South American Journal of Logic
Vol. 2, n. 2, pp. 245–296, 2016
ISSN: 2446-6719

Propositional Equality, Identity Types, and Computational Paths

Ruy J.G.B. de Queiroz, Anjolina G. de Oliveira and Arthur F. Ramos[1]

Abstract

In proof theory the notion of canonical proof is rather basic, and it is usually taken for granted that a canonical proof of a sentence must be unique up to certain minor syntactical details (such as, e.g., change of bound variables). When setting up a proof theory for equality one is faced with a rather unexpected situation where there may not be a unique canonical proof of an equality statement. Indeed, in a (1994–5) proposal for the formalisation of proofs of propositional equality in the Curry–Howard style [41], we have already uncovered such a peculiarity. Totally independently, and in a different setting, Hofmann & Streicher (1994) [14] have shown how to build a model of Martin-Löf's Type Theory in which uniqueness of canonical proofs of identity types does not hold. The intention here is to show that, by considering as sequences of rewrites and substitution, it comes a rather natural fact that two (or more) distinct proofs may be yet canonical and are none to be preferred over one another. By looking at proofs of equality as rewriting (or computational) paths this approach will be in line with the recently proposed connections between type theory and homotopy theory via identity types, since elements of identity types will be, concretely, paths (or homotopies).

Keywords: equality, identity type, type theory, homotopy theory, labelled deduction, natural deduction

[1]This paper is an extension of [48], uploaded to arXiv in its first version in July 2011. The authors would like to thank the anonymous referees for their very careful scrutiny of the paper, leading to significant improvements both in content and presentation. It has to be mentioned that the exchanges of e-mail messages with Thomas Streicher, which happened around June–July 2011 while preparing the first version, as well as with Michael Shulman (2012–2013), were of extraordinary value. Any mistakes or misconceptions, however, are the fault of the authors of this paper. Parts of this material have been presented at a number of places, including: Tech Univ Darmstadt's *Logik Seminar* (19 Aug 2013); *Encontro Brasileiro de Lógica 2014* (11 Apr 2014); Univ Federal do Ceará's *I Workshop Científico do LOGIA* (21 Nov 2014); Univ of Brasília's *Workshop de Lógica Aplicada* (05 Feb 2015); Indiana Univ's *Computer Science Colloquium* (24 Jul 2015); Univ of São Paulo's *Logic and Applications: in honor of Francisco Miraglia on the occasion of his 70th birthday* (17 Sep 2016).

1 Introduction

There seems to be hardly any doubt that the so-called "identity types" are the most intriguing concept of intensional Martin-Löf type theory [15, 53]. From the description of a workshop entitled *Identity Types – Topological and Categorical Structure*, organised Nov 13–14, 2006, with support from the Swedish Research Council (VR) and the mathematics departments of Uppsala University and Stockholm University:

> "The identity type, the type of proof objects for the fundamental propositional equality, is one of the most intriguing constructions of intensional dependent type theory (also known as Martin-Löf type theory). Its complexity became apparent with the Hofmann-Streicher groupoid model of type theory. This model also hinted at some possible connections between type theory and homotopy theory and higher categories. Exploration of this connection is intended to be the main theme of the workshop."

Indeed, a whole new research avenue has recently been explored by people like Vladimir Voevodsky [55] and Steve Awodey [3] in trying to make a bridge between type theory and homotopy theory, mainly via the groupoid structure exposed in the Hofmann-Streicher countermodel to the principle of Uniqueness of Identity Proofs (UIP). This has opened the way to, in Awodey's words, "a new and surprising connection between Geometry, Algebra, and Logic, which has recently come to light in the form of an interpretation of the constructive type theory of Per Martin-Löf into homotopy theory, resulting in new examples of certain algebraic structures which are important in topology".

Furthermore, there have been several important strands in the area of categorical semantics for Martin-Löf's type theory, giving rise to rather unexpected links between type theory, abstract homotopy theory and higher-dimensional category theory, as pointed out by van den Berg and Garner [7]. And this is all due to the peculiar structure brought about by the so-called identity types:

> "All of this work can be seen as an elaboration of the following basic idea: that in Martin-Löf type theory, a type A is analogous to a topological space; elements $a, b \in A$ to points of that space; and elements of an identity type $p, q \in \mathtt{Id}_A(a, b)$ to *paths* or *homotopies* $p, q : a \to b$ in A." [7]

Computational paths. Motivated by looking at equalities in type theory as arising from the existence of computational paths between two formal objects, our purpose here is to offer a different perspective on the role and the power of the notion of propositional equality as formalised in the so-called Curry-Howard functional interpretation. We begin by recalling our previous observation [41] pertaining to the fact that the formulation of the identity type by Martin-Löf, both in the intensional and in the extensional versions, did not take into account an important entity, namely, identifiers for sequences of rewrites, and this has led to a false dichotomy. The missing entity has also made it difficult to formulate the *introduction* rule for both the intensional and the extensional version without having to resort to the use of the reflexivity operator "r" as in:

$$\frac{a : A}{\mathtt{r}(a) : \mathtt{Id}_A(a, a)}$$

when this should come as a consequence of the general principle of equality saying that, for all elements a of a type A, equality is by definition a reflexive relation, rather than taking part of the definition of the identity type. Instead, if the introduction rule for the identity type takes the form of:

$$\frac{a =_s b : A}{s(a, b) : \mathtt{Id}_A(a, b)}$$

where the identifier 's' is supposed to denote a sequence of rewrites and substitutions which would have started from a and arrived at b, it becomes rather natural to see members of identity types as *computational* (or *rewriting*) *paths*.[2] By having the general rules for equality defined as:

reflexivity *symmetry* *transitivity*

$$\frac{x : A}{x =_\rho x : A} \qquad \frac{x =_t y : A}{y =_{\sigma(t)} x : A} \qquad \frac{x =_t y : A \qquad y =_u z : A}{x =_{\tau(t,u)} z : A}$$

(where 'σ' and 'τ' are the symmetry and transitivity rewriting operators) one would then be able to infer that

$$\frac{\dfrac{a : A}{a =_\rho a : A}}{\rho(a) : \mathtt{Id}_A(a, a)}$$

Taking an identifier from the meta-language to the object-language. As we can see from the above example, one may start from '$a : A$', i.e. a is an element of type A, and take the a to the object-language by inferring that $\rho(a) : \mathtt{Id}_A(a, a)$. That is to say, in the latter judgement, the object a is being predicated about in the object language ('a is equal to itself'). It is only via identity types that this can be done in the framework of the Curry-Howard functional interpretation.

[2] An anonymous reviewer has pointed out that this is common in the "judgemental" approach to logic: one finds the judgement that underlies a proposition (Cf. [32] and [19, 20]).

Iteration. In the same aforementioned workshop, B. van den Berg in his contribution "Types as weak omega-categories" draws attention to the power of the identity type in the iterating types to form a globular set:

> Fix a type X in a context Γ. Define a globular set as follows: A_0 consists of the terms of type X in context Γ, modulo definitional equality; A_1 consists of terms of the types $Id(X;p;q)$ (in context Γ) for elements p,q in A_0, modulo definitional equality; A_2 consists of terms of well-formed types $Id(Id(X;p;q);r;s)$ (in context Γ) for elements p,q in A_0, r,s in A_1, modulo definitional equality; etcetera...

Indeed, one may start from:

$$\frac{p =_r q : X}{r(p,q) : \mathtt{Id}_X(p,q)} \quad \text{and} \quad \frac{p =_s q : X}{s(p,q) : \mathtt{Id}_X(p,q)}$$

and move up one level as in:

$$\frac{r(p,q) =_u s(p,q) : \mathtt{Id}_X(p,q)}{u(r(p,q),s(p,q)) : \mathtt{Id}_{\mathtt{Id}_X(p,q)}(r(p,q),s(p,q))}$$

and so on... This has been made precise by a theorem of Peter Lumsdaine [21] and, independently, by Benno van den Berg and Richard Garner [6, 7] to the effect that, for any type X in Martin-Löf's (intensional) type theory, the globular set $\hat{X}$ of terms of type $X; \mathtt{Id}_X; \mathtt{Id}_{\mathtt{Id}_X}; ...$ carries a natural weak ω-groupoid structure.

Among other things, this makes it possible to formalise UIP in the theory, as pointed out in Hofmann–Streicher's (1996) survey [15]:

> "We will call *UIP* (*U*niqueness of *I*dentity *P*roofs) the following property. If a_1, a_2 are objects of type A then for any proofs p and q of the proposition "a_1 *equals* a_2" there is another proof establishing equality of p and q. (...) Notice that in traditional logical formalism a principle like *UIP* cannot even be sensibly expressed as proofs cannot be referred to by terms of the object language and thus are not within the scope of propositional equality."

The principle of UIP was originally rendered as [14]:

$$x : A, \quad p : \mathtt{Id}_A(x,x) \quad \vdash \quad \mathtt{Id}_{\mathtt{Id}_A(x,x)}(p, \mathtt{r}_A(x))$$

or in the form of a variant for x and y not assumed to be necessarily equal:

$$x : A, \quad y : A, \quad p,q : \mathtt{Id}_A(x,y) \quad \vdash \quad \mathtt{Id}_{\mathtt{Id}_A(x,y)}(p,q)$$

Counter to the principle, put forward by Martin-Löf, that a type is determined by its canonical object, the model of the identity type constructed by Hofmann & Streicher contains more than one canonical object, and therefore the UIP does not hold. Although this is sharp contrast with the theory of meaning for type theory as developed mainly by Martin-Löf, Prawitz and Dummett, it is in perfect agreement with an alternative theory of meaning based on reduction rules as meaning-giving which we have been advocating for some time now [33, 34, 35, 45, 46, 36, 37, 38, 39, 40].

Elimination rules and the general principles of equality. Another aspect of Martin-Löf's formulation of identity types which has posed difficulties in understanding the notion of normal proofs of equality statements is the framing of elimination rules for identity types as something of the following sort: `Id`-*elimination*

$$\frac{a : A \quad b : A \quad c : \mathtt{Id}_A(a,b) \quad \begin{array}{c}[x{:}A]\\ d(x){:}C(x,x,\mathtt{r}(x))\end{array} \quad \begin{array}{c}[x{:}A,y{:}A,z{:}\mathtt{Id}_A(x,y)]\\ C(x,y,z)\ \mathit{type}\end{array}}{\mathtt{J}(c,d) : C(a,b,c)}$$

together with the conversion rule: `Id`-*conversion*

$$\frac{a : A \quad \begin{array}{c}[x : A]\\ d(x) : C(x,x,\mathtt{r}(x))\end{array} \quad \begin{array}{c}[x : A, y : A, z : \mathtt{Id}_A(x,y)]\\ C(x,y,z)\ \mathit{type}\end{array}}{\mathtt{J}(\mathtt{r}(a), d(x)) = d(a/x) : C(a,a,\mathtt{r}(a))}$$

To the elimination operator '`J`' it is sometimes associated the definition of the usual properties of the equality relation:

> "Surprisingly enough, the `J`-eliminator is sufficient for constructing terms refl, symm, trans and subst inhabiting the types corresponding to the propositions expressing reflexivity, symmetry, transitivity and replacement." [14]

Same as in [15]:

> "The elimination operator J is motivated by the view of $Id(A, \_, \_)$ as an inductively defined family with with constructor *refl*. Accordingly, J permits one to define an object of type $(a_1, a_2 : A)(s : Id(A, a_1, a_2)C(a_1, a_2, s)$ by prescribing its behaviour for arguments of canonical form, i.e. $a_1 = a_2 = a$ and $s = \mathit{refl}(A, a)$.
>
> In the presence of Π-sets, this elimination operation J allows one to derive the following replacement rule in the presence of Π-sets.
>
> $$\mathit{subst} : (A : \mathit{Set})(P : (a : A)\mathit{Set})(a_1, a_2 : A)(s : Id(a_1, a_2))P(a_1) \to P(a_2)$$
>
> satisfying
>
> $$\mathit{subst}(\mathit{refl}(a), p) = p \text{ ''}$$

Nevertheless, in Martin-Löf's type theory the general properties of equality are given at the level of definitional equality, independently of the $\mathtt{J}$ elimination operator for identity types. Moreover, as soon as the formulation of the rules for the identity types take into account the aforementioned "missing entity", and thus the existential force of propositional equality, the existence of proofs of transitivity and symmetry for propositional equality follow from the application of the rules. Our formulation would be as in:

$\mathtt{Id}$-*elimination*

$$\frac{c : \mathtt{Id}_A(x,y) \qquad \begin{array}{c}[x =_t y : A]\\ d(t) : C\end{array}}{\mathtt{J}(c, \acute{t}d(t)) : C}$$

(where $\acute{t}$ is an abstraction over the variable 't') with the following conversion rule:
$\mathtt{Id}$-*conversion*

$$\frac{\dfrac{a =_s b : A}{s(a,b) : \mathtt{Id}_A(a,b)}\mathtt{Id}\text{-}intro \qquad \begin{array}{c}[a =_t b : A]\\ d(t) : C\end{array}}{\mathtt{J}(s(a,b), \acute{t}d(t)) : C}\mathtt{Id}\text{-}elim \qquad \rhd_\beta \qquad \begin{array}{c} a =_s b : A\\ d(s/t) : C\end{array}$$

giving us the equality:

$$\mathtt{J}(s(a,b), \acute{t}d(t)) =_\beta d(s/t)$$

With this formulation, we can see that it is by virtue of the elimination rule combined with the general rules of equality on the level of judgements that one can prove transitivity and symmetry for propositional equality:

Construction 1.1 (inv_A)

$$\dfrac{\dfrac{\dfrac{\dfrac{[c(x,y) : \mathtt{Id}_A(x,y)] \qquad \dfrac{\dfrac{[x =_t y : A]}{y =_{\sigma(t)} x : A}}{(\sigma(t))(y,x) : \mathtt{Id}_A(y,x)}}{\mathtt{J}(c(x,y), (\sigma(\acute{t}))(y,x)) : \mathtt{Id}_A(y,x)}}{\lambda c.\mathtt{J}(c(x,y), (\sigma(\acute{t}))(y,x)) : \mathtt{Id}_A(x,y) \to \mathtt{Id}_A(y,x)}}{\lambda y.\lambda c.\mathtt{J}(c(x,y), (\sigma(\acute{t}))(y,x)) : \Pi y : A.(\mathtt{Id}_A(x,y) \to \mathtt{Id}_A(y,x))}}{\lambda x.\lambda y.\lambda c.\mathtt{J}(c(x,y), (\sigma(\acute{t}))(y,x)) : \Pi x : A.\Pi y : A.(\mathtt{Id}_A(x,y) \to \mathtt{Id}_A(y,x))}$$

where σ is the symmetry operator introduced by the general rule of symmetry given as part of the definition of equality on the lefthand side.

Construction 1.2 (cmp_A)

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[w(x,y):\mathtt{Id}_A(x,y)] \qquad \dfrac{[s(y,z):\mathtt{Id}_A(y,z)] \qquad \dfrac{\dfrac{[x=_t y:A] \qquad [y=_u z:A]}{x=_{\tau(t,u)} z:A}}{(\tau(t,u))(x,z):\mathtt{Id}_A(x,z)}}{\mathtt{J}(s(y,z),\acute{u}(\tau(t,u))(x,z)):\mathtt{Id}_A(x,z)}}{\mathtt{J}(w(x,y),\acute{t}\mathtt{J}(s(y,z),\acute{u}(\tau(t,u))(x,z))):\mathtt{Id}_A(x,z)}}{\lambda s.\mathtt{J}(w(x,y),\acute{t}\mathtt{J}(s(y,z),\acute{u}(\tau(t,u))(x,z))):(\mathtt{Id}_A(y,z)\to\mathtt{Id}_A(x,z))}}{\lambda w.\lambda s.\mathtt{J}(w(x,y),\acute{t}\mathtt{J}(s(y,z),\acute{u}(\tau(t,u))(x,z))):(\mathtt{Id}_A(x,y)\to(\mathtt{Id}_A(y,z)\to\mathtt{Id}_A(x,z)))}}{\lambda z.\lambda w.\lambda s.\mathtt{J}(w(x,y),\acute{t}\mathtt{J}(s(y,z),\acute{u}(\tau(t,u))(x,z))):\Pi z{:}A.(\mathtt{Id}_A(x,y)\to(\mathtt{Id}_A(y,z)\to\mathtt{Id}_A(x,z)))}}{\lambda y.\lambda z.\lambda w.\lambda s.\mathtt{J}(w(x,y),\acute{t}\mathtt{J}(s(y,z),\acute{u}(\tau(t,u))(x,z))):\Pi y{:}A.\Pi z{:}A.(\mathtt{Id}_A(x,y)\to(\mathtt{Id}_A(y,z)\to\mathtt{Id}_A(x,z)))}}{\lambda x.\lambda y.\lambda z.\lambda w.\lambda s.\mathtt{J}(w(x,y),\acute{t}\mathtt{J}(s(y,z),\acute{u}(\tau(t,u))(x,z))):\Pi x{:}A.\Pi y{:}A.\Pi z{:}A.(\mathtt{Id}_A(x,y)\to(\mathtt{Id}_A(y,z)\to\mathtt{Id}_A(x,z)))}
$$

The final proof terms above are called, respectively, inv_A and cmp_A by Streicher [53]:

> "Using $\mathtt{J}$ one can define operations
>
> $cmp_A \in (\Pi x, y, z : A)\mathtt{Id}_A(x, y) \to \mathtt{Id}_A(y, z) \to \mathtt{Id}_A(x, z)$
>
> $inv_A \in (\Pi x, y : A)\mathtt{Id}_A(x, y) \to \mathtt{Id}_A(y, x)$ "

The groupoid laws. It so happens that the existence of cmp_A and inv_A validates the following groupoid laws as pointed out in Streicher's talk at the aforementioned workshop "Identity Types vs. Weak ω-Groupoids – Some Ideas and Problems" [53]:

$$
\begin{array}{ll}
(a) & (\Pi x, y, z, u : A) \\
 & (\Pi f : \mathtt{Id}_A(x, y))(\Pi g : \mathtt{Id}_A(y, z))(\Pi h : \mathtt{Id}_A(z, u)) \\
 & \mathtt{Id}_{\mathtt{Id}_A(x,u)}(cmp_A(f, cmp_A(g, h)), cmp_A(cmp_A(f, g), h)) \\
(b) & (\Pi x, y : A)\mathtt{Id}_{\mathtt{Id}_A(x,y)}(cmp_A(\mathtt{r}(x), f), f) \wedge \mathtt{Id}_{\mathtt{Id}_A(y,x)}(cmp_A(g, \mathtt{r}(y)), g) \\
(c) & (\Pi x, y : A)(\Pi f : \mathtt{Id}_A(x, y)) \\
 & \mathtt{Id}_{\mathtt{Id}_A(x,x)}(cmp_A(f, inv_A(f)), \mathtt{r}(x)) \wedge \mathtt{Id}_{\mathtt{Id}_A(y,y)}(cmp_A(inv_A(f), f), \mathtt{r}(y))
\end{array}
$$

This makes type A an internal groupoid where the groupoid equations hold only in the sense of propositional equality. Indeed, via the reduction rules defined over the terms corresponding to equality proofs, one can see that the laws are validated. Just to motivate the reductions between proofs of equality, let us recall that the rule of *symmetry* is the only rule which changes the direction of an equation. So, its use must be controlled. Here we give two reductions over proofs of equality which are related to such a need for controlling the use of symmetry. (The rewriting system with all reductions between terms of identity types is given in Definition 3.21.)

Definition 1.3 (reductions involving ρ and σ)

$$
\dfrac{x =_\rho x : A}{x =_{\sigma(\rho)} x : A} \quad \triangleright_{sr} \quad x =_\rho x : A
$$

$$
\dfrac{\dfrac{x =_r y : A}{y =_{\sigma(r)} x : A}}{x =_{\sigma(\sigma(r))} y : A} \quad \triangleright_{ss} \quad x =_r y : A
$$

Associated rewritings:
$\sigma(\rho) \triangleright_{sr} \rho$
$\sigma(\sigma(r)) \triangleright_{ss} r$

By applying the rule of propositional equality to the level of $\mathtt{Id}_{\mathtt{Id}_A(x,x)}$ we can get:

$$\frac{\dfrac{x =_\rho x : A}{x =_{\sigma(\rho)} x : A}}{(\sigma(\rho))(x) : \mathtt{Id}_A(x,x)} \quad \triangleright_{sr} \quad \frac{x =_\rho x : A}{\rho(x) : \mathtt{Id}_A(x,x)}$$

i.e., '$(\sigma(\rho))(x)$' and '$\rho(x)$' are two equal proofs of $\mathtt{Id}_A(x,x)$. So,

$$\frac{\sigma(\rho) =_{sr} \rho : \mathtt{Id}_A(x,x)}{(sr)(\sigma(\rho),\rho) : \mathtt{Id}_{\mathtt{Id}_A(x,x)}(\sigma(\rho),\rho)}$$

And similarly:

$$\frac{\dfrac{\dfrac{x =_r y : A}{y =_{\sigma(r)} x : A}}{x =_{\sigma(\sigma(r))} y : A}}{(\sigma(\sigma(r)))(x,y) : \mathtt{Id}_A(x,y)} \quad \triangleright_{ss} \quad \frac{x =_r y : A}{r(x,y) : \mathtt{Id}_A(x,y)}$$

Thus:

$$\frac{\sigma(\sigma(r)) =_{ss} r : \mathtt{Id}_A(x,y)}{(ss)(\sigma(\sigma(r)),r) : \mathtt{Id}_{\mathtt{Id}_A(x,y)}(\sigma(\sigma(r)),r)}$$

Similarly, the transitivity operation on proofs of equality brings us the following reductions:

Definition 1.4 (τ and τ)

$$\frac{\dfrac{x =_t y : A \quad y =_r w : A}{x =_{\tau(t,r)} w : A} \qquad w =_s z : A}{x =_{\tau(\tau(t,r),s)} z : A} \quad \triangleright_{tt} \quad \frac{x =_t y : A \qquad \dfrac{y =_r w : A \quad w =_s z : A}{y =_{\tau(r,s)} z : A}}{x =_{\tau(t,\tau(r,s))} z : A}$$

Associated rewriting:
$\tau(\tau(t,r),s) \triangleright_{tt} \tau(t,\tau(r,s))$

So,

$$\frac{\dfrac{\dfrac{x =_t y : A \quad y =_r w : A}{x =_{\tau(t,r)} w : A} \qquad w =_s z : A}{x =_{\tau(\tau(t,r),s)} z : A}}{(\tau(\tau(t,r),s))(x,z) : \mathtt{Id}_A(x,z)}$$

$$\rhd_{tt} \quad \dfrac{\dfrac{x =_t y : A \qquad \dfrac{y =_r w : A \quad w =_s z : A}{y =_{\tau(r,s)} z : A}}{x =_{\tau(t,\tau(r,s))} z : A}}{(\tau(t,\tau(r,s)))(x,z) : \mathtt{Id}_A(x,z)}$$

Thus

$$\dfrac{\tau(\tau(t,r),s) =_{tt} \tau(t,\tau(r,s)) : \mathtt{Id}_A(x,z)}{(tt)(\tau(\tau(t,r),s),\tau(t,\tau(r,s))) : \mathtt{Id}_{\mathtt{Id}_A(x,z)}(\tau(\tau(t,r),s),\tau(t,\tau(r,s)))}$$

Notice that, although the type $\mathtt{Id}_{\mathtt{Id}_A(x,z)}(\tau(\tau(t,r),s),\tau(t,\tau(r,s)))$ is inhabited, i.e. there is a proof-term of that type, this does not pressupose that, seeing t,r,s as functions, $t \circ (r \circ s) = (t \circ r) \circ s$. This is similar to Hofmann-Streicher's statement on Proposition 4.1 of [15]:

> "If $a_1, a_2, a_3, a_4 : A$ and $s_1 : \mathtt{Id}_A(a_1, a_2)$ and $s_2 : \mathtt{Id}_A(a_2, a_3)$ and $s_3 : \mathtt{Id}_A(a_3, a_4)$ then
>
> $$\mathit{trans}(s_3, (\mathit{trans}(s_2, s_1)) =_{\mathit{prop}} \mathit{trans}(\mathit{trans}(s_3, s_2), s_1)\text{"}$$

(where '$s_1 =_{\mathit{prop}} s_2$' meant that the type $\mathtt{Id}_{\mathtt{Id}_A(a_1,a_2)}(s_1, s_2)$ was inhabited). Notice again that it was not required that $s_3 \circ (s_2 \circ s_1) = (s_3 \circ s_2) \circ s_1$.

The same observation is made by Warren [56]:

> "For example, given terms f of type $\mathtt{Id}_A(a,b)$ and g of type $\mathtt{Id}_A(b,c)$, there exists a "composite" $(g \cdot f)$ of type $\mathtt{Id}_A(a,c)$. However, this composition and the identities mentioned above fail to satisfy the actual category axioms "on-the-nose", but only up to the existence of terms of further "higher-dimensional" identity types. Thus, given f and g as above together with a further term h of type $\mathtt{Id}_A(c,d)$, the type
>
> $$\mathtt{Id}_{\mathtt{Id}_A(a,d)}(h \cdot (g \cdot f), (h \cdot g) \cdot f)$$
>
> is inhabited; but it is not in general the case that $h \cdot (g \cdot f) = (h \cdot g) \cdot f$."

The fact that the structure brought about by identity types satisfy the groupoid laws, but only 'at the propositional equality', is also highlighted by Steve Awodey in his recent survey:

> "In the intensional theory, each type A is thus endowed by the identity types $\mathtt{Id}_A(a,b)$ with a non-trivial structure. Indeed, this structure was observed by Hofmann and Streicher in [HS98] to satisfy conditions analogous to the familiar laws for groupoids. Specifically, the posited refexivity of propositional equality produces identity proofs $\mathtt{r}(a) : \mathtt{Id}_A(a,a)$ for any term $a : A$,

> playing the role of a unit arrow 1_a for a; and when $f : \mathtt{Id}_A(a, b)$ is an identity proof, then (corresponding to the symmetry of identity) there also exists a proof $f^{-1} : \mathtt{Id}_A(b, a)$, to be thought of as the inverse of f; finally, when $f : \mathtt{Id}_A(a, b)$ and $g : \mathtt{Id}_A(b, c)$ are identity proofs, then (corresponding to transitivity) there is a new proof $g \circ f : \mathtt{Id}_A(a, c)$, thought of as the composite of f and g. Moreover, this structure on each type A can be shown to satisfy the usual groupoid laws, but significantly, only **up to propositional equality**." [3]

In what follows we will spell out a refinement of the approach to propositional equality which was presented in a previous paper on the functional interpretation of direct computations [47]. The intention, as already put forward above, is to offer a formulation of a proof theory for propositional equality very much in the style of identity types which, besides being a reformulation of Martin-Löf's own intensional identity types into one which dissolves what we see as a false dichotomy, turns out to validate the groupoid laws as uncovered by Hofmann & Streicher as well as to refute the principle of uniqueness of identity proofs. So, we are left with a sort of 'weak' type theory in which the connections between the deductive system and the semantics of "a type is a space of paths", which is not so obvious in Martin-Löf's own formulation of intensional type theory, becomes rather natural: propositional equality is indeed the type of (computational) sequences/paths between two elements of a type.

The main point of this paper is to show the connections between the approach to propositional equality that we have been developing since the early 1990's to the one put forward by the Hofmann-Streicher-Voevodsky-Awodey tradition which has now come to be documented in a phenomenal and collective production entitled *Homotopy Type Theory* [16] (Aug 2013). This is useful because, in spite of the differences in details, the approaches seem to have arrived at similar conclusions: elements of the identity type are paths/sequences-of-rewrites from an object to another object of a certain type, which gives rise to all these exciting connections to homotopy. The aim is not to make a formal comparison of the two approaches, but rather to explore both the similarities and the differences between them, and at the same time expose the interesting convergence of groundbreaking conclusions with respect to the connections between type theory and homotopy theory.

2 Normal form for proofs of equality

The clarification of the notion of normal form for equality reasoning took an important step with the work of Statman in the late 1970's [51, 52]. The concept of *direct computation* was instrumental in the development of Statman's approach. By way of motivation, let us take a simple example from the λ-calculus.

$$\begin{array}{lllllll}
(\lambda x.(\lambda y.yx)(\lambda w.zw))v & \rhd_\eta & (\lambda x.(\lambda y.yx)z)v & \rhd_\beta & (\lambda y.yv)z & \rhd_\beta & zv \\
(\lambda x.(\lambda y.yx)(\lambda w.zw))v & \rhd_\beta & (\lambda x.(\lambda w.zw)x)v & \rhd_\eta & (\lambda x.zx)v & \rhd_\beta & zv \\
(\lambda x.(\lambda y.yx)(\lambda w.zw))v & \rhd_\beta & (\lambda x.(\lambda w.zw)x)v & \rhd_\beta & (\lambda w.zw)v & \rhd_\eta & zv
\end{array}$$

There is at least one sequence of conversions, i.e. one computational path, from the initial term to the final term. (In this case we have given three!) Thus, in the formal theory of λ-calculus, the term $(\lambda x.(\lambda y.yx)(\lambda w.zw))v$ is declared to be **equal** to zv.

Now, some natural questions arise:

1. Are the sequences/paths themselves *normal*?
2. Are there non-normal sequences/paths?
3. If yes, how are the latter to be identified and (possibly) normalized?
4. What happens if general rules of equality are involved?

Of course, if one considers only the β-contractions, the traditional choice is for the so-called *outermost and leftmost reduction* [13].

Nevertheless, we are interested in an approach to these questions that would be applicable both to λ-calculus and to proofs in Gentzen's style Natural Deduction. As rightly pointed out by Le Chenadec in [9], the notion of normal proof has been somewhat neglected by the systems of equational logic: "In proof-theory, since the original work of Gentzen (1969) on sequent calculus, much work has been devoted to the normalization process of various logics, Prawitz (1965), Girard (1988). Such an analysis was lacking in equational logic (the only exceptions we are aware of are Statman (1977), Kreisel and Tait (1961))." The works of Statman [51, 52] and Le Chenadec [9] represent significant attempts to fill this gap. Statman studies proof transformations for the equational calculus E of Kreisel–Tait [18]. Le Chenadec defines an equational proof system (the *LE system*) and gives a normalization procedure.

What is a proof of an equality statement? The so-called Brouwer-Heyting-Kolmogorov Interpretation defines logical connectives by taking proof, rather than truth-values, as a primitive notion:

a proof of the proposition:	**is given by:**
$A \wedge B$	a proof of A **and** a proof of B
$A \vee B$	a proof of A **or** a proof of B
$A \to B$	a **function** that turns a proof of A into a proof of B
$\forall x^D.P(x)$	a **function** that turns an element a into a proof of $P(a)$
$\exists x^D.P(x)$	an element a (witness) **and a proof of** $P(a)$

Based on the Curry-Howard functional interpretation of logical connectives, one can formulate the BHK-interpretation in formal terms as following:

a proof of the proposition:	**has the canonical form of:**
$A \wedge B$	$\langle p, q \rangle$ where p is a proof of A and q is a proof of B
$A \vee B$	$i(p)$ where p is a proof of A or $j(q)$ where q is a proof of B (‘i’ and ‘j’ abbreviate ‘into the left/right disjunct’)
$A \to B$	$\lambda x.b(x)$ where $b(p)$ is a proof of B provided p is a proof of A
$\forall x^A.B(x)$	$\Lambda x.f(x)$ where $f(a)$ is a proof of $B(a)$ provided a is an arbitrary individual chosen from the domain A
$\exists x^A.B(x)$	$\varepsilon x.(f(x), a)$ where a is a witness from the domain A, $f(a)$ is a proof of $B(a)$

(The term ‘$\varepsilon x.(f(x), a)$’ is framed so as to formalise the notion of a function carrying its own argument [42].)

A question remains, however:

> What is a proof of an equality statement?

An answer to such a question will help us extend the BHK-interpretation with an explanation of what is a proof of an equality statement:

a proof of the proposition:	**is given by:**
$t_1 = t_2$	? (Perhaps a sequence of rewrites starting from t_1 and ending in t_2?)

Two related questions naturally arise:

1. What is the logical status of the symbol “=”?
2. What would be a canonical/direct proof of $t_1 = t_2$?

In a previous work [49] we have tried to show how the framework of labelled natural deduction can help us formulate a proof theory for the “logical connective” of propositional equality.[3] The connective is meant to be used in reasoning about equality between

[3] An old question is in order here: what is a logical connective? We shall take it that from the point of view of proof theory (natural deduction style) a logical connective is whatever logical symbol which is analysable into rules of *introduction* and *elimination*.

referents (i.e. the terms alongside formulas/types), as well as with a general notion of substitution which is needed for the characterization of the so-called *term declaration logics* [2].

In order to account for the distinction between the equalities that are:

> *definitional*, i.e. those equalities that are given as rewrite rules (equations), orelse originate from general functional principles (e.g. β, η, etc.),

and those that are:

> *propositional*, i.e. the equalities that are supported (or otherwise) by an evidence (a composition of rewrites),

we need to provide for an equality sign as a symbol for *rewrite* (i.e. as part of the functional calculus on the terms), and an equality sign as a symbol for a *relation* between referents (i.e. as part of the logical calculus on the formulas/types).

Single steps of reduction come from definitional equalities, and those single steps can be composed leading to sequences of rewrites, which can then turned into a propositional equality. It helps to remember that in "$t : A$", the logical interpretation is that "t" is a (functional) term, and "A" is a statement. So, the equality is propositional when it is a statement, i.e., in "$q : \mathtt{Id}$", "$\mathtt{Id}$" is a statement which is supported by the term "q" (which, in its turn, can be an equational term like "$a =_s b$"). So, while "q" will carry definitional content (be it single or composed), "$\mathtt{Id}$" will carry propositional content.

Definitional equalities. Let us recall from the theory of λ-calculus, that:

Definition 2.1 ([13], (Definition 6.2 and Notation 7.1)) *The formal theory of $\lambda\beta\eta$ equality has the following axioms:*

$$
\begin{array}{lll}
(\alpha) & \lambda x.M = \lambda y.[y/x]M & (y \notin FV(M)) \\
(\beta) & (\lambda x.M)N = [N/x]M & \\
(\eta) & (\lambda x.Mx) = M & (x \notin FV(M)) \\
(\rho) & M = M &
\end{array}
$$

and the following inference rules:

$$
(\mu)\ \frac{M = M'}{NM = NM'} \qquad (\tau)\ \frac{M = N \qquad N = P}{M = P}
$$

$$
(\nu)\ \frac{M = M'}{MN = M'N} \qquad (\sigma)\ \frac{M = N}{N = M}
$$

$$
(\xi)\ \frac{M = M'}{\lambda x.M = \lambda x.M'}
$$

$$
(\zeta)\ \frac{Mx = Nx}{M = N} \qquad \textit{if } x \notin FV(MN)
$$

In Martin-Löf's type theory the axioms are introduced as:

$$(\beta)\quad \frac{N : A \quad \begin{array}{c}[x : A]\\ M : B\end{array}}{(\lambda x.M)N = M[N/x] : B}$$

$$(\eta)\quad \frac{M : (\Pi x : A)B}{(\lambda x.Mx) = M : (\Pi x : A)B}\ (x \notin FV(M))$$

$$(\rho)\quad \frac{M : A}{M = M : A}$$

$$(\mu)\quad \frac{M = M' : A \quad N : (\Pi x : A)B}{NM = NM' : B} \qquad (\tau)\quad \frac{M = N : A \qquad N = P : A}{M = P : A}$$

$$(\nu)\quad \frac{N : A \quad M = M' : (\Pi x : A)B}{MN = M'N : B} \qquad (\sigma)\quad \frac{M = N : A}{N = M : A}$$

$$(\xi)\quad \frac{\begin{array}{c}[x : A]\\ M = M' : B\end{array}}{\lambda x.M = \lambda x.M' : (\Pi x : A)B}$$

Propositional equality. Again, let us recall from the formal theory of λ-calculus, that:

> **Definition 1.37 (β-equality)** [13]
> We say that P is β-equal or β-convertible to Q (notation $P =_\beta Q$) iff Q can be obtained from P by a finite (perhaps empty) series of β-contractions and reversed β-contractions and changes of bound variables. That is, $P =_\beta Q$ iff there exist $P_0, \ldots, P_n$ $(n \geq 0)$ such that
>
> $$(\forall i \leq n-1)(P_i \triangleright_{1\beta} P_{i+1} \text{ or } P_{i+1} \triangleright_{1\beta} P_i \text{ or } P_i \equiv_\alpha P_{i+1}).$$
> $$P_0 \equiv P, \qquad P_n \equiv Q.$$

NB: equality with an **existential** force.

The same happens with $\lambda\beta\eta$-equality:

> **Definition 7.5 ($\lambda\beta\eta$-equality)** [13]
> The equality-relation determined by the theory $\lambda\beta\eta$ is called $=_{\beta\eta}$; that is, we define
>
> $$M =_{\beta\eta} N \quad \Leftrightarrow \quad \lambda\beta\eta \vdash M = N.$$

Note again that two terms are $\lambda\beta\eta$-equal if **there exists** a proof of their equality in the theory of $\lambda\beta\eta$-equality.

Remark 2.2 *In setting up a set of Gentzen's ND-style rules for equality we need to account for:*
1. the dichotomy definitional versus propositional equality;
2. there may be more than one normal *proof of a certain equality statement;*
3. given a (possibly non-normal) proof, the process of bringing it to a normal form should be finite and confluent.

The missing entity. Within the framework of the functional interpretation (*à la* Curry–Howard [17]), the definitional equality is often considered by reference to a judgement of the form:

$$a = b : A$$

which says that a and b are equal elements from domain or type A. Notice that the 'reason' why they are equal does not play any part in the judgement. This aspect of 'forgetting contextual information' is, one might say, the first step towards 'extensionality' of equality, for whenever one wants to introduce intensionality into a logical system one invariably needs to introduce information of a 'contextual' nature, such as, where the identification of two terms (i.e. equation) comes from.

We feel that a first step towards finding an alternative formulation of the proof theory for propositional equality which takes care of the intensional aspect is to allow the 'reason' for the equality to play a more significant part in the form of judgement. We also believe that from the point of view of the logical calculus, if there is a 'reason' for two expressions to be considered equal, the proposition asserting their equality will be true, regardless of what particular composition of rewrites (definitional equalities) amounts to the evidence in support of the proposition concerned. Given these general guidelines, we shall provide what may be seen as a middle ground solution between the intensional [23, 22] and the extensional [24] accounts of Martin-Löf's propositional equality. The intensionality is taken care by the functional calculus on the labels, while the extensionality is catered by the logical calculus on the formulas. In order to account for the intensionality in the labels, we shall make the composition of rewrites (definitional equalities) appear as indexes of the equality sign in the judgement with a variable denoting a sequence of equality identifiers (we have seen that in the Curry–Howard functional interpretation there are at least four 'natural' equality identifiers: β, η, ξ and μ). So, instead of the form above, we shall have the following pattern for the equality judgement:

$$a =_s b : A$$

where 's' is meant to be a sequence of equality identifiers.

In the sequel we shall be discussing in some detail the need to identify the kind of definitional equality, as well as the need to have a logical connective of 'propositional equality' in order to be able to reason about the functional objects (those to the left hand side of the ':' sign).

Term rewriting. Deductive systems based on the Curry–Howard isomorphism [17] have an interesting feature: normalization and strong normalization (Church–Rosser property) theorems can be proved by reductions on the terms of the functional calculus. Exploring this important characteristic, we have proved these theorems for the *Labelled Natural Deduction – LND* [44, 50] via a term rewriting system constructed from the *LND*-terms of the functional calculus [29]. Applying this same technique to the *LND* equational fragment, we obtain the normalization theorems for the equational logic of the *Labelled Natural Deduction* System [28, 30, 31].

This technique is used given the possibility of defining two measures of redundancy for the *LND* system that can be dealt with in the object level: the terms on the functional calculus and the *rewrite reason* (composition of rewrites), the latter being indexes of the equations in the *LND* equational fragment.

In the *LND* equational logic [41], the equations have the following pattern:

$$a =_s b : A$$

where one is to read: a is equal to b *because* of 's' ('s' being the *rewrite reason*); 's' is a term denoting a sequence of equality identifiers (β, η, α, etc.), i.e. a composition of rewrites. In other words, 's' denotes the *computational path* from a to b.

In this way, the *rewrite reason* (reason, for short) represents an **orthogonal measure of redundancy** for the *LND*, which makes the *LND* equational fragment an "enriched" system of equational logic. Unlike the traditional equational logic systems, in *LND* equational fragment there is a gain in local control by the use of *reason*. All the proof steps are recorded in the composition of rewrites (reasons). Thus, consulting the reasons, one should be able to see whether the proof has the normal form. We have then used this powerful mechanism of controlling proofs to present a precise normalization procedure for the *LND* equational fragment. Since the reasons can be dealt with in the object level, we can employ a computational method to prove the normalization theorems: we built a term rewriting system based on an algebraic calculus on the "*rewrite reasons*", which compute normal proofs. With this we believe we are making a step towards filling a gap in the literature on equational logic and on proof theory (natural deduction).

Kreisel–Tait's system. In [18] Kreisel and Tait define the system E for equality reasoning as consisting of axioms of the form $t = t$, and the following rules of inference:

$$(E1) \qquad \frac{E[t/x] \quad t = u}{E[u/x]}$$

$$(E2) \qquad \frac{s(t) = s(u)}{t = u}$$

$$(E3) \quad \frac{0 = s(t)}{A} \quad \text{for any formula } A$$

$$(E4_n) \quad \frac{t = s^n(t)}{A} \quad \text{for any formula } A$$

where t and u are terms, '0' is the first natural number (zero), 's(-)' is the successor function.

Statman's normal form theorem. In order to prove the normalization results for the calculus E Statman defines two subsets of E: (i) a natural deduction based calculus for equality reasoning NE; (ii) a sequent style calculus SE.

The NE calculus is defined as having axioms of the form $a = a$, and the rule of substituting equals for equals:

$$(=) \quad \frac{E[a/u] \quad a \approx b}{E[b/u]}$$

where E is any set of equations, and $a \approx b$ is ambiguously $a = b$ and $b = a$.

Statman arrives at various important results on normal forms and bounds for proof search in NE. In this case, however, a rather different notion of normal form is being used: the 'cuts' do not arise out of an *inversion principle*, as it is the case for the logical connectives, but rather from a certain form of sequence of equations which Statman calls *computation*, and whose normal form is called *direct computation*. With the formulation of a proof theory for the 'logical connective' of propositional equality we wish to analyse equality reasoning into its basic components: rewrites, on the one hand, and statements about the existence of rewrites, on the other hand. This type of analysis came to the surface in the context of constructive type theory and the Curry–Howard functional interpretation.

Martin-Löf's Identity type. There has been essentially two approaches to the problem of characterizing a proof theory for propositional equality, both of which originate in P. Martin-Löf's work on *Intuitionistic Type Theory*: the intensional [23] and the extensional [24, 25] formulations.

The extensional version. In his [24] and [25] presentations of *Intuitionistic Type Theory* P. Martin-Löf defines the type of *extensional* propositional equality '$\mathtt{Id}$' (here called '$\mathtt{Id}^{ext}$') as:

$\mathtt{Id}^{ext}$-*formation*

$$\frac{A\ type \qquad a : A \qquad b : A}{\mathtt{Id}_A^{ext}(a, b)\ type}$$

$\mathtt{Id}^{ext}$-*introduction*

$$\frac{a = b : A}{\mathtt{r} : \mathtt{Id}_A^{ext}(a, b)}$$

$\mathtt{Id}^{ext}$-*elimination*[4]

$$\frac{c : \mathtt{Id}_A^{ext}(a, b)}{a = b : A}$$

$\mathtt{Id}^{ext}$-*equality*

$$\frac{c : \mathtt{Id}_A^{ext}(a, b)}{c = \mathtt{r} : \mathtt{Id}_A^{ext}(a, b)}$$

Note that the above account of propositional equality does not 'keep track of all proof steps': both in the $\mathtt{Id}^{ext}$-*introduction* and in the $\mathtt{Id}^{ext}$-*elimination* rules there is a considerable loss of information concerning the deduction steps. While in the $\mathtt{Id}^{ext}$-*introduction* rule the 'a' and the 'b' do not appear in the 'trace' (the label/term alongside the logical formula/type), the latter containing only the canonical element '$\mathtt{r}$', in the rule of $\mathtt{Id}^{ext}$-*elimination* all the trace that might be recorded in the term 'c' simply disappears from label of the conclusion. If by 'intensionality' we understand a feature of a logical system which identifies as paramount the concern with issues of *context* and *provability*, then it is quite clear that any logical system containing $\mathtt{Id}^{ext}$-type can hardly be said to be 'intensional': as we have said above, neither its *introduction* rule nor its *elimination* rule carry the necessary *contextual* information from the premise to the conclusion.

The intensional version. Another version of the propositional equality, which has its origins in Martin-Löf's early accounts of *Intuitionistic Type Theory* [22, 23], and is apparently in the most recent, as yet unpublished, versions of type theory, is defined in [54] and [27]. In a section dedicated to the *intensional vs. extensional* debate, [54] (p.633) says that:

[4]The set of rules given in [24] contained the additional *elimination* rule:

$$\frac{c : \mathtt{Id}_A^{ext}(a, b) \qquad d : C(\mathtt{r}/z)}{\mathtt{J}(c, d) : C(c/z)}$$

which may be seen as reminiscent of the previous *intensional* account of propositional equality [23].

> "Martin-Löf has returned to an intensional point of view, as in Martin-Löf (1975), that is to say, $t = t' \in A$ is understood as "t and t' are definitionally equal". As a consequence the rules for identity types have to be adapted."

If we try to combine the existing accounts of the *intensional* equality type '$\texttt{Id}_A$' [23, 54, 27], here denoted '$\texttt{Id}^{int}$', the rules will look like:

$\texttt{Id}^{int}$-*formation*

$$\frac{A\ type \qquad a : A \qquad b : A}{\texttt{Id}^{int}_A(a,b)\ type}$$

$\texttt{Id}^{int}$-*introduction*

$$\frac{a : A}{\texttt{r}(a) : \texttt{Id}^{int}_A(a,a)} \qquad \frac{a = b : A}{\texttt{r}(a) : \texttt{Id}^{int}_A(a,b)}$$

$\texttt{Id}^{int}$-*elimination*

$$\frac{a : A \quad b : A \quad c : \texttt{Id}^{int}_A(a,b) \quad \begin{array}{c}[x:A]\\ d(x):C(x,x,\texttt{r}(x))\end{array} \quad \begin{array}{c}[x:A,y:A,z:\texttt{Id}^{int}_A(x,y)]\\ C(x,y,z)\ type\end{array}}{\texttt{J}(c,d):C(a,b,c)}$$

$\texttt{Id}^{int}$-*equality*

$$\frac{a : A \;\; \begin{array}{c}[x : A]\\ d(x) : C(x,x,\texttt{r}(x))\end{array} \qquad \begin{array}{c}[x : A, y : A, z : \texttt{Id}^{int}_A(x,y)]\\ C(x,y,z)\ type\end{array}}{\texttt{J}(\texttt{r}(a), d(x)) = d(a/x) : C(a,a,\texttt{r}(a))}$$

With slight differences in notation, the 'adapted' rules for identity type given in [54] and [27] resembles the one given in [23]. It is called *intensional* equality because there remains no direct connection between judgements like '$a = b : A$' and '$c : \texttt{Id}^{int}_A(a,b)$'.

A labelled proof theory for propositional equality. Now, it seems that an alternative formulation of propositional equality within the functional interpretation, which will be a little more elaborate than the extensional $\texttt{Id}^{ext}_A$-type, and simpler than the intensional $\texttt{Id}^{int}_A$-type, could prove more convenient from the point of view of the 'logical interpretation'. It seems that whereas in the former we have a considerable loss of information in the $\texttt{Id}^{ext}$-*elimination*, in the sense that propositional equality and definitional equality are collapsed into one, in the latter we have an $\texttt{Id}^{int}$-*elimination* too heavily loaded with (perhaps unnecessary) information. If, on the one hand, there is an *over*explicitation of information in $\texttt{Id}^{int}$, on the other hand, in $\texttt{Id}^{ext}$ we have a case of *under*explicitation. With the formulation of a proof theory for equality via labelled natural deduction we wish to find a middle ground solution between those two extremes.

3 Labelled deduction

The functional interpretation of logical connectives via deductive systems which use some sort of labelling mechanism [25, 11, 12] can be seen as the basis for a general framework characterizing logics via a clear separation between a functional calculus on the *labels*, i.e. the referents (names of individuals, expressions denoting the record of proof steps used to arrive at a certain formula, names of 'worlds', etc.) and a logical calculus on the formulas. The key idea is to make these two dimensions as harmonious as possible, i.e. that the functional calculus on the labels matches the logical calculus on the formulas at least in the sense that to every abstraction on the variables of the functional calculus there corresponds a discharge of an assumption-formula of the logical calculus. One aspect of such interpretation which stirred much discussion in the literature of the past ten years or so, especially in connection with *Intuitionistic Type Theory* [25], was that of whether the logical connective of propositional equality ought to be dealt with 'extensionally' or 'intensionally'. Here we attempt to formulate what appears to be a middle ground solution, in the sense that the intensional aspect is dealt with in the functional calculus on the labels, whereas the extensionality is kept to the logical calculus. We also intend to demonstrate that the connective of propositional equality (cf. Aczel's [1] '`Id`') needs to be dealt with in a similar manner to 'Skolem-type' connectives (such as disjunction and existential quantification), where notions like *hiding*, *choice* and *dependent variables* play crucial rôles.

3.1 Identifiers for (compositions of) equalities

In the functional interpretation, where a functional calculus on the labels go hand in hand with a logical calculus on the formulas, we have a classification of equalities, whose identifications are carried along as part of the deduction: either β-, η-, ξ-, μ- or α- equality will have been part of an expression labelling a formula containing '`Id`'. There one finds the key to the idea of 'hiding' in the *introduction* rule, and opening local (Skolem-type) assumptions in the *elimination* rule. (Recall that in the case of disjunction we also have alternatives: either into the left disjunct, or into the right disjunct.) So, we believe that it is not unreasonable to start off the formalization of propositional equality with the parallel to the disjunction and existential cases in mind. Only, the witness of the type of propositional equality are not the 'a's and 'b's of '$a = b : A$', but the actual (sequence of) equalities (β-, η-, ξ-, α-) that might have been used to arrive at the judgement '$a =_s b : A$' (meaning '$a = b$' *because* of 's'), 's' being a sequence made up of β-, η-, ξ- and/or α-equalities, perhaps with some of the general equality rules of reflexivity, symmetry and transitivity. So, in the *introduction* rule of the type we need to form the canonical proof as if we were *hiding* the actual sequence. Also, in the rule of *elimination* we need to open a new local assumption introducing a new variable denoting a possible sequence as a (Skolem-type) new constant. That is, in order to eliminate the connective '$\mathtt{Id}_A$' (i.e. to deduce something from a proposition like '$\mathtt{Id}_A(a,b)$'), we start by choosing a new variable to denote the reason why the two terms are equal: 'let t be an expression (sequence of equalities) justifying the equality between the terms'. If we then arrive at an arbitrary formula 'C' labelled with an expression where the t still occurs free, then we can conclude that the same C can be obtained from the Id-formula regardless of the identity of the chosen t, meaning that the label alongside C in the conclusion will have been abstracted from the free occurrences of t.

Observe that now we are still able to 'keep track' of all proof steps (which does not happen with Martin-Löf's $\mathtt{Id}^{ext}_A$-type) [24, 25], and we have an easier formulation (as compared with Martin-Löf's $\mathtt{Id}^{int}_A$-type) [23] of how to perform the *elimination* step.

3.2 The proof rules

In formulating the propositional equality connective, which we shall identify by '$\mathtt{Id}$', we shall keep the pattern of inference rules essentially the same as the one used for the other logical connectives (as in, e.g. [42]), and we shall provide an alternative presentation of propositional equality as follows:

$\mathtt{Id}$-*formation*

$$\frac{A\ type \qquad a : A \qquad b : A}{\mathtt{Id}_A(a,b)\ type}$$

$\mathtt{Id}$-*introduction*

$$\frac{a =_s b : A}{s(a,b) : \mathtt{Id}_A(a,b)} \qquad \frac{a =_s b : A \quad a =_t b : A \quad s =_z t : \mathtt{Id}_A(a,b)}{s(a,b) =_{\xi(z)} t(a,b) : \mathtt{Id}_A(a,b)}$$

(Notice that the ξ rule for $\mathtt{Id}_A$ has an extra hypothesis, which, though apparently circular, is concerned with making sure that not all sequences of rewrites from a to b are definitionally equal: in order to be declared ξ-equal, two sequences need to be equal from some other reason.)

$\mathtt{Id}$-*elimination*

$$\frac{p : \mathtt{Id}_A(a,b) \qquad \begin{matrix}[a =_t b : A] \\ d(t) : C\end{matrix}}{\mathtt{J}(p, \acute{t}d(t)) : C} \qquad \frac{p =_r q : \mathtt{Id}_A(a,b) \qquad \begin{matrix}[a =_t b : A] \\ d(t) : C\end{matrix}}{\mathtt{J}(p, \acute{t}d(t)) =_{\mu(r)} \mathtt{J}(q, \acute{t}d(t)) : C}$$

$\mathtt{Id}$-*reduction*

$$\frac{\dfrac{a =_s b : A}{s(a,b) : \mathtt{Id}_A(a,b)}\,\mathtt{Id}\text{-}intr \qquad \begin{matrix}[a =_t b : A] \\ d(t) : C\end{matrix}}{\mathtt{J}(s(a,b), \acute{t}d(t)) : C}\,\mathtt{Id}\text{-}elim \qquad \rhd_\beta \qquad \begin{matrix}a =_s b : A \\ d(s/t) : C\end{matrix}$$

giving rise to the equality

$$\mathtt{J}(s(a,b), \acute{t}d(t)) =_\beta d(s/t) : C$$

$\mathtt{Id}$-*induction*

$$\cfrac{e : \mathtt{Id}_A(a,b) \qquad \cfrac{[a =_t b : A]}{t(a,b) : \mathtt{Id}_A(a,b)}\,\mathtt{Id}\text{-}intr}{\mathtt{J}(e, \acute{t}t(a,b)) : \mathtt{Id}_A(a,b)}\,\mathtt{Id}\text{-}elim \qquad \rhd_\eta \qquad e : \mathtt{Id}_A(a,b)$$

giving rise to the equality

$$\mathtt{J}(e, \acute{t}t(a,b)) =_\eta e : \mathtt{Id}_A(a,b)$$

where '$\acute{}$' is an abstractor which binds the occurrences of the (new) variable 't' introduced with the local assumption '$[a =_t b : A]$' as a kind of 'Skolem'-type constant denoting the (presumed) 'reason' why 'a' was assumed to be equal to 'b'. (Recall the Skolem-type procedures of introducing new local assumptions in order to allow for the elimination of logical connectives where the notion of 'hiding' is crucial, e.g. disjunction and existential quantifier – in [42].)

Now, having been defined as a 'Skolem'-type connective, '$\mathtt{Id}$' needs to have a conversion stating the non-interference of the newly opened branch (the local assumption in the $\mathtt{Id}$-*elimination* rule) with the main branch. Thus, we have:

$\mathtt{Id}$-(*permutative*) *reduction*

$$\cfrac{\cfrac{e : \mathtt{Id}_A(a,b) \qquad \begin{matrix}[a =_t b : A] \\ d(t) : C\end{matrix}}{\mathtt{J}(e, \acute{t}d(t)) : C}}{w(\mathtt{J}(e, \acute{t}d(t))) : W}\,\mathsf{r} \quad \rhd_\zeta \quad \cfrac{e : \mathtt{Id}_A(a,b) \qquad \cfrac{\begin{matrix}[a =_t b : A] \\ d(t) : C\end{matrix}}{w(d(t)) : W}\,\mathsf{r}}{\mathtt{J}(e, \acute{t}w(d(t))) : W}$$

provided w does not disturb the existing dependencies in the term e (the main branch), i.e. provided that rule 'r' does not discharge any assumption on which '$\mathtt{Id}_A(a,b)$' depends. The corresponding ζ-equality is:

$$w(\mathtt{J}(e, \acute{t}d(t))) =_\zeta \mathtt{J}(e, \acute{t}w(d(t)))$$

The equality indicates that the operation w can be pushed inside the $\acute{}$-abstraction term, provided that it does not affect the dependencies of the term e.

Since we are defining the logical connective '$\mathtt{Id}$' as a connective which deals with singular terms, where the 'witness' is supposed to be hidden, we shall not be using direct *elimination* like Martin-Löf's $\mathtt{Id}^{ext}$-*elimination*. Instead, we shall be using the following $\mathtt{Id}$-*elimination*:

$$\frac{e : \mathtt{Id}_A(a,b) \qquad \begin{matrix}[a =_t b : A] \\ d(t) : C\end{matrix}}{\mathtt{J}(e, \acute{t}d(t)) : C} \qquad \frac{e =_s f : \mathtt{Id}_A(a,b) \qquad \begin{matrix}[a =_t b : A] \\ d(t) : C\end{matrix}}{\mathtt{J}(e, \acute{t}d(t)) =_{\mu(s)} \mathtt{J}(f, \acute{t}d(t)) : C}$$

The *elimination* rule involves the introduction of a new local assumption (and corresponding variable in the functional calculus), namely '$[a =_t b : A]$' (where 't' is the new variable) which is only discharged (and 't' bound) in the conclusion of the rule. The intuitive explanation would be given in the following lines. In order to eliminate the equality $\mathtt{Id}$-connective, where one does not have access to the 'reason' (i.e. a sequence of 'β', 'η', 'ξ' or 'ζ' equalities) why the equality holds because '$\mathtt{Id}$' is supposed to be a connective dealing with singular terms (as are '$\vee$' and '$\exists$'), in the first step one has to open a new local assumption supposing the equality holds because of, say 't' (a new variable). The new assumption then stands for 'let t be the unknown equality'. If a third (arbitrary) statement can be obtained from this new local assumption via an unspecified number of steps which does not involve any binding of the new variable 't', then one discharges the newly introduced local assumption binding the free occurrences of the new variable in the label alongside the statement obtained, and concludes that that statement is to be labelled by the term '$\mathtt{J}(e, \acute{t}d(t))$' where the new variable (i.e. t) is bound by the '´'-abstractor.

Another feature of the $\mathtt{Id}$-connective which is worth noticing at this stage is the equality under 'ξ' of all its elements (see second *introduction* rule). This does not mean that the labels serving as evidences for the $\mathtt{Id}$-statement are all identical to a constant (cf. constant '$\mathtt{r}$' in Martin-Löf's $\mathtt{Id}_{ext}$-type), but simply that if two (sequences of) equality are obtained as witnesses of the equality between, say 'a' and 'b' of domain A, then they are taken to be equal under ξ-equality. It would not seem unreasonable to think of the $\mathtt{Id}$-connective of propositional equality as expressing the proposition which, whenever true, indicates that the two elements of the domain concerned are equal under some (unspecified, *hidden*) composition of definitional equalities. It is as if the proposition points to the existence of a term (witness) which depends on both elements and on the kind of equality judgements used to arrive at its proof. So, in the logical side, one forgets about what was the actual witness. Cf. the existential generalization:

$$\frac{F(t)}{\exists x.F(x)}$$

where the actual witness is in fact 'abandoned'. Obviously, as we are interested in keeping track of relevant information introduced by each proof step, in labelled natural deduction system the witness is not abandoned, but is carried over as an unbounded name in the label of the corresponding conclusion formula.

$$\frac{t : A \qquad f(t) : F(t)}{\varepsilon x.(f(x), t) : \exists x^A.F(x)}$$

Note, however, that it is carried along *only* in the functional side, the logical side not keeping any trace of it at all.

Now, notice that if the functional calculus on the labels is to match the logical calculus on the formulas, than we must have the resulting label on the left of the '$\triangleright_\beta$' as β-convertible to the concluding label on the right. So, we must have the convertibility equality:

$$\mathtt{J}(s(a,b), \acute{t}d(t)) =_\beta d(s/t) : C$$

The same holds for the η-equality:

$$\mathtt{J}(e, \acute{t}t(a,b)) =_\eta e : \mathtt{Id}_A(a,b)$$

Parallel to the case of disjunction, where two different `constructors` distinguish the two alternatives, namely 'i' and 'j', we here have any (sequence of) equality identifiers ('β', 'η', 'μ', 'ξ', etc.) as `constructor`s of proofs for the `Id`-connective. They are meant to denote the alternatives available.

Function extensionality. In setting out to build a proof of functional extensionality using our formulation of the `Id`-type, i.e.:

$$\Pi f^{A\to B}\Pi g^{A\to B}(\Pi x^A \mathtt{Id}_B(APP(f,x), APP(g,x)) \to \mathtt{Id}_{A\to B}(f,g))$$

we end up proving a weakened version which says that if A is non-empty, then the above principle of function extensionality over $A \to B$ holds:

$$A \to (\Pi f^{A\to B}\Pi g^{A\to B}(\Pi x^A \mathtt{Id}_B(APP(f,x), APP(g,x)) \to \mathtt{Id}_{A\to B}(f,g)))$$

As a matter of fact, in type theory, when the type A is empty, one can prove that the type $A \to B$ has only one member, i.e., $\lambda x.x$, but then every (closed) lambda expression denotes that member. This phenomenon is brought up in one of Nuprl's primitives (`http://www.cs.cornell.edu/home/sfa/Nuprl/NuprlPrimitives/Xvoid_doc.html`):

> Thm* Void$\to$ A =ext $\{\lambda x.x$:Void$\to$ $A\}$
>
> The last theorem above states that Void$\to$ A has only the one member ($\lambda x.x$). This is a degenerate function in that it cannot be applied to anything in Void. Indeed, the body of the function expression is irrelevant since it will never be evaluated as part of application to any argument of type Void.
>
> Thm* (λx.whatever) = ($\lambda x.x$) $\in$ Void$\to$ A
>
> So, although Void$\to$ A has only one member, every (closed) lambda expression denotes that member.

In the proof shown at the Appendix B, notice that the leftmost occurrence of the variable z in the proof term of the formula immediately above the conclusion is free. So, in type theory we can apply $\rightarrow$-introduction and do an abstraction on the variable, discharging the assumption $[z : A]$, and thus introduction the implication: A implies the premiss of the rule.

General rules of equality. Apart from the already mentioned 'constants' (identifiers) which compose the reasons for equality (i.e. the indexes to the equality on the functional calculus), it is reasonable to expect that the following rules are taken for granted: *reflexivity*, *symmetry* and *transitivity*.

Substitution without involving quantifiers. We know from logic programming, i.e. from the theory of unification, that substitution can take place even when no quantifier is involved. This is justified when, for some reason a certain referent can replace another under some condition for identifying the one with the other.

Now, what would be counterpart to such a 'quantifier-less' notion of substitution in a labelled natural deduction system. Without the appropriate means of handling equality (definitional and propositional) we would hardly be capable of finding such a counterpart. Having said all that, let us think of what we ought to do at a certain stage in a proof (deduction) where the following two premises would be at hand:

$$a =_g y : A \qquad \text{and} \qquad f(a) : P(a)$$

We have that a and y are equal ('identifiable') under some arbitrary sequence of equalities (rewrites) which we name g. We also have that the predicate formula $P(a)$ is labelled by a certain functional expression f which depends on a. Clearly, if a and y are 'identifiable', we would like to infer that P, being true of a, will also be true of y. So, we shall be happy in inferring (on the logical calculus) the formula $P(y)$. Now, given that we ought to compose the label of the conclusion out of a composition of the labels of the premises, what label should we insert alongside $P(y)$? Perhaps various good answers could be given here, but we shall choose one which is in line with our 'keeping record of what (relevant) data was used in a deduction'. We have already stated how much importance we attach to names of individuals, names of formula instances, and of course, what kind of deduction was performed (i.e. what kind of connective was introduced or eliminated). In this section we have also insisted on the importance of, not only 'classifying' the equalities, but also having variables for the kinds of equalities that may be used in a deduction. Let us then formulate our rule of 'quantifier-less' substitution as:

$$\frac{a =_g y : A \qquad f(a) : P(a)}{g(a, y) \cdot f(a) : P(y)}$$

which could be explained in words as follows: if a and y are 'identifiable' due to a certain g, and $f(a)$ is the evidence for $P(a)$, then let the composition of $g(a, y)$ (the label for the propositional equality between a and y) with $f(a)$ (the evidence for $P(a)$) be the evidence for $P(y)$.

By having this extra rule of substitution added to the system of rules of inference, we are able to validate one half of the so-called 'Leibniz's law', namely:

$$\forall x^A \forall y^A.(\mathtt{Id}_A(x,y) \to (P(x) \to P(y)))$$

The *LND* equational fragment. As we already mentioned, in the *LND* equational logic, the equations have an index (the *reason*) which keeps all proof steps. The *reasons* is defined by the kind of rule used in the proof and the equational axioms (*definitional equalities*) of the system. The rules are divided into the following classes: (i) general rules; (ii) subterm substitution rule; (iii) ξ- and μ-rules.

Since the *LND* system is based on the Curry–Howard isomorphism [17], terms represent proof constructions, thus proof transformations correspond to equalities between terms. In this way, the *LND* equational logic can deal with equalities between *LND* proofs. The proofs in the *LND* equational fragment which deals with equalities between deductions are built from the basic proof transformations for the *LND* system, given in [44, 42, 50]. These basic proof transformations form an equational system, composed by *definitional equalities* (β, η and ζ).

General rules.

Definition 3.1 (equation) *An equation in LND_{EQ} is of the form:*

$$s =_r t : A$$

where s and t are terms, r is the identifier for the rewrite reason*, and A is the type (formula).*

Definition 3.2 (system of equations) *A system of equations S is a set of equations:*

$$\{s_1 =_{r_1} t_1 : A_1, \ldots, s_n =_{r_n} t_n : A_n\}$$

where r_i is the rewrite reason *identifier for the ith equation in S.*

Definition 3.3 (rewrite reason) *Given a system of equations S and an equation $s =_r t : A$, if $S \vdash s =_r t : A$, i.e. there is a deduction/computation of the equation starting from the equations in S, then the* rewrite reason *r is built up from:*

(i) *the constants for* rewrite reasons*: $\{ \rho, \beta, \eta, \zeta \}$;*

(ii) the r_i*'s;*

using the substitution operations:

(iii) $\mathtt{sub_L}$*;*

(iv) $\mathtt{sub_R}$*;*

and the operations for building new rewrite reasons*:*

(v) σ, τ, ξ, μ.

Definition 3.4 (general rules of equality) *The general rules for equality (reflexivity, symmetry and transitivity) are defined as follows:*

$$\begin{array}{ccc} \textit{reflexivity} & \textit{symmetry} & \textit{transitivity} \\ \dfrac{x : A}{x =_\rho x : A} & \dfrac{x =_t y : A}{y =_{\sigma(t)} x : A} & \dfrac{x =_t y : A \qquad y =_u z : A}{x =_{\tau(t,u)} z : A} \end{array}$$

The "subterm substitution" rule. Equational logic as usually presented has the following inference rule of substitution:

$$\frac{s = t}{s\theta = t\theta}$$

where θ is a substitution.

Note that the substitution θ "appeared" in the conclusion of the rule. As rightly pointed out by Le Chenadec in [9], from the viewpoint of the subformula property (objects in the conclusion of some inference should be subobjects of the premises), this rule is unsatisfactory. He then defines two rules:

$$IL\frac{M = N \quad C[N] = O}{C[M] = O} \qquad IR\frac{M = C[N] \quad N = O}{M = C[O]}$$

where M, N and O are terms and the context $C[\_]$ is adopted in order to distinguish subterms.

In [29] we have formulated an inference rule called "subterm substitution" which deals in a *explicit way*[5] with substitutions. In fact, the *LND* [44, 50] can be seen as an enriched system which brings to the object language terms, and now substitutions.

[5]In [44] de Queiroz and Gabbay recall Girard, who describes the intimate connections between constructivity and explicitation, and claim that "...one of the aims of inserting a label alongside formulas (accounting for the steps made to arrive at each particular point in the deduction) is exactly that of making *explicit* the use of formulas (and instances of formulas and individuals) throughout a deduction ..."

Definition 3.5 (subterm substitution) *The rule of "subterm substitution" is framed as follows:*

$$\frac{x =_r \mathcal{C}[y] : A \quad y =_s u : A'}{x =_{\mathtt{sub_L}(r,s)} \mathcal{C}[u] : A} \qquad \frac{x =_r w : A' \quad \mathcal{C}[w] =_s u : A}{\mathcal{C}[x] =_{\mathtt{sub_R}(r,s)} u : A}$$

where $\mathcal{C}$ is the context in which the subterm detached by '[ ]' appears and A' could be a subdomain of A, equal to A or disjoint to A. ($\mathcal{C}[u]$ is the result of replacing all occurrences of y by u in C.[6]

The symbols $\mathtt{sub_L}$ and $\mathtt{sub_R}$ denote in which side (L – *left* or R – *right*) is the premiss that contains the subterm to be substituted.

Note that the transitivity rule previously defined can be seen as a special case for this rule when $A' = A$ and the context $\mathcal{C}$ is empty.

The ξ- and μ-rules. In the Curry–Howard "formulae-as-types" interpretation [17], the ξ-rule[7] states when two *canonical* elements are equal, and the μ-rule[8] states when two *noncanonical* elements are equal. So, each introduction rule for the *LND* system has associated to it a ξ-rule and each elimination rule has a related μ-rule. For instance, the ξ-rule and μ-rule for the connective $\wedge$ are defined as follows:

$$\frac{x =_u y : A \qquad s : B}{\langle x, s\rangle =_{\xi_1(u)} \langle y, s\rangle : A \wedge B} \qquad \frac{x : A \qquad s =_v t : B}{\langle x, s\rangle =_{\xi_2(v)} \langle x, t\rangle : A \wedge B}$$

[6]We should like to thank an anonymous referee who pointed out the ambiguity which would remain in case this condition is not made clear.

[7]The ξ-rule is the formal counterpart to Bishop's constructive principle of definition of a set [8] (page 2) which says: "To define a set we prescribe, at least implicitly, what we have (the constructing intelligence) must to do in order to construct an element of the set, and what we must do to show that two elements of the set are equal." Cf. also [8] (page 12) Bishop defines a product of set as "The *cartesian product*, or simply *product*, $X \equiv X_1 \times \ldots \times X_n$ of sets $X_1, X_2, \ldots, X_n$ is defined to be the set of all ordered n-tuples $(x_1, \ldots, x_n)$ and $(y_1, \ldots, y_n)$ of X are equal if the coordinates x_i and y_i are equal for each i." See also [25] (p.8): "... a set A is defined by prescribing how a canonical element of A is formed as well as how two equal canonical elements of A are formed." We also know from the theory of Lambda Calculus the definition of ξ-rule, see e.g. [5] (pp. 23 and 78): "$\xi : M = N \Rightarrow \lambda x.M = \lambda x.N$"

[8]The μ-rule is also defined in the theory of Lambda Calculus, see e.g. [26]: "The equational axioms and inference rules are as follows, where $[N/x]M$ denotes substitution of N for x in M. ...

$$(\mu) \qquad \frac{\Gamma \rhd M_1 = M_2 : \sigma \Rightarrow \tau \quad \Gamma \rhd N_1 = N_2 : \sigma}{\Gamma \rhd M_1 N_1 = M_2 N_2 : \tau}\text{"}$$

and is divided into two equalities μ and ν in [13] (p.66):

$$(\mu)\frac{M = M'}{NM = NM'} \qquad (\nu)\frac{M = M'}{MN = M'N}$$

$$\frac{x =_r y : A \wedge B}{FST(x) =_{\mu_1(r)} FST(y) : A} \qquad \frac{x =_r y : A \wedge B}{SND(x) =_{\mu_2(r)} SND(y) : B}$$

In the Appendix we give a list, for each type-forming operator, of rules of definition, in the style of Bishop, Curry-Howard, and Martin-Löf, of what elements the type contains, and when two elements are defined to be equal.

Term rewriting system for *LND* with equality. In [30] we have proved termination and confluence for the rewriting system arising out of the proof rules given for the proposed natural deduction system for equality.

The idea is to analyse all possible occurrences of redundancies in proofs which involve the rules of rewriting, and the most obvious case is the nested application of the rule of symmetry. But there are a number of cases when the application of rewriting rules is redundant, but which is not immediately obvious that there is a redundancy. Take, for instance, the following case:

Definition 3.6 (reductions involving τ)

$$\frac{x =_r y : A \quad y =_{\sigma(r)} x : A}{x =_{\tau(r,\sigma(r))} x : A} \quad \rhd_{tr} \quad x =_\rho x : A$$

$$\frac{y =_{\sigma(r)} x : A \quad x =_r y : A}{y =_{\tau(\sigma(r),r)} y : A} \quad \rhd_{tsr} \quad y =_\rho y : A$$

$$\frac{u =_r v : A \quad v =_\rho v : A}{u =_{\tau(r,\rho)} v : A} \quad \rhd_{trr} \quad u =_r v : A$$

$$\frac{u =_\rho u : A \quad u =_r v : A}{u =_{\tau(\rho,r)} v : A} \quad \rhd_{tlr} \quad u =_r v : A$$

Associated rewriting rule over the reason:
$\tau(r, \sigma(r)) \rhd_{tr} \rho$
$\tau(\sigma(r), r) \rhd_{tsr} \rho$
$\tau(r, \rho) \rhd_{trr} r$
$\tau(\rho, r) \rhd_{tlr} r$.

Below is another less obvious case of occurrence of redundancy:

Definition 3.7

$$\frac{a : A \quad \dfrac{\begin{matrix}[x : A] \\ \vdots \\ b(x) =_r g(x) : B\end{matrix}}{\lambda x.b(x) =_{\xi(r)} \lambda x.g(x) : A \to B} \to\text{-}intr}{APP(\lambda x.b(x), a) =_{\nu(\xi(r))} APP(\lambda x.g(x), a) : B} \to\text{-}elim$$

$$\triangleright_{mxl} \quad \begin{array}{c} a : A \\ b(a/x) =_r g(a/x) : B \end{array}$$

Associated rewriting rule:
$\nu(\xi(r)) \triangleright_{mxl} r$.

As an example:

Example 3.8

$$\dfrac{\dfrac{x =_r y : A}{i(x) =_{\xi_1(r)} i(y) : A + B} \quad \dfrac{\dfrac{x =_r y : A}{y =_{\sigma(r)} x : A}}{i(y) =_{\xi_1(\sigma(r))} i(x) : A + B}}{i(x) =_{\tau(\xi_1(r), \xi_1(\sigma(r)))} i(x) : A + B}$$

$$\triangleright_{tr} \quad \dfrac{x =_r y : A}{x =_{\xi_1(r)} y : A + B}$$

Associated rewriting:
$\tau(\xi_1(r), \xi_1(\sigma(r))) \triangleright_{tr} \xi_1(r)$.

Definition 3.9 (reductions involving ρ and σ)

$$\dfrac{x =_\rho x : A}{x =_{\sigma(\rho)} x : A} \quad \triangleright_{sr} \quad x =_\rho x : A$$

$$\dfrac{\dfrac{x =_r y : A}{y =_{\sigma(r)} x : A}}{x =_{\sigma(\sigma(r))} y : A} \quad \triangleright_{sr} \quad x =_r y : A$$

Associated rewritings:
$\sigma(\rho) \triangleright_{sr} \rho$
$\sigma(\sigma(r)) \triangleright_{sr} r$

Definition 3.10 (reductions involving τ)

$$\dfrac{x =_r y : A \quad y =_{\sigma(r)} x : A}{x =_{\tau(r, \sigma(r))} x : A} \quad \triangleright_{tr} \quad x =_\rho x : A$$

$$\dfrac{y =_{\sigma(r)} x : A \quad x =_r y : A}{y =_{\tau(\sigma(r), r)} y : A} \quad \triangleright_{tsr} \quad y =_\rho y : A$$

$$\dfrac{u =_r v : A \quad v =_\rho v : A}{u =_{\tau(r, \rho)} v : A} \quad \triangleright_{trr} \quad u =_r v : A$$

$$\dfrac{u =_\rho u : A \quad u =_r v : A}{u =_{\tau(\rho, r)} v : A} \quad \triangleright_{tlr} \quad u =_r v : A$$

Associated rewritings:
$\tau(r, \sigma(r)) \rhd_{tr} \rho$
$\tau(\sigma(r), r) \rhd_{tsr} \rho$
$\tau(r, \rho) \rhd_{tsr} r$
$\tau(\rho, r) \rhd_{tlr} r$

Note that the first two reductions identify the case in which a *reason* which is part of a rewrite sequence meets its inverse.

These reductions can be generalized to transformations where the reasons r and $\sigma(r)$ (transf. 1 and 2) and r and ρ (transf. 3 and 4) appear in some context, as illustrated by the following example:

Example 3.11

$$\dfrac{\dfrac{x =_r y : A}{i(r) =_{\xi_1(r)} i(y) : A + B} \quad \dfrac{\dfrac{x =_r y : A}{y =_{\sigma(r)} x : A}}{i(y) =_{\xi_1(\sigma(r))} i(x) : A + B}}{i(x) =_{\tau(\xi_1(r), \xi_1(\sigma(r)))} i(x) : A + B}$$

$$\rhd_{tr} \quad \dfrac{\dfrac{x : A}{x =_\rho x : A}}{i(x) =_{\xi_1(\rho)} i(x) : A + B}$$

Associated rewriting:
$\tau(\xi_1(r), \xi_1(\sigma(r))) \rhd_{tr} \xi_1(\rho)$

For the general context $\mathcal{C}[\ ]$:
Associated rewritings:
$\tau(\mathcal{C}[r], \mathcal{C}[\sigma(r)]) \rhd_{tr} \mathcal{C}[\rho]$
$\tau(\mathcal{C}[\sigma(r)], \mathcal{C}[r]) \rhd_{tsr} \mathcal{C}[\rho]$
$\tau(\mathcal{C}[r], \mathcal{C}[\rho]) \rhd_{trr} \mathcal{C}[r]$
$\tau(\mathcal{C}[\rho], \mathcal{C}[r]) \rhd_{tlr} \mathcal{C}[r]$

Definition 3.12 (substitution rules)

$$\dfrac{u =_r \mathcal{C}[x] : A \quad x =_\rho x : A'}{u =_{\mathtt{sub}_{\mathtt{L}}(r,\rho)} \mathcal{C}[x] : A} \quad \rhd_{slr} \quad u =_r \mathcal{C}[x] : A$$

$$\dfrac{x =_\rho x : A' \quad \mathcal{C}[x] =_r z : A}{\mathcal{C}[x] =_{\mathtt{sub}_{\mathtt{R}}(\rho,r)} z : A} \quad \rhd_{srr} \quad \mathcal{C}[x] =_r z : A$$

$$\dfrac{\dfrac{z =_s \mathcal{C}[y] : A \quad y =_r w : A'}{z =_{\mathtt{sub}_{\mathtt{L}}(s,r)} \mathcal{C}[w] : D} \quad \dfrac{y =_r w : A'}{w =_{\sigma(r)} y : D'}}{z =_{\mathtt{sub}_{\mathtt{L}}(\mathtt{sub}_{\mathtt{L}}(s,r),\sigma(r))} \mathcal{C}[y] : A} \quad \rhd_{sls} \quad z =_s \mathcal{C}[y] : A$$

$$\cfrac{\cfrac{z =_s \mathcal{C}[y] : A \quad y =_r w : A'}{z =_{\mathtt{sub_L}(s,r)} \mathcal{C}[w] : A} \qquad \cfrac{y =_r w : A'}{w =_{\sigma(r)} y : A'}}{z =_{\mathtt{sub_L}(\mathtt{sub_L}(s,r),\sigma(r))} \mathcal{C}[y] : A} \triangleright_{slss} \; z =_s \mathcal{C}[y] : A$$

$$\cfrac{x =_s w : A' \qquad \cfrac{\cfrac{x =_s w : A'}{w =_{\sigma(s)} x : A'} \quad \mathcal{C}[x] =_r z : A}{\mathcal{C}[w] =_{\mathtt{sub_R}(\sigma(s),r)} z : A}}{\mathcal{C}[x] =_{\mathtt{sub_R}(s,\mathtt{sub_R}(\sigma(s),r))} z : A} \triangleright_{srs} \; \mathcal{C}[x] =_r z : A$$

$$\cfrac{\cfrac{x =_s w : A'}{w =_{\sigma(s)} x : A'} \qquad \cfrac{x =_s w : A' \quad \mathcal{C}[w] =_r z : A}{\mathcal{C}[x] =_{\mathtt{sub_R}(s,r)} z : A}}{\mathcal{C}[w] =_{\mathtt{sub_R}(\sigma(s),\mathtt{sub_R}(s,r))} z : A} \triangleright_{srrr} \; \mathcal{C}[w] =_r z : A$$

Associated rewritings:
$\mathtt{sub_L}(\mathcal{C}[r], \mathcal{C}[\rho]) \triangleright_{slr} \mathcal{C}[r]$
$\mathtt{sub_R}(\mathcal{C}[\rho], \mathcal{C}[r]) \triangleright_{srr} \mathcal{C}[r]$
$\mathtt{sub_L}(\mathtt{sub_L}(s, \mathcal{C}[r]), \mathcal{C}[\sigma(r)]) \triangleright_{sls} s$
$\mathtt{sub_L}(\mathtt{sub_L}(s, \mathcal{C}[\sigma(r)]), \mathcal{C}[r]) \triangleright_{slss} s$
$\mathtt{sub_R}(s, \mathtt{sub_R}(\mathcal{C}[\sigma(s)], r)) \triangleright_{srs} r$
$\mathtt{sub_R}(\mathcal{C}[\sigma(s)], \mathtt{sub_R}(\mathcal{C}[s], r)) \triangleright_{srrr} r$

Definition 3.13

β_{rewr}-$\times$-*reduction*

$$\cfrac{\cfrac{x =_r y : A \qquad z : B}{\langle x, z\rangle =_{\xi_1(r)} \langle y, z\rangle : A \times B} \times\textit{-intr}}{FST(\langle x, z\rangle) =_{\mu_1(\xi_1(r))} FST(\langle y, z\rangle) : A} \times\textit{-elim} \qquad \triangleright_{mx2l} \quad x =_r y : A$$

$$\cfrac{\cfrac{x =_r x' : A \qquad y =_s z : B}{\langle x, y\rangle =_{\xi_\wedge(r,s)} \langle x', z\rangle : A \times B} \times\textit{-intr}}{FST(\langle x, y\rangle) =_{\mu_1(\xi_\wedge(r,s))} FST(\langle x', z\rangle) : A} \times\textit{-elim} \qquad \triangleright_{mx2l} \quad x =_r x' : A$$

$$\cfrac{\cfrac{x =_r y : A \qquad z =_s w : B}{\langle x, z\rangle =_{\xi_\wedge(r,s)} \langle y, w\rangle : A \times B} \times\textit{-intr}}{SND(\langle x, z\rangle) =_{\mu_2(\xi_\wedge(r,s))} SND(\langle y, w\rangle) : B} \times\textit{-elim} \qquad \triangleright_{mx2r} \quad z =_s w : B$$

$$\cfrac{\cfrac{x : A \qquad z =_s w : B}{\langle x, z\rangle =_{\xi_2(s)} \langle x, w\rangle : A \times B} \times\textit{-intr}}{SND(\langle x, z\rangle) =_{\mu_2(\xi_2(s))} SND(\langle x, w\rangle) : B} \times\textit{-elim}$$

$$\triangleright_{mx2r} \quad z =_s w : B$$

Associated rewritings:
$\mu_1(\xi_1(r)) \triangleright_{mx2l1} r$
$\mu_1(\xi_\wedge(r,s)) \triangleright_{mx2l2} r$
$\mu_2(\xi_\wedge(r,s)) \triangleright_{mx2r1} s$
$\mu_2(\xi_2(s)) \triangleright_{mx2r2} s$

β_{rewr}*-+-reduction*

$$\frac{\dfrac{a =_r a' : A}{i(a) =_{\xi_1(r)} i(a') : A + B}\ +\ \textit{-intr} \quad \begin{matrix}[x : A]\\ f(x) =_s k(x) : C\end{matrix} \quad \begin{matrix}[y : B]\\ g(y) =_u h(y) : C\end{matrix}}{D(i(a), \acute{x}f(x), \acute{y}g(y)) =_{\mu(\xi_1(r),s,u)} D(i(a'), \acute{x}k(x), \acute{y}h(y)) : C}\ + \textit{-elim}$$

$$\triangleright_{mx3l} \quad \begin{matrix} a =_r a' : A \\ f(a/x) =_s k(a'/x) : C \end{matrix}$$

$$\frac{\dfrac{b =_r b' : B}{j(b) =_{\xi_2(r)} j(b') : A + B}\ + \textit{-intr} \quad \begin{matrix}[x : A]\\ f(x) =_s k(x) : C\end{matrix} \quad \begin{matrix}[y : B]\\ g(y) =_u h(y) : C\end{matrix}}{D(j(b), \acute{x}f(x), \acute{y}g(y)) =_{\mu(\xi_2(r),s,u)} D(j(b'), \acute{x}k(x), \acute{y}h(y)) : C}\ + \textit{-elim}$$

$$\triangleright_{mx3r} \quad \begin{matrix} b =_s b' : B \\ g(b/y) =_u h(b'/y) : C \end{matrix}$$

Associated rewritings:
$\mu(\xi_1(r), s, u) \triangleright_{mx3l} s$
$\mu(\xi_2(r), s, u) \triangleright_{mx3r} u$

β_{rewr}-Π*-reduction*

$$\frac{a : A \quad \dfrac{\begin{matrix}[x : A]\\ f(x) =_r g(x) : B(x)\end{matrix}}{\lambda x.f(x) =_{\xi(r)} \lambda x.g(x) : \Pi x : A.B(x)}}{APP(\lambda x.f(x), a) =_{\nu(\xi(r))} APP(\lambda x.g(x), a) : B(a)}$$

$$\triangleright_{mxl} \quad \begin{matrix} a : A \\ f(a/x) =_r g(a/x) : B(a) \end{matrix}$$

Associated rewriting:
$\nu(\xi(r)) \triangleright_{mxl} r$

β_{rewr}-Σ*-reduction*

$$\frac{\dfrac{a =_r a' : A \quad f(a) : B(a)}{\varepsilon x.(f(x), a) =_{\xi_1(r)} \varepsilon x.(f(x), a') : \Sigma x : A.B(x)} \quad \begin{matrix}[t : A, g(t) : B(t)]\\ d(g,t) =_s h(g,t) : C\end{matrix}}{E(\varepsilon x.(f(x), a), \acute{g}\acute{t}d(g,t)) =_{\mu(\xi_1(r),s)} E(\varepsilon x.(f(x), a'), \acute{g}\acute{t}h(g,t)) : C}$$

$$\triangleright_{mxr} \quad \begin{matrix} a =_r a' : A \quad f(a) : B(a) \\ d(f/g, a/t) =_s h(f/g, a'/t) : C \end{matrix}$$

$$\frac{\dfrac{a : A \quad f(a) =_r i(a) : B(a)}{\varepsilon x.(f(x),a) =_{\xi_2(r)} \varepsilon x.(i(x),a) : \Sigma x : A.B(x)} \quad \begin{array}{c}[t : A, g(t) : B(t)] \\ d(g,t) =_s h(g,t) : C\end{array}}{E(\varepsilon x.(f(x),a), \acute{g}\acute{t}d(g,t)) =_{\mu(\xi_2(r),s)} E(\varepsilon x.(i(x),a), \acute{g}\acute{t}h(g,t)) : C}$$

$$\triangleright_{mxl} \quad \begin{array}{c} a : A \quad f(a) =_r i(a) : B(a) \\ d(f/g, a/t) =_s h(i/g, a/t) : C \end{array}$$

Associated rewritings:
$\mu(\xi_1(r), s) \triangleright_{mxr} s$
$\mu(\xi_2(r), s) \triangleright_{mxl} s$

Definition 3.14 (η_{rewr})
η_{rewr}- $\times$-*reduction*

$$\frac{\dfrac{x =_r y : A \times B}{FST(x) =_{\mu_1(r)} FST(y) : A} \times\text{-}elim \quad \dfrac{x =_r y : A \times B}{SND(x) =_{\mu_2(r)} SND(y) : B} \times\text{-}elim}{\langle FST(x), SND(x)\rangle =_{\xi(\mu_1(r),\mu_2(r))} \langle FST(y), SND(y)\rangle : A \times B} \times\text{-}intr$$

$$\triangleright_{mx} \quad x =_r y : A \times B$$

η_{rewr}- $+$-*reduction*

$$\frac{c =_t d : A + B \quad \dfrac{[a_1 =_r a_2 : A]}{i(a_1) =_{\xi_1(r)} i(a_2) : A + B} +\text{-}intr \quad \dfrac{[b_1 =_s b_2 : B]}{j(b_1) =_{\xi_2(s)} j(b_2) : A + B} +\text{-}intr}{D(c, \acute{a_1}i(a_1), \acute{b_1}j(b_1)) =_{\mu(t,\xi_1(r),\xi_2(s))} D(d, \acute{a_2}i(a_2), \acute{b_2}j(b_2))} +\text{-}elim$$

$$\triangleright_{mxx} \quad c =_t d : A + B$$

Π-η_{rewr}-*reduction*

$$\frac{\dfrac{[t : A] \quad c =_r d : \Pi x : A.B(x)}{APP(c,t) =_{\nu(r)} APP(d,t) : B(t)} \Pi\text{-}elim}{\lambda t.APP(c,t) =_{\xi(\nu(r))} \lambda t.APP(d,t) : \Pi t : A.B(t)} \Pi\text{-}intr$$

$$\triangleright_{xmr} \quad c =_r d : \Pi x : A.B(x)$$

where c and d do not depend on x.

Σ-η_{rewr}-*reduction*

$$\frac{c =_s b : \Sigma x : A.B(x) \quad \dfrac{[t : A] \quad [g(t) =_r h(t) : B(t)]}{\varepsilon y.(g(y),t) =_{\xi_2(r)} \varepsilon y.(h(y),t) : \Sigma y : A.B(y)} \Sigma\text{-}intr}{E(c, \acute{g}\acute{t}\varepsilon y.(g(y),t)) =_{\mu(s,\xi_2(r))} E(b, \acute{h}\acute{t}\varepsilon y.(h(y),t)) : \Sigma y : A.B(y)} \Sigma\text{-}elim$$

$$\triangleright_{mxlr} \quad c =_s b : \Sigma x : A.B(x)$$

Associated rewritings:
$\xi(\mu_1(r), \mu_2(r)) \triangleright_{mx} r$
$\mu(t, \xi_1(r), \xi_2(s)) \triangleright_{mxx} t$
$\xi(\nu(r)) \triangleright_{xmr} r$
$\mu(s, \xi_2(r)) \triangleright_{mxlr} s$

Definition 3.15 (σ **and** τ)

$$\frac{\dfrac{x =_r y : A \quad y =_s w : A}{x =_{\tau(r,s)} w : A}}{w =_{\sigma(\tau(r,s))} x : A} \quad \triangleright_{stss} \quad \frac{\dfrac{y =_s w : A}{w =_{\sigma(s)} y : A} \quad \dfrac{x =_r y : A}{y =_{\sigma(r)} x : A}}{w =_{\tau(\sigma(s),\sigma(r))} x : A}$$

Associated rewriting:
$\sigma(\tau(r,s)) \triangleright_{stss} \tau(\sigma(s),\sigma(r))$

Definition 3.16 (σ **and** $\mathtt{sub}$)

$$\frac{\dfrac{x =_r \mathcal{C}[y] : A \quad y =_s w : A'}{x =_{\mathtt{sub_L}(r,s)} \mathcal{C}[w] : A}}{\mathcal{C}[w] =_{\sigma(\mathtt{sub_L}(r,s))} x : A} \quad \triangleright_{ssbl} \quad \frac{\dfrac{y =_s w : A'}{w =_{\sigma(s)} y : A'} \quad \dfrac{x =_r \mathcal{C}[y] : A}{\mathcal{C}[y] =_{\sigma(r)} x : A}}{\mathcal{C}[w] =_{\mathtt{sub_R}(\sigma(s),\sigma(r))} x : A}$$

$$\frac{\dfrac{x =_r y : A' \quad \mathcal{C}[y] =_s w : A}{\mathcal{C}[x] =_{\mathtt{sub_R}(r,s)} w : A}}{w =_{\sigma(\mathtt{sub_R}(r,s))} \mathcal{C}[x] : D} \quad \triangleright_{ssbr} \quad \frac{\dfrac{\mathcal{C}[y] =_s w : A}{w =_{\sigma(s)} \mathcal{C}[y] : A} \quad \dfrac{x =_r y : A'}{y =_{\sigma(r)} x : A'}}{w =_{\mathtt{sub_L}(\sigma(s),\sigma(r))} \mathcal{C}[x] : A}$$

Associated rewritings:
$\sigma(\mathtt{sub_L}(r,s)) \triangleright_{ssbl} \mathtt{sub_R}(\sigma(s),\sigma(r))$
$\sigma(\mathtt{sub_R}(r,s)) \triangleright_{ssbr} \mathtt{sub_L}(\sigma(s),\sigma(r))$

Definition 3.17 (σ **and** ξ)

$$\frac{\dfrac{x =_r y : A}{i(x) =_{\xi_1(r)} i(y) : A + B}}{i(y) =_{\sigma(\xi_1(r))} i(x) : A + B} \quad \triangleright_{sx} \quad \frac{\dfrac{x =_r y : A}{y =_{\sigma(r)} x : A}}{i(y) =_{\xi_1(\sigma(r))} i(x) : A + B}$$

$$\frac{\dfrac{x =_r y : A \quad z =_s w : B}{\langle x, z\rangle =_{\xi(r,s)} \langle y, w\rangle : A \times B}}{\langle y, w\rangle =_{\sigma(\xi(r,s))} \langle x, z\rangle : A \times B} \quad \triangleright_{sxss} \quad \frac{\dfrac{x =_r y : A}{y =_{\sigma(r)} x : A} \quad \dfrac{z =_s w : B}{w =_{\sigma(s)} z : B}}{\langle y, w\rangle =_{\xi(\sigma(r),\sigma(s))} \langle x, z\rangle : A \times B}$$

$$\frac{\dfrac{\begin{array}{c}[x : A] \\ f(x) =_s g(x) : B(x)\end{array}}{\lambda x.f(x) =_{\xi(s)} \lambda x.g(x) : \Pi x : A.B(x)}}{\lambda x.g(x) =_{\sigma(\xi(s))} \lambda x.f(x) : \Pi x : A.B(x)} \quad \triangleright_{smss} \quad \frac{\dfrac{\begin{array}{c}[x : A] \\ f(x) =_s g(x) : B(x)\end{array}}{g(x) =_{\sigma(s)} f(x) : B(x)}}{\lambda x.g(x) =_{\xi(\sigma(s))} \lambda x.f(x) : \Pi x : A.B(x)}$$

Associated rewritings:
$\sigma(\xi(r)) \triangleright_{sx} \xi(\sigma(r))$
$\sigma(\xi(r,s)) \triangleright_{sxss} \xi(\sigma(r),\sigma(s))$
$\sigma(\xi(s) \triangleright_{smss} \xi(\sigma(s))$

Definition 3.18 (σ and μ)

$$\frac{\dfrac{x =_r y : A \times B}{FST(x) =_{\mu_1(r)} FST(y) : A}}{FST(y) =_{\sigma(\mu_1(r))} FST(x) : A} \quad \rhd_{sm} \quad \frac{\dfrac{x =_r y : A \times B}{y =_{\sigma(r)} x : A \times B}}{FST(y) =_{\mu_1(\sigma(r))} FST(x) : A}$$

$$\frac{\dfrac{x =_r y : A \times B}{SND(x) =_{\mu_2(r)} SND(y) : A}}{SND(y) =_{\sigma(\mu_2(r))} SND(x) : A} \quad \rhd_{sm} \quad \frac{\dfrac{x =_r y : A \times B}{y =_{\sigma(r)} x : A \times B}}{SND(y) =_{\mu_2(\sigma(r))} SND(x) : A}$$

$$\frac{\dfrac{x =_s y : A \quad f =_r g : A \to B}{APP(f,x) =_{\mu(s,r)} APP(g,y) : B}}{APP(g,y) =_{\sigma(\mu(s,r))} APP(f,x) : B}$$

$$\rhd_{smss} \quad \frac{\dfrac{x =_s y : A}{y =_{\sigma(s)} x : A} \quad \dfrac{f =_r g : A \to B}{g =_{\sigma(r)} f : A \to B}}{APP(g,y) =_{\mu(\sigma(s),\sigma(r))} APP(f,x) : B}$$

$$\frac{\dfrac{x =_r y : A + B \quad \begin{matrix}[s : A] \\ \vdots \\ d(s) =_u f(s) : C\end{matrix} \quad \begin{matrix}[t : B] \\ \vdots \\ e(t) =_v g(t) : C\end{matrix}}{D(x, \acute{s}d(s), \acute{t}e(t)) =_{\mu(r,u,v)} D(y, \acute{s}f(s), \acute{t}g(t)) : C}}{D(y, \acute{s}f(s), \acute{t}g(t)) : C =_{\sigma(\mu(r,u,v))} D(x, \acute{s}d(s), \acute{t}e(t)) : C}$$

$$\rhd_{smsss} \frac{\dfrac{x =_r y : A + B}{y =_{\sigma(r)} x : A + B} \quad \dfrac{\begin{matrix}[s : A] \\ d(s) =_u f(s) : C\end{matrix}}{f(s) =_{\sigma(u)} d(s) : C} \quad \dfrac{\begin{matrix}[t : B] \\ e(t) =_v g(t) : C\end{matrix}}{g(t) =_{\sigma(v)} e(t) : C}}{D(y, \acute{s}f(s), \acute{t}g(t)) =_{\mu(\sigma(r),\sigma(u),\sigma(v))} D(x, \acute{s}d(s), \acute{t}e(t)) : C}$$

$$\frac{\dfrac{e =_s b : \Sigma x : A.B(x) \quad \begin{matrix}[t : A,\ g(t) : B(t)] \\ d(g,t) =_r f(g,t) : C\end{matrix}}{E(e, \acute{g}\acute{t}d(g,t)) =_{\mu(s,r)} E(b, \acute{g}\acute{t}f(g,t)) : C}}{E(b, \acute{g}\acute{t}f(g,t)) =_{\sigma(\mu(s,r))} E(e, \acute{g}\acute{t}d(g,t)) : C}$$

$$\rhd_{smss} \frac{\dfrac{e =_s b : \Sigma x : A.B(x)}{b =_{\sigma(s)} e : \Sigma x : A.B(x)} \quad \dfrac{\begin{matrix}[t : A,\ g(t) : B(t)] \\ d(g,t) =_r f(g,t) : C\end{matrix}}{f(g,t) =_{\sigma(r)} d(g,t) : C}}{E(b, \acute{g}\acute{t}f(g,t)) =_{\mu(\sigma(s),\sigma(r))} E(e, \acute{g}\acute{t}d(g,t)) : C}$$

Associated rewritings:
$\sigma(\mu_1(r)) \rhd_{sm} \mu_1(\sigma(r))$
$\sigma(\mu_2(r)) \rhd_{sm} \mu_2(\sigma(r))$
$\sigma(\mu(s,r)) \rhd_{smss} \mu(\sigma(s), \sigma(r))$
$\sigma(\mu(r,u,v)) \rhd_{smsss} \mu(\sigma(r), \sigma(u), \sigma(v))$

Definition 3.19 (τ **and** $\mathtt{sub}$)

$$\dfrac{\dfrac{x =_r \mathcal{C}[y] : A \quad y =_s w : A'}{x =_{\mathtt{sub}_{\mathtt{L}}(r,s)} \mathcal{C}[w] : A} \quad \mathcal{C}[w] =_t z : A}{x =_{\tau(\mathtt{sub}_{\mathtt{L}}(r,s),t)} z : A} \triangleright_{tsbll} \dfrac{x =_r \mathcal{C}[y] : A \quad \dfrac{y =_s w : A' \quad \mathcal{C}[w] =_t z : A}{\mathcal{C}[y] =_{\mathtt{sub}_{\mathtt{R}}(s,t)} z : A}}{x =_{\tau(r,\mathtt{sub}_{\mathtt{R}}(s,t))} z : A}$$

$$\dfrac{\dfrac{y =_s w : A \quad \mathcal{C}[w] =_t z : A}{\mathcal{C}[y] =_{\mathtt{sub}_{\mathtt{R}}(s,t)} z : A} \quad z =_u v : A}{\mathcal{C}[y] =_{\tau(\mathtt{sub}_{\mathtt{R}}(s,t),u)} v : A} \triangleright_{tsbrl} \dfrac{y =_s w : D' \quad \dfrac{\mathcal{C}[w] =_t z : A \quad z =_u v : A}{\mathcal{C}[w] =_{\tau(t,u)} v : A}}{\mathcal{C}[y] =_{\mathtt{sub}_{\mathtt{R}}(s,\tau(t,u))} v : A}$$

$$\dfrac{x =_r \mathcal{C}[z] : A \quad \dfrac{\mathcal{C}[z] =_\rho \mathcal{C}[z] : A \quad z =_s w : A'}{\mathcal{C}[z] =_{\mathtt{sub}_{\mathtt{L}}(\rho,s)} \mathcal{C}[w] : A}}{x =_{\tau(r,\mathtt{sub}_{\mathtt{L}}(\rho,s))} \mathcal{C}[w] : A} \triangleright_{tsblr} \dfrac{x =_r \mathcal{C}[z] : A \quad z =_s w : A'}{x =_{\mathtt{sub}_{\mathtt{L}}(r,s)} \mathcal{C}[w] : A}$$

$$\dfrac{x =_r \mathcal{C}[w] : A \quad \dfrac{w =_s z : A' \quad \mathcal{C}[z] =_\rho \mathcal{C}[z] : A}{\mathcal{C}[w] =_{\mathtt{sub}_{\mathtt{R}}(s,\rho)} \mathcal{C}[z] : A}}{x =_{\tau(r,\mathtt{sub}_{\mathtt{R}}(s,\rho))} \mathcal{C}[z] : A} \triangleright_{tsbrr} \dfrac{x =_r \mathcal{C}[w] : D \quad w =_s z : A'}{x =_{\mathtt{sub}_{\mathtt{L}}(r,s)} \mathcal{C}[z] : A}$$

Definition 3.20 (τ **and** τ)

$$\dfrac{\dfrac{x =_t y : A \quad y =_r w : A}{x =_{\tau(t,r)} w : A} \quad w =_s z : A}{x =_{\tau(\tau(t,r),s)} z : A} \triangleright_{tt} \dfrac{x =_t y : A \quad \dfrac{y =_r w : A \quad w =_s z : A}{y =_{\tau(r,s)} z : A}}{x =_{\tau(t,\tau(r,s))} z : A}$$

Associated rewritings:
$\tau(\mathtt{sub}_{\mathtt{L}}(r,s),t) \triangleright_{tsbll} \tau(r,\mathtt{sub}_{\mathtt{R}}(s,t))$
$\tau(\mathtt{sub}_{\mathtt{R}}(s,t),u)) \triangleright_{tsbrl} \mathtt{sub}_{\mathtt{R}}(s,\tau(t,u))$
$\tau(r,\mathtt{sub}_{\mathtt{L}}(\tau,s)) \triangleright_{tsblr} \mathtt{sub}_{\mathtt{L}}(r,s)$
$\tau(r,\mathtt{sub}_{\mathtt{R}}(s,\tau)) \triangleright_{tsbrr} \mathtt{sub}_{\mathtt{L}}(r,s)$
$\tau(\tau(t,r),s) \triangleright_{tt} \tau(t,\tau(r,s))$

By analysing all cases of redundant proofs involving equality we arrive at following set of associated rewriting rules. (NB. In the same way the definitional equalities (coming from rewriting rules) over terms of the λ calculus had to be given names – β, η, ξ, μ, etc. –, we will need to assign a name to each rewriting rule for terms representing computational paths. For the lack of a better naming choice at this point, we have tried to use abbreviations related to the operations involved.)

Definition 3.21 (*LND_{EQ}-TRS*)
1. $\sigma(\rho) \triangleright_{sr} \rho$
2. $\sigma(\sigma(r)) \triangleright_{ss} r$
3. $\tau(\mathcal{C}[r], \mathcal{C}[\sigma(r)]) \triangleright_{tr} \mathcal{C}[\rho]$
4. $\tau(\mathcal{C}[\sigma(r)], \mathcal{C}[r]) \triangleright_{tsr} \mathcal{C}[\rho]$
5. $\tau(\mathcal{C}[r], \mathcal{C}[\rho]) \triangleright_{rrr} \mathcal{C}[r]$
6. $\tau(\mathcal{C}[\rho], \mathcal{C}[r]) \triangleright_{lrr} \mathcal{C}[r]$
7. $\mathtt{sub_L}(\mathcal{C}[r], \mathcal{C}[\rho]) \triangleright_{slr} \mathcal{C}[r]$
8. $\mathtt{sub_R}(\mathcal{C}[\rho], \mathcal{C}[r]) \triangleright_{srr} \mathcal{C}[r]$
9. $\mathtt{sub_L}(\mathtt{sub_L}(s, \mathcal{C}[r]), \mathcal{C}[\sigma(r)]) \triangleright_{sls} s$
10. $\mathtt{sub_L}(\mathtt{sub_L}(s, \mathcal{C}[\sigma(r)]), \mathcal{C}[r]) \triangleright_{slss} s$
11. $\mathtt{sub_R}(\mathcal{C}[s], \mathtt{sub_R}(\mathcal{C}[\sigma(s)], r)) \triangleright_{srs} r$
12. $\mathtt{sub_R}(\mathcal{C}[\sigma(s)], \mathtt{sub_R}(\mathcal{C}[s], r)) \triangleright_{srrr} r$
13. $\mu_1(\xi_1(r)) \triangleright_{mx2l1} r$
14. $\mu_1(\xi_\wedge(r, s)) \triangleright_{mx2l2} r$
15. $\mu_2(\xi_\wedge(r, s)) \triangleright_{mx2r1} s$
16. $\mu_2(\xi_2(s)) \triangleright_{mx2r2} s$
17. $\mu(\xi_1(r), s, u) \triangleright_{mx3l} s$
18. $\mu(\xi_2(r), s, u) \triangleright_{mx3r} u$
19. $\nu(\xi(r)) \triangleright_{mxl} r$
20. $\mu(\xi_2(r), s) \triangleright_{mxr} s$
21. $\xi(\mu_1(r), \mu_2(r)) \triangleright_{mx} r$
22. $\mu(t, \xi_1(r), \xi_2(s)) \triangleright_{mxx} t$
23. $\xi(\nu(r)) \triangleright_{xmr} r$
24. $\mu(s, \xi_2(r)) \triangleright_{mx1r} s$
25. $\sigma(\tau(r, s)) \triangleright_{stss} \tau(\sigma(s), \sigma(r))$
26. $\sigma(\mathtt{sub_L}(r, s)) \triangleright_{ssbl} \mathtt{sub_R}(\sigma(s), \sigma(r))$
27. $\sigma(\mathtt{sub_R}(r, s)) \triangleright_{ssbr} \mathtt{sub_L}(\sigma(s), \sigma(r))$
28. $\sigma(\xi(r)) \triangleright_{sx} \xi(\sigma(r))$
29. $\sigma(\xi(s, r)) \triangleright_{sxss} \xi(\sigma(s), \sigma(r))$
30. $\sigma(\mu_1(r)) \triangleright_{sm} \mu_1(\sigma(r))$
31. $\sigma(\mu_2(r)) \triangleright_{sm} \mu_2(\sigma(r))$
32. $\sigma(\mu(s, r)) \triangleright_{smss} \mu(\sigma(s), \sigma(r))$
33. $\sigma(\mu(r, u, v)) \triangleright_{smsss} \mu(\sigma(r), \sigma(u), \sigma(v))$

34. $\tau(r, \mathtt{sub}_{\mathtt{L}}(\rho, s)) \rhd_{tsbll} \mathtt{sub}_{\mathtt{L}}(r, s)$
35. $\tau(r, \mathtt{sub}_{\mathtt{R}}(s, \rho)) \rhd_{tsbrl} \mathtt{sub}_{\mathtt{L}}(r, s)$
36. $\tau(\mathtt{sub}_{\mathtt{L}}(r, s), t) \rhd_{tsblr} \tau(r, \mathtt{sub}_{\mathtt{R}}(s, t))$
37. $\tau(\mathtt{sub}_{\mathtt{R}}(s, t), u) \rhd_{tsbrr} \mathtt{sub}_{\mathtt{R}}(s, \tau(t, u))$
38. $\tau(\tau(t, r), s) \rhd_{tt} \tau(t, \tau(r, s))$
39. $\tau(\mathcal{C}[u], \tau(\mathcal{C}[\sigma(u)], v)) \rhd_{tts} v$
40. $\tau(\mathcal{C}[\sigma(u)], \tau(\mathcal{C}[u], v)) \rhd_{tst} u$.

3.3 Termination property for the LND_{EQ}-TRS

Theorem 3.22 (Termination property for LND_{EQ}-TRS) *LND_{EQ}-TRS is terminating.*

The proof of the termination property for LND_{EQ}-TRS is obtained by using a special kind of ordering: *recursive path ordering*, proposed by N. Dershowitz in 1982 [10]:

Definition 3.23 (recursive path ordering) *Let $>$ be a partial ordering on a set of operators F. The recursive path ordering $>^*$ on the set T(F) of terms over F is defined recursively as follows:*

$$s = f(s_1, \ldots, s_m) >^* g(t_1, \ldots, t_n) = t,$$

if and only if

1. *$f = g$ and $\{s_1, \ldots, s_m\} \gg^* \{t_1, \ldots, t_n\}$, or*
2. *$f > g$ and $\{s\} \gg^* \{t_1, \ldots, t_n\}$, or*
3. *$f \not\geq g$ and $\{s_1, \ldots, s_m\} \gg^*$ or $= \{t\}$*

where $\gg^*$ is the extension of $>^*$ to multisets.

Note that this definition uses the notion of ordering on multisets. A given partial ordering $>$ on a set S may be extended to a partial ordering $\gg$ on finite multisets of elements of S, wherein a multiset is reduced by removing one or more elements and replacing them with any finite number of elements, each oh which is smaller than one of the elements removed [10].

The proof of termination property via a recursive path ordering is made by showing that for all rules $e \rightarrow d$ of the system, $e >^* d$.

The *recursive path ordering* can be extended in order to allow some function of a term $f(t_1, \ldots, t_n)$ to play the role of the operator f. As explained in [10], we can consider the k-th operand t_k to be the operator, and compare two terms by first recursively comparing their k-th operands.

In the proof of termination property for the LND_{EQ}-TRS, we use the precedence ordering on the rewrite operators for the rules from 1 to 32 defined as follows:

$$\begin{aligned}
&\sigma > \tau > \rho,\\
&\sigma > \xi,\\
&\sigma > \xi_\wedge,\\
&\sigma > \xi_1,\\
&\sigma > \xi_2,\\
&\sigma > \mu,\\
&\sigma > \mu_1,\\
&\sigma > \mu_2,\\
&\sigma > \mathtt{sub}_{\mathtt{L}},\\
&\sigma > \mathtt{sub}_{\mathtt{R}},\\
&\tau > \mathtt{sub}_{\mathtt{L}}
\end{aligned}$$

We can combine the recursive path idea used for the rules 1–34 with an extension of recursive path ordering for rules from 35 to 37, where the first operand is used as operator. When comparing two "τ" and "τ with $\mathtt{sub}_{\mathtt{R}}$", we use the first operand as operator. This proof is similar to Example (H) given in pp. 299–300 of [10].

The confluence proof is built by the Knuth–Bendix superposition algorithm applied to the rules of the system.

We have proved termination and confluence of the rewriting system LND_{EQ}-TRS [28, 29, 30]. As a matter of fact, rules 38 and 39 came out of the Knuth–Bendix completion procedure applied to the the rewriting system. As we have previously pointed out, although the rewriting system is terminating and confluent, we have observed an interesting phenomenon here: there may be more than one normal proof of an equality statement. This is not a contradiction since the confluence property only says that the term for the equality reason can be brought to a unique normal form regardless of the order in which it is reduced. But there may be a different, yet normal/canonical, proof of the same equality statement.

3.4 Rewrite equality

As we have just seen, LND_{EQ}-TRS gives us a total of 39 reductions rules. We call each rule as a *rewrite rule* (abbreviation: *rw-rule*). We have the following definition:

Definition 3.24 (rw-contraction, rw-reduction) *Let s and t be computational paths. We say that $s \rhd_{1rw} t$ (read as: a rw-contracts to b) iff we can obtain t from s by an application of only one rw-rule. If s can be reduced to t by a finite number of rw-contractions, then we say that $s \rhd_{rw} t$ (read as s rw-reduces to t).*

Definition 3.25 (rw-equality) *Let s and t be computational paths. We say that $s =_{rw} t$ (read as: s is rw-equal to t) iff t can be obtained from s by a finite (perhaps empty) series of rw-contractions and reversed rw-contractions. In other words, $s =_{rw} t$ iff there exists a sequence $R_0, ..., R_n$, with $n \geq 0$, such that*

$$(\forall i \leq n-1)(R_i \rhd_{1rw} R_{i+1} \text{ or } R_{i+1} \rhd_{1rw} R_i)$$
$$R_0 \equiv s, \quad R_n \equiv t$$

In the introduction, specifically in *definition 1.3* and *definition 1.4*, we showed that an application of a rw-rule yields a term of the identity type. For example, the rule $\sigma(\rho) \rhd_{sr} \rho$, with $x =_\rho x : A$, yields a term of type $\mathtt{Id}_{\mathtt{Id}_A(x,x)}(\sigma(\rho), \rho)$. Since rw-equalities are just applications of rw-rules (or inverse rw-rules), then one could think of an rw-equality as a term of the identity type. That way, if we have paths $s(a,b) : \mathtt{Id}_A(a,b)$, $t(a,b) : \mathtt{Id}_A(a,b)$ and an rw-equality θ such that $s =_{rw_\theta} t$, then θ is the following term of the identity type: $\theta : \mathtt{Id}_{\mathtt{Id}_A(a,b)}(s,t)$.

In the sequel, we use rw-equality to develop a groupoid model of a type.

4 Uniqueness of identity proofs

In this section, our objective is to show that the approach introduced in this work, i.e., the identity type as the type of computational paths, refutes the *Uniqueness of Identity Proofs (UIP)* property. To do that, we first introduce a groupoid model of a type using computational paths.

4.1 Groupoid structure

The groupoid structure uses the idea that computational paths are terms of the identity type. That way, given a type A, objects are seen as elements $a, b : A$ of the type and morphisms are terms of the identity type, i.e., computational paths $a =_s b$ between the objects. This structure will be referred to as A_{rw}.

Before we formally define this structure, we need to make clear that we are working using the concept of a weak category. As we are going to check, the groupoid laws will hold only up to rw-equality. In this sense, we use the term weak to indicate this fact, i.e., that the laws will not hold 'on the nose', but only up to rw-equality. With that, we first show that the structure is a weak category

Proposition 4.1 *A_{rw} is a weak category*

Proof. First, we need to define compositions of morphisms. Given two paths $s : a \to b$ and $t : b \to c$, then we can compose them to obtain $t \circ s = \tau(s,t)$. The identity arrow 1_a is defined as $1_a = \rho_a$. We now need to check the associative and identity laws. The associativity is given by a straightforward application of rule tt:

$$\tau(\tau(s,r),t) =_{rw_{tt}} \tau(s,\tau(r,t)).$$

The identity laws are given by straightforward applications of rules tlr and trr:

$$s \circ 1_a = s \circ \rho_a = \tau(\rho_a, s) =_{rw_{tlr}} s$$
$$1_b \circ s = \rho_b \circ s = \tau(s, \rho_b) =_{rw_{trr}} s$$

As one can clearly notice, the rules only hold up to rw-equality. In this sense, A_{rw} is a weak category.

■

To complete our groupoid structure, we need to prove that, in fact, A_{rw} is a weak groupoid. First, let's recall the concept of categorical groupoid [4]:

Definition 4.2 (Isomorphism) *Let $f : A \to B$ be an arrow of any category. f is called an isomorphism if there exists a $g : B \to A$ such that $g \circ f = 1_A$ and $f \circ g = 1_B$. g is called the inverse of f and can be written as f^{-1}.*

Definition 4.3 (Groupoid) *A groupoid is a category in which every arrow is an isomorphism.*

Proposition 4.4 *A_{rw} is a weak groupoid.*

Proof. We need to show that every path $s : a \to b$ of A_{rw} is an isomorphism. To do that, we need to find an inverse s'. Just do $s' = \sigma(s)$, then, by straightforward applications of rules tr and tsr, we have:

$$s \circ s' = s \circ \sigma(s) = \tau(\sigma(s), s) =_{rw_{tsr}} \rho_b$$
$$s' \circ s = \sigma(s) \circ s = \tau(s, \sigma(s)) =_{rw_{tr}} \rho_a$$

That way, the groupoid laws hold up to rw-equality. We conclude that A_{rw} is a weak groupoid.

■

4.2 Refutation of UIP

We use A_{rw} to show that our approach refutes the principle of uniqueness of identity proofs. Hofmann and Streicher [15] defines $UIP(A)$ as the type that is inhabited iff, given $a, b : A$, and $s, s' : \mathtt{Id}_A(a,b)$, then we have a witness $p : \mathtt{Id}_{\mathtt{Id}_A(a,b)}(s,s')$. In this sense, every pair of equality proofs (s, s') are propositionally equal. Using computational paths, one can interpret that $UIP(A)$ is inhabited iff given terms $a, b : A$ and any paths $a =_s b$ and $a =_t b$, then $s =_{rw} t$.

Using the groupoid interpretation, if UIP holds, then every type A is a trivial groupoid structure in which there is only one morphism between every pair of objects [15]. To show that UIP is false, one only needs to construct an A such that there is more than one morphism between a pair of objects (i.e., there is at least one pair of objects $a, b : A$ and a pair of paths s and t between them such that $s \neq_{rw} t$). To achieve this objective, consider the type A and that $(\lambda x.(\lambda y.yx)(\lambda w.zw))v : A$:

Theorem 4.5 *The type UIP is empty.*

Proof. We construct A_{rw}:

$$\begin{tikzcd}
& (\lambda x.(\lambda y.yx)(\lambda w.zw))v \arrow[dl, "\eta"'] \arrow[dr, "\beta"] \arrow[ddd, bend right, "\tau(\tau(\eta,\beta),\beta)"'] \arrow[ddd, bend left, "\tau(\tau(\beta,\beta),\eta)"] & \\
(\lambda x.(\lambda y.yx)x)v \arrow[d, "\beta"'] & & (\lambda x.(\lambda w.zw)x)v \arrow[d, "\beta"] \\
(\lambda y.yv)z \arrow[dr, "\beta"'] & & (\lambda w.zw)v \arrow[dl, "\eta"] \\
& zv &
\end{tikzcd}$$

As depicted above, there are (at least) two possible paths between $(\lambda x.(\lambda y.yx)(\lambda w.zw))v$ and zv. The first path is given by $\tau(\tau(\eta, \beta), \beta)$ and the second by $\tau(\tau(\beta, \beta), \eta)$. Moreover, looking at all rw-rules, there is no rule that establishes the rw-equality between these two paths. That way, A_{rw} is not a trivial groupoid and thus, UIP is empty. ■

With that, we conclude that our approach based on computational paths refutes the principle of uniqueness of identity proofs.

5 Conclusion

Motivated by looking at equalities in type theory as arising from the existence of computational paths between two formal objects, our purpose here was to offer an alternative perspective (to the one prevailing on the literature) on the role and the power of the so-called identity types, as well as of the notion of propositional equality as formalised in the so-called Curry-Howard functional interpretation. We started by recalling our previous observation [41] pertaining to the fact that the formulation of the identity type by Martin-Löf, both in the intensional and in the extensional versions, did not take into account an important entity, namely, identifiers for sequences of rewrites, and this has led to a false dichotomy.

Next, by considering as sequences of rewrites and substitution, we have shown that it comes a rather natural fact that two (or more) distinct proofs may be yet canonical and are none to be preferred over one another. By looking at proofs of equality as rewriting (or computational) paths this approach fits well with the recently proposed connections between type theory and homotopy theory via identity types, since elements of identity types will be, concretely, paths (or homotopies). In the end, our formulation of a proof theory for propositional equality is still very much in the style of type-theoretic identity types which, besides being a reformulation of Martin-Löfs own intensional identity types into one which dissolves what we see as a false dichotomy, turned out to validate the groupoid laws as uncovered by Hofmann & Streicher as well as to refute the principle of uniqueness of identity proofs.

References

[1] P. H. G. Aczel. Frege Structures and the Notions of Proposition, Truth and Set, in J. Barwise, H.-J. Keisler, and K. Kunen (eds.), *The Kleene Symposium*, Vol. 101 of *Studies in Logic and The Foundations of Mathematics*, pp 31–59, North-Holland Publishing Co., Amsterdam, xx+425pp, 1980. Proceedings of the Symposium held in June 18–24, 1978, at Madison, Wisconsin, USA

[2] P. H. G. Aczel. Term Declaration Logic and Generalised Composita, in *Sixth Annual IEEE Symposium on Logic in Computer Science* (*LICS'91*), pp 22–30, IEEE Press, 1991. Proceedings of the Symposium held July 15–18 1991, in Amsterdam, The Netherlands

[3] S. Awodey. Type Theory and Homotopy. arxiv.org/abs/1010.1810v1, Oct 2010.

[4] S. Awodey. *Category Theory*. Oxford University Press, 2010.

[5] H.P. Barendregt. *The Lambda Calculus, Its Syntax and Semantics*, volume 103 of *Studies in Logic and the Foundation of Mathematics*. North-Holland. Amsterdam. New York. Oxford., revised edition, 1984.

[6] B. van den Berg and R. Garner. Types are weak ω-groupoids. http://arxiv.org/abs/0812.0298v1, 2008.

[7] B. van den Berg and R. Garner. Topological and simplicial models of identity types. http://arxiv.org/abs/1007.4638v1, 2010. To appear in *ACM Transactions on Computational Logic*.

[8] Errett Bishop. *Foundations of Constructive Analysis. McGraw-Hill series in Higher Mathematics*. McGraw-Hill Book Company, New York, xiv+371pp., 1967.

[9] Ph. Le Chenadec. On the logic of unification. *J. Symbolic Computation*, 8(1 and 2):141–199, July/August 1989.

[10] N. Dershowitz. Orderings for term-rewriting systems. *Theor. Comp. Sci.* 17:279–301, 1982.

[11] D. M. Gabbay. *Labelled Deductive Systems, Volume I - Foundations.* Oxford University Press, 1996.

[12] D. M. Gabbay and R. J. G. B. de Queiroz. Extending the Curry-Howard interpretation to linear, relevant and other resource logics. *The Journal of Simbolic Logic*, 57(4):1319–1365, December 1992.

[13] J. R. Hindley and J. P. Seldin. *Lambda Calculus and Combinators. An Introduction*. Cambridge University Press, 2008.

[14] M. Hofmann and T. Streicher. The groupoid model refutes uniqueness of identity proofs. In *Logic in Computer Science, 1994* (*LICS '94*), pp. 208–212, 1994.

[15] M. Hofmann and T. Streicher. The groupoid interpretation of type theory. In *Twenty-five years of constructive type theory (Venice, 1995)*, G. Sambin and J. Smith (eds.), volume 36 of Oxford Logic Guides, Oxford Univ. Press, New York, 1998, pp. 83-111.

[16] *Homotopy Type Theory: Univalent Foundations of Mathematics*. Univalent Foundations Program, Institute for Advanced Study, Princeton, 2013. (https://github.com/HoTT/book/issues?state=open)

[17] W. A. Howard. The formulae-as-types notion of construction. In J. R. Seldin and J.R. Hindley, editors, *To H. B. Curry: Essays on Combinatory Logic Lambda Calculus and Formalism*. Academic Press, 1980. xxv+606pp.

[18] G. Kreisel and W. Tait. Finite definability of number theoretic functions and parametric completeness of equational calculi. *Zeitschr. f. Math. Logik und Grundlagen d. Math*, 7:28–38, 1961.

[19] D. R. Licata and R. Harper. Canonicity for 2-dimensional type theory. *Electronic Notes in Theoretical Computer Science* 276:263–289, September 2011.

[20] D. R. Licata and R. Harper. Two-Dimensional Directed Type Theory. *POPL '12 Proceedings of the 39th annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 337–348, 2012.

[21] P. Lumsdaine. Weak omega-categories from intensional type theory. *Logical Methods in Computer Science* 6(3:24):1–19, 2010.

[22] P. Martin-Löf. About Models for Intuitionistic Type Theories and the Notion of Definitional Equality, in S. Kanger (ed.), *Proceedings of the Third Scandinavian Logic Symposium*, Series *Studies in Logic and The Foundations of Mathematics*, pp 81–109, North-Holland, Amsterdam, 1975. Symposium held in 1973

[23] P. Martin-Löf. An intuitionistic theory of types: predicative part, in H. E. Rose and J. C. Shepherdson (eds.), *Logic Colloquium '73*, Vol. 80 of *Studies in Logic and The Foundations of Mathematics*, pp 73–118, North-Holland, Amsterdam, viii+513pp, 1975. Proceedings of the Colloquium held in Bristol, UK, in 1973

[24] P. Martin-Löf. Constructive Mathematics and Computer Programming, in L. J. Cohen, J. Łos, H. Pfeiffer, and K.-P. Podewski (eds.), *Logic, Methodology and Philosophy of Science VI*, Series *Studies in Logic and The Foundations of Mathematics*, pp 153–175, North-Holland, Amsterdam, xiii+738pp, 1992. Proceedings of the International Congress held in Hannover, August 22–29 1979

[25] P. Martin-Löf. *Intuitionistic Type Theory.* Series Studies in Proof Theory. Bibliopolis Naples, iv+91pp., 1984. Notes by Giovanni Sambin of a series of lectures given in Padova, June 1980.

[26] J. C. Mitchell and A. Scedrov. Notes on sconing and relators. In E. Boerger et al., editor, *Computer Science Logic '92, Selected Papers*, pages 352–378. Springer LNCS 702, 1993. Available by anonymous ftp from host ftp.cis.upenn.edu and the file pub/papers/scedrov/rel.dvi.

[27] B. Nordström, K. Petersson, and J. M. Smith. *Programming in Martin-Löf's Type Theory. An Introduction*, Vol. 7 of *The International Series of Monographs on Computer Science*, Clarendon Press, Oxford, x+221pp, 1990.

[28] A. G. de Oliveira. *Proof Transformations for Labelled Natural Deduction via Term Rewriting.* (In Portuguese). MSc thesis, Depto. de Informática, Universidade Federal de Pernambuco, Recife, Brazil, April 1995.

[29] A. G. de Oliveira and R. J. G. B. de Queiroz. Term Rewriting Systems with Labelled Deductive Systems. *Proceedings of Brazilian Symposium on Artificial Intelligence* (*SBIA'94*), Lecture Notes in Artificial Intelligence, Springer, pages 59–72, 1994.

[30] A. G. de Oliveira and R. J. G. B. de Queiroz. A Normalization Procedure for the Equational Fragment of Labelled Natural Deduction'. *Logic Journal of the Interest Group in Pure and Applied Logics* 7(2):173–215, 1999.

[31] A. G. de Oliveira and R. J. G. B. de Queiroz. A New Basic Set of Proof Transformations. In *We Will Show Them! Essays in Honour of Dov Gabbay. Volume 2*, S. Artemov, H. Barringer, A. Garcez, L. Lamb and J. Woods (eds.), College Publications, London, ISBN 1904987125, October 2005, pp. 499–528.

[32] F. Pfenning and R. Davies. A judgemental reconstruction of modal logic. *Mathematical Structures in Computer Science* 11:511-540, 2001.

[33] R. J. G. B. de Queiroz. A proof-theoretic account of programming and the rôle of reduction rules. *Dialectica* 42(4):265–282, 1988.

[34] R. J. G. B. de Queiroz. The mathematical language and its semantics: to show the consequences of a proposition is to give its meaning. In Weingartner, P. and Schurz, G., editors, *Reports of the Thirteenth International Wittgenstein Symposium 1988*, volume 18 of *Schriftenreihe der Wittgenstein-Gesellschaft*, Vienna, 1989, pp. 259–266, Hölder-Pichler-Tempsky. Symposium held in Kirchberg/Wechsel, Austria, August 14–21 1988.

[35] R. J. G. B. de Queiroz. *Proof Theory and Computer Programming. The Logical Foundations of Computer Programming*. PhD thesis, Imperial College, London, February 1990.

[36] R. J. G. B. de Queiroz. Meaning as grammar *plus* consequences. *Dialectica* 45(1):83–86, 1991.

[37] R. J. G. B. de Queiroz. *Grundgesetze* alongside *Begriffsschrift* (abstract). In *Abstracts of Fifteenth International Wittgenstein Symposium.* pp. 15–16. Symposium held in Kirchberg/Wechsel, August 16–23 1992.

[38] R. J. G. B. de Queiroz. Normalisation and Language-Games. *Dialectica* 48(2):83–125, 1994.

[39] R. J. G. B. de Queiroz. Meaning, function, purpose, usefulness, *consequences* – interconnected concepts. *Logic Journal of the Interest Group in Pure and Applied Logics* 9(5):693–734, 2001.

[40] R. J. G. B. de Queiroz. On reduction rules, meaning as use, and proof-theoretic semantics *Studia Logica* 90(2):211–247, November 2008.

[41] R. J. G. B. de Queiroz and D. M. Gabbay. Equality in Labelled Deductive Systems and the functional interpretation of propositional equality. In P. Dekker and M. Stokhof, editors, *Proceedings of the 9th Amsterdam Colloquium*, pages 547–565, 1994.

[42] R. J. G. B. de Queiroz and D. M. Gabbay. The functional interpretation of the existential quantifier. *Bulletin of the Interest Group in Pure and Applied Logics*, 3(2 and 3):243–290, 1995. Abstract in JSL 58(2):753–754, 1993. (Presented at *Logic Colloquium '91*, Uppsala, August 9–16.)

[43] R. J. G. B. de Queiroz and D. M. Gabbay. The functional interpretation of modal necessity. In de Rijke, Maarten, editor, *Advances in Intensional Logic*, Applied Logic Series, Kluwer Academic Publishers, September 1997, pp. 61–91.

[44] R. J. G. B. de Queiroz and D. M. Gabbay. Labelled Natural Deduction. In *Logic, Language and Reasoning. Essays in Honor of Dov Gabbay's 50th Anniversary*, H.J. Ohlbach and U. Reyle (eds.), Kluwer Academic Publishers, June 1999, pp. 173–250.

[45] R. J. G. B. de Queiroz and T. S. E. Maibaum. Proof Theory and Computer Programming. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik* 36:389–414, 1990.

[46] R. J. G. B. de Queiroz and T. S. E. Maibaum. Abstract Data Types and Type Theory: Theories as Types. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik* 37:149–166, 1991.

[47] R. J. G. B. de Queiroz and A. G. de Oliveira. The Functional Interpretation of Direct Computations. *Electronic Notes in Theoretical Computer Science* (Elsevier, ISSN 1571-0661) 269:19–40, April 2011.

[48] R. J. G. B. de Queiroz and A. G. de Oliveira. Propositional equality, identity types and direct computational paths. arXiv:1107.1901. Jul 2011.

[49] R. J. G. B. de Queiroz and A. G. de Oliveira. Natural Deduction for Equality: The Missing Entity. In *Advances in Natural Deduction. A Celebration of Dag Prawitz's Work*, L. C. Pereira, E. H. Haeusler & V. de Paiva, (eds.), volume 39 of *Trends in Logic*, Springer, 2014, pp. 63–91.

[50] R. J. G. B. de Queiroz, A. G. de Oliveira, D. M. Gabbay. *The Functional Interpretation of Logical Deduction*, Imperial College Press / World Scientific, Oct 2011.

[51] R. Statman. Herbrand's Theorem and Gentzen's notion of a direct proof. In J. Barwise, editor, *Handbook of Mathematical Logic*. North-Holland, Amsterdam, 1977.

[52] R. Statman. Bounds for proof-search and speed-up in the predicate calculus. *Annals of Mathematical Logic* 15:225–287, 1978.

[53] T. Streicher. Identity Types vs. Weak ω-Groupoids: Some Ideas and Problems. Slides of a talk given at *Identity Types - Topological and Categorical Structure*, Workshop, Uppsala, November 13–14, 2006.

[54] A. S. Troelstra and D. van Dalen. *Constructivism in Mathematics: An Introduction. Vol. II*, Vol. 123 of *Studies in Logic and The Foundations of Mathematics*, North-Holland, Amsterdam, xvii+535pp, 1988.

[55] V. Voevodsky. Univalent Foundations of Mathematics (abstract). In *Logic, Language, Information and Computation. 18th International Workshop, WoLLIC 2011*, Philadelphia, PA, USA, L. Beklemishev & R. de Queiroz (eds.), LNCS 6642, p.4, Springer, 2011.

[56] M. Warren. *Homotopy theoretic aspects of constructive type theory*, Ph.D. thesis, Carnegie Mellon University, 2008.

Ruy J.G.B. de Queiroz
Centro de Informática
Universidade Federal de Pernambuco
Av. Jornalista Anibal Fernandes, s/n
50740-560 Recife, PE, Brazil
E-mail: ruy@cin.ufpe.br

Anjolina G. de Oliveira
Centro de Informática
Universidade Federal de Pernambuco
Av. Jornalista Anibal Fernandes, s/n
50740-560 Recife, PE, Brazil
E-mail: ago@cin.ufpe.br

Arthur F. Ramos
Centro de Informática
Universidade Federal de Pernambuco
Av. Jornalista Anibal Fernandes, s/n
50740-560 Recife, PE, Brazil
E-mail: afr@cin.ufpe.br

Appendices

A Definitional Equality Rules

$$(\beta) \quad \frac{a : A \quad \begin{matrix}[x:A] \\ b : B\end{matrix}}{(\lambda x.b)a =_\beta b[a/x] : B}$$

$$(\eta) \quad \frac{f : (\Pi x : A)B}{\lambda x.APP(f, x) =_\eta f : (\Pi x : A)B} \; (x \notin FV(f))$$

$$(\rho) \quad \frac{a : A}{a =_\rho a : A}$$

$$(\mu) \quad \frac{a =_s a' : A \quad f : (\Pi x : A)B}{APP(f, a) =_{\mu(s)} APP(f, a') : B}$$

$$(\tau) \quad \frac{a =_s a' : A \qquad a' =_t a'' : A}{a =_{\tau(s,t)} a'' : A}$$

$$(\nu) \quad \frac{a : A \quad f =_s f' : (\Pi x : A)B}{APP(f, a) =_{\nu(s)} APP(f', a) : B}$$

$$(\sigma) \quad \frac{a =_s a' : A}{a' =_{\sigma(s)} a : A}$$

$$(\xi) \quad \frac{\begin{matrix}[x:A] \\ f(x) =_s f'(x) : B\end{matrix}}{\lambda x.f(x) =_{\xi(s)} \lambda x.f'(x) : (\Pi x : A)B}$$

$$(\mu_1) \quad \frac{p =_s q : A \times B}{FST(p) =_{\mu_1(s)} FST(q) : A}$$

$$(\mu_2) \quad \frac{p =_s q : A \times B}{SND(p) =_{\mu_2(s)} SND(q) : B}$$

$$(\xi_\wedge) \quad \frac{a =_r a' : A \quad b =_s b' : B}{\langle a, b\rangle =_{\xi_\wedge(r,s)} \langle a', b'\rangle : A \times B}$$

$$(\xi_1) \quad \frac{a =_s a' : A \quad b : B}{\langle a, b\rangle =_{\xi_1(s)} \langle a', b\rangle : A \times B}$$

$$(\xi_2) \quad \frac{a : A \quad b =_s b' : B}{\langle a, b\rangle =_{\xi_2(s)} \langle a, b'\rangle : A \times B}$$

$$(\xi_1) \quad \frac{a =_s a' : A}{i(a) =_{\xi_1(s)} i(a') : A + B}$$

$$(\xi_2) \quad \frac{b =_s b' : B}{j(b) =_{\xi_2(s)} j(b') : A + B}$$

$$(\nu) \quad \frac{p =_s q : A+B \quad \begin{array}{c}[x:A]\\ f(x):C\end{array} \quad \begin{array}{c}[y:B]\\ g(y):C\end{array}}{D(p, \acute{x}f(x), \acute{y}g(y)) =_{\nu(s)} D(q, \acute{x}f(x), \acute{y}g(y)) : C}$$

$$(\mu_1) \quad \frac{p : A+B \quad \begin{array}{c}[x:A]\\ f(x) =_s f'(x):C\end{array} \quad \begin{array}{c}[y:B]\\ g(y):C\end{array}}{D(p, \acute{x}f(x), \acute{y}g(y)) =_{\mu_1(s)} D(p, \acute{x}f'(x), \acute{y}g(y)) : C}$$

$$(\mu_2) \quad \frac{p : A+B \quad \begin{array}{c}[x:A]\\ f(x):C\end{array} \quad \begin{array}{c}[y:B]\\ g(y) =_s g'(y):C\end{array}}{D(p, \acute{x}f(x), \acute{y}g(y)) =_{\mu_2(s)} D(p, \acute{x}f(x), \acute{y}g'(y)) : C}$$

$$(\xi_1) \quad \frac{a =_s a' : A \quad b(a) : B(a)}{\varepsilon x.(b(x), a) =_{\xi_1(s)} \varepsilon x.(b(x), a') : (\Sigma x : A)B(x)}$$

$$(\xi_2) \quad \frac{a : A \quad b(a) =_s b'(a) : B(a)}{\varepsilon x.(b(x), a) =_{\xi_2(s)} \varepsilon x.(b'(x), a) : (\Sigma x : A)B(x)}$$

$$(\nu) \quad \frac{p =_s q : (\Sigma x : A)B(x) \quad \begin{array}{c}[t:A,\ f(t):B(t)]\\ h(t,f):C\end{array}}{E(p, \acute{f}\acute{t}h(t,f)) =_{\nu(s)} E(q, \acute{f}\acute{t}h(t,f)) : C} \qquad (\mu) \quad \frac{p : (\Sigma x : A)B(x) \quad \begin{array}{c}[t:A,\ f(t):B(t)]\\ h(t,f) =_s h'(t,f):C\end{array}}{E(p, \acute{f}\acute{t}h(t,f)) =_{\mu(s)} E(p, \acute{f}\acute{t}h'(t,f)) : C}$$

B Function extensionality

A proof of (weak) Function Extensionality:

$$[f : A \to B]$$

$$[g : A \to B]$$

$$\dfrac{\dfrac{[z : A] \quad [v : \Pi x^{A}.Id_{B}(APP(f,x), APP(g,x))]}{APP(v,z) : Id_{B}(APP(f,z), AP(g,z))} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[f : A \to B]}{\lambda z.APP(f,z) =_{\eta} f : A \to B}}{f =_{\sigma(\eta)} \lambda z.APP(f,z) : A \to B} \quad \dfrac{[APP(f,z) =_{t} APP(g,z) : B]}{\lambda z.APP(f,z) =_{\xi(t)} \lambda z.APP(g,z) : A \to B}}{f =_{\tau(\sigma(\eta),\xi(t))} \lambda z.APP(g,z) : A \to B} \quad \dfrac{[g : A \to B]}{\lambda z.AP(g,z) =_{\eta} g : A \to B}}{f =_{\tau(\tau(\sigma(\eta),\xi(t)),\eta)} g : A \to B}}{(\tau(\tau(\sigma(\eta),\xi(t)),\eta))(f,g) : Id_{A \to B}(f,g)}}{J(APP(v,z), \acute{t}.(\tau(\tau(\sigma(\eta),\xi(t)),\eta))(f,g)) : Id_{A \to B}(f,g)}$$

$$\dfrac{J(APP(v,z), \acute{t}.(\tau(\tau(\sigma(\eta),\xi(t)),\eta))(f,g)) : Id_{A \to B}(f,g)}{\lambda v.J(APP(v,z), \acute{t}.(\tau(\tau(\sigma(\eta),\xi(t)),\eta))(f,g)) : \Pi x^{A}.Id_{B}(APP(f,x), AP(g,x)) \to Id_{A \to B}(f,g)}$$

$$\dfrac{}{\lambda g.\lambda v.J(APP(v,z), \acute{t}.(\tau(\tau(\sigma(\eta),\xi(t)),\eta))(f,g)) : \Pi g^{A \to B}(\Pi x^{A}.Id_{B}(APP(f,x), AP(g,x)) \to Id_{A \to B}(f,g))}$$

$$\dfrac{}{\lambda f.\lambda g.\lambda v.J(AP(v,z), \acute{t}.(\tau(\tau(\sigma(\eta),\xi(t)),\eta))(f,g)) : \Pi f^{A \to B}\Pi g^{A \to B}(\Pi x^{A}.Id_{B}(APP(f,x), AP(g,x)) \to Id_{A \to B}(f,g))}$$

$$\dfrac{}{\lambda z.\lambda f.\lambda g.\lambda v.J(APP(v,z), \acute{t}.(\tau(\tau(\sigma(\eta),\xi(t)),\eta))(f,g)) : A \to (\Pi f^{A \to B}\Pi g^{A \to B}(\Pi x^{A}.Id_{B}(APP(f,x), AP(g,x)) \to Id_{A \to B}(f,g)))}$$