

Multi-Clocked Guarded Recursion Beyond ω

Rasmus Ejlers Møgelberg  

IT University of Copenhagen, Denmark

Abstract

Type theories with multi-clocked guarded recursion provide a flexible framework for programming with coinductive types encoding productivity in types. Combining this with solutions to general guarded domain equations one can also construct relatively simple denotational models of programming languages with advanced features. These constructions have previously been explored in the setting of extensional type theory through a presheaf model, which proves correctness of encodings of W-types. That model has been adapted to presheaves of cubical sets (functors into the category of cubical sets), where the model verifies correctness of encodings also of coinductive types whose definitions involve quotient inductive types such as finite powersets or finite distributions. Likewise the cubical model also verifies correctness of coinductive predicates defined using existential quantification and allows the results to be related to the global world of cubical sets.

This paper looks at how to extend the extensional presheaf model of multi-clocked guarded recursion to higher ordinals, so that correctness of encodings of coinductive types can be extended from W-types to those involving finite powersets and finite distributions, as well as coinductive predicates involving existential quantification. This extension will allow results previously proved in Clocked Cubical Type Theory to be interpreted in a model based on set-theory, proving the correctness of these results as understood in their usual set theoretic interpretation.

2012 ACM Subject Classification Theory of computation $\rightarrow$ Type theory; Theory of computation $\rightarrow$ Denotational semantics; Theory of computation $\rightarrow$ Categorical semantics

Keywords and phrases Guarded Recursion, Coinductive Types, Dependent Type Theory, Accessible Functors, Algebraic Theories

Digital Object Identifier 10.4230/LIPIcs.TYPES.2025.11

Funding *Rasmus Ejlers Møgelberg*: This work was supported by the Independent Research Fund Denmark grant number 2032-00134B.

Acknowledgements I thank Jiří Adámek for discussions of the material of Section 6.1.

1 Introduction

The complex numbers form a mathematical paradise in which all polynomials have roots. However, they are not just a dream world without connection to the more earthly world of the real numbers. For example, computing with complex numbers sometimes leads to the construction of real solutions to polynomial equations. In the world of programming language semantics the equations of interest are recursive domain equations $X \cong F(X)$. Many of these do not have solutions in the category of sets, but the topos of trees [7] serves as a mathematical paradise in which guarded versions of these equations $X \cong F(\triangleright X)$ all have solutions. One can therefore construct models of advanced programming languages using the topos of trees, but can one then relate these results to the real world?

The answer depends on the type of results, and what is meant by the “real world”. For example, the original paper on the topos of trees [7] constructs a model of a programming language with higher order store in the internal language of the topos of trees ($\mathbf{Set}^{\omega_{\text{op}}}$), and then argues how these results relate to the real world of the category of sets. In some cases, this is not possible, however. For example, a proof of an existential quantification of a predicate in the topos of trees does not always externalise to a proof of an existential in sets. For this reason, Transfinite Iris [22] uses higher indexing than ω to reason about liveness



© Rasmus Ejlers Møgelberg;
licensed under Creative Commons License CC-BY 4.0

31st International Conference on Types for Proofs and Programs (TYPES 2025).

Editors: Fredrik Nordvall Forsberg and James McKinna; Article No. 11; pp. 11:1–11:22

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

properties. Likewise, as we shall explain below, passing to the real world does not preserve functors like P_f and D_f used to model non-determinism and probabilistic choice, and this causes problems when relating models of languages with these constructs to the real world.

The problem can be stated type theoretically using *multi-clocked guarded recursion* as in Clocked Type Theory (CloTT) [6] and its presheaf model [18]. In CloTT the delay modality $\triangleright^\kappa$ is indexed by a clock κ , and one can quantify these clocks universally. The model contains a copy of the standard set theoretic model of type theory as well as a copy of the topos of trees, and universal quantification acts as a form of externalisation, relating results proved in the topos of tree to those in the real world of set theory. In this setting, the problem of concluding from existential quantification in the topos of trees to actual existence in sets can be expressed as commutativity of quantification over clocks with existential quantification.

Multi-clocked guarded recursion also offers a way of programming with coinductive types and predicates in type theory, using guarded recursion to express the requirement of productivity in types [5, 8]. Specifically, for functors F commuting with clock quantification, the solution to the guarded recursive domain equation $X(\kappa) \cong F(\triangleright^\kappa X(\kappa))$ externalises to a terminal coalgebra with carrier $\forall \kappa. X(\kappa)$ in the real world of set theory. Unfortunately, standard functors such as finite powerset functor P_f or the finite distribution functor D_f do not commute with with clock quantification in the standard model, and so one cannot use these tools to program with non-deterministic or probabilistic processes.

By varying the notion of “real world”, however, the situation can be improved. Kristensen et al. [17] construct a model of the combination of CloTT and Cubical Type Theory [13] by considering presheaves valued in cubical sets. In this model, existential quantification and functors like P_f and D_f , constructed as quotient inductive types on the category of hsets *do* commute with clock quantification. This means that guarded recursion can be used as a way of programming with coinductive types and predicates in Cubical Type Theory, including those involving P_f and D_f and existential quantification. Clocked Cubical Type Theory has been used to model programming languages with non-determinism [20] and probabilistic choice [23]. These models are constructed by solving guarded recursive domain equations (also with negative occurrences), and using the commutativity results above, the models can be related to the operational semantics of these languages, defined as coinductive non-deterministic and probabilistic processes. Likewise, Clocked Cubical Type Theory has been used to prove existence and non-existence of distributive laws between the coinductive delay monad and other monads [21], and to model programming languages with gradual typing [14]. Some of these results have been formalised in Guarded Cubical Agda [25], an extension of the Cubical Agda proof assistant [26].

The question is now how tied these results are to cubical type theory. For example, the results proved about existence and non-existence of distributive laws between monads as interpreted in the model of Kristensen et al. are results about these monads as seen as monads of hsets in the cubical model. Do the same results hold for these monads on **Set**? Likewise, results about contextual equivalence of programs proved using the cubical model a priori only say something about their semantics as read in cubical sets. How do these relate to the ordinary set theoretic interpretations?

In this paper we return to the set theoretic model of CloTT and look at how varying the indexing ordinal used in the model construction implies commutativity of clock quantification with certain existential quantifications and with functors constructed as quotient inductive types such as P_f and D_f . The contributions are as follows

- We analyse existing applications of Clocked Cubical Type Theory [17, 23, 21] and isolate requirements on the model that will allow for these to be interpreted in the set theoretic

$$\begin{array}{c}
\frac{}{\Gamma, \kappa : \text{clock} \vdash} \quad \frac{\kappa : \text{clock} \in \Gamma}{\Gamma, \alpha : \kappa \vdash} \quad \frac{\Gamma \vdash t : \triangleright (\alpha : \kappa).A \quad \Gamma, \beta : \kappa, \Gamma' \vdash}{\Gamma, \beta : \kappa, \Gamma' \vdash t[\beta] : A[\beta/\alpha]} \\
\\
\frac{\Gamma, \alpha : \kappa \vdash t : A}{\Gamma \vdash \lambda(\alpha : \kappa).t : \triangleright (\alpha : \kappa).A} \quad \frac{\Gamma, \kappa : \text{clock} \vdash t : A}{\Gamma \vdash \Lambda \kappa. t : \forall \kappa. A} \quad \frac{\Gamma \vdash t : \forall \kappa. A \quad \kappa' : \text{clock} \in \Gamma}{\Gamma \vdash t[\kappa'] : A[\kappa'/\kappa]} \\
\\
\frac{\kappa : \text{clock} \in \Gamma \quad \Gamma \vdash A \text{ type}}{\Gamma \vdash \text{fix}^\kappa : (\triangleright^\kappa A \rightarrow A) \rightarrow A} \quad \frac{x : A \in \Gamma}{\Gamma \vdash x : A}
\end{array}$$

■ **Figure 1** Selected typing rules for Clocked Type Theory [6, 18].

model.

- We state the conservativity of the model of Clocked Type Theory over the standard set theoretic model of type theory. This is an easy consequence of the model construction of [18], but has not previously been stated explicitly. We show how this allows us to relate results proved in CloTT to the real world of set theory.
- We give semantic conditions for monads on sets to commute with clock quantification in the model of Clocked Type Theory indexed by regular ordinals, and we give a simple condition on algebraic theories to generate such monads. This accounts for the functors P_f and D_f mentioned above, but also monads of higher arity, such as the countable powerset monad.
- We give conditions allowing existential quantification to commute with quantification over clocks in the model

As a result we see that the applications mentioned above can all be expressed in the internal language of a variant of the set theoretic model indexed by a sufficiently large ordinal. We hope that these results can form part of the semantic basis for formalisation of applications as those mentioned above also in proof assistants not based on cubical type theory.

We remark that the metatheory in which the models are constructed is classical set theory. In particular, the proof of Theorem 8 is very classical.

Overview

We start by recalling Clocked Type Theory and the encoding of coinductive types from guarded recursive types in Sections 2 and 3. Section 4 recalls the applications of Clocked Cubical Type Theory and derives the requirements needed to express these in the version of Clocked Type Theory based on the set theoretic model. Section 5 recalls the presheaf model of Clocked Type Theory and describes the conservativity result over the set theoretic model. Section 6 gives conditions on finitary monads to commute with clock quantification as well as the syntactic condition on algebraic theories ensuring this. Section 7 considers commutativity of existential quantification and clock quantification, and Section 8 concludes.

2 Clocked Type Theory

This section gives a brief introduction to Clocked Type Theory. We refer readers to [18] for full details.

Clocked Type Theory is a dependently typed type theory with Π - and Σ -types, extensional identity types, binary and nullary sums and products, as well as a modality $\triangleright$ for delays

and a guarded fixed point operator. Delays are associated with clocks, so that $\triangleright^\kappa A$ can be understood as a type of data of type A delayed by one time step on clock κ . To program with clocks, the type theory has a special form of assumptions $\kappa : \text{clock}$ that can be placed in a context. Note that *clock* is not a type, but rather a pretype, similar to the status of the interval in Cubical Type Theory [13]. Introduction and elimination for $\triangleright$ is by abstracting and applying ticks on clocks, following the Fitch-style rules for programming with modal types [11, 12]. Selected typing rules can be found in Figure 1. The notation $\triangleright^\kappa A$ is used for $\triangleright(\alpha : \kappa).A$ when the tick α does not appear free in A . Note that variables can be introduced from anywhere in a context, also if they are behind ticks. This means that one can define a term $\text{next}^\kappa \stackrel{\text{def}}{=} \lambda x. \lambda(\alpha : \kappa). x$ of type $A \rightarrow \triangleright^\kappa A$. Given $f : \triangleright^\kappa A \rightarrow A$, the fixed point operator computes the unique fixed point of $f \circ \text{next}^\kappa$. The language of ticks also allows one to define a dependent version of an applicative functor action for $\triangleright$:

$$\lambda f. \lambda x. \lambda(\alpha : \kappa). f[\alpha](x[\alpha]) : \triangleright^\kappa(\Pi(x : A). B(x)) \rightarrow \Pi(x : \triangleright^\kappa A). \triangleright(\alpha : \kappa). B(x[\alpha])$$

This is, for example, useful for reasoning about elements of modal type. It can also be used to express an extensionality principle for $\triangleright$:

$$(t =_{\triangleright(\alpha : \kappa). A} u) \cong \triangleright(\alpha : \kappa). (t[\alpha] =_A u[\alpha])$$

For these applications, it is important that the same ticks can be used on both the logical and term side, and this is one benefit of using type theory rather than a language with a clear separation between term language and logic, such as topos logic.

2.1 Axioms

One of the main purposes of quantification over clocks is to allow for a controlled elimination of $\triangleright$. In Clocked Type Theory this is done by an operation of application to a tick constant $\diamond$, with very special typing rules, which can be used to construct a term of type

$$\text{force} : (\forall \kappa. \triangleright^\kappa A) \rightarrow \forall \kappa. A.$$

This, together with the *tick irrelevance axiom*

$$\frac{\Gamma \vdash t : \triangleright^\kappa A}{\text{tirr}^\kappa t : \triangleright(\alpha : \kappa). \triangleright(\alpha' : \kappa). t[\alpha] =_A t[\alpha']} \quad (1)$$

allows one to prove the following

$$\text{IsEquiv}(\lambda(x : \forall \kappa. A). \lambda \kappa. \lambda(\alpha : \kappa). x[\alpha]) \quad (2)$$

where $\text{IsEquiv}(-)$ is defined precisely as in homotopy type theory [24]. In the present case of extensional type theory, this axiom just states that the canonical map $(\forall \kappa. A) \rightarrow \forall \kappa. \triangleright^\kappa A$ is an isomorphism.

The axiom of *clock irrelevance* states

$$\text{IsEquiv}(\lambda x : A. \lambda \kappa. x) \quad (3)$$

Again, in our setting, this just states that the canonical map $A \rightarrow \forall \kappa. A$ is an isomorphism when κ is not free in A . In Clocked Cubical Type Theory [17] clock irrelevance is not an axiom, but can be proved for many types. The other axioms mentioned can all be given computational content in Clocked Cubical Type Theory, but it is unclear how to generalise this to type theories not based on Cubical Type Theory. In this paper we are mostly concerned with properties of the model, and so we will assume clock irrelevance and tick irrelevance as axioms, deriving (2) as a consequence.

$$\begin{array}{c}
\frac{\Delta = \{\kappa_1, \dots, \kappa_n\} \quad \Gamma \vdash \kappa_i : \text{clock for } i = 1, \dots, n}{\Gamma \vdash U_\Delta \text{ type}} \quad \frac{\Gamma \vdash t : U_\Delta}{\Gamma \vdash \text{El}_\Delta(t) \text{ type}} \\
\\
\frac{\Gamma \vdash t : U_\Delta \quad \Delta \subseteq \Delta'}{\Gamma \vdash \text{in}_{\Delta, \Delta'}(t) : U_{\Delta'}} \quad \frac{\Gamma, \kappa : \text{clock} \vdash A : U_{\Delta, \kappa}}{\Gamma \vdash \bar{\forall} \kappa. A : U_\Delta} \quad \frac{\Gamma, \alpha : \kappa \vdash A : U_\Delta \quad \kappa \in \Delta}{\Gamma \vdash \bar{\exists}(\alpha : \kappa). A : U_\Delta}
\end{array}$$

■ **Figure 2** Selected typing rules for universes.

2.2 Universes

It is inconsistent with the clock irrelevance axiom to assume a single universe closed under $\triangleright$ [18]. To solve this, CloTT has a family of universes indexed by finite sets of clocks. Figure 2 shows some basic rules for these. Judgemental equality rules state that codes for type operations indeed encode those type operation, as well as commutativity rules for universe inclusions $\text{in}_{\Delta, \Delta'}$ and codes for type operations. For example

$$\text{El}_{\Delta'}(\text{in}_{\Delta, \Delta'}(t)) = \text{El}_\Delta(t) \quad \text{El}_\Delta(\bar{\exists}(\alpha : \kappa). A) = \triangleright(\alpha : \kappa).(\text{El}_\Delta(A))$$

Note that a universe U_Δ is only closed under $\triangleright^\kappa$ for the κ in Δ . The notation Δ, κ is shorthand for the union of Δ and $\{\kappa\}$.

Unlike previous presentations of CloTT, we include a family of universes of propositions Prop_Δ , also indexed by clock contexts Δ . All rules of Figure 2 have corresponding rules for universes of propositions. Moreover, these universes are closed under basic operations of predicate logic such as

$$\frac{\Gamma \vdash t : U_\Delta \quad \Gamma \vdash \phi : \text{El}_\Delta(t) \rightarrow \text{Prop}_\Delta}{\Gamma \vdash \exists(x : \text{El}_\Delta(t)). \phi(x) : \text{Prop}_\Delta} \quad \frac{\Gamma \vdash t : U_\Delta \quad \Gamma \vdash u, s : \text{El}_\Delta(t)}{\Gamma \vdash u = s : \text{Prop}_\Delta}$$

and a similar rule for universal quantification.

3 Encoding Coinductive Types

One benefit of universes is that they allow for encoding guarded recursive types simply as fixed points for endomaps on universes. Suppose, for example, that $A : U_{\Delta, \kappa}$. Defining $\bar{D}^\kappa(A)$ as the fixed point of the map

$$\lambda(x : \triangleright^\kappa U_{\Delta, \kappa}). A + \bar{\exists}^\kappa x$$

then $D^\kappa(A) \stackrel{\text{def}}{=} \text{El}_{\Delta, \kappa}(\bar{D}^\kappa(A))$ satisfies $D^\kappa(A) = A + \triangleright^\kappa D^\kappa(A)$. Note that this is an actual judgemental equality of types. In Clocked Cubical Type Theory, fixed points only unfold up to propositional equality, and the above type equation is only an equivalence of types. The type constructor D^κ is often referred to as the *guarded delay monad*.

Guarded recursive types can be used as a stepping stone towards defining coinductive types. For example, the more standard coinductive delay monad D can be encoded as the map that maps $A : U_\Delta$ to $\bar{\forall} \kappa. \bar{D}^\kappa(A)$ where κ is a fresh clock. This gives the coinductive solution to $D(A) \cong A + D(A)$. Using guarded recursion has the benefit that one can program with D using the guarded fixed point operator, which allows productivity of coinductive definitions to be captured in types.

To describe the general setting of the encoding of inductive types, note first that for any type I , the type $I \rightarrow U_\Delta$ describes the objects of a category in the naive sense: Morphisms from X to Y are simply terms of the type $\Pi(i : I).X(i) \rightarrow Y(i)$. Here we have left the “elements of” operation El_Δ implicit as we shall do below also for operations of universe inclusions to avoid clutter. If $\Delta \subset \Delta'$, the universe inclusion $\text{in}_{\Delta, \Delta'}$ induces a functor from $I \rightarrow U_\Delta$ to $I \rightarrow U_{\Delta'}$. In the opposite direction, universal quantification over clocks defines a functor $(\forall \kappa.(I \rightarrow U_{\Delta, \kappa})) \rightarrow (I \rightarrow U_\Delta)$ (assuming κ fresh for I) by

$$(\forall \kappa.X)(i) \stackrel{\text{def}}{=} \bar{\forall} \kappa.X[\kappa](i)$$

where $\forall \kappa.(I \rightarrow U_{\Delta, \kappa})$ is considered a category in the naive sense. In the next definition we write $\text{Func}(\mathbb{C}, \mathbb{D})$ for the type of functors between categories $\mathbb{C}$ and $\mathbb{D}$.

► **Definition 1.** A functor $F : (I \rightarrow U_\Delta) \rightarrow (J \rightarrow U_\Delta)$ commutes with clock quantification if there exists a functor $F' : \forall \kappa.\text{Func}(I \rightarrow U_{\Delta, \kappa}, J \rightarrow U_{\Delta, \kappa})$ extending F in the sense that

$$\begin{array}{ccc} (I \rightarrow U_\Delta) & \xrightarrow{F} & (J \rightarrow U_\Delta) \\ \downarrow & & \downarrow \\ (I \rightarrow U_{\Delta, \kappa}) & \xrightarrow{F'} & (J \rightarrow U_{\Delta, \kappa}) \end{array}$$

such that the canonical morphism

$$F(\forall \kappa.X) \rightarrow \forall \kappa.F'[\kappa](X)$$

is an isomorphism in $J \rightarrow U_\Delta$ for all X in $\forall \kappa.(I \rightarrow U_{\Delta, \kappa})$.

Suppose, for example, that $A : U_\Delta$. Then $F(X) \stackrel{\text{def}}{=} A + X$ defines a functor $U_\Delta \rightarrow U_\Delta$ (omitting $I = 1$), which extends to a functor $F' : \forall \kappa.(U_{\Delta, \kappa} \rightarrow U_{\Delta, \kappa})$ because the universe inclusions commute with binary sums. To prove that F commutes with clock quantification, one needs universal quantification to distribute over sums ($\forall \kappa.(B + C) \cong \forall \kappa.B + \forall \kappa.C$) and that A is clock irrelevant which is an axiom (3). In the following we shall use the same symbol for the functor F and its extension F' .

► **Theorem 2** ([5, 19]). Let $F : (I \rightarrow U_\Delta) \rightarrow (I \rightarrow U_\Delta)$ be a functor commuting with universal quantification over clocks. There is a unique object $\mu^\kappa X.F(\triangleright^\kappa X)$ in $(I \rightarrow U_{\Delta, \kappa})$ satisfying $F(\triangleright^\kappa(\mu^\kappa X.F(\triangleright^\kappa X))) \cong \mu^\kappa X.F(\triangleright^\kappa X)$. The object $\forall \kappa.\mu^\kappa X.F(\triangleright^\kappa X)$ is a final coalgebra for F .

The encoding was first proved with respect to a denotational model by Atkey and McBride [5], and later internalised in type theory by the author [19]. Although the setting above is slightly different from the original one, due to the indexing of universes, the proof carries over verbatim from [19]. To get a feeling for the proof, note how the structure isomorphism of the final coalgebra can be easily constructed:

$$\begin{aligned} \forall \kappa.\mu^\kappa X.F(\triangleright^\kappa X) &\cong \forall \kappa.F(\triangleright^\kappa(\mu^\kappa X.F(\triangleright^\kappa X))) \\ &\cong F(\forall \kappa.\triangleright^\kappa(\mu^\kappa X.F(\triangleright^\kappa X))) \\ &\cong F(\forall \kappa.\mu^\kappa X.F(\triangleright^\kappa X)) \end{aligned}$$

A version of Theorem 2 can be proved also for the universes of propositions Prop_Δ . This allows for encoding of inductive predicates.

For Theorem 2 to be useful, one needs a large collection of functors commuting with clock quantification. Many functors have this property just by virtue of being defined by type constructors that commute with clock quantification. For example, $\forall\kappa$ distributes over Σ -types just like Π does, and commutes with Π , so polynomial functors commute with clock quantification:

$$\begin{aligned}\forall\kappa.(\Sigma(a : A).\Pi(b : B(a).X)) &\cong \Sigma(a : \forall\kappa.A).\forall\kappa.\Pi(b : B(a[\kappa]).X) \\ &\cong \Sigma(a : A).\forall\kappa.\Pi(b : B(a).X) \\ &\cong \Sigma(a : A).\Pi(b : B(a).\forall\kappa.X)\end{aligned}$$

using clock irrelevance of A in the second line. In the model, universal quantification over clocks distributes over binary sums of types, but this is not directly provable in the type theory unless one encodes sum as a Σ -type indexed by booleans and uses clock-irrelevance for booleans. In this paper we will just consider this an axiom, and so it proves correctness of the encoding of the coinductive delay monad. The model also verifies that $\forall\kappa$ commutes with $\triangleright^{\kappa'}$, and this can be proved internally if the rule for tick application is extended as in [17]. In this paper we will look at conditions for $\forall\kappa$ commuting with quotient inductive types, in particular monads induced by algebraic theories.

On the propositional side, one can prove that $\forall\kappa$ commutes with conjunction, as well as universal quantification over clocks as well as ordinary variables. Similarly to the case of types, the model also verifies that $\forall\kappa$ commutes with disjunction and $\triangleright^{\kappa'}$. Existential quantification is more difficult. In Clocked Cubical Type Theory, existential quantification can be encoded using Σ and propositional truncation, and both of these can be proved to commute with clock quantification. In the extensional model clock quantification does not commute with propositional truncation (see Example 7 below). We will see, however, that by varying the indexing ordinal of the model, it is safe to assume commutativity with quantification over sets of small cardinality. E.g., by indexing over ω_1 (the first uncountable ordinal), clock quantification commutes with existential quantification over countable set. We first look at what is needed for applications.

4 Applications

This section recalls the applications of Clocked Cubical Type Theory to coalgebra [17], semantics of programming languages with probabilistic choice [23] and to reasoning about weak bisimilarity [21]. The aim is to determine the precise requirements needed to interpret these in a model of Clocked Type Theory.

4.1 Coalgebra

Coalgebras in general describe systems, and final coalgebras behaviours of such systems. Often the systems of interest are non-deterministic or probabilistic, and can be described as coalgebras for functors involving powersets or distributions. For example, coalgebras for the functor $F(X) = P_f(A \times X)$, where P_f is the finite powerset functor, are finitely branching labelled transition systems with labels in A . Similarly, coalgebras for $F(X) = A \times D_f(X)$, where D_f is the finite distribution functor, are Markov-processes with labels in A . Since clock quantification commutes with products, Theorem 2 will give encodings of final coalgebras for these functors, if P_f and D_f commute with quantification over clocks. Note that P_f and D_f are both generated by algebraic theories. For example, $P_f(X)$ is the free join semilattice on X , so generated by $\emptyset$, finite union, and equations for associativity, commutativity, idempotency

and a unit axiom for $\emptyset$. (See Section 6.2 for details.) The natural way to program with these in type theory is therefore as quotient inductive types, using the associated recursion and induction principles. As an example, we recall the induction principle for $P_f(X)$ just in the case of predicates: Given $\phi : P_f(X) \rightarrow \text{Prop}_\Delta$, in order to prove $\forall A : P_f(X). \phi(A)$, it suffices to prove $\phi(\emptyset)$, $\phi(\{x\})$ for all $x : X$, and that $\phi(A)$ and $\phi(B)$ implies $\phi(A \cup B)$.

Given a coalgebra $\xi : X \rightarrow F(X)$ one would also like to define notions of similarity and bisimilarity as coinductive predicates. For example, in the case of $F(X) = P_f(A \times X)$, R is a simulation relation if whenever $R(x, x')$ holds, also the following holds

$$\forall a, y. (a, y) \in \xi(x) \rightarrow \exists y'. (a, y') \in \xi(x') \wedge R(y, y')$$

and simulation is the largest simulation relation. Here, the predicate $(a, y) \in \xi(x)$ is defined by P_f -recursion. Supposing, for example, that $A, X : \mathbf{U}_\emptyset$, simulation should be the final coalgebra for the functor mapping $R : X \times X \rightarrow \text{Prop}_\emptyset$ to

$$G(R)(x, x') \stackrel{\text{def}}{=} \forall a, y. (a, y) \in \xi(x) \rightarrow \exists y'. (a, y') \in \xi(x') \wedge R(y, y')$$

In order to define simulation, one first defines $\lesssim^\kappa$ by guarded recursion to satisfy

$$x \lesssim^\kappa x' \stackrel{\text{def}}{=} \forall a, y. (a, y) \in \xi(x) \rightarrow \exists y'. (a, y') \in \xi(x') \wedge \triangleright^\kappa (y \lesssim^\kappa y')$$

then define $x \lesssim x'$ as $\forall \kappa. x \lesssim^\kappa x'$. This definition will work if G commutes with clock quantification. The crucial step in showing that, is to show that existential quantification and universal quantification over clocks commute. As we shall see, this is not a property that holds generally in the model. However, in this particular case, the natural thing to do is to redefine the predicate $\exists y'. (a, y') \in \xi(x') \wedge R(y, y')$ by P_f -recursion over $\xi(x')$:

- If $A = \emptyset$ then $\exists y'. (a, y') \in A \wedge R(y, y')$ is false
- If $A = \{(a', y')\}$ then $\exists y'. (a, y') \in A \wedge R(y, y')$ is defined to be $a = a' \wedge R(y, y')$
- If $A = B \cup C$ then $\exists y'. (a, y') \in A \wedge R(y, y')$ is defined as

$$(\exists y'. (a, y') \in B \wedge R(y, y')) \vee (\exists y'. (a, y') \in C \wedge R(y, y'))$$

Commutativity of the two quantifications can then be proved by P_f -induction if just $\forall \kappa. (\phi \vee \psi)$ is equivalent to $(\forall \kappa. \phi) \vee (\forall \kappa. \psi)$ and $\forall \kappa. \text{false}$ is false. Both of these hold in the presheaf model of Clocked Type Theory independently of the indexing ordinal.

4.2 Modelling Programming Languages with Probabilistic Choice

Stassen et al. [23] construct a model of Probabilistic FPC, a programming language with recursive types and finite probabilistic choice, in Clocked Cubical Type Theory. One of the basic ingredients in this interpretation is a monad modelling the combination of recursion and probabilistic choice. In its guarded form, this monad maps a type $A : \mathbf{U}_\emptyset$ to the guarded recursive type $D^\kappa A$ satisfying $D^\kappa A \cong D_f(A + \triangleright^\kappa(D^\kappa A))$ where D_f is the finite distribution monad. The coinductive version of this monad satisfies $D^\forall A \cong D_f(A + D^\forall A)$. Elements of this monad are distributions of values of type A and computations that can run for one more step. As before, the correctness of the definition $D^\forall A \stackrel{\text{def}}{=} \forall \kappa. D^\kappa A$ requires D_f commutes with clock quantification.

The monads D^κ and D are used for both denotational and operational semantics. In order to prove an adequacy theorem, the paper constructs a relation between syntax and semantics using guarded recursion. One crucial ingredient in this definition is a notion of lifting of a relation on values to one on computations. In terms of the universes of this paper,

one can express this lifting as mapping a relation $R : A \rightarrow B \rightarrow \text{Prop}_{\Delta, \kappa}$ where $A : \mathcal{U}_{\Delta, \kappa}$ and $B : \mathcal{U}_{\Delta}$ to a relation $\bar{R}^{\kappa} : D^{\kappa}A \rightarrow D^{\vee}B \rightarrow \text{Prop}_{\Delta, \kappa}$. Note the asymmetry: A and B live in different universes, and different monads are applied to them. The definition of $\bar{R}^{\kappa}$ is by case analysis of the first argument: An element in $D^{\kappa}A$ is either a distribution just of values of type A , one just of delayed computations, or a convex combination of the two: $\mu_1 \oplus_p \mu_2$ where $\mu_1 : D_f(A)$ and $\mu_2 : D_f(\triangleright^{\kappa}(D^{\kappa}A))$. (Here and below we leave the inclusions of $D_f(A)$ and $D_f(\triangleright^{\kappa}(D^{\kappa}A))$ into $D^{\kappa}A$ implicit for readability.) Which of the three cases an input is in is decidable and, in the latter case μ_1 and μ_2 and p can be computed from the input. In the latter case the definition reads

$$(\mu_1 \oplus_p \mu_2) \bar{R}^{\kappa} \nu \stackrel{\text{def}}{=} \exists \nu_1 : D_f(B), \nu_2 : D_f(D^{\vee}B). (\nu \rightsquigarrow^{\approx} \nu_1 \oplus_p \nu_2) \wedge \mu_1 \bar{R}^{\kappa} \nu_1 \wedge \mu_2 \bar{R}^{\kappa} \nu_2.$$

where $\rightsquigarrow^{\approx}$ is a form of reduction relation on $D^{\vee}$.

One important result of the paper (Lemma 7.3) states that if $A : \mathcal{U}_{\Delta}$, the relation $\bar{R} : D^{\vee}A \rightarrow D^{\vee}B \rightarrow \text{Prop}_{\Delta}$ defined as $\bar{R}(\mu, \nu) \stackrel{\text{def}}{=} \forall \kappa. \bar{R}^{\kappa}(\mu[\kappa], \nu)$ is equivalent to a coinductive definition of $\bar{R}$ which is similar to the definition of $\bar{R}^{\kappa}$. The proof uses that this coinductive definition can be defined via a guarded recursive definition, and this in turn relies on the existential quantifications of ν_1 and ν_2 to commute with quantification over clocks. Unlike the existential quantifications used in the previous paragraph, these quantifications cannot be rewritten to inductions over a quotient inductive type. In the models presented in this paper, existential quantification will only commute with clock quantification if the existential quantification is of bounded size. The result as stated in this generality will therefore not be true in these models. However, Stassen et al. only apply this theorem to the very specific case of $\nu : D^{\vee}1$, and the quantifications are therefore over $\nu_1 : D_f(1)$ and $\nu_2 : D_f(D^{\vee}1)$, so of bounded size.

4.3 Weak Bisimilarity

The coinductive delay monad satisfying $D(A) \cong A + D(A)$, is an intensional model of recursion because it counts computation steps. Often one wants to reason up to weak bisimilarity. However, working with weak bisimilarity is quite subtle. For example, proving that the quotient of the coinductive delay monad by weak bisimilarity is a monad seems to require countable choice [10]. Rather than working with the quotient, one therefore often works in a category of setoids. For example, Møgelberg and Zwart [21] prove that if T is a free model monad for an algebraic theory with no drop equations, then there is a distributive law of T over the coinductive delay monad quotiented as a setoid.

Recall that coinductive delay monad can be defined as $D(A) \stackrel{\text{def}}{=} \forall \kappa. D^{\kappa}(A)$ where $D^{\kappa}(A) \cong A + \triangleright^{\kappa} D^{\kappa}(A)$. Given a setoid (X, R) , we would like to define the setoid $(D(X), \sim_R)$ capturing weak bisimilarity. Writing

$$\begin{aligned} \text{now} & : A \rightarrow D(A) & \text{step} & : D(A) \rightarrow D(A) \\ \text{now}^{\kappa} & : A \rightarrow D^{\kappa}(A) & \text{step}^{\kappa} & : \triangleright^{\kappa} D^{\kappa}(A) \rightarrow D^{\kappa}(A) \end{aligned}$$

for the left and right inclusions into the sum, the relation $\sim_R$ should be defined by coinduction and cases:

$$\begin{aligned} \text{now}(a) \sim_R y & \stackrel{\text{def}}{=} \exists (n : \mathbb{N}, y' : X). y = \text{step}^n(\text{now}(y')) \wedge R(a, y') \\ x \sim_R \text{now}(a) & \stackrel{\text{def}}{=} \exists (n : \mathbb{N}, x' : X). x = \text{step}^n(\text{now}(x')) \wedge R(x', a) \\ \text{step}(x) \sim_R \text{step}(y) & \stackrel{\text{def}}{=} x \sim_R y \end{aligned}$$

This coinductive predicate can be encoded using guarded recursion, because the function mapping a relation $\sim_R$ to $F(\sim_R)$ defined as above commutes with clock quantification. Indeed, $x \sim_R y$ does not appear in the two first clauses, and so in these cases the required condition follows from the axiom of clock irrelevance. In the last clause the case is immediate.

However, in order to use guarded recursion for programming with weak bisimilarity, Møgelberg and Zwart [21] use a different definition: They define

$$x \sim_R y \stackrel{\text{def}}{=} \forall \kappa. x[\kappa] \sim_R^\kappa y[\kappa]$$

where $\sim_R^\kappa: D^\kappa X \rightarrow D^\kappa X \rightarrow \text{Prop}_{\Delta, \kappa}$ is defined also for $X: \mathbf{U}_{\Delta, \kappa}$ and $R: X \rightarrow X \rightarrow \text{Prop}_{\Delta, \kappa}$ as the unique relation satisfying

$$\text{now}^\kappa(a) \sim_R^\kappa y \stackrel{\text{def}}{=} \exists(n: \mathbb{N}, y': X). y = (\text{step}^\kappa \circ \text{next})^n(\text{now}^\kappa(y')) \wedge R(a, y')$$

$$x \sim_R^\kappa \text{now}^\kappa(a) \stackrel{\text{def}}{=} \exists(n: \mathbb{N}, x': X). x = (\text{step}^\kappa \circ \text{next})^n(\text{now}^\kappa(x')) \wedge R(x', a)$$

$$\text{step}^\kappa(x) \sim_R^\kappa \text{step}^\kappa(y) \stackrel{\text{def}}{=} \triangleright(\alpha: \kappa). (x[\alpha] \sim_R^\kappa y[\alpha])$$

Is this the same as the usual coinductive definition? When comparing the two we can assume $X: \mathbf{U}_\Delta$ and $R: X \rightarrow X \rightarrow \text{Prop}_\Delta$. The critical cases are the first two, so suppose x is $\text{now}(a)$. Then $\forall \kappa. x[\kappa] \sim_R^\kappa y[\kappa]$ reduces to

$$\forall \kappa. \exists(n: \mathbb{N}, y': X). y[\kappa] = (\text{step}^\kappa \circ \text{next})^n(\text{now}^\kappa(y')) \wedge R(a, y')$$

We would like to commute $\forall \kappa$ past the two existential quantification, after which the equivalence follows by the relationship between step and step^κ and the clock irrelevance axiom applied to $R(a, y')$. The first quantification is over a countable set, so will commute with $\forall \kappa$ if the indexing in the model is over a sufficiently high ordinal. The other one is more difficult because we want to define a generally applicable monad, and so should not put restrictions on the cardinality of X . On the other hand, the y' is essentially unique because

$$(\text{step}^\kappa \circ \text{next})^n(\text{now}^\kappa(y')) = (\text{step}^\kappa \circ \text{next})^n(\text{now}^\kappa(y''))$$

implies $\triangleright^n(y' = y'')$. It therefore suffices to show that clock quantification commutes with existential quantifications that are unique in this sense.

4.4 Summary Requirements

Having analysed the applications in detail, we now summarise the requirements needed for universal quantification over clocks in order to express the applications above. They are

- Universal quantification needs to commute with certain quotient-inductive type constructors such as finite powersets and finite distributions
- Universal quantification needs to commute with bounded existential quantification, so over sets that are below a certain cardinality
- Universal quantification needs to commute with existential quantification that is essentially unique in the sense that if $X: \mathbf{U}_\Delta$ and $\phi: X \rightarrow \text{Prop}_{\Delta, \kappa}$ is such that $\phi(x) \wedge \phi(y)$ implies $(\triangleright^\kappa)^n(x = y)$, then $\forall \kappa. \exists x: X. \phi(x)$ implies $\exists(x: X). \forall \kappa. \phi(x)$.

5 The Presheaf Model

The section recalls the model of multi-clocked guarded recursion described by Manna et al. [18]. The model is indexed by a limit ordinal ρ , but Manna et al. only describe the specific case of $\rho = \omega$. All constructions generalise directly, however¹. The model uses a

¹ The abstract description of the left adjoint to $\blacktriangleright$ does not immediately generalise, but the concrete version, which is used throughout that paper, does.

presheaf category over a category $\mathbb{T}_\rho$ of time objects defined as pairs $(\mathcal{E}; \delta)$ where $\mathcal{E}$ is a finite subset of some fixed countably infinite set of clock names and $\delta : \mathcal{E} \rightarrow \rho$. A morphism from $(\mathcal{E}; \delta)$ to $(\mathcal{E}'; \delta')$ is a map $\sigma : \mathcal{E} \rightarrow \mathcal{E}'$ such that $\delta' \circ \sigma \leq \delta$ in the pointwise order.

Let $\mathbf{GR}$ be the category $\mathbf{PSh}(\mathbb{T}_\rho) \stackrel{\text{def}}{=} \mathbf{Set}^{\mathbb{T}_\rho}$ of *covariant* presheaves on $\mathbb{T}_\rho$. We write $\sigma \cdot x$ for $X(\sigma)(x)$ when X is a presheaf, $x \in X(\mathcal{E}; \delta)$ and $\sigma : (\mathcal{E}; \delta) \rightarrow (\mathcal{E}'; \delta')$. There is an object of clocks $\mathbf{Clk}$ in $\mathbf{GR}$ defined as $\mathbf{Clk}(\mathcal{E}; \delta) = \mathcal{E}$. This is used to model assumptions of the form $\kappa : \text{clock}$ in contexts.

The operation $\triangleright$ can be described semantically using an endofunctor $\blacktriangleright$ on the slice category $\mathbf{GR}/\mathbf{Clk}$. To describe this, first recall that any slice category of a presheaf category is equivalent to the category of presheaves over the elements of the slicing object. In this particular case, the category of elements $\int \mathbf{Clk}$ has as objects triples $(\mathcal{E}; \delta; \lambda)$, where $\lambda \in \mathcal{E}$ and morphisms from $(\mathcal{E}; \delta; \lambda)$ to $(\mathcal{E}'; \delta'; \lambda')$ are $\mathbb{T}_\rho$ -morphisms $\sigma : (\mathcal{E}; \delta) \rightarrow (\mathcal{E}'; \delta')$ such that $\sigma(\lambda) = \lambda'$. The modality $\blacktriangleright$ can then be defined as

$$(\blacktriangleright X)(\mathcal{E}; \delta; \lambda) = \lim_{\alpha < \delta(\lambda)} X(\mathcal{E}; \delta[\lambda \mapsto \alpha]; \lambda)$$

Note how this generalises the definition of $\blacktriangleright$ in the case of the topos of trees ($\mathbf{PSh}(\omega^{\text{op}})$): $(\blacktriangleright X)(0) = 1$, $(\blacktriangleright X)(n+1) = X(n)$ to higher ordinals and the multi-clocked setting.

The model constructed by Manna et al. follows the standard construction of a semantic model of type theory from presheaf models: Semantic contexts are objects Γ of $\mathbf{GR}$, and semantic types in semantic context Γ are presheaves over $\int \Gamma$. So a semantic type assigns, to each $(\mathcal{E}; \delta)$ and $\gamma \in \Gamma(\mathcal{E}; \delta)$ a set $X(\mathcal{E}; \delta; \gamma)$, and to each $\sigma : (\mathcal{E}; \delta) \rightarrow (\mathcal{E}'; \delta')$ a morphism $\sigma \cdot (-) : X(\mathcal{E}; \delta; \gamma) \rightarrow X(\mathcal{E}'; \delta'; \sigma \cdot \gamma)$. In order to interpret the clock irrelevance axiom, the model restricts attention to semantic types that are clock irrelevant in the following sense.

► Definition 3 ([9]). *Let Γ be an object of $\mathbf{GR}$. A presheaf X over $\int \Gamma$ is said to be invariant under clock introduction if, for any $(\mathcal{E}; \delta)$, $\lambda \notin \mathcal{E}$ and α , the map*

$$\iota \cdot (-) : X(\mathcal{E}; \delta; \gamma) \rightarrow X((\mathcal{E}, \lambda); \delta[\lambda \mapsto \alpha]; \iota \cdot \gamma)$$

is an isomorphism, where $\mathcal{E}, \lambda$ is shorthand for $\mathcal{E} \cup \{\lambda\}$ and $\iota : \mathcal{E} \rightarrow (\mathcal{E}, \lambda)$ is the inclusion.

For example, if Γ is the terminal object 1 , a presheaf X in $\mathbf{PSh}(\int 1) \cong \mathbf{GR}$ is invariant under clock introduction if and only if each $\sigma \cdot (-) : X(\emptyset; !) \rightarrow X(\mathcal{E}; \delta)$ is an isomorphism, and so X is a constant presheaf. Similarly, the subcategory of presheaves over $\int \mathbf{Clk}$ invariant under clock introduction is equivalent to the topos of trees $\mathbf{Set}^{\omega^{\text{op}}}$. On the other hand, $\mathbf{Clk}$ is an example of a presheaf that is not invariant under clock introduction.

The collection of presheaves invariant under clock introduction is closed under standard type formers such as Σ , Π and universal quantification over clocks [9]. Universal quantification over clocks in fact is just modelled as a Π -type indexed by $\mathbf{Clk}$. We recall [9] an alternative, direct definition of this which we will need in the following sections. Suppose Γ is an object of $\mathbf{GR}$ and A is a (semantic) dependent type over $\Gamma \times \mathbf{Clk}$. This means that for any $(\mathcal{E}; \delta)$, $\gamma \in \Gamma(\mathcal{E}; \delta)$ and $\lambda \in \mathcal{E}$, $A(\mathcal{E}; \delta; \gamma; \lambda)$ is a set. Suppose we have a function that assigns, to each $\mathcal{E}$ an element $\lambda_{\mathcal{E}} \notin \mathcal{E}$, and define $\forall(A)$ as a dependent type over Γ by

$$\forall(A)(\mathcal{E}; \delta; \gamma) \stackrel{\text{def}}{=} \lim_{\alpha < \rho} A((\mathcal{E}, \lambda_{\mathcal{E}}); \delta[\lambda_{\mathcal{E}} \mapsto \alpha]; \iota \cdot \gamma; \lambda_{\mathcal{E}})$$

where $\iota : \mathcal{E} \rightarrow (\mathcal{E}, \lambda_{\mathcal{E}})$ is the inclusion. The axiom (2) can then be proved as follows

$$\begin{aligned} \forall(\blacktriangleright A)(\mathcal{E}; \lambda; \gamma) &\stackrel{\text{def}}{=} \lim_{\alpha < \rho} \lim_{\beta < \alpha} A((\mathcal{E}, \lambda_{\mathcal{E}}); \delta[\lambda_{\mathcal{E}} \mapsto \beta]; \iota \cdot \gamma; \lambda_{\mathcal{E}}) \\ &\cong \lim_{\alpha < \rho} A((\mathcal{E}, \lambda_{\mathcal{E}}); \delta[\lambda_{\mathcal{E}} \mapsto \alpha]; \iota \cdot \gamma; \lambda_{\mathcal{E}}) \\ &= \forall(A)(\mathcal{E}; \lambda; \gamma) \end{aligned}$$

which works only because ρ is assumed to be a limit ordinal. Note also that this definition makes it clear that clock quantification distributes over sums.

► **Terminology.** *In the following, given an object Γ of GR, we shall refer to a presheaf X over $\int \Gamma$ which is invariant under clock introduction as a family over Γ . The sets $X(\mathcal{E}; \delta; \gamma)$ are referred to as the fibres of X , and if all fibres of X are subsingletons, we say that X is a predicate on Γ .*

5.1 Universes

If Δ is a clock context (so a finite set of clock variables), form the presheaf Clk^{Δ} in GR as the Δ -fold product of Clk by itself. Write $\text{GR}[\Delta]$ for $\text{PSh}(\int \text{Clk}^{\Delta})$. Assuming a set theoretic universe of small sets, there is an object $\mathcal{U}^{\Delta}$ in $\text{GR}[\Delta]$ and a family $\mathcal{E}l^{\Delta}$ over it such that

- $\mathcal{U}^{\Delta}$ is invariant under clock introduction
- For any Γ in $\text{GR}[\Delta]$ and A a family over Γ with small fibres, there is a unique map $\ulcorner A \urcorner : \Gamma \rightarrow \mathcal{U}^{\Delta}$ such that $A = \mathcal{E}l^{\Delta}[\ulcorner A \urcorner]$.
- This universe is closed under Π - and Σ -types, universal quantification over clocks, $\triangleright^{\kappa}$ for $\kappa \in \Delta$.

The construction uses a variant of the Hofmann-Streicher construction of universes in presheaf categories [16]: An element of $\mathcal{U}^{\Delta}(\mathcal{E}; \delta; f : \Delta \rightarrow \mathcal{E})$, is a family over $y(f[\Delta], \delta|_{f[\Delta]})$ in GR with small fibres, where y is the yoneda embedding, and $f[\Delta] \subseteq \mathcal{E}$ is the image of f . Concretely, this means that an element is a family of small sets A_{σ} indexed by the diagram whose components are maps $\sigma : (f[\Delta], \delta|_{f[\Delta]}) \rightarrow (\mathcal{E}', \delta')$, which is invariant under clocks in the sense that $\iota \cdot (-) : A_{\sigma} \rightarrow A_{\iota \circ \sigma}$ is an isomorphism for all ι as in Definition 3. For example, $\mathbf{U}_{\emptyset}$ is an object in $\text{GR}[\emptyset] \cong \text{GR}$, and this is the constant presheaf of small GR objects that are invariant under clock introduction. The universes of Clocked Type Theory can be modelled using these universes by appropriate reindexing. We refer to Manna et al. [18] for further details. For this paper we also need a family of universes of propositions.

► **Proposition 4.** *There exists an object $\mathcal{U}_{\text{Prop}}^{\Delta}$ of $\text{GR}[\Delta]$ and a predicate $\mathcal{E}l_{\text{Prop}}^{\Delta}$ on it such that*

1. $\mathcal{U}_{\text{Prop}}^{\Delta}$ is invariant under clock introduction
2. For any Γ in $\text{GR}[\Delta]$ and A predicate on Γ , there is a unique map $\ulcorner A \urcorner : \Gamma \rightarrow \mathcal{U}_{\text{Prop}}^{\Delta}$ such that $A \cong \mathcal{E}l_{\text{Prop}}^{\Delta}[\ulcorner A \urcorner]$.
3. This universe is closed under $\forall$ - and $\exists$ -quantification over objects invariant under clock introduction, finite conjunction and disjunction, universal quantification over clocks, and $\triangleright^{\kappa}$ for $\kappa \in \Delta$.

Proof. Let $\mathcal{U}_{\text{Prop}}^{\Delta}(\mathcal{E}; \delta; f : \Delta \rightarrow \mathcal{E})$ be the subset of $\mathcal{U}^{\Delta}(\mathcal{E}; \delta; f : \Delta \rightarrow \mathcal{E})$ consisting of families where each A_{σ} is a subset of $1 = \{\star\}$. The proof of almost all the properties above then carry over verbatim from the case of $\mathcal{U}^{\Delta}$, except for closure under existential quantification. For that, suppose Γ is an object of $\text{GR}[\Delta]$, A a family over Γ , $\ulcorner B \urcorner : \Gamma.A \rightarrow \mathcal{U}_{\text{Prop}}^{\Delta}$, and $B = \mathcal{E}l_{\text{Prop}}^{\Delta}[\ulcorner B \urcorner]$. Here

$$\Gamma.A(\mathcal{E}; \delta) = \{(\gamma, a) \mid \gamma \in \Gamma(\mathcal{E}; \delta) \wedge a \in A(\mathcal{E}; \delta; \gamma)\}$$

We must show that the presheaf $\exists(A, B)$ over $\int \Gamma$ defined by

$$\exists(A, B)(\mathcal{E}; \delta; \gamma) \stackrel{\text{def}}{=} \bigcup_{a \in A(\mathcal{E}; \delta; \gamma)} B(\mathcal{E}; \delta; \gamma; a)$$

has a code $\ulcorner \exists(A, B) \urcorner : \Gamma \rightarrow \mathcal{U}_{\text{Prop}}^\Delta$. By item 2 of the proposition, it suffices to show that $\exists(A, B)$ is invariant under clock introduction, and so a predicate. So let $\iota : (\mathcal{E}; \delta) \rightarrow ((\mathcal{E}, \lambda); \delta[\lambda \mapsto \alpha])$ be given by the inclusion of $\mathcal{E}$ into $(\mathcal{E}, \lambda)$. Then

$$\begin{aligned} \exists(A, B)((\mathcal{E}, \lambda); \delta[\lambda \mapsto \alpha]; \iota \cdot \gamma) &\stackrel{\text{def}}{=} \bigcup_{a \in A((\mathcal{E}, \lambda); \delta[\lambda \mapsto \alpha]; \iota \cdot \gamma)} B((\mathcal{E}, \lambda); \delta[\lambda \mapsto \alpha]; \iota \cdot \gamma; a) \\ &= \bigcup_{a \in A(\mathcal{E}; \delta; \gamma)} B((\mathcal{E}, \lambda); \delta[\lambda \mapsto \alpha]; \iota \cdot \gamma; \iota \cdot a) \\ &= \bigcup_{a \in A(\mathcal{E}; \delta; \gamma)} B(\mathcal{E}; \delta; \gamma; a) \\ &= \exists(A, B)(\mathcal{E}; \delta; \gamma) \end{aligned}$$

using invariance under clock introduction for A and B . ◀

5.2 Conservativity Over Set

One important consequence of the requirement that families be invariant under clock introduction is that the fragment of the type theory that does not mention clocks, ticks or guarded recursion is interpreted using constant presheaves only, and so the interpretation of this fragment agrees with that of the standard model in **Set**. In particular, any proof of such a statement in the model based on **GR** gives rise to a proof in the model based on **Set**.

► **Theorem 5.** *Let $\Gamma \vdash A$ type be a wellformed statement in Clocked Type Theory that can also be expressed in the internal type theory of **Set**. If there is a term t such that $\Gamma \vdash t : A$, then also the dependent type constructed by interpreting $\Gamma \vdash A$ type in **Set** can be inhabited.*

Note that Γ and A can be constructed using guarded recursion as long as they can be expressed in the internal language of **Set**. For example, any endofunctor F on **Set** mapping small sets to small sets induces a functor $F : \mathcal{U}_\emptyset \rightarrow \mathcal{U}_\emptyset$ in the internal language of our model. The type $\forall \kappa. \mu X. F(\mathfrak{p}^\kappa X)$ is a closed type, so is interpreted as a set. We can therefore view the statement that $\forall \kappa. \mu X. F(\mathfrak{p}^\kappa X)$ is a final coalgebra for F (Theorem 2) as a statement in the internal language of **Set**, and since this statement has a proof in Clocked Type Theory if F commutes with clock quantification, then, by Theorem 5, it also holds in the set-theoretic interpretation. Of course, this statement will talk about the action of F as an endofunctor on $\mathcal{U}_\emptyset$, which, as mentioned above is the set of all small objects of **GR** that are invariant under clock introduction. However, the notion of functor expressed using this universe in **Set** is equivalent to the one expressed using the usual universe of small sets. Many of the end results proved in the applications mentioned in Section 4, are of this form, for example, existence or non-existence of distributive laws of monads on **Set** or setoids, and contextual equivalence of programs of Probabilistic FPC. Their proofs use guarded recursion, but by Theorem 5 these proofs also prove their usual set theoretic interpretation.

6 Algebraic data types

The construction of lifting a set functor F preserving smallness to an endomap on a universe $\mathcal{U}^\emptyset$ extends to any of the universes $\mathcal{U}^\Delta$. For such a functor to commute with universal quantification over clocks means that the canonical map

$$F(\lim_{\alpha < \rho} A((\mathcal{E}, \lambda_\mathcal{E}); \delta[\lambda_\mathcal{E} \mapsto \alpha]; \iota \cdot \gamma; \lambda_\mathcal{E})) \rightarrow \lim_{\alpha < \rho} F(A((\mathcal{E}, \lambda_\mathcal{E}); \delta[\lambda_\mathcal{E} \mapsto \alpha]; \iota \cdot \gamma; \lambda_\mathcal{E}))$$

must be an isomorphism. Since A can be anything, this means that F needs to commute with ρ^{op} -indexed limits. Theorem 2 is therefore simply a synthetic restatement of Adámek’s fixed point theorem [1]: The type $\forall \kappa. \mu^\kappa X. F(\triangleright^\kappa X)$ is the limit of the terminal sequence

$$1 \leftarrow F(1) \leftarrow F^2(1) \leftarrow \dots \leftarrow F^\omega(1) \leftarrow F^{\omega+1} \leftarrow \dots$$

(where $F^\beta(1) = \lim_{\alpha < \beta} F^\alpha$ for limit ordinals β), and Adámek’s theorem states that this is a terminal coalgebra if F preserves ρ^{op} -indexed limits.

In this section we look at conditions for functors to commute with ordinal-indexed limits. We are particularly interested in the cases of monads generated by algebraic theories, as in the examples of finite powersets and finite distributions. First a few examples.

► **Example 6.** The finite powerset monad P_f does not commute with ω^{op} -limits. In fact the limit of the ω^{op} -long terminal sequence

$$1 \leftarrow P_f(1) \leftarrow P_f^2(1) \leftarrow \dots$$

contains infinite sets [2, 27]. Worrel [27] proved that the terminal sequence converges after $\omega + \omega$ steps, but in general P_f does not commute with $(\omega + \omega)^{\text{op}}$ -indexed limits. For a counter example, just append the terminal sequence to an ω -long sequence of 1. In general, however, P_f does commute with ω_1^{op} -limits, where ω_1 is the first uncountable ordinal [2].

The next example shows that some monads generated by algebraic theories do not commute with any ρ^{op} -limits.

► **Example 7.** Propositional truncation is the monad $T(X) = \{\star \mid \exists x : X.\text{true}\}$. This monad does not commute with ρ^{op} -limits for any ρ . Consider, for example, the sequence

$$X_\alpha = \{\beta \in \rho \mid \beta > \alpha\}$$

with maps $X_\alpha \rightarrow X_\gamma$ for $\gamma \leq \alpha$ given by inclusions. The limit of this sequence is empty, so also $T(\lim X) = \emptyset$, but $T(X_\alpha) = 1$, so $\lim T(X) = 1$.

6.1 A Semantic Condition

We now state conditions that imply that a functor preserves ρ^{op} -indexed limits. Let κ be a regular cardinal. A set Y is κ -small if it has cardinality strictly less than κ , and we write $Y \subseteq_\kappa X$ to mean that Y is a κ -small subset of X . Recall [4] that a functor $F : \mathbf{Set} \rightarrow \mathbf{Set}$ is said to be κ -accessible, if it preserves κ -filtered colimits. In particular, for any X , the map

$$\text{colim}_{X' \subseteq_\kappa X} F(X') \rightarrow F(X)$$

must be an isomorphism (in fact this statement is equivalent to being κ -accessible [3]). In words, this means that any $x \in F(X)$ lives already in $F(X')$ for some κ -small subset X' of X . Say that a functor F preserves pullbacks of monos, if, for any pullback diagram

$$\begin{array}{ccc} A & \longrightarrow & B \\ \downarrow & \lrcorner & \downarrow m \\ C & \longrightarrow & D \end{array}$$

where m is a mono, also F applied to the diagram is a pullback. In particular, this means that F preserves monos, also the non-split ones from the empty set.

Say that a functor F preserves κ -small intersections if, for all families $(X_i \subseteq X)_{i \in I}$ indexed by κ -small sets I , $F(\cap_{i \in I} X_i)$ is the limit of the diagram

$$\begin{array}{ccccc} F(X_i) & & \dots & & F(X_j) & & \dots \\ & \searrow & & & \swarrow & & \\ & F(X) & & & & & \end{array}$$

► **Theorem 8.** *Let $F : \mathbf{Set} \rightarrow \mathbf{Set}$ be a κ -accessible functor which preserves κ -small intersections and pullbacks of monomorphisms. If ρ is a regular ordinal, and there exists an $\alpha \in \rho$ of the same cardinality as κ , then F commutes with ρ^{op} -indexed limits.*

For the proof we need the following.

► **Lemma 9.** *Let ρ be a regular ordinal, and let $X : \rho^{\text{op}} \rightarrow \mathbf{Set}$ be a diagram whose maps $X_\beta \rightarrow X_\alpha$ are all surjective. If, for any $\alpha \in \rho$, there exists an ordinal $\beta > \alpha$ such that $X_\beta \rightarrow X_\alpha$ is not injective, then, for any $\alpha \in \rho$ there is a $\beta(\alpha)$ and an injection $\alpha \rightarrow X_{\beta(\alpha)}$.*

Proof. To construct $\beta(\alpha)$ we construct first a map $\gamma : \rho \rightarrow \rho$ by induction on the input: If α is a successor, so $\alpha = s(\alpha')$, define $\gamma(\alpha)$ to be an ordinal such that $X_{\gamma(\alpha)} \rightarrow X_{\gamma(\alpha')}$ is not injective. If α is a limit ordinal, define

$$\gamma(\alpha) = \bigvee_{\alpha' < \alpha} \gamma(\alpha')$$

In this case $\gamma(\alpha)$ is in ρ because ρ is regular. Using this, define $\beta(\alpha) = \gamma(s(\alpha))$. Next we construct injections $f_{\alpha'} : \alpha' \rightarrow X_{\beta(\alpha)}$ for $\alpha' < s(\alpha)$ by induction on α' such that

- $f_{\alpha'}$ extends $f_{\alpha''}$ whenever $\alpha'' < \alpha'$ and
- the composition $X_{\beta(\alpha), \gamma(\alpha')} \circ f_{\alpha'} : \alpha' \rightarrow X_{\beta(\alpha)} \rightarrow X_{\gamma(\alpha')}$ is injective for all α' .

Again, we proceed by cases of α' : If $\alpha' = s(\alpha'')$, since $X_{\gamma(\alpha')} \rightarrow X_{\gamma(\alpha'')}$ is not injective, and $X_{\beta(\alpha), \gamma(\alpha'')} \circ f_{\alpha''}$ is injective, there must be an element $x \in X_{\gamma(\alpha')}$ outside the image of $X_{\beta(\alpha), \gamma(\alpha'')} \circ f_{\alpha''}$. We can therefore extend $f_{\alpha''}$ to $f_{\alpha'}$ by picking $f_{\alpha'}(\alpha'')$ to be an element in $X_{\beta(\alpha)}$ that maps to x by $X_{\beta(\alpha), \gamma(\alpha')}$. When α' is a limit ordinal, we can simply define $f_{\alpha'}(\alpha'') = f_{s(\alpha'')}(\alpha'')$. ◀

Proof of Theorem 8. First note that if $x \in F(X)$, then there exists a smallest subset $X' \subseteq_\kappa X$ such that $x \in F(X')$ (leaving the action of F on the inclusion implicit): There exists an X' because F is κ -accessible, and the smallest is then the intersection of all $X' \setminus \{x'\}$ such that $x \in F(X' \setminus \{x'\})$.

Let X be a ρ^{op} -indexed diagram. We must show that the canonical map

$$F(\lim X) \rightarrow \lim F(X)$$

is an isomorphism. To show surjectivity, suppose $x \in \lim F(X)$. Let X'_α be the smallest subset of X_α such that $x_\alpha \in F(X'_\alpha)$. We show that these form a subdiagram of X . Suppose $\beta \leq \alpha$ and let $r : X_\alpha \rightarrow X_\beta$ be the restriction map. Then clearly $X'_\beta \subseteq r[X'_\alpha]$ by minimality of X'_β . Since F preserves pullbacks of monomorphisms, the diagram

$$\begin{array}{ccc} F(r^{-1}(X'_\beta) \cap X'_\alpha) & \longrightarrow & F(X'_\beta) \\ \downarrow & & \downarrow \\ F(X'_\alpha) & \longrightarrow & F(r[X'_\alpha]) \end{array}$$

is a pullback, so $x_\alpha \in F(r^{-1}(X'_\beta))$, and by minimality then $r^{-1}(X'_\beta) \cap X'_\alpha = X'_\alpha$. So

$$r[X'_\alpha] = r[r^{-1}(X'_\beta) \cap X'_\alpha] \subseteq X'_\beta$$

and so $X'_\beta = r[X'_\alpha]$. So X' is a subdiagram, and each map in X' is surjective.

We next show that the maps $X'_\beta \rightarrow X'_\alpha$ must be bijections from some point on. If not, by Lemma 9, for any $\alpha \in \rho$ there must be an injection of α into some $X'_{\beta(\alpha)}$, but since there is an α of cardinality κ , this contradicts $X'_{\beta(\alpha)}$ being κ -small.

So there must be a stage α such that all maps $X'_\beta \rightarrow X'_\alpha$ are isomorphisms. Then $F(\lim X') \cong F(X'_\alpha) \cong \lim F(X')$, and so x is in the image of the comparison map via the inclusion of $F(\lim X')$ into $F(\lim X)$:

$$\begin{array}{ccc} F(\lim X') & \xrightarrow{\cong} & \lim F(X') \ni x \\ \downarrow & & \downarrow \\ F(\lim X) & \longrightarrow & \lim F(X) \end{array}$$

For injectivity, suppose $a, b \in F(\lim X)$ map to the same element in $\lim F(X)$, so that $F(\pi_\alpha)a = F(\pi_\alpha)b$ for any α . Let $X'_a \subseteq \lim X$ be the smallest subset such that $a \in F(X'_a)$ and similarly for X'_b . For each $x, y \in X'_a$ with $x \neq y$ there is an α such that $x_\alpha \neq y_\alpha$. Let α be the upper bound of these as x, y range over all pairs of different elements in X'_a . Note that this is an element in ρ because ρ is regular, and because X_a has cardinality strictly smaller than some element in ρ . Then $\pi_\beta : X'_a \rightarrow \pi_\beta[X'_a]$ is a bijection whenever $\beta \geq \alpha$. Similarly, there is a γ such that for all $\beta \geq \gamma$, $\pi_\beta : X'_b \rightarrow \pi_\beta[X'_b]$ is a bijection. Let β be the least upper bound of α and γ . Then $\pi_\beta[X'_a]$ is the smallest subset of X_β such that $F(\pi_\beta)(a)$ is in $F(\pi_\beta[X'_a])$, because a smaller subset $X'' \subset \pi_\beta[X'_a]$ would give a subset $\pi_\beta^{-1}(X'') \subset X'_a$ with $a \in F(\pi_\beta^{-1}(X''))$. Similarly, $\pi_\beta[X'_b]$ is the smallest subset such that $F(\pi_\beta)(b)$ is in $F(\pi_\beta[X'_b])$. Since $F(\pi_\beta)(a) = F(\pi_\beta)(b)$, we must have $\pi_\beta[X'_b] = \pi_\beta[X'_a]$. So then, $X'_a \cong \pi_\beta[X'_a] \cong X'_b$ via an isomorphism f such that $f(x)_\beta = x_\beta$ from some β on. We conclude that $X'_a = X'_b$. So $a, b \in F(X'_a)$ map to the same element in $F(\pi_\beta(X'_a))$ by the isomorphism $F(\pi_\beta)$, and therefore must be equal. $\blacktriangleleft$

6.2 A Syntactic Condition

We recall the notion of κ -ary algebraic theory for κ a regular cardinal. A signature consists of a set of operations Σ all associated with an arity, which is a κ -small set. Given a signature, one can form the set $\mathbf{Tm}(X)$ of terms with free variables in X inductively in the standard way:

- If $x \in X$ then $x \in \mathbf{Tm}(X)$
- If $\text{op} \in \Sigma$ has arity A and $t : A \rightarrow \mathbf{Tm}(X)$, then $\text{op}(t) \in \mathbf{Tm}(X)$

An equation is a pair of terms (t, s) in some $\mathbf{Tm}(X)$. An algebraic theory is a signature together with a set of equations over that signature.

An algebraic theory induces a monad on $\mathbf{Set}$ by the definition

$$T(X) = \mathbf{Tm}(X) / \sim$$

where $\sim$ is the smallest congruence relation generated by the equations of the theory. It is well-known (see e.g. [4]) that the underlying functor of T is then κ -accessible.

► **Example 10.** The finite powerset monad is generated by the finitary ($\aleph_0$ -ary) theory consisting of two operations $\perp$ of arity $\emptyset$ and $\vee$ of arity $\{0, 1\}$. We write the latter as an infix binary operator. The equations are

$$x \vee y = y \vee x \quad (x \vee y) \vee z = x \vee (y \vee z) \quad x \vee x = x \quad x \vee \perp = x$$

► **Example 11.** The countable powerset monad is generated by the theory consisting of $\perp$ of arity $\emptyset$ and $\bigvee$ of arity $\mathbb{N}$. Following Chapman et al. [10], the equations can be expressed by writing $x \vee y$ for $\bigvee(x, y, y, y, \dots)$ as

$$\begin{aligned} x \vee y &= y \vee x & (x \vee y) \vee z &= x \vee (y \vee z) & x \vee x &= x & x \vee \perp &= x \\ x_i \vee \left(\bigvee x\right) &= \bigvee x & \left(\bigvee x\right) \vee y &= \bigvee (x_i \vee y)_{i \in \mathbb{N}} \end{aligned}$$

This is an $\aleph_1$ -ary algebraic theory.

► **Example 12.** The finite distributions monad is generated by a family of binary operations $\oplus_p$ indexed by p in the open interval $(0, 1)$ together with the equations

$$x \oplus_p x = x \quad x \oplus_p y = y \oplus_{1-p} x \quad (x \oplus_p y) \oplus_q z = x \oplus_{pq} (y \oplus_{\frac{(1-p)q}{1-pq}} z)$$

► **Example 13.** The propositional truncation monad of Example 7 is generated by the algebraic theory of no constructors and a single equation $(x = y)$.

We now give a syntactic condition for an algebraic theory ensuring that the generated monad satisfies the conditions of Theorem 8.

► **Definition 14.** An equation (t, s) is a drop equation if $\text{fv}(t) \neq \text{fv}(s)$.

Here $\text{fv}(t)$ for $t \in \text{Tm}(X)$ is the smallest subset $X' \subseteq X$ such that $t \in \text{Tm}(X')$. Of the four examples mentioned above, only propositional truncation has drop equations. Note that if a theory has no drop equations, then also the least congruence on terms generated by the equations has no drop equations.

► **Theorem 15.** Suppose T is a monad generated by a κ -ary algebraic theory with no drop equations. Then T preserves pullbacks of monomorphisms and κ -small intersections. As a consequence, if ρ is a regular ordinal with an $\alpha \in \rho$ of same cardinality as κ , then T preserves ρ^{op} -limits.

Proof. Let $f : X \rightarrow Y$ be a map and let $Z \subseteq Y$. We must show that the following is a pullback

$$\begin{array}{ccc} T(f^{-1}(Z)) & \xrightarrow{T(f)} & T(Z) \\ \downarrow & & \downarrow T(i) \\ T(X) & \xrightarrow{T(f)} & T(Y) \end{array}$$

where $i : Z \rightarrow Y$ is the inclusion. So suppose given $[s]_Z \in T(Z)$ and $[t]_X \in T(X)$ such that $T(i)[s]_Z = T(f)[t]_X$. Let t_f be the result of substituting the free variables of t using f , so that $T(f)[t]_X = [t_f]_Y$. The assumption says that t_f and s are related in the congruence relation generated by the equalities of the algebraic theory. Since none of these are drop equations, t_f and s have the same free variables. Since the free variables of s are in Z , the free variables of t must be in $f^{-1}(Z)$. So $[t] \in T(f^{-1}(Z))$ and maps to $[s]_Z \in T(Z)$ and $[t]_X \in T(X)$. This proves existence.

To prove uniqueness, it suffices to show that T preserves monomorphisms, implying that the map $T(f^{-1}(Z)) \rightarrow T(X)$ is mono. So suppose $A \subseteq B$, $j : A \rightarrow B$ is the inclusion, and that t and s are terms with free variables in A such that $T(j)[s] = T(j)[t]$. Since there are no more equations between terms in B than in A , also $[t] = [s]$.

Showing that T preserves κ -small intersections is similar: Given a family $([t_i] \in T(X_i))_{i \in I}$, with each $X_i \subseteq X$, the free variables in all t_i must be the same, so that they all represent the same equivalence class in $T(\cap_{i \in I} X_i)$. $\blacktriangleleft$

As a non-example, the pullback

$$\begin{array}{ccc} \emptyset & \longrightarrow & \{0\} \\ \downarrow & \lrcorner & \downarrow \\ \{1\} & \longrightarrow & \{0, 1\} \end{array}$$

witnesses that propositional truncation does not preserve pullbacks of monomorphisms.

7 Existential quantification

The previous section handled the first requirement listed in Section 4.4. We now look at the last two, which both concern commutativity of universal quantification over clocks and existential quantification. The property we need can be stated in Clocked Type Theory as

$$\Delta, X : \mathbf{U}_\Delta, \phi : X \rightarrow \forall \kappa. \text{Prop}_{\Delta, \kappa} \vdash \exists (x : X). \forall \kappa. \phi(x) [\kappa] = \forall \kappa. \exists (x : X). \phi(x) [\kappa] : \text{Prop}_\Delta \quad (4)$$

where $\Delta = \kappa_1 : \text{clock}, \dots, \kappa_n : \text{clock}$ is a clock context of length n . We first note that this principle does not hold in general.

► **Example 16.** Let Δ be the empty context and let $X = \rho$ be the indexing ordinal. Let ϕ be the predicate on $\rho \times \text{Clk}$ defined as

$$\phi(\mathcal{E}; \delta; \alpha; \lambda) = \{\star \mid \delta(\lambda) \leq \alpha\}$$

Since this predicate is invariant under clock introduction, it corresponds to a map $X \rightarrow \forall \kappa. \text{Prop}_{\Delta, \kappa}$, but $\forall \kappa. \exists (\alpha : \rho). \phi(\alpha) [\kappa] = \{\star\}$ and

$$\begin{aligned} (\exists (\alpha : \rho). \forall \kappa. \phi(\alpha) [\kappa])(\mathcal{E}; \delta) &= \{\star \mid \exists \alpha : \rho. \forall \beta. \phi((\mathcal{E}, \lambda_\mathcal{E}); \delta[\lambda_\mathcal{E} \mapsto \beta]; \alpha; \lambda_\mathcal{E})\} \\ &= \emptyset \end{aligned}$$

because the condition fails when $\beta > \alpha$. Such β always exist, because ρ is assumed to be a limit ordinal, as it needs to be to interpret axiom (2).

The above example is a variant of the well-known phenomenon that existential quantification in the internal language of a presheaf topos is local and does not imply global existence. This has been observed previously also in the setting of guarded recursion [7, 22].

To understand the principle more generally, we look at substitutions from some object Γ in $\mathbf{GR}$ into the context of (4). A mapping $\Gamma \rightarrow \text{Clk}^\Delta$ corresponds to an object of $\mathbf{GR}/\text{Clk}^\Delta$ which is equivalent to $\mathbf{GR}[\Delta]$, so a mapping into the context of (4) corresponds to an object Γ in $\mathbf{GR}[\Delta]$, a family X over Γ , and finally, a predicate ϕ on $X \times \text{Clk}$. The equality (4) then translates to equality of $(\exists (x : X). \forall \kappa. \phi(x))((\mathcal{E}, \delta); \chi; \gamma)$ which is

$$\{\star \mid \exists x : X((\mathcal{E}, \delta); \chi; \gamma). \forall \alpha < \rho. \phi((\mathcal{E}, \lambda_\mathcal{E}); \delta[\lambda_\mathcal{E} \mapsto \alpha]; \iota\chi; \iota \cdot \gamma; \iota \cdot x; \lambda_\mathcal{E})\} \quad (5)$$

and $(\forall \kappa. \exists (x : X). \phi(x))(\mathcal{E}; \delta; \chi; \gamma)$ which is

$$\{\star \mid \forall \alpha < \rho. \exists x : X((\mathcal{E}, \lambda_{\mathcal{E}}); \delta[\lambda_{\mathcal{E}} \mapsto \alpha]; \iota\chi; \iota \cdot \gamma). \phi((\mathcal{E}, \lambda_{\mathcal{E}}); \delta[\lambda_{\mathcal{E}} \mapsto \alpha]; \iota\chi; \iota \cdot \gamma; x; \lambda_{\mathcal{E}})\} \quad (6)$$

for all $\gamma \in \Gamma(\mathcal{E}; \delta; \chi)$ and $\chi : \Delta \rightarrow \mathcal{E}$.

► **Theorem 17.** *If ρ is a regular ordinal and all fibres $X(\mathcal{E}; \delta; \chi; \gamma)$ are of cardinality strictly smaller than ρ , then (5) and (6) are equal.*

Proof. Since X is invariant under clock introduction, (6) can be rewritten as

$$\{\star \mid \forall \alpha < \rho. \exists x : X(\mathcal{E}; \delta; \chi; \gamma). \phi((\mathcal{E}, \lambda_{\mathcal{E}}); \delta[\lambda_{\mathcal{E}} \mapsto \alpha]; \iota\chi; \iota \cdot \gamma; \iota \cdot x; \lambda_{\mathcal{E}})\} \quad (7)$$

Clearly the condition of (5) implies that of (7). For the other direction, first note that the condition is downward closed in α because, for $\beta \leq \alpha$, the identity on $(\mathcal{E}, \lambda_{\mathcal{E}})$ tracks a map σ which gives

$$\sigma \cdot (-) : \phi((\mathcal{E}, \lambda_{\mathcal{E}}); \delta[\lambda_{\mathcal{E}} \mapsto \alpha]; \iota\chi; \iota \cdot \gamma; \iota \cdot x; \lambda_{\mathcal{E}}) \rightarrow \phi((\mathcal{E}, \lambda_{\mathcal{E}}); \delta[\lambda_{\mathcal{E}} \mapsto \beta]; \iota\chi; \iota \cdot \gamma; \iota \cdot x; \lambda_{\mathcal{E}})$$

Now, arguing classically, suppose (5) is false. For each x there must exist an α_x , such that

$$\phi((\mathcal{E}, \lambda_{\mathcal{E}}); \delta[\lambda_{\mathcal{E}} \mapsto \beta]; \iota\chi; \iota \cdot \gamma; \iota \cdot x; \lambda_{\mathcal{E}})$$

is false for all $\beta \geq \alpha_x$. By the condition of the cardinality of the fibres of X and since ρ is regular, the least upper bound α of all the α_x is an ordinal which is smaller than ρ . This ordinal then proves (6) false. ◀

The above proof is essentially the same as the proof of the “existential property” forming part of the basis of Transfinite Iris [22, Theorem 5.2], but adapted to the multi-clocked setting. The existential property states that any existential proved in the logic of Transfinite Iris externalises to an existential in the meta-logic. Transfinite Iris uses a large indexing ordinal to ensure that the indexing ordinal is larger than the existentially quantified type. We will discuss this point further in the conclusion.

Universal quantification over clocks also commutes with essentially unique existential quantifications.

► **Theorem 18.** *Let Γ be an object of $\text{GR}[\Delta]$, X a family over Γ , and ϕ a predicate over $X \times \text{Clk}$, and let $n : \Gamma \rightarrow \mathbb{N}$ be a map of presheaves. If $\phi(x, \kappa) \wedge \phi(y, \kappa)$ implies $(\triangleright^{\kappa})^n(x = y)$ then (5) and (6) are equal.*

Proof. The predicate

$$\Delta, \Gamma, x : X, y : X, \kappa : \text{clock} \vdash (\triangleright^{\kappa})^n(x = y)$$

is true at $(\mathcal{E}; \delta; \chi; \gamma; x; y; \lambda)$ if and only if either $x = y$ or $\delta(\lambda) < n(\gamma)$. So, for α larger than $n(\gamma)$, there exists at most a single x such that

$$\phi((\mathcal{E}, \lambda_{\mathcal{E}}); \delta[\lambda_{\mathcal{E}} \mapsto \alpha]; \iota\chi; \iota \cdot \gamma; \iota \cdot x; \lambda_{\mathcal{E}})$$

Suppose now that (7) holds, and pick α larger than $n(\gamma)$. By (7) there is an x such that the above holds for that α and x . By downward closedness, it then also holds at all $\beta \leq \alpha$. For $\beta \geq \alpha$, there is a y such that the above holds at β for that y . By downwards closure and uniqueness, we then conclude $x = y$, so that x witnesses (5). ◀

8 Conclusion

We have seen that by choosing the indexing ordinal of the model to be sufficiently big, all the requirements of the applications studied in Section 4 can be met. For example, to program with finitely branching non-deterministic processes, and reasoning about bisimilarity of these using guarded recursion, as done by Kristensen et al [17] the indexing ordinal can be ω_1 (the first uncountable ordinal) or higher. For modelling Probabilistic FPC and reasoning about contextual equivalence of programs, the indexing ordinal needs to be larger than the cardinality of $D_f(D^\forall 1)$, which is uncountable. Finally, for reasoning about the coinductive delay monad up to weak bisimilarity as a monad on the category of setoids, one needs the indexing monad to be ω_1 or higher, such that quantification over clocks commutes with existential quantification over $\mathbb{N}$. In all these examples, we can now interpret the results proved in previous papers into the model and conclude the expected set theoretic results. For example, Theorem 7.7 of [21] now proves the existence of a distributive law of the monad generated by an algebraic theory with no drop equations over the coinductive delay monad up to weak bisimilarity as monads on the usual category of setoids. Likewise the results about contextual equivalence of Probabilistic FPC programs proved in [23] now hold for the operational semantics given in that paper as interpreted in the usual set-theoretic way, where $D^\forall$ is a coinductively defined monad.

The syntactic requirements on algebraic theories to generate monads commuting with clock quantification are quite natural from a type theoretic point of view: The arities should be of a particular form and the equations non-drop. The latter corresponds to the equations being typeable in a relevant type system (in the technical sense of substructural types). The requirements on cardinality for existential quantifications, on the other hand, are less natural in type theory. One way to avoid this is to use the ordinal of all ordinals as indexing, as done in Transfinite Iris [22]. Since this ordinal is large (lives in the next universe), it suffices to commute clock quantification with all small existential quantifications. This simplification comes at a price, however: Universal quantification over clocks now raises the universe level for small types. Monads such as $D^\forall$ defined using universe quantification will therefore only be monads on the category of sets in the next universe. In particular, the type $D^\forall 1$ will not be small, and so existential quantification over it will not automatically commute with universal quantification over clocks, unless one can prove $D^\forall 1$ isomorphic to a small set.

An alternative to proving type constructors commute with clock quantification is to use Worrells proof [27] that for any κ -accessible endofunctor F on **Set**, the terminal sequence converges in $\kappa + \kappa$ steps, producing a terminal coalgebra for F . This proves that the encoding of coinductive types for those functors is correct, externally. While this result is more general (in the sense that it works for all cardinals κ), it is also less natural and flexible from the type theoretic point of view. From the computational point of view, Kristensen et al. [17] develop a notion of induction under clocks that provides computational content to axioms of higher-inductive types or quotient inductive types commuting with clock quantification. It is not clear how to provide computational content to axioms of being a terminal coalgebra.

Multimodal Type Theory [15] gives a different approach to reasoning about the relationship between **Set** and the topos of trees in one type theory. Each of these categories correspond to a mode of the type theory, and the functors of constant presheaves and global elements are internalised as modalities between these two modes. While the use of modes makes the relationship between these categories explicit in the language, as we saw in Section 5 the same two models form submodels of the model of multiclocked guarded recursion. The results of Sections 6 and 7 can also be applied in the setting of Multimodal Type Theory.

References

- 1 Jiří Adámek. Free algebras and automata realizations in the language of categories. *Commentationes Mathematicae Universitatis Carolinae*, 15(4):589–602, 1974.
- 2 Jiří Adámek and Václav Koubek. On the greatest fixed point of a set functor. *Theoretical Computer Science*, 150(1):57–75, 1995. doi:10.1016/0304-3975(95)00011-K.
- 3 Jiří Adámek, Stefan Milius, Lurdes Sousa, and Thorsten Wissmann. On finitary functors. *Theory and Applications of Categories*, 34(35):1134–1164, 2019.
- 4 Jiří Adámek and Jiří Rosický. *Locally Presentable and Accessible Categories*, volume 189 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, 1994.
- 5 Robert Atkey and Conor McBride. Productive coprogramming with guarded recursion. *ACM SIGPLAN Notices*, 48(9):197–208, 2013. doi:10.1145/2500365.2500597.
- 6 Patrick Bahr., Hans Bugge Grathwohl, and Rasmus Ejlers Møgelberg. The clocks are ticking: No more delays! In *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–12. IEEE, 2017. doi:10.1109/LICS.2017.8005097.
- 7 Lars Birkedal, Rasmus Ejlers Møgelberg, Jan Schwinghammer, and Kristian Støvring. First steps in synthetic guarded domain theory: step-indexing in the topos of trees. *Logical Methods in Computer Science*, 8(4), 2012. doi:10.2168/LMCS-8(4:1)2012.
- 8 Aleš Bizjak, Hans Bugge Grathwohl, Ranald Clouston, Rasmus E Møgelberg, and Lars Birkedal. Guarded dependent type theory with coinductive types. In *International Conference on Foundations of Software Science and Computation Structures*, pages 20–35. Springer, 2016.
- 9 Aleš Bizjak and Rasmus Ejlers Møgelberg. Denotational semantics for guarded dependent type theory. *Mathematical Structures in Computer Science*, 30(4):342–378, 2020. doi:10.1017/S0960129520000080.
- 10 James Chapman, Tarmo Uustalu, and Niccolò Veltri. Quotienting the delay monad by weak bisimilarity. *Mathematical Structures in Computer Science*, 29(1):67–92, 2019. doi:10.1017/S0960129517000184.
- 11 Ranald Clouston. Fitch-style modal lambda calculi. In *International Conference on Foundations of Software Science and Computation Structures*, pages 258–275. Springer, 2018. doi:10.1007/978-3-319-89366-2_14.
- 12 Ranald Clouston, Bassel Mannaa, Rasmus Ejlers Møgelberg, Andrew M. Pitts, and Bas Spitters. Modal dependent type theory and dependent right adjoints. *Mathematical Structures in Computer Science*, 30(2):118–138, 2020. doi:10.1017/S0960129519000197.
- 13 Cyril Cohen, Thierry Coquand, Simon Huber, and Anders Mörtberg. Cubical type theory: A constructive interpretation of the univalence axiom. In *21st International Conference on Types for Proofs and Programs (TYPES 2015)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.TYPES.2015.5.
- 14 Eric Giovannini, Tingting Ding, and Max S. New. Denotational semantics of gradual typing using synthetic guarded domain theory. *Proc. ACM Program. Lang.*, 9(POPL):772–801, 2025. doi:10.1145/3704863.
- 15 Daniel Gratzer, GA Kavvos, Andreas Nuyts, and Lars Birkedal. Multimodal dependent type theory. In *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 492–506, 2020.
- 16 Martin Hofmann and Thomas Streicher. Lifting Grothendieck universes. Unpublished, 1999. URL: www.mathematik.tu-darmstadt.de/~streicher/NOTES/lift.pdf.
- 17 Magnus Baunsgaard Kristensen, Rasmus Ejlers Møgelberg, and Andrea Vezzosi. Greatest hits: Higher inductive types in coinductive definitions via induction under clocks. In Christel Baier and Dana Fisman, editors, *LICS '22: 37th Annual ACM/IEEE Symposium on Logic in Computer Science, Haifa, Israel, August 2 - 5, 2022*, pages 42:1–42:13. ACM, 2022. doi:10.1145/3531130.3533359.
- 18 Bassel Mannaa, Rasmus Ejlers Møgelberg, and Niccolò Veltri. Ticking clocks as dependent right adjoints: Denotational semantics for clocked type theory. *Logical Methods in Computer Science*, 16, 2020. URL: <https://lmcs.episciences.org/6980>.

- 19 Rasmus Ejlers Møgelberg. A type theory for productive coprogramming via guarded recursion. In *Proceedings of the Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–10, 2014. doi:10.1145/2603088.2603132.
- 20 Rasmus Ejlers Møgelberg and Andrea Vezzosi. Two guarded recursive powerdomains for applicative simulation. In Ana Sokolova, editor, *Proceedings 37th Conference on Mathematical Foundations of Programming Semantics, MFPS 2021, Hybrid: Salzburg, Austria and Online, 30th August - 2nd September, 2021*, volume 351 of *EPTCS*, pages 200–217, 2021. doi:10.4204/EPTCS.351.13.
- 21 Rasmus Ejlers Møgelberg and Maaïke Zwart. What monads can and cannot do with a few extra pages. *Logical Methods in Computer Science*, 21, 2025. doi:10.46298/LMCS-21(4:5)2025.
- 22 Simon Spies, Lennard Gäher, Daniel Gratzer, Joseph Tassarotti, Robbert Krebbers, Derek Dreyer, and Lars Birkedal. Transfinite iris: resolving an existential dilemma of step-indexed separation logic. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*, pages 80–95, 2021. doi:10.1145/3453483.3454031.
- 23 Philipp Stassen, Rasmus Ejlers Møgelberg, Maaïke Zwart, Alejandro Aguirre, and Lars Birkedal. Modelling recursion and probabilistic choice in guarded type theory. *Proc. ACM Program. Lang.*, 9(POPL):1417–1445, 2025. doi:10.1145/3704884.
- 24 The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. <https://homotopytypetheory.org/book>, Institute for Advanced Study, 2013.
- 25 Niccolò Veltri and Andrea Vezzosi. Formalizing π -calculus in guarded cubical agda. In *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*, CPP 2020, pages 270–283, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3372885.3373814.
- 26 Andrea Vezzosi, Anders Mörtberg, and Andreas Abel. Cubical agda: a dependently typed programming language with univalence and higher inductive types. *Proceedings of the ACM on Programming Languages*, 3(ICFP):1–29, 2019. doi:10.1145/3341691.
- 27 James Worrell. On the final sequence of a finitary set functor. *Theor. Comput. Sci.*, 338(1-3):184–199, 2005. doi:10.1016/j.tcs.2004.12.009.