Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 336 (2018) 299–314

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Non-deterministic Effects in a Realizability Model

Niels F.W. Voorneveld [1]

Faculty of Mathematics and Physics
University of Ljubljana
Ljubljana, Slovenia

Abstract

We model non-deterministic effects for Turing computability by working in the assemblies of Kleene's first partial combinatory algebra. Two methods will be discussed, one using equivalence relations on trees and one using topological descriptions of powerdomains. We describe these models for a selection of non-deterministic paradigms: angelic, demonic, convex and probabilistic. Though the first approach has a connection to traditional non-deterministic computability, the second approach works better for combining non-determinism with recursion. We establish morphisms from the tree models to the powerdomain models, which are bijective at ground type and give isomorphisms for all but the demonic case. We also see that any of the powerdomain models can be interpreted as a sub-monad of a continuation monad.

Keywords: Non-determinism, realizability, computability, powerdomains

1 Introduction

There are many approaches to understanding computability for higher-order computation [10]. A prominent one uses the notion of *assembly* on Kleene's first partial combinatory algebra $\mathcal{K}_1$. This type of computability represents data using numerical codes and models extensional transformations of data using computable *realizers*. The category of assemblies is cartesian closed and so models higher type computability, which can accommodate computer science features like recursion at higher type [9]. In this paper we study how to model non-deterministic computation in assemblies interpreted in its various forms; angelic, demonic, convex and probabilistic. We thus model computability for higher order non-deterministic computation.

An initial attempt to do something similar was made by Taylor and Phoa in unpublished work from 1990 [12]. A modern perspective on their approach would

[1] Email: niels.voorneveld@fmf.uni-lj.si
Thanks to my PhD supervisor, Alex Simpson
This material is based upon work supported by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF under Award No. FA9550-14-1-0096

https://doi.org/10.1016/j.entcs.2018.03.029

be to view it as an instance of Power and Plotkin's theory of algebraic effects [14]. Non-determinism can be modelled via powerdomains which are abstractly given as free algebras with respect to equational or inequational theories. This approach combines readily with categories of *modest sets* which are small subcategories of assemblies that enjoy remarkable completeness properties [6,7]. Due to these completeness properties, powerdomains as free algebras are guaranteed to exist. See [1] for a related line of development using realizability over Kleene's second combinatory algebra $\mathcal{K}_2$.

The disadvantage of such abstractly defined powerdomains is that one does not a priori get an explicit description of the construction. In this paper we approach the problem from a different angle, we consider explicit constructions of a powerdomain assembly PX for any assembly X, such that we can interpret the function space $X \to PY$ as the non-deterministic functions from X to Y.

After some technical background and preliminary results, we start by defining an assembly TX of trees over an assembly X. We add various equivalence relations that interpret the trees using the different flavours of non-determinism. We have a functor sending an assembly to the resulting collection of equivalence classes on trees which forms a strong monad. Moreover, a relationship with non-determinism can be established via a connection with non-deterministic Turing machines.

Sadly, these tree models appear not to combine well with general recursion. In order to model recursion, assemblies need to be *complete*, meaning that sequences of approximations have a unique limit [9,11]. But this property seems difficult to establish for such tree models. For this reason, we study alternative models of non-determinism, inspired by topological descriptions of the familiar domain theoretic powerdomains [13,16,8]. These overcome the problem with completeness.

Moreover, we establish a relationship between the models, consisting of an embedding in one direction that is bijective at base type. These bijections at base type give isomorphisms for all but the demonic paradigm. Lastly, we look at how each of these models arise as a sub-monad of the continuation monad via theorems that can be seen as representation results for the computable powerdomains.

2 Technical background

Kleene's first partial algebra $\mathcal{K}_1$ can be defined as the set $\mathbb{N}$ together with the partial algebraic map $\mathbb{N} \times \mathbb{N} \rightharpoonup \mathbb{N}$ denoted $(a, b) \mapsto ab$, giving the result of applying the a-th Turing machine on input b and outputting the result. We write $ab \downarrow$ if this terminates with a value, and $ab \uparrow$ if the program fails to terminate. Repeated application uses left association, $abc = (ab)c$. On terms created from constants, variables and applications one has lambda abstraction with weak head reduction.

An assembly $\mathcal{A}$ on $\mathcal{K}_1$ is given by a set $|\mathcal{A}|$ together with a map $||-||_{\mathcal{A}}$ assigning to each element a of $|\mathcal{A}|$ a non-empty set of natural numbers called the *realizers* of a. A (computable) morphism $f : \mathcal{A} \to \mathcal{B}$ between assemblies is given by a map $f : |\mathcal{A}| \to |\mathcal{B}|$ such that there is an $n \in \mathbb{N}$ with the property that $\forall a \in |\mathcal{A}|, x \in ||a||_X : nx \downarrow \wedge nx \in ||f(a)||_{\mathcal{A}}$. This gives a cartesian closed category of assemblies

Asm on $\mathcal{K}_1$, where the function object $\mathcal{B}^{\mathcal{A}}$ consists of the morphisms $f : |\mathcal{A}| \to |\mathcal{B}|$ whose representations are all possible $n \in \mathbb{N}$ satisfying the above property. This category has finite products and finite co-products (pairing and summation), and:

(i) The terminal assembly $\mathbf{1}$, where $|\mathbf{1}| = \{*\}$ and $|| * ||_{\mathbf{1}} = \mathbb{N}$.

(ii) The natural numbers object $\mathcal{N}$ with $|\mathcal{N}| = \mathbb{N}$ and $||n||_{\mathcal{N}} = \{n\}$.

(iii) Finite binary sequences $\mathbf{2}^*$, where $||\alpha||_{\mathbf{2}^*} = \{\sum_{0 \le i < \text{length}(\alpha)} (\alpha_i + 1)2^{-i}\}$

We look at a specific assembly Σ, coinciding with the dominance in [15].

Definition 2.1 We take Σ such that: $|\Sigma| = \{\top, \bot\}$, with $||\top||_\Sigma = \{n \in \mathbb{N} : n0\downarrow\}$ and $||\bot||_\Sigma = \{n \in \mathbb{N} : n0\uparrow\}$

Because of the undecidability of the Halting problem, we do not have the switch morphism $\Sigma \to \Sigma$ sending $\top \mapsto \bot$ and $\bot \mapsto \top$. We get that Σ imposes a preorder, which we call the Σ-*order*, on each assembly $\mathcal{A}$ where for each $a, b \in \mathcal{A}$ we have:

$$x \le_{\mathcal{A}} y :\Leftrightarrow \forall u \in \Sigma^{\mathcal{A}}, (u(x) = \top \Rightarrow u(y) = \top)$$

For example we have $\bot \le_\Sigma \top$ but not $\top \le_\Sigma \bot$. By composition of morphisms we also have that for any $f \in \mathcal{B}^{\mathcal{A}}$ and $x \le_{\mathcal{A}} y$ we get that $f(x) \le_{\mathcal{B}} f(y)$. So all morphisms preserve the Σ-order. We have the following morphisms:

(i) $\vee : \Sigma \times \Sigma \to \Sigma$ with $t \vee r = \top \Leftrightarrow t = \top$ or $r = \top$.

(ii) $\wedge : \Sigma \times \Sigma \to \Sigma$ with $t \wedge r = \top \Leftrightarrow t = \top$ and $r = \top$.

(iii) $\bigvee : \Sigma^{\mathcal{N}} \to \Sigma$ where $\bigvee f = \top \Leftrightarrow \exists n, f(n) = \top$.

We do however **not** have a map $\bigwedge : \Sigma^{\mathcal{N}} \to \Sigma$ such that $\bigwedge f = \top \Leftrightarrow \forall n : f(n) = \top$. Such a map would need to check for an infinite number of programs whether they terminate, which one can't do in a finite amount of time. By imposing the primitive operations we have similar properties on $\Sigma^{\mathcal{A}}$ for any assembly $\mathcal{A}$. Hence $\Sigma^{\mathcal{A}}$ has the structure of a computable lattice of opens on $\mathcal{A}$ as we can interpret a map $f : \mathcal{A} \to \Sigma$ as the subset $\{a \in \mathcal{A} | f(a) = \top\}$ of $\mathcal{A}$. We call $\Sigma^{\mathcal{A}}$ the Σ-*topology* on $\mathcal{A}$, which differs from a classical topology because of its computable nature. Morphisms are automatically continuous with respect to these Σ-topologies.

Definition 2.2 Let $(-)_\bot : \mathbf{Asm} \to \mathbf{Asm}$ be the functor where $|\mathcal{A}_\bot| = |\mathcal{A}| \cup \{\bot\}$ (disjoint union), with realizers: $||a||_{\mathcal{A}_\bot} := \{n | n0\downarrow, n0 \in ||a||_{\mathcal{A}}\}$ and $||\bot||_{\mathcal{A}_\bot} = ||\bot||_\Sigma$

This functor adds a bottom element to each assembly. We get that $\forall a \in |\mathcal{A}| : \bot \le_{\mathcal{A}_\bot} a$ but not $a \le_{\mathcal{A}_\bot} \bot$. This functor forms a monad adding partiality; let $\eta^\bot_{\mathcal{A}} : \mathcal{A} \to \mathcal{A}_\bot$ denote the appropriate natural transformation. The morphisms $\mathcal{N} \to \mathcal{N}_\bot$ are precisely the partial computable functions from $\mathbb{N}$ to $\mathbb{N}$.

For an assembly $\mathcal{A}$, the *regular subobject* given by $B \subset |\mathcal{A}|$ is the assembly $\mathcal{B}$ where $|\mathcal{B}| = B$ and $||b||_{\mathcal{B}} = ||b||_{\mathcal{A}}$ for all $b \in B$.

Definition 2.3 Let ω be the *regular subobject* of $\Sigma^{\mathcal{N}}$ consisting of the morphisms $f : \mathcal{N} \to \Sigma$ satisfying:

(i) $\forall n, m \in \mathcal{N}, (m > n \wedge f(n) = \bot) \Rightarrow f(m) = \bot$

(ii) $\exists n \in \mathcal{N}, f(n) = \bot$

Let $\bar{\omega}$ be the regular subobject of $\Sigma^{\mathcal{N}}$ of elements only satisfying condition (i).

The assemblies ω and $\overline{\omega}$ are carriers respectively for the initial algebra and final co-algebra of the functor $(-)_\bot$. For $a \in \mathbb{N} \cup \{\infty\}$, we write $\widehat{a} \in \Sigma^{\mathcal{N}}$ for the function given by $\widehat{a}(n) = \top \Leftrightarrow (n < a)$. We get that $|\omega| = \{\widehat{a} | a \in \mathbb{N}\}$ and $|\overline{\omega}| = |\omega| \cup \{\widehat{\infty}\}$.

These assemblies can be seen as ascending chains in the Σ-order, since for $a, b \in \mathbb{N} \cup \{\infty\}$ we have $\widehat{a} \leq_{\overline{\omega}} \widehat{b}$ precisely when $a \leq b$. The topology on $\overline{\omega}$ also induces some interesting *continuity* properties, as for any $f \in \Sigma^{\overline{\omega}}, f(\widehat{\infty}) = \top \Leftrightarrow \exists \widehat{a} \in \omega, f(\widehat{a}) = \top$.

A map $f : \overline{\omega} \to \mathcal{A}$ can be seen as an approximation of an element in $\mathcal{A}$, where the sequence $\{f(\widehat{n})\}_{n \in \mathbb{N}}$ approximates the limit $f(\widehat{\infty})$. Take $\mathcal{A}^{i_\omega} : \mathcal{A}^{\overline{\omega}} \to \mathcal{A}^{\omega}$ to be the morphism composing maps with the inclusion from ω to $\overline{\omega}$. The following property of an assembly $\mathcal{A}$ implements the completeness property: that every sequence in $\mathcal{A}$ has a unique limit.

Definition 2.4 An assembly $\mathcal{A}$ is *complete* if $\mathcal{A}^{i_\omega}$ is an isomorphism.

So for any sequence $f : \omega \to \mathcal{A}$ on a complete assembly, we can compute its unique limit $(\mathcal{A}^{i_\omega})^{-1}(f)(\widehat{\infty})$. Completeness is an important property, as it is used to derive computable fixed point operators. By [9] we know that the assemblies $\mathbf{1}$ and $\mathcal{N}$ are complete, and that completeness is preserved under $(-)_\bot$. For any $\mathcal{A}$, completeness is also preserved under $(-)^{\mathcal{A}}$, as we can take the completion pointwise: we consider $F \in (X^{\mathcal{A}})^{\omega}$ as a morphism $\mathcal{A} \to X^{\omega}$ and use completeness of X.

We will investigate assemblies of the form $\Sigma^{\Sigma^{\mathcal{A}}}$ later on in the paper, so we end this chapter with some properties about them.

Lemma 2.5 *For $u, v \in \Sigma^{\mathcal{A}}$ we have $\forall a \in \mathcal{A}, u(a) \leq_\Sigma v(a)$ if and only if $u \leq_{\Sigma^{\mathcal{A}}} v$.*

Proof. Assume $\forall a \in \mathcal{A}, u(a) \leq_\Sigma v(a)$. Let $c : \Sigma \to \Sigma^{\mathcal{A}}$ be $t \mapsto (a \mapsto (u(a) \vee (t \wedge v(a))))$, then $c(\bot) = u$ and $c(\top) = v$. For $Q \in \Sigma^{\Sigma^{\mathcal{A}}}$, we have a map $t \mapsto Q(c(t)) : \Sigma \to \Sigma$. This map must preserve $\leq_\Sigma$, so $Q(u) \leq_\Sigma Q(v)$. For the converse, we can use for $a \in \mathcal{A}$ the evaluation map $w \mapsto w(a)$ in $\Sigma^{\Sigma^{\mathcal{A}}}$ to show that $u(a) \leq_\Sigma v(a)$. □

We write $\chi_V \in \Sigma^{\mathcal{N}}$ for the membership test on $V \subset \mathbb{N}$, so $\chi_V(n) = \top \Leftrightarrow n \in V$. These exist for all finite V's since we have decidable equality on $\mathcal{N}$. The following result is essentially the Rice-Shapiro theorem from computability theory.

Lemma 2.6 *For $Q \in \Sigma^{\Sigma^{\mathcal{N}}}$ and $u \in \Sigma^{\mathcal{N}}$ such that $Q(u) = \top$, there is some finite set $V \subset \mathcal{N}$ such that $\forall n \in V, u(n) = \top$ and $Q(\chi_V) = \top$.*

Proof. Consider the map $t : \overline{\omega} \to \Sigma$ given by $\hat{a} \mapsto Q(u \wedge \hat{a})$. We have that $t(\hat{\infty}) = Q(u \wedge \hat{\infty}) = Q(u) = \top$. By continuity, there is an $n \in \mathbb{N}$ such that $t(\hat{n}) = \top$. So for the finite set $V := \{m \in \mathcal{N} | m < n, u(m) = \top\}$ we have $\chi_V = u \wedge \hat{n}$, and $Q(\chi_V) = t(\hat{n}) = \top$. Hence V has the desired properties. □

Corollary 2.7 *With $Q, R \in \Sigma^{\Sigma^{\mathcal{N}}}$, if for all finite sets $V \subset \mathbb{N}$ we have $Q(\chi_V) = R(\chi_V)$ then $Q = R$. So $\Sigma^{\Sigma^{\mathcal{N}}}$ is 'characterised' by these χ_V.*

3 Monads of trees

We think of non-deterministic computations as a computation with a series of binary choices until it outputs a result or goes into a loop. We can represent this structure using binary trees. A *tree* on $\mathcal{A}$ is a binary tree whose leaves are labelled with elements from $\mathcal{A}_\perp$ and which may have infinite branches. Each node or leaf of a tree has a *location*, a binary sequence $\alpha \in \mathbf{2}^*$ which describes a path to a node with a series of left and right instructions. We say a tree t on $\mathcal{A}$ is *constructed* by a morphism $f : \mathbf{2}^* \to (\mathcal{A}+\mathbf{1})_\perp$ if for any sequence $\alpha \in \mathbf{2}^*$ and any $\beta \in \mathbf{2}^*$ extending or equal to α (written as $\alpha \sqsubseteq \beta$, α is a subsequence of β) we have:

(i) If t has a node (not a leaf) at α, then $f(\alpha) = \eta^\perp_{(\mathcal{A}+\mathbf{1})}(inr(*))$

(ii) If t has a leaf $\perp$ at α, then $f(\beta) = \perp_{(\mathcal{A}+\mathbf{1})}$

(iii) If t has a leaf $a \in |\mathcal{A}|$ at α, then $f(\beta) = \eta^\perp_{(\mathcal{A}+\mathbf{1})}(inl(a))$

Note that a morphism constructing a computable tree is necessarily unique. Any tree which is constructed by some morphism is called *computable*. The *assembly of trees* on $\mathcal{A}$, denoted $T\mathcal{A}$, consists of computable trees on $\mathcal{A}$ which are represented by realizers of the unique morphism that constructs them. We can represent the algebraic operator $or : T\mathcal{A} \times T\mathcal{A} \to T\mathcal{A}$ that simply joins two trees t and r into a single tree, which has a base node which branches into both t and r. For a morphism $m : \mathcal{A} \to \mathcal{B}$, let $Tm : T\mathcal{A} \to T\mathcal{B}$ be the morphism that replaces the leaves of a tree $t \in T\mathcal{A}$ with $(m)_\perp(t)$. So $T(-)$ forms an endofunctor on **Asm**.

Lemma 3.1 *The endofunctor $T(-)$ on* ***Asm*** *forms a strong monad.*

Proof. For each $\mathcal{A}$ we have a morphism $\eta^T_\mathcal{A}$ sending $x \in \mathcal{A}$ to the trivial tree only having a single leaf x. It is realised by the code for $\lambda ab.a$.

The morphism $\mu^T_\mathcal{A} : TT\mathcal{A} \to T\mathcal{A}$ is given by replacing leaves with the trees within them. To see it is computable, consider that given a morphism $f \in ((\mathcal{A}+\mathbf{1})^{\mathbf{2}^*}_\perp + \mathbf{1})^{\mathbf{2}^*}_\perp$ and $\alpha \in \mathbf{2}^*$ one can search for the smallest subsequence $\beta \sqsubseteq \alpha$ such that $f(\beta) = \eta^\perp(\text{inl}(g))$ with $g \in (\mathcal{A}+\mathbf{1})^{\mathbf{2}^*}_\perp$ and then for the smallest sequence γ such that α extends the concatenation $\beta\gamma$ and $g(\gamma) = \eta^\perp(\text{inl}(a))$ for some $a \in \mathcal{A}$. We can construct a computation that returns $\eta^\perp(\text{inr}(a))$ if it finds this a, returns $\perp$ if it got into a loop and $*$ if either $f(\alpha) = \eta^\perp(\text{inr}(*))$ or $g(\gamma) = \eta^\perp(\text{inr}(*))$ for all checked γ in the procedure. The resulting computation represents $\mu^T_\mathcal{A}$ as a map from $((\mathcal{A}+\mathbf{1})^{\mathbf{2}^*}_\perp + \mathbf{1})^{\mathbf{2}^*}_\perp$ to $(\mathcal{A}+\mathbf{1})^{\mathbf{2}^*}_\perp$, the morphisms constructing $TT\mathcal{A}$ and $T\mathcal{A}$.

Strength can be established by defining for any two assemblies $\mathcal{A}$ and $\mathcal{B}$, the map $t^T_{\mathcal{A},\mathcal{B}} : \mathcal{A} \times T\mathcal{B} \to T(\mathcal{A}\times\mathcal{B})$ by sending (a,t) to the tree created by replacing each leaf l of t with (a,l). This map is computable since it uses some elementary morphisms like pairing and currying on the morphisms constructing the trees. $\square$

Lemma 3.2 *$T\mathcal{A}$ is a carrier of the final co-algebra of $X \mapsto (X \times X + \mathcal{A})_\perp$.*

Proof. We have an isomorphism $\Phi : (T\mathcal{A} \times T\mathcal{A} + \mathcal{A})_\perp \to T\mathcal{A}$, sending $\perp$ to the trivial $\perp$-tree, $\eta^\perp(\text{inr}(a))$ to $\eta^T_\mathcal{A}(a)$ and $\eta^\perp(\text{inl}(f,g))$ to (f or g).

Let $\mathcal{B}$ be an assembly with a morphism $\Psi : \mathcal{B} \to (T\mathcal{B}\times T\mathcal{B}+\mathcal{B})_\perp$. We recursively define a map $m : (T\mathcal{B} \times T\mathcal{B} + \mathcal{B})_\perp \to T\mathcal{A}$ where: $m(\perp) =$ trivial tree with leaf $\perp$, $m(\eta^\perp(\mathrm{inl}(x, y))) = (m(\Psi(x))$ or $m(\Psi(y)))$ and $m(\eta^\perp(\mathrm{inr}(a))) =$ trivial tree with leaf a. Hence $m \circ \Psi$ is a morphism from $\mathcal{B}$ to $T\mathcal{A}$, so $T\mathcal{A}$ has the desired properties.□

The given definition has a connection with Turing machine models of non-deterministic computations. The morphisms $\mathcal{N} \to T\mathcal{N}$ correspond to a representation of non-deterministic Turing machines which on an input n will make binary choices until it outputs a value or gets into a loop. Here, no equivalences are taken into account as (0 or 1) is different from (1 or 0). So this model is quite intensional. We derive equivalence relations on $T\mathcal{A}$ according to non-deterministic paradigms.

We use the assembly of *observations* $\Sigma^{\mathcal{A}}$ which describe semi-decidable properties of leaves. For $a \in \mathcal{A}$ and $u \in \Sigma^{\mathcal{A}}$, we say that *a satisfies u* if $u(a) = \top$. For any $u \in \Sigma^{\mathcal{A}}$ we write $u_\perp \in \Sigma^{\mathcal{A}_\perp}$ for the morphism with $u_\perp(\perp) = \perp$ and $u_\perp(\eta^\perp_{\mathcal{A}}(a)) = u(a)$ for any $a \in \mathcal{A}$. Since leaves of trees from $T\mathcal{A}$ are elements of $\mathcal{A}_\perp$, they can be 'tested' by observations in $\Sigma^{\mathcal{A}_\perp}$.

Definition 3.3 We say that $t, r \in T\mathcal{A}$ are *lower-equivalent* ($t \sim_l r$) when for any $u \in \Sigma^{\mathcal{A}}$, if either t or r has a leaf a for which $u_\perp(a) = \top$ then both have a leaf satisfying $u_\perp$.

This corresponds to what is called *angelic non-determinism*, as it looks at which properties may hold for some possible sequence of choices. Another type, *demonic non-determinism* checks which properties must hold whatever the choices are. To effectively check this in a finite amount of time, one must furthermore require that the trees are finite.

Definition 3.4 We say that $f, g \in T\mathcal{A}$ are *upper-equivalent* ($f \sim_u g$) if for any $u \in \Sigma^{\mathcal{A}}$ we have, f is finite and all its leaves satisfy $u_\perp$ if and only if g is finite and all its leaves satisfy $u_\perp$.

Definition 3.5 Two trees f and g are *convex-equivalent* ($f \sim_c g$) if they are both lower and upper equivalent.

Interpreting the tree as a combination of 50-50 chances (coin-flips), we define a function $\wp : \Sigma^{\mathcal{A}} \times T\mathcal{A} \to [0, 1]$ which for (t, u) gives the probability that t satisfies $u_\perp$. More concretely, let L^t_u be the set of locations $\alpha \in \mathbf{2}^*$ at which t has a leaf satisfying $u_\perp$. Then we can define $\wp(u, t) := \Sigma_{\alpha \in L^t_u} 2^{-\mathrm{length}(\alpha)}$. Here $\mathrm{length}(\alpha)$ denotes the height of the leaf, so $2^{-\mathrm{length}(\alpha)}$ is the probability to end up in that leaf.

Definition 3.6 The trees $t, r \in T\mathcal{A}$ are *probabilistically-equivalent* ($t \sim_p r$) if for all $u \in \Sigma^{\mathcal{A}}$, $\wp(t, u) = \wp(r, u)$.

Maps $\mathcal{N} \to T_l(\mathcal{N})$ correspond to non-deterministic Turing machines which, for any input n, exhibit corresponding non-deterministic computations. The equivalence relation implements an angelic equivalence by which, for example, the computations 0 and (0 or 1) are considered equivalent. The maps $\mathcal{N} \to T_u(\mathcal{N})$ corresponds to demonic non-deterministic Turing machines where $\perp$ is considered the same as (0 or $\perp$). The convex non-deterministic and probabilistic Turing machines are represented respectively by the morphisms $\mathcal{N} \to T_c(\mathcal{N})$ and $\mathcal{N} \to T_p(\mathcal{N})$. So each of the tree models can be seen as modelling Turing computability for a particular non-deterministic paradigm.

So for any of the notions of equivalence $i \in \{l, u, c, p\}$ we can define an endofunctor $T_i(-)$ on **Asm** given by $T(-)/\equiv_i$. The equivalences are preserved under Kleisli lifts ($t \equiv r \Rightarrow f^*(t) \equiv f^*(r)$) as will be illustrated later. They also preserve the algebraic operator *or*. However, we do not know how to find a cartesian closed subcategory closed under T_i that only contains complete objects.

4 Topologically derived powerdomains

In order to get a model that combines non-determinism with completeness, we look towards complete powerdomains in domain theory. There are many descriptions of powerdomains, each sensitive to the domain theoretical setting. We want to choose the description that suits our setting best.

In **Asm**, we have structures that can be used to synthetically define a form of computable topology. In particular, Σ has the structure of the Sierpiński space $\mathbb{S}$, and $\Sigma^{\mathcal{A}}$ is a computable analogue to the lattice of open sets on $\mathcal{A}$, just like in topology opens of X can be given by continuous maps $X \to \mathbb{S}$.

Described in [16], the lower powerdomain on some preorder X is the set of completely prime upwards closed opens on the lattice of open sets $\mathcal{O}(X)$. In [3] this space has been proven to be equivalent to the continuous maps $\mathcal{O}(X) \to \mathbb{S}$ which preserve top, bottom and binary joins. We can use morphisms $\Sigma^X \to \Sigma$, which are automatically continuous with respect to the Σ-topologies, to describe these maps. We define our computable lower powerdomain as follows:

Definition 4.1 The *lower-powerdomain* of $\mathcal{A}$, written $P_l(\mathcal{A})$, is the regular subobject of $\Sigma^{\Sigma^{\mathcal{A}}}$ consisting of the top and bottom preserving morphisms Q satisfying the property, $\forall u, v \in \Sigma^{\mathcal{A}}, Q(u \vee v) = Q(u) \vee Q(v)$

The upper powerdomain on X from [16] can be seen as Scott-continuous filters on the lattice of open sets $\mathcal{O}(X)$. By [3] these can be described as top and bottom preserving continuous functions $\mathcal{O}(X) \to \mathbb{S}$ which preserve finite meets. We again use the assembly $\Sigma^{\Sigma^{\mathcal{A}}}$ to describe these maps.

Definition 4.2 The *upper-powerdomain* of $\mathcal{A}$, written $P_u(\mathcal{A})$, is the regular subobject of $\Sigma^{\Sigma^{\mathcal{A}}}$ consisting of the top and bottom preserving elements Q such that $\forall u, v \in \Sigma^{\mathcal{A}}, Q(u \wedge v) = Q(u) \wedge Q(v)$.

The convex powerdomain introduced in [13] is a construction defined on coherent continuous domains. It is the set of convex patch-compact subsets with the Egli-Milner order. An alternative description used in [3] combines the lower and upper powerdomain. This model uses the topological space $\mathbb{A} = \{\bot, I, \top\}$ whose non-trivial opens are $\{I, \top\}$ and $\{\top\}$. The powerdomain is then described as the space of continuous top and bottom preserving maps $c : \mathcal{O}(X) \to \mathbb{A}$ with the properties: $\forall A, B \in \mathcal{O}(X) : c(A) = \bot \Rightarrow c(B) = c(A \cup B)$ and $c(A) = \top \Rightarrow c(B) = c(A \cap B)$.

To describe $\mathbb{A}$ in **Asm**, we use the assembly $\Phi := \Sigma_\bot$, renaming the elements of $|\Phi|$ as $\{\bot, I, \top\}$ such that $\bot \leq_\Phi I \leq_\Phi \top$. This way, the assembly $\Phi^{\Sigma^{\mathcal{A}}}$ describes continuous maps $\mathcal{O}(\mathcal{A}) \to \mathbb{A}$.

Definition 4.3 The *convex-powerdomain* over $\mathcal{A}$ is the subobject $P_c(\mathcal{A})$ of $\Phi^{\Sigma^{\mathcal{A}}}$ made of top and bottom preserving elements Q such that for all $u, v \in \Sigma^{\mathcal{A}}$:

(i) $Q(u) = \bot \Rightarrow Q(u \vee v) = Q(v)$

(ii) $Q(u) = \top \Rightarrow Q(u \wedge v) = Q(v)$

The probabilistic powerdomain on a space X as described in [8] is represented using valuations. These are maps $\mu : \mathcal{O}(X) \to [0,1]$ assigning to each open of X a non-negative value that denotes the probability of that set. They satisfy the properties that for any $A, B \in \mathcal{O}(X)$, $\mu(\emptyset) = 0$, $\mu(A) + \mu(B) = \mu(A \cup B) + \mu(A \cap B)$ and $A \subset B \Rightarrow \mu(A) \leq \mu(B)$. The probabilistic powerdomain consists of those valuations for which $\mu(X) = 1$, hence these μ are both top and bottom preserving.

In our setting, we use the left computably enumerable (c.e.) real numbers in the interval $[0,1]$ described in [4]. A real number x is as such if the set $\{(p,q) \in \mathcal{N}^2 | \frac{p}{q} < x\}$ is semi-decidable. We define the assembly $\overline{[0,1]}$ of left c.e. real numbers realised by those $r \in \mathbb{N}$ such that $r(\text{pair } p\ q) \downarrow \Leftrightarrow \frac{p}{q} < x$. For this assembly we have a computable operation $\oplus : \overline{[0,1]} \times \overline{[0,1]} \to \overline{[0,1]}$ that sends (x,y) to $(x+y)/2$.

Definition 4.4 The *probabilistic powerdomain* $P_p(\mathcal{A})$ on $\mathcal{A}$ is the regular subobject of $\overline{[0,1]}^{\Sigma^{\mathcal{A}}}$ of top and bottom preserving Q satisfying for all $u, v \in \Sigma^{\mathcal{A}}$,

$$Q(u) \oplus Q(v) = Q(u \vee v) \oplus Q(u \wedge v)$$

Proposition 4.5 *For any assembly $\mathcal{A}$, we have that $P_l(\mathcal{A})$, $P_u(\mathcal{A})$, $P_c(\mathcal{A})$ and $P_p(\mathcal{A})$ are all complete.*

Proof. We know that Σ and $\Phi = \Sigma_\bot$ are complete. For the assembly of left c.e. reals $\overline{[0,1]}$, consider $f : \overline{\omega} \to \overline{[0,1]}$. By continuity we must have that $f(\widehat{\infty}) > p/q$ if and only if there is an $\widehat{a} \in \omega$ such that $f(\widehat{a}) > p/q$. Also, if $f(\widehat{a}) > p/q$ then $f(\widehat{\infty}) > p/q$. So $f(\widehat{\infty}) = \lim_{a \to \infty} f(\widehat{a})$. Hence there is only one completion of maps $g : \omega \to \overline{[0,1]}$ constructed by taking defining $g' : \overline{\omega} \to \overline{[0,1]}$ as $g'(\widehat{a}) > p/q$ precisely when $\bigvee_n (\hat{a}(n) \wedge g(\widehat{n}) > p/q) = \top$. So $\overline{[0,1]}$ is complete, and hence by closure properties so are $\Sigma^{\Sigma^{\mathcal{A}}}$, $\Phi^{\Sigma^{\mathcal{A}}}$ and $\overline{[0,1]}^{\Sigma^{\mathcal{A}}}$.

For the algebraically defined regular sub-objects, we can use the following trick. With $P_l(\mathcal{A}) \subset \Sigma^{\Sigma^{\mathcal{A}}}$, we consider the equaliser of the two maps $Q \mapsto \lambda uv.(Q(u) \vee Q(v))$ and $Q \mapsto \lambda uv.Q(u \vee v)$ from $\Sigma^{\Sigma^{\mathcal{A}}}$ to $\Sigma^{\Sigma^{\mathcal{A}} \times \Sigma^{\mathcal{A}}}$. This assembly is complete since it is the equaliser on a complete assembly. $P_l(\mathcal{A})$ is the regular sub-object of this equaliser, defined by the bottom preservation condition. However for $f : \overline{\omega} \to \Sigma^{\Sigma^{\mathcal{A}}}$ we have by continuity that $f(\widehat{\infty})(\lambda a.\bot) = \bot$ only if $f(\widehat{a})(\lambda a.\bot) = \bot$ for all $\widehat{a} \in \omega$, and $f(\widehat{\infty})(\lambda a.\top) = \top$ only if there is an $\widehat{a} \in \omega$ with $f(\widehat{a})(\lambda a.\top) = \bot$. So top and bottom preservation is conserved in the ω-limits, hence $P_l(\mathcal{A})$ is complete.

$P_u(\mathcal{A})$ and $P_p(\mathcal{A})$ go similarly, as both are top and bottom preserving regular sub-objects of algebraically defined equalisers on complete assemblies.

For the convex case we need to be more careful. For $f' : \omega \to \Phi^{\Sigma^{\mathcal{A}}}$ we can always find the unique completion $f : \overline{\omega} \to \Phi^{\Sigma^{\mathcal{A}}}$ argument-wise. So we only need to check that if $f(\widehat{a}) \in P_c(\mathcal{A})$ for all $\widehat{a} \in \omega$, we have $f(\widehat{\infty}) \in P_c(\mathcal{A})$. Firstly, $f(\widehat{\infty})(u) = \bot$ iff $f(\widehat{a})(u) = \bot$ for all $\widehat{a} \in \omega$, hence $f(\widehat{a})(v) = f(\widehat{a})(u \vee v)$. So we have by continuity that $f(\widehat{\infty})(v) = f(\widehat{\infty})(u \vee v)$. For the other property, note that if $f(\widehat{\infty})(u) = \top$, then there is an $\widehat{a} \in \omega$ such that $f(\widehat{a})(u) = \top$. So $f(\widehat{b})(v) = f(\widehat{b})(u \wedge v)$ for any $\hat{b} \in \omega$ with $b \geq a$. Hence $f(\widehat{\infty})(v) = f(\widehat{\infty})(u \vee v)$. So $P_c(\mathcal{A})$ is complete as well. □

5 Interpreting the trees

We have given two descriptions for each of our considered paradigms of non-determinism, one by establishing equivalences between trees and one using topological descriptions of power domains. In this section we will establish that we can computably interpret trees from the former description as elements in the latter. This will also allow us to computably test the equivalence classes on trees, such that we can lift a function $f : \mathcal{A} \to T_i\mathcal{B}$ to the Kleisli lift $f^* : T_i\mathcal{A} \to T_i\mathcal{B}$.

The lower equivalence relation checks for which observations $u \in \Sigma^{\mathcal{A}}$ there is a leaf a such that $u_\bot(a) = \top$. We want to package this information into a single top and bottom preserving element of $\Sigma^{\Sigma^{\mathcal{A}_\bot}}$. We define $\exists_{\mathcal{A}} : T(\mathcal{A}) \to \Sigma^{\Sigma^{\mathcal{A}_\bot}}$:

$$\exists_{\mathcal{A}}(t)(u) = \top \Leftrightarrow (u(\bot) = \top) \text{ or } (t \text{ has a leaf } a \text{ such that } u(a))$$

We give an explanation as to why this map is a computable morphism. Let $f : \mathbf{2}^* \to (\mathcal{A} + \mathbf{1})_\bot$ be the morphism constructing the tree t. We use the obvious morphism $\gamma : \mathcal{A}_\bot \to (\mathcal{A} + \mathbf{1})_\bot$. Note that $a \in \mathcal{A}_\bot$ is a leaf of t at α if and only if $f(\beta) = \gamma(a)$ for any β extending α. This means there is a 1-1 correspondence between leaves of t and the set 'image$(f) \cap$image(γ)'. Let $(\mathring{-}) : \Sigma^{\mathcal{A}_\bot} \to \Sigma^{(\mathcal{A}+\mathbf{1})_\bot}$ be such that $\mathring{u}(\eta^\bot(\mathrm{inr}(*))) = \bot$ and $\mathring{u}(\gamma(a)) = u(a)$. To check the different leaves of a tree t, we use the enumeration $e : \mathcal{N} \to \mathbf{2}^*$ of binary sequences to enumerate the domain of f. So we can find a representation of $\exists_{\mathcal{A}}$ using a morphism on the maps that construct the trees: $f \mapsto \lambda u : \Sigma^{\mathcal{A}_\bot}.u(\bot) \vee \bigvee(\mathring{u} \circ f \circ e) : (\mathcal{A} + \mathbf{1})_\bot^{\mathbf{2}^*} \to \Sigma^{\Sigma^{\mathcal{A}_\bot}}$

As an immediate consequence of the definition of $\exists_{\mathcal{A}}$, we have that for any two trees $t, r \in T\mathcal{A}$: $\exists_{\mathcal{A}}(t) = \exists_{\mathcal{A}}(r)$ if and only if $t \equiv_l r$, and $\exists_{\mathcal{A}}(t \text{ or } r) = \exists_{\mathcal{A}}(t) \vee \exists_{\mathcal{A}}(r)$. The following establishes that $\exists_{\mathcal{A}}$ maps into $P_l(\mathcal{A}_\bot)$.

Lemma 5.1 *For any $t \in T\mathcal{A}$ we have $\exists_{\mathcal{A}}(t) \in P_l(\mathcal{A}_\bot)$*

Proof. For $u \in \Sigma^{\mathcal{A}_\bot}$ the top element, $\exists_{\mathcal{A}}(t)(u) \geq_\Sigma u(\bot) = \top$. If u is the bottom element, then $\exists_{\mathcal{A}}(t)(u) = \bot \vee \bigvee(\lambda n.\bot) = \bot$. So top and bottom are preserved.

Taking $u, v \in \Sigma^{\mathcal{A}_\bot}$ both not the top element, then $u \vee v$ is not the top element either. We have $\exists_{\mathcal{A}}(f)(u \vee v) = \top$ if and only if there is a leaf a of t such that $(u \vee v)(a) = \top$. So either a is a leaf such that $u_\bot(a) = \top$ or it is a leaf such that $v_\bot(a) = \top$. Such a leaf exists if and only if $\exists_{\mathcal{A}}(t)(u) \vee \exists_{\mathcal{A}}(t)(v) = \top$. □

We can conclude that $\exists_{\mathcal{A}}$ can be seen as an injective map from $T_l(\mathcal{A})$ to $P_l(\mathcal{A}_\bot)$, meaning it can be factored through an injective morphism $\exists'_{\mathcal{A}} : T_l(\mathcal{A}) \to P_l(\mathcal{A}_\bot)$.

For the upper case, we define $\forall_{\mathcal{A}} : T\mathcal{A} \to \Sigma^{\Sigma^{\mathcal{A}_\perp}}$ using the description of the upper-equivalence relation and adding top preservation:

$$\forall_{\mathcal{A}}(t)(u) = \top \Leftrightarrow (u(\perp) = \top) \text{ or } (t \text{ is finite and all leaves satisfy } u)$$

In the category **Asm**, a binary tree is finite if and only if there is some maximal length to its branches. Our enumeration $e : \mathcal{N} \to \mathbf{2}^*$ can be defined such that it enumerates the sequences in order of length, so we can use $\bigvee_n \bigwedge_{2^n-1 \leq m < 2^{n+1}-1} (\mathring{u} \circ f \circ e(m))$ to check whether for $f \in (\mathcal{A} + \mathbf{1})_\perp^{\mathbf{2}^*}$ there is some n such for all binary sequences α of length n we have $\mathring{u}(f(\alpha)) = \top$. This is true for f precisely when the tree it constructs is finite and all its leaves satisfy u. Note that $\bigwedge_{2^n-1 \leq m < 2^{n+1}-1}$ uses a finite combination of $\wedge$ operations, hence it is computable. So we can computably represent $\forall_{\mathcal{A}}$ using this construction, making it a computable morphism for any $\mathcal{A}$.

We have by definition that $\forall_{\mathcal{A}}(t) = \forall_{\mathcal{A}}(r)$ if and only if $t \equiv_u r$, and $\forall_{\mathcal{A}}(t \text{ or } r) = \forall_{\mathcal{A}}(t) \wedge \forall_{\mathcal{A}}(r)$. Mimicking the proof of 5.1 we also get the following,

Lemma 5.2 *For all $t \in T\mathcal{A}$ we have $\forall_{\mathcal{A}}(t) \in P_u(\mathcal{A}_\perp)$*

So $\forall_{\mathcal{A}}$ can also be seen as an injective map from $T_u(\mathcal{A})$ to $P_u(\mathcal{A})$.

For the convex case, we want to combine the upper and lower case. For this purpose we define the following map, $\diamond : \Phi \times \Phi \to \Phi$ as:

$$a \diamond b = \begin{cases} \top & \text{if } a = b = \top \\ \perp & \text{if } a = b = \perp \\ I & \text{otherwise} \end{cases}$$

Let $\phi : \Sigma \to \Phi$ be the map preserving top and bottom. Using these we simply define $\exists\forall_{\mathcal{A}} := (\phi \circ \exists_{\mathcal{A}}) \diamond (\phi \circ \forall_{\mathcal{A}}) : T\mathcal{A} \to \Phi^{\Sigma^{\mathcal{A}_\perp}}$. Note that $\forall_{\mathcal{A}} \leq_{\Sigma^{\Sigma \mathcal{A}_\perp}} \exists_{\mathcal{A}}$, which intuitively means: if all leaves satisfy $u_\perp \in \Sigma^{\mathcal{A}_\perp}$ then certainly there is a leaf that satisfies $u_\perp$. One can see that $\exists\forall_{\mathcal{A}}$ contains all the information of both $\exists_{\mathcal{A}}$ and $\forall_{\mathcal{A}}$.

Since it uses a combination of $\exists_{\mathcal{A}}$ and $\forall_{\mathcal{A}}$, we have that $\exists\forall_{\mathcal{A}}(t) = \exists\forall_{\mathcal{A}}(r)$ if and only if $t \equiv_l r$ and $t \equiv_u r$, which is precisely when $t \equiv_c r$. Moreover, $\exists\forall_{\mathcal{A}}(t \text{ or } r) = \exists\forall_{\mathcal{A}}(t) \diamond \exists\forall_{\mathcal{A}}(r)$. This map again has its image within the appropriate powerdomain.

Lemma 5.3 *The image of $\exists\forall_{\mathcal{A}}$ is included in $P_c(\mathcal{A}_\perp)$.*

Proof. Top and bottom preservation properties are carried over from $\exists_{\mathcal{A}}$ and $\forall_{\mathcal{A}}$. Let $t \in T\mathcal{A}$ and take $u, v \in \Sigma^{\mathcal{A}_\perp}$ two observations which are not $\lambda x.\top$. If $\exists\forall_{\mathcal{A}}(t)(u) = \perp$, then $\exists_{\mathcal{A}}(f)(u) = \perp$ hence there is no leaf of t satisfying u. So any leaf satisfying $(u \vee v)$ must satisfy v, hence $\exists_{\mathcal{A}}(t)(u \vee v) = \exists_{\mathcal{A}}(t)(v)$ and $\forall_{\mathcal{A}}(t)(u \vee v) = \forall_{\mathcal{A}}(t)(v)$. If $\exists\forall_{\mathcal{A}}(t)(u) = \top$ we have $\forall_{\mathcal{A}}(t)(u) = \top$. This means t is finite and all its leaves satisfy u. So for any leaf we get that it satisfies $(u \wedge v)$ if and only if it satisfies v. So $\exists_{\mathcal{A}}(t)(u \wedge v) = \exists_{\mathcal{A}}(t)(v)$ and $\forall_{\mathcal{A}}(t)(u \wedge v) = \forall_{\mathcal{A}}(t)(v)$.□

We can conclude that $\exists\forall_{\mathcal{A}}$ can be seen as an injective map from $T_c(\mathcal{A})$ to $P_c(\mathcal{A}_\perp)$.

Like in the probabilistic equivalence relation, we can also interpret trees as a series of 50-50 coin flips, and define the map $\Theta_{\mathcal{A}} : T\mathcal{A} \to \overline{[0,1]}^{\Sigma^{\mathcal{A}_\perp}}$ as

$$\Theta_{\mathcal{A}}(t)(u_\perp) = \wp(u,t) \text{ for any } u \in \Sigma^{\mathcal{A}} \qquad \Theta_{\mathcal{A}}(t)(\lambda x.\top) = 1$$

Here $\wp$ is the same as the one used in 3.6, except here we consider its output to be in $\overline{[0,1]}$. Of course, not all reals are left computably enumerable. So we check whether this map forms a well-defined computable morphism. Let t be a computable tree and f be the unique morphism constructing that tree. We define the n-th approximation of $\wp(u,t)$ as: $\wp_n(u,t) := \sum_{\alpha \in L^t_u, \text{length}(\alpha) \leq n} 2^{-\text{length}(\alpha)}$. Note that for α with $\text{length}(\alpha) = m \leq n$, there are precisely 2^{n-m} binary sequences of length n extending α. Since $2^{n-m} * 2^{-n} = 2^{-m}$ we have that with f constructing t, $\wp_n(u,t) = 2^{-n}\#\{\alpha \in \mathbf{2}^* | \text{length}(\alpha) = n, \mathring{u}(f(\alpha))\}$. Hence $\wp_n(u,t)$ is left computably enumerable. We get that $\wp(u,t) = \lim_{n\to\infty} \wp_n(u,t)$ is left computably enumerable, since given $(p,q) \in \mathbb{N}^2$ we can compute $\wp(u,t) > \frac{p}{q}$ by looking for an n such that $\wp_n(u,t) > \frac{p}{q}$. Hence $\Theta_{\mathcal{A}}$ is a well-defined and computable morphism.

We have by definition that $\Theta_{\mathcal{A}}(t) = \Theta_{\mathcal{A}}(r) \Leftrightarrow t \equiv_c r$. With the following lemma, we can conclude that $\Theta_{\mathcal{A}}$ can be seen as an injective map from $T_p(\mathcal{A})$ to $P_p(\mathcal{A}_\perp)$.

Lemma 5.4 *For $t \in T\mathcal{A}$ we have that $\Theta_{\mathcal{A}}(t) \in P_p(\mathcal{A}_\perp)$*

Proof. Obviously $\Theta_{\mathcal{A}}(t)$ preserves top and bottom. For $u, v \in \Sigma^{\mathcal{A}}$, the real number $\Theta_{\mathcal{A}}(t)(u_\perp) \oplus \Theta_{\mathcal{A}}(t)(v_\perp)$ is given by two summations, one over the leaves satisfying $u_\perp$ and one over the leaves satisfying $v_\perp$. If a leaf is used in either of the summations, it satisfies $u_\perp \vee v_\perp = (u \vee v)_\perp$, and if it is used in both summations it satisfies $u_\perp \wedge v_\perp = (u \wedge v)_\perp$. So one can see that the summations used in $\Theta_{\mathcal{A}}(t)((u \vee v)_\perp) \oplus \Theta_{\mathcal{A}}(t)((u \wedge v)_\perp)$ can be created by redistributing the elements of the summations used in $\Theta_{\mathcal{A}}(t)(u_\perp) \oplus \Theta_{\mathcal{A}}(t)(v_\perp)$. Hence they approach the same value. □

Take $f : \mathcal{A} \to T_l(\mathcal{B})$ and the Kleislie lift $f^* : T\mathcal{A} \to T_l(\mathcal{B})$ via T. Assume $t \equiv_l r$, and $u \in \Sigma^{\mathcal{B}}$ such that $\exists_{\mathcal{B}}(f^*(t))(u) = \top$. Then there is a leaf a of t such that there is a leaf of $f(a)$ satisfying u. Hence $\exists_{\mathcal{A}}(t)(a \mapsto \exists_{\mathcal{B}}(f(a))(u)) = \top$, so since $t \equiv_l r$ and by injectivity we have $\exists_{\mathcal{A}}(r)(a \mapsto \exists_{\mathcal{B}}(f(a))(u)) = \top$ meaning $\exists_{\mathcal{B}}(f^*(r))(u) = \top$. We can conclude that $f^*(t) \equiv_l f^*(r)$. So f^* can be seen as a map $T_l\mathcal{A} \to T_l(\mathcal{B})$. Similarly we have Kleisli lifts for the other non-deterministic paradgims, using the results of Section 7 for the convex and probabilistic case.

6 Base case bijections

We have seen by construction that the maps $\exists_{\mathcal{A}}$, $\forall_{\mathcal{A}}$, $\exists\forall_{\mathcal{A}}$ and $\Theta_{\mathcal{A}}$ can be seen as injective maps from the tree models $T_i(\mathcal{A})$ to the powerdomain models $P_i(\mathcal{A}_\perp)$. We denote these specific injective maps as $\exists'_{\mathcal{A}}$, $\forall'_{\mathcal{A}}$, $\exists\forall'_{\mathcal{A}}$ and $\Theta'_{\mathcal{A}}$ respectively. In this section we will establish that these maps are bijections in the case that $\mathcal{A} = \mathcal{N}$, and form isomorphisms in most of the base cases.

Note that $|\Sigma^{\mathcal{A}_\perp}| = \{u_\perp | u \in \Sigma^{\mathcal{A}}\} \cup \{\lambda x.\top\}$ since $\perp \leq_{\mathcal{A}_\perp} \eta^\top_{\mathcal{A}}(a)$ for any $a \in \mathcal{A}$. Since all elements of the computable powerdomains on $\mathcal{A}_\perp$ are defined as top preserving maps on domain $\Sigma^{\mathcal{A}_\bot}$, we can conclude that these elements are completely determined by their output on observations of the form $u_\bot$ with $u \in \Sigma^{\mathcal{A}}$.

Lemma 6.1 *The map $\exists'_{\mathcal{N}} : T_l(\mathcal{N}) \to P_l(\mathcal{N}_\bot)$ is bijective and has an inverse.*

Proof. Take $Q \in P_l(\mathcal{N}_\bot)$. By 2.7, Q is completely determined by its output on the lifts of finite membership tests $(\chi_V)_\bot$. Now, since $\chi_V = \bigvee_{n\in V} \chi_{\{n\}}$ we have by the $\vee$ preservation property of Q that $Q((\chi_V)_\bot) = \bigvee_{n\in V} Q((\chi_{\{n\}})_\bot)$. So for any $R \in P_l(\mathcal{N}_\bot)$ we have that if $\forall n, Q((\chi_{\{n\}})) = R((\chi_{\{n\}}))$ then $Q = R$.

Hence for any tree t in $T\mathcal{N}$ whose leaves are precisely those n such that $Q((\chi_{\{n\}})_\bot) = \top$, we have $\exists_{\mathcal{N}}(t) = Q$. We can define the inverse with a morphism Δ from $\Sigma^{\Sigma^{\mathcal{N}_\bot}}$ to $T\mathcal{N}$ which for each n has a leaf at location $\alpha = 0^n1$ which is n if $Q((\chi_{\{n\}})_\bot) = \top$, else $\bot$. This gives a computable inverse of $\exists'_{\mathcal{N}}$. □

Lemma 6.2 *The morphism $\forall'_{\mathcal{N}} : T_u(\mathcal{N}) \to P_u(\mathcal{N}_\bot)$ is bijective, but does not have a computable inverse.*

Proof. Let $Q \in P_u(\mathcal{N}_\bot)$. By the characterisation of $\Sigma^{\Sigma^{\mathcal{N}}}$ in 2.7 we know that Q is determined by the values it outputs on inputs $(\chi_V)_\bot$ for finite sets $V \subset \mathbb{N}$. Now we have that $Q((\chi_V)_\bot) \wedge Q((\chi_W)_\bot) = Q((\chi_V \wedge \chi_W)_\bot) = Q((\chi_{V\cap W})_\bot)$, hence there is a single finite set V such that $Q((\chi_V)_\bot) = \top$ and $Q((\chi_W)_\bot) = \top$ iff $V \subset W$. So with $t \in T\mathcal{N}$ some finite tree whose leaves are precisely the elements of V, we have $\forall_{\mathcal{N}}(t) = Q$. Hence $\forall'_{\mathcal{N}}$ is bijective.

Let $\tau : \Sigma \to P_u(\mathcal{N}_\bot)$ be defined as $\tau(s)(u_\bot) := (u(0) \wedge u(1)) \vee (u(0) \wedge s)$ for any $u \in \Sigma^{\mathcal{N}}$, and $\tau(s)(\lambda x.\top) = \top$. Then $\tau(\top)((\chi_V)_\bot) = \top$ if and only if $0 \in V$ and $\tau(\bot)((\chi_V)_\bot) = \top$ if and only if $\{0,1\} \subset V$. We define a map $\rho : T_u(\mathcal{N}) \to \Sigma$ that checks for $t \in T\mathcal{N}$ whether it is finite and it has a leaf labelled 1. This is well defined since it is invariant under the upper-equivalence. If we have a computable inverse $\kappa : P_u(\mathcal{N}_\bot) \to T_u(\mathcal{N})$, then we have a map $\rho \circ \kappa \circ \tau : \Sigma \to \Sigma$. This map sends $\top$ to $\bot$ since the inverse of $\tau(\top)$ is the equivalence class of finite trees with only leaves labelled 0 (no 1 leaves), and $\bot$ is sent to $\top$ since the trees in the equivalence classes mapped to $\tau(\bot)$ are finite trees with 1-leaves. So we have the switch map, which is impossible. We must conclude that the inverse is not computable. □

In the convex case, things work out more neatly, because we have more information. Let $Q = \exists\forall_{\mathcal{N}}(t)$ and $ter \in \Sigma^{\mathcal{N}}$ be the test $(\lambda x.\top)$, then $Q(ter_\bot) = \top$ if and only if t is finite and has no $\bot$-leaves. For each $m \in \mathbb{N}$ we have the mutually exclusive tests; $I \leq_\Phi Q((\bigvee_{n\geq m} \chi_{\{n\}})_\bot)$ checking whether t has a leaf $m \in \mathcal{N}$ with $m \geq n$, and $Q((\chi_{V_{<m}})_\bot) = \top$ with $V_{<m} := \{n | n < m\}$ checking whether t is finite and all its leaves are natural numbers below m. If $Q(ter_\bot) = \top$, meaning t is finite, one of these two tests must terminate, hence making a decidable test of whether all leaves are below m. We also construct a map $sm : \Phi^{\Sigma^{\mathcal{N}}} \to \mathcal{N}_\bot$ that gives $\bot$ if $Q(ter_\bot) \neq \top$, and otherwise finds the first $n \in \mathcal{N}$ for which the now decidable check $I \leq_\Phi Q((\chi_{\{n\}})_\bot)$ holds (it is decidable since otherwise $Q((\chi_{\mathbb{N}-\{n\}})_\bot) = \top$ holds). So if t is finite without $\bot$-leaves, $sm(Q)$ will give the smallest leaf of t. We create a morphism $\Lambda : P_p(\mathcal{N}_\bot) \to T_c\mathcal{N}$ as follows, where we define what leaf or node $\Lambda(Q)$ has at certain locations if no leaves preceded that location:

(i) $\Lambda(Q)$ has leaf $sm(Q)$ at 0^{n+1} if $Q((\chi_{V_{<m}})_\perp) = \top$ holds, and a node $*$ if $I \leq_\Phi Q((\bigvee_{n \geq m} \chi_{\{n\}})_\perp)$ holds. If neither, it necessarily gives $\perp$.

(ii) $\Lambda(Q)$ has a leaf at $0^n 1$ if no leaf preceded it. The leaf will be n if $I \leq_\Phi Q((\chi_{\{n\}})_\perp)$ holds, and $sm(Q)$ if $Q((\chi_{\mathbb{N}-\{n\}})_\perp) = \top$. If neither it will be $\perp$.

From this construction we can see that $\Lambda(Q)$ gives a tree whose leaves are precisely the leaves of t, and is infinite or has a leaf $\perp$ if t is infinite.

With this map we can retrieve from Q some tree $\Lambda(Q)$ such that $\Lambda(Q) \equiv_c t$. The following lemma establishes that all $Q \in P_c(\mathcal{N})$ originate from $T\mathcal{N}$ via $\exists\forall_\mathcal{N}$.

Lemma 6.3 *The morphism $\exists\forall'_\mathcal{N}$ is bijective and hence Λ forms an inverse.*

Proof. Let $Q \in P_c(\mathcal{N}_\perp)$ and for convenience of notation let $Q' \in \Phi^{\Sigma^\mathcal{N}}$ be defined such that $Q'(u) := Q(u_\perp)$. We do a case distinction by value of $Q'(ter)$. Since for any $u \in \Sigma^\mathcal{N}$ we have $u \leq_{\Sigma^\mathcal{N}} ter$ we get $Q'(u) \leq_\Sigma Q'(ter)$. So a Q' with $Q'(ter) = \perp$ will always give $\perp$ except for the top element. Hence Q is reached by trees whose only leaves are $\perp$. Now for the other cases:

If $Q'(ter) = I$, we need a tree that is somewhere terminating but not everywhere. We have that for $u \in \Sigma^\mathcal{N}$, $Q'(u) \leq_\Sigma Q'(ter) = I$. So Q' only outputs I or $\perp$. If $Q'(u) = \perp$ we have $Q'(u \vee v) = Q'(v) = Q'(u) \vee Q'(v)$ for all $v \in \Sigma^\mathcal{N}$. If $Q'(u) = I = Q'(v)$, then by $u \leq_{\Sigma^\mathcal{N}} u \vee v$ we have $Q'(u \vee v) = I = I \vee I = Q'(u) \vee Q'(v)$. So Q' is now a lattice with $Q'(u \vee v) = Q'(u) \vee Q'(v)$, and so is Q. Taking $m : \Phi \to \Sigma$ sending $\perp \mapsto \perp$ and $I \mapsto \top$, we get that $m \circ Q$ is now in the lower powerdomain $P_l(\mathcal{N})$. Take $t := \Delta(m \circ Q)$ the infinite tree constructed in 6.1, then $\exists\forall_\mathcal{N}(t) = Q$.

If $Q'(ter) = \top$, we know by continuity that there is a finite set $V \subset \mathbb{N}$ such that $Q'(\chi_V) = \top$. Looking at all such finite sets, we know we can find a smallest one (by property (ii) of the powerdomain). Let V be as such, we have $Q'(u) = \top \Leftrightarrow \chi_V \leq_{\Sigma^\mathcal{N}} u$. Now if $Q'(u) = \perp$, then there are elements $x \in V$ such that $u(x) = \perp$. Take $W \subset V$ all those elements. So $\chi_W \wedge u = \lambda x.\perp$ and $\chi_V \leq_{\Sigma^\mathcal{N}} u \vee \chi_W$. Since $Q'(u) = \perp$, $\top = Q'(u \vee \chi_W) = Q'(\chi_W)$, so $\chi_V \leq_{\Sigma^\mathcal{N}} \chi_W$ which means $W = V$. Hence $u \wedge \chi_V = \lambda x.\perp$. Conversely, if we assume $u \wedge \chi_V = \lambda x.\perp$, then $\perp = Q'(\lambda x.\perp) = Q'(\chi_V \wedge u) = Q'(u)$. So $Q'(u) = \perp \Leftrightarrow \chi_V \wedge u = \lambda x.\perp \Leftrightarrow \forall x \in V, u(x) = \perp$. We can conclude that all values for Q' and Q are determined by this one finite set V. So the finite tree whose leaves are precisely those from V is mapped into Q by $\exists\forall_\mathcal{N}$.□

For the lower and convex case, the inverses are established by constructing a specific tree in $T\mathcal{N}$. So the inverses factor through $T\mathcal{N}$ and don't need the equivalence classes to be well defined. In the probabilistic case however, we will depend heavily on the specific realizers of elements in the powerdomain. We use the convenient decidable test on specific codes, $ab \downarrow^n$ that checks whether the a-th Turing machine applied to b terminates in n steps. In constructing a tree from some code of a powerdomain element, you can be careful by only checking a finite amount of things, so your tests are always decidable and never get you into a $\perp$-loop.

Lemma 6.4 *The morphism $\Theta'_\mathcal{N}$ is bijective and has an inverse.*

Proof. Take $Q \in P_p(\mathcal{N}_\perp)$. Let $\chi_V \in \Sigma^\mathcal{N}$ be some observation where V is a finite set.

Since $\chi_V = \bigvee_{n \in V} \chi_{\{n\}}$ and $\chi_{\{n\}} \wedge \chi_{\{m\}} = \lambda x.\bot$ if $n \neq m$, we have that $Q((\chi_V)_\bot)$ is the sum of all $Q((\chi_{\{n\}})_\bot)$ over $n \in V$. Since any $u \in \Sigma^{\mathcal{N}}$ is approached by these finite χ_V (see 2.6) we have that Q is completely determined by the values $Q((\chi_{\{n\}})_\bot)$. So we need to construct a computable tree such that the probability of ending up at a leaf n is $a_n := Q((\chi_{\{n\}})_\bot) \in \overline{[0,1]}$.

Let $r_Q \in ||Q||_{P_p(\mathcal{N}_\bot)}$, and x_e a realiser of the map $n \mapsto (\chi_{\{n\}})_\bot$. For each m, we give a lower approximation of the probabilities by checking for all $0 \leq n < m$, $0 \leq b \leq m$ and $0 < a < 2^b$ whether $r_Q(x_e\ n)(\text{pair}\ a\ 2^b) \downarrow^m$. This term will yield true for some m precisely if $a_n > \frac{a}{2^b}$. This gives us a decidable approximation of the probabilities given by a_n. Note that for higher m we always gain information, never lose it. Now we generate the tree by using the m-th approximation at height m in the tree and outputting leaves whenever one has sufficient enough information to do so. E.g. when at m we know that $a_n > \frac{a}{2^b}$ we can make sure that for $a2^{m-b}$ different binary sequences there is a leaf n at some subsequence. Using such a procedure, we can construct a tree approximating the probabilities level by level. This will give us an inverse morphism $\Theta_{\mathcal{N}}^{-1} : P_p(\mathcal{N}_\bot) \to T_p(\mathcal{N})$. □

7 Representation theorems

In the upper and lower case, our powerdomains are subobjects of $\Sigma^{\Sigma^{\mathcal{A}}}$, which has the structure of a continuation type. The functor $\Sigma^{\Sigma^{(-)}}$ forms a monad, with the natural morphisms defined as $\eta_{\mathcal{A}}^{\Sigma}(a) := (u \mapsto u(a))$ and $\mu_{\mathcal{A}}^{\Sigma}(F) := (u \mapsto F(Q \mapsto Q(u)))$. The morphism $\eta_{\mathcal{A}}^{\Sigma}(a)$ preserves both $\vee$ and $\wedge$ for any a, and if F preserves $\vee$ and/or $\wedge$ then so does $\mu_{\mathcal{A}}^{\Sigma}(F)$. Hence $P_l(-)$ and $P_u(-)$ form sub-monads of this monad.

We end this paper by constructing a similar sub-monad of a continuation monad in the convex and probabilistic case. For the first one, we define the following structure as in [3], using the assembly $\Phi := \Sigma_\bot$ with renamed elements $\{\bot, I, \top\}$.

Definition 7.1 The *continuation convex powerdomain* of $\mathcal{A}$, denoted $C_c(\mathcal{A})$, is the regular subobject of $\Phi^{\Phi^{\mathcal{A}}}$ of top and bottom preserving maps Q satisfying:

(i) $\forall u, v \in \Phi^{\mathcal{A}}, Q(u \diamond v) = Q(u) \diamond Q(v)$

(ii) $\forall u, v \in \Phi^{\mathcal{A}}, Q(u) = \bot \Rightarrow Q(u \vee v) = Q(v)$

(iii) $\forall u, v \in \Phi^{\mathcal{A}}, Q(u) = \top \Rightarrow Q(u \wedge v) = Q(v)$

Proposition 7.2 *For each $\mathcal{A}$ there is an isomorphism between $P_c(\mathcal{A})$ and $C_c(\mathcal{A})$.*

Proof. This is essentially a computable version of what is done in [3]. So here we will only give the definition of the isomorphism. Let $\pi_0, \pi_1 : \Phi \to \Sigma$ be the morphisms given by $\pi_0(a) = \top \Leftrightarrow a \geq I$ and $\pi_1(a) = \top \Leftrightarrow a > I$. The isomorphism is given by the following two maps:

$$Q \mapsto \lambda u : \Phi^{\mathcal{A}}.\pi_0(Q(\pi_0 \circ u)) \diamond \pi_1(Q(\pi_1 \circ u)) \quad : \quad P_c(\mathcal{A}) \to C_c(\mathcal{A})$$
$$Q \mapsto \lambda u : \Sigma^{\mathcal{A}}.Q(\phi \circ u) \quad : \quad C_c(\mathcal{A}) \to P_c(\mathcal{A})$$

□

Definition 7.3 The *continuation probabilistic powerdomain* of $\mathcal{A}$, denoted $C_p(\mathcal{A})$

is the regular subobject of $\overline{[0,1]}^{\overline{[0,1]}^{\mathcal{A}}}$ of top and bottom preserving maps Q such that for all $u, v \in \overline{[0,1]}^{\mathcal{A}}$ we have $Q(u \oplus v) = Q(u) \oplus Q(v)$.

The following is a computable version of the Riesz representation theorem.

Theorem 7.4 *For any $\mathcal{A}$ there is an isomorphism between $P_p(\mathcal{A})$ and $C_p(\mathcal{A})$.*

Proof. Let $\rho : \Sigma \to \overline{[0,1]}$ be the morphism such that $\rho(\bot) = 0$ and $\rho(\top) = 1$, the top and bottom elements of $\overline{[0,1]}$. We define:

$$\Omega : C_p(\mathcal{A}) \to P_p(\mathcal{A}), \quad \Omega(W)(u) = W(\rho \circ u)$$

We check that Ω is well defined: $\Omega(W)(u) \oplus \Omega(W)(v) = W(\rho \circ u) \oplus W(\rho \circ v) = W(\rho \circ u \oplus \rho \circ v) = W(\rho \circ (u \vee v) \oplus \rho \circ (u \wedge v)) = W(\rho \circ (u \vee v)) \oplus W(\rho \circ (u \wedge v)) = \Omega(W)(u \vee v) \oplus \Omega(W)(u \wedge v)$.

Using $\oplus$ we can define $\bigoplus_{m \leq 2^n} p_m := \frac{\sum_{1 \leq i \leq 2^m} p_i}{2^m}$. Let $\Upsilon : P_p(\mathcal{A}) \to C_p(\mathcal{A})$ be:

$$\Upsilon(V)(f) = \lim_{n \mapsto \infty} \bigoplus_{0 < m \leq 2^n} V(\lambda a : \mathcal{A}.(f(a) > \frac{m}{2^n}))$$

Which can be alternatively expressed as $\Upsilon(V)(f) = \lim_{n \to \infty} \sum_{1 \leq m \leq 2^n} 2^{-n} V(\lambda a : \mathcal{A}.(f(a) > \frac{m}{2^n}))$. Here the $>$ is interpreted as the morphism $\overline{[0,1]} \times \mathcal{N}^2 \to \Sigma$ sending $(f(a), m, 2^n)$ to $\top$ iff the inequality holds. The sequence of reals within the limit does not decrease as n gets higher, hence it gives a computable lower approximation of some real number. So $\Upsilon(V)(f)$ is some lower computably enumerable real in $\overline{[0,1]}$ computable from V and f. We can approximate $\Upsilon(V)(f \oplus g)$ with $\lim_{k \to \infty} \lim_{n \to \infty} \bigoplus_{0 < m \leq 2^n} V(\bigvee_{0 \leq l \leq 2^k} \lambda a : \mathcal{A}.(f(a) > \frac{l}{2^k}, g(a) > (\frac{m}{2^n} - \frac{l}{2^k})))$. So using the main property of V dividing the area under the functions f, g and $f \oplus g$ up into blocks, we get that $\Upsilon(V)(f \oplus g) > \frac{p}{q}$ if and only if $\Upsilon(V)(f) \oplus \Upsilon(V)(g) > \frac{p}{q}$. Hence $\Upsilon(V)(f \oplus g) = \Upsilon(V)(f) \oplus \Upsilon(V)(g)$. We can conclude that Υ is a well-defined computable morphism into $C_p(\mathcal{A})$.

Now to prove they are each other's inverses. Take $V \in P_p(\mathcal{A})$ and $u \in \Sigma^{\mathcal{A}}$. We have $\Omega(\Upsilon(V))(u) = \Upsilon(V)(\rho \circ u) = \lim_{n \to \infty} \sum_{1 \leq m \leq 2^n} 2^{-n} V(\lambda a : \mathcal{A}.\rho(u(a)) > \frac{m}{2^n}) = \lim_{n \to \infty} \sum_{1 \leq m \leq 2^n} 2^{-n} V(\lambda a : \mathcal{A}.u(a) \wedge (1 > \frac{m}{2^n}))$. Now, $V(\lambda a : \mathcal{A}.u(a) \wedge (1 > \frac{m}{2^n}))$ is $V(u)$ for $m < 2^n$ and 0 for $m = 2^n$. Hence the limit over n approximates $V(u)$. We can conclude that $\Omega(\Upsilon(V))(u) = V(u)$.

For $W \in C_p(\mathcal{A})$, $\Upsilon(\Omega(W))(f) = \lim_{n \to \infty} \bigoplus_{m \leq 2^n} \Omega(V)(\lambda a : \mathcal{A}.f(a) > \frac{m}{2^n}) = \lim_{n \to \infty} \bigoplus_{m \leq 2^n} V(\lambda a : \mathcal{A}.\rho(f(a) > \frac{m}{2^n})) = \lim_{n \to \infty} V(\bigoplus_{m \leq 2^n} \lambda a : \mathcal{A}.\rho(f(a) > \frac{m}{2^n}))$ which is by continuity the same as $V(\bigvee_n \bigoplus_{m \leq 2^n} \lambda a : \mathcal{A}.\rho(f(a) > \frac{m}{2^n})) = V(\lambda a : \mathcal{A}. \max\{\frac{m}{2^n} | (m, n) \in \mathcal{N}^2, \frac{m}{2^n} < f(a)\}) = V(\lambda a : \mathcal{A}.f(a)) = V(f)$.

We can conclude that Ω and Υ form an isomoprhism. $\square$

A direct result of this theorem relates to affine maps between probability spaces.

Corollary 7.5 *An isomorphism exists between $\mathcal{A} \to P_p(\mathcal{B})$ and the space of computable affine morphisms $\overline{[0,1]}^{\mathcal{B}} \underset{affine}{\longrightarrow} \overline{[0,1]}^{\mathcal{A}}$*

References

[1] Ingo Battenfeld, "Topological Domain Theory", PhD thesis, (2008)

[2] Ingo Battenfeld, *Observationally-induced Effects in Cartesian Closed Categories*, Electronic Notes in Theoretical Computer Science, **286** (2012) pp. 43-56

[3] I. Battenfeld, K. Keimel, T. Streicher, *Observationally-induced Algebras in Domain Theory*, Logical Methods in Computer Science **10(3:18)** (2014), pp. 1-26

[4] Rodney G. Downey and Denis R. Hirschfeldt, "Algorithmic Randomness and Complexity", (2010)

[5] Gerhard Gierz, Karl Heinrich Hofmann, Klaus Keimel, Jimmie Lawson, Michael Mislove, and Dana S. Scott. "Continuous Lattices and Domains". Cambridge University Press, Cambridge, (2003)

[6] J.M.E. Hyland, *A Small Complete Category*, Annals of Pure and Applied Logic, **40** (1988) pp. 135-165

[7] J.M.E. Hyland, E.P. Robinson, G. Rosolini, *The discrete objects in the effective topos*, Journal Proceedings of the London mathematical society **3(1)** (1990) pp. 1-36

[8] Claire Jones, "Probabilistic Non-determinism", PhD thesis. (1989)

[9] J. R. Longley, "Realizability Toposes and Language Semantics", PhD thesis, (1994).

[10] John Longley, Dag Normann, "Higher-Order Computability" Springer-Verlag, (2005)

[11] John Longley, Alex Simpson, *A uniform approach to domain theory in realizability models* Math. Struct. in Comp. Sci. **7(5)** (1997) pp. 469-505

[12] Wesley Phoa and Paul Taylor, *The Synthetic Plotkin Powerdomain*, Unpublished notes, (1990)

[13] G. D. Plotkin, *A Powerdomain Construction*, Siam J. Comput. **5(3),** (1976), pp. 452-487

[14] Gordon Plotkin and John Power *Algebraic Operations and Generic Effects*, J. Applied Categorical Structures, **11** (2003) pp. 69-94

[15] Giuseppe Rosolini, "Continuity and effectiveness in topoi", PhD thesis (1989)

[16] M. Smyth, *Power domains*, J. Comput. System Sci., **16** (1978), pp. 23-36

[17] M. Smyth, *Power domains and predicate transformers: a topological view*, Proc. Internat. Colloquium on Automata, Languages and Programs, **154** (1983), pp. 662-676