Under consideration for publication in Math. Struct. in Comp. Science

Denotational semantics for guarded dependent type theory

ALEŠ BIZJAK[1†] and RASMUS EJLERS MØGELBERG[2]
[1] *Aarhus University*
[2] *IT University of Copenhagen*

Received February 11, 2018

We present a new model of Guarded Dependent Type Theory (GDTT), a type theory with guarded recursion and multiple clocks in which one can program with, and reason about coinductive types. Productivity of recursively defined coinductive programs and proofs is encoded in types using guarded recursion, and can therefore be checked modularly, unlike the syntactic checks implemented in modern proof assistants.
The model is based on a category of covariant presheaves over a category of time objects, and quantification over clocks is modelled using a presheaf of clocks. To model the *clock irrelevance axiom*, crucial for programming with coinductive types, types must be interpreted as presheaves orthogonal to the object of clocks. In the case of dependent types, this translates to a unique lifting condition similar to the one found in homotopy theoretic models of type theory. Since the universes defined by the standard Hofmann-Streicher construction in this model do not satisfy this property, the universes in GDTT must be indexed by contexts of clock variables. A large and technical part of the paper is devoted to showing that these can be constructed in such a way that inclusions between universes induced by inclusions of clock variable contexts commute on the nose with type operations on the universes.

1. Introduction

Type theories with dependent types such as Martin-Löf Type Theory (Martin-Löf, 1973), Homotopy Type Theory (Univalent Foundations Program, 2013), or the Extended Calculus of Constructions (Luo, 1994) are systems that can be simultaneously thought of as programming languages and logical systems. One reason why this is useful is that programs, their specification and the proof that a program satisfies this specification, can be expressed in the same language. In these systems, the logical interpretation of terms forces a totality requirement on the programming language, i.e., rules out general recursion, since nonterminating programs can inhabit any type, and thus be interpreted as proofs of false statements.

The lack of general recursion is a limitation both from a programming and a logical perspective. For example, when programming with coinductive types, the natural way to program and reason about these is by recursion. For example, the constant stream of zeros can be naturally described as the solution to the equation $\mathsf{zeros} = 0 :: \mathsf{zeros}$. To ensure logical consistency, such recursive

† Corresponding author. Full address: Department of Computer Science, Aabogade 34, DK-8200 Aarhus N, Denmark. Email: abizjak@cs.au.dk

definitions must be productive, in the sense that any finite segment of the stream can be computed in finite time. Modern proof assistants such as Coq (The Coq Development Team, 2004) and Agda (Norell, 2007) do support coinductive types and recursive definitions such as the above but the productivity checks are based on a syntactical analysis of terms, and are not modular. This means that using these in larger applications requires sophisticated tricks (Danielsson, 2010). This paper is concerned with a new technique using guarded recursion to express productivity in types.

Guarded recursion in the sense of Nakano (Nakano, 2000) is a safe way of adding recursion to type theory without breaking logical consistency. The idea is to guard all unfoldings of recursive equations by time steps in the form of a modal type constructor $\blacktriangleright$. The type $\blacktriangleright A$ should be thought of as a type of elements of A available one time step from now. Values can be preserved by time steps using an operator next satisfying $\mathsf{next}\, t : \blacktriangleright A$ whenever $t : A$. The fixed point operator has type $\mathsf{fix} : (\blacktriangleright A \to A) \to A$ and computes, for any f, a fixed point for $f \circ \mathsf{next}$. This is particularly useful when programming with *guarded recursive types*, i.e., recursive types where all occurrences of the type parameter appears guarded by a $\blacktriangleright$. For example, a guarded recursive type of streams would satisfy $\mathsf{Str} = \mathsf{Nat} \times \blacktriangleright \mathsf{Str}$ and the stream of zeros can be defined as $\mathsf{fix}(\lambda xs.\, \langle 0, xs\rangle)$. The type $\blacktriangleright \mathsf{Str} \to \mathsf{Str}$ in fact exactly captures productive recursive stream definitions. Using universes, the type Str can itself be computed as a guarded recursive fixed point. In this paper we use universes a la Tarski, i.e., for any term $A : \mathsf{U}$ there is a type $\mathsf{El}(A)$. If we assume an operation $\overline{\blacktriangleright} : \blacktriangleright U \to U$ satisfying $\mathsf{El}(\overline{\blacktriangleright}(\mathsf{next}(A))) = \blacktriangleright \mathsf{El}(A)$, then the type of guarded streams can be encoded as $\mathsf{Str} \stackrel{\text{def}}{=} \mathsf{El}(\mathsf{fix}(\lambda X.\, \mathsf{Nat} \times \overline{\blacktriangleright}(X)))$.

The guarded recursive type of streams above is not the usual type of streams. In particular, a term of type $\mathsf{Str} \to \mathsf{Str}$ must always be *causal* in the sense that the n first element of output only depend on the n first elements of input. Indeed, causality of maps is crucial for the encoding of productivity in types. On the other hand, a *closed* term of type Str does denote a full stream of numbers, and likewise a term of type Str in a context consisting solely of a variable $x : \mathsf{Nat}$ gives rise to an assignment of numbers to full streams of numbers. In general, this holds if the context is stable, i.e., consists entirely of time-independent types.

1.1. *Guarded recursion with multiple clocks*

Atkey and McBride (Atkey and McBride, 2013) proposed a way to program with coinductive types using this idea, expressing time-independence by indexing all $\blacktriangleright$ operators, next and fix by clocks. For example, if $t : A$ and κ is a clock, $\mathsf{next}^\kappa\, t : \overset{\kappa}{\blacktriangleright} A$ and the type $\overset{\kappa}{\blacktriangleright} A$ is to be thought of as elements of type A available one κ-time step from now. Likewise the guarded recursive type of streams must be indexed with a clock and assumed to satisfy $\mathsf{Str}^\kappa = \mathsf{Nat} \times \overset{\kappa}{\blacktriangleright} \mathsf{Str}^\kappa$. There are no operations on clocks, only clock variables, although we will see that a single clock constant can be useful. We refer to this as *guarded recursion with multiple clocks*, and the case of a single operator $\blacktriangleright$ as *guarded recursion with a single clock* or sometimes simply *the single clock case*.

In turn, *clock quantification* of guarded dependent type theory allows us to define the coinductive type of streams from the guarded recursive type of streams as $\forall\kappa.\, \mathsf{Str}^\kappa$. Clock quantification behaves similarly to the dependent product type in the sense it has analogous introduction and elimination rules; terms of this type are introduced by clock abstraction $\Lambda\kappa.t$, and eliminated using clock application $t[\kappa']$, provided κ' is a valid clock. However clock quantification additionally satisfies the *clock irrelevance* property, which is crucial for showing that types such as $\forall\kappa.\, \mathsf{Str}^\kappa$

satisfy the properties expected of coinductive types, i.e., that they are final coalgebras. Using these constructs and properties we can program with streams using guarded recursion, ensuring productivity of definitions using types.

This paper presents a model of GDTT (Bizjak et al., 2016), an extensional type theory with guarded recursion and clocks, in which one can program with, and reason about guarded recursive and coinductive types. To motivate some of the constructions of GDTT, we now take a closer look at the encoding of coinductive streams as $\forall\kappa.\,\mathsf{Str}^\kappa$. As a minimal requirement for this to work, we need an isomorphism of types $\forall\kappa.\,\mathsf{Str}^\kappa \cong \mathsf{Nat} \times \forall\kappa.\,\mathsf{Str}^\kappa$. This isomorphism is a composition of three isomorphisms

$$\begin{aligned}\forall\kappa.\,\mathsf{Str}^\kappa &= \forall\kappa.\,\mathsf{Nat} \times \overset{\kappa}{\blacktriangleright}\mathsf{Str}^\kappa \\ &\cong (\forall\kappa.\,\mathsf{Nat}) \times \forall\kappa.\,\overset{\kappa}{\blacktriangleright}\mathsf{Str}^\kappa \\ &\cong \mathsf{Nat} \times \forall\kappa.\,\overset{\kappa}{\blacktriangleright}\mathsf{Str}^\kappa \\ &\cong \mathsf{Nat} \times \forall\kappa.\,\mathsf{Str}^\kappa\end{aligned}$$

We shall see in this paper that the first isomorphism follows from the fact that $\forall\kappa.A$ behaves essentially as the dependent product type $\prod(\kappa : \mathsf{clock}).A$, and thus distributes over binary products. For the second isomorphism, we need $\mathsf{Nat} \cong \forall\kappa.\,\mathsf{Nat}$. One direction of this isomorphism maps $x : \mathsf{Nat}$ to $\Lambda\kappa.x$, and the opposite way evaluates an element in $\prod(\kappa : \mathsf{clock}).A$ at a *clock constant* κ_0. The composition on Nat is obviously the identity, but for the other composition to be the identity, we need to assume the η-axiom for $\forall\kappa.A$, and the *clock irrelevance axiom*, which states that whenever $t : \forall\kappa.A$ and κ is not in A, then evaluating t at different clocks give the same result. One of the main contributions of this paper is that this axiom can be modelled using a notion of orthogonality. The last isomorphism requires an inverse $\mathsf{force} : \forall\kappa.\,\overset{\kappa}{\blacktriangleright} A \to \forall\kappa.A$ to the map induced by next^κ.

In this paper we focus on modelling GDTT, and refer to the reader to (Møgelberg, 2014) for a proof of correctness of the coinductive type encodings.

1.2. *A model of guarded recursion with multiple clocks*

In the single clock case guarded recursion can be modelled in the topos of trees, i.e., the category $\mathrm{Set}^{\omega^{\mathrm{op}}}$ of presheaves over the ordered natural numbers ω. In this model, a closed type is modelled as a sequence of sets $(X_n)_{n\in\mathbb{N}}$ together with restriction maps $X_{n+1} \to X_n$. We think of X_n as the type as it looks if we have n steps to reason about it. For example, in the guarded recursive type of streams, since the tail takes one computation step to compute, one can compute the $n+1$ first elements of the stream in n steps. We can represent this by the object defined as $\mathsf{Str}_n = \mathbb{N}^{n+1}$ with restriction maps as projections.

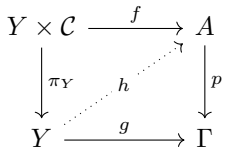
In this model $\blacktriangleright X$ is the object given by $(\blacktriangleright X)_0 = 1$ and $(\blacktriangleright X)_{n+1} = X_n$. Redefining Str_n to be $\mathbb{N}^{n+1} \times 1$ (and associating products to the right) one gets $\mathsf{Str} = \mathbb{N} \times \blacktriangleright\mathsf{Str}$. In the empty context a term $t : \overset{\kappa}{\blacktriangleright} A \to A$ is modelled as a family of maps $t_{n+1} : A_n \to A_{n+1}$ and $t_0 : 1 \to A_0$. The fixed point operator maps such a family to the global element $\mathsf{fix}(t) : 1 \to A$ defined as $\mathsf{fix}(t)_n = t_n \circ \cdots \circ t_0$. We refer to (Birkedal et al., 2012) for further details.

In this paper we extend this to a model of guarded recursion with multiple clocks. The model is a presheaf category over a category $\mathbb{T}$ of time objects. In the single clock case, a time object was simply a number indicating the number of ticks left on the unique clock. In the multi clock case,

a time object consists of a finite number of clocks $\mathcal{E}$, together with a map $\delta : \mathcal{E} \to \mathbb{N}$ indicating the number of ticks left on each clock. A morphism of time objects $\sigma : (\mathcal{E}, \delta) \to (\mathcal{E}', \delta')$ is a map $\sigma : \mathcal{E} \to \mathcal{E}'$ such that $\delta'(\sigma(\lambda)) \leq \delta(\lambda)$ for each $\lambda \in \mathcal{E}$. Such a morphism can rename clocks, introduce new clocks (elements of $\mathcal{E}'$ outside the image of σ) and even synchronise clocks (by mapping them to the same clock). The inequality requirement corresponds to the inequalities between numbers in the topos of trees.

We consider *covariant* presheaves on $\mathbb{T}$, i.e., the category of functors $\mathbb{T} \to \mathrm{Set}$. In this category there is an object of clocks given by $\mathcal{C}(\mathcal{E}, \delta) = \mathcal{E}$, which we use to model clock variables. Clock quantification is modelled as a dependent product over $\mathcal{C}$. With this interpretation, for a type A in which κ does not appear free, the type $\forall\kappa.A$ is modelled as a simple function type $\mathcal{C} \to A$. The clock irrelevance axiom mentioned above, then states that the map $A \to (\mathcal{C} \to A)$ mapping an element x in A to the constant map to x is an isomorphism. Of course this does not hold for all presheaves A, and so we must show that this holds for the interpretation of any type. Note that it does not hold for $A = \mathcal{C}$, and so, although $\forall\kappa.A$ is modelled as a dependent product over the presheaf of clocks, there is no type of clocks in the type theory. This is similar to the status of the interval in cubical type theory (Cohen et al., 2016), which is not itself a type, but still the set of types is closed under dependent products over the interval (these are path types).

For dependent types the condition becomes a unique lifting property. Recall that a type A depending on a context Γ is modelled as a context projection map $p : A \to \Gamma$. These must satisfy the condition that for all Y, and for all commutative squares as in the outer square below (where π_Y is the projection), there exists a unique h such that the two triangles commute.



This condition is similar to the notion of fibration used in models of homotopy theoretic models of type theory (Awodey and Warren, 2009; Kapulkin and Lumsdaine, 2012) and cubical type theory (Bezem et al., 2013), and is proved by induction on the structure of types. We say that such a map p is *orthogonal* to $\mathcal{C}$.

1.3. *Universes*

Since our model is a presheaf category, one would hope that modelling universes would follow the standard Hofmann-Streicher construction (Hofmann and Streicher, 1999), restricting to the elements orthogonal to $\mathcal{C}$. Unfortunately, this universe $\mathcal{U}$ is not itself orthogonal to $\mathcal{C}$. The reason is that there is a map $\mathcal{U} \times \mathcal{C} \to \mathcal{U}$ mapping a type A and a clock κ to $\overset{\kappa}{\blacktriangleright} A$, and this map is not constant in the $\mathcal{C}$ component. This is a new semantic manifestation of a known problem, and we follow the solution used in GDTT, which is to have a family of universes $(\mathsf{U}_\Delta)_\Delta$ in the syntax, indexed by finite sets of clock variables. Each universe U_Δ is to be thought of as the universe of types independent of the clocks outside of Δ, and the type operation $\overset{\kappa}{\blacktriangleright}$ is restricted on the universe U_Δ to the κ in Δ.

This means that universes are indexed by a new dimension, similar to the indexing of universes by natural numbers used to avoid Russels paradox (Martin-Löf, 1973). Fortunately, there are inclusions $\mathsf{U}_\Delta \to \mathsf{U}_{\Delta'}$ for $\Delta \subseteq \Delta'$, and a large and rather technical part of the paper is devoted

to proving universe polymorphism in this dimension. This means that operations on types such as dependent product can be defined on the universes in such a way that they commute with the inclusions mentioned above, not just up to isomorphism, but indeed up to identity. This requires great care when modelling the universes and the operations on them. We hope that, as a consequence of this result, the indexing of universes by clock contexts can be suppressed in practical applications, just like the indexing by natural numbers is often suppressed.

1.4. *Related work*

The notion of guarded recursion studied in this paper originates with Nakano (Nakano, 2000). Much of the recent interest in guarded recursion is due to the guarded recursive types, which can even have negative occurences and thus, by adding $\blacktriangleright$ operators in appropriate places, provide approximations to solutions to equations that can not be solved in set theory. These have been used to construct syntactic models and operational reasoning principles for (also combinations of) advanced programming language features including general references, recursive types, countable non-determinism and concurrency (Birkedal et al., 2012; Bizjak et al., 2014; Svendsen and Birkedal, 2014). This technique can be understood as an abstract form of step-indexing (Appel and McAllester, 2001), the connection to which was first discovered by Appel et al (Appel et al., 2007). Most of these applications have been constructed using logics with guarded recursion, such as the internal language of the topos of trees (Birkedal et al., 2012), but recently GDTT has been used to construct denotational models of programming languages like FPC (Møgelberg and Paviotti, 2016), modelling the recursive types of these as guarded recursive types.

Most type theories with guarded recursion considered until now have been extensional, with the exception of *guarded cubical type theory* (Birkedal et al., 2016). This has, however, only been developed in the single clock case, although there exists an experimental version with multiple clocks.

Guarded recursion with multiple clocks was first developed in the simply typed setting by Atkey and McBride (Atkey and McBride, 2013). Møgelberg (Møgelberg, 2014) extended these results to a model of dependent type theory and proved correctness of the coinductive type encodings inside a type theory with guarded recursion. These two early works used a restricted version of clock application, allowing $t[\kappa'] : A\,[\kappa'/\kappa]$ for $t : \forall\kappa.A$ only if κ' does not appear free in $\forall\kappa.A$. This condition can be thought of as disallowing the clocks κ and κ' to be synchronised in A, and was motivated by the models considered at the time. This restriction has unfortunate consequences for the syntactic metatheory. In particular, the present authors do not know how to prove type preservation for clock β-reductions in these systems.

This led us to suggest a different model (Bizjak and Møgelberg, 2015) given by a family of presheaf categories $\mathsf{GR}(\Delta)$ indexed by clock contexts (finite sets of clock variables) Δ. This model should in principle lead to a model of GDTT, but this was never done in detail, due to a problem with modelling substitution of clock variables. Such substitutions are given by maps $\sigma : \Delta \to \Delta'$ and must correspond semantically to functors $\mathsf{GR}(\Delta) \to \mathsf{GR}(\Delta')$. While these functors can be defined in a natural way, they do not commute with dependent function types up to identity, only up to isomorphism. This problem can be thought of as a coherence problem, similar to the one arising when modelling type theory in locally cartesian closed categories (Hofmann, 1994). It is very likely that Hofmann's solution to the latter problem (Hofmann, 1994) can be adapted to construct an equivalent family of categories for which the functors preserve construction on

the nose, but we prefer the solution presented here, which organises all these categories inside one big presheaf category, thereby reducing the model construction to the known construction of modelling type theory in a presheaf category. The precise relation to the categories $\mathsf{GR}(\Delta)$ is presented in Section 9.

Recently, GDTT has been refined to *clocked type theory* (CloTT) (Bahr et al., 2017), which has better operational properties, and indeed strong normalisation has been proved for clocked type theory in the setting without identity types. The main novel feature of CloTT is the notion of ticks on a clock introduced in contexts as assumptions of the form $\alpha : \kappa$, for κ a clock. Ticks can be used to encode the delayed substitutions (see Section 6) of GDTT, and reduce most of the equalities between these to β and η equalities.

We are currently working on extending the model of this paper to a model of CloTT. The major complication in doing this, is that there appears to be no object of ticks on a type. Instead, the operation of extending a context by a tick must be modelled in a different, more complicated way. For this reason, we have chosen to present the model in the simpler setting of GDTT first. We expect that much of the work presented here, including the universes, can be reused for the future model of CloTT.

In recent unpublished work on guarded computational type theory Harper and Sterling (private communication) have independently discovered a presheaf category very similar to the one used in this paper.

1.5. *Overview*

Section 2 presents a basic type theory Core-GDTT for guarded recursion with multiple clocks. This can be thought of as the core of GDTT (Bizjak et al., 2016) although we use a slightly different presentation. Section 3 then presents a basic model of Core-GDTT in the presheaf category $\mathrm{Set}^{\mathbb{T}}$, and Section 4 shows how to model the clock irrelevance axiom. The following sections 5 and 6 then extend Core-GDTT with extensional identity types and delayed substitutions, a construction needed for reasoning about guarded recursive and coinductive types. The second half of the paper (Sections 7 and 8) are devoted to universes and modelling universe polymorphism in the clock context dimension. Finally the relations to the categories $\mathsf{GR}(\Delta)$ constructed in previous work (Bizjak and Møgelberg, 2015) by the authors is discussed in Section 9.

2. A basic type theory for guarded recursion

This section introduces Core-GDTT a presentational variant of a fragment of the type theory GDTT (Bizjak et al., 2016). The fragment is the one not mentioning universes, delayed substitutions and identity types. All these will be treated in Sections 5–8. The variation referred to above is in the treatment of clocks, which in previous work (Bizjak et al., 2016; Møgelberg, 2014; Bizjak and Møgelberg, 2015) had a separate context. Here we simply include them in the context as if they were ordinary variables to simplify the presentation of the denotational semantics. Section 2.1 sketches an equivalence between Core-GDTT and the corresponding fragment of GDTT.

The rules for context formation, type judgements and equalities can be found in Figure 1. Note that clock has a special status. In particular, it is not a type. Its status is similar to that of the interval type in cubical type theory (Cohen et al., 2016). Ignoring $\blacktriangleright$ and the clock irrelevance

Wellformed contexts

$$\frac{}{\cdot \vdash} \qquad \frac{\Gamma \vdash A\ \mathsf{type} \qquad x \notin \Gamma}{\Gamma, x : A \vdash} \qquad \frac{\Gamma \vdash \qquad \kappa \notin \Gamma}{\Gamma, \kappa : \mathsf{clock} \vdash}$$

Wellformed clocks

$$\frac{\kappa : \mathsf{clock} \in \Gamma}{\Gamma \vdash \kappa : \mathsf{clock}}$$

Type formation

$$\frac{\Gamma, x : A \vdash B\ \mathsf{type}}{\Gamma \vdash \prod (x : A).B\ \mathsf{type}} \qquad \frac{\Gamma, x : A \vdash B\ \mathsf{type}}{\Gamma \vdash \sum (x : A).B\ \mathsf{type}}$$

$$\frac{\Gamma \vdash A\ \mathsf{type} \qquad \Gamma \vdash \kappa : \mathsf{clock}}{\Gamma \vdash \overset{\kappa}{\blacktriangleright} A\ \mathsf{type}} \qquad \frac{\Gamma, \kappa : \mathsf{clock} \vdash A\ \mathsf{type}}{\Gamma \vdash \forall \kappa. A\ \mathsf{type}}$$

Typing judgements

$$\frac{}{\Gamma, x : A, \Gamma' \vdash x : A} \qquad \frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : \prod (x : A).B} \qquad \frac{\Gamma \vdash t : \prod (x : A).B \qquad \Gamma \vdash u : A}{\Gamma \vdash t\, u : B\,[u/x]}$$

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash u : B\,[t/x]}{\Gamma \vdash \langle t, u \rangle : \sum (x : A).B} \qquad \frac{\Gamma \vdash t : \sum (x : A).B}{\Gamma \vdash \pi_1 t : A} \qquad \frac{\Gamma \vdash t : \sum (x : A).B}{\Gamma \vdash \pi_2 t : B\,[\pi_1 t/x]}$$

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash \kappa : \mathsf{clock}}{\Gamma \vdash \mathsf{next}^\kappa\, t : \overset{\kappa}{\blacktriangleright} A} \qquad \frac{\Gamma, \kappa : \mathsf{clock} \vdash t : \overset{\kappa}{\blacktriangleright} A}{\Gamma \vdash \mathsf{prev}\, \kappa.t : \forall \kappa. A} \qquad \frac{\Gamma, x : \overset{\kappa}{\blacktriangleright} A \vdash t : A}{\Gamma \vdash \mathsf{fix}^\kappa\, x.t : A}$$

$$\frac{\Gamma, \kappa : \mathsf{clock} \vdash t : A}{\Gamma \vdash \Lambda \kappa.t : \forall \kappa. A} \qquad \frac{\Gamma \vdash t : \forall \kappa. A \qquad \Gamma \vdash \kappa' : \mathsf{clock}}{\Gamma \vdash t\big[\kappa'\big] : A\,\big[\kappa'/\kappa\big]} \qquad \frac{\Gamma \vdash t : A \qquad \Gamma \vdash A \equiv B}{\Gamma \vdash t : B}$$

Equalities

$$(\lambda x.t)u = t\,[u/x] \qquad \lambda x.tx = t \quad (\text{if } x \notin t)$$

$$\pi_i\, \langle t_1, t_2 \rangle = t_i \qquad \langle \pi_1 t, \pi_2 t \rangle = t$$

$$(\Lambda \kappa.t)\kappa' = t\,\big[\kappa'/\kappa\big] \qquad \Lambda \kappa.t[\kappa] = t \quad (\text{if } \kappa \notin t)$$

$$\mathsf{prev}\, \kappa.\, (\mathsf{next}^\kappa\, t) = \Lambda \kappa.t \qquad \mathsf{next}^\kappa\, ((\mathsf{prev}\, \kappa.t)[\kappa]) = t$$

$$\mathsf{fix}^\kappa\, x.t = t\,[\mathsf{next}^\kappa (\mathsf{fix}^\kappa\, x.t)/x]$$

Clock irrelevance axiom

$$\frac{\Gamma \vdash t : \forall \kappa. A \qquad \Gamma \vdash A\ \mathsf{type} \qquad \Gamma \vdash \kappa' : \mathsf{clock} \qquad \Gamma \vdash \kappa'' : \mathsf{clock}}{\Gamma \vdash t\big[\kappa'\big] \equiv t\big[\kappa''\big] : A}$$

Figure 1. Syntax of Core-GDTT, a fragment of GDTT.

axiom, the type theory Core-GDTT is in fact just a fragment of a type theory with a base type clock in which types like $\prod(x : A).\mathsf{clock}$ or $\sum(\kappa : \mathsf{clock}).A$ are not allowed. Under this view, the type $\forall\kappa.A$ can be thought of as a dependent product type $\prod(\kappa : \mathsf{clock}).A$, in fact its basic behaviour is exactly like a dependent product, as can be seen from the equality rules. We make use of this view to establish soundness of the model given in Section 3. What distinguishes it from an ordinary dependent product is the clock irrelevance axiom stated at the bottom of Figure 1. The assumption $\Gamma \vdash A\ \mathsf{type}$ mean that κ is not free in A, and so in that case $\forall\kappa.A$ reduces to a simple function space $\mathsf{clock} \to A$. The axiom states that all maps of this type are constant. In Section 4 we explain how to model the type theory with this additional axiom.

In Figure 1 the equalities should be understood as equalities of terms in a context. For brevity we have omitted the context in most statements except the clock irrelevance axiom, which, unlike the other rules, is type directed.

The term constructor $\mathsf{prev}\,\kappa$ is a restricted elimination form for $\overset{\kappa}{\blacktriangleright}$, and binds κ. An unrestricted eliminator $\mathsf{prev}\,\kappa$ would be unsafe, because terms of the form $\mathsf{fix}^{\kappa}\, x.\, \mathsf{prev}\,\kappa.t$ would inhabit any type. As the model presented in this paper shows, however, it is safe to eliminate a $\overset{\kappa}{\blacktriangleright}$, as long as κ does not appear in the ordinary (non-clock) variables of the context. This is ensured in the rule for $\mathsf{prev}\,\kappa$ by requiring that κ is at the end of the context. One might have expected a simpler rule of the form

$$\frac{\Gamma, \kappa : \mathsf{clock} \vdash t : \overset{\kappa}{\blacktriangleright} A}{\Gamma, \kappa : \mathsf{clock} \vdash \mathsf{prev}\,\kappa.t : A}$$

but this rule is not closed under substitution of clock variables. This problem is solved by binding κ.

The notion of substitution of contexts is defined in the standard way using the rules

$$\frac{}{\cdot : \Gamma \to \cdot} \qquad \frac{\rho : \Gamma \to \Gamma' \qquad \Gamma \vdash t : A\rho}{\rho[x \mapsto t] : \Gamma \to \Gamma', x : A}$$

$$\frac{\rho : \Gamma \to \Gamma' \qquad \Gamma \vdash \kappa' : \mathsf{clock}}{\rho[\kappa \mapsto \kappa'] : \Gamma \to \Gamma', \kappa : \mathsf{clock}}$$

and likewise the notion of substitution is defined in the standard way.

Lemma 2.1. If $\rho : \Gamma \to \Gamma'$ is a substitution, then

— if $\Gamma' \vdash A\ \mathsf{type}$ also $\Gamma \vdash A\rho\ \mathsf{type}$
— if $\Gamma' \vdash t : A$ also $\Gamma \vdash t\rho : A\rho$

Some example terms. We refer to previous work for more extensive and detailed motivation and explanation of the usage of the type theory. We briefly show here some example terms on streams. The type Str^{κ} of *guarded streams* of natural numbers is the unique type satisfying $\mathsf{Str}^{\kappa} \equiv \mathbf{N} \times \overset{\kappa}{\blacktriangleright} \mathsf{Str}^{\kappa}$. To understand this example it is not important how this type can be defined, only that it satisfies the stated judgemental equality. For readers familiar with guarded dependent type theory we remark that it can be defined as using the guarded fixed point on the universe $\mathsf{U}_{\{\kappa\}}$ as outlined in the introduction of this paper. Using the mentioned judgemental equality we

can type

$$\mathsf{head}^\kappa : \mathsf{Str}^\kappa \to \mathbf{N} \qquad \mathsf{tail}^\kappa : \mathsf{Str}^\kappa \to \overset{\kappa}{\blacktriangleright} \mathsf{Str}^\kappa$$
$$\mathsf{head}^\kappa \triangleq \lambda xs.\pi_1(xs) \qquad \mathsf{tail}^\kappa \triangleq \lambda xs.\pi_2(xs)$$

Notice that the tail^κ introduces a $\overset{\kappa}{\blacktriangleright}$ modality: The tail of a guarded streams is only available later. This prevents non-productive stream definitions. However once the streams are defined we wish to use them without introducing later modalities. This can be achieved by the type Str of *streams* of natural numbers. It is defined from the type of guarded streams as $\mathsf{Str} \triangleq \forall\kappa.\, \mathsf{Str}^\kappa$. Thus, the tail function on *streams* is defined as

$$\mathsf{tail} : \mathsf{Str} \to \mathsf{Str}$$
$$\mathsf{tail} \triangleq \lambda xs.\, \mathsf{prev}\, \kappa\, \mathsf{tail}^\kappa(xs[\kappa])$$

2.1. *Relation to previous presentations*

Judgements of GDTT as presented in (Bizjak et al., 2016) have a separate context for clock variables. For example, typing judgements of have the form $\Gamma \vdash_\Delta t : A$ where Δ is a clock context of the form $\kappa_1, \ldots \kappa_n$, and Γ consists exclusively of ordinary variable declarations. The two presentations are equivalent in the sense that $\Gamma \vdash_\Delta t : A$ is a valid judgement in the presentation of (Bizjak et al., 2016) iff $\kappa_1 : \mathsf{clock}, \ldots, \kappa_n : \mathsf{clock}, \Gamma \vdash t : A$ is valid in the presentation used here.

Another minor difference is that GDTT as presented in (Bizjak et al., 2016) has a clock constant κ_0. The clock constant can be easily added to Core-GDTT by a precompilation adding a fresh clock variable to the left of the context in each judgement.

3. A presheaf model

This section defines the category $\mathrm{Set}^{\mathbb{T}}$ as that of covariant presheaves on the category of *time objects* $\mathbb{T}$. As any presheaf category, $\mathrm{Set}^{\mathbb{T}}$ has enough structure to model dependent type theory. The category $\mathrm{Set}^{\mathbb{T}}$ contains an object $\mathcal{C}$ of *clocks* which can be used to model clock quantification and guarded recursion. We show that $\mathrm{Set}^{\mathbb{T}}$ validates almost all the rules of guarded dependent type theory, apart from the clock irrelevance axiom, which is the topic of Section 4.

We write **Fin** for the category of finite sets and functions whose objects are finite subset $\mathcal{E}$ of some given, countably infinite, set of clocks.[†]

Definition 3.1. Let $\mathbb{T}$ be the category with objects pairs $(\mathcal{E}, \delta)$ with $\mathcal{E} \in \mathbf{Fin}$ and $\delta : \mathcal{E} \to \mathbb{N}$ a function. A morphism $(\mathcal{E}, \delta) \to (\mathcal{E}', \delta')$ in $\mathbb{T}$ is a *function* $\tau : \mathcal{E} \to \mathcal{E}'$ in **Fin** such that $\delta' \circ \tau \leq \delta$ in the pointwise ordering.

We use λ to range over elements of $\mathcal{E}$ and write $\mathcal{E}, \lambda$ for the union of $\mathcal{E}$ with $\{\lambda\}$ assuming $\lambda \notin \mathcal{E}$. Likewise, when $\mathcal{E}$ and $\mathcal{E}'$ are disjoint, we write $\mathcal{E}, \mathcal{E}'$ for their union. We use the notation $\delta[\lambda \mapsto n]$ for both the update of δ (when $\lambda \in \mathcal{E}$) and the extension of δ (when $\lambda \notin \mathcal{E}$).

[†] The assumption that the objects are subset of a fixed set, as opposed to arbitrary finite sets keeps the category **Fin**, and thus also $\mathbb{T}$, small, thus simplifying definitions of, e.g., dependent products. Moreover it is needed to satisfy Assumption 8.1.

The indexing category $\mathbb{T}$ should be thought of as a category of *time objects*. A *time object* is a finite set of semantic clocks $\mathcal{E}$ which each have a finite number of ticks left on them as indicated by δ. During a computation, three things can happen: time can pass on the existing clocks, as captured by a map $\mathrm{id}_{\mathcal{E}} : (\mathcal{E}, \delta) \to (\mathcal{E}, \delta')$ where $\delta' \leq \delta$, new clocks can be introduced as captured by set inclusions $i : (\mathcal{E}, \delta) \to ((\mathcal{E}, \lambda), \delta[\lambda \mapsto n])$, and clocks can be synchronised as captured by a map

$$\mathrm{id}_{\mathcal{E}}[\lambda \mapsto \lambda'', \lambda' \mapsto \lambda''] : ((\mathcal{E}, \lambda, \lambda'), \delta[\lambda \mapsto n, \lambda' \mapsto m]) \to ((\mathcal{E}, \lambda''), \delta[\lambda'' \mapsto \mathsf{min}(n, m)]).$$

Finally, clocks can be renamed, e.g., via an isomorphism $\sigma : \mathcal{E} \cong \mathcal{E}'$ inducing an isomorphism $\sigma : (\mathcal{E}, \delta \circ \sigma) \to (\mathcal{E}', \delta)$. Any map in the indexing category $\mathbb{T}$ can be written as a composition of these three kinds of maps.

Variables of the form $\kappa : \mathsf{clock}$ will be modelled as the *object of clocks* $\mathcal{C}$, which is simply the first projection

$$\mathcal{C}(\mathcal{E}, \delta) = \mathcal{E}.$$

Lemma 3.1. Let λ be a clock. There is an isomorphism of objects of $\mathrm{Set}^{\mathbb{T}}$

$$\mathcal{C} \cong \varinjlim_{n \in \mathbb{N}} y\,(\{\lambda\}, n)$$

where $y : \mathbb{T}^{\mathrm{op}} \to \mathrm{Set}^{\mathbb{T}}$ is the (co)Yoneda embedding, and we write $(\{\lambda\}, n)$ for the $\mathbb{T}$ object $(\{\lambda\}, [\lambda \mapsto n])$, i.e., the first component is the singleton containing λ, and the second component is the map which maps λ to n.

Proof. The objects of the diagram are

$$\begin{aligned} y\,(\{\lambda\}, n)\,(\mathcal{E}, \delta) &= \mathsf{Hom}_{\mathbb{T}}((\{\lambda\}, n)\,, (\mathcal{E}, \delta)) \\ &\cong \{\lambda' \in \mathcal{E} \mid \delta(\lambda') \leq n\} \end{aligned}$$

and up to this isomorphism, the arrows are inclusions of sets. Since colimits are compute pointwise in presheaf categories, the isomorphism follows. □

As mentioned above, the fragment of Core-GDTT not mentioning $\blacktriangleright$, can be considered a fragment of dependent type theory with a single base type for clocks. This can be modelled soundly in any presheaf category once the interpretation of the base type is given, using standard techniques (Hofmann, 1997). We now briefly recall how this is done in our case.

3.1. *Interpreting type theory in categories of presheaves*

A context $\Gamma \vdash$ is modelled as an object of $\mathrm{Set}^{\mathbb{T}}$. Concretely, this means that for each $(\mathcal{E}, \delta) \in \mathbb{T}$ we have a set $[\![\Gamma]\!]_{(\mathcal{E},\delta)}$, and for each morphism $\sigma : (\mathcal{E}, \delta) \to (\mathcal{E}', \delta')$ a restriction map

$$\sigma \cdot - : [\![\Gamma]\!]_{(\mathcal{E},\delta)} \to [\![\Gamma]\!]_{(\mathcal{E}',\delta')}$$

satisfying the following two functoriality properties

$$\mathrm{id} \cdot x = x \tag{1}$$

$$(\sigma \circ \tau) \cdot x = \sigma \cdot (\tau \cdot x). \tag{2}$$

A type in context $\Gamma \vdash A\ \mathsf{type}$ is modelled as an object $[\![\Gamma \vdash A\ \mathsf{type}]\!]$, sometimes written simply

as $[\![A]\!]$, over the category of elements of the presheaf $[\![\Gamma]\!]$. Concretely this means that for each $(\mathcal{E},\delta) \in \mathbb{T}$ and $\gamma \in [\![\Gamma]\!]_{(\mathcal{E},\delta)}$ there is a set $A_{(\mathcal{E},\delta)}(\gamma)$ and for each morphism $\sigma : (\mathcal{E},\delta) \to (\mathcal{E}',\delta')$ in $\mathbb{T}$ and $\gamma \in [\![\Gamma]\!]_{(\mathcal{E},\delta)}$ a restriction map

$$\sigma \cdot (-) : A_{(\mathcal{E},\delta)}(\gamma) \to A_{(\mathcal{E}',\delta')}(\sigma \cdot \gamma)$$

satisfying the functoriality properties (1) and (2). Note that the notation $\sigma \cdot x$ is overloaded both for a restriction of contexts as well as types.

A term $\Gamma \vdash t : A$ is modelled as a family of elements

$$[\![t]\!]_{(\mathcal{E},\delta)}(\gamma) \in [\![A]\!]_{(\mathcal{E},\delta)}(\gamma)$$

indexed by $(\mathcal{E},\delta) \in \mathbb{T}$ and $\gamma \in [\![\Gamma]\!]_{(\mathcal{E},\delta)}$, such that for any morphism $\sigma : (\mathcal{E},\delta) \to (\mathcal{E}',\delta')$ in $\mathbb{T}$ and any $\gamma \in [\![\Gamma]\!]_{(\mathcal{E},\delta)}$ we have

$$\sigma \cdot [\![t]\!]_{(\mathcal{E},\delta)}(\gamma) = [\![t]\!]_{(\mathcal{E}',\delta')}(\sigma \cdot \gamma).$$

A substitution $\rho : \Gamma \to \Gamma'$ is modelled as a morphism from the presheaf $[\![\Gamma]\!]$ to the presheaf $[\![\Gamma']\!]$. Concretely for each $(\mathcal{E},\delta)$ in $\mathbb{T}$ and $\gamma \in [\![\Gamma]\!]_{(\mathcal{E},\delta)}$ there is an element $[\![\rho]\!]_{(\mathcal{E},\delta)}(\gamma) \in [\![\Gamma']\!]_{(\mathcal{E},\delta)}$. It is defined in Figure 2. In the case of clocks, the figure uses that an element $\gamma \in [\![\Gamma \vdash]\!]_{(\mathcal{E},\delta)}$ is a tuple of the same length as Γ, and thus can be seen as a map defined on the variables in Γ. In particular, if κ is in Γ, then $\gamma(\kappa) \in \mathcal{E}$.[‡] The interpretation satisfies that for any $\sigma : (\mathcal{E},\delta) \to (\mathcal{E}',\delta')$ we have

$$\sigma \cdot [\![\rho]\!]_{(\mathcal{E},\delta)}(\gamma) = [\![\rho]\!]_{(\mathcal{E}',\delta')}(\sigma \cdot \gamma).$$

A summary of the interpretation of types and terms of Core-GDTT is in Figures 2 and 3. As stated quantification over clocks is interpreted like dependent product types, but we include it in Figure 2 for completeness. The interpretation of $\overset{\kappa}{\blacktriangleright}$ and guarded recursion (also included in the figures) will be discussed in the next sections.

The restriction maps are not defined in the figure. In the case of comprehension, this is defined as

$$\sigma \cdot (\gamma, x) = (\sigma \cdot \gamma, \sigma \cdot x).$$

and likewise the case of dependent sum types is defined pointwise. In the case of dependent products, the restriction is by precomposition, i.e., if $\tau : (\mathcal{E},\delta) \to (\mathcal{E}',\delta')$ then

$$\tau \cdot (f_\sigma)_{\sigma:(\mathcal{E},\delta)\to(\mathcal{E}'',\delta'')} = (f_{\sigma\tau})_{\sigma:(\mathcal{E}',\delta')\to(\mathcal{E}'',\delta'')}$$

We will not recall the details of the proof of soundness of the interpretation, but just recall the crucial substitution lemma, which must be proved simultaneously with the definition of the interpretation. This lemma will be used in the following sections to define the interpretations of the constructions of GDTT, and must likewise be extended to these constructions.

Lemma 3.2. If $\rho : \Gamma \to \Gamma'$ is a substitution, then

— if $\Gamma' \vdash A$ type then

$$[\![\Gamma \vdash A\rho \text{ type}]\!]_{(\mathcal{E},\delta)}(\gamma) = [\![\Gamma' \vdash A \text{ type}]\!]_{(\mathcal{E},\delta)}([\![\rho]\!]_{(\mathcal{E},\delta)}(\gamma))$$

[‡] An alternative way to understand $\gamma(\kappa)$ is to view the clock κ as a morphism $[\![\kappa]\!] : [\![\Gamma \vdash]\!] \to \mathcal{C}$. In this view $\gamma(\kappa)$ is the application of $[\![\kappa]\!]_{(\mathcal{E},\delta)}$ to $\gamma \in [\![\Gamma \vdash]\!]_{(\mathcal{E},\delta)}$.

— if $\Gamma' \vdash t : A$ then

$$[\![\Gamma \vdash t\rho : A\rho]\!]_{(\mathcal{E},\delta)}(\gamma) = [\![\Gamma' \vdash t : A]\!]_{(\mathcal{E},\delta)}([\![\rho]\!]_{(\mathcal{E},\delta)}(\gamma))$$

3.2. *Modelling* $\blacktriangleright$ *and guarded recursion*

Extending the interpretation to include the later modality and associated term constructor is straightforward, as we now explain.

If $\delta(\gamma(\kappa)) > 0$ we write $\delta^{-\gamma(\kappa)}$ for the function which agrees with δ everywhere except on $\gamma(\kappa)$ where $\delta^{-\gamma(\kappa)}(\gamma(\kappa)) = \delta(\gamma(\kappa)) - 1$. It is elementary that the identity function is a morphism of type

$$(\mathcal{E},\delta) \to \left(\mathcal{E}, \delta^{-\gamma(\kappa)}\right).$$

in $\mathbb{T}$. We write $\mathrm{tick}^{\gamma(\kappa)}$ for this morphism.

With this notation we can define the interpretation of $\blacktriangleright$ as

$$\left[\!\!\left[\overset{\kappa}{\blacktriangleright} A\right]\!\!\right]_{(\mathcal{E},\delta)}(\gamma) = \begin{cases} \{\star\} & \text{if } \delta(\gamma(\kappa)) = 0 \\ [\![A]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}(\mathrm{tick}^{\gamma(\kappa)} \cdot \gamma) & \text{otherwise} \end{cases}$$

with restrictions given by those of $[\![A]\!]$ or uniquely determined by the fact that the codomain is the singleton set.

Analogously, the term $\mathsf{next}^\kappa\, t$ is modelled as

$$[\![\mathsf{next}^\kappa\, t]\!]_{(\mathcal{E},\delta)}(\gamma) = \begin{cases} \star & \text{if } \delta(\gamma(\kappa)) = 0 \\ [\![t]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}(\mathrm{tick}^{\gamma(\kappa)} \cdot \gamma) & \text{otherwise} \end{cases}$$

Modelling previous. Rather than modeling prev directly, we model the construct

$$\frac{\Gamma \vdash t : \forall\kappa.\overset{\kappa}{\blacktriangleright} A}{\Gamma \vdash \mathsf{force}\, t : \forall\kappa.A}$$

Using this, one can define prev as

$$\mathsf{prev}\,\kappa.t \overset{\mathrm{def}}{=} \mathsf{force}(\Lambda\kappa.t) \tag{3}$$

In the following, we will assume a choice of fresh clock names $\mathcal{E} \mapsto \lambda_\mathcal{E}$, such that $\lambda_\mathcal{E} \notin \mathcal{E}$. Given $(\mathcal{E},\delta) \in \mathbb{T}$ and $\gamma \in [\![\Gamma]\!]_{(\mathcal{E},\delta)}$ let $\iota^n : (\mathcal{E},\delta) \to ((\mathcal{E},\lambda_\mathcal{E}), \delta[\lambda_\mathcal{E} \mapsto n])$ be the inclusion for $n \in \mathbb{N}$.

Lemma 3.3. Let $\Gamma, \kappa : \mathsf{clock} \vdash A\ \mathsf{type}$, $(\mathcal{E},\delta) \in \mathbb{T}$ and $\gamma \in [\![\Gamma]\!]_{(\mathcal{E},\delta)}$. The set

$$[\![\Gamma \vdash \forall\kappa.A\ \mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma)$$

is isomorphic to the set

$$\{(f_n)_{n\in\mathbb{N}} \mid \forall n. f_n \in [\![\Gamma, \kappa : \mathsf{clock} \vdash A\ \mathsf{type}]\!]_{((\mathcal{E},\lambda_\mathcal{E}),\delta[\lambda_\mathcal{E}\mapsto n])}(\iota^n \cdot \gamma, \lambda_\mathcal{E}), \forall n. \mathrm{tick}^{\lambda_\mathcal{E}} \cdot f_{n+1} = f_n\}$$

i.e., to the limit of the diagram

$$X_0 \xleftarrow{\mathrm{tick}^{\lambda_\mathcal{E}} \cdot -} X_1 \xleftarrow{\mathrm{tick}^{\lambda_\mathcal{E}} \cdot -} X_2 \xleftarrow{\mathrm{tick}^{\lambda_\mathcal{E}} \cdot -} \cdots$$

Interpretation of contexts

$$[\![\cdot \vdash]\!]_{(\mathcal{E},\delta)} = \{\star\}$$
$$[\![\Gamma, x : A \vdash]\!]_{(\mathcal{E},\delta)} = \left\{(\gamma, x) \;\middle|\; \gamma \in [\![\Gamma \vdash]\!]_{(\mathcal{E},\delta)}, x \in [\![\Gamma \vdash A \;\mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma)\right\}$$

Interpretation of substitutions

$$[\![\rho : \Gamma \to \cdot]\!]_{(\mathcal{E},\delta)}(\gamma) = \star$$
$$[\![\rho[x \mapsto t] : \Gamma \to \Gamma', x : A]\!]_{(\mathcal{E},\delta)}(\gamma) = \left([\![\rho : \Gamma \to \Gamma']\!]_{(\mathcal{E},\delta)}(\gamma), [\![\Gamma \vdash t : A\rho]\!]_{(\mathcal{E},\delta)}(\gamma)\right).$$
$$[\![\rho[\kappa \mapsto \kappa'] : \Gamma \to \Gamma', \kappa : \mathsf{clock}]\!]_{(\mathcal{E},\delta)}(\gamma) = \left([\![\rho : \Gamma \to \Gamma']\!]_{(\mathcal{E},\delta)}(\gamma), \gamma(\kappa')\right).$$

Interpretation of types

$$[\![\Gamma \vdash \textstyle\sum (x : A).B \;\mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma) =$$
$$\left\{(a, b) \;\middle|\; a \in [\![\Gamma \vdash A \;\mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma), b \in [\![\Gamma, x : A \vdash B \;\mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma, a)\right\}$$
$$[\![\Gamma \vdash \textstyle\prod (x : A).B \;\mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma) =$$
$$\left\{(f_\sigma)_{\sigma:(\mathcal{E},\delta)\to(\mathcal{E}',\delta')} \;\middle|\; \begin{array}{l}\forall \mathcal{E}', \delta', \sigma : (\mathcal{E},\delta) \to (\mathcal{E}', \delta'), \forall a \in [\![\Gamma \vdash A \;\mathsf{type}]\!]_{(\mathcal{E}',\delta')}(\sigma \cdot \gamma),\\ f_\sigma(a) \in [\![\Gamma, x : A \vdash B \;\mathsf{type}]\!]_{(\mathcal{E}',\delta')}(\sigma \cdot \gamma, a) \text{ such that}\\ \tau \cdot f_\sigma(a) = f_{\tau\circ\sigma}(\tau \cdot a) \text{ for composable } \tau, \sigma\end{array}\right\}$$

$$[\![\forall \kappa.A]\!]_{(\mathcal{E},\delta)}(\gamma) = \left\{(f_\sigma)_{\sigma:(\mathcal{E},\delta)\to(\mathcal{E}',\delta')} \;\middle|\; \begin{array}{l}\forall \mathcal{E}', \delta', \forall \sigma : (\mathcal{E},\delta) \to (\mathcal{E}', \delta'), \forall \kappa \in \mathcal{E}',\\ f_\sigma(\kappa) \in [\![A]\!]_{(\mathcal{E}',\delta')}(\sigma \cdot \gamma, \kappa) \text{ such that}\\ \tau \cdot f_\sigma(\kappa) = f_{\tau\circ\sigma}(\tau(\kappa)) \text{ for composable } \tau, \sigma\end{array}\right\}$$

$$\left[\!\!\left[\blacktriangleright^\kappa A\right]\!\!\right]_{(\mathcal{E},\delta)}(\gamma) = \begin{cases}\{\star\} & \text{if } \delta(\gamma(\kappa)) = 0\\ [\![A]\!]_{\mathcal{E},\delta^{-\gamma(\kappa)}}(\mathrm{id}_{\mathcal{E}}^{\gamma(\kappa)} \cdot \gamma) & \text{otherwise}\end{cases}$$

Figure 2. Interpretation of types, contexts and substitutions of Core-GDTT.

$$[\![\mathsf{next}^\kappa\, t]\!]_{(\mathcal{E},\delta)}(\gamma) = \begin{cases}\star & \text{if } \delta(\gamma(\kappa)) = 0\\ [\![t]\!]_{\mathcal{E},\delta^{-\gamma(\kappa)}}(\mathrm{id}_{\mathcal{E}}^{\gamma(\kappa)} \cdot \gamma) & \text{otherwise}\end{cases}$$
$$[\![\Gamma \vdash \Lambda\kappa.t : \forall \kappa.A]\!]_{(\mathcal{E},\delta)}(\gamma) = \left([\![\Gamma, \kappa : \mathsf{clock} \vdash t : A]\!]_{(\mathcal{E}',\delta')}(\sigma \cdot \gamma, -)\right)_{\sigma:(\mathcal{E},\delta)\to(\mathcal{E}',\delta')}$$
$$[\![\Gamma \vdash t[\kappa'] : A[\kappa'/\kappa]]\!]_{(\mathcal{E},\delta)}(\gamma) = \left([\![\Gamma \vdash t : \forall \kappa.A]\!]_{(\mathcal{E},\delta)}(\gamma)\right)_{\mathrm{id}}(\gamma(\kappa'))$$

Figure 3. Interpretation of selected terms of Core-GDTT.

where

$$X_n = [\![\Gamma, \kappa : \mathsf{clock} \vdash A \;\mathsf{type}]\!]_{((\mathcal{E},\lambda_\mathcal{E}),\delta[\lambda_\mathcal{E}\mapsto n])}(\iota^n \cdot \gamma, \lambda_\mathcal{E}).$$

Up to this isomorphism, the restriction map

$$\sigma \cdot (-) : [\![\Gamma \vdash \forall \kappa.A \;\mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma) \to [\![\Gamma \vdash \forall \kappa.A \;\mathsf{type}]\!]_{(\mathcal{E}',\delta')}(\sigma \cdot \gamma)$$

maps the family $(f_n)_n$ to the family $(\sigma[\lambda_{\mathcal{E}} \mapsto \lambda_{\mathcal{E}'}] \cdot f_n)_n$, and the interpretation of terms is

$$\left(\llbracket \Gamma \vdash \Lambda\kappa.t : \forall\kappa.A \rrbracket_{(\mathcal{E},\delta)}(\gamma)\right)_n = \llbracket \Gamma, \kappa : \mathsf{clock} \vdash t : A \rrbracket_{((\mathcal{E},\lambda_{\mathcal{E}}),\delta[\lambda_{\mathcal{E}} \mapsto n])}(\iota^n \cdot \gamma, \lambda_{\mathcal{E}})$$

$$\llbracket \Gamma \vdash t[\kappa'] : A[\kappa'/\kappa] \rrbracket_{(\mathcal{E},\delta)}(\gamma) = \mathrm{id}_{\mathcal{E}}[\lambda_{\mathcal{E}} \mapsto \gamma(\kappa')] \cdot \left(\llbracket \Gamma \vdash t : \forall\kappa.A \rrbracket_{(\mathcal{E},\delta)}(\gamma)\right)_{\delta(\gamma(\kappa'))}$$

Proof. An element in $\llbracket \Gamma \vdash \forall\kappa.A\ \mathsf{type} \rrbracket_{(\mathcal{E},\delta)}(\gamma)$ is given by a family (f_σ) indexed by $\sigma : (\mathcal{E},\delta) \to (\mathcal{E}',\delta')$. One direction of the isomophism maps such a family to the family

$$\phi((f_\sigma)_\sigma) = (f_{\iota^n}(\lambda_{\mathcal{E}}))_n$$

For the opposite direction, note that for any $\sigma : (\mathcal{E},\delta) \to (\mathcal{E}',\delta')$ and any $\lambda \in \mathcal{E}'$, σ can be factored through $\iota^{\delta'(\lambda)}$ as in the following diagram

$$\begin{array}{ccc} (\mathcal{E},\delta) & \xrightarrow{\iota^{\delta'(\lambda)}} & ((\mathcal{E},\lambda_{\mathcal{E}}),\delta[\lambda_{\mathcal{E}} \mapsto \delta'(\lambda)]) \\ & \searrow^{\sigma} & \downarrow{\scriptstyle \sigma[\lambda_{\mathcal{E}} \mapsto \lambda]} \\ & & (\mathcal{E}',\delta') \end{array} \tag{4}$$

The other direction of the isomorphism can then be defined as

$$(\psi((f_n)_n))_\sigma(\lambda) = \sigma[\lambda_{\mathcal{E}} \mapsto \lambda] \cdot f_{\delta'(\lambda)}$$

To see that this is well-defined, we must show that

$$(\psi((f_n)_n))_{\tau\sigma}(\tau(\lambda)) = \tau \cdot ((\psi((f_n)_n))_\sigma(\lambda))$$

whenever $\tau : (\mathcal{E}',\delta') \to (\mathcal{E}'',\delta'')$. For this, let

$$\omega : ((\mathcal{E},\lambda_{\mathcal{E}}),\delta[\lambda_{\mathcal{E}} \mapsto \delta'(\lambda)]) \to ((\mathcal{E},\lambda_{\mathcal{E}}),\delta[\lambda_{\mathcal{E}} \mapsto \delta''(\tau(\lambda))])$$

be the map tracked by the identity, and note that $\tau(\sigma[\lambda_{\mathcal{E}} \mapsto \lambda]) = (\tau\sigma)[\lambda_{\mathcal{E}} \mapsto \tau(\lambda)]$. Now

$$\begin{aligned} (\psi((f_n)_n))_{\tau\sigma}(\tau(\lambda)) &= (\tau\sigma)[\lambda_{\mathcal{E}} \mapsto \tau(\lambda)] \cdot f_{\delta'(\tau(\lambda))} \\ &= (\tau\sigma)[\lambda_{\mathcal{E}} \mapsto \tau(\lambda)] \cdot (\omega \cdot f_{\delta(\lambda)}) \\ &= \tau(\sigma[\lambda_{\mathcal{E}} \mapsto \lambda]) \cdot f_{\delta(\lambda)} \\ &= \tau \cdot ((\psi((f_n)_n))_\sigma(\lambda)) \end{aligned}$$

This proves that ϕ and ψ are well-defined. We must show that they are each others inverses. To this end we compute

$$\begin{aligned} \phi(\psi((f_m)_m))_n &= (\psi((f_m)_m))_{\iota^n}(\lambda_{\mathcal{E}}) \\ &= \iota^n[\lambda_{\mathcal{E}} \mapsto \lambda_{\mathcal{E}}] \cdot f_{(\delta[\lambda_{\mathcal{E}} \mapsto n])(\lambda_{\mathcal{E}})} \\ &= f_n \end{aligned}$$

where the last line uses $\iota^n[\lambda_{\mathcal{E}} \mapsto \lambda_{\mathcal{E}}] = \mathrm{id}_{(\mathcal{E},\lambda_{\mathcal{E}})}$. For the opposite direction, let $\sigma : (\mathcal{E},\delta) \to (\mathcal{E}',\delta')$, $\lambda \in \mathcal{E}'$ and let $n = \delta'(\lambda)$. Then

$$\begin{aligned} \psi(\phi((f_\tau)_\tau)_\sigma)(\lambda) &= \sigma[\lambda_{\mathcal{E}} \mapsto \lambda] \cdot (\phi((f_\tau)_\tau)_n \\ &= \sigma[\lambda_{\mathcal{E}} \mapsto \lambda] \cdot (f_{\iota^n}(\lambda_{\mathcal{E}})) \\ &= (f_{(\sigma[\lambda_{\mathcal{E}} \mapsto \lambda])\iota^n}(\sigma[\lambda_{\mathcal{E}} \mapsto \lambda](\lambda_{\mathcal{E}}))) \\ &= f_\sigma(\lambda) \end{aligned}$$

Using the naturality assumption on f and the commutative diagram (4). The final statements of the lemma follow by easy calculations. □

Corollary 3.1. Let $\Gamma, \kappa : \mathsf{clock} \vdash A\ \mathsf{type}$, $(\mathcal{E}, \delta) \in \mathbb{T}$ and $\gamma \in [\![\Gamma]\!]_{(\mathcal{E},\delta)}$. The interpretation of the term $\Lambda\kappa.\, \mathsf{next}^\kappa(t[\kappa])$ is an isomorphism of sets

$$[\![\Gamma \vdash \forall\kappa.A\ \mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma) \cong \left[\!\!\left[\Gamma \vdash \forall\kappa.\overset{\kappa}{\blacktriangleright} A\ \mathsf{type}\right]\!\!\right]_{(\mathcal{E},\delta)}(\gamma)$$

Proof. By Lemma 3.3, $\left[\!\!\left[\Gamma \vdash \forall\kappa.\overset{\kappa}{\blacktriangleright} A\ \mathsf{type}\right]\!\!\right]_{(\mathcal{E},\delta)}(\gamma)$ is isomorphic to

$$\{(f_n)_{n\in\mathbb{N}} \mid \forall n.f_n \in \left[\!\!\left[\Gamma, \kappa : \mathsf{clock} \vdash \overset{\kappa}{\blacktriangleright} A\ \mathsf{type}\right]\!\!\right]_{((\mathcal{E},\lambda_\mathcal{E}),\delta[\lambda_\mathcal{E}\mapsto n])}(\iota^n\cdot\gamma, \lambda_\mathcal{E}), \forall n.\mathrm{tick}^{\lambda_\mathcal{E}}\cdot f_{n+1} = f_n\}$$

which is equal to

$$\left\{ (f_n)_{n\in\mathbb{N}} \;\middle|\; \begin{array}{l} f_0 = \star, \\ \forall n.f_{n+1} \in [\![\Gamma, \kappa : \mathsf{clock} \vdash A\ \mathsf{type}]\!]_{((\mathcal{E},\lambda_\mathcal{E}),\delta[\lambda_\mathcal{E}\mapsto n])}(\iota^n\cdot\gamma, \lambda_\mathcal{E}), \\ \forall n.\mathrm{tick}^{\lambda_\mathcal{E}}\cdot f_{n+2} = f_{n+1} \end{array} \right\}$$

where the restriction map $\mathrm{tick}^{\lambda_\mathcal{E}}$ refers to that of A, rather than $\overset{\kappa}{\blacktriangleright} A$. The interpretation of $\Lambda\kappa.\, \mathsf{next}^\kappa(t[\kappa])$ maps a family $(f_n)_n$ to the family $(g_m)_m$ given by $g_0 = \star$ and $g_{n+1} = f_n$. This map clearly is an isomorphism. □

We can thus define the interpretation of force to be the isomorphism given by Lemma 3.1. To show that this is well defined, we must verify that $\tau \cdot ([\![\mathsf{force}(t)]\!]_{(\mathcal{E},\delta)}(\gamma)) = [\![\mathsf{force}(t)]\!]_{(\mathcal{E}',\delta')}(\tau\cdot\gamma)$ for all appropriate γ and τ. This follows from the similar property for the interpretation of $\Lambda\kappa.\, \mathsf{next}^\kappa(t[\kappa])$ and by induction on t.

Finally we must show that equations are modelled soundly.

Proposition 3.1. The following equations hold for the interpretation of prev

$$[\![\mathsf{prev}\,\kappa.\,(\mathsf{next}^\kappa\, t)]\!] = [\![\Lambda\kappa.t]\!] \qquad\qquad [\![\mathsf{next}^\kappa\,((\mathsf{prev}\,\kappa.t)[\kappa])]\!] = [\![t]\!]$$

whenever the terms on the left hand sides are well typed.

Proof. Recall that prev is modelled using (3). Both equalities follow from the fact that force is modelled as an inverse to $\Lambda\kappa.\, \mathsf{next}^\kappa(t[\kappa])$. For example, the first is proved as

$$\begin{aligned} [\![\mathsf{prev}\,\kappa.\,(\mathsf{next}^\kappa\, t)]\!] &= [\![\mathsf{force}(\Lambda\kappa.\,(\mathsf{next}^\kappa\, t))]\!] \\ &= [\![\mathsf{force}(\Lambda\kappa.\,(\mathsf{next}^\kappa(\Lambda\kappa.t)[\kappa]))]\!] \\ &= [\![\Lambda\kappa.t]\!] \end{aligned}$$

The other equality is proved similarly. □

Modelling fixed points. The interpretation of fixed points is defined by induction over $\delta(\gamma(\kappa))$ as

$$[\![\mathsf{fix}^\kappa\, x.t]\!]_{(\mathcal{E},\delta)}(\gamma) = \begin{cases} [\![t]\!]_{(\mathcal{E},\delta)}(\gamma, \star) & \text{if } \delta(\gamma(\kappa)) = 0 \\ [\![t]\!]_{(\mathcal{E},\delta)}\left(\gamma, [\![\mathsf{fix}^\kappa\, x.t]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}(\mathrm{tick}^{\gamma(\kappa)}\cdot\gamma)\right) & \text{otherwise} \end{cases}$$

This is clearly well-defined and moreover correctly commutes with substitution, as is easy to see by computation. This definition is forced by the fixed point property and naturality of the actions.

Lemma 3.4. The fixed point axiom is sound in the sense that

$$[\![\mathsf{fix}^\kappa\, x.t]\!] = [\![t\,[\mathsf{next}^\kappa(\mathsf{fix}^\kappa\, x.t)/x]]\!]$$

Proof. By the substitution lemma

$$[\![t\,[\mathsf{next}^\kappa(\mathsf{fix}^\kappa\, x.t)/x]]\!]_{(\mathcal{E},\delta)}\,(\gamma) = [\![t]\!]_{(\mathcal{E},\delta)}\,(\gamma, [\![\mathsf{next}^\kappa(\mathsf{fix}^\kappa\, x.t)]\!]_{(\mathcal{E},\delta)}\,(\gamma))$$

Since $[\![\mathsf{next}^\kappa(\mathsf{fix}^\kappa\, x.t)]\!]_{(\mathcal{E},\delta)}\,(\gamma))$ is $\star$ if $\delta(\gamma(\kappa)) = 0$ and $[\![\mathsf{fix}^\kappa\, x.t]\!]_{\left(\mathcal{E},\delta^{-\gamma(\kappa)}\right)}\,(\mathrm{tick}^{\gamma(\kappa)} \cdot \gamma)$ otherwise, the lemma follows. □

With these definitions we can extend the interpretation to the whole of Core-GDTT. However the interpretation only validates the basic axioms, i.e., β and η laws. It does not validate the clock irrelevance axiom. To soundly interpret Core-GDTT we need to require that the objects and terms are suitably constant. This is the subject of the next section. First, however, we state the substitution lemma for Core-GDTT.

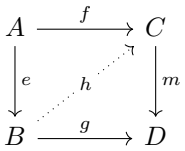
Lemma 3.5. The statement of Lemma 3.2 extends to all of Core-GDTT.

The above lemma can be proved by a trivial computation that we omit.

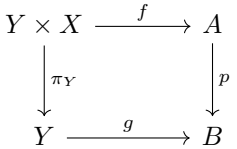
4. Modelling clock irrelevance using orthogonality

In the interpretation above $\forall\kappa.A$ is interpreted as an ordinary dependent product $\prod(\kappa : \mathsf{clock})\,.A$. Under this interpretation, the clock irrelevance axiom concerns functions f of type $\mathcal{C} \to A$ and states that each such function must be constant. To model this, we restrict attention in the model to those objects satisfying this property, and show that these objects are closed under the type constructions of Core-GDTT. The property will be called *orthogonality* and for the case of dependent types we consider orthogonality of an object and a morphism.

Recall that two morphisms $e : A \to B$ and $m : C \to D$ are *orthogonal* if all commutative squares as below have a *unique* filler h.



Definition 4.1. In a cartesian category $\mathbb{C}$, say a morphism $p : A \to B$ is *orthogonal to an object* X if it is orthogonal to any projection of the form $\pi_Y : Y \times X \to Y$, i.e., if for any Y and any commutative square



there exists a *unique* $h : Y \to A$ such that the diagram

$$\begin{array}{ccc} Y \times X & \xrightarrow{f} & A \\ \downarrow{\scriptstyle \pi_Y} & \overset{h}{\nearrow} & \downarrow{\scriptstyle p} \\ Y & \xrightarrow{g} & B \end{array}$$

commutes.

A map $p : A \to B$ in $\mathrm{Set}^{\mathbb{T}}$ is *invariant under clock introduction* if it is orthogonal to any object of the form $y\,(\{\kappa\}, n)$.

Definition 4.2. The interpretation of a type $\Gamma \vdash A\ \mathsf{type}$ is *invariant under clock introduction* if the context projection $[\![\Gamma, x : A \vdash]\!] \to [\![\Gamma \vdash]\!]$ is invariant under clock introduction in the sense of Definition 4.1.

The terminology of being invariant under clock introduction is justified by the following lemma, the proof of which is on page 18 after preliminary Lemmas 4.2 and 4.3.

Lemma 4.1. A morphism $p : A \to B$ in $\mathrm{Set}^{\mathbb{T}}$ is invariant under clock introduction if and only if for all $(\mathcal{E}, \delta) \in \mathbb{T}$, and any (equivalently all) $\lambda \notin \mathcal{E}$ and any n the square

$$\begin{array}{ccc} A(\mathcal{E}, \delta) & \xrightarrow{A(\iota)} & A\left((\mathcal{E}, \lambda), \delta[\lambda \mapsto n]\right) \\ \downarrow{\scriptstyle p_{(\mathcal{E},\delta)}} & \lrcorner & \downarrow{\scriptstyle p_{((\mathcal{E},\lambda),\delta[\lambda\mapsto n])}} \\ B(\mathcal{E}, \delta) & \xrightarrow{B(\iota)} & B\left((\mathcal{E}, \lambda), \delta[\lambda \mapsto n]\right) \end{array}$$

is a pullback, where $\iota : \mathcal{E} \to \mathcal{E}, \lambda$ is the inclusion.

In particular, for any presheaf A, the unique map $A \to 1$ is invariant under clock introduction iff A is a constant presheaf. It will be an invariant of the interpretation defined here that the interpretation of any type is invariant under clock introduction.

Lemma 4.1 can be restated in the following way for interpretations of types.

Corollary 4.1. The type $\Gamma \vdash A\ \mathsf{type}$ is invariant under clock introduction if and only if for any $\mathcal{E}$, any $\lambda \notin \mathcal{E}$, any inclusion $\iota : \mathcal{E} \to \mathcal{E}, \lambda$, any n and any $\gamma \in [\![\Gamma]\!]_{(\mathcal{E},\delta)}$, the action of ι on $[\![A]\!]$

$$\iota \cdot (-) : [\![\Gamma \vdash A\ \mathsf{type}]\!]_{(\mathcal{E},\delta)}\,(\gamma) \to [\![\Gamma \vdash A\ \mathsf{type}]\!]_{((\mathcal{E},\lambda),\delta[\lambda\mapsto n])}\,(\iota \cdot \gamma) \tag{5}$$

is an isomorphism.

The proof of Lemma 4.1 uses the characterisation of orthogonality in Lemma 4.2 together with the characterisation of exponentiation with certain representable functors in Lemma 4.3.

The following lemma is proved by a straightforward diagram chase.

Lemma 4.2. Suppose $\mathbb{C}$ is cartesian closed and $X \in \mathbb{C}$ is an object and $p : A \to B$ a morphism. Then p is orthogonal to X if and only if

$$\begin{array}{ccc} A & \xrightarrow{c_A} & A^X \\ \downarrow{\scriptstyle p} & \lrcorner & \downarrow{\scriptstyle p^X} \\ B & \xrightarrow{c_B} & B^X \end{array}$$

is a pullback. Here c_A and c_B are exponential transposes of projections $A \times X \to A$ and $B \times X \to B$ and p^X is postcomposition with p.

By the pullback lemma (MacLane, 1998, Exercise III.4.8), we derive the following corollary.

Corollary 4.2. If the morphisms $p \circ q$ and p are orthogonal to X then so is q.

Lemma 4.3. Let A be an object of $\mathrm{Set}^{\mathbb{T}}$. Let λ be a clock and $n \in \mathbb{N}$. Then

$$A^{y(\{\lambda\},n)}(\mathcal{E},\delta) \cong A((\mathcal{E},\lambda_{\mathcal{E}}),\delta[\lambda_{\mathcal{E}} \mapsto n])$$

and up to this isomorphism, $c_A = A(\iota)$, where $\iota : (\mathcal{E},\delta) \to ((\mathcal{E},\lambda_{\mathcal{E}}),\delta[\lambda_{\mathcal{E}} \mapsto n])$ is the inclusion.

Proof. In $\mathbb{T}$, the object $((\mathcal{E},\lambda_{\mathcal{E}}),\delta[\lambda_{\mathcal{E}} \mapsto n])$ is a coproduct of $(\mathcal{E},\delta)$ and (λ,n) with coproduct inclusions given by set inclusions (mapping λ to $\lambda_{\mathcal{E}}$). Since $y : \mathbb{T}^{\mathrm{op}} \to \mathrm{Set}^{\mathbb{T}}$ preserves products, we get the following series of isomorphisms using the Yoneda lemma and standard definitions of exponentials in presheaf categories:

$$\begin{aligned} A^{y(\{\lambda\},n)}(\mathcal{E},\delta) &= \mathsf{Hom}(y(\mathcal{E},\delta) \times y\left(\{\lambda\},n\right), A) \\ &\cong \mathsf{Hom}(y((\mathcal{E},\lambda_{\mathcal{E}}),\delta[\lambda_{\mathcal{E}} \mapsto n]), A) \\ &\cong A((\mathcal{E},\lambda_{\mathcal{E}}),\delta[\lambda_{\mathcal{E}} \mapsto n]) \end{aligned}$$

The morphism c_A maps $x \in A(\mathcal{E},\delta)$ to the natural transformation given by the composition of the first projection $y(\mathcal{E},\delta) \times y\left(\{\lambda\},n\right) \to y(\mathcal{E},\delta)$ and the morphism $y(\mathcal{E},\delta) \to A$ corresponding to x under the Yoneda lemma. Since the projection corresponds to composition with ι, the second statement of the lemma follows. □

Proof of Lemma 4.1 Follows from Lemma 4.2 and Lemma 4.3. □

Below we prove that dependent types are modelled as maps invariant under clock introduction, and use this to prove soundness of the clock irrelevance axiom. In fact, just to prove that, we only need types to be modelled as maps orthogonal to $\mathcal{C}$. This is a slightly weaker statement than being invariant under clock introduction, as the next lemma states. We have chosen to work with the latter because of the natural characterisation of Lemma 4.1.

Lemma 4.4. Let $\mathbb{C}$ be a cartesian closed category $\mathbb{C}$ and let $X = \varinjlim_i X_i$ be a connected colimit. If $p : A \to B$ is orthogonal to all X_i, then it is also orthogonal to X. As a consequence, if p is invariant under clock introduction, it is also orthogonal to $\mathcal{C}$.

The second statement of the lemma follows from the first by Lemma 3.1.

The notion of orthogonality can be shown to be equivalent to the one used in Hyland et. al. (Hyland et al., 1990), and the next lemma follows from (Hyland et al., 1990, Proposition 2.1). Rather than proving this equivalence, we give here a direct proof.

Proposition 4.1. Suppose $\mathbb{C}$ is a locally cartesian closed category and X is an object in $\mathbb{C}$. The notion of being orthogonal to X is then closed under composition, pullback (along *arbitrary* maps), dependent products (along *arbitrary* maps) and all isomorphisms are orthogonal to X.

Proof. Closure under composition and the fact that isomorphisms are orthogonal to X follow straightforwardly from Lemma 4.2.

To prove the statement for pullbacks, suppose $p : B \to D$ is orthogonal to X and $q : A \to C$ is

the pullback of p along some map g not assumed to be orthogonal to X. By the pullback pasting lemma then the outer square below is a pullback.

$$\begin{array}{ccccc} A & \xrightarrow{f} & B & \xrightarrow{c_B} & B^X \\ {\scriptstyle q}\downarrow & \lrcorner & \downarrow{\scriptstyle p} & \lrcorner & \downarrow{\scriptstyle p^X} \\ C & \xrightarrow{g} & D & \xrightarrow{c_D} & D^X \end{array}$$

By naturality of c, the below outer square is equal to the one above, and thus also a pullback.

$$\begin{array}{ccccc} A & \xrightarrow{c_A} & A^X & \xrightarrow{f^X} & B^X \\ {\scriptstyle q}\downarrow & & \downarrow{\scriptstyle q^X} & & \downarrow{\scriptstyle p^X} \\ C & \xrightarrow{c_C} & C^X & \xrightarrow{g^X} & D^X \end{array} \tag{6}$$

Since $-^X$ has a left adjoint it preserves pullbacks and so right square of (6) is a pullback. By the pullback lemma, also the left square is a pullback, and thus q is orthogonal to X.

For dependent products, suppose $p : A \to B$ is orthogonal to X, and $f : B \to C$. We must show that $\Pi_f(p)$ is orthogonal to X, where $\Pi_f : \mathbb{C}/B \to \mathbb{C}/C$ is the right adjoint to pullback along f. We write $f^*(h) : B \times_C Y \to B$ for the result of applying the pullback functor to an object $h : Y \to C$ of $\mathbb{C}/C$ and use the notation

$$\widehat{(-)} : \mathsf{Hom}_{\mathbb{C}/C}(h, \Pi_f(p)) \to \mathsf{Hom}_{\mathbb{C}/B}(f^*(h), p)$$

for the isomorphism of hom-sets, given $h : Y \to C$.

Given Y, h, k as in the outer square on the left below, by naturality, the isomorphism $\widehat{(-)}$ extends to a bijective correspondence of diagonal fillers in the following two diagrams.

$$\begin{array}{ccc} Y \times X & \xrightarrow{k} & \Pi_B A \\ {\scriptstyle \pi_Y}\downarrow & \nearrow & \downarrow{\scriptstyle \Pi_f(p)} \\ Y & \xrightarrow{h} & C \end{array} \qquad \begin{array}{ccc} B \times_C (Y \times X) & \xrightarrow{\widehat{k}} & A \\ {\scriptstyle B \times_C \pi_Y}\downarrow & \nearrow & \downarrow{\scriptstyle p} \\ B \times_C Y & \xrightarrow{f^*(h)} & B \end{array} \tag{7}$$

where $B \times_C \pi_Y$ is the pullback functor applied to the morphism $\pi_Y : (h \circ \pi_Y) \to h$ in $\mathbb{C}/C$.

By the pullback pasting lemma, the following outer diagram is a pullback

$$\begin{array}{ccc} (B \times_C Y) \times X & \xrightarrow{\pi_Y^C \times \mathrm{id}_X} & Y \times X \\ {\scriptstyle \pi_{(B \times_C Y)}}\downarrow & \lrcorner & \downarrow{\scriptstyle \pi_Y} \\ B \times_C Y & \xrightarrow{\pi_Y^C} & Y \\ {\scriptstyle f^*(h)}\downarrow & \lrcorner & \downarrow{\scriptstyle h} \\ B & \xrightarrow{f} & C \end{array}$$

From this we conclude that there is an isomorphism $\phi : (B \times_C Y) \times X \cong B \times_C (Y \times X)$. An

easy diagram chase verifies (using the universal property of the lower diagram above) that

$$(B \times_C \pi_Y) \circ \phi = \pi_{B \times_C Y} : (B \times_C Y) \times X \to (B \times_C Y).$$

Thus, the fillers of (7) are in bijective correspondence with the fillers of

$$\begin{array}{ccc} (B \times_C Y) \times X & \xrightarrow{\widehat{k} \circ \phi} & A \\ {\scriptstyle \pi_{B \times_C Y}} \downarrow & \nearrow & \downarrow {\scriptstyle p} \\ B \times_C Y & \xrightarrow{f^*(h)} & B \end{array}$$

Since p is assumed to be orthogonal to X, there is a unique filler of the diagram above, and thus a unique filler of the left diagram of (7). This proves that $\Pi_f(p)$ is orthogonal to X as desired. □

Lemma 4.5. If the type $\Gamma \vdash A$ type is invariant under clock introduction and $\Gamma \vdash \kappa : \mathsf{clock}$ then $\Gamma \vdash \overset{\kappa}{\blacktriangleright} A$ type is invariant under clock introduction.

Proof. We use the characterisation of invariance under clock introduction from Corollary 4.1. Thus let $(\mathcal{E}, \delta) \in \mathbb{T}$, and $n \in \mathbb{N}$. Let $\iota : (\mathcal{E}, \delta) \to ((\mathcal{E}, \lambda_{\mathcal{E}}), \delta[\lambda_{\mathcal{E}} \mapsto n])$ be the inclusion. It is immediate that $(\iota \cdot \gamma)(\kappa) = \gamma(\kappa)$ and thus both are in $\mathcal{E}$.

Hence the action $\iota \cdot (-)$ of $\overset{\kappa}{\blacktriangleright} A$ is either the identity function, when $\delta(\gamma(\kappa)) = 0$, or is the action of the inclusion

$$\iota' : \left(\mathcal{E}, \delta^{-\gamma(\kappa)}\right) \to ((\mathcal{E}, \lambda_{\mathcal{E}}), \delta^{-\gamma(\kappa)}[\lambda_{\mathcal{E}} \mapsto n])$$

if $\delta(\gamma(\kappa)) > 0$. The former is trivially an isomorphism, and the latter is an isomorphism by assumption that A is invariant under clock introduction, and Corollary 4.1. □

Corollary 4.3. If $\Gamma \vdash A$ type is a valid judgement of Core-GDTT, the projection $[\![\Gamma, x : A]\!] \to [\![\Gamma]\!]$ is invariant under clock introduction.

Proof. By induction on type derivations. □

We now show that Corollary 4.3 implies the soundness of the clock irrelevance axiom.

Lemma 4.6. Suppose $\Gamma \vdash t : \forall \kappa.A$, $\Gamma \vdash A$ type, i.e., $\kappa \notin A$, $\Gamma \vdash \kappa' : \mathsf{clock}$ and $\Gamma \vdash \kappa'' : \mathsf{clock}$. If the context projection $[\![\Gamma, x : A \vdash]\!] \to [\![\Gamma \vdash]\!]$ is invariant under clock introduction, then $[\![t[\kappa']]\!] = [\![t[\kappa'']]\!]$.

Proof. The denotation of the term t corresponds uniquely to a morphism f which makes the following diagram commute (where p is the context projection).

$$\begin{array}{ccc} [\![\Gamma \vdash]\!] \times \mathcal{C} & \xrightarrow{f} & [\![\Gamma, x : A \vdash]\!] \\ {\scriptstyle \pi_{[\![\Gamma \vdash]\!]}} \downarrow & & \downarrow {\scriptstyle p} \\ [\![\Gamma \vdash]\!] & \xrightarrow{\mathrm{id}} & [\![\Gamma \vdash]\!] \end{array}$$

Analogously the clocks κ' and κ'' correspond to sections $[\![\kappa']\!]$ and $[\![\kappa'']\!]$ of $\pi_{[\![\Gamma \vdash]\!]}$, and by this correspondence the terms $t[\kappa']$ and $t[\kappa'']$ correspond to compositions of $[\![\kappa']\!]$ and $[\![\kappa'']\!]$ with f.

By Corollary 4.3 the context projection p is invariant under clock introduction and, thus by

Lemma 4.4 the context projection p is orthogonal to $\mathcal{C}$. This means there is a unique filler u making the following diagram commute.

$$\begin{array}{ccc} [\![\Gamma \vdash]\!] \times \mathcal{C} & \xrightarrow{f} & [\![\Gamma, x : A \vdash]\!] \\ \downarrow{\scriptstyle \pi_{[\![\Gamma\vdash]\!]}} & \overset{u}{\nearrow} & \downarrow{\scriptstyle p} \\ [\![\Gamma \vdash]\!] & \xrightarrow{\mathrm{id}} & [\![\Gamma \vdash]\!] \end{array}$$

Thus $f \circ [\![\kappa']\!] = u \circ \pi_{[\![\Gamma\vdash]\!]} \circ [\![\kappa']\!] = u$ and $f \circ [\![\kappa'']\!] = u \circ \pi_{[\![\Gamma\vdash]\!]} \circ [\![\kappa'']\!] = u$. Hence the denotations of $t[\kappa']$ and $t[\kappa'']$ are equal. □

Theorem 4.1 (Soundness). The model validates all the judgemental equalities of Core-GDTT. More precisely

— If $\Gamma \vdash A \equiv B$ is derivable then for any $(\mathcal{E}, \delta) \in \mathbb{T}$ and $\gamma \in [\![\Gamma \vdash]\!]_{(\mathcal{E},\delta)}$

$$[\![\Gamma \vdash A \ \mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma) = [\![\Gamma \vdash B \ \mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma).$$

— If $\Gamma \vdash t \equiv s : A$ is derivable then for any $(\mathcal{E}, \delta) \in \mathbb{T}$ and $\gamma \in [\![\Gamma \vdash]\!]_{(\mathcal{E},\delta)}$

$$[\![\Gamma \vdash t : A]\!]_{(\mathcal{E},\delta)}(\gamma) = [\![\Gamma \vdash s : B]\!]_{(\mathcal{E},\delta)}(\gamma).$$

5. Identity types

Since $\mathrm{Set}^{\mathbb{T}}$ is a presheaf category it models *extensional* identity types, i.e., identity types with the identity reflection axiom.

Concretely the interpretation of the identity type is as follows

$$[\![\Gamma \vdash \mathsf{Id}_A(t,s) \ \mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma) = \left\{ \star \;\middle|\; [\![\Gamma \vdash t : A]\!]_{(\mathcal{E},\delta)}(\gamma) = [\![\Gamma \vdash s : A]\!]_{(\mathcal{E},\delta)}(\gamma) \right\}.$$

The interpretation of reflexivity term $\Gamma \vdash \mathsf{refl}_A\, t : \mathsf{Id}_A(t,t)$ is simple:

$$[\![\Gamma \vdash \mathsf{refl}_A\, t : \mathsf{Id}_A(t,t)]\!]_{(\mathcal{E},\delta)}(\gamma) = \star.$$

Similarly, for any term $\Gamma \vdash p : \mathsf{Id}_A(t,s)$, any $(\mathcal{E}, \delta) \in \mathbb{T}$, and any $\gamma \in [\![\Gamma \vdash]\!]_{(\mathcal{E},\delta)}$ there is an element $[\![\Gamma \vdash p : \mathsf{Id}_A(t,s)]\!]_{(\mathcal{E},\delta)}(\gamma) \in [\![\Gamma \vdash \mathsf{Id}_A(t,s) \ \mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma)$ which in particular means that for all such $(\mathcal{E}, \delta)$ and γ we have

$$[\![\Gamma \vdash t : A]\!]_{(\mathcal{E},\delta)}(\gamma) = [\![\Gamma \vdash s : A]\!]_{(\mathcal{E},\delta)}(\gamma)$$

or in other words

$$[\![\Gamma \vdash t : A]\!] = [\![\Gamma \vdash s : A]\!]$$

which shows soundness of the equality reflection rule.

For soundness of the clock irrelevance axiom in this extended system, we must show that also the interpretation of identity type $\mathsf{Id}_A(t,s)$ is invariant under clock introduction, provided the type A is.

Proposition 5.1. Let $\Gamma \vdash t : A$ and $\Gamma \vdash s : A$ be two terms. Suppose the context projection $\pi_A : [\![\Gamma, x : A \vdash]\!] \to [\![\Gamma \vdash]\!]$ is *invariant under clock introduction*. Then the context projection

$$\pi_{\mathsf{Id}_A(t,s)} : [\![\Gamma, p : \mathsf{Id}_A(t,s) \vdash]\!] \to [\![\Gamma \vdash]\!]$$

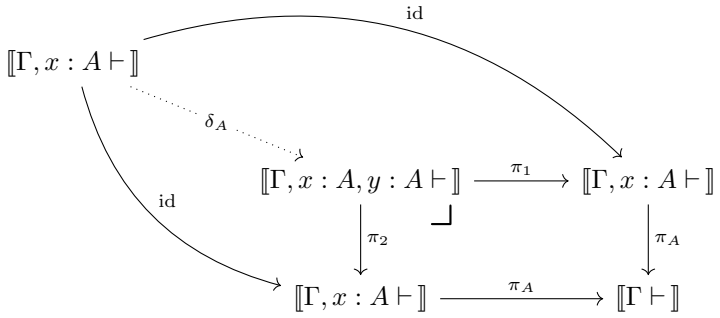
is also invariant under clock introduction.

Proof. By definition $\pi_{\mathsf{Id}_A(t,s)}$ fits into the following pullback

$$\begin{array}{ccc} [\![\Gamma, p : \mathsf{Id}_A\,(t,s) \vdash]\!] & \longrightarrow & [\![\Gamma, x : A, y : A, p : \mathsf{Id}_A\,(x,y) \vdash]\!] \\ \downarrow{\scriptstyle \pi_{\mathsf{Id}_A(t,s)}} & \lrcorner & \downarrow{\scriptstyle \pi_{\mathsf{Id}_A(x,y)}} \\ [\![\Gamma \vdash]\!] & \xrightarrow{\langle [\![t]\!], [\![s]\!] \rangle_{[\![\Gamma \vdash]\!]}} & [\![\Gamma, x : A, y : A \vdash]\!] \end{array}$$

so by Proposition 4.1 it suffices to show that $\pi_{\mathsf{Id}_A(x,y)}$ is invariant under clock introduction.

By definition $\pi_{\mathsf{Id}_A(x,y)}$ is (isomorphic to) the unique map δ_A (the diagonal) making the following diagram commute.



By assumption π_A is invariant under clock introduction. Thus so is π_2 by Proposition 4.1. By the same proposition id is invariant under clock introduction and thus by Corollary 4.2 so is δ_A. By Proposition 4.1 so is $\pi_{\mathsf{Id}_A(x,y)}$, since it is isomorphic to δ_A. □

6. Delayed substitutions

In the simply typed setting the applicative functor McBride and Paterson (2008) structure of the later modality is essential. For instance, it allows us to apply a term f of type $\overset{\kappa}{\blacktriangleright}(A \to B)$ to a term t of type $\overset{\kappa}{\blacktriangleright} A$ to get a term $f \circledast^\kappa t$ of type $\overset{\kappa}{\blacktriangleright} B$; that is, if we have a function after one κ-step and if after one κ-step we have an argument, we can apply the function at the time, and get the result after one κ-step.

In GDTT the function types can be dependent, and thus to be able to use the later modality to its fullest, the applicative functor structure needs to be generalised, so that we can apply a term f of type $\overset{\kappa}{\blacktriangleright}(\prod (x : A) .B)$ to a term t of type $\overset{\kappa}{\blacktriangleright} A$. In GDTT the type of the delayed application $f \circledast^\kappa t$ becomes $\overset{\kappa}{\blacktriangleright}[x \leftarrow t] .B$ where $[x \leftarrow t]$ is a *delayed substitution*, and x is bound in $\overset{\kappa}{\blacktriangleright}[x \leftarrow t] .B$. If at some point we learn that t is of the form $\mathsf{next}^\kappa\, t'$ for some t' we can actually perform the substitution and get the type $\overset{\kappa}{\blacktriangleright} B[t'/x]$. This process can be iterated, e.g., if B is also a dependent product $\prod (y : C) .D$ and s is a term of type $\overset{\kappa}{\blacktriangleright}[x \leftarrow t] .C$ then the delayed application $f \circledast^\kappa t \circledast^\kappa s$ is well-typed with type $\overset{\kappa}{\blacktriangleright}[x \leftarrow t, y \leftarrow s] .D$.

Delayed substitutions satisfy convenient judgemental equalities (listed in Figure 5) which ensure that delayed substitutions can be manipulated in an intuitive way. For example, if the type A is well-formed without x then the delayed substitution in $\overset{\kappa}{\blacktriangleright}[x \leftarrow t] A$ is redundant, and thus

$\overset{\kappa}{\blacktriangleright}[x \leftarrow t]\, A \equiv \overset{\kappa}{\blacktriangleright} A$. Further, as explained above, if the term t is of type $\mathsf{next}^\kappa\, t'$ then we can perform an actual substitution, and thus $\overset{\kappa}{\blacktriangleright}[x \leftarrow \mathsf{next}^\kappa\, t']\, .B \equiv \overset{\kappa}{\blacktriangleright} B[t'/x]$. Finally, the order of bindings in $\overset{\kappa}{\blacktriangleright}[x \leftarrow t, y \leftarrow s]\, .D$ matters only in as much as it usually does in dependent type theory. That is, $x \leftarrow t$ and $y \leftarrow s$ can be exchanged provided x does not appear in the type of y.

To conclude this introduction to delayed substitutions we remark that they can be attached to the term former $\mathsf{next}^\kappa\, t$ as well and they enjoy analogous rules. As shown in previous work (Bizjak et al., 2016) a calculus with just these generalised next^κ and $\overset{\kappa}{\blacktriangleright}$ can express the delayed application construct which was primitive in simply typed calculi with guarded recursion. We refer to (Bizjak et al., 2016) for extensive examples of how to use delayed substitutions for reasoning about guarded recursive and coinductive terms.

The typing rules for delayed substitutions and related constructs are recalled in Figure 4 and the equality rules are recalled in Figure 5. We write $\xi : \Gamma \to^\kappa \Gamma'$ for the delayed substitution ξ from Γ to Γ'. Note that Γ' is not a context, but a telescope such that Γ, Γ' is a well-formed context. The delayed substitution ξ is a list of pairs written as $x \leftarrow t$, which are successively well-typed in context Γ of types derived from Γ', as stated in the formation rule in Figure 4.

Delayed substitutions

$$\frac{\Gamma \vdash}{\cdot : \Gamma \to^\kappa \cdot} \qquad \frac{\xi : \Gamma \to^\kappa \Gamma' \qquad \Gamma \vdash t : \overset{\kappa}{\blacktriangleright} \xi.A}{\xi\,[x \leftarrow t] : \Gamma \to^\kappa \Gamma', x : A}$$

Well-formed types

$$\frac{\Gamma, \Gamma' \vdash A\ \mathsf{type} \qquad \xi : \Gamma \to^\kappa \Gamma'}{\Gamma \vdash \overset{\kappa}{\blacktriangleright} \xi.A\ \mathsf{type}}$$

Well-typed terms

$$\frac{\Gamma, \Gamma' \vdash t : A \qquad \xi : \Gamma \to^\kappa \Gamma'}{\Gamma \vdash \mathsf{next}^\kappa\, \xi.t : \overset{\kappa}{\blacktriangleright} \xi.A}$$

Figure 4. Typing rules involving delayed substitutions.

The typing rule for prev is generalised in (Bizjak et al., 2016) to allow elimination also of $\blacktriangleright$ with attached delayed substitutions. We now recall that rule and show that it is admissible.

Proposition 6.1. If $\Gamma, \kappa : \mathsf{clock} \vdash t : \overset{\kappa}{\blacktriangleright} \xi.A$, then also $\Gamma \vdash \mathsf{prev}\, \kappa.t : \forall \kappa.A(\mathsf{adv}^\kappa\, \xi)$ where

$$\mathsf{adv}^\kappa\, \xi : \Gamma, \kappa : \mathsf{clock} \to \Gamma, \kappa : \mathsf{clock}, \Gamma'$$

is a substitution defined by induction on ξ using the rules

$$\mathsf{adv}^\kappa\, (\cdot) = \mathrm{id}_{\Gamma,\kappa}$$
$$\mathsf{adv}^\kappa\, (\xi[x \mapsto t]) = (\mathsf{adv}^\kappa\, \xi)[x \mapsto (\mathsf{prev}\, \kappa.t)[\kappa]]$$

Moreover, the following equality rule holds

$$\frac{\Gamma, \kappa, \Gamma' \vdash t : A \qquad \xi : (\Gamma, \kappa : \mathsf{clock}) \to^\kappa \Gamma'}{\Gamma \vdash \mathsf{prev}\, \kappa.\, \mathsf{next}^\kappa\, \xi.t \equiv \Lambda\kappa.t(\mathsf{adv}^\kappa(\xi)) : \forall \kappa.A(\mathsf{adv}^\kappa(\xi))}$$

Type equality

$$\frac{\xi\,[x \leftarrow t] : \Gamma \twoheadrightarrow^{\kappa} \Gamma', x : B \qquad \Gamma, \Gamma' \vdash A\ \mathsf{type}}{\Gamma \vdash \overset{\kappa}{\blacktriangleright}\xi\,[x \leftarrow t]\,.A \equiv \overset{\kappa}{\blacktriangleright}\xi.A}$$

$$\frac{\xi\,[x \leftarrow t, y \leftarrow u]\,\xi' : \Gamma \twoheadrightarrow^{\kappa} \Gamma', x : B, y : C, \Gamma'' \qquad \Gamma, \Gamma' \vdash C\ \mathsf{type} \qquad \Gamma, \Gamma', x : B, y : C, \Gamma'' \vdash A\ \mathsf{type}}{\Gamma \vdash \overset{\kappa}{\blacktriangleright}\xi\,[x \leftarrow t, y \leftarrow u]\,\xi'.A \equiv \overset{\kappa}{\blacktriangleright}\xi\,[y \leftarrow u, x \leftarrow t]\,\xi'.A}$$

$$\frac{\xi : \Gamma \twoheadrightarrow^{\kappa} \Gamma' \qquad \Gamma, \Gamma', x : B \vdash A\ \mathsf{type} \qquad \Gamma, \Gamma' \vdash t : B}{\Gamma \vdash \overset{\kappa}{\blacktriangleright}\xi\,[x \leftarrow \mathsf{next}^{\kappa}\,\xi.t]\,.A \equiv \overset{\kappa}{\blacktriangleright}\xi.A\,[t/x]}$$

$$\frac{\xi : \Gamma \twoheadrightarrow^{\kappa} \Gamma' \qquad \Gamma, \Gamma' \vdash t : A \qquad \Gamma, \Gamma' \vdash s : A}{\Gamma \vdash \mathsf{Id}_{\overset{\kappa}{\blacktriangleright}\xi.A}\,(\mathsf{next}^{\kappa}\,\xi.t, \mathsf{next}^{\kappa}\,\xi.s) \equiv \overset{\kappa}{\blacktriangleright}\xi.\mathsf{Id}_A\,(t, s)}$$

Term equality

$$\frac{\xi\,[x \leftarrow t] : \Gamma \twoheadrightarrow^{\kappa} \Gamma', x : B \qquad \Gamma, \Gamma' \vdash u : A}{\Gamma \vdash \mathsf{next}^{\kappa}\,\xi\,[x \leftarrow t]\,.u \equiv \mathsf{next}^{\kappa}\,\xi.u : \overset{\kappa}{\blacktriangleright}\xi.A}$$

$$\frac{\xi\,[x \leftarrow t, y \leftarrow u]\,\xi' : \Gamma \twoheadrightarrow^{\kappa} \Gamma', x : B, y : C, \Gamma'' \qquad \Gamma, \Gamma' \vdash C\ \mathsf{type} \qquad \Gamma, \Gamma', x : B, y : C, \Gamma'' \vdash v : A}{\Gamma \vdash \mathsf{next}^{\kappa}\,\xi\,[x \leftarrow t, y \leftarrow u]\,\xi'.v \equiv \mathsf{next}^{\kappa}\,\xi\,[y \leftarrow u, x \leftarrow t]\,\xi'.v : \overset{\kappa}{\blacktriangleright}\xi\,[x \leftarrow t, y \leftarrow u]\,\xi'.A}$$

$$\frac{\xi : \Gamma \twoheadrightarrow^{\kappa} \Gamma' \qquad \Gamma, \Gamma', x : B \vdash u : A \qquad \Gamma, \Gamma' \vdash t : B}{\Gamma \vdash \mathsf{next}^{\kappa}\,\xi\,[x \leftarrow \mathsf{next}^{\kappa}\,\xi.t]\,.u \equiv \mathsf{next}^{\kappa}\,\xi.u\,[t/x] : \overset{\kappa}{\blacktriangleright}\xi.A\,[t/x]} \qquad \frac{\Gamma \vdash t : \overset{\kappa}{\blacktriangleright}\xi.A}{\Gamma \vdash \mathsf{next}^{\kappa}\,\xi\,[x \leftarrow t]\,.x \equiv t : \overset{\kappa}{\blacktriangleright}\xi.A}$$

$$\frac{\Gamma \vdash A\ \mathsf{type} \qquad \Gamma, \Gamma', \Gamma'' \vdash u : A \qquad \xi : \Gamma \twoheadrightarrow^{\kappa} \Gamma' \qquad \xi' : \Gamma \twoheadrightarrow^{\kappa} \Gamma''}{\Gamma \vdash \mathsf{next}^{\kappa}\,\xi.\,\mathsf{next}^{\kappa}\,\xi'.u \equiv \mathsf{next}^{\kappa}\,\xi'.\,\mathsf{next}^{\kappa}\,\xi.u : \overset{\kappa}{\blacktriangleright}\overset{\kappa}{\blacktriangleright}A}$$

Figure 5. Equality rules involving delayed substitutions.

Note that in the definition of $\mathsf{adv}^{\kappa}\,(\xi[x \mapsto t])$, the typing assumption on t is

$$\Gamma \vdash t : \overset{\kappa}{\blacktriangleright}\xi.A$$

and so the typing of adv relies on the first part of the proposition. Thus the statements of welltypedness of $\mathsf{adv}^{\kappa}(\xi)$ and of $\mathsf{prev}\,\kappa.t$ must be proved by simultaneous induction over the length of ξ.

Proof. We first define the concept of applying next^{κ} to a substitution obtaining a delayed substitution. This should be thought of as an inverse operation to advancing a delayed substitution. Let $\sigma : \Gamma \to \Gamma, \Gamma'$ be a substitution which restricted to the context Γ is the identity. Define

$$\mathsf{next}^{\kappa}(\sigma) : \Gamma \twoheadrightarrow^{\kappa} \Gamma'$$

by induction on the size of Γ' by

$$\mathsf{next}^{\kappa}\,(\sigma[x \mapsto u]) = \mathsf{next}^{\kappa}(\sigma)[x \leftarrow \mathsf{next}^{\kappa}(u)]$$

This is welltyped, since by assumption $\Gamma \vdash u : A\sigma$ and so $\Gamma \vdash \mathsf{next}^\kappa(u) : \overset{\kappa}{\blacktriangleright}(A\sigma)$ and

$$\overset{\kappa}{\blacktriangleright}(A\sigma) \equiv \overset{\kappa}{\blacktriangleright} \mathsf{next}^\kappa(\sigma).A.$$

Since $\mathsf{next}^\kappa(\mathsf{adv}^\kappa(\xi)) \equiv \xi$ by the η rule for prev the assumed type of t in the statement of the proposition is

$$\begin{aligned}\overset{\kappa}{\blacktriangleright} \xi.A &\equiv \overset{\kappa}{\blacktriangleright} \mathsf{next}^\kappa(\mathsf{adv}^\kappa(\xi)).A \\ &\equiv \overset{\kappa}{\blacktriangleright} A(\mathsf{adv}^\kappa(\xi))\end{aligned}$$

by repeated application of the first and third rule of Figure 5. Thus, $\Gamma \vdash \mathsf{prev}\,\kappa.t : \forall\kappa.A(\mathsf{adv}^\kappa\,\xi)$ as desired. The equality rule stated at the end of the proposition follows analogously as

$$\begin{aligned}\mathsf{prev}\,\kappa.\,\mathsf{next}^\kappa\,\xi.t &\equiv \mathsf{prev}\,\kappa.\,\mathsf{next}^\kappa\,[\mathsf{next}^\kappa(\mathsf{adv}^\kappa(\xi))]\,.t \\ &\equiv \mathsf{prev}\,\kappa.\,\mathsf{next}^\kappa\,(t(\mathsf{adv}^\kappa(\xi))) \\ &\equiv \Lambda\kappa.t(\mathsf{adv}^\kappa(\xi))\end{aligned}$$

where the last equality is the β rule for prev from Figure 1. □

6.1. *Semantics of delayed substitutions.*

A delayed substitution $\xi : \Gamma \rightarrowtriangle^\kappa \Gamma'$ is interpreted as a family of elements

$$[\![\xi : \Gamma \rightarrowtriangle^\kappa \Gamma']\!]_{(\mathcal{E},\delta)}\,(\gamma) \in [\![\Gamma, \Gamma' \vdash]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}$$

parametrised by $(\mathcal{E},\delta) \in \mathbb{T}$ and $\gamma \in [\![\Gamma \vdash]\!]_{(\mathcal{E},\delta)}$ such that $\delta(\gamma(\kappa)) > 0$. Moreover these elements must satisfy

$$\pi^\Gamma_{(\mathcal{E},\delta^{-\gamma(\kappa)})}([\![\xi : \Gamma \rightarrowtriangle^\kappa \Gamma']\!]_{(\mathcal{E},\delta)}\,(\gamma)) = \mathrm{tick}^{\gamma(\kappa)} \cdot \gamma \tag{8}$$

where $\pi^\Gamma_{(\mathcal{E},\delta)} : [\![\Gamma, \Gamma' \vdash]\!]_{(\mathcal{E},\delta)} \to [\![\Gamma \vdash]\!]_{(\mathcal{E},\delta)}$ is the composition of context projections. The family of elements needs to be natural in the usual sense with respect to the actions of $[\![\Gamma \vdash]\!]$ and $\Gamma, \Gamma' \vdash$, that is, if $\sigma : (\mathcal{E},\delta) \to (\mathcal{E}',\delta')$ and $\gamma \in [\![\Gamma \vdash]\!]_{(\mathcal{E},\delta)}$ are such that $\delta(\gamma(\kappa)) > 0$ and $\delta'(\sigma(\gamma(\kappa))) > 0$ then

$$\sigma \cdot \left([\![\xi : \Gamma \rightarrowtriangle^\kappa \Gamma']\!]_{(\mathcal{E},\delta)}\,(\gamma)\right) = [\![\xi : \Gamma \rightarrowtriangle^\kappa \Gamma']\!]_{(\mathcal{E}',\delta')}\,(\sigma \cdot \gamma). \tag{9}$$

Here, on the left hand side, σ is considered a map $(\mathcal{E}, \delta^{-\gamma(\kappa)}) \to (\mathcal{E}', \delta'^{-\gamma(\kappa)})$

The interpretation is defined by induction on the length of ξ (equivalently, length of Γ'). When ξ is empty the interpretation is uniquely determined by property (8). Otherwise

$$[\![\xi\,[x \leftarrow t] : \Gamma \rightarrowtriangle^\kappa \Gamma', x : A]\!]_{(\mathcal{E},\delta)}\,(\gamma) = \left([\![\xi : \Gamma \rightarrowtriangle^\kappa \Gamma']\!]_{(\mathcal{E},\delta)}\,(\gamma), \left[\!\!\left[\Gamma \vdash t : \overset{\kappa}{\blacktriangleright} \xi.A\right]\!\!\right]_{(\mathcal{E},\delta)}\,(\gamma)\right)$$

With this the interpretation of the extended later type and the next construct are straightforward.

$$\left[\!\!\left[\Gamma \vdash \overset{\kappa}{\blacktriangleright} \xi.A\ \mathsf{type}\right]\!\!\right]_{(\mathcal{E},\delta)}\,(\gamma) = \begin{cases}\{\star\} & \text{if } \delta(\gamma(\kappa)) = 0 \\ [\![A]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}\left([\![\xi]\!]_{(\mathcal{E},\delta)}\,(\gamma)\right) & \text{otherwise}\end{cases}$$

$$[\![\mathsf{next}^\kappa\,\xi.t]\!]_{(\mathcal{E},\delta)}\,(\gamma) = \begin{cases}\star & \text{if } \delta(\gamma(\kappa)) = 0 \\ [\![t]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}\left([\![\xi]\!]_{(\mathcal{E},\delta)}\,(\gamma)\right) & \text{otherwise}\end{cases}$$

Note that (8) implies that in case of ξ being an empty delayed substitution, these definitions specialise to the ones given in Section 3.2.

Proposition 6.2. The interpretation is sound wrt. the equalities of Figure 5.

Proof. Most of these rules follow from the substitution lemma, and we just show a few.
To show that

$$\left[\!\left[\Gamma \vdash \overset{\kappa}{\blacktriangleright} \xi\,[x \leftarrow \mathsf{next}^\kappa\,\xi.t]\,.A\ \mathsf{type}\right]\!\right] = \left[\!\left[\Gamma \vdash \overset{\kappa}{\blacktriangleright} \xi.A\,[t/x]\ \mathsf{type}\right]\!\right]$$

under the assumptions of the rule, suppose given $(\mathcal{E}, \delta)$ and γ, and consider the interesting case of $\delta(\gamma(\kappa)) > 0$. Then

$$\begin{aligned}
&\left[\!\left[\Gamma \vdash \overset{\kappa}{\blacktriangleright} \xi\,[x \leftarrow \mathsf{next}^\kappa\,\xi.t]\,.A\ \mathsf{type}\right]\!\right]_{(\mathcal{E},\delta)}(\gamma) \\
&= [\![\Gamma, \Gamma', x : B \vdash A\ \mathsf{type}]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}\left([\![\xi]\!]_{(\mathcal{E},\delta)}(\gamma), [\![\mathsf{next}^\kappa\,\xi.t]\!]_{(\mathcal{E},\delta)}(\gamma)\right) \\
&= [\![\Gamma, \Gamma', x : B \vdash A\ \mathsf{type}]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}\left([\![\xi]\!]_{(\mathcal{E},\delta)}(\gamma), [\![t]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}([\![\xi]\!]_{(\mathcal{E},\delta)}(\gamma))\right) \\
&= [\![\Gamma, \Gamma' \vdash A\,[t/x]\ \mathsf{type}]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}\left([\![\xi]\!]_{(\mathcal{E},\delta)}(\gamma)\right) \\
&= \left[\!\left[\Gamma \vdash \overset{\kappa}{\blacktriangleright} \xi.A\,[t/x]\ \mathsf{type}\right]\!\right]_{(\mathcal{E},\delta)}(\gamma)
\end{aligned}$$

For soundness of the rule

$$\Gamma \vdash \mathsf{Id}_{\overset{\kappa}{\blacktriangleright}\xi.A}(\mathsf{next}^\kappa\,\xi.t, \mathsf{next}^\kappa\,\xi.s) \equiv \overset{\kappa}{\blacktriangleright}\xi.\mathsf{Id}_A(t,s).$$

The left hand side of the equality is interpreted as

$$\left[\!\left[\mathsf{Id}_{\overset{\kappa}{\blacktriangleright}\xi.A}(\mathsf{next}^\kappa\,\xi.t, \mathsf{next}^\kappa\,\xi.s)\right]\!\right]_{(\mathcal{E},\delta)}(\gamma) = \left\{\star \;\middle|\; [\![\mathsf{next}^\kappa\,\xi.t]\!]_{(\mathcal{E},\delta)}(\gamma) = [\![\mathsf{next}^\kappa\,\xi.s]\!]_{(\mathcal{E},\delta)}(\gamma)\right\}$$

And by simple inspection we have

$$[\![\mathsf{next}^\kappa\,\xi.t]\!]_{(\mathcal{E},\delta)}(\gamma) = [\![\mathsf{next}^\kappa\,\xi.s]\!]_{(\mathcal{E},\delta)}(\gamma)$$

if and only if $\delta(\gamma(\kappa)) = 0$ or

$$[\![t]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}\left(\mathrm{tick}^{\gamma(\kappa)} \cdot \gamma, [\![\xi]\!]_{(\mathcal{E},\delta)}(\gamma)\right) = [\![s]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}\left(\mathrm{tick}^{\gamma(\kappa)} \cdot \gamma, [\![\xi]\!]_{(\mathcal{E},\delta)}(\gamma)\right).$$

This immediately shows soundness by inspecting the definition of the interpretation of the later type $\overset{\kappa}{\blacktriangleright}\xi.\mathsf{Id}_A(t,s)$.

Finally, we show that

$$\left[\!\left[\Gamma \vdash \mathsf{next}^\kappa\,\xi.\,\mathsf{next}^\kappa\,\xi'.u : \overset{\kappa}{\blacktriangleright}\overset{\kappa}{\blacktriangleright} A\right]\!\right]_{(\mathcal{E},\delta)}(\gamma) = \left[\!\left[\Gamma \vdash \mathsf{next}^\kappa\,\xi'.\,\mathsf{next}^\kappa\,\xi.u : \overset{\kappa}{\blacktriangleright}\overset{\kappa}{\blacktriangleright} A\right]\!\right]_{(\mathcal{E},\delta)}(\gamma)$$

under the assumptions of the rule. For simplicity of presentation, we abuse notation and write $(\mathrm{tick}^{\gamma(\kappa)} \cdot \gamma, [\![\xi]\!]_{(\mathcal{E},\delta)}(\gamma))$ instead of $[\![\xi]\!]_{(\mathcal{E},\delta)}(\gamma)$ using (9), i.e., using $[\![\xi]\!]_{(\mathcal{E},\delta)}$ for just the non-trivial part of the substitution.

Note that the equation is trivial unless $\delta(\gamma(\kappa)) > 1$, so assume that is the case, and write δ^{--}

for $(\delta^{-\gamma(\kappa)})^{-\gamma(\kappa)}$ and tick for $\mathrm{tick}^{\gamma(\kappa)}$. Then

$$\begin{aligned}
&\left[\!\left[\Gamma \vdash \mathsf{next}^\kappa\, \xi.\, \mathsf{next}^\kappa\, \xi'.u : \overset{\kappa}{\blacktriangleright}\overset{\kappa}{\blacktriangleright} A\right]\!\right]_{(\mathcal{E},\delta)}(\gamma)\\
&= \left[\!\left[\Gamma, \Gamma' \vdash \mathsf{next}^\kappa\, \xi'.u : \overset{\kappa}{\blacktriangleright} A\right]\!\right]_{(\mathcal{E},\delta^{-\gamma(\kappa)}))}(\mathrm{tick}\cdot\gamma, [\![\xi]\!]_{(\mathcal{E},\delta)}(\gamma))\\
&= [\![\Gamma, \Gamma', \Gamma'' \vdash u : A]\!]_{(\mathcal{E},\delta^{--})}(\mathrm{tick}^2\cdot\gamma, \mathrm{tick}\cdot([\![\xi]\!]_{(\mathcal{E},\delta)}(\gamma)), [\![\xi']\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}(\mathrm{tick}\cdot\gamma))\\
&= [\![\Gamma, \Gamma', \Gamma'' \vdash u : A]\!]_{(\mathcal{E},\delta^{--})}(\mathrm{tick}^2\cdot\gamma, [\![\xi]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}(\mathrm{tick}\cdot\gamma)), [\![\xi']\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}(\mathrm{tick}\cdot\gamma))
\end{aligned}$$

A similar computation shows that $\left[\!\left[\Gamma \vdash \mathsf{next}^\kappa\, \xi'.\, \mathsf{next}^\kappa\, \xi.u : \overset{\kappa}{\blacktriangleright}\overset{\kappa}{\blacktriangleright} A\right]\!\right]_{(\mathcal{E},\delta)}(\gamma)$ reduces to the same. □

In addition to soundness of the rules, we must show that the substitution lemma 3.2 still holds in order for the retain soundness of the interpretation of the more basic features of the type theory.

Lemma 6.1. The statements of Lemma 3.2 hold for Core-GDTT extended with identity types and delayed substitutions.

Again this lemma is proved by extending the standard inductive proof of Lemma 3.2 in a straightforward way, and we omit the details.

To retain the soundness of the clock irrelevance axiom, the following lemma is needed.

Lemma 6.2. If the type $\Gamma, \Gamma' \vdash A\ \mathsf{type}$ is invariant under clock introduction, $\Gamma \vdash \kappa : \mathsf{clock}$ is a valid clock, and $\xi : \Gamma \to^\kappa \Gamma'$ is a delayed substitution then $\Gamma \vdash \overset{\kappa}{\blacktriangleright}\xi.A\ \mathsf{type}$ is invariant under clock introduction.

The proof is completely analogous to the proof of Lemma 4.5. The additional property which is needed is naturality of the interpretation of delayed substitution (9).

7. Universes

We now assume we are given a set theoretic universe with its induced notion of small sets. Being a presheaf category, $\mathrm{Set}^{\mathbb{T}}$ has a universe object $\mathcal{U}$ and a dependent type $\mathcal{E}l$ of elements defined as in (Hofmann and Streicher, 1999), as we now recall. If $(\mathcal{E}, \delta)$ is a time object, then the set $\mathcal{U}(\mathcal{E}, \delta)$ is the set of small dependent types over $y(\mathcal{E}, \delta)$. Concretely, an element X in $\mathcal{U}(\mathcal{E}, \delta)$ assigns to each $\sigma : (\mathcal{E}, \delta) \to (\mathcal{E}', \delta')$ a small set X_σ and to each $\tau : (\mathcal{E}', \delta') \to (\mathcal{E}'', \delta'')$ a map $\tau \cdot (-) : X_\sigma \to X_{\tau\sigma}$ in a functorial way. The action $\sigma \cdot (-) : \mathcal{U}(\mathcal{E}, \delta) \to \mathcal{U}(\mathcal{E}', \delta')$ maps an X to the family $(X_{\tau\sigma})_\tau$. The dependent type $\mathcal{E}l$ is defined as $\mathcal{E}l_{(\mathcal{E},\delta)}(X) = X_{\mathrm{id}}$ with action $\sigma \cdot (-) : \mathcal{E}l_{(\mathcal{E},\delta)}(X) \to \mathcal{E}l_{(\mathcal{E}',\delta')}(\sigma \cdot X)$ defined as $\sigma \cdot (-) : X_{\mathrm{id}} \to X_\sigma$.

One might hope that this universe could be used to model an extension of Core-GDTT with one universe. However, $\mathcal{U}$ is not a constant presheaf, and therefore not invariant under clock introduction. Another way to see this is that the map

$$\blacktriangleright : \mathcal{C} \times \mathcal{U} \to \mathcal{U}$$

defined, at $(\mathcal{E}, \delta) \in \mathrm{Set}^{\mathbb{T}}$, as

$$(\blacktriangleright(\lambda, X))_{\sigma:(\mathcal{E},\delta)\to(\mathcal{E}',\delta')} = \begin{cases} 1 & \text{if } \delta'(\sigma(\lambda)) = 0 \\ X_{\mathrm{tick}^{\sigma(\lambda)}\circ\sigma} & \text{else} \end{cases}$$

does not factor through the second projection. Restricting $\mathcal{U}$ to the families invariant under clock introduction does not rule out this map, and so does not eliminate the problem. Note that $\blacktriangleright$ above does indeed encode the type constructor $\blacktriangleright$ since if $\Gamma \vdash A : \mathcal{U}$ and $\kappa : \mathsf{clock} \in \Gamma$ then, if $\delta(\gamma(\kappa)) > 0$,

$$\begin{aligned}(\blacktriangleright(\gamma(\kappa), [\![\Gamma \vdash A : \mathcal{U}]\!]_{(\mathcal{E},\delta)}(\gamma)))_{\mathrm{id}} &= ([\![\Gamma \vdash A : \mathcal{U}]\!]_{(\mathcal{E},\delta)}(\gamma)))_{\mathrm{tick}^{\gamma(\kappa)}} \\ &= (\mathrm{tick}^{\gamma(\kappa)} \cdot [\![\Gamma \vdash A : \mathcal{U}]\!]_{(\mathcal{E},\delta)}(\gamma)))_{\mathrm{id}} \\ &= [\![\Gamma \vdash A : \mathcal{U}]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}(\mathrm{tick}^{\gamma(\kappa)} \cdot \gamma))_{\mathrm{id}} \\ &= \left[\!\!\left[\Gamma \vdash \overset{\kappa}{\blacktriangleright} \mathsf{El}(A)\ \mathsf{type}\right]\!\!\right]_{(\mathcal{E},\delta)}(\gamma)\end{aligned}$$

To avoid this problem we follow the approach of GDTT and introduce, for each finite set of clock variables Δ, a universe of types depending on the clocks in Δ. An element in this universe is to be thought of as being constant in the dimensions outside Δ, and the operation $\overset{\kappa}{\blacktriangleright}$ is only defined on the universe for $\kappa \in \Delta$. This rules out the more general $\blacktriangleright$ operation mentioned above. It also means that universes are now indexed over a new dimension (clock contexts). We show that the operations on types are polymorphic in this dimension.

7.1. *A family of semantic universes*

If Δ is a finite set of clocks, consider the object $\mathcal{C}^\Delta$ defined as the Δ-fold product of $\mathcal{C}$ by itself, i.e., $\mathcal{C}^\Delta(\mathcal{E}, \delta) = \mathcal{E}^\Delta$. We define a dependent family $\mathcal{U}^\Delta$ over $\mathcal{C}^\Delta$ and a dependent family $\mathcal{E}l^\Delta$ over $\mathcal{U}^\Delta$ below. But first we need some notation. For $\gamma \in \mathcal{C}^\Delta(\mathcal{E}, \delta)$, i.e., a function from Δ to $\mathcal{E}$, we write $\gamma[\Delta]$ for its image. More generally, if $\gamma \in [\![\Gamma]\!]_{(\mathcal{E},\delta)}$ and Δ is some subset of clock names in Γ then we write $\gamma[\Delta]$ for the subset $\{\gamma(\kappa) \mid \kappa \in \Delta\}$ of $\mathcal{E}$. Similarly for $\gamma \in [\![\Gamma]\!]_{(\mathcal{E},\delta)}$ and any subset Δ of clock variables of Γ we write $\gamma|_\Delta$ for the *sublist* of γ on the clock variables in Δ.[§]

With this notation we define $\mathcal{U}^\Delta_{(\mathcal{E},\delta)}(\gamma)$ to be the set of small dependent types on $y(\gamma[\Delta], \delta|_{\gamma[\Delta]})$ invariant under clock introduction. The notion of small dependent types should be understood as described above, i.e., as families of sets X_σ indexed over morphisms σ with domain $(\gamma[\Delta], \delta|_{\gamma[\Delta]})$ together with maps $\tau \cdot (-) : X_\sigma \to X_{\tau\sigma}$ satisfying functoriality. The requirement of invariance under clock introduction means that if $\iota : (\mathcal{E}', \delta') \to ((\mathcal{E}', \lambda), \delta'[\lambda \mapsto n])$ is an inclusion, then $\iota \cdot (-)$ must be an isomorphism.

For $\sigma : (\mathcal{E}, \delta) \to (\mathcal{E}', \delta')$ we must define $\sigma \cdot (-) : \mathcal{U}^\Delta_{(\mathcal{E},\delta)}(\gamma) \to \mathcal{U}^\Delta_{(\mathcal{E}',\delta')}(\sigma \cdot \gamma)$. Note first that $(\sigma \cdot \gamma)[\Delta] = \sigma[\gamma[\Delta]]$ and so we can consider the restriction and corestriction of σ:

$$\overline{\sigma}^{\gamma[\Delta]} : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\sigma[\gamma[\Delta]], \delta'|_{\sigma[\gamma[\Delta]]}) .$$

Using this, we define the family $\sigma \cdot X$ as the family whose component at a map τ with domain $(\sigma[\gamma[\Delta]], \delta'|_{\sigma[\gamma[\Delta]]})$ is $X_{\tau \circ \overline{\sigma}^{\gamma[\Delta]}}$. Note that this is well-defined, i.e., if X is invariant under clock introduction, so is $\sigma \cdot X$.

Lemma 7.1. The dependent family $\mathcal{U}^\Delta$ over $\mathcal{C}^\Delta$ is invariant under clock introduction.

[§] An alternative way to understand $\gamma|_\Delta$ is to view Δ as a morphism $[\![\Delta]\!] : [\![\Gamma \vdash]\!] \to \mathcal{C}^\Delta$, where $\mathcal{C}^\Delta$ is the $|\Delta|$-fold product of the presheaf $\mathcal{C}$. Then $\gamma|_\Delta$ is the application of $[\![\Delta]\!]_{(\mathcal{E},\delta)}$ to $\gamma \in [\![\Gamma \vdash]\!]_{(\mathcal{E},\delta)}$.

Proof. If $\iota : (\mathcal{E}, \delta) \to ((\mathcal{E}, \lambda), \delta[\lambda \mapsto n])$ is the inclusion then $(\iota \cdot \gamma)[\Delta] = \gamma[\Delta]$, so

$$((\iota \cdot \gamma)[\Delta], \delta[\lambda \mapsto n]|_{(\iota \cdot \gamma)[\Delta]}) = (\gamma[\Delta], \delta|_{\gamma[\Delta]})$$

and thus $\mathcal{U}^{\Delta}_{(\mathcal{E},\delta)}(\gamma) = \mathcal{U}^{\Delta}_{((\mathcal{E},\lambda),\delta[\lambda\mapsto n])}(\iota \cdot \gamma)$. Since $\overline{\iota}^{\gamma[\Delta]}$ is the identity, also $\iota \cdot (-) : \mathcal{U}^{\Delta}_{(\mathcal{E},\delta)}(\gamma) \to \mathcal{U}^{\Delta}_{((\mathcal{E},\lambda),\delta[\lambda\mapsto n])}(\iota \cdot \gamma)$ is the identity map, in particular an isomorphism. □

If X is an element in $\mathcal{U}^{\Delta}_{(\mathcal{E},\delta)}(\gamma)$, we define

$$\mathcal{E}l^{\Delta}_{(\mathcal{E},\delta)}(\gamma, X) = X_{i:(\gamma[\Delta],\delta|_{\gamma[\Delta]})\to(\mathcal{E},\delta)}$$

where i is the inclusion. If $\sigma : (\mathcal{E}, \delta) \to (\mathcal{E}', \delta')$ we must define

$$\sigma \cdot (-) : \mathcal{E}l^{\Delta}_{(\mathcal{E},\delta)}(\gamma, X) \to \mathcal{E}l^{\Delta}_{(\mathcal{E}',\delta')}(\sigma \cdot \gamma, \sigma \cdot X)$$

The codomain of this map is

$$\begin{aligned}\mathcal{E}l^{\Delta}_{(\mathcal{E}',\delta')}(\sigma \cdot \gamma, \sigma \cdot X) &= (\sigma \cdot X)_j \\ &= X_{j \circ \overline{\sigma}^{\gamma[\Delta]}} \\ &= X_{\sigma \circ i}\end{aligned}$$

where $j : \sigma[\gamma[\Delta]] \to \mathcal{E}'$ is the inclusion. We can therefore define $\sigma \cdot (-) : X_i \to X_{\sigma \circ i}$ to be the map that is part of the structure of X.

Lemma 7.2. The dependent family $\mathcal{E}l^{\Delta}$ over $\mathcal{U}^{\Delta}$ is invariant under clock introduction.

Proof. Let $\iota : (\mathcal{E}, \delta) \to ((\mathcal{E}, \lambda), \delta[\lambda \mapsto n])$ be the inclusion and let X be an element in $\mathcal{U}^{\Delta}_{(\mathcal{E},\delta)}(\gamma)$. We must show that

$$\iota \cdot (-) : \mathcal{E}l^{\Delta}_{(\mathcal{E},\delta)}(\gamma, X) \to \mathcal{E}l^{\Delta}_{((\mathcal{E},\lambda),\delta[\lambda\mapsto n])}(\iota \cdot \gamma, \iota \cdot X)$$

is an isomorphism. By definition of $\mathcal{E}l^{\Delta}$ this map is $\iota \cdot (-) : X_i \to X_{\iota \circ i}$, which is part of the structure of X. Since X is an element in the universe $\mathcal{U}^{\Delta}$ it must be invariant under clock introduction, which means exactly that all maps of the form $\iota \cdot (-)$ are isomorphisms. □

7.2. *Basic syntax and semantics of universes*

The syntax and interpretation of universes is presented in Figure 6. For the moment, we postpone codes of type operations on universes, and so the figure only shows the very basic rules. In the interpretation of $\Gamma \vdash \mathsf{U}_{\Delta}\ \mathsf{type}$, if $\kappa \in \Delta$, then κ is mentioned in Γ, so $\gamma(\kappa) \in \mathcal{C}(\mathcal{E}, \delta)$. So $\gamma|_{\Delta} \in \mathcal{E}^{\Delta}$, i.e., it is an element of $\mathcal{C}^{\Delta}_{(\mathcal{E},\delta)}$.

We now show the cases necessary for the appropriate extension of Corollary 4.3.

Lemma 7.3. Context projections of the form $[\![\Gamma, X : \mathsf{U}_{\Delta} \vdash]\!] \to [\![\Gamma \vdash]\!]$ and $[\![\Gamma, x : \mathsf{El}_{\Delta}(t) \vdash]\!] \to [\![\Gamma \vdash]\!]$ are invariant under clock introduction.

Proof. The context projection $[\![\Gamma, X : \mathsf{U}_{\Delta} \vdash]\!] \to [\![\Gamma \vdash]\!]$ is the pullback of $\mathcal{U}^{\Delta} \to \mathcal{C}^{\Delta}$ along the map that maps $\gamma \in [\![\Gamma \vdash]\!]_{(\mathcal{E},\delta)}$ to $\gamma|_{\Delta}$. Since orthogonality is preserved by pullback (Proposition 4.1) the statement follows from Lemma 7.1. The second statement follows similarly from Lemma 7.2. □

We then prove the relevant new cases for the substitution lemma (Lemma 3.2).

Formation and typing rules

$$\frac{\Gamma \vdash \kappa_1 : \mathsf{clock} \qquad \ldots \qquad \Gamma \vdash \kappa_n : \mathsf{clock}}{\Gamma \vdash \mathsf{U}_{\kappa_1,\ldots,\kappa_n}\ \mathsf{type}} \qquad \frac{\Gamma \vdash t : \mathsf{U}_\Delta}{\Gamma \vdash \mathsf{El}_\Delta(t)\ \mathsf{type}}$$

$$\frac{\Gamma \vdash t : \mathsf{U}_\Delta \qquad \Gamma \vdash \mathsf{U}_{\Delta'}\ \mathsf{type} \qquad \Delta \subseteq \Delta'}{\Gamma \vdash \mathsf{in}_{\Delta,\Delta'}(t) : \mathsf{U}_{\Delta'}}$$

Equations

$$\mathsf{U}_\Delta \equiv \mathsf{U}_{\Delta'} \qquad \text{if } \Delta = \Delta' \text{ as sets}$$
$$\mathsf{El}_{\Delta'}(\mathsf{in}_{\Delta,\Delta'}(t)) \equiv \mathsf{El}_\Delta(t)$$

Interpretation

$$[\![\Gamma \vdash \mathsf{U}_\Delta\ \mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma) = \mathcal{U}^\Delta_{(\mathcal{E},\delta)}(\gamma|_\Delta)$$
$$[\![\Gamma \vdash \mathsf{El}_\Delta(t)\ \mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma) = \mathcal{E}l^\Delta_{(\mathcal{E},\delta)}(\gamma|_\Delta, [\![t]\!]_{(\mathcal{E},\delta)}(\gamma))$$
$$([\![\Gamma \vdash \mathsf{in}_{\Delta,\Delta'}(t) : \mathsf{U}_{\Delta'}]\!]_{(\mathcal{E},\delta)}(\gamma))_\sigma = ([\![\Gamma \vdash t : \mathsf{U}_\Delta]\!]_{(\mathcal{E},\delta)}(\gamma))_{(\sigma\circ i)}$$
where $i : \gamma[\Delta] \to \gamma[\Delta']$ is the inclusion

Figure 6. Syntax and semantics of universes.

Lemma 7.4. Let $\rho : \Gamma \to \Gamma'$ be a substitution.

— if $\Gamma' \vdash \mathsf{U}_\Delta\ \mathsf{type}$ then also $\Gamma \vdash (\mathsf{U}_\Delta)\rho\ \mathsf{type}$ and

$$[\![\Gamma \vdash \mathsf{U}_\Delta\, \rho\ \mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma) = [\![\Gamma' \vdash \mathsf{U}_\Delta\ \mathsf{type}]\!]_{(\mathcal{E},\delta)}([\![\rho]\!](\gamma)),$$

— if $\Gamma' \vdash t : \mathsf{U}_\Delta$ and $\Gamma \vdash t\rho : (\mathsf{U}_\Delta)\rho$ and $\Gamma' \vdash \mathsf{U}_{\Delta'}\ \mathsf{type}$ then

$$\Gamma \vdash (\mathsf{El}_\Delta(t))\rho\ \mathsf{type} \qquad \text{and} \qquad \Gamma \vdash (\mathsf{in}_{\Delta,\Delta'}(t))\rho : (\mathsf{U}_{\Delta'})\rho,$$

— if, moreover,

$$[\![\Gamma \vdash t\rho : (\mathsf{U}_\Delta)\rho]\!]_{(\mathcal{E},\delta)}(\gamma) = [\![\Gamma' \vdash t : \mathsf{U}_\Delta]\!]_{(\mathcal{E},\delta)}([\![\rho]\!](\gamma))$$

then

$$[\![\Gamma \vdash (\mathsf{El}_\Delta(t))\rho\ \mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma) = [\![\Gamma \vdash \mathsf{El}_\Delta(t)\ \mathsf{type}]\!]_{(\mathcal{E},\delta)}([\![\rho]\!](\gamma))$$

and

$$[\![\Gamma \vdash (\mathsf{in}_{\Delta,\Delta'}(t))\rho : (\mathsf{U}_{\Delta'})\rho]\!]_{(\mathcal{E},\delta)}(\gamma) = [\![\Gamma \vdash \mathsf{in}_{\Delta,\Delta'}(t) : \mathsf{U}_{\Delta'}]\!]_{(\mathcal{E},\delta)}([\![\rho]\!](\gamma)).$$

Proof. Suppose $\Gamma' \vdash \mathsf{U}_\Delta\ \mathsf{type}$ and $\Delta = \kappa_1, \ldots \kappa_n$. Then $\Gamma \vdash \rho(\kappa_i) : \mathsf{clock}$ for all i, and so $\Gamma \vdash \mathsf{U}_{\rho(\kappa_1),\ldots,\rho(\kappa_n)}\ \mathsf{type}$. For the semantics,

$$[\![\Gamma \vdash \mathsf{U}_{\rho(\kappa_1),\ldots,\rho(\kappa_n)}\ \mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma) = \mathcal{U}^{\rho[\Delta]}_{(\mathcal{E},\delta)}(\gamma|_{\rho[\Delta]})$$

and since $[\![\rho]\!](\gamma)(\kappa_i) = \gamma(\rho(\kappa_i))$

$$[\![\Gamma' \vdash \mathsf{U}_\Delta\ \mathsf{type}]\!]_{(\mathcal{E},\delta)}([\![\rho]\!](\gamma)) = \mathcal{U}^\Delta_{(\mathcal{E},\delta)}((\gamma\circ\rho)|_\Delta)$$

Both $\mathcal{U}^{\rho[\Delta]}_{(\mathcal{E},\delta)}(\gamma|_{\rho[\Delta]})$ and $\mathcal{U}^\Delta_{(\mathcal{E},\delta)}((\gamma\circ\rho)|_\Delta)$ are the set of small dependent families on the object $y(\gamma[\rho[\Delta]], \delta'|_{\gamma[\rho[\Delta]]})$ invariant under clock introduction.

The other statements follow similarly. □

Lemma 7.5. The interpretation is sound.

Proof. We must show that equal terms and types are interpreted equally. If $\Delta = \Delta'$ as sets, then $\gamma[\Delta] = \gamma[\Delta']$ and $\gamma|_\Delta = \gamma|_{\Delta'}$ so $[\![\Gamma \vdash \mathsf{U}_\Delta\ \mathsf{type}]\!] = [\![\Gamma \vdash \mathsf{U}_{\Delta'}\ \mathsf{type}]\!]$.

For the other equality, suppose $\Delta \subseteq \Delta'$ and let $j : \gamma[\Delta'] \to \mathcal{E}$ and $i : \gamma[\Delta] \to \gamma[\Delta']$ be the inclusions. Then

$$\begin{aligned}[\![\Gamma \vdash \mathsf{El}_{\Delta'}(\mathsf{in}_{\Delta,\Delta'}(t))\ \mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma) &= ([\![\Gamma \vdash \mathsf{in}_{\Delta,\Delta'}(t) : \mathsf{U}_{\Delta'}]\!]_{(\mathcal{E},\delta)}(\gamma))_j \\ &= ([\![\Gamma \vdash t : \mathsf{U}_{\Delta'}]\!]_{(\mathcal{E},\delta)}(\gamma))_{ji} \\ &= [\![\Gamma \vdash \mathsf{El}_\Delta(t)\ \mathsf{type}]\!]_{(\mathcal{E},\delta)}(\gamma)\,.\end{aligned}$$

□

8. Codes of type constructors on universes

Having established the basic framework of universes, we now show how to add codes for type constructors to the universes. We start with dependent sums and products.

8.1. *Codes for dependent sums and products*

Figure 7 describes the rules for these. Note that the left hand sides of the two last equalities are well-formed because $\mathsf{El}_{\Delta'}(\mathsf{in}_{\Delta,\Delta'}(A)) \equiv \mathsf{El}_\Delta(A)$.

Formation and typing rules

$$\frac{\Gamma \vdash A : \mathsf{U}_\Delta \qquad \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta}{\Gamma \vdash \overline{\prod}_\Delta\,(x : A)\,.B : \mathsf{U}_\Delta} \qquad \frac{\Gamma \vdash A : \mathsf{U}_\Delta \qquad \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta}{\Gamma \vdash \overline{\sum}_\Delta\,(x : A)\,.B : \mathsf{U}_\Delta}$$

Equations

$$\begin{aligned}\mathsf{El}_\Delta(\overline{\prod}_\Delta\,(x : A)\,.B) &\equiv \prod\,(x : \mathsf{El}_\Delta\,A)\,.\,\mathsf{El}_\Delta\,B \\ \mathsf{El}_\Delta(\overline{\sum}_\Delta\,(x : A)\,.B) &\equiv \sum\,(x : \mathsf{El}_\Delta\,A)\,.\,\mathsf{El}_\Delta\,B \\ \overline{\prod}_{\Delta'}\,(x : \mathsf{in}_{\Delta,\Delta'}(A))\,.\,\mathsf{in}_{\Delta,\Delta'}(B) &\equiv \mathsf{in}_{\Delta,\Delta'}(\overline{\prod}_\Delta\,(x : A)\,.B) \\ \overline{\sum}_{\Delta'}\,(x : \mathsf{in}_{\Delta,\Delta'}(A))\,.\,\mathsf{in}_{\Delta,\Delta'}(B) &\equiv \mathsf{in}_{\Delta,\Delta'}(\overline{\sum}_\Delta\,(x : A)\,.B)\end{aligned}$$

Figure 7. Syntax for codes for dependent products

We now turn to the semantics of these codes. Recall that if A and B are as in the premises of the rule, then

$$[\![\Gamma \vdash A : \mathsf{U}_\Delta]\!]_{(\mathcal{E},\delta)}(\gamma)$$

is a small dependent type on $y(\gamma[\Delta], \delta|_{\gamma[\Delta]})$. To model the code for dependent products, we would like to view the interpretation of B as a small dependent type on $[\![\Gamma \vdash A : \mathsf{U}_\Delta]\!]_{(\mathcal{E},\delta)}(\gamma)$. More precisely, we will show (Lemma 8.2 below) that the interpretation of B gives rise to a family of small sets $([\![\Gamma \vdash (x : A).B]\!]_{(\mathcal{E},\delta)}(\gamma))_\sigma(x)$ indexed over $\sigma : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\mathcal{E}', \delta')$ and

$x \in (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\gamma))_\sigma$. We can use this to model $\left(\llbracket \Gamma \vdash \overline{\sum}_\Delta (x : A).B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\gamma) \right)_\sigma$ as the set of pairs (x, y) where $x \in (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\gamma))_\sigma$ and $y \in (\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_\sigma(x)$. Note that one can not define $(\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_\sigma(x)$ simply as

$$\llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash \mathsf{El}_\Delta(B)\ \mathsf{type} \rrbracket_{(\mathcal{E}',\delta')}(\sigma \cdot \gamma, x)$$

as one might hope, since $\gamma \in \llbracket \Gamma \rrbracket_{(\mathcal{E},\delta)}$ and $\sigma : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\mathcal{E}', \delta')$, and so $\sigma \cdot \gamma$ is not well defined. As a first step we need to extend any such map σ to all of $(\mathcal{E}, \delta)$, and to do that we need coproducts in **Fin**. Precisely, we will assume the following.

Assumption 8.1. The category **Fin** comes equipped with a choice of coproducts such that for any $\mathcal{E}, \mathcal{E}'$ the left inclusion $\mathsf{inl} : \mathcal{E} \to \mathcal{E} + \mathcal{E}'$ satisfies $\mathsf{inl}(\lambda) = \lambda$ for all λ in $\mathcal{E}$.

For example, if the infinite set of clocks over which the category **Fin** is defined is the set of natural numbers, one can define

$$\mathcal{E} + \mathcal{E}' = \mathcal{E} \cup \{n + 1 + \mathsf{max}(\mathcal{E}) \mid n \in \mathcal{E}'\}$$

if $\mathcal{E}$ is non-empty. The requirement that $\mathsf{inl}(\lambda) = \lambda$ is needed to make certain equalities hold up to identity rather than just isomorphism.

We start by defining the sets of the family $(\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_\sigma(x)$. The definition uses the following slight abuse of notation: If $\sigma : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\mathcal{E}', \delta')$ and $i : \mathcal{E} \setminus \gamma[\Delta] \to \mathcal{E}$ is the inclusion we write $\sigma + i : \mathcal{E} \to \mathcal{E}' + \mathcal{E}$ for the map defined as

$$(\sigma + i)(\lambda) = \begin{cases} \mathsf{inl}(\sigma(\lambda)) & \text{if } \lambda \in \gamma[\Delta] \\ \mathsf{inr}(\lambda) & \text{else} \end{cases}$$

Definition 8.1. Let $\Gamma \vdash A : \mathsf{U}_\Delta$, and $\Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta$, and $\gamma \in \llbracket \Gamma \rrbracket_{(\mathcal{E},\delta)}$. If, moreover, $\sigma : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\mathcal{E}', \delta')$ and $x \in (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\gamma))_\sigma$ define $(\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_\sigma(x)$ as

$$\llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\sigma + i) \cdot \gamma, \mathsf{inl} \cdot x)_{j:(\sigma\gamma[\Delta],\delta'|_{\sigma\gamma[\Delta]}) \to (\mathcal{E}',\delta')}$$

where j is the inclusion.

Note that the inclusion j is into $\mathcal{E}'$ and not $\mathcal{E}' + \mathcal{E}$, so this is not an application of the denotation of $\mathsf{El}_\Delta(B)$. If one defines this as the denotation of $\mathsf{El}_\Delta(B)$ at $((\sigma + i) \cdot \gamma, \mathsf{inl} \cdot x)$, then Lemma 8.3 fails.

To see Definition 8.1 is well-defined, we must show that

$$\mathsf{inl} \cdot x \in \llbracket \Gamma \vdash \mathsf{El}_\Delta(A)\ \mathsf{type} \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\sigma + i) \cdot \gamma)$$

and that the domain of j is $((\sigma + i) \cdot \gamma)[\Delta]$. The latter follows from Assumption 8.1 since that gives $((\sigma + i) \cdot \gamma)[\Delta] = \sigma\gamma[\Delta]$. For the former, a priori, the type of $\mathsf{inl} \cdot x$ is $(\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\mathsf{inl} \circ \sigma}$. and so we must prove

$$(\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\mathsf{inl} \circ \sigma} = \llbracket \Gamma \vdash \mathsf{El}_\Delta(A)\ \mathsf{type} \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\sigma + i) \cdot \gamma)$$

which follows from the following lemma, since $(\sigma + i)h = \mathsf{inl} \circ \sigma$, where $h : \gamma[\Delta] \to \mathcal{E}$ is the inclusion.

Lemma 8.1. Let $h : \gamma[\Delta] \to \mathcal{E}$ be the inclusion, let $\tau : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\mathcal{E}', \delta')$ and let $\omega :$

$(\mathcal{E},\delta) \to (\mathcal{E}',\delta')$ be an extension of τ in the sense that $\omega h = \tau$. Then, for any $\gamma \in [\![\Gamma]\!]_{(\mathcal{E},\delta)}$,

$$([\![\Gamma \vdash A : \mathsf{U}_\Delta]\!]_{(\mathcal{E},\delta)}(\gamma))_\tau = [\![\Gamma \vdash \mathsf{El}_\Delta(A)\ \mathsf{type}]\!]_{(\mathcal{E}',\delta')}(\omega \cdot \gamma)$$

Proof. Let $\overline{\omega} : \gamma[\Delta] \to \omega[\gamma[\Delta]]$ be the restriction and corestriction of ω. Then $\tau = j\overline{\omega}$, where $j : \omega[\gamma[\Delta]] \to \mathcal{E}'$ is the inclusion. So

$$\begin{aligned}([\![\Gamma \vdash A : \mathsf{U}_\Delta]\!]_{(\mathcal{E},\delta)}(\gamma))_\tau &= ([\![\Gamma \vdash A : \mathsf{U}_\Delta]\!]_{(\mathcal{E},\delta)}(\gamma))_{j\overline{\omega}} \\ &= (\omega \cdot ([\![\Gamma \vdash A : \mathsf{U}_\Delta]\!]_{(\mathcal{E},\delta)}(\gamma)))_j \\ &= ([\![\Gamma \vdash A : \mathsf{U}_\Delta]\!]_{(\mathcal{E}',\delta')}(\omega \cdot \gamma))_j \\ &= [\![\Gamma \vdash \mathsf{El}_\Delta(A)\ \mathsf{type}]\!]_{(\mathcal{E}',\delta')}(\omega \cdot \gamma)\end{aligned}$$

□

Lemma 8.2. Suppose $\Gamma \vdash A : \mathsf{U}_\Delta$, and $\Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta$, and $\gamma \in [\![\Gamma]\!]_{(\mathcal{E},\delta)}$. The family of sets $([\![\Gamma \vdash (x : A).B]\!]_{(\mathcal{E},\delta)}(\gamma))_\sigma(x)$ defined as in Definition 8.1 extends to a small dependent type over $[\![\Gamma \vdash A : \mathsf{U}_\Delta]\!]_{(\mathcal{E},\delta)}(\gamma)$ considered as a small dependent type over $y(\gamma[\Delta], \delta|_{\gamma[\Delta]})$. In other words, for all $\sigma : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\mathcal{E}',\delta')$, $x \in ([\![\Gamma \vdash A : \mathsf{U}_\Delta]\!]_{(\mathcal{E},\delta)}(\gamma))_\sigma$, and $\tau : (\mathcal{E}',\delta') \to (\mathcal{E}'',\delta'')$ there is a map

$$\tau \cdot (-) : ([\![\Gamma \vdash (x : A).B]\!]_{(\mathcal{E},\delta)}(\gamma))_\sigma(x) \to ([\![\Gamma \vdash (x : A).B]\!]_{(\mathcal{E},\delta)}(\gamma))_{\tau\sigma}(\tau \cdot x)\,. \tag{10}$$

satisfying $\mathrm{id} \cdot y = y$ and $\omega \cdot (\tau \cdot y) = (\omega\tau) \cdot y$. Moreover, this family is invariant under clock introduction.

Proof. First note that if $k : (\tau\sigma\gamma[\Delta], \delta''|_{\tau\sigma\gamma[\Delta]}) \to (\mathcal{E}'',\delta'')$ is the inclusion, then

$$\begin{aligned}&([\![\Gamma \vdash (x : A).B]\!]_{(\mathcal{E},\delta)}(\gamma))_{\tau\sigma}(\tau \cdot x) \\ &= ([\![\Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta]\!]_{(\mathcal{E}''+\mathcal{E},[\delta'',\delta])}((\tau\sigma + i) \cdot \gamma, \mathsf{inl} \cdot (\tau \cdot x))_k \\ &= ([\![\Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta]\!]_{(\mathcal{E}''+\mathcal{E},[\delta'',\delta])}((\tau + \mathrm{id}) \cdot (\sigma + i) \cdot \gamma, (\tau + \mathrm{id}) \cdot \mathsf{inl} \cdot x))_k \\ &= ((\tau + \mathrm{id}) \cdot [\![\Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta]\!]_{(\mathcal{E}''+\mathcal{E},[\delta'',\delta])}((\sigma + i) \cdot \gamma, \mathsf{inl} \cdot x))_k \\ &= ([\![\Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta]\!]_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\sigma + i) \cdot \gamma, \mathsf{inl} \cdot x))_{k\overline{(\tau+\mathrm{id})}}\end{aligned}$$

where $\overline{(\tau + \mathrm{id})} : (\sigma + i)[\gamma[\Delta]] \to (\tau\sigma + i)[\gamma[\Delta]]$ is restriction and corestriction of $\tau + \mathrm{id}$. By Assumption 8.1 $(\sigma + i)\gamma[\Delta] = \sigma\gamma[\Delta]$ and $\overline{(\tau + \mathrm{id})}^{\sigma\gamma[\Delta]} = \overline{\tau}^{\sigma\gamma[\Delta]}$, so

$$\begin{aligned}([\![\Gamma \vdash (x : A).B]\!]_{(\mathcal{E},\delta)}(\gamma))_{\tau\sigma}(\tau \cdot x) &= [\![\Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta]\!]_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\sigma + i) \cdot \gamma, \mathsf{inl} \cdot x))_{k\overline{\tau}^{\sigma\gamma[\Delta]}} \\ &= ([\![\Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta]\!]_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\sigma + i) \cdot \gamma, \mathsf{inl} \cdot x))_{\tau j}\end{aligned}$$

where the last equality uses

$$\begin{array}{ccc} \sigma\gamma[\Delta] & \xrightarrow{\overline{\tau}^{\sigma\gamma[\Delta]}} & \tau\sigma\gamma[\Delta] \\ \downarrow{\scriptstyle j} & & \downarrow{\scriptstyle k} \\ \mathcal{E}' & \xrightarrow{\ \tau\ } & \mathcal{E}'' \end{array}$$

Since $[\![\Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta]\!]_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\sigma + i) \cdot \gamma, \mathsf{inl} \cdot x)$ is an element in $\mathcal{U}^\Delta(((\sigma + i) \cdot \gamma)|_\Delta)$

$$\left(\left[\!\left[\Gamma \vdash \overline{\sum}_{\Delta}\left(x : A\right).B : \mathsf{U}_{\Delta}\right]\!\right]_{(\mathcal{E},\delta)}(\gamma)\right)_{\sigma} = \left\{ (x,y) \;\middle|\; \begin{array}{l} x \in (\llbracket \Gamma \vdash A : \mathsf{U}_{\Delta} \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\sigma}, \\ y \in (\llbracket \Gamma \vdash (x:A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\sigma}(x) \end{array} \right\}$$

$$\tau \cdot (x,y) = (\tau \cdot x, \tau \cdot y)$$

$$\left(\left[\!\left[\Gamma \vdash \overline{\prod}_{\Delta}\left(x : A\right).B : \mathsf{U}_{\Delta}\right]\!\right]_{(\mathcal{E},\delta)}(\gamma)\right)_{\sigma} =$$

$$\left\{ (f_{\tau})_{\tau:(\mathcal{E}',\delta')\to(\mathcal{E}'',\delta'')} \;\middle|\; \begin{array}{l} \forall \tau. f_{\tau} : (x : (\llbracket \Gamma \vdash A : \mathsf{U}_{\Delta} \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau\sigma}) \\ \qquad \to (\llbracket \Gamma \vdash (x:A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau\sigma}(x), \\ \forall \rho, x. f_{\rho\tau}(\rho \cdot x) = \rho \cdot f_{\tau}(x) \end{array} \right\}$$

$$(\tau \cdot f)_{\rho} = f_{\rho\tau}$$

Figure 8. Interpretation of dependent products and sums. The function σ is assumed to have type $\sigma : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\mathcal{E}', \delta')$ for some $(\mathcal{E}', \delta')$.

there is a map

$$\begin{aligned} \tau \cdot (-) : \llbracket \Gamma, x : \mathsf{El}_{\Delta}(A) \vdash B : \mathsf{U}_{\Delta} \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\sigma + i) \cdot \gamma, \mathsf{inl} \cdot x)_j \\ \to \llbracket \Gamma, x : \mathsf{El}_{\Delta}(A) \vdash B : \mathsf{U}_{\Delta} \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\sigma + i) \cdot \gamma, \mathsf{inl} \cdot x))_{\tau j} \end{aligned} \tag{11}$$

Thus we can define the desired map (10) to be (11). Equations $\mathrm{id} \cdot z = z$ and $\rho \cdot (\tau \cdot z) = (\rho \circ \tau) \cdot z$ then follow because they hold for the action (11). Invariance under clock introduction follows likewise from $\llbracket \Gamma, x : \mathsf{El}_{\Delta}(A) \vdash B : \mathsf{U}_{\Delta} \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\sigma + i) \cdot \gamma, \mathsf{inl} \cdot x)$ being an element in $\mathcal{U}^{\Delta}(\gamma|_{\Delta})$ □

We can thus define $\left[\!\left[\Gamma \vdash \overline{\sum}_{\Delta}\left(x : A\right).B : \mathsf{U}_{\Delta}\right]\!\right]_{(\mathcal{E},\delta)}(\gamma)$ and $\left[\!\left[\Gamma \vdash \overline{\prod}_{\Delta}\left(x : A\right).B : \mathsf{U}_{\Delta}\right]\!\right]_{(\mathcal{E},\delta)}(\gamma)$ to be the dependent products and sums for the dependent type $\llbracket \Gamma \vdash (x:A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma)$. This gives small dependent types over $y(\gamma[\Delta], \delta|_{\gamma[\Delta]})$ and these are invariant under introduction of clocks because the collection of these types is closed under dependent products and sums by Proposition 4.1. The resulting interpretation is summarised Figure 8.

For this to be a well-defined interpretation of terms, we must show that it is natural with respect to the restriction maps as stated in the following lemma.

Lemma 8.3. If $\gamma \in \llbracket \Gamma \rrbracket_{(\mathcal{E},\delta)}$ and $\tau : (\mathcal{E}, \delta) \to (\mathcal{E}', \delta')$ then

$$\left[\!\left[\Gamma \vdash \overline{\sum}_{\Delta}\left(x : A\right).B : \mathsf{U}_{\Delta}\right]\!\right]_{(\mathcal{E}',\delta')}(\tau \cdot \gamma) = \tau \cdot \left(\left[\!\left[\Gamma \vdash \overline{\sum}_{\Delta}\left(x : A\right).B : \mathsf{U}_{\Delta}\right]\!\right]_{(\mathcal{E},\delta)}(\gamma)\right)$$

$$\left[\!\left[\Gamma \vdash \overline{\prod}_{\Delta}\left(x : A\right).B : \mathsf{U}_{\Delta}\right]\!\right]_{(\mathcal{E}',\delta')}(\tau \cdot \gamma) = \tau \cdot \left(\left[\!\left[\Gamma \vdash \overline{\prod}_{\Delta}\left(x : A\right).B : \mathsf{U}_{\Delta}\right]\!\right]_{(\mathcal{E},\delta)}(\gamma)\right)$$

For the proof of Lemma 8.3 we need the following lemma.

Lemma 8.4. Suppose $\gamma \in \llbracket \Gamma \rrbracket_{(\mathcal{E},\delta)}$, $\tau : (\mathcal{E}, \delta) \to (\mathcal{E}', \delta')$ and $\sigma : (\tau\gamma[\Delta], \delta'|_{\tau\gamma[\Delta]}) \to (\mathcal{E}'', \delta'')$. If $x \in (\llbracket \Gamma \vdash A : \mathsf{U}_{\Delta} \rrbracket_{(\mathcal{E}',\delta')}(\tau \cdot \gamma))_{\sigma}$ then

$$(\llbracket \Gamma \vdash (x:A).B \rrbracket_{(\mathcal{E}',\delta')}(\tau \cdot \gamma))_{\sigma}(x) = (\llbracket \Gamma \vdash (x:A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\sigma\overline{\tau}}(x) \tag{12}$$

where $\overline{\tau} = \overline{\tau}^{\gamma[\Delta]} : \gamma[\Delta] \to \tau\gamma[\Delta]$ is the restriction/corestriction of τ. Moreover, for any ω :

$(\mathcal{E}'', \delta'') \to (\mathcal{E}''', \delta''')$ the maps

$$\omega \cdot (-) : (\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E}', \delta')} (\tau \cdot \gamma))_\sigma(x) \to (\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E}', \delta')} (\tau \cdot \gamma))_{\omega\sigma}(\omega \cdot x)$$

and

$$\omega \cdot (-) : (\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E}, \delta)} (\gamma))_{\sigma\overline{\tau}}(x) \to (\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E}, \delta)} (\gamma))_{\omega\sigma\overline{\tau}}(\omega \cdot x)$$

are equal.

For the right hand side of (12) to be well-defined we need $x \in (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}, \delta)} (\gamma))_{\sigma\overline{\tau}}$. This holds because

$$\begin{aligned} (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}', \delta')} (\tau \cdot \gamma))_\sigma &= (\tau \cdot (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}, \delta)} (\gamma)))_\sigma \\ &= (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}, \delta)} (\gamma))_{\sigma\overline{\tau}} \end{aligned} \tag{13}$$

Proof. By definition

$$(\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E}', \delta')} (\tau \cdot \gamma))_\sigma(x) = \llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}'' + \mathcal{E}', [\delta'', \delta'])} ((\sigma + j) \cdot \tau \cdot \gamma, \mathsf{inl} \cdot x)_h$$

where $h : (\sigma\tau\gamma[\Delta], \delta''|_{\sigma\tau\gamma[\Delta]}) \to (\mathcal{E}'', \delta'')$ and $j : (\mathcal{E}' \setminus \tau\gamma[\Delta]) \to \mathcal{E}'$ are inclusions.

Consider the map $\phi : ((\mathcal{E}'' + \mathcal{E}), [\delta'', \delta]) \to ((\mathcal{E}'' + \mathcal{E}'), [\delta'', \delta'])$ defined as

$$\begin{aligned} \phi(\mathsf{inl}(\lambda)) &= \mathsf{inl}(\lambda) \\ \phi(\mathsf{inr}(\lambda)) &= \begin{cases} \mathsf{inl}(\sigma\tau(\lambda)) & \text{if } \tau(\lambda) \in \tau\gamma[\Delta] \\ \mathsf{inr}(\tau(\lambda)) & \text{if } \tau(\lambda) \notin \tau\gamma[\Delta] \end{cases} \end{aligned}$$

and note that the following diagram commutes

$$\begin{array}{ccc} \mathcal{E} & \xrightarrow{(\sigma\overline{\tau}+i)} & \mathcal{E}'' + \mathcal{E} \\ \downarrow{\scriptstyle \tau} & & \downarrow{\scriptstyle \phi} \\ \mathcal{E}' & \xrightarrow{\sigma+j} & \mathcal{E}'' + \mathcal{E}' \end{array}$$

which can be seen by considering the three cases: $\lambda \in \gamma[\Delta], \lambda \in \tau^{-1}(\tau\gamma[\Delta]) \setminus \gamma[\Delta]$, and $\lambda \notin \tau^{-1}(\tau\gamma[\Delta])$. Since $\phi \circ \mathsf{inl} = \mathsf{inl}$ we get

$$\begin{aligned} &(\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E}', \delta')} (\tau \cdot \gamma))_\sigma(x) \\ &\quad = \llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}'' + \mathcal{E}', [\delta'', \delta'])} (\phi \cdot (\sigma\overline{\tau} + i) \cdot \gamma, \phi \cdot \mathsf{inl} \cdot x)_h \\ &\quad = \llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}'' + \mathcal{E}, [\delta'', \delta])} ((\sigma\overline{\tau} + i) \cdot \gamma, \mathsf{inl} \cdot x)_{h \circ \overline{\phi}} \\ &\quad = \llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}'' + \mathcal{E}, [\delta'', \delta])} ((\sigma\overline{\tau} + i) \cdot \gamma, \mathsf{inl} \cdot x)_h \\ &\quad = (\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E}, \delta)} (\gamma))_{\sigma\overline{\tau}}(x) \end{aligned}$$

where the third equality uses that the restriction/corestriction $\overline{\phi} : (\sigma\overline{\tau} + i)\gamma[\Delta] \to \sigma\tau\gamma[\Delta]$ is the identity.

The last statement of the lemma follows likewise. □

Proof of Lemma 8.3 If $\sigma : (\tau\gamma[\Delta], \delta|_{\tau\gamma[\Delta]}) \to (\mathcal{E}', \delta')$ then, by definition,

$$\left(\tau \cdot \left(\left[\!\left[\Gamma \vdash \overline{\textstyle\sum}_\Delta (x : A) .B : \mathsf{U}_\Delta \right]\!\right]_{(\mathcal{E}, \delta)} (\gamma)\right)\right)_\sigma = \left(\left[\!\left[\Gamma \vdash \overline{\textstyle\sum}_\Delta (x : A) .B : \mathsf{U}_\Delta \right]\!\right]_{(\mathcal{E}, \delta)} (\gamma)\right)_{\sigma\overline{\tau}}$$

which equals

$$\{(x,y) \mid x \in (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\sigma\overline{\tau}}, y \in (\llbracket \Gamma \vdash (x:A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\sigma\overline{\tau}}(x)\}.$$

By (13) and Lemma 8.4 this equals

$$\{(x,y) \mid x \in (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}',\delta')}(\tau\cdot\gamma))_{\sigma}, y \in (\llbracket \Gamma \vdash (x:A).B \rrbracket_{(\mathcal{E}',\delta')}(\tau\cdot\gamma))_{\sigma}(x)\}$$

which is just

$$\left(\left\llbracket \Gamma \vdash \overline{\textstyle\sum}_\Delta (x:A).B : \mathsf{U}_\Delta \right\rrbracket_{(\mathcal{E}',\delta')}(\tau\cdot\gamma)\right)_\sigma$$

Equality of the actions of the form $\rho\cdot(-)$ follow likewise.

By a similar calculation it follows that

$$\left(\tau\cdot\left(\left\llbracket \Gamma \vdash \overline{\textstyle\prod}_\Delta (x:A).B : \mathsf{U}_\Delta \right\rrbracket_{(\mathcal{E},\delta)}(\gamma)\right)\right)_\sigma$$

is the set of families $(f_\omega)_\omega$ indexed by maps $\omega : (\mathcal{E}',\delta') \to (\mathcal{E}'',\delta'')$ such that

$$f_\omega : (x : (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}',\delta')}(\tau\cdot\gamma))_{\omega\sigma}) \to (\llbracket \Gamma \vdash (x:A).B \rrbracket_{(\mathcal{E}',\delta')}(\tau\cdot\gamma))_{\omega\sigma}(x)$$

satisfying $f_{\rho\omega}(\rho\cdot x) = \rho\cdot f_\omega(x)$. This is precisely

$$\left(\left\llbracket \Gamma \vdash \overline{\textstyle\prod}_\Delta (x:A).B : \mathsf{U}_\Delta \right\rrbracket_{(\mathcal{E}',\delta')}(\tau\cdot\gamma)\right)_\sigma$$

as desired. □

It remains to show that the equalities of Figure 7 hold under this interpretation. The main work for showing this is contained in the next two lemmas.

Lemma 8.5. Suppose $\gamma \in \llbracket\Gamma\rrbracket_{(\mathcal{E},\delta)}$ and let $j : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\mathcal{E},\delta)$ be the inclusion and suppose that

$$x \in (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\gamma))_j = \llbracket \Gamma \vdash \mathsf{El}_\Delta(A)\ \mathsf{type} \rrbracket_{(\mathcal{E},\delta)}(\gamma)$$

Then

$$(\llbracket \Gamma \vdash (x:A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_j(x) = \llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash \mathsf{El}_\Delta(B)\ \mathsf{type} \rrbracket_{(\mathcal{E},\delta)}(\gamma, x)$$

and for any $\tau : (\mathcal{E},\delta) \to (\mathcal{E}',\delta')$ the map

$$\tau\cdot(-) : \llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash \mathsf{El}_\Delta(B)\ \mathsf{type} \rrbracket_{(\mathcal{E},\delta)}(\gamma, x) \to \llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash \mathsf{El}_\Delta(B)\ \mathsf{type} \rrbracket_{(\mathcal{E}',\delta')}(\tau\cdot\gamma, \tau\cdot x)$$

is the same as the map

$$\tau\cdot(-) : (\llbracket \Gamma \vdash (x:A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_j(x) \to (\llbracket \Gamma \vdash (x:A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau j}(\tau\cdot x)$$

of Lemma 8.2.

For the second statement of the lemma, to see that the maps have the same types, let $h : (\tau\gamma[\Delta], \delta|_{\tau\gamma[\Delta]}) \to (\mathcal{E}',\delta')$ be the inclusion and compute

$$\begin{aligned}
\llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash \mathsf{El}_\Delta(B)\ \mathsf{type} \rrbracket_{(\mathcal{E}',\delta')}(\tau\cdot\gamma, \tau\cdot x) &= (\llbracket \Gamma \vdash (x:A).B \rrbracket_{(\mathcal{E}',\delta')}(\tau\cdot\gamma))_h(\tau\cdot x) \\
&= (\llbracket \Gamma \vdash (x:A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{h\circ\overline{\tau}}(\tau\cdot x) \\
&= (\llbracket \Gamma \vdash (x:A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau\circ j}(\tau\cdot x)
\end{aligned}$$

where the second equality follows from Lemma 8.4.

Proof. By definition

$$(\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_j(x) = \llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}+\mathcal{E},[\delta,\delta])}((j+i)\cdot\gamma, \mathsf{inl}\cdot x)_h$$

where $h : (j+i)\gamma[\Delta] = \gamma[\Delta] \to \mathcal{E}+\mathcal{E}$ is the inclusion.

Consider the map

$$\phi = [\mathsf{inl}, \mathsf{inl}] : (\mathcal{E}+\mathcal{E}, [\delta,\delta]) \to (\mathcal{E}+\mathcal{E}, [\delta,\delta])$$

Note that

$$\phi \circ \mathsf{inl} = \mathsf{inl} \qquad\qquad \overline{\phi} = \mathrm{id}$$
$$\phi \circ (j+i) = \mathsf{inl}$$

Using these identities, we compute:

$$\begin{aligned}
(\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_j(x) &= (\llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}+\mathcal{E},[\delta,\delta])}((j+i)\cdot\gamma, \mathsf{inl}\cdot x))_{h\circ\overline{\phi}} \\
&= (\llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}+\mathcal{E},[\delta,\delta])}(\phi(j+i)\cdot\gamma, (\phi\circ\mathsf{inl})\cdot x))_h \\
&= (\llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}+\mathcal{E},[\delta,\delta])}(\mathsf{inl}\cdot\gamma, \mathsf{inl}\cdot x))_h \\
&= (\llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\gamma, x))_{h\circ\overline{\mathsf{inl}}} \\
&= (\llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\gamma, x))_h \\
&= \llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash \mathsf{El}_\Delta(B)\ \mathsf{type} \rrbracket_{(\mathcal{E},\delta)}(\gamma, x)\,.
\end{aligned}$$

The second statement of the lemma follows from the fact that both actions $\tau\cdot(-)$ derive from the action of $\llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket$. □

Lemma 8.6. Let $\gamma \in \llbracket \Gamma \vdash \rrbracket_{(\mathcal{E},\delta)}$, let $\Delta \subseteq \Delta'$ be sets of clocks in contexts Γ and let $j : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\gamma[\Delta'], \delta|_{\gamma[\Delta']})$ be the inclusion. If $\tau : (\gamma[\Delta'], \delta|_{\gamma[\Delta']}) \to (\mathcal{E}', \delta')$ and

$$x \in (\llbracket \Gamma \vdash \mathsf{in}_{\Delta,\Delta'}(A) : \mathsf{U}_{\Delta'} \rrbracket_{(\mathcal{E},\delta)}(\gamma))_\tau = (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau j}$$

then

$$(\llbracket \Gamma \vdash (x : \mathsf{in}_{\Delta,\Delta'}(A)).\, \mathsf{in}_{\Delta,\Delta'}(B) \rrbracket_{(\mathcal{E},\delta)}(\gamma))_\tau(x) = (\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau j}(x)$$

Proof. By definition $(\llbracket \Gamma \vdash (x : \mathsf{in}_{\Delta,\Delta'}(A)).\, \mathsf{in}_{\Delta,\Delta'}(B) \rrbracket_{(\mathcal{E},\delta)}(\gamma))_\tau(x)$ is

$$(\llbracket \Gamma, x : \mathsf{El}_{\Delta'}(\mathsf{in}_{\Delta,\Delta'}(A)) \vdash \mathsf{in}_{\Delta,\Delta'}(B) : \mathsf{U}_{\Delta'} \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\tau+i)\cdot\gamma, \mathsf{inl}\cdot x))_{h:(\tau\gamma[\Delta'],\delta'|_{\tau\gamma[\Delta']})\to(\mathcal{E}',\delta')}$$

where $i : \mathcal{E} \setminus \gamma[\Delta'] \to \mathcal{E}$ is the inclusion. Let $k : \tau\gamma[\Delta] \to \tau\gamma[\Delta']$ be the inclusion, then the above equals

$$(\llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\tau+i)\cdot\gamma, \mathsf{inl}\cdot x))_{hk} \tag{14}$$

This must be proved equal to $(\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau j}(x)$ which by definition is

$$(\llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\tau j+l)\cdot\gamma, \mathsf{inl}\cdot x))_{hk} \tag{15}$$

where $l : \mathcal{E} \setminus \gamma[\Delta] \to \mathcal{E}$ is the inclusion.

Define $\phi : (\mathcal{E}'+\mathcal{E}, [\delta',\delta]) \to (\mathcal{E}'+\mathcal{E}, [\delta',\delta])$ by

$$\phi(\mathsf{inl}(\lambda)) = \mathsf{inl}(\lambda)$$
$$\phi(\mathsf{inr}(\lambda)) = \begin{cases} \mathsf{inl}(\tau(\lambda)) & \text{if } \lambda \in \gamma[\Delta'] \setminus \gamma[\Delta] \\ \mathsf{inr}(\lambda) & \text{else} \end{cases}$$

and note that

$$\phi \circ (\tau j + l) = (\tau + i) \qquad\qquad \phi \circ \mathsf{inl} = \mathsf{inl}$$

The first of these can be proved by considering cases: If $\lambda \in \gamma[\Delta]$ then

$$\phi \circ (\tau j + l)(\lambda) = \phi(\mathsf{inl}(\tau(\lambda))) = \mathsf{inl}(\tau(\lambda)) = (\tau + i)(\lambda)$$

If $\lambda \in \gamma[\Delta'] \setminus \gamma[\Delta]$ then

$$\phi \circ (\tau j + l)(\lambda) = \phi(\mathsf{inr}(\lambda)) = \mathsf{inl}(\tau(\lambda)) = (\tau + i)(\lambda)$$

and if $\lambda \in \mathcal{E} \setminus \gamma[\Delta']$ then both sides equal $\mathsf{inr}(\lambda)$. Note also that the restriction/corestriction $\overline{\phi} : (\tau j + l)\gamma[\Delta] \to (\tau + i)\gamma[\Delta]$ is the identity on $\tau\gamma[\Delta]$.

Now, (14) equals

$$\begin{aligned}
&(\llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}\,(\phi \cdot (\tau j + l) \cdot \gamma, \phi \cdot \mathsf{inl} \cdot x))_{hk} \\
=&(\llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}\,((\tau j + l) \cdot \gamma, \mathsf{inl} \cdot x))_{hk\overline{\phi}} \\
=&(\llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}\,((\tau j + l) \cdot \gamma, \mathsf{inl} \cdot x))_{hk}
\end{aligned}$$

which is just (15). □

Proposition 8.1. All the equalities of Figure 7 hold in the model, i.e., if $\gamma \in \llbracket \Gamma \vdash \rrbracket_{(\mathcal{E},\delta)}$ then the following equalities hold:

$$\left(\left\llbracket \Gamma \vdash \mathsf{El}_\Delta(\overline{\textstyle\prod}_\Delta\,(x : A)\,.B)\ \mathsf{type} \right\rrbracket\right)_{(\mathcal{E},\delta)}(\gamma) = (\llbracket \Gamma \vdash \textstyle\prod\,(x : \mathsf{El}_\Delta(A))\,.\,\mathsf{El}_\Delta(B)\ \mathsf{type} \rrbracket)_{(\mathcal{E},\delta)}(\gamma)$$

$$\left(\left\llbracket \Gamma \vdash \mathsf{El}_\Delta(\overline{\textstyle\sum}_\Delta\,(x : A)\,.B)\ \mathsf{type} \right\rrbracket\right)_{(\mathcal{E},\delta)}(\gamma) = (\llbracket \Gamma \vdash \textstyle\sum\,(x : \mathsf{El}_\Delta(A))\,.\,\mathsf{El}_\Delta(B)\ \mathsf{type} \rrbracket)_{(\mathcal{E},\delta)}(\gamma)$$

$$\left(\left\llbracket \Gamma \vdash \overline{\textstyle\prod}_{\Delta'}\,(x : \mathsf{in}_{\Delta,\Delta'}(A))\,.\,\mathsf{in}_{\Delta,\Delta'}(B) : \mathsf{U}_{\Delta'} \right\rrbracket\right)_{(\mathcal{E},\delta)}(\gamma) = \left(\left\llbracket \Gamma \vdash \mathsf{in}_{\Delta,\Delta'}(\overline{\textstyle\prod}_\Delta\,(x : A)\,.B) : \mathsf{U}_{\Delta'} \right\rrbracket\right)_{(\mathcal{E},\delta)}(\gamma)$$

$$\left(\left\llbracket \Gamma \vdash \overline{\textstyle\sum}_{\Delta'}\,(x : \mathsf{in}_{\Delta,\Delta'}(A))\,.\,\mathsf{in}_{\Delta,\Delta'}(B) : \mathsf{U}_{\Delta'} \right\rrbracket\right)_{(\mathcal{E},\delta)}(\gamma) = \left(\left\llbracket \Gamma \vdash \mathsf{in}_{\Delta,\Delta'}(\overline{\textstyle\sum}_\Delta\,(x : A)\,.B) : \mathsf{U}_{\Delta'} \right\rrbracket\right)_{(\mathcal{E},\delta)}(\gamma)$$

In the first two cases, the equalities extend to equalities of presheaf actions on types.

Proof. By definition $\left(\left\llbracket \Gamma \vdash \mathsf{El}_\Delta(\overline{\sum}_\Delta\,(x : A)\,.B)\ \mathsf{type} \right\rrbracket\right)_{(\mathcal{E},\delta)}(\gamma)$ is

$$\{(x, y) \mid x \in (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}\,(\gamma))_j, y \in (\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E},\delta)}\,(\gamma))_j(x)\}$$

where $j : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\mathcal{E}, \delta)$ is the inclusion. By Lemma 8.5 this is

$$\{(x, y) \mid x \in \llbracket \Gamma \vdash \mathsf{El}_\Delta(A)\ \mathsf{type} \rrbracket_{(\mathcal{E},\delta)}\,(\gamma), y \in \llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash \mathsf{El}_\Delta(B)\ \mathsf{type} \rrbracket_{(\mathcal{E},\delta)}\,(\gamma, x)\}$$

which is precisely $(\llbracket \Gamma \vdash \prod\,(x : \mathsf{El}_\Delta\,A)\,.\,\mathsf{El}_\Delta\,B\ \mathsf{type} \rrbracket)_{(\mathcal{E},\delta)}\,(\gamma)$.

Likewise $\left(\left\llbracket \Gamma \vdash \mathsf{El}_\Delta(\overline{\prod}_\Delta\,(x : A)\,.B)\ \mathsf{type} \right\rrbracket\right)_{(\mathcal{E},\delta)}(\gamma)$ is by definition the set of families (f_τ) indexed by maps $\tau : (\mathcal{E}, \delta) \to (\mathcal{E}', \delta')$ such that

$$f_\tau : (x : (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}\,(\gamma))_{\tau j}) \to (\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E},\delta)}\,(\gamma))_{\tau j}(x) \tag{16}$$

for all τ and $f_{\rho\tau}(\rho \cdot x) = \rho \cdot f_\tau(x)$ for all x and ρ. Now,

$$\begin{aligned}(\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau j} &= (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{h\overline{\tau}} \\ &= (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}',\delta')}(\tau \cdot \gamma))_h \\ &= (\llbracket \Gamma \vdash \mathsf{El}_\Delta(A)\ \mathsf{type} \rrbracket_{(\mathcal{E}',\delta')}(\tau \cdot \gamma))\end{aligned}$$

where $h : (\tau\gamma[\Delta], \delta|_{\tau\gamma[\Delta]}) \to (\mathcal{E}, \delta)$ is the inclusion, and

$$\begin{aligned}(\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau j}(x) &= (\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{h\overline{\tau}}(x) \\ &= (\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E}',\delta')}(\tau \cdot \gamma))_h(x) && \text{(Lemma 8.4)} \\ &= \llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash \mathsf{El}_\Delta(B)\ \mathsf{type} \rrbracket_{(\mathcal{E}',\delta')}(\tau \cdot \gamma)(x) && \text{(Lemma 8.5)}\end{aligned}$$

So the type of each f_τ is

$$f_\tau : (x : \llbracket \Gamma \vdash \mathsf{El}_\Delta(A)\ \mathsf{type} \rrbracket_{(\mathcal{E}',\delta')}(\tau \cdot \gamma)) \to \llbracket \Gamma, x : \mathsf{El}_\Delta(A) \vdash \mathsf{El}_\Delta(B)\ \mathsf{type} \rrbracket_{(\mathcal{E}',\delta')}(\tau \cdot \gamma)(x) \quad (17)$$

and by the second statement of Lemma 8.5 the requirement of $f_{\rho\tau}(\rho \cdot x) = \rho \cdot f_\tau(x)$ for all x and ρ as formulated using the typing in (16) is equivalent to the one as formulated using the typing in (17). Thus we conclude that

$$\left(\left\llbracket \Gamma \vdash \mathsf{El}_\Delta(\overline{\prod}_\Delta\,(x : A)\,.B)\ \mathsf{type} \right\rrbracket\right)_{(\mathcal{E},\delta)}(\gamma) = (\llbracket \Gamma \vdash \textstyle\prod\,(x : \mathsf{El}_\Delta\,A)\,.\,\mathsf{El}_\Delta\,B\ \mathsf{type} \rrbracket)_{(\mathcal{E},\delta)}(\gamma)$$

By definition, if $\sigma : (\gamma[\Delta'], \delta|_{\gamma[\Delta']}) \to (\mathcal{E}', \delta')$ then

$$\left(\left(\left\llbracket \Gamma \vdash \overline{\textstyle\sum}_{\Delta'}\,(x : \mathsf{in}_{\Delta,\Delta'}(A))\,.\,\mathsf{in}_{\Delta,\Delta'}(B) : \mathsf{U}_{\Delta'} \right\rrbracket\right)_{(\mathcal{E},\delta)}(\gamma)\right)_\sigma$$

is

$$\{(x, y) \mid x \in (\llbracket \Gamma \vdash \mathsf{in}_{\Delta,\Delta'}(A) : \mathsf{U}_{\Delta'} \rrbracket_{(\mathcal{E},\delta)}(\gamma))_\sigma, y \in (\llbracket \Gamma \vdash (x : \mathsf{in}_{\Delta,\Delta'}(A)).\,\mathsf{in}_{\Delta,\Delta'}(B) \rrbracket_{(\mathcal{E},\delta)}(\gamma))_\sigma(x)\}$$

which by Lemma 8.6 equals

$$\{(x, y) \mid x \in (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\sigma j}, y \in (\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\sigma j}(x)\}$$

where $j : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\gamma[\Delta'], \delta|_{\gamma[\Delta']})$ is the inclusion. The latter equals

$$\left(\left(\left\llbracket \Gamma \vdash \overline{\textstyle\sum}_\Delta\,(x : A)\,.B : \mathsf{U}_\Delta \right\rrbracket\right)_{(\mathcal{E},\delta)}(\gamma)\right)_{\sigma j}$$

which by definition is

$$\left(\left(\left\llbracket \Gamma \vdash \mathsf{in}_{\Delta,\Delta'}(\overline{\textstyle\sum}_{\Delta'}\,(x : A)\,.B) : \mathsf{U}_{\Delta'} \right\rrbracket\right)_{(\mathcal{E},\delta)}(\gamma)\right)_\sigma$$

Likewise, by definition $\left(\left(\left\llbracket \Gamma \vdash \overline{\textstyle\prod}_{\Delta'}\,(x : \mathsf{in}_{\Delta,\Delta'}(A))\,.\,\mathsf{in}_{\Delta,\Delta'}(B) : \mathsf{U}_{\Delta'} \right\rrbracket\right)_{(\mathcal{E},\delta)}(\gamma)\right)_\sigma$ is the set of families (f_τ) indexed by maps $\tau : (\mathcal{E}', \delta') \to (\mathcal{E}'', \delta'')$ where each f_τ has type

$$f_\tau : (x : (\llbracket \Gamma \vdash \mathsf{in}_{\Delta,\Delta'}(A) : \mathsf{U}_{\Delta'} \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau\sigma}) \to (\llbracket \Gamma \vdash (x : \mathsf{in}_{\Delta,\Delta'}(A)).\,\mathsf{in}_{\Delta,\Delta'}(B) \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau\sigma}(x)$$

such that for all τ, ρ and x it holds that $f_{\rho\tau}(\rho \cdot x) = \rho \cdot f_\tau(x)$. Using Lemma 8.6 the type of f_τ can be rewritten to

$$f_\tau : (x : (\llbracket \Gamma \vdash A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau\sigma j}) \to (\llbracket \Gamma \vdash (x : A).B \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau\sigma j}(x)$$

and thus we can conclude that

$$\begin{aligned}
&\left(\left(\left[\!\left[\Gamma \vdash \overline{\prod}_{\Delta'}\left(x : \mathsf{in}_{\Delta,\Delta'}(A)\right).\mathsf{in}_{\Delta,\Delta'}(B) : \mathsf{U}_{\Delta'}\right]\!\right]\right)_{(\mathcal{E},\delta)}(\gamma)\right)_{\sigma} \\
&= \left(\left(\left[\!\left[\Gamma \vdash \overline{\prod}_{\Delta}\left(x : A\right).B : \mathsf{U}_{\Delta}\right]\!\right]\right)_{(\mathcal{E},\delta)}(\gamma)\right)_{\sigma j} \\
&= \left(\left(\left[\!\left[\Gamma \vdash \mathsf{in}_{\Delta,\Delta'}(\overline{\prod}_{\Delta}\left(x : A\right).B) : \mathsf{U}_{\Delta'}\right]\!\right]\right)_{(\mathcal{E},\delta)}(\gamma)\right)_{\sigma}
\end{aligned}$$

□

8.1.1. *Substitution lemma* Finally we state the needed substitution lemma.

Lemma 8.7. Suppose $\Gamma' \vdash A : \mathsf{U}_\Delta$ and $\Gamma', x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta$, let $\sigma : \Gamma \to \Gamma'$ be a substitution and assume that x is not in Γ. Let $\gamma \in [\![\Gamma \vdash]\!]_{(\mathcal{E},\delta)}$ and suppose that for all $(\mathcal{E}', \delta')$ and $\gamma' \in [\![\Gamma \vdash]\!]_{(\mathcal{E}',\delta')}$

$$[\![\Gamma \vdash A\sigma : (\mathsf{U}_\Delta)\sigma]\!]_{(\mathcal{E}',\delta')}(\gamma') = [\![\Gamma' \vdash A : \mathsf{U}_\Delta]\!]_{(\mathcal{E}',\delta')}([\![\sigma]\!]_{(\mathcal{E}',\delta')}(\gamma'))$$

and for all $z \in [\![\Gamma \vdash (\mathsf{El}_\Delta\, A)\sigma\ \mathsf{type}]\!]_{(\mathcal{E}',\delta')}(\gamma')$

$$[\![\Gamma, x : (\mathsf{El}_\Delta\, A)\sigma \vdash B\sigma : (\mathsf{U}_\Delta)\sigma]\!]_{(\mathcal{E}',\delta')}(\gamma', z) = [\![\Gamma', x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta]\!]_{(\mathcal{E}',\delta')}([\![\sigma]\!]_{(\mathcal{E}',\delta')}(\gamma'), z)$$

Then

$$\left[\!\left[\Gamma \vdash (\overline{\sum}_\Delta\left(x : A\right).B)\sigma : (\mathsf{U}_\Delta)\sigma\right]\!\right]_{(\mathcal{E},\delta)}(\gamma) = \left[\!\left[\Gamma' \vdash \overline{\sum}_\Delta\left(x : A\right).B : \mathsf{U}_\Delta\right]\!\right]_{(\mathcal{E},\delta)}([\![\sigma]\!]_{(\mathcal{E},\delta)}\,\gamma)$$

$$\left[\!\left[\Gamma \vdash (\overline{\prod}_\Delta\left(x : A\right).B)\sigma : (\mathsf{U}_\Delta)\sigma\right]\!\right]_{(\mathcal{E},\delta)}(\gamma) = \left[\!\left[\Gamma' \vdash \overline{\prod}_\Delta\left(x : A\right).B : \mathsf{U}_\Delta\right]\!\right]_{(\mathcal{E},\delta)}([\![\sigma]\!]_{(\mathcal{E},\delta)}\,\gamma)$$

Proof. If $\Delta = \kappa_1, \ldots, \kappa_n$, then $(\mathsf{U}_\Delta)\sigma = \mathsf{U}_{\sigma[\Delta]}$ where $\sigma[\Delta] = \sigma(\kappa_1), \ldots, \sigma(\kappa_n)$. By definition, if $\tau : (\gamma\sigma[\Delta], \delta|_{\gamma\sigma[\Delta]}) \to (\mathcal{E}', \delta')$ then

$$\left(\left[\!\left[\Gamma \vdash (\overline{\sum}_\Delta\left(x : A\right).B)\sigma : (\mathsf{U}_\Delta)\sigma\right]\!\right]_{(\mathcal{E},\delta)}(\gamma)\right)_{\tau}$$

is the set of pairs (z, w) where

$$\begin{aligned}
z &\in \left([\![\Gamma \vdash A\sigma : (\mathsf{U}_\Delta)\sigma]\!]_{(\mathcal{E},\delta)}(\gamma)\right)_{\tau} \\
w &\in \left([\![\Gamma \vdash (x : A\sigma).B\sigma]\!]_{(\mathcal{E},\delta)}(\gamma)\right)_{\tau}(z)
\end{aligned}$$

By assumption, the type of z equals

$$([\![\Gamma' \vdash A : \mathsf{U}_\Delta]\!]_{(\mathcal{E},\delta)}([\![\sigma]\!]_{(\mathcal{E},\delta)}(\gamma)))_\tau$$

and the type of w is

$$\begin{aligned}
&([\![\Gamma, x : (\mathsf{El}_\Delta(A))\sigma \vdash B\sigma : (\mathsf{U}_\Delta)\sigma]\!]_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\tau + i)\cdot\gamma, \mathsf{inl}\cdot z))_j \\
&= ([\![\Gamma', x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta]\!]_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}([\![\sigma]\!]_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\tau + i)\cdot\gamma), \mathsf{inl}\cdot z))_j \\
&= ([\![\Gamma', x : \mathsf{El}_\Delta(A) \vdash B : \mathsf{U}_\Delta]\!]_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\tau + i)\cdot[\![\sigma]\!]_{(\mathcal{E},\delta)}(\gamma), \mathsf{inl}\cdot z))_j \\
&= \left([\![\Gamma' \vdash (x : A).B]\!]_{(\mathcal{E},\delta)}([\![\sigma]\!]_{(\mathcal{E},\delta)}(\gamma))\right)_\tau(z)
\end{aligned}$$

and thus the (z, w) are exactly the elements of

$$\left(\left[\!\left[\Gamma' \vdash \overline{\sum}_\Delta (x : A) . B : \mathsf{U}_\Delta\right]\!\right]_{(\mathcal{E},\delta)} (\llbracket\sigma\rrbracket_{(\mathcal{E},\delta)}\, \gamma)\right)_\tau$$

proving the first equality.

For the second equality, the elements of

$$\left(\left[\!\left[\Gamma \vdash (\overline{\prod}_\Delta (x : A) . B)\sigma : (\mathsf{U}_\Delta)\sigma\right]\!\right]_{(\mathcal{E},\delta)} (\gamma)\right)_\tau$$

are the families $(f_\omega)_\omega$ indexed over maps $\omega : (\mathcal{E}', \delta') \to (\mathcal{E}'', \delta'')$ of maps with type

$$f_\omega : \left(z : \llbracket\Gamma \vdash A\sigma : (\mathsf{U}_\Delta)\sigma\rrbracket_{(\mathcal{E},\delta)} (\gamma)\right)_{\omega\tau} \to \left(\llbracket\Gamma \vdash (x : A\sigma).B\sigma\rrbracket_{(\mathcal{E},\delta)} (\gamma)\right)_{\omega\tau} (z)$$

satisfying $f_{\rho\omega}(\rho \cdot z) = \rho \cdot f_\omega(z)$ for all ω, ρ and z. By the calculations above, we can rewrite the type of f_ω to

$$f_\omega : (z : \llbracket\Gamma' \vdash A : \mathsf{U}_\Delta\rrbracket_{(\mathcal{E},\delta)} (\llbracket\sigma\rrbracket_{(\mathcal{E},\delta)} (\gamma)))_{\omega\tau} \to \left(\llbracket\Gamma' \vdash (x : A).B\rrbracket_{(\mathcal{E},\delta)} (\llbracket\sigma\rrbracket_{(\mathcal{E},\delta)} (\gamma))\right)_{\omega\tau} (z)$$

and thus we see that the families $(f_\omega)_\omega$ are exactly the elements of

$$\left(\left[\!\left[\Gamma' \vdash \overline{\prod}_\Delta (x : A) . B : \mathsf{U}_\Delta\right]\!\right]_{(\mathcal{E},\delta)} (\llbracket\sigma\rrbracket_{(\mathcal{E},\delta)}\, \gamma)\right)_\tau$$

proving the second equality. □

8.2. *Universal quantification over clocks*

We now turn to the code for universal quantification over clocks. Figure 9 presents the typing and equality rules for these.

Formation and typing rules

$$\frac{\Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \qquad \kappa \notin \Delta}{\Gamma \vdash \overline{\forall}\kappa.A : \mathsf{U}_\Delta}$$

Equations

$$\begin{aligned} \mathsf{El}_\Delta(\overline{\forall}\kappa.A) &\equiv \forall\kappa.\, \mathsf{El}_{\Delta,\kappa}(A) \\ \mathsf{in}_{\Delta,\Delta'}\,(\overline{\forall}\kappa.A) &\equiv \overline{\forall}\kappa.\, \mathsf{in}_{(\Delta,\kappa),(\Delta',\kappa)}\,(A) \end{aligned}$$

Figure 9. Syntax for codes for universal quantification over clocks. The second equality has the side condition that $\kappa \notin \Delta'$

Since universal quantification over clocks is just a dependent product over $\mathcal{C}$, the interpretation of the code for this is very similar to the interpretation of the code for dependent product, and the proofs presented in this section are very similar to the ones presented in Section 8.1. However, we cannot view $\overline{\forall}\kappa.A$ directly as a special case of $\overline{\prod}_\Delta (x : A) . B$ for two reasons: The shift of universe from $\mathsf{U}_{\Delta,\kappa}$ to U_Δ, and because $\mathcal{C}$ is not invariant under clock introduction, so not an element of any of the universes. The latter problem is not a serious one, since we have already

seen that the types invariant under clock introduction are closed under universal quantification over clocks.

Lemma 8.8. Suppose $\Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa}$, and $\gamma \in [\![\Gamma]\!]_{(\mathcal{E},\delta)}$. The family of sets

$$([\![\Gamma \vdash (\kappa : \mathsf{clock}).A]\!]_{(\mathcal{E},\delta)}(\gamma))_\sigma(\lambda)$$

indexed over $\sigma : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\mathcal{E}', \delta')$ and $\lambda \in \mathcal{E}'$, which is defined as

$$[\![\Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa}]\!]_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\sigma + i)\cdot\gamma, \lambda)_{j:((\sigma\gamma[\Delta],\lambda),\delta'|_{\sigma\gamma[\Delta],\lambda})\to(\mathcal{E}',\delta')}$$

(where j is the inclusion) extends to a small dependent type over $\mathcal{C}$ considered as a small dependent type over $y(\gamma[\Delta], \delta|_{\gamma[\Delta]})$, i.e., a small dependent type in context $x : y(\gamma[\Delta], \delta|_{\gamma[\Delta]}), \kappa : \mathcal{C} \vdash$. Moreover, this family is invariant under clock introduction.

We have written $(\sigma\gamma[\Delta], \lambda)$ for the union of $\sigma\gamma[\Delta]$ and the singleton $\{\lambda\}$, *which need not be disjoint.*

Proof. If σ is as above and $\tau : (\mathcal{E}', \delta') \to (\mathcal{E}'', \delta'')$ we must define

$$\tau \cdot (-) : ([\![\Gamma \vdash (\kappa : \mathsf{clock}).A]\!]_{(\mathcal{E},\delta)}(\gamma))_\sigma(\lambda) \to ([\![\Gamma \vdash (\kappa : \mathsf{clock}).A]\!]_{(\mathcal{E},\delta)}(\gamma))_{\tau\sigma}(\tau(\lambda))\,. \tag{18}$$

First note that

$$\begin{aligned}
&([\![\Gamma \vdash (\kappa : \mathsf{clock}).A]\!]_{(\mathcal{E},\delta)}(\gamma))_{\tau\sigma}(\tau(\lambda))\\
=&([\![\Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa}]\!]_{(\mathcal{E}''+\mathcal{E},[\delta'',\delta])}((\tau\sigma + i)\cdot\gamma, \tau(\lambda)))_k\\
=&([\![\Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa}]\!]_{(\mathcal{E}''+\mathcal{E},[\delta'',\delta])}((\tau + \mathrm{id})(\sigma + i)\cdot\gamma, (\tau + \mathrm{id})(\lambda)))_k
\end{aligned}$$

where $k : ((\tau\sigma\gamma[\Delta], \tau(\lambda)), \delta''|_{\tau\sigma\gamma[\Delta],\tau(\lambda)}) \to (\mathcal{E}'', \delta'')$ is the inclusion. Since $(\sigma + i)\gamma[\Delta] = \sigma\gamma[\Delta]$ and $\overline{(\tau + \mathrm{id})}^{(\sigma\gamma[\Delta],\lambda)} = \overline{\tau}^{(\sigma\gamma[\Delta],\lambda)}$, by naturality of the interpretation of A we get

$$\begin{aligned}
&([\![\Gamma \vdash (\kappa : \mathsf{clock}).A]\!]_{(\mathcal{E},\delta)}(\gamma))_{\tau\sigma}(\tau(\lambda))\\
=&([\![\Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa}]\!]_{(\mathcal{E}''+\mathcal{E},[\delta'',\delta])}((\sigma + i)\cdot\gamma, \lambda))_{k\circ\overline{\tau}^{(\sigma\gamma[\Delta],\lambda)}}\\
=&([\![\Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa}]\!]_{(\mathcal{E}''+\mathcal{E},[\delta'',\delta])}((\sigma + i)\cdot\gamma, \lambda))_{\tau j}
\end{aligned}$$

where the last equality uses

$$\begin{array}{ccc}
\sigma\gamma[\Delta], \lambda & \xrightarrow{\overline{\tau}^{(\sigma\gamma[\Delta],\lambda)}} & \tau\sigma\gamma[\Delta], \tau(\lambda)\\
\Big\downarrow{\scriptstyle j} & & \Big\downarrow{\scriptstyle k}\\
\mathcal{E}' & \xrightarrow{\quad\tau\quad} & \mathcal{E}''
\end{array}$$

We can therefore define the action (18) using the structure coming from

$$([\![\Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa}]\!]_{(\mathcal{E}''+\mathcal{E},[\delta'',\delta])}((\sigma + i)\cdot\gamma, \lambda))$$

being an element in $\mathcal{U}^{\Delta,\kappa}((\gamma|_\Delta)[\kappa \mapsto \lambda])$. The rest of the proof now follows as in the proof of Lemma 8.2. □

We can therefore define $[\![\Gamma \vdash \overline{\forall}\kappa.A : \mathsf{U}_\Delta]\!]_{(\mathcal{E},\delta)}(\gamma)$ to be the dependent product of the type $([\![\Gamma \vdash (\kappa : \mathsf{clock}).A]\!]_{(\mathcal{E},\delta)}(\gamma))$ over $\mathcal{C}$. This gives a small dependent type over $y(\gamma[\Delta], \delta|_{\gamma[\Delta]})$ which is

$$\Big(\big[\!\big[\Gamma \vdash \overline{\forall}\kappa.A : \mathsf{U}_\Delta\big]\!\big]_{(\mathcal{E},\delta)}(\gamma)\Big)_\sigma = \{(f_\tau)_{\tau:(\mathcal{E}',\delta')\to(\mathcal{E}'',\delta'')} \mid \forall\tau.f_\tau : (\lambda : \mathcal{E}'') \to (\big[\!\big[\Gamma \vdash (\kappa : \mathsf{clock}).A\big]\!\big]_{(\mathcal{E},\delta)}(\gamma))_{\tau\sigma}(\lambda), \forall\rho,\lambda.f_{\rho\tau}(\rho(\lambda)) = \rho \cdot f_\tau(\lambda)\}$$
$$(\tau \cdot f)_\rho = f_{\rho\tau}$$

Figure 10. Interpretation of codes for universal quantification over clocks. The function σ is assumed to have type $\sigma : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\mathcal{E}', \delta')$ for some $(\mathcal{E}', \delta')$.

invariant under clock introduction by Proposition 4.1. The resulting interpretation is summarised Figure 10.

For this to be a well-defined interpretation of terms, we must show that it is natural with respect to the restriction maps as stated in the following lemma.

Lemma 8.9. If $\gamma \in \big[\!\big[\Gamma\big]\!\big]_{(\mathcal{E},\delta)}$ and $\tau : (\mathcal{E}, \delta) \to (\mathcal{E}', \delta')$ then

$$\big[\!\big[\Gamma \vdash \overline{\forall}\kappa.A : \mathsf{U}_\Delta\big]\!\big]_{(\mathcal{E}',\delta')}(\tau \cdot \gamma) = \tau \cdot \Big(\big[\!\big[\Gamma \vdash \overline{\forall}\kappa.A : \mathsf{U}_\Delta\big]\!\big]_{(\mathcal{E},\delta)}(\gamma)\Big)$$

For the proof of Lemma 8.9 we need the following lemma.

Lemma 8.10. Suppose $\gamma \in \big[\!\big[\Gamma\big]\!\big]_{(\mathcal{E},\delta)}$, $\tau : (\mathcal{E}, \delta) \to (\mathcal{E}', \delta')$ and $\sigma : (\tau\gamma[\Delta], \delta'|_{\tau\gamma[\Delta]}) \to (\mathcal{E}'', \delta'')$. If $\lambda \in \mathcal{E}''$ then

$$(\big[\!\big[\Gamma \vdash (\kappa : \mathsf{clock}).A\big]\!\big]_{(\mathcal{E}',\delta')}(\tau \cdot \gamma))_\sigma(\lambda) = (\big[\!\big[\Gamma \vdash (\kappa : \mathsf{clock}).A\big]\!\big]_{(\mathcal{E},\delta)}(\gamma))_{\sigma\overline{\tau}}(\lambda)$$

where $\overline{\tau} = \overline{\tau}^{\gamma[\Delta]} : \gamma[\Delta] \to \tau\gamma[\Delta]$ is the restriction/corestriction of τ. Moreover, for any $\omega : (\mathcal{E}'', \delta'') \to (\mathcal{E}''', \delta''')$ the maps

$$\omega \cdot (-) : (\big[\!\big[\Gamma \vdash (\kappa : \mathsf{clock}).A\big]\!\big]_{(\mathcal{E}',\delta')}(\tau \cdot \gamma))_\sigma(\lambda) \to (\big[\!\big[\Gamma \vdash (\kappa : \mathsf{clock}).A\big]\!\big]_{(\mathcal{E}',\delta')}(\tau \cdot \gamma))_{\omega\sigma}(\omega(\lambda))$$

and

$$\omega \cdot (-) : (\big[\!\big[\Gamma \vdash (\kappa : \mathsf{clock}).A\big]\!\big]_{(\mathcal{E},\delta)}(\gamma))_{\sigma\overline{\tau}}(\lambda) \to (\big[\!\big[\Gamma \vdash (\kappa : \mathsf{clock}).A\big]\!\big]_{(\mathcal{E},\delta)}(\gamma))_{\omega\sigma\overline{\tau}}(\omega(\lambda))$$

are equal.

Proof. By definition

$$(\big[\!\big[\Gamma \vdash (\kappa : \mathsf{clock}).A\big]\!\big]_{(\mathcal{E}',\delta')}(\tau \cdot \gamma))_\sigma(\lambda) = \big[\!\big[\Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa}\big]\!\big]_{(\mathcal{E}''+\mathcal{E}',[\delta'',\delta'])}((\sigma + j) \cdot \tau \cdot \gamma, \lambda)_h$$

where $h : ((\sigma\tau\gamma[\Delta], \lambda), \delta''|_{(\sigma\tau\gamma[\Delta],\lambda)}) \to (\mathcal{E}'', \delta'')$ and $j : (\mathcal{E}' \setminus \tau\gamma[\Delta]) \to \mathcal{E}'$ are inclusions.

As in the proof of Lemma 8.4 we define the map

$$\phi : ((\mathcal{E}'' + \mathcal{E}), [\delta'', \delta]) \to ((\mathcal{E}'' + \mathcal{E}'), [\delta'', \delta'])$$

as

$$\phi(\mathsf{inl}(\lambda')) = \mathsf{inl}(\lambda')$$
$$\phi(\mathsf{inr}(\lambda')) = \begin{cases} \mathsf{inl}(\sigma\tau(\lambda')) & \text{if } \tau(\lambda') \in \tau\gamma[\Delta] \\ \mathsf{inr}(\tau(\lambda')) & \text{if } \tau(\lambda') \notin \tau\gamma[\Delta] \end{cases}$$

satisfying $\phi(\sigma\overline{\tau}+i) = (\sigma+j)\phi$ and $\phi(\lambda) = \lambda$. By naturality of the interpretation of A we get

$$\begin{aligned}
&(\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E}',\delta')} (\tau \cdot \gamma))_\sigma(\lambda) \\
=&\llbracket \Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E}''+\mathcal{E}',[\delta'',\delta'])} (\phi \cdot (\sigma\overline{\tau}+i) \cdot \gamma, \phi(\lambda))_h \\
=&\llbracket \Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E}''+\mathcal{E},[\delta'',\delta])} ((\sigma\overline{\tau}+i) \cdot \gamma, \lambda)_{h \circ \overline{\phi}} \\
=&\llbracket \Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E}''+\mathcal{E},[\delta'',\delta])} ((\sigma\overline{\tau}+i) \cdot \gamma, \lambda)_h \\
=&(\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)} (\gamma))_{\sigma\overline{\tau}}(\lambda)
\end{aligned}$$

where the third equality uses that the restriction/corestriction $\overline{\phi} : ((\sigma\overline{\tau}+i)\gamma[\Delta], \lambda) \to (\sigma\tau\gamma[\Delta], \lambda)$ is the identity.

The last statement of the lemma follows likewise. □

Proof of Lemma 8.9 If $\sigma : (\tau\gamma[\Delta], \delta|_{\tau\gamma[\Delta]}) \to (\mathcal{E}', \delta')$ then, by definition,

$$\left(\tau \cdot \left(\llbracket \Gamma \vdash \overline{\forall}\kappa.A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)} (\gamma)\right)\right)_\sigma = \left(\llbracket \Gamma \vdash \overline{\forall}\kappa.A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)} (\gamma)\right)_{\sigma\overline{\tau}}$$

which is the set of families $(f_\omega)_\omega$ indexed by maps $\omega : (\mathcal{E}', \delta') \to (\mathcal{E}'', \delta'')$ such that

$$f_\omega : (\lambda : \mathcal{E}'') \to (\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)} (\gamma))_{\omega\sigma\overline{\tau}}(\lambda)$$

satisfying $f_{\rho\omega}(\rho(\lambda)) = \rho \cdot f_\omega(\lambda)$. By Lemma 8.10, the type of f_ω can be equally expressed as

$$f_\omega : (\lambda : \mathcal{E}'') \to (\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E}',\delta')} (\tau \cdot \gamma))_{\omega\sigma}(\lambda)$$

and so the families of $(f_\omega)_\omega$ are precisely those of

$$\left(\llbracket \Gamma \vdash \overline{\forall}\kappa.A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E}',\delta')} (\tau \cdot \gamma)\right)_\sigma$$

as desired. □

It remains to show that the equalities of Figure 9 hold under this interpretation. The main work for showing this is contained in the next two lemmas.

Lemma 8.11. Suppose $\gamma \in \llbracket \Gamma \rrbracket_{(\mathcal{E},\delta)}$ let $j : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\mathcal{E}, \delta)$ be the inclusion and suppose that $\lambda \in \mathcal{E}$. Then

$$(\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)} (\gamma))_j(\lambda) = \llbracket \Gamma, \kappa : \mathsf{clock} \vdash \mathsf{El}_{\Delta,\kappa}(A)\ \mathsf{type} \rrbracket_{(\mathcal{E},\delta)} (\gamma, \lambda)$$

and for any $\tau : (\mathcal{E}, \delta) \to (\mathcal{E}', \delta')$ the map

$$\tau \cdot (-) : \llbracket \Gamma, \kappa : \mathsf{clock} \vdash \mathsf{El}_{\Delta,\kappa}(A)\ \mathsf{type} \rrbracket_{(\mathcal{E},\delta)} (\gamma, \lambda) \to \llbracket \Gamma, \kappa : \mathsf{clock} \vdash \mathsf{El}_{\Delta,\kappa}(A)\ \mathsf{type} \rrbracket_{(\mathcal{E}',\delta')} (\tau \cdot \gamma, \tau(\lambda))$$

is the same as the map

$$\tau \cdot (-) : (\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)} (\gamma))_j(\lambda) \to (\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)} (\gamma))_{\tau j}(\tau(\lambda))$$

of Lemma 8.8.

For the second statement of the lemma, to see that the maps have the same types, let h :

$(\tau\gamma[\Delta], \delta|_{\tau\gamma[\Delta]}) \to (\mathcal{E}', \delta')$ be the inclusion and compute

$$\begin{aligned}\llbracket \Gamma, \kappa : \mathsf{clock} \vdash \mathsf{El}_{\Delta,\kappa}(A)\ \mathsf{type} \rrbracket_{(\mathcal{E}',\delta')}(\tau \cdot \gamma, \tau(\lambda)) &= (\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E}',\delta')}(\tau \cdot \gamma))_h(\tau(\lambda)) \\ &= (\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{h \circ \overline{\tau}}(\tau(\lambda)) \\ &= (\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau \circ j}(\tau(\lambda))\end{aligned}$$

where the second equality follows from Lemma 8.10.

Proof. By definition

$$(\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)}(\gamma))_j(\lambda) = \llbracket \Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E}+\mathcal{E},[\delta,\delta])}((j+i) \cdot \gamma, \lambda)_h$$

where $h : ((j+i)\gamma[\Delta], \lambda) = (\gamma[\Delta], \lambda) \to \mathcal{E} + \mathcal{E}$ is the inclusion.

As in the proof of Lemma 8.5 we consider the map

$$\phi = [\mathsf{inl}, \mathsf{inl}] : (\mathcal{E} + \mathcal{E}, [\delta, \delta]) \to (\mathcal{E} + \mathcal{E}, [\delta, \delta])$$

and note that

$$\begin{aligned}\phi(\lambda) &= \mathsf{inl}(\lambda) & \overline{\phi}^{(\gamma[\Delta],\lambda)} &= \mathrm{id} \\ \phi \circ (j+i) &= \mathsf{inl}\end{aligned}$$

Using these identities, we compute:

$$\begin{aligned}(\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)}(\gamma))_j(\lambda) &= (\llbracket \Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E}+\mathcal{E},[\delta,\delta])}((j+i) \cdot \gamma, \lambda))_{h \circ \overline{\phi}} \\ &= (\llbracket \Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E}+\mathcal{E},[\delta,\delta])}(\phi(j+i) \cdot \gamma, \phi(\lambda)))_h \\ &= (\llbracket \Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E}+\mathcal{E},[\delta,\delta])}(\mathsf{inl} \cdot \gamma, \mathsf{inl}(\lambda)))_h \\ &= (\llbracket \Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E},\delta)}(\gamma, \lambda))_{h \circ \overline{\mathsf{inl}}} \\ &= (\llbracket \Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E},\delta)}(\gamma, \lambda))_h \\ &= \llbracket \Gamma, \kappa : \mathsf{clock} \vdash \mathsf{El}_{\Delta,\kappa}(A)\ \mathsf{type} \rrbracket_{(\mathcal{E},\delta)}(\gamma, \lambda)\,.\end{aligned}$$

The second statement of the lemma follows from the fact that both actions $\tau \cdot (-)$ derive from the action of $\llbracket \Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket$. □

Lemma 8.12. Let $\Delta \subseteq \Delta'$ be sets of clocks in contexts Γ and suppose $\Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa}$. Let $\gamma \in \llbracket \Gamma \vdash \rrbracket_{(\mathcal{E},\delta)}$, and $j : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\gamma[\Delta'], \delta|_{\gamma[\Delta']})$ be the inclusion. If $\tau : (\gamma[\Delta'], \delta|_{\gamma[\Delta']}) \to (\mathcal{E}', \delta')$ and $\lambda' \in \mathcal{E}'$ then

$$(\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).\, \mathsf{in}_{(\Delta,\kappa),(\Delta',\kappa)}(A) \rrbracket_{(\mathcal{E},\delta)}(\gamma))_\tau(\lambda') = (\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau j}(\lambda')$$

Proof. By definition $(\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).\, \mathsf{in}_{(\Delta,\kappa),(\Delta',\kappa)}(A) \rrbracket_{(\mathcal{E},\delta)}(\gamma))_\tau(\lambda')$ is

$$(\llbracket \Gamma, \kappa : \mathsf{clock} \vdash \mathsf{in}_{(\Delta,\kappa),(\Delta',\kappa)}(A) : \mathsf{U}_{\Delta',\kappa} \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\tau + i) \cdot \gamma, \lambda'))_{h:((\tau\gamma[\Delta'],\lambda'),\delta'|_{(\tau\gamma[\Delta'],\lambda')}) \to (\mathcal{E}',\delta')}$$

where $i : \mathcal{E} \setminus \gamma[\Delta'] \to \mathcal{E}$ is the inclusion. Let $k : (\tau\gamma[\Delta], \lambda') \to (\tau\gamma[\Delta'], \lambda')$ be the inclusion, then the above equals

$$(\llbracket \Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\tau + i) \cdot \gamma, \lambda'))_{hk} \tag{19}$$

This must be proved equal to $(\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau j}(\lambda')$ which by definition is

$$(\llbracket \Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\tau j + l) \cdot \gamma, \lambda'))_{hk} \tag{20}$$

where $l : \mathcal{E} \setminus \gamma[\Delta] \to \mathcal{E}$ is the inclusion.

Define $\phi : (\mathcal{E}' + \mathcal{E}, [\delta', \delta]) \to (\mathcal{E}' + \mathcal{E}, [\delta', \delta])$ as in the proof of Lemma 8.6 by

$$\phi(\mathsf{inl}(\lambda)) = \mathsf{inl}(\lambda)$$
$$\phi(\mathsf{inr}(\lambda)) = \begin{cases} \mathsf{inl}(\tau(\lambda)) & \text{if } \lambda \in \gamma[\Delta'] \setminus \gamma[\Delta] \\ \mathsf{inr}(\lambda) & \text{else} \end{cases}$$

and note that $\phi \circ (\tau j + l) = (\tau + i)$ (see proof of Lemma 8.6) and $\phi(\lambda') = \lambda'$.

Now, by naturality of the interpretation of A, (19) equals

$$\begin{aligned} &(\llbracket \Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])} (\phi \cdot (\tau j + l) \cdot \gamma, \phi(\lambda')))_{hk} \\ =&(\llbracket \Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])} ((\tau j + l) \cdot \gamma, \lambda'))_{hk\overline{\phi}} \\ =&(\llbracket \Gamma, \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])} ((\tau j + l) \cdot \gamma, \lambda'))_{hk} \end{aligned}$$

which is just (20). □

Proposition 8.2. All the equalities of Figure 9 hold in the model, i.e., if $\gamma \in \llbracket \Gamma \vdash \rrbracket_{(\mathcal{E},\delta)}$ then the following equalities hold:

$$\left(\llbracket \Gamma \vdash \mathsf{El}_\Delta(\overline{\forall}\kappa.A)\ \mathsf{type} \rrbracket\right)_{(\mathcal{E},\delta)} (\gamma) = (\llbracket \Gamma \vdash \forall\kappa.\, \mathsf{El}_{\Delta,\kappa}(A)\ \mathsf{type} \rrbracket)_{(\mathcal{E},\delta)} (\gamma)$$
$$\left(\llbracket \Gamma \vdash \mathsf{in}_{\Delta,\Delta'} (\overline{\forall}\kappa.A) : \mathsf{U}_{\Delta'} \rrbracket\right)_{(\mathcal{E},\delta)} (\gamma) = \left(\llbracket \Gamma \vdash \overline{\forall}\kappa.\, \mathsf{in}_{(\Delta,\kappa),(\Delta',\kappa)} (A) : \mathsf{U}_{\Delta'} \rrbracket\right)_{(\mathcal{E},\delta)} (\gamma)$$

In the first case, the equality extends to an equality of the presheaf actions on types.

Proof. By definition $\left(\llbracket \Gamma \vdash \mathsf{El}_\Delta(\overline{\forall}\kappa.A)\ \mathsf{type} \rrbracket\right)_{(\mathcal{E},\delta)} (\gamma)$ is the set of families (f_τ) indexed by maps $\tau : (\mathcal{E}, \delta) \to (\mathcal{E}', \delta')$ such that

$$f_\tau : (\lambda' : \mathcal{E}') \to (\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)} (\gamma))_{\tau j}(\lambda') \tag{21}$$

for all τ (where $j : (\gamma[\Delta], \delta|_{\gamma[\Delta]}) \to (\mathcal{E}, \delta)$ is the inclusion) and $f_{\rho\tau}(\rho(\lambda')) = \rho \cdot f_\tau(\lambda')$ for all λ' and ρ. Now,

$$\begin{aligned} &(\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)} (\gamma))_{\tau j}(\lambda') \\ =&(\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)} (\gamma))_{h\overline{\tau}}(\lambda') \\ =&(\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E}',\delta')} (\tau \cdot \gamma))_h(\lambda') && \text{(Lemma 8.10)} \\ =&\llbracket \Gamma, \kappa : \mathsf{clock} \vdash \mathsf{El}_{\Delta,\kappa}(A)\ \mathsf{type} \rrbracket_{(\mathcal{E}',\delta')} (\tau \cdot \gamma, \lambda') && \text{(Lemma 8.11)} \end{aligned}$$

where $h : (\tau\gamma[\Delta], \delta|_{\tau\gamma[\Delta]}) \to (\mathcal{E}, \delta)$ is the inclusion.

So the type of each f_τ must be

$$f_\tau : (\lambda' : \mathcal{E}') \to \llbracket \Gamma, \kappa : \mathsf{clock} \vdash \mathsf{El}_{\Delta,\kappa}(A)\ \mathsf{type} \rrbracket_{(\mathcal{E}',\delta')} (\tau \cdot \gamma, \lambda') \tag{22}$$

and by the second statement of Lemmas 8.10 and 8.11 the requirement of $f_{\rho\tau}(\rho \cdot \lambda') = \rho \cdot f_\tau(\lambda')$ for all λ' and ρ as formulated using the typing in (21) is equivalent to the one as formulated using the typing in (22). Thus we conclude that

$$\left(\llbracket \Gamma \vdash \mathsf{El}_\Delta(\overline{\forall}\kappa.A)\ \mathsf{type} \rrbracket\right)_{(\mathcal{E},\delta)} (\gamma) = (\llbracket \Gamma \vdash \forall\kappa.\, \mathsf{El}_{\Delta,\kappa}(A)\ \mathsf{type} \rrbracket)_{(\mathcal{E},\delta)} (\gamma)$$

For the second equation recall that if $\sigma : (\gamma[\Delta'], \delta|_{\gamma[\Delta']}) \to (\mathcal{E}', \delta')$ then

$$\left(\left(\llbracket \Gamma \vdash \overline{\forall}\kappa.\, \mathsf{in}_{(\Delta,\kappa),(\Delta',\kappa)} (A) : \mathsf{U}_{\Delta'} \rrbracket\right)_{(\mathcal{E},\delta)} (\gamma)\right)_\sigma$$

is the set of families (f_τ) indexed by maps $\tau : (\mathcal{E}', \delta') \to (\mathcal{E}'', \delta'')$ where each f_τ has type

$$f_\tau : (\lambda'' : \mathcal{E}'') \to (\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).\, \mathsf{in}_{(\Delta,\kappa),(\Delta',\kappa)}(A) \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau\sigma}(\lambda'')$$

such that for all τ, ρ and x it holds that $f_{\rho\tau}(\rho(\lambda'')) = \rho \cdot f_\tau(\lambda'')$. Using Lemma 8.12 the type of f_τ can be rewritten to

$$f_\tau : (\lambda'' : \mathcal{E}'') \to (\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)}(\gamma))_{\tau\sigma j}(\lambda'')$$

and thus we can conclude that

$$\begin{aligned}\left(\left(\llbracket \Gamma \vdash \overline{\forall}\kappa.\, \mathsf{in}_{(\Delta,\kappa),(\Delta',\kappa)}(A) : \mathsf{U}_{\Delta'} \rrbracket\right)_{(\mathcal{E},\delta)}(\gamma)\right)_\sigma &= \left(\left(\llbracket \Gamma \vdash \overline{\forall}\kappa.A : \mathsf{U}_{\Delta'} \rrbracket\right)_{(\mathcal{E},\delta)}(\gamma)\right)_{\sigma j} \\ &= \left(\left(\llbracket \Gamma \vdash \mathsf{in}_{\Delta,\Delta'}(\overline{\forall}\kappa.A) : \mathsf{U}_{\Delta'} \rrbracket\right)_{(\mathcal{E},\delta)}(\gamma)\right)_\sigma\end{aligned}$$

□

8.2.1. *Substitution lemma*

Lemma 8.13. Suppose $\Gamma', \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa}$ and let $\sigma : \Gamma \to \Gamma'$ be a substitution and κ not in Γ. Let $\gamma \in \llbracket \Gamma \vdash \rrbracket_{(\mathcal{E},\delta)}$ and suppose that for all $(\mathcal{E}', \delta')$ and $\gamma' \in \llbracket \Gamma \vdash \rrbracket_{(\mathcal{E}',\delta')}$ and $\lambda' \in \mathcal{E}'$

$$\llbracket \Gamma, \kappa : \mathsf{clock} \vdash A\sigma : (\mathsf{U}_{\Delta,\kappa})\sigma \rrbracket_{(\mathcal{E}',\delta')}(\gamma', \lambda') = \llbracket \Gamma', \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E}',\delta')}(\llbracket \sigma \rrbracket_{(\mathcal{E}',\delta')}(\gamma'), \lambda')$$

Then

$$\llbracket \Gamma \vdash (\overline{\forall}\kappa.A)\sigma : (\mathsf{U}_\Delta)\sigma \rrbracket_{(\mathcal{E},\delta)}(\gamma) = \llbracket \Gamma' \vdash \overline{\forall}\kappa.A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\llbracket \sigma \rrbracket_{(\mathcal{E},\delta)}(\gamma))$$

Proof. By definition, if $\tau : (\gamma\sigma[\Delta], \delta|_{\gamma\sigma[\Delta]}) \to (\mathcal{E}', \delta')$ then

$$\left(\llbracket \Gamma \vdash (\overline{\forall}\kappa.A)\sigma : (\mathsf{U}_\Delta)\sigma \rrbracket_{(\mathcal{E},\delta)}(\gamma)\right)_\tau$$

is the set of families $(f_\omega)_\omega$ indexed over maps $\omega : (\mathcal{E}', \delta') \to (\mathcal{E}'', \delta'')$ of maps with type

$$f_\omega : (\lambda'' : \mathcal{E}'') \to \left(\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A\sigma \rrbracket_{(\mathcal{E},\delta)}(\gamma)\right)_{\omega\tau}(\lambda'')$$

satisfying $f_{\rho\omega}(\rho(\lambda'')) = \rho \cdot f_\omega(\lambda'')$ for all ω, ρ and z. Now,

$$\begin{aligned}&\left(\llbracket \Gamma \vdash (\kappa : \mathsf{clock}).A\sigma \rrbracket_{(\mathcal{E},\delta)}(\gamma)\right)_{\omega\tau}(\lambda'') \\ &= \llbracket \Gamma, \kappa : \mathsf{clock} \vdash A\sigma : (\mathsf{U}_{\Delta,\kappa})\sigma \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\omega\tau + i) \cdot \gamma, \lambda'')_j \\ &= \llbracket \Gamma', \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}(\llbracket \sigma \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\omega\tau + i) \cdot \gamma), \lambda'')_j \\ &= \llbracket \Gamma', \kappa : \mathsf{clock} \vdash A : \mathsf{U}_{\Delta,\kappa} \rrbracket_{(\mathcal{E}'+\mathcal{E},[\delta',\delta])}((\omega\tau + i) \cdot \llbracket \sigma \rrbracket_{(\mathcal{E},\delta)}(\gamma), \lambda'')_j \\ &= \left(\llbracket \Gamma' \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)}(\llbracket \sigma \rrbracket_{(\mathcal{E},\delta)}(\gamma), \lambda'')\right)_{\omega\tau}\end{aligned}$$

where $j : ((\omega\tau\gamma[\Delta], \lambda''), \delta''|_{\omega\tau\gamma[\Delta],\lambda''}) \to (\mathcal{E}'', \delta'')$ is the inclusion. So the type of f_ω is

$$f_\omega : (\lambda'' : \mathcal{E}'') \to \left(\llbracket \Gamma' \vdash (\kappa : \mathsf{clock}).A \rrbracket_{(\mathcal{E},\delta)}(\llbracket \sigma \rrbracket_{(\mathcal{E},\delta)}(\gamma), \lambda'')\right)_{\omega\tau}$$

and thus we see that the families $(f_\omega)_\omega$ are exactly the elements of

$$\left(\llbracket \Gamma' \vdash \overline{\forall}\kappa.A : \mathsf{U}_\Delta \rrbracket_{(\mathcal{E},\delta)}(\llbracket \sigma \rrbracket_{(\mathcal{E},\delta)}(\gamma))\right)_\tau$$

proving the lemma. □

Formation and typing rules

$$\frac{\kappa \in \Delta \qquad \Gamma \vdash A : \overset{\kappa}{\blacktriangleright} \mathsf{U}_\Delta}{\Gamma \vdash \overline{\blacktriangleright}^{\kappa} A : \mathsf{U}_\Delta}$$

Equations

$$\mathsf{El}_\Delta \left(\overline{\blacktriangleright}^{\kappa} \mathsf{next}^{\kappa}\, \xi.A\right) \equiv \overset{\kappa}{\blacktriangleright} \xi.\, (\mathsf{El}_\Delta\, A)$$

$$\mathsf{in}_{\Delta,\Delta'} \left(\overline{\blacktriangleright}^{\kappa} \mathsf{next}^{\kappa}\, \xi.A\right) \equiv \overline{\blacktriangleright}^{\kappa} \mathsf{next}^{\kappa}\, \xi.\, \mathsf{in}_{\Delta,\Delta'}\, (A)$$

Figure 11. Codes for the later modality.

Compared to codes for dependent products and clock quantification the codes for the later types are straightforward and completely analogous to the codes for later types in previous models (Bizjak and Møgelberg, 2015). The typing rules and judgemental equalities for the new construct are listed in Figure 11. The judgemental equalities have a side-condition that the clock κ appears in the clock context Δ and assume A is of type U_Δ.

The interpretation of the code for later is as follows. Let $(\mathcal{E}, \delta)$ be a time object and $\gamma \in [\![\Gamma \vdash]\!]_{(\mathcal{E},\delta)}$ and let $\sigma : \left(\gamma[\Delta], \delta|_{\gamma[\Delta]}\right) \to (\mathcal{E}', \delta')$. We define

$$\left([\![\Gamma \vdash \overline{\blacktriangleright}^{\kappa} A : \mathsf{U}_\Delta]\!]_{(\mathcal{E},\delta)}\, (\gamma)\right)_\sigma = \begin{cases} \{\star\} & \text{if } \delta'(\sigma(\gamma(\kappa))) = 0 \\ \left(\left[\!\left[\Gamma \vdash A : \overset{\kappa}{\blacktriangleright} \mathsf{U}_\Delta\right]\!\right]_{(\mathcal{E},\delta)}\, (\gamma)\right)_{\sigma^{\gamma(\kappa)}} & \text{if } \delta'(\sigma(\gamma(\kappa))) > 0 \end{cases}$$

where

$$\sigma^{\gamma(\kappa)} : \left(\gamma[\Delta], \delta^{-\gamma(\kappa)}|_{\gamma[\Delta]}\right) \to \left(\mathcal{E}', \delta'^{-\sigma(\gamma(\kappa))}\right)$$

is σ, but with changed domain and codomain as a morphism in $\mathbb{T}$. Note that the second case above is well defined since

$$\left[\!\left[\Gamma \vdash A : \overset{\kappa}{\blacktriangleright} \mathsf{U}_\Delta\right]\!\right]_{(\mathcal{E},\delta)} (\gamma) \in \mathcal{U}^{\Delta}_{(\mathcal{E},\delta^{-\gamma(\kappa)})} \left(\mathrm{tick}^{\gamma(\kappa)} \cdot \gamma\right).$$

The restrictions $\tau \cdot (-)$ are again either the unique maps into $\{\star\}$, or inherited from the restrictions of $\left[\!\left[\Gamma \vdash A : \overset{\kappa}{\blacktriangleright} \mathsf{U}_\Delta\right]\!\right]_{(\mathcal{E},\delta)} (\gamma)$.

Checking the judgemental equalities is a straightforward computation as we now demonstrate

by proving one of the judgemental equalities.

$$
\begin{aligned}
&[\![\mathsf{El}_\Delta\,(\overline{\blacktriangleright}^\kappa\,\mathsf{next}^\kappa\,\xi.A)]\!]_{(\mathcal{E},\delta)}\,(\gamma)\\
&\quad= \left([\![\overline{\blacktriangleright}^\kappa\,\mathsf{next}^\kappa\,\xi.A]\!]_{(\mathcal{E},\delta)}\,(\gamma)\right)_{\iota:\left(\gamma[\Delta],\delta|_{\gamma[\Delta]}\right)\to(\mathcal{E},\delta)}\\
&\quad= \begin{cases} \{\star\} & \text{if } \delta(\gamma(\kappa))=0\\ \left([\![\mathsf{next}^\kappa\,\xi.A]\!]_{(\mathcal{E},\delta)}\,(\gamma)\right)_{\iota^{\gamma(\kappa)}:\left(\gamma[\Delta],\delta^{-\gamma(\kappa)}|_{\gamma[\Delta]}\right)\to(\mathcal{E},\delta^{-\gamma(\kappa)})} & \text{otherwise}\end{cases}\\
&\quad= \begin{cases} \{\star\} & \text{if } \delta(\gamma(\kappa))=0\\ \left([\![A]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}\,([\![\xi]\!]_{(\mathcal{E},\delta)}\,(\gamma))\right)_{\iota^{\gamma(\kappa)}:\left(\gamma[\Delta],\delta^{-\gamma(\kappa)}|_{\gamma[\Delta]}\right)\to(\mathcal{E},\delta^{-\gamma(\kappa)})} & \text{otherwise}\end{cases}\\
&\quad= \begin{cases} \{\star\} & \text{if } \delta(\gamma(\kappa))=0\\ [\![\mathsf{El}_\Delta\,A]\!]_{(\mathcal{E},\delta^{-\gamma(\kappa)})}\,([\![\xi]\!]_{(\mathcal{E},\delta)}\,(\gamma)) & \text{otherwise}\end{cases}\\
&\quad= \left[\!\!\left[\overset{\kappa}{\blacktriangleright}\xi.\,(\mathsf{El}_\Delta\,A)\right]\!\!\right]_{(\mathcal{E},\delta)}\,(\gamma)
\end{aligned}
$$

where the last equality follows directly by definition of the interpretation of the later type.

Finally we extend the substitution lemma to the new construct.

Lemma 8.14. Suppose $\Gamma' \vdash A : \overset{\kappa}{\blacktriangleright}\mathsf{U}_\Delta$ and $\kappa \in \Delta$. Let $\rho : \Gamma \to \Gamma'$ be a substitution. Let $(\mathcal{E},\delta)$ be a time object and $\gamma \in [\![\Gamma \vdash]\!]_{(\mathcal{E},\delta)}$ an element of the interpretation of the context. If

$$
\left[\!\!\left[\Gamma' \vdash A : \overset{\kappa}{\blacktriangleright}\mathsf{U}_\Delta\right]\!\!\right]_{(\mathcal{E},\delta)}\,([\![\rho]\!]_{(\mathcal{E},\delta)}\,(\gamma)) = \left[\!\!\left[\Gamma \vdash A\rho : \overset{\rho(\kappa)}{\blacktriangleright}\mathsf{U}_{\rho[\Delta]}\right]\!\!\right]_{(\mathcal{E},\delta)}\,(\gamma) \tag{23}
$$

then

$$
[\![\Gamma' \vdash \overline{\blacktriangleright}^\kappa\,A : \mathsf{U}_\Delta]\!]_{(\mathcal{E},\delta)}\left([\![\rho]\!]_{(\mathcal{E},\delta)}\,(\gamma)\right) = \left[\!\!\left[\Gamma \vdash \overline{\blacktriangleright}^{\rho(\kappa)}\,A\rho : \mathsf{U}_{\rho[\Delta]}\right]\!\!\right]_{(\mathcal{E},\delta)}\,(\gamma) \tag{24}
$$

where recall that we use $\rho(\kappa)$ for the clock κ' satisfying $\Gamma \vdash \kappa' : \mathsf{clock}$ which gets substituted for κ, and analogously $\rho[\Delta]$.

Proof. Since both sides of (24) are elements of $\mathcal{U}^{\gamma[\rho[\Delta]]}_{(\mathcal{E},\delta)}$ we must show that whenever both sides are applied to a morphism of time objects $\sigma : \left(\gamma[\rho[\Delta]], \delta|_{\gamma[\rho[\Delta]]}\right) \to (\mathcal{E}',\delta')$ they yield equal results. If $\delta'(\sigma(\gamma(\rho(\kappa)))) = 0$ then both sides of the equation are clearly going to be chosen singletons, hence equal. If $\delta'(\sigma(\gamma(\rho(\kappa)))) > 0$ then also $\delta(\gamma(\rho(\kappa))) > 0$ and so

$$
\begin{aligned}
\left([\![\Gamma' \vdash \overline{\blacktriangleright}^\kappa\,A : \mathsf{U}_\Delta]\!]_{(\mathcal{E},\delta)}\left([\![\rho]\!]_{(\mathcal{E},\delta)}\,(\gamma)\right)\right)_\sigma &= \left(\left[\!\!\left[\Gamma' \vdash A : \overset{\kappa}{\blacktriangleright}\mathsf{U}_\Delta\right]\!\!\right]_{(\mathcal{E},\delta)}\,([\![\rho]\!]_{(\mathcal{E},\delta)}\,(\gamma))\right)_{\sigma^{([\![\rho]\!]_{(\mathcal{E},\delta)}(\gamma))(\kappa)}}\\
&= \left(\left[\!\!\left[\Gamma' \vdash A : \overset{\kappa}{\blacktriangleright}\mathsf{U}_\Delta\right]\!\!\right]_{(\mathcal{E},\delta)}\,([\![\rho]\!]_{(\mathcal{E},\delta)}\,(\gamma))\right)_{\sigma^{\gamma(\rho(\kappa))}}\\
&= \left(\left[\!\!\left[\Gamma \vdash A\rho : \overset{\rho(\kappa)}{\blacktriangleright}\mathsf{U}_{\rho[\Delta]}\right]\!\!\right]_{(\mathcal{E},\delta)}\,(\gamma)\right)_{\sigma^{\gamma(\rho(\kappa))}}\\
&= \left(\left[\!\!\left[\Gamma \vdash \overline{\blacktriangleright}^{\rho(\kappa)}\,A\rho : \mathsf{U}_{\rho[\Delta]}\right]\!\!\right]_{(\mathcal{E},\delta)}\,(\gamma)\right)_\sigma
\end{aligned}
$$

□

9. Recovering the categories $\mathsf{GR}(\Delta)$

In this final section we discuss the relation to the family of categories $\mathsf{GR}(\Delta)$ defined in previous work by the authors (Bizjak and Møgelberg, 2015). As mentioned in the introduction, this gives a model of guarded recursion with multiple clocks up to a coherence problem. We first recall the definition of the categories $\mathsf{GR}(\Delta)$.

For a finite set of clock variables Δ the category $\mathsf{GR}(\Delta)$ is the category of presheaves on the poset $\mathfrak{I}(\Delta)$. The elements of this poset are pairs (E, δ) where E is an equivalence relation on Δ and $\delta : \Delta \to \mathbb{N}$ is a function which respects the equivalence relation E. The order on $\mathfrak{I}(\Delta)$ is defined so that $(E, \delta) \leq (E', \delta')$ if E is coarser than E' (i.e., $E' \subseteq E$ as subsets of $\Delta \times \Delta$) and δ is pointwise less than δ'. The idea behind this poset is that δ records how much time is left on each clock, and the equivalence relation E states which clocks are identified. The order is defined so that we can pass to a state where there is less time available on each clock, but we can also identify different clocks, i.e., make the equivalence relation coarser.

The intension of the categories $\mathsf{GR}(\Delta)$ is that types and terms in clock variable context Δ should be modelled in $\mathsf{GR}(\Delta)$. In the present paper, the corresponding fragment is modelled in the slice category $\mathrm{Set}^{\mathbb{T}}/\mathcal{C}^{\Delta}$ with the restriction that types must be invariant under clock introduction. Thus the next theorem states that the two models are equivalent.

Theorem 9.1. Let Δ be a finite set of clocks. The full subcategory of $\mathrm{Set}^{\mathbb{T}}/\mathcal{C}^{\Delta}$ on objects invariant under clock introduction is equivalent to the category $\mathsf{GR}(\Delta)$.

Proof (sketch) Recall (Johnstone, 2002, Proposition 1.1.7) that the slice category $\mathrm{Set}^{\mathbb{T}}/\mathcal{C}^{\Delta}$ is equivalent to the category of presheaves over the category of elements of $\mathcal{C}^{\Delta}$. Spelling out the definition of the category of elements, this gives the category of covariant presheaves on the category $\mathcal{R}(\Delta)$ whose objects are triples $(\mathcal{E}, \sigma, \delta)$ with $\sigma : \Delta \to \mathcal{E}$ and $\delta : \mathcal{E} \to \mathbb{N}$. An $\mathcal{R}(\Delta)$ morphism $(\mathcal{E}, \sigma, \delta) \to (\mathcal{E}', \sigma', \delta')$ is a function $\tau : \mathcal{E} \to \mathcal{E}'$ such that $\tau \circ \sigma = \sigma'$ and $\delta' \circ \tau \leq \delta$.

Further, notice that the indexing poset $\mathfrak{I}(\Delta)$ is equivalent to the preorder $\mathcal{S}(\Delta)^{\mathrm{op}}$ where $\mathcal{S}(\Delta)$ is the full subcategory of $\mathcal{R}(\Delta)$ on those objects $(\mathcal{E}, \sigma, \delta)$ where σ is surjective. Indeed, this equivalence follows from the fact that every surjective function σ on Δ determines an equivalence relation on Δ, and every equivalence relation E on Δ gives rise to the surjective quotient function $q : \Delta \to \Delta/E$. Straightforward calculations show this extends to the claimed equivalence of the poset $\mathfrak{I}(\Delta)$ and the preorder $\mathcal{S}(\Delta)^{\mathrm{op}}$.

Thus we have that $\mathsf{GR}(\Delta)$ is equivalent to the category of covariant presheaves on $\mathcal{S}(\Delta)$. By definition there is an inclusion functor $g : \mathcal{S}(\Delta) \to \mathcal{R}(\Delta)$ which gives rise, by precomposition, to a functor $g^* : \mathrm{Set}^{\mathbb{T}}/\mathcal{C}^{\Delta} \to \mathsf{GR}(\Delta)$. Moreover, there is a functor $f : \mathcal{R}(\Delta) \to \mathcal{S}(\Delta)$ which maps $(\mathcal{E}, \sigma, \delta)$ to $\left(\sigma[\Delta], \mathsf{im}(\sigma), \delta|_{\sigma[\Delta]}\right)$. This functor gives rise to a functor $f^* : \mathsf{GR}(\Delta) \to \mathrm{Set}^{\mathbb{T}}/\mathcal{C}^{\Delta}$. It is easy to see $f \circ g = \mathrm{id}$ and that there is a natural transformation $\varepsilon : g \circ f \to \mathrm{id}$ whose component at $(\mathcal{E}, \sigma, \delta)$ is given by the inclusion $\sigma[\Delta] \to \mathcal{E}$. These transformations define an adjunction $g \dashv f$.

Thus $f^* \dashv g^*$ as well; first from $f \circ g = \mathrm{id}$ we have $g^* \circ f^* = \mathrm{id}$, and so the unit η^* of the adjunction is the identity natural transformation, and, second, from the counit of the adjunction $g \dashv f$ we define the counit ε^* of the adjunction $f^* \dashv g^*$ pointwise, as in

$$(\varepsilon^*_X)_{(\mathcal{E},\sigma,\delta)} = X\left(\varepsilon_{(\mathcal{E},\sigma,\delta)}\right).$$

It is standard that an adjunction restricts to an equivalence of full subcategories $\mathbb{C}$ of $\mathsf{GR}(\Delta)$ and $\mathbb{D}$ of $\mathrm{Set}^{\mathbb{T}}/\mathcal{C}^{\Delta}$ on objects where the unit and the counit are isomorphisms, respectively. Because the unit η^* of the adjunction $f^* \dashv g^*$ is an isomorphism the category $\mathbb{C}$ is $\mathsf{GR}(\Delta)$.

The category $\mathbb{D}$ on the other hand is the category of those objects $X \in \mathrm{Set}^{\mathbb{T}}/\mathcal{C}^{\Delta}$ where for every $(\mathcal{E}, \sigma, \delta) \in \mathcal{R}(\Delta)$ the component of the counit

$$(\varepsilon^*_X)_{(\mathcal{E},\sigma,\delta)} = X\left(\varepsilon_{(\mathcal{E},\sigma,\delta)}\right) = X(\iota)$$

where $\iota : \left(\sigma[\Delta], \mathsf{im}(\sigma), \delta|_{\sigma[\Delta]}\right) \to (\mathcal{E}, \sigma, \delta)$ is the inclusion, is an isomorphism. By Lemma 4.1 this holds precisely when the object X is invariant under clock introduction. Hence, the adjunction $f^* \dashv g^*$ restricts to the equivalence of $\mathsf{GR}(\Delta)$ and the full subcategory of $\mathrm{Set}^{\mathbb{T}}/\mathcal{C}^{\Delta}$ on objects invariant under clock introduction. □

Notice, however, that the categories in Theorem 9.1 are *not* isomorphic. This is the key to achieving preservation of structure, chiefly dependent products, in the present model, up to *equality*, as opposed to only up to isomorphism, as in the previous model (Bizjak and Møgelberg, 2015).

Acknowledgements

We thank Patrick Bahr, Lars Birkedal, Hans Bugge Grathwohl and Bassel Mannaa for helpful discussions. We thank the anonymous reviewer for helpful suggestions, and in particular for suggesting a more direct proof of Lemma 4.6. Bizjak was supported by the ModuRes Sapere Aude Advanced Grant from The Danish Council for Independent Research for the Natural Sciences (FNU). Møgelberg was supported by a research grant (13156) from VILLUM FONDEN and DFF-Research Project 1 Grant no. 4002-00442, from The Danish Council for Independent Research for the Natural Sciences (FNU).

References

A. W. Appel and D. A. McAllester. An indexed model of recursive types for foundational proof-carrying code. *ACM Trans. Program. Lang. Syst*, 23(5):657–683, 2001.

Andrew W. Appel, Paul-André Melliès, Christopher D. Richards, and Jérôme Vouillon. A very modal model of a modern, major, general type system. In *POPL*, pages 109–122, 2007.

Robert Atkey and Conor McBride. Productive coprogramming with guarded recursion. In *Proceedings of ICFP 2013*, pages 197–208. ACM, 2013.

Steve Awodey and Michael A Warren. Homotopy theoretic models of identity types. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 146, pages 45–55. Cambridge University Press, 2009.

Patrick Bahr, Hans Bugge Grathwohl, and Rasmus Ejlers Møgelberg. The clocks are ticking: No more delays! In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pages 1–12, 2017.

Marc Bezem, Thierry Coquand, and Simon Huber. A model of type theory in cubical sets. In *19th International Conference on Types for Proofs and Programs, TYPES 2013, April 22-26, 2013, Toulouse, France*, pages 107–128, 2013.

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.

Lars Birkedal, Aleš Bizjak, Ranald Clouston, Hans Bugge Grathwohl, Bas Spitters, and Andrea Vezzosi. Guarded cubical type theory: Path equality for guarded recursion. In *25th EACSL*

Annual Conference on Computer Science Logic, CSL 2016, August 29 - September 1, 2016, Marseille, France, pages 23:1–23:17, 2016.

Aleš Bizjak, Lars Birkedal, and Marino Miculan. A model of countable nondeterminism in guarded type theory. In *RTA-TLCA*, pages 108–123, 2014.

Aleš Bizjak, Hans Bugge Grathwohl, Ranald Clouston, Rasmus Ejlers Møgelberg, and Lars Birkedal. Guarded dependent type theory with coinductive types. In *FoSSaCS*, pages 20–35, 2016.

Aleš Bizjak and Rasmus Ejlers Møgelberg. A model of guarded recursion with clock synchronisation. *Electronic Notes in Theoretical Computer Science*, 319:83 – 101, 2015. ISSN 1571-0661. . The 31st Conference on the Mathematical Foundations of Programming Semantics (MFPS XXXI).

Cyril Cohen, Thierry Coquand, Simon Huber, and Anders Mörtberg. Cubical type theory: a constructive interpretation of the univalence axiom. *CoRR*, abs/1611.02108, 2016. URL http://arxiv.org/abs/1611.02108.

N. A. Danielsson. Beating the productivity checker using embedded languages. In *PAR*, volume 43, pages 29–48, 2010.

Martin Hofmann. On the interpretation of type theory in locally cartesian closed categories. In *Proceedings of Computer Science Logic, Lecture Notes in Computer Science*, pages 427–441. Springer, 1994.

Martin Hofmann. Syntax and semantics of dependent types. In *Extensional Constructs in Intensional Type Theory*, pages 13–54. Springer, 1997.

Martin Hofmann and Thomas Streicher. Lifting Grothendieck universes. Unpublished, 1999. URL www.mathematik.tu-darmstadt.de/~streicher/NOTES/lift.pdf.

J Martin E Hyland, Edmund P Robinson, and Giuseppe Rosolini. The discrete objects in the effective topos. *Proceedings of the London mathematical society*, 3(1):1–36, 1990.

Peter T. Johnstone. *Sketches of an elephant: a topos theory compendium. Vol. 1*, volume 43 of *Oxford Logic Guides*. The Clarendon Press Oxford University Press, New York, 2002. ISBN 0-19-853425-6.

Chris Kapulkin and Peter LeFanu Lumsdaine. The simplicial model of univalent foundations (after voevodsky). *CoRR*, abs/1211.2851, 2012. URL https://arxiv.org/abs/1211.2851.

Z. Luo. *Computation and Reasoning. A Type Theory for Computer Science*. Number 11 in International Series of Monographs on Computer Science. Oxford University Press, 1994.

Saunders MacLane. *Categories for the Working Mathematician*. Graduate Texts in Mathematics. Springer New York, second edition, 1998. ISBN 9780387984032.

P. Martin-Löf. An intuitionistic theory of types: Predicative part. In H.E. Rose and J.C. Shepherdson, editors, *Logic Colloquium*, pages 73–118, Amsterdam, 1973. North-Holland.

Conor McBride and Ross Paterson. Applicative programming with effects. *J. Funct. Programming*, 18(1):1–13, 2008.

Rasmus Ejlers Møgelberg. A type theory for productive coprogramming via guarded recursion. In *Proceedings of CSL-LICS 2014*, pages 71:1–71:10. ACM, 2014.

Rasmus Ejlers Møgelberg and Marco Paviotti. Denotational semantics of recursive types in synthetic guarded domain theory. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pages 317–326, 2016.

Hiroshi Nakano. A modality for recursion. In *Proceedings of LICS 2000*, pages 255–266. IEEE, 2000.

Ulf Norell. *Towards a practical programming language based on dependent type theory*. PhD thesis, Chalmers University of Technology, 2007.

Kasper Svendsen and Lars Birkedal. Impredicative concurrent abstract predicates. In *ESOP*, 2014.

The Coq Development Team. *The Coq proof assistant reference manual*. LogiCal Project, 2004. URL http://coq.inria.fr. Version 8.0.

The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. 2013.