Extending models of second order predicate logic to models of second order dependent type theory

Herman Geuvers*

Faculty of Mathematics and Informatics
Technological University of Eindhoven, The Netherlands

Abstract. We describe a method for constructing a model of second order dependent type theory out of a model of classical second order predicate logic. Apart from the construction being of interest by itself, this also suggests a way of proving the *completeness of the formulas-as-types embedding* from second order predicate logic to second order dependent type theory. Under this embedding, formulas are interpreted as types, and derivability (of a formaula) in the logic should correspond to *inhabitation* (i.e. the associated type being nonempty) in the type system. This correspondence works in one way (called *soundness*): if a formula is derivable, then the associated type is inhabited (there is a term of that type). It's an open problem whether the correspondence works in the other direction (called *completeness*): if the type associated with formula φ is inhabited, then φ is derivable.
The completeness is proved if any model $\mathcal{M}$ of second order logic can *faithfully* be extended to a model $\mathcal{S}(\mathcal{M})$ of second order dependent type theory. That is, for all formulas φ, $\mathcal{M} \models \varphi$ if and only if φ is inhabited in $\mathcal{S}(\mathcal{M})$. In this paper we show that such a *faithfull extension* is possible if $\mathcal{M}$ is a full model of classical second order predicate logic. This implies that a second order formula that is derivable in classical $\lambda P2$ is true in all full models (but it may not be derivable in classical second order logic, due to the existence of non-full models of classical second order logic). We give one small application of the method to an axiomatization of finite structures.

1 Introduction

We describe a method for constructing a model of second order dependent type theory ($\lambda P2$) out of a model of classical second order predicate logic (CPRED2). We show that for *full* models of CPRED2, this construction is *faithful* to the formulas-as-types embedding of CPRED2 into $\lambda P2$ in the following sense. Consider the CPRED2-model $\mathcal{M}$ over signature Σ, where $\mathcal{M}$ is *full*, i.e. the domains in the structure in which the predicates are interpreted are allways the full powerset (for example, a quantification over unary predicates over D ranges over the full powerset $\wp(D)$.) We show how to extend this model to the $\lambda P2$-model

* email: herman@win.tue.nl

$\mathcal{S}(\mathcal{M})$ such that for each second order sentence φ in the language of Σ,

$$(1) \quad \mathcal{M} \models \varphi \text{ iff } \models [\![\overline{\varphi}]\!]^{\mathcal{S}(\mathcal{M})} \neq \emptyset.$$

Here, $\overline{\varphi}$ denotes the interpretation of φ as a type in $\lambda P2$ and $[\![\overline{\varphi}]\!]^{\mathcal{S}(\mathcal{M})}$ denotes the interpretation of $\overline{\varphi}$ in $\mathcal{S}(\mathcal{M})$. An additional property of the model $\mathcal{S}(\mathcal{M})$ is that it models Γ_Σ, the context that interprets the signature Σ in $\lambda P2$, and Γ_{CL}, the context that makes the logic of $\lambda P2$ classical. (So, Γ_{CL} is just $z : \Pi\alpha{:}\star.(\neg\neg\alpha){\to}\alpha.$)

As a consequence of the above property, we find that, for φ a sentence of CPRED2 over the language of Σ,

(2) if $\not\vdash_{\text{CPRED2}} \varphi$ and there is a full countermodel,
then $\neg\exists M[\Gamma_{CL}\Gamma_\Sigma \vdash_{\lambda P2} M : \overline{\varphi}]$.

Viz. suppose that $\mathcal{M}$ is a CPRED2-countermodel to $\vdash_{\text{CPRED2}} \varphi$ and suppose that $\Gamma_{CL}\Gamma_\Sigma \vdash_{\lambda P2} M : \overline{\varphi}$ for some M. Extending $\mathcal{M}$ to the $\lambda P2$-model $\mathcal{S}(\mathcal{M})$, we find that $[\![\overline{\varphi}]\!]^{\mathcal{S}(\mathcal{M})} = \emptyset$ (due to property (1)). On the other hand, $[\![\overline{\varphi}]\!]^{\mathcal{S}(\mathcal{M})} \neq \emptyset$, because the interpretation of M is an element of $[\![\overline{\varphi}]\!]^{\mathcal{S}(\mathcal{M})}$ (due to the soundness of the $\lambda P2$-model). From this contradiction we conclude that there is no M such that $\Gamma_{CL}\Gamma_\Sigma \vdash_{\lambda P2} M : \overline{\varphi}$.

Property (2) above suggests a way of proving the completeness of the formulas-as-types embedding from PRED2 (constructive second order predicate logic) to $\lambda P2$ by semantical methods. It is relatively easy to see (and well-known) that this embedding is *sound*: for φ a sentence of PRED2 over the language of Σ, if $\vdash_{\text{PRED2}} \varphi$, then $\Gamma_\Sigma \vdash_{\lambda P2} M : \overline{\varphi}$ for some term M. (This M can be constructed canonically out of the derivation in PRED2.) The reverse, *completeness*, is still an open problem. It states that, if $\Gamma_\Sigma \vdash_{\lambda P2} M : \overline{\varphi}$ for some term M, then $\vdash_{\text{PRED2}} \varphi$. The reason why this completeness result is not immediate is that in $\lambda P2$, formulas (of the logic) and domains (of the logic) are not only treated in a similar way (as 'types' of 'proofterms', respectively 'elements'), but they also live in the same universe, which means that constructions valid for formulas can also be applied to domains (and vice versa). For the Calculus of Constructions, that 'corresponds to' constructive higher order order predicate logic (in the same way that $\lambda P2$ corresponds to PRED2), it is known that the formulas-as-types embedding is *not* conservative. (See e.g. [Berardi 1990] or [Geuvers 1995].) However, for λP, that 'corresponds to' constructive minimal first order order predicate logic, the formulas-as-types embedding is conservative. (See e.g. [Geuvers 1993].)

That (2) above does not prove completeness is due to the fact that CPRED2 (in general) also has non-full models. So, it occurs for specific φ that $\not\vdash_{\text{CPRED2}} \varphi$, but $\mathcal{M} \models \varphi$ for all full models. The result in this paper only states, that if $\Gamma_{CL}\Gamma_\Sigma \vdash_{\lambda P2} M : \varphi$, then φ is true in all full models, which does not imply that $\vdash_{\text{CPRED2}} \varphi$.

2 Second order dependent type theory

The system of second order dependent type theory, $\lambda P2$, was first introduced in [Longo and Moggi 1988]. It can be seen as a subsystem of the Calculus of Constructions ([Coquand and Huet 1988], [Coquand 1990]), where the operations of forming type constructors are restricted to second order ones. (So, one can quantify over type constructors of kind $\sigma \rightarrow \star$, but one can not form type constructors of kind $(\sigma \rightarrow \star) \rightarrow \star$.) It can also be seen as an extension of the first order system λP, where quantification over type constructors has been added. For an extensive discussion on these systems and their relations, we refer to [Barendregt 1992] or [Geuvers 1993]. Here we just define the system $\lambda P2$ and give some initial motivation for it.

Definition 1. The type system $\lambda P2$ is defined as follows. The set of *pseudoterms*, T, is defined by

$$\mathsf{T} ::= \star \mid \Box \mid \mathrm{Var} \mid (\Pi \mathrm{Var}{:}\mathsf{T}.\mathsf{T}) \mid (\lambda \mathrm{Var}{:}\mathsf{T}.\mathsf{T}) \mid \mathsf{T}\mathsf{T},$$

where Var is a countable set of variables. On T we have the usual notion of β-reduction, $\longrightarrow_\beta$. We adopt from the untyped lambda calculus the conventions of denoting the transitive reflexive closure of $\longrightarrow_\beta$ by $\longrightarrow\!\!\!\!\!\rightarrow_\beta$ and the transitive symmetric closure of $\longrightarrow\!\!\!\!\!\rightarrow_\beta$ by $=_\beta$.

The typing of terms is done under the assumption of specific types for the free variables that occur in the term. This is done in a *context*, a finite sequence of declarations $\Gamma = v_1{:}T_1, \ldots, v_n{:}T_n$ (the v are variables and the T are pseudoterms). Typing judgements are written as $\Gamma \vdash M : T$, with Γ a context and M and T pseudoterms.

The deduction rules for $\lambda P2$ are as follows. (v ranges over Var, s, s_1 and s_2 range over $\{\star, \Box\}$ and M, N, T and U range over T.)

$$\text{(axiom)}\ \vdash \star : \Box \qquad \text{(var)}\ \frac{\Gamma \vdash T : \star/\Box}{\Gamma, v{:}T \vdash v : T} \qquad \text{(weak)}\ \frac{\Gamma \vdash T : \star/\Box \quad \Gamma \vdash M : U}{\Gamma, v{:}T \vdash M : U}$$

$$(\Pi)\ \frac{\Gamma \vdash T : s_1 \quad \Gamma, v{:}T \vdash U : s_2}{\Gamma \vdash \Pi v{:}T.U : s_2} \text{ if } (s_1, s_2) \neq (\Box, \Box)$$

$$(\lambda)\ \frac{\Gamma, v{:}T \vdash M : U \quad \Gamma \vdash \Pi v{:}T.U : s}{\Gamma \vdash \lambda v{:}T.M : \Pi v{:}T.U}$$

$$\text{(app)}\ \frac{\Gamma \vdash M : \Pi v{:}T.U \quad \Gamma \vdash N : T}{\Gamma \vdash MN : U[N/v]} \qquad (\mathrm{conv}_\beta)\ \frac{\Gamma \vdash M : T \quad \Gamma \vdash U : s}{\Gamma \vdash M : U} \text{ if } T =_\beta U$$

The equality in the conversion rule (conv_β) is the β-equality on the set of pseudoterms T. In the rules (var) and (weak) it is always assumed that the newly declared variable is fresh, that is, it has not yet been declared in Γ. For convenience, we split up the set Var into a set $\mathrm{Var}^\star$, the *object variables*, and $\mathrm{Var}^\Box$, the *constructor variables*. Object variables will be denoted by $x, y, z, \ldots$ and constructor variables by $\alpha, \beta, \ldots$. In the rules (var) and (weak), we take the variable v out of $\mathrm{Var}^\star$ if $s = \star$ and out of $\mathrm{Var}^\Box$ if $s = \Box$.

We call a pseudoterm M *well-typed* if there is a context Γ and another pseudoterm N such that either $\Gamma \vdash M : N$ or $\Gamma \vdash N : M$ is derivable. The well-typed terms can be split into the following disjoint subsets: $\{\Box\}$, the set of *kinds* (terms of type $\Box$; including $\star$), the set of constructors (terms of type a 'kind'; including the *types*, terms of type $\star$) and the *objects* (terms of type a 'type').

Convention We denote kinds by $A, B, C, \ldots$, types by $\sigma, \tau, \ldots$, constructors by $P, Q, \ldots$ and objects by $t, q, \ldots$.
If v is not free in U, we denote – as usual – $\Pi x{:}T.U$ by $T \to U$. Furthermore, we let brackets associate to the right, so $T \to T \to T$ denotes $T \to (T \to T)$.

Formulas-as-types embedding from PRED2 *to* $\lambda P2$ There is a *formulas-as-types* embedding from constructive second order predicate logic into $\lambda P2$. We do not describe it in detail, as it is well-known. If we want to define the formulas-as-types embedding from classical second order predicate logic into $\lambda P2$, we just add Γ_{CL} $(= z : \Pi\alpha{:}\star.(\neg\neg\alpha)\to\alpha)$ as an initial part of each context of $\lambda P2$. We remark that, if Σ is a signature of second order predicate logic (determining the second order language $\mathcal{L}_\Sigma$), then there is a (canonical) $\lambda P2$-context Γ_Σ, that contains declarations for dealing with the symbols of Σ. For example, a binary relation symbol P in Σ yields a declaration $\mathsf{P} : \alpha_D \to \alpha_D \to \star$ and a binary function symbol f in Σ yields a declaration $\mathsf{f} : \alpha_D \to \alpha_D \to \alpha_D$ in Γ_Σ. The declaration $\alpha_D : \star$ is the declaration for the type of the elements of the domain of the logic. A (well-known) example of a formula that is provable in PRED2 is

$$\forall X^{(2)}(\forall x, y(X(x,y) \supset X(y,x) \supset \bot)) \supset (\forall x(X(x,x) \supset \bot)),$$

denoting the fact that every asymmetric relation is antireflexive. Its interpretation in $\lambda P2$ is the type

$$\Pi\beta{:}\alpha_D \to \alpha_D \to \star.(\Pi x, y{:}\alpha_D.\beta xy \to \beta yx \to \bot) \to (\Pi x{:}\alpha_D.\beta xx \to \bot),$$

typable in the context $\Gamma = \alpha_D{:}\star$. The formulas-as-types embedding from PRED2 into CPRED2 will be denoted by $\overline{}$, so if φ is a second order formula, then $\overline{\varphi}$ is the associated type in $\lambda P2$. It is a well-known fact that the formulas-as-types embedding is sound:

Fact 2.

$$\Delta \vdash_{\text{PRED2}} \varphi \;\Rightarrow \Gamma_\Sigma, \Gamma_\Delta, \Gamma_\varphi \vdash_{\lambda P2} M : \overline{\varphi} \text{ for some } M,$$

for φ a formula and Δ a set of formulas of PRED2 *over the language of Σ. Here, the context Γ_Δ contains declarations of exactly those variables that are needed to make sure that $\Gamma_\Sigma, \Gamma_\Delta \vdash \overline{\psi} : \star$ for every $\psi \in \Delta$ plus a declaration $z : \overline{\psi}$ for every $\psi \in \Delta$ (to turn ψ into an 'assumption'). The context Γ_φ contains only declarations of those variables to make sure that $\Gamma_\Sigma, \Gamma_\Delta, \Gamma_\varphi \vdash \overline{\varphi} : \star$.*

It is a well-known fact that the term M can be constructed canonically out of the derivation of $\Delta \vdash \varphi$ in PRED2. As an example we remark that

$$\alpha_D{:}\star \vdash \lambda\beta{:}\alpha_D \to \alpha_D \to \star.\lambda h{:}(\Pi x, y{:}\alpha_D.\beta xy \to \beta yx \to \bot).\lambda x{:}\alpha_D.\lambda q{:}\beta xx.hxxqq :$$
$$\Pi\beta{:}\alpha_D \to \alpha_D \to \star.(\Pi x, y{:}\alpha_D.\beta xy \to \beta yx \to \bot) \to (\Pi x{:}\alpha_D.\beta xx \to \bot).$$

3 Second order predicate logic

Consider a (one-sorted) second order signature

$$\Sigma = (\mathsf{P}_1, \ldots, \mathsf{P}_k, \mathsf{f}_1, \ldots, \mathsf{f}_l),$$

where $\mathsf{P}_1, \ldots, \mathsf{P}_k$ is the set of relation symbols (all with fixed arity) and $\mathsf{f}_1, \ldots, \mathsf{f}_l$ is the set of function symbols (all with fixed arity; these include the constants). The terms and the formulas of the second order language over Σ are given by composing the symbols of Σ with other terms, starting from infinitely many first order variables $x_1, x_2, \ldots$ and infinitely many second order variables $X_1^{(n)}, X_2^{(n)}, \ldots$ of all arities n. One can either choose to have just the primitive connectives $\supset$ and $\forall$ (first and second order) and define the other connectives in terms of these, or to have also the other connectives ($\vee, \wedge, \neg, \bot$ and first and higher order $\exists$) as primitives. For reasons of simplicity, we choose for the first option, but for the rest of our exposition it does not really matter. The derivation rules for second order predicate logic are the usual natural deduction style ones. We distinguish between *classical* second order predicate logic, CPRED2, which has the 'double negation' rule and *constructive* second order predicate logic, PRED2, which doesn't.

A *second order structure* for the above signature Σ is a tuple

$$\mathcal{M} = (D, \{D_n\}_{n \in \mathbf{N}}, R_1, \ldots, R_k, g_1, \ldots, g_l),$$

where D is a set, $\{D_n\}_{n \in \mathbf{N}^+}$ is a family of sets such that $D_n \in \wp(D^n)$ for every n, $R_1, \ldots, R_k$ are sets (with $R_i \in D_n$ if the arity of R_i is n) and $g_1, \ldots, g_l$ are functions (with $g_i \in D^n \to D$ if the arity of g_i is n).

For $\mathcal{M}$ a second order structure for Σ, *validity* of a formula φ in $\mathcal{M}$ is defined in the usual Tarskian way by induction on the structure of φ, using *valuations* b_1 and b_2, assigning values to second order, respectively first order variables, and an *interpretation function* I assigning a function (of $\mathcal{M}$) to each function symbol f (respecting the arities) and assigning a set (of $\mathcal{M}$) to each relation symbol P (respecting the arities). The fact that φ is valid in the model $\langle \mathcal{M}, I \rangle$ under valuations b_1, b_2 is denoted by

$$\mathcal{M}, I, b_1, b_2 \models \varphi.$$

Two examples of cases in the inductive definition of $\mathcal{M}, I, b_1, b_2 \models \varphi$ are

$$\mathcal{M}, I, b_1, b_2 \models \forall X^{(n)}.\psi \quad \text{if} \quad \mathcal{M}, I, b_1(X^{(n)} := P), b_2 \models \psi \text{ for all } P \in D_n,$$
$$\mathcal{M}, I, b_1, b_2 \models \exists x.\psi \quad \text{if} \quad \mathcal{M}, I, b_1, b_2(x := d) \models \psi \text{ for some } d \in D.$$

Here, $b_2(x := d)$ denotes the function that assigns $b_2(y)$ to y if $y \neq x$ and d to x. Similarly for $b_2(X^{(n)} := P)$. As usual,

$$\mathcal{M}, I \models \varphi$$

(φ is valid in $\mathcal{M}$ under the interpretation I) denotes the fact that $\mathcal{M}, I, b_1, b_2 \models \varphi$ for all valuations b_1 and b_2.

Not all second order structures for Σ can be called 'models', because the interpretation is not always sound (there are no conditions on the sets D_n, so they may be too small to interpret all definable sets). There are two ways to proceed, both can be found in the literature (see e.g. [van Dalen 1994]). One way is to restrict to so called *full structures*.

Definition 3. A structure $\mathcal{M}$ is *full* if $D_n = \wp(D^n)$ for each n. A model $\langle \mathcal{M}, I \rangle$ is called *full* if the structure $\mathcal{M}$ is full. The class of full models has its associated *satisfaction* relation ($\models_f$), defined as follows. For Δ a set of formulas and φ a formula,

$$\Delta \models_f \varphi,$$

(Δ *f-satisfies* φ) if for all *full* structures $\mathcal{M}$ and interpretation functions I, if $\mathcal{M}, I \models \psi$ $(\forall \psi \in \Delta)$, then $\mathcal{M}, I \models \varphi$.

There is a *soundness* result, saying

$$\Delta \vdash \varphi \Rightarrow \Delta \models_f \varphi.$$

The reverse implication (*completeness*) does not hold (see e.g. [van Dalen 1994]):

$$\Delta \models_f \varphi \not\Rightarrow \Delta \vdash \varphi.$$

Because of the incompleteness of the class of full models, a second approach is usually taken, which is to restrict the class of second order structures to those in which the *comprehension schema* is valid.

Definition 4. A second order structure $\mathcal{M}$ with interpretation function I is called a second order model if the *comprehension schema* is valid in $\langle \mathcal{M}, I \rangle$. That is, for all formulas φ with free variables contained in $\{x_1, \ldots, x_n\}$,

$$\mathcal{M}, I \models \exists X^{(n)} \forall x_1, \ldots, x_n [\varphi \leftrightarrow X^{(n)}(x_1, \ldots, x_n)],$$

where $X^{(n)}$ may not occur in φ. The *satisfaction* relation ($\models$), associated with the class of second order models is then defined as follows. For Δ a set of formulas and φ a formula,

$$\Delta \models \varphi,$$

(Δ *satisfies* φ) if for all second order models $\langle \mathcal{M}, I \rangle$, if $\mathcal{M}, I \models \psi$ $(\forall \psi \in \Delta)$, then $\mathcal{M}, I \models \varphi$.

Now there is a soundness and completeness result saying

$$\Delta \vdash \varphi \Leftrightarrow \Delta \models \varphi,$$

for Δ a set of formulas and φ a formula. The completeness can be proved by faithfully translating second order predicate logic into first order predicate logic and using completeness for the first order predicate logic. (For details see again [van Dalen 1994].)

4 Model construction for $\lambda P2$

To be able to extend a second order model to a model of $\lambda P2$, we first discuss a model notion for $\lambda P2$. This notion is not a general (categorical) one, but a description of a class of models. It is an adaptation of models for the Calculus of Constructions as described in [Stefanova and Geuvers 1996]. As a matter of fact, we simplify the construction in [Stefanova and Geuvers 1996] quite a bit, because the 'countermodels' that we are interested in are of a relatively simple kind.

To extend second order models to $\lambda P2$-models, there are two basic problems. The first is that in $\lambda P2$, there is no distinction between 'sets' and 'formulas', as they both live in the universe $\star$. This implies that the domain from the logic (containing the first order elements) and the formulas from the logic have to be interpreted in the same way in the $\lambda P2$-model. A second problem is that in $\lambda P2$, functions 'compute', whereas in the logic functions are interpreted in a set-theoretic way (as graphs). We shall see later how to solve these problems.

Our models of $\lambda P2$ are built from *weakly extensional combinatory algebras* (weca for short). A *combinatory algebra* (ca for short) is a tuple $\mathcal{A} = \langle \mathbf{A}, \cdot, \mathbf{k}, \mathbf{s}, =_A \rangle$, with $\mathbf{A}$ a set, $\cdot$ a binary function from $\mathbf{A} \times \mathbf{A}$ to $\mathbf{A}$ (usually denoted by infix notation), $\mathbf{k}, \mathbf{s} \in \mathbf{A}$ and $=_A$ an equivalence relation on $\mathbf{A}$, such that $=_A$ is compatible with $\cdot$ and moreover $(\mathbf{k}\cdot a)\cdot b =_A a$ and $((\mathbf{s}\cdot a)\cdot b)\cdot c =_A (a\cdot c)\cdot(b\cdot c)$. For $\mathcal{A}$ a combinatory algebra, the set of *terms over* $\mathcal{A}$, $\mathcal{T}(\mathcal{A})$, is defined by letting $\mathcal{T}(\mathcal{A})$ contain infinitely many variables $v_1, v_2, \ldots$ and distinct elements c_a for every $a \in \mathbf{A}$, and letting $\mathcal{T}(\mathcal{A})$ be closed under application (the operation $\cdot$). Given a term t and a valuation ρ, mapping variables to elements of $\mathbf{A}$, the *interpretation of* t *in* $\mathbf{A}$ *under* ρ, notation $[\![t]\!]^{\mathcal{A}}_{\rho}$ is defined in the usual way ($[\![c_a]\!]^{\mathcal{A}}_{\rho} = a$, $[\![MN]\!]^{\mathcal{A}}_{\rho} = [\![M]\!]^{\mathcal{A}}_{\rho} \cdot [\![N]\!]^{\mathcal{A}}_{\rho}$, etcetera). An important property of cas is that they are *combinatory complete*, i.e. if $t[v] \in \mathcal{T}(\mathcal{A})$ is a term with free variable v, then there is an element in $\mathbf{A}$, usually denoted by $\lambda^* v.t[v]$, such that $\forall x (\lambda^* v.t[v]) \cdot x = t[x]$ in $\mathcal{A}$. (More technically, this means that $\mathcal{A}, \rho \models (\lambda^* v.t[v]) \cdot x = t[x]$ for all ρ.) A ca is *weakly extensional* if $\mathcal{A} \models \forall v (t = t') \rightarrow (\lambda^* v.t) = (\lambda^* v.t')$. This implies that, if $[\![t_1]\!]^{\mathcal{A}}_{\rho(x:=a)} = [\![t_2]\!]^{\mathcal{A}}_{\rho(x:=a)}$ for all $a \in \mathbf{A}$, then $[\![\lambda^* x.t_1]\!]^{\mathcal{A}}_{\rho} = [\![\lambda^* x.t_2]\!]^{\mathcal{A}}_{\rho}$. The need for *weakly extensional* cas comes from the fact that we need

$$M =_\beta N \Rightarrow ([M])_\rho = ([N])_\rho \text{ for all } \rho,$$

where $([-])_\rho$ interprets pseudoterms as elements of $\mathbf{A}$, using a valuation ρ for the free variables. Of course, $([-])_\rho$ is close to $[\![-]\!]^{\mathcal{A}}_{\rho}$, except for the fact that now we also have to interpret abstraction: under $([-])_\rho$, λ is interpreted as λ^*. Now, in general $M =_\beta N \not\Rightarrow ([M])_\rho = ([N])_\rho$ for cas (e.g. take combinatory logic and $M \equiv x$, $N \equiv Ix$). However, for wecas this implication holds. A standard example of a weca is $\Lambda/\beta\eta$ (the classes of open λ-terms modulo $\beta\eta$-equality) with the usual (class)equality. Here is why $\Lambda/\beta\eta$ is a weca: let $t_1, t_2 \in \Lambda$ (for simplicity we reason with representants of classes)

$$\text{for all } q \in \Lambda [t_1[q/x] = t_2[q/x]] \Rightarrow t_1 = t_2$$

$$\Rightarrow (\lambda^* x.t_1)x = (\lambda^* x.t_2)x$$
$$\stackrel{\xi}{\Rightarrow} \lambda x.(\lambda^* x.t_1)x = \lambda x.(\lambda^* x.t_2)x$$
$$\stackrel{\eta}{\Rightarrow} \lambda^* x.t_1 = \lambda^* x.t_2,$$

where the second implication is by $(\lambda^* x.t)q = t[q/x]$ and the third and fourth are by the indicated rules.

The types of $\lambda P2$ will be interpreted as subsets of $\mathbf{A}/{=_A}$, so a type is interpreted as a set of equivalence classes (modulo $=_A$). It is well-known that application generalizes from an operation on $\mathbf{A}$ to an operation on $\mathbf{A}/{=_A}$ (define $[M]\cdot[N]$ as $[M \cdot N]$. Due to the fact that we are in a *weakly extensional* ca, also abstraction generalizes from an operation on $\mathbf{A}$ to an operation on $\mathbf{A}/{=_A}$ by defining $\lambda^* x.[M]$ as $[\lambda^* x.M]$.

Definition 5. A *polyset structure* over the combinatory algebra $\mathcal{A}$ is a collection $\mathcal{P} \subset \wp(\mathbf{A}/{=_A})$ such that

1. $\mathbf{A}/{=_A} \in \mathcal{P}$,
2. $\mathcal{P}$ is closed under arbitrary intersection $\bigcap$,
3. $\mathcal{P}$ is closed under *dependent products*, i.e. if $X \in \mathcal{P}$ and $F : X \to \mathcal{P}$, then $\Pi_{t\in X} F(t) \in \mathcal{P}$, where $\Pi_{t\in X} F(t)$ is defined as

$$\{a \in \mathbf{A}/{=_A} \mid \forall t \in X.a \cdot t \in F(t)\}.$$

The elements of a polyset structure are called *polysets*. In case the function F is the constant function with value Y, we write $X \stackrel{t}{\to} Y$ in stead of $\Pi_{t\in X} Y$.

The dependent product of a polyset structure will be used to interpret types of the form $\Pi x{:}\sigma.\tau$, where both σ and τ are types. The intersection will be used to interpret types of the form $\Pi\alpha{:}A.\sigma$, where σ is a type and A is a kind. To interpret kinds we need a *predicate structure*.

Definition 6. For $\mathcal{P}$ a polystructure, a *predicative structure over* $\mathcal{P}$ is a collection of sets $\mathcal{N}$ such that

1. $\mathcal{P} \in \mathcal{N}$,
2. $\mathcal{N}$ is closed under *set-theoretical dependent products*, i.e. if $X \in \mathcal{P}$ and $F : X \to \mathcal{N}$, then $\prod_{t\in X} F(t) \in \mathcal{N}$, where $\prod_{t\in X} F(t)$ is defined as

$$\{f \mid \forall a \in X(f(a) \in F(a))\}.$$

In case F is a constant function with value A, we write $X \stackrel{k}{\to} A$ in stead of $\prod_{t\in X} A$.
If $\mathcal{A}$ is a combinatory algebra, $\mathcal{P}$ a polystructure over $\mathcal{A}$ and $\mathcal{N}$ a predicative structure over $\mathcal{P}$, then we call the tuple $\langle \mathcal{A}, \mathcal{P}, \mathcal{N}\rangle$ a $\lambda P2$-*structure*.

A predicative structure over a polyset structure $\mathcal{P}$ is intended to give a domain of interpretation for the kinds. For example, if the type σ is interpreted as the polyset X, then the kind $\sigma \rightarrow \sigma \rightarrow \star$ is interpreted as $\prod_{t \in X} \prod_{q \in X} \mathcal{P}$, for which we usually write $X \xrightarrow{k} X \xrightarrow{k} \mathcal{P}$.

We now define three interpretation functions, one for kinds, $\mathcal{V}(-)$, that maps kinds to elements of $\mathcal{N}$, one for constructors (and types), $[\![-]\!]$, that maps constructors to elements of $\bigcup \mathcal{N}$ (and types to elements of $\mathcal{P}$, which is a subset of $\bigcup \mathcal{N}$) and one for objects, $(\![-]\!)$, that maps objects to elements of the combinatory algebra $\mathcal{A}$. All these interpretations are parametrised by *valuations*, assigning values to the free variables (declared in the context).

Let in the following $\mathcal{S} = \langle \mathcal{A}, \mathcal{P}, \mathcal{N} \rangle$ be a $\lambda P2$-structure: $\mathcal{A} = \langle \mathbf{A}, \cdot, \mathbf{k}, \mathbf{s}, =_A \rangle$ is a combinatory algebra, $\mathcal{P}$ is a polyset structure over $\mathcal{A}$ and $\mathcal{N}$ is a predicative structure over the polyset structure $\mathcal{P}$.

Definition 7. A *constructor variable valuation* is a map ξ from $\mathrm{Var}^{\Box}$ to $\bigcup \mathcal{N}$. An *object variable valuation* is a map ρ from $\mathrm{Var}^{\star}$ to $\mathbf{A}/=_A$.

Definition 8. For ρ an object variable valuation, we define the map $(\![-]\!)^{\mathcal{S}}_{\rho}$ from the set of objects to $\mathbf{A}$ as follows. (We usually leave the structure $\mathcal{S}$ implicit.)

$$\begin{aligned}
(\![x]\!)_\rho &:= \rho(x), \\
(\![tq]\!)_\rho &:= (\![t]\!)_\rho \cdot (\![q]\!)_\rho, \text{ if } q \text{ is an object}, \\
(\![tQ]\!)_\rho &:= (\![t]\!)_\rho, \text{ if } Q \text{ is a constructor}, \\
(\![\lambda x{:}\sigma.t]\!)_\rho &:= \lambda^* v.(\![t]\!)_{\rho(x:=v)}, \text{ if } \sigma \text{ is a type}, \\
(\![\lambda \alpha{:}A.t]\!)_\rho &:= (\![t]\!)_\rho, \text{ if } A \text{ is a kind}.
\end{aligned}$$

Definition 9. For ρ an object variable valuation and ξ a constructor variable valuation, we define the maps $\mathcal{V}(-)^{\mathcal{S}}_{\xi\rho}$ and $[\![-]\!]^{\mathcal{S}}_{\xi\rho}$ respectively from kinds to $\mathcal{N}$ and from constructors to $\bigcup \mathcal{N}$ as follows. (We usually leave the structure $\mathcal{S}$ implicit.)

$$\begin{aligned}
\mathcal{V}(\star)_{\xi\rho} &:= \mathcal{P}, \\
\mathcal{V}(\Pi x{:}\sigma.B)_{\xi\rho} &:= \prod_{t \in [\![\sigma]\!]_{\xi\rho}} \mathcal{V}(B)_{\xi\rho(x:=t)}, \\
[\![\alpha]\!]_{\xi\rho} &:= \xi(\alpha), \\
[\![\Pi \alpha{:}A.\tau]\!]_{\xi\rho} &:= \bigcap_{a \in \mathcal{V}(A)_{\xi\rho}} [\![\tau]\!]_{\xi(\alpha:=a)\rho}, \text{ if } A \text{ is a kind}, \\
[\![\Pi x{:}\sigma.\tau]\!]_{\xi\rho} &:= \Pi_{t \in [\![\sigma]\!]_{\xi\rho}} [\![\tau]\!]_{\xi\rho(x:=t)}, \text{ if } \sigma \text{ is a type}, \\
[\![Pt]\!]_{\xi\rho} &:= [\![P]\!]_{\xi\rho}((\![t]\!)_\rho), \\
[\![\lambda x{:}\sigma.P]\!]_{\xi\rho} &:= \lambda t \in [\![\sigma]\!]_{\xi\rho}.[\![P]\!]_{\xi\rho(x:=t)}.
\end{aligned}$$

There is a soundness result saying that if the valuations ξ and ρ *fullfill* the context Γ, then if $\Gamma \vdash P : A$ (A a kind), then $[\![A]\!]_{\xi\rho} \in \mathcal{V}(A)_{\xi\rho}$ and if $\Gamma \vdash t : \sigma$ (σ a type), then $(\![t]\!)_\rho \in [\![\sigma]\!]_{\xi\rho}$.

Definition 10. For Γ a $\lambda P2$-context, ρ an object variable valuation and ξ a constructor variable valuation, we say that ξ, ρ *fullfills* Γ, notation $\xi, \rho \models \Gamma$, if for all $x \in \mathrm{Var}^{\star}$ and $\alpha \in \mathrm{Var}^{\Box}$,

$$x : \sigma \in \Gamma \Rightarrow \rho(x) \in [\![\sigma]\!]_{\xi\rho},$$
$$\alpha : A \in \Gamma \Rightarrow \xi(\alpha) \in \mathcal{V}(A)_{\xi\rho}.$$

Definition 11. The notion of *truth in a $\lambda P2$-structure*, notation $\models^{\mathcal{S}}$ and of *truth*, notation $\models$ are defined as follows. For Γ a context, t an object, σ a type, P a constructor and A a kind of $\lambda P2$,

$$\Gamma \models^{\mathcal{S}} t : \sigma \text{ if } \forall \xi, \rho[\xi, \rho \models \Gamma \Rightarrow (\!|t|\!)_{\rho} \in [\![\sigma]\!]_{\xi\rho}],$$
$$\Gamma \models^{\mathcal{S}} P : A \text{ if } \forall \xi, \rho[\xi, \rho \models \Gamma \Rightarrow [\![P]\!]_{\xi\rho} \in \mathcal{V}(A)_{\xi\rho}].$$

Quantifying over the class of all $\lambda P2$-structures, we obtain

$$\Gamma \models t : \sigma \text{ if } \Gamma \models^{\mathcal{S}} t : \sigma \text{ for all } \lambda P2\text{-structures } \mathcal{S},$$
$$\Gamma \models P : A \text{ if } \Gamma \models^{\mathcal{S}} P : A \text{ for all } \lambda P2\text{-structures } \mathcal{S},$$

Theorem 12 Soundness. *For Γ a context, M an object or a constructor and and T a type or a kind of $\lambda P2$,*

$$\Gamma \vdash M : T \Rightarrow \Gamma \models M : T.$$

Proof. By induction on the derivation of $\Gamma \vdash M : T$ in $\lambda P2$. □

5 From a CPRED2-model to a $\lambda P2$-model

Now, we show how to construct a $\lambda P2$-structure $\mathcal{S}(\mathcal{M})$ out of the second order model $\mathcal{M}$ such that, if $\mathcal{M}$ is a full model, then *validity* is preserved. We know what validity of a formula φ means in a second order ('logical') model: whether the interpretation of φ is true in the model or not. In a type theoretical model, we call a type *valid* if its interpretation is nonempty. This conforms with the 'formulas-as-types' embedding from PRED2 to $\lambda P2$, where a formula is interpreted as the type of its proofs. (Hence, a formula is provable iff its associated type is nonempty.)

Definition 13. For $\mathcal{S}$ a $\lambda P2$-structure, Γ a context, σ a type in Γ and ξ, ρ valuations such that $\xi, \rho \models \Gamma$, we say that σ *is valid in* $\mathcal{S}$ *under* ξ, ρ, notation $\mathcal{S}, \xi, \rho \models^{\lambda P2} \sigma$, if

$$[\![\sigma]\!]^{\mathcal{S}}_{\xi\rho} \neq \emptyset.$$

So, what we are looking for is a method for constructing out of the full second order model $\mathcal{M}$ a $\lambda P2$-structure $\mathcal{S}(\mathcal{M})$ such that for all formulas φ in the language of Σ and for all valuations ξ, ρ such that $\xi, \rho \models \Gamma_\Sigma$,

$$\mathcal{S}(\mathcal{M}), \xi, \rho \models^{\lambda P2} \overline{\varphi} \text{ iff } \mathcal{M}, b_\xi, b_\rho \models \varphi,$$

where b_ξ and b_ρ are valuations defined in some canonical way from ξ and ρ (to be made precise later).

To motivate the construction, we perform it step by step, giving hints on why specific choices are made. So, let in the following Σ be a fixed signature and let $\mathcal{M} = (D, \{D_n\}_{n\in\mathbf{N}}, R_1, \ldots, R_k, g_1, \ldots, g_l)$ be a full second order model for this signature.

The combinatory algebra should contain elements that correspond to the elements of D and elements to interpret the function symbols of the signature Σ. As a basis for the combinatory algebra we take the untyped lambda calculus.

Definition 14. We define the set Λ^+ as follows.

1. $\Lambda \subset \Lambda^+$,
2. if $d \in D$, then $c_d \in \Lambda^+$,
3. $\mathsf{f} \in \Lambda^+$ for each function symbol f in the signature Σ.

On Λ^+ we have the usual notion of $\beta\eta$-reduction. We add a reduction relation $\longrightarrow_R$:

$$\begin{aligned} \mathsf{f}c_{d_1}\cdots c_{d_n} &\longrightarrow_R c_d, \text{ if } g(d_1,\ldots,d_n) = d \text{ in } \mathcal{M}, \\ c_d P_1 \cdots P_n &\longrightarrow_R c_d, \text{ for } d \in D, P_1, \ldots, P_n \in \Lambda^+, \end{aligned}$$

where g is the interpretation of f in the model $\mathcal{M}$. We shall often be a bit informal and just write d for the element $c_d \in \Lambda^+$.
We define the tuple $\mathcal{A}(\mathcal{M})$ as $\langle \Lambda^+, \mathrm{app}, K, S, =_{\beta\eta R}\rangle$. Here, app denotes the (not written) application in λ-calculus, K is $\lambda xy.x$ and S is $\lambda xyz.xz(yz)$.

The reduction rule for c_d may look arbitrary, but is esential for the interpretation of types later. Types are interpreted as sets of $\beta\eta R$-equivalence classes and the domain type α_D is interpreted as the set C, of equivalence classes $[c_d]$ ($d \in D$). Then the R is essential for making sure that if C is a subset of both X and Y, then $C \subset X \to Y$.

Lemma 15. *$\mathcal{A}(\mathcal{M})$ as defined above is a combinatory algebra.*

Proof. By a standard result from Mitschke (see [Barendregt 1984], p. 401) it follows that $\beta\eta R$-reduction is Church-Rosser. Hence $K \neq S$. The other requirements for a ca are satisfied immediately. □

The polyset structure should be defined in such a way that we can prove $\mathcal{S}(\mathcal{M}), \xi, \rho \models^{\lambda P2} \overline{\varphi}$ iff $\mathcal{M}, b_\xi, b_\rho \models \varphi$. This property is proved by proving the following results by simultaneous induction on the structure of φ.

$$[\![\overline{\varphi}]\!]_{\xi\rho} \neq \emptyset \Rightarrow \mathcal{M}, b_\xi, b_\rho \models \varphi,$$
$$[\![\overline{\varphi}]\!]_{\xi\rho} = \emptyset \Rightarrow \mathcal{M}, b_\xi, b_\rho \not\models \varphi.$$

We take a closer look at the case for $\varphi = \forall X^{(1)}.\psi$ in the second implication, so we try to prove $[\![\overline{\forall X^{(1)}.\psi}]\!]_{\xi\rho} = \emptyset \Rightarrow \mathcal{M}, b_\xi, b_\rho \not\models \forall X^{(1)}.\psi$. We have

$$[\![\overline{\varphi}]\!]_{\xi\rho} = \emptyset \text{ iff } \bigcap_{Q \in [\![D]\!]_{\xi\rho} \xrightarrow{k} \mathcal{P}} [\![\overline{\psi}]\!]_{\xi(X:=Q)\rho} = \emptyset,$$

$$\mathcal{M}, b_\xi, b_\rho \not\models \forall X^{(1)}.\psi \text{ iff for all } P \in D_1, \mathcal{M}, b_\xi(X := P), b_\rho \not\models \psi.$$

We have as induction hypothesis that for all ξ, ρ, if $[\![\overline{\psi}]\!]_{\xi\rho} = \emptyset$, then $\mathcal{M}, b_\xi, b_\rho \not\models \psi$. So, we want to reason as follows

$$\begin{aligned}
\bigcap_{Q \in [\![D]\!]_{\xi\rho} \xrightarrow{k} \mathcal{P}} [\![\overline{\psi}]\!]_{\xi(X:=Q)\rho} = \emptyset &\overset{(1)}{\Rightarrow} [\![\overline{\psi}]\!]_{\xi(X:=Q)\rho} = \emptyset \text{ for some } Q \in [\![D]\!]_{\xi\rho} \xrightarrow{k} \mathcal{P} \\
&\overset{(\mathrm{IH})}{\Rightarrow} \mathcal{M}, b_{\xi(X:=Q)}, b_\rho \not\models \psi \text{ for some } Q \in [\![D]\!]_{\xi\rho} \xrightarrow{k} \mathcal{P} \\
&\overset{(2)}{\Rightarrow} \mathcal{M}, b_\xi(X := P), b_\rho \not\models \psi \text{ for some } P \in D_1 \\
&\Rightarrow \mathcal{M}, b_\xi, b_\rho \not\models \forall X^{(1)}.\psi.
\end{aligned}$$

However, the implications (1) and (2) are problematic. For (2), we have to make sure that every constructor variable valuation ξ can be adapted to a valuation b_ξ of second order variables. For this we need the model to be full. (Because $[\![D]\!]_{\xi\rho} \xrightarrow{k} \mathcal{P}$ is the full function space from $[\![D]\!]_{\xi\rho}$ to $\mathcal{P}$, D_1 will have to be the full powerset $\wp(D_1)$.) For (1), we have to make sure that, if $\bigcap_{Q \in [\![D]\!]_{\xi\rho} \xrightarrow{k} \mathcal{P}} [\![\overline{\psi}]\!]_{\xi(X:=Q)\rho} = \emptyset$, then $[\![\overline{\psi}]\!]_{\xi(X:=Q)\rho} = \emptyset$ for some $Q \in [\![D]\!]_{\xi\rho} \xrightarrow{k} \mathcal{P}$. This is of course not the case for all models: we have to choose $\mathcal{P}$ in such a way that this property holds. Similar requirements pop up in the cases $\varphi = \forall x.\psi$ (the set $\Pi_{t \in [\![D]\!]_{\xi\rho}} [\![\psi]\!]_{\xi\rho(x:=t)}$ can be empty without any of the $[\![\psi]\!]_{\xi\rho(x:=t)}$ being empty) and $\varphi = \psi \supset \chi$ (the set $[\![\psi]\!]_{\xi\rho} \xrightarrow{t} [\![\chi]\!]_{\xi\rho}$ can be empty without $[\![\psi]\!]_{\xi\rho} \neq \emptyset$ and $[\![\chi]\!]_{\xi\rho} = \emptyset$). We have the following requirements for the polystructure $\mathcal{P}$.

$$\bigcap_{i \in I} X_i = \emptyset \text{ iff } \exists i \in I.X_i = \emptyset,$$
$$\Pi_{m \in X} Y_m = \emptyset \text{ iff } \exists m \in X.Y_m = \emptyset,$$
$$X \xrightarrow{t} Y = \emptyset \text{ iff } X \neq \emptyset \& Y = \emptyset.$$

So, in a sense, $\mathcal{P}$ must be 'classical'. Note however that it should on the other hand not be too classical: the straightforward choice $\mathcal{P} = \{\emptyset, \Lambda^+\}$ does not work, because then $[\![D]\!]_{\xi\rho} = \Lambda^+$, which contains too many elements.

Definition 16. Define the set $C \subset \Lambda^+/\beta\eta R$ by $C := \{[d] \mid d \in D\}$, the set of $\beta\eta R$-equivalence classes of elements of D. Define the polyset structure $\mathcal{P}(\mathcal{M})$ over Λ^+ as follows.

$$\mathcal{M}(\mathcal{P}) = \{X \mid C \subseteq X \subseteq \Lambda^+/\beta\eta R\} \cup \{\emptyset\}.$$

So, $X \in \mathcal{P}(\mathcal{M})$ if X is empty or X is a set of $\beta\eta R$-equivalence classes that contains C. In the following, we often abbreviate $\mathcal{P}(\mathcal{M})$ to $\mathcal{P}$. Note that C is the smallest nonempty polyset.

We have the following property, whose proof is straightforward.

Lemma 17. *The set $\mathcal{P}(\mathcal{M})$ defined above is a polyset structure and it satisfies the properties*

$$\bigcap_{i \in I} X_i = \emptyset \text{ iff } \exists i \in I. X_i = \emptyset,$$
$$\Pi_{m \in X} Y_m = \emptyset \text{ iff } \exists m \in X. Y_m = \emptyset,$$
$$X \xrightarrow{t} Y = \emptyset \text{ iff } X \neq \emptyset \& Y = \emptyset.$$

(If $X_i \in \mathcal{P}$ for all $i \in I$, $X, Y \in \mathcal{P}$ and $Y_m \in \mathcal{P}$ for every $m \in X$.)

The predicative structure $\mathcal{N}(\mathcal{M})$ over $\mathcal{P}(\mathcal{M})$ and $\mathcal{A}(\mathcal{M})$ is taken to be full hierarchy of dependent-set-theoretical function spaces over $\mathcal{P}$. That is, the smallest set $\mathcal{N}$ such that $\mathcal{P} \in \mathcal{N}$ and if $X \in \mathcal{P}$ and F a set-theoretic function from X to $\mathcal{N}$, then $\prod_{t \in X} F(t) \in \mathcal{N}$.

The valuations ξ and ρ are fixed on the Γ_Σ (the part of the context that represents the signature).

Definition 18. Define the valuations ξ and ρ (such that $\xi, \rho \models \Gamma_\Sigma$) as follows.

$$\begin{aligned}
\xi(\alpha_D) &:= C, \\
\xi(\mathsf{P}^{(n)}) &:= \text{the map in } \xi(\alpha_D) \xrightarrow{k} \cdots \xrightarrow{k} \xi(\alpha_D) \xrightarrow{k} \mathcal{P}(\mathcal{M}) \text{ with} \\
&\qquad \xi(\mathsf{P}^{(n)})([d_1]) \cdots ([d_n]) = C \text{ if } (d_1 \ldots, d_n) \in I(\mathsf{P}^{(n)}), \\
&\qquad\qquad = \emptyset \text{ if } (d_1 \ldots, d_n) \notin I(\mathsf{P}^{(n)}), \\
\rho(\mathsf{f}) &:= \mathsf{f}.
\end{aligned}$$

It is not hard to verify that $\xi, \rho \models \Gamma_\Sigma$.

The valuations ξ and ρ are fixed for the variables declared in Γ_Σ. For the variables declared in $\Gamma_\Delta, \Gamma_\varphi$, the valuations ξ and ρ range over a whole set of possible ones. For each ξ, ρ that fullfills $\Gamma_\Delta, \Gamma_\varphi$, we have to define logical valuations b_ξ and b_ρ from ξ and ρ.

Definition 19. Let ξ and ρ be valuations such that $\xi, \rho \models \Gamma_\Sigma, \Gamma_\Delta \Gamma_\varphi$. Define b_ξ and b_ρ as follows.

1. For $X^{(n)} : \alpha_D \to \cdots \to \alpha_D \to \star$ in $\Gamma_\Delta \Gamma_\varphi$, define

$$b_\xi(X^{(n)}) := \{(d_1, \ldots, d_n) \mid \xi(X^{(n)})([d_1]) \cdots ([d_n]) \neq \emptyset\}.$$

2. For $x : \alpha_D$ in $\Gamma_\Delta \Gamma_\varphi$ (so $\rho(x) \in C$), define

$$b_\rho(x) = d \text{ if } \rho(x) = [d].$$

3. For $v : T$ in Γ_Σ and $x : \overline{\psi}$ in $\Gamma_\Delta \Gamma_\varphi$, nothing has to be defined.

Proposition 20. *Let $\mathcal{M}$ be a full second order model over the signature Σ and let $\mathcal{S}(\mathcal{M})$ be the associated $\lambda P2$-structure. Then for all formulas φ in the language of Σ and valuations ξ and ρ such that $\xi, \rho \models \Gamma_\Sigma, \Gamma_\varphi$,*

$$[\![\overline{\varphi}]\!]_{\xi\rho} \neq \emptyset \Rightarrow \mathcal{M}, b_\xi, b_\rho \models \varphi,$$
$$[\![\overline{\varphi}]\!]_{\xi\rho} = \emptyset \Rightarrow \mathcal{M}, b_\xi, b_\rho \not\models \varphi.$$

Proof. By induction on the structure of φ. □

Corollary 21. *For $\mathcal{M}$ a full second order model over the signature Σ, $\mathcal{S}(\mathcal{M})$ the associated $\lambda P2$-structure, φ a formula in the language of Σ and ξ and ρ valuations such that $\xi, \rho \models \Gamma_\Sigma, \Gamma_\varphi$,*

$$\mathcal{S}(\mathcal{M}), \xi, \rho \models^{\lambda P2} \overline{\varphi} \text{ iff } \mathcal{M}, b_\xi, b_\rho \models \varphi.$$

Corollary 22. *If $\Delta \not\vdash \varphi$ in* CPRED2 *and the countermodel is full, then*

$$\neg\exists t[\Gamma_\Sigma, \Gamma_{\Delta,\varphi} \vdash t : \overline{\varphi}].$$

Proof. Let $\mathcal{M}$ be a CPRED2-countermodel to $\Delta \vdash_{\text{CPRED2}} \varphi$. Suppose that $\Gamma_\Sigma, \Gamma_{\Delta,\varphi} \vdash_{\lambda P2} M : \overline{\varphi}$. We find that $[\![\overline{\varphi}]\!]^{\mathcal{S}(\mathcal{M})} = \emptyset$ by the Theorem, but also that $([M]) \in [\![\overline{\varphi}]\!]$ by soundness. We conclude that there is no M such that $\Gamma_\Sigma \vdash_{\lambda P2} M : \overline{\varphi}$. □

So, what has been proven in this paper about the formulas-as-types embedding from second order predicate logic to $\lambda P2$ is the following.

Theorem 23. *For Σ a second order structure, Δ a finite set of formulas over Σ and φ a formula over Σ,*

$$\exists M[\Gamma_\Sigma, \Gamma_{CL}, \Gamma_\Delta \vdash_{\lambda P2} M : \overline{\varphi}] \text{ iff } \Delta \vdash_f \varphi,$$

i.e. $\overline{\varphi}$ is inhabited iff φ is true in all full models of Δ.

6 A small application of the method

We apply our method to show that for second order signatures that describe finite structures of fixed cardinality, the conservativity holds. More precisely: let Σ be a signature containing n constants $c_1, \ldots, c_n$ and let μ_n and ν_n be the following formulas.

$$\mu_n := \forall x[x = c_1 \vee \cdots \vee x = c_n],$$
$$\nu_n := c_1 \neq c_2 \wedge c_1 \neq c_3 \wedge \cdots \wedge c_{n-1} \neq c_n.$$

If Δ is a consistent finite set of formulas (in the language Σ) containing μ_n and ν_n, then, for all second order sentences φ (in the language Σ),

$$\Delta \vdash_{\mathrm{CPRED2}} \varphi \quad \text{iff} \quad \exists t[\Gamma_\Sigma, \Gamma_{CL}, \Gamma_\Delta \vdash_{\lambda P2} t : \overline{\varphi}] \tag{1}$$

This fact comes as a corollary of Theorem 23, using the fact that all models of Δ are full (every subset of the structure can be described by a predicate in the language Σ).

We motivate that the use of Theorem 23 is essential, and that the result in (1) itself is not at all straightforward, by looking at the situation for $\Delta = \{\mu_2, \nu_2\}$. So, let in the following Δ be the set $\{\mu_2, \nu_2\}$. First note that equality is interpreted in $\lambda P2$ as *Leibniz* equality, denoted by $=_L$: for $x, y : \alpha_D$, $x =_L y$ denotes $\Pi\beta{:}\alpha_D \to \star . Px \to Py$. This equality satisfies all the nice properties that one would expect from an equality. Now, we have the following.

Observation 24. *1. Every second order model of Δ is finite,*
2. Every $\lambda P2$-model of Δ is essentially *infinite.*

The first is immediate: the first order domain consists of interpretations for the constants c_1 and c_2 only, hence all the predicate-domains are finite as well. For the second we have to be a bit more precise in what we mean with *essentially*: we mean to say that there are types in $\lambda P2$ that can only be interpreted as infinite sets. One such type is nat, the type of Curch numerals defined by

$$\mathrm{nat} := \Pi\alpha{:}\star . \alpha \to (\alpha \to \alpha) \to \alpha.$$

The natural numbers can be represented as closed terms (the *Church numerals*) of type nat by defining for $m \in \mathbb{N}$ the term $\underline{m}$ as $\lambda\alpha{:}\star.\lambda x{:}\alpha.\lambda f{:}\alpha\to\alpha . f^m(x)$. Now, from $z : \overline{\nu}_2$, it follows that the type $\underline{k} \neq \underline{m}$ is inhabited for every $k, m \in \mathbb{N}$ with $k \neq m$. This can be observed by noticing that for $k > 0$, $\underline{k}\alpha_D c_1(\lambda x{:}\alpha_D . c_2) =_\beta c_2$ and $\underline{0}\alpha_D c_1(\lambda x{:}\alpha_D . c_2) =_\beta c_1$, so $\underline{k} \neq \underline{0}$ is inhabited for $k > 0$. Furthermore, the predecessor function is definable on Church numerals, so, if $\underline{k+1} = \underline{m+1}$ is inhabited, then $\underline{k} = \underline{m}$ is inhabited. Hence we can conclude that, if $k \neq m$, then $\underline{k} \neq \underline{m}$ is inhabited.

All this implies that all Church numerals $\underline{k}$ have a different interpretation in any (sound) $\lambda P2$-model of the context Δ. So, every $\lambda P2$-model of Δ is *essentially* infinite. The fact that the interpretation of Δ in $\lambda P2$ is nevertheless conservative, means that one can not really use this infiniteness to prove new results.

References

[Barendregt 1984] H.P. Barendregt, *The lambda calculus: its syntax and semantics*, revised edition. Studies in Logic and the Foundations of Mathematics, North Holland.

[Barendregt 1992] H.P. Barendregt, Typed lambda calculi. In *Handbook of Logic in Computer Science*, eds. Abramski et al., Oxford Univ. Press.

[Berardi 1990] S. Berardi, Type dependence and constructive mathematics, Ph.D. Thesis, Universita di Torino, Italy.

[Berardi 1993] S. Berardi, Encoding of data types in Pure Construction Calculus: a semantic justification, in *Logical Environments*, eds. G. Huet and G. Plotkin, Cambridge University Press, pp 30–60.

[Coquand 1990] Th. Coquand, Metamathematical investigations of a calculus of constructions. In *Logic and Computer Science*, ed. P.G. Odifreddi, APIC series, vol. 31, Academic Press, pp 91-122.

[Coquand and Huet 1988] Th. Coquand and G. Huet, The calculus of constructions, *Information and Computation*, 76, pp 95-120.

[van Dalen 1994] D. van Dalen, *Logic and Structure*, third edition. Springer Verlag.

[Geuvers 1993] J.H. Geuvers, *Logics and Type systems*, Ph.D. Thesis, University of Nijmegen, Netherlands.

[Geuvers 1995] J.H. Geuvers, The Calculus of Constructions and Higher Order Logic, in *The Curry-Howard isomorphism*, ed. Ph. de Groote, Volume 8 of the "Cahiers du Centre de logique" (Université catholique de Louvain), Academia, Louvain-la-Neuve (Belgium), pp. 139-191.

[Stefanova and Geuvers 1996] M. Stefanova and J.H. Geuvers, A simple semantics for the Calculus of Constructions, to appear in the Proceedings of the ESPRIT-BRA 'Types' meeting, Turin, Italy, 1995.

[Girard et al. 1989] J.-Y. Girard, Y. Lafont and P. Taylor, *Proofs and types*, Camb. Tracts in Theoretical Computer Science 7, Cambridge University Press.

[Harper et al. 1987] R. Harper, F. Honsell and G. Plotkin, A framework for defining logics. *Proceedings Second Symposium on Logic in Computer Science*, (Ithaca, N.Y.), IEEE, Washington DC, pp 194-204.

[Howard 1980] W.A. Howard, The formulas-as-types notion of construction. In *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, eds. J.P. Seldin, J.R. Hindley, AcademicPress, New York, pp 479-490.

[Longo and Moggi 1988] G. Longo and E. Moggi, Constructive Natural Deduction and its "Modest" Interpretation. Report CMU-CS-88-131.

[Nederpelt et al. 1994] R.P. Nederpelt, J.H. Geuvers and R.C. de Vrijer (editors), *Selected Papers on Automath*, Volume 133 in Studies in Logic and the Foundations of Mathematics, North-Holland, Amsterdam, 1994, pp 1024.

[Streicher 1991] T. Streicher, Independence of the induction principle and the axiom of choice in the pure calculus of constructions, *TCS* 103(2), pp 395 - 409.

This article was processed using the LaTeX macro package with LLNCS style