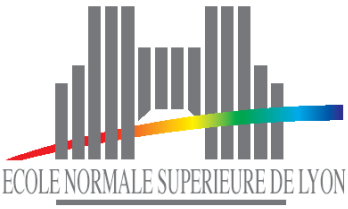
Training period
June–August 2004
Magistère d'Informatique et de Modélisation
2nd year



Decidability of Equality in Categories with Families

Samuel Mimram

17th September 2004

Directed by Thierry Coquand

Datavetenskap
Chalmers tekniska högskola
Göteborg, Sverige
http://www.cs.chalmers.se/

Abstract *Categories with families* (cwfs) is a theory which was introduced to be a categorical model of *dependently-typed* λ*-calculus* (LF). We first define both theories and discuss about possible variations of the syntax and their consequences; we also see some related theories and see why categories with families is a rather natural one in the sense that it captures the essential constructions of λ-calculus. Then we show that it is decidable to know whether two dependently-typed λ-terms are equal or not by giving an algorithm and the proof of its termination. Finally we show that the theories of cwfs and LF are equivalent (i.e. the same theorems hold in both theories, modulo interpretation) by giving two syntactic interpretations, of one theory into the other one, which are proven to have some good soundness properties. This proves that the equality is also decidable for cwfs.

1 Introduction to categories and categorical models

Formal systems are a convenient theoretical model to study programming languages and the notion of calculus in general. The relations between a syntactic theory, its semantics and an appropriate logic enable mathematical methods to be used for reasoning about those theories and understanding how relevant and natural their constructions are.

The theory of categories (def. 2) provides an abstract framework to define a collection of mathematical objects along with structured relations between them. It is abstract in the sense that only the essential properties of the concerned objects are taken in account and hardly no reference to the object themselves is ever made. Moreover this theory has been shown to be suitable as a foundation of mathematics (e.g. instead of set-theory). It is therefore important to give categorical models to theories to try to understand the respective merits and defaults of fundamental theories of mathematics. The relation of category theory to logic was first established by Lawvere (around 1970). The link between a type theory and an appropriate category is given by a categorical semantics, i.e. an interpretation of the type theory in a category. As an example, this analogy establishes the following correspondence between simply-typed λ-calculus, cartesian-closed categories (cccs) and propositional logic (see [LS86]):

Typed λ-calculus	Cartesian Closed Categories	Propositional Logic
Product types	Products	Conjunction
Function spaces	Exponentiation	Implication

Actually cartesian-closed categories have been proven (e.g. in [LS86], [Cur86] or [San87]) to be equivalent to simply-typed λ-calculus. This means that a judgment is valid in one theory iff its interpretation (which basically makes a correspondence between product types and products, function space and exponentiation, etc.) is also valid: both theories are essentially the same in the sense that, modulo interpretation, the same judgments are valid in both theories. Therefore anyone of the two presentation can be chosen when dealing with λ-calculus, it does not matter. And categorical models have some very interesting properties.

When using λ-calculus or proving theorems about it, the use of variables seem rather unnatural. Usually we only want to talk about λ-calculus modulo α-conversion[1] and often many lemmata have to be proven only to handle problems related to this. De Bruijn indexes[2] partly solve this issue but are not completely satisfactory (the β-reduction is a rather difficult to express for example). Another point is the fact that the substitution is a meta-operation in λ-calculus (it is not part of the calculus). λ-calculi with explicit substitutions were introduced for the substitution to be part of the theory but some the rules involved in the definition of such calculi seem to be too complicated

[1] α*-conversion* means renaming of bound variables. For example, the λ-term $\lambda xy.xt$ is α-convertible to $\lambda yz.yt$ but not to $\lambda xy.xu$.

[2] The idea of De Bruijn is to use natural numbers instead of variables. An index n is bound to the n-th λ-abstraction encountered when going up in the syntactic tree of the term. For example $\lambda x.x$ would be written $\lambda.0$, $\lambda xy.x$ would be written $\lambda.\lambda.1$, etc. Two λ-terms are α-convertible iff they have the same De Bruijn representation.

and improvable. Categorical models address those problems thanks to the use of combinators. In those models, in cccs for examples, bindings are expressed without the need of variables nor De Bruijn indexes which is very pleasant since variables do not seem to be part of the core λ-calculus; substitution is also expressed without being a higher order operation.

Categories with families were introduced by P. Dybjer in [Dyb96], with the aim of being a categorical model of dependently-typed λ-calculus (which is an extension of simply-typed λ-calculus). In this paper, after having defined rigorously both theories, we are going to prove that they are actually equivalent. We will also show that the equality in our dependently-typed λ-calculus (LF) is decidable and an important consequence of the equivalence between both theories is the fact that the equality is also decidable in categories with families which is an important property for equational theories because it can be the basis of numerous decidability theorems and even algorithms since we actually give an algorithm to decide the equality.

The general approach is quite simple and we can see that the definitions of the theories that are going to be proven to be equivalent look very much like one another. However, the proofs will turn out to be rather technical and far from trivial when looking at them closely. We will see that some details are very interesting because they are key points to understand what typed λ-calculus is essentially.

2 Definition of some related (or not) theories

2.1 Generalized algebraic theories

We must first define formally the theories we are going to use (mainly categories with families and a logical framework). But we need a framework to define rigorously those theories. *Generalized algebraic theories* (that we shall abbreviate in gats and we should write **Gat** when referring to the theory) provide such a framework. They were introduced by Cartmell in [Car86]. A detailed presentation – somehow syntactically easier to read – can be found in [Pit95]. The definition given here is slightly different on one point: the contexts are not required to be finite in the theory[3].

There are two main reasons which motivate the definition and the use of gats. First, it is important to have a formal definition of what is a definition of an equational theory[4]. Basically, in an equational theory we want to have "types" (or *sorts*) which are constructed using function symbols (sort constructors) and which can depend on a finite number of terms of a fixed type (the list of those terms along with their type is the *context* where those terms are defined); we also want to have terms which are constructed from term-valued function symbols (term constructors) which can depend on terms of a given type just like types; the last thing we want to be able to express in our theory is equations between terms and equations between types (that is why the theories defined in **Gat** are said to be *equational*). The collection of rules to construct types and terms as well as the equations between types of between terms are called the *axioms*.

For example, the theory of natural number with addition can be defined using the following axioms:

- "natural numbers" (Nat) is a sort
$$\diamond \vdash \mathsf{Nat}$$
(the symbol $\diamond$ stands for the empty context, the sort Nat does not need to depend on any term; the $\vdash$ symbol is here to separate the hypothesis and the conclusion)
- zero is a natural number
$$\diamond \vdash \mathsf{O} : \mathsf{Nat}$$

[3] The finiteness condition on contexts seemed to us rather unnatural, unnecessary and would complicate the definition of cwfs to keep the equivalence between **Gat** and **Cwf**. However, we did not have time to check in details that this does not introduce complications or inconsistencies.

[4] Even though there are foundational problems which obviously arise: in what formal system shall we define **Gat** itself? We will show that cwfs provide a nice answer to this question: we can define the theory **Cwf** which equivalent to **Gat** inside **Gat**.

– if n is a natural number, so is its successor

$$n : \mathsf{Nat} \vdash \mathsf{S}\,(n) : \mathsf{Nat}$$

– if n and m are natural numbers, so is their sum

$$n : \mathsf{Nat}, m : \mathsf{Nat} \vdash \mathsf{Add}\,(n, m) : \mathsf{Nat}$$

– finally, we want to express the equalities which really define the addition

$$n : \mathsf{Nat} \vdash \mathsf{Add}\,(\mathsf{O}, n) = \mathsf{O}$$
$$n : \mathsf{Nat}, m : \mathsf{Nat} \vdash \mathsf{Add}\,(\mathsf{S}\,(n)\,, m) = \mathsf{Add}\,(n, \mathsf{S}\,(m))$$

The rules for derivating *theorems* in **Gat** are show in figure 1. These are here to ensure that gats behave quite naturally[5]. In particular, we want our equality to behave like real equality.

– It must be reflexive, symmetric and transitive (which is expressed by the rules (Ty-Eq-Refl), (Ty-Eq-Sym), (Ty-Eq-Trans) and the corresponding rules for equalities between terms).
– Type and term constructors, as well as equalities must be compatible with substitution. This is expressed by rules related to substitution. For example, we want to be able to derive $\diamond \vdash \mathsf{S}\,(\mathsf{O}) : \mathsf{Nat}$ (1 is a natural number) which can be done by substituting O to n in our second axiom, or we want to be able to be able to derive $\diamond \vdash \mathsf{Add}\,(\mathsf{O}, \mathsf{O}) = \mathsf{O}$ which can be done by substituting O to n in the first equality axiom.
– If two terms (or types) are contstructed by the same function symbols with equal arguments, we want the two terms (or types) to be equal. This is expressed by the rules for axioms.

Context morphisms have been introduced to be able to express substitution (in fact a morphism corresponds to a finite number of successive substitutions) in a type-safe manner.

Definition 1 (Generalized algebraic theory (gat)). *A* context *is a list of (variable, type)-pairs defined inductively by:*

– *the* empty context*, written $\diamond$, is a context;*
– *if Γ is a context, A is a type such that* $\mathrm{FV}\,(A) \subseteq \mathrm{DV}\,(\Gamma)$ *and x is a variable which is not in* $\mathrm{DV}\,(\Gamma)$*, then $\Gamma, x : A$ is a context.*

In the previous definition $\mathrm{FV}\,(A)$ *represents the set of variables free in A and* $\mathrm{DV}\,(\Gamma)$ *represents the set of variables free in Γ. Those are defined as usual (e.g. see def. 10).*

A generalized algebraic theory *(gat) is collection of meta-constants (the type- and term-constructors):*

– *the n-ary* type-valued function symbols*:* $\mathrm{Term}^n \to \mathrm{Sort}$*;*
– *the n-ary* term-valued function symbols*:* $\mathrm{Term}^n \to \mathrm{Term}$*;*

for each natural number n, with

– *for each function symbol S a judgment*

$$\Gamma_S \vdash S(\overrightarrow{x})$$

called the introductory axiom *of s ($\overrightarrow{x}$ is of course supposed to have the same arity as s);*
– *for each term-valued function symbol F a judgment*

$$\Gamma_F \vdash F(\overrightarrow{x}) : A_F$$

called the introductory axiom *of F ($\overrightarrow{x}$ is supposed to have the same arity as F);*
– *a collection of judgments of the form $\Gamma \vdash A = A'$ called the* type-equality axioms*;*
– *a collection of judgments of the form $\Gamma \vdash M = M' : A$ called the* term-equality axioms.

Contexts

$$\frac{}{\diamond \vdash}(\textsc{C-Emp}) \qquad \frac{\Gamma \vdash A \qquad x \notin \mathrm{DV}\,(\Gamma)}{\Gamma, x : A \vdash}(\textsc{C-Ext})$$

$$\frac{}{\diamond = \diamond}(\textsc{C-Emp-Eq}) \quad \frac{\Gamma = \Gamma' \qquad \Gamma \vdash A = A'\left[\overrightarrow{x}/\overrightarrow{x}'\right]}{\Gamma, x : A = \Gamma', x' : A'}(\textsc{C-Conv})$$

Types

$$\frac{\Gamma \vdash A}{\Gamma \vdash A = A}(\textsc{Ty-Eq-Refl}) \quad \frac{\Gamma \vdash A = B}{\Gamma \vdash B = A}(\textsc{Ty-Eq-Sym}) \quad \frac{\Gamma \vdash A = B \qquad \Gamma \vdash B = C}{\Gamma \vdash A = C}(\textsc{Ty-Eq-Trans})$$

$$\frac{\gamma : \Delta \to \Gamma \qquad \Gamma \vdash A}{\Delta \vdash A\,[\gamma/\overrightarrow{x}]}(\textsc{Ty-S}) \quad \frac{\gamma = \gamma' : \Delta \to \Gamma \qquad \Gamma \vdash A = B}{\Delta \vdash A\,[\gamma/\overrightarrow{x}] = A'\,[\gamma'/\overrightarrow{x}]}(\textsc{Ty-S-Conv})$$

Terms

$$\frac{\Gamma, x : A, \Delta \vdash}{\Gamma, x : A, \Delta \vdash x : A}(\textsc{Var})$$

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash M = M : A}(\textsc{Tm-Eq-Refl}) \quad \frac{\Gamma \vdash M = N : A}{\Gamma \vdash N = M : A}(\textsc{Tm-Eq-Sym}) \quad \frac{\Gamma \vdash M = N : A \qquad \Gamma \vdash N = P : A}{\Gamma \vdash M = P : A}(\textsc{Tm-Eq-Trans})$$

$$\frac{\Gamma \vdash M : A \qquad \Gamma \vdash A = B}{\Gamma \vdash M : B}(\textsc{Tm-Conv}) \quad \frac{\Gamma \vdash M = N : A \qquad \Gamma \vdash A = B}{\Gamma \vdash M = N : B}(\textsc{Tm-Eq-Conv})$$

$$\frac{\gamma : \Delta \to \Gamma \qquad \Gamma \vdash M : A}{\Delta \vdash M\,[\gamma/\overrightarrow{x}] : A\,[\gamma/\overrightarrow{x}]}(\textsc{Tm-S}) \quad \frac{\gamma = \delta : \Delta \to \Gamma \qquad \Gamma \vdash M = N : A}{\Delta \vdash M\,[\gamma/\overrightarrow{x}] = N\,[\delta/\overrightarrow{x}] : A\,[\gamma/\overrightarrow{x}]}(\textsc{Tm-S-Conv})$$

Context morphisms

$$\frac{\Gamma \vdash}{\langle\rangle : \Gamma \to \diamond}(\textsc{M-Emp}) \qquad \frac{\gamma : \Delta \to \Gamma \qquad \Gamma \vdash A \qquad \Delta \vdash M : A\,[\gamma/\overrightarrow{x}]}{\langle \gamma, M\rangle : \Delta \to \Gamma, x : A}(\textsc{M-Ext})$$

$$\frac{\Gamma \vdash}{\langle\rangle = \langle\rangle : \Gamma \to \diamond}(\textsc{M-Emp-Refl}) \quad \frac{\gamma = \delta : \Delta \to \Gamma \qquad \Gamma \vdash A \qquad \Delta \vdash M = N : A\,[\gamma/\overrightarrow{x}]}{\langle \gamma, M\rangle = \langle \delta, N\rangle : \Delta \to \Gamma, x : A}(\textsc{M-Conv})$$

Rules for axioms

$$\frac{\Gamma_S \vdash}{\Gamma_S \vdash S\,(\overrightarrow{x})}(\textsc{Ax-Ty-I}) \qquad \frac{\Gamma_F \vdash A_F}{\Gamma_F \vdash F(\overrightarrow{x}) : A_F}(\textsc{Ax-Tm-I})$$

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A = B}(\textsc{Ax-Ty-Eq}) \quad \frac{\Gamma \vdash M : A \qquad \Gamma \vdash N : A}{\Gamma \vdash M = N : A}(\textsc{Ax-Tm-Eq})$$

Figure 1. Definition of gat

Given such a theory, the theorems *are the judgments which are derivable using the rules shown in figure 1.*

The intended meaning of the judgments is

Judgment	Meaning
$\Gamma \vdash$	Γ is a well-formed context
$\Gamma \vdash A$	A is a well-formed type in the context Γ
$\Gamma \vdash M : A$	M is a well-formed term of type A in the context Γ
$\Gamma \vdash A = A'$	In the context Γ, the types A and A' are equal and well-formed
$\Gamma \vdash M = M' : A$	In the context Γ, M and M' are well-formed terms of type A

It is important to understand that **Gat** is a meta-theory in the sense that it is a framework for defining some[6] equational theories by giving a finite number of axioms.

In the following, since it is easier to read, we will use the inference bar instead of the symbol "$\vdash$" when writing axioms and the empty context will be simply denoted by nothing in hypothesis. This inference bar must not be confused with the one used to define **Gat**. For example, for now on, the axioms of natural numbers should be written

$$\frac{}{\mathsf{O} : \mathsf{Nat}} \quad \frac{n : \mathsf{Nat}}{\mathsf{S}\,(n) : \mathsf{Nat}} \quad \frac{n\ m : \mathsf{Nat}}{\mathsf{Add}\,(n, m) : \mathsf{Nat}} \quad \text{etc.}$$

To distinguish between the equality in the equational theories and the equality used to define things, the former will still be written $=$ while the latter will be written $\equiv$ (i.e. $a \equiv b$ iff a and b are equal by definition).

2.2 Categories, cartesian closed categories

The notion of *category* is going to be used intensively (and not always explicitly) in the rest of this paper. Unfortunately, we can only make a very short presentation of the category theory which is a rather developed and complicated theory. The reader can find good introductions to category theory in many books, in [Awo03] for example.

A category $\mathbf{C}$ is given by the following data:

- a collection of *objects* $|\mathbf{C}|$;
- for each two objects A and B of $\mathbf{C}$, a collection $\mathrm{Hom}\,(A, B)$ of *arrows* (or *morphisms*) between A and B (if $f \in \mathrm{Hom}\,(A, B)$ then A is called the *domain* of f and B its *codomain* and f might be written $A \xrightarrow{f} B$);
- for each morphisms f and g such that $A \xrightarrow{f} B \xrightarrow{g} C$, there is a morphism $g \circ f$ called the *composite* of f and g;
- for each object A a morphism id_A from A to A called the *identity morphism*.

These are required to satisfy the following rules:

- associativity: $h \circ (g \circ f) = (h \circ g) \circ f$;
- unit: $f \circ \mathrm{id}_A = f = \mathrm{id}_B \circ f$ for all arrow $f \in \mathrm{Hom}\,(A, B)$.

For example, we can define the category **Grp** of groups where objects are groups and arrows are morphisms of groups (the composition and the identity are defined as usual and they satisfy the required properties).

The definition is quite abstract. One of the goals of categories is to be able to talk about "things" (the objects of the category) which have some structured relations between them (the morphisms)

[5] Actually the "natural" behavior is motivated by more than intuitive wishes: it will turn out that **Gat** is equivalent to **Cwf** and thus to dependently-typed λ-calculus.

[6] Of course, not every equational theory can be defined as a gat.

without having to explicitly refer to objects. This make category theory a very general theory – it has even been shown to be suitable to be a foundational theory for mathematics. As we will see later, the framework provided by category theory is much wider than usual morphisms between structured sets: in particular, we can also define categories where morphisms are proof of a type (a logic formula) having an other type in hypothesis. It is important to give categorical models to theories in order to understand what are the essential properties of those theories and to be able to show properties of those theories which do not depend on a particular definition of the theory, etc.

In the following, we might write $x \in \mathbf{C}$ to mean that x is an object of $\mathbf{C}$.

Definition 2 (Cartegory). *The theory of categories is defined as a gat by the "category axioms" of figure 2.*

Our goal is to show that the *categories with families* (cwfs) is a notion that subsumes cartesian closed categories: it is a categorical equivalent of λ-calculus with dependent types (instead of λ-calculus with simple types). That's why we first quickly present cartesian closed categories.

We won't explain it in details since most of its constructions can be found in categories with families (cwfs), in an extended version though, and we will explain those. The only constructions which are not needed anymore in cwfs are pairing operations ($\wedge$, $\langle\ ,\ \rangle$, π and π' which are defined by "product axioms" rules) and curyfying operations (* and ε which are defined in "exponential axioms" rules). This can be explained by the fact that objects in cwfs are not types but contexts and thus there is no need to "encode" contexts into types. Details about this are provided in section 3.3.

Definition 3 (Cartesian closed category). *A cartesian closed theory is a category with finite products and exponentials. The formal definition is given in fig. 2.*

Remark 1. We have relaxed the notations for the axioms of a gat: the function symbols do not depend on all the terms they should (the remark 4 tries to discuss this point in details).

Remark 2. The application operator ε could have equivalently been defined as a constant of type $\mathrm{Hom}\,(\langle x \Rightarrow y \wedge x, y\rangle)$ instead of a "meta-operator" (a function symbol in **Gat**).

2.3 Categories with families

Some other theories which are closely related to categories with families (contextual categories and categories with attributes) are presented in annex A.

Remember that some diagram was required to be a pullback in contextual categories and cwa. This condition is rather unnatural in the sense that is very hard to express equationally. *Categories with families* (**Cwf**) is a theory which is equivalent to the previously mentioned ones which was introduced by P. Dybjer in [Dyb96] to deal with this problem. In this presentation, the pullback diagram is a property which is deduced from the definition (cf. proposition 1) rather than imposed by the definition and therefore **Cwf** can be defined in **Gat** which is very nice.

Definition 4 (Category of families of sets (Fam)). *An object of the category* **Fam** *is a family of sets $(B(x))_{x\in A}$ and a morphism with source $(B(x))_{x\in A}$ and target $(B'(x))_{x\in A'}$ is a pair consisting of a function $f : A \to A'$ and a family of functions $g(x) : B(x) \to B'(f(x))$ indexed by $x \in A$.*

The notion of family fits well to dependent types and terms: intuitively a dependent type can be modeled as a set which depends on its context; its is the same for the set of terms which depend on a context. The definition of categories with families formalizes this. Some operators have to be introduced to be able to access the terms which are in the context (q represents the last element of a context and p the beginning of the context). Morphisms of contexts are also introduced to represent substitution in contexts.

Definition 5 (Category with families (cwf)). *A* category with families *consists of:*

Category axioms

Sort symbols:

$$\frac{\mathrm{Ob} : \mathrm{Sort} \qquad x\ y : \mathrm{Ob}}{\mathrm{Hom}(x,y) : \mathrm{Sort}}$$

Operator symbols:

$$\frac{x\ y\ z : \mathrm{Ob} \qquad f : \mathrm{Hom}(y,z) \qquad g : \mathrm{Hom}(x,y)}{f \circ g : \mathrm{Hom}(x,z)}$$

$$\frac{x : \mathrm{Ob}}{\mathrm{id}_x : \mathrm{Hom}(x,x)}$$

Equations:

$$\frac{x\ y\ z\ t : \mathrm{Ob} \qquad f : \mathrm{Hom}(z,t) \qquad g : \mathrm{Hom}(y,z) \qquad h : \mathrm{Hom}(x,y)}{(f \circ g) \circ h = f \circ (g \circ h) : \mathrm{Hom}(x,t)}$$

$$\frac{x\ y : \mathrm{Ob} \qquad f : \mathrm{Hom}(x,y)}{\mathrm{id}_y \circ f = f : \mathrm{Hom}(x,y)}$$

$$\frac{x\ y : \mathrm{Ob} \qquad f : \mathrm{Hom}(x,y)}{f \circ \mathrm{id}_x = f : \mathrm{Hom}(x,y)}$$

Terminal object axioms

Operator symbols:

$$\frac{1 : \mathrm{Ob} \qquad x : \mathrm{Ob}}{\bigcirc_x : \mathrm{Hom}(x,1)}$$

Equations:

$$\frac{x,y : \mathrm{Ob} \qquad f : \mathrm{Hom}(x,y)}{\bigcirc_y \circ f = \bigcirc_y : \mathrm{Hom}(y,1)}$$

$$\frac{}{\mathrm{id}_1 = \bigcirc_1 : \mathrm{Hom}(1,1)}$$

Product axioms

Operator symbols:

$$\frac{x\ y : \mathrm{Ob}}{x \wedge y : \mathrm{Ob}}$$

$$\frac{x\ y\ z : \mathrm{Ob} \qquad f : \mathrm{Hom}(x,y) \qquad g : \mathrm{Hom}(x,z)}{\langle f,g \rangle : \mathrm{Hom}(x, y \wedge z)}$$

$$\frac{x\ y : \mathrm{Ob}}{\pi_{x,y} : \mathrm{Hom}(x \wedge y, x)}$$

$$\frac{x\ y : \mathrm{Ob}}{\pi'_{x,y} : \mathrm{Hom}(x \wedge y, y)}$$

Equations:

$$\frac{x\ y\ z : \mathrm{Ob} \qquad f : \mathrm{Hom}(x,y) \qquad g : \mathrm{Hom}(x,z)}{\pi_{y,z} \circ \langle f,g \rangle = f : \mathrm{Hom}(x,y)}$$

$$\frac{x\ y\ z : \mathrm{Ob} \qquad f : \mathrm{Hom}(x,y) \qquad g : \mathrm{Hom}(x,z)}{\pi'_{y,z} \circ \langle f,g \rangle = g : \mathrm{Hom}(x,z)}$$

$$\frac{x\ y\ z\ t : \mathrm{Ob} \qquad f : \mathrm{Hom}(y,z) \qquad g : \mathrm{Hom}(y,t) \qquad h : \mathrm{Hom}(x,y)}{\langle f,g \rangle \circ h = \langle f \circ h, g \circ h \rangle : \mathrm{Hom}(x, z \wedge t)}$$

$$\frac{x\ y : \mathrm{Ob}}{\mathrm{id}_{x \wedge y} = \left\langle \pi_{x,y}, \pi'_{x,y} \right\rangle : \mathrm{Hom}(x \wedge y, x \wedge y)}$$

Exponentials axioms

Operator symbols:

$$\frac{x\ y : \mathrm{Ob}}{x \Rightarrow y : \mathrm{Ob}}$$

$$\frac{x\ y\ z : \mathrm{Ob} \qquad f : \mathrm{Hom}(x \wedge y, z)}{f^* : \mathrm{Hom}(x, y \Rightarrow z)}$$

$$\frac{x\ y\ z : \mathrm{Ob} \qquad f : \mathrm{Hom}(x, y \Rightarrow z) \qquad g : \mathrm{Hom}(x,y)}{\varepsilon(f,g) : \mathrm{Hom}(x,z)}$$

Equations:

$$\frac{x\ y\ z\ t : \mathrm{Ob} \qquad f : \mathrm{Hom}(y \wedge z, t) \qquad g : \mathrm{Hom}(x,y)}{f^* \circ g = (f \circ \left\langle g \circ \pi_{x,y}, \pi'_{x,z} \right\rangle)^* : \mathrm{Hom}(x, z \Rightarrow t)}$$

$$\frac{x\ y\ z\ t : \mathrm{Ob} \qquad f : \mathrm{Hom}(y, z \Rightarrow t) \qquad g : \mathrm{Hom}(y,z) \qquad h : \mathrm{Hom}(x,y)}{\varepsilon(f,g) \circ h = \varepsilon(f \circ h, g \circ h) : \mathrm{Hom}(x,t)}$$

$$\frac{x\ y\ z : \mathrm{Ob} \qquad f : \mathrm{Hom}(x \wedge y, z) \qquad g : \mathrm{Hom}(x,y)}{\varepsilon(f^*, g) = f \circ \langle \mathrm{id}_x, g \rangle : \mathrm{Hom}(x,z)}$$

$$\frac{x\ y\ z : \mathrm{Ob} \qquad f : \mathrm{Hom}(x, y \Rightarrow z)}{\left(\varepsilon(f \circ \pi_{x,y}, \pi'_{x,y})\right)^* = f}$$

Figure 2. Definition of cccs in gat

– *A base category* $\mathbf{C}$ *whose objects are called* contexts *and whose morphisms are called* substitutions.
– *A functor*[7] $(\mathrm{Type}, \mathrm{Term}) : \mathbf{C}^{\mathrm{op}} \to \mathbf{Fam}$. *We write* $\mathrm{Term}\,(\Gamma) \equiv (\Gamma \vdash A)_{A \in \mathrm{Type}(\Gamma)}$, *where* $\Gamma \in \mathbf{C}$ *and call it the family of* terms indexed by types *in the context* Γ. *Moreover, if* γ *is a morphism of* $\mathbf{C}$ *then the two components of* $\mathrm{Type}\,(\Gamma)$ *and* $\mathrm{Term}\,(\gamma)$ *interpret substitution in types and terms respectively. We write* $A[\gamma]$ *for the application of the first component to a type* A *and* $a[\gamma]$ *for the application of the second component to a term* a.
– *A terminal object* $\diamond$ *called the* empty context.
– *A* context comprehension *operation which to an object* Γ *of* $\mathbf{C}$ *and a type* $A \in \mathrm{Type}\,(\Gamma)$ *associates:*
  - *an object* Γ, A *of* $\mathbf{C}$*;*
  - *a morphism* $p_A^\Gamma : \Gamma, A \to \Gamma$ *of* C *(the* first projection*);*
  - *a term* $q_A^\Gamma \in (\Gamma, A \vdash A[p_A^\Gamma])$ *(the* second projection*).*

  The following universal property holds: for each object Δ *in* $\mathbf{C}$, *morphism* $\gamma : \Delta \to \Gamma$, *and term* $M \in \Delta \vdash A[\gamma]$, *there is a unique morphism* $\delta = \langle \gamma, M \rangle : \Delta \to \Gamma, A$ *such that* $p_A^\Gamma \circ \delta = \gamma$ *and* $q_A^\Gamma[\delta] = a$.

  Formally, we can define the rule that must hold by defining cwfs in gat (see fig. 3).

Remark 3. The operators $\vdash$, $\to$, $[\,]$, etc. are not the same as the on used in the definition of **Gat** (fig. 1), we reuse those notations with different meanings (in fact, their meanings are closely related but are not on the same "meta-level").

Remark 4 (About the importance of the syntax). To alleviate the notations, some type indications where or will be omitted, as discussed in [Car86] (§10, *Informal syntax*). Cartmell namely distinguishes between two types of what he calls "informal syntax" (i.e. a syntax where the operators do not explicitly depend on all the elements of the context where they were defined in order to have lighter notations):

– the omission of some formally necessary variables (e.g. writing $\mathrm{App}\,(M, N)$ instead of $\mathrm{App}_{\Pi(A,B)}\,(M, N)$);
– the overloading of operators (e.g. writing p instead of p_A^Γ).

The justification of the dropping of those arguments is that they can be recovered from the context. For example we can simply write p instead of p_Γ^A because if we know that a combinator $p_?^?$ has type $\Gamma, A \to \Gamma$ then we know that it must be p_Γ^A (we are able to recover Γ and A from the context). However there is no precise theorem which justifies that (Cartmell only gives a necessary condition which is that the implicit parameters of a function symbol must occur implicitly in the context of the introductory rule of the concerned symbol).

The rules given in fig. 3 are given in informal syntax. However, we might sometimes write explicitly p_A^Γ and q_A^Γ instead of p and q for the sake of clarity. We believe that this syntax in unambiguous (i.e. that implicit arguments can be recovered from the context) but this has yet to be proven – or the proofs should be done more precisely with the formal syntax.

Moreover, since we are only human beings, we do not want to deal with loads of indexes and want to keep our proofs a bit readable. Therefore we might omit the index from $\circ$, $[\,]$ or even from App. This is not correct from a theoretical point of view because some properties might be verified in one presentation but not in the other one. We have only used it as a relatively short way to write terms but the proofs have been checked using the syntax given in fig. 3.

Those considerations might not seem to to be so much interesting but actually this kind of syntactic details matters. As we will see later, the proof of the equivalence between $\mathbf{Cwf_{LF}}$ and **LF** would have been much simpler with an untyped operator App. However we need this type

[7] A functor φ is a morphism between two categories $\mathbf{C}_1$ and $\mathbf{C}_2$ which sends objects of $\mathbf{C}_1$ to objects of $\mathbf{C}_2$ and morphisms of $\mathbf{C}_1$ to morphisms of $\mathbf{C}_2$ such that if f is a morphism from A to B in $\mathbf{C}_1$ then $\varphi(f)$ is a morphism from $\varphi(A)$ to $\varphi(B)$ in $\mathbf{C}_2$; moreover φ is required to be compatible with composition and identity.

Rules for the category C

Sort symbols:

$$\mathrm{Ctxt} : \mathrm{Sort} \quad \text{(C-I)}$$

$$\frac{\Delta\ \Gamma : \mathrm{Ctxt}}{\Delta \to \Gamma : \mathrm{Sort}}\ \text{(M-I)}$$

Operator symbols:

$$\frac{\Theta\ \Delta\ \Gamma : \mathrm{Ctxt} \qquad \gamma : \Delta \to \Gamma \qquad \delta : \Theta \to \Delta}{\gamma \circ_\Delta \delta : \Theta \to \Gamma}\ \text{(M-C)}$$

$$\frac{\Gamma : \mathrm{Ctxt}}{\mathrm{id} : \Gamma \to \Gamma}\ \text{(M-Id)}$$

Equations:

$$\frac{\Gamma\ \Delta\ \Theta\ \Psi : \mathrm{Ctxt} \qquad \gamma : \Theta \to \Psi \qquad \delta : \Delta \to \Theta \qquad \theta : \Gamma \to \Delta}{(\gamma \circ_\Theta \delta) \circ_\Delta \theta = \gamma \circ_\Theta (\delta \circ_\Delta \theta) : \Gamma \to \Psi}\ \text{(M-Assoc)}$$

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt} \qquad \gamma : \Gamma \to \Delta}{\mathrm{id} \circ_\Delta \gamma = \gamma : \Gamma \to \Delta}\ \text{(M-Id-L)}$$

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt} \qquad \gamma : \Gamma \to \Delta}{\gamma \circ_\Gamma \mathrm{id} = \gamma : \Gamma \to \Delta}\ \text{(M-Id-R)}$$

Rules for the functor (Type, Term)

Sort symbols:

$$\frac{\Gamma : \mathrm{Ctxt}}{\mathrm{Type}\ (\Gamma) : \mathrm{Sort}}\ \text{(Ty-I)}$$

$$\frac{\Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\ (\Gamma)}{\Gamma \vdash A : \mathrm{Sort}}\ \text{(Ty-Abs)}$$

Operator symbols:

$$\frac{\Delta\ \Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\ (\Gamma) \qquad \gamma : \Delta \to \Gamma}{A[\gamma]_\Gamma : \mathrm{Type}\ (\Delta)}\ \text{(Ty-S)}$$

$$\frac{\Delta\ \Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\ (\Gamma) \qquad M : \Gamma \vdash A \qquad \gamma : \Delta \to \Gamma}{M[\gamma]^A_\Gamma : \Delta \vdash A[\gamma]_\Gamma}\ \text{(Tm-S)}$$

Equations:

$$\frac{\Gamma\ \Delta\ \Theta : \mathrm{Ctxt} \qquad A : \mathrm{Type}\ (\Theta) \qquad \gamma : \Delta \to \Theta \qquad \delta : \Gamma \to \Delta}{A[\gamma \circ_\Delta \delta]_\Theta = A[\gamma]_\Theta[\delta]_\Delta : \mathrm{Type}\ (\Gamma)}\ \text{(Ty-S-C)}$$

$$\frac{\Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\ (\Gamma)}{A[\mathrm{id}]_\Gamma = A : \mathrm{Type}\ (\Gamma)}\ \text{(Ty-S-Id)}$$

$$\frac{\Gamma\ \Delta\ \Theta : \mathrm{Ctxt} \qquad A : \mathrm{Type}\ (\Theta) \qquad M : \Theta \vdash A \qquad \delta : \Gamma \to \Delta \qquad \gamma : \Delta \to \Theta}{M[\gamma \circ_\Delta \delta]^A_\Theta = M[\gamma]^A_\Theta[\delta]^{A[\gamma]}_\Delta : \Gamma \vdash A[\gamma \circ_\Delta \delta]_\Theta}\ \text{(Tm-S-C)}$$

$$\frac{\Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\ (\Gamma) \qquad M : \Gamma \vdash A}{M[\mathrm{id}]_\Gamma = M : \Gamma \vdash A}\ \text{(Tm-S-Id)}$$

Rules for the terminal object

Operator symbols:

$$\frac{}{\diamond : \mathrm{Ctxt}}\ \text{(C-Emp)}$$

$$\frac{\Gamma : \mathrm{Ctxt}}{\langle\rangle : \Gamma \to \diamond}\ \text{(M-Emp)}$$

Equations:

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt} \qquad \gamma : \Gamma \to \Delta}{\langle\rangle \circ_\Delta \gamma = \langle\rangle : \Gamma \to \diamond}\ \text{(M-Emp-L)}$$

$$\frac{}{\mathrm{id} = \langle\rangle : \diamond \to \diamond}\ \text{(M-Emp-Id)}$$

Rules for context comprehension

Operator symbols:

$$\frac{\Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\ (\Gamma)}{\Gamma, A : \mathrm{Ctxt}}\ \text{(C-Ext)}$$

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt} \qquad A : \mathrm{Type}\ (\Delta) \qquad \gamma : \Gamma \to \Delta \qquad M : \Gamma \vdash A[\gamma]_\Delta}{\langle \gamma, M \rangle : \Gamma \to \Delta, A}\ \text{(M-Ext)}$$

$$\frac{\Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\ (\Gamma)}{p : \Gamma, A \to \Gamma}\ \text{(M-E-L)}$$

$$\frac{\Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\ (\Gamma)}{q : \Gamma, A \vdash A[p]_\Gamma}\ \text{(M-E-R)}$$

Equations:

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt} \qquad A : \mathrm{Type}\ (\Gamma) \qquad \gamma : \Gamma \to \Delta \qquad M : \Gamma \vdash A[\gamma]_\Delta}{p \circ_{\Gamma,A} \langle \gamma, M \rangle = \gamma : \Gamma \to \Delta}\ \text{(M-C-L)}$$

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt} \qquad A : \mathrm{Type}\ (\Gamma) \qquad \gamma : \Gamma \to \Delta \qquad M : \Gamma \vdash A[\gamma]_\Delta}{q[\langle \gamma, M \rangle]_{\Delta,A} = M : \Gamma \vdash A[\gamma]_\Delta}\ \text{(M-C-R)}$$

$$\frac{\Gamma\ \Delta\ \Theta : \mathrm{Ctxt} \qquad \gamma : \Delta \to \Gamma \qquad \delta : \Gamma \to \Theta \qquad A : \mathrm{Type}\ (\Theta) \qquad M : \Gamma \vdash A\,[\delta]_\Theta}{\langle \delta, M \rangle \circ_\Gamma \gamma = \left\langle \delta \circ_\Gamma \gamma, M[\gamma]^{A[\delta]_\Theta}_\Gamma \right\rangle : \Delta \to \Theta}\ \text{(M-Ext-S)}$$

$$\frac{\Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\ (\Gamma)}{\mathrm{id} = \langle p, q \rangle : \Gamma, A \to \Gamma, A}\ \text{(M-Ext-Id)}$$

Figure 3. Definition of cwfs in gat

information. The reason for that is that we want the term-model (i.e. the model of syntactic terms) of the theory of $\mathbf{Cwf_{LF}}$ to be initial in the category of models[8] i.e. that for every model $\mathscr{M}$ there exists one and only one morphism (i.e. an interpretation which preserves the judgments) from the term model $\mathscr{M}_0$ to $\mathscr{M}$. This is important because we want a property to be true in the term model (in particular the decidability of the equality that we are going to prove) iff it is true in all models. We did not have enough time to write this in details but the idea to prove that $\mathscr{M}_0$ is initial is that we are going to need to interpret the term model $\mathscr{M}_0$ in a model $\mathscr{M}$ and this will have to be done recursively[9]. Moreover, terms will have to be interpreted with an interpretation which depends on their type (and certainly also on the context). Therefore we will have to guess the types recursively. For example, if we write $[\![\ ]\!]_A^\Gamma$ this interpretation, we will have

$$\left[\!\left[\mathrm{App}_{\Pi(A,B)}\left(M,N\right)\right]\!\right]^\Gamma_{B\left[\langle \mathrm{id},N\rangle\right]_{\Gamma,A}} \equiv \mathrm{App}_{\Pi(A,B)}\left([\![M]\!]_B^{\Gamma,A}, [\![N]\!]_A^\Gamma\right)$$

Clearly the "inferred" type information $B\left[\langle \mathrm{id},N\rangle\right]_{\Gamma,A}$ is not enough to recover the types A and B. That is why we need to have a typed application operator. As we will see later (§3.3), this will have important consequences on the complexity of the proof.

The notion of category with families is more natural than the notion of category with attributes (def. 30) because the pullback-condition does not need to be imposed; it is rather deduced from the definition. This is why cwf can be simply defined in gat, contrarily to cwa and justifies the introduction of this theory: it seems to be a rather natural and elegant categorical model of dependent type theory which can be represented inside itself (of course we cannot prove internally its consistency). However this remains to be proven and this is precisely one of the goals of this paper to show the equivalence between the two theories.

Proposition 1. *With the notations of the definition 5, the diagram*

$$\begin{array}{ccc} \Delta, A[\gamma] & \xrightarrow{\langle \gamma\circ p^\Delta_{A[\gamma]}, q^\Delta_{A[\gamma]}\rangle} & \Gamma, A \\ \downarrow{\scriptstyle p^\Delta_{A[\gamma]}} & & \downarrow{\scriptstyle p^\Gamma_A} \\ \Delta & \xrightarrow[\gamma]{} & \Gamma \end{array}$$

is a pullback.

The proof is given in annex, section A.3.

Proposition 2. Cwf *can be defined as a gat.*

Proof. The proof is sketched in [Dyb96]. □

We will now introduce the notion of *weakening* which will turn out later to be useful to "remove useless informations from contexts" (that is why it is called weakening, because it is related to the weakening rules of the logics).

Definition 6 (Weakening). *If Γ and Δ are contexts, A an element of* $\mathrm{Type}\left(\Gamma\right)$*, and $\gamma : \Delta \to \Gamma$ a morphism, the context morphism $\tilde{p}\left(\gamma, A\right) : \Delta, A[\gamma] \to \Gamma, A$ called the* weakening *of γ by A is defined by*

$$\tilde{p}\left(\gamma, A\right) \equiv \langle \gamma \circ p, q\rangle$$

[8] The category of models of a theory is the category where the objects are the models of the theory and morphisms are the functions between models which are compatible with the interpretations i.e. φ is a morphism between two models $\mathscr{M}_1$ and $\mathscr{M}_2$ iff for all term t we have $\mathcal{I}_2\left(t\right) = \varphi\left(\mathcal{I}_1\left(t\right)\right)$ where $\mathcal{I}_1$ and $\mathcal{I}_2$ are the interpretations of the theory respectively in $\mathscr{M}_1$ and $\mathscr{M}_2$.

[9] There might be an other way to do that which would not require the combinators to be indexed by uninferable types but we did not find an easy one.

We define (fig. 7) a notion of cwf supporting Π-types (product types, intended to be types of functions). Some operators are introduced to be able to represent functions (in the λ-calculus sense) and their type in cwfs. For example, if, in a context Γ, A and B are both types then $\Pi(A, B)$ (introduced by rule (Exp)) is the type of functions which, given an argument of type A return a term of type B. Some constructions are also introduced for the terms, those should be quite natural for anyone who is familiar with λ-calculus. For example, if whenever we add a term of type A to the context, M is a term of type B then $\lambda(M)$ can be seen as the function which, whenever given a term of type A as argument returns the term M where the "hole" in M, introduced by the supposition of a term of type A in the context when typing M, is filled with N (this is expressed by rules (Abs) and (Π-C)). This is clearly closely related to the abstraction in λ-calculus: if whenever we suppose that x is a variable of type A, M is a term of type B then $\lambda x.M$ is a term of type $\Pi x : A.B$ and can be seen as the function which, to each term N of type A, associates the term $M[N/x]$. From this comparison, we can see that the categorical syntax (of cwfs) is more natural than the syntax of λ-calculus and actually – this is going to be proven in section 3.3 – the two theories are equivalent (which means that, modulo interpretation, if a theorem holds in one theory then it holds in the other one). In particular, there is no need of variables to express bindings which is convenient because usually λ-terms are considered modulo α-conversion (i.e. renaming of bound variables). Moreover, substitution in cwfs is not a meta-operation like in λ-calculus. Concerning this point, it is certainly closer to λ-calculi with explicit substitutions but we did not want to use those since they have been much less studied than without. From this point of view the categorical syntax seems to be much closer to the essence of the λ-calculus. This is one of the main reasons why people try to define categorical models (**Cwf** is one of them).

The rules (Π-S), (λ-S) and (App-S) are here to define inductively the application of a morphism to syntactic terms build from the new operators (Π, λ and App). Finally, (Π-η) is the categorical formulation of the usual η-conversion rule[10].

The formulation of some of the rules might seem surprisingly complicated. For example, why did we not simply write $\mathrm{App}_{\Pi(A,B)}(M, N) : \Gamma \vdash B$ in the conclusion of the rule (App) (without the substitution on B)? This is because we wanted those rules to be suitable for dependent types, which are going to be introduced in the next definition.

Definition 7 (Cwfs supporting Π-types). *A cwf* **C** *supports* Π*-types if the derivations rules and equations of the figure 7 hold.*

To have dependent types, we add an operator which injects (modulo equality) terms into types: for each term M, $\mathsf{El}\ M$ is a type and $\mathsf{El}\ M = \mathsf{El}\ M'$ iff $M = M'$. Thanks to this operator, types can depend on terms. For example, we could imagine an extension of our typing system which has natural numbers and where, for each $n : \mathsf{Nat}$, $\mathsf{List}(n)$ is the type of lists of length n (the type $\mathsf{List}(n)$ *depends* on the term n). This would be useful to define functions which are guaranteed to return lists of same length as their argument for example (those functions would have type $\Pi(\mathsf{List}(n), \mathsf{List}(n))$).

Definition 8 (Cwfs supporting LFs). *A cwf* **C** *supports LFs if it supports* Π*-types and the derivation rules of the figure 8 hold.*

The theory of cwfs supporting LFs will be named $\mathbf{Cwf_{LF}}$.

Remark 5 (Why Star*).* Of course the Star operator might seem useless here since the rule (Star) can only be used before a rule (Elem). Both rules could have been merged into the rule

$$\frac{\Gamma : \mathrm{Ctxt}}{\mathrm{Elem}(M) : \mathrm{Type}(\Gamma)}$$

However our presentation can be much more easily generalized (in the case we would want to introduce a sort of natural integers in the theory for example).

[10] In untyped λ-calculus this equality would be simply written $M = \lambda x.Mx$.

Operator symbols:

$$\frac{\Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\,(\Gamma) \qquad B : \mathrm{Type}\,(\Gamma, A)}{\Pi(A,B) : \mathrm{Type}\,(\Gamma)}(\textsc{Exp})$$

$$\frac{\Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\,(\Gamma) \qquad B : \mathrm{Type}\,(\Gamma, A) \qquad M : \Gamma, A \vdash B}{\lambda(M) : \Gamma \vdash \Pi(A,B)}(\textsc{Abs})$$

$$\frac{\Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\,(\Gamma) \qquad B : \mathrm{Type}\,(\Gamma, A) \qquad M : \Gamma \vdash \Pi(A,B) \qquad N : \Gamma \vdash A}{\mathrm{App}_{\Pi(A,B)}\,(M,N) : \Gamma \vdash B[\langle \mathrm{id}, N\rangle]_{\Gamma,A}}(\textsc{App})$$

Equations:

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt} \qquad A : \mathrm{Type}\,(\Gamma) \qquad B : \mathrm{Type}\,(\Gamma, A) \qquad \gamma : \Delta \to \Gamma}{\Pi\,(A,B)\,[\gamma]_\Gamma = \Pi\left(A[\gamma]_\Gamma, B\,[\langle \gamma \circ p, q\rangle]_{\Gamma,A}\right) : \mathrm{Type}\,(\Delta)}(\Pi\text{-S})$$

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt} \qquad A : \mathrm{Type}\,(\Gamma) \qquad M : \Gamma \vdash A \qquad \gamma : \Delta \to \Gamma}{\lambda(M)[\gamma]_\Gamma = \lambda\left(M\,[\langle \gamma \circ p, q\rangle]_{\Gamma,A}\right) : \mathrm{Type}\,(\Delta)}(\lambda\text{-S})$$

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt} \qquad A : \mathrm{Type}\,(\Gamma) \qquad B : \mathrm{Type}\,(\Gamma, A) \qquad N : \Gamma \vdash A \qquad M : \Gamma \vdash \Pi\,(A,B) \qquad \gamma : \Delta \to \Gamma}{\mathrm{App}_{\Pi(A,B)}\,(M,N)\,[\gamma]_\Gamma = \mathrm{App}_{\Pi(A,B)}\left(M\,[\gamma]_\Gamma^{\Pi(A,B)}, N\,[\gamma]_\Gamma^{A}\right) : \Delta \vdash B\left[\left\langle \mathrm{id}, N\,[\gamma]_\Gamma^{A}\right\rangle\right]_{\Delta, A[\gamma]_\Gamma}}(\textsc{App-S})$$

$$\frac{\Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\,(\Gamma) \qquad B : \mathrm{Type}\,(\Gamma, A) \qquad N : \Gamma \vdash A \qquad M : \Gamma \vdash \Pi\,(A,B)}{\mathrm{App}_{\Pi(A,B)}\,(\lambda\,(M)\,, N) = M\,[\langle \mathrm{id}, N\rangle]_{\Gamma,A} : B\,[\langle \mathrm{id}, N\rangle]_{\Gamma,A}}(\Pi\text{-C})$$

$$\frac{\Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\,(\Gamma) \qquad B : \mathrm{Type}\,(\Gamma, A) \qquad M : \Gamma \vdash \Pi(A,B)}{\lambda\left(\mathrm{App}_{\Pi\left(A, B[p]_\Gamma\right)}\left(M\,[p]_\Gamma^{\Pi(A,B)}, q\right)\right) = M : \Gamma \vdash \Pi(A,B)}(\Pi\text{-}\eta)$$

Figure 4. Rules for Π-types in cwfs

Operator symbols:

$$\frac{\Gamma : \mathrm{Ctxt}}{\mathrm{Star} : \mathrm{Type}\,(\Gamma)}(\textsc{Star}) \qquad \frac{\Gamma : \mathrm{Ctxt} \qquad M : \Gamma \vdash \mathrm{Star}}{\mathrm{Elem}\,(M) : \mathrm{Type}\,(\Gamma)}(\textsc{Elem})$$

Equations:

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt} \qquad \gamma : \Delta \to \Gamma}{\mathrm{Star}\,[\gamma]_\Gamma = \mathrm{Star} : \mathrm{Type}\,(\Delta)}(\textsc{Star-S})$$

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt} \qquad A : \mathrm{Type}\,(\Gamma) \qquad M : \Gamma \vdash A \qquad \mathrm{Star} : \mathrm{Type}\,(\Gamma) \qquad \gamma : \Delta \to \Gamma}{\mathrm{Elem}\,(M)\,[\gamma]_\Gamma = \mathrm{Elem}\left(M\,[\gamma]_\Gamma\right) : \mathrm{Type}\,(\Delta)}(\textsc{Elem-S})$$

Figure 5. Additional rules for cwfs supporting LFs

2.4 Logical frameworks

We first present logical frameworks as defined in [CPT03] (we will use later theorems proven in this paper). It is a λ-calculus with dependent types without pairing (or currifying) operations since we do not need them to encode context in types as previously mentioned. Most of the rules used to define should seem quite natural to anyone who is already familiar with simply-typed λ-calculus. Those rules are direct extensions of the rules of simply-typed λ-calculus to support dependent types. Here "dependent" means that a type can depend on a term. This is possible thank to the operator El which "injects" terms into types (for each term M, there is a corresponding type $\mathsf{El}\ M$). In a more elaborated type system with natural numbers (Nat) we could imagine to define, for all natural number n, the type $\mathsf{List}\,(n)$ of lists of length n by the rules

$$\frac{\Gamma : \mathrm{Ctxt} \qquad n : \mathsf{Nat}}{\mathsf{List}\,(n) : \mathrm{Type}\,(\Gamma)}$$

$$\frac{\Gamma : \mathrm{Ctxt}}{[] : \Gamma \vdash \mathsf{List}\,(\mathsf{O})} \qquad \frac{\Gamma : \mathrm{Ctxt} \qquad n : \mathsf{Nat} \qquad l : \Gamma \vdash \mathsf{List}\,(n) \qquad A : \mathrm{Type}\,(\Gamma) \qquad M : \Gamma \vdash A}{l :: M : \Gamma \vdash \mathsf{List}\,(\mathsf{S}\,(n))}$$

The type $\mathsf{List}\,(n)$ *depends* on the term n. This kind of types are present in proof checkers such as COQ and turn out to be very useful and convenient to manipulate.

Definition 9 (Logical framewok (LF)). *A logical framework is defined by three classes*

- terms *:* $M, N \quad ::= \quad x \quad | \quad \lambda x.M \quad | \quad \mathrm{App}_{\Pi x:A.B}\,(M, N)$*;*
- types *:* $A, B \quad ::= \quad \star \quad | \quad \mathsf{El}\ M \quad | \quad \Pi x : A.B$*;*
- contexts *:* $\Gamma \quad ::= \quad \diamond \quad | \quad \Gamma, x : A$*;*

which are such that the inference rules of fig. 6 hold.

Remark 6. App, π *and* π' *could have equivalently been defined as constants of the language.*

Remark 7. In the following we might write MN *for* $\mathrm{App}_{\Pi x:A.B}\,(M, N)$ *to have lighter notations. However it is important to understand that the type index of the application is theoretically necessary to have the equivalence with* $\mathbf{Cwf_{LF}}$ *(cf. remark 4), it is just an informal and concise way of writing the application.*

Remark 8. The rules of defining LFs (fig. 6) are very similar to the rules defining gats (fig. 1). And actually, we will see later that the theory **LF** *is equivalent to the theory* **Gat**.

Definition 10 (Free, defined variables). *The set of variables* $\mathrm{FV}\,(M)$ free *in a term* M *is defined inductively by*

$$\begin{aligned} \mathrm{FV}\,(x) &\equiv \{x\} \\ \mathrm{FV}\,(\lambda x.M) &\equiv \mathrm{FV}\,(M) \setminus \{x\} \\ \mathrm{FV}\,(\mathrm{App}_{\Pi x:A.B}\,(M, N)) &\equiv \mathrm{FV}\,(M) \cup \mathrm{FV}\,(N) \\ \mathrm{FV}\,(\langle M, N\rangle) &\equiv \mathrm{FV}\,(M) \cup \mathrm{FV}\,(N) \end{aligned}$$

The set of variables $\mathrm{DV}\,(\Gamma)$ defined *in a context is defined inductively by*

$$\begin{aligned} \mathrm{DV}\,(x : A) &\equiv \{x\} \\ \mathrm{DV}\,(\Gamma, x : A) &\equiv \mathrm{DV}\,(\Gamma) \cup \{x\} \end{aligned}$$

We will now prove some lemmata which show that "everything is going on well" i.e. that the rules are natural in the sense that they are compatible with most operations among which substitution and reduction. Those are going to be used later to show that the equality is decidable in LFs. The proofs of most of those lemmata are given in annex, in section B.1. They are mostly inductions on the derivation of the hypothesis.

Contexts

$$\frac{}{\diamond \vdash}(\text{C-Emp}) \quad \frac{\Gamma \vdash A \qquad x \notin \mathrm{DV}(\Gamma)}{\Gamma, x : A \vdash}(\text{C-Ext})$$

Types

$$\frac{\Gamma \vdash}{\Gamma \vdash \star}(\textsc{Star}) \quad \frac{\Gamma \vdash M : \star}{\Gamma \vdash \mathsf{El}\ M}(\textsc{Elem})$$

$$\frac{\Gamma, x : A \vdash B}{\Gamma \vdash \Pi x : A.B}(\textsc{Exp})$$

Type equalities

$$\frac{\Gamma \vdash A}{\Gamma \vdash A = A}(\text{Ty-Eq-Refl}) \quad \frac{\Gamma \vdash A = B}{\Gamma \vdash B = A}(\text{Ty-Eq-Sym}) \quad \frac{\Gamma \vdash A = B \qquad \Gamma \vdash B = C}{\Gamma \vdash A = C}(\text{Ty-Eq-Trans})$$

$$\frac{\Gamma \vdash M = N : \star}{\Gamma \vdash \mathsf{El}\ M = \mathsf{El}\ N}(\text{El-Eq-C})$$

$$\frac{\Gamma \vdash \Pi x : A.B \qquad \Gamma \vdash A = A' \qquad \Gamma, x : A \vdash B = B'}{\Gamma \vdash \Pi x : A.B = \Pi x : A'.B'}(\Pi\text{-Eq-C})$$

Terms

$$\frac{\Gamma, x : A, \Delta \vdash}{\Gamma, x : A, \Delta \vdash x : A}(\textsc{Var}) \quad \frac{\Gamma \vdash M : A \qquad \Gamma \vdash A = B}{\Gamma \vdash M : B}(\text{Tm-Conv})$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : \Pi x : A.B}(\textsc{Abs}) \quad \frac{\Gamma \vdash M : \Pi x : A.B \qquad \Gamma \vdash N : A}{\Gamma \vdash \mathrm{App}_{\Pi x:A.B}(M, N) : B[N/x]}(\textsc{App})$$

Term equalities

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash M = M : A}(\text{Tm-Eq-Refl}) \quad \frac{\Gamma \vdash M = M' : A}{\Gamma \vdash M' = M : A}(\text{Tm-Eq-Sym}) \quad \frac{\Gamma \vdash M = M' : A \qquad \Gamma \vdash M' = M'' : A}{\Gamma \vdash M = M'' : A}(\text{Tm-Eq-Trans})$$

$$\frac{\Gamma \vdash M = N : A \qquad \Gamma \vdash A = B}{\Gamma \vdash M = N : B}(\text{Tm-Eq-Conv}) \qquad \frac{\Gamma \vdash \mathrm{App}_A(M, N) : B \qquad \Gamma \vdash A = A'}{\Gamma \vdash \mathrm{App}_A(M, N) = \mathrm{App}_{A'}(M, N) : B}(\text{App-Eq-Conv})$$

$$\frac{\Gamma \vdash \lambda x.M : \Pi x : A.B \qquad \Gamma, x : A \vdash M = M' : B}{\Gamma \vdash \lambda x.M = \lambda x.M' : \Pi x : A.B}(\Pi\text{-I-Eq}) \quad \frac{\Gamma \vdash M : \Pi x : A.B \qquad \Gamma \vdash N = N' : A}{\Gamma \vdash \mathrm{App}_{\Pi x:A.B}(M, N) = \mathrm{App}_{\Pi x:A.B}\left(M, N'\right) : B\left[N/x\right]}(\textsc{App-Eq})$$

$$\frac{\Gamma \vdash \lambda x.M : \Pi x : A.B \qquad \Gamma \vdash N : A}{\Gamma \vdash \mathrm{App}_{\Pi x:A.B}((\lambda x.M), N) = M[N/x] : B[N/x]}(\Pi\text{-C})$$

$$\frac{\Gamma \vdash M : \Pi x : A.B}{\Gamma \vdash M = \lambda x.\mathrm{App}_{\Pi x:A.B}(M, x) : \Pi x : A.B}(\Pi\text{-}\eta)$$

Figure 6. Typing rules in LF

Types

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash \star\,[M/x] = \star}(\textsc{Star-S})$$

$$\frac{\Gamma, x : A \vdash M : \star \qquad \Gamma \vdash N : A}{\Gamma \vdash (\mathsf{El}\ M)\,[N/x] = \mathsf{El}\ \ (M\,[N/x])}(\textsc{El-S})$$

$$\frac{\Gamma, x : A \vdash \Pi y : B.C \qquad \Gamma \vdash M : A \qquad x \notin \mathrm{DV}\,(M)}{\Gamma \vdash (\Pi y : B.C) = \Pi y : (B\,[M/x])\,.\,(C\,[M/x])}(\Pi\text{-S})$$

Terms

$$\frac{\Gamma \vdash x : A \qquad \Gamma \vdash M : A}{\Gamma \vdash x\,[M/x] = M : A}(\textsc{Var-S}) \quad \frac{\Gamma, x : A \vdash y : B \qquad \Gamma \vdash M : A \qquad y \neq x}{\Gamma \vdash y\,[M/x] = y : B\,[M/x]}(\textsc{Var-S'})$$

$$\frac{\Gamma, y : B \vdash \lambda x.M \qquad \Gamma \vdash N : B \qquad x \notin \mathrm{DV}\,(M)}{\Gamma \vdash (\lambda x.M)\,[N/y] = \lambda x.\,(M\,[N/y])}(\textsc{Abs-S})$$

$$\frac{\Gamma, y : B \vdash \mathrm{App}_{\Pi x:C.D}\,(M, N) : A \qquad \Gamma \vdash P : B}{\Gamma \vdash (\mathrm{App}_{\Pi x:C.D}\,(M, N))\,[P/y] = \mathrm{App}_{\Pi x:C[P/y].D[P/y]}\,((M\,[P/y])\,,(N\,[P/y]))}(\textsc{App-S})$$

Figure 7. Rules for substitutions in LF

Lemma 1 (Well-formedness). *Let $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$ be a context, A and B two types and M a term. We shall write $\lceil\Gamma\rceil_i \equiv x_1 : A_1, \ldots, x_i : A_i$. The following rules hold*

1. *if $\Gamma \vdash$ is derivable then for all i such that $1 < i \leq n$, $x_i \notin \mathrm{DV}\,(\lceil\Gamma\rceil_{i-1})$, $\mathrm{FV}\,(A_i) \subset \mathrm{DV}\,(\lceil\Gamma\rceil_{i-1})$ and $\lceil\Gamma\rceil_{i-1} \vdash A_i$ appears in the derivation;*
2. *if $\Gamma \vdash \mathcal{J}$ is derivable then $\mathrm{FV}\,(\mathcal{J}) \subset \mathrm{DV}\,(\Gamma)$ and $\Gamma \vdash$ appears in the derivation, where $\mathcal{J}$ is either a type, a typed term, an equality between types or a typed equality between terms.*

Lemma 2. *The following rules hold*

1. *if $\Gamma \vdash A = B$ is derivable then $\Gamma \vdash A$ and $\Gamma \vdash B$ are derivable;*
2. *if $\Gamma \vdash M = N : A$ is derivable then $\Gamma \vdash M : A$ and $\Gamma \vdash N : A$ derivable.*

Definition 11 (Concatenation of two contexts). *The concatenation (written ",") of a context and a variable-type couple can be extended to the concatenation of two contexts (that we shall write "$\tilde{,}$ " here for clarity but that we will write "," in the remaining of the paper) by*

$$\Gamma \mathbin{\tilde{,}} \diamond \equiv \Gamma$$
$$\Gamma \mathbin{\tilde{,}} (\Delta, x : A) \equiv (\Gamma \mathbin{\tilde{,}} \Delta)\,, x : A$$

Definition 12 (Substitution on contexts). *The substitution can be extended on contexts by*

$$\diamond\,[M/x] \equiv \diamond$$
$$(\Gamma, x : A)\,[M/x] \equiv \Gamma$$
$$(\Gamma, y : A)\,[M/x] \equiv \Gamma\,[M/x]\,, y : A\,[M/x]$$

Lemma 3 (Weakening). *If $x \notin \mathrm{DV}\,(\Gamma) \cup \mathrm{DV}\,(\Delta)$ and $\Gamma \vdash C$ then*

1. *if Γ, Δ is a context then $\Gamma, x : C, \Delta$ is a context;*
2. *if $\Gamma, \Delta \vdash A$ then $\Gamma, x : C, \Delta \vdash A$;*
3. *if $\Gamma, \Delta \vdash A = B$ then $\Gamma, x : C, \Delta \vdash A = B$;*
4. *if $\Gamma, \Delta \vdash M : A$ then $\Gamma, x : C, \Delta \vdash M : A$;*
5. *if $\Gamma, \Delta \vdash M = N : A$ then $\Gamma, x : C, \Delta \vdash M = N : A$.*

Lemma 4. *The following rule holds*

$$\frac{\Gamma \vdash M : \Pi x : A.B \qquad \Gamma \vdash N = N' : A}{\Gamma \vdash \mathrm{App}_{\Pi x:A.B}\,(M, N) = \mathrm{App}_{\Pi x:A.B}\,(M, N') : B\,[N/x]}$$

Lemma 5. *If $\Gamma, x : A, \Delta \vdash x : B$ then $\Gamma \vdash A = B$.*

Lemma 6 (Soundness of the substitution). *If $\Gamma \vdash N : B$ is derivable then*

1. *if $\Gamma, x : B, \Delta$ is a context then $\Gamma, \Delta[N/x]$ is a context;*
2. *if $\Gamma, x : B, \Delta \vdash A$ is derivable then $\Gamma, \Delta[N/x] \vdash A[N/x]$ is derivable;*
3. *if $\Gamma, x : B, \Delta \vdash A = A'$ is derivable then $\Gamma, \Delta[N/x] \vdash A[N/x] = A'[N/x]$ is derivable;*
4. *if $\Gamma, x : B, \Delta \vdash M : A$ is derivable then $\Gamma, \Delta[N/x] \vdash M[N/x] : A[N/x]$ is derivable;*
5. *if $\Gamma, x : B, \Delta \vdash M = M' : A$ is derivable then $\Gamma, \Delta[N/x] \vdash M[N/x] = M'[N/x] : A[N/x]$ is derivable.*

Definition 13 (β-reduction). *The β-reduction relation, written $\to_\beta$ is defined inductively on terms by*

$$\frac{}{(\lambda x.M)\, N \to_\beta M[N/x]}(\beta\text{-Red-App}) \quad \frac{M \to_\beta M'}{\lambda x.M \to_\beta \lambda x.M'}(\beta\text{-Red-Abs-C})$$
$$\frac{M \to_\beta M'}{MN \to_\beta M'N}(\beta\text{-Red-App-C-L}) \quad \frac{N \to_\beta N'}{MN \to_\beta MN'}(\beta\text{-Red-App-C-R})$$

The transitive closure of $\to_\beta$ will be written $\xrightarrow{\beta}$.

The β-convertibility *relation, written $\overset{\beta}{=}$, is the reflexive-, transitive- and symmetric-closure of $\xrightarrow{\beta}$ i.e.*

1. *(β-Eq-Refl): $M \overset{\beta}{=} M$;*
2. *(β-Eq-Ext-R): if $M \overset{\beta}{=} N$ and $N \to_\beta N'$ then $M \overset{\beta}{=} N'$;*
3. *(β-Eq-Ext-L): if $M \overset{\beta}{=} N$ and $N_\beta \leftarrow N'$ then $M \overset{\beta}{=} N'$.*

Lemma 7. *The relation $\overset{\beta}{=}$ is an equivalence relation.*

Lemma 8. *The substitution preserves β-convertibility:*

1. *if $M \overset{\beta}{=} M'$ then $M[N/x] \overset{\beta}{=} M'[N/x]$;*
2. *if $N \overset{\beta}{=} N'$ then $M[N/x] \overset{\beta}{=} M[N'/x]$.*

Lemma 9 (Subject reduction). *If $\Gamma \vdash M : A$ and $M \to_\beta M'$ then $\Gamma \vdash M = M' : A$.*

Lemma 10. *If $\Gamma \vdash M : A$ and $M \overset{\beta}{=} M'$ then $\Gamma \vdash M = M' : A$.*

Lemma 11 (Soundness of a β-convertible substitution). *If $\Gamma \vdash N : B$ is derivable and $N \overset{\beta}{=} N'$ then*

1. *if $\Gamma, x : B, \Delta \vdash M = M' : A$ is derivable then $\Gamma, \Delta[N/x] \vdash M[N/x] = M'[N'/x] : A[N/x]$ is derivable.*
2. *if $\Gamma, x : B, \Delta \vdash A = A'$ is derivable then $\Gamma, \Delta[N/x] \vdash A[N/x] = A'[N'/x]$ is derivable;*

Theorem 1 (Church-Rosser). *If $M \overset{\beta}{=} M'$ then there exists a term N such that $M \xrightarrow{\beta} N$ and $M' \xrightarrow{\beta} N$.*

Definition 14 (η-reduction). *The* η-reduction*, written $\to_\eta$, is defined inductively on terms by*

- *(η-Red): $\lambda x.Mx \to_\eta M$;*
- *(η-Red-Abs-C): if $M \to_\eta M'$ then $\lambda x.M \to_\eta \lambda x.M'$;*
- *(η-Red-App-C-L): if $M \to_\eta M'$ then $MN \to_\eta M'N$;*
- *(η-Red-App-C-R): if $N \to_\eta N'$ then $MN \to_\eta MN'$.*

The η-equivalence is the reflexive-, symmetric-, transitive-closure of $\rightarrow_\eta$ and is written $\overset{\eta}{=}$.

The $\beta\eta$-equivalence is the composition of the equivalences $\overset{\beta}{=}$ and $\overset{\eta}{=}$.

The following lemma is will not be used in following proofs but is helpful to understand that the equality in the theory is basically the same notion that the untyped $\beta\eta$-equality. However we have chosen to use a λ-calculus with a typed equality because this way of presenting the theory seemed to be more natural and extensible. Moreover, typed equality is quite convenient and natural to use. For example[11] we have

$$2 = 4 : (\mathbb{Z} \bmod 4)$$

but

$$2 \neq 4 : \mathbb{Z}$$

Lemma 12. *If $\Gamma \vdash M : A$, then $\Gamma \vdash M = M' : A$ iff $M \overset{\beta\eta}{=} M'$.*

3 Decidability of equality in Cwf

3.1 Decidability of the equality in LF

In this section, we are going to prove that the equality is decidable in **LF**. More precisely, we are going to show that there exists an algorithm which, given two terms M and N and the proofs of their common type in an environment Γ (i.e. the proofs of $\Gamma \vdash M : A$ and $\Gamma \vdash N : A$), will answer in a finite time whether the judgment $\Gamma \vdash M = N : A$ is derivable or not.

Deciding the equality of two terms is considered to be an important property of theories because it is a fundamental result to have decidability results – and those results can be effective (i.e. have algorithms) since we explicitly give an algorithm to decide the equality.

Moreover, this result is not self-evident. As shown in lemma 12, the equality is basically the $\beta\eta$-equivalence on typable terms, and the fact that it is only on typable terms is crucial. If we consider the $\beta\eta$-equivalence on untyped λ-terms then this equality is not decidable anymore. Here is a sketch of the proof. λ-calculus has been shown to be Turing-complete and two λ-terms are extensionally equal iff they are $\beta\eta$-convertible; thus, deciding $\beta\eta$-convertibility would be the same as deciding extensional equality which is undecidable by reduction to the halt problem.

The proof we are going to make is a generalization of the proof given in [Fau02] for the non-dependent case (the simply-typed λ-calculus). We are also going to use some results of [CPT03]. Our result is also a generalization of their because they show the decidability of the equality only for a particular model of the dependently-typed λ-calculus: the per-model (see definition 20).

The structure of the proof is the following. First we are going to define a particular η-expansion based on a type A which is called the *incarnation* and is written η_A. It has been shown in [CPT03] that if $\vdash M : A$ is derivable in **LF** then $\eta_A(M)$ is normalizable and therefore, if $\vdash N : A$ is also derivable in **LF** the relation $\eta_A(M) \overset{\beta}{=} \eta_A(N)$ is decidable (by theorem 1 it is sufficient to check if they reduce to the same normal form). Then, we are going to prove that $\vdash M = N : A$ holds in **LF** iff $\eta_A(M) \overset{\beta}{=} \eta_A(N)$, which implies the decidability of the equality in **LF**. Only a little more work is needed to show that we can also decide judgments with a context (of the form $\Gamma \vdash M = N : A$).

The proofs of most of the properties given here can be found in annex, in section B.2. They are mostly inductions on the structure of the derivation of the hypothesis.

Let's first define *environments*. These are functions which can be seen as a functional and untyped equivalent of context morphisms in **LF** in the sense that they also associate terms to variables. Actually, for each context Γ, we will be able to define a special environment ρ_Γ which will turn out to be helpful to get rid of contexts when deciding equality by replacing each variable by its incarnation based on its type in the context.

[11] Of course this cannot be expressed in our type system (we do not have integers for example) but could be in an extension of it.

Definition 15 (Environment). *An* environment ρ *is a function which to each* **LF***-variable associates an* **LF***-term. The identity environment* $\mathrm{id}_{\mathrm{env}}$ *is such that for all variable* x, $\mathrm{id}_{\mathrm{env}}(x) = x$. *Given an environment* ρ *and a (variable, term)-couple, we can define the* update *of* ρ *by*

$$\langle \rho, x \mapsto M \rangle (y) = \begin{cases} M & \text{if } y = x \\ \rho(y) & \text{else} \end{cases}$$

We will write $M[\rho]$ *(resp.* $A[\rho]$*) the simultaneous (for all variables* x*) substitution of* $\rho(x)$ *for all free occurrence of* x *in* M *(resp.* A*). The composition of environments is defined by* $\rho_1 \circ \rho_2(x) = \rho_1(x)[\rho_2]$. *Therefore we have* $M[\rho_1 \circ \rho_2] = (M[\rho_1])[\rho_2]$.

Lemma 13. *If* $M \overset{\beta}{=} N$ *then* $M[\rho] \overset{\beta}{=} N[\rho]$.

We will now introduce the concept of *incarnation* which was first defined by J.-Y. Girard. The version we use here is a more syntactic definition proposed by T. Coquand in [CPT03]. It is a syntactic transformation of a term w.r.t. a given type, such that the new term is built according to the considered type i.e. we forget about the information which is not related to the type and focus on the part of the term that gives information related to the type. It might not be very clear here since we work with a really minimal λ-calculus (in particular we do not have a unit element or paring) but it is really the idea behind this definition. This incarnation will pre-η-expand terms according to their type in such manner that to check if two terms of same type are equal we will only need to check if their incarnations are β-convertible (instead of $\beta\eta$-convertible, since by lemma 12 the equality of the theory is the same as the $\beta\eta$-convertibility).

Definition 16 (Incarnation). *The* incarnation $\eta_A(M)$ *of a term* M *w.r.t. the type* A *is defined inductively on* A *by:*

- $\eta_\star(M) \equiv M$;
- $\eta_{\mathsf{El}\ N}(M) \equiv M$;
- $\eta_{\Pi x:A.B}(M) \equiv \lambda z.\eta_{B[\eta_A(z)/x]}\left(\mathrm{App}_{\Pi x:A.B}\left(M,\ \eta_A\left(z\right)\right)\right)$ *with* z *not free in* B.

The incarnation $\eta_\Gamma(\rho)$ *of an environment* ρ *w.r.t the context* Γ *is defined inductively on* Γ *by:*

- $\eta_\diamond(\rho) = \rho$;
- $\eta_{\Gamma,x:A}(\rho) = \left\langle \eta_\Gamma(\rho), x \mapsto \eta_{A[\eta_\Gamma(\rho)]}\left(\rho x\right) \right\rangle$.

We will write ρ_Γ *for* $\eta_\Gamma(\mathrm{id}_{\mathrm{env}})$. *The second rule gives immediately the recursive definition*

$$\rho_{\Gamma,x:A} = \left\langle \rho_\Gamma, x \mapsto \eta_{A[\rho_\Gamma]}\left(x\right) \right\rangle$$

Remark 9. Of course we have $\eta_{B[\eta_A(z)/x]}(M) \equiv \eta_B(M)$ since substitution on types only replaces variables in types of the form $\mathsf{El}\ N$ and the incarnation $\eta_{\mathsf{El}\ N}$ does not depend on N. Thus, incarnation of Π-types could have equivalently and more simply been defined by $\eta_{\Pi x:A.B}(M) \equiv \lambda z.\eta_B\left(M\ \eta_A\left(z\right)\right)$ (without the substitution on B). However this definition is more natural in the sense that it should be more scalable, i.e. it should suit better if we wanted to extend this notion to a more complex type theory.

The incarnation on contexts was defined in order to "get rid" of the context. In fact the equivalence we are going to prove is

$$\Gamma \vdash M = N : A \qquad \text{iff} \qquad \eta_A(M)[\rho_\Gamma] \overset{\beta}{=} \eta_A(N)[\rho_\Gamma]$$

The incarnation ρ_Γ of the context Γ is here to close the terms by substituting to the free variables in M and N their incarnation w.r.t their type in the context of their value in the context since we want the resulting terms to be β-convertible and not only $\beta\eta$-convertible.

We will now prove some technical lemmata which are of small interest in themselves but are going to be helpful in the following proofs. In particular, since most proofs are inductions (on the derivation of a judgment in hypothesis), we will need recursive definitions for some of the previously defined notions (the application of an environment and the incarnation of a context).

Lemma 14. *The following rules hold, giving us a recursive definition of the application of an environment to a term.*

1. *If* $x \notin \mathrm{FV}(\mathcal{J})$ *then* $\mathcal{J}[\langle \rho, x \mapsto M\rangle] = \mathcal{J}[\rho]$;
2. $(\Pi x : A.B)[\rho] = \Pi x : (A[\rho]).(B[\langle \rho, x \mapsto x\rangle])$;
3. $(\lambda x.M)[\rho] = \lambda x.(M[\langle \rho, x \mapsto x\rangle])$;
4. $(MN)[\rho] = (M[\rho])(N[\rho])$;
5. $\eta_A(M)[\rho] \stackrel{\beta}{=} \eta_{A[\rho]}(M[\rho])$.

Lemma 15. $\mathrm{FV}(\eta_A(M)) = \mathrm{FV}(M)$.

Proof. By induction on A. □

Lemma 16 (Recursive definition of $[\rho_\Gamma]$). *If* $\Gamma, x : A$ *is a context then*

$$M[\rho_{\Gamma,x:A}] \equiv M\left[\left\langle \rho_\Gamma, x \mapsto \eta_{A[\rho_\Gamma]}(x)\right\rangle\right] \equiv M[\eta_A(x)][\rho_\Gamma]$$

Proof. By induction on the length of Γ. □

Remember that we want to relate equality in the **LF**-theory and β-conversion of incarnation of terms. Towards this goal we need some intermediate lemmata which relate both equalities.

Lemma 17. *If* $M \stackrel{\beta}{=} N$ *then* $\eta_A(M) \stackrel{\beta}{=} \eta_A(N)$.

Proof. By induction on A. □

Lemma 18. *If* $\Gamma \vdash A = B$ *then* $\eta_A(M) \stackrel{\beta}{=} \eta_B(M)$.

Proof. By induction on the derivation of $\Gamma \vdash A = B$. □

We now want to prove that if $\Gamma \vdash M = N : A$ then $\eta_A(M)[\rho_\Gamma] \stackrel{\beta}{=} \eta_A(N)[\rho_\Gamma]$ which is one of the two implications of the goal equivalence. The first idea is of course to prove that by induction on the derivation of $\Gamma \vdash M = N : A$. However this cannot be done since we would not be able to use the induction hypothesis when dealing with the rule (App-Eq). This is due to the fact that the definition of the incarnation of an application is not recursive: we do not have $\eta_{B[N/x]}(MN) \equiv \eta_{\Pi x:A.B}(M)\eta_B(N)$ but rather $\eta_{B[N/x]}(MN) \equiv (\eta_{\Pi x:A.B}(M))\,N$. Actually the proof turned out to be much more complicated than we thought it would be.

We have to interpret **LF** in a particular model, the model of *partial equivalence relations*; the equality in this model has been proven in [CPT03] to imply the result we want (the validity of the interpretation of the judgment $\Gamma \vdash M = N : A$ in this model implies $\eta_A(M)[\rho_\Gamma] \stackrel{\beta}{=} \eta_A(N)[\rho_\Gamma]$). It would be interesting – but would require much more work and time than we actually had – to understand what really is the cause of the impossibility to make a purely syntactic proof, which would be more satisfactory since the result we want is purely syntactic.

Write $\mathcal{O}$ the set of objects i.e. syntactic terms of the form

$$M, N \quad ::= \quad x \quad | \quad \lambda x.M \quad | \quad \mathrm{App}_{\Pi x:A.B}(M, N)$$

Partial equivalence relations are going to be defined in order to have a model of type theory. They are based on the intuitive idea that we can prove that $M = M : A$ iff the type A is inhabited (else this equation is quite meaningless).

Definition 17 (per). *A* partial equivalence relation *(*per*) on a set $\mathfrak{D}$ is a binary relation $\mathcal{A}$ on $\mathfrak{D}$ which is symmetric and transitive.*

We may write $u_1 = u_2 : \mathcal{A}$ for $\mathcal{A}(u_1, u_2)$ and $u : \mathcal{A}$ for $u = u : \mathcal{A}$. Write $[\![A]\!]$ for the set of all $u \in \mathfrak{D}$ such that $u : \mathcal{A}$ and $\mathrm{per}(\mathfrak{D})$ for the set of all pers on $\mathfrak{D}$. If $\mathcal{A} \in \mathrm{per}(\mathfrak{D})$ then $\mathrm{Fam}(\mathcal{A})$ is the set of all functions $\mathcal{F} : [\![\mathcal{A}]\!] \to \mathrm{per}(\mathfrak{D})$ such that $\mathcal{F}(u_1) = \mathcal{F}(u_2)$ (where $=$ is the extensional equality[12]) whenever $u_1 = u_2 : \mathcal{A}$.

If $\mathcal{A} \in \mathrm{per}(\mathcal{O})$ and $\mathcal{F} \in \mathrm{Fam}(\mathcal{A})$ we can form $\Pi(\mathcal{A}, \mathcal{F}) \in \mathrm{per}(\mathcal{O})$ defined by $w_1 = w_2 : \Pi(\mathcal{A}, \mathcal{F})$ iff $u_1 = u_2 : \mathcal{A}$ implies $w_1 u_1 = w_2 u_2 : \mathcal{F}(u_1)$.

Definition 18 (Neutral terms). *A term is* neutral *iff it is (weakly) normalizable and of the form*

$$\nu \quad ::= \quad x \quad | \quad \nu M$$

Definition 19 (Saturation). *A relation $\mathcal{A} \in \mathrm{per}(\mathcal{O})$ is* saturated *iff*

- *$\nu : \mathcal{A}$ for every neutral ν;*
- *if $u : \mathcal{A}$ then u is normalizable;*
- *if $u_1 = u_2 : \mathcal{A}$ then $u_1 \overset{\beta}{=} u_2$.*

Definition 20 (Per-interpretation). *A* per-interpretation *(in a* per-model*) of types is constituted of*

- *an interpretation $\overline{\star} \in \mathrm{per}(\mathcal{O})$ of $\star$;*
- *a family $\mathcal{E} \in \mathrm{Fam}(\overline{\star})$ such that for all $M : \overline{\star}$, the interpretation $\overline{\mathsf{El}\ M} \equiv \mathcal{E}(M)$ is saturated.*

The interpretation of $\Pi x : A.B$ is $\overline{\Pi x : A.B} \equiv \Pi(\overline{A}, M \mapsto \overline{B\,[M/x]})$.

The intentional equality *is defined by the rules of figure 8. It is intentional in the sense that*

$$\frac{}{\star \cong \star}(\textsc{Star-Eq-C}) \quad \frac{M = N : \overline{\star}}{\mathsf{El}\ M \cong \mathsf{El}\ N}(\textsc{El-Eq-C})$$

$$\frac{M \mapsto \overline{B_1(M)} \in \mathrm{Fam}(\overline{A_1}) \qquad \overset{A_1 \cong A_2}{M \mapsto \overline{B_2(M)} \in \mathrm{Fam}(\overline{A_2})} \qquad M_1 = M_2 : \overline{A_1} \Rightarrow B_1(M_1) \cong B_2(M_2)}{\Pi x : A_1.B_1 \cong \Pi x : A_2.B_2}(\Pi\text{-Eq-C})$$

Figure 8. Rules of intentional equality

$\overline{A} = \overline{B}$ (extensionally) does not imply $A \cong B$.

We will sometime write $M : \overline{A}$ for $M = M : \overline{A}$ and writing $\overline{A}$ will always imply $A \cong A$.

Belonging to the interpretation of a Π-type can be defined inductively:

Lemma 19. *If $N = N' : \overline{A} \Rightarrow M\,[N/x] = M'\,[N'/x] : \overline{B\,[N/x]}$ then $\lambda x.M = \lambda x.M' : \overline{\Pi x : A.B}$.*

Proof. By definition of $\overline{\Pi x : A.B}$. □

In this model we are able to define a typed equality between environments. This equality is the counterpart in per-models of typed equality between morphisms of **LF** or of $\mathbf{Cwf_{LF}}$.

[12] Two functions f and g with a common domain $\mathscr{D}$ are said to be extensionally equal iff $\forall x \in \mathscr{D}, f(x) = g(x)$. The same definition holds for typed λ-terms: two λ-terms M and M' of type $\Pi x : A.B$ in a context Γ are extensionally equal iff the rule

$$\frac{\Gamma \vdash N : A}{\Gamma \vdash MN = M'N : B\,[N/x]}$$

holds.

Definition 21 (Judgements in per-models). *The judgments $\rho_1 = \rho_2 : \Gamma$ (which means that the two environments ρ_1 and ρ_2 are equal in the context Γ), $\Gamma : \overline{\text{Ctxt}}$ (Γ is a valid context), $A_1 = A_2[\![\Gamma]\!]$ (the two types A_1 and A_2 are equal in the context Γ) and $M_1 = M_2 : A[\![\Gamma]\!]$ (the two terms M_1 and M_2 of type A are equal in the context Γ) are defined by induction by the rules of the figure 9.*

We might write $A[\![\Gamma]\!]$ for $A = A[\![\Gamma]\!]$ and $M : A[\![\Gamma]\!]$ for $M = M : A[\![\Gamma]\!]$.

Rules for environments

$$\frac{}{\rho_1 = \rho_2 : \diamond}(\text{Env-C-Emp}) \quad \frac{\rho_1 = \rho_2 : \Gamma \qquad A[\rho_1] \cong A[\rho_2] \qquad \rho_1 x = \rho_2 x : \overline{A[\rho_1]}}{\rho_1 = \rho_2 : \Gamma, x : A}(\text{Env-C-Ext})$$

Rules for contexts

$$\frac{}{\diamond : \overline{\text{Ctxt}}}(\text{C-Emp}) \quad \frac{x \notin \text{DV}(\Gamma) \qquad A[\![\Gamma]\!]}{\Gamma, x : A : \overline{\text{Ctxt}}}(\text{C-Ext})$$

Rules for equalities

$$\frac{\Gamma : \overline{\text{Ctxt}} \qquad \forall \rho_1, \rho_2,\ \rho_1 = \rho_2 : \Gamma \Rightarrow A_1[\rho_1] \cong A_2[\rho_2]}{A_1 = A_2[\![\Gamma]\!]}(\text{Ty-Eq})$$

$$\frac{A[\![\Gamma]\!] \qquad \forall \rho_1, \rho_2,\ \rho_1 = \rho_2 : \Gamma \Rightarrow M_1[\rho_1] = M_2[\rho_2] : \overline{A[\rho_1]}}{M_1 = M_2 : A[\![\Gamma]\!]}(\text{Tm-Eq})$$

Figure 9. Rules for per-models

With those definitions, some derivation rules which are really similar to the derivations in **LF** can be proven to be valid. These will help to show that this model is compatible with the judgments in **LF**.

Lemma 20. *The rules of the figure 10 have been proven to hold in per-models in [CPT03].*

Lemma 21. *The relations $\cong$ and $=$ (on environments in a context, on types in a context and on typed terms in a context) are equivalence relation.*

Proof. By induction on the structure of the derivations. □

Lemma 22. *If $\rho = \rho' : \Gamma$ and $y \notin \Gamma$ then for any M, $\rho = \langle \rho', y \mapsto M \rangle : \Gamma$.*

Proof. By induction on the proof of $\rho = \rho' : \Gamma$. □

We can now prove that the interpretation is compatible with the judgments in **LF**.

Lemma 23 (Interpretation of LF in a per-model). *The following rules hold*

- *if $\Gamma \vdash$ then $\Gamma : \overline{\text{Ctxt}}$;*
- *if $\Gamma \vdash A$ then $A[\![\Gamma]\!]$;*
- *if $\Gamma \vdash A_1 = A_2$ then $A_1 = A_2[\![\Gamma]\!]$;*
- *if $\Gamma \vdash M : A$ then $M : A[\![\Gamma]\!]$;*
- *if $\Gamma \vdash M_1 = M_2 : A$ then $M_1 = M_2 : A[\![\Gamma]\!]$;*

Proof. By induction on the structure of the derivation of the hypothesis. □

Lemma 24. *If $\Gamma \vdash M = N : A$ then $\eta_A(M)[\rho_\Gamma] \stackrel{\beta}{=} \eta_A(N)[\rho_\Gamma]$.*

Proof. Suppose that $\Gamma \vdash M = N : A$. Then by lemma 23 we have $M = N : A[\![\Gamma]\!]$ and this has been proven to imply $\eta_A(M)[\rho_\Gamma] \stackrel{\beta}{=} \eta_A(N)[\rho_\Gamma]$ in [CPT03]. □

Type formation and equalities

$$\frac{\Gamma : \overline{\mathrm{Ctxt}}}{\star[\![\Gamma]\!]}(\textsc{Star}) \qquad \frac{M = N : \star[\![\Gamma]\!]}{\mathsf{El}\ M = \mathsf{El}\ N[\![\Gamma]\!]}(\textsc{El-Eq-C}) \qquad \frac{A_1 = A_2[\![\Gamma]\!] \qquad B_1 = B_2[\![\Gamma, x : A_1]\!]}{\Pi x : A_1.B_1 = \Pi x : A_2.B_2[\![\Gamma]\!]}(\Pi\text{-Eq-C})$$

Terms

$$\frac{\Gamma, x : A : \overline{\mathrm{Ctxt}}}{x : A[\![\Gamma, x : A]\!]}(\textsc{Var}) \qquad \frac{M : B[\![\Gamma, x : A]\!]}{\lambda x.M : \Pi x : A.B[\![\Gamma]\!]}(\textsc{Abs}) \qquad \frac{M : \Pi x : A.B[\![\Gamma]\!] \qquad N : A[\![\Gamma]\!]}{MN : B\,[N/x]\,[\![\Gamma]\!]}(\textsc{App})$$

Type conversion

$$\frac{M = N : A[\![\Gamma]\!] \qquad A = B[\![\Gamma]\!]}{M = N : B[\![\Gamma]\!]}(\textsc{Tm-Eq-Conv})$$

Weakening

$$\frac{B_1 = B_2[\![\Gamma]\!] \qquad \Gamma, x : A : \overline{\mathrm{Ctxt}}}{B_1 = B_2[\![\Gamma, x : A]\!]} \qquad \frac{M = N : B[\![\Gamma]\!] \qquad \Gamma, x : A : \overline{\mathrm{Ctxt}}}{M = N : B[\![\Gamma, x : A]\!]}$$

Figure 10. Derivable rules in per-models

Lemma 25. *If $\Gamma \vdash M : A$ then $\Gamma \vdash M = \eta_A\,(M) : A$.*

Proof. By induction on A. □

Lemma 26. *If $\Gamma \vdash M : A$ then $\Gamma \vdash M = M\,[\rho_\Gamma] : A$.*

Proof. By induction on the derivation of $\Gamma \vdash M : A$. □

Lemma 27. *If $\Gamma \vdash M : A$, $\Gamma \vdash N : A$ and $\eta_A(M)\,[\rho_\Gamma] \stackrel{\beta}{=} \eta_A(N)\,[\rho_\Gamma]$ then $\Gamma \vdash M = N : A$.*

Proof. Since by hypothesis we have $\Gamma \vdash M : A$, using lemmata 25 and 2 we also have $\Gamma \vdash \eta_A(M) : A$ and by lemmata 26 and 2 we have $\Gamma \vdash \eta_A(M)\,[\rho_\Gamma] : A$. Similarly, we can show that $\Gamma \vdash \eta_A(N)\,[\rho_\Gamma] : A$. By hypothesis, we also have $\eta_A(M)\,[\rho_\Gamma] \stackrel{\beta}{=} \eta_A(N)\,[\rho_\Gamma]$, which implies by lemma 10 that $\Gamma \vdash \eta_A(M)\,[\rho_\Gamma] = \eta_A(N)\,[\rho_\Gamma] : A$. Using lemmata 25, 26 and (Tm-Eq-Trans), we have $\Gamma \vdash M = \eta_A\,(M)\,[\rho_\Gamma] : A$ and $\Gamma \vdash N = \eta_A\,(N)\,[\rho_\Gamma] : A$. Finally, using (Tm-Eq-Trans) and (Tm-Eq-Refl), we can conclude that $\Gamma \vdash M = N : A$. □

Theorem 2. *If $\Gamma \vdash M : A$, $\Gamma \vdash N : A$ then: $\Gamma \vdash M = N : A$ iff $\eta_A(M)\,[\rho_\Gamma] \stackrel{\beta}{=} \eta_A(N)\,[\rho_\Gamma]$.*

Proof. This results directly from lemmata 24 and 27. □

Theorem 3. *The equality is decidable in* **LF**.

Proof. It has been shown in [CPT03] that the relation $\eta_A(M)\,[\rho_\Gamma] \stackrel{\beta}{=} \eta_A(N)\,[\rho_\Gamma]$ is decidable. It comes from the fact that typable terms are normalizable. Thus to decide equality of two terms M and N, it is enough to compute their normal forms by making successive β-reductions and check if they are the same. We can show that two β-convertible normalizable terms have the same normal form using theorem 1. □

3.2 Definition of LF as a pseudo-gat

Let's first introduce a slightly different definition of **LF**. This definition looks much more like a gat. We do this in order to be able not to handle explicitly the rules implied by gat (the type conversion rules for example).

Definition 22 (Equivalent definition of LF). LF *can be equivalently defined as a* pseudo-*gat i.e. the theory defined by the rules (and the notations) of the figure 11 and where additionally the "meta"-rules implied by* **Gat** *(fig. 1) are required to hold – in fact, those rules are the rules of fig. 6 under sections "type equalities" and "term equalities" excepting (*Π-C*) and (*Π-η*) and the rule of lemma 4. We also have slightly changed the notations to make it look much more like the definition of* **Gat** *(fig. 1) in order for the equivalence between the two systems to be visually clearer.*

This theory will be written $\mathbf{LF_{Gat}}$ *in the proof of the equivalence of the two definitions but* **LF** *after that (this is motivated by the equivalence).*

We do not mention it explicitly but x *is supposed to be* fresh *(i.e. not in the variables defined in the preceding context* Γ*).*

Remark 10. The theory defined in fig. 11 is not a proper gat since we use the substitution in those rules and substitution is a higher-order operation (that is why we have called it a *pseudo*-gat). To be defined in gat, **LF** should be defined with explicit substitutions and De Bruijn indexes. However we did not want to use explicit substitutions even though a λ-calculus with explicit substitutions would be more like $\mathbf{Cwf_{LF}}$ (which would have lead to a simpler proof of equivalence) because the λ-calculus with explicit substitutions has been less studied than the usual λ-calculus.

Definition 23 (Substution on contexts). *Substitution can naturally be extended on contexts by*

$$\begin{aligned}\diamond[M/x] &\equiv \diamond\\ (\Gamma, x:A)[M/x] &\equiv \Gamma\\ (\Gamma, y:A)[M/x] &\equiv \Gamma[M/x], y:A[M/x]\end{aligned}$$

Lemma 28 (Soundess of application of context morphisms). *The following rules are derivable*

$$\frac{\Gamma : \text{Ctxt} \qquad A : \text{Type}(\Gamma) \qquad x : \Gamma \vdash A \qquad M : \Gamma \vdash A \qquad \Gamma, x : A, \Delta : \text{Ctxt} \qquad B : \text{Type}(\Gamma, x : A, \Delta)}{B[M/x] : \text{Type}(\Gamma, \Delta[M/x])}$$

$$\frac{\Gamma : \text{Ctxt} \qquad A : \text{Type}(\Gamma) \qquad x : \Gamma \vdash A \qquad M : \Gamma \vdash A \qquad \Gamma, x : A, \Delta : \text{Ctxt} \qquad B : \text{Type}(\Gamma, x : A, \Delta) \qquad N : \Gamma, x : A, \Delta \vdash B}{N[M/x] : \Gamma, \Delta[M/x] \vdash B[M/x]}$$

Definition 24 (Context morphism). *A context morphism* γ *is a list of variable-term couples. Morphisms of* **LF** *and associated operators are defined by the rules of figure 12.*

3.3 Equivalence between LF and $\mathbf{Cwf_{LF}}$

In this section, we show the equivalence between **LF** and $\mathbf{Cwf_{LF}}$.

The elaboration of the proof of the equivalence of the two theories was one of the hardest points we have faced. The first decision we had to make was to decide whether we would chose a semantic or a syntactic approach.

Historically, the first proof of the equivalence between simply-typed λ-calculus and cartesian-closed categories (ccc) – proof of which this proof can be seen as an extension – was a semantic one and is given in [LS86]. It proves the equivalence between the two theories by proving the equivalence of the respective categories of models. More precisely, for each model of one theory an equivalent model is given and the equivalence is proven by giving two reciprocal interpretations that are proven

Sort symbols:

$$\frac{}{\text{Ctxt} : \text{Sort}}$$

$$\frac{\Gamma : \text{Ctxt}}{\text{Type}\,(\Gamma) : \text{Sort}}$$

$$\frac{\Gamma : \text{Ctxt} \qquad A : \text{Type}\,(\Gamma)}{\Gamma \vdash A : \text{Sort}}$$

Contexts

Operator symbols:

$$\frac{}{\diamond : \text{Ctxt}}(\text{C-Emp})$$

$$\frac{\Gamma : \text{Ctxt} \qquad A : \text{Type}\,(\Gamma)}{\Gamma, x : A : \text{Ctxt}}(\text{C-Ext})$$

Types

Operator symbols:

$$\frac{\Gamma : \text{Ctxt}}{\text{Star} : \text{Type}\,(\Gamma)}(\textsc{Star})$$

$$\frac{\Gamma : \text{Ctxt} \qquad M : \Gamma \vdash \text{Star}}{\text{Elem}\,(M) : \text{Type}\,(\Gamma)}(\textsc{Elem})$$

$$\frac{\Gamma : \text{Ctxt} \qquad A : \text{Type}\,(\Gamma) \qquad B : \text{Type}\,(\Gamma, x : A)}{\Pi x : A.B : \text{Type}\,(\Gamma)}(\textsc{Exp})$$

Terms

Operator symbols:

$$\frac{\Gamma : \text{Ctxt} \qquad A : \text{Type}\,(\Gamma)}{x : (\Gamma, x : A \vdash A)}(\textsc{Var})$$

$$\frac{\Gamma : \text{Ctxt} \qquad A\ B : \text{Type}\,(\Gamma) \qquad x : \Gamma \vdash A \qquad x \not\equiv y}{x : (\Gamma, y : B \vdash A)}(\textsc{Var-Ext})$$

$$\frac{\Gamma : \text{Ctxt} \qquad A : \text{Type}\,(\Gamma) \qquad B : \text{Type}\,(\Gamma, x : A) \qquad M : \Gamma, x : A \vdash B}{\lambda x.M : \Gamma \vdash \Pi x : A.B}(\textsc{Abs})$$

$$\frac{\Gamma : \text{Ctxt} \qquad A : \text{Type}\,(\Gamma) \qquad B : \text{Type}\,(\Gamma, x : A) \qquad M : \Gamma \vdash \Pi x : A.B \qquad N : \Gamma \vdash A}{\text{App}_{\Pi x:A.B}\,(M, N) : \Gamma \vdash B\,[N/x]}(\textsc{App})$$

Equations:

$$\frac{\Gamma : \text{Ctxt} \qquad A : \text{Type}\,(\Gamma) \qquad B : \text{Type}\,(\Gamma, x : A) \qquad M : \Gamma, x : A \vdash B \qquad \Gamma : N \vdash A}{\text{App}_{\Pi x:A.B}\,((\lambda x.M)\,, N) = M\,[N/x] : \Gamma \vdash B\,[N/x]}(\Pi\text{-C})$$

$$\frac{\Gamma : \text{Ctxt} \qquad A : \text{Type}\,(\Gamma) \qquad B : \text{Type}\,(\Gamma, x : A) \qquad M : \Gamma \vdash \Pi x : A.B}{\lambda x.\,(\text{App}_{\Pi x:A.B}\,(M, x)) = M : \Gamma \vdash \Pi x : A.B}(\Pi\text{-}\eta)$$

Figure 11. Definition of **LF** as a pseudo-gat

Sort symbols:

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt}}{\Delta \to \Gamma : \mathrm{Sort}}$$

Operator symbols:

$$\frac{\Gamma : \mathrm{Ctxt}}{\langle\rangle : \Gamma \to \diamond}$$

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt} \qquad \gamma : \Delta \to \Gamma \qquad A : \mathrm{Type}\,(\Gamma) \qquad x : \Gamma \vdash A \qquad M : \Delta \vdash A\,[\gamma]}{\langle \gamma, x \mapsto M \rangle : \Delta \to \Gamma, x : A}$$

$$\frac{\Gamma : \mathrm{Ctxt}}{\mathrm{id}_\Gamma : \Gamma \to \Gamma}$$

$$\frac{\Theta\ \Delta\ \Gamma : \mathrm{Ctxt} \qquad \gamma : \Delta \to \Gamma \qquad \delta : \Theta \to \Delta}{\gamma \circ \delta : \Theta \to \Delta}$$

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt} \qquad \gamma : \Delta \to \Gamma \qquad A : \mathrm{Type}\,(\Gamma)}{A\,[\Gamma] : \mathrm{Type}\,(\Delta)}$$

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt} \qquad \gamma : \Delta \to \Gamma \qquad A : \mathrm{Type}\,(\Gamma) \qquad M : \Gamma \vdash A}{M\,[\gamma] : \Delta \vdash A\,[\gamma]}$$

Equations:

$$\frac{}{\langle\rangle = \mathrm{id}_\diamond : \diamond \to \diamond}$$

$$\frac{\Gamma\ \Delta : \mathrm{Ctxt} \qquad \gamma : \Delta \to \Gamma \qquad A : \mathrm{Type}\,(\Gamma) \qquad x : \Gamma \vdash A \qquad M : \Delta \vdash A\,[\gamma] \qquad B : \mathrm{Type}\,(\Gamma, x : A)}{B\,[\langle \gamma, x \mapsto M \rangle] = (B\,[M/x])\,[\gamma] : \mathrm{Type}\,(\Delta)}$$

$$\frac{\Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\,(\Gamma)}{A\,[\mathrm{id}_\Gamma] = A : \mathrm{Type}\,(\Gamma)}$$

$$\frac{\Gamma : \mathrm{Ctxt} \qquad A : \mathrm{Type}\,(\Gamma) \qquad M : \Gamma \vdash A}{M\,[\mathrm{id}_\Gamma] = M : \Gamma \vdash A}$$

Figure 12. Rules for context morphisms in **LF**

to respect some soundness properties (i.e. basically they are compatible with the morphisms of the categories). This proof is rather complicated with many technical details: they have to introduce polynomial λ-calculi (which are λ-calculi with extra term-variables added), their calculus has to have a natural number object (i.e. "contain" $\mathbb{N}$), etc. Some more technical details seem to be very unnatural. The proof is a also quite difficult to follow because it involves many concepts of category theory (for example the equivalence between the categories of models is defined using the notion of natural transformation).

The proof we chose to do is a syntactic one which consists in giving two interpretations of one theory to the other one. However these interpretations are purely syntactical (no reference is ever made to a model). This is inspired by work already made to show the equivalence between cccs and simply-typed λ-calculus. A sketch of the proof is given in [Hue86]. Some encoding is needed because objects of cccs are types and not contexts. Therefore the translation can only be done on terms in a context with one element which is not restrictive; the pairing operations[13] and the curryfying operation[14] are required to be in both theories to "encode" the context into a type: the context $x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n$ is encoded to $x : (((A_1 \wedge A_2) \wedge \ldots) \wedge A_{n-1}) \wedge A_n$ and the variables x_i can be recovered by using projections and is replaced by $\pi' \circ \pi^{n-i}$ (in cwfs, this structure is in contexts and no encoding is needed which might seem more natural).

Both theories have roughly the same operators and the same structure. So it might seem quite easy to prove the equivalence. However, when looking at it attentively many technical details have to be dealt with (see [Cur86] and [San87]) especially when relating the substitution of λ-calculus (which is a semantic operation, of high-order) with the substitution of cwfs (which is a syntactic operation and is part of the calculus). Many presentations of the calculi are possible and are not all equivalent, in particular some operators can be typed or not (the abstraction and the application for example).

The second approach seems more natural and simple. That is why we chose it. Actually, you have to deal with less technical details and the proof is conceptually simpler. However the first one may seem more satisfactory in the sense that what we really want to talk about is the category of models and not so much the theory which has generated this category.

We have adopted the structure of the proof given in [San87] for the equivalence between simply-typed λ-calculus and cccs or of the one given in [Rit92] for the equivalence between the Calculus of Constructions (a calculus a bit different from our LF) and categories with attributes (a theory similar to cwfs). The sketch of the proof is rather simple. We are going to define two interpretations of one theory in one another. Those are entirely syntactical. $[\![\ ]\!]_{\mathbf{Cwf_{LF}}}$ interprets **LF**-terms, -types and -contexts in **Cwf** and reciprocally $[\![\ ]\!]_{\mathbf{LF}}$ will interpret **Cwf**-terms, -types and -contexts in **LF**. Those interpretation will be then proven to be sound which means that if a judgment $\Gamma \vdash \mathcal{J}$ holds in **LF** then its interpretation $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}} \vdash [\![\mathcal{J}]\!]_{\mathbf{Cwf_{LF}}}$ in $\mathbf{Cwf_{LF}}$ holds and reciprocally. Finally, we will show that the two interpretations are inverse of one another modulo equality. Those conditions are enough to show that an equality $\Gamma \vdash M = N : A$ holds in $\mathbf{Cwf_{LF}}$ iff its interpretation holds in **LF**. Therefore, since we have shown that the equality was decidable in **LF**, we will be able to conclude that the equality is decidable in $\mathbf{Cwf_{LF}}$.

The general idea may seem to be simple but we will see that we will have face some complicated details in particular when we will have to relate the substitution of the two theories. The main problem comes from the fact that substitution is in the theory in cwfs whereas it is a meta-operation in λ-calculus. The proof of the equivalence might have been easier if we had chosen a λ-calculus with explicit substitutions but we did not want to do that since λ-calculi with explicit substitution have been much less studied than without.

We have already explained that the operator App of $\mathbf{Cwf_{LF}}$ has to be typed in order to have the decidability of the initial model of $\mathbf{Cwf_{LF}}$; the App of $\mathbf{Cwf_{LF}}$ must therefore also be typed in

[13] For each two types A and B there is a sum type $A \wedge B$ and there is also a sum object for each pair of morphisms of same domain, along with the respective projections.

[14] The curryfying operator is basically the operation which to each morphism f associates the morphism "$\lambda x.\lambda y.f\ \langle x, y\rangle$" (in OCaml this operator would be defined by `let curry f x y = f (x, y)`).

order for the equivalence not to be too complicated (it might even have been unfeasible). However it would have been much simpler and more clean with untyped operators in both theories (in fact this is the proof we had begun to do before we became aware of the initial model problem). For example in [Rit92], they have untyped theories and they have completed their interpretation with "stubs" (i.e. meaningless values) in order to have a total interpretation. For example the interpretation of x_i in an empty context is quite meaningless (or on the other way, the interpretation of q in an empty context). They have however given an arbitrary value (a stub) to the interpretation of x_i; this way, their interpretation is total and they can avoid using the Kleene equality[15] which is rather difficult to manipulate and not very natural since we always have to wonder whereas the objects we manipulate are defined or not.

Remark 11. To avoid having to pass the type of the interpreted term as an argument of the interpretations, we are going to use slightly different notations than previously mentioned for the abstraction in **LF** and **Cwf**: those will be indexed by the type of the argument. For example we will write $\lambda x_A.M$ instead of $\lambda x.M$ in **LF** and $\lambda_A(M)$ instead of $\lambda(M)$ in **Cwf**. This is not strictly necessary but the syntax for the interpretations is already hard enough to read as you will see.

The proofs of most properties can be found in annex C.

Interpretation of LF into $\mathbf{Cwf_{LF}}$ We first provide an interpretation of terms, types and contexts of objects of **LF** into objects of **LF**. It is largely inspired of [Hof97] and [Rit92].

The theory **LF** can be interpreted into a $\mathbf{Cwf_{LF}}$ with the interpretation $[\![\ ]\!]_{\mathbf{Cwf}}$ defined by

– contexts of **LF** are interpreted into contexts (objects) of $\mathbf{Cwf_{LF}}$

$$\begin{aligned}[\![\diamond]\!]_{\mathbf{Cwf_{LF}}} &\equiv \diamond\\ [\![\Gamma, x:A]\!]_{\mathbf{Cwf_{LF}}} &\equiv [\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}, [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\end{aligned}$$

– types of **LF** in a context Γ are interpreted into types of $\mathbf{Cwf_{LF}}$ in the context $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$

$$\begin{aligned}[\![\mathrm{Star}]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} &\equiv \mathrm{Star}\\ [\![\mathrm{Elem}(M)]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} &\equiv \mathrm{Elem}\left([\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\right)\\ [\![\Pi x:A.B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} &\equiv \Pi\left([\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}, [\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}\right)\end{aligned}$$

– terms of **LF** of type A in a context $\Gamma \equiv x_1:A_1,\ldots,x_n:A_n$ are interpreted into terms of $\mathbf{Cwf_{LF}}$ of type $[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ in the context $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$ (we write Γ' the context $x_A:A_1,\ldots,x_{n-1}:A_{n-1}$)

$$\begin{aligned}[\![x_i]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} &\equiv \begin{cases} q_{[\![A_n]\!]^{\Gamma'}_{\mathbf{Cwf_{LF}}}} & \text{if } i=n\\ [\![x_i]\!]^{\Gamma'}_{\mathbf{Cwf_{LF}}}\left[p_{[\![A_n]\!]^{\Gamma'}_{\mathbf{Cwf_{LF}}}}\right] & \text{else}\end{cases}\\ [\![\lambda x_A.M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} &\equiv \lambda_A\left([\![M]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}\right)\\ [\![\mathrm{App}_{\Pi x:A.B}(M,N)]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} &\equiv \mathrm{App}_{\Pi\left([\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}, [\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}\right)}\left([\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}, [\![N]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\right)\end{aligned}$$

Proof (Well-foundedness). This definition is well founded since:

– the interpretation of contexts calls itself recursively on structurally smaller contexts;
– the interpretation of types calls itself recursively on structurally smaller types;

[15] The Kleene equality $a \cong b$ is defined by: a is defined iff b is defined and in this case a and b are equal.

– the interpretation of terms calls itself recursively on structurally smaller terms except for the rule $[\![x_i]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ which only recursively calls $[\![x_i]\!]^{\Gamma'}_{\mathbf{Cwf_{LF}}}$ with a context Γ' structurally smaller than Γ. □

Remark 12. The interpretation is partial: the interpretation of a term might not be defined. This is why we are going to need to use Kleene equality. It would have been better to have a total interpretation but this seems much more difficult to do.

To show that this translation has good soundness properties (cf. proposition 3), we must first relate the syntactic substitution in LFs to semantic substitution in cwfs. This is why the following definitions are required.

Definition 25 (Projection and unprojection morphisms). *Let Γ and Δ be two contexts, A a type and M a term.*

The projection morphism P_x is defined inductively by:

$$\begin{aligned} P_x(\Gamma, x : A) &\equiv p_{[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}} \\ P_x(\Gamma, x : A, \Delta, y : B) &\equiv \tilde{p}\left(P_x\left(\Gamma, x : A, \Delta\right), [\![B]\!]^{\Gamma,\Delta}_{\mathbf{Cwf_{LF}}}\right) \\ &= \left\langle P_x\left(\Gamma, x : A, \Delta\right) \circ p_{[\![B]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}}, q_{[\![B]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}} \right\rangle \end{aligned}$$

where $\tilde{p}$ is the weakening as defined in definition 6.

Similarly, the unprojection morphism U^M_x is defined inductively by:

$$\begin{aligned} U^M_x\left(\Gamma, x : A\right) &\equiv \left\langle \mathrm{id}_{\Gamma}, [\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} \right\rangle \\ U^M_x\left(\Gamma, x : A, \Delta, y : B\right) &\equiv \tilde{p}\left(U^M_x\left(\Gamma, x : A, \Delta\right), [\![B]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}\right) \\ &= \left\langle U^M_x\left(\Gamma, x : A, \Delta\right) \circ p_{[\![B]\!]^{\Gamma,\Delta[M/x]}_{\mathbf{Cwf_{LF}}}}, q_{[\![B]\!]^{\Gamma,\Delta[M/x]}_{\mathbf{Cwf_{LF}}}} \right\rangle \end{aligned}$$

The idea is that $P_x\left(\Gamma, x : A, \Delta\right)$ is a morphism

$$P_x\left(\Gamma, x : A, \Delta\right) : [\![\Gamma, x : A, \Delta]\!]_{\mathbf{Cwf_{LF}}} \to [\![\Gamma, \Delta]\!]_{\mathbf{Cwf_{LF}}}$$

in $\mathbf{Cwf_{LF}}$ projecting out the A-part. Similarly

$$U^M_x\left(\Gamma, x : A, \Delta\right) : [\![\Gamma, \Delta[M/x]]\!]_{\mathbf{Cwf_{LF}}} \to [\![\Gamma, x : A, \Delta]\!]_{\mathbf{Cwf_{LF}}}$$

is a morphism in $\mathbf{Cwf_{LF}}$.

For possibly undefined expression s and t, we shall write $s \cong t$ to mean that if either side is defined then so is the other one and both agree (Kleene equality).

Lemma 29 (Weakening). *Let Γ and Δ be pre-contexts, A and B be pre-types, M be a pre-term and x a fresh variable. Let $\mathcal{J}$ be either M or A. The expression $P_x\left(\Gamma, x : A, \Delta\right)$ is defined iff $[\![\Gamma, x : A, \Delta]\!]_{\mathbf{Cwf_{LF}}}$ and $[\![\Gamma, \Delta]\!]_{\mathbf{Cwf_{LF}}}$ are defined and in this case is a morphism from the former to the latter. If $[\![\mathcal{J}]\!]^{\Gamma,\Delta}_{\mathbf{Cwf_{LF}}}$ is defined then*

$$[\![\mathcal{J}]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}} \cong [\![\mathcal{J}]\!]^{\Gamma,\Delta}_{\mathbf{Cwf_{LF}}}\left[P_x\left(\Gamma, x : A, \Delta\right)\right]$$

Proof. The proof proceeds by induction on the lengths of the involved pre-terms, -types and -contexts. □

Lemma 30 (Substitution). *Let Γ and Δ be pre-contexts, A and B be pre-types, M and N be pre-terms and x be a fresh variable. Let $\mathcal{J}$ be either A or M and suppose that $[\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ is defined. The expression $U^M_x\left(\Gamma, x : A, \Delta\right)$ is defined iff $[\![\Gamma, \Delta\left[M/x\right]]\!]_{\mathbf{Cwf_{LF}}}$ and $[\![\Gamma, x : A, \Delta]\!]_{\mathbf{Cwf_{LF}}}$ are both defined and in this case is a morphism from the former to the latter. If $[\![\mathcal{J}]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}$ is defined then*

$$[\![\mathcal{J}\left[M/x\right]]\!]^{\Gamma,\Delta[M/x]}_{\mathbf{Cwf_{LF}}} \cong [\![\mathcal{J}]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}\left[U^M_x\left(\Gamma, x : A, \Delta\right)\right]$$

Proof. The proof proceeds by induction on the lengths of the involved pre-terms, -types and -contexts as in the proof of lemma 29. □

Proposition 3 (Soundness). *The interpretation function enjoys the following soundness properties*

1. *if* $\Gamma \vdash$ *then* $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}} : \mathrm{Ctxt}$ *is derivable in* $\mathbf{Cwf_{LF}}$*;*
2. *if* $\Gamma \vdash A$ *then* $[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} : \mathrm{Type}\left([\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}\right)$ *is derivable in* $\mathbf{Cwf_{LF}}$*;*
3. *if* $M : \Gamma \vdash A$ *then* $[\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} : [\![\Gamma]\!]_{\mathbf{Cwf_{LF}}} \vdash [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ *is derivable in* $\mathbf{Cwf_{LF}}$*;*
4. *if* $\Gamma \vdash A = B$ *then* $[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} = [\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} : \mathrm{Type}\,(\Gamma)$ *is derivable in* $\mathbf{Cwf_{LF}}$*;*
5. *if* $M = N : \Gamma \vdash A$ *then* $[\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} = [\![N]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} : [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ *is derivable in* $\mathbf{Cwf_{LF}}$*.*

Proof. The proof is done by induction on the derivations. □

Interpretation of $\mathbf{Cwf_{LF}}$ into LF $\mathbf{Cwf_{LF}}$ can be interpreted into **LF** with the interpretation $[\![\ ]\!]_{\mathbf{LF}}$ defined by:

– contexts of $\mathbf{Cwf_{LF}}$ are interpreted into contexts of **LF**

$$[\![\Gamma]\!]_{\mathbf{LF}} = \begin{cases} \diamond & \text{if } \Gamma = \diamond \\ [\![\Gamma']\!]_{\mathbf{LF}}, x : [\![A]\!]^{\Gamma'}_{\mathbf{LF}} & \text{else, where } x \text{ is a "fresh" variable (i.e. } x \notin \mathrm{DV}\,([\![\Gamma']\!]_{\mathbf{LF}})) \end{cases}$$

– types of $\mathbf{Cwf_{LF}}$ in a context Γ are interpreted into types of **LF** in the context $[\![\Gamma]\!]_{\mathbf{LF}}$

$$\begin{aligned} [\![\mathrm{Star}]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}} &\equiv \mathrm{Star} \\ [\![\mathrm{Elem}\,(M)]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}} &\equiv \mathrm{Elem}\left([\![M]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}}\right) \\ [\![\Pi(A,B)]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}} &\equiv \Pi x : [\![A]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}} . [\![B]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}}}_{\mathbf{LF}} && \text{with } x \notin \mathrm{DV}\,(()\,[\![\Gamma]\!]_{\mathbf{LF}}) \\ [\![\Sigma(A,B)]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}} &\equiv \Sigma x : [\![A]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}} . [\![B]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}}}_{\mathbf{LF}} && \text{with } x \notin \mathrm{DV}\,([\![\Gamma]\!]_{\mathbf{LF}}) \end{aligned}$$

– terms of $\mathbf{Cwf_{LF}}$ in a context Γ are interpreted into terms of **LF** in the context $[\![\Gamma]\!]_{\mathbf{LF}}$

$$\begin{aligned} [\![q]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}}}_{\mathbf{LF}} &\equiv x && \text{with } x \notin \mathrm{DV}\,([\![\Gamma]\!]_{\mathbf{LF}}) \\ [\![\lambda_A(M)]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}} &\equiv \lambda x_A . \left([\![M]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]^{\Gamma}_{\mathbf{LF}}}_{\mathbf{LF}}\right) && \text{with } x \notin \mathrm{DV}\,([\![\Gamma]\!]_{\mathbf{LF}}) \\ [\![\mathrm{App}\,(M,N)]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}} &\equiv \left([\![M]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}}\right)\left([\![N]\!]^{\Gamma}_{\mathbf{LF}}\right) \end{aligned}$$

– morphisms of $\mathbf{Cwf_{LF}}$ are interpreted into context morphisms of **LF**

$$\begin{aligned} [\![\langle\rangle]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}} &\equiv \langle\rangle \\ [\![\langle\gamma, M\rangle]\!]^{[\![\Gamma]\!], x:[\![A]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}}}_{\mathbf{LF}} &\equiv \left\langle [\![\gamma]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}}, x \mapsto [\![M]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}}}_{\mathbf{LF}} \right\rangle && \text{with } x \notin \mathrm{DV}\,([\![\Gamma]\!]_{\mathbf{LF}}) \end{aligned}$$

Definition 26 (Projection and unprojection morphisms). *Let* Γ *and* Δ *be two* **LF***-contexts in* $\mathbf{Cwf_{LF}}$*,* A *a* $\mathbf{Cwf_{LF}}$*-type in the context* Γ *and* M *an* **LF***-term.*

The projection morphism $\overline{P_x}$ *is defined inductively by:*

$$\begin{aligned} \overline{P_x}(\Gamma, x : A) &\equiv \mathrm{id}_\Gamma \\ \overline{P_x}(\Gamma, x : A, \Delta, y : B) &\equiv \langle P_x\,(\Gamma, x : A, \Delta), y \mapsto y\rangle \end{aligned}$$

The unprojection morphism $\overline{U^M_x}$ *is defined inductively by:*

$$\begin{aligned} \overline{U^M_x}\,(\Gamma, x : A) &\equiv \mathrm{id}_\Gamma, M^{\Gamma}_{\mathbf{LF}} \\ \overline{U^M_x}\,(\Gamma, x : A, \Delta, y : B) &\equiv \left\langle U^M_x\,(\Gamma, x : A, \Delta), y \mapsto y\right\rangle \end{aligned}$$

Just like for the previous way, the idea is that $\overline{P_x}\,(\Gamma, x : A, \Delta)$ is a context morphism

$$\overline{P_x}\left(\llbracket\Gamma\rrbracket_{\mathbf{LF}}, x : \llbracket A\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}}, \llbracket\Delta\rrbracket_{\mathbf{LF}}\right) : \llbracket\Gamma, A, \Delta\rrbracket_{\mathbf{LF}} \to \llbracket\Gamma, \Delta\rrbracket_{\mathbf{LF}}$$

in **LF** projecting out the A-part. Similarly, if M is a term of type $\Gamma \vdash A$ in $\mathbf{Cwf}_{\mathbf{LF}}$ then

$$\overline{U_x^M}\left(\llbracket\Gamma\rrbracket_{\mathbf{LF}}, x : \llbracket A\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}}, \Delta\right) : \llbracket\Gamma, \Delta\left[\langle \mathrm{id}_\Gamma, M\rangle\right]\rrbracket_{\mathbf{LF}} \to \llbracket\Gamma, A, \Delta\rrbracket_{\mathbf{LF}}$$

is a context morphism in **LF**.

Lemma 31 (Weakening). *Let Γ and Δ be pre-contexts, A and B be pre-types, M be a pre-term and x a fresh variable. Let $\mathcal{J}$ be either M or A. The expression $P_x\left(\llbracket\Gamma\rrbracket_{\mathbf{LF}}, x : \llbracket A\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}}, \Delta\right)$ is defined iff $\llbracket\Gamma, A, \Delta\rrbracket_{\mathbf{LF}}$ and $\llbracket\Gamma, \Delta\rrbracket_{\mathbf{LF}}$ are defined and in this case is a morphism from the former to the latter. If $\llbracket\mathcal{J}\rrbracket_{\mathbf{Cwf}_{\mathbf{LF}}}^{\Gamma,\Delta}$ is defined then*

$$\llbracket\mathcal{J}\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma, A, \Delta\rrbracket} \cong \llbracket\mathcal{J}\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma,\Delta\rrbracket_{\mathbf{LF}}}\left[P_x\left(\llbracket\Gamma\rrbracket_{\mathbf{LF}}, x : \llbracket A\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}}, \llbracket\Delta\rrbracket_{\mathbf{LF}}\right)\right]$$

Proof. The proof proceeds by induction on the lengths of the involved pre-terms, -types and -contexts as in the proof of lemma 29. □

Lemma 32 (Substitution). *Let Γ and Δ be pre-contexts, A and B be pre-types, M and N be pre-terms and x be a fresh variable. Let $\mathcal{J}$ be either A or M and suppose that $\llbracket M\rrbracket_{\mathbf{LF}}^{\Gamma}$ is defined. The expression $U_x^M\left(\llbracket\Gamma\rrbracket_{\mathbf{LF}}, x : \llbracket A\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}}, \llbracket\Delta\rrbracket_{\mathbf{LF}}\right)$ is defined iff $\left\llbracket\Gamma, \Delta\left[\langle \mathrm{id}_{\llbracket\Gamma\rrbracket_{\mathbf{LF}}}, M\rangle\right]\right\rrbracket_{\mathbf{LF}}$ and $\llbracket\Gamma, A, \Delta\rrbracket_{\mathbf{LF}}$ are both defined and in this case is a morphism from the former to the latter. If $\llbracket\mathcal{J}\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}, x:\llbracket A\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}}, \llbracket\Delta\rrbracket_{\mathbf{LF}}}$ is defined then*

$$\left\llbracket\mathcal{J}\left[\langle \mathrm{id}_{\llbracket\Gamma\rrbracket_{\mathbf{LF}}}, M\rangle\right]\right\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}, \left[\Delta\left[\langle \mathrm{id}_{\llbracket\Gamma\rrbracket_{\mathbf{LF}}}, M\rangle\right]\right]_{\mathbf{LF}}} \cong \llbracket\mathcal{J}\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}, x:\llbracket A\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}}, \llbracket\Delta\rrbracket_{\mathbf{LF}}}\left[U_x^M\left(\llbracket\Gamma\rrbracket_{\mathbf{LF}}, x : \llbracket A\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}}, \llbracket\Delta\rrbracket_{\mathbf{LF}}\right)\right]$$

Proof. The proof proceeds by induction on the lengths of the involved pre-terms, -types and -contexts as in the proof of lemma 31. □

Proposition 4 (Soundness). *The interpretation function enjoys the following soundness properties*

1. *if $\Gamma : \mathrm{Ctxt}$ then $\llbracket\Gamma\rrbracket_{\mathbf{LF}}$ is a context;*
2. *if $A : \mathrm{Type}\,(\Gamma)$ then $\llbracket A\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}}$ is a type in the context $\llbracket\Gamma\rrbracket$;*
3. *if $M : \Gamma \vdash A$ then $\llbracket M\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}}$ is a term of type $\llbracket\Gamma\rrbracket_{\mathbf{LF}} \vdash \llbracket M\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}} : \llbracket A\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}}$;*
4. *if $\Gamma \vdash A = B$ then $\llbracket A\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}} = \llbracket B\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}} : \mathrm{Type}\,(\llbracket\Gamma\rrbracket_{\mathbf{LF}})$;*
5. *if $M = N : \Gamma \vdash A$ then $\llbracket M\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}} = \llbracket N\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}} : \llbracket\Gamma\rrbracket_{\mathbf{LF}} \vdash \llbracket A\rrbracket_{\mathbf{LF}}^{\llbracket\Gamma\rrbracket_{\mathbf{LF}}}$;*
6. *if $\gamma : \Delta \to \Gamma$ then $\llbracket\gamma\rrbracket_{\mathbf{LF}}^{\llbracket\Delta\rrbracket_{\mathbf{LF}}} : \llbracket\Delta\rrbracket_{\mathbf{LF}} \to \llbracket\Gamma\rrbracket_{\mathbf{LF}}$;*
7. *if $\gamma = \delta : \Delta \to \Gamma$ then $\llbracket\gamma\rrbracket_{\mathbf{LF}}^{\llbracket\Delta\rrbracket_{\mathbf{LF}}} = \llbracket\delta\rrbracket_{\mathbf{LF}}^{\llbracket\Delta\rrbracket_{\mathbf{LF}}} : \llbracket\Delta\rrbracket_{\mathbf{LF}} \to \llbracket\Gamma\rrbracket_{\mathbf{LF}}$.*

The last two properties are not strictly required for the proof of the equivalence between **LF** and $\mathbf{Cwf}_{\mathbf{LF}}$ but are required to prove the other properties.

Proof. By induction on the derivation rules. □

Proposition 5. *The interpretations inverse of one another modulo equality*

1. *if $\Gamma \vdash_{\mathbf{LF}} A$ then $\left[\!\left[\llbracket M\rrbracket_{\mathbf{Cwf}_{\mathbf{LF}}}^{\Gamma}\right]\!\right]_{\mathbf{LF}}^{\Gamma} = M : \Gamma \vdash A$;*
2. *if $A : \mathrm{Type}_{\mathbf{LF}}\,(\Gamma)$ then $\left[\!\left[\llbracket A\rrbracket_{\mathbf{Cwf}_{\mathbf{LF}}}^{\Gamma}\right]\!\right]_{\mathbf{LF}}^{\Gamma} = A : \mathrm{Type}\,(\Gamma)$;*

3. if $\Gamma \vdash_{\mathbf{Cwf_{LF}}} A$ *then* $\left[\!\left[[\![M]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}} \right]\!\right]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{Cwf_{LF}}} = M : \Gamma \vdash A$;

4. if $A : \mathrm{Type}_{\mathbf{Cwf_{LF}}}(\Gamma)$ *then* $\left[\!\left[[\![A]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}} \right]\!\right]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{Cwf_{LF}}} = A : \mathrm{Type}(\Gamma)$.

Proof. By induction on terms, types and contexts. □

Theorem 4. *The theories* **LF** *and* $\mathbf{Cwf_{LF}}$ *are equivalent.*

Theorem 5 (Decidability of the equality in $\mathbf{Cwf_{LF}}$). *The equality is decidable in* $\mathbf{Cwf_{LF}}$.

Proof. This is a direct consequence of theorems 3 and 4. □

4 Possible formalization of the proof

We have tried to see how difficult it would be to formalize our proofs in COQ. The first question which arises is: how can we define category theory in COQ's type theory (which is based on the calculus of inductive constructions)? We have used the work already done by Saïbi to define category theory (see [Sai96]). One important point is that the equality of COQ is the Leibnitz equality (two terms are equal iff they are provably equal) whereas the equality used to define categories is the extensional equality (two functions f and g are extensionally equal iff $\forall x, f(x) = g(x)$). This is why the notion of *setoid* needs to be introduced to be able to make proofs about categories in an intentional type theory like COQ's without depending on a particular equality: instead of simply sets, the collections of objects and of morphisms of a category are required to be a setoid.

Definition 27 (Setoid). *A* setoid *is a quintuple* $\mathcal{A} = \langle A, \sim_A, \mathrm{refl}_A, \mathrm{sym}_A, \mathrm{trans}_A \rangle$. *The set* A *a is called the* carrier *of* $\mathcal{A}$ *and* $\sim_A$ *is an equivalence relation on* A *along with* refl_A, sym_A *and* trans_A *which are the proofs of respectively reflexivity, symmetry and transitivity of* $\sim_A$.

Since the proofs are not using COQ's equality but the equivalence relations of the setoids, the proofs get very long and fastidious: the rewrite tactics cannot be used and everything has to be done by hand using the reflexivity, the symmetry and the transitivity properties of the equivalence relation. For example, it took us more than 500 lines of COQ only to define the category of families (definition 4, not to be confused with categories *with* families). Defining generalized algebraic theories, categories with families as a gat, the logical framework and completely formalizing the proof of the equivalence between the two theories would require tremendous amounts of time and work (which would not be very complicated but very technical). `Rewrite`-like tactics could be developed using reflexion techniques to make the proofs easier and shorter but the elaboration of such tactics would be rather technical and fastidious.

Another way of solving this problem could be to formalize the proof in a prover like *NuPrl* which is based on an extensional type theory and would maybe lead to simpler proofs.

Another problem arises from a limitation of COQ: we cannot define mutually inductive types with different parameters (else COQ raises the error `User error: Parameters should be syntactically the same for each inductive type`). And this is needed to define gats: $\mathrm{Type}(\Gamma)$ depends on the context Γ, $\Gamma \vdash A$ depends on the context Γ and on the type A, etc. This might not be a theoretical limitation of COQ. The explanation given in the manual is:

> It is also possible to parameterize these inductive definitions. However, parameters correspond to a local context in which the whole set of inductive declarations is done. For this reason, the parameters must be strictly the same for each inductive types.

5 Conclusion

We have proven that the equality was decidable in LF (a dependently-typed λ-calculus) and have shown the equivalence between LF and categories with families by giving two reciprocal interpretation of the syntax of one theory into the other one. This proves that the equality is decidable in cwfs.

There is many work left to see why the syntax and the proofs are so complicated: this is a sign that there are things left to understand and maybe to change in the involved theories.

Acknowledgements

I would like to thank Thierry Coquand for having accepted to direct my training period in Chalmers, for giving me the opportunity to discover the world of type and category theories and for the time he spent with me.

I also want to thank Peter Dybjer who kindly explained to me what were his motivations when defining cwfs.

Finally, thank you Martin and #sos for being someone to talk to during those three months in Sweden and Caroline for being too much of the ball.

A From Gat to Cwf

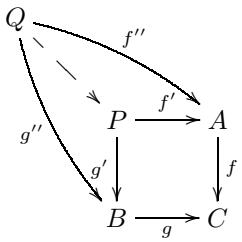
A.1 Contextual categories

In this section, we introduce a few notions that are close to *categories with families* (in fact they are even equivalent to cwfs) and which were historically introduced before (in [Car86]). They are interesting for several reasons. First they inspired the definition of cwfs. Actually cwfs were introduced to solve some of the problems those theories had (in particular, terms cannot be easily manipulated in those theories and they have a pullback condition which is rather unnatural and prevents them from being defined as gats). Moreover, they have been studied and it is quite easy to see that they are all equivalent. We will not be able to make a detailed presentation of those theories. The interested reader could read [Car86] and [Hof97].

The multiplicity slighly different categorical models of dependently-typed λ-calculus reflects the fact that the categorical interpretation of dependent types is undoubtly complicated.

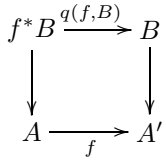
We first introduce the categorical notion of *pullback* which is used in the definition of **Con** and **Cwa**.

Definition 28. *Let* $\mathbf{C}$ *be a category and* f *and* g *two morphisms of same codomain consists of an object* P *and two morphisms* f' *and* g' *such that* $f \circ f' = g \circ g'$ *and, for all* Q *such that there exists two morphisms* f'' *and* g'' *such that* $f \circ f'' = g \circ g''$*, there exists a unique morphism from* Q *to* P*:*



Definition 29 (Contextual category). *A* contextual category *consists of*

- *A category* $\mathbf{C}$ *with terminal object* $\diamond$*.*
- *A tree structure on the objects of* $\mathbf{C}$ *such that the terminal object* $\diamond$ *is the unique least element of the tree. This means that there exists a function* p *(*father function*) on objects of* $\mathbf{C}$ *such that* $p(\diamond) = \diamond$*,* p *is injective on* $\mathbf{C} \setminus \{\diamond\}$*. If* $p(B) = A$ *we will write* $A \lhd B$*.*
- *For all* $A, A' \in \mathbf{C}$*, for all* $f : A \to A'$ *in* $\mathbf{C}$ *for all* $B \in \mathbf{C}$ *such that* $A' \lhd B$*, an object* f^*B *of* $\mathbf{C}$ *and a morphism* $q(f, B) : f^*B \to B$ *such that the diagram*



is a pullback in $\mathbf{C}$*; for all* $A, B \in \mathbf{C}$*, if* $A \lhd B$ *then* $\mathrm{id}_A^* B = B$ *and* $q(\mathrm{id}_A, B) = \mathrm{id}_B$*; and for all* $A, B, C \in \mathbf{C}$*,* $f : A \to A'$ *and* $f' : A' \to A''$*, the identities* $(f \circ f')^* B = f^*(f'^* B)$ *and* $q(f \circ F', B) = q(f, f'^* B) \circ q(f', B)$ *hold.*

Remark 13. We have removed the condition: for each $\Gamma \in \mathbf{C}$ there is a minimal integer n (the *level* of Γ) such that $p^n(\Gamma) = \diamond$ because we do not require the contexts to be finite.

Proposition 6. *The theories* **Gat** *and* **Con** *are equivalent.*

Proof. Given in [Car86]. □

In contextual categories, types cannot be easily manipulated which was corrected by the definition of *categories with attributes*.

A.2 Categories with attributes

Definition 30 (Category with attributes (cwa)). *A* category with attributes *consists of*

- *a category* $\mathbf{C}$ *with a terminal object* $\diamond$*;*
- *for each object* Γ *in* $\mathbf{C}$*, a collection* $\mathrm{Type}\,(\Gamma)$*, whose elements are called* Γ-indexed types *in* $\mathbf{C}$ *and a function* $f^* : \mathrm{Type}\,(\Gamma) \to \mathrm{Type}\,(\Delta)$ *for each* $f : \Delta \to \Gamma$ *such that for all* $A \in \mathrm{Type}\,(\Gamma)$*, the relations* $\mathrm{id}^*_\Gamma A = A$ *and* $(g \circ f)^* A = g^*(f^* A)$ *hold;*
- *for each* $A \in \mathrm{Type}\,(\Gamma)$ *and object* Γ, A *and a morphism* $\pi_A : \Gamma, A \to \Gamma$*;*
- *for each* $f : \Delta \to \Gamma$ *and* $A \in \mathrm{Type}\,(\Gamma)$*, a pullback diagram*

$$\begin{array}{ccc} \Delta, f^*A & \xrightarrow{\langle f, A\rangle} & \Gamma, A \\ {\scriptstyle \pi_{f^*A}}\downarrow & & \downarrow{\scriptstyle \pi_A} \\ \Delta & \xrightarrow[f]{} & \Gamma \end{array}$$

such that $\langle \mathrm{id}_\Gamma, A\rangle = \mathrm{id}_{\Gamma.A}$ *and* $\langle f \circ g, A\rangle = \langle f, A\rangle \circ \langle g, f^* A\rangle$*.*

Proposition 7. *The theories* **Con** *and* **Cwa** *are equivalent.*

Proof. A proof is given in [Hof97]. Two functors $O : \mathbf{Cwf} \to \mathbf{Con}$ and $W : \mathbf{Con} \to \mathbf{Cwf}$ such that $OW \cong \mathrm{id}_{\mathbf{Con}}$ and $WO \cong \mathrm{id}_{\mathbf{Cwf}}$ are defined, which proves the equivalence. □

By transitivity of the equivalence of categories, we can claim:

Lemma 33. *The theories* **Gat** *and* **Cwa** *are equivalent.*

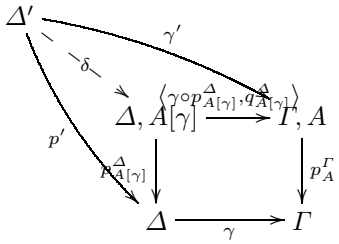
A.3 Pullback in Cwf

Proposition 8. *With the notations of the definition 5, the diagram*

$$\begin{array}{ccc} \Delta, A[\gamma] & \xrightarrow{\langle \gamma \circ p^\Delta_{A[\gamma]}, q^\Delta_{A[\gamma]}\rangle} & \Gamma, A \\ {\scriptstyle p^\Delta_{A[\gamma]}}\downarrow & & \downarrow{\scriptstyle p^\Gamma_A} \\ \Delta & \xrightarrow[\gamma]{} & \Gamma \end{array}$$

is a pullback.

Proof. Let Δ be a context and $p' : \Delta' \to \Delta$ and $\gamma' : \Delta' \to \Gamma, A$ be two morphisms such that $\gamma \circ p' = p^\Gamma_A \circ \gamma'$. Therefore the diagram



(without δ) is commutative. We must show that there exists a unique $\delta : \Delta' \to \Delta, A[\gamma]$ such that $\left\langle \gamma \circ p^\Delta_{A[\gamma]}, q^\Delta_{A[\gamma]} \right\rangle \circ \delta = \gamma'$ and $p^\Delta_{A[\gamma]} \circ \delta = p'$.

We can decompose γ' as $\gamma' \overset{\text{(M-Ext-Id)}}{=} \langle p_A^\Gamma, q_A^\Gamma \rangle \circ \gamma' \overset{\text{(M-Ext-S)}}{=} \langle p_A^\Gamma \circ \gamma', q_A^\Gamma[\gamma'] \rangle = \langle \gamma \circ p', M \rangle$ where $M \equiv q_A^\Gamma[\gamma']$ (of type $\Delta' \vdash A[\gamma \circ p']$). Let δ be the morphism

$$\delta \equiv \langle p', M \rangle : \Delta' \to \Delta, A[\gamma]$$

which is suitable since $p^\Delta_{A[\gamma]} \circ \delta \overset{\text{(M-C-L)}}{=} p'$ and

$$\begin{aligned}
\left\langle \gamma \circ p^\Delta_{A[\gamma]}, q^\Delta_{A[\gamma]} \right\rangle \circ \delta &\overset{\text{(M-Ext-S)}}{=} \left\langle \gamma \circ p^\Delta_{A[\gamma]} \circ \delta, q^\Delta_{A[\gamma]}[\delta] \right\rangle \\
&\overset{\text{(M-C-R)}}{=} \langle \gamma \circ p', q_A^\Gamma[\gamma'] \rangle \\
&= \langle p_A^\Gamma \circ \gamma', q_A^\Gamma[\gamma'] \rangle \\
&\overset{\text{(M-Ext-S)}}{=} \langle p_A^\Gamma, q_A^\Gamma \rangle \circ \gamma' \\
&\overset{\text{(M-Ext-Id)}}{=} \gamma'
\end{aligned}$$

Conversely, let's suppose that there is an other morphism $\delta' : \Delta' \to \Delta, A[\gamma]$ such that $\left\langle \gamma \circ p^\Delta_{A[\gamma]}, q^\Delta_{A[\gamma]} \right\rangle \circ \delta' = \gamma'$ and $p^\Delta_{A[\gamma]} \circ \delta' = p'$. As before, we have $\delta' = \left\langle p^\Delta_{A[\gamma]} \circ \delta', q^\Delta_{A[\gamma]}[\delta'] \right\rangle = \langle p', N \rangle$ where $N \equiv q^\Delta_{A[\gamma]}[\delta']$. The following equalities hold:

$$\begin{aligned}
N &= q^\Delta_{A[\gamma]}[\delta'] \\
&= q_A^\Gamma \left[\left\langle \gamma \circ p^\Delta_{A[\gamma]} \circ \delta', q^\Delta_{A[\gamma]}[\delta'] \right\rangle \right] \\
&= q_A^\Gamma \left[\left\langle \gamma \circ p^\Delta_{A[\gamma]}, q^\Delta_{A[\gamma]} \right\rangle \circ \delta' \right] \\
&= q_A^\Gamma[\gamma'] \\
&= M
\end{aligned}$$

and therefore $\delta' = \delta$. □

B Proof of the decidability of equality in LF

B.1 Soundness of LF

Lemma 34 (Well-formedness). *Let $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$ be a context, A and B two types and M a term. We shall write $\lceil \Gamma \rceil_i \equiv x_1 : A_1, \ldots, x_i : A_i$. The following rules hold*

1. *if $\Gamma \vdash$ is derivable then for all i such that $1 < i \leq n$, $x_i \notin \mathrm{DV}(\lceil \Gamma \rceil_{i-1})$, $\mathrm{FV}(A_i) \subset \mathrm{DV}(\lceil \Gamma \rceil_{i-1})$ and $\lceil \Gamma \rceil_{i-1} \vdash A_i$ appears in the derivation;*
2. *if $\Gamma \vdash \mathcal{J}$ is derivable then $\mathrm{FV}(\mathcal{J}) \subset \mathrm{DV}(\Gamma)$ and $\Gamma \vdash$ appears in the derivation, where $\mathcal{J}$ is either a type, a typed term, an equality between types or a typed equality between terms.*

Proof. Both properties can simultaneously be proven by a straightforward induction on the structure of the derivation of the hypothesis. □

Remark 14. A requirement for the context $\Gamma, x : A$ to be well-formed is the property $x \notin \Gamma$. However, to improve the readability of the proofs, we might omit some of the arguments related to this in the following but they have been verified. The main reason for that it that it leads to very long and technical proofs which are not really difficult.

Lemma 35. *The following rules hold*

1. *if $\Gamma \vdash A = B$ is derivable then $\Gamma \vdash A$ and $\Gamma \vdash B$ are derivable;*

2. *if* $\Gamma \vdash M = N : A$ *is derivable then* $\Gamma \vdash M : A$ *and* $\Gamma \vdash N : A$ *derivable.*

Proof. We will only show the first results of the conclusions (i.e. $\Gamma \vdash A$ and $\Gamma \vdash M : A$) since the other ones can be obtained using (Ty-Eq-Sym) and (Tm-Eq-Sym).

By induction on the derivation of the hypothesis.

- (Ty-Eq-Refl): $B \equiv A$ and $\Gamma \vdash A$ was derived.
- (Ty-Eq-Sym): $\Gamma \vdash B = A$ was derived and we can conclude by application of the induction hypothesis.
- (Ty-Eq-Trans): $\Gamma \vdash A = C$ and $\Gamma \vdash C = B$ were derived and we can conclude by application of the induction hypothesis.
- (El-Eq-C): A and B are of the form $A \equiv \mathsf{El}\ M$ and $B \equiv \mathsf{El}\ N$ and $\Gamma \vdash M = N : \star$ was derived. By induction hypothesis we have $\Gamma \vdash M : \star$ and $\Gamma \vdash N : \star$. We can conclude using (El-Eq-C).
- (Π-Eq-C): A and B are of the form $A \equiv \Pi x : A'.B'$ and $B \equiv \Pi x : A''.B''$ and $\Gamma \vdash \Pi x : A'.B'$ was derived.
- (Tm-Eq-Refl): $N \equiv M$ and $\Gamma \vdash M : A$ was derived.
- (Tm-Eq-Sym): $\Gamma \vdash N = M : A$ was derived and we can conclude by application of the induction hypothesis.
- (Tm-Eq-Trans): $\Gamma \vdash M = P : A$ and $\Gamma \vdash P = N : A$ were derived and we can conclude by application of the induction hypothesis.
- (Tm-Eq-Conv): $\Gamma \vdash M = N : B$ and $\Gamma \vdash B = A$ were derived. By induction hypothesis we have $\Gamma \vdash M : B$ and we can conclude that $\Gamma \vdash M : A$ by (Tm-Conv).
- (App-Eq-Conv): the goal is of the form $\Gamma \vdash \mathrm{App}_A\,(M, N) = \mathrm{App}_{A'}\,(M, N) : B$ and $\Gamma \vdash \mathrm{App}_A\,(M, N) : B$ was derived.
- (Π-I-Eq): M, N and A are of the form $M \equiv \lambda x.M'$, $N \equiv \lambda x.N'$ and $A \equiv \Pi x : A'.B'$ and $\Gamma \vdash \lambda x.M' : \Pi x : A'.B'$ was derived.
- (App-Eq): M, N and A are of the form $M \equiv \mathrm{App}_{\Pi x:A'.B'}\,(()\,M', N')$, $N \equiv \mathrm{App}_{\Pi x:A'.B'}\,(M', N'')$ and $A \equiv B'\,[N'/x]$ and $\Gamma \vdash M' : \Pi x : A'.B'$ and $\Gamma \vdash N' = N'' : A'$ were derived. By induction hypothesis $\Gamma \vdash N' : A'$ holds and therefore we have $\Gamma \vdash \mathrm{App}_{\Pi x:A'.B'}\,(M', N') : B'\,[N/x]$ using (App).
- (Π-C): M, N and A are of the form $M \equiv \mathrm{App}_{\Pi x:A'.B'}\,((\lambda x.M')\,, N')$, $N \equiv M'\,[N'/x]$ and $A \equiv B'\,[N'/x]$ and $\Gamma \vdash \lambda x.M' : \Pi x : A'.B'$ and $\Gamma \vdash N' : A'$. By (App), we can conclude that $\Gamma \vdash \mathrm{App}_{\Pi x:A'.B'}\,((\lambda x.M')\,, N')\,x : B'\,[N'/x]$.
- (Π-η): A is of the form $A \equiv \Pi x : A'.B'$ and $\Gamma \vdash M : \Pi x : A'.B'$ was derived. □

Lemma 36 (Weakening). *If* $x \notin \mathrm{DV}\,(\Gamma) \cup \mathrm{DV}\,(\Delta)$ *and* $\Gamma \vdash C$ *then*

1. *if* Γ, Δ *is a context then* $\Gamma, x : C, \Delta$ *is a context;*
2. *if* $\Gamma, \Delta \vdash A$ *then* $\Gamma, x : C, \Delta \vdash A$*;*
3. *if* $\Gamma, \Delta \vdash A = B$ *then* $\Gamma, x : C, \Delta \vdash A = B$*;*
4. *if* $\Gamma, \Delta \vdash M : A$ *then* $\Gamma, x : C, \Delta \vdash M : A$*;*
5. *if* $\Gamma, \Delta \vdash M = N : A$ *then* $\Gamma, x : C, \Delta \vdash M = N : A$*.*

Proof. By induction on the structure of the derivation of the hypothesis.

- (C-Emp): we have $\Gamma \equiv \Delta \equiv \diamond$ and by hypothesis $\diamond \vdash C$ therefore $\diamond, x : C, \diamond$ is a context by (C-Ext).
- (C-Ext): Δ is of the form $\Delta \equiv \Delta', y : A$ and $\Gamma, \Delta' \vdash A$ was derived. By induction hypothesis we have $\Gamma, x : C, \Delta' \vdash A$ and therefore $\Gamma, x : C, \Delta', y : A$ is a context.
- (Star): since Γ, Δ is a context, by induction hypothesis $\Gamma, x : C, \Delta$ is a context and by (Star) we have $\Gamma, x : C, \Delta \vdash \star$.
- (Elem): $\Gamma, \Delta \vdash \star$ was derived, therefore by induction hypothesis $\Gamma, x : C, \Delta \vdash \star$ holds and by (Elem) we have $\Gamma, x : C, \Delta \vdash \mathsf{El}\ M$.
- (Exp): A is of form $A \equiv \Pi y : A'.B'$ and $\Gamma, \Delta, y : A' \vdash B'$ was derived, therefore by induction hypothesis $\Gamma, x : C, \Delta, y : A \vdash B$ holds and by (Exp) we have $\Gamma, x : C, \Delta \vdash \Pi y : A'.B'$.

– (Ty-Eq-Refl): $B \equiv A$ and $\Gamma, \Delta \vdash A$ was derived. By induction hypothesis we have $\Gamma, x : C, \Delta \vdash A$ and we conclude using (Ty-Eq-Refl).
– etc. The property is proven for all the other rules using the same model, by applying the induction hypothesis to the judgments obtained by inverting the rules and reapplying the rule to the obtained judgments. □

Lemma 37. *The following rule holds*

$$\frac{\Gamma \vdash M : \Pi x : A.B \qquad \Gamma \vdash N = N' : A}{\Gamma \vdash \mathrm{App}_{\Pi x:A.B}(M, N) = \mathrm{App}_{\Pi x:A.B}(M, N') : B[N/x]}$$

Proof. Suppose that $\Gamma \vdash M = M' : \Pi x : A.B$ and $\Gamma \vdash N : A$ hold. The following derivation is valid

$$\dfrac{\dfrac{\dfrac{}{\Gamma, x : \Pi y : A.B \vdash x : \Pi y : A.B}(\textsc{Var}) \qquad \dfrac{\dfrac{}{\Gamma \vdash N : A}(\text{Hypothesis})}{\Gamma, x : \Pi y : A.B \vdash N : A}(\text{Lemma 3})}{\Gamma, x : \Pi y : A.B \vdash xN : B[N/y]}(\textsc{App})}{\Gamma \vdash \lambda x.xN : \Pi x : (\Pi y : A.B).B[N/y]}(\textsc{Abs})$$

Since $\Gamma \vdash M = M' : \Pi x : A.B$, by lemma 1 we know that $y \notin \mathrm{FV}(B)$ and therefore $B[N/y] \equiv B$. By (App-Eq) we have $\Gamma \vdash (\lambda x.xN)\, M = (\lambda x.xN)\, M' : B[M/x]$ and by (Π-C) we have $\Gamma \vdash (\lambda x.xN)\, M = MN : B[M/x]$ and $\Gamma \vdash (\lambda x.xN)\, M' = M'N : B[M/x]$. Finally we can conclude that $\Gamma \vdash MN = M'N : B[M/x]$ using (Tm-Eq-Sym) and (Tm-Eq-Trans). □

Lemma 38. *If $\Gamma, x : A, \Delta \vdash x : B$ then $\Gamma \vdash A = B$.*

Proof. The proof is done by induction on the derivation of $\Gamma, x : A, \Delta \vdash x : B$.

– (Var): we have $B \equiv A$ and the result follows by (Ty-Eq-Refl).
– (Tm-Conv): $\Gamma, x : A, \Delta \vdash x : C$ and $\Gamma, x : A, \Delta \vdash C = B$ where derived for some type C. By induction hypothesis $\Gamma, x : A, \Delta \vdash A = C$ holds and the result follows using (Tm-Eq-Trans). □

Lemma 39 (Soundness of the substitution). *If $\Gamma \vdash N : B$ is derivable then*

1. *if $\Gamma, x : B, \Delta$ is a context then $\Gamma, \Delta[N/x]$ is a context;*
2. *if $\Gamma, x : B, \Delta \vdash A$ is derivable then $\Gamma, \Delta[N/x] \vdash A[N/x]$ is derivable;*
3. *if $\Gamma, x : B, \Delta \vdash A = A'$ is derivable then $\Gamma, \Delta[N/x] \vdash A[N/x] = A'[N/x]$ is derivable;*
4. *if $\Gamma, x : B, \Delta \vdash M : A$ is derivable then $\Gamma, \Delta[N/x] \vdash M[N/x] : A[N/x]$ is derivable;*
5. *if $\Gamma, x : B, \Delta \vdash M = M' : A$ is derivable then $\Gamma, \Delta[N/x] \vdash M[N/x] = M'[N/x] : A[N/x]$ is derivable.*

Proof. The proof is done by induction on the derivation of the hypothesis.

– (C-Emp): this rule cannot have been used to prove that $\Gamma, x : B, \Delta$ since $\Gamma, x : B, \Delta \not\equiv \diamond$.
– (C-Ext): Δ is of the form $\Delta \equiv \Delta', y : A$ and $\Gamma, x : B, \Delta' \vdash A$ was derived and by induction hypothesis $\Gamma, \Delta'[N/x] \vdash A[N/x]$ holds. We can conclude using (C-Ext).
– (Star): we have $\star[N/x] \equiv \star$ and $\Gamma, x : B, \Delta \vdash$ was derived. By induction hypothesis $\Gamma, \Delta[N/x]$ holds and $\Gamma, \Delta[N/x] \vdash \star$ is derivable using (Star).
– (Elem): A is of the form $A \equiv \mathsf{El}\ M$, therefore $A[N/x] \equiv \mathsf{El}\ (M[N/x])$ and $\Gamma, x : B, \Delta \vdash M : \star$ was derived. By induction hypothesis we have $\Gamma, \Delta[N/x] \vdash M[N/x] : \star[N/x]$ and therefore, since $\star[N/x] \equiv \star$, $\Gamma, \Delta[N/x] \vdash \mathsf{El}\ (M[N/x])$ is derivable using (Elem).
– (Exp): A is of the form $A \equiv \Pi y : A'.B'$ and $\Gamma, x : B, \Delta, y : B' \vdash B'$ was derived. By induction hypothesis we have $\Gamma, \Delta[N/x], y : A'[N/x] \vdash B'[N/x]$ and by (Exp) we conclude that $\Gamma, \Delta[N/x] \vdash (\Pi y : A'.B')[N/x]$.

– ...
– (Var):
  • Suppose that $M \equiv x$. Then $M[N/x] \equiv N$ and since $\Gamma, x : B, \Delta \vdash$ was derived, by induction hypothesis we know that $\Gamma, \Delta[N/x]$ is a context. By hypothesis $\Gamma \vdash N : B$ holds. This implies that $\Gamma, \Delta[N/x] \vdash N : B$ is also derivable by induction on Δ.
  The property is immediate if $\Delta \equiv \diamond$.
  Suppose that $\Delta \equiv \Delta', y : C$ and assume that $\Gamma, \Delta'[N/x] \vdash N : B$. Since by hypothesis $\Gamma, x : B, \Delta', y : C \vdash x : A$, by lemma 1 $\Gamma, x : B, \Delta', y : C$ is a context and therefore $y \notin \mathrm{DV}(\Gamma) \cup \mathrm{DV}(\Delta') \cup \{x\}$. Since $\Gamma \vdash N : B$, by lemma 1, $\mathrm{FV}(N) \subseteq \mathrm{DV}(\Gamma)$ which implies $y \notin \mathrm{DV}(N)$. Thus $y \notin \mathrm{DV}(\Delta) \cup \mathrm{DV}(\Delta'[N/x])$. Moreover since $\Gamma, x : B, \Delta', y : C$ is a context, by lemma 1 $\Gamma, x : B, \Delta' \vdash C$ was derived and by induction hypothesis $\Gamma, \Delta'[N/x] \vdash C[N/x]$. By lemma 3 we can finally conclude that $\Gamma, \Delta'[N/x], y : C[N/x] \vdash N : B$.
  By lemma 5, since $\Gamma, x : B, \Delta \vdash x : A$ was derived, $\Gamma \vdash A = B$ holds. Since we also have $\Gamma, \Delta[N/x] \vdash N : B$, we can conclude that $\Gamma, \Delta[N/x] \vdash M[N/x] : A$ by using (Tm-Conv).
  • Suppose that $M \equiv y$. Then $\Gamma, x : B, \Delta \equiv \Gamma', y : A, \Delta'$. The proof can be done by distinguishing the cases where $y : A$ appears in Γ or in Δ, applying the induction hypothesis to the judgments obtained by inversion of the rules and concluding using (Var).
– ...
The omitted cases can all be proven the same way, by applying the induction hypothesis to the judgments obtained by inversion of the rules and concluding using (Var). □

Lemma 40. *The relation $\overset{\beta}{=}$ is an equivalence relation.*

Proof. By definition of $\overset{\beta}{=}$. □

Lemma 41. *The substitution preserves β-convertibility:*

1. *if $M \overset{\beta}{=} M'$ then $M[N/x] \overset{\beta}{=} M'[N/x]$;*
2. *if $N \overset{\beta}{=} N'$ then $M[N/x] \overset{\beta}{=} M[N'/x]$.*

Proof. 1. By induction on the proof of $M \overset{\beta}{=} M'$.
  – (β-Eq-Refl): In this case $M' \equiv M$ and the result follows by reflexivity of $\overset{\beta}{=}$.
  – (β-Eq-Ext-R): We have $M \overset{\beta}{=} M''$ and $M'' \to_\beta M'$. We can show that $M''[N/x] \overset{\beta}{=} M'[N/x]$ by induction on the proof of $M'' \to_\beta M'$ (the proof is quite straightforward). By induction hypothesis we have $M[N/x] \overset{\beta}{=} M''[N/x]$ and we conclude using the transitivity of $\overset{\beta}{=}$.
  – (β-Eq-Ext-L): The proof is similar to the previous case.
2. By induction on M.
  – $M \equiv x$: $M[N/x] \overset{\beta}{=} N \overset{\beta}{=} N' \overset{\beta}{=} M[N'/x]$.
  – $M \equiv y$: $M[N/x] \overset{\beta}{=} M \overset{\beta}{=} M[N'/x]$.
  – $M \equiv \lambda y.M'$: by induction hypothesis $M'[N/x] \overset{\beta}{=} M'[N'/x]$ and therefore $(\lambda y.M')[N/x] \equiv \lambda y.M'[N/x] \overset{(\beta\text{-Red-Abs-C})}{=} \lambda y.M'[N'/x] \equiv (\lambda y.M')[N'/x]$.
  – $M \equiv M'M''$: the result is obtained similarly using the induction hypothesis and (β-Red-App-C-L) and (β-Red-App-C-R). □

Lemma 42 (Subject reduction). *If $\Gamma \vdash M : A$ and $M \to_\beta M'$ then $\Gamma \vdash M = M' : A$.*

Proof. The proof is done by induction on the derivation of $\Gamma \vdash M : A$.

– (Var): M is a variable and cannot β-reduce.
– (Tm-Conv): $\Gamma \vdash M : B$ and $\Gamma \vdash B = A$ were derived and by induction hypothesis we have $\Gamma \vdash M = M' : B$. We can conclude using (Tm-Eq-Conv).

– (Abs): M and A are of the form $M \equiv \lambda x.N$ and $A \equiv \Pi x : A'.B'$ and $\Gamma, x : A' \vdash N : B'$ was derived. By hypothesis, $M \to_\beta M'$ holds and it must have been derived using (β-Red-Abs-C): M' must be of the form $M' \equiv \lambda x.N'$ with $N \to_\beta N'$. By induction hypothesis we have $\Gamma, x : A' \vdash N = N' : B'$. And we conclude using (Π-I-Eq).
– (App): We distinguish cases according to the rule used to prove $M \to_\beta M'$.
  • (β-Red-App): M, M' and B are of the form $M \equiv (\lambda x.M')\, N$, $M' \equiv M'\,[N/x]$ and $B \equiv B'\,[N/x]$ and $\Gamma \vdash \lambda x.M' : \Pi x : A'.B'$ and $\Gamma \vdash N : A'$ were derived. We can conclude using (Π-C).
  • (β-Red-App-C-L): M, M' and B are of the form $M \equiv M'N$, $M' \equiv M''N'$ and $B \equiv B'\,[N'/x]$, $\Gamma \vdash M' : \Pi x : A'.B'$ and $\Gamma \vdash N' : A'$ were derived and $M' \to_\beta M''$. By induction hypothesis we have $\Gamma \vdash M' = M'' : \Pi x : A'.B'$ and we can conclude by lemma 4 that $\Gamma \vdash M'N' = M''N' : B'\,[N'/x]$.
  • (β-Red-App-C-R): M, M' and B are of the form $M \equiv M'N'$, $M' \equiv M'N''$ and $B \equiv B'\,[N'/x]$, $\Gamma \vdash M' : \Pi x : A'.B'$ and $\Gamma \vdash N' : A'$ were derived and $N' \to_\beta N''$. By induction hypothesis we have $\Gamma \vdash N' = N'' : A$ and we conclude by (App-Eq) that $\Gamma \vdash M'N' = M'N'' : B'\,[N'/x]$. □

Lemma 43. *If $\Gamma \vdash M : A$ and $M \overset{\beta}{=} M'$ then $\Gamma \vdash M = M' : A$.*

Proof. By induction on the proof of $M \overset{\beta}{=} M'$.

– (β-Eq-Refl): $M' \equiv M$ and we conclude using (Tm-Eq-Refl).
– (β-Eq-Ext-R): We have $M \overset{\beta}{=} M'' \to_\beta M'$. By induction hypothesis we have $\Gamma \vdash M = M'' : A$ and by lemma 9 we have $\Gamma \vdash M'' = M' : A$. We can conclude using (Tm-Eq-Trans).
– (β-Eq-Ext-L): We have $M \overset{\beta}{=} M''_\beta \leftarrow M'$. By induction hypothesis we have $\Gamma \vdash M = M'' : A$ and by lemma 9 we have $\Gamma \vdash M' = M'' : A$. We can conclude using (Tm-Eq-Sym) and (Tm-Eq-Trans). □

Lemma 44 (Soundness of a β-convertible substitution). *If $\Gamma \vdash N : B$ is derivable and $N \overset{\beta}{=} N'$ then*

1. *if $\Gamma, x : B, \Delta \vdash M = M' : A$ is derivable then $\Gamma, \Delta\,[N/x] \vdash M\,[N/x] = M'\,[N'/x] : A\,[N/x]$ is derivable.*
2. *if $\Gamma, x : B, \Delta \vdash A = A'$ is derivable then $\Gamma, \Delta\,[N/x] \vdash A\,[N/x] = A'\,[N'/x]$ is derivable;*

Proof. 1. Since $\Gamma, x : B, \Delta \vdash M = M' : A$ is derivable, by lemma 2 we have $\Gamma, x : B, \Delta \vdash M : A$. Therefore $\Gamma, \Delta\,[N/x] \vdash M\,[N/x] : A\,[N/x]$ by lemma 6. We also have $N \overset{\beta}{=} N'$, which implies, by lemma 8, $M\,[N/x] \overset{\beta}{=} M\,[N'/x]$. Finally, by lemma 10 we can conclude that $\Gamma, \Delta\,[N/x] \vdash M\,[N/x] = M\,[N'/x] : A\,[N/x]$.

2. The proof is done by induction on the derivation of $\Gamma, x : B \vdash A = A'$.
   – (Ty-Eq-Refl): $A' \equiv A$ and the result is obtained by (Ty-Eq-Refl).
   – (Ty-Eq-Sym): $\Gamma, x : B, \Delta \vdash A' = A$ was derived and by induction hypothesis $\Gamma, \Delta\,[N/x] \vdash A'\,[N/x] = A\,[N'/x]$ is derivable. We can conclude using (Ty-Eq-Sym).
   – (Ty-Eq-Trans): $\Gamma, x : B, \Delta \vdash A = A''$ and $\Gamma, x : B, \Delta \vdash A'' = A'$ were derived, thus by induction hypothesis $\Gamma, \Delta\,[N/x] \vdash A\,[N/x] = A''\,[N'/x]$ and by lemma 6 $\Gamma, \Delta\,[N/x] \vdash A''\,[N'/x] = A'\,[N'/x]$. We conclude using (Ty-Eq-Trans).
   – (El-Eq-C): $A \equiv \mathsf{El}\ M$, $A' \equiv \mathsf{El}\ M'$ and $\Gamma, x : B, \Delta \vdash M = M' : \star$ was derived. By 1. we deduce that $\Gamma, \Delta\,[N/x] \vdash M\,[N/x] = M'\,[N'/x] : \star$ (since $\star\,[N/x] \equiv \star$). Thus by (El-Eq-C) we conclude that $\Gamma, \Delta\,[N/x] \vdash \mathsf{El}\ M\,[N/x] = \mathsf{El}\ M'\,[N'/x]$.
   – (Π-Eq-C): $A \equiv \Pi x : A'.B'$, $A' \equiv \Pi x : A''.B''$ and $\Gamma, x : B, \Delta \vdash \Pi y : A'.B'$, $\Gamma, x : B, \Delta \vdash A' = A''$ and $\Gamma, x : B, \Delta, y : A' \vdash B' = B''$ were derived. By lemma 6 we have $\Gamma, \Delta\,[N/x] \vdash \Pi y : A'\,[N/x].B'\,[N/x]$ and by induction hypothesis we have $\Gamma, \Delta\,[N/x] \vdash A'\,[N/x] = A''\,[N'/x]$ and $\Gamma, \Delta\,[N/x], y : A'\,[N/x] \vdash B'\,[N'/x] = B''\,[N/x]$. We can then conclude using (Π-Eq-C). □

Theorem 6 (Church-Rosser). *If $M \overset{\beta}{=} M'$ then there exists a term N such that $M \xrightarrow{\beta} N$ and $M' \xrightarrow{\beta} N$.*

Proof. This property is classic; we will not reproduce its demonstration here (see for example [LS86] or [Fau02]).

Lemma 45. *If $\Gamma \vdash M : A$, then $\Gamma \vdash M = M' : A$ iff $M \overset{\beta\eta}{=} M'$.*

Proof. Two quite straightforward inductions. □

B.2 Decidability of equality in LF

Lemma 46. *If $M \overset{\beta}{=} N$ then $M[\rho] \overset{\beta}{=} N[\rho]$.*

Proof. The proof is similar to the one of lemma 8. □

Lemma 47. *The following rules hold, giving us a recursive definition of the application of an environment to a term.*

1. *If $x \notin \mathrm{FV}(\mathcal{J})$ then $\mathcal{J}[\langle \rho, x \mapsto M\rangle] = \mathcal{J}[\rho]$;*
2. *$(\Pi x : A.B)[\rho] = \Pi x : (A[\rho]).(B[\langle \rho, x \mapsto x\rangle])$;*
3. *$(\lambda x.M)[\rho] = \lambda x.(M[\langle \rho, x \mapsto x\rangle])$;*
4. *$(MN)[\rho] = (M[\rho])(N[\rho])$;*
5. *$\eta_A(M)[\rho] \overset{\beta}{=} \eta_{A[\rho]}(M[\rho])$.*

Proof. The lemmata can be proven independently.

1. Immediate.
2. Immediate.
3. Immediate.
4. Immediate.
5. By induction on A:
   - if $A \equiv \star$ or $A \equiv \mathsf{El}\ N$ then $\eta_A(M)[\rho] \overset{\beta}{=} M[\rho] \overset{\beta}{=} \eta_{A[\rho]}(M[\rho])$
   - if $A \equiv \Pi x : A'.B'$ then

$$\begin{aligned}
\eta_{\Pi x:A'.B'}(M)[\rho] &\overset{\beta}{=} \left(\lambda z.\eta_{B'[\eta_{A'}(z)/x]}\left(M\eta_{A'}(z)\right)\right)[\rho] \\
&\overset{\beta}{=} \lambda z.\left(\eta_{B'[\eta_{A'}(z)/x]}\left(M\eta_{A'}(z)\right)[\langle \rho, z \mapsto z\rangle]\right) \\
&\overset{\text{IH}}{=} \lambda z.\left(\eta_{B'[\eta_{A'}(z)/x][\langle \rho, z\mapsto z\rangle]}\left(\left(M\eta_{A'}(z)\right)[\langle \rho, z \mapsto z\rangle]\right)\right) \\
&\overset{\beta}{=} \lambda z.\eta_{B'[\eta_{A'}(z)/x][\langle \rho, z\mapsto z\rangle]}\left(\left(M[\langle \rho, z \mapsto z\rangle]\right)\eta_{A'[\langle \rho, z\mapsto z\rangle]}(z)\right) \\
&\overset{\beta}{=} \lambda z.\eta_{B'[\langle \rho, x\mapsto x\rangle][\eta_{A'[\rho](z)/x}]}\left(\left(M[\rho]\right)\eta_{A'[\rho]}(z)\right) \\
&\overset{\beta}{=} \eta_{\Pi x:(A'[\rho]).(B'[\langle \rho, x\mapsto x\rangle])}(M[\rho]) = \eta_{(\Pi x:A.B)[\rho]}(M[\rho])
\end{aligned}$$

□

Remark 15. To prove the last point, we took care not to use the fact that $\eta_{A[\rho]} \equiv \eta_A$ (cf. remark 9) in order to improve the extensibility of the proof.

Lemma 48. $\mathrm{FV}(\eta_A(M)) = \mathrm{FV}(M)$.

Proof. Straightforward induction on A. □

Lemma 49 (Recursive definition of $[\rho_\Gamma]$). *If $\Gamma, x : A$ is a context then*

$$M[\rho_{\Gamma,x:A}] \equiv M\left[\left\langle \rho_\Gamma, x \mapsto \eta_{A[\rho_\Gamma]}(x)\right\rangle\right] \equiv M[\eta_A(x)][\rho_\Gamma]$$

Proof. By induction on the length of Γ.

- If $\Gamma \equiv \diamond$ then the property is immediate.
- Else, since $\Gamma, x : A$ is a context $x \notin \mathrm{DV}(\Gamma)$. Moreover by lemma 15, $\mathrm{FV}\left(\eta_{A[\rho_\Gamma]}(x)\right) = \{x\}$. Therefore $M\left[\left\langle \rho_\Gamma, x \mapsto \eta_{A[\rho_\Gamma]}(x)\right\rangle\right] \equiv M\left[\eta_{A[\rho_\Gamma]}(x)\right][\rho_\Gamma]$ since ρ_Γ will not do any substitution on x. Finally $\eta_A(M) \equiv \eta_{A[\rho]}$ since substitution is only done in $\mathsf{El}\ N$ which does not change the incarnation. □

Lemma 50. *If $M \overset{\beta}{=} N$ then $\eta_A(M) \overset{\beta}{=} \eta_A(N)$.*

Proof. By induction on A.

- The property is immediate if $A \equiv \star$ or $A \equiv \mathsf{El}\ P$.
- If $A \equiv \Pi x : B.C$ then, since $M\ \eta_B(z) \overset{\beta}{=} N\ \eta_B(z)$ because $M \overset{\beta}{=} N$, using the induction hypothesis, we have

$$\begin{aligned}\eta_A(M) &\equiv \lambda z.\eta_{C[\eta_B(z)/x]}(M\ \eta_B(z)) \\ &\overset{\beta}{=} \lambda z.\eta_{C[\eta_B(z)/x]}(N\ \eta_B(z)) \\ &\equiv \eta_A(N)\end{aligned}$$

This induction is well-founded since the number of Π in A strictly decreases for each recursive application of the lemma. □

Lemma 51. *If $\Gamma \vdash A = B$ then $\eta_A(M) \overset{\beta}{=} \eta_B(M)$.*

Proof. By induction on the derivation of $\Gamma \vdash A = B$.

- (Ty-Eq-Refl), (Ty-Eq-Sym) and (Ty-Eq-Trans): the result is obtained by using the induction hypothesis and respectively the reflexivity, symmetry and transitivity of $\overset{\beta}{=}$.
- (El-Eq-C): A and B are of the form $A \equiv \mathsf{El}\ M'$ and $B \equiv \mathsf{El}\ N'$ therefore $\eta_A(M) \equiv M \equiv \eta_B(M)$.
- (Π-Eq-C): A and B are of the form $A \equiv \Pi x : A'.B'$ and $B \equiv \Pi x : A''.B''$ and $\Gamma \vdash A' = A''$ and $\Gamma, x : A' \vdash B' = B''$ were derived. From $\Gamma \vdash A' = A''$, using the induction hypothesis, we can deduce that for all term N $\eta_{A'}(N) \overset{\beta}{=} \eta_{A''}(N)$. From the last one we can deduce $\Gamma, x : A' \vdash B'\left[\eta_{A'}(z)/x\right] = B''\left[\eta_{A'}(z)/x\right]$ by lemma 6. Therefore, by induction hypothesis, for all term N we have $\eta_{B'[\eta_{A'}(z)/x]}(N) \overset{\beta}{=} \eta_{B''[\eta_{A'}(z)/x]}(N)$. Thus, we can write using lemma 17

$$\begin{aligned}\eta_A(M) &\equiv \eta_{\Pi x:A'.B'}(M) \\ &\equiv \lambda z.\eta_{B'[\eta_{A'}(z)/x]}(M\ \eta_{A'}(z)) \\ &\overset{\beta}{=} \lambda z.\eta_{B'[\eta_{A'}(z)/x]}(M\ \eta_{A''}(z)) \\ &\overset{\beta}{=} \lambda z.\eta_{B''[\eta_{A'}(z)/x]}(M\ \eta_{A''}(z))\end{aligned}$$

□

Lemma 52. *If $N = N' : \overline{A} \Rightarrow M[N/x] = M'[N'/x] : \overline{B[N/x]}$ then $\lambda x.M = \lambda x.M' : \overline{\Pi x : A.B}$.*

Proof. By definition of $\overline{\Pi x : A.B}$. □

Lemma 53. *The rules of the figure 10 have been proven to hold in per-models in [CPT03].*

Lemma 54. *The relations $\cong$ and $=$ (on environments in a context, on types in a context and on typed terms in a context) are equivalence relation.*

Proof. Straightforward induction on the structure of the derivations. □

Type formation and equalities

$$\frac{\Gamma : \overline{\mathrm{Ctxt}}}{\star[\![\Gamma]\!]}(\textsc{Star}) \qquad \frac{M = N : \star[\![\Gamma]\!]}{\mathsf{El}\ M = \mathsf{El}\ N[\![\Gamma]\!]}(\textsc{El-Eq-C}) \qquad \frac{A_1 = A_2[\![\Gamma]\!] \qquad B_1 = B_2[\![\Gamma, x : A_1]\!]}{\Pi x : A_1.B_1 = \Pi x : A_2.B_2[\![\Gamma]\!]}(\Pi\text{-Eq-C})$$

Terms

$$\frac{\Gamma, x : A : \overline{\mathrm{Ctxt}}}{x : A[\![\Gamma, x : A]\!]}(\textsc{Var}) \qquad \frac{M : B[\![\Gamma, x : A]\!]}{\lambda x.M : \Pi x : A.B[\![\Gamma]\!]}(\textsc{Abs}) \qquad \frac{M : \Pi x : A.B[\![\Gamma]\!] \qquad N : A[\![\Gamma]\!]}{MN : B\,[N/x]\,[\![\Gamma]\!]}(\textsc{App})$$

Type conversion

$$\frac{M = N : A[\![\Gamma]\!] \qquad A = B[\![\Gamma]\!]}{M = N : B[\![\Gamma]\!]}(\textsc{Tm-Eq-Conv})$$

Weakening

$$\frac{B_1 = B_2[\![\Gamma]\!] \qquad \Gamma, x : A : \overline{\mathrm{Ctxt}}}{B_1 = B_2[\![\Gamma, x : A]\!]} \qquad \frac{M = N : B[\![\Gamma]\!] \qquad \Gamma, x : A : \overline{\mathrm{Ctxt}}}{M = N : B[\![\Gamma, x : A]\!]}$$

Figure 13. Derivable rules in per-models

Lemma 55. *If $\rho = \rho' : \Gamma$ and $y \notin \Gamma$ then for any M, $\rho = \langle \rho', y \mapsto M \rangle : \Gamma$.*

Proof. By induction on the proof of $\rho = \rho' : \Gamma$.

Consider a proof ending with

$$\frac{\rho = \rho' : \Gamma \qquad A\,[\rho] = A\,[\rho'] \qquad \rho x = \rho' x : \overline{A\,[\rho]}}{\rho = \rho' : \Gamma, x : A}$$

If suffices to show $\rho = \langle \rho', y \mapsto M \rangle : \Gamma$ and $\rho x = \langle \rho', y \mapsto M \rangle\, x : \overline{A\,[\rho]}$. The former holds by induction hypothesis, the latter because $y \notin \mathrm{DV}\,(\Gamma, x : A)$, so $y \neq x$. □

Lemma 56 (Interpretation of LF in a per-model). *The following rules hold*

- *if $\Gamma \vdash$ then $\Gamma : \overline{\mathrm{Ctxt}}$;*
- *if $\Gamma \vdash A$ then $A[\![\Gamma]\!]$;*
- *if $\Gamma \vdash A_1 = A_2$ then $A_1 = A_2[\![\Gamma]\!]$;*
- *if $\Gamma \vdash M : A$ then $M : A[\![\Gamma]\!]$;*
- *if $\Gamma \vdash M_1 = M_2 : A$ then $M_1 = M_2 : A[\![\Gamma]\!]$;*

Proof. By induction on the structure of the derivation of the hypothesis.

- (C-Emp): by (Env-C-Emp).
- (C-Ext): by (Env-C-Ext).
- (Star): by (Star).
- (Elem): by (El-Eq-C).
- (Exp): by (Π-Eq-C), using the fact that by lemma 1, $\Gamma \vdash A$ was derived and therefore $A[\![\Gamma]\!]$ holds.
- (Ty-Eq-Refl), (Ty-Eq-Sym) and (Ty-Eq-Trans): by lemma 21.
- (El-Eq-C): by (El-Eq-C).
- (Π-Eq-C): by (Π-Eq-C).

– (Var): by (Var).
– (Tm-Conv): by (Tm-Eq-Conv).
– (Abs): by (Abs).
– (App): by (App).
– (Tm-Eq-Refl), (Tm-Eq-Sym) and (Tm-Eq-Trans): by lemma 21.
– (Tm-Eq-Conv): by (Tm-Eq-Conv).
– (Π-I-Eq): we want to derive $\Gamma \vdash \lambda x.M = \lambda x.M' : \Pi x : A.B$ and by inversion $\Gamma \vdash \lambda x.M : \Pi x : A.B$ and $\Gamma, x : A \vdash M = M' : B$. By induction hypothesis we have $\lambda x : M : \Pi x : A.B[\![\Gamma]\!]$ and $M = M' : B[\![\Gamma, x : A]\!]$. Obviously, we are going to use the rule (Tm-Eq) to prove the conclusion. By lemma 1 $\Gamma \vdash A$ was derived and therefore by induction hypothesis $A[\![\Gamma]\!]$ holds. What remains to be proven is $\forall \rho, \rho',\ \rho = \rho' : \Gamma \Rightarrow (\lambda x.M)\,[\rho] = (\lambda x.M')\,[\rho'] : \overline{(\Pi x : A.B)\,[\rho]}$.
Let ρ and ρ' be two environments such that $\rho = \rho' : \Gamma$.
Let N and N' be two terms such that $N = N' : \overline{A\,[\rho]}$. Since $\Gamma, x : A \vdash M = M' : B$ holds, by lemma 1 $\Gamma, x : A$ is a context and $x \notin \mathrm{DV}(\Gamma)$. Therefore, by lemma 22 we have $\langle \rho, x \mapsto N\rangle = \langle \rho', x \mapsto N'\rangle : \Gamma$, and by inversion on (Ty-Eq) which was used to derive $A[\![\Gamma]\!]$, we have $A\,[\langle \rho, x \mapsto N\rangle] \cong A\,[\langle \rho', x \mapsto N'\rangle]$. By definition of N and N' we also have $\langle \rho, x \mapsto N\rangle\, x \equiv N = N' \equiv \langle \rho', x \mapsto N'\rangle\, x : \overline{A\,[\rho]}$. We can now use (Env-C-Ext), which shows that $\langle \rho, x \mapsto N\rangle = \langle \rho', x \mapsto N'\rangle : \Gamma, x : A$.
By inversion of the rule (Tm-Eq) on the hypothesis $M = M' : B[\![\Gamma, x : A]\!]$, we have therefore $M\,\langle \rho, x \mapsto N\rangle = M'\,\langle \rho', x \mapsto N'\rangle : \overline{B\,[\langle \rho, x \mapsto N\rangle]}$. This can be rewritten as $M\,[\rho]\,[N/x] = M'\,[\rho']\,[N'/x] : \overline{B\,[\rho]\,[N/x]}$. Finally, by lemma 19, this implies $\lambda x.M\,[\rho] = \lambda x.M'\,[\rho'] : \overline{\Pi x : A\,[\rho]\,.B\,[\rho]}$.
– The other rules ((App-Eq), (Π-C), (Π-η)) can be handled using the same kind of arguments. □

Lemma 57. *If $\Gamma \vdash M = N : A$ then $\eta_A(M)\,[\rho_\Gamma] \overset{\beta}{=} \eta_A(N)\,[\rho_\Gamma]$.*

Proof. Suppose that $\Gamma \vdash M = N : A$. Then by lemma 23 we have $M = N : A[\![\Gamma]\!]$ and this has been proven to imply $\eta_A(M)\,[\rho_\Gamma] \overset{\beta}{=} \eta_A(N)\,[\rho_\Gamma]$ in [CPT03]. □

Lemma 58. *If $\Gamma \vdash M : A$ then $\Gamma \vdash M = \eta_A\,(M) : A$.*

Proof. By induction on A.

– If $A \equiv \star$ or $A \equiv \mathsf{El}\ M$ then $\eta_A\,(M) \equiv M$ and the result is obtained by (Tm-Eq-Refl).
– Suppose that $A \equiv \Pi x : A'.B'$. Then $\eta_A\,(M) \equiv \lambda z.\eta_{B'[\eta_{A'}(z)/x]}\,(M\ \eta_{A'}\,(z))$. Since $\Gamma, z : A' \vdash z : A'$, by induction hypothesis we have $\Gamma, z : A' \vdash z = \eta_{A'}(z) : A'$. Therefore, by rules (App-Eq), we have $\Gamma, z : A' \vdash Mz = M\,(\eta_{A'}\,(z)) : B'[z/x]$. By induction hypothesis and (Tm-Eq-Trans) this implies $\Gamma, z : A' \vdash Mz = \eta_{B'[\eta_{A'}(z)/x]}\,(M\,(\eta_{A'}\,(z))) : B'[z/x]$. Then, by (Abs), we have $\Gamma \vdash \lambda z.Mz = \lambda z.\eta_{B'[\eta_{A'}(z)/x]}\,(M\,(\eta_{A'}\,(z))) : \Pi x : A'.B'$. Finally, we can conclude by (Π-η) and (Tm-Eq-Trans) that $\Gamma \vdash M = \lambda z.\eta_{B'[\eta_{A'}(z)/x]}\,(M\,(\eta_{A'}\,(z))) : \Pi x : A'.B'$ holds. □

Lemma 59. *If $\Gamma \vdash M : A$ then $\Gamma \vdash M = M\,[\rho_\Gamma] : A$.*

Proof. By induction on the derivation of $\Gamma \vdash M : A$.

As usual the only case to be handled with care is (Var) by distinguishing whether the variable is changed by ρ_Γ – in which case the conclusion is obtained using lemma 25 – or not – in this case the conclusion is immediately obtained by (Tm-Eq-Refl). All the other cases can be handled by applying the induction hypothesis to the judgments obtained by inversion. □

Lemma 60. *If $\Gamma \vdash M : A$, $\Gamma \vdash N : A$ and $\eta_A(M)\,[\rho_\Gamma] \overset{\beta}{=} \eta_A(N)\,[\rho_\Gamma]$ then $\Gamma \vdash M = N : A$.*

Proof. Since by hypothesis we have $\Gamma \vdash M : A$, using lemmata 25 and 2 we also have $\Gamma \vdash \eta_A(M) : A$ and by lemmata 26 and 2 we have $\Gamma \vdash \eta_A(M)\,[\rho_\Gamma] : A$. Similarly, we can show that $\Gamma \vdash \eta_A(N)\,[\rho_\Gamma] : A$. By hypothesis, we also have $\eta_A(M)\,[\rho_\Gamma] \overset{\beta}{=} \eta_A(N)\,[\rho_\Gamma]$, which implies by lemma 10 that $\Gamma \vdash \eta_A(M)\,[\rho_\Gamma] = \eta_A(N)\,[\rho_\Gamma] : A$. Using lemmata 25, 26 and (Tm-Eq-Trans), we have $\Gamma \vdash M = \eta_A\,(M)\,[\rho_\Gamma] : A$ and $\Gamma \vdash N = \eta_A\,(N)\,[\rho_\Gamma] : A$. Finally, using (Tm-Eq-Trans) and (Tm-Eq-Refl), we can conclude that $\Gamma \vdash M = N : A$. □

Theorem 7. *If $\Gamma \vdash M : A$, $\Gamma \vdash N : A$ then: $\Gamma \vdash M = N : A$ iff $\eta_A(M)[\rho_\Gamma] \overset{\beta}{=} \eta_A(N)[\rho_\Gamma]$.*

Proof. This results directly from lemmata 24 and 27. □

Theorem 8. *The equality is decidable in* **LF**.

Proof. It has been shown in [CPT03] that the relation $\eta_A(M)[\rho_\Gamma] \overset{\beta}{=} \eta_A(N)[\rho_\Gamma]$ is decidable. It comes from the fact that typable terms are normalizable. Thus to decide equality of two terms M and N, it is enough to compute their normal forms by making successive β-reductions and check if they are the same. We can show that two β-convertible normalizable terms have the same normal form using theorem 1. □

C Proof of the equivalence between $\mathbf{Cwf_{LF}}$ and LF

C.1 Interpretation of LF into $\mathbf{Cwf_{LF}}$

Lemma 61 (Weakening). *Let Γ and Δ be pre-contexts, A and B be pre-types, M be a pre-term and x a fresh variable. Let $\mathcal{J}$ be either M or A. The expression $P_x(\Gamma, x : A, \Delta)$ is defined iff $[\![\Gamma, x : A, \Delta]\!]_{\mathbf{Cwf_{LF}}}$ and $[\![\Gamma, \Delta]\!]_{\mathbf{Cwf_{LF}}}$ are defined and in this case is a morphism from the former to the latter. If $[\![\mathcal{J}]\!]^{\Gamma,\Delta}_{\mathbf{Cwf_{LF}}}$ is defined then*

$$[\![\mathcal{J}]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}} \cong [\![\mathcal{J}]\!]^{\Gamma,\Delta}_{\mathbf{Cwf_{LF}}}[P_x(\Gamma, x : A, \Delta)]$$

Proof. The proof proceeds by induction on the lengths of the involved pre-terms, -types and -contexts.

Base case of the induction on Δ. Suppose that the argument of P is of the form $\Gamma, x : A$ ($\Delta \equiv \diamond$). Then $P_x(\Gamma, x : A) \equiv p_A\left([\![\Gamma, x : A]\!]_{\mathbf{Cwf_{LF}}}\right)$ is defined iff $[\![\Gamma, x : A]\!]_{\mathbf{Cwf_{LF}}}$ and $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$ are defined and in this case the relation

$$[\![\mathcal{J}]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}} \cong [\![\mathcal{J}]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}[P_x(\Gamma, x : A)]$$

is verified since we have (by induction on length of involved pre-terms and -types).

For types, we have

– $\mathcal{J} \equiv \mathrm{Star}$:

$$\begin{aligned}
[\![\mathrm{Star}]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}} &\overset{\mathrm{def}}{\cong} \mathrm{Star} \\
&\cong \mathrm{Star}[P_x(\Gamma, x : A)] \\
&\cong [\![\mathrm{Star}]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}[P_x(\Gamma, x : A)]
\end{aligned}$$

– $\mathcal{J} \equiv \mathrm{Elem}(M)$:

$$\begin{aligned}
[\![\mathrm{Elem}(M)]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}} &\overset{\mathrm{def}}{\cong} \mathrm{Elem}\left([\![M]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}\right) \\
&\overset{\mathrm{ind}}{\cong} \mathrm{Elem}\left([\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}[P_x(\Gamma, x : A)]\right) \\
&\cong \mathrm{Elem}\left([\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\right)[P_x(\Gamma, x : A)] \\
&\cong [\![\mathrm{Elem}(M)]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}[P_x(\Gamma, x : A)]
\end{aligned}$$

– $\mathcal{J} \equiv \Pi y : B.C$:

$$\begin{aligned}
[\![\Pi y : B.C]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}} &\overset{\text{def}}{\cong} \Pi\left([\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}, [\![C]\!]^{\Gamma,x:A,y:B}_{\mathbf{Cwf_{LF}}}\right) \\
&\overset{\text{ind}}{\cong} \Pi\left([\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\left[P_x\left(\Gamma, x : A\right)\right], [\![C]\!]^{\Gamma,y:B}_{\mathbf{Cwf_{LF}}}\left[P_x\left(\Gamma, x : A, y : B\right)\right]\right) \\
&\cong \Pi\left([\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}, [\![C]\!]^{\Gamma,y:B}_{\mathbf{Cwf_{LF}}}\right)\left[P_x\left(\Gamma, x : A\right)\right] \\
&\cong [\![\Pi y : B.C]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\left[P_x\left(\Gamma, x : A\right)\right]
\end{aligned}$$

The penultimate equality is verified because we have

$$\begin{aligned}
&\Pi\left([\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}, [\![C]\!]^{\Gamma,y:B}_{\mathbf{Cwf_{LF}}}\right)\left[P_x\left(\Gamma, x : A\right)\right] = \\
&\quad \Pi\left([\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\left[P_x\left(\Gamma, x : A\right)\right], [\![C]\!]^{\Gamma,y:B}_{\mathbf{Cwf_{LF}}}\left[\left\langle P_x\left(\Gamma, x : A\right) \circ p_{[\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}}, q_{[\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}}\right\rangle\right]\right)
\end{aligned}$$

and

$$\begin{aligned}
\left\langle P_x\left(\Gamma, x : A\right) \circ p_{[\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}}, q_{[\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}}\right\rangle &= \left\langle P_x\left(\Gamma, x : A\right) \circ p_{[\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}[P_x(\Gamma,x:A)]}, q_{[\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}[P_x(\Gamma,x:A)]}\right\rangle \\
&= \tilde{p}\left(P_x\left(\Gamma, x : A\right), [\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}\right) \\
&= P_x\left(\Gamma, x : A, y : B\right)
\end{aligned}$$

We won't mention such details in the following and will write $\overset{*}{\equiv}$ when they have been omitted to justify the equality.

and for terms

– $\mathcal{J} \equiv x$: $[\![x]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ is not defined
– $\mathcal{J} \equiv y$:

$$\begin{aligned}
[\![y]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}} &\overset{\text{def}}{\cong} [\![y]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\left[p_{[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}}\right] \\
&\cong [\![y]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\left[P_x\left(\Gamma, x : A\right)\right]
\end{aligned}$$

– $\mathcal{J} \equiv \lambda y_B.M$:

$$\begin{aligned}
[\![\lambda y_B.M]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}} &\overset{\text{def}}{\cong} \lambda_B\left([\![M]\!]^{\Gamma,x:A,y:B}_{\mathbf{Cwf_{LF}}}\right) \\
&\overset{\text{ind}}{\cong} \lambda_B\left([\![M]\!]^{\Gamma,y:B}_{\mathbf{Cwf_{LF}}}\left[P_x\left(\Gamma, x : A, y : B\right)\right]\right) \\
&\overset{*}{\equiv} \lambda_B\left([\![M]\!]^{\Gamma,y:B}_{\mathbf{Cwf_{LF}}}\right)\left[P_x\left(\Gamma, x : A\right)\right] \\
&\cong [\![\lambda y_B.M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\left[P_x\left(\Gamma, x : A\right)\right]
\end{aligned}$$

– $\mathcal{J} \equiv MN$:

$$\begin{aligned}
[\![MN]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}} &\overset{\text{def}}{\cong} \text{App}\left([\![M]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}, [\![N]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}\right) \\
&\cong \text{App}\left([\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\left[P_x\left(\Gamma, x : A\right)\right], [\![N]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\left[P_x\left(\Gamma, x : A\right)\right]\right) \\
&\cong \text{App}\left([\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}, [\![N]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\right)\left[P_x\left(\Gamma, x : A\right)\right] \\
&\cong [\![MN]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\left[P_x\left(\Gamma, x : A\right)\right]
\end{aligned}$$

Induction step of the induction on Δ. Suppose that the argument of P is of the form $\Gamma, x : A, \Delta, y : B$. Then $P_x\left(\Gamma, x : A, \Delta, y : B\right) \equiv p_A\left(\left[\!\left[\Gamma, x : A, \Delta, y : B\right]\!\right]_{\mathbf{Cwf_{LF}}}\right)$ is defined iff $\left[\!\left[\Gamma, x : A, \Delta, y : B\right]\!\right]_{\mathbf{Cwf_{LF}}}$ and $\left[\!\left[\Gamma, \Delta, y : B\right]\!\right]_{\mathbf{Cwf_{LF}}}$ are defined and in this case the relation

$$\left[\!\left[\mathcal{J}\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, x:A, \Delta, y:B} \cong \left(\left[\!\left[\mathcal{J}\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, \Delta, y:B}\right)\left[P_x\left(\Gamma, x : A, \Delta, y : B\right)\right]$$

is verified since we have (by induction on length of involved pre-terms and -types).

For types, we have

– $\mathcal{J} \equiv \mathrm{Star}$:

$$\begin{aligned}
\left[\!\left[\mathrm{Star}\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, x:A, \Delta, y:B} &\overset{\mathrm{def}}{\cong} \mathrm{Star} \\
&\cong \mathrm{Star}\left[P_x\left(\Gamma, x : A, \Delta, y : B\right)\right] \\
&\cong \left[\!\left[\mathrm{Star}\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, \Delta, y:B}\left[P_x\left(\Gamma, x : A, \Delta, y : B\right)\right]
\end{aligned}$$

– $\mathcal{J} \equiv \mathrm{Elem}\left(M\right)$:

$$\begin{aligned}
\left[\!\left[\mathrm{Elem}\left(M\right)\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, x:A, \Delta, y:B} &\overset{\mathrm{def}}{\cong} \mathrm{Elem}\left(\left[\!\left[M\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, x:A, \Delta, y:B}\right) \\
&\overset{\mathrm{ind}}{\cong} \mathrm{Elem}\left(\left[\!\left[M\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, \Delta, y:B}\left[P_x\left(\Gamma, x : A, \Delta, y : B\right)\right]\right) \\
&\cong \mathrm{Elem}\left(\left[\!\left[M\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, \Delta, y:B}\right)\left[P_x\left(\Gamma, x : A, \Delta, y : B\right)\right] \\
&\cong \left[\!\left[\mathrm{Elem}\left(M\right)\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, \Delta, y:B}\left[P_x\left(\Gamma, x : A, \Delta, y : B\right)\right]
\end{aligned}$$

– $\mathcal{J} \equiv \Pi z : C.D$:

$$\begin{aligned}
\left[\!\left[\Pi z : C.D\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, x:A, \Delta, y:B} &\overset{\mathrm{def}}{\cong} \Pi\left(\left[\!\left[C\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, x:A, \Delta, y:B}, \left[\!\left[D\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, x:A, \Delta, y:B, z:C}\right) \\
&\overset{\mathrm{ind}}{\cong} \Pi\left(\left[\!\left[C\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, \Delta, y:B}\left[P_x\left(\Gamma, x : A, \Delta, y : B\right)\right], \left[\!\left[D\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, \Delta, y:B, z:C}\left[P_x\left(\Gamma, x : A, \Delta, y : B, z : C\right)\right]\right) \\
&\overset{*}{\equiv} \Pi\left(\left[\!\left[C\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, \Delta, y:B}, \left[\!\left[D\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, \Delta, y:B, z:C}\right)\left[P_x\left(\Gamma, x : A, \Delta, y : B\right)\right] \\
&\cong \left[\!\left[\Pi z : C.D\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, \Delta, y:B}\left[P_x\left(\Gamma, x : A, \Delta, y : B\right)\right]
\end{aligned}$$

and for terms

– $\mathcal{J} \equiv y$:

$$\begin{aligned}
\left[\!\left[y\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, x:A, \Delta, y:B} &\overset{\mathrm{def}}{\cong} q_{\left[\!\left[B\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, x:A, \Delta}} \\
&\overset{\mathrm{ind}}{\cong} q_{\left(\left[\!\left[B\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, \Delta}\left[P_x\left(\Gamma, x:A, \Delta\right)\right]\right)} \\
&\overset{*}{\equiv} q_{\left[\!\left[B\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, \Delta}}\left[P_x\left(\Gamma, x : A, \Delta, y : B\right)\right] \\
&\cong \left[\!\left[y\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, \Delta, y:B}\left[P_x\left(\Gamma, x : A, \Delta, y : B\right)\right]
\end{aligned}$$

– $\mathcal{J} \equiv x$: $\left[\!\left[x\right]\!\right]_{\mathbf{Cwf_{LF}}}^{\Gamma, \Delta, y:B}$ is not defined

– $\mathcal{J} \equiv z$ with $z \in \mathrm{DV}(\Gamma) \cup \mathrm{DV}(\Delta)$:

$$\begin{aligned}
[\![z]\!]^{\Gamma,x:A,\Delta,y:B}_{\mathbf{Cwf_{LF}}} &\overset{\text{def}}{\cong} [\![z]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}} \left[p_{[\![B]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}}\right] \\
&\overset{\text{ind}}{\cong} [\![z]\!]^{\Gamma,\Delta}_{\mathbf{Cwf_{LF}}} \left[P_x(\Gamma, x:A, \Delta)\right] \left[p_{[\![B]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}}\right] \\
&\cong [\![z]\!]^{\Gamma,\Delta}_{\mathbf{Cwf_{LF}}} \left[P_x(\Gamma, x:A, \Delta) \circ p_{[\![B]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}}\right] \\
&\cong [\![z]\!]^{\Gamma,\Delta}_{\mathbf{Cwf_{LF}}} \left[p_{[\![B]\!]^{\Gamma,\Delta}_{\mathbf{Cwf_{LF}}}} \circ \left\langle P_x(\Gamma, x:A, \Delta) \circ p_{[\![B]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}}, q_{[\![B]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}}\right\rangle\right] \\
&\cong [\![z]\!]^{\Gamma,\Delta}_{\mathbf{Cwf_{LF}}} \left[p_{[\![B]\!]^{\Gamma,\Delta}_{\mathbf{Cwf_{LF}}}}\right] \left[\left\langle P_x(\Gamma, x:A, \Delta) \circ p_{[\![B]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}}, q_{[\![B]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}}\right\rangle\right] \\
&\cong [\![z]\!]^{\Gamma,\Delta,y:B}_{\mathbf{Cwf_{LF}}} \left[\left\langle P_x(\Gamma, x:A, \Delta) \circ p_{[\![B]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}}, q_{[\![B]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}}\right\rangle\right] \\
&\cong [\![z]\!]^{\Gamma,\Delta,y:B}_{\mathbf{Cwf_{LF}}} \left[\left\langle P_x(\Gamma, x:A, \Delta) \circ p_{[\![B]\!]^{\Gamma,\Delta}_{\mathbf{Cwf_{LF}}}[P_x(\Gamma,x:A,\Delta)]}, q_{[\![B]\!]^{\Gamma,\Delta}_{\mathbf{Cwf_{LF}}}[P_x(\Gamma,x:A,\Delta)]}\right\rangle\right] \\
&\cong [\![z]\!]^{\Gamma,\Delta,y:B}_{\mathbf{Cwf_{LF}}} \left[\tilde{p}\left(P_x(\Gamma, x:A, \Delta), [\![B]\!]^{\Gamma,\Delta}_{\mathbf{Cwf_{LF}}}\right)\right] \\
&\cong [\![z]\!]^{\Gamma,\Delta,y:B}_{\mathbf{Cwf_{LF}}} \left[P_x(\Gamma, x:A, \Delta, y:B)\right]
\end{aligned}$$

– $\mathcal{J} \equiv \lambda z_C.M$:

$$\begin{aligned}
[\![\lambda z_C.M]\!]^{\Gamma,x:A,\Delta,y:B}_{\mathbf{Cwf_{LF}}} &\overset{\text{def}}{\cong} \lambda_C\left([\![M]\!]^{\Gamma,x:A,\Delta,y:B,z:C}_{\mathbf{Cwf_{LF}}}\right) \\
&\overset{\text{ind}}{\cong} \lambda_C\left([\![M]\!]^{\Gamma,\Delta,y:B,z:C}_{\mathbf{Cwf_{LF}}}\left[P_x(\Gamma, x:A, \Delta, y:B)\right]\right) \\
&\overset{*}{\equiv} \lambda_C\left([\![M]\!]^{\Gamma,\Delta,y:B,z:C}_{\mathbf{Cwf_{LF}}}\right)\left[P_x(\Gamma, x:A, \Delta)\right] \\
&\cong [\![\lambda z_C.M]\!]^{\Gamma,\Delta,y:B}_{\mathbf{Cwf_{LF}}}\left[P_x(\Gamma, x:A, \Delta, y:B)\right]
\end{aligned}$$

– $\mathcal{J} \equiv MN$:

$$\begin{aligned}
[\![MN]\!]^{\Gamma,x:A,\Delta,y:B}_{\mathbf{Cwf_{LF}}} &\overset{\text{def}}{\cong} [\![M]\!]^{\Gamma,x:A,\Delta,y:B}_{\mathbf{Cwf_{LF}}} [\![N]\!]^{\Gamma,x:A,\Delta,y:B}_{\mathbf{Cwf_{LF}}} \\
&\overset{\text{ind}}{\cong} \left([\![M]\!]^{\Gamma,\Delta,y:B}_{\mathbf{Cwf_{LF}}}\left[P_x(\Gamma, x:A, \Delta, y:B)\right]\right)\left([\![N]\!]^{\Gamma,\Delta,y:B}_{\mathbf{Cwf_{LF}}}\left[P_x(\Gamma, x:A, \Delta, y:B)\right]\right) \\
&\cong \left([\![M]\!]^{\Gamma,\Delta,y:B}_{\mathbf{Cwf_{LF}}}\right)\left([\![N]\!]^{\Gamma,\Delta,y:B}_{\mathbf{Cwf_{LF}}}\right)\left[P_x(\Gamma, x:A, \Delta, y:B)\right] \\
&\cong [\![MN]\!]^{\Gamma,\Delta,y:B}_{\mathbf{Cwf_{LF}}}\left[P_x(\Gamma, x:A, \Delta, y:B)\right]
\end{aligned}$$

□

Lemma 62 (Substitution). *Let Γ and Δ be pre-contexts, A and B be pre-types, M and N be pre-terms and x be a fresh variable. Let $\mathcal{J}$ be either A or M and suppose that $[\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ is defined. The expression $U^M_x(\Gamma, x:A, \Delta)$ is defined iff $[\![\Gamma, \Delta[M/x]]\!]_{\mathbf{Cwf_{LF}}}$ and $[\![\Gamma, x:A, \Delta]\!]_{\mathbf{Cwf_{LF}}}$ are both defined and in this case is a morphism from the former to the latter. If $[\![\mathcal{J}]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}$ is defined then*

$$[\![\mathcal{J}[M/x]]\!]^{\Gamma,\Delta[M/x]}_{\mathbf{Cwf_{LF}}} \cong [\![\mathcal{J}]\!]^{\Gamma,x:A,\Delta}_{\mathbf{Cwf_{LF}}}\left[U^M_x(\Gamma, x:A, \Delta)\right]$$

Proof. The proof proceeds by induction on the lengths of the involved pre-terms, -types and -contexts as in the proof of lemma 29. □

Proposition 9 (Soundness). *The interpretation function enjoys the following soundness properties*

1. *if $\Gamma \vdash$ then $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}} : \mathrm{Ctxt}$ is derivable in $\mathbf{Cwf_{LF}}$;*

2. *if* $\Gamma \vdash A$ *then* $[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} : \mathrm{Type}\left([\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}\right)$ *is derivable in* $\mathbf{Cwf_{LF}}$*;*
3. *if* $M : \Gamma \vdash A$ *then* $[\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} : [\![\Gamma]\!]_{\mathbf{Cwf_{LF}}} \vdash [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ *is derivable in* $\mathbf{Cwf_{LF}}$*;*
4. *if* $\Gamma \vdash A = B$ *then* $[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} = [\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} : \mathrm{Type}\,(\Gamma)$ *is derivable in* $\mathbf{Cwf_{LF}}$*;*
5. *if* $M = N : \Gamma \vdash A$ *then* $[\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} = [\![N]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} : [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ *is derivable in* $\mathbf{Cwf_{LF}}$.

Proof. The proof is done by induction on the derivations.

- Contexts rules
  - $(\textsc{C-Emp})_{\mathbf{LF}}$: $[\![\diamond]\!]_{\mathbf{Cwf_{LF}}} \equiv \diamond$ is a context by rule $(\textsc{C-Emp})_{\mathbf{Cwf_{LF}}}$.
  - $(\textsc{C-Ext})_{\mathbf{LF}}$: if Γ is a context and A an element of $\mathrm{Type}\,(\Gamma)$ then by induction hypothesis $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$ is a context and $[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ is an element of $\mathrm{Type}\left([\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}\right)$ and therefore $[\![\Gamma, x : A]\!]_{\mathbf{Cwf_{LF}}} \equiv [\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}, [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ is a context by rule $(\textsc{C-Ext})_{\mathbf{Cwf_{LF}}}$.
- Types rules
  - $(\textsc{Star})_{\mathbf{LF}}$: if Γ is a context then by induction hypothesis $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$ is a context and therefore Star is an element of $\mathrm{Type}\left([\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}\right)$.
  - $(\textsc{Elem})_{\mathbf{LF}}$: if Γ is a context and $M : \Gamma \vdash \mathrm{Star}$ then by induction hypothesis $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$ is a context and $[\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} \vdash [\![\mathrm{Star}]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ which can be rewritten as $[\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} \vdash \mathrm{Star}$ since $[\![\mathrm{Star}]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} \equiv \mathrm{Star}$ and therefore $\mathrm{Elem}\left([\![M]\!]_{\mathbf{Cwf_{LF}}}\right)$ is an element of $\mathrm{Type}\left([\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}\right)$ by rule $(\textsc{Elem})$.
  - $(\textsc{Exp})_{\mathbf{LF}}$: if Γ is a context, A a type in the context Γ and B a type in the context $\Gamma, x : A$ then by induction hypothesis $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$ is a context, $[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ is a type in the context $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$ and $[\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}$ is a type in the context $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}, [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ and therefore $[\![\Pi x : A.B]\!] \equiv \Pi\left([\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}, [\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\right)$ is a type in the context Γ by rule $(\textsc{Exp})_{\mathbf{Cwf_{LF}}}$.
- Terms rules
  - $(\textsc{Var})_{\mathbf{LF}}$: if Γ is a context and A is a type in the context Γ then, by induction hypothesis, $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$ is a context and $[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ is a type in $[\![\Gamma]\!]$ and therefore $[\![x]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}} \equiv q_{[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}}$ is a term of type $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}, [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} \vdash [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\left[p_{[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}}\right]$ by rule $(\text{M-E-R})_{\mathbf{Cwf_{LF}}}$, which we can rewrite into $[\![\Gamma, x : A]\!]_{\mathbf{Cwf_{LF}}} \vdash [\![A]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}$ by lemma 29 since $p_{[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}} \equiv P_x\,(\Gamma, x : A)$.
  - $(\textsc{Var-Ext})_{\mathbf{LF}}$: if Γ is a context, A and B are types in the context Γ and x is a variable of type $\Gamma \vdash A$ then, by induction hypothesis, $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$ is a context, $[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ and $[\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ are types in the context $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$ and $[\![x]\!]_{\mathbf{Cwf_{LF}}}$ is a term of type $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}} \vdash [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ and therefore we have $[\![x]\!]^{\Gamma,y:B}_{\mathbf{Cwf_{LF}}} \equiv [\![x]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\left[p_{[\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}}\right]$ which is of type $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}, [\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} \vdash [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ by rule $(\text{T}\textsc{m}\text{-S})_{\mathbf{Cwf_{LF}}}$since $p_{[\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}}$ is a morphism of type $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}, [\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} \to [\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$ and $[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\left[p_{[\![B]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}}\right] \equiv [\![A]\!]^{\Gamma,y:B}_{\mathbf{Cwf_{LF}}}$. This type can be rewritten $[\![\Gamma, B]\!]_{\mathbf{Cwf_{LF}}} \vdash [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$.
  - $(\textsc{Abs})_{\mathbf{LF}}$: if Γ is a context, A is a type in the context Γ, B is a type in the context $(\Gamma, x : A)$ and M is a term of type $(\Gamma, x : A \vdash B)$ then by induction hypothesis $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$ is a context, $[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ is a type in the context $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$, $[\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}$ is a type in the context $[\![\Gamma, x : A]\!]_{\mathbf{Cwf_{LF}}} \equiv [\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}, [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ and therefore $[\![\lambda x_A.M]\!] \equiv \lambda_{[\![A]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}}$ is a term of type $\Pi\left([\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}, [\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}\right) \equiv [\![\Pi x : A.M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ by rule $(\textsc{Abs})_{\mathbf{Cwf_{LF}}}$.
  - $(\textsc{App})_{\mathbf{LF}}$: if Γ is a context, A a type in the context Γ, B a type in the context $\Gamma, x : A$, M a term of type $\Gamma \vdash \Pi x : A.B$ and N a term of type $\Gamma \vdash A$ then by induction hypothesis $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$ is a context, $[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ is a type in the context $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$, $[\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}$ is a type in the context $[\![\Gamma, x : A]\!]_{\mathbf{Cwf_{LF}}} \equiv [\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}, [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$, $[\![M]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}$ is a term of type $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}} \vdash \Pi\left([\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}, [\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}\right)$ and N is a term of type $[\![\Gamma]\!] \vdash [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$

and therefore $[\![MN]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} \equiv \mathrm{App}\left([\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}, [\![N]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\right)$ is a term of type $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}} \vdash [\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}\left[\left\langle \mathrm{id}_{[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}}, [\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\right\rangle\right] \equiv [\![\Gamma]\!]_{\mathbf{Cwf_{LF}}} \vdash [\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}\left[U^M_x(\Gamma, x:A)\right] \cong [\![\Gamma]\!]_{\mathbf{Cwf_{LF}}} \vdash [\![B\,[M/x]]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$.

- $(\Pi\text{-C})_{\mathbf{LF}}$: if Γ is a context, A is a type in the context Γ, B a type in the context $\Gamma, x:A$, M a term of type $\Gamma \vdash \Pi x:A.B$ and N a term of type $\Gamma \vdash A$ then by induction hypothesis $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$ is a context, $[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ is a type in the context $[\![\Gamma]\!]\,\mathbf{Cwf_{LF}}$, $[\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}$ is a type in the context $[\![\Gamma, x:A]\!]_{\mathbf{Cwf_{LF}}} \equiv [\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}, [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$, $[\![M]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}$ is a term of type $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}} \vdash \Pi\left([\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}, [\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}\right)$ and N is a term of type $[\![\Gamma]\!] \vdash [\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ and therefore $[\![(\lambda x_A.M)\,N]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} \equiv \mathrm{App}\left(\lambda\left([\![M]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}\right), [\![N]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\right) \overset{(\Pi\text{-C})}{=} [\![N]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}\left[U^M_x(\Gamma, x:A)\right] \cong [\![N\,[M/x]]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ by lemma 30.
- $(\Pi\text{-}\eta)_{\mathbf{LF}}$: if Γ is a context, A is a type in the context Γ, B is a type in the context $\Gamma, x:A$ and M is a term of type $\Gamma \vdash \Pi x:A.B$ then by induction hypothesis $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$ is a context, $[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ is a type in the context $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}}$, $[\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}$ is a type in the context $[\![\Gamma, x:A]\!]_{\mathbf{Cwf_{LF}}}$ and $[\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}$ is a term of type $[\![\Gamma]\!]_{\mathbf{Cwf_{LF}}} \vdash \Pi\left([\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}, [\![B]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}\right)$ and therefore

$$\begin{aligned}
[\![\lambda x_A.\,(Mx)]\!] &\equiv \lambda\left(\mathrm{App}\left([\![M]\!]^{\Gamma,x:A}_{\mathbf{Cwf_{LF}}}, [\![x]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\right)\right)\\
&\equiv \lambda\left(\mathrm{App}\left([\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}\left[p_{[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}}\right], q_{[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}}\right)\right)\\
&\overset{(\Pi\text{-}\eta)}{=} [\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}}
\end{aligned}$$

Moreover, the rules of gat are verified in both theories and are preserved on the nose by the interpretation. □

C.2 Interpretation of $\mathbf{Cwf_{LF}}$ into LF

Lemma 63 (Weakening). *Let Γ and Δ be pre-contexts, A and B be pre-types, M be a pre-term and x a fresh variable. Let $\mathcal{J}$ be either M or A. The expression $P_x\left([\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}}, \Delta\right)$ is defined iff $[\![\Gamma, A, \Delta]\!]_{\mathbf{LF}}$ and $[\![\Gamma, \Delta]\!]_{\mathbf{LF}}$ are defined and in this case is a morphism from the former to the latter. If $[\![\mathcal{J}]\!]^{\Gamma,\Delta}_{\mathbf{Cwf_{LF}}}$ is defined then*

$$[\![\mathcal{J}]\!]^{[\![\Gamma,A,\Delta]\!]}_{\mathbf{LF}} \cong [\![\mathcal{J}]\!]^{[\![\Gamma,\Delta]\!]_{\mathbf{LF}}}_{\mathbf{LF}}\left[P_x\left([\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}}, [\![\Delta]\!]_{\mathbf{LF}}\right)\right]$$

Proof. The proof proceeds by induction on the lengths of the involved pre-terms, -types and -contexts as in the proof of lemma 29. □

Lemma 64 (Substitution). *Let Γ and Δ be pre-contexts, A and B be pre-types, M and N be pre-terms and x be a fresh variable. Let $\mathcal{J}$ be either A or M and suppose that $[\![M]\!]^{\Gamma}_{\mathbf{LF}}$ is defined. The expression $U^M_x\left([\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}}, [\![\Delta]\!]_{\mathbf{LF}}\right)$ is defined iff $[\![\Gamma, \Delta\left[\left\langle \mathrm{id}_{[\![\Gamma]\!]_{\mathbf{LF}}}, M\right\rangle\right]]\!]_{\mathbf{LF}}$ and $[\![\Gamma, A, \Delta]\!]_{\mathbf{LF}}$ are both defined and in this case is a morphism from the former to the latter. If $[\![\mathcal{J}]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}}, [\![\Delta]\!]_{\mathbf{LF}}}_{\mathbf{LF}}$ is defined then*

$$[\![\mathcal{J}\left[\left\langle \mathrm{id}_{[\![\Gamma]\!]_{\mathbf{LF}}}, M\right\rangle\right]]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}, [\![\Delta\left[\left\langle \mathrm{id}_{[\![\Gamma]\!]_{\mathbf{LF}}}, M\right\rangle\right]]\!]_{\mathbf{LF}}}_{\mathbf{LF}} \cong [\![\mathcal{J}]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}}, [\![\Delta]\!]_{\mathbf{LF}}}_{\mathbf{LF}}\left[U^M_x\left([\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}}, [\![\Delta]\!]_{\mathbf{LF}}\right)\right]$$

Proof. The proof proceeds by induction on the lengths of the involved pre-terms, -types and -contexts as in the proof of lemma 31. □

Proposition 10 (Soundness). *The interpretation function enjoys the following soundness properties*

1. *if* $\Gamma : \mathrm{Ctxt}$ *then* $[\![\Gamma]\!]_{\mathbf{LF}}$ *is a context;*
2. *if* $A : \mathrm{Type}\,(\Gamma)$ *then* $[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}$ *is a type in the context* $[\![\Gamma]\!]$*;*
3. *if* $M : \Gamma \vdash A$ *then* $[\![M]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}$ *is a term of type* $[\![\Gamma]\!]_{\mathbf{LF}} \vdash [\![M]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}} : [\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}$*;*
4. *if* $\Gamma \vdash A = B$ *then* $[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}} = [\![B]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}} : \mathrm{Type}\,([\![\Gamma]\!]_{\mathbf{LF}})$*;*
5. *if* $M = N : \Gamma \vdash A$ *then* $[\![M]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}} = [\![N]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}} : [\![\Gamma]\!]_{\mathbf{LF}} \vdash [\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}$*;*
6. *if* $\gamma : \Delta \to \Gamma$ *then* $[\![\gamma]\!]_{\mathbf{LF}}^{[\![\Delta]\!]_{\mathbf{LF}}} : [\![\Delta]\!]_{\mathbf{LF}} \to [\![\Gamma]\!]_{\mathbf{LF}}$*;*
7. *if* $\gamma = \delta : \Delta \to \Gamma$ *then* $[\![\gamma]\!]_{\mathbf{LF}}^{[\![\Delta]\!]_{\mathbf{LF}}} = [\![\delta]\!]_{\mathbf{LF}}^{[\![\Delta]\!]_{\mathbf{LF}}} : [\![\Delta]\!]_{\mathbf{LF}} \to [\![\Gamma]\!]_{\mathbf{LF}}$*.*

The last two properties are not strictly required for the proof of the equivalence between **LF** and $\mathbf{Cwf}_{\mathbf{LF}}$ but are required to prove the other properties.

Proof. By induction on the derivation rules.

- Rules for category: the interpretation is clearly compatible with the rules (M-Assoc), (M-Id-L) and (M-Id-R) by definition of context morphisms (def. 24).
- Rules for the functor T: the interpretation is clearly compatible with the rules (Ty-I), (Ty-Abs) thanks to the corresponding rules in **LF**
- ...
- Rules for Π-types
  - the interpretation is clearly compatible with the rules (Exp) and (Abs) thanks to the corresponding rules in **LF**
  - (Π-C): if Γ is a context, A a type in the context Γ, B a type in the context Γ, A, N a term of type $\Gamma \vdash \Pi(A, B)$ and M a term of type $\Gamma, A \vdash B$ then, by induction hypothesis, $[\![\Gamma]\!]_{\mathbf{LF}}$ is a context, $[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}$ is a type in the context $[\![\Gamma]\!]_{\mathbf{LF}}$, and, supposing that $[\![\Gamma, A]\!]_{\mathbf{LF}} = [\![\Gamma]\!]_{\mathbf{LF}}, [\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}$, $[\![B]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}}$ is a type in the context $[\![\Gamma]\!]_{\mathbf{LF}}, x : [\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}$, $[\![N]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}$ is a term of type $[\![\Gamma]\!]_{\mathbf{LF}} \vdash [\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}$ and $[\![M]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}}$ is a term of type $[\![\Gamma]\!]_{\mathbf{LF}}, x : [\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}} \vdash [\![B]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}}$ and therefore, by rule $(\Pi\text{-C})_{\mathbf{LF}}$ we have $\left(\lambda x_{[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}} . [\![M]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}}\right) [\![N]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}} = [\![M]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}} \left[[\![N]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}} / x\right] : [\![\Gamma]\!]_{\mathbf{LF}} \vdash [\![B]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}} \left[[\![N]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}} / x\right]$. Thus we have

$$\begin{aligned}
[\![\mathrm{App}\,(\lambda_A\,(M)\,, N)]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}} &\cong \left(\lambda x_{[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}} . [\![M]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}}\right) [\![N]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}} \\
&= [\![M]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}} \left[[\![N]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}} / x\right] \\
&\cong [\![M]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}} \left[\left\langle \mathrm{id}_{[\![\Gamma]\!]_{\mathbf{LF}}}, [\![N]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}} \right\rangle\right] \\
&\equiv [\![M]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}} \left[\overline{U}_x^{[\![N]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}} ([\![\Gamma]\!]_{\mathbf{LF}})\right] \\
&\cong \left[\!\left[M \left[\left\langle \mathrm{id}_{[\![\Gamma]\!]_{\mathbf{LF}}}, [\![N]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}} \right\rangle\right]\right]\!\right]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}
\end{aligned}$$

  in the type $[\![\Gamma]\!]_{\mathbf{LF}} \vdash [\![B]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}, x:[\![A]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}}} \left[[\![N]\!]_{\mathbf{LF}}^{[\![\Gamma]\!]_{\mathbf{LF}}} / x\right]$.
  - ...
- ... (the other rules can be handled similarly).

□

Proposition 11. *The interpretations inverse of one another modulo equality*

1. *if* $\Gamma \vdash_{\mathbf{LF}} A$ *then* $\left[\!\left[[\![M]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} \right]\!\right]^{\Gamma}_{\mathbf{LF}} = M : \Gamma \vdash A$;
2. *if* $A : \mathrm{Type}_{\mathbf{LF}}(\Gamma)$ *then* $\left[\!\left[[\![A]\!]^{\Gamma}_{\mathbf{Cwf_{LF}}} \right]\!\right]^{\Gamma}_{\mathbf{LF}} = A : \mathrm{Type}(\Gamma)$;
3. *if* $\Gamma \vdash_{\mathbf{Cwf_{LF}}} A$ *then* $\left[\!\left[[\![M]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}} \right]\!\right]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{Cwf_{LF}}} = M : \Gamma \vdash A$;
4. *if* $A : \mathrm{Type}_{\mathbf{Cwf_{LF}}}(\Gamma)$ *then* $\left[\!\left[[\![A]\!]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{LF}} \right]\!\right]^{[\![\Gamma]\!]_{\mathbf{LF}}}_{\mathbf{Cwf_{LF}}} = A : \mathrm{Type}(\Gamma)$.

Proof. By induction on terms, types and contexts. □

References

[Abb03] M. G. Abbott. *Categories of Containers*. PhD thesis, University of Leicester, August 2003. http://www.mcs.le.ac.uk/~mabbott/docs/thesis.ps.

[Awo03] S. Awodey. *Categories for Everybody*. Draft version, 2003. http://www.andrew.cmu.edu/course/80-413-713/notes/draft/catbook.ps.

[Car86] J. Cartmell. Generalised Algebraic Theories and Contextual Categories. *Annals of Pure and Applied Logic*, 32:209–243, 1986.

[CPT03] T. Coquand, R. Pollack, and M. Takeyama. A Logical Framework with Dependently Typed Records. In *Typed Lambda Calculus and Applications, TLCA'03*, volume 2701 of *LNCS*. Springer-Verlag, 2003. http://www.cs.nott.ac.uk/~gmh/appsem-slides/pollack.pdf.

[Cur86] P.-L. Curien. *Categorical Combinators*. PhD thesis, 1986.

[Dyb96] P. Dybjer. Internal Type Theory. *LNCS*, 1158:120–134, 1996. http://www.cs.chalmers.se/~peterd/papers/InternalTT.ps.

[Fau02] Germain Faure. Decidability of the Typed Equality in the Simply Typed λ-calculus with Subtyping. Technical report, Chalmers University of Computer Science, 2002. http://www.loria.fr/~faure/faure_files/report2002.ps.gz.

[Hof97] M. Hofmann. *Semantics of Logics of Computation*, chapter Syntax and Semantics of Dependent Types. P. Dybjer and A. Pitts, eds., Cambridge University Press, 1997. http://www.dcs.ed.ac.uk/home/mxh/cupart.dvi.gz.

[Hue86] G. Huet. *Formal Structures for Computation and Deduction*, chapter 7 – 8. Course notes, 1986. http://pauillac.inria.fr/~huet/PUBLIC/Formal_Structures.ps.gz.

[Jac92] B. Jacobs. Simply Typed and Untyped Lambda Calculus Revisited. In P.T. Johnstone M.P. Fourman and A.M. Pitts, editors, *Applications of Category Theory in Computer Science*, volume LMS 177, pages 119 – 142. Camb. Univ. Press, 1992. http://www.cs.kun.nl/~bart/PAPERS/Durham.ps.Z.

[LS86] J. Lambek and P. J. Scott. *Intruduction to Higher Order Categorical Logic*. Cambridge University Press, 1986.

[Pak02] Scott Pakin. The Comprehensive LaTeX Symbol List, octobre 2002. http://www.ctan.org/tex-archive/info/symbols/comprehensive/.

[Pit95] A. M. Pitts. *Handbook of Logic in Computer Science*, chapter Categorical Logic. Oxford University Press, 1995. http://www.cl.cam.ac.uk/~amp12/papers/catl/catl.ps.gz.

[Rit92] E. Ritter. *Categorical Abstract Machines for Higher-Order Typed λ-Calculi*. PhD thesis, University of Cambridge, 1992. ftp://ftp.cs.bham.ac.uk/pub/authors/E.Ritter/phd.ps.gz.

[Sai96] Amokrane Saibi. Théorie Constructive des Catégories. Draft, 1996. http://pauillac.inria.fr/~saibi/Cat_monographie.ps.

[San87] H. P. Sander. *Categorical Combinators*. PhD thesis, Departement of Computer Science, Chalmers University of Technology and University of Gteborg, may 1987.