The Scott Model of PCF in Univalent Type Theory

Tom de Jong

University of Birmingham, United Kingdom

CCC, 5 September 2019

UNIVERSITY OF BIRMINGHAM

I would like to thank Martín Escardó for suggesting and supervising this project.

I have also benefited from Benedikt Ahrens' support and his help with UniMath.

Outline

Outline

1. PCF and the Scott Model
2. Motivations

Outline

Outline

Outline

PCF

PCF is a typed programming language with a fixed point combinator.

- PCF types:
  - ι for natural numbers;
  - $\sigma \Rightarrow \tau$ for functions.

PCF

PCF is a typed programming language with a fixed point combinator.

- PCF types:
  - ι for natural numbers;
  - $\sigma \Rightarrow \tau$ for functions.
- Examples of PCF terms:
  - $\underline{0}, \underline{1}, \underline{2}, \ldots : \iota$;
  - pred : $\iota \Rightarrow \iota$ and pred $\underline{5} : \iota$;
  - $\mathsf{fix}_\sigma : (\sigma \Rightarrow \sigma) \Rightarrow \sigma$.

PCF

PCF is a typed programming language with a fixed point combinator.

- PCF types:
  - ι for natural numbers;
  - $\sigma \Rightarrow \tau$ for functions.
- Examples of PCF terms:
  - $\underline{0}, \underline{1}, \underline{2}, \ldots : \iota$;
  - $\mathsf{pred} : \iota \Rightarrow \iota$ and $\mathsf{pred}\,\underline{5} : \iota$;
  - $\mathsf{fix}_\sigma : (\sigma \Rightarrow \sigma) \Rightarrow \sigma$.
- Examples of reduction:
  - $\mathsf{pred}\,\underline{n+1} \rhd^* \underline{n}$;
  - $\mathsf{fix}\, f \rhd^* f\,(\mathsf{fix}\, f)$.

Scott model of PCF

Scott model

- interpret PCF types as directed complete posets with a least element;
- for function types use continuous maps.

Because of the fixed point combinator, a "standard" set-theoretic interpretation will not work (i.e. one where function types are interpreted as exponentials in Set).

Dana Scott realised that, instead of sets, one should consider particular posets, namely directed complete posets with a least element, or dcpo with bottom for short.
A map between dcpos (with bottom) is continuous if it preserves directed suprema. The point is that such maps have fixed points.
The continuous maps between two dcpos with bottom form another dcpo with bottom with the pointwise ordering.

Soundness and computational adequacy

- Soundness:

$$\text{if } s \rhd^* t, \text{ then } [\![s]\!] = [\![t]\!].$$

- Computational adequacy:

$$\text{if } [\![s]\!] = [\![\underline{n}]\!], \text{then } s \rhd^* \underline{n}.$$

PCF together with $\rhd^*$ and the Scott model should work well together. This is expressed through soundness and computational adequacy.

Soundness expresses that the model (denotational semantics) respects the reduction relation (operational semantics). If a term s in PCF reduces ("computes") to a term t, then the interpretations of s and t are equal in the model.

Computational adequacy can be regarded as a partial converse to soundness. (A lack of function extensionality in PCF makes a full converse impossible.) An interesting consequence of computational adequacy is that it allows one to reason semantically about termination (reduction to a numeral) in PCF.

Main results

- Scott model of PCF in constructive predicative univalent type theory
- Soundness
- Computational adequacy
- All formalised

Why construct the Scott model in this setting?

- Test case for univalent foundations.

Propositional truncation in use, as we will see later.

Since PCF has a fixed-point combinator, it has non-termination. This is what makes a constructive type-theoretic semantics challenging.
To constructively account for the non-termination in PCF, we work with the partial map classifier monad (also known as the lifting monad) from topos theory [Koc91], which has been extended to constructive type theory by Reus and Streicher [RS99] and to univalent type theory by Knapp and Escardó [EK17; Kna18].

Coquand et al. have shown that countable choice is independent over univalent type theory [CMR17; Coq18]. We show that one can do without choice or HIITS.

Directed completeness needs to be formulated in terms of families, rather than subsets, because of the predicative framework.

Rather than setoids, we work with the identity types, as usual in univalent type theory. Quotienting the setoids is problematic as it needs choice to yield another monad [CUV17].

Why construct the Scott model in this setting?

- Test case for univalent foundations.
- Use lifting monad to account for non-termination of PCF.
  Other approaches to partiality need some form of countable choice [CUV17] or higher inductive-inductive types [ADK17].

Propositional truncation in use, as we will see later.

Since PCF has a fixed-point combinator, it has non-termination. This is what makes a constructive type-theoretic semantics challenging.
To constructively account for the non-termination in PCF, we work with the partial map classifier monad (also known as the lifting monad) from topos theory [Koc91], which has been extended to constructive type theory by Reus and Streicher [RS99] and to univalent type theory by Knapp and Escardó [EK17; Kna18].

Coquand et al. have shown that countable choice is independent over univalent type theory [CMR17; Coq18]. We show that one can do without choice or HIITS.

Directed completeness needs to be formulated in terms of families, rather than subsets, because of the predicative framework.

Rather than setoids, we work with the identity types, as usual in univalent type theory. Quotienting the setoids is problematic as it needs choice to yield another monad [CUV17].

Why construct the Scott model in this setting?

- Test case for univalent foundations.
- Use lifting monad to account for non-termination of PCF.
  Other approaches to partiality need some form of countable choice [CUV17] or higher inductive-inductive types [ADK17].
- Can have directed completeness in a predicative framework.

Propositional truncation in use, as we will see later.

Since PCF has a fixed-point combinator, it has non-termination. This is what makes a constructive type-theoretic semantics challenging.
To constructively account for the non-termination in PCF, we work with the partial map classifier monad (also known as the lifting monad) from topos theory [Koc91], which has been extended to constructive type theory by Reus and Streicher [RS99] and to univalent type theory by Knapp and Escardó [EK17; Kna18].

Coquand et al. have shown that countable choice is independent over univalent type theory [CMR17; Coq18]. We show that one can do without choice or HIITS.

Directed completeness needs to be formulated in terms of families, rather than subsets, because of the predicative framework.

Rather than setoids, we work with the identity types, as usual in univalent type theory. Quotienting the setoids is problematic as it needs choice to yield another monad [CUV17].

Why construct the Scott model in this setting?

- Test case for univalent foundations.
- Use lifting monad to account for non-termination of PCF.
  Other approaches to partiality need some form of countable choice [CUV17] or higher inductive-inductive types [ADK17].
- Can have directed completeness in a predicative framework.
- Related work [BKV09]:
  - based on Capretta's delay monad;
  - only ω-complete posets;
  - uses Coq's impredicative `Prop` universe;
  - works with setoids.

Propositional truncation in use, as we will see later.

Since PCF has a fixed-point combinator, it has non-termination. This is what makes a constructive type-theoretic semantics challenging.
To constructively account for the non-termination in PCF, we work with the partial map classifier monad (also known as the lifting monad) from topos theory [Koc91], which has been extended to constructive type theory by Reus and Streicher [RS99] and to univalent type theory by Knapp and Escardó [EK17; Kna18].

Coquand et al. have shown that countable choice is independent over univalent type theory [CMR17; Coq18]. We show that one can do without choice or HIITS.

Directed completeness needs to be formulated in terms of families, rather than subsets, because of the predicative framework.

Rather than setoids, we work with the identity types, as usual in univalent type theory. Quotienting the setoids is problematic as it needs choice to yield another monad [CUV17].

Type-theoretic framework

We work in intensional Martin-Löf Type Theory with two extensionality axioms and propositional truncation.

- For X a type with elements x, y, write $x = y$ for $\mathsf{Id}_X(x, y)$.
- We reserve $\equiv$ for the judgemental equality.

We review some of the features of this type theory that are instrumental to the development of the Scott model of PCF in this framework.
We will assume some familiarity with dependent type theory, e.g. $\prod$- and $\sum$-types.

Subsingletons and sets

Definition

A type X is a subsingleton (or proposition) if we have an element of the type

$$\text{is-a-subsingleton}(X) :\equiv \prod_{x:X} \prod_{y:X} x = y.$$

Definition

A type X is a set if the type

$$\text{is-a-set}(X) :\equiv \prod_{x:X} \prod_{y:X} \text{is-a-subsingleton}(x = y)$$

is inhabited.

There is a stratification of types in terms of the complexity of their identity types: Voevodsky's hlevels. For our development, we only need to consider two hlevels: the subsingletons and sets.
In a subsingleton, all elements are identified/equal. There is at most one element (up to $=$).

In a set, elements are identified/equal in at most one way.

Propositional truncation

For every type X, there is a proposition $\|X\|$ and a map $X \to \|X\|$, such that every map from X to a *proposition* P factors through it.

$$\begin{tikzcd}
X \arrow[rr] \arrow[dr] & & P \\
& \|X\| \arrow[ur, dashed] &
\end{tikzcd}$$

Borrowing terminology from category theory, we might call propositional truncation subsingleton reflection.

The dashed map is necessarily unique, because of function extensionality (assumed later) and the fact that P is a subsingleton.

Function and propositional extensionality

- Function extensionality asserts that

$$\left(\prod_{x:X} f(x) = g(x)\right) \to f = g$$

is inhabited for every two functions $f, g : X \to Y$.

- Propositional extensionality asserts that

$$((P \to Q) \times (Q \to P)) \to P = Q$$

is inhabited for every two propositions P and Q.

Type universes

- A type universe is a "type of types", closed under $\sum$, $\prod$ and $+$.
- We assume a tower of universes $\mathcal{U}_0 : \mathcal{U}_1 : \mathcal{U}_2 : \ldots$ with $\mathbf{0}, \mathbf{1}, \mathbf{N} : \mathcal{U}_0$.
- We write $\mathcal{U}_i \sqcup \mathcal{U}_j :\equiv \mathcal{U}_{\max(i,j)}$ and $\mathcal{U}_i^+ :\equiv \mathcal{U}_{i+1}$.
- If $A : \mathcal{U}$ and $B : \mathcal{V}$, then $\sum_{x:A} B(x), \prod_{x:A} B(x) : \mathcal{U} \sqcup \mathcal{V}$.

Formally, the universes do not contain types, but "codes" for types. That is, formally, we have universes à la Tarski.

Here, $\mathbf{0}, \mathbf{1}$ and $\mathbf{N}$ respectively denote the empty type, unit type and the natural numbers type.

What is univalent about our development?

- The univalence axiom is an extensionality axiom for type universes.
- Function and propositional extensionality are consequences of univalence. We do not need full univalence.
- Univalent type theory is about much more than the univalence axiom!

We do not need full univalence, because the types under consideration are all propositions and sets (dcpos).

Arguably, univalent type theory is much more about the concept of hlevels than about the univalence axiom.

Outline

Directed complete families

Since we work predicatively, we consider families, rather than power sets.

Definition

Let $(P, \leq)$ be a poset. A family $\alpha : I \to P$ is directed if it is inhabited (that is, $\|I\|$) and $\prod_{i,j:I} \|\sum_{k:I} \alpha_i \leq \alpha_k \times \alpha_j \leq \alpha_k\|$.

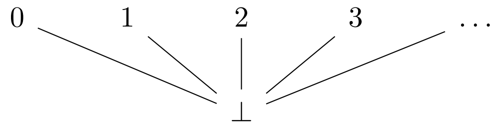
Observe that this is property, rather than structure.

Definition

A poset $(P, \leq)$ is $\mathcal{U}$-directed-complete if it has suprema for every directed family indexed by a type in $\mathcal{U}$.

Partiality, constructively

Classically, the poset



is directed complete and is used to interpret the base type ι.

But directed completeness of this poset implies LLPO, a constructive taboo.

LPO stands for Bishop's Limited Principle of Omniscience.
Type theoretically, LPO may be formulated as:

$$\prod_{\alpha:\mathbf{N}\to\mathbf{2}} \left(\prod_{n:\mathbf{N}} \alpha(n) = 0\right) + \left(\sum_{k:\mathbf{N}} \alpha(k) = 1\right).$$

Now given $\alpha : \mathbf{N} \to \mathbf{2}$, define $\beta : \mathbf{N} \to \mathbf{N} + \mathbf{1}$ by

$$\beta(n) = \begin{cases} \mathsf{inl}(k) & \text{if } k \text{ is the least number} \leq n \text{ such that } \alpha(k) = 1; \\ \mathsf{inr}(\star) & \text{else.} \end{cases}$$

Then β is directed and therefore, if $N + 1$ is directed complete, has a supremum s. But we can decide if $s = \mathsf{inl}(k)$ for some $k : \mathbf{N}$ or if $s = \mathsf{inr}(\star)$. But the former implies $\alpha(k) = 1$, while the latter implies $\prod_{n:\mathbf{N}} \alpha(n) = 0$.

The lifting of a type

Fix a type universe $\mathcal{T}$ and write $\Omega :\equiv \sum_{P:\mathcal{T}} \text{is-a-subsingleton}(P)$.

Definition

The lifting $\mathcal{L}(X)$ of a type X is defined as

$$\mathcal{L}(X) :\equiv \sum_{P:\Omega} (P \to X).$$

Definition

We can embed a type into its lifting: $\eta_X : X \to \mathcal{L}(X)$, $x \mapsto (\mathbf{1}, \lambda t.x)$.

Definition

The element $\bot :\equiv (\mathbf{0}, \text{from-empty}_X) : \mathcal{L}(X)$ represents "undefined".

With the law of excluded middle, this is all, i.e. $\mathcal{L}(X) \simeq X + \mathbf{1}$, where $\simeq$ denotes Voevodsky's notion of type equivalence.

The lifting of a set is a dcpo with bottom

Lemma

If X is a set, then so is $\mathcal{L}(X)$.

Definition

Let is-defined : $\mathcal{L}(X) \to \Omega$ be $(P, \varphi) \mapsto P$.

Definition

Let X be a set. Define a partial order $\sqsubseteq$ on $\mathcal{L}(X)$ by:

$$l \sqsubseteq m :\equiv \text{is-defined}\, l \to l = m.$$

Theorem (Knapp, Escardó)

For X a set, $(\mathcal{L}(X), \sqsubseteq)$ is $\mathcal{U}_0$-directed complete and $\bot$ is its least element.

The proof of the theorem uses [Kra+17, Theorem 5.4]: any constant map $f : X \to Y$ to a set factors through $\|X\|$.
Assume $\alpha : I \to \mathcal{L}(X)$ directed. We define

$$\bigsqcup \alpha :\equiv \left(\left\| \sum_{i:I} \text{is-defined}(\alpha_i) \right\|, \Psi \right),$$

where $\Psi : \|\sum_{i:I} \text{is-defined}(\alpha_i)\| \to X$ is such that

$$\begin{tikzcd}
\sum_{i:I} \text{is-defined}(\alpha_i) \arrow[rr, "\Phi"] \arrow[dr] & & X \\
& \|\sum_{i:I} \text{is-defined}(\alpha_i)\| \arrow[ur, "\Psi"] &
\end{tikzcd}$$

commutes and if $\alpha_i :\equiv (P_i, \varphi_i)$, then $\Phi(i, d) :\equiv \varphi_i(d)$.
This map is constant. If $(i, d_i), (j, d_j) : \sum_{i:I}$ is-defined(α_i), then to prove $\Phi(i, d_i) = \Phi(j, d_j)$, note that this is a proposition, so as α is directed, we find $k : I$ with $\alpha_i, \alpha_j \sqsubseteq \alpha_k$. But d_i : is-defined(α_i) and d_j : is-defined(α_j), so $\alpha_i = \alpha_k = \alpha_j$. Hence, $\Phi(i, d_i) = \Phi(j, d_j)$, as desired.

The lifting as a monad

Theorem (Knapp, Escardó, dJ)

Any $f : X \to \mathcal{L}(Y)$ can be extended to $f^{\#} : \mathcal{L}(X) \to \mathcal{L}(Y)$.

$$\begin{tikzcd} & \mathcal{L}(X) \arrow[d, dashed, "f^{\#}"] \\ X \arrow[ur, "\eta_X"] \arrow[r, "f"'] & \mathcal{L}(Y) \end{tikzcd}$$

This extension is continuous and $\left(\mathcal{L}, \eta, (-)^{\#}\right)$ satisfies the Kleisli triple equations.

The "monad" bumps universe levels. This is not a problem, however, as the Kleisli triple equations can be type checked and proved nonetheless.

The Scott model using the lifting monad (PCF types)

Definition (Interpreting PCF types)

The base type (for natural numbers) is interpreted as

$$[\![\iota]\!] :\equiv \mathcal{L}(\mathbf{N}).$$

The function types are interpreted using the dcpo of continuous maps:

$$[\![\sigma \Rightarrow \tau]\!] :\equiv [\![\tau]\!]^{[\![\sigma]\!]}$$

The continuous maps between two dcpos with bottom form another dcpo with bottom with the pointwise ordering.

The Scott model using the lifting monad (PCF terms)

Definition (Interpreting PCF terms)

The PCF terms are interpreted using the lifting monad, e.g.

- $[\![\underline{n}]\!] :\equiv \eta(n)$ for every $n : \mathbf{N}$;
- $[\![\mathsf{pred}]\!] :\equiv \mathcal{L}(P)$, where P is the predecessor map on $\mathbf{N}$.

As usual,

$$[\![\mathsf{fix}]\!](f) :\equiv \bigsqcup_{n:\mathbf{N}} f^n(\bot).$$

Theorem

The model is sound and computationally adequate.

Proving soundness and computational adequacy

- Soundness is proved using a standard induction on the rules of the reduction relation $\triangleright^*$.
  Also use the Kleisli triple equations, e.g. to reduce $f^{\#}(\eta(n))$ to $f(n)$.
- Computational adequacy is proved using a standard logical relation.

To avoid dealing with variables, we work with a combinatory version of PCF and therefore a combinatory version of the operational semantics.

The logical relation is necessary, because the statement of computational adequacy does not allow for a direct proof by induction.

Characterising PCF propositions

Question

Recall: if t is a PCF term of the base type, then $[\![t]\!] : \mathcal{L}(\mathbf{N})$.
Can we characterise the propositions of the form is-defined$([\![t]\!])$?

Characterising PCF propositions

Question

Recall: if t is a PCF term of the base type, then $[\![t]\!] : \mathcal{L}(\mathbf{N})$.
Can we characterise the propositions of the form is-defined$([\![t]\!])$?

First observation

$$
\begin{aligned}
\text{is-defined}([\![t]\!]) &\longleftrightarrow \sum_{n:\mathbf{N}} [\![t]\!] = \eta(n) \\
&\longleftrightarrow \sum_{n:\mathbf{N}} [\![t]\!] = [\![\underline{n}]\!] \\
&\longleftrightarrow \sum_{n:\mathbf{N}} t \rhd^* \underline{n} \quad \text{(by soundness and computational adequacy)}
\end{aligned}
$$

Characterising PCF propositions

Second observation

$$\begin{aligned}\text{is-defined}(\llbracket t \rrbracket) &\longleftrightarrow \sum_{n:\mathbf{N}} t \rhd^* \underline{n} \\ &\longleftrightarrow \sum_{n:\mathbf{N}} \sum_{k:\mathbf{N}} t \rhd^k \underline{n},\end{aligned}$$

where $\rhd^k$ is reduction k steps.

We can prove $t \rhd^k \underline{n}$ to be decidable.
Thus, is-defined($\llbracket t \rrbracket$) is semi-decidable.

To prove that is-defined($\llbracket t \rrbracket$) is semi-decidable, we prove that $t \rhd^k \underline{n}$ is decidable.
We have the following general theorem. Let R be relation on a type X. If

(i) X has decidable equality;

(ii) R is single-valued;

(iii) $\sum_{y:X} xRy$ is decidable for every $x : X$;

then, the k-step reflexive transitive closure R^k of R is decidable for every natural number k.
So we need the PCF terms to have decidable equality. This follows from a more general result on decidable equality of indexed W-types originally due to Jasper Hugunin.

Restricting the propositions in the lifting?

Question

Could we have used

$$\mathcal{L}_{\mathsf{sd}}(X) :\equiv \sum_{P:\Omega_{\mathsf{sd}}} (P \to X),$$

where Ω_{sd} is the type of semi-decidable propositions of Ω?

Restricting the propositions in the lifting?

Question

Could we have used

$$\mathcal{L}_{\mathsf{sd}}(X) :\equiv \sum_{P:\Omega_{\mathsf{sd}}} (P \to X),$$

where Ω_{sd} is the type of semi-decidable propositions of Ω?

No, because one needs some form of countable choice to prove the Kleisli equations [EK17; Kna18].

Universe levels of the lifting

- Recall $\mathcal{L}(X) :\equiv \sum_{P:\Omega}(P \to X)$ with $\Omega :\equiv \sum_{P:\mathcal{T}}$ is-a-subsingleton(P). Hence,

$$\mathcal{L}(X : \mathcal{U}) : \mathcal{T}^{+} \sqcup \mathcal{U}.$$

Universe levels of the lifting

- Recall $\mathcal{L}(X) :\equiv \sum_{P:\Omega}(P \to X)$ with $\Omega :\equiv \sum_{P:\mathcal{T}}$ is-a-subsingleton(P). Hence,

$$\mathcal{L}(X : \mathcal{U}) : \mathcal{T}^{+} \sqcup \mathcal{U}.$$

- From now on, take $\mathcal{T} :\equiv \mathcal{U}_0$. Note:

$$\mathcal{L}(\mathsf{N} : \mathcal{U}_0) : \mathcal{U}_1$$

and

$$\sqsubseteq : \mathcal{L}(\mathsf{N}) \to \mathcal{L}(\mathsf{N}) \to \mathcal{U}_1.$$

Universe levels and dcpos

- Write $\mathcal{W}$-DCPO$_{\mathcal{U},\mathcal{V}}$ for the type of dcpos with bottom with
  - underlying type in $\mathcal{U}$;
  - partial order taking values in $\mathcal{V}$;
  - suprema for directed $\mathcal{W}$-families.

  So,

$$\mathcal{L}(\mathsf{N}) : \mathcal{U}_0\text{-DCPO}_{\mathcal{U}_1,\mathcal{U}_1}.$$

We are interested in these universe levels, because to interpret PCF types like $\iota \Rightarrow \iota \Rightarrow \iota \Rightarrow \iota$, we need to be able to iterate the exponential.

For the Scott model, we need $\mathcal{U}_0$-directed completeness, because we (only) need N-indexed families.

In general,

$$(-)^{(-)} : \mathcal{W}\text{-DCPO}_{\mathcal{U},\mathcal{V}} \to \mathcal{W}\text{-DCPO}_{\mathcal{U}',\mathcal{V}'} \to \mathcal{W}\text{-DCPO}_{\mathcal{W}^+ \sqcup \mathcal{U} \sqcup \mathcal{V} \sqcup \mathcal{U}' \sqcup \mathcal{V}', \mathcal{U} \sqcup \mathcal{V}'}.$$

In particular,

$$(-)^{(-)} : \mathcal{U}_0\text{-DCPO}_{\mathcal{U}_1,\mathcal{U}_1} \to \mathcal{U}_0\text{-DCPO}_{\mathcal{U}_1,\mathcal{U}_1} \to \mathcal{U}_0\text{-DCPO}_{\mathcal{U}_1,\mathcal{U}_1}.$$

These universe levels have been checked using Agda (see slide after Conclusion).

Universe levels and dcpos

- Write $\mathcal{W}$-DCPO$_{\mathcal{U},\mathcal{V}}$ for the type of dcpos with bottom with
  - underlying type in $\mathcal{U}$;
  - partial order taking values in $\mathcal{V}$;
  - suprema for directed $\mathcal{W}$-families.

  So,
  $$\mathcal{L}(\mathbb{N}) : \mathcal{U}_0\text{-DCPO}_{\mathcal{U}_1,\mathcal{U}_1}.$$
- The type of continuous functions mentions all directed families, so taking exponentials potentially increases universe levels.

We are interested in these universe levels, because to interpret PCF types like $\iota \Rightarrow \iota \Rightarrow \iota \Rightarrow \iota$, we need to be able to iterate the exponential.

For the Scott model, we need $\mathcal{U}_0$-directed completeness, because we (only) need $\mathbb{N}$-indexed families.

In general,

$$(-)^{(-)} : \mathcal{W}\text{-DCPO}_{\mathcal{U},\mathcal{V}} \to \mathcal{W}\text{-DCPO}_{\mathcal{U}',\mathcal{V}'} \to \mathcal{W}\text{-DCPO}_{\mathcal{W}^+ \sqcup \mathcal{U} \sqcup \mathcal{V} \sqcup \mathcal{U}' \sqcup \mathcal{V}', \mathcal{U} \sqcup \mathcal{V}'}.$$

In particular,

$$(-)^{(-)} : \mathcal{U}_0\text{-DCPO}_{\mathcal{U}_1,\mathcal{U}_1} \to \mathcal{U}_0\text{-DCPO}_{\mathcal{U}_1,\mathcal{U}_1} \to \mathcal{U}_0\text{-DCPO}_{\mathcal{U}_1,\mathcal{U}_1}.$$

These universe levels have been checked using Agda (see slide after Conclusion).

Universe levels and dcpos

- Write $\mathcal{W}$-DCPO$_{\mathcal{U},\mathcal{V}}$ for the type of dcpos with bottom with
  - underlying type in $\mathcal{U}$;
  - partial order taking values in $\mathcal{V}$;
  - suprema for directed $\mathcal{W}$-families.

  So,

$$\mathcal{L}(\mathsf{N}) : \mathcal{U}_0\text{-DCPO}_{\mathcal{U}_1,\mathcal{U}_1}.$$

- The type of continuous functions mentions all directed families, so taking exponentials potentially increases universe levels.
- Nonetheless,

$$[\![-]\!] : \text{PCF-types} \to \mathcal{U}_0\text{-DCPO}_{\mathcal{U}_1,\mathcal{U}_1}.$$

We are interested in these universe levels, because to interpret PCF types like $\iota \Rightarrow \iota \Rightarrow \iota \Rightarrow \iota$, we need to be able to iterate the exponential.

For the Scott model, we need $\mathcal{U}_0$-directed completeness, because we (only) need N-indexed families.

In general,

$$(-)^{(-)} : \mathcal{W}\text{-DCPO}_{\mathcal{U},\mathcal{V}} \to \mathcal{W}\text{-DCPO}_{\mathcal{U}',\mathcal{V}'} \to \mathcal{W}\text{-DCPO}_{\mathcal{W}^+ \sqcup \mathcal{U} \sqcup \mathcal{V} \sqcup \mathcal{U}' \sqcup \mathcal{V}', \mathcal{U} \sqcup \mathcal{V}'}.$$

In particular,

$$(-)^{(-)} : \mathcal{U}_0\text{-DCPO}_{\mathcal{U}_1,\mathcal{U}_1} \to \mathcal{U}_0\text{-DCPO}_{\mathcal{U}_1,\mathcal{U}_1} \to \mathcal{U}_0\text{-DCPO}_{\mathcal{U}_1,\mathcal{U}_1}.$$

These universe levels have been checked using Agda (see slide after Conclusion).

Conclusion

Scott model of PCF in constructive predicative univalent type theory:

- uses the lifting monad
- soundness and computational adequacy
- important role for propositional truncation
- careful handling of universes

The point is that lifting works: we can define the Scott model of PCF in constructive predicative univalent type theory and prove fundamental properties like soundness and computational adequacy.
We deviate from the classical treatment in

- using the lifting monad from Knapp and Escardó;
- our use of the propositional truncation;
- a careful treatment of directed completeness and universe levels (because of the predicative setting).

Conclusion

Scott model of PCF in constructive predicative univalent type theory:

- uses the lifting monad
- soundness and computational adequacy
- important role for propositional truncation
- careful handling of universes

Thank you for your attention!

The point is that lifting works: we can define the Scott model of PCF in constructive predicative univalent type theory and prove fundamental properties like soundness and computational adequacy.
We deviate from the classical treatment in

- using the lifting monad from Knapp and Escardó;
- our use of the propositional truncation;
- a careful treatment of directed completeness and universe levels (because of the predicative setting).

Full paper and formalisation

Tom de Jong. *The Scott model of PCF in univalent type theory*. June 2019. arXiv: 1904.09810 [math.LO].

Any comments, questions, feedback are most welcome!

Coq formalisation using UniMath:
`https://github.com/tomdjong/UniMath/tree/paper/UniMath`

Agda formalisation (to check the universe levels) using Martín Escardó's Agda development:
`https://github.com/martinescardo/TypeTopology`

At present, it is impossible to check universe levels in UniMath. Therefore, we have redone part of our development in Agda.

References I

[AJ94] S. Abramsky and A. Jung. 'Domain Theory'. In: *Handbook of Logic in Computer Science*. Ed. by S. Abramsky, D. M. Gabbay and T. S. E. Maibaum. Vol. 3. Updated online version available at: `https://www.cs.bham.ac.uk/~axj/pub/papers/handy1.pdf`. Clarendon Press, 1994, pp. 1–168.

References II

[ADK17] Thorsten Altenkirch, Nils Anders Danielsson and Nicolai Kraus. 'Partiality, Revisited: The Partiality Monad as a Quotient Inductive-Inductive Type'. In: *Foundations of Software Science and Computation Structures*. Ed. by Javier Esparza and Andrzej S. Murawski. Vol. 10203. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2017, pp. 534–549. DOI: 10.1007/978-3-662-54458-7_31.

References III

[BKV09] Nick Benton, Andrew Kennedy and Carsten Varming. 'Some Domain Theory and Denotational Semantics in Coq'. In: *Theorem Proving in Higher Order Logics*. Ed. by Stefan Berghofer et al. Vol. 5674. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2009, pp. 115–130. DOI: 10.1007/978-3-642-03359-9_10.

[CUV17] James Chapman, Tarmo Uustalu and Niccolò Veltri. 'Quotienting the delay monad by weak bisimilarity'. In: *Mathematical Structures in Computer Science* 29.1 (2017), pp. 67–92. DOI: 10.1017/s0960129517000184.

References IV

[Coq18] Thierry Coquand. 'A survey of constructive presheaf models of univalence'. In: *SIGLOG News* 5.3 (2018), pp. 54–65. DOI: 10.1145/3242953.3242962.

[CMR17] Thierry Coquand, Bassel Mannaa and Fabian Ruch. 'Stack semantics of type theory'. In: *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. June 2017. DOI: 10.1109/LICS.2017.8005130.

References V

[EK17] Martín H. Escardó and Cory M. Knapp. 'Partial Elements and Recursion via Dominances in Univalent Type Theory'. In: *26th EACSL Annual Conference on Computer Science Logic (CSL 2017)*. Ed. by Valentin Goranko and Mads Dam. Vol. 82. Leibniz International Proceedings in Informatics (LIPIcs). Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2017, 21:1–21:16. DOI: `10.4230/LIPIcs.CSL.2017.21`.

[Kna18] Cory Knapp. 'Partial Functions and Recursion in Univalent Type Theory'. PhD thesis. School of Computer Science, University of Birmingham, June 2018.

References VI

[Koc91] Anders Kock. 'Algebras for the partial map classifier monad'. In: *Category Theory*. Ed. by Aurelio Carboni, Maria Cristina Pedicchio and Guiseppe Rosolini. Vol. 1488. Lecture Notes in Mathematics. Springer Berlin Heidelberg, 1991, pp. 262–278. DOI: 10.1007/BFb0084225.

[Kra+17] Nicolai Kraus et al. 'Notions of Anonymous Existence in Martin-Löf Type Theory'. In: *Logical Methods in Computer Science* 13 (1 2017). DOI: 10.23638/LMCS-13(1:15)2017.

References VII

[RS99] Bernhard Reus and Thomas Streicher. 'General synthetic domain theory – a logical approach'. In: *Mathematical Structures in Computer Science* 9.2 (1999), pp. 177–223. DOI: 10.1017/S096012959900273X.

[Str06] Thomas Streicher. *Domain-Theoretic Foundations of Functional Programming*. World Scientific, 2006. DOI: 10.1142/6284.