $U : Alg(\mathcal{P}_{\kappa} + I) \longrightarrow Set$ be the forgetful functor. A logical translator τ for

λ is a natural transformation $\tau : (U)^\eta \longrightarrow TU$ such that that the following diagram



commutes, in *Set*, for every set S. Here $\tau_{\check{\mathcal{P}}S}$ is the component corresponding to the power set algebra $\check{\mathcal{P}}S$. We call the functions $\tau_{\check{\mathcal{P}}S}$ the components of the *power set restriction* of τ. Also notice that the two vertices at the top should be $U\check{\mathcal{P}}S$ and $TU\check{\mathcal{P}}S$, respectively, but we omit the forgetful functor to simplify our notation.

We would like to make some observations before presenting some examples.

Observation 1: notice that the components of a boolean transformation τ are indexed over algebras $Alg(\mathcal{P}_\kappa + I)$. This means that given an algebra $\mathfrak{A}$, the $\mathfrak{A}$ component of τ is a function $\tau_{\mathfrak{A}} : A \longrightarrow TA$, where A is the carrier set of $\mathfrak{A}$. This also implies that for each algebraic structure over A there should be a function from A to TA. In principle different algebraic structures with the same carrier set have different components.

Observation 2: Notice that a natural transformation $\tau : (U)^\eta \longrightarrow TU$ is a logical translator for λ iff for each η-sequence $\overrightarrow{X}$ of subsets of S the following holds

$$\lambda_S(\overrightarrow{X}) = \{s \in TS \,|\, (s, \tau_{\check{\mathcal{P}}S}(\overrightarrow{X})) \in \overline{T}(\in_S)\}.$$

Observation 3: Notice that a natural transformation $\tau : (U)^\eta \longrightarrow TU$ defines a natural transformation $\tau_{\check{\mathcal{P}}} : (\check{\mathcal{P}})^\eta \longrightarrow T\check{\mathcal{P}}$, as follows: Given a set S, the S-component of $\tau_{\check{\mathcal{P}}}$ is the S^+-component of τ, where S^+ is the power set algebra with underlying set $\check{\mathcal{P}}S$. In other words $(\tau_{\check{\mathcal{P}}})_S = \tau_{S^+}$. We call the natural transformation $\tau_{\check{\mathcal{P}}}$ the *power set restriction* of τ. Since the underlying set of S^+ is $\check{\mathcal{P}}S$, we write $\tau_{\check{\mathcal{P}}S}$ instead of $(\tau_{\check{\mathcal{P}}})_S$. This justifies the terminology used in the previous definition.

3.2 Examples of Logical Translators

Now we would like to present some examples of logical translators. We will also make the translations produced from those logical translators explicit. These translations can be computed as a consequence of Theorem 4.2; we will always assume the formula inside the predicate lifting can be translated. In all the examples presented here we assume $2 = \{\top, \bot\}$. We start with the covariant power set functor.

Example 3.5 Let $\Diamond$ be the predicate lifting associated with the existential modality (for the covariant power set functor). We define a logical translator τ for $\Diamond$ with the following components: $\tau_{\mathfrak{A}} : A \longrightarrow \mathcal{P}A$ maps an element $x \in A$ to $\tau_{\mathfrak{A}}(x) = \{x, \top\}$. This logical translator produces the translation $t(\Diamond\varphi) = \nabla\{t(\varphi), \top\}$.

Example 3.6 Let T be the covariant power set functor and consider the predicate lifting $\lambda_{\{\top\}}$ associated with the set $\{\top\} \in \mathcal{P}2$. Recall that this predicate lifting has the following components: a set $X \subseteq S$ is mapped to

$$\lambda_{\{\top\}}(X) = \{U \subseteq S \,|\, U \neq \emptyset \text{ and } U \subseteq X\}.$$

We define a logical translator τ for $\lambda_{\{\top\}}$ where the function $\tau_{\mathfrak{A}} : A \longrightarrow \mathcal{P}A$ maps an element $x \in A$ to $\tau_{\mathfrak{A}}(x) = \{x\}$. This logical translator produces the translation $t(\lambda_{\{\top\}}\varphi) = \nabla\{t(\varphi)\}$.

Example 3.7 Let T be the covariant power set functor and consider the predicate lifting $\lambda_{\{\top,\bot\}}$ associated with the set $\{\top, \bot\} \in \mathcal{P}2$. Recall that this predicate lifting has the following components: a set $X \subseteq S$ is mapped to

$$\lambda_{\{\top,\bot\}}(X) = \{U \subseteq S \,|\, U \cap X \neq \emptyset \text{ and } U \cap \neg_S X \neq \emptyset\}.$$

We define a logical translator τ for $\lambda_{\{\top,\bot\}}$ where the function $\tau_{\mathfrak{A}} : A \longrightarrow \mathcal{P}A$ maps an element $x \in A$ to $\tau_{\mathfrak{A}}(x) = \{x, \neg_{\mathfrak{A}} x\}$. This logical translator produces the translation $t(\lambda_{\{\top,\bot\}}\varphi) = \nabla\{t(\varphi), \neg t(\varphi)\}$.

In the cases of multisets and distributions we have the following situation.

Example 3.8 Let $\lambda_{(n,m)}$ be a singleton lifting for the multiset functor. We define a logical translator τ for $\lambda_{(n,m)}$ where the function $\tau_{\mathfrak{A}} : A \longrightarrow \mathcal{B}_{\mathbb{N}}A$ maps an element $x \in A$ to the following bag: $B_{(x,n,m)} : A \longrightarrow \mathbb{N}$

$$B_{(x,n,m)}(x) = n,\ B_{(x,n,m)}(\neg_{\mathfrak{A}} x) = m, \text{ and } B_{(x,n,m)}(a) = 0 \text{ for any other element.}$$

Using this we obtain the following translation $t(\lambda_{(n,m)}\varphi) = \nabla B_{(t(\varphi),n,m)}$.

Example 3.9 Let λ_q be a singleton lifting for the distribution functor. A logical translator τ for λ has the following components: The function $\tau_{\mathfrak{A}} : A \longrightarrow \mathcal{D}A$ maps an element x to the following distribution $\mu_{(x,q)} : A \longrightarrow [0, 1]$

$$\mu_{(x,q)}(x) = q,\ \mu_{(x,q)}(\neg_{\mathfrak{A}} x) = 1 - q, \text{ and } B_{(x,q)}(a) = 0 \text{ for any other element.}$$

The translation induced by this logical translator is $t(\lambda_q\varphi) = \nabla\mu_{(t(\varphi),q)}$.

Now we will present examples using the identity functor and constant functors; these examples will be used in the proof of Theorem 4.10.

Example 3.10 Let λ_d be a singleton lifting for a functor with constant value D. We define a logical translator τ where the $\mathfrak{A}$-component is the function $\tau_{\mathfrak{A}} : A \longrightarrow D$ with constant value d. This logical translator produces translation $t(\lambda_d\varphi) = \nabla d$.

Example 3.11 Let $\lambda_{\{\top\}}$ and $\lambda_{\{\bot\}}$ be the two singleton liftings of the identity functor. The identity natural transformation $1_U : U \longrightarrow U$ is a logical translator for $\lambda_{\{\top\}}$. This logical translator produces the translation $t(\lambda_{\{\top\}}) = \nabla t(\varphi)$.

We define a logical translator for $\lambda_{\{\bot\}}$ as follows: the $\mathfrak{A}$-component is the function $\neg_{\mathfrak{A}} : A \longrightarrow A$ corresponding to negations, notice that every algebra in

$Alg(\mathcal{P}_\kappa + I)$ has one of such functions. This logical translator produces the translation $t(\lambda_{\{\bot\}}) = \nabla\neg t(\varphi)$.

Now we will show how we can combine logical translators to obtain logical translators of more complex functors. The following examples (constructions) will be used in the proof of Theorem 4.6. We start composing with the covariant power set functor.

Example 3.12 Let T be a functor and fix $P \in \mathcal{P}T(2)$. There are two facts to notice here. First fact is that $P \subseteq T(2)$, therefore for each $p \in P$ there exists a predicate lifting $\lambda_p : \check{\mathcal{P}} \longrightarrow \check{\mathcal{P}}T$ for T. The second fact is that P itself defines a singleton lifting $\lambda_P : \check{\mathcal{P}} \longrightarrow \check{\mathcal{P}}\mathcal{P}T$ for $\mathcal{P}T$.

Using example 2.27 we can prove the following. If for each $p \in P$ there exists a natural translator $\tau_p : U \longrightarrow TU$ for the predicate lifting $\lambda_p : \check{\mathcal{P}} \longrightarrow \check{\mathcal{P}}T$ for T, then the natural transformation $\tau_P : U \longrightarrow \mathcal{P}TU$, in which the $\mathfrak{A}$-component is the function $(\tau_P)_{\mathfrak{A}} : A \longrightarrow \mathcal{P}TA$ mapping an element $x \in A$ to the set $(\tau_P)_{\mathfrak{A}}(x) = \{(\tau_p)_{\mathfrak{A}}(x) \,|\, p \in P\}$, is a logical translator for the singleton lifting $\lambda_P : \check{\mathcal{P}} \longrightarrow \mathcal{P}T\check{\mathcal{P}}$ associated with P for the functor $\mathcal{P}T$. This also works for polyadic predicate liftings.

Now we will illustrate the case of coproducts.

Example 3.13 Let τ be a logical translator for an η-ary predicate lifting λ for a functor T, and let T' be any other functor, for which we can define Moss' language. By the universal property of coproducts there exists a natural transformation

$$i_T : TU \longrightarrow TU + T'U,$$

where U is the appropriate forgetful functor. Composing i_T with τ we obtain a new natural transformation $i_T\tau : (U)^\eta \longrightarrow TU + T'U$, this natural transformation happens to be a logical translator for the predicate lifting $(\lambda, \lambda_\bot)$ for $T + T'$, where $\lambda_\bot$ is the predicate lifting of Example 2.21. In other words, we extended τ via coproducts. This procedure also works for polyadic predicate liftings. Notice that if λ is a singleton lifting associated with $p \in T(2^\eta)$, then $(\lambda, \lambda_\bot)$ is the singleton lifting associated with p, but for the functor $T + T'$.

Logical translators can also be extended using products.

Example 3.14 Fix two functors T_1 and T_2 and two singleton liftings λ_{p_1} and λ_{p_2}, respectively. Assume τ_{p_1} is a logical translator for λ_{p_1}, and τ_{p_2} is a logical translator for λ_{p_2}. By the universal property of products, there exists a unique natural transformation

$$\tau = (\tau_{p_1}, \tau_{p_2}) : U \longrightarrow T_1U \times T_2U.$$

Given an algebra $\mathfrak{A}$, of the appropriate type, $\tau_{\mathfrak{A}}$ maps an element $x \in A$ to $((\tau_{p_1})_{\mathfrak{A}}(x), (\tau_{p_2})_{\mathfrak{A}}(x))$. Furthermore τ is a logical translator for the predicate lifting $\lambda_{(p_1,p_2)} : \check{\mathcal{P}} \longrightarrow \check{\mathcal{P}}(T_1 \times T_2)$. Recall that $\lambda_{(p_1,p_2)}$ is a singleton lifting for $T_1 \times T_2$ and maps a set $X \subseteq S$ to $\lambda_{(p_1,p_2)}(X) = \lambda_{p_1}(X) \times \lambda_{p_2}(X)$. This procedure still works for polyadic predicate liftings.

3.3 Logical translators vs Natural transformations

Based on the previous examples we can explain a more general phenomenon, namely that *every natural transformation* $\tau : (-)^\eta \longrightarrow T$ *defines an homonymous natural transformation* $\tau : (U)^\eta \longrightarrow TU$ *that is in fact a logical translator.* This property will become important in Section 5. A first remark is the following result.

Proposition 3.15 *Every natural transformation* $\tau : (-)^\eta \longrightarrow T$ *defines an homonymous natural transformation* $\tau : (U)^\eta \longrightarrow TU$*, where* $U : Alg(\mathcal{P}_\kappa + I) \longrightarrow Set$ *is the forgetful functor.*

Proof. Given a natural transformation $\tau : (-)^\eta \longrightarrow T$ we define an homonymous natural transformation $\tau : (U)^\eta \longrightarrow TU$ taking the same function for all algebras with the same carrier set. More explicit, given algebras $\mathfrak{A}$ and $\mathfrak{A}'$ with underlying set A we define

$$\tau_{\mathfrak{A}} = \tau_{\mathfrak{A}'} = \tau_A.$$

□

Another interesting property is presented in the following proposition.

Proposition 3.16 *Every natural transformation* $\tau : (U)^\eta \longrightarrow TU$ *is a logical translator for some polyadic predicate lifting* λ_τ*.*

Proof. Define λ_τ composing the power set restriction (Observation 3 after Definition 3.4) of τ, i.e $\tau_{\check{\mathcal{P}}} : (\check{\mathcal{P}})^\eta \longrightarrow T\check{\mathcal{P}}$, with the natural transformation associated with Moss' modality. In other words the S component of λ_τ is the function $(\lambda_\tau)_S = \nabla_S \tau_{\check{\mathcal{P}}S}$. It is clear from the construction that τ will be a logical translator for λ_τ. □

Corollary 3.17 *Every natural transformation* $\tau : (-)^\eta \longrightarrow T$ *defines an homonymous natural transformation* $\tau : (U)^\eta \longrightarrow TU$ *that is in fact a logical translator.*

4 Properties of Translators

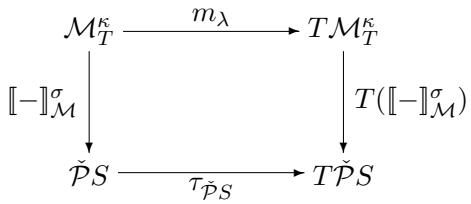
In the previous section, we defined the concept of logical translator and illustrated it with examples. Now we will show how to obtain translations from logical translators.

4.1 Translations from Translators

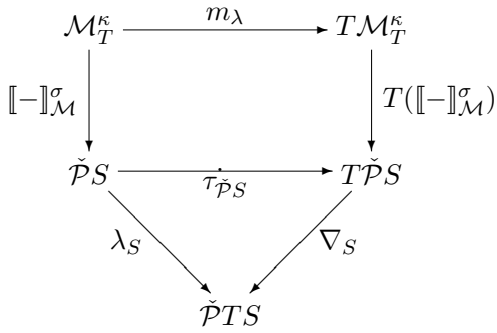
Notice that If τ is a logical translator for a predicate lifting λ, since Moss' language $\mathcal{M}_T^\kappa$ is an expansion of an algebra in $Alg(\mathcal{P}_\kappa + I)$, there exists a function $\tau_{\mathcal{M}_T^\kappa} : (\mathcal{M}_T^\kappa)^\eta \longrightarrow T\mathcal{M}_T^\kappa$.

Definition 4.1 If τ is a logical translator for a predicate lifting λ, for a functor T, the function $\tau_{\mathcal{M}_T^\kappa} : (\mathcal{M}_T^\kappa)^\eta \longrightarrow T\mathcal{M}_T^\kappa$ is called the *signature of the translator.* Since τ is a logical translator, we write m_λ instead of $\tau_{\mathcal{M}_T^\kappa}$. We do this to keep in mind the predicate lifting λ.

Another relevant observation is: since $[\![-]\!]^{\sigma}_{\mathcal{M}}$ is a morphism of L-algebras, in particular it is a morphism of $\mathcal{P}_{\kappa}+I$-algebras. Therefore, the naturality of τ implies that the following diagram



commutes, where m_{λ} is the signature of the translator. More explicitly, it says that for any formula $\psi \in \mathcal{M}^{\kappa}_{T}$ the equation $\tau_{\check{\mathcal{P}}S}[\![\psi]\!]^{\sigma}_{\mathcal{M}} = T([\![-]\!]^{\sigma}_{\mathcal{M}})m_{\lambda}(\psi)$ holds. Notice that the bottom function in the previous rectangle coincides with the function at the top of the triangle defining logical translators. Putting those two diagrams together we obtain the following diagram



This diagram commutes, for every set S, because it was constructed from two commutative diagrams. The commutativity of this last diagram implies that for every formula $\psi \in \mathcal{M}^{\kappa}_{T}$ the equation

$$\lambda_S[\![\psi]\!]^{\sigma}_{\mathcal{M}} = \nabla_S T([\![-]\!]^{\sigma}_{\mathcal{M}})m_{\lambda}(\psi) \tag{3}$$

holds. Compare this last equation with equation (2) on page 11. Notice that a similar equation still holds for polyadic predicate liftings. Readers that worry about the mix of λ with the semantic of a Moss' formula should recall that λ is a natural transformation and therefore the right side on the previous equation makes sense.

Now we can construct our function t from a logical translator τ. We will illustrate the construction proving the following result.

Theorem 4.2 *For every accessible and weak pullback preserving functor T, every predicate lifting λ for T, and every formula $\varphi \in \mathcal{L}^{\kappa}_{T}(\Lambda)$, if there exists a logical translator for λ and φ can be translated into Moss' language, then the formula $\lambda\varphi$ can be translated into Moss' language for T.*

Proof. Based on the discussion above, it is enough to define a formula $t(\lambda\varphi)$ such that $\lambda_S[\![\varphi]\!]^{\sigma}_{\mathcal{L}} = \nabla_S T([\![-]\!]^{\sigma}_{\mathcal{M}})t(\lambda\varphi)$, the translation of $\lambda\varphi$ will then be $\nabla t(\lambda\varphi)$. We

define the desired formula as follows

$$t(\lambda\varphi) = m_\lambda(t(\varphi)),$$

where m_λ is the signature of the translator and $t(\varphi)$ is a translation of φ. Now we will show that it works. By hypotheses we already know that $[\![\varphi]\!]^\sigma_{\mathcal{L}} = [\![t(\varphi)]\!]^\sigma_{\mathcal{M}}$. Using this we conclude

$$\begin{aligned}
\lambda_S[\![\varphi]\!]^\sigma_{\mathcal{L}} &= \lambda_S[\![t(\varphi)]\!]^\sigma_{\mathcal{M}} \\
&= \nabla_S T([\![-]\!]^\sigma_{\mathcal{M}})m_\lambda(t(\varphi)) && \text{(Equation (3) with } \psi = t(\varphi)) \\
&= \nabla_S T([\![-]\!]^\sigma_{\mathcal{M}})t(\lambda\varphi) && \text{(Definition of } t).
\end{aligned}$$

The two extremes of this chain are the equation we wanted to prove. This concludes our construction. □

Notice that the previous result still holds for polyadic predicate liftings. If we have enough disjunctions we can prove the following result.

Corollary 4.3 *Let T be an accessible and weak pullback preserving functor, let Λ_0 be a set of predicate liftings such that for each of its members there exists a logical translator. If $\mathcal{L}^\kappa_T(\Lambda)$ is a PL-language such that $\Lambda_0 \subseteq \Lambda$, and each predicate lifting $\lambda \in \Lambda$ can be obtained as a join of predicate liftings in Λ_0, then there exist a translation from $\mathcal{L}^\kappa_T(\Lambda)$ into $\mathcal{M}^\kappa_T$.*

Proof. We will define a function $t : \mathcal{L}^\kappa_T(\Lambda) \longrightarrow \mathcal{M}^\kappa_T$ by induction on the complexity of the formulas. The base case, i.e $\varphi = \top$ and boolean cases are done as usual. We have to show how to define t for formulas of the form $\lambda\varphi$. Since every predicate lifting in Λ can be obtained as a join of predicate liftings in Λ_0, it is enough to define the function t for each predicate lifting in Λ_0, i.e we can assume $\lambda \in \Lambda_0$. We will define $t(\lambda\varphi)$ by induction on the complexity of φ.

Assume $\varphi = \top$, we define $t(\lambda\varphi) = \nabla m_\lambda t(\varphi) = m_\lambda\top$, where m_λ is the signature of the translator for λ. The cases for conjunctions, disjunctions and negations are similar.

Assume $\varphi = \lambda'\varphi'$, where $\lambda' \in \Lambda$. Our inductive hypothesis is that the function t is already defined on φ'. By hypothesis we have $\lambda = \bigvee_{i\in I}\lambda_i$ for some predicate liftings in Λ_0, the previous theorem implies that we can translate all the formulas $\lambda_i\varphi'$, i.e. we can define the function t for each formula $\lambda_i\varphi'$. Using this and the previous theorem, we can see that defining $t(\lambda\varphi)$ to be

$$t(\lambda\varphi) = \nabla m_\lambda(\bigvee_{i\in I} t(\lambda_i\varphi')) = \nabla m_\lambda(\bigvee_{i\in I} m_{\lambda_i} t(\varphi')))$$

works. This concludes the construction of t in formulas of the form $\lambda\varphi$. We conclude that the language $\mathcal{L}^\kappa_T(\Lambda)$ can be translated into Moss' language for T. □

The previous corollary will help us in the translation of KPFs, but also consider the following situation. If we can translate a predicate lifting λ into Moss' language,

then we can also translate the predicate lifting $\neg\lambda$ into Moss' language. If we can also translate another predicate lifting λ', then we can also translate predicate liftings like $\lambda \wedge \lambda'$. In a nutshell, if we can translate a set of predicate liftings Λ_0 we can translate all boolean expressions over Λ_0, i.e the elements in the free boolean algebra over Λ_0. From this we can see that the previous corollary is a particular case of the following result.

Corollary 4.4 *Let T be an accessible and weak pullback preserving functor, let Λ_0 be a set of predicate liftings such that for each of its members there exists a logical translator. If $\mathcal{L}^\kappa_T(\Lambda)$ is a PL-language such that $\Lambda_0 \subseteq \Lambda$, and each predicate lifting $\lambda \in \Lambda$ can be obtained as a boolean expression of predicate liftings in Λ_0, then there exist a translation from $\mathcal{L}^\kappa_T(\Lambda)$ into $\mathcal{M}^\kappa_T$.*

Remark 4.5 Given a set of predicate liftings Λ, we write $\mathcal{P}_\kappa + \Lambda + I$ for the functor with factor I^η for each η-ary predicate lifting $\lambda \in \Lambda$. The conditions in the previous corollaries (requiring that each predicate lifting $\lambda \in \Lambda$ can be obtained as join or as a boolean expression, respectively, of predicate liftings in Λ_0) are important because of the following situation. If they hold, we can define a functor, which we call it the translation functor,

$$F : Alg(\mathcal{P}_\kappa + T + I) \longrightarrow Alg(\mathcal{P}_\kappa + \Lambda + I).$$

Assume $\Lambda = \Lambda_0$. We can then define F as follows: an algebra $\mathfrak{A}$ in $Alg(\mathcal{P}_\kappa + T + I)$ is mapped to an algebra $F(\mathfrak{A})$, with the same carrier set A and the following operations: it has the same conjunction and negations of $\mathfrak{A}$. Given $\lambda \in \Lambda_0$ we define $\lambda_{F(\mathfrak{A})} = \nabla_{\mathfrak{A}}\tau_{\mathfrak{A}}$, where τ is a logical translator for λ. The functor F is the identity on arrows. The fact that τ is a natural transformation $\tau : (U)^\eta \longrightarrow TU$ is what we need to prove that F is a functor. If not every predicate lifting in Λ can be expressed as a boolean expression of predicate liftings in Λ_0, in principle, we can not define the functor F.

The category $Alg(\mathcal{P}_\kappa + \Lambda + I)$ is interesting because, for a fixed set of predicate liftings Λ, the language $\mathcal{L}^\kappa_T(\Lambda)$ is the carrier set of an initial algebra in $Alg(\mathcal{P}_\kappa + \Lambda + I)$. Moreover, we can define a functor $S_T : Coalg(T)^{op} \longrightarrow Alg(\mathcal{P}_\kappa + \Lambda + I)$ analogue to the complex algebra functor (Definition 2.11) and then the satisfaction relation $[\![-]\!]^\sigma_{\mathcal{L}}$ appears as a morphism from this initial algebra to $S_T(\sigma)$.

The functor F is interesting because we are defining an appropriate algebra structure on Moss' language, actually we are defining and appropriate structure over all L algebras. More important is to notice that the translation $t : \mathcal{L}^\kappa_T(\Lambda) \longrightarrow \mathcal{M}^\kappa_T$, described on the previous corollaries, is the initial arrow from $\mathcal{L}^\kappa_T(\Lambda)$ to $F(\mathcal{M}^\kappa_T)$ in $Alg(\mathcal{P}_\kappa + \Lambda + I)$, recall that F preserves carrier sets.

Furthermore, this can be done more abstractly. Any functor $F : Alg(\mathcal{P}_\kappa + T + I) \longrightarrow Alg(\mathcal{P}_\kappa + \Lambda + I)$ such that (i) faithful, (ii) $U_M = U_{\mathcal{L}}F$ and (iii) $S_T = FM_T$, where U_M and $U_{\mathcal{L}}$ are the appropriate forgetful functors, defines a translation from the language $\mathcal{L}^\kappa_T(\Lambda)$ into Moss' language as an initial arrow. We call functor with these properties translation functors.

4.2 Extending Translators

Logical translators are particularly useful because they can be combined to obtain logical translators of more complex functors. In other words, logical translators can be extended via products, coproducts, and composition with the covariant power set functor. This is summarized in the following result.

Theorem 4.6 *For every pair of functors T_1 and T_2, if every (monadic) singleton lifting for T_i has a logical translator, with $i = 1, 2$; then every singleton lifting for $T \in \{T_1 + T_2, T_1 \times T_2, \mathcal{P}T_1\}$ has a logical translator.*

Proof. The proof of this theorem was already done along the text, it can be found in the examples. We will only point to the appropriate examples. The case of coproducts follows from Example 3.13, the case of products follows from Example 3.14, and the case of the power set functor was discussed in Examples 3.12, and 2.27.

□

The next corollary easily follows from the previous theorem.

Corollary 4.7 *For every Kripke polynomial functor T, every (monadic) singleton predicate lifting for T has a logical translator.*

Proof. The proof goes by induction on the complexity of T. Examples 3.10 and 3.11, in the previous section, show that for each singleton lifting of the base cases, the identity and constant functors, there exists a logical translator. The previous theorem implies that those logical translators can be inductively extended to any singleton lifting for a fixed KPF T. This concludes the proof. □

This together with Corollary 4.3 implies the following theorem.

Theorem 4.8 *For every Kripke polynomial functor T, if Λ_0 is the set of monadic singleton liftings for T, then we can find a translation*

$$t' : \mathcal{L}^{\omega}_{T}(\Lambda) \longrightarrow \mathcal{M}^{\omega}_{T},$$

where Λ is the set of all boolean expressions over Λ_0,

Examples 3.8 and 3.9 show that in fact we can prove the following.

Corollary 4.9 *For every $(\mathcal{B}_{\mathbb{N}}, \mathcal{D})$-Kripke polynomial functor T, if Λ_0 is the set of monadic singleton liftings for T, then we can find a translation*

$$t' : \mathcal{L}^{\omega}_{T}(\Lambda) \longrightarrow \mathcal{M}^{\omega}_{T},$$

where Λ is the set of all boolean expressions over Λ_0,

Notice that these translations are finitary, i.e. we are only using formulas of finite arity. Under the assumptions of Remarks 2.29 and 3.2 on pages 9 and 10, respectively, we can prove the following theorem.

Theorem 4.10 *Let T be a Kripke polynomial over a set D. If $|D| < \kappa$, then we can find a translation*

$$t' : \mathcal{L}_T^{\kappa}(\Lambda) \longrightarrow \mathcal{M}_T^{\kappa},$$

where Λ is the set of all monadic predicate lifting for T.

At the light of Example 3.8 we have.

Corollary 4.11 *Let T be a $\mathcal{B}_{\mathbb{N}}$-Kripke polynomial over a set D. If $max\{|D|, \aleph_0\} < \kappa$, then we can find a translation*

$$t' : \mathcal{L}_T^{\kappa}(\Lambda) \longrightarrow \mathcal{M}_T^{\kappa},$$

where Λ is the set of all monadic predicate lifting for T.

Based on Example 3.9 we have

Corollary 4.12 *Let T be a $\mathcal{D}$-Kripke polynomial over a set D. If $max\{|D|, 2^{\aleph_0}\} < \kappa$, then we can find a translation*

$$t' : \mathcal{L}_T^{\kappa}(\Lambda) \longrightarrow \mathcal{M}_T^{\kappa},$$

where Λ is the set of all monadic predicate lifting for T.

5 From Moss' language to PL-languages

We concluded our last section illustrating how to translate singleton liftings, in the case of KPFs, using logical translators. As we saw with the example of the existential modality, not only singleton liftings can be translated using logical translators. In this section, we show how we can even translate some polyadic predicate liftings using logical translators. As an interesting consequence, we will obtain a translation backwards, i.e. we will be able to translate Moss' modality. We start with a representation theorem for endofunctors in the category *Set*.

Theorem 5.1 (see [2]) *For every accessible functor T, there exists a cardinal number κ such that for each set A, there exists a surjection*

$$e_A : \sum_{\eta \in \kappa} T(\eta) \times A^{\eta} \longrightarrow T(A).$$

Moreover, these surjections constitute a natural transformation e.

Proof. This result was proved by Jiří Adámek and Vera Trnková, see [2], we will only explain how to define the functions e_A.

By the universal property of the coproduct, it is enough to define e_A in each factor. Fix a cardinal number η, and pick a pair $(p, s) \in T(\eta) \times A^{\eta}$. Notice that an element $s \in A^{\eta}$ is a function $s : \eta \longrightarrow A$, applying T to s we obtain a function $T(s) : T(\eta) \longrightarrow T(A)$, evaluating this last function on p we obtain the action of $(e_{\eta})_A$. In other words

$$(e_{\eta})_A(p, s) = T(s)(p).$$

The function e_A is the coproduct of the functions $(e_\eta)_A$. The property of accessibility is used to prove that there exists a cardinal number κ such that the function e_A is onto for each set A. □

An immediate consequence of the previous theorem is the following result.

Lemma 5.2 *Every $p \in T(\eta)$ defines a natural transformation*

$$e_p : (-)^\eta \longrightarrow T.$$

Proof. The previous theorem implies that for every η, there exists a natural transformation $e_\eta : T(\eta) \times (-)^\eta \longrightarrow T$. Fixing the first component with p we obtain a natural transformation $e_p : (-)^\eta \longrightarrow T$. This natural transformation maps an element $s \in A^\eta$ to $e_p(s) = T(s)(p)$. □

Until here, we have not addressed the issue of translating polyadic predicate liftings. We have not developed a general method to translate polyadic liftings, but using logical translators we can translate a large class of polyadic predicate liftings. At a first glance, it might seem unlikely that a formula of the form $\lambda(\varphi_1, \cdots, \varphi_\eta)$, where λ is an η-ary predicate lifting, can be translated into a formula of the form $\nabla\Psi$. The previous results imply that such polyadic liftings exists. Combining the previous lemma with Proposition 3.15 and Proposition 3.16 on page 16, we obtain the following proposition.

Proposition 5.3 *Every $p \in T(\eta)$ defines an η-ary predicate lifting λ_p associated with p, furthermore e_p is a logical translator for λ_p.*

Example 5.4 If T is the covariant power set functor and $1 = \{\top\}$, we have $\mathcal{P}(1) = \{\emptyset, \{\top\}\}$. we can easily see that $e_{\{\top\}}$ produces the natural translator $\lambda_{\{\top\}}$ of Example 3.6, furthermore $e_{\{\top\}}$ is the logical translator produced using Theorem 4.6.

Notice that not all logical translators can be obtained using this method. The logical translator associated with the existential modality can not be obtained from an element in $\mathcal{P}(1)$. This last result illustrates a method to construct polyadic predicate liftings that can be translated using logical translators. We will say no more concerning translations of polyadic predicate liftings, we will go back to this issue in the conclusions.

Now we will address the issue of translating Moss' modality. The examples in Section 3 and the previous result suggest that it might be hard to translate all formulas of type $\nabla\Psi$ using a single formula schema. The work done in this paper draws a path where a translation of Moss' modality is parametric in Ψ, i.e for each $\Psi \in T\mathcal{M}_T^\kappa$ we should find a polyadic predicate lifting λ_Ψ and a sequence of formulas $\overrightarrow{\varphi}$ such that $[\![\lambda_\Psi(\overrightarrow{\varphi})]\!]^\sigma_{\mathcal{L}} = [\![\nabla\Psi]\!]^\sigma_{\mathcal{M}}$. In fact we will prove something stronger, we will make our construction independent of the coalgebraic structure, just as it was explained in Section 3, more explicit we will construct λ_Ψ and $\overrightarrow{\varphi}$ such that

$$(\lambda_\Psi)_S(\overrightarrow{[\![\varphi]\!]^\sigma_{\mathcal{L}}}) = \nabla_S T([\![-]\!]^\sigma_{\mathcal{M}})(\Psi).$$

There is still one piece missing to construct our translation. We would like to define our translation inductively. Consider the following: given a formula $\nabla\Psi$, by definition, we know $\Psi \in T(\mathcal{M}_T^\kappa)$. Using Theorem 5.1, we may pick $\eta \in \kappa$, $p \in T(\eta)$, and a sequence of formulas $\overrightarrow{\psi} = (\psi_i)_{i\in\eta} \in (\mathcal{M}_T^\kappa)^\eta$ such that $e_{\mathcal{M}_T^\kappa}(p, \overrightarrow{\psi}) = \Psi$. The intuition is that each formula ψ_i is less complex that Ψ then we can assume it has been already translated. This idea was suggested by Yde Venema in [7] Section 5. For lack of space we can not explain his idea in detail, we will only quote his definition, a small modification for non-standard functors will work in our framework.

Definition 5.5 [see [7]] Let T be a standard finitary functor, and let X be a set of objects to be called variables. Inductively we define, for each natural number n, the set $\mathcal{M}_T^{\#n}(X)$ of *coalgebraic (Moss') formulas over X of depth n*:

- $\mathcal{M}_T^{\#0}(X)$ is the smallest set M which contains $\bot, \top$, and all variables in X and satisfy (i) if p and q belong to M, then so do $\neg p$ and $p \wedge q$.
- $\mathcal{M}_T^{\#n+1}(X)$ is the smallest superset of $\mathcal{M}_T^{\#n}$ which contains the formula $\nabla\Psi$ for each Ψ that belongs to TQ for some finite $Q \subseteq T\mathcal{M}_T^{\#n}$ and is closed under the formation rule (i).

The following holds $\mathcal{M}_T^\omega = \bigcup_{n\in\omega} \mathcal{M}_T^{\#n}(\emptyset)$.

Using this definition, in the case of standard functors, we can choose the formulas $(\psi_i)_{i\in\eta}$, defined above to be less complex that Ψ. This is done as follows: lets assume $\nabla\Psi$ has depth $n+1$, then there exists a finite set $Q \subseteq \mathcal{M}_T^{\#n}(\emptyset)$ such that $\Psi \in TQ$, applying Theorem 5.1 to TQ, we obtain our desired sequence, using the naturality of e we can see that this works. With this observation in mind we can translate Moss' modality.

Theorem 5.6 *For every accessible functor T that preserves weak pullback, there exists a cardinal number κ and a translation from $\mathcal{M}_T^\omega$ into the language $\mathcal{L}_T^\omega(\Lambda)$, where Λ contains all η-ary predicate liftings for all $\eta \in \kappa$.*

Proof. Let κ be as in Theorem 5.1, with this assumption the idea behind this proof is the same idea used in the proof of Theorem 4.2. We will inductively define a translation $t : \mathcal{M}_T^\kappa \longrightarrow \mathcal{L}_T^\kappa(\Lambda)$. We only explain how to define it in formulas of the type $\nabla\Psi$. By definition we know $\Psi \in T(\mathcal{M}_T^\kappa)$. Using Theorem 5.1 and the observation above, we may pick $\eta \in \omega$, $p \in T(\eta)$, and a sequence of formulas $\overrightarrow{\psi} = (\psi_i)_{i\in\eta} \in (\mathcal{M}_T)^{\#\eta}$ less complex than Ψ such that $e_{\mathcal{M}_T^\kappa}(p, \overrightarrow{\psi}) = \Psi$. Using this, we define t as follows

$$t(\nabla\Psi) = \lambda_p(t(\psi_i))_{i\in\eta}.$$

Now we show that this works. By inductive hypothesis we have $(\forall i \in \eta)([\![t(\psi_i)]\!]_{\mathcal{L}}^\sigma =$

$[\![\psi_i]\!]^\sigma_{\mathcal{M}}$). Using this we conclude that for all sets S we have

$$\begin{aligned}
(\lambda_p)_S([\![t(\psi_i)]\!]^\sigma_{\mathcal{L}})_{i\in\eta} &= (\lambda_p)_S([\![\psi_i]\!]^\sigma_{\mathcal{M}})_{i\in\eta} \\
&= \nabla_S(e_p)_S([\![\psi_i]\!]^\sigma_{\mathcal{M}})_{i\in\eta} && \text{(Definition of } \lambda_p\text{)} \\
&= \nabla_S T([\![-]\!]^\sigma_{\mathcal{M}})(e_p)_{\mathcal{M}^\kappa_T}(\overrightarrow{\psi}) && \text{(Naturality of } e_p \text{ over } [\![-]\!]^\sigma_{\mathcal{M}}\text{)} \\
&= \nabla_S T([\![-]\!]^\sigma_{\mathcal{M}})e_{\mathcal{M}^\kappa_T}(p, \overrightarrow{\psi}) && \text{(Definition of } e_p\text{)} \\
&= \nabla_S T([\![-]\!]^\sigma_{\mathcal{M}})(\Psi) && \text{(Definition of } \overrightarrow{\psi} \text{ and } p\text{)}.
\end{aligned}$$

This concludes the proof. □

6 Conclusions

We established a general relation between the expressive power of Moss' language and the expressive power of PL-languages. We showed that the expressive power of Moss' modality is strictly weaker than the expressive power of polyadic predicate liftings. This is stated in Theorem 5.6 where a non-constructive translation of Moss' modality is defined for any accessible weak pullback functor, and in Example 3.1 where we present a concrete example of a predicate lifting that is not expressible in Moss' language. However, the fact that Moss' modality is strictly weaker than predicate liftings does not imply that Moss' language is less expressive because the lack of expressive power of Moss' modality can be overcome in the boolean part of Moss' language, see corollaries 4.3 and 4.4.

In the backwards direction, we showed that, under appropriate assumptions, every monadic predicate lifting for a KPF can be translated using Moss' modality, see Theorems 4.9 and 4.10. As corollary of those theorems we proved that we can also add the finite multiset functor and the finite distribution functor as base cases. In the general case, we illustrated how every functor T has a large class of predicate liftings that can be translated into Moss' language; we don't know if it is possible to translate all polyadic predicate liftings.

We have presented a new technique to compare coalgebraic modal languages which is based on the use of logical translators to define translations. The key property of logical translators is that they are natural transformations. This was an essential characteristic used in the proofs of Theorem 4.2, Theorem 4.6, and Theorem 5.6.

The translation functor defined in Remark 4.5 seems to be a useful tool to disprove conjectures about expressive power. This because such conjectures could, in principle, be translated into properties concerning the categories of algebras.

Some questions of interest are left open. Given a polyadic predicate lifting λ, Theorem 4.2 provides sufficient conditions to express λ into Moss' language, a natural problem is to determine whether they are also necessary. The evidence shown in the examples suggests that if λ can be translated into Moss' language using a formula of shape $\nabla\Psi$, then this formula can be obtained using a logical translator. We conjecture that such a translation exist iff there is a logical translator

that produces it. Our intuition is that Moss' language should be better understood as a three sorted algebra. One sort of elements, one sort of subsets of the previous sort, and one sort for the image of the first sort under the functor T. The idea is that logical translators are the term representable functions and that translations can only be obtained using term representable functions. The main issue being to explain what does it mean for a function $f : A \longrightarrow TA$ to be term representable in this sorted signature.

Let Λ_0 be the set of polyadic predicate liftings, for a functor T, for which there exists a logical translator. Theorem 4.4 shows that these predicate liftings have a privileged role among predicate liftings. The exact properties of Λ_0 are still to be determined. Notice that Theorem 5.6 implies that for a κ-accessible functor T, the language $\mathcal{L}^{\kappa}_{T}(\Lambda_0)$ is expressive.

We don't know whether every predicate lifting has a logical translator. Our intuition is that this can not be the case. Since logical translators produce translations, we conjecture that a predicate lifting can have at most one logical translator modulo isomorphisms. Following the same believes, we also conjecture that the translation functor (Remark 4.5) is the unique functor, modulo isomorphisms, with the following properties: (i) faithfulness, (ii) $U_M = U_{\mathcal{L}}F$, and (iii) $S_T = FM_T$. We also do not know whether the translation of Theorem 5.6 has a more constructive presentation or can be produced as an initial arrow using some kind of translation functor.

Recent work of the author together with Alexander Kurz shows that for every cardinal η and every η-ary singleton lifting λ there exists a natural transformation $\tau : (\check{\mathcal{P}})^{\eta} \longrightarrow T\check{\mathcal{P}}$ such that $\lambda = \nabla_T \tau$. This implies that for every set of predicate liftings Λ, we can always define a functor $F : C_M \longrightarrow Alg(\mathcal{P}_\kappa + \Lambda + I)$, where C_M are the algebras under the image of Moss' functor. As a consequence we can see that the existence of a translation from the language $\mathcal{L}^{\kappa}_{T}(\Lambda)$ into Moss' language reduces to extend the functor F to a functor F' with the following properties: (1) It has as domain a subcategory of $Alg(\mathcal{P}_\kappa + T + I)$ containing Moss' language and the satisfaction relations for Moss' language. (2) It is an extension of F. (3) It satisfies the conditions (i)-(iii) in the previous paragraph. The extension of F to the functor F' is *the extension problem*. Logical translators are a maximal solution to the extension problem, i.e. they allow us to extend F to a functor with domain $Alg(\mathcal{P}_\kappa + T + I)$ (Remark 4.5). We remark that Example 3.1 shows that the extension problem is not always solvable.

Another question relates to Theorem 4.6 and was suggested by Dirk Pattinson. We showed that logical translators can be extended when the functor T is composed with the covariant power set functor, so what is so special about $\mathcal{P}$? Can we do extensions of logical translators using other functors? We conjecture that logical translators can be extended whenever the functor T is composed with a monad.

Singleton liftings seem to be interesting by their own, more research on their properties should be carried further. For example, as a consequence of the expressivity results in [6] we can show that the existence of a separating set of predicate liftings implies that the language $\mathcal{L}^{\kappa}_{T}(\Lambda_s)$, where Λ_s is the set of polyadic singleton

liftings for T up to arity κ, is expressive. Following Schröder, we notice that a separating set of predicate liftings exists iff the set Λ_s is separating.

References

[1] Jiří Adámek, *Free algebras and automata realizations in the language of categories*, Commentationes Mathematicae Universitatis Carolinae **15** (1974), pp. 589–602.

[2] Jiří Adámek and Vera Trnková, *Automata and algebras in categories*, Kluwer Academic Publishers, Norwell, MA, USA, 1990.

[3] Raúl Andrés Leal, *Expresivity of coalgebraic modal languages*, Master's thesis, ILLC, Universiteit van Amsterdam, December 2007.

[4] Lawrence S. Moss, *Coalgebraic logic.*, Ann. Pure Appl. Logic **96** (1999), no. 1-3, 277–317, Erratum published Ann.P.Appl.Log 99:241-259, 1999.

[5] Dirk Pattinson, *An introduction to the theory of coalgebras*, Course notes for NASSLLI, 2003.

[6] Lutz Schröder, *Expressivity of coalgebraic modal logic: The limits and beyond.*, FoSSaCS, 2005, pp. 440–454.

[7] Yde Venema, *Automata and fixed point logic: A coalgebraic perspective*, Information and Computation **204** (2006), pp. 637–678.

[8] ———, *Algebras and coalgebras*, in Handbook of Modal Logic, ch. 6, pp. 331–426, Elsevier B.V, 2007.

[9] James Worrell, *Terminal sequences for accessible endofunctors*, Electr. Notes Theor. Comput. Sci. **19** (1999), 39–53.