# Extensional Normalization in the Logical Framework with Proof Irrelevant Equality

Andreas Abel
Department of Computer Science
Ludwig-Maximilians-University Munich

## Abstract

*We extend the Logical Framework by proof irrelevant equality types and present an algorithm that computes unique long normal forms. The algorithm is inspired by normalization-by-evaluation. Equality proofs which are not reflexivity are erased to a single object $*$. The algorithm decides judgmental equality, its completeness is established by a PER model.*

## 1. Introduction

In intensional Martin-Löf type theory (ITT), but also in the Calculus of Inductive Constructions which underlies the Coq proof assistant, we distinguish between *definitional equality* of terms $t$ and $t'$ of type $T$, given by the judgement $\Gamma \vdash t = t' : T$, and *propositional equality* which is established by providing an inhabitant of the identity set $\mathsf{Id}_T\, t\, t'$. While definitional equality is decidable and invoked during type checking, propositional equality is not. Inhabitants of $\mathsf{Id}_T\, t\, t'$ can be assumed and constructed by induction. However, there is at most one *canonical* inhabitant, the polymorphic constant $\mathsf{refl}$; if $t$ is definitionally equal to $t'$, then $\Gamma \vdash \mathsf{refl} : \mathsf{Id}_T\, t\, t'$ is canonical, otherwise, there are only non-canonical inhabitants.

In this respect, identity sets are similar to the unit set which has exactly one canonical inhabitant. For the unit set one easily proves that any two inhabitants are propositionally equal. The question whether this can be shown for the identity set as well has been answered negatively by Hofmann and Streicher [11]. The principle UIP (uniqueness of identity proofs) which states that any two proofs of propositional equality are propositionally equal themselves is refuted by interpreting each set $T$ of ITT as a groupoid, and each identity set $\mathsf{Id}_T\, t\, t'$ as a set of isomorphisms between the objects $t$ and $t'$ in the groupoid $T$. This model leaves room for more than one isomorphism per each pair of objects. Awodey and Warren [8] and Gambino and Garner [10] continue this program by looking at isomorphisms of isomorphisms and towers of identity sets leading to weak $\omega$-groupoids and homotopy theory.

While UIP is not provable and adding it as a constant might destroy canonicity,[1] we can add the principle of proof irrelevance for identity sets, as Altenkirch suggested [5].

$$\frac{\Gamma \vdash p, q : \mathsf{Id}_T\, t\, t'}{\Gamma \vdash p = q : \mathsf{Id}_T\, t\, t'}$$

Any two proofs of a propositional equality are *definitionally* equal. In the presence of this rule, UIP has a trivial proof. An instance of proof irrelevance (for $q = \mathsf{refl}$) is Martin-Löf's $\eta$-expansion for the identity type [12]:

$$\frac{\Gamma \vdash p : \mathsf{Id}_T\, t\, t}{\Gamma \vdash p = \mathsf{refl} : \mathsf{Id}_T\, t\, t}$$

Streicher's axiom $K$ [14] is the propositional variant of this principle, again, it has a trivial proof.

In this article, we are concerned with a decision procedure for definitional equality in the presence of proof irrelevant identity sets. We aim at computing unique $\eta$-long $\beta$-normal forms of well-typed terms which can then be compared syntactically. There are a few challenging issues:

1. Unlike in settings with proof-irrelevant propositions that are never eliminated [4, 7], we need to distinguish a $\mathsf{refl}$-proof from a merely hypothetical proof of a possibly false identity in order to resolve identity elimination.

$$\frac{\begin{array}{c}\Gamma \vdash S : \mathsf{Set} \qquad \Gamma \vdash s, s' : S \qquad \Gamma \vdash p : \mathsf{Id}_S\, s\, s' \\ \Gamma \vdash C : (x, x' : S) \to \mathsf{Id}_S\, x\, x' \to \mathsf{Set} \\ \Gamma \vdash h : (x : S) \to C\, x\, x\, \mathsf{refl}\end{array}}{\Gamma \vdash \mathsf{idElim}_{S,s,s'}\, p\, C\, h : C\, s\, s'\, p}$$

$$\mathsf{idElim}_{S,s,s}\, \mathsf{refl}\, C\, h = h\, s$$

2. An $\eta$-expansion at identity type can trigger a new $\beta$-reduction.[2]

$$\mathsf{idElim}_{A,a,a}\, p\, C\, h \longrightarrow_{\eta^-} \mathsf{idElim}_{A,a,a}\, \mathsf{refl}\, C\, h \longrightarrow_{\beta} h\, a$$

[1] Canonicity means that each closed term reduces to a canonical form, e.g., each closed term of natural number type reduces to a numeral

Hence, we cannot simply compute $\beta$-normal forms which we $\eta$-expand in a second phase.

Our solution is as follows: We introduce a symbol $*$ for irrelevant proofs which may be different from refl.

$$\frac{\Gamma \vdash p : \mathsf{Id}_T\, t\, t'}{\Gamma \vdash * : \mathsf{Id}_T\, t\, t'}$$

To obtain unique normal forms, we replace each proof $p : \mathsf{Id}_T\, t\, t'$ with refl if $t$ and $t'$ have the same normal form, and with $*$ otherwise. This step may not be done too early, in particular, bottom-up normalization (like hereditary substitutions [15]) does not work. For example, consider the following computation in context $X : \mathsf{Set}, x : X$.

$$\begin{aligned} &\mathsf{nf}((\lambda y{:}X.\lambda p{:}\mathsf{Id}_X\, x\, y.\, p)\, x) \\ &= \mathsf{nf}(\lambda y{:}X.\lambda p{:}\mathsf{Id}_X\, x\, y.\, p)\, \mathsf{nf}(x) \\ &= (\lambda y{:}X.\lambda p{:}\mathsf{Id}_X\, x\, y.\, *)\, x \\ &= \lambda p{:}\mathsf{Id}_X\, x\, x.\, * \end{aligned}$$

We have checked for identity of $x$ and $y$ too early. A top-down normalization function might work, but it is not compositional, thus, a proof of normalization could be hard.

In this article, we consider a strategy that is inspired by normalization-by-evaluation. It is type-directed; a term of function type is normalized by applying it to a fresh, $\eta$-expanded variable. At base type, it is sufficient to compute the $\beta$-normal form. However, here we have the same issue that $\eta$-expansion may not be executed eagerly, otherwise we will produce $*$s where we could have refls. For example, a variable $f$ of type $(x{:}X) \to (y{:}X) \to \mathsf{Id}_X\, x\, y$ will eagerly expand to $\lambda x \lambda y.\, *$, missing that $f\, x\, x = \mathsf{refl}$. We therefore treat $\eta$-expansion of variables lazily, we introduce markers $\mathsf{Up}^T t$ which remember that $t$ needs to be $\eta$-expanded.

For the moment, we can show completeness of our normalization procedure only for the logical framework [13] with identity sets (see the rules in Sec. 2). In particular, we did not succeed in adding function sets. However, adding a natural number set and other base types imposes no problem.

*Overview.* In Sec. 2 we recapitulate syntax and inference rules of the logical framework with proof irrelevant identity sets. In Sec. 3 we define unique normal forms and present a partial normalization function whose soundness is immediate. For completeness, we construct a PER model in Sec. 4. For two semantic objects, we show that if they are related in the PER of their type, then they have the same normal form. The opposite direction could only be shown for objects of base type. Termination and completeness of the normalization function follows from the usual fundamental theorem.

[2]Note that this is unlike the case for function types where *unrestricted* $\eta$ expansion can create a $\beta$-redex whose reduction leads back to the original term but makes no progress.

$$t^{A\to B}\, x \longrightarrow_{\eta^-} (\lambda y.\, t\, y)\, x \longrightarrow_{\beta} t\, x$$

## 2. Syntax

We consider Martin-Löf's logical framework with Id as the only set former. While it is unproblematic to add a set of natural numbers, we do not know how to extend the present word to function sets or $\Sigma$-sets yet.

Raw expressions and typing contexts are given by the following grammar, where we use uppercase letter to indicate types and lowercase letters to indicate terms.

$$\begin{aligned} \mathsf{Exp} \ni p, q, r, s, t ::= {} & x \mid \lambda x t \mid r\, s \mid (x{:}S) \to T \\ & \mid \mathsf{Set} \mid \mathsf{Type} \\ & \mid \mathsf{refl} \mid \mathsf{idElim}_{S,s,s'}\, p\, T\, t \mid \mathsf{Id}_T\, t\, t' \\ \mathsf{Cxt} \ni \Gamma \quad ::= {} & \diamond \mid \Gamma, x{:}T \end{aligned}$$

Note that we use refl polymorphically, as opposed to adding a monomorphic constant $\mathsf{refl} : (T : \mathsf{Set}) \to (t : T) \to \mathsf{Id}_T\, t\, t$ to the logical framework. Typing and equality is specified via the following five judgements, which are introduced by the rules in Figure 1. Note that we have omitted all congruence rules for equality.

| | |
|---|---|
| $\Gamma \vdash$ | $\Gamma$ is a well-formed context |
| $\Gamma \vdash T$ | $T$ is a well-formed type in $\Gamma$ |
| $\Gamma \vdash t : T$ | $t$ has type $T$ in $\Gamma$ |
| $\Gamma \vdash T = T'$ | $T$ and $T'$ are equal types in $\Gamma$ |
| $\Gamma \vdash t = t' : T$ | $t$ and $t'$ are equal terms of type $T$ in $\Gamma$ |

We write $\Gamma \vdash J$ to mean one of these judgements. These judgements enjoy the usual properties such as weakening, context conversion, substitution, and inversion of typing.

We extend Exp to $\mathsf{Exp}^*$ by adding the constructors $*$ and $\mathsf{Up}^T t$. Normalization will erase all hypothetical identity proofs to $*$, i.e., all proofs that do not normalize to refl but to a term with a variable in head position. In order to define well-typed normal forms, we introduce another set of judgements $\vdash^*$ which are clones of the $\vdash$ judgements, with the additional rule

$$\frac{\Gamma \vdash^* p : \mathsf{Id}_T\, t\, t'}{\Gamma \vdash^* * : \mathsf{Id}_T\, t\, t'}.$$

Our normalization procedure will use marked terms $\mathsf{Up}^T t$ which express that term $t$ is to be eta-expanded at type $T$. These markers have only administrative purpose, so we add the rules

$$\frac{\Gamma \vdash^* t : T}{\Gamma \vdash^* \mathsf{Up}^T t : T} \qquad \frac{\Gamma \vdash^* t : T}{\Gamma \vdash^* \mathsf{Up}^T t = t : T}.$$

**Theorem 1 (Conservativity)** *If* $\Gamma \vdash^* J$ *and* $\Gamma, J$ *do not mention* $*$ *and* Up*, then* $\Gamma \vdash J$.

Well-formed contexts.

$$\frac{}{\diamond \vdash} \qquad \frac{\Gamma \vdash \qquad \Gamma \vdash T : \mathsf{Type}}{\Gamma, x{:}T \vdash}$$

Dependently-typed $\lambda$-calculus.

$$\frac{\Gamma \vdash \qquad (x{:}T) \in \Gamma}{\Gamma \vdash x : T} \qquad \frac{\Gamma \vdash t : T \qquad \Gamma \vdash T = T' : \mathsf{Type}}{\Gamma \vdash t : T'}$$

$$\frac{\Gamma \vdash S : \mathsf{Type} \qquad \Gamma, x{:}S \vdash T : \mathsf{Type}}{\Gamma \vdash (x{:}S) \to T : \mathsf{Type}} \qquad \frac{\Gamma, x{:}S \vdash t : T}{\Gamma \vdash \lambda x t : (x{:}S) \to T} \qquad \frac{\Gamma \vdash r : (x{:}S) \to T \qquad \Gamma \vdash s : S}{\Gamma \vdash r\, s : T[s/x]}$$

Equality axioms.

$$\frac{\Gamma, x{:}S \vdash t : T \qquad \Gamma \vdash s : S}{\Gamma \vdash (\lambda x t)\, s = t[s/x] : T[s/x]} \qquad \frac{\Gamma \vdash t : (x{:}S) \to T}{\Gamma \vdash t = \lambda x.\, t\, x : (x{:}S) \to T}\ x \notin \mathsf{FV}(t)$$

Sets.

$$\frac{\Gamma \vdash}{\Gamma \vdash \mathsf{Set} : \mathsf{Type}} \qquad \frac{\Gamma \vdash T : \mathsf{Set}}{\Gamma \vdash T : \mathsf{Type}}$$

Identity set.

$$\frac{\Gamma \vdash T : \mathsf{Set} \qquad \Gamma \vdash t : T \qquad \Gamma \vdash t' : T}{\Gamma \vdash \mathsf{Id}_T\, t\, t' : \mathsf{Set}} \qquad \frac{\Gamma \vdash T : \mathsf{Set} \qquad \Gamma \vdash t : T}{\Gamma \vdash \mathsf{refl} : \mathsf{Id}_T\, t\, t}$$

$$\frac{\Gamma \vdash S : \mathsf{Set} \qquad \Gamma \vdash s, s' : S \qquad \Gamma \vdash p : \mathsf{Id}_S\, s\, s' \qquad \Gamma \vdash C : (x, x'{:}S) \to \mathsf{Id}_S\, x\, x' \to \mathsf{Set} \qquad \Gamma \vdash h : (x{:}S) \to C\, x\, x\, \mathsf{refl}}{\Gamma \vdash \mathsf{idElim}_{S,s,s'}\, p\, C\, h : C\, s\, s'\, p}$$

Equality axioms.

$$\frac{\Gamma \vdash S : \mathsf{Set} \qquad \Gamma \vdash s : S \qquad \Gamma \vdash C : (x, x'{:}S) \to \mathsf{Id}_S\, x\, x' \to \mathsf{Set} \qquad \Gamma \vdash h : (x{:}S) \to C\, x\, x\, \mathsf{refl}}{\Gamma \vdash \mathsf{idElim}_{S,s,s}\, \mathsf{refl}\, C\, h = h\, s : C\, s\, s\, \mathsf{refl}}$$

$$\frac{\Gamma \vdash p, q : \mathsf{Id}_T\, t\, t'}{\Gamma \vdash p = q : \mathsf{Id}_T\, t\, t'}$$

**Figure 1. Inference rules**

*Proof.* As in [4]: Replace in the derivation of $\Gamma \vdash^* J$ each occurrence of $*$, which has been introduced by forgetting proof $p$, by just this $p$. Also, replace every $\mathsf{Up}^T t$ by $t$. Remove or replace the affected inference steps. □

## 3. Normalization

In this section, we exhibit a type-directed algorithm $\mathsf{norm}^{\Gamma \vdash T} t$ which, given a well-typed term $\Gamma \vdash t : T$, computes the unique long normal form of $t$ where all identity proofs have been replaced by either $\mathsf{refl}$ or $*$. The purpose of the algorithm is to decide judgmental equality, in particular we ask for the following properties. Let $\Gamma \vdash t, t' : T$.

1. Soundness: If $\mathsf{norm}^{\Gamma \vdash T} t = \mathsf{norm}^{\Gamma \vdash T} t'$ then $\Gamma \vdash t = t' : T$.
2. Completeness: If $\Gamma \vdash t = t' : T$ then $\mathsf{norm}^{\Gamma \vdash T} t = \mathsf{norm}^{\Gamma \vdash T} t'$.
3. Termination: $\mathsf{norm}^{\Gamma \vdash T} t$ is defined.

We will define normalization on terms, hence, the algorithm will be sound by construction. Our soundness theorem is no more than subject reduction. Completeness and termination will follow from the construction of a PER model in Section 4.

### 3.1. Normal forms

It is illustrative to first present a grammar of normal forms, to see where the algorithm is supposed to take us. We distinguish long normal forms $v$, identified by a judgement $\Gamma \vdash v \Uparrow T$, from neutral normal forms $u$, identified by $\Gamma \vdash u \Downarrow T$. Neutral normal forms are terms whose head is a variable or $*$ and whose spine consists of long normal forms. Only certain expressions are candidates for long normal forms. In any case, they need to be $\beta$-normal; in particular, they must be generated by the following grammar.

$$\begin{aligned} u, U &::= x \mid u\, v \mid \mathsf{idElim}_{V,v,v'} * C\, h \\ v, V &::= u \mid \lambda x v \mid \mathsf{refl} \mid * \\ &\quad \mid (x{:}V) \to V' \mid \mathsf{Id}_V\, v\, v' \mid \mathsf{Set} \end{aligned}$$

*Neutral normal forms* $\Gamma \vdash^* u \Downarrow T$.

$$\frac{\Gamma \vdash^*}{\Gamma \vdash^* x \Downarrow \Gamma(x)}$$

$$\frac{\Gamma \vdash^* u \Downarrow (x{:}S) \to T \qquad \Gamma \vdash^* v \Uparrow S}{\Gamma \vdash^* u\, v \Downarrow T[v/x]}$$

$$\frac{\begin{array}{c}\Gamma \vdash^* V \Uparrow \mathsf{Set} \qquad \Gamma \vdash^* v, v' \Uparrow V \qquad v \not\equiv v' \\ \Gamma \vdash^* C \Uparrow (x, x'{:}V) \to \mathsf{Id}_V\, x\, x' \to \mathsf{Set} \\ \Gamma \vdash^* h \Uparrow (x{:}V) \to C\, x\, x\, \mathsf{refl}\end{array}}{\Gamma \vdash^* \mathsf{idElim}_{V,v,v'} * C\, h \Downarrow C\, v\, v' *}$$

$$\frac{\Gamma \vdash^* u \Downarrow T \qquad \Gamma \vdash^* T = T' : \mathsf{Type}}{\Gamma \vdash^* u \Downarrow T'}$$

Usually, one would consider $\mathsf{idElim}_{V,v,v'}\, u\, C\, h$ neutral if $u$ was. In our case, we only have two possible proofs $u$ of identity: $\mathsf{refl}$ and $*$. The first would constitute a $\beta$-redex, thus, only the second is possible, and it forces $v \not\equiv v'$.

*(Long) normal forms* (lnf) $\Gamma \vdash^* v \Uparrow T$.

$$\frac{\Gamma \vdash^* U \Downarrow \mathsf{Set} \qquad \Gamma \vdash^* u \Downarrow U}{\Gamma \vdash^* u \Uparrow U} \qquad \frac{\Gamma \vdash^* t : T}{\Gamma \vdash^* \mathsf{refl} \Uparrow \mathsf{Id}_T\, t\, t}$$

$$\frac{\Gamma \vdash^* v, v' \Uparrow T \qquad v \not\equiv v' \qquad \Gamma \vdash^* p : \mathsf{Id}_T\, v\, v'}{\Gamma \vdash^* * \Uparrow \mathsf{Id}_T\, v\, v'}$$

$$\frac{\Gamma, x{:}S \vdash^* v \Uparrow T}{\Gamma \vdash^* \lambda x v \Uparrow (x{:}S) \to T}$$

$$\frac{\Gamma \vdash^* v \Uparrow T \qquad \Gamma \vdash^* T = T' : \mathsf{Type}}{\Gamma \vdash^* v \Uparrow T'}$$

Neutral nfs of base type $U$ are lnfs, yet in our case, we exclude the identity type $\mathsf{Id}_T\, t\, t'$ as base type, thus, only neutral types $\Gamma \vdash^* U \Downarrow \mathsf{Set}$ remain. The two possible lnfs of identity type have their own introduction rules: $*$ is a lnf of identity type $\mathsf{Id}_T\, v\, v'$ if $v$ and $v'$ are different lnfs and there the identity type is inhabited by some proof $p$.

*Normal types* $\Gamma \vdash^* V \Uparrow \mathsf{Set}/\mathsf{Type}$.

$$\frac{\Gamma \vdash^* U \Downarrow \mathsf{Set}}{\Gamma \vdash^* U \Uparrow \mathsf{Set}} \qquad \frac{\Gamma \vdash^* V \Uparrow \mathsf{Set} \qquad \Gamma \vdash^* v, v' \Uparrow V}{\Gamma \vdash^* \mathsf{Id}_V\, v\, v' \Uparrow \mathsf{Set}}$$

$$\frac{\Gamma \vdash^* V \Uparrow \mathsf{Set}}{\Gamma \vdash^* V \Uparrow \mathsf{Type}} \qquad \frac{\Gamma \vdash^*}{\Gamma \vdash^* \mathsf{Set} \Uparrow \mathsf{Type}}$$

$$\frac{\Gamma \vdash^* V \Uparrow \mathsf{Type} \qquad \Gamma, x{:}V \vdash^* V' \Uparrow \mathsf{Type}}{\Gamma \vdash^* (x{:}V) \to V' \Uparrow \mathsf{Type}}$$

The three judgements entail well-typedness and are preserved under weakening with well-formed contexts. Note that these judgements do not immediately give us an method to check whether a term is normal, since they rely on the type equality judgement.

**Remark 2** *Note that while we can derive* $\Gamma \vdash x \Downarrow \mathsf{Id}_T\, t\, t'$ *for a suitable* $\Gamma$*, we do not get* $\Gamma \vdash x \Uparrow \mathsf{Id}_T\, t\, t'$*, since* $\mathsf{Id}_T\, t\, t'$ *is not a neutral type.*

### 3.2. Normalization Algorithm

We denote parallel substitution by $t\sigma$, where $\sigma$ is a map from variables to expressions such that $\sigma(x) \neq x$ for only finitely many $x$. Update of $\sigma$ at $x$ by $t$ is written $(\sigma, x = t)$.

A weak head normal form (whnf) $w$ is an expression without $\beta$-redex in weak head position. In our case, this means concretely:

$$\begin{array}{rcl} w & ::= & u \mid \mathsf{Up}^{(x:S)\to T} u \mid \lambda x t \mid (x:S) \to T \\ & \mid & \mathsf{refl} \mid * \mid \mathsf{Id}_T\, t\, t' \mid \mathsf{Set} \end{array}$$

The algorithm is given by the entry point function $\mathsf{norm}^{\Gamma \vdash T}\, t$ and six mutual recursive partial functions:

| | |
|---|---|
| $\mathsf{nf}^W w$ | compute lnf of whnf $w$, directed by type $W$ |
| $\mathsf{NF}\, W$ | compute lnf of type $W$ |
| $\uparrow^W u$ | $\eta$-expand neutral nf $u$ at type $W$ |
| $(\!\|t\|\!)$ | compute whnf of term $t$ |
| $w @ s$ | compute whnf of application $w\, s$ |
| $\mathsf{idE}\ \ldots$ | compute whnf of idElim |

*The top-level normalization function* replaces free variables $x : S$ by their reflection $\uparrow^S x$ and calls nf on the whnfs.

$$\mathsf{norm}^{\Gamma \vdash T}\, t = \mathsf{nf}^{(\!|T\sigma|\!)}\, (\!|t\sigma|\!) \text{ where } \sigma(x) = \uparrow^{(\!|\Gamma(x)|\!)} x.$$

*Type-directed normalization* $\eta$-expands its argument, a whnf, at function type, using *active application* to a reflected fresh variable $\uparrow^{\cdot} x$. Active application triggers further normalization. At identity type, whnfs are already nfs, and at neutral types $U$ we only encounter neutral nfs $u$.

$$\begin{array}{lcl} \mathsf{nf}^{(x:S)\to T} w & = & \lambda x.\, \mathsf{nf}^{(\!|T[s/x]|\!)}\, w @ s \\ & & \text{where } x \notin \mathsf{FV}(w) \text{ and } s = \uparrow^{(\!|S|\!)} x \\ \mathsf{nf}^{\mathsf{Set}} W & = & \mathsf{NF}\, W \\ \mathsf{nf}^{\mathsf{Type}} W & = & \mathsf{NF}\, W \\ \mathsf{nf}^{\mathsf{Id}_S\, s\, s'} w & = & w \\ \mathsf{nf}^U u & = & u \end{array}$$

*Type normalization.* This is just nf for types.

$$\begin{array}{lcl} \mathsf{NF}\, \mathsf{Set} & = & \mathsf{Set} \\ \mathsf{NF}\, ((x:S) \to T) & = & (x : \mathsf{NF}\, (\!|S|\!)) \to \mathsf{NF}\, (\!|T[(\uparrow^{(\!|S|\!)} x)/x]|\!) \\ \mathsf{NF}\, (\mathsf{Id}_T\, t\, t') & = & \mathsf{Id}_{(\mathsf{NF}\, W)}\, (\mathsf{nf}^W (\!|t|\!))\, (\mathsf{nf}^W (\!|t'|\!)) \\ & & \text{where } W = (\!|T|\!) \\ \mathsf{NF}\, U & = & U \end{array}$$

*Eta-expansion (reflection)* acts on a neutral nf $u$ of type $W$. In case of an identity type $\mathsf{Id}_T\, t\, t'$, if $t$ and $t'$ have the same normal form, we can safely expand $u$ to refl, and if not, then to $*$. It is crucial that the point of evaluating $\uparrow^{\mathsf{Id}_T\, t\, t'} u$ we have the complete information about $t$ and $t'$. This is ensured by two means:

1. Free variables in any term involved in normalization are reflected, i. e., they can only appear inside marked terms $\mathsf{Up}^{(x:S)\to T} u$ or neutral nfs $u$ of neutral type.
2. Reflection at function types is delayed: $\uparrow^{(x:S)\to T} u$ produces a marked term $\mathsf{Up}^{(x:S)\to T} u$ which waits for an argument. Only after application, reflection continues (see below).

$$\begin{array}{lcl} \uparrow^{(x:S)\to T} u & = & \mathsf{Up}^{(x:S)\to T} u \\ \uparrow^{\mathsf{Id}_T\, t\, t'} u & = & \begin{cases} \mathsf{refl} & \text{if } \mathsf{nf}^{(\!|T|\!)} (\!|t|\!) \equiv \mathsf{nf}^{(\!|T|\!)} (\!|t'|\!) \\ * & \text{otherwise} \end{cases} \\ \uparrow^{\mathsf{Set}} U & = & U \\ \uparrow^U u & = & u \end{array}$$

*Weak head normalization* is standard: It eliminates all top level $\beta$-redexes.

$$\begin{array}{lcl} (\!|r\, s|\!) & = & (\!|r|\!) @ s \\ (\!|\mathsf{idElim}_{S,s,s'}\, p\, C\, h|\!) & = & \mathsf{idE}_{S,s,s'}\, (\!|p|\!)\, C\, h \\ (\!|w|\!) & = & w \end{array}$$

*Active application* eliminates functions, which are either $\lambda$-abstractions or delayed reflections $\mathsf{Up}^{(x:S)\to T} u$ at function type.

$$\begin{array}{lcl} (\lambda x t) @ s & = & (\!|t[s/x]|\!) \\ (\mathsf{Up}^{(x:S)\to T} u) @ s & = & \uparrow^{(\!|T[s/x]|\!)} (u\, (\mathsf{nf}^{(\!|S|\!)} (\!|s|\!))) \end{array}$$

*Identity proof elimination* of refl is a $\beta$-redex which is resolved by idE. Elimination of $*$ leads to a neutral normal form, which is subjected to $\eta$-expansion. This way, idElim can never stack up in neutrals.

$$\begin{array}{lcl} \mathsf{idE}_{S,s,s'}\, \mathsf{refl}\, C\, h & = & (\!|h\, s|\!) \\ \mathsf{idE}_{S,s,s'}\, *\, C\, h & = & \uparrow^{(\!|C\, s\, s'\, *|\!)} (\mathsf{idElim}_{V_S, v_s, v_{s'}}\, *\, V_C\, v_h) \\ \text{where} \quad W_S & = & (\!|S|\!) \\ V_S & = & \mathsf{NF}\, W_S \\ v_s & = & \mathsf{nf}^{W_S} (\!|s|\!) \\ v_{s'} & = & \mathsf{nf}^{W_S} (\!|s'|\!) \\ V_C & = & \mathsf{nf}^{(x,x':S)\to \mathsf{Id}_S\, x\, x' \to \mathsf{Set}} (\!|C|\!) \\ v_h & = & \mathsf{nf}^{(x:S)\to C\, x\, x\, \mathsf{refl}} (\!|h|\!) \end{array}$$

### 3.3. Soundness

Soundness of normalization follows simply by inspection of the equations. Formally, we prove:

**Theorem 3 (Subject reduction)** *Let* $\Gamma \vdash^* t : T$.

1. *If* $\mathsf{norm}^{\Gamma \vdash T}\, t = v$ *then* $\Gamma \vdash^* t = v : T$.
2. *If* $\mathsf{nf}^T t = v$ *then* $\Gamma \vdash^* t = v : T$.
3. *If* $\mathsf{NF}\, T = V$ *then* $\Gamma \vdash^* T = V : \mathsf{Type}$.
4. *If* $\uparrow^T t = w$ *then* $\Gamma \vdash^* t = w : T$.
5. *If* $(\!|t|\!) = w$ *then* $\Gamma \vdash^* t = w : T$.

6. *If* $\Gamma \vdash^* t\,s : S$ *and* $t@s = w$ *then* $\Gamma \vdash^* t\,s = w : S$.

7. *If* $\Gamma \vdash \mathsf{idElim}_{S,s,s'}\,p\,C\,h : T$ *and* $\mathsf{idE}_{S,s,s'}\,p\,C\,h = w$ *then* $\Gamma \vdash \mathsf{idElim}_{S,s,s'}\,p\,C\,h = w : T$.

*Proof.* Simultaneously, by induction on the evaluation trace of the functions. More precisely, we view the functions as mutually inductive relations, e.g., $(\!|\_|\!)$ as the binary relation $(\!|\_|\!) = \_$, which are given by inference rules, for instance,

$$\frac{(\!|T|\!) = W \qquad (\!|t|\!) = w \qquad (\!|t'|\!) = w' \qquad \mathsf{nf}^W w = v \qquad \mathsf{nf}^W w' = v}{\uparrow^{\mathsf{Id}_T\,t\,t'} u = \mathsf{refl}}$$

Then, we show the theorem by mutual induction on these relations. In case of the rule above, we have $\Gamma \vdash^* u : \mathsf{Id}_T\,t\,t'$ which entails $\Gamma \vdash^* T : \mathsf{Set}$ and $\Gamma \vdash^* t, t' : T$ by inversion. By induction hypotheses we get $\Gamma \vdash^* T = W : \mathsf{Set}$, $\Gamma \vdash^* t = w : T$, $\Gamma \vdash^* t' = w' : T$, $\Gamma \vdash^* w = v : W$ and $\Gamma \vdash^* w' = v : W$. Putting these equalities together, we get $\Gamma \vdash^* t = t' : T$, hence, $\Gamma \vdash^* \mathsf{refl} : \mathsf{Id}_T\,t\,t'$. By proof irrelevance, $\Gamma \vdash^* u = \mathsf{refl} : \mathsf{Id}_T\,t\,t'$. □

**Corollary 4 (Soundness)** *If* $\Gamma \vdash t, t' : T$ *and* $\mathsf{norm}^{\Gamma \vdash T}\,t = \mathsf{norm}^{\Gamma \vdash T}\,t'$ *then* $\Gamma \vdash t = t' : T$.

*Proof.* Certainly, $\Gamma \vdash^* t, t' : T$, hence, by subject reduction $\Gamma \vdash^* t = \mathsf{norm}^{\Gamma \vdash T}\,t : T$ and $\Gamma \vdash^* t' = \mathsf{norm}^{\Gamma \vdash T}\,t' : T$. It follows $\Gamma \vdash^* t = t' : T$ and, by conservativity, $\Gamma \vdash t = t' : T$. □

## 4. Model and Completeness

In this section we prove that normalization terminates and is complete for judgmental equality, i. e., if $\Gamma \vdash t = t' : T$ then $\mathsf{norm}^{\Gamma \vdash T}\,t$ and $\mathsf{norm}^{\Gamma \vdash T}\,t'$ are defined and $\alpha$-equivalent. To this end, we construct a partial equivalence relation (PER) $[W]$ for each type $W$ in whnf such that:

1. If $\Gamma \vdash t = t' : T$ and $\sigma(x) = \uparrow^{(\!|\Gamma(x)|\!)} x$ for all $x \in \mathsf{dom}(\Gamma)$ then $((\!|t\sigma|\!), (\!|t'\sigma|\!)) \in [(\!|T\sigma|\!)]$.

2. If $(w, w') \in [W]$ then $\mathsf{nf}^W w$ and $\mathsf{nf}^W w'$ are defined and equal.

We do not prove here that $\mathsf{nf}^W w$ is actually a long normal form $\Gamma \vdash \mathsf{nf}^W w \Uparrow W$; this result could be achieved by switching to a Kripke model where $\Gamma$ is available. Also, the PER model is unsound: every type is inhabited in the model. This does not bother us either; soundness can also be obtained by switching to a Kripke model.

### 4.1. PER Model

Dealing with a predicative type theory, we can construct a model from below. For convenience, we use induction-recursion [9]. Our construction is analogous to previous work [1, 2].

We write $w = w' \in W$ for $(w, w') \in [W]$ and extend this to all terms by letting $t = t' \in T$ be $(\!|t|\!) = (\!|t'|\!) \in (\!|T|\!)$, which more precisely means the conjunction of $(\!|t|\!) = w$, $(\!|t'|\!) = w'$, $(\!|T|\!) = W$, and $w = w' \in W$. Also $t \in T$ is short for $t = t \in T$.

First, we construct a PER $\_ = \_ \in \mathsf{Set}$ by induction, and simultaneously for each $W \in \mathsf{Set}$ its extension, a PER $[W]$, by recursion on $W \in \mathsf{Set}$. Then, we proceed with $\_ = \_ \in \mathsf{Type}$ and $[W]$ for $W \in \mathsf{Type}$. We will not repeat the proofs that the obtained relations are indeed PERs and that related sets or types have the same extension [1]. In our case, the construction of $\mathsf{Set}$ is trivial: there are only two types of sets, namely neutral sets and identity sets.

*Neutral sets.*

$$\frac{}{U = U \in \mathsf{Set}} \qquad [U] = \{(u, u)\}$$

*Identity sets.*

$$\frac{T = T' \in \mathsf{Set} \qquad t_1 = t_1' \in T \qquad t_2 = t_2' \in T}{\mathsf{Id}_T\,t_1\,t_2 = \mathsf{Id}_{T'}\,t_1'\,t_2' \in \mathsf{Set}}$$

$$\begin{aligned}[\mathsf{Id}_T\,t_1\,t_2] = \{&(\mathsf{refl}, \mathsf{refl}) \mid \mathsf{nf}^{(\!|T|\!)}\,(\!|t_1|\!) \equiv \mathsf{nf}^{(\!|T|\!)}\,(\!|t_2|\!)\} \\ \cup \{&(*, *) \quad\;\; \mid \mathsf{nf}^{(\!|T|\!)}\,(\!|t_1|\!) \not\equiv \mathsf{nf}^{(\!|T|\!)}\,(\!|t_2|\!)\}\end{aligned}$$

If the normal form of $t_1$ or $t_2$ is undefined, then $[\mathsf{Id}_T\,t_1\,t_2]$ is empty, according to the definition. However, this will never happen, since $t_1, t_2 \in T$ by definition. Thus, $[\mathsf{Id}_T\,t_1 t_2]$ will be the singleton $\{(\mathsf{refl}, \mathsf{refl})\}$ if $t_1$ and $t_2$ have the same normal form, and $\{(*, *)\}$ otherwise. Since $\alpha$-equivalence is decidable, our construction does not leave intuitionistic logic.

*Universe of sets.*

$$\frac{W = W' \in \mathsf{Set}}{W = W' \in \mathsf{Type}} \qquad \frac{}{\mathsf{Set} = \mathsf{Set} \in \mathsf{Type}}$$

No new PERs $[W]$ need to be constructed here, since $[\mathsf{Set}] = \{(W, W') \mid W = W' \in \mathsf{Set}\}$ already by our notational convention.

*Function types.* The standard construction.

$$\frac{S = S' \in \mathsf{Type} \qquad \forall s = s' \in S.\, T[s/x] = T[s'/x]' \in \mathsf{Type}}{(x\!:\!S) \to T = (x\!:\!S') \to T' \in \mathsf{Type}}$$

$$[(x\!:\!S) \to T] = \{(w, w') \mid \forall s = s' \in S.\ w@s = w'@s' \in T[s/x]\}$$

## 4.2. Escape Lemma

We would like to establish that $\mathsf{nf}$ decides model equality, i. e., if $w, w' \in W$ then $w = w' \in W$ if and only if $\mathsf{nf}^W w \equiv \mathsf{nf}^W w'$. We show it for sets $W$, but for function types we can only prove the direction "only if".

In the following, when we formulate a property for the PER $\mathsf{Type}$, we mean it also for $\mathsf{Set}$, and a proof by induction on $W = W' \in \mathsf{Type}$ shall mean that we first do a induction on $W = W' \in \mathsf{Set}$.

**Lemma 5 (Out of and into of the model)** *Let* $W = W' \in \mathsf{Type}$. *Then,*

1. $\mathsf{NF}\, W \equiv \mathsf{NF}\, W'$.
2. *If* $w = w' \in W$ *then* $\mathsf{nf}^W w \equiv \mathsf{nf}^{W'} w'$.
3. $\uparrow^W u = \uparrow^{W'} u \in W$.

*Proof.* By induction on $W = W' \in \mathsf{Type}$.

*Case*

$$\frac{}{U = U \in \mathsf{Set}}$$

1. Clear, since $\mathsf{NF}\, U = U$.
2. Assume $u = u \in U$. We have $\mathsf{nf}^U u = u$.
3. By definition, $\uparrow^U u = \uparrow^U u \in U$.

*Case*

$$\frac{T = T' \in \mathsf{Set} \qquad t_1 = t_1' \in T \qquad t_2 = t_2' \in T}{\mathsf{Id}_T\, t_1\, t_2 = \mathsf{Id}_{T'}\, t_1'\, t_2' \in \mathsf{Set}}$$

1. Clear, using the induction hypotheses.
2. *Case*
$$\frac{\mathsf{nf}^{(\!|T|\!)}\,(\!|t_1|\!) \equiv \mathsf{nf}^{(\!|T|\!)}\,(\!|t_2|\!)}{\mathsf{refl} = \mathsf{refl} \in \mathsf{Id}_T\, t_1\, t_2}$$
Trivial, since $\mathsf{nf}^{\mathsf{Id}_T\, t_1\, t_2}\mathsf{refl} = \mathsf{refl}$.
*Case*
$$\frac{\mathsf{nf}^{(\!|T|\!)}\,(\!|t_1|\!) \not\equiv \mathsf{nf}^{(\!|T|\!)}\,(\!|t_2|\!)}{* = * \in \mathsf{Id}_T\, t_1\, t_2}$$
Trivial, since $\mathsf{nf}^{\mathsf{Id}_T\, t_1\, t_2} * = *$.
3. Let $v_1 := \mathsf{nf}^{(\!|T|\!)}\,(\!|t_1|\!)$ and $v_2 := \mathsf{nf}^{(\!|T|\!)}\,(\!|t_2|\!)$ which are both defined by induction hypothesis. If $v_1 \equiv v_2$ then $\uparrow^{\mathsf{Id}_T\, t_1\, t_2} u = \mathsf{refl}$. Since by induction hypothesis also $\mathsf{nf}^{(\!|T'|\!)}\,(\!|t_1'|\!) = v_1 \equiv v_2 = \mathsf{nf}^{(\!|T'|\!)}\,(\!|t_2'|\!)$, we have $\uparrow^{\mathsf{Id}_{T'}\, t_1'\, t_2'} u = \mathsf{refl}$, thus $\uparrow^{\mathsf{Id}_T\, t_1\, t_2} u = \uparrow^{\mathsf{Id}_{T'}\, t_1'\, t_2'} u \in \mathsf{Id}_T\, t_1\, t_2$. If $v_1 \not\equiv v_2$, then $\uparrow^{\mathsf{Id}_T\, t_1\, t_2} u = * = \uparrow^{\mathsf{Id}_{T'}\, t_1'\, t_2'} u$, thus, also $\uparrow^{\mathsf{Id}_T\, t_1\, t_2} u = \uparrow^{\mathsf{Id}_{T'}\, t_1'\, t_2'} u \in \mathsf{Id}_T\, t_1\, t_2$.

*Case*

$$\frac{S = S' \in \mathsf{Type} \qquad \forall s = s' \in S.\, T[s/x] = T[s'/x]' \in \mathsf{Type}}{(x\!:\!S) \to T = (x\!:\!S') \to T' \in \mathsf{Type}}$$

1. Observe that $\mathsf{NF}\,((x\!:\!S) \to T) = (x\!:\!\mathsf{NF}(\!|S|\!)) \to \mathsf{NF}\,(\!|T[(\uparrow^{(\!|S|\!)} x)/x]|\!)$. By induction hypothesis, we have $\uparrow^{(\!|S|\!)} x = \uparrow^{(\!|S'|\!)} x \in (\!|S|\!)$. Thus, by induction hypothesis $\mathsf{NF}\,(\!|T[(\uparrow^{(\!|S|\!)} x)/x]|\!) \equiv \mathsf{NF}\,(\!|T'[(\uparrow^{(\!|S'|\!)} x)/x]|\!)$, and also $\mathsf{NF}\,(\!|S|\!) \equiv \mathsf{NF}\,(\!|S'|\!)$. Together, $\mathsf{NF}\,((x\!:\!S) \to T) \equiv \mathsf{NF}\,((x\!:\!S') \to T')$.
2. Assume $w = w' \in (x\!:\!S) \to T$. Observe that $\mathsf{nf}^{(xS)\to T} w = \lambda x.\, \mathsf{nf}^{(\!|T[(\uparrow^{(\!|S|\!)} x)/x]|\!)}\,(w@\uparrow^{(\!|S|\!)} x)$. Similar as in the last case, we have $w@\uparrow^{(\!|S|\!)} x = w'@\uparrow^{(\!|S'|\!)} x \in (\!|T[(\uparrow^{(\!|S|\!)} x)/x]|\!)$, from which $\mathsf{nf}^{(xS)\to T} w \equiv \mathsf{nf}^{(xS')\to T'} w'$ follows.
3. To show $\uparrow^{(xS)\to T} u = \uparrow^{(xS')\to T'} u \in (x : S) \to T$, assume arbitrary $s = s' \in S$ and show $(\uparrow^{(xS)\to T} u)@s = (\uparrow^{(xS')\to T'} u)@s' \in (\!|T[s/x]|\!)$. Observe that $(\uparrow^{(xS)\to T} u)@s = \uparrow^{(\!|T[s/x]|\!)}\,(u\,(\mathsf{nf}^{(\!|S|\!)}\,(\!|s|\!)))$. Let $v := \mathsf{nf}^{(\!|S|\!)}\,(\!|s|\!)$. By induction hypothesis, $v \equiv \mathsf{nf}^{(\!|S'|\!)} s'$ and $\uparrow^{(\!|T[s/x]|\!)}\,(u\,v) = \uparrow^{(\!|T'[s'/x]|\!)}\,(u\,v) \in (\!|T[s/x]|\!)$.

□

**Lemma 6 (Eta-expansion in the model)** *Let* $W \in \mathsf{Set}$ *and* $w \in W$.

1. $\mathsf{NF}\, W = W \in \mathsf{Set}$.
2. $\mathsf{nf}^W w = w \in W$.
3. $\uparrow^W w = w \in W$.

*Proof.* By induction on $W \in \mathsf{Set}$.

*Case*

$$\frac{}{U \in \mathsf{Set}}$$

1. We have $\mathsf{NF}\, U = U$, hence, $\mathsf{NF}\, U = U \in \mathsf{Set}$.
2. Since $\mathsf{nf}^U u = u$, we have $\mathsf{nf}^U u = u \in U$.

3. With $\uparrow^U u = u$ we immediately obtain $\uparrow^U u = u \in U$.

*Case*

$$\frac{(\!|T|\!) \in \mathsf{Set} \qquad (\!|t_1|\!) \in (\!|T|\!) \qquad (\!|t_2|\!) \in (\!|T|\!)}{\mathsf{Id}_T\, t_1\, t_2 \in \mathsf{Set}}$$

Let $W = (\!|T|\!)$, $w_1 = (\!|t_1|\!)$ and $w_2 = (\!|t_2|\!)$.

1. $\mathsf{NF}\,(\mathsf{Id}_T\, t_1\, t_2) = \mathsf{Id}_{\mathsf{NF}\,W}\,(\mathsf{nf}^W w_1)\,(\mathsf{nf}^W w_2)$ entails $\mathsf{NF}\,(\mathsf{Id}_T\, t_1\, t_2) = \mathsf{Id}_T\, t_1\, t_2 \in \mathsf{Set}$ by the induction hypotheses.
2. $\mathsf{nf}^{\mathsf{Id}_T\, t_1\, t_2} w = w$ by definition.
3. We distinguish two cases:

$$\frac{\mathsf{nf}^W w_1 \equiv \mathsf{nf}^W w_2}{\mathsf{refl} \in \mathsf{Id}_T\, t_1\, t_2}$$

Since $\uparrow^{\mathsf{Id}_T\, t_1\, t_2} w = \mathsf{refl}$ for any $w$, clearly $\uparrow^{\mathsf{Id}_T\, t_1\, t_2} \mathsf{refl} = \mathsf{refl} \in \mathsf{Id}_T\, t_1\, t_2$.

$$\frac{\mathsf{nf}^W w_1 \not\equiv \mathsf{nf}^W w_2}{* \in \mathsf{Id}_T\, t_1\, t_2}$$

Since $\uparrow^{\mathsf{Id}_T\, t_1\, t_2} w = *$ for any $w$, clearly $\uparrow^{\mathsf{Id}_T\, t_1\, t_2} * = * \in \mathsf{Id}_T\, t_1\, t_2$.

□

**Corollary 7** *If* $W \in \mathsf{Set}$, $w, w' \in W$, *and* $\mathsf{nf}^W w = \mathsf{nf}^W w'$, *then* $w = w' \in W$.

It is unclear how to extend Lemma 6 to function types. Hence, we could not prove it for types $W \in \mathsf{Type}$, and we can not extend Set by function sets.

### 4.3. Fundamental Theorem

In this section, we show that judgmental equality is modeled by the PERs, and as a consequence, is decided by normalization.

Let $\sigma = \sigma' \in \Gamma$ iff $\sigma(x) = \sigma'(x) \in \Gamma(x)$ for all $x \in \mathsf{dom}(\Gamma)$. We define $\Gamma \Vdash J$, meaning that $\Gamma \vdash J$ is valid in the PER model by induction on $\Gamma$:

$$\begin{array}{lcl}
\diamond \Vdash & :\Longleftrightarrow & \text{true} \\
\Gamma, x{:}T \Vdash & :\Longleftrightarrow & \Gamma \Vdash T : \mathsf{Type} \\
\Gamma \Vdash t : T & :\Longleftrightarrow & \Gamma \Vdash t = t : T \\
\Gamma \Vdash t = t' : T & :\Longleftrightarrow & \Gamma \Vdash T \text{ and} \\
 & & \forall \sigma = \sigma' \in \Gamma.\ (\!|t\sigma|\!) = (\!|t'\sigma'|\!) \in (\!|T\sigma|\!) \\
\Gamma \Vdash T & :\Longleftrightarrow & \text{either } T = \mathsf{Type} \text{ and } \Gamma \Vdash \\
 & & \text{or } \Gamma \Vdash T : \mathsf{Type}
\end{array}$$

The proof of the fundamental theorem requires Lemma 6 for the soundness of the typing of idElim.

**Theorem 8 (Fundamental theorem)** *If* $\Gamma \vdash J$ *then* $\Gamma \Vdash J$.

*Proof.* By induction on $\Gamma \vdash J$.

*Case*

$$\frac{\Gamma \vdash t : T}{\Gamma \vdash \mathsf{refl} : \mathsf{Id}_T\, t\, t}$$

Assume $\sigma = \sigma' \in \Gamma$. Let $w := (\!|t\sigma|\!)$ and $W := (\!|T\sigma|\!)$. By induction hypothesis, $w \in W$. Hence $v := \mathsf{nf}^W w$ is defined. Clearly, $\mathsf{refl} \in \mathsf{Id}_W\, w\, w$.

*Case*

$$\frac{\begin{array}{c}\Gamma \vdash S : \mathsf{Set} \qquad \Gamma \vdash s, s' : S \\ \Gamma \vdash p : \mathsf{Id}_S\, s\, s' \\ \Gamma \vdash C : (x, x' {:} S) \to \mathsf{Id}_S\, x\, x' \to \mathsf{Set} \\ \Gamma \vdash h : (x{:}S) \to C\, x\, x\, \mathsf{refl}\end{array}}{\Gamma \vdash \mathsf{idElim}_{S,s,s'}\, p\, C\, h : C\, s\, s'\, p}$$

Let $W = (\!|S\sigma|\!)$, $w = (\!|s\sigma|\!)$, and $w' = (\!|s'\sigma|\!)$. By induction hypothesis, $(\!|p\sigma|\!) = (\!|p\sigma'|\!) \in (\!|(\mathsf{Id}_S\, s\, s')\sigma|\!)$.

In case $\mathsf{nf}^W w \equiv \mathsf{nf}^W w'$ we have $(\!|p\sigma|\!) = (\!|p\sigma'|\!) = \mathsf{refl}$ and $(\!|(\mathsf{idElim}_{S,s,s'}\, p\, C\, h)\sigma|\!) = (\!|(h\, s)\sigma|\!)$. By Cor. 7 $w = w' \in W$, hence, $s\sigma = s'\sigma \in S\sigma$. Also by induction hypothesis, $C\sigma = C\sigma \in (x, x' : S\sigma) \to \mathsf{Id}_{S\sigma}\, x\, x' \to \mathsf{Set}$, which implies $(C\, s\, s'\, p)\sigma = (C\, s\, s\, \mathsf{refl})\sigma \in \mathsf{Set}$. By induction hypothesis $h\sigma = h\sigma' \in (x : S\sigma) \to C\sigma\, x\, x\, \mathsf{refl}$, which implies $(h\, s)\sigma = (h\, s)\sigma' \in (C\, s\, s\, \mathsf{refl})\sigma$, and finally $(\mathsf{idElim}_{S,s,s'}\, p\, C\, h)\sigma = (\mathsf{idElim}_{S,s,s'}\, p\, C\, h)\sigma' \in (C\, s\, s'\, p)\sigma$.

In case $\mathsf{nf}^W w \not\equiv \mathsf{nf}^W w'$ we have $(\!|p\sigma|\!) = (\!|p\sigma'|\!) = *$. The proof is finished using the definition of $\mathsf{idE}_{S\sigma,s\sigma,s'\sigma}\, *\, C\sigma\, h\sigma$ and Lemma 5.

*Case*

$$\frac{\begin{array}{c}\Gamma \vdash S : \mathsf{Set} \qquad \Gamma \vdash s : S \\ \Gamma \vdash C : (x, x' {:} S) \to \mathsf{Id}_S\, x\, x' \to \mathsf{Set} \\ \Gamma \vdash h : (x{:}S) \to C\, x\, x\, \mathsf{refl}\end{array}}{\Gamma \vdash \mathsf{idElim}_{S,s,s}\, \mathsf{refl}\, C\, h = h\, s : C\, s\, s\, \mathsf{refl}}$$

By induction hypothesis, $h\sigma = h\sigma' \in (x : S\sigma) \to C\sigma\, x\, x\, \mathsf{refl}$ and $s\sigma = s\sigma' \in S\sigma$. Since $(\!|(\mathsf{idElim}_{S,s,s}\, \mathsf{refl}\, C\, h)\sigma|\!) = (\!|h\sigma|\!)@(s\sigma)$ and $(\!|(h\, s)\sigma'|\!) = (\!|h\sigma'|\!)@(s\sigma')$, the goal $(\!|(\mathsf{idElim}_{S,s,s}\, \mathsf{refl}\, C\, h)\sigma|\!) = (\!|(h\, s)\sigma'|\!) \in (\!|(C\, s\, s\, \mathsf{refl})\sigma|\!)$ follows immediately.

*Case*

$$\frac{\Gamma \vdash p, q : \mathsf{Id}_T\, t\, t'}{\Gamma \vdash p = q : \mathsf{Id}_T\, t\, t'}$$

This is trivial by definition of the semantics of the identity type. Here is a proof: Assume $\sigma = \sigma' \in \Gamma$ and show $p' = q' \in W$ where $p' := (\!|p\sigma|\!)$, $q' := (\!|q\sigma'|\!)$, and $W := (\!|(\mathsf{Id}_T\, t\, t')\sigma|\!)$. By induction hypothesis and

Lemma 5, $v := \mathsf{nf}^{(\!|T|\!)}\,(\!|t|\!)$ and $v' := \mathsf{nf}^{(\!|T|\!)}\,(\!|t'|\!)$ are both defined. If they are identical, it must be the case that $p = q = \mathsf{refl}$, otherwise, $p = q = *$. In both cases $p' = q' \in W$.

*Case*

$$\frac{\Gamma, x{:}S \vdash t : T \qquad \Gamma \vdash s : S}{\Gamma \vdash (\lambda x t)\, s = t[s/x] : T[s/x]}$$

Assume $\sigma = \sigma' \in \Gamma$. W.l.o.g., choose $x$ such it does not interfere with $\sigma$ or $\sigma'$. Observe that $(\!|((\lambda x t)\, s)\sigma)|\!) = (\!|(\lambda x.\, t(\sigma, x = x))\, s\sigma|\!) = (\!|\lambda x.\, t(\sigma, x = x)|\!)@s\sigma = (\!|t(\sigma, x = x)[s\sigma/x]|\!) = (\!|t(\sigma, x = s\sigma)|\!)$. Further, $(\!|(t[s/x])\sigma'|\!) = (\!|t(\sigma', x = s\sigma')|\!)$. By induction hypothesis, $(\!|s\sigma|\!) = (\!|s\sigma'|\!) \in (\!|S|\!)$, thus, $s\sigma = s\sigma' \in S$. Hence, $(\sigma, x = s\sigma) = (\sigma', x = s\sigma') \in \Gamma, x : S$. By induction hypothesis again, $\Delta \vdash (\!|t(\sigma, x = s\sigma)|\!) = (\!|t(\sigma', x = s\sigma')|\!) \in (\!|T(\sigma, x = s\sigma)|\!)$. Using the calculations above, this implies the goal $(\!|((\lambda x t)\, s)\sigma|\!) = (\!|(t[s/x])\sigma'|\!) \in (\!|(T[s/x])\sigma|\!)$. □

**Corollary 9 (Completeness of Normalization)** *Let* $\Gamma \vdash t = t' : T$ *and* $\sigma(x) = \uparrow^{(\!|\Gamma(x)|\!)} x$. *Then* $\mathsf{nf}^{(\!|T\sigma|\!)}\,(\!|t\sigma|\!) \equiv \mathsf{nf}^{(\!|T\sigma|\!)}\,(\!|t'\sigma|\!)$.

*Proof.* By Lemma 5, $\sigma = \sigma \in \Gamma$. Then fundamental theorem yields $(\!|t\sigma|\!) = (\!|t'\sigma|\!) \in (\!|T\sigma|\!)$. The goal follows by Lemma 5. □

## 5. Conclusion

We have shown how to decide judgmental equality in a logical framework with proof irrelevant identity types. Identities are restricted to base types. It is not clear yet how to extend our approach to identities between function types, future research will show.

### 5.1. More Related Work

In his habilitation thesis, Streicher [14] considers the $\eta$-rule $\mathsf{idElim}_{S,s,s'}\, p\, C\, (\lambda x.\, e\, x\, x\, \mathsf{refl}) = e$ for $e : (x, y : S) \to (q{:}\mathsf{Id}_S\, x\, y) \to C\, x\, y\, q$ and proves that it entails equality reflection, i.e., Extensional Type Theory. Clearly, this rule destroys decidability of definitional equality and type checking. Streicher also proves that in the presence of UIP, idElim can be replaced by the simpler substitution operator $\mathsf{subst} : (S : \mathsf{Set}) \to (s, s'{:}S) \to \mathsf{Id}_S\, s\, s' \to (P{:}S \to \mathsf{Set}) \to P\, s \to P\, s'$. Thus, we could have chosen subst instead of idElim without losing expressivity.

Werner [16] considers the Calculus of Inductive Constructions (CIC) with proof-irrelevant propositions. During equality checking $t =_\epsilon t'$, all proofs are replaced by $\epsilon$ (our $*$), in particular, canonical and hypothetical proofs of propositional equality are no longer distinguished. Consequently, for identity elimination Werner switches to the reduction rule $\mathsf{subst}_{S,s,s'}\, p\, P\, x \longrightarrow x$ in case $s =_\epsilon s'$. This is generalized to all inductive propositions (which can have at most one constructor in the CIC). We hope to extend our approach to the CIC as well; this would integrate $\eta$-equality into the CIC with proof-irrelevance.

Very interesting related work is *Observational Type Theory* by Altenkirch, McBride, and Swierstra [6]. They implement a heterogeneous propositional equality which features functional extensionality by definition. Proof irrelevance is obtained by type-directed quotation (very similar to contextual reification [3]) which happens after evaluation and "boxes" all proofs such that they can be disregarded in the syntactical equality test. How their approach relates to mine exactly I do not know yet.

*Acknowledgments.* The idea to look at $\eta$-expansion for the identity type came up in discussions with James Chapman and Tarmo Uustalu during a visit to IOC Tallinn in March 2009. I am grateful for their invitation.

## References

[1] A. Abel, K. Aehlig, and P. Dybjer. Normalization by evaluation for Martin-Löf type theory with one universe. In M. Fiore, editor, *Proceedings of the 23rd Conference on the Mathematical Foundations of Programming Semantics (MFPS XXIII), New Orleans, LA, USA, 11-14 April 2007*, volume 173 of *Electronic Notes in Theoretical Computer Science*, pages 17–39. Elsevier, 2007.

[2] A. Abel, T. Coquand, and P. Dybjer. Normalization by evaluation for Martin-Löf Type Theory with typed equality judgements. In *22nd IEEE Symposium on Logic in Computer Science (LICS 2007), 10-12 July 2007, Wroclaw, Poland, Proceedings*, pages 3–12. IEEE Computer Society Press, 2007.

[3] A. Abel, T. Coquand, and P. Dybjer. Verifying a semantic $\beta\eta$-conversion test for Martin-Löf type theory. In P. Audebaud and C. Paulin-Mohring, editors, *Mathematics of Program Construction, 9th International Conference, MPC 2008, Marseille, France, July 15-18, 2008. Proceedings*, volume 5133 of *Lecture Notes in Computer Science*, pages 29–56. Springer-Verlag, 2008.

[4] A. Abel, T. Coquand, and M. Pagano. A modular type-checking algorithm for type theory with singleton types and proof irrelevance. In P.-L. Curien, editor, *Typed Lambda Calculi and Applications, 9th International Conference, TLCA 2009, Brasilia, Brazil, July 1-3, 2009, Proceedings*, volume 5608 of *Lecture Notes in Computer Science*, pages 5–19. Springer-Verlag, 2009.

[5] T. Altenkirch. Extensional equality in intensional type theory. In *14th Annual IEEE Symposium on Logic in Computer*

*Science, 2-5 July, 1999, Trento, Italy, Proceedings*, pages 412–420, 1999.

[6] T. Altenkirch, C. McBride, and W. Swierstra. Observational equality, now! In A. Stump and H. Xi, editors, *Proceedings of the ACM Workshop Programming Languages meets Program Verification, PLPV 2007, Freiburg, Germany, October 5, 2007*, pages 57–68. ACM Press, 2007.

[7] S. Awodey and A. Bauer. Propositions as [types]. *J. Log. Comput.*, 14(4):447–471, 2004.

[8] S. Awodey and M. A. Warren. Homotopy theoretic models of identity types. *Mathematical Proceedings of the Cambridge Philosophical Society*, 146:45–55, 2009.

[9] P. Dybjer. A general formulation of simultaneous inductive-recursive definitions in type theory. *The Journal of Symbolic Logic*, 65(2):525–549, 2000.

[10] N. Gambino and R. Garner. The identity type weak factorisation system. *Theoretical Computer Science*, 409(1):94–109, 2008.

[11] M. Hofmann and T. Streicher. The groupoid interpretation of type theory. In *Twenty-five Years of Constructive Type Theory, Festschrift, Venice 1995*, volume 36 of *Oxford Logic Guides*, pages 83–111. Oxford University Press, 1998.

[12] P. Martin-Löf. Constructive mathematics and computer programming. In *Proc. of a discussion meeting of the Royal Society of London on Mathematical logic and programming languages*, pages 167–184, Upper Saddle River, NJ, USA, 1985. Prentice-Hall, Inc.

[13] B. Nordström, K. Petersson, and J. M. Smith. *Programming in Martin Löf's Type Theory: An Introduction*. Clarendon Press, Oxford, 1990.

[14] T. Streicher. Investigations into Intensional Type Theory, 1993. Habilitation thesis, Ludwig-Maximilians-University Munich.

[15] K. Watkins, I. Cervesato, F. Pfenning, and D. Walker. A concurrent logical framework I: Judgements and properties. Technical report, School of Computer Science, Carnegie Mellon University, Pittsburgh, 2003.

[16] B. Werner. On the strength of proof-irrelevant type theories. *Logical Methods in Computer Science*, 4, 2008.