Pedagogical lambda-cube: the λ^2 case

Vincent Demange

Pedagogical lambda-cube: the λ^2 case

Vincent Demange
Loria, University of Lorraine, Nancy, France
vincent.demange@loria.fr

Abstract. In pedagogical formal systems one needs to systematically give examples of hypotheses made. This main characteristic is not the only one needed, and a formal definition of *pedagogical sub-systems of the Calculus of Constructions* (CC) has already been stated. Here we give such a pedagogical sub-system of CC corresponding to the second-order pedagogical λ-calculus of Colson and Michel. It thus illustrates the appropriateness of the formal definition, and opens the study to stronger systems of the λ-cube, for which CC is the most expressive representative. In addition we study the type-checking problem for the formalisms of those pedagogical calculi of second-order.

Keywords: typed lambda-calculus, calculus of constructions, pedagogical formal systems, mathematical logic, negationless mathematics, constructive mathematics.

1 Introduction

The Poincaré criterion The main feature of pedagogical formal systems is to always require the user to give examples of used hypotheses. This need for systematic *exemplification* has lead to the terminology of *pedagogical formal systems*, because it is the formal counterpart of the usual informal teaching practice consisting of giving examples of newly introduced notions. The necessity of such a constraint was already observed by Poincaré [34] in the case of definitions by postulates: "A definition by postulate has value only when the existence of the object defined has been proved [...] by means of examples [...].". Since every set of hypotheses made on some objects (e.g. propositions or λ-terms) can be seen as a set of definitions by postulates, in the following when for a formal system every set of used hypotheses can be exemplified we will say it meets the *Poincaré criterion*.

Formal pedagogy More formally, for instance in propositional natural deduction systems —studied by Colson and Michel up to the propositional second-order calculus in [4, 5]— whenever one wants to use the set of formulas Δ as hypotheses she must give a substitution σ (the examples) from propositional variables to formulas such that $\vdash\sigma(A)$ for each $A \in \Delta$. Through the propositions-as-types correspondence [21] this requirement extends to type systems —second-order λ-calculus studied by Colson and Michel in [6]: a typing environment $x_1 : A_1, \ldots, x_n : A_n$ can be exemplified if there are terms t_i and a substitution σ from type variables to types such that $\vdash t_i : \sigma(A_i)$.

From a logical point of view and in an intuitionist framework, this *pedagogical constraint* does not allow the use of negation and reasoning by contradiction: it is no more possible to assume a formula A that will reveal to be a contradiction since no instance of this formula can be proved. It then agrees with the negationless mathematics advocated by Griss as a refinement of intuitionism [16, 17, 18, 19].

From a computational point of view, it means that for every type at least one of its instances has to be inhabited by a term. This last property leads to the notion of *usefulness* of λ-terms in pedagogical type systems: every function f of type $A \to B$ can be applied to a term u of type A when A is closed.

Overview of the article In this article, we will focus on the extension of these results to the type systems of the Barendregt's λ-cube [1]. Indeed those systems have logical and computational meaning, and the most powerful is the Calculus of Constructions (CC) of Coquand [7] for which a formal study of pedagogy has already been investigated by Colson and Demange in [3]. First the formalism of CC being more explicit, the Poincaré criterion become: if an environment $x_1 : A_1, \ldots, x_n : A_n$ is well-formed [1] then there are terms t_i such that $\vdash t_i : A_i[x_1, \ldots, x_{i-1} \leftarrow t_1, \ldots, t_{i-1}]$ where $[\cdot \leftarrow \cdot]$ is the usual substitution from variables to terms. The conclusion of the investigation was a complete formal definition of a *pedagogical subsystem of CC* (see def. 10): the formal system has to (i) be a subsystem of CC; (ii) verify subject reduction; (iii) meet the Poincaré criterion; (iv) and meet the *converse of the Poincaré criterion*. The converse of the Poincaré criterion is needed to ensure expressiveness: in [3] a system $\mathrm{CC_r}$ satisfying (i), (ii) and (iii) but not (iv) has been exhibited with a good computational power but strong logical limitations. Also in [5] a *weakly pedagogical second-order calculus* $\mathrm{P_s}$-Prop^2 has been stated for which a type system can be obtained satisfying (i), (iii) and (iv) but not (ii).

At the end of the study about pedagogical CC, it was suggested that it is possible to build a *pedagogical subsystem of CC* in the precise sense of the previous definition, corresponding to the pedagogical second-order λ-calculus P-Prop^2 of [6]. This construction is the main subject of this present paper, the difficulties were mainly due to a difference in formalism, the one of P-Prop^2—and of Girard's System F [13]— being more liberal than the one of CC, and the need to stick to the definition. Especially the (i) of the previous definition does not allow the addition of constant symbols (initial examples) to the calculus, which was the case in P-Prop^2.

Outline of the article In section 2 we recall the usual notations, definitions and well-known results about the calculus of constructions (CC) and its subsystem of second order λ^2. In sections $3, 4, 5$ we define and study pedagogical subsystems of CC of second order: first with explicit and total examples (also called motivations) λ^2_e, then with total motivations λ^2_t and finally with partial motivations λ^2_p. Each is obtained from the previous by relaxing some constraints, the last one fully satisfying the definition of pedagogical subsystem of CC. Then in section 6 we link those systems with the previously stated pedagogical second order λ-calculus P-Prop^2 of [6]. We end in section 7 by showing the undecidabilty of type checking for all those type systems. Finally we conclude in section 8 by suggesting a formalism to recover type checking

[1] The *well-formedness* of environments Γ are formal judgements in CC written $\Gamma\,\mathsf{wf}$.

in pedagogical formal systems, and open the study toward more expressive systems of the λ-cube based on the current work.

Related works Obviously the works on pedagogical formal systems previously mentioned are relevant: the minimal propositional calculus over $\rightarrow$, $\wedge$ and $\vee$ has been studied in [4]; the second order propositional calculus in [5]; the second-order λ-calculus in [6]; and an investigation on the whole Calculus of Constructions in [3]. A great overview of those works can be found in the introduction of [3], to which we can add the following unmentioned and unpublished [2] result of Michel in [28]: every λ-term of the second-order λ-calculus admit a *continuation passing style* translation that can be typed in the pedagogical second-order λ-calculus, ensuring the preservation of programs.

Also in an intuitionistic framework, which is the case here, *pedagogical mathematics* are linked with the negationless mathematics philosophy. The idea of negationless mathematics appeared in the middle of the last century when Griss expressed it as a step further of the intuitionistic philosophy of Brouwer. Indeed, in intuitionistic mathematics, a proof of a negative statement $\neg A$ impose to assume A in order to obtain a contradiction. But assuming A is no *intuitive* method for Griss since it will reveal to be an impossible construction. First works of Griss [16, 17, 18, 19] constitute an informal outline of a geometry, an arithmetic, a set theory and an analysis without negation. Heyting [20] and Franchella [10] summarize differences of viewpoint about intuitionism of Brouwer and that of Griss. Some formal developments of the Griss *desiderata* has been proposed, from which we can cite those of Vredenduin [38], Gilmore [12], Valpola [37], Nelson [32, 31], Minichiello [29], López-Escobar [24, 25], Mezhlumbekova [27] and more recently of Krivtsov [22, 23], dealing with negationless predicate logic and arithmetic in natural deduction systems or in sequent calculus. One of the main ideas is the introduction of a *quantified implication* $A(\vec{x}) \rightarrow_{\vec{x}} B(\vec{x})$ which is interpreted in intuitionistic logic by $\forall \vec{x}\ A(\vec{x}) \rightarrow B(\vec{x}) \wedge \exists \vec{x} A(\vec{x})$. Mints [30] provides a good overview of those works.

2 Background and Notations

In this section, we briefly recall usual notations, definitions and results about the Calculus of Constructions (CC) and its subsystem of second-order λ^2. At the end we recall the formal definition of *pedagogical sub-system of the Calculus of Constructions* resulting of the study in [3].

2.1 Definitions and notations

We try to use $x, y, \ldots$ as symbols for variables, $u, v, w, t, \ldots$ to denote terms, $A, B, \ldots$ for types or formulas, $\Gamma, \Gamma', \ldots$ for environments.

The set of raw terms of CC is defined by induction: the variables x, and constants Prop and Type are raw terms; $\lambda x^A.u$, $\forall x^A.B$ and $u\ v$ are raw terms if x is a variable

[2] Actually a stronger but non-constructive result concerning the preservation of programs that can be typed in the $\lambda\mu$-calculus of Parigot [33] is present in [6].

and u, v, A, B are raw terms. $\mathcal{S}(u)$ is the set of sub-terms of u, containing u. For brevity, in the following *terms* will refer to raw terms.

$\equiv$ is the syntactical equality of terms modulo renaming of bound variables[3]. We note by $\rightsquigarrow_\beta$ the usual beta-reduction relation between terms; $\overset{*}{\rightsquigarrow}_\beta$ its reflexive and transitive closure; and $=_\beta$ its equivalence closure. A term u is in normal form if it is not reducible, i.e. there is no term t such that $u \rightsquigarrow_\beta t$. If all possible reductions from a term u lead to a normal form, then the term u is said to be strongly normalizing.

$\mathcal{V}(t)$ is the set of free variables of t. If $\mathcal{V}(t) = \emptyset$ then t is said to be closed. The usual capture avoiding substitution of u for x in t is noted $t[x \leftarrow u]$; and $t[x_1, \ldots, x_n \leftarrow u_1, \ldots, u_n]$ is the simultaneous substitution of u_1 for x_1, u_2 for x_2, etc. in t. When dealing with substitutions as mathematical objects, we will use list symbolism: $[]$ is the empty substitution, and if σ is a substitution then $\sigma::(x \mapsto a)$ is a new substitution mapping all variables $y \neq x$ to $\sigma(y)$ and x to a. The application of a substitution is extended from variables to terms in the usual way: if $\sigma \equiv (x_1 \mapsto u_1)::\ldots::(x_n \mapsto u_n)$ then $\sigma(t) = t[x_1, \ldots, x_n \leftarrow u_1, \ldots, u_n]$.

To shorten notations, we might use a vector symbolism: $\vec{t}$ denotes a sequence of terms $t_1, \ldots, t_n$; and $\forall \vec{x}^{\vec{A}}.B$ denotes $\forall x_1^{A_1} \ldots \forall x_n^{A_n}.B$. As usual, $A \to B$ is short for $\forall x^A.B$ when x does not appear in $\mathcal{V}(B)$.

An environment is a finite list of associations variable-term. The empty environment is noted $[]$ or omitted, otherwise it is of the form $x_1 : A_1, \ldots, x_n : A_n$, or Γ, Γ' where Γ and Γ' are environments. The domain of an environment is the finite set of its variables: $\mathrm{dom}(x_1 : A_1, \ldots, x_n : A_n) = \{x_1, \ldots, x_n\}$. Substitutions can be applied to environments: $(x_1 : A_1, \ldots, x_n : A_n)[y \leftarrow u] \equiv x_1 : A_1[y \leftarrow u], \ldots, x_n : A_n[y \leftarrow u]$.

$\Gamma' \equiv x_1 : A_1, \ldots, x_i : A_i$ is an initial segment of $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$ when $i \leq n$, abbreviated by $\Gamma' \preccurlyeq \Gamma$. Similarly for substitutions $\sigma' \preccurlyeq \sigma$. We will also write $\Gamma_{\leq i}$ or $\Gamma_{<i}$ for the first i-th (resp. $i-1$-th) elements of Γ, similarly with $\sigma_{\leq i}$ or $\sigma_{<i}$.

In CC there are two kinds of judgements: $\Gamma\ \mathsf{wf}^{\mathsf{C}}$ means that the environment Γ is syntactically well-formed, and $\Gamma \vdash^{\mathsf{C}} t : A$ expresses that the term t is of type A in the environment Γ.

Implicitly, $\Gamma \vdash^{\mathsf{C}} A : \kappa$ signifies that there is $\kappa \in \{\mathrm{Prop}, \mathrm{Type}\}$ such that this previous statement holds. $\Gamma \vdash^{\mathsf{C}} A_1 : A_2 : \ldots : A_n$ is the contraction of $\Gamma \vdash^{\mathsf{C}} A_1 : A_2$, etc. and $\Gamma \vdash^{\mathsf{C}} A_{n-1} : A_n$. If the contraction appears as a premise of a rule it denotes $n-1$ premises, and as a conclusion of a rule it expands to $n-1$ possible conclusions (i.e. $n-1$ rules).

Rules of CC are presented in fig. 1: close presentations can be found in [8], with well formed judgements; in [2] avoiding weakening rule; or [1] presenting usual properties of CC. Removing some rules of CC we obtain λ^2 of fig. 2, a subsystem corresponding to the polymorphic λ-calculus also known as the system F of Girard-Reynolds [14, 35]. Notice that the raw-terms stay the same.

As usual a derivation of a judgement is a finite tree rooted by the judgement and where leafs are instances of inference rules without premise. A sub-derivation is then a sub-tree, and a strict sub-derivation is a sub-tree which is not the whole tree.

[3] As in [9], we assume De Bruijn indexes for bound variables and identifiers for free variables. So there is no need for α-conversion notion which is implicit.

$$\frac{}{[\,]\,\mathsf{wf}^{\mathsf{c}}}\ (\text{c-env}_1) \qquad \frac{\Gamma \vdash^{\mathsf{c}} A : \kappa \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^{\mathsf{c}}}\ (\text{c-env}_2)$$

$$\frac{\Gamma\,\mathsf{wf}^{\mathsf{c}}}{\Gamma \vdash^{\mathsf{c}} \mathrm{Prop} : \mathrm{Type}}\ (\text{c-ax}) \qquad \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}^{\mathsf{c}}}{\Gamma, x : A, \Gamma' \vdash^{\mathsf{c}} x : A}\ (\text{c-var})$$

$$\frac{\Gamma, x : A \vdash^{\mathsf{c}} u : B : \kappa}{\Gamma \vdash^{\mathsf{c}} \lambda x^A.u : \forall x^A.B}\ (\text{c-abs}) \qquad \frac{\Gamma \vdash^{\mathsf{c}} u : \forall x^A.B \quad \Gamma \vdash^{\mathsf{c}} v : A}{\Gamma \vdash^{\mathsf{c}} u\ v : B[x \leftarrow v\,]}\ (\text{c-app})$$

$$\frac{\Gamma, x : A \vdash^{\mathsf{c}} B : \kappa}{\Gamma \vdash^{\mathsf{c}} \forall x^A.B : \kappa}\ (\text{c-prod}) \qquad \frac{\Gamma \vdash^{\mathsf{c}} t : A \quad \Gamma \vdash^{\mathsf{c}} A' : \kappa \quad A =_\beta A'}{\Gamma \vdash^{\mathsf{c}} t : A'}\ (\text{c-conv})$$

where κ stands for Prop or for Type.

Figure 1: Inference rules of CC.

$$\frac{}{[\,]\,\mathsf{wf}^2}\ (\text{env}_1) \qquad \frac{\Gamma \vdash^2 A : \kappa \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^2}\ (\text{env}_2)$$

$$\frac{\Gamma\,\mathsf{wf}^2}{\Gamma \vdash^2 \mathrm{Prop} : \mathrm{Type}}\ (\text{ax}) \qquad \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}^2}{\Gamma, x : A, \Gamma' \vdash^2 x : A}\ (\text{var})$$

$$\frac{\Gamma, x : A \vdash^2 u : B : \mathrm{Prop}}{\Gamma \vdash^2 \lambda x^A.u : \forall x^A.B}\ (\text{abs}) \qquad \frac{\Gamma \vdash^2 u : \forall x^A.B \quad \Gamma \vdash^2 v : A}{\Gamma \vdash^2 u\ v : B[x \leftarrow v\,]}\ (\text{app})$$

$$\frac{\Gamma, x : A \vdash^2 B : \mathrm{Prop}}{\Gamma \vdash^2 \forall x^A.B : \mathrm{Prop}}\ (\text{prod})$$

Figure 2: Inference rules of λ^2.

2.2 Properties of CC

In the sequel we shall need the following well-known results about CC and λ^2 (omitted proofs can be found in [1]). Starred relations refer to both CC and λ^2, meaning that the property holds in both systems.

Property 1 (free variables)

(i) If $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^\star$ or $x_1 : A_1, \ldots, x_n : A_n \stackrel{\star}{\vdash} w : C$ then for all i, $\mathcal{V}(A_{i+1}) \subseteq \{x_1, \ldots, x_i\}$ and $x_i \not\equiv x_j$ for all $i \neq j$;

(ii) If $x_1 : A_1, \ldots, x_n : A_n \stackrel{\star}{\vdash} w : C$ then in addition $\mathcal{V}(w, C) \subseteq \{x_1, \ldots, x_n\}$.

Property 2 If $\Gamma \,\mathsf{wf}^\star$ or $\Gamma \stackrel{\star}{\vdash} w : C$ then $\mathrm{Type} \notin \mathcal{S}(\Gamma)$ and $\mathrm{Type} \notin \mathcal{S}(w)$.

Property 3 (environments validity)

(i) if $\Gamma \,\mathsf{wf}^\star$, then for all environments $\Gamma' \preccurlyeq \Gamma$, $\Gamma' \,\mathsf{wf}^\star$ is a sub-derivation;

(ii) if $\Gamma \stackrel{\star}{\vdash} w : C$, then for all environments $\Gamma' \preccurlyeq \Gamma$, $\Gamma' \,\mathsf{wf}^\star$ is a strict sub-derivation.

Property 4 (environment types validity) If $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^\star$, then for all i there is κ such that $x_1 : A_1, \ldots, x_i : A_i \stackrel{\star}{\vdash} A_{i+1} : \kappa$ is a strict sub-derivation.

Property 5 (type uniqueness) If $\Gamma \stackrel{2}{\vdash} w : C$ and $\Gamma \stackrel{2}{\vdash} w : C'$ then $C \equiv C'$.

Property 6 (type correctness) If $\Gamma \stackrel{\star}{\vdash} w : C$ then $C \equiv \mathrm{Type}$ or $\Gamma \stackrel{\star}{\vdash} C : \kappa$.

Property 7

(i) If $\Gamma \stackrel{2}{\vdash} C : \mathrm{Type}$ then $C \equiv \mathrm{Prop}$ and the last used rule is (ax);

(ii) If $\Gamma \stackrel{2}{\vdash} C : \mathrm{Prop}$ then the last used rule is (var) or (prod).

Proof by case analysis on the last used rule.

(i) **(var)** Impossible case because Type can not be in the environment (prop. 2).

(app) There are two cases:

- $B \equiv x$ and $v \equiv \mathrm{Type}$: which is impossible (prop. 2);
- $B \equiv \mathrm{Type}$: hence $\Gamma \stackrel{2}{\vdash} \forall x^A.\, \mathrm{Type} : \kappa$ (prop. 6), which is impossible (prop. 2).

(ii) **(app)** We have $\Gamma \stackrel{2}{\vdash} \forall x^A.B : \kappa$ (prop. 6) which has to be obtained by the (prod) rule, hence $\Gamma, x : A \stackrel{2}{\vdash} B : \mathrm{Prop}$. From $B[x \leftarrow v\,] \equiv \mathrm{Prop}$ we have two cases:

- $B \equiv x$ and $v \equiv \mathrm{Prop}$: the second premise is then $\Gamma \stackrel{2}{\vdash} \mathrm{Prop} : A$ obtained by the (ax) rule, hence $A \equiv \mathrm{Type}$ which is impossible (prop. 2);
- $B \equiv \mathrm{Prop}$: then $\Gamma, x : A \stackrel{2}{\vdash} \mathrm{Prop} : \mathrm{Prop}$ which is impossible.

□

Property 8 If $\Gamma \stackrel{2}{\vdash} C : \mathrm{Prop}$ then for all $x \in \mathcal{V}(C)$, $(x : \mathrm{Prop}) \in \Gamma$.

Proof by structural induction on the derivation: we only need to consider the rules (var) and (prod) (prop. 7). □

Property 9 If $\Gamma \vdash^2 w : C$ or $\Gamma \vdash^2 C : \kappa$ or $\Gamma\,\mathsf{wf}^2$ where $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$ then C and the A_i are in normal form.

Proof The proof can be split in two simple steps:

- if $\Gamma\,\mathsf{wf}^2$ or $\Gamma \vdash^2 C : \kappa$ with $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$ then λ does not appear in C nor in any A_i, proved by structural induction on the derivation;
- every reducible raw term u contains the symbol λ, proved by induction on the usual inductive definition of $u \rightsquigarrow_\beta u'$.

□

Definition 10 (pedagogical subsystem of CC)

CC* is a pedagogical subsystem of CC if:
(i) CC* is a subsystem of CC: $\Gamma\,\mathsf{wf}^\star$ implies $\Gamma\,\mathsf{wf}^\mathsf{c}$, and $\Gamma \vdash^\star t : C$ implies $\Gamma \vdash^\mathsf{c} t : C$.
(ii) CC* satisfies subject reduction: if $\Gamma \vdash^\star t : C$ and $t \rightsquigarrow_\beta t'$ then $\Gamma \vdash^\star t' : C$.
(iii) CC* meets the Poincaré criterion and its converse: $x_1 : A_1, \ldots, x_n : A_n\,\mathsf{wf}^\star$ **if and only if** $x_1 : A_1, \ldots, x_n : A_n\,\mathsf{wf}^\mathsf{c}$ and there are terms $t_1, \ldots, t_n$ such that

$$\vdash^\star t_1 : A_1 \qquad \vdash^\star t_2 : A_2[x_1 \leftarrow t_1\,] \quad \ldots \quad \vdash^\star t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}\,]$$

3 Total and explicit motivations

Usually the current state of a proof is indicated by a sequent $\Gamma \vdash t : A$ meaning that "t is a proof of A under the assumptions Γ". In the pedagogical practice we also need examples of the hypotheses of Γ which we can make explicit using *enhanced sequents* of the form $\Gamma \vdash_\sigma t : A$ meaning "t is a proof of A under the assumptions Γ exemplified by σ" where σ is a substitution from the variables of Γ to terms. In the same way we switch from judgements $\Gamma\,\mathsf{wf}$ to $\Gamma\,\mathsf{wf}_\sigma$. Each assumption/variable of Γ has to be examplified by σ, hence the *total and explicit motivations* system λ_e^2 of fig. 3.

Making the examples/motivations explicit have at least two benefits. First it allows to better reflect the practice of pedagogical mathematics by using a global example during a proof. Second it simplifies and specifies the statements about the formalism: we can act on the motivations and then appreciate the constraints they impose or they are subject to.

3.1 System definition

We extend the raw terms with the two constants o and $\top$. Inference rules of λ_e^2 are presented on fig. 3. The (prod) rule of λ^2 is constrained as (e-prod) in λ_e^2 in order to avoid empty types as soon as possible (e.g. $\forall A^{\mathrm{Prop}}.A$). Indeed those empty types can not be examplified, and allowing to manipulate them could break the subject reduction property (see [5]) or the Poincaré criterion if we introduce them into environments. The added constraint then requires that the formed type to be compatible with the current motivation σ, namely that the instance $\sigma(\forall x^A.B)$ be inhabited.

Also the additional (second) premise of the rule (e-env$_2$) should not be considered as a constraint: the term a is already contained in the derivation of the first premise

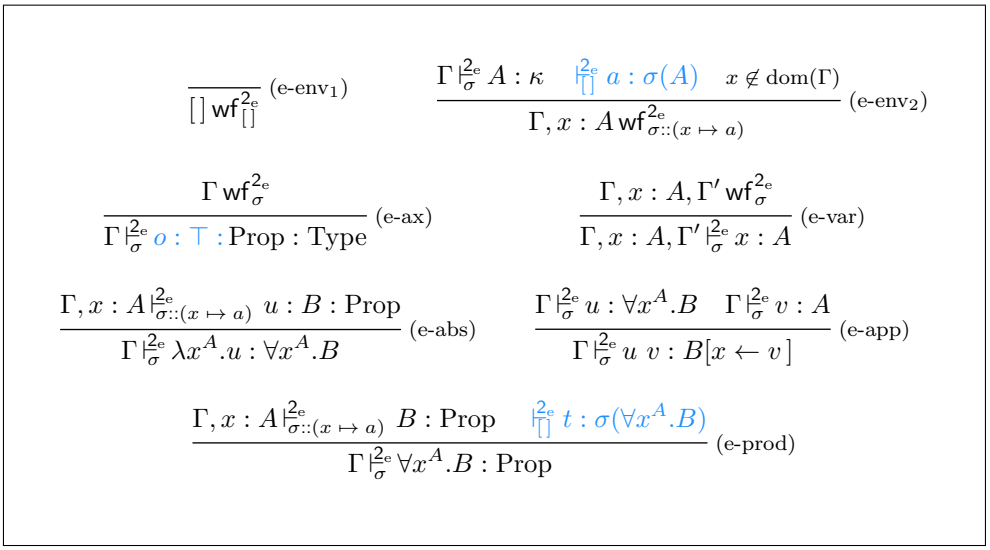


Figure 3: Inference rules of λ^2_e.

(see lem. 17). This last fact is important for explicit motivations systems: if $\Gamma \vdash^{2_e}_{\sigma} A : \kappa$ does not permit us to build an example a of $\sigma(A)$ then it means the *motivabilty*, and consequently the usability, of the type A has not been tested soon enough.

Remark 11

Substitutions and environments related by wf^{2_e} or $\vdash^{2_e}$ match: they have the same size, and to each variable of the environment correspond a raw term at the same position in the substitution (see lem. 13).

The constants o and $\top$, the initial examples, are mandatory to begin derivations: otherwise one would only be allowed to derive $[\,]\,\mathsf{wf}^{2_e}_{[\,]}$ and $\vdash^{2_e}_{[\,]} \mathrm{Prop} : \mathrm{Type}$.

In this section we show that λ^2_e *almost* satisfies the required properties of a pedagogical subsystem of CC: indeed in λ^2_e judgements and raw-terms are modified with respect to those of λ^2 and then CC.

3.2 Preliminary results

The properties $1, 2, 3, 4, 5, 6, 8, 9$ are still valid for λ^2_e, modulo the addition of the corresponding explicit motivations.

Theorem 12 (λ^2_e is a subsystem of λ^2)

(i) if $\Gamma\,\mathsf{wf}^{2_e}_{\sigma}$, then $\Gamma\,\mathsf{wf}^{2}$;

(ii) if $\Gamma \vdash^{2_e}_{\sigma} w : C$, then $\Gamma \vdash^{2} w : C$.

Proof immediate by structural induction on the derivation: it is enough to "forget" explicit motivations and to interpret in λ^2 the constants o and $\top$ of λ^2_e by, respectively, $\lambda A^{\mathrm{Prop}}.\lambda x^A.x$ and $\forall A^{\mathrm{Prop}}.A \to A$. □

Lemma 13 If $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{2_e}_\sigma$ or $x_1 : A_1, \ldots, x_n : A_n \vdash^{2_e}_\sigma w : C$ where $\sigma \equiv (y_1 \mapsto t_1) :: \ldots :: (y_m \mapsto t_m)$ then $m = n$, and for all i $x_i \equiv y_i$ and t_i is closed.

Lemma 14 (generation) If $\Gamma \vdash^{2_e}_\sigma t : T$ then one of these cases holds:
(i) if $t \equiv o$, then $T \equiv \top$;
(ii) if $t \equiv \top$, then $T \equiv \mathrm{Prop}$;
(iii) if $t \equiv \mathrm{Prop}$, then $T \equiv \mathrm{Type}$;
(iv) if $t \equiv x$, then there is $(x : A) \in \Gamma$ with $T \equiv A$;
(v) if $t \equiv \lambda x^A.u$, then there are B and a such that $\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} u : B : \mathrm{Prop}$ is a strict sub-derivation with $T \equiv \forall x^A.B$;
(vi) if $t \equiv u\ v$, then there are A and B such that $\Gamma \vdash^{2} u : \forall x^A.B$ and $\Gamma \vdash^{2} v : A$ are strict sub-derivations with $T \equiv B[x \leftarrow v\,]$;
(vii) if $t \equiv \forall x^A.B$, then there are a and t such that $\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} B : \mathrm{Prop}$ and $\vdash^{2_e}_{[]} t : \sigma(\forall x^A.B)$ are strict sub-derivations with $T \equiv \mathrm{Prop}$.

Lemma 15
(i) If $\Gamma \vdash^{2_e}_\sigma C : \mathrm{Type}$ then $C \equiv \mathrm{Prop}$ and the last derivation rule is (e-ax);
(ii) If $\Gamma \vdash^{2_e}_\sigma C : \mathrm{Prop}$ then the last derivation rule is (e-ax) or (e-var) or (e-prod).

Proof by case analysis on the last used rule, similar to the proof for λ^2 (prop. 7): to show that a derivation is impossible for λ^2_e, it is enough to notice that λ^2_e is a subsystem of λ^2 (thm. 12) and that the corresponding derivation is already impossible in λ^2. □

3.3 Results concerning pedagogy

Theorem 16 (λ^2_e meets the Poincaré criterion)
If $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{2_e}_\sigma$, then for all i $\vdash^{2_e}_{[]} \sigma(x_i) : \sigma(A_i)$ are **strict sub-derivations**.

Proof by structural induction on the derivation of $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{2_e}_\sigma$:

$$\textbf{(e-env}_\mathbf{2}\textbf{)}\ \frac{\Gamma \vdash^{2_e}_\sigma A : \kappa \quad \vdash^{2_e}_{[]} a : \sigma(A) \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A \,\mathsf{wf}^{2_e}_{\sigma::(x \mapsto a)}}$$

From $\Gamma \vdash^{2_e}_\sigma A : \kappa$ we know that $\Gamma \,\mathsf{wf}^{2_e}_\sigma$ is a strict sub-derivation (prop. 3), hence by induction hypothesis, with $\Gamma := y_1 : B_1, \ldots, y_n : B_n$, we have $\vdash^{2_e}_{[]} \sigma(y_i) : \sigma(B_i)$ are strict sub-derivations of $\Gamma \vdash^{2_e}_\sigma A : \kappa$. The second premise allows us to conclude for x.

□

Lemma 17 If $\Gamma \vdash^{2_e}_\sigma C : \kappa$, then there is a term t such that $\vdash^{2_e}_{[]} t : \sigma(C)$.

Proof by structural induction on the derivation. The only rules to consider are (e-ax), (e-prod) and (e-var) (lem. 15), and only the (e-var) case is non-trivial:

$$\textbf{(e-var)}\ \frac{\Gamma, x : \mathrm{Prop}, \Gamma' \,\mathsf{wf}^{2_e}_{\sigma}}{\Gamma, x : \mathrm{Prop}, \Gamma' \mid\!\!\!\sim^{2_e}_{\sigma} x : \mathrm{Prop}}$$

By the Poincaré criterion (thm. 16) applied to the premise, $\mid\!\!\!\sim^{2_e}_{[]} \sigma(x) : \mathrm{Prop}$ is a strict sub-derivation. Hence by induction hypothesis there is a term t such that $\mid\!\!\!\sim^{2_e}_{[]} t : \sigma(x)$.

□

Lemma 18 (weakening) If $\Gamma \mid\!\!\!\sim^{2_e}_{\sigma} w : C, \Gamma' \,\mathsf{wf}^{2_e}_{\sigma'}, \Gamma \subseteq \Gamma'$ and $\sigma \subseteq \sigma'$, then $\Gamma' \mid\!\!\!\sim^{2_e}_{\sigma'} w : C$.

Proof by structural induction on the derivation:

$$\textbf{(e-abs)}\ \frac{\Gamma, x : A \mid\!\!\!\sim^{2_e}_{\sigma::(x \mapsto a)} u : B : \mathrm{Prop}}{\Gamma \mid\!\!\!\sim^{2_e}_{\sigma} \lambda x^A.u : \forall x^A.B}$$ Let $\Gamma' \,\mathsf{wf}^{2_e}_{\sigma'}$ with $\Gamma \subseteq \Gamma'$ and $\sigma \subseteq \sigma'$.

From one premise we have that $\Gamma \mid\!\!\!\sim^{2_e}_{\sigma} A : \kappa$ is a sub-derivation (prop. 3, 4), on which we can apply induction hypothesis to get $\Gamma' \mid\!\!\!\sim^{2_e}_{\sigma'} A : \kappa$ and since also $\mid\!\!\!\sim^{2_e}_{[]} a : \sigma(A)$ (thm. 16) hence $\mid\!\!\!\sim^{2_e}_{[]} a : \sigma'(A)$ then finally by the rule (e-env$_2$) we have $\Gamma', x : A \,\mathsf{wf}^{2_e}_{\sigma'::(x \mapsto a)}$. The induction hypothesis applied on the premises gives $\Gamma', x : A \mid\!\!\!\sim^{2_e}_{\sigma'::(x \mapsto a)} u : B : \mathrm{Prop}$ and the (e-abs) rule finishes the proof.

(e-prod) Just as for the (e-abs) rule to be able to apply induction hypothesis.

□

Lemma 19 If $\mid\!\!\!\sim^{2_e}_{[]} w : C : \kappa$ and $z \notin \mathrm{dom}(\Gamma)$ then:
(i) if $\Gamma[z \leftarrow w] \,\mathsf{wf}^{2_e}_{\sigma}$ then $z : C, \Gamma \,\mathsf{wf}^{2_e}_{(z \mapsto w)::\sigma}$;
(ii) if $\Gamma[z \leftarrow w] \mid\!\!\!\sim^{2_e}_{\sigma} D[z \leftarrow w] : \kappa'$ then $z : C, \Gamma \mid\!\!\!\sim^{2_e}_{(z \mapsto w)::\sigma} D : \kappa'$.

Proof by structural induction on the derivation:
(i)

(e-env$_1$) From $\mid\!\!\!\sim^{2_e}_{[]} w : C : \kappa$ by (e-env$_2$) we have $z : C \,\mathsf{wf}^{2_e}_{[(z \mapsto w)]}$.

$$\textbf{(e-env}_2\textbf{)}\ \frac{\Gamma[z \leftarrow w] \mid\!\!\!\sim^{2_e}_{\sigma} A[z \leftarrow w] : \kappa'' \quad \mid\!\!\!\sim^{2_e}_{[]} a : \sigma(A[z \leftarrow w]) \quad x \notin \mathrm{dom}(\Gamma[z \leftarrow w])}{\Gamma[z \leftarrow w], x : A[z \leftarrow w] \,\mathsf{wf}^{2_e}_{\sigma::(x \mapsto a)}}$$

By induction hypothesis $z : C, \Gamma \mid\!\!\!\sim^{2_e}_{(z \mapsto w)::\sigma} A : \kappa''$ and also the second premise can be rewritten as $\mid\!\!\!\sim^{2_e}_{[]} a : (z \mapsto w)::\sigma(A)$ since w is closed and $z \notin \mathrm{dom}(\sigma)$ (lem. 13), then by (e-env$_2$) we get the result.

(ii) The case where $D \equiv z$ can be processed in the following way:

From $\Gamma[z \leftarrow w] \mid\!\!\!\sim^{2_e}_{\sigma} D[z \leftarrow w] : \kappa'$ it follows that $\Gamma[z \leftarrow w] \,\mathsf{wf}^{2_e}_{\sigma}$ is a strict sub-derivation (prop. 3), then by induction hypothesis $z : C, \Gamma \,\mathsf{wf}^{2_e}_{(z \mapsto w)::\sigma}$ and using the (e-var) rule $z : C, \Gamma \mid\!\!\!\sim^{2_e}_{(z \mapsto w)::\sigma} z : C$. Also $C \equiv \kappa'$ by type uniqueness (prop. 5) since:

- we have $\Gamma[z \leftarrow w] \mid\!\!\!\sim^{2_e}_{\sigma} w : \kappa'$ by hypothesis;

- from $\vdash^{2_e}_{[]} w : C$ we get $\Gamma[z \leftarrow w] \vdash^{2_e}_{\sigma} w : C$ by weakening (lem. 18).

Let us now deal with the cases where $D \not\equiv z$, we only need to consider the rules (e-ax), (e-var) and (e-prod) (lem. 15):

$$\textbf{(e-ax)}\ \frac{\Gamma[z \leftarrow w]\ \mathsf{wf}^{2_e}_{\sigma}}{\Gamma[z \leftarrow w] \vdash^{2_e}_{\sigma} \top : \mathrm{Prop}} \quad \text{with } D[z \leftarrow w] \equiv \top \text{ and } D \not\equiv z, \text{ hence } D \equiv \top.$$

By induction hypothesis $z : C, \Gamma\ \mathsf{wf}^{2_e}_{(z \mapsto w)::\sigma}$ and then using the (e-ax) rule we have $z : C, \Gamma \vdash^{2_e}_{(z \mapsto w)::\sigma} \top : \mathrm{Prop}$. We do the same for $\Gamma[z \leftarrow w] \vdash^{2_e}_{\sigma} \mathrm{Prop} : \mathrm{Type}$.

$$\textbf{(e-var)}\ \frac{\Gamma[z \leftarrow w], x : \kappa', \Gamma'[z \leftarrow w]\ \mathsf{wf}^{2_e}_{\sigma::(x \mapsto t)::\sigma'}}{\Gamma[z \leftarrow w], x : \kappa', \Gamma'[z \leftarrow w] \vdash^{2_e}_{\sigma::(x \mapsto t)::\sigma'} x : \kappa'} \quad \text{with } D[z \leftarrow w] \equiv x \text{ and } D \not\equiv z, \text{ hence } D \equiv x.$$

The induction hypothesis gives $z : C, \Gamma, x : \kappa', \Gamma'\ \mathsf{wf}^{2_e}_{(z \mapsto w)::\sigma::(x \mapsto t)::\sigma'}$ then the (e-var) rule finishes the proof.

$$\textbf{(e-prod)}\ \frac{\Gamma[z \leftarrow w], x : A[z \leftarrow w] \vdash^{2_e}_{\sigma::a} B[z \leftarrow w] : \mathrm{Prop} \quad \vdash^{2_e}_{[]} t : \sigma((\forall x^A.B)[z \leftarrow w])}{\Gamma[z \leftarrow w] \vdash^{2_e}_{\sigma} \forall x^{A[z \leftarrow w]}.B[z \leftarrow w] : \mathrm{Prop}}$$

By induction hypothesis $z : C, \Gamma, x : A \vdash^{2_e}_{(z \mapsto w)::\sigma::(x \mapsto a)} B : \mathrm{Prop}$, moreover the second premise can be rewritten to $\vdash^{2_e}_{[]} t : (z \mapsto w)::\sigma(\forall x^A.B)$ hence the result by (e-prod).

□

Theorem 20 (λ^2_e meets the converse of the Poincaré criterion)
If

$$\vdash^{2_e}_{[]} t_1 : A_1 : \kappa_1 \quad \ldots \quad \vdash^{2_e}_{[]} t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}] : \kappa_n$$

(with the x_i pairwise distinct), then

$$x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n\ \mathsf{wf}^{2_e}_{(x_1 \mapsto t_1)::(x_2 \mapsto t_2):: \ldots ::(x_n \mapsto t_n)}$$

Proof by induction on n:

By hypothesis $\vdash^{2_e}_{[]} A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}] : \kappa_n$ which can be rewritten to $\vdash^{2_e}_{[]} A_n[x_1, \ldots, x_{n-2} \leftarrow t_1, \ldots, t_{n-2}][x_{n-1} \leftarrow t_{n-1}] : \kappa_n$ since the x_i are pairwise distinct and the t_i are closed (lem. 13). We can then generalize over x_{n-1} (lem. 19) since we have $\vdash^{2_e}_{[]} t_{n-1} : A_{n-1}[x_1, \ldots, x_{n-2} \leftarrow t_1, \ldots, t_{n-2}] : \kappa_{n-1}$ in order to obtain $x_{n-1} : A_{n-1}[x_1, .., x_{n-2} \leftarrow t_1, .., t_{n-2}] \vdash^{2_e}_{(x_{n-1} \mapsto t_{n-1})} A_n[x_1, .., x_{n-2} \leftarrow t_1, .., t_{n-2}] : \kappa_n$.

Proceeding the same, we generalize over the variables from x_{n-2} to x_1 to finally obtain $x_1 : A_1, \ldots, x_{n-1} : A_{n-1} \vdash^{2_e}_{(x_1 \mapsto t_1)::\ldots::(x_{n-1} \mapsto t_{n-1})} A_n : \kappa_n$. Now since also $\vdash^{2_e}_{[]} t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}]$ then by (e-env$_2$) we finally get the result. □

Lemma 21
(i) If $z : C, \Gamma\ \mathsf{wf}^{2_e}_{(z \mapsto c)::\sigma}$ then $\Gamma[z \leftarrow c]\ \mathsf{wf}^{2_e}_{\sigma}$;

(ii) If $z : C, \Gamma \vdash^{2_e}_{(z \mapsto c)::\sigma} w : D$ then $\Gamma[z \leftarrow c] \vdash^{2_e}_{\sigma} w[z \leftarrow c] : D[z \leftarrow c]$.

Proof by structural induction on the derivation:

(e-var) $\dfrac{z : C, \Gamma \,\mathsf{wf}^{2_e}_{(z \mapsto c)::\sigma}}{z : C, \Gamma \vdash^{2_e}_{(z \mapsto c)::\sigma} z : C}$ is the only non-trivial case.

By induction hypothesis, we have $\Gamma[z \leftarrow c] \,\mathsf{wf}^{2_e}_{\sigma}$. And $\vdash^{2_e}_{[]} c : C$ by the Poincaré criterion (thm. 16), hence by weakening (lem. 18) we finally obtain $\Gamma[z \leftarrow c] \vdash^{2_e}_{\sigma} c : C$.

(e-app) $\dfrac{z : C, \Gamma \vdash^{2_e}_{(z \mapsto c)::\sigma} u : \forall x^A.B \quad z : C, \Gamma \vdash^{2_e}_{(z \mapsto c)::\sigma} v : A}{z : C, \Gamma \vdash^{2_e}_{(z \mapsto c)::\sigma} u\ v : B[x \leftarrow v]}$

By induction hypothesis, both $\Gamma[z \leftarrow c] \vdash^{2_e}_{\sigma} u[z \leftarrow c] : \forall x^{A[z \leftarrow c]}.B[z \leftarrow c]$ and $\Gamma[z \leftarrow c] \vdash^{2_e}_{\sigma} v[z \leftarrow c] : A[z \leftarrow c]$. Hence applying the (e-app) rule on those we have $\Gamma[z \leftarrow c] \vdash^{2_e}_{\sigma} u[z \leftarrow c]\ v[z \leftarrow c] : B[z \leftarrow c][x \leftarrow v[z \leftarrow c]]$, but since c is closed (lem. 13) then $B[z \leftarrow c][x \leftarrow v[z \leftarrow c]] \equiv B[x \leftarrow v][z \leftarrow c]$.

(e-prod) $\dfrac{z : C, \Gamma, x : A \vdash^{2_e}_{(z \mapsto c)::\sigma::(x \mapsto a)} B : \mathrm{Prop} \quad \vdash^{2_e}_{[]} t : (z \mapsto c)::\sigma(\forall x^A.B)}{z : C, \Gamma \vdash^{2_e}_{(z \mapsto c)::\sigma} \forall x^A.B : \mathrm{Prop}}$

By induction hypothesis $\Gamma[z \leftarrow c], x : A[z \leftarrow c] \vdash^{2_e}_{\sigma::(x \mapsto a)} B[z \leftarrow c] : \mathrm{Prop}$. And $(z \mapsto c)::\sigma(\forall x^A.B) \equiv \sigma((\forall x^A.B)[z \leftarrow c])$ since c is closed and $z \notin \mathrm{dom}(\sigma)$ (lem. 13). Therefore $\vdash^{2_e}_{[]} t : \sigma(\forall x^{A[z \leftarrow c]}.B[z \leftarrow c])$ and the (e-prod) rule allows us to conclude.

□

Lemma 22 If $\Gamma \vdash^{2_e}_{\sigma} w : C$ then $\vdash^{2_e}_{[]} \sigma(w) : \sigma(C)$.

Proof by induction on the size of the environment:

Let $\Gamma := x_1 : A_1, \ldots, x_n : A_n$ and $\sigma := (x_1 \mapsto t_1)::\ldots::(x_n \mapsto t_n)$. We have $\vdash^{2_e}_{[]} w[x_1 \leftarrow t_1] \ldots [x_n \leftarrow t_n] : C[x_1 \leftarrow t_1] \ldots [x_n \leftarrow t_n]$ after n substitutions of the motivations (lem. 21). And since the t_i are closed and the x_i are pairwise distinct (lem. 13) then $w[x_1 \leftarrow t_1] \ldots [x_n \leftarrow t_n] \equiv w[x_1, \ldots, x_n \leftarrow t_1, \ldots, t_n] \equiv \sigma(w)$ and $C[x_1 \leftarrow t_1] \ldots [x_n \leftarrow t_n] \equiv C[x_1, \ldots, x_n \leftarrow t_1, \ldots, t_n] \equiv \sigma(C)$. □

Lemma 23 If $\Gamma, z : C, \Gamma' \vdash^{2_e}_{\sigma} w : D$ and $z \notin \mathcal{V}(\Gamma', w)$, then $z \notin \mathcal{V}(D)$.

Proof immediate by structural induction on the derivation. □

Lemma 24 (strengthening)

(i) If $\Gamma, z : C, \Gamma' \,\mathsf{wf}^{2_e}_{\sigma::(z \mapsto c)::\sigma'}$ and $z \notin \mathcal{V}(\Gamma')$, then $\Gamma, \Gamma' \,\mathsf{wf}^{2_e}_{\sigma::\sigma'}$;

(ii) If $\Gamma, z : C, \Gamma' \vdash^{2_e}_{\sigma::(z \mapsto c)::\sigma'} w : D$ and $z \notin \mathcal{V}(\Gamma', w)$, then $\Gamma, \Gamma' \vdash^{2_e}_{\sigma::\sigma'} w : D$.

Proof by structural induction on the derivation, similar to [26, lem. 3.2.9]. The only non-immediate case is the following one:

(e-abs) $\dfrac{\Gamma, z : C, \Gamma', x : A \vdash^{2_e}_{\sigma::(z \mapsto c)::\sigma'::(x \mapsto a)} u : B : \mathrm{Prop}}{\Gamma, z : C, \Gamma' \vdash^{2_e}_{\sigma::(z \mapsto c)::\sigma'} \lambda x^A.u : \forall x^A.B}$ with $z \notin \mathcal{V}(\Gamma', \lambda x^A.u)$.

We have $z \notin \mathcal{V}(\Gamma', A, u)$, therefore also $z \notin \mathcal{V}(B)$ (lem. 23). We can then apply the induction hypothesis to get $\Gamma, \Gamma', x : A \vdash^{2_e}_{\sigma::\sigma'::(x \mapsto a)} u : B : \mathrm{Prop}$ and by (e-abs) the result.

□

Lemma 25 If $\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} u : B : \mathrm{Prop}$, then $\Gamma \vdash^{2_e}_{\sigma} \lambda x^A.u : \forall x^A.B : \mathrm{Prop}$.

Proof By (e-abs) on the hypotheses we have $\Gamma \vdash^{2_e}_{\sigma} \lambda x^A.u : \forall x^A.B$, so we obtain $\vdash^{2_e}_{[]} \sigma(\lambda x^A.u) : \sigma(\forall x^A.B)$ (lem. 22) which allows us to apply the (e-prod) and conclude.
□

Lemma 26 If $\Gamma\, \mathsf{wf}^{2_e}_{\sigma}$ and $\vdash^{2_e}_{[]} c : \sigma(C) : \kappa$ with $z \notin \mathrm{dom}(\Gamma)$, then $\Gamma, z : C\, \mathsf{wf}^{2_e}_{\sigma::(z \mapsto c)}$.

Proof Let $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$ and $\sigma \equiv (x_1 \mapsto a_1)::\ldots::(x_n \mapsto a_n)$.

By the Poincaré criterion (thm. 16) we have the derivations

$$\vdash^{2_e}_{[]} a_1 : A_1 \quad \vdash^{2_e}_{[]} a_2 : A_2[x_1 \leftarrow a_1] \quad \ldots \quad \vdash^{2_e}_{[]} a_n : A_n[x_1, \ldots, x_{n-1} \leftarrow a_1, \ldots, a_{n-1}]$$

and since for all i $x_1 : A_1, \ldots, x_{i-1} : A_{i-1} \vdash^{2_e}_{\sigma_{<i}} A_i : \kappa_i$ (prop. 4) then by substitutions (lem. 22) $\vdash^{2_e}_{[]} A_i[x_1, \ldots, x_{i-1} \leftarrow a_1, \ldots, a_{i-1}] : \kappa_i$. The result then follows by applying the converse of the Poincaré criterion (thm. 20) on:

$$\vdash^{2_e}_{[]} a_1 : A_1 : \kappa_1 \quad \ldots \quad \vdash^{2_e}_{[]} a_n : A_n[x_1, \ldots, x_{n-1} \leftarrow a_1, \ldots, a_{n-1}] : \kappa_n$$

$$\vdash^{2_e}_{[]} c : C[x_1, \ldots, x_n \leftarrow a_1, \ldots, a_n] : \kappa$$

□

Lemma 27 (replacement of equivalents)

If $\Gamma \vdash^{2_e}_{\sigma} w : E[z_1, \ldots, z_n \leftarrow C_1, \ldots, C_n] : \mathrm{Prop}$ and there are terms $(f_i)_{1 \le i \le n}$ and $(g_i)_{1 \le i \le n}$ such that for all i

$$\begin{array}{l} \Gamma \vdash^{2_e}_{\sigma} f_i : C_i \to D_i \\ \Gamma \vdash^{2_e}_{\sigma} g_i : D_i \to C_i \end{array} \quad \text{and} \quad \begin{array}{l} \Gamma \vdash^{2_e}_{\sigma} C_i : \mathrm{Prop} \\ \Gamma \vdash^{2_e}_{\sigma} D_i : \mathrm{Prop} \end{array}$$

then there is a term w' such that $\Gamma \vdash^{2_e}_{\sigma} w' : E[z_1, \ldots, z_n \leftarrow D_1, \ldots, D_n] : \mathrm{Prop}$.

Proof by induction on the raw term E (generalize [6, lem. 14]):

Let us first notice that if $E \equiv z_i$, then $w' := f_i\ w$ suits. Now let us deal with the cases when E is different from all the z_i. We proceed by case analysis on the last used rule producing $\Gamma \vdash^{2_e}_{\sigma} E[z_1, \ldots, z_n \leftarrow C_1, \ldots, C_n] : \mathrm{Prop}$, which limits the analysis to three rules (lem. 15):

(e-ax) In this case $E \equiv \top$ and then $w' := w$ suits.

(e-var) In this case $E \equiv y$ is a variable different from the z_i and then $w' := w$ suits.

(e-prod) Let $F[\vec{z} \leftarrow \vec{C}\,]$ abbreviates $F[z_1, \ldots, z_n \leftarrow C_1, \ldots, C_n\,]$:

$$\frac{\Gamma, x : A[\vec{z} \leftarrow \vec{C}\,] \mathrel{|\!\!\sim^{2_e}_{\sigma::(x \mapsto a)}} B[\vec{z} \leftarrow \vec{C}\,] : \mathrm{Prop} \quad \mathrel{|\!\!\sim^{2_e}_{[]}} t : \sigma(\forall x^{A[\vec{z} \leftarrow \vec{C}\,]}.B[\vec{z} \leftarrow \vec{C}\,])}{\Gamma \mathrel{|\!\!\sim^{2_e}_{\sigma}} \forall x^{A[\vec{z} \leftarrow \vec{C}\,]}.B[\vec{z} \leftarrow \vec{C}\,] : \mathrm{Prop}}$$

Since $\Gamma \mathrel{|\!\!\sim^{2_e}_{\sigma}} A[\vec{z} \leftarrow \vec{C}\,] : \kappa$ (prop. 3, 4), we distinguish two cases depending on κ:

- $\kappa \equiv \mathrm{Type}$: then $A[\vec{z} \leftarrow \vec{C}\,] \equiv \mathrm{Prop}$ (lem. 15). If $A \equiv z_i$ then $\Gamma \mathrel{|\!\!\sim^{2_e}_{\sigma}} C_i : \mathrm{Type}$, which is not allowed by type uniqueness (prop. 5). Necessarily $A \not\equiv z_i$ for all i and then $A \equiv \mathrm{Prop}$. The rule can then be rewritten in the following simpler way:

$$\frac{\Gamma, x : \mathrm{Prop} \mathrel{|\!\!\sim^{2_e}_{\sigma::(x \mapsto a)}} B[\vec{z} \leftarrow \vec{C}\,] : \mathrm{Prop} \quad \mathrel{|\!\!\sim^{2_e}_{[]}} t : \sigma(\forall x^{\mathrm{Prop}}.B[\vec{z} \leftarrow \vec{C}\,])}{\Gamma \mathrel{|\!\!\sim^{2_e}_{\sigma}} \forall x^{\mathrm{Prop}}.B[\vec{z} \leftarrow \vec{C}\,] : \mathrm{Prop}}$$

  Weakening (lem. 18) with $\Gamma, x : \mathrm{Prop}\ \mathsf{wf}^{2_e}_{\sigma::(x \mapsto a)}$ (prop. 3) on the derivations $\Gamma \mathrel{|\!\!\sim^{2_e}_{\sigma}} w : \forall x^{\mathrm{Prop}}.B[\vec{z} \leftarrow \vec{C}\,] : \mathrm{Prop}$, we get $\Gamma, x : \mathrm{Prop} \mathrel{|\!\!\sim^{2_e}_{\sigma::(x \mapsto a)}} w : \forall x^{\mathrm{Prop}}.B[\vec{z} \leftarrow \vec{C}\,] : \mathrm{Prop}$. Then using (e-var) and (e-app): $\Gamma, x : \mathrm{Prop} \mathrel{|\!\!\sim^{2_e}_{\sigma::(x \mapsto a)}} w\ x : B[\vec{z} \leftarrow \vec{C}\,]$. Now since $\Gamma, x : \mathrm{Prop} \mathrel{|\!\!\sim^{2_e}_{\sigma::(x \mapsto a)}} B[\vec{z} \leftarrow \vec{C}\,] : \mathrm{Prop}$ then by induction hypothesis there is a term u such that $\Gamma, x : \mathrm{Prop} \mathrel{|\!\!\sim^{2_e}_{\sigma::(x \mapsto a)}} u : B[\vec{z} \leftarrow \vec{D}\,] : \mathrm{Prop}$. Hence $\Gamma \mathrel{|\!\!\sim^{2_e}_{\sigma}} \lambda x^{\mathrm{Prop}}.u : \forall x^{\mathrm{Prop}}.B[\vec{z} \leftarrow \vec{D}\,] : \mathrm{Prop}$ (lem. 25), namely $w' := \lambda x^{\mathrm{Prop}}.u$ suits.

- $\kappa \equiv \mathrm{Prop}$: then $A[\vec{z} \leftarrow \vec{C}\,] \not\equiv \mathrm{Prop}$ (lem. 14) and $x \notin \mathcal{V}(B[\vec{z} \leftarrow \vec{C}\,])$ (prop. 8). From the first premise, we get $\mathrel{|\!\!\sim^{2_e}_{[]}} a : \sigma(A[\vec{z} \leftarrow \vec{C}\,]) : \mathrm{Prop}$ (thm. 16 and lem. 22) which can be rewritten to $\mathrel{|\!\!\sim^{2_e}_{[]}} a : A[\vec{z}, \vec{y} \leftarrow \sigma(\vec{C}), \sigma(\vec{y})\,] : \mathrm{Prop}$ with $\vec{y}$ denoting the free variables of $A[\vec{z} \leftarrow \vec{C}\,]$. Now since we have (lem. 22):

$$\begin{array}{ll} \mathrel{|\!\!\sim^{2_e}_{[]}} \sigma(f_i) : \sigma(C_i) \to \sigma(D_i) & \mathrel{|\!\!\sim^{2_e}_{[]}} \sigma(C_i) : \mathrm{Prop} \\ \mathrel{|\!\!\sim^{2_e}_{[]}} \sigma(g_i) : \sigma(D_i) \to \sigma(C_i) & \mathrel{|\!\!\sim^{2_e}_{[]}} \sigma(D_i) : \mathrm{Prop} \end{array}$$

  and also (prop. 3, 8 and lem. 22):

$$\mathrel{|\!\!\sim^{2_e}_{[]}} \sigma(y_i) : \mathrm{Prop} \qquad \mathrel{|\!\!\sim^{2_e}_{[]}} \lambda z^{\sigma(y_i)}.z : \sigma(y_i) \to \sigma(y_i)$$

  we can then apply the induction hypothesis on A to build a term a' such that $\mathrel{|\!\!\sim^{2_e}_{[]}} a' : A[\vec{z}, \vec{y} \leftarrow \sigma(\vec{D}), \sigma(\vec{y})\,] : \mathrm{Prop}$, namely $\mathrel{|\!\!\sim^{2_e}_{[]}} a' : \sigma(A[\vec{z} \leftarrow \vec{D}\,]) : \mathrm{Prop}$. And since $\Gamma\, \mathsf{wf}^{2_e}_{\sigma}$ (prop. 3), we then have $\Gamma, x : A[\vec{z} \leftarrow \vec{D}\,]\, \mathsf{wf}^{2_e}_{\sigma::(x \mapsto a')}$ (lem. 26).

  Therefore by (e-var) we have $\Gamma, x : A[\vec{z} \leftarrow \vec{D}\,] \mathrel{|\!\!\sim^{2_e}_{\sigma::(x \mapsto a')}} x : A[\vec{z} \leftarrow \vec{D}\,]$ and also $\Gamma, x : A[\vec{z} \leftarrow \vec{D}\,] \mathrel{|\!\!\sim^{2_e}_{\sigma::(x \mapsto a')}} A[\vec{z} \leftarrow \vec{D}\,] : \mathrm{Prop}$ (prop. 4, lem. 14, 15). Hence the induction hypothesis gives a term u such that $\Gamma, x : A[\vec{z} \leftarrow \vec{D}\,] \mathrel{|\!\!\sim^{2_e}_{\sigma::(x \mapsto a')}} u : A[\vec{z} \leftarrow \vec{C}\,] : \mathrm{Prop}$.

  By weakening (lem. 18) on the hypothesis and using the (e-app) rule we get $\Gamma, x : A[\vec{z} \leftarrow \vec{D}\,] \mathrel{|\!\!\sim^{2_e}_{\sigma::(x \mapsto a')}} w\ u : B[\vec{z} \leftarrow \vec{C}\,]$ and from the first premise $\Gamma, x : A[\vec{z} \leftarrow \vec{C}\,] \mathrel{|\!\!\sim^{2_e}_{\sigma::(x \mapsto a)}} B[\vec{z} \leftarrow \vec{C}\,] : \mathrm{Prop}$, then by strengthening (lem. 24) we can remove

x from the environment, and by weakening (lem. 18) with $x : A[\vec{z} \leftarrow \vec{D}\,]$ we get $\Gamma, x : A[\vec{z} \leftarrow \vec{D}\,] \mid^{2_e}_{\sigma::(x \mapsto a')} B[\vec{z} \leftarrow \vec{C}\,] : \mathrm{Prop}$. Hence by induction hypothesis we have a term v such that $\Gamma, x : A[\vec{z} \leftarrow \vec{D}\,] \mid^{2_e}_{\sigma::(x \mapsto a')} v : B[\vec{z} \leftarrow \vec{D}\,] : \mathrm{Prop}$ and finally $\Gamma \mid^{2_e}_{\sigma} \lambda x^{A[\vec{z} \leftarrow \vec{D}\,]}.v : A[\vec{z} \leftarrow \vec{D}\,] \to B[\vec{z} \leftarrow \vec{D}\,] : \mathrm{Prop}$ (lem. 25).

□

Lemma 28 If $\Gamma \mid^{2_e}_{\sigma} C : \mathrm{Prop}$, $\Gamma \mid^{2_e}_{\sigma} D : \mathrm{Prop}$ with C and D closed, then there are two terms f and g such that $\Gamma \mid^{2_e}_{\sigma} f : C \to D : \mathrm{Prop}$ and $\Gamma \mid^{2_e}_{\sigma} g : D \to C : \mathrm{Prop}$.

Proof Since C and D are closed, then by strengthening (lem. 24) $\mid^{2_e}_{[]} C : \mathrm{Prop}$ and $\mid^{2_e}_{[]} D : \mathrm{Prop}$ and there are terms u and v such that $\mid^{2_e}_{[]} u : C : \mathrm{Prop}$ and $\mid^{2_e}_{[]} v : D : \mathrm{Prop}$ (lem. 17). By (e-env$_2$) $z : C\, \mathsf{wf}^{2_e}_{(z \mapsto u)}$ and $z : D\, \mathsf{wf}^{2_e}_{(z \mapsto v)}$. Weakening (lem. 18) then gives $z : C \mid^{2_e}_{(z \mapsto u)} v : D : \mathrm{Prop}$ and $z : D \mid^{2_e}_{(z \mapsto v)} u : C : \mathrm{Prop}$. Simultaneous use of the (e-abs) and (e-prod) rules (lem. 25) gives $\mid^{2_e}_{[]} \lambda z^C.v : C \to D : \mathrm{Prop}$ and $\mid^{2_e}_{[]} \lambda z^D.u : D \to C : \mathrm{Prop}$. Finally by weakening (lem. 18) with $\Gamma\, \mathsf{wf}^{2_e}_{\sigma}$ (prop. 3) we obtain $\Gamma \mid^{2_e}_{\sigma} \lambda z^C.v : C \to D : \mathrm{Prop}$ and $\Gamma \mid^{2_e}_{\sigma} \lambda z^D.u : D \to C : \mathrm{Prop}$. □

Lemma 29 (motivations exchange) If $\Gamma \mid^{2_e}_{\sigma} w : C$ and $\Gamma\, \mathsf{wf}^{2_e}_{\sigma'}$, then $\Gamma \mid^{2_e}_{\sigma'} w : C$.

Proof by structural induction on the derivation of $\Gamma \mid^{2_e}_{\sigma} w : C$:

(e-abs) $$\frac{\Gamma, x : A \mid^{2_e}_{\sigma::(x \mapsto a)} u : B : \mathrm{Prop}}{\Gamma \mid^{2_e}_{\sigma} \lambda x^A.u : \forall x^A.B}$$

Since $\Gamma \mid^{2_e}_{\sigma} A : \kappa$ is a strict sub-derivation (prop. 3, 4), then by induction hypothesis $\Gamma \mid^{2_e}_{\sigma'} A : \kappa$. Hence we get a' such that $\Gamma, x : A\, \mathsf{wf}^{2_e}_{\sigma'::(x \mapsto a')}$ (lem. 17 and (e-env$_2$)). Now we can apply the induction hypothesis on the premises followed by an application of the (e-abs) rule to obtain the result.

(e-prod) $$\frac{\Gamma, x : A \mid^{2_e}_{\sigma::(x \mapsto a)} B : \mathrm{Prop} \quad \mid^{2_e}_{[]} t : \sigma(\forall x^A.B)}{\Gamma \mid^{2_e}_{\sigma} \forall x^A.B : \mathrm{Prop}}$$

As previously, we can start to show that $\Gamma, x : A\, \mathsf{wf}^{2_e}_{\sigma'::(x \mapsto a')}$ for some a'. Hence by induction hypothesis $\Gamma, x : A \mid^{2_e}_{\sigma'::(x \mapsto a')} B : \mathrm{Prop}$.

We can rewrite the second premise as $\mid^{2_e}_{[]} t : (\forall x^A.B)[y_1, \ldots, y_m \leftarrow \sigma(y_1), \ldots, \sigma(y_m)\,]$ where the y_i are the free variables of $\forall x^A.B$. Furthermore $(y_i : \mathrm{Prop}) \in \Gamma$ (prop. 8), then also $\mid^{2_e}_{[]} \sigma(y_i) : \mathrm{Prop}$ and $\mid^{2_e}_{[]} \sigma'(y_i) : \mathrm{Prop}$ (thm. 16).

Since the $\sigma(y_i)$ and the $\sigma'(y_i)$ are all closed (lem. 13), we then have terms f_i and g_i such that $\mid^{2_e}_{[]} f_i : \sigma'(y_i) \to \sigma(y_i)$ and $\mid^{2_e}_{[]} g_i : \sigma(y_i) \to \sigma'(y_i)$ (lem. 28). Hence replacing the equivalents (lem. 27) there is a term t' such that $\mid^{2_e}_{[]} t' : (\forall x^A.B)[y_1, \ldots, y_m \leftarrow \sigma'(y_1), \ldots, \sigma'(y_m)\,]$, namely $\mid^{2_e}_{[]} t' : \sigma'(\forall x^A.B)$. We are then allowed to conclude using the (e-prod) rule.

□

Lemma 30 (substitution lemma)

(i) If $\Gamma, y : C, \Gamma' \,\mathsf{wf}^{2_e}_{\sigma::(y \mapsto c)::\sigma'}$ and $\Gamma \vdash^{2_e}_{\sigma} w : C$, then there is a substitution ρ such that $\Gamma, \Gamma'[y \leftarrow w] \,\mathsf{wf}^{2_e}_{\sigma::\rho}$;

(ii) If $\Gamma, y : C, \Gamma' \vdash^{2_e}_{\sigma::(y \mapsto c)::\sigma'} d : D$ and $\Gamma \vdash^{2_e}_{\sigma} w : C$, then there is a substitution ρ such that $\Gamma, \Gamma'[y \leftarrow w] \vdash^{2_e}_{\sigma::\rho} d[y \leftarrow w] : D[y \leftarrow w]$.

Proof by structural induction on the first derivation:

(e-env$_2$) immediate by the induction hypothesis on the first premise followed by (e-env$_2$) and lem. 17.

(e-var) There are three cases depending on the position in the environment of the extracted variable: before y, being y or after y. They are solved as usual using the induction hypothesis, see [1, lem. 5.2.11]. The second case need an application of weakening (lem. 18) on $\Gamma \vdash^{2_e}_{\sigma} w : C$ in order to obtain $\Gamma, \Gamma'[y \leftarrow w] \vdash^{2_e}_{\sigma::\rho} w : C$.

$$\textbf{(e-abs)}\ \frac{\Gamma, y : C, \Gamma', x : A \vdash^{2_e}_{\sigma::(y \mapsto c)::\sigma'::(x \mapsto a)} u : B : \mathrm{Prop}}{\Gamma, y : C, \Gamma' \vdash^{2_e}_{\sigma::(y \mapsto c)::\sigma'} \lambda x^A.u : \forall x^A.B}$$

Induction hypothesis on the premises gives two substitutions ρ' and ρ'' such that

$$\Gamma, \Gamma'[y \leftarrow w], x : A[y \leftarrow w] \vdash^{2_e}_{\sigma::\rho'::(x \mapsto a')} u[y \leftarrow w] : B[y \leftarrow w]$$
$$\Gamma, \Gamma'[y \leftarrow w], x : A[y \leftarrow w] \vdash^{2_e}_{\sigma::\rho''::(x \mapsto a'')} B[y \leftarrow w] : \mathrm{Prop}$$

And we can exchange the motivation of the second one (lem. 29 and prop. 3) to obtain

$$\Gamma, \Gamma'[y \leftarrow w], x : A[y \leftarrow w] \vdash^{2_e}_{\sigma::\rho'::(x \mapsto a')} B[y \leftarrow w] : \mathrm{Prop}$$

Finally we get the result by applying the rule (e-abs) with $\rho := \rho'$.

(e-app) As previously, since the induction hypothesis applied to the two premises gives two substitutions ρ' and ρ'' potentially different, we chose one (lem. 29) and deduce the result by the rule (e-app).

$$\textbf{(e-prod)}\ \frac{\Gamma, y : C, \Gamma', x : A \vdash^{2_e}_{\sigma::(y \mapsto c)::\sigma'::(x \mapsto a)} B : \mathrm{Prop} \quad \vdash^{2_e}_{[]} t : \sigma::(y \mapsto c)::\sigma'(\forall x^A.B)}{\Gamma, y : C, \Gamma' \vdash^{2_e}_{\sigma::(y \mapsto c)::\sigma'} \forall x^A.B : \mathrm{Prop}}$$

First, by induction hypothesis, we have a substitution ρ' and a term a' such that

$$\Gamma, \Gamma'[y \leftarrow w], x : A[y \leftarrow w] \vdash^{2_e}_{\sigma::\rho'::(x \mapsto a')} B[y \leftarrow w] : \mathrm{Prop}$$

And transferring the motivation to the conclusion (lem. 22) and the second premise

$$\vdash^{2_e}_{[]} t : \sigma::(y \mapsto c)::\sigma'(\forall x^A.B) : \mathrm{Prop} \qquad (*)$$

Second since all free variable z of $\forall x^A.B$ are of type Prop (prop. 8) then:

- when $z \not\equiv y$: the Poincaré criterion (thm. 16) on the previous well-formed environments (prop. 3) gives us $\vdash^{2_e}_{[]} \sigma::(y \mapsto c)::\sigma'(z) : \mathrm{Prop}$ and $\vdash^{2_e}_{[]} \sigma::\rho'(z) : \mathrm{Prop}$;

- when $z \equiv y$: the Poincaré criterion (thm. 16) and the transfer of the motivation to the conclusion (lem. 22) gives us $\vdash^{2_e}_{[]} \sigma::(y \mapsto c)::\sigma'(z) : \mathrm{Prop}$ and $\vdash^{2_e}_{[]} \sigma(w) : \mathrm{Prop}$.

Since all those types are closed (prop. 1) they are equivalent (lem. 28), and we can then freely exchange them (lem. 27) in $(*)$ to build a term t' such that

$$\vdash^{2_e}_{[]} t' : \sigma::(y \mapsto \sigma(w))::\rho'(\forall x^A.B) : \mathrm{Prop} \quad \text{i.e.} \quad \vdash^{2_e}_{[]} t' : \sigma::\rho'((\forall x^A.B)[y \leftarrow w\,]) : \mathrm{Prop}$$

which allows us to conclude using (e-prod).

□

Theorem 31 (subject reduction) If $\Gamma \vdash^{2_e}_{\sigma} t : C$ and $t \rightsquigarrow_\beta t'$, then $\Gamma \vdash^{2_e}_{\sigma} t' : C$.

Proof by structural induction on the derivation followed by case analysis on the definition of $\rightsquigarrow_\beta$, similar to the one of [7, prop. 7] or [1, thm. 5.2.15]:

$$\textbf{(e-abs)}\ \frac{\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} u : B : \mathrm{Prop}}{\Gamma \vdash^{2_e}_{\sigma} \lambda x^A.u : \forall x^A.B}$$

A being in normal form (prop. 9), only the case $u \rightsquigarrow_\beta u'$ can happen: it is trivially solved using the induction hypothesis on the first premise.

$$\textbf{(e-app)}\ \frac{\Gamma \vdash^{2_e}_{\sigma} u : \forall x^A.B \quad \Gamma \vdash^{2_e}_{\sigma} v : A}{\Gamma \vdash^{2_e}_{\sigma} u\ v : B[x \leftarrow v\,]}$$

There are three cases:

- $u \rightsquigarrow_\beta u'$: trivial using the induction hypothesis on the first premise and (e-app).
- $v \rightsquigarrow_\beta v'$: there are three more cases (prop. 6):
  - $A \equiv \mathrm{Type}$: impossible (prop. 6, 2);
  - $\Gamma \vdash^{2_e}_{\sigma} A : \mathrm{Type}$: then $A \equiv \mathrm{Prop}$ (lem. 15) hence v is not reducible (prop. 9);
  - $\Gamma \vdash^{2_e}_{\sigma} A : \mathrm{Prop}$: then $A \not\equiv \mathrm{Prop}$ (lem. 14) and then $x \notin \mathcal{V}(B)$ (prop. 8) hence we have $B[x \leftarrow v'\,] \equiv B \equiv B[x \leftarrow v\,]$, and it is enough to apply the induction hypothesis on the second premise followed by (e-app).
- $u \equiv \lambda x^C.w$ and $u\ v \rightsquigarrow_\beta w[x \leftarrow v\,]$: generation (lem. 14) gives $\Gamma, x : A \vdash^{2_e}_{\sigma} w : B$ and by substitution (lem. 30) we have the result.

(e-prod) A term of type Prop is not reducible (prop. 9).

□

Lemma 32 (type correctness, see prop. 6**)**

If $\Gamma \vdash^{2_e}_{\sigma} w : C$ then $C \equiv \mathrm{Type}$ or there is κ such that $\Gamma \vdash^{2_e}_{\sigma} C : \kappa$.

Proof by structural induction on the derivation (similar to prop. 6): for the (e-abs) rule, the previous lemma 25 immediately gives us the result. □

Theorem 33 (λ^2_e is a *pseudo* pedagogical sub-system of CC)

λ_e^2 satisfies the following properties:
(i) λ_e^2 is a sub-system of CC;

(ii) If $\Gamma \vdash^{2e}_{\sigma} t : C$ and $t \rightsquigarrow_\beta t'$, then $\Gamma \vdash^{2e}_{\sigma} t' : C$.

(iii) $x_1 : A_1, .., x_n : A_n \; \mathsf{wf}^{2e}_{(x_1 \mapsto t_1)::..::(x_n \mapsto t_n)}$ **if and only if** $x_1 : A_1, .., x_n : A_n \; \mathsf{wf}^{\mathsf{c}}$ and

$$\vdash^{2e}_{[]} t_1 : A_1 \qquad \vdash^{2e}_{[]} t_2 : A_2[x_1 \leftarrow t_1] \quad \ldots \quad \vdash^{2e}_{[]} t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}]$$

Proof

(i) λ_e^2 is a sub-system of λ^2 (thm. 12), itself a sub-system of CC.

(ii) It is exactly the statement of the theorem 31.

(iii) $\Rightarrow$ It is exactly the statement of the theorem 16.

$\Leftarrow$ From $\vdash^{2e}_{[]} t_i : A_i[x_1, \ldots, x_{i-1} \leftarrow t_1, \ldots, t_{i-1}]$ and since $A_i \not\equiv \text{Type}$ because $x_1 : A_1, \ldots, x_n : A_n \; \mathsf{wf}^{\mathsf{c}}$ and $t_i \not\equiv \text{Type}$ (prop. 2), thanks to type correctness (lem. 32) we have $\vdash^{2e}_{[]} A_i[x_1, \ldots, x_{i-1} \leftarrow t_1, \ldots, t_{i-1}] : \kappa_i$ and we can then apply the theorem 20 to obtain the result.

□

4 Total motivations

In λ_e^2 examples has to be maintained during the whole proof: all premisses of rules use the same motivation. But we have seen that motivations can be exchanged (lem. 29): if $\Gamma \vdash^{2e}_{\sigma} w : C$ and $\Gamma \; \mathsf{wf}^{2e}_{\sigma'}$ then $\Gamma \vdash^{2e}_{\sigma'} w : C$. Hence we relax this constraint in the system λ_t^2 (fig. 4) and allow for different motivations to be used during sub-proofs. We then make the motivations *implicit* but still require them to completely exemplifies environments when needed. Leaving enhanced judgements leads us a step closer to a real pedagogical subsystem of CC (additional constants are maintained).

4.1 System definition

The following definitions of motivations of an environment or a type depend on the formal system λ_t^2 (fig. 4). To solve the apparent circularity, we can break those definitions in two parts: first a convenient abbreviation needed for the definition of the system; and second an effective definition once the inference rules of the system have been stated.

Definition 34 (Motivation of an environment) A substitution σ *motivates* an environment $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$, abbreviated $\sigma \operatorname{mot} \Gamma$, if for all $i \vdash^{2t} \sigma(x_i) : \sigma(A_i)$.

Definition 35 (Motivation of a type) A substitution σ *motivate a type C relatively to an environment* Γ, abbreviated $\sigma \operatorname{mot}_\Gamma C$ if (i) $\sigma \operatorname{mot} \Gamma$ and (ii) there is a term t such that $\vdash^{2t} t : \sigma(C)$.

Depending on the context, $\sigma \operatorname{mot} \Gamma$ will denote the derivations $\vdash^{2t} \sigma(x_i) : \sigma(A_i)$, or the fact that the environment Γ can be motivated by σ. The same applies for the $\sigma \operatorname{mot}_\Gamma C$ notation too.

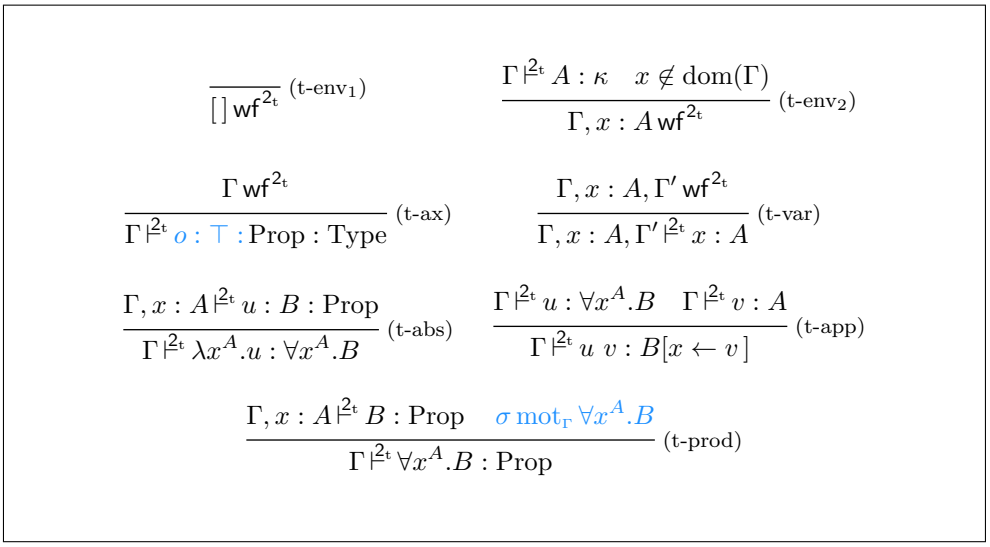


Figure 4: Inference rules of λ_t^2.

4.2 Results

The properties $1, 3, 4$ are still valid for λ_t^2.

Theorem 36 (λ_t^2 is a subsystem of λ^2)

(i) if $\Gamma\,\mathsf{wf}^{2_t}$ then $\Gamma\,\mathsf{wf}^{2}$;

(ii) if $\Gamma \vdash^{2_t} w : C$ then $\Gamma \vdash^{2} w : C$.

Lemma 37 (see lem. 15)

(i) If $\Gamma \vdash^{2_t} C : \mathrm{Type}$ then $C \equiv \mathrm{Prop}$ and the last rule of the derivation is (t-ax);

(ii) If $\Gamma \vdash^{2_t} C : \mathrm{Prop}$ then the last rule of the derivation is (t-ax), (t-var) or (t-prod).

Lemma 38 (λ_e^2 is a sub-system of λ_t^2) For every substitution σ:

(i) if $\Gamma\,\mathsf{wf}^{2_e}_{\sigma}$ then $\Gamma\,\mathsf{wf}^{2_t}$;

(ii) if $\Gamma \vdash^{2_e}_{\sigma} w : C$ then $\Gamma \vdash^{2_t} w : C$.

Proof by structural induction on the derivation. Every cases but (e-prod) are immediate: since we *forget* the explicit motivation, the rules are the same (or more constrained in the case of e-env$_2$) in λ_e^2.

$$\textbf{(e-prod)}\ \frac{\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} B : \mathrm{Prop} \quad \vdash^{2_e}_{[]} t : \sigma(\forall x^A.B)}{\Gamma \vdash^{2_e}_{\sigma} \forall x^A.B : \mathrm{Prop}}\ \text{with } \Gamma \equiv y_1 : D_1, \ldots, y_n : D_n.$$

By the first premise we have the sub-derivation $\Gamma\,\mathsf{wf}^{2_e}_{\sigma}$ (prop. 3) and the Poincaré criterion (thm. 16) gives $\vdash^{2_e}_{[]} \sigma(y_i) : \sigma(D_i)$ as strict sub-derivations, on which we can apply induction hypothesis to obtain $\vdash^{2_t} \sigma(y_i) : \sigma(D_i)$, namely $\sigma \,\mathrm{mot}\, \Gamma$. Moreover,

induction hypothesis applied on the second premise gives us $\vdash^{2_t} t : \sigma(\forall x^A.B)$ and we then get $\sigma \operatorname{mot}_\Gamma \forall x^A.B$. The induction hypothesis applied on the first premise and the (t-prod) rule allow us to conclude.

□

Lemma 39 (λ_t^2 is a sub-system of λ_e^2)

(i) if $\Gamma \,\mathsf{wf}^{2_t}$ then there is a substitution σ such that $\Gamma \,\mathsf{wf}^{2_e}_\sigma$;

(ii) if $\Gamma \vdash^{2_t} w : C$ then there is a substitution σ such that $\Gamma \vdash^{2_e}_\sigma w : C$.

Proof by structural induction on the derivation:

$$\textbf{(t-env}_\mathbf{2}\textbf{)}\ \frac{\Gamma \vdash^{2_t} A : \kappa \quad x \notin \operatorname{dom}(\Gamma)}{\Gamma, x : A \,\mathsf{wf}^{2_t}}$$

By induction hypothesis we have a substitution σ' such that $\Gamma \vdash^{2_e}_{\sigma'} A : \kappa$, and then (lem. 17) there is a term a such that $\vdash^{2_e}_{[\,]} a : \sigma'(A)$. Hence by (e-env$_2$) we obtain the result with $\sigma := \sigma' :: (x \mapsto a)$.

$$\textbf{(t-abs)}\ \frac{\Gamma, x : A \vdash^{2_t} u : B : \mathrm{Prop}}{\Gamma \vdash^{2_t} \lambda x^A.u : \forall x^A.B}$$

By induction hypothesis we have $\Gamma, x : A \vdash^{2_e}_{\sigma_1 :: (x \mapsto a_1)} u : B$ for a substitution σ_1 and a term a_1, and also $\Gamma, x : A \vdash^{2_e}_{\sigma_2 :: (x \mapsto a_2)} B : \mathrm{Prop}$ for σ_2 and a_2. Hence by exchange of motivations (prop. 3 and lem. 29) we also have $\Gamma, x : A \vdash^{2_e}_{\sigma_1 :: (x \mapsto a_1)} B : \mathrm{Prop}$ and finally the result by (e-abs).

(t-app) Performed as for (t-abs).

$$\textbf{(t-prod)}\ \frac{\Gamma, x : A \vdash^{2_t} B : \mathrm{Prop} \quad \sigma \operatorname{mot}_\Gamma \forall x^A.B}{\Gamma \vdash^{2_t} \forall x^A.B : \mathrm{Prop}}$$

In the following, (IH) will be the name of the induction hypothesis, which is applicable to every strict sub-derivation of $\Gamma \vdash^{2_t} \forall x^A.B : \mathrm{Prop}$.

Let $\Gamma \equiv y_1 : D_1, \ldots, y_n : D_n$. First we show by induction on i that

$$\forall i \quad y_1 : D_1, \ldots, y_i : D_i \,\mathsf{wf}^{2_e}_{\sigma_{\leq i}}$$

- $i = 0$: by (e-env$_1$) we have $[\,] \,\mathsf{wf}^{2_e}_{[\,]}$.
- Assume
$$y_1 : D_1, \ldots, y_i : D_i \,\mathsf{wf}^{2_e}_{\sigma_{\leq i}} \qquad \text{(IHi)}$$
By the definition of $\sigma \operatorname{mot}_\Gamma \forall x^A.B$ we have $\vdash^{2_t} \sigma(y_{i+1}) : \sigma(D_{i+1})$ as a sub-derivation, on which we can apply (IH) to obtain $\vdash^{2_e}_{[\,]} \sigma(y_{i+1}) : \sigma(D_{i+1})$. Since $y_1 : D_1, \ldots, y_i : D_i \vdash^{2_t} D_{i+1} : \kappa$ is a sub-derivation of the first premise (prop. 3, 4), using the induction hypothesis (IH) we can build a

substitution ρ such that $y_1 : D_1, \ldots, y_i : D_i \models^{2e}_{\rho} D_{i+1} : \kappa$, hence by motivations exchange (lem. 29) using (IHi) $y_1 : D_1, \ldots, y_i : D_i \models^{2e}_{\sigma_{\leq i}} D_{i+1} : \kappa$. We then transfer the motivation to the conclusion (lem. 22) to obtain $\models^{2e}_{[]} \sigma_{\leq i}(D_{i+1}) : \kappa$.

Finally since $y_1 : D_1, \ldots, y_i : D_i \,\mathsf{wf}^{2e}_{\sigma_{\leq i}}$ (IHi) and $\models^{2e}_{[]} \sigma(y_{i+1}) : \sigma_{\leq i}(D_{i+1}) : \kappa$, then $y_1 : D_1, \ldots, y_i : D_i, y_{i+1} : D_{i+1} \,\mathsf{wf}^{2e}_{\sigma_{\leq i+1}}$ (lem. 26) which closes this sub-proof.

Now, when $i = n$, we have $\Gamma \,\mathsf{wf}^{2e}_{\sigma}$. The induction hypothesis (IH) applied to the first premise gives ρ and a' such that $\Gamma, x : A \models^{2e}_{\rho::(x \mapsto a')} B : \mathrm{Prop}$. Hence $\Gamma \models^{2e}_{\rho} A : \kappa$ (prop. 3, 4), so $\Gamma \models^{2e}_{\sigma} A : \kappa$ by exchange of motivations (lem. 29), and there is a such that $\models^{2e}_{[]} a : \sigma(A)$ (lem. 17). Using (e-env$_2$) we have $\Gamma, x : A \,\mathsf{wf}^{2e}_{\sigma::(x \mapsto a)}$. By exchange of motivations (lem. 29) we then get $\Gamma, x : A \models^{2e}_{\sigma::(x \mapsto a)} B : \mathrm{Prop}$.

The definition of $\sigma \,\mathrm{mot}_{\Gamma}\, \forall x^A.B$ implies the existence of t such that $\models^{2t} t : \sigma(\forall x^A.B)$ is a sub-derivation on which we can apply (HI) to obtain $\models^{2e}_{[]} t : \sigma(\forall x^A.B)$. Finally the (e-prod) rule gives the result.

□

Theorem 40 (λ^2_t is a *pseudo* pedagogical sub-system of CC)

λ^2_t satisfies the following properties:

(i) λ^2_t is a sub-system of CC;

(ii) If $\Gamma \models^{2t} t : C$ and $t \rightsquigarrow_\beta t'$ then $\Gamma \models^{2t} t' : C$.

(iii) $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{2t}$ **if and only if** $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{\mathsf{c}}$ and there are terms $t_1, \ldots, t_n$ such that

$$\models^{2t} t_1 : A_1 \qquad \models^{2t} t_2 : A_2[x_1 \leftarrow t_1] \quad \ldots \quad \models^{2t} t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}]$$

Proof

(i) λ^2_t is a sub-system of λ^2 (thm. 36) itself a sub-system of CC.

(ii) From $\Gamma \models^{2t} t : C$ we have a substitution σ such that $\Gamma \models^{2e}_{\sigma} t : C$ (lem. 39) and since $t \rightsquigarrow_\beta t'$, then $\Gamma \models^{2e}_{\sigma} t' : C$ (thm. 33) hence $\Gamma \models^{2t} t' : C$ (lem. 38).

(iii) $\Rightarrow$ From $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{2t}$ we have $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{2e}_{\sigma}$ (lem. 39), hence $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{\mathsf{c}}$ and $\models^{2e}_{[]} \sigma(x_{i+1}) : \sigma(A_{i+1})$ (thm. 33) and finally $\models^{2t} \sigma(x_{i+1}) : \sigma(A_{i+1})$ (lem. 38).

$\Leftarrow$ Similarly we move back and forth from λ^2_t to λ^2_e (lem. 38, 39).

□

5 Partial motivations

In [3] we designed $\mathrm{CC_r}$ a subsystem of CC able to derive $\lambda A^{\mathrm{Prop}}.\lambda x^A.x$ of type $\forall A^{\mathrm{Prop}}.A \to A$, those two terms acting as initial examples like the constants o and $\top$ do for λ^2_e and λ^2_t (and P-Prop^2 of [6]). In $\mathrm{CC_r}$ the (c-prod) rule of CC is constrained

such that *every occurrences* of the formed type $\forall x^A.B$ has to be inhabited. In λ_e^2 and λ_t^2 only one occurrence need to be inhabited, but it has lead us to use motivations dealing with all the possible variables of the type to be motivated, namely all the variables of the environments, making the motivations total. In order to recover this behaviour of $\mathrm{CC_r}$ and remove the need for additional constants, we can make the motivations partial, that is allowing them to act on some variables of the environments.

5.1 System definition

As for λ_t^2 the following definitions of partial motivation of an environment or a type refer to the formal system λ_p^2 (fig. 5) and the apparent circularity can be circumvented in the same way.

Definition 41 (Application of a partial motivation)

The application of the substitution σ to the environment Γ, whose result is an environment abbreviated by $\sigma(\Gamma)$, is recursively defined as:

$$\sigma([\,]) := [\,]$$
$$\sigma(\Gamma, x : A) := \begin{cases} \sigma(\Gamma) & \text{if } x \in \mathrm{dom}(\sigma) \\ \sigma(\Gamma), x : \sigma(A) & \text{otherwise} \end{cases}$$

Definition 42 (Partial motivation of an environment) A substitution σ *partially motivates* the environment $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$, abbreviated $\sigma \widetilde{\mathrm{mot}}\ \Gamma$, if for all i $x_i \in \mathrm{dom}(\sigma) \Rightarrow \sigma(\Gamma_{<i}) \stackrel{2_\mathrm{p}}{\vdash} \sigma(x_i) : \sigma(A_i)$.

Definition 43 (Partial motivation of a type) A substitution σ *partially motivates a type* C relatively to an environment Γ, abbreviated $\sigma\ \widetilde{\mathrm{mot}}_\Gamma\ C$ if (i) $\sigma\ \widetilde{\mathrm{mot}}\ \Gamma$ and (ii) there is a term t such that $\sigma(\Gamma) \stackrel{2_\mathrm{p}}{\vdash} t : \sigma(C)$.

Depending on the context, $\sigma\ \widetilde{\mathrm{mot}}\ \Gamma$ will denote the previous derivations, or the fact that the environment Γ can be *partially motivated* by σ. The same applies for the $\sigma\ \widetilde{\mathrm{mot}}_\Gamma\ C$ notation.

Example 44 $\sigma := [x_2 \mapsto t_2, x_4 \mapsto t_4]$ partially motivates the type C relatively to $\Gamma := x_1 : A_1, \ldots, x_5 : A_5$ if:

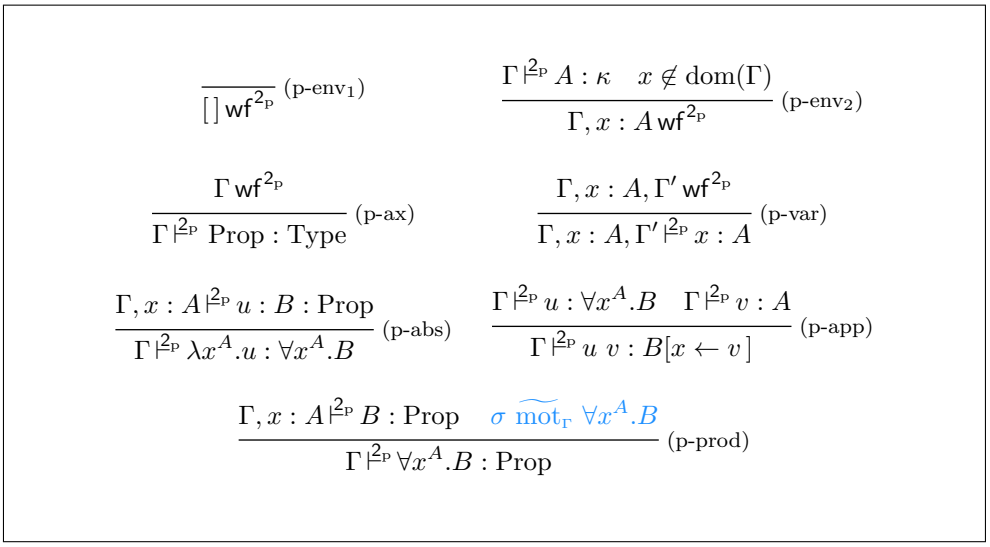
(i) $x_1 : A_1 \stackrel{2_\mathrm{p}}{\vdash} t_2 : A_2$ and $x_1 : A_1, x_3 : A_3[x_2 \leftarrow t_2\,] \stackrel{2_\mathrm{p}}{\vdash} t_4 : A_4[x_2 \leftarrow t_2\,]$;

(ii) there is t such that $x_1 : A_1, x_3 : A_3[x_2 \leftarrow t_2\,], x_5 : A_5[x_2, x_4 \leftarrow t_2, t_4\,] \stackrel{2_\mathrm{p}}{\vdash} t : \sigma(C)$.

Remark 45

When $\mathrm{dom}(\Gamma) \subseteq \mathrm{dom}(\sigma)$ we have the total motivation definition of λ_t^2. When $\mathrm{dom}(\sigma) = \emptyset$ the behaviour of $\mathrm{CC_r}$ is recovered.

For every environment Γ, $[\,]\ \widetilde{\mathrm{mot}}\ \Gamma$ holds. However, for a type C, we of course do not always have $[\,]\ \widetilde{\mathrm{mot}}_\Gamma\ C$.



$$\frac{}{[\,]\,\mathsf{wf}^{2_\mathrm{p}}}\ (\text{p-env}_1) \qquad \frac{\Gamma \vdash^{2_\mathrm{p}} A : \kappa \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^{2_\mathrm{p}}}\ (\text{p-env}_2)$$

$$\frac{\Gamma\,\mathsf{wf}^{2_\mathrm{p}}}{\Gamma \vdash^{2_\mathrm{p}} \mathrm{Prop} : \mathrm{Type}}\ (\text{p-ax}) \qquad \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}^{2_\mathrm{p}}}{\Gamma, x : A, \Gamma' \vdash^{2_\mathrm{p}} x : A}\ (\text{p-var})$$

$$\frac{\Gamma, x : A \vdash^{2_\mathrm{p}} u : B : \mathrm{Prop}}{\Gamma \vdash^{2_\mathrm{p}} \lambda x^A.u : \forall x^A.B}\ (\text{p-abs}) \qquad \frac{\Gamma \vdash^{2_\mathrm{p}} u : \forall x^A.B \quad \Gamma \vdash^{2_\mathrm{p}} v : A}{\Gamma \vdash^{2_\mathrm{p}} u\ v : B[x \leftarrow v\,]}\ (\text{p-app})$$

$$\frac{\Gamma, x : A \vdash^{2_\mathrm{p}} B : \mathrm{Prop} \quad \sigma\ \widetilde{\mathrm{mot}}_\mathrm{r}\ \forall x^A.B}{\Gamma \vdash^{2_\mathrm{p}} \forall x^A.B : \mathrm{Prop}}\ (\text{p-prod})$$

Figure 5: Inference rules of λ_p^2.

5.2 Results

In this section, we will identify the constants o and $\top$ of the previous systems λ_e^2 and λ_t^2 to their definitions in λ_p^2, namely $o := \lambda A^{\mathrm{Prop}}.\lambda x^A.x$ and $\top := \forall A^{\mathrm{Prop}}.A \to A$.

Lemma 46 We have the following derived rules:

$$\frac{\Gamma\,\mathsf{wf}^{2_\mathrm{p}}}{\Gamma \vdash^{2_\mathrm{p}} o : \top : \mathrm{Prop} : \mathrm{Type}}$$

Proof immediate by using an empty motivation whenever the (p-prod) rule is used (similar to the proof for CC_r in [3, sec. 3.4]) . □

Theorem 47 (λ_p^2 is a subsystem of λ^2)

(i) if $\Gamma\,\mathsf{wf}^{2_\mathrm{p}}$ then $\Gamma\,\mathsf{wf}^2$;

(ii) if $\Gamma \vdash^{2_\mathrm{p}} w : C$ then $\Gamma \vdash^2 w : C$.

Proof immediate by structural induction on the derivation. □

Lemma 48 (λ_t^2 is a subsystem of λ_p^2)

(i) if $\Gamma\,\mathsf{wf}^{2_\mathrm{t}}$ then $\Gamma\,\mathsf{wf}^{2_\mathrm{p}}$;

(ii) if $\Gamma \vdash^{2_\mathrm{t}} w : C$ then $\Gamma \vdash^{2_\mathrm{p}} w : C$.

Proof immediate by structural induction on the derivation:

- the (t-ax) case is done in the previous lemma 46;

- for the (t-prod) case, applying the induction hypothesis on all the derivations of $\sigma\,\mathrm{mot}_\Gamma\,\forall x^A.B$ is enough to obtain $\sigma\,\widetilde{\mathrm{mot}}_\Gamma\,\forall x^A.B$ and to conclude using (p-prod).

□

In order to prove the converse of the previous lemma, namely that λ_p^2 is a subsystem of λ_t^2, we will need to complete partial motivation to make them total. Therefore there is a need to define the substitution resulting of the composition of two substitutions:

Definition 49 (Composition of substitutions)

$\sigma \odot \rho$ is the *composition substitution* of the two substitutions σ and ρ defined by:

$$\sigma \odot \rho := \rho^\sigma :: \sigma\backslash_{\mathrm{dom}(\rho)}$$

where

$$\begin{array}{rcl} [\,]^\sigma & := & [\,] \\ ((y \mapsto v) :: \tau)^\sigma & := & (y \mapsto \sigma(v)) :: \tau^\sigma \end{array}$$

and $\sigma\backslash_{\mathrm{dom}(\rho)}$ is σ where all $(x \mapsto v)$ such that $x \in \mathrm{dom}(\rho)$ are removed.

Lemma 50 For every raw term t and substitutions σ and ρ we have $\sigma \odot \rho(t) \equiv \sigma(\rho(t))$. Moreover $\mathrm{dom}(\sigma \odot \rho) = \mathrm{dom}(\sigma) \cup \mathrm{dom}(\rho)$.

Proof immediate by induction on the raw term t. □

Lemma 51 (λ_p^2 is a subsystem of λ_t^2)

(i) if $\Gamma\,\mathsf{wf}^{2_\mathrm{p}}$ then $\Gamma\,\mathsf{wf}^{2_\mathrm{t}}$;

(ii) if $\Gamma \vdash^{2_\mathrm{p}} w : C$ then $\Gamma \vdash^{2_\mathrm{t}} w : C$.

Proof by structural induction on the derivation:

$$\textbf{(p-prod)}\ \frac{\Gamma, x : A \vdash^{2_\mathrm{p}} B : \mathrm{Prop} \quad \sigma\ \widetilde{\mathrm{mot}}_\Gamma\ \forall x^A.B}{\Gamma \vdash^{2_\mathrm{p}} \forall x^A.B : \mathrm{Prop}}\ \text{with } \Gamma \equiv y_1 : D_1, \ldots, y_n : D_n.$$

By the definition of $\sigma\ \widetilde{\mathrm{mot}}_\Gamma\ \forall x^A.B$, we have a term t such that $\sigma(\Gamma) \vdash^{2_\mathrm{p}} t : \sigma(\forall x^A.B)$ is a sub-derivation, and then by induction hypothesis $\sigma(\Gamma) \vdash^{2_\mathrm{t}} t : \sigma(\forall x^A.B)$. Hence there is a substitution ρ (lem. 39) such that

$$\sigma(\Gamma) \vdash_\rho^{2_\mathrm{e}} t : \sigma(\forall x^A.B) \qquad (*)$$

We then have $\rho \odot \sigma\,\mathrm{mot}_\Gamma\,\forall x^A.B$ since:

- if $y_i \in \mathrm{dom}(\sigma)$, by the definition of $\sigma\ \widetilde{\mathrm{mot}}_\Gamma\ \forall x^A.B$ we have $\sigma(\Gamma_{<i}) \vdash^{2_\mathrm{p}} \sigma(y_i) : \sigma(D_i)$ is a sub-derivation, and then by induction hypothesis $\sigma(\Gamma_{<i}) \vdash^{2_\mathrm{t}} \sigma(y_i) : \sigma(D_i)$. Hence there is ρ' such that $\sigma(\Gamma_{<i}) \vdash_{\rho'}^{2_\mathrm{e}} \sigma(y_i) : \sigma(D_i)$ (lem. 39) and then by exchange of motivations (lem. 29 and prop. 3) $\sigma(\Gamma_{<i}) \vdash_{\rho_{<i}}^{2_\mathrm{e}} \sigma(y_i) : \sigma(D_i)$. Then transferring the motivation to the conclusion (lem. 22) $\vdash_{[\,]}^{2_\mathrm{e}} \rho(\sigma(y_i)) : \rho(\sigma(D_i))$ and then also $\vdash^{2_\mathrm{t}} \rho \odot \sigma(y_i) : \rho \odot \sigma(D_i)$ (lem. 38, 50).

- if $y_i \notin \mathrm{dom}(\sigma)$ then $y_i \in \mathrm{dom}(\sigma(\Gamma))$, and then from $(*)$ using the Poincaré criterion (thm. 16 and prop. 3) $\vdash^{2_e}_{[]} \rho(y_i) : \rho(\sigma(D_i))$, namely, since $y_i \notin \mathrm{dom}(\sigma)$, $\vdash^{2_e}_{[]} \rho(\sigma(y_i)) : \rho(\sigma(D_i))$. Hence $\vdash^{2_t} \rho \odot \sigma(y_i) : \rho \odot \sigma(D_i)$ (lem. 38, 50).
- finally from $(*)$, transferring the motivation to the conclusion (lem. 22) we have $\vdash^{2_e}_{[]} \rho(t) : \rho(\sigma(\forall x^A.B))$. Hence $\vdash^{2_t} \rho(t) : \rho \odot \sigma(\forall x^A.B)$ (lem. 38, 50).

Thus the induction hypothesis applied to the first premise gives $\Gamma, x : A \vdash^{2_t} B : \mathrm{Prop}$ and the (t-prod) allows to conclude.

□

Theorem 52 (λ^2_p is a pedagogical sub-system of CC)

λ^2_p satisfies the following properties:
(i) λ^2_p is a subsystem of CC;

(ii) If $\Gamma \vdash^{2_P} t : C$ and $t \rightsquigarrow_\beta t'$, then $\Gamma \vdash^{2_P} t' : C$.

(iii) $x_1 : A_1, \ldots, x_n : A_n\ \mathsf{wf}^{2_P}$ **if and only if** $x_1 : A_1, \ldots, x_n : A_n\ \mathsf{wf}^{\mathsf{c}}$ and there are terms $t_1, \ldots, t_n$ such that

$$\vdash^{2_P} t_1 : A_1 \qquad \vdash^{2_P} t_2 : A_2[x_1 \leftarrow t_1\,] \quad \ldots \quad \vdash^{2_P} t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}\,]$$

Proof (i) holds since λ^2_p is a sub-system of λ^2 (thm. 47) itself a sub-system of CC. For (ii) and (iii) it is enough to notice that λ^2_t are λ^2_p equivalent (lem. 48, 51) in order to import the results of the former (thm. 40) into the later. □

Let us emphasize that λ^2_p is a pedagogical sub-system of CC in the sense of the formal definition given at the beginning (def. 10).

6 Pedagogical system F

λ^2_p is a pedagogical subsystem of CC, syntactically equivalent to the systems λ^2_e and λ^2_t (lem. 38, 39, 48, 51). In this section we link those systems with the second order pedagogical λ-calculus P-Prop^2 of [6]. First we recall the system P-Prop^2, then we show that it is equivalent to λ^2_t.

6.1 System definition

Definition 53 (Types of P-Prop^2) Types of P-Prop^2 are built according to the following rules: (i) $\top$ is a type; (ii) types variables $\alpha, \beta, \gamma, \ldots$ are types; (iii) if A and B are types then $A \rightarrow B$ is a type; (iv) if α is a type variable and A a type then $\forall \alpha.A$ is a type. The finite set of free variables of a type A, noted $\mathcal{V}(A)$, is defined in the usual way.

Definition 54 (Terms of P-Prop^2) Terms of P-Prop^2 are built according to the following rules: (i) $\boldsymbol{o}$ is a term; (ii) term variables $x, y, z, \ldots$ are terms; (iii) if x is a term variable, A a type and t a term then $\boldsymbol{\lambda} x^A.t$ is a term; (iv) if α is a type variable and t a term then $\boldsymbol{\Lambda}\alpha.t$ is a term; (v) if t and u are terms then $t\ u$ is a term; (vi) if t is a term and U a type then $t\ U$ is a term.

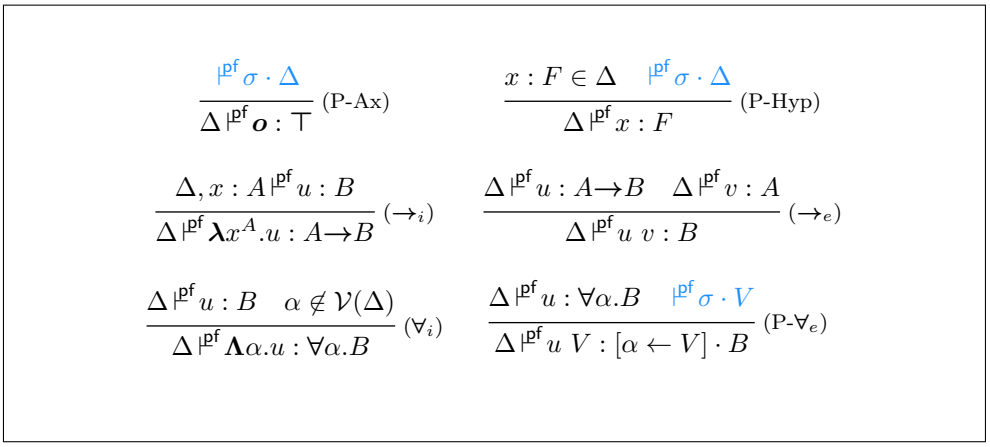


Figure 6: Inference rules of P-Prop^2.

Definition 55 (Substitutions of P-Prop^2) A substitution of P-Prop^2 is an application from type variables to types. The application of a substitution σ to a type A, defined in the usual way, is noted $\sigma \cdot A$. A constant substitution but in a finite number of points $\alpha_1, \ldots, \alpha_n$, associated respectively to the types $V_1, \ldots, V_n$, is noted $[\alpha_1, \ldots, \alpha_n \leftarrow V_1, \ldots, V_n]$.

Definition 56 (Contexts of P-Prop^2) A context Δ of P-Prop^2 is a finite set of couples $x : A$ where x is a term variable and A a type. Moreover if $x : A$ and $x : B$ are into the set Δ then $A = B$. The context $\{x_1 : A_1, \ldots, x_n : A_n\}$ is abbreviated to $x_1 : A_1, \ldots, x_n : A_n$. The set of free variables of a context $\Delta = x_1 : A_1, \ldots, x_n : A_n$, noted $\mathcal{V}(\Delta)$, is defined the usual way as the union of the $\mathcal{V}(A_i)$.

The following definitions of motivation refer to the formal system P-Prop^2 (fig. 6):

Definition 57 (Motivations of P-Prop^2) A substitution σ of P-Prop^2 *motivates* a type A, noted $\Vdash^{\mathsf{pf}} \sigma \cdot A$, if there is a term t such that $\Vdash^{\mathsf{pf}} t : \sigma \cdot A$. By extension, a substitution σ *motivate a context* $\Delta = x_1 : A_1, \ldots, x_n : A_n$, noted $\Vdash^{\mathsf{pf}} \sigma \cdot \Delta$, if for all i we have $\Vdash^{\mathsf{pf}} \sigma \cdot A_i$.

Remark 58 In P-Prop^2, and unlike λ_t^2, substitutions and contexts are set based, terms and types are disjoint, and a motivated type is not necessarily closed. Also since types are not built into the system P-Prop^2 every rules introducing new types need to be constrained (see fig. 6).

6.2 Results

Definition 59 (Translation from P-Prop^2 to λ_t^2)

Let $[\![\cdot]\!]$ be the translation from types and terms of P-Prop^2 to the raw terms of λ_t^2 defined by:

$$\begin{array}{rlrl} [\![\top]\!] &:= \top & [\![x]\!] &:= x \\ [\![\alpha]\!] &:= \alpha & [\![\boldsymbol{\lambda} x^A.t]\!] &:= \lambda x^{[\![A]\!]}.[\![t]\!] \\ [\![A{\rightarrow}B]\!] &:= [\![A]\!] \to [\![B]\!] & [\![\boldsymbol{\Lambda}\alpha.t]\!] &:= \lambda\alpha^{\mathrm{Prop}}.[\![t]\!] \\ [\![\forall\alpha.A]\!] &:= \forall\alpha^{\mathrm{Prop}}.[\![A]\!] & [\![t\ u]\!] &:= [\![t]\!]\ [\![u]\!] \\ [\![\boldsymbol{o}]\!] &:= o & [\![t\ U]\!] &:= [\![t]\!]\ [\![U]\!] \end{array}$$

where α is a type variable and x is a term variable.

Remark 60 We implicitly assumed that variables of λ_t^2 contains type and term variables of P-Prop^2.

Lemma 61 For all types A, B and all type variable α of P-Prop^2

$$[\![[\alpha \leftarrow B] \cdot A]\!] \equiv [\![A]\!][\alpha \leftarrow [\![B]\!]\,]$$

Proof by structural induction on the type A of P-Prop^2. □

Let us notice some results simplifying the extension of the translation of the contexts of P-Prop^2 to the environments of λ_t^2, in order to use the later like sets instead of lists:

Lemma 62 (exchange in λ_e^2) If $y \notin \mathcal{V}(D)$ then:
(i) If $\Gamma, y{:}C, z{:}D, \Gamma' \,\mathsf{wf}^{2_e}_{\sigma::(y\mapsto c)::(z\mapsto d)::\sigma'}$ then $\Gamma, z{:}D, y{:}C, \Gamma' \,\mathsf{wf}^{2_e}_{\sigma::(z\mapsto d)::(y\mapsto c)::\sigma'}$;
(ii) If $\Gamma, y{:}C, z{:}D, \Gamma' \vdash^{2_e}_{\sigma::(y\mapsto c)::(z\mapsto d)::\sigma'} w{:}E$ then $\Gamma, z{:}D, y{:}C, \Gamma' \vdash^{2_e}_{\sigma::(z\mapsto d)::(y\mapsto c)::\sigma'} w{:}E$.

Proof by structural induction on the derivation:

$$\textbf{(e-env}_\mathbf{2}\textbf{)}\ \frac{\Gamma, y : C \vdash^{2_e}_{\sigma::(y\,\mapsto\, c)} D : \kappa \quad \vdash^{2_e}_{[]} d : \sigma::(y \mapsto c)(D) \quad z \notin \mathrm{dom}(\Gamma, y)}{\Gamma, y : C, z : D \,\mathsf{wf}^{2_e}_{\sigma::(y\,\mapsto\, c)::(z\,\mapsto\, d)}}$$

Since $y \notin \mathcal{V}(D)$ by strengthening (lem. 24) on the first premise we get $\Gamma \vdash^{2_e}_{\sigma} D : \kappa$ then $\vdash^{2_e}_{[]} d : \sigma(D)$ (lem. 22). Hence by (e-env$_2$) we have $\Gamma, z : D \,\mathsf{wf}^{2_e}_{\sigma::(z\,\mapsto\, d)}$.

From the first premise we deduce $\Gamma \vdash^{2_e}_{\sigma} C : \kappa$ (prop. 4), which we can weaken (lem. 18) to obtain a derivation of $\Gamma, z : D \vdash^{2_e}_{\sigma::(z\,\mapsto\, d)} C : \kappa$.

Since $\vdash^{2_e}_{[]} c : \sigma(C)$ (thm. 16) and $z \notin \mathcal{V}(C)$ (lem. 13) then also $\vdash^{2_e}_{[]} c : \sigma::(z \mapsto d)(C)$, and by (e-env$_2$) we finally obtain the result $\Gamma, z : D, y : C \,\mathsf{wf}^{2_e}_{\sigma::(z\,\mapsto\, d)::(y\,\mapsto\, c)}$.

□

Lemma 63 (exchange in λ_t^2) If $y \notin \mathcal{V}(D)$ then:
(i) If $\Gamma, y : C, z : D, \Gamma' \,\mathsf{wf}^{2_t}$ then $\Gamma, z : D, y : C, \Gamma' \,\mathsf{wf}^{2_t}$;
(ii) If $\Gamma, y : C, z : D, \Gamma' \vdash^{2_t} w : E$ then $\Gamma, z : D, y : C, \Gamma' \vdash^{2_t} w : E$.

Proof immediate (lem. 62) since λ_t^2 and λ_e^2 are equivalents (lem. 38, 39). □

Lemma 64 If $\Gamma \vdash^{2_t} w : C$ then we can split Γ in two environments Γ_1 and Γ_2 such that: (i) Γ is a permutation of Γ_1, Γ_2; (ii) $\Gamma_1, \Gamma_2 \vdash^{2_t} w : C$; (iii) for all $y : D \in \Gamma_1$, $D \equiv \mathrm{Prop}$; (iv) for all $y : D \in \Gamma_2$, $D \not\equiv \mathrm{Prop}$.

Proof Let $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$. Since we have $x_1 : A_1, \ldots, x_i : A_i \vdash^{2_t} A_{i+1} : \kappa$ (prop. 4): either $\kappa \equiv \text{Type}$ and then $A_{i+1} \equiv \text{Prop}$ (lem. 37); or $\kappa \equiv \text{Prop}$ and then $A_{i+1} \not\equiv \text{Prop}$ (lem. 14, 39). We can then put all the $x_i : A_i$ where $A_i \equiv \text{Prop}$ in front of the environment (lem. 63) to constitute the Γ_1 part, the others constituting the Γ_2 part. □

Remark 65 The elements of Γ_1 can appear in any order (lem. 63). The same holds also for Γ_2 since the A_i only depend on the variables $x_j : \text{Prop}$ of Γ_1 (prop. 8 and lem. 39).

In the following, we will assume that the Γ_1 part of Γ in judgements $\Gamma\,\mathsf{wf}^{2_t}$ or $\Gamma \vdash^{2_t} w : C$ is implicit and then we will omit mentioning it. It can be reconstituted by putting in it every free variables of Γ, w and C. This is allowed by the properties of strengthening (lem. 24) and weakening (lem. 18) permitting us to add and remove elements of type Prop into Γ.

Those observations allow for a simpler definition of the translation of contexts of P-Prop^2 to environments of λ_t^2:

Definition 66 (Translation of a context of P-Prop^2)
The *translation of a context* of P-Prop^2 to an environment of λ_t^2 is defined by:

$$[\![x_1 : A_1, \ldots, x_n : A_n]\!] := x_1 : [\![A_1]\!], \ldots, x_n : [\![A_n]\!]$$

Lemma 67 (type correctness of λ_t^2) If $\Gamma \vdash^{2_t} w : C$, then $C \equiv \text{Type}$ or $\Gamma \vdash^{2_t} C : \kappa$.

Proof immediate: it already holds for λ_e^2 (lem. 32), equivalent to λ_t^2 (lem. 38, 39). □

Lemma 68 If $\Gamma \vdash^{2_t} w : [\![C]\!]$, then $\Gamma \vdash^{2_t} [\![C]\!] : \text{Prop}$.

Proof From $\Gamma \vdash^{2_t} w : [\![C]\!]$ we deduce that there is κ such that $\Gamma \vdash^{2_t} [\![C]\!] : \kappa$ (lem. 67). But if $\kappa \equiv \text{Type}$, then $[\![C]\!] \equiv \text{Prop}$ (lem. 37), which is not possible by the definition of $[\![\cdot]\!]$. As a consequence $\kappa \equiv \text{Prop}$. □

Definition 69 (Universal trivial motivation) The *universal trivial motivation* τ is the constant substitution associating $\top$ to every type variable.

Property 70 If $\Delta \Vdash^{\mathsf{pf}} u : F$ then for every sub-type G of Δ, F we have $\Vdash^{\mathsf{pf}} \tau \cdot G$.

Proof in [6, thm. 19]. □

Lemma 71 If $\Delta \Vdash^{\mathsf{pf}} u : F$ then there is a derivation of $\Delta \Vdash^{\mathsf{pf}} u : F$ using only the trivial motivation τ in the premise of the rules (P-Ax), (P-Hyp) and (P-$\forall_e$).

Proof by structural induction on the derivation. For each of the three rules, every motivated formulas appear as a sub-type of the conclusion sequent. Thus they are also motivable by τ (prop. 70). We can then replace everywhere the premise $\Vdash^{\mathsf{pf}} \sigma \cdot \Delta$ by $\Vdash^{\mathsf{pf}} \tau \cdot \Delta$. □

Lemma 72 If $\Delta \Vdash^{\mathsf{pf}} w : C$ is a derivation using only the trivial motivation τ, then $[\![\Delta]\!] \vdash^{2_t} [\![w]\!] : [\![C]\!]$.

Proof by structural induction on the derivation:

(P-Ax) $\dfrac{\vdash^{\mathsf{pf}} \tau \cdot \Delta}{\Delta \vdash^{\mathsf{pf}} \boldsymbol{o} : \top}$ where $\Delta = \{x_1 : A_1, \ldots, x_n : A_n\}$.

By hypothesis we have some terms t_i such that $\vdash^{\mathsf{pf}} t_i : \tau \cdot A_i$. Hence by induction hypothesis $\vdash^{2_t} [\![t_i]\!] : [\![\tau \cdot A_i]\!]$. But $[\![\tau \cdot A_i]\!] \equiv [\![A_i]\!][\vec{y} \leftarrow \top]$ (lem. 61) where $\vec{y}$ are the free variables of A_i. And since $\vdash^{2_t} [\![\tau \cdot A_i]\!] : \mathrm{Prop}$ (lem. 68) then by the reciprocal of the Poincaré criterion (thm. 40) $x_1 : [\![A_1]\!], \ldots, x_n : [\![A_n]\!]\ \mathsf{wf}^{2_t}$ and then by the (t-ax) rule we obtain the result.

(P-Hyp) $\dfrac{x : F \in \Delta \quad \vdash^{\mathsf{pf}} \tau \cdot \Delta}{\Delta \vdash^{\mathsf{pf}} x : F}$

As for (P-Ax), we show that $[\![\Delta]\!]\ \mathsf{wf}^{2_t}$. But since $x : F \in \Delta$ implies $x : [\![F]\!] \in [\![\Delta]\!]$, then by (t-var) we have $[\![\Delta]\!] \vdash^{2_t} x : [\![F]\!]$.

$(\to_i)$ $\dfrac{\Delta, x : A \vdash^{\mathsf{pf}} u : B}{\Delta \vdash^{\mathsf{pf}} \boldsymbol{\lambda} x^A.u : A \to B}$

By the induction hypothesis $[\![\Delta]\!], x : [\![A]\!] \vdash^{2_t} [\![u]\!] : [\![B]\!]$ and $[\![\Delta]\!], x : [\![A]\!] \vdash^{2_t} [\![B]\!] : \mathrm{Prop}$ (lem. 68). Hence by (t-abs) we obtain $[\![\Delta]\!] \vdash^{2_t} \lambda x^{[\![A]\!]}.[\![u]\!] : [\![A]\!] \to [\![B]\!]$ (because $x \notin \mathcal{V}(B)$ implies $x \notin \mathcal{V}([\![B]\!])$).

$(\to_e)$ $\dfrac{\Delta \vdash^{\mathsf{pf}} u : A \to B \quad \Delta \vdash^{\mathsf{pf}} v : A}{\Delta \vdash^{\mathsf{pf}} u\ v : B}$

It is enough to apply the induction hypothesis to the two premises and use (t-app).

$(\forall_i)$ $\dfrac{\Delta \vdash^{\mathsf{pf}} u : B \quad \alpha \notin \mathcal{V}(\Delta)}{\Delta \vdash^{\mathsf{pf}} \boldsymbol{\Lambda} \alpha.u : \forall \alpha.B}$

By induction hypothesis we have $[\![\Delta]\!] \vdash^{2_t} [\![u]\!] : [\![B]\!]$. There are two cases depending on whether $\alpha \in \mathcal{V}(B)$ or not:

- $\alpha \in \mathcal{V}(B)$: then $\alpha : \mathrm{Prop}$ is in the hidden implicit part of the translated environment, and since it does not appear in $\mathcal{V}(\Delta)$ it does not appear either in $\mathcal{V}([\![\Delta]\!])$. We can then *bubble up* $\alpha : \mathrm{Prop}$ in head position by successive permutations (lem. 63) to obtain $[\![\Delta]\!], \alpha : \mathrm{Prop} \vdash^{2_t} [\![u]\!] : [\![B]\!]$.
- $\alpha \notin \mathcal{V}(B)$: then α does not appear in the hidden part of the environment, and we can then add $\alpha : \mathrm{Prop}$ to $[\![\Delta]\!]$ by weakening (lem. 18, 38, 39) to obtain $[\![\Delta]\!], \alpha : \mathrm{Prop} \vdash^{2_t} [\![u]\!] : [\![B]\!]$.

In both cases we also have $[\![\Delta]\!], \alpha : \mathrm{Prop} \vdash^{2_t} [\![B]\!] : \mathrm{Prop}$ (lem. 68) and (t-abs) allows us to conclude.

(P-$\forall_e$) $\dfrac{\Delta \vdash^{\mathsf{pf}} u : \forall \alpha.B \quad \vdash^{\mathsf{pf}} \tau \cdot V}{\Delta \vdash^{\mathsf{pf}} u\ V : [\alpha \leftarrow V] \cdot B}$

As for (P-Ax) and (P-Hyp), from $\vdash^{\mathsf{pf}} \tau \cdot V$ we deduce $z : [\![V]\!]\,\mathsf{wf}^{2_t}$ where z is a fresh variable. We then get $[\![\Delta]\!] \vdash^{2_t} [\![V]\!] : \kappa$ (prop. 4) where $\kappa \equiv \mathrm{Prop}$ (otherwise $[\![V]\!] \equiv \mathrm{Prop}$ by prop. 8 and lem. 38, 39 which is impossible).

By the induction hypothesis $[\![\Delta]\!] \vdash^{2_t} [\![u]\!] : \forall \alpha^{\mathrm{Prop}}.[\![B]\!]$ and then $[\![\Delta]\!]\,\mathsf{wf}^{2_t}$ (prop. 3). Thus by weakening we also have $[\![\Delta]\!] \vdash^{2_t} [\![V]\!] : \mathrm{Prop}$ and then using the (t-app) rule $[\![\Delta]\!] \vdash^{2_t} [\![u]\!]\ [\![V]\!] : [\![B]\!][\alpha \leftarrow [\![V]\!]]$. But $[\![B]\!][\alpha \leftarrow [\![V]\!]] \equiv [\![[\alpha \leftarrow V] \cdot B]\!]$ (lem. 61).

□

Lemma 73 If $\Gamma \vdash^{2_t} w : C$ and $w \not\equiv \mathrm{Prop}$ then there is a term or a type w' of P-Prop^2 such that $[\![w']\!] \equiv w$.

Proof by structural induction on the derivation:

$$\textbf{(t-abs)}\ \frac{\Gamma, x : A \vdash^{2_t} u : B : \mathrm{Prop}}{\Gamma \vdash^{2_t} \lambda x^A.u : \forall x^A.B}$$

First the induction hypothesis gives us a term u' such that $[\![u']\!] \equiv u$. Then, since $\Gamma \vdash^{2_t} A : \kappa$ (prop. 4) is a sub-derivation, we are faced to two cases:

- if $\kappa \equiv \mathrm{Type}$, then $A \equiv \mathrm{Prop}$ (lem. 37) and in this case $w' := \mathbf{\Lambda} x.u'$ fits;
- if $\kappa \equiv \mathrm{Prop}$, then $A \not\equiv \mathrm{Prop}$, and we can apply the induction hypothesis to get a term A' such that $[\![A']\!] \equiv A$; hence $w' := \boldsymbol{\lambda} x^{A'}.u'$ fits.

$$\textbf{(t-prod)}\ \frac{\Gamma, x : A \vdash^{2_t} B : \mathrm{Prop} \quad \sigma\,\mathrm{mot}_\Gamma\,\forall x^A.B}{\Gamma \vdash^{2_t} \forall x^A.B : \mathrm{Prop}}$$

The induction hypothesis gives us a term B' such that $[\![B']\!] \equiv B$. We have to consider two cases in the sub-derivation $\Gamma \vdash^{2_t} A : \kappa$:

- if $\kappa \equiv \mathrm{Type}$, then $A \equiv \mathrm{Prop}$ (lem. 37) and in this case $w' := \forall x.B'$ fits;
- if $\kappa \equiv \mathrm{Prop}$, then $A \not\equiv \mathrm{Prop}$ and then $x \notin \mathcal{V}(B)$ (prop. 8), and the induction hypothesis gives us a term A' such that $[\![A']\!] \equiv A$, hence $w' := A' \to B'$ fits.

□

Corollary 74 If $y_1 : D_1, \ldots, y_n : D_n \vdash^{2_t} w : C$ then:
(i) if $D_i \not\equiv \mathrm{Prop}$ then there is a term or a type D_i' of P-Prop^2 such that $[\![D_i']\!] \equiv D_i$;
(ii) if $w \not\equiv \mathrm{Prop}$ then there is a term or a type w' of P-Prop^2 such that $[\![w']\!] \equiv w$;
(iii) if $C \not\equiv \mathrm{Prop}, \mathrm{Type}$ then there is a term or a type C' of P-Prop^2 such that $[\![C']\!] \equiv C$.

Proof

(i) From $y_1 : D_1, \ldots, y_i : D_i \vdash^{2_t} D_{i+1} : \kappa$ (prop. 4) we can distinguish two cases:

- if $\kappa \equiv \mathrm{Type}$, then $D_{i+1} \equiv \mathrm{Prop}$ (lem. 37);
- if $\kappa \equiv \mathrm{Prop}$, then $D_{i+1} \not\equiv \mathrm{Prop}$ and the lemma 73 finishes the proof.

(ii) This is exactly the lemma 73.

(iii) We have three cases (lem. 67):

- $C \equiv \text{Type}$: then the implication is valid by vacuity;
- $y_1 : D_1, \ldots, y_n : D_n \vdash^{2_t} C : \text{Type}$: then $C \equiv \text{Prop}$ (lem. 37);
- $y_1 : D_1, \ldots, y_n : D_n \vdash^{2_t} C : \text{Prop}$: then $C \not\equiv \text{Prop}$ and the lemma 73 concludes.

□

Lemma 75

(i) If $[\![\Delta]\!]\,\mathsf{wf}^{2_t}$ then there is a substitution ρ such that $\vDash^{\mathsf{pf}} \rho \cdot \Delta$;

(ii) If $[\![\Delta]\!] \vdash^{2_t} [\![C]\!] : \text{Prop}$ then there is a substitution ρ such that $\vDash^{\mathsf{pf}} \rho \cdot \Delta$ and $\vDash^{\mathsf{pf}} \rho \cdot C$;

(iii) If $[\![\Delta]\!] \vdash^{2_t} [\![w]\!] : [\![C]\!]$ then $\Delta \vDash^{\mathsf{pf}} w : C$.

Proof by structural induction on the derivation:

(t-env$_1$) $\dfrac{}{[\![\emptyset]\!]\,\mathsf{wf}^{2_t}}$ With ρ the empty substitution, we have trivially $\vDash^{\mathsf{pf}} \rho \cdot \emptyset$.

(t-env$_2$) $\dfrac{[\![\Delta]\!] \vdash^{2_t} A : \kappa \quad x \notin \mathrm{dom}([\![\Delta]\!])}{[\![\Delta]\!], x : A\,\mathsf{wf}^{2_t}}$

There are two cases:

- if $\kappa \equiv \text{Prop}$ then $A \not\equiv \text{Prop}$, and A is the image of some A' by $[\![\cdot]\!]$ (cor. 74); the induction hypothesis on the premise gives a substitution ρ satisfying $\vDash^{\mathsf{pf}} \rho \cdot (\Delta, A')$;
- if $\kappa \equiv \text{Type}$ then $A \equiv \text{Prop}$ (lem. 37) and $x : A$ is in the hidden part of the environment; but since $[\![\Delta]\!]\,\mathsf{wf}^{2_t}$ (prop. 3) is a sub-derivation, then the induction hypothesis gives a substitution ρ such that $\vDash^{\mathsf{pf}} \rho \cdot \Delta$.

(t-ax) $\dfrac{[\![\Delta]\!]\,\mathsf{wf}^{2_t}}{[\![\Delta]\!] \vdash^{2_t} [\![\boldsymbol{o}]\!] : [\![\top]\!] : \text{Prop}}$

By induction hypothesis on the premise we have a substitution ρ such that $\vDash^{\mathsf{pf}} \rho \cdot \Delta$, we can then derive $\Delta \vDash^{\mathsf{pf}} \boldsymbol{o} : \top$ by (P-Ax), and then we have also $\vDash^{\mathsf{pf}} \rho \cdot (\Delta, \top)$.

(t-var) $\dfrac{[\![\Delta, x : A, \Delta']\!]\,\mathsf{wf}^{2_t}}{[\![\Delta, x : A, \Delta']\!] \vdash^{2_t} [\![x]\!] : [\![A]\!]}$

By induction hypothesis we have a substitution ρ such that $\vDash^{\mathsf{pf}} \rho \cdot (\Delta, x : A, \Delta')$ and then by (P-Hyp) we get $\Delta, x : A, \Delta' \vDash^{\mathsf{pf}} x : A$.

(t-abs) Depending on the type of x, we are faced to one on those two cases:

$$\frac{[\![\Delta]\!], x : [\![A]\!] \vdash^{2_t} [\![u]\!] : [\![B]\!] : \text{Prop}}{[\![\Delta]\!] \vdash^{2_t} [\![\boldsymbol{\lambda} x^A.u]\!] : [\![A \to B]\!]} \qquad \frac{[\![\Delta]\!], x : \text{Prop} \vdash^{2_t} [\![u]\!] : [\![B]\!] : \text{Prop}}{[\![\Delta]\!] \vdash^{2_t} [\![\boldsymbol{\Lambda} x.u]\!] : [\![\forall x.B]\!]}$$

Each case can be easily solved using the induction hypothesis on the first premise and the $(\to_i)$ and $(\forall_i)$ rules (respectively).

(t-app) As previously, depending on the type of x we have two cases:

$$\frac{[\![\Delta]\!] \stackrel{2_t}{\vdash} [\![u]\!] : [\![A{\to}B]\!] \quad [\![\Delta]\!] \stackrel{2_t}{\vdash} [\![v]\!] : [\![A]\!]}{[\![\Delta]\!] \stackrel{2_t}{\vdash} [\![u\ v]\!] : [\![B]\!]} \qquad \frac{[\![\Delta]\!] \stackrel{2_t}{\vdash} [\![u]\!] : [\![\forall x.B]\!] \quad [\![\Delta]\!] \stackrel{2_t}{\vdash} [\![v]\!] : \mathrm{Prop}}{[\![\Delta]\!] \stackrel{2_t}{\vdash} [\![u\ v]\!] : [\![B]\!][x \leftarrow [\![v]\!]\,]}$$

The induction hypothesis and the $(\to_e)$ and $(\forall_e)$ rules (respectively) solve them.

(t-prod) Once again, depending on the type of x we have to deal with two cases:

- $\dfrac{[\![\Delta]\!], x : [\![A]\!] \stackrel{2_t}{\vdash} [\![B]\!] : \mathrm{Prop} \quad \sigma\,\mathrm{mot}_{[\![\Delta]\!]}[\![A{\to}B]\!]}{[\![\Delta]\!] \stackrel{2_t}{\vdash} [\![A{\to}B]\!] : \mathrm{Prop}}$ where $\Delta \equiv y_1 : D_1, \ldots, y_n : D_n$.

  By the definition of $\sigma\,\mathrm{mot}_{[\![\Delta]\!]}[\![A{\to}B]\!]$, we have:

  - terms t_i such that $\stackrel{2_t}{\vdash} t_i : [\![D_i]\!][\vec{\alpha} \leftarrow \vec{E}\,]$ where $\vec{\alpha}$ are the free variables of the D_i and then the E_j are such that $\stackrel{2_t}{\vdash} E_j : \mathrm{Prop}$ (prop. 8). We then have terms E'_j and t'_i such that $[\![E'_j]\!] \equiv E_j$ and $[\![t'_i]\!] \equiv t_i$ (cor. 74). Therefore $\stackrel{2_t}{\vdash} [\![t'_i]\!] : [\![D_i]\!][\vec{\alpha} \leftarrow [\![\vec{E'}]\!]\,]$ namely $\stackrel{2_t}{\vdash} [\![t'_i]\!] : [\![[\vec{\alpha} \leftarrow \vec{E'}\,] \cdot D_i]\!]$ (lem. 61). The induction hypothesis gives $\stackrel{\mathsf{pf}}{\vdash} t'_i : [\vec{\alpha} \leftarrow \vec{E'}\,] \cdot D_i$ namely $\stackrel{\mathsf{pf}}{\vdash} \rho \cdot D_i$ where $\rho := [\vec{\alpha} \leftarrow \vec{E'}\,]$.
  - and a term u such that $\stackrel{2_t}{\vdash} u : [\![A{\to}B]\!][\vec{\alpha} \leftarrow \vec{E}\,]$ which by the same way leads us to $\stackrel{\mathsf{pf}}{\vdash} \rho \cdot (A{\to}B)$.

- $\dfrac{[\![\Delta]\!], x : \mathrm{Prop} \stackrel{2_t}{\vdash} [\![B]\!] : \mathrm{Prop} \quad \sigma\,\mathrm{mot}_{[\![\Delta]\!]}[\![\forall x.B]\!]}{[\![\Delta]\!] \stackrel{2_t}{\vdash} [\![\forall x.B]\!] : \mathrm{Prop}}$ Solved as previously.

□

Theorem 76 $\Delta \stackrel{\mathsf{pf}}{\vdash} w : C$ if and only if $[\![\Delta]\!] \stackrel{2_t}{\vdash} [\![w]\!] : [\![C]\!]$.

Proof $\Rightarrow$ From $\Delta \stackrel{\mathsf{pf}}{\vdash} w : C$, we build a derivation using only τ as motivation (lem. 71), and then $[\![\Delta]\!] \stackrel{2_t}{\vdash} [\![w]\!] : [\![C]\!]$ (lem. 72).

$\Leftarrow$ It is exactly the (iii) of lemma 75 above. □

Corollary 77 We can embed the second order propositional calculus Prop^2 and λ^2 in the calculi λ^2_e, λ^2_t and λ^2_p.

Proof The next property 79 recalls an embedding from Prop^2 to P-Prop^2, which is enough because we can embed P-Prop^2 in λ^2_t (thm. 76), λ^2_t being equivalent to λ^2_e and λ^2_p (lem. 38, 39, 48, 51). Also λ^2 and Prop^2 are two different formalizations of the same calculus (can be shown similarly as what we did for λ^2_t and P-Prop^2). □

7 Type checking

In this section we show that for all the pedagogical type systems of second-order presented so far the so-called type-checking problem is not decidable. We use the fact that the type inhabitation problem for Prop^2 is not decidable. Prop^2 is P-Prop^2 without the constraints, also known as System F such as presented in [15].

Definition 78 (Type inhabitation) For a given formal system, *the type inhabitation problem* is:

input: a context (or an environment) Γ, and a type A;

output: "true" if there is a term t such that $\Gamma \vdash^{\star} t : A$, and "false" otherwise.

Property 79 The type inhabitation problem for Prop^2 can be reduced to the type inhabitation problem for P-Prop^2: for every Δ and A there is t such that $\Delta \vdash^{\mathsf{f}} t : A$ *if and only if* there is t' such that $\Delta^\gamma \vdash^{\mathsf{pf}} t' : A^\gamma$, where $^\gamma$ is a translation from formulas of Prop^2 to formulas of P-Prop^2.

Proof A (constructive) proof can be found in [5] about formal systems corresponding to the type systems Prop^2 and P-Prop^2. The translation $^\gamma$, inspired by the A-translation of [11], consists in replacing every occurrences of type variables α by $\alpha \vee \gamma$ where γ is a fresh type variable. □

Property 80 Type inhabitation for Prop^2 is undecidable.

Proof by Urzyczyn in [36]. □

Lemma 81 Type inhabitation for P-Prop^2 is undecidable.

Proof by contradiction. Assume that type inhabitation for P-Prop^2 can be decided by an algorithm D: $D(\Delta, A) = \text{true}$ *if and only if* there is a term t such that $\Delta \vdash^{\mathsf{pf}} t : A$. We can then build an algorithm D' able to decide the problem of type inhabitation for Prop^2: $D'(\Delta, A) := D(\Delta^\gamma, A^\gamma)$. Indeed:

$$\begin{array}{lll} D'(\Delta, A) = \text{true} & \textit{iff} & D(\Delta^\gamma, A^\gamma) = \text{true} \\ & \textit{iff} & \text{there is } t' \text{ such that } \Delta^\gamma \vdash^{\mathsf{pf}} t' : A^\gamma \\ & \textit{iff} & \text{there is } t \text{ such that } \Delta \vdash^{\mathsf{f}} t : A \qquad \text{(prop. 79)} \end{array}$$

But we noticed that the type inhabitation for Prop^2 is undecidable (prop. 80). □

Definition 82 (Type checking) For a given type system, *the problem of type checking* is:

input: a context (or an environment) Γ, a term t and a type A;

output: "true" if there is a derivation of $\Gamma \vdash^{\star} t : A$, and "false" otherwise.

Lemma 83 The type inhabitation problem for P-Prop^2 with an empty context can be reduced to the type checking problem for λ^2_t with an empty context: for every type A there is t such that $\vdash^{\mathsf{pf}} t : A$ with A closed *if and only if* $\vdash^{2_t} [\![A]\!] : \text{Prop}$.

Proof $\Rightarrow$ From $\vdash^{\mathsf{pf}} t : A$ we can deduce $\vdash^{2_t} [\![t]\!] : [\![A]\!]$ (thm. 76), and by type correctness (lem. 67) $\vdash^{2_t} [\![A]\!] : \kappa$. But $\kappa \not\equiv \text{Type}$ because otherwise $[\![A]\!] \equiv \text{Prop}$ (lem. 37) which is not possible by the definition of $[\![\cdot]\!]$, hence $\kappa \equiv \text{Prop}$.

$\Leftarrow$ From $\vdash^{2_t} [\![A]\!] : \mathrm{Prop}$ we can build a term a such that $\vdash^{2_t} a : [\![A]\!]$ (lem. 17, 38, 39). But a is the image of a term t by $[\![\cdot]\!]$ (cor. 74), i.e. $[\![t]\!] \equiv a$, hence $\vdash^{2_t} [\![t]\!] : [\![A]\!]$ and finally $\vdash^{f} t : A$ (thm. 76).

□

Lemma 84 The type inhabitation problem for P-Prop^2 can be reduced to the type inhabitation problem for P-Prop^2 with an empty context: for every type A there is t such that $\Delta \vdash^{\mathsf{pf}} t : A$ *if and only if* there is t' such that $\vdash^{\mathsf{pf}} t' : \forall\vec{\alpha}.\Delta{\to}A$, where $\forall\vec{\alpha}.\Delta{\to}A$ is closed, $\vec{\alpha}$ are the free variables of Δ and A, and $\Delta{\to}A$ denotes $B_1{\to}\ldots{\to}B_n{\to}A$ with $\Delta = \{y_1 : B_1, \ldots, y_n : B_n\}$.

Proof $\Rightarrow$ From $\Delta \vdash^{\mathsf{pf}} t : A$ we have $\vdash^{\mathsf{pf}} \boldsymbol{\lambda}\Delta.t : \Delta{\to}A$ using $(\to_i)$ and then $\vdash^{\mathsf{pf}} \boldsymbol{\Lambda}\vec{\alpha}.\boldsymbol{\lambda}\Delta.t : \forall\vec{\alpha}.\Delta{\to}A$ using $(\forall_i)$. So $t' := \boldsymbol{\Lambda}\vec{\alpha}.\boldsymbol{\lambda}\Delta.t$ fits.

$\Leftarrow$ Conversely from $\vdash^{\mathsf{pf}} t' : \forall\vec{\alpha}.\Delta{\to}A$ using $(\forall_e)$ we have $\vdash^{\mathsf{pf}} t'\ \vec{\alpha} : \Delta{\to}A$ since the $\vec{\alpha}$ are motivable $\top$, and then by weakening we have $\Delta \vdash^{\mathsf{pf}} t'\ \vec{\alpha} : \Delta{\to}A$ and finally using $(\to_e)$ we obtain $\Delta \vdash^{\mathsf{pf}} t'\ \vec{\alpha}\ \Delta : A$, namely $t := t'\ \vec{\alpha}\ \Delta$ fits.

Weakening for P-Prop^2 has been proved in [6, prop. 21] if the introduced formula can be motivated: here the formulas of Δ are all motivable by the trivial substitution τ since they appear as sub-formulas in $\forall\vec{\alpha}.\Delta{\to}A$ (prop. 70). □

Theorem 85 The type checking problem for λ_t^2 is undecidable.

Proof by contradiction. Let us assume that the type checking problem for λ_t^2 can be decided by an algorithm D: $D(\Gamma, t, A) = \text{true}$ *if and only if* $\Gamma \vdash^{2_t} t : A$. We can then build an algorithm D' to decide the type inhabitation problem for P-Prop^2: $D'(\Delta, A) := D([\,], [\![\forall\vec{\alpha}.\Delta{\to}A]\!], \mathrm{Prop})$ with $\vec{\alpha}$ the free variables of Δ and A. Indeed:

$$\begin{aligned} D'(\Delta, A) = \text{true} \text{ iff } & D([\,], [\![\forall\vec{\alpha}.\Delta{\to}A]\!], \mathrm{Prop}) = \text{true} \\ \text{iff } & \vdash^{2_t} [\![\forall\vec{\alpha}.\Delta{\to}A]\!] : \mathrm{Prop} \\ \text{iff there is } t \text{ such that } & \vdash^{\mathsf{pf}} t : \forall\vec{\alpha}.\Delta{\to}A \quad \text{(lem. 83)} \\ \text{iff there is } t' \text{ such that } & \Delta \vdash^{\mathsf{pf}} t' : A \quad \text{(lem. 84)} \end{aligned}$$

But the type inhabitation problem for P-Prop^2 is undecidable (lem. 81). □

Corollary 86 The type checking problem for λ_p^2 is undecidable.

Proof is an immediate consequence of the equivalence of λ_t^2 and λ_p^2 (lem. 38, 39). □

Definition 87 (Type checking with explicit motivations)

For a given type system with explicit motivations, *the type checking problem for explicit motivations* is the following:

input: a context (or environment) Γ, a substitution σ, a term t and a type A;

output: "true" if there is a derivation of $\Gamma \vdash^{\star}_{\sigma} t : A$, and "false" otherwise.

Theorem 88 The type checking problem for λ_e^2 is undecidable.

Proof by contradiction. Let us assume that the type checking problem for λ_e^2 can be decided by an algorithm D: $D(\Gamma, \sigma, t, A) = \text{true}$ *if and only if* $\Gamma \overset{2_e}{\underset{\sigma}{\vdash}} t : A$. We can then build an algorithm D' to decide the type checking problem for λ_t^2:

$$D'(\Gamma, t, A) := \begin{cases} D([\,], [\,], \forall\Gamma.\top, \text{Prop}) & \text{if } A \equiv \text{Type and } t \equiv \text{Prop} \\ \text{false} & \text{if } A \equiv \text{Type and } t \not\equiv \text{Prop} \\ \text{false} & \text{if } A \equiv \text{Prop and } t \equiv \text{Prop} \\ D([\,], [\,], \forall\Gamma.\forall z^t.\top, \text{Prop}) & \text{if } A \equiv \text{Prop and } t \not\equiv \text{Prop} \\ D([\,], [\,], \lambda\Gamma.t, \forall\Gamma.A) & \text{otherwise} \end{cases}$$

with $\lambda\Gamma.A \equiv \lambda y_1^{B_1}.\ldots.\lambda y_n^{B_n}.A$ if $\Gamma \equiv y_1 : B_1, \ldots, y_n : B_n$, and similarly for $\forall\Gamma.A$. First we show that $D([\,], [\,], \forall\Gamma.\top, \text{Prop}) = \text{true}$ *iff* there is σ such that $\Gamma\ \mathsf{wf}_\sigma^{2_e}$:

$\Rightarrow$ From $\overset{2_e}{\underset{[\,]}{\vdash}} \forall\Gamma.\top : \text{Prop}$ by generation (lem. 14) we obtain a substitution σ such that $\Gamma \overset{2_e}{\underset{\sigma}{\vdash}} \top : \kappa$, and finally (prop. 3) $\Gamma\ \mathsf{wf}_\sigma^{2_e}$.

$\Leftarrow$ From $\Gamma\ \mathsf{wf}_\sigma^{2_e}$ using (e-ax) we have $\Gamma \overset{2_e}{\underset{\sigma}{\vdash}} o : \top : \text{Prop}$ and then using (e-abs) and (e-prod) (lem. 25) $\overset{2_e}{\underset{[\,]}{\vdash}} \lambda\Gamma.o : \forall\Gamma.\top : \text{Prop}$, so $D([\,], [\,], \forall\Gamma.\top, \text{Prop}) = \text{true}$.

Now we can show that $D'(\Gamma, t, A) = \text{true}$ *iff* $\Gamma \overset{2_t}{\vdash} t : A$:

- $A \equiv \text{Type}$ and $t \equiv \text{Prop}$:

$$\begin{array}{lll} D'(\Gamma, t, A) = \text{true} & \textit{iff} & D([\,], [\,], \forall\Gamma.\top, \text{Prop}) = \text{true} \\ & \textit{iff} & \text{there is } \sigma\ \Gamma\ \mathsf{wf}_\sigma^{2_e} \\ & \textit{iff} & \text{there is } \sigma\ \Gamma \overset{2_e}{\underset{\sigma}{\vdash}} \text{Prop} : \text{Type} \quad \text{((e-ax) and prop. 3)} \\ & \textit{iff} & \Gamma \overset{2_t}{\vdash} \text{Prop} : \text{Type} \quad \text{(lem. 38, 39)} \end{array}$$

- $A \equiv \text{Type}$ and $t \not\equiv \text{Prop}$: $D'(\Gamma, t, A) = \text{false}$ and $\Gamma \not\overset{2_t}{\vdash} t : \text{Type}$ (lem. 15).
- $A \equiv \text{Prop}$ and $t \equiv \text{Prop}$: $D'(\Gamma, t, A) = \text{false}$ and $\Gamma \not\overset{2_t}{\vdash} \text{Prop} : \text{Prop}$ (lem. 14)
- $A \equiv \text{Prop}$ and $t \not\equiv \text{Prop}$:

$$\begin{array}{lll} D'(\Gamma, t, A) = \text{true} & & \\ \quad \textit{iff} & D([\,], [\,], \forall\Gamma.\forall z^t.\top, \text{Prop}) = \text{true} & \\ \quad \textit{iff} & \text{there are } \sigma \text{ and } w \quad \Gamma, z : t\ \mathsf{wf}_{\sigma::(z \mapsto w)}^{2_e} & \\ \quad \textit{iff} & \text{there is } \sigma\ \Gamma \overset{2_e}{\underset{\sigma}{\vdash}} t : \kappa & \text{(prop. 4, lem. 17, (e-env}_2\text{))} \\ \quad \textit{iff} & \text{there is } \sigma\ \Gamma \overset{2_e}{\underset{\sigma}{\vdash}} t : \text{Prop} & \text{(lem. 15)} \\ \quad \textit{iff} & \Gamma \overset{2_t}{\vdash} t : \text{Prop} & \text{(lem. 38, 39)} \end{array}$$

- $A \not\equiv \kappa$:

$$\begin{array}{lll} D'(\Gamma, t, A) = \text{true} & \textit{iff} & D([\,], [\,], \lambda\Gamma.t, \forall\Gamma.A) = \text{true} \\ & \textit{iff} & \overset{2_e}{\underset{[\,]}{\vdash}} \lambda\Gamma.t : \forall\Gamma.A \\ & \textit{iff} & \overset{2_e}{\underset{[\,]}{\vdash}} \lambda\Gamma.t : \forall\Gamma.A : \kappa' \quad \text{(lem. 32)} \\ & \textit{iff} & \overset{2_e}{\underset{[\,]}{\vdash}} \lambda\Gamma.t : \forall\Gamma.A : \text{Prop} \quad \text{(lem. 15)} \\ & \textit{iff} & \text{there is } \sigma\ \Gamma \overset{2_e}{\underset{\sigma}{\vdash}} t : A : \text{Prop} \quad \text{(lem. 14, 25)} \\ & \textit{iff} & \Gamma \overset{2_t}{\vdash} t : A \quad \text{(lem. 38, 39)} \end{array}$$

But the type checking problem for λ_t^2 is undecidable (thm. 85). □

8 Conclusion

In this paper, we have given an example of the formal definition of pedagogical subsystem of the Calculus of Constructions of [3] that we called λ_p^2, corresponding precisely to the pedagogical second-order λ-calculus of Colson and Michel [6]. Moreover the formalism of CC used in the definition allows for an homogeneous description of various type systems. For instance the introduced constraints for the second-order necessarily need to be transferred to higher orders pedagogical calculi; conversely once a pedagogical Calculus of Constructions will be obtained, pedagogical versions of the λ-cube systems should appear by deletion of some rules and simplification of associated constraints. Furthermore a pedagogical Calculus of Constructions can open the study toward pedagogical *pure type systems* [1]. Thus we believe the objective of giving a uniform formal handling of the study of formal pedagogy has been reached.

During the building of our system λ_p^2 we uncovered a formalism making explicit into the judgements the needed motivations, λ_e^2. This kind of formalism seems to be natural for expressing pedagogical calculi. Also it allows to state more precise and intuitive *meta*-mathematical properties about these systems. However we have shown it does not carry enough useful information to consider an implementation, especially because the type-checking is still undecidable.

As a conclusion, we suggest a simple solution to this problem: let us annotate types with terms to ensure their motivability, just like the typed λ-calculus annotate pure λ-terms with types to ensure their normalization. As an example we give modified rules (env_2) and (prod) implementing this (term annotation is at the bottom of types):

$$\frac{\Gamma_\sigma \vdash A_a : \kappa \quad x \notin \text{dom}(\Gamma)}{\Gamma_\sigma, x : A_a \,\mathsf{wf}}\ (\text{env}_2) \qquad \frac{\Gamma_\sigma, x : A_a \vdash B_b : \text{Prop} \quad \vdash t : \sigma(\forall x^{A_a}.B_b)}{\Gamma_\sigma \vdash (\forall x^{A_a}.B_b)_t : \text{Prop}}\ (\text{prod})$$

In such a formalism, terms should contain the needed information to allow the rebuild of the derivation and then type-checking.

References

[1] Henk Barendregt. *Lambda calculi with types*, volume 2 of *Handbook of Logic in Computer Science*, pages 117–309. Oxford University Press, 1992.

[2] M.W. Bunder and Jonathan P. Seldin. Variants of the Basic Calculus of Constructions. *Journal of Applied Logic*, 2(2):191–217, 2004.

[3] Loïc Colson and Vincent Demange. Investigations on a pedagogical calculus of constructions. *Journal of Universal Computer Science*, 19(6):729–749, 2013.

[4] Loïc Colson and David Michel. Pedagogical natural deduction systems: the propositional case. *Journal of Universal Computer Science*, 13(10):1396–1410, 2007.

[5] Loïc Colson and David Michel. Pedagogical Second-order Propositional Calculi. *Journal of Logic and Computation*, 18(4):669–695, 2008.

[6] Loïc Colson and David Michel. Pedagogical second-order λ-calculus. *Theoretical Computer Science*, 410:4190–4203, 2009.

[7] Thierry Coquand. *Une théorie des constructions*. PhD thesis, Université Paris VII, 31 January 1985.

[8] Thierry Coquand. An analysis of girard's paradox. In *Proceedings of the First Annual IEEE Symposium on Logic in Computer Science (LICS 1986)*, pages 227–236. IEEE Computer Society Press, June 1986.

[9] Thierry Coquand. Metamathematical investigations of a calculus of constructions. Technical Report 1088, INRIA, September 1989.

[10] Miriam Franchella. Brouwer and Griss on intuitionistic negation. *Modern Logic 4*, 3:256–265, 1994.

[11] H. Friedman. Classically and intuitionistically provably recursive functions. In Springer, editor, *Higher Set Theory*, volume 669, pages 21–27, 1978.

[12] P.C.G. Gilmore. The effect of Griss' criticism of the intuitionistic logic on deductive theories formalized within the intuitionistic logic. *Indagationes Mathematicæ*, 15:162–174, 175–186, 1953.

[13] J.-Y. Girard. *Interprétation fonctionnelle et élimination des coupures dans l'arithmétique d'ordre supérieur*. Thèse de doctorat d'état, Université Paris VII, 1972.

[14] Jean-Yves Girard. Le lambda-calcul du second ordre. In *Séminaire N. Bourbaki*, number 678, pages 173–185, February 1987.

[15] Jean-Yves Girard, Paul Taylor, and Yves Lafont. *Proofs and types*. Cambridge University Press, 1990.

[16] G.F.C. Griss. Negationless intuitionistic mathematics. *Indagationes Mathematicæ*, 8:675–681, 1946.

[17] G.F.C. Griss. Negationless intuitionistic mathematics II. *Indagationes Mathematicæ*, 12:108–115, 1950.

[18] G.F.C. Griss. Negationless intuitionistic mathematics III. *Indagationes Mathematicæ*, 13:193–199, 1951.

[19] G.F.C. Griss. Negationless intuitionistic mathematics IVa, IVb. *Indagationes Mathematicæ*, 13:452–462,463–471, 1951.

[20] Arendt Heyting. G. F. C. Griss and his negationless intuitionistic mathematics. *Synthese*, 9:91–96, 1955.

[21] William A. Howard. The formulas-as-types notion of construction. In J. P. Seldin and J. R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus, and Formalism*, pages 479–490. Academic Press, 1980.

[22] Victor N. Krivtsov. A Negationless Interpretation of Intuitionistic Theories. I. *Studia Logica*, 64(3):323–344, 2000.

[23] Victor N. Krivtsov. A Negationless Interpretation of Intuitionistic Theories. II. *Studia Logica*, 65(2):155–179, 2000.

[24] E. G. K. López-Escobar. Constructions and negationless logic. *Studia Logica*, 30(1):7–22, 1972.

[25] E. G. K. López-Escobar. Elementary interpretations of negationless arithmetic. *Fundamenta Mathematicae*, 82(1):25–38, 1974.

[26] Zhaohui Luo. *An Extended Calculus of Constructions*. PhD thesis, University of Edinburgh, 1990.

[27] V. Mezhlumbekova. Deductive capabilities of negationless intuitionistic arithmetic. *Moscow University Mathematical Bulletin*, 30(2), 1975.

[28] David Michel. *Systèmes formels et systèmes fonctionnels pédagogiques*. PhD thesis, Université Paul-Verlaine – Metz, 2008.

[29] John Kent Minichiello. An extension of negationless logic. *Notre Dame J. Formal Logic*, 10:298–302, 1969.

[30] Grigori Mints. Notes on Constructive Negation. *Synthese*, 148(3):701–717, February 2006.

[31] D. Nelson. A complete negationless system. *Studia Logica*, 32:41–49, 1973.

[32] David Nelson. Non-Null Implication. *The Journal of Symbolic Logic*, 31(4):562–572, December 1966.

[33] Michel Parigot. $\lambda\mu$-calculus: An Algorithmic Interpretation of Classical Natural Deduction. In Andrei Voronkov, editor, *LPAR*, volume 624 of *Lecture Notes in Computer Science*, pages 190–201. Springer, 1992.

[34] Henri Poincaré. *Dernières pensées*. Flammarion, 1913.

[35] John Reynolds. Towards a theory of type structure. In B. Robinet, editor, *Programming Symposium*, volume 19 of *Lecture Notes in Computer Science*, pages 408–425. Springer Berlin / Heidelberg, 1974.

[36] Paweł Urzyczyn. Inhabitation in typed lambda-calculi (a syntactic approach). In Philippe de Groote and J. Roger Hindley, editors, *Typed Lambda Calculi and Applications*, volume 1210 of *Lecture Notes in Computer Science*, pages 373–389. Springer Berlin / Heidelberg, 1997.

[37] V. Valpola. Ein system der negationlosen Logik mit ausschliesslich realisierbaren Prädicaten. *Acta Philosophica Fennica*, 9:1–247, 1955.

[38] P.G.J. Vredenduin. The logic of negationless mathematics. *Compositio Mathematica*, 11:204–277, 1953.