A Rewrite System for Strongly Normalizable Terms

Olivier Hermant[1] and Ronan Saillard[1,2]

1 **CRI, MINES ParisTech, PSL Research University**
35 rue Saint-Honoré, 77 300 Fontainebleau
olivier.hermant@mines-paristech.fr

2 **INRIA**
23 avenue d'Italie, CS 81321, 75214 Paris Cedex 13
ronan.saillard@inria.fr

Abstract

In a 2012 paper, Richard Statman exhibited an inference system, based on second order monadic logic and non-terminating rewrite rules, that exactly types all strongly normalizable lambda-terms. In this paper, we show that this system can be simplified to first-order minimal logic with rewrite rules, along the Deduction modulo lines. We show that our rewrite system is terminating and that the conversion rule respects weak versions of invertibility of the arrow and of quantifiers. This requires additional care, in particular in the treatment of the latter. Then we study proof reduction, and show that every typable proof term is strongly normalizable and vice-versa.

1998 ACM Subject Classification F.4.1 Mathematical Logic, Dummy classification – please refer to http://www.acm.org/about/class/ccs98-html

Keywords and phrases Deduction modulo, Type systems, Intersection types

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Introduction

Intersection types [3, 5] have been used to show that every strongly normalizing λ-term is typable [13, 11]. In a 2013 paper [14], Rick Statman introduces a type system, which is an extension of second-order monadic logic, that is also capable to do so.The goal of this paper is to bring down a similar result to *first-order (minimal) logic*.

The central ingredient of [14] is a ternary second-order predicate D, that is a discriminator symbol [4]. D behaves like an `if` instruction on its first argument in the following sense:

$$D\,0\,F\,G \;\equiv\; F \qquad\qquad D\,1\,F\,G \;\equiv\; G$$

It is notable, that the properties of the D predicate, such as the behavior of D with respect to 0 and 1, or a form of commutativity with respect to the implication connective and the universal quantifier, are defined via a *rewriting relation*. This rewriting system is not terminating in the naive sense, but enjoys termination and confluence modulo the equivalence relation defined by the quantifier permutation rewrite rule.

This last feature made us realized, that defining such a system in *Deduction modulo theory* [9], a framework that combines *first-order* logic and rewriting rules *on formulas*, could be the right way to achieve our goal.

The will to stick to first order is very constraining: first of all, we must find a way to reflect the second-order predicate D. This is of moderate difficulty, since we mainly follow

Conference title on which this volume is based on.
Editors: Billy Editor and Bill Editors; pp. 1–20
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

the lines of the encoding of higher-order logic in Deduction modulo theory [10]. However, we also need to reflect through additional rewrite rules the properties on D, that are in [14]. In consequence, many questions, linked to rewriting in the first place but that have also an impact on the rest of the work, arise, whose answer is a lot more involved.

This document is organized as follows: we first recall the Deduction modulo theory version of many-sorted minimal logic (with simple types). In Sec. 3 right after, we introduce the rewrite system we will work on, and motivate our design choices. Those design choices, and in particular the non-confluence of the rewrite system, will then force us, in Sec. 4, to precisely state and prove the properties of the rewrite system that will be needed for the rest of the paper. In the next section, we define and prove correct a $(\forall_E)/(\forall_I)$ cut reduction in typing derivations, which allows us in Sec. 6 and Sec. 7 to derive the same results as [14], that is to say all strongly normalizable terms are typable, and vice versa.

2 Deduction Modulo

Deduction modulo has two ingredients: a logic and a rewrite system. We elude the (standard) definition of rewrite system, termination, confluence, and we refer to textbooks [15], if the reader is unfamiliar with these notions. The only peculiarity that has to be noticed is that we allow rewriting of atomic formulas into (potentially) non-atomic ones. For instance, $P \longrightarrow \forall x.(P \wedge P)$ would be an acceptable rewrite rule. Rewriting non-atomic formulas is forbidden (this would allow to immediately break the Brouwer-Heyting-Kolmogorov interpretation).

We describe in more detail the many-sorted language, and the inference rules.

2.1 Simple Types, Terms and Formulas

▶ **Definition 1** (Simple Types). We let ι and o be two base types. A (restricted) simple type is:

- either a base type, or a compound type $(\iota \to o) \to o$ or $o \to o \to o$;
- or a compound type $\iota \to \tau$ where τ is a simple type.

The language is composed of typed variables and constants, an adjustable parameter that is defined in Sec. 3 below. For each simple types τ_1 and τ_2, we define the application symbol α_{τ_1,τ_2} of arity $\langle \tau_1 \to \tau_2, \tau_1, \tau_2 \rangle$. Given two terms t, u of respective types $\tau_1 \to \tau_2$ and τ_1, $\alpha_{\tau_1,\tau_2}(t, u)$ (written $t\ u$) is a term of type τ_2.

At the propositional level, besides predicate symbols, defined as well in Sec. 3, we enjoy the sole binary connective $\to$ and the quantifier $\forall$, that binds only *variables of type* ι.

Anticipating a little bit, the "propositional" type o will serve to reflect formulas at the *term* level. In order not to encode the full higher-order logic, as in [10, 8], we need to restrict the available simple types. On the same vein, we allow only quantification over variables of type ι (denoted u, v), thus reflecting only a fragment of second-order logic. The full power of second-order logic, and in particular quantification over predicates, was neither necessary in [14]. In fact, only D accounts for the choice of second-order logic.

2.2 Proof Terms and Inference Rules

We assume familiarity with untyped and typed lambda-calculi, and discuss only the inference rules of Fig. 1. Variables are denoted x, y, z, while λ-terms are denoted X, Y, Z. The set of free variables of a term X is noted $FV(X)$. When X is strongly normalizing, we write $|X|$ the depth of its reduction tree.

$$\frac{(x:F)\in\Gamma}{\Gamma\vdash x:F}\ (\text{Axiom}) \qquad \frac{\Gamma\vdash X:F \qquad F\equiv G}{\Gamma\vdash X:G}\ (\text{Conv})$$

$$\frac{\Gamma\vdash X:F\to G \qquad \Gamma\vdash Y:F}{\Gamma\vdash XY:G}\ (\to_E) \qquad \frac{\Gamma,x:F\vdash X:G}{\Gamma\vdash \lambda x.X:F\to G}\ (\to_I)$$

$$\frac{\Gamma\vdash X:F \qquad v:\iota \qquad v \text{ does not occur in } \Gamma}{\Gamma\vdash X:\forall v.F}\ (\forall_I) \qquad \frac{\Gamma\vdash X:\forall v.F \qquad t:\iota \qquad t \text{ free for } v \text{ in } F}{\Gamma\vdash X:F[v/t]}\ (\forall_E)$$

Figure 1 Typing Rules of Minimal Natural Deduction Modulo Theory

Contexts are unordered *sets* of typed variables, which is possible since we do not have dependent types. As said above, we quantify only on variables of type ι, hence we allow to instantiate universally quantified formulas only by terms of type ι. Extensions to other types would require the variable and the term to have the same type. To indicate many applications of the same rule, we use a double inference bar.

The main rule to discuss is the $(Conv)$ rule. This rule allows to change the type of a λ-term X along the congruence $(Conv)$ generated by the rewrite rules, granting to Deduction modulo theory all its (typing, in our case) power. It obviously primarily depends on the rewrite system under consideration, which is the topic of the next section.

3 The Rewrite System

We are now about to define an embedding of second-order logic. The original idea is to embed formulas inside terms, via the type o, and to *decode* them at the propositional level via the predicate symbol ε. This idea has already been used to embed logics [8, 10] in Deduction modulo theory as well as in the $\lambda\Pi$-calculus modulo [1, 6].

However, another critical choice has to be made: to reflect quantification at the term level, the notion of *binder* should be available, which is not customary. We face a choice: either natively express an encoding of this notion for terms, via explicit substitution similarly to what was done in [8], or use a *combinatorial calculus* [2] instead. We choose the latter as a preliminary analysis of [8] showed that the many and complex rewrite rules could interfere badly with the ones we add on D, and that the technical overload of this system should better be avoided, if possible.

The price to pay is to have a *non extensional* version of the simply typed lambda-calculus, that will soon lead us into its own complications.

3.1 The Language

We consider a language composed of the following function symbols:

- $\dot\forall$ of type $(\iota\to o)\to o$.
- $\dot\Rightarrow$ of type $o\to o\to o$.
- $\dot p$ of type o.
- D of type $\iota\to o\to o\to o$.
- 0 and 1 of type ι.
- I of type $\iota\to\iota$.
- K_τ of type $\tau\to\iota\to\tau$, for each simple type τ.
- S_{τ_1,τ_2} of type $(\iota\to\tau_1\to\tau_2)\to(\iota\to\tau_1)\to\iota\to\tau_2$, for each simple types τ_1 and τ_2.

The language has a unique predicate symbol ε, of rank $\langle o\rangle$. We use infix notation for $\dot{\Rightarrow}$. The "predicate" symbol D, now lives at the *term* level, thus allowing for it to have a complex, morally second-order, type. Technically speaking, it is now a propositional term (whose type ends by o).

Notice that we have only one atomic propositional term of type o, $\dot{p}$. There is no need for other symbols, or for predicate with arguments, although it should be possible to add them.

According to the S, K, I combinatorial calculus, we define the abstraction on x in an expression as:

▶ **Definition 2.** λ^x_τ is the following function, which associates to a term of type τ with variable x of type ι, a term of type $\iota \to \tau$ where x does not appear:

$$\begin{array}{lll} \lambda^x_\iota(x) & := & I \\ \lambda^x_\tau(t) & := & K_\tau\ t \qquad\qquad\qquad\qquad \text{if } x \notin t \\ \lambda^x_\tau(\alpha(t,u)) & := & S_{\rho,\tau}\ (\lambda^x_{\rho\to\tau}(t))\ (\lambda^x_\rho(u)) \quad \text{if } x \in t \text{ or } x \in u \end{array}$$

3.2 Rewrite Rules

We consider the following rewrite rules, where τ, τ_1, τ_2 are any simple type.

$$\varepsilon(A\dot{\Rightarrow}B) \longrightarrow \varepsilon(A) \to \varepsilon(B) \qquad (1)$$

$$\varepsilon(\dot{\forall}A) \longrightarrow \forall x.\varepsilon(A\ x) \qquad (2)$$

$$I\ x \longrightarrow x \qquad (3)$$

$$K_\tau\ x\ y \longrightarrow x \qquad (4)$$

$$S_{\tau_1,\tau_2}\ x\ y\ z \longrightarrow x\ z\ (y\ z) \qquad (5)$$

Rules (1)-(2) are here to decode propositional terms into formulas [8, 10], while rules (3)-(5) are the usual rules for the S, K, I-calculus. All those rules are quite standard. Next come rules on the propositional term D, where in the last two rules x is chosen fresh:

$$D0FG \longrightarrow F \qquad (6)$$

$$D1FG \longrightarrow G \qquad (7)$$

$$Dv(F\dot{\Rightarrow}H)(G\dot{\Rightarrow}K) \longrightarrow (DvFG)\dot{\Rightarrow}(DvHK) \qquad (8)$$

$$Dv(\dot{\forall}F)G \longrightarrow \dot{\forall}\lambda^x(Dv(F\ x)G) \qquad (9)$$

$$DvF(\dot{\forall}G) \longrightarrow \dot{\forall}\lambda^x(DvF(G\ x)) \qquad (10)$$

▶ Remark.

- $\lambda^x(DvF(G\ x))$ is in fact $\dot{\forall}(S(K(DvF))(S(KG)I))$, and $\lambda^x(Dv(F\ x)G)$ has a similar shape. So, (9)-(10) are not schemata (while (4)-(5) are), and the structure and redexes of v, F, G are preserved by λ^x. This would not be the case for a more aggressive abstraction.
- $(\lambda^x_\tau(t))x \longrightarrow^* t$, but even if $t \longrightarrow t'$, $\lambda^x_\tau(t) \longrightarrow^* \lambda^x_\tau(t')$ does not hold in general. No combinatorial calculus is known to be extensional.
- The correspondence with the rules of [14] is straightforward: (6) corresponds to (0), (7) to (1), (8) to ($\to$), (9) and (10) to the two rewrite rules ($\wedge$). We have no equivalent of

  $$(\$)\ \forall x.F \longrightarrow F\ (x \text{ not free in } F) \text{ and } (\$\$)\ \forall x.\forall y.F \longrightarrow \forall y.\forall x.F$$

  for a number of reasons, among which:
  - on terms, we are not in position to inspect the argument of $\dot{\forall}$, due to the lack of extensionality: we must apply the argument to some x and reduce it to see its shape. This is exactly what (2) does, and this bit of extensionality is crucial in Lem. 18;

  - one can think, as a result of the previous remark, of defining ($) and ($$) at the formula level, but we are forbidden to define rewrite rules on non atomic formulas.
- Rules (9) (and (10)) is expressed at the term level, instead of being defined more directly as the coarser $\varepsilon(Dv(\dot\forall F)G) \longrightarrow \forall x.\varepsilon(Dv(F\ x)G)$. This is once again crucial for Lem. 18 to hold, and complicates the matter of Sec. 4.

▶ Remark. This calculus is trivially not confluent since some critical peaks [15] are not joinable. To recover weak versions of confluence, we need to examine them later. They are:
- (1) $F \dot\Rightarrow H \longleftarrow D0(F \dot\Rightarrow H)(G \dot\Rightarrow K) \longrightarrow (D0FG) \dot\Rightarrow (D0HK)$.
- (2) $G \dot\Rightarrow K \longleftarrow D1(F \dot\Rightarrow H)(G \dot\Rightarrow K) \longrightarrow (D1FG) \dot\Rightarrow (D1HK)$.
- (3) $\dot\forall(\lambda^x_o(D0(F\ x)G)) \longleftarrow D0(\dot\forall F)G \longrightarrow \dot\forall F$.
- (4) $\dot\forall(\lambda^x_o(D1F(G\ x))) \longleftarrow D1F(\dot\forall G) \longrightarrow \dot\forall G$.
- (5) $\dot\forall(\lambda^x_o(Dv(Fx)(\dot\forall G))) \longleftarrow Dv(\dot\forall F)(\dot\forall G) \longrightarrow \dot\forall(\lambda^y(Dv(\dot\forall F)(Gy)))$
- (6) $\dot\forall(\lambda^x_o(D0F(Gx))) \longleftarrow D0F(\dot\forall G) \longrightarrow F$.
- (7) $\dot\forall(\lambda^x_o(D1(F\ x)G)) \longleftarrow D1(\dot\forall F)G \longrightarrow G$.

4 Properties

We establish two very important properties of any rewrite system: termination, and a limited form of confluence. We also examine the consequences of those results.

4.1 Termination

▶ **Lemma 3.** *A term of type ι cannot contain a propositional symbol ($\dot\forall, \dot\Rightarrow$).*

Proof. By induction on the term, given the limited functions and types we allow. ◀

▶ **Lemma 4.** *The rewrite system restricted to rules* (4) *to* (7) *terminates.*

Proof. The proof of termination for the simple typed combinators S, K, I can easily cope with the addition of rules (6) and (7). See for instance [12]. ◀

▶ **Lemma 5** (Strong Normalization)**.** $\longrightarrow$ *is strongly normalizing.*

Proof. The following pair, ordered lexicographically, strictly decreases at each rewriting step: (number of $\dot\Rightarrow$ and $\dot\forall$, height of the $SKI01$-reduction tree). Indeed only rules (5) and (8) duplicate some argument but as it is of type ι it cannot contain a $\dot\Rightarrow$ or $\dot\forall$. ◀

▶ **Lemma 6.** *A term of type o without head redex is either $\dot p$, or begins with D, $\dot\forall$ or $\dot\Rightarrow$.*

Proof. By typing. The head cannot be S or K, it would reduce. ◀

4.2 Confluence: For a Handful of Dollars

It is possible to define the equations ($) and ($$) of [14] on formulas, but not on terms, so confluence of rewriting fails. Instead to strive to this, we shall focus on the desired "weak confluence", or consistency, properties. To this end we introduce a relation $\approxeq$ which goal is to relate the propositional structure of normal forms of convertible formulas.

▶ **Definition 7.** We write $A \approxeq B$ if A and B are normal and
- either $A = \forall\vec{x}.\varepsilon(t_A), B = \forall\vec{y}.\varepsilon(t_B)$ and $\varepsilon(t_A) \equiv \varepsilon(t_B)$
- or $A = \forall\vec{x}.(A_1 \to A_2), B = \forall\vec{y}.(B_1 \to B_2)$ and $A_1 \equiv B_1$ and $A_2 \equiv B_2$.

▶ Remark. $\approxeq$ is a reflexive, symmetric and transitive relation.

▶ **Lemma 8.** *If $A^{*}\!\longleftarrow B \longrightarrow^{*} B_0$ with B_0 normal then there exists A_0 normal such that $A \longrightarrow^{*} A_0$ and $A_0 \approxeq B_0$.*

Proof. Induction on $|B|$, the height of the reduction tree of B. The cases B normal or equal to A are immediate. Otherwise, we have: $A^{*}\!\longleftarrow A' \longleftarrow B \longrightarrow B' \longrightarrow^{*} B_0$.

The strategy it to look for a C_0 such that $A' \longrightarrow^{*} C_0$ and $C_0 \approxeq B_0$. Since $|A'| < |B|$, we conclude by induction hypothesis, that there exists A_0 such that $A \longrightarrow A_0$ and $A_0 \approxeq C_0 \approxeq B_0$.

If the reductions $B \longrightarrow A'$ and $B \longrightarrow B'$ are non overlapping then there exists C such that $A' \longrightarrow^{*} C^{*}\!\longleftarrow B'$ and since $|C| < |B|$, C_0 is obtained by induction hypothesis.

Thus, it only remains to exhibit C_0 in the overlapping case. We look at each critical pair separately, each time using a customized auxiliary induction hypothesis. We only detail one case here, see appendix for a complete proof.

Critical pair (3) right/left

$$A' = \mathcal{K}[\dot\forall F] \longleftarrow B = \mathcal{K}[D0(\dot\forall F)G] \longrightarrow B' = \mathcal{K}[\dot\forall(\lambda^x_o(D0(F\ x)G))] \longrightarrow^{*} B_0$$

We let $t_1 \sim t_2$ if there exist a context $\mathcal{K}$ and two terms θ_1, θ_2 such that:

- $A' \longrightarrow^{*} t_1 = \mathcal{K}[\dot\forall\theta_1]$,
- $B' \longrightarrow^{*} t_2 = \mathcal{K}[\dot\forall\theta_2] \longrightarrow^{*} B_0$,
- $\theta_2 x \longrightarrow^{*} {}^{*}\!\longleftarrow_{SKI} \theta_1 x$, meaning confluence with the only rules (4)-(6),
- if $\theta_2 \longrightarrow^{*} \theta_2'$ then there exists θ_1' such that $\theta_1 \longrightarrow^{*} \theta_1'$ and $\theta_2' x \longrightarrow^{*} {}^{*}\!\longleftarrow_{SKI} \theta_1' x$.

$A' \sim B'$ (with the obvious $\theta_1 = F$ and $\theta_2 = \lambda^x_o(D0(F\ x)G)$). We prove, by induction on the length of $t_2 \longrightarrow^{*} B_0$, that if $t_1 \sim t_2$ then C_0 exists. If $t_2 = B_0$ then C_0 can be any normal form of t_1, for rewriting can occur only in θ_1. Otherwise, $t_2 \longrightarrow t_2' \longrightarrow^{*} B_0$, and:

- either $t_2' = \mathcal{K}[\dot\forall\theta_2']$ with $\theta_2 \longrightarrow \theta_2'$. Then $\mathcal{K}[\dot\forall\theta_1'] \sim t_2'$ for the θ_1' obtained by the last assumption of $t_1 \sim t_2$ and we apply the induction hypothesis;
- or $t_2' = \mathcal{K}'[\dot\forall\theta_2]$ with $\mathcal{K} \longrightarrow \mathcal{K}'$. Then $\mathcal{K}'[\dot\forall\theta_1] \sim t_2'$ and the induction hypothesis applies. Notice that this can erase, but never duplicate, θ_2 since its type is o (*cf.* Lem. 3).
- or $t_2 = \mathcal{L}[\varepsilon(\dot\forall\theta_2)]$ and $t_2' = \mathcal{L}[\forall x.\varepsilon(\theta_2 x)]$. Then we have $t_1 \longrightarrow \mathcal{L}[\forall x.\varepsilon(\theta_1 x)] \longrightarrow^{*} C^{*}\!\longleftarrow t_2' \longrightarrow^{*} \forall\vec{x}.B_0$. But $|t_2'| < |B|$, and the main induction hypothesis gives us $C \longrightarrow^{*} C_0$;
- or $t_2 = \mathcal{L}[Dv(\dot\forall\theta_2)Z]$ and $t_2' = \mathcal{L}[\dot\forall\lambda^x(Dv(\theta_2 x)Z)]$. Then $\mathcal{L}[\dot\forall\lambda^x(Dv(\theta_1 x)Z)] \sim t_2'$ and the induction hypothesis applies. Similarly if $t_2 = \mathcal{L}[DvZ(\dot\forall\theta_2)]$.

◀

▶ **Corollary 9.** *If $A \equiv B \longrightarrow^{*} B_0$ with B_0 normal, then there exists a normal form A_0 of A, such that $A_0 \approxeq B_0$.*

▶ **Corollary 10** ($\rightarrow$-Compatibility)**.** *If $F \rightarrow G \equiv H \rightarrow K$ then $F \equiv H$ and $G \equiv K$.*

▶ **Corollary 11** ($\forall$-Compatibility)**.** *If $A \equiv B \longrightarrow^{*} \forall\vec{y}B_0$ with B_0 normal and not quantified, then there exists A_0 normal and not quantified such that $A \longrightarrow^{*} \forall\vec{x}.A_0$ and $A_0 \equiv B_0$.*

4.3 Reification of Formulas: Digging Terms

To merge two derivations of the same λ-term [14, 13], we must be able to combine two formulas with the D operator. In our case, this imposes to reify formulas, since D combines propositional term, and to show the suitable coherence results.

▶ **Definition 12.** Let F be a formula. We define the term, of type o, $\gamma(F)$ (noted $\dot{F}$) as:

$$\begin{array}{lcl} \gamma(\varepsilon(t)) & := & t \\ \gamma(F \to G) & := & \gamma(F) \dot{\Rightarrow} \gamma(G) \\ \gamma(\forall x.F) & := & \dot{\forall}(\lambda_o^x(\gamma(F))) \end{array}$$

▶ Remark. $\varepsilon(\dot{F}) \longrightarrow^* F$, and $\gamma(F[x/t]) = \gamma(F)[x/t]$.

Merging two derivations heavily relies on the following result. If $F \equiv F'$ then

$$Dv\dot{F}\dot{G} \equiv Dv\dot{F'}\dot{G} \tag{11}$$

For this, we need to dig out from $\dot{F}$ the propositional structure of F. Rewrite rules (8)-(10) can be used for this.

▶ **Definition 13** (Compound Terms)**.** A term t of type o is compound if it reduces either to $A \dot{\Rightarrow} B$ or to $\dot{\forall} A$. It is implicational if it reduces to $A \dot{\Rightarrow} B$ or to $\dot{\forall} A$, with Ax implicational.

▶ Remark. If t is implicational, then $\varepsilon(t) \longrightarrow^* \forall \vec{x}.A \to B$.

The rewriting relation is too weak to ensure (11) on the nose. Due to the lack extensionality, we can not always rewrite further into a term: in particular the topmost D-redex of $\dot{F}$ is frozen. To dig deeper, we introduce a bit of extensionality by applying ε, that releases frozen redexes after application of rules (2)-(5).

Unfortunately, lifting (11) at the formula level by ε is not sufficient, since when $\dot{F}$ is an implication, we also need for $\dot{G}$ to be an implication, or at least to reduce to it. When G is, for instance $\dot{p}$, this is virtually impossible. The key insight is that, without jeopardizing typing judgements or the rewriting relation, we can replace $\dot{p}$, an inert atomic propositional term, by anything of the same type including, of course, an implicational term. This allows to dig further into G, up to the point, where $\dot{\Rightarrow}$ pops up.

▶ **Definition 14** (Refinment)**.**
- We define the term $\dot{p}^n$ by induction. $\dot{p}^0 := \dot{p}$, and $\dot{p}^{n+1} := \dot{p}^n \dot{\Rightarrow} \dot{p}^n$.
- $F\{n\}$ (resp. $t\{n\}$) is the replacement in F (resp. t) of all the atomic terms $\dot{p}$ by $\dot{p}^n$.

▶ Remark. $F\{n\}\{m\} = F\{n+m\}$

▶ **Lemma 15.** *If $F \equiv F'$ then $F\{n\} \equiv F'\{n\}$.*

▶ **Lemma 16.** *If $\Gamma \vdash X : F$ then $\Gamma\{n\} \vdash X : F\{n\}$.*

▶ Remark. The derivation structure is preserved.

▶ **Lemma 17.** *For any term t of type o, for any $n \geq 1$, $t\{n\}$ is implicational.*

Proof. Induction on any normal form of $t\{n\}$. ◀

Modulo those definitions, we are able to unpack a term, using rules (1)-(2), apply the corresponding propositional structure rule (8)-(10), and pack it back to regain the lost conversion relation.

▶ **Lemma 18.** *If $F_1 \equiv F_2$ and $G_1 \equiv G_2$ then, for some n, $\varepsilon(Dv\dot{F_1}\{n\}\dot{G_1}\{n\}) \equiv \varepsilon(Dv\dot{F_2}\{n\}\dot{G_2}\{n\})$.*

Proof. It suffices to show that if $F_1 \longrightarrow F_2$ then, for some n, $\varepsilon(Dv\dot{F_1}H\{n\}) \equiv \varepsilon(Dv\dot{F_2}H\{n\})$. The same holds for $G_1 \longrightarrow G_2$, and Lem. 15 concludes. We proceed by induction on F_1.

- If $F_1 = \varepsilon(t_1)$ we take $n = 0$. Either $F_2 = \varepsilon(t_2)$ and $t_1 \longrightarrow t_2$, or $t_1 = A \dot{\Rightarrow} B$ and $F_2 = \varepsilon(A) \to \varepsilon(B)$ (trivial cases), otherwise $t_1 = \dot{\forall} A$ and $F_2 = \forall x.\varepsilon(A\ x)$. We join to $\forall x.\varepsilon(Dv(\dot{A}x)H)$.
- If $F_1 = A_1 \to B$ and $F_2 = A_2 \to B$ with $A_1 \longrightarrow A_2$. By Lem. 17 $\varepsilon(H\{1\}) \longrightarrow^* \vec{\dot{\forall}}\varepsilon(C \dot{\Rightarrow} D)$. By induction hypothesis, $\varepsilon(Dv\dot{A}_1 C\{m\}) \equiv \varepsilon(Dv\dot{A}_2 C\{m\})$ for some m. It follows $\varepsilon(Dv(\dot{A}_1 \dot{\Rightarrow} \dot{B}))H\{m{+}1\}) \longrightarrow^* \forall\vec{x}.\varepsilon(Dv(\dot{A}_1 \dot{\Rightarrow} \dot{B})(C\{m\} \dot{\Rightarrow} D\{m\})) \longrightarrow \forall\vec{x}.(\varepsilon(Dv\dot{A}_1 C\{m\}) \to \varepsilon(Dv\dot{B}D\{m\})) \equiv \forall\vec{x}.(\varepsilon(Dv\dot{A}_2 C\{m\}) \to \varepsilon(Dv\dot{B}D\{m\})) \equiv \varepsilon(Dv(\dot{A}_2 \dot{\Rightarrow} \dot{B}))H\{m+1\})$.
- The case $F_1 = A \to B_1$ and $F_2 = A \to B_2$ with $B_1 \longrightarrow B_2$ is similar.
- If $F_1 = \forall x.A_1$ and $F_2 = \forall x.A_2$ with $A_1 \longrightarrow A_2$. By induction hypothesis, $\varepsilon(Dv(\dot{A}_1 x)H\{n\}) \equiv \varepsilon(Dv\dot{A}_2 H\{n\})$ for some n. It follows $\varepsilon(Dv(\dot{\forall}\lambda^x(\dot{A}_1))H\{n\}) \longrightarrow^* \forall x.\varepsilon(Dv\dot{A}_1 H\{n\}) \equiv \forall x.\varepsilon(Dv\dot{A}_2 H\{n\})^* \longleftarrow \varepsilon(Dv(\dot{\forall}\lambda^x(\dot{A}_2))H\{n\})$.

◀

5 Reduction of Derivations

We now follow the path pioneered by Statman [14], taking into account the modifications imposed by our framework.

5.1 Elementary Reduction

▶ **Lemma 19.** *We can turn a derivation* $(\forall_I), (Conv), (\forall_E)$ *into a derivation* $(\forall_E)^n, (Conv), (\forall_I)^m$.

Proof. Assume the derivation:

$$\dfrac{\dfrac{\dfrac{\Gamma \vdash X : A}{\Gamma \vdash X : \forall x_0.A}\,(\forall_I)}{\Gamma \vdash X : \forall y_0.B}\,(\text{Conv})}{\Gamma \vdash X : B[y_0/t]}\,(\forall_E)$$

Let $\forall\vec{x}.A_0$ be a non quantified normal form of $\forall x_0.A$. From Lem. 11, there is B_0 such that $\forall y_0.B_0 \longrightarrow^* \forall\vec{y}.B_0$ and $A_0 \equiv B_0$. We build the following derivation:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Gamma \vdash X : A}{\Gamma \vdash X : \forall x_1 \cdots x_n.A_0}\,(\text{Conv})}{\Gamma \vdash X : A_0[y_0/t]}\,(\forall_E),\ n-1 \text{ times}}{\Gamma \vdash X : B_0[y_0/t]}\,(\text{Conv})}{\Gamma \vdash X : \forall y_1 \cdots y_m.B_0[y_0/t]}\,(\forall_I),\ m-1 \text{ times}}{\Gamma \vdash X : B[y_0/t]}\,(\text{Conv})$$

◀

5.2 Segments, Merge and Reduction

▶ **Definition 20** (Segment)**.** A segment in a typing derivation is a sequence of (Conv), $(\forall_I)$ and $(\forall_E)$ inference rules.

If Γ_1 and Γ_2 are two contexts and v is a fresh variable, we write $Dv\Gamma_1\Gamma_2$ the context such that:

$$(Dv\Gamma_1\Gamma_2)(x) = \begin{cases} \Gamma_1(x) & \text{if } x \in \Gamma_1 \text{ and } x \notin \Gamma_2 \\ \Gamma_2(x) & \text{if } x \notin \Gamma_1 \text{ and } x \in \Gamma_2 \\ \varepsilon(Dv(\gamma(\Gamma_1(x)))(\gamma(\Gamma_2(x)))) & \text{if } x \in \Gamma_1 \text{ and } x \in \Gamma_2 \end{cases}$$

▶ **Lemma 21** (Segment Merge)**.** *If we have two segments* $\begin{array}{c}\Gamma_1 \vdash X : F \\ \hline\hline \Gamma_1 \vdash X : F'\end{array}$ *and* $\begin{array}{c}\Gamma_2 \vdash X : G \\ \hline\hline \Gamma_2 \vdash X : G'\end{array}$ *then, for any fresh* v*, and some* n*, we can build the following segment:*

$$\begin{array}{c}(Dv\Gamma_1\Gamma_2)\{n\} \vdash X : \varepsilon(Dv\dot{F}\{n\}\dot{G}\{n\}) \\ \hline\hline (Dv\Gamma_1\Gamma_2)\{n\} \vdash X : \varepsilon(Dv\dot{F}'\{n\}\dot{G}'\{n\})\end{array}$$

Proof. Induction on the size of the segments, that can be made equal by addition of trivial conversion steps. We examine the six possible combinations of last two rules. Lem. 18 is used in the first three cases, and for the sake of readability we make explicit the dependency on $\{n\}$ only in the first case.

-
$$\dfrac{\dfrac{\vdots}{\Gamma_1 \vdash X : F}}{\Gamma_1 \vdash X : F'}(\text{Conv}) \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash X : G}}{\Gamma_2 \vdash X : G'}(\text{Conv}) \quad \hookrightarrow \quad \dfrac{\dfrac{\vdots}{(Dv\Gamma_1\Gamma_2)\{n\} \vdash X : \varepsilon(Dv\dot{F}\{n\}\dot{G}\{n\})}}{(Dv\Gamma_1\Gamma_2)\{n+m\} \vdash X : \varepsilon(Dv\dot{F}'\{n+m\}\dot{G}'\{n+m\})}(\text{Conv})$$

-
$$\dfrac{\dfrac{\vdots}{\Gamma_1 \vdash X : F}}{\Gamma_1 \vdash X : F'}(\text{Conv}) \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash X : G}}{\Gamma_2 \vdash X : \forall x.G}(\forall_I) \quad \hookrightarrow \quad \dfrac{\dfrac{\dfrac{\vdots}{Dv\Gamma_1\Gamma_2 \vdash X : \varepsilon(Dv\dot{F}\dot{G})}}{Dv\Gamma_1\Gamma_2 \vdash X : \forall x.\varepsilon(Dv\dot{F}\dot{G})}(\forall_I)}{Dv\Gamma_1\Gamma_2 \vdash X : \varepsilon(Dv\dot{F}'(\dot{\forall}\lambda^x\dot{G}))}(\text{Conv})$$

-
$$\dfrac{\dfrac{\vdots}{\Gamma_1 \vdash X : F}}{\Gamma_1 \vdash X : F'}(\text{Conv}) \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash X : \forall x.G}}{\Gamma_2 \vdash X : G[x/t]}(\forall_E) \quad \hookrightarrow \quad \dfrac{\dfrac{\dfrac{\vdots}{Dv\Gamma_1\Gamma_2 \vdash X : \varepsilon(Dv\dot{F}(\dot{\forall}\lambda^x\dot{G}))}}{Dv\Gamma_1\Gamma_2 \vdash X : \forall x.\varepsilon(Dv\dot{F}'\dot{G})}(\text{Conv})}{Dv\Gamma_1\Gamma_2 \vdash X : \varepsilon(Dv\dot{F}'G[\dot{x}/t])}(\forall_E)$$

-
$$\dfrac{\dfrac{\vdots}{\Gamma_1 \vdash X : F}}{\Gamma_1 \vdash X : \forall x.F}(\forall_I) \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash X : G}}{\Gamma_2 \vdash X : \forall y.G}(\forall_I) \quad \hookrightarrow \quad \dfrac{\dfrac{\dfrac{\dfrac{\vdots}{Dv\Gamma_1\Gamma_2 \vdash X : \varepsilon(Dv\dot{F}\dot{G})}}{Dv\Gamma_1\Gamma_2 \vdash X : \forall x.\varepsilon(Dv\dot{F}\dot{G})}(\forall_I)}{Dv\Gamma_1\Gamma_2 \vdash X : \forall y.\forall x.\varepsilon(Dv\dot{F}\dot{G})}(\forall_I)}{Dv\Gamma_1\Gamma_2 \vdash X : \varepsilon(Dv(\dot{\forall}\lambda^x\dot{F})(\dot{\forall}\lambda^y\dot{G}))}(\text{Conv})$$

-
$$\dfrac{\dfrac{\vdots}{\Gamma_1 \vdash X : F}}{\Gamma_1 \vdash X : \forall x.F}(\forall_I) \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash X : \forall y.G}}{\Gamma_2 \vdash X : G[y/t]}(\forall_E) \quad \hookrightarrow \quad \dfrac{\dfrac{\dfrac{\dfrac{\vdots}{Dv\Gamma_1\Gamma_2 \vdash X : \varepsilon(Dv\dot{F}(\dot{\forall}\lambda^y\dot{G}))}}{Dv\Gamma_1\Gamma_2 \vdash X : \forall x.\varepsilon(Dv\dot{F}(\dot{\forall}\lambda^y\dot{G}))}(\forall_I)}{Dv\Gamma_1\Gamma_2 \vdash X : \forall y.\varepsilon(Dv(\dot{\forall}\lambda^x\dot{F})\dot{G})}(\text{Conv})}{Dv\Gamma_1\Gamma_2 \vdash X : \varepsilon(Dv(\dot{\forall}\lambda^x\dot{F})G[\dot{y}/t])}(\forall_E)$$

-
$$\dfrac{\dfrac{\vdots}{\Gamma_1 \vdash X : \forall x.F}}{\Gamma_1 \vdash X : F[x/t]}(\forall_E) \qquad \dfrac{\dfrac{\vdots}{\Gamma_2 \vdash X : \forall y.G}}{\Gamma_2 \vdash X : G[y/u]}(\forall_E) \quad \hookrightarrow \quad \dfrac{\dfrac{\dfrac{\dfrac{\vdots}{Dv\Gamma_1\Gamma_2 \vdash X : \varepsilon(Dv(\dot{\forall}\lambda^x\dot{F})(\dot{\forall}\lambda^y\dot{G}))}}{Dv\Gamma_1\Gamma_2 \vdash X : \forall x.\forall y.\varepsilon(Dv\dot{F}\dot{G})}(\text{Conv})}{Dv\Gamma_1\Gamma_2 \vdash X : \forall y.\varepsilon(DvF[\dot{x}/t]\dot{G})}(\forall_E)}{Dv\Gamma_1\Gamma_2 \vdash X : \varepsilon(DvF[\dot{x}/t]G[\dot{y}/u])}(\forall_E)$$

◀

▶ **Lemma 22.** *In a segment we can assume that no* $(\forall_I)$ *precedes an* $(\forall_E)$*.*

Proof. For a formula F, we let $w(F)$ be $\max\{|\vec{x}|,\ F \longrightarrow^* \forall\vec{x}.A_0\}$, that is to say the maximum number of head quantifiers of any reduct of F.

Given a segment of length n:

$$\dfrac{\Gamma \vdash X : F_1}{\Gamma \vdash X : F_n} \text{(segment)}$$

we let m be $\max\{w(F_1), \cdots, w(F_n)\}$ We define $w(s)$ to be the m-uple of integers $\langle n_m, \cdots, n_1 \rangle$, where n_i is the sum of:

- the number of rules $\dfrac{\Gamma \vdash X : F_j}{\Gamma \vdash X : F_{j+1}} (\forall_E)$ such that $w(F_j) = i$ and there is at least one $(\forall_I)$ rule above in the segment;
- and the number of rules $\dfrac{\Gamma \vdash X : F_j}{\Gamma \vdash X : F_{j+1}} (\forall_E)$ such that $w(F_j) = i$ and there is at least one $(\forall_E)$ rule below in the segment.

Lem. 19 makes $w(s)$, lexicographically ordered, decrease. Indeed it replaces the formulæ $\forall x_0.A$ and $\forall y_0.B$ by (many) lighter formulas: $n_{\max(w(\forall x_0.A), w(\forall y_0.B))}$ decreases strictly, and only values at index strictly lower than $\max(w(\forall x_0.A), w(\forall y_0.B))$ can increase. Therefore the process terminates. ◀

▶ **Lemma 23.** *If there is a segment between $\Gamma \vdash X : F \to G$ and $\Gamma \vdash X : H \to K$ then $F \equiv H$ and $G \equiv K$.*

Proof. By Lem. 22, we can shrink the segment into a single (Conv) rule. We conclude by Arrow Compatibility (Lem. 10). ◀

6 Strongly Normalizing Terms are Typable

▶ **Lemma 24** (Inversion). *Let $\Gamma \vdash X : F$ be a derivation. X has three possible shapes. It is either x, or $\lambda y.Y$, or YZ. Accordingly, the last rules of the derivation are:*

$$\dfrac{\dfrac{(x:G) \in \Gamma}{\Gamma \vdash x : G}\,(\mathit{Axiom})}{\Gamma \vdash x : F}\,(\mathit{segment}) \qquad \dfrac{\dfrac{\Gamma(y:K) \vdash Y : L}{\Gamma \vdash \lambda y.Y : K \to L}\,(\to_I)}{\Gamma \vdash \lambda y.Y : F}\,(\mathit{segment}) \qquad \dfrac{\dfrac{\Gamma \vdash Y : K \to L \qquad \Gamma \vdash Z : K}{\Gamma \vdash YZ : L}\,(\to_E)}{\Gamma \vdash YZ : F}\,(\mathit{segment})$$

Proof. By induction on the typing derivation. ◀

▶ **Lemma 25** (Lemma 1 of [14]). *If $\Gamma_1 \vdash X : F$ and $\Gamma_2 \vdash X : G$ then, for some n, $Dv\Gamma_1\Gamma_2\{n\} \vdash X : \varepsilon(Dv\dot{F}\{n\}\dot{G}\{n\})$.*

Proof. Induction on X.

- Case $X = x$. By Lem. 24,

$$\dfrac{\dfrac{(x:F') \in \Gamma_1}{\Gamma_1 \vdash x : F'}\,(\text{Axiom})}{\Gamma_1 \vdash x : F}\,(\text{segment}) \quad \text{and} \quad \dfrac{\dfrac{(x:G') \in \Gamma_2}{\Gamma_2 \vdash x : G'}\,(\text{Axiom})}{\Gamma_2 \vdash x : G}\,(\text{segment})$$

  Lem. 21 gives, for some n, the derivation

$$\dfrac{\dfrac{(x : \varepsilon(Dv\dot{F}'\dot{G}')\{n\}) \in Dv\Gamma_1\Gamma_2\{n\}}{Dv\Gamma_1\Gamma_2\{n\} \vdash x : \varepsilon(Dv\dot{F}'\dot{G}')\{n\}}}{Dv\Gamma_1\Gamma_2\{n\} \vdash x : \varepsilon(Dv\dot{F}\dot{G})\{n\}}\,(\text{segment})$$

- Case $X = YZ$. By Lem. 24,

$$\dfrac{\dfrac{\Gamma_1 \vdash Y : K \to L \qquad \Gamma_1 \vdash Z : K}{\Gamma_1 \vdash YZ : L}\,(\to_E)}{\Gamma_1 \vdash YZ : F}\,(\text{segment}) \quad \text{and} \quad \dfrac{\dfrac{\Gamma_2 \vdash Y : K' \to L' \qquad \Gamma_2 \vdash Z : K'}{\Gamma_2 \vdash YZ : L'}\,(\to_E)}{\Gamma_2 \vdash YZ : G}\,(\text{segment})$$

  By induction hypothesis, Lem. 16 and Lem. 21 we get the derivation, for some n

$$\dfrac{\dfrac{D v \Gamma_1 \Gamma_2\{n\} \vdash Y : \varepsilon(Dv(\dot{K} \dot{\Rightarrow} \dot{L})(\dot{K}' \dot{\Rightarrow} \dot{L}')\{n\})}{D v \Gamma_1 \Gamma_2\{n\} \vdash Y : \varepsilon(D v \dot{K} \dot{K}')\{n\} \to \varepsilon(D v \dot{L} \dot{L}')\{n\}}\,(\text{Conv}) \qquad D v \Gamma_1 \Gamma_2\{n\} \vdash Z : \varepsilon(D v \dot{K} \dot{K}')\{n\}}{\dfrac{D v \Gamma_1 \Gamma_2\{n\} \vdash Y Z : \varepsilon(D v \dot{L} \dot{L}')\{n\}}{D v \Gamma_1 \Gamma_2\{n\} \vdash Y Z : \varepsilon(D v \dot{F} \dot{F}')\{n\}}\,(\text{segment})}\,(\to_E)$$

- Case $X = \lambda y.Y$. By Lem. 24,

$$\dfrac{\dfrac{\Gamma_1, y : K \vdash Y : L}{\Gamma_1 \vdash \lambda y.Y : K \to L}\,(\to_I)}{\Gamma_1 \vdash \lambda y.Y : F}\,(\text{segment}) \qquad \text{and} \qquad \dfrac{\dfrac{\Gamma_2, y : K' \vdash Y : L'}{\Gamma_2 \vdash \lambda y.Y : K' \to L'}\,(\to_I)}{\Gamma_2 \vdash \lambda y.Y : G}\,(\text{segment})$$

By induction hypothesis, Lem. 16 and Lem. 21 we get the derivation, for some n

$$\dfrac{\dfrac{\dfrac{D v \Gamma_1 \Gamma_2, y : D v \dot{K} \dot{K}'\{n\} \vdash Y : \varepsilon(D v \dot{L} \dot{L}')\{n\}}{D v \Gamma_1 \Gamma_2\{n\} \vdash \lambda y.Y : \varepsilon(D v \dot{K} \dot{K}')\{n\} \to \varepsilon(D v \dot{L} \dot{L}')\{n\}}\,(\to_I)}{D v \Gamma_1 \Gamma_2\{n\} \vdash \lambda y.Y : \varepsilon(Dv(\dot{K} \dot{\Rightarrow} \dot{L})(\dot{K}' \dot{\Rightarrow} \dot{L}'))\{n\}}\,(\text{Conv})}{D v \Gamma_1 \Gamma_2\{n\} \vdash \lambda y.Y : \varepsilon(D v \dot{F} \dot{G})\{n\}}\,(\text{segment})$$

◀

▶ **Lemma 26.** *If $\Gamma, x : F \vdash X : G$ then, for any H of type o, $\Gamma, x : \forall v.\varepsilon(D v \dot{F} H) \vdash X : G$ and $\Gamma, x : \forall v.\varepsilon(D v H \dot{F}) \vdash X : G$.*

Proof. Induction on the derivation. The only interesting case is (Axiom) with $X = x$. We apply (Axiom), $(\forall_E)$ to substitute 0 (resp. 1) to v, and (Conv). ◀

▶ Remark. The structure of the derivation is preserved.

▶ **Lemma 27.** *If $\Gamma_i \vdash X_i : F_i$ for $i \in \{1 \cdots n\}$ then there exists Γ such that $\Gamma \vdash X_i : F_i$.*

Proof. Induction on n.

- The case $n = 1$ is trivial.
- For $n \geq 2$, consider $\Gamma_n \vdash X_n : F_n$ and Γ' obtained by induction hypothesis. If $(x : F) \in \Gamma_n$ and $(x : G) \in \Gamma'$ we take $\Gamma(x) = \forall v.\varepsilon(D v \dot{F} \dot{G})$ (see Lem. 26). Otherwise, we take $\Gamma(x) = (x : F)$, or $(x : G)$.

◀

▶ **Lemma 28** (Lemma 2 of [14])**.** *If X is normal then $\Gamma \vdash X : F$ for some context Γ and formula F.*

Proof. Induction on X, whose general form is $\lambda x_1 \cdots \lambda x_r.x_i X_1 \cdots X_s$. By induction hypothesis, $\Gamma_i \vdash X_i : F_i$. Let $F_0 = F_1 \to \cdots \to F_s \to F$ for some formula F. By Lem. 27, let Γ such that $\Gamma \vdash X_i : F_i$ and $\Gamma \vdash x_i : F_0$. Types of variables in Γ may change, including x_i. If some $x_j \notin \Gamma$, complete it arbitrarily with $(x_j : G_j)$. We deduce $\Gamma \vdash X : \Gamma(x_1) \to \cdots \to \Gamma(x_r) \to F$. ◀

If Γ_1 and Γ_2 are two contexts, we note $\forall u.D u \Gamma_1 \Gamma_2$ the context such that:

$$(\forall u.D u \Gamma_1 \Gamma_2)(x) = \begin{cases} \Gamma_1(x) & \text{if } x \in \Gamma_1 \text{ and } x \notin \Gamma_2 \\ \Gamma_2(x) & \text{if } x \notin \Gamma_1 \text{ and } x \in \Gamma_2 \\ \forall u.\varepsilon(Du(\gamma(\Gamma_1(x)))(\gamma(\Gamma_2(x)))) & \text{if } x \in \Gamma_1 \text{ and } x \in \Gamma_2 \end{cases}$$

▶ **Lemma 29.** *If $D v \Gamma_1 \Gamma_2 \vdash X : F$ then $\forall u.D u \Gamma_1 \Gamma_2 \vdash X : F$.*

Proof. Induction on the typing derivation. ◀

▶ **Lemma 30** (Lemma 3 of [14]). *If $\Gamma, \Gamma_1 \vdash X[x/Y] : F$, $x \in X$ and $dom(\Gamma_1) = FV(Y)$, then, for some n and Γ_2 such that $dom(\Gamma_2) = FV(Y)$, we have $\Gamma\{n\}, \Gamma_2 \vdash (\lambda x.X)Y : F\{n\}$.*

Proof. We show by induction on the typing derivation that: if $\Gamma, \Gamma_1 \vdash X[x/Y] : F$, $x \in X$ and $dom(\Gamma_1) = FV(Y)$, then $\Gamma_2 \vdash Y : G$ and $\Gamma\{n\}, \Gamma_2, x : G \vdash X : F\{n\}$, for some integer n, context Γ_2 such that $dom(\Gamma_2) = FV(Y)$ and formula G.

- **(Trivial)** If $X = x$ then $\Gamma_1 \vdash Y : F$ and $\Gamma, \Gamma_1, x : F \vdash X : F$. We omit those cases below.
- **(Axiom)** If $y = X[x/Y]$ and $x \in X$ then we are in the **(Trivial)** case.
- **(Congruence)**, ($\forall_I$), ($\forall_E$). By induction hypothesis.
- ($\rightarrow_I$) $X = \lambda y.A$ and letting $X_0 = A[x/Y]$, $\Gamma, y : H, \Gamma_1 \vdash X_0 : K$ and $\Gamma, y : H, \Gamma_1 \vdash \lambda y.X_0 : H \rightarrow K$. By induction hypothesis, $\Gamma_2 \vdash Y : G$ and $\Gamma\{n\}, y : H\{n\}, \Gamma_2, x : G \vdash A : K\{n\}$. It follows that $\Gamma\{n\}, \Gamma_2, x : G \vdash X : (H \rightarrow K)\{n\}$.
- ($\rightarrow_E$) $X = AB$. Letting $X_1 = A[x/Y]$ and $X_2 = B[x/Y]$, we have $\Gamma, \Gamma_1 \vdash X_1 : H \rightarrow F$, $\Gamma, \Gamma_1 \vdash X_2 : H$, $\Gamma, \Gamma_1 \vdash X_1X_2 : F$.
  - If $x \in A$ and $x \in B$ then, by induction hypothesis, $\Gamma_A \vdash Y : G_A$, $\Gamma\{n_A\}, \Gamma_A, x : G_A \vdash A : (H \rightarrow F)\{n_A\}$, $\Gamma_B \vdash Y : G_2$ and $\Gamma\{n_B\}, \Gamma_B, x : G_B \vdash B : H\{n_B\}$.
    By Lem. 25, $(Dv\Gamma_A\Gamma_B)\{m\} \vdash Y : (DvG_AG_B)\{m\}$ and by Lem. 29 and ($\forall_I$), $(\forall v.Dv\Gamma_A\Gamma_B)\{m\} \vdash Y : (\forall v.DvG_AG_B)\{m\}$.
    Let $n = max\{n_A, n_B\} + m$, $\Gamma_2 = (\forall v.Dv\Gamma_A\Gamma_B)\{n\}$ and $G = (\forall v.DvG_AG_B)\{n\}$. By Lem. 26 and Lem. 16, $\Gamma\{n\}, \Gamma_2, x : G \vdash A : (H \rightarrow F)\{n\}$ and $\Gamma\{n\}, \Gamma_2, x : G \vdash B : H\{n\}$. Thus, by ($\rightarrow_I$), $\Gamma\{n\}, \Gamma_2, x : G \vdash X : F\{n\}$.
  - If $x \in A$ and $x \notin B$ (ie $B = X_2$) then, by induction hypothesis, $\Gamma_A \vdash Y : G$ and $\Gamma\{n\}, \Gamma_A, x : G \vdash A : (H \rightarrow F)\{n\}$. Thus, for $\Gamma_2 = \forall v.Dv\Gamma_A\Gamma_1$, by Lem. 26 and ($\rightarrow_I$), $\Gamma\{n\}, \Gamma_2, x : G \vdash X : F\{n\}$.
  - If $x \in B$ and $x \notin A$ (ie $A = X_1$) then, by induction hypothesis, $\Gamma_2 \vdash Y : G$ and $\Gamma\{n\}, \Gamma_B, x : G \vdash B : H\{n\}$. Thus, for $\Gamma_2 = \forall v.Dv\Gamma_1\Gamma_B$, by Lem. 26 and ($\rightarrow_I$), $\Gamma\{n\}, \Gamma_2, x : G \vdash X : F\{n\}$.

◀

▶ **Theorem 31** (Proposition 1 of [14]). *If X is strongly normalizable then $\Gamma \vdash X : F$, for some context Γ and formula F.*

Proof. Double induction on $|X|$ and X, whose general form is $\lambda x_1 \cdots \lambda x_r.YX_1 \cdots X_s$, with $Y = x_i$ or $Y = \lambda x.X_0$.

- if $r > 0$, then we apply induction hypothesis on X.
- if $X = x_iX_1 \cdots X_s$ then it is essentially Lem. 28.
- if $X = (\lambda x.X_0)X_1 \cdots X_s$ then, by induction on $|X|$, we have $\Gamma \vdash X_0[x/X_1]X_2 \cdots X_s : F$. At some point of the derivation (by Lem. 24), we have $\Gamma \vdash X_0[x/X_1] : G$. To derive $\Gamma \vdash X : F$, it suffices to derive $\Gamma \vdash (\lambda x.X_0)X_1 : G$ and to plug the corresponding sub-derivation. If $x \notin X_0$ then $\Gamma, x : H \vdash X_0 : G$ holds.
  Otherwise, Lem. 30 applies, and $\Gamma' \vdash (\lambda x.X_0)X_1 : G\{n\}$, with $dom(\Gamma') = dom(\Gamma)$. Lem. 26 and Lem. 16 give $\forall u.Du(\Gamma\{n\})\Gamma' \vdash (\lambda x.X_0)X_1 : G\{n\}$ and $\forall u.Du(\Gamma\{n\})\Gamma' \vdash X_0[x/X_1]X_2 \cdots X_s : F\{n\}$. The structure is preserved, so we can still plug the corresponding sub-derivation.

◀

7 Well-Typed Terms are Strongly Normalizing

The results of this section do not differ significantly from [14]. We mainly include it for self-containment and we will be sketchy.

▶ **Definition 32** (F_∞)**.** Given a term M, we define $F_\infty(M)$ as follow:
- If $M = C[(\lambda x.P)Q]$ where $(\lambda x.P)Q$ is the left-most redex in M then
  - if $x \in P$ then $F_\infty(M) = C[P[x/Q]]$
  - if $x \notin P$ and Q normal then $F_\infty(M) = C[P]$
  - if $x \notin P$ and Q not normal then $F_\infty(M) = C[(\lambda x.P)(F_\infty(Q))]$
- Otherwise M is normal and $F_\infty(M) = M$.

▶ **Theorem 33** (Barendregt's perpetual reduction strategy [2])**.** *If $F_\infty(M)$ is strongly normalizing, so is M.*

▶ **Definition 34.** The $\to$-height of a formula F is the maximum number of nested $\to$ in a normal form of F. We write $h(F)$.

▶ **Lemma 35.** *If $F \equiv G$, then $h(F) = h(G)$.*

Proof. Corollary of Lem. 8. ◀

▶ **Lemma 36** (Lemma 4 of [14])**.** *If $\Gamma \vdash Y : F$, $\Gamma, y : F \vdash X : G$ and X, Y are strongly normalizing then $\Gamma \vdash X[y/Y] : G$ and $X[y/Y]$ is strongly normalizing.*

Proof. Induction on the tuple $(h(F)$,$|Y|$,$|X|$,$X)$ ordered lexicographically. X has the general shape $\lambda x_1 \dots \lambda x_r.ZX_1 \dots X_s$, with $Z = z$ or $Z = \lambda x.X_0$.

We prove that, for some n, $F_\infty^n(X[y/Y])$ is strongly normalizing. By Thm. 33, this entails that $X[y/Y]$ is strongly normalizing.

- If $r > 0$ or $r = 0$ and $Z = z \neq y$ then the induction hypothesis on X applies.
- If $X = (\lambda x.X_0)X_1 \dots X_s$ then we have the following derivation:

$$\dfrac{\dfrac{\dfrac{\Gamma, y : F, x : H \vdash X_0 : J}{\Gamma, y : F \vdash \lambda x.X_0 : H \to J}}{\Gamma, y : F \vdash \lambda x.X_0 : K \to L}\text{(segment)} \qquad \Gamma, y : F \vdash X_1 : K}{\Gamma, y : F \vdash (\lambda x.X_0)X_1 : L}$$

  By the property on segments between arrows (Lem. 23), we have $H \equiv K$ and $J \equiv L$. If $x \in X_0$ then $F_\infty(X[y/Y]) = (X_0[x/X_1]X_2 \dots X_s)[y/Y]$. If $x \notin X_0$ then, because $X_1[y/Y]$ is strongly normalizable by induction hypothesis on X_1, for some n, $F_\infty^n(X[y/Y]) = (X_0X_2 \dots X_s)[y/Y]$. Since $\Gamma, y : F \vdash X_0[x/X_1]X_2 \cdots X_s : G$, in both cases the induction hypothesis on $|X|$ applies.
- If $X = yX_1 \dots X_s$ then, by induction hypothesis on X, the $X_i[y/Y]$ are strongly normalizing. We distinguish three sub-cases depending on the shape of Y.
  - If $Y = zY_1 \dots Y_q$ then $X[x/Y] = zY_1 \dots Y_qX_1[y/Y] \dots X_s[y/Y]$. Thus it is strongly normalizing by induction hypothesis on X.
  - If $Y = (\lambda z.Z)Y_1 \dots Y_q$, let $A = x(X_1[y/Y]) \dots (X_s[y/Y])$ and $Y' = Z[z/Y_1]Y_2 \dots Y_q$. A is strongly normalizing because its components are. Y' is strongly normalizing because it is a reduct of Y. By induction on $|Y|$, $A[x/Y']$ is strongly normalizing. Now, since Y_1 is strongly normalizing, we have, for some n, $F_\infty^n(X[y/Y]) = Z[z/Y_1]Y_2 \dots Y_qX_1[x/Y] \dots X_s[x/Y] = A[x/Y']$.

- if $Y = \lambda z.Z$ then $X[y/Y] = (\lambda z.Z)X_1[y/Y] \cdots X_s[y/Y]$ and we have the following derivations:

$$\dfrac{\dfrac{\Gamma(y:F) \vdash y:F}{\Gamma(y:F) \vdash y:H \to K}\,(\text{segment}) \qquad \Gamma(y:F) \vdash X_1:H}{\Gamma(y:F) \vdash yX_1:K} \qquad \dfrac{\dfrac{\Gamma(z:L) \vdash Z:M}{\Gamma \vdash \lambda z.Z:L \to M}}{\Gamma \vdash \lambda z.Z:F}\,(\text{segment})$$

  Thus we have $H \to K \equiv F \equiv L \to M$ and by Lem. 10, $H \equiv L$ and $K \equiv M$.
  - If $z \in Z$ then $F_\infty(X[y/Y]) = (Z[z/X_1[y/Y]])(X_2[y/Y])\ldots(X_s[y/Y])$. Since $h(H) < h(H \to K) = h(F)$, by induction hypothesis, $Z[z/X_1[y/Y]]$ is strongly normalizing. Since $h(M) < h(L \to M) = h(F)$, by the same argument, $F_\infty(X[y/Y]) = (x(X_2[y/Y])\ldots(X_s[y/Y]))[x/Z[z/X_1[y/Y]]]$ is strongly normalizing.
  - If $z \notin Z$ then for some n, $F^n_\infty(X[x/Y]) = Z(X_2[y/Y])\ldots(X_s[y/Y])$. Again, since $h(M) < h(F)$, by induction hypothesis, $F^n_\infty(X[y/Y]) = (x(X_2[y/Y])\ldots(X_s[y/Y]))[x/Z]$ is strongly normalizing.

◀

▶ **Corollary 37** (Subject reduction). *If $\Gamma \vdash X : F$, X strongly normalizing and $X \to_\beta X'$ then $\Gamma \vdash X' : F$.*

▶ **Theorem 38** (Proposition 2 of [14]). *If $\Gamma \vdash X : T$ then X is strongly normalizing.*

Proof. We proceed by induction on X.

$$X = \lambda x_1 \ldots \lambda x_r Y X_1 \ldots X_s \text{ with } Y = x_i \text{ or } Y = \lambda x.X_0$$

- If $r > 0$ or $r = 0$ and $Y = x_i$ then X is strongly normalizing induction hypothesis.
- Otherwise $X = (\lambda x.X_0)X_1 \ldots X_s$.
  We prove by induction on $(s, |Y| + \sum |X_i|)$ that if $X = YX_1 \ldots X_s$ is typable and the Y, X_i are strongly normalizing then X is strongly normalizing. We show that every reduct X' of X is strongly normalizing.
  - If the reduction is in Y or X_i then, since we have subject reduction (Lem. 37), we can apply the induction hypothesis on $|Y| + \sum |X_i|$.
  - If $Y = \lambda x.X_0$ and the $X' = (X_0[x/X_1])X_2 \cdots X_s$ then, by Lem. 36, $(X_0[x/X_1])$ is strongly normalizing and we can apply the induction hypothesis on s.

◀

8 Conclusion

We have defined a typing system in minimal logic with first-order rewriting rules that is able to type exactly the strongly normalizable terms. Moreover, we have done so without the equivalence relation on formulas induced by the quantifiers rules ($) (omission) and ($$) (permutation), which was conjectured in [14].

To achieve this goal, we used a combinatorial calculus and propositional rewrite rules. We also needed results on the rewrite system, among which termination and a weak form of confluence, made difficult due to lack of extensionality.

We probably can simplify the proofs herein by using higher-level results, as confluence modulo a well-chosen equivalence relation, and by being more accurate in refining types, for instance. We leave this as further work. We also need to explicit the relation with intersection types, by defining a sound and complete translation between both systems. To many extents, $\forall v.\varepsilon(DvFG)$ behaves like $F \cap G$.

Lastly, we designed this rewrite system to investigate its super-consistency [7]. It has been conjectured that all rewrite systems in Deduction modulo theory , for which the typable λ-terms are strongly normalizable, are super-consistent. It would be very interesting to answer this question for our system, that we have kept simple on that purpose. If it appears not to be super-consistent, then it answers (negatively) to the conjecture and, otherwise, we would get, as a byproduct, a reducibility-candidate model for strongly normalizable λ-terms.

9 Acknowledgments

We would like to thanks Ali Assaf and Gilles Dowek for their comments on this work.

References

1 Ali Assaf. Conservativity of embeddings in the lambda-Pi calculus modulo rewriting, 2015. Submitted to TLCA 2015.

2 Henk Barendregt. *The Lambda Calculus: Its Syntax and Semantics.* Studies in Logic and the Foundations of Mathematics. Elsevier Science, 1985.

3 Henk Barendregt, Mario Coppo, and Mariangiola Dezani-Ciancaglini. A filter lambda model and the completeness of type assignment. *J. Symb. Log.*, 48(4):931–940, 1983.

4 Stephen L. Bloom and Ralph Tindell. Varieties of "if-then-else". *SIAM J. Comput.*, 12(4):677–707, 1983.

5 Mario Coppo and Mariangiola Dezani-Ciancaglini. An extension of the basic functionality theory for the λ-calculus. *Notre Dame Journal of Formal Logic*, 21(4):685–693, 1980.

6 Denis Cousineau and Gilles Dowek. Embedding pure type systems in the lambda-pi-calculus modulo. In Simona Ronchi Della Rocca, editor, *Typed Lambda Calculi and Applications, 8th International Conference, TLCA 2007*, volume 4583 of *Lecture Notes in Computer Science*, pages 102–117. Springer, 2007.

7 Gilles Dowek. Truth values algebras and normalization. In Thorsten Altenkirch and Conor McBride, editors, *TYPES for proofs and programs*, volume 4502 of *Lecture Notes in Computer Science*, pages 110–124. Springer, 2006.

8 Gilles Dowek, Thérèse Hardin, and Claude Kirchner. Hol-lambda-sigma: an intentional first-order expression of higher-order logic. *Mathematical Structures in Computer Science*, 11:1–25, 2001.

9 Gilles Dowek, Thérèse Hardin, and Claude Kirchner. Theorem proving modulo. *Journal of Automated Reasoning*, 31:33–72, 2003.

10 Gilles Dowek and Benjamin Werner. Proof normalization modulo. *The Journal of Symbolic Logic*, 68(4):1289–1316, December 2003.

11 Silvia Ghilezan. Strong normalization and typability with intersection types. *Notre Dame Journal of Formal Logic*, 37(1):44–52, 1996.

12 J.R. Hindley and J.P. Seldin. *Lambda-Calculus and Combinators: An Introduction.* Cambridge University Press, 2008.

13 G. Pottinger. A type assignment for the strongly normalizable lambda-terms. In J. Hindley and J. Seldin, editors, *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 561–577. Academic Press, 1980.

14 Rick Statman. A New Type Assignment for Strongly Normalizable Terms. In Simona Ronchi Della Rocca, editor, *Computer Science Logic 2013 (CSL 2013)*, volume 23 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 634–652, Dagstuhl, Germany, 2013. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

15 TeReSe. *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science.* Cambridge University Press, 2003.

A Detailed proof of Lemma 8

▶ **Lemma 39.** *If $A^*\!\longleftarrow B \longrightarrow^* B_0$ with B_0 normal then there exists A_0 normal such that $A \longrightarrow^* A_0$ and $A_0 \approxeq B_0$.*

Proof. We proceed by induction on $|B|$, the height of the reduction tree of B.

If $|B| = 0$ or $A = B$ then the conclusion is immediate.

Otherwise we have the following diagram: If $A^*\!\longleftarrow A' \longleftarrow B \longrightarrow B' \longrightarrow^* B_0$.

If we are able to find a C_0 such that $A' \longrightarrow C_0$ and $C_0 \approxeq B_0$ then we can conclude since, $|A'| < |B|$ and by induction hypothesis, there exists A_0 such that $A \longrightarrow A_0$ and $A_0 \approxeq C_0 \approxeq B_0$.

If the reduction $B \longrightarrow A'$ and $B \longrightarrow B'$ are non overlapping then there exists C such that such that $A' \longrightarrow^* C^*\!\longleftarrow B'$ and since $|C| < |B|$, such C_0 exists by induction hypothesis.

Thus, it remains to show that such C_0 exists in the overlapping case. We look at each critical pair separately, each time using an auxiliary induction hypothesis.

Critical pair (1) left/right

$$A' = \mathcal{K}[F\Rrightarrow H] \longleftarrow B = \mathcal{K}[D0(F\Rrightarrow H)(G\Rrightarrow K)] \longrightarrow B' = \mathcal{K}[(D0FG)\Rrightarrow(D0HK)] \longrightarrow^* B_0$$

We write $t_1 \sim t_2$ if there exist a context $\mathcal{K}$ and terms θ^i, Z^i such that

- $A' \longrightarrow^* t_1$,
- $B' \longrightarrow^* t_2 \longrightarrow^* B_0$,
- $t_1 = \mathcal{K}[\theta^1, \cdots, \theta^k]$ and
- $t_2 = \mathcal{K}[D0\theta^1 Z^1, \cdots, D0\theta^k Z^k]$.

We have $A' \sim B'$. We prove, by induction on the length of $t_2 \longrightarrow^* B_0$, that if $t_1 \sim t_2$ then C_O exists.

If $t_2 = B_0$ then, since B_0 is normal, $k = 0$, $t_1 = t_2$ and we can take $C_0 = B_0$.

If $t_2 \longrightarrow t_2' \longrightarrow^* B_0$ then

- either $t_2' = \mathcal{K}'[D0\theta^1 Z^1, \cdots, D0\theta^k Z^k]$ with $\mathcal{K} \longrightarrow \mathcal{K}'$ then we have $\mathcal{K}'[\theta^1, \cdots, \theta^k] \sim t_2'$ and the induction hypothesis applies. Remark that this can erase a θ^i but never duplicate it since its type is o (cf Lemma 3).
- or $t_2' = \mathcal{K}[D0\theta^1 Z^1, \cdots, \theta^j, \cdots, D0\theta^k Z^k]$ or $t_2' = \mathcal{K}[D0\theta^1 Z^1, \cdots, D0\theta^j X, \cdots, D0\theta^k Z^k]$ with $Z^j \longrightarrow X$ then $t_1 \sim t_2'$ and the induction hypothesis applies.
- or $t_2' = \mathcal{K}[D0\theta^1 Z^1, \cdots, D0XZ^j, , \cdots, D0\theta^k Z^k]$ with $\theta^j \longrightarrow X$ then $\mathcal{K}[\theta^1, \cdots, X, \cdots, \theta^k] \sim t_2'$ and the induction hypothesis applies.
- or $\theta^j = A\Rrightarrow B$, $Z^j = C\Rrightarrow D$ and $t_2' = \mathcal{K}[D0\theta^1 Z^1, \cdots, (D0AC)\Rrightarrow(D0BD), \cdots, D0\theta^k Z^k] = \mathcal{L}[D0\theta^1 Z^1, \cdots, D0AC, D0BD, \cdots, D0\theta^k Z^k]$ then $\mathcal{L}[\theta^1, \cdots, A, B, \cdots, \theta^k] \sim t_2'$ and the induction hypothesis applies.

Critical pair (1) right/left

$$A' = \mathcal{K}[(D0FG)\Rrightarrow(D0HK)] \longleftarrow B = \mathcal{K}[D0(F\Rrightarrow H)(G\Rrightarrow K)] \longrightarrow B' = \mathcal{K}[F\Rrightarrow H] \longrightarrow^* B_0$$

Since $A' \longrightarrow^* B'$ we can take $C_0 = B_0$.

Critical pair (2) left/right

$$A' = \mathcal{K}[G\Rrightarrow K]) \longleftarrow B = \mathcal{K}[D1(F\Rrightarrow H)(G\Rrightarrow K)] \longrightarrow B' = \mathcal{K}[(D1FG)\Rrightarrow(D1HK)] \longrightarrow^* B_0$$

Same as Critical pair (1) left/right.

Critical pair (2) right/left

$$A' = \mathcal{K}[(D1FG)\dot{\Rightarrow}(D1HK)] \longleftarrow B = \mathcal{K}[D1(F\dot{\Rightarrow}H)(G\dot{\Rightarrow}K)] \longrightarrow B' = \mathcal{K}[G\dot{\Rightarrow}K] \longrightarrow^* B_0$$

Since $A' \longrightarrow^* B'$ we can take $C_0 = B_0$.

Critical pair (3) left/right

$$A' = \mathcal{K}[\dot{\forall}(\lambda_o^x(D0(F\ x)G))]) \longleftarrow B = \mathcal{K}[D0(\dot{\forall}F)G] \longrightarrow B' = \mathcal{K}[\dot{\forall}F] \longrightarrow^* B_0$$

We write $t_1 \sim t_2$ if there exist a context $\mathcal{K}$ and two terms θ_1, θ_2 such that

- $A' \longrightarrow^* t_1$,
- $B' \longrightarrow^* t_2 \longrightarrow^* B_0$,
- $t_1 = \mathcal{K}[\dot{\forall}\theta_1]$
- $t_2 = \mathcal{K}[\dot{\forall}\theta_2]$
- $\theta_1 x \longrightarrow^* {}^*\!\!\longleftarrow_{SKI} \theta_2 x$

We have $A' \sim B'$. We prove, by induction on the length of $t_2 \longrightarrow^* B_0$, that if $t_1 \sim t_2$ then C_O exists.

If $t_2 = B_0$ then we take C_0 to be a normal form of t_1.

If $t_2 \longrightarrow t_2' \longrightarrow^* B_0$ then

- either $t_2' = \mathcal{K}[\dot{\forall}\theta_2']$ with $\theta_2 \longrightarrow \theta_2'$ then, since $\longrightarrow_{SKI}$ commutes with $\longrightarrow$, we have $t_1 \sim t_2'$ and the induction hypothesis applies
- or $t_2' = \mathcal{K}'[\dot{\forall}\theta_2]$ with $\mathcal{K} \longrightarrow \mathcal{K}'$ then $\mathcal{K}'[\dot{\forall}\theta_1] \sim t_2'$ and the induction hypothesis applies
- or $t_2 = \mathcal{L}[\varepsilon(\dot{\forall}\theta_2)]$ and $t_2' = \mathcal{L}[\forall x.\varepsilon(\theta_2 x)]$ then $t_1 = \mathcal{L}[\varepsilon(\dot{\forall}\theta_1)] \longrightarrow \mathcal{L}[\forall x.\varepsilon(\theta_1 x)] \longrightarrow^* {}^*\!\!\longleftarrow_{SKI} t_2' \longrightarrow^* \forall\vec{x}.B_0$ and, since $\longrightarrow_{SKI}$ commutes with $\longrightarrow$ and B_0 is normal, $t_1 \longrightarrow^* B_0$. Thus we can take $C_0 = B_0$.
- or $t_2 = \mathcal{L}[DvZ(\dot{\forall}\theta_2)]$ and $t_2' = \mathcal{L}[\dot{\forall}\lambda^x(DvZ(\theta_2 x))]$ then we have $\mathcal{L}[\dot{\forall}\lambda^x(Dv(\theta_1 x)Z)] \sim t_2'$ and we can apply the induction hypothesis.
- or $t_2 = \mathcal{L}[Dv(\dot{\forall}\theta_2)Z]$ and $t_2' = \mathcal{L}[\dot{\forall}\lambda^x(Dv(\theta_2 x)Z)]$ then we proceed as in the previous case.

Critical pair (3) right/left

$$A' = \mathcal{K}[\dot{\forall}F] \longleftarrow B = \mathcal{K}[D0(\dot{\forall}F)G] \longrightarrow B' = \mathcal{K}[\dot{\forall}(\lambda_o^x(D0(F\ x)G))] \longrightarrow^* B_0$$

We write $t_1 \sim t_2$ if there exist a context $\mathcal{K}$ and two terms θ_1, θ_2 such that

- $A' \longrightarrow^* t_1$,
- $B' \longrightarrow^* t_2 \longrightarrow^* B_0$,
- $t_1 = \mathcal{K}[\dot{\forall}\theta_1]$
- $t_2 = \mathcal{K}[\dot{\forall}\theta_2]$
- $\theta_2 x \longrightarrow^* {}^*\!\!\longleftarrow_{SKI} \theta_1 x$
- if $\theta_2 \longrightarrow^* \theta_2'$ then there exists θ_1' such that $\theta_1 \longrightarrow^* \theta_1'$ and $\theta_2' x \longrightarrow^* {}^*\!\!\longleftarrow_{SKI} \theta_1' x$.

We have $A' \sim B'$. We prove, by induction on the length of $t_2 \longrightarrow^* B_0$, that if $t_1 \sim t_2$ then C_O exists.

If $t_2 = B_0$ then we take C_0 to be a normal form of t_1.

If $t_2 \longrightarrow t_2' \longrightarrow^* B_0$ then

- either $t_2' = \mathcal{K}[\dot{\forall}\theta_2']$ with $\theta_2 \longrightarrow \theta_2'$ then $\mathcal{K}[\dot{\forall}\theta_1'] \sim t_2'$ for the θ_1' obtained by the last assumption of $t_1 \sim t_2$ and we can apply the induction hypothesis;
- or $t_2' = \mathcal{K}'[\dot{\forall}\theta_2]$ with $\mathcal{K} \longrightarrow \mathcal{K}'$ then $\mathcal{K}'[\dot{\forall}\theta_1] \sim t_2'$ and the induction hypothesis applies;
- or $t_2 = \mathcal{L}[\varepsilon(\dot{\forall}\theta_2)]$ and $t_2' = \mathcal{L}[\forall x.\varepsilon(\theta_2 x)]$ then we have $t_1 \longrightarrow \mathcal{L}[\forall x.\varepsilon(\theta_1 x)] \longrightarrow^* {}^*\!\!\longleftarrow t_2' \longrightarrow^* \forall\vec{x}.B_0$. Thus, since $|t_2'| < |B|$, we can conclude by the main induction hypothesis;

- or $t_2 = \mathcal{L}[Dv(\dot\forall\theta_2)Z]$ and $t_2' = \mathcal{L}[\dot\forall\lambda^x(Dv(\theta_2 x)Z)]$ then $\mathcal{L}[\dot\forall\lambda^x(Dv(\theta_1 x)Z)] \sim t_2'$ and the induction hypothesis applies.
- or $t_2 = \mathcal{L}[DvZ(\dot\forall\theta_2)]$ and we proceed as in the previous case.

Critical pair (4) left/right

$$A' = \mathcal{K}[\dot\forall(\lambda^x_o(D1F(G\ x)))] \longleftarrow B = \mathcal{K}[D1F(\dot\forall G)] \longrightarrow B' = \mathcal{K}[\dot\forall G] \longrightarrow^* B_0$$

Same as Critical pair (3) left/right.

Critical pair (4) right/left

$$A' = \mathcal{K}[\dot\forall G] \longleftarrow B = \mathcal{K}[D1F(\dot\forall G)] \longrightarrow B' = \mathcal{K}[\dot\forall(\lambda^x_o(D1F(G\ x)))] \longrightarrow^* B_0$$

Same as Critical pair (3) right/left.

Critical pair (5) left/right

$$A' = \mathcal{K}[\dot\forall(\lambda^x_o(Dv(Fx)(\dot\forall G)))] \longleftarrow B = \mathcal{K}[Dv(\dot\forall F)(\dot\forall G) \longrightarrow B' = \mathcal{K}[\dot\forall(\lambda^y(Dv(\dot\forall F)(Gy)))]]) \longrightarrow^* B_0$$

We write $t_1 \sim t_2$ if there exist a context $\mathcal{K}$ and terms $\theta_1, \theta_2, H_1, H_2, R$ such that

- $A' \longrightarrow^* t_1$,
- $B' \longrightarrow^* t_2 \longrightarrow^* B_0$,
- $t_1 = \mathcal{K}[\dot\forall\theta_1]$
- $t_2 = \mathcal{K}[\dot\forall\theta_2]$
- $\dot\forall\theta_1 \equiv \dot\forall\theta_2$
- $\theta_1 x \longrightarrow \dot\forall H_1$
- $\theta_2 x \longrightarrow \dot\forall H_2$
- $H_1 y \longrightarrow^*_{SKI} R\ {}^*\!\!\longleftarrow_{SKI} H_2 x$
- and if $\theta_2 \longrightarrow^* \theta_2'$ then there exist θ_1', H_1', H_2', R having the same properties as above.

We have $A' \sim B'$. We prove, by induction on the length of $t_2 \longrightarrow^* B_0$, that if $t_1 \sim t_2$ then C_O exists.

If $t_2 = B_0$ then we take C_0 to be a normal form of t_1.

If $t_2 \longrightarrow t_2' \longrightarrow^* B_0$ then

- either $t_2' = \mathcal{K}[\dot\forall\theta_2']$ with $\theta_2 \longrightarrow \theta_2'$ then $\mathcal{K}[\dot\forall\theta_1'] \sim t_2'$ (for θ_1' obtained by the last assumption of $t_1 \sim t_2$) and the induction hypothesis applies;
- or $t_2' = \mathcal{K}'[\dot\forall\theta_2]$ with $\mathcal{K} \longrightarrow \mathcal{K}'$ then $\mathcal{K}'[\dot\forall\theta_1] \sim t_2'$ and the induction hypothesis applies;
- or $t_2 = \mathcal{L}[\varepsilon(\dot\forall\theta_2)]$ and $t_2' = \mathcal{L}[\forall y.\varepsilon(\theta_2 y)]$ then $\mathcal{L}[\forall y.\forall x.\varepsilon(R)]^* \!\!\longleftarrow t_2' \longrightarrow^* B_0$. Since $|t_2'| < |B|$, there exists, by the main induction hypothesis, D_0 such that $\mathcal{L}[\forall y.\forall x.\varepsilon(R)] \longrightarrow^* D_0$ and $D_0 \approxeq B_0$. Moreover we have $t_1 \longrightarrow^* \mathcal{L}[\forall x.\forall y.\varepsilon(R)]$.
  - if $\mathcal{L} = \forall\vec{w}.\Box$ then $D_0 = \forall\vec{w}.\forall y.\forall x.G$ with $\varepsilon(R) \longrightarrow^* G$ then we take C_0 to be $\forall\vec{w}.\forall x.\forall y.G$.
  - if $\mathcal{L} = \forall\vec{w}.(L_1 \to L_2[\Box])$ (resp. $\mathcal{L} = \forall\vec{w}.(L_1[\Box] \to L_2)$) then $D_0 = \forall\vec{w}.(L_1 \to L_2[G_1])$ (resp. $\forall\vec{w}.(L_1[G_1] \to L_2)$) with G_1 a normal form of $\forall y.\forall x.\varepsilon(R)$ then we take $C_0 = \forall\vec{w}.(L_1 \to L_2[G_2])$ (resp. $C_0 = \forall\vec{w}.(L_1 \to L_2[G_2])$) with G_2 a normal form of $\forall x.\forall y.\varepsilon(R)$. We have $C_0 \approxeq D_0$ since $G_2\,{}^*\!\!\longleftarrow \varepsilon(\dot\forall(\theta_1)) \equiv \varepsilon(\dot\forall\theta_2) \longrightarrow^* G_1$.
- or $t_2 = \mathcal{L}[Dv(\dot\forall\theta_2)Z]$ and $t_2' = \mathcal{L}[\dot\forall\lambda^y(Dv(\theta_2 y)Z)]$ then $t_1 = \mathcal{L}[Dv(\dot\forall\theta_1)Z]$ and we have $\mathcal{L}[\dot\forall\lambda^x(Dv(\theta_1 x)Z)] \sim t_2'$ and the induction hypothesis applies.
- or $t_2 = \mathcal{L}[DvZ(\dot\forall\theta_2)]$ and we proceed as in the previous case.

Critical pair (5) right/left

$A' = \mathcal{K}[\dot\forall(\lambda^y(Dv(\dot\forall F)(Gy)))] \longleftarrow B = \mathcal{K}[Dv(\dot\forall F)(\dot\forall G)] \longrightarrow B' = \mathcal{K}[\dot\forall(\lambda^x_o(Dv(Fx)(\dot\forall G)))] \longrightarrow^* B_0$

Same as Critical pair (5) left/right.

Critical pair (6) right/left

$$A' = \mathcal{K}[\dot\forall(\lambda^x_o(D0F(Gx)))] \longleftarrow B = \mathcal{K}[D0F(\dot\forall G)] \longrightarrow B' = \mathcal{K}[F] \longrightarrow^* B_0$$

We write $t_1 \sim t_2$ if there exist a context $\mathcal{K}$ and two terms θ_1, θ_2 such that
- $A' \longrightarrow^* t_1$,
- $B' \longrightarrow^* t_2 \longrightarrow^* B_0$,
- $t_1 = \mathcal{K}[\dot\forall\theta_1]$
- $t_2 = \mathcal{K}[\theta_2]$
- $\theta_1 x \longrightarrow^* \theta_2$
- $\dot\forall\theta_1 \equiv \theta_2$

We have $A' \sim B'$. We prove, by induction on the length of $t_2 \longrightarrow^* B_0$, that if $t_1 \sim t_2$ then C_O exists.

If $t_2 = B_0$ then we take C_0 to be a normal form of t_1.

If $t_2 \longrightarrow t_2' \longrightarrow^* B_0$ then
- either $t_2' = \mathcal{K}[\theta_2']$ with $\theta_2 \longrightarrow \theta_2'$ then $t_1 \sim t_2'$ and the induction hypothesis applies;
- or $t_2' = \mathcal{K}'[\theta_2]$ with $\mathcal{K} \longrightarrow \mathcal{K}'$ then $\mathcal{K}'[\dot\forall\theta_1] \sim t_2'$ and the induction hypothesis applies;

Critical pair (6) right/left

$$A' = \mathcal{K}[F] \longleftarrow B = \mathcal{K}[D0F(\dot\forall G)] \longrightarrow B' = \mathcal{K}[\dot\forall(\lambda^x_o(D0F(Gx)))] \longrightarrow^* B_0$$

We write $t_1 \sim t_2$ if there exist a context $\mathcal{K}$ and two terms θ_1, θ_2 such that
- $A' \longrightarrow^* t_1$,
- $B' \longrightarrow^* t_2 \longrightarrow^* B_0$,
- $t_1 = \mathcal{K}[\theta_1]$
- $t_2 = \mathcal{K}[\dot\forall\theta_2]$
- $\theta_2 x \longrightarrow^* \theta_1$
- $\theta_1 \equiv \dot\forall\theta_2$
- if $\theta_2 \longrightarrow^* \theta_2'$ then there exists θ_1' such that $\theta_1 \longrightarrow^* \theta_1'$ and $\theta_2' x \longrightarrow^* \theta_1'$.

We have $A' \sim B'$. We prove, by induction on the length of $t_2 \longrightarrow^* B_0$, that if $t_1 \sim t_2$ then C_O exists.

If $t_2 = B_0$ then we take C_0 to be a normal form of t_1.

If $t_2 \longrightarrow t_2' \longrightarrow^* B_0$ then
- either $t_2' = \mathcal{K}[\dot\forall\theta_2']$ with $\theta_2 \longrightarrow \theta_2'$ then $\mathcal{K}[\theta_1'] \sim t_2'$ for θ_1' obtained by the last assumption of $t_1 \sim t_2$ and the induction hypothesis applies;
- or $t_2' = \mathcal{K}'[\dot\forall\theta_2]$ with $\mathcal{K} \longrightarrow \mathcal{K}'$ then $\mathcal{K}'[\theta_1] \sim t_2'$ and the induction hypothesis applies;
- or $t_2 = \mathcal{L}[\varepsilon(\dot\forall\theta_2)]$, $t_1 = \mathcal{L}[\varepsilon(\theta_1)]$ and $t_2' = \mathcal{L}[\forall x.\varepsilon(\theta_2 x)]$ then $\mathcal{L}[\forall x.\varepsilon(\theta_1)]^*\!\longleftarrow t_2' \longrightarrow^* \forall\vec{x}.B_0$. By the main induction hypothesis there exists D_0 such that $\mathcal{L}[\forall x.\varepsilon(\theta_1)] \longrightarrow^* D_0$ and $D_0 \approxeq C_0$.
  - if $\mathcal{L} = \forall\vec{w}.\Box$ then $D_0 = \forall\vec{w}.\forall x.G$ with $\varepsilon(\theta_1) \longrightarrow^* G$ and we take C_0 to be $\forall\vec{w}.G$.
  - if $\mathcal{L} = \forall\vec{w}.(L_1 \to L_2[\Box])$ (resp. $\mathcal{L} = \forall\vec{w}.(L_1[\Box] \to L_2)$) then $D_0 = \forall\vec{w}.(L_1 \to L_2[\forall x.G])$ (resp. $\forall\vec{w}.(L_1[\forall x.G] \to L_2)$) with $\varepsilon(\theta_1) \longrightarrow^* G$ and we take $C_0 = \forall\vec{w}.(L_1 \to L_2[G])$ (resp. $C_0 = \forall\vec{w}.(L_1 \to L_2[G])$). We have $C_0 \approxeq D_0$ since $G^*\!\longleftarrow \varepsilon(\theta_1) \equiv \varepsilon(\dot\forall\theta_2) \longrightarrow^* \forall x.G$.

- or $t_2 = \mathcal{L}[Dv(\dot{\forall}\theta_2)Z]$ and $t'_2 = \mathcal{L}[\dot{\forall}\lambda^x(Dv(\theta_2 x)Z)]$ then $t_1 = \mathcal{L}[Dv\theta_1 Z] \sim t'_2$ and the induction hypothesis applies;
- or $t_2 = \mathcal{L}[DvZ(\dot{\forall}\theta_2)]$ and we proceed as in the previous case.

Critical pair (7) left/right

$$A' = \mathcal{K}[\dot{\forall}(\lambda^x_o(D1(F\ x)G))] \longleftarrow B = \mathcal{K}[D1(\dot{\forall}F)G] \longrightarrow B' = \mathcal{K}[G] \longrightarrow^* B_0$$

Same as Critical pair (6) left/right.

Critical pair (7) right/left

$$A' = \mathcal{K}[G] \longleftarrow B = \mathcal{K}[D1(\dot{\forall}F)G] \longrightarrow B' = \mathcal{K}[\dot{\forall}(\lambda^x_o(D1(F\ x)G))] \longrightarrow^* B_0$$

Same as Critical pair (6) right/left.

◀