On the Groupoid Model of Computational Paths

Arthur F. Ramos
Centro de Informática
Universidade Federal de Pernambuco
Recife, Brazil

Ruy J. G. B. de Queiroz
Centro de Informática
Universidade Federal de Pernambuco
Recife, Brazil

Anjolina G. de Oliveira
Centro de Informática
Universidade Federal de Pernambuco
Recife, Brazil

Abstract
The main objective of this work is to study mathematical properties of computational paths. Originally proposed by de Queiroz & Gabbay (1994) as ‘sequences or rewrites’, computational paths are taken to be terms of the identity type of Martin-Löf’s Intensional Type Theory, since these paths can be seen as the grounds on which the propositional equality between two computational objects stand. From this perspective, this work aims to show that one of the properties of the identity type is present on computational paths. We are referring to the fact that that the identity type induces a groupoid structure, as proposed by Hofmann & Streicher (1994). Using categorical semantics, we show that computational paths induce a groupoid structure. We also show that computational paths are capable of inducing higher categorical structures.

Keywords: Computational paths, groupoid model, equality theory, term rewriting systems, type theory, category theory, higher categorical structures.

1 Introduction
There seems to be little doubt that the identity type is one of the most intriguing concepts of Martin-Löf’s Type Theory. This claim is supported by recent ground-
breaking discoveries. In 2005, Vladimir Voevodsky [17] discovered the Univalent Models, resulting in a new area of research known as Homotopy Type Theory [2]. This theory is based on the fact that a term of some identity type, for example $p : \text{Id}_{A}(a, b)$, has a clear homotopical interpretation. The interpretation is that the witness $p$ can be seen as a homotopical path between the points $a$ and $b$ within a topological space $A$. This simple interpretation has made clear the connection between Type Theory and Homotopy Theory, generating groundbreaking results, as one can see in [15,2]. It is important to emphasize that one important fact of the homotopic interpretation is that the homotopic paths exist only in the semantic sense. In other words, there is no formal entity in type theory that represents these paths. They are not present in the syntax of Type Theory.

In this work, we are interested in a concept known as computational path, originally proposed by [9] and further explored by [7]. A computational path is an entity that establishes the equality between two terms of the same type. It differs from the homotopical path, since it is not only a semantic interpretation. It is a formal entity of the equality theory, as proposed by [7]. In fact, in our work entitled "On the Identity Type as the Type of Computational Paths" [14], we claim that computational paths should be considered as a formal entity of Type Theory and that the identity type is just the type of this new entity. That way, the computational path would exist not only in a semantic sense, but it would be also a formal entity of the syntax of Type Theory.

Since the connection between computational paths and identity types have been studied in the work that we have just mentioned, this connection will not be the objective here. Rather, we are interested in important mathematical properties that arise naturally from the concept of computational paths. Specifically, we want to show that computational paths naturally induce a structure known as a groupoid. In fact, that the identity type induces a groupoid structure is a well-known fact, established by [11] in 1994. Nevertheless, this work differs substantially from [11]. The main difference is the fact that in [11], the induced groupoid is constructed by the construction of several terms of the identity type, using the elimination rule (i.e., using the constructor $J$). In fact, we will not use the identity type as originally formulated or any elimination rule. We will work directly with the concept of computational path and concepts related to it. We will show that, using these concepts, we can induce a category that has the properties of a groupoid.

## 2 Computational Paths

Since computational path is a generic term, it is important to emphasize the fact that we are using the term computational path in the sense defined by [7]. A computational path is based on the idea that it is possible to formally define when two computational objects $a, b : A$ are equal. These two objects are equal if one can reach $b$ from $a$ applying a sequence of axioms or rules. This sequence of operations forms a path. Since it is between two computational objects, it is said that this path is a computational one. Also, an application of an axiom or a rule transforms (or rewrite) an term in another. For that reason, a computational path is also known as a sequence of rewrites. Nevertheless, before we define formally a computational
path, we can take a look at one famous equality theory, the \( \lambda \beta \eta - \text{equality} \) [10]:

**Definition 2.1** The \( \lambda \beta \eta \)-equality is composed by the following axioms:

- \((\alpha)\) \( \lambda x. M = \lambda y. [y/x]M \) if \( y \notin FV(M) \);
- \((\beta)\) \( (\lambda x. M)N = [N/x]M \);
- \((\rho)\) \( M = M \);
- \((\eta)\) \( (\lambda x.Mx) = M \) \( x \notin FV(M) \).

And the following rules of inference:

\[
\begin{align*}
(\mu) & \quad \frac{M = M'}{NM = NM'} \\
(\tau) & \quad \frac{M = N}{M = P}
\end{align*}
\]

\[
\begin{align*}
(\nu) & \quad \frac{M = M'}{MN = M'N} \\
(\sigma) & \quad \frac{M = N}{N = M}
\end{align*}
\]

\[
\begin{align*}
(\xi) & \quad \frac{M = M'}{\lambda x. M = \lambda x. M'}
\end{align*}
\]

**Definition 2.2** ([10]) \( P \) is \( \beta \)-equal or \( \beta \)-convertible to \( Q \) (notation \( P =_\beta Q \)) iff \( Q \) is obtained from \( P \) by a finite (perhaps empty) series of \( \beta \)-contractions and reversed \( \beta \)-contractions and changes of bound variables. That is, \( P =_\beta Q \) iff there exist \( P_0, \ldots, P_n \) \( (n \geq 0) \) such that \( P_0 \equiv P, P_n \equiv Q, (\forall i \leq n-1)(P_i \triangleright_1 P_{i+1} \text{ or } P_{i+1} \triangleright_1 P_i \text{ or } P_i \equiv_\alpha P_{i+1}) \).

(NB: equality with an existential force, which will show in the proof rules for the identity type.)

The same happens with \( \lambda \beta \eta \)-equality:

**Definition 2.3** (\( \lambda \beta \eta \)-equality [10]) The equality-relation determined by the theory \( \lambda \beta \eta \) is called \( =_{\beta \eta} \); that is, we define

\[
M =_{\beta \eta} N \iff \lambda \beta \eta \vdash M = N.
\]

**Example 2.4** Take the term \( M \equiv (\lambda x.(\lambda y.yx)(\lambda w.zw))v \). Then, it is \( \beta \eta \)-equal to \( N \equiv zv \) because of the sequence:

\[
(\lambda x.(\lambda y.yx)(\lambda w.zw))v, (\lambda x.(\lambda y.yx)z)v, (\lambda y.yv)z, zv
\]

which starts from \( M \) and ends with \( N \), and each member of the sequence is obtained via 1-step \( \beta \)- or \( \eta \)-contraction of a previous term in the sequence. To take this sequence into a path, one has to apply transitivity twice, as we do in the example below.

**Example 2.5** The term \( M \equiv (\lambda x.(\lambda y.yx)(\lambda w.zw))v \) is \( \beta \eta \)-equal to \( N \equiv zv \) because of the sequence:

\[
(\lambda x.(\lambda y.yx)(\lambda w.zw))v, (\lambda x.(\lambda y.yx)z)v, (\lambda y.yv)z, zv
\]

Now, taking this sequence into a path leads us to the following:

The first is equal to the second based on the grounds:

\[
\eta((\lambda x.(\lambda y.yx)(\lambda w.zw))v, (\lambda x.(\lambda y.yx)z)v)
\]

The second is equal to the third based on the grounds:
\[ \beta((\lambda x.(\lambda y.yx)z)v, (\lambda y.yv)z) \]

Now, the first is equal to the third based on the grounds:
\[ \tau(\eta((\lambda x.(\lambda y.yx)(\lambda w.zw))v, (\lambda x.(\lambda y.yx)z)v), \beta((\lambda x.(\lambda y.yx)z)v, (\lambda y.yv)z)) \]

Now, the third is equal to the fourth one based on the grounds:
\[ \beta((\lambda y.yv)z, zv) \]

Thus, the first one is equal to the fourth one based on the grounds:
\[ \tau(\eta((\lambda x.(\lambda y.yx)(\lambda w.zw))v, (\lambda x.(\lambda y.yx)z)v), \beta((\lambda x.(\lambda y.yx)z)v, (\lambda y.yv)z)), \beta((\lambda y.yv)z, zv)) \]

The aforementioned theory establishes the equality between two \( \lambda \)-terms. Since we are working with computational objects as terms of a type, we need to translate the \( \lambda \beta \eta \)-equality to a suitable equality theory based on Martin L"of’s type theory. We obtain:

**Definition 2.6** The equality theory of Martin L"of’s type theory has the following basic proof rules for the \( \Pi \)-type:

\[
\begin{align*}
(\beta) & \quad \frac{N : A}{\lambda x.M} = M[N/x] : B[N/x] & (\xi) & \quad \frac{M = M' : B}{\lambda x.M = \lambda x.M' : (\Pi x : A)B} \\
(\rho) & \quad \frac{M = M' : A}{N = M : A} & (\mu) & \quad \frac{M = M' : A}{N = N : (\Pi x : A)B} \\
(\sigma) & \quad \frac{N = N : A}{M = M : A} & (\nu) & \quad \frac{M : (\Pi x : A)B}{MN = M'N : B[N/x]} \\
(\tau) & \quad \frac{M = N : A}{M = P : A} \\
(\eta) & \quad \frac{M : (\Pi x : A)B}{(\lambda x.M) = M : (\Pi x : A)B} (x \notin FV(M))
\end{align*}
\]

We are finally able to formally define computational paths:

**Definition 2.7** Let \( a \) and \( b \) be elements of a type \( A \). Then, a *computational path* \( s \) from \( a \) to \( b \) is a composition of rewrites (each rewrite is an application of the inference rules of the equality theory of type theory or is a change of bound variables). We denote that by \( a =_s b \).

As we have seen in example 2.5, composition of rewrites are applications of the rule \( \tau \). Since change of bound variables is possible, consider that each term is considered up to \( \alpha \)-equivalence.

### 3 A Term Rewriting System for Paths

As we have just showed, a computational path establishes when two terms of the same type are equal. From the theory of computational paths, an interesting case arises. Suppose we have a path \( s \) that establishes that \( a =_s b : A \) and a path \( t \)
that establishes that \( a =_t b : A \). Consider that \( s \) and \( t \) are formed by distinct compositions of rewrites. Is it possible to conclude that there are cases that \( s \) and \( t \) should be considered equivalent? The answer is yes. Consider the following examples:

**Example 3.1** Consider the path \( a =_t b : A \). By the symmetric property, we obtain \( b =_{\sigma(t)} a : A \). What if we apply the property again on the path \( \sigma(t) \)? We would obtain a path \( a =_{\sigma(\sigma(t))} b : A \). Since we applied symmetry twice in succession, we obtained a path that is equivalent to the initial path \( t \). For that reason, we conclude the act of applying symmetry twice in succession is a redundancy. We say that the path \( \sigma(\sigma(t)) \) can be reduced to the path \( t \).

**Example 3.2** Consider the reflexive path \( a =_\rho a : A \). Since it is a trivial path, the symmetric path \( a =_{\sigma(\rho)} a : A \) is equivalent to the initial one. For that reason, the application of the symmetry on the reflexive path is a redundancy. The path \( \sigma(\rho) \) can be reduced to the path \( \rho \).

**Example 3.3** Consider the path \( a =_t b : A \) and the inverse path \( b =_{\sigma(t)} a : A \). We can apply the transitive property in these paths, obtaining \( a =_{\tau(t,\sigma(t))} a : A \). Since the paths are inversions of each other, the transitive path is equivalent to the trivial path \( \rho \). Therefore, this transitive application is a redundancy. The path \( \tau(t,\sigma(t)) \) can be reduced to the trivial path \( \rho \).

As one could see in the aforementioned examples, different paths should be considered equal if one is just a redundant form of the other. The examples that we have just seen are straightforward and simple cases. Since the equality theory has a total of 7 axioms, the possibility of combinations that could generate redundancies are high. Fortunately, all possible redundancies were thoroughly mapped by De Oliveira (1995)[3]. In this work, a system that establishes all redundancies and creates rules that solve them was proposed. This system, known as \( LND_{EQ-TRS} \), maps a total of 39 rules that solve redundancies. Since we have a special interest on the groupoid model, we are not interested in all redundancy rules, but in a very specific subset of these rules (all have been taken from [3,6]):

- Rules involving \( \sigma \) and \( \rho \):

\[
\frac{x =_\rho x : A}{x =_{\sigma(\rho)} x : A} \quad \triangleright_{sr} \quad x =_\rho x : A
\]

\[
\frac{x =_r y : A}{y =_{\sigma(\sigma)} x : A} \quad \triangleright_{ss} \quad x =_r y : A
\]

- Rules involving \( \tau \):

\[
\frac{x =_r y : A \quad y =_{\sigma(\sigma)} x : A}{x =_{\tau(\tau,\sigma(\sigma))} x : A} \quad \triangleright_{tr} \quad x =_\rho x : A
\]
\[
\begin{align*}
\frac{y = \sigma(r) \ x : A}{y = \tau(\sigma(r), x) \ y : A} & \quad \triangleright_{tsr} \quad y = \rho \ y : A \\
\frac{x = r \ y : A}{x = \tau(r, \rho) \ y : A} & \quad \triangleright_{trr} \quad x = r \ y : A \\
\frac{x = \rho \ x : A}{x = \tau(\rho, r) \ y : A} & \quad \triangleright_{tlr} \quad x = r \ y : A
\end{align*}
\]

- Rule involving \( \tau \) and \( \tau \)
\[
\begin{align*}
\frac{x = t \ y : A}{x = \tau(t, r) \ w : A} & \quad \triangleright_{tt} \quad y = \tau(\tau(t, r), s) \ z : A \\
\frac{y = r \ w : A}{w = s \ z : A} & \quad \triangleright_{tt} \quad y = \tau(r, s) \ z : A \\
\frac{x = t \ y : A}{x = \tau(t, \tau(r, s)) \ z : A}
\end{align*}
\]

**Definition 3.4** An \( rw \)-rule is any of the rules defined in \( LND_{EQ} - TRS \).

**Definition 3.5** Let \( s \) and \( t \) be computational paths. We say that \( s \triangleright_{1rw} t \) (read as: \( s \) \( rw \)-contracts to \( t \)) iff we can obtain \( t \) from \( s \) by an application of only one \( rw \)-rule. If \( s \) can be reduced to \( t \) by finite number of \( rw \)-contractions, then we say that \( s \triangleright_{rw} t \) (read as \( s \) \( rw \)-reduces to \( t \)).

**Definition 3.6** 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 \( R_0, \ldots, R_n \), with \( n \geq 0 \), such that
\[
(\forall i \leq n - 1)(R_i \triangleright_{1rw} R_{i+1} \text{ or } R_{i+1} \triangleright_{1rw} R_i) \\
R_0 \equiv s, \quad R_n \equiv t
\]

**Proposition 3.7** is transitive, symmetric and reflexive.

**Proof.** Comes directly from the fact that \( rw \)-equality is the transitive, reflexive and symmetric closure of \( rw \).

Before talking directly from the fact that \( rw \)-equality is the transitive, reflexive and symmetric closure of \( rw \).

3.1 **\( LND_{EQ} - TRS_2 \)**

Until now, this section has concluded that there exist redundancies which are resolved by a system called \( LND_{EQ} - TRS \). This system establishes rules that reduces these redundancies. Moreover, we concluded that these redundancies are just redundant uses of the equality axioms showed in section 2. In fact, since these axioms just defines an equality theory for type theory, one can specify and say that these are redundancies of the equality of type theory. As we mentioned, the
\( LND_{EQ} - TRS \) has a total of 39 rules \[3\]. Since the \( rw \)-equality is based on the rules of \( LND_{EQ} - TRS \), one can just imagine the high number of redundancies that \( rw \)-equality could cause. In fact, a thoroughly study of all the redundancies caused by these rules could generate an entire new work. Fortunately, we are only interested in the redundancies caused by the fact that \( rw \)-equality is transitive, reflexive and symmetric with the addition of only one specific \( rw \)-rule. Let’s say that we have a system, called \( LND_{EQ} - TRS_2 \), that resolves all the redundancies caused by \( rw \)-equality (the same way that \( LND_{EQ} - TRS \) resolves all the redundancies caused by equality). Since we know that \( rw \)-equality is transitive, symmetric and reflexive, it should have the same redundancies that the equality had involving only these properties. Since \( rw \)-equality is just a sequence of \( rw \)-rules (also similar to equality, since equality is just a computational path, i.e., a sequence of identifiers), then we could put a name on these sequences. For example, if \( s \) and \( t \) are \( rw \)-equal because there exists a sequence \( \theta : R_0, \ldots, R_n \) that justifies the \( rw \)-equality, then we can write that \( s =_{rw} t \). Thus, we can rewrite, using \( rw \)-equality, all the rules that originated the rules involving \( \tau \), \( \sigma \) and \( \rho \). For example, we have:

\[
\frac{x =_{rw} y : A \quad y =_{rw} w : A}{x =_{rw}(t,r) w : A \quad w =_{rw} z : A} \quad \frac{x =_{rw}(t,r,s) z : A}{\quad x =_{rw}(t,r,s) z : A} \quad x =_{rw} y : A \quad y =_{rw} w : A \quad w =_{rw} z : A} \quad \frac{x =_{rw}(t,r,s) z : A}{x =_{rw}(t,r,s) z : A} \quad \frac{\triangleright_{t2}}{x =_{rw} y : A \quad y =_{rw} w : A \quad w =_{rw} z : A}
\]

Therefore, we obtain the rule \( tt_2 \), that resolves one of the redundancies caused by the transitivity of \( rw \)-equality (the 2 in \( tt_2 \) indicates that it is a rule that resolves a redundancy of \( rw \)-equality). In fact, using the same reasoning, we can obtain, for \( rw \)-equality, all the redundancies that we have showed in page 5. In other words, we have \( tr_2, tsr_2, trr_2, tl_2, sr_2, ss_2 \) and \( tt_2 \). Since we have now rules of \( LND_{EQ} - TRS_2 \), we can use all the concepts that we have just defined for \( LND_{EQ} - TRS \). The only difference is that instead of having \( rw \)-rules and \( rw \)-equality, we have \( rw_2 \)-rules and \( rw_2 \)-equality.

There is an important rule specific to this system. It stems from the fact that transitivity of reducible paths can be reduced in different ways, but generating the same result. For example, consider the simple case of \( \tau(s, t) \) and consider that it is possible to reduce \( s \) to \( s' \) and \( t \) to \( t' \). There is two possible \( rw \)-sequences that reduces this case: The first one is \( \theta : \tau(s, t) \triangleright_{1rw} \tau(s', t) \triangleright_{1rw} \tau(s', t') \) and the second \( \theta' : \tau(s, t) \triangleright_{1rw} \tau(s, t') \triangleright_{1rw} \tau(s', t') \). Both \( rw \)-sequences obtained the same result in similar ways, the only difference being the choices that have been made at each step. Since the variables, when considered individually, followed the same reductions, these \( rw \)-sequences should be considered redundant relative to each other and, for that reason, there should be \( rw_2 \)-rule that establishes this reduction. This rule is called \textit{choice independence} and is denoted by \( cd_2 \). In fact, independent of the quantity of transitivities and variables, if the sole difference between the \( rw \)-
sequences are the choices that were made in each step, then this rule will establish the \( rw_2 \)-equality between the sequences.

**Proposition 3.8** \( rw_2 \)-equality is transitive, symmetric and reflexive.

**Proof.** Analogous to Proposition 3.7. \( \square \)

## 4 The Groupoid Induced by Computational Paths

The objective of this section is to show that computational paths, together with the reduction rules discussed in the last section, are capable of inducing structures known as groupoids. To do that, we will use a categorical interpretation.

Before we conclude that computational paths induces categories with groupoid properties, we need to make clear the difference between a strict and a weak category. As one will see, the word weak will appear many times. This will be the case because some of the categorical equalities will not hold “on the nose”, so to say. They will hold up to \( rw \)-equality or up to higher levels of \( rw \)-equalities. This is similar to the groupoid model of the identity type proposed by [11]. In [11], the equalities do not hold “on the nose”, they hold up to propositional equality (or up to homotopy if one uses the homotopic interpretation). To indicate that these equalities hold only up to some property, we say that the induced structure is a weak categorical structure.

For each type \( A \), computational paths induces a weak categorical structure called \( A_{rw} \). In \( A_{rw} \), the objects are terms \( a \) of the type \( A \) and morphisms between terms \( a : A \) and \( b : A \) are arrows \( s : a \rightarrow b \) such that \( s \) is a computational path between the terms, i.e., \( a =_{s} b : A \). One can easily check that \( A_{rw} \) is a category. To do that, just define the composition of morphisms as the transitivity of two paths and the reflexive path as the identity arrow. The \( rw \)-equality of the associativity comes directly from the \( tt \) rule and the \( rw \)-equality of the identity laws comes from the \( trr \) and \( trl \) rules. Of course, as discussed before, this only forms a weak structure, since the equalities only hold up to \( rw \)-equality. The most interesting fact about \( A_{rw} \) is the following proposition:

**Proposition 4.1** The induced structure \( A_{rw} \) has a weak groupoidal structure.

**Proof.** A groupoid is just a category in which every arrow is an isomorphism. Since we are working in a weak sense, the isomorphism equalities need only to hold up to \( rw \)-equality. To show that, for every arrow \( s : a \rightarrow b \), we need to show a \( t : b \rightarrow a \) such that \( t \circ s =_{rw} 1_{a} \) and \( s \circ t =_{rw} 1_{b} \). To do that, recall that every computational path has an inverse path \( \sigma(s) \). Put \( t = \sigma(s) \). Thus, we have that \( s \circ t = s \circ \sigma(s) = \tau(\sigma(s), s) \triangleright_{tr} \rho_{b} \). Since \( \rho_{b} = 1_{b} \), we conclude that \( s \circ t =_{rw} 1_{b} \). We also have that \( t \circ s = \sigma(s) \circ s = \tau(s, \sigma(s)) \triangleright_{tr} \rho_{a} \). Therefore, \( t \circ s = 1_{a} \). We conclude that every arrow is a weak isomorphism and thus, \( A_{rw} \) is a weak groupoidal structure. \( \square \)

With that, we conclude that computational paths have a groupoid model.
4.1 Higher structures

We have just showed that computational paths induce a weak groupoidal structure known as $A_{rw}$. We also know that the arrows (or morphisms) of $A_{rw}$ are computational paths between two terms of a type $A$. As we saw in the previous section, sometimes a computational path can be reduced to another by rules that we called $rw$-rules. That way, if we have terms $a, b : A$ and paths between these terms, we can define a new structure. This new structure, called $A_{2rw}(a, b)$, has, as objects, computational paths between $a$ and $b$ and the set of morphisms $\text{Hom}_{A_{2rw}(a, b)}(s, t)$ between paths $a =_s b$ and $a =_t b$ corresponds to the set of sequences that prove $s =_{rw} t$. Since $rw$-equality is transitive, reflexive and symmetric, $A_{2rw}$ is a weak categorical structure which the equalities hold up to $rw_2$-equality.

The proof of this fact is analogous to the proposition 4.1. The sole difference is the fact that since the morphisms are $rw$-equalities, instead of computational paths, all the equalities will hold up to $rw_2$-equality. To see this, take the example of the associativity. Looking at the $LND_{EQ} - TRS_2$ system, we have that $\tau(\tau(\theta, \sigma), \phi) \tau(\theta, \tau(\sigma, \phi)) = \tau(\theta, \tau(\sigma, \phi))$. The associative law holds up to $rw_2$-equality. As one can easily check, the identity law will also hold up to $rw_2$-equality. Therefore, $A_{2rw}(a, b)$ has a weak categorical structure. Analogous to proposition 4.4, the groupoid law will also hold up to $rw_2$-equality.

Instead of considering $A_{rw}$ and $A_{2rw}(a, b)$ as separated structures, we can think of a unique structure with 2-levels. The first level is $A_{rw}$ and the second one is the structure $A_{2rw}(a, b)$ between each pair of objects $a, b \in A_{rw}$. We call this structure $2 - A_{rw}$. The morphisms of the first level are called 1-morphisms and the ones of the second level are called 2-morphisms (also known as 2-arrows or 2-cells). Since it has multiple levels, it is considered a higher structure. We want to prove that this structure is a categorical structure known as weak 2-category. The main problem is the fact that in a weak 2-category, the last level (i.e., the second level) needs to hold up in a strict sense. This is not the case for $2 - A_{2rw}$, since each $A_{2rw}(a, b)$ only holds up to $rw_2$-equality. Nevertheless, there still a way to induce this weak 2-category. Since $rw$-equality is an equivalence relation (because it is transitive, symmetric and reflexive), we can consider a special $A_{2rw}(a, b)$, where the arrows are the arrows of $A_{2rw}(a, b)$ modulo $rw_2$-equality. That way, since the equalities hold up to $rw_2$-equality in $A_{2rw}(a, b)$, they will hold in a strict sense when we consider the equivalence classes of $rw_2$-equality. We call this structure $[A_{2rw}(a, b)]$. In this structure, consider the composition of arrows defined as: $[\theta]_{rw_2} \circ [\phi]_{rw_2} = [\theta \circ \phi]_{rw_2}$.

Now, we can think of the structure $[2 - A_{rw}]$. This structure is similar to $2 - A_{rw}$. The difference is that the categories of the second level are $[A_{2rw}(a, b)]$ instead of $A_{2rw}(a, b)$. We can now prove that $[2 - A_{rw}]$ is a weak 2-category:

**Proposition 4.2** Computational paths induce a weak 2-category called $[2 - A_{rw}]$.

**Proof.** First of all, let’s draw a diagram that represents $[2 - A_{rw}]$:
In this diagram we represent 1-arrows and 2-arrows between these 1-arrows. The fact that 2-arrows are equivalence classes is represented by the brackets.

Given $[\alpha]_{rw_2} : [s = \alpha_1, \ldots, \alpha_n = t]$ and $[\theta]_{rw_2} : [r = \theta_1, \ldots, \theta_m = w]$, then we define the horizontal composition $(([\theta]_{rw_2} \circ_h [\alpha]_{rw_2})$ as the sequence $[\tau(s = \alpha_1, r = \theta_1), \ldots, \tau(\alpha_n, \theta_1), \ldots, \tau(\alpha_n, t, \theta_m = w)]_{rw_2}$.

We also need to verify the associative and identity law for $\circ_h$. Since we are working with a weak 2-category, these laws should hold up to natural isomorphism [12]. To verify these laws, the idea is that every 2-morphism of $[A_{rw_2}]$ is an isomorphism. The proof of this fact is analogous to the one of Proposition 5.1, but using $rw_2$-rules instead of $rw$-rules. Since a natural transformation is a natural isomorphism iff every component is an isomorphism (as one can check in [1]), we conclude that finding isomorphisms for the associative and identity laws is just a matter of finding the correct morphisms.

For the associative law, we need to check that there is a natural isomorphism $\text{assoc}$ between $(([\psi]_{rw_2} \circ_h [\theta]_{rw_2}) \circ_h [\alpha]_{rw_2})$ and $([\psi]_{rw_2} \circ_h ([\theta]_{rw_2} \circ_h [\alpha]_{rw_2}))$. To do this, by the definition of horizontal composition, a component of $(([\psi]_{rw_2} \circ_h [\theta]_{rw_2}) \circ_h [\alpha]_{rw_2})$ is a term of the form $\tau(x, \tau(\theta(y, \psi)), z)$, with $x, y,$ and $z$ being suitable natural numbers that respect the order of horizontal composition. Analogously, the same component of $([\psi]_{rw_2} \circ_h ([\theta]_{rw_2} \circ_h [\alpha]_{rw_2}))$ is just a suitable term $\tau(\tau(\alpha, \theta(y), \psi), \tau(\alpha, \theta(y), \psi))$.

The isomorphism between these component is clearly established by the inverse $tt$ rule, i.e., $\tau(\tau(\alpha, \theta(y), \psi)) = [\psi]_{rw_{2tt}}(\tau(\alpha, \theta(y), \psi))$

The identity laws use the same idea. We need to check that $([\alpha]_{rw_2} \circ_h [\rho_{p_{a}}]_{rw_2}) = [\alpha]_{rw_2}$. To do that, we need to take components $(\rho_{p_{a}}, \alpha_{y})$ and $\alpha_{y}$ and establish their isomorphism $\text{r}_{w_{2tt}} : (\rho_{p_{a}}, \alpha_{y}) = \rho_{p_{a}} \circ_h \alpha_{y}$.

The other natural isomorphism $\text{r}_{w_{2tt}}$, i.e., the isomorphism between $([\rho_{p_{a}}]_{rw_2} \circ_h [\alpha]_{rw_2})$ and $[\alpha]_{rw_2}$ can be established in an analogous way, just using the rule $trr$ instead of $tlt$. Just for purpose of clarification, $\rho_{p_{a}}$ comes from the reflexive property of $rw$-equality. Since $\rho_{a}$ is the identity path, using the reflexivity we establish that $\rho_{a} = \rho_{a}$.

With the associative and identity isomorphisms established, we now need to check the interchange law [12]. We need to check that:

$([\varphi]_{rw_2} \circ \theta_{2}) \circ_h ([\chi]_{rw_2} \circ \alpha_{2}) = ([\varphi]_{rw_2} \circ_h [\chi]_{rw_2}) \circ ([\theta]_{rw_2} \circ_h [\alpha]_{rw_2})$

From $(([\varphi]_{rw_2} \circ \theta_{2}) \circ_h ([\chi]_{rw_2} \circ \alpha_{2}))$, we have:

$([\varphi]_{rw_2} \circ \theta_{2}) \circ_h ([\chi]_{rw_2} \circ \alpha_{2}) = ([\varphi]_{rw_2} \circ \theta_{2}) \circ_h ([\chi]_{rw_2} \circ \alpha_{2}) = [	au(\theta, \varphi)]_{rw_2} \circ_h [\tau(\alpha, \chi)]_{rw_2} = [\theta_{1}, \ldots, \theta_{m} = \varphi_{1}, \ldots, \varphi_{m'}]_{rw_2} \circ_h [\alpha_{1}, \ldots, \alpha_{m} = \chi_{1}, \ldots, \chi_{m'}]_{rw_2} = [	au(\alpha_{1}, \theta_{1}), \ldots, \tau(\alpha_{m} = \chi_{1}, \theta_{1}), \ldots, \tau(\chi_{n}, \theta_{1}), \ldots, \tau(\chi_{n}, \varphi_{n'})]_{rw_2}$

From $(([\varphi]_{rw_2} \circ_h [\chi]_{rw_2}) \circ ([\theta]_{rw_2} \circ_h [\alpha]_{rw_2}))$: 

10
\[(\langle [\varphi]_{rw_2} \circ_h [\chi]_{rw_2} \rangle \circ \langle [\theta]_{rw_2} \circ_h [\alpha]_{rw_2} \rangle) (r \circ s) = \]
\[(\langle [\tau(\chi_1, \varphi_1), ..., \tau(\chi_n, \varphi_1), ..., \tau(\chi_n, \varphi_n')]_{rw_2} \rangle \circ \]
\[\langle [\tau(\alpha_1, \theta_1), ..., \tau(\alpha_m, \theta_1), ..., \tau(\alpha_m, \theta_m')]_{rw_2} \rangle = \]
\[\langle [\tau(\alpha_1, \theta_1), ..., \tau(\alpha_m, \theta_1), ..., \tau(\alpha_m, \theta_m'), ..., \tau(\chi_n, \varphi_1), ..., \tau(\chi_n, \varphi_n')]_{rw_2} \rangle \]

If one looks closely, one can notice that this is suitable to apply \(cd_2\). Individually, every variable that appears in the sequence of transitivities follows the same expansion in both cases. The only difference is how the choices have been made. Therefore, the \(rw_2\)-equality is established by \(cd_2\). Since we are working with equivalence classes, this equality holds strictly.

About the coherence laws, one can easily check that Mac Lane’s pentagon and triangle diagrams are commutative by simple and straightforward applications of natural isomorphisms \(assoc, r^s\) and \(l^s\). One can check appendix A for a detailed proof of this part. \(\square\)

We can also conclude that \([2 - A_{rw}]\) has a weak 2-groupoid structure. That is the case because we already know (from proposition 4.4) that the groupoid laws are satisfied by 1-morphisms up to the isomorphism of the next level, i.e., up to \(rw\)-equality and the 2-morphisms, as we have just seen, are isomorphisms (that hold in a strict way, since the second level is using classes of equivalence).

Our objective in a future work is to define higher levels of \(rw\)-equalities and, from that, we will try to obtain even higher groupoid structures. Eventually, our goal will be the construction of a weak \(\omega\)-groupoid. We believe that it is possible to achieve these results, since it was proved by [13,16] that the identity type induces such structure. Given the connection between computational paths and terms of identity type, we should be able to prove that computational paths also induces a weak \(\omega\)-groupoid.

\section{Conclusion}

The main purpose of this work was the study of mathematical properties of an entity known as computational paths. Inspired by our previous work, which has established the connection between computational paths and the identity type [14], our main motivation was the belief that some of the properties of the identity type of Martin Löf’s Intensional Type Theory should be present on computational paths. The property that we have investigated is the fact that the identity type induces a groupoid structure. Our idea has been the fact that computational paths should also be capable of naturally inducing such structure.

To achieve our results, we have defined a computational path as a sequence of equality identifiers between two terms of a type, where each identifier is either an axiom of the equality theory of Type Theory or a change of bound variables. Then, we have shown, using examples, that the usage of the equality axioms can cause redundancies within a computational path. We mentioned that there already exists a system called \(LND_{EQ} - TRS\) that maps and resolves these redundancies. We have gone further, proposing the existence of higher \(LND_{EQ} - TRS_n\) systems, using the idea that a \(LND_{EQ} - TRS_n\) resolves the redundancies of a \(LND_{EQ} - TRS_{n-1}\). Using \(LND_{EQ} - TRS\), we have proved that a computational path is capable of
inducing a weak groupoid structure. Using the higher rewriting systems, we have showed that it is possible to induce a structure known as weak 2-groupoid. Based on the fact that there exist infinite $LND_{EQ} - TRS_n$ systems, we opened the way, in a future work, for a possible proof establishing that computational paths induces a weak $w$-groupoid.

References

[1] Steve Awodey. *Category theory*. Oxford University Press, 2010.

[2] Steve Awodey. Type theory and homotopy. In P. Dybjer, Sten Lindström, Erik Palmgren, and G. Sundholm, editors, *Epistemology versus Ontology*, volume 27 of *Logic, Epistemology, and the Unity of Science*, pages 183–201. Springer Netherlands, 2012.

[3] A. G. de Oliveira. *Proof transformations for labelled natural deduction via term rewriting*, 1995. Master’s thesis, Depto. de Informática, Universidade Federal de Pernambuco, Recife, Brazil, April 1995.

[4] 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 IGPL*, 7(2):173–215, 1999.

[5] R. J. G. B. de Queiroz and A. G. de Oliveira. Term rewriting systems with labelled deductive systems. In *Proceedings of Brazilian Symposium on Artificial Intelligence (SBIA ’94)*, pages 59–72, 1994.

[6] R. J. G. B. de Queiroz and A. G. de Oliveira. Propositional equality, identity types, and direct computational paths. 2013. http://arxiv.org/abs/1107.1901.

[7] R. J. G. B. de Queiroz and A. G. de Oliveira. Natural deduction for equality: The missing entity. In *Advances in Natural Deduction*, pages 63–91. Springer, 2014.

[8] R. J. G. B. de Queiroz, A. G. de Oliveira, and D. M. Gabbay. *The Functional Interpretation of Logical Deduction*. World Scientific, 2011.

[9] R. J. G. B. de Queiroz and D. M. Gabbay. Equality in labelled deductive systems and the functional interpretation of propositional equality. In *Proceedings of the 9th Amsterdam Colloquium*, pages 547–565. ILLC/Department of Philosophy, University of Amsterdam, 1994.

[10] J. Roger Hindley and Jonathan P. Seldin. *Lambda-calculus and combinators: an introduction*. Cambridge University Press, 2008.

[11] Martin Hofmann and Thomas Streicher. The groupoid model refutes uniqueness of identity proofs. In *Logic in Computer Science, 1994. LICS ’94. Proceedings*, Symposium on, pages 208–212. IEEE, 1994.

[12] Tom Leinster. Basic bicategories. *arXiv preprint math.CT/9810017*, 1998.

[13] Peter LeFanu Lumsdaine. Weak $\omega$-categories from intensional type theory. In *Typed lambda calculi and applications*, volume 5608 of *LNCS*, pages 172–187. Springer, 2009.

[14] Arthur F. Ramos, R. J. G. B. de Queiroz, and A. G. de Oliveira. On the identity type as the type of computational paths. 2015. http://arxiv.org/abs/1504.04759.

[15] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. https://homotopytypetheory.org/book, Institute for Advanced Study, 2013.

[16] Benno van den Berg and Richard Garner. Types are weak $\omega$-groupoids. *Proceedings of the London Mathematical Society*, 102(2):370–394, 2011.

[17] V. Voevodsky. Univalent foundations and set theory. Univalent Foundations and Set Theory, Lecture at IAS, Princeton, New Jersey, Mar 2014.
A Coherence Laws

In this section, we prove that Mac Lane’s pentagon holds for $[2 - \mathbb{A}]$. In other words, we should check the following fact:

Given the following diagram of $1$-morphisms:

The following diagrams should commute:

The proofs are straightforward. Remember that $\text{assoc}$ is just an application of $\sigma(tl)$, $r^*_r$ is an application of $\text{trr}$ and $l^*_s$ an application of $\text{tlr}$. First, let’s start with $((u \circ p) \circ r) \circ s = \tau(s, \tau(r, \tau(p, u)))$ going to the right of the diagram:

Now, starting from the same $\tau(s, \tau(r, \tau(p, u)))$ and going bottom left:

Therefore, the first diagram commutes. We now need to check the second one.
Let’s start from \(((r \circ \rho_b) \circ s) = \tau(s, \tau(\rho_b, r))\) and going to the right of the diagram:

\[
\begin{align*}
assoc(\tau(s, \tau(\rho_b, r))) &= \tau(\tau(s, \rho_b), r) \\
([\rho_r]_{rw_2} \circ h_l^*) \tau(s, \rho_b, r) &= \tau((l_r^* \tau(s, \rho_b)), r) = \\
\tau(s, r) &= r \circ s
\end{align*}
\]

Now, starting from the same \(\tau(s, \tau(\rho_b, r))\) and going right bottom:

\[
\begin{align*}
(r_r^* \circ h \rho_s) \tau(s, \tau(\rho_b, r)) &= \tau(s, r_r^* \tau(\rho_b, r)) = \\
\tau(s, r) &= r \circ s
\end{align*}
\]

Thus, the second diagram commutes. The coherence laws hold.