Coherence for Monoidal Groupoids in HoTT

Stefano Piceghello
Department of Informatics and Department of Mathematics, University of Bergen, Norway
stefano.piceghello@uib.no

Abstract

We present a proof of coherence for monoidal groupoids in homotopy type theory. An important role in the formulation and in the proof of coherence is played by groupoids with a free monoidal structure; these can be represented by 1-truncated higher inductive types, with constructors freely generating their defining objects, natural isomorphisms and commutative diagrams. All results included in this paper have been formalised in the proof assistant Coq.

2012 ACM Subject Classification Theory of computation → Type theory; Theory of computation → Constructive mathematics

Keywords and phrases homotopy type theory, coherence, monoidal categories, groupoids, higher inductive types, formalisation, Coq

Digital Object Identifier 10.4230/LIPIcs.TYPES.2019.8

Supplementary Material Formalised proofs are available at `https://github.com/spiceghello/FSMG`.

Acknowledgements The author wishes to thank Kristian S. Alfsvåg, Marc Bezem, Floris van Doorn, Bjørn Ian Dundas, Peter Dybjer, Favonia and Håkon R. Gylterud for providing valuable insight on the topic and the anonymous reviewers for constructive comments, and acknowledges the support of the Centre for Advanced Study (CAS) in Oslo, Norway, which funded and hosted the research project *Homotopy Type Theory and Univalent Foundations* during the 2018/19 academic year.

1 Introduction

Homotopy type theory (HoTT) [21] is a version of Martin-Löf type theory, enhanced with a definition of identity types that allows the interpretation of terms in a type as points in spaces, and terms in identity types as paths; iterating the construction of identity types promotes the interpretation of types as ∞-groupoids [17, 21, 3]. One of the key features of HoTT is the use of higher inductive types (HITs) [18, 21, 8], which integrate inductive constructions with the higher groupoid structure of types. This has lead to the formalisation of numerous results and computations in homotopy theory (see e.g. [21, Chapter 8] for a nonexhaustive list), which have largely been checked using proof assistants such as Coq [12, 11], Agda [7] and Lean [9]. In this paper we employ the functionalities of HITs to formalise in HoTT the result in category theory known as coherence for monoidal categories.

A (weak) *monoidal category* consists of a category $\mathcal{C}$ together with a *monoidal structure*, i.e. a bifunctor $\bullet : \mathcal{C} \times \mathcal{C} \to \mathcal{C}$ that serves as product, an object $e \in \mathsf{ob}(\mathcal{C})$ that serves as unit for the product, and natural isomorphisms describing associativity and unitality of the product (w.r.t. the unit object) and making two classes of diagrams – namely, the *coherence diagrams* in Figures 1a and 1b – commute. A monoidal category is said to be *strict* if the associativity and unitality natural isomorphisms are identities. Monoidal categories satisfy a theorem of coherence, which states that every monoidal category is monoidally equivalent to a strict one; an equivalent formulation [14, 19] is the following:

▶ **Statement 1.** *Every diagram in a free monoidal category commutes.*

This means that all (nontrivial) diagrams in a free monoidal category can be expressed as combinations of instances of the coherence diagrams and of the naturality diagrams of associativity and unitality.

Different proofs of Statement 1 have been formalised in intensional MLTT in [4, 5] using the proof assistant ALF [20] (with Axiom K) and later in [1] using HOL [10]; there, a category is given by a set of objects and a family of Hom-setoids. After noticing that a free monoidal category is a groupoid (since all its arrows are built upon instances of the natural isomorphisms defining the monoidal structure, and hence they are invertible), a version of the same statement can be proved in HoTT by exploiting the mentioned groupoid structure of types (Theorem 7). Indeed, monoidal groupoids can be represented by 1-types, using the correspondence between objects in a groupoid and terms in the type, arrows and paths,[1] and commutative diagrams and 2-paths, where monoidality refers to the presence in the type of the necessary terms, paths and 2-paths to define a monoidal structure. Moreover, HITs offer a way to describe objects satisfying certain universal properties, allowing us to define types that can be interpreted as *free* monoidal groupoids.

Since every path in a type has an inverse, this framework only allows us to express groupoids and not categories; however, a formulation of Statement 1 for groupoids is, from a mathematical standpoint, equivalent to the original one, as the free objects in the categories of monoidal groupoids and of monoidal categories coincide. The choice of proving coherence as formulated in Statement 1, without referring to strict monoidal structures, is also due to the design of the theory, as discussed in Section 5.

The results included in this paper have been formalised using the Coq proof assistant (see Section 6 for further details); the code is available as supplementary material.

Background and Notation

We assume familiarity with the basics of HoTT; our main reference is [21], from which we largely borrow our notation. In particular:

- we will use the symbol $\equiv$ for judgmental equality and $:\equiv$ for definitions;
- we will not distinguish, in the notation, between different members of an assumed hierarchy of universes, and will instead denote them uniformly by $\mathcal{U}$;
- for a family of types $B : A \to \mathcal{U}$, dependent functions are denoted by $f :\equiv (x \mapsto f(x)) : \Pi(x : A).B(x)$; the identity function is indicated by $\mathsf{id}_A :\equiv \mathsf{id} : A \to A$; dependent pairs are denoted by $\langle a, b\rangle : \Sigma(x : A).B(x)$; terms in nested Σ-types are denoted by tuples $\langle a, b, c, \ldots\rangle$;
- identity types are denoted by $x =_A y$ or simply $x = y$ for $x, y : A$; their terms are called *paths* in A and their elimination principle is called *path induction*; identity (reflexivity) paths, which inductively generate identity types, are denoted by $\mathsf{refl}_x :\equiv \mathsf{refl} : x = x$; terms $r : p = q$, where p and q are paths in a type, are called 2-*paths*, and so on;
- the inverse of a path p is denoted by p^{-1}; the concatenation of paths $p : x = y$ and $q : y = z$ is denoted by $p \cdot q : x = z$; this operation is provably associative and unital with respect to the identity path and it satisfies inverse laws, giving rise to the umbrella-expression "path algebra" to encompass all proofs of existence of 2-paths of the sort;
- the action of $f : A \to B$ on a path $p : x =_A y$ is denoted by $\mathsf{ap}_f(p) : f(x) =_B f(y)$; functoriality of ap will also be referred to as path algebra;

[1] Observe that, by their very nature, the categories we are describing are univalent.

- given a family of types $P : A \to \mathcal{U}$ and a path $p : x =_A y$, the transport of terms in $P(x)$ along p is denoted by $p^P_* :\equiv p_* : P(x) \to P(y)$; the action of $f : \Pi(a : A).P(a)$ on p is indicated by $\mathsf{apd}_f(p) : p_*(f(x)) =_{P(y)} f(y)$; a term in the identity type $p_*(u) = v$ is called a *pathover* [15];
- several results are assumed about transporting in families of paths; in particular, we will implicitly use that, given functions $f, g : A \to B$ and paths $p : x =_A y$ and $q : f(x) =_B g(x)$, there is a (2-)pathover of type $p_*^{(a \mapsto f(a) = g(a))}(q) = \mathsf{ap}_f(p)^{-1} \cdot q \cdot \mathsf{ap}_g(p)$;
- pointwise equalities of functions f and $g : A \to B$ are called *homotopies* and denoted by $f \sim g :\equiv \Pi(x : A).f(x) = g(x)$; if $h : B \to A$ and $f \circ h \sim \mathsf{id}_B$, f is said to be a *retraction* of h and B a *retract* of A;
- we denote by $A \simeq B :\equiv \Sigma(f : A \to B, g : B \to A).(g \circ f \sim \mathsf{id}_A) \times (f \circ g \sim \mathsf{id}_B)$ the type of *equivalences* between A and B; f and g are said to be half adjoint in such an equivalence;
- the prefix "0-" or "1-" for types refers to their *truncation level*; a 0-type is a type A such that there is a term in $\Pi(x, y : A).\Pi(p, q : x = y).p = q$, embodying the notion of a *set* of terms, while A is a 1-type (i.e. a *groupoid*) if all its identity types are 0-types; every 0-type is a 1-type; the property of A being a 1-type is denoted by $\mathsf{IsTrunc}_1(A)$;
- as mentioned, the theory assumes HITs; in the presentation we will informally call "computation rules" also the assumed identities involving the (dependent) application of the elimination principle of a HIT on higher constructors, although no computation takes place [21, Chapter 6].

We will, moreover, refer to a (2-)path as "trivial" if it is either the identity path or it can be obtained by path algebra.

In figures presenting 2-paths, we choose to display paths $p : x = y$ as arrows $x \to y$, both to preserve the analogy with categories and to keep track of the endpoints; in such diagrams, all arrows are invertible, as all paths are. A dotted line denotes instead judgmental equality. Figures relevant to proofs are included in Appendix A.

2 Monoidal Groupoids

In this section we will provide the definitions of the objects of our study, building a categorical framework within which to formulate the theorem of coherence for monoidal groupoids.

▶ **Definition 2.** *A* ***groupoid*** *is the data given by a type G and a proof that G is a 1-type; we call* $\mathsf{Gpd} :\equiv \Sigma(G : \mathcal{U}).\mathsf{IsTrunc}_1(G)$ *the subuniverse of 1-types in the universe $\mathcal{U}$. A (groupoid)* ***functor*** *F between groupoids is simply a function between the underlying (1-)types; a* ***natural isomorphism*** *between two functors is a homotopy between the functions.*

In Definition 2 we bestow the title of "functor" on simple functions, as the functorial action on paths is provided by ap. We will use the same notation for a groupoid $G : \mathsf{Gpd}$ and its underlying type $G : \mathcal{U}$. This framework allows us to represent categories with all arrows invertible (i.e. *groupoids*) as 1-types, with paths taking the role of arrows and 2-paths that of commutative diagrams.[2]

▶ **Definition 3.** *For a type M, the type* $\mathsf{MonStr}(M)$ *of* ***monoidal structures*** *on M is the Σ-type encoding the following data:*

[2] This translation implies a "relaxation" of certain strict categorical properties: for example, associativity of the composition of the arrows in a category is strict, while associativity of concatenation of paths only holds up to a coherent choice of higher paths. Moreover, we remark that, given a groupoid $G : \mathsf{Gpd}$, we do not have access to the discrete subcategory of its objects.

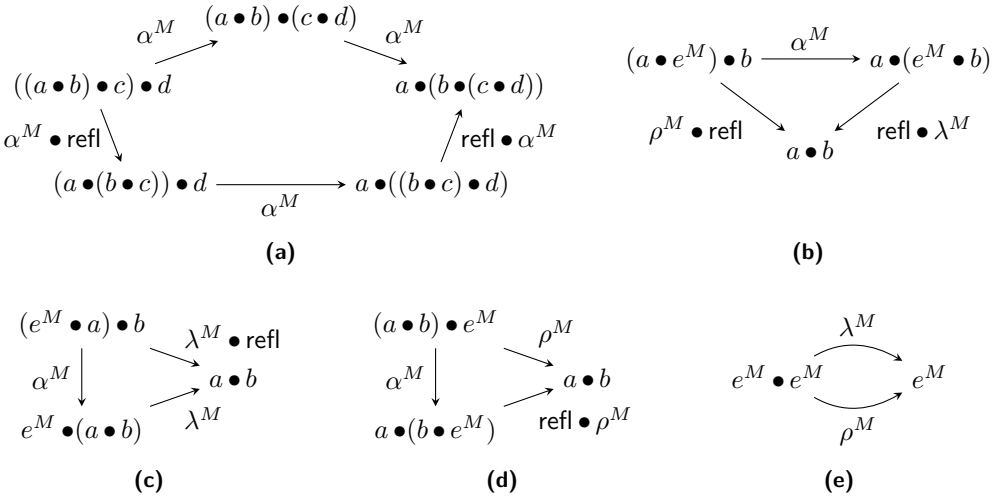


Figure 1 **(a)** and **(b)**: coherence diagrams $\pentagon^M$ and $\triangledown^M$, respectively, where α^M, λ^M and ρ^M are associativity and left and right unitality morphisms; **(c)**, **(d)** and **(e)**: coherence diagrams derivable from $\pentagon^M$, $\triangledown^M$ and naturality of α^M, λ^M and ρ^M; the derivation is shown in Figures 5 and 6. Here $\bullet^M$ is denoted by $\bullet$ for clarity.

- $e^M : M$ *(unit);*
- $\bullet^M : M \to M \to M$ *(monoidal product, infix notation);*
- $\alpha^M : \Pi(a, b, c : M).(a \bullet^M b) \bullet^M c = a \bullet^M (b \bullet^M c)$ *(associativity);*
- $\lambda^M : \Pi(b : M).e^M \bullet^M b = b$ *(left unitality);*
- $\rho^M : \Pi(a : M).a \bullet^M e^M = a$ *(right unitality);*
- *families* $\pentagon^M$ *and* $\triangledown^M$ *of* 2*-paths filling the coherence diagrams in Figures 1a and 1b.*

The type of ***monoidal groupoids*** *is defined as* $\mathsf{MonGpd} :\equiv \Sigma(M : \mathsf{Gpd}).\mathsf{MonStr}(M)$.

We will use the same notation for a monoidal groupoid M and its carrier. The functorial action of $\bullet^M$ on paths and 2-paths, denoted in this paper by the same symbol, is given by the following functions, each defined by path induction (on both arguments):

$$\bullet^M : (a_1 =_M b_1) \to (a_2 =_M b_2) \to (a_1 \bullet^M a_2 =_M b_1 \bullet^M b_2),$$
$$\bullet^M : (p =_{(a_1 = b_1)} p') \to (q =_{(a_2 = b_2)} q') \to (p \bullet^M q =_{(a_1 \bullet^M a_2 = _M b_1 \bullet^M b_2)} p' \bullet^M q').$$

We emphasise that the given definition of a monoidal structure only pertains the levels of coherence for associativity and unitality that might be present in a (non-higher) groupoid ("1-coherent" monoidal structure, [6]), i.e., no higher diagrams need to be described, as they are present already.

By path induction, α^M, λ^M and ρ^M are natural in all their arguments. Moreover, the 2-paths in Figures 1c to 1e can be derived by instances of the coherence diagrams $\pentagon^M$ and $\triangledown^M$, together with naturality of associativity and unitality [13], as shown in Figures 5 and 6.

▶ **Definition 4.** *The type* $\mathsf{MonGpd}(M, N)$ *of (strong)* ***monoidal functors*** *between two monoidal groupoids* $\langle M, e^M, \bullet^M, \ldots\rangle$ *and* $\langle N, e^N, \bullet^N, \ldots\rangle$ *is defined as the* Σ*-type encoding the following data:*

- *a functor* $F : M \to N$*;*
- *a path* $F_0 : e^N = F(e^M)$ *and a family of paths* $F_2 : \Pi(a, b : M).F(a) \bullet^N F(b) = F(a \bullet^M b)$*;*
- *families of* 2*-paths* F_α*,* F_λ *and* F_ρ*, as depicted in Figure 2.*

$$\begin{array}{ccc}
(F(a) \bullet^N F(b)) \bullet^N F(c) & \xrightarrow{\alpha^N} & F(a) \bullet^N (F(b) \bullet^N F(c)) \\
{\scriptstyle F_2 \bullet^N \mathsf{refl}}\downarrow & & \downarrow{\scriptstyle \mathsf{refl} \bullet^N F_2} \\
F(a \bullet^M b) \bullet^N F(c) & & F(a) \bullet^N F(b \bullet^M c) \\
{\scriptstyle F_2}\downarrow & & \downarrow{\scriptstyle F_2} \\
F((a \bullet^M b) \bullet^M c) & \xrightarrow[\mathsf{ap}_F(\alpha^M)]{} & F(a \bullet^M (b \bullet^M c))
\end{array}$$

$$\begin{array}{ccc}
e^N \bullet^N F(b) & \xrightarrow{\lambda^N} & F(b) \\
{\scriptstyle F_0 \bullet^N \mathsf{refl}}\downarrow & & \uparrow{\scriptstyle \mathsf{ap}_F(\lambda^M)} \\
F(e^M) \bullet^N F(b) & \xrightarrow[F_2]{} & F(e^M \bullet^M b)
\end{array}
\qquad
\begin{array}{ccc}
F(a) \bullet^N e^N & \xrightarrow{\rho^N} & F(a) \\
{\scriptstyle \mathsf{refl} \bullet^N F_0}\downarrow & & \uparrow{\scriptstyle \mathsf{ap}_F(\rho^M)} \\
F(a) \bullet^N F(e^M) & \xrightarrow[F_2]{} & F(a \bullet^M e^M)
\end{array}$$

Figure 2 Coherence conditions $F_\alpha(a, b, c)$, $F_\lambda(b)$ and $F_\rho(a)$ (respectively: top, bottom left and bottom right) for monoidal functors, for a, b, $c : M$.

$$\begin{array}{ccc}
 & e^N & \\
{\scriptstyle F_0}\swarrow & & \searrow{\scriptstyle G_0} \\
F(e^M) & \xrightarrow[\theta(e^M)]{} & G(e^M)
\end{array}
\qquad
\begin{array}{ccc}
F(a) \bullet^N F(b) & \xrightarrow{F_2} & F(a \bullet^M b) \\
{\scriptstyle \theta(a) \bullet^N \theta(b)}\downarrow & & \downarrow{\scriptstyle \theta(a \bullet^M b)} \\
G(a) \bullet^N G(b) & \xrightarrow[G_2]{} & G(a \bullet^M b)
\end{array}$$

Figure 3 Coherence conditions θ_0 and $\theta_2(a, b)$ for monoidal natural isomorphisms, for a, $b : M$.

We will use the same notation for a monoidal functor $F : \mathsf{MonGpd}(M, N)$ and its underlying function. This implementation allows us to provide sound definitions of identity and composition of monoidal functors:

$$\begin{aligned}
(\mathsf{id}_M)_0 &:\equiv \mathsf{refl} : e_M = e_M \\
(\mathsf{id}_M)_2(a, b) &:\equiv \mathsf{refl} : a \bullet^M b = a \bullet^M b, \text{ and} \\
\\
(G \circ F)_0 &:\equiv G_0 \cdot \mathsf{ap}_G(F_0) : e_P = G(F(e_M)) \\
(G \circ F)_2(a, b) &:\equiv G_2(F(a), F(b)) \cdot \mathsf{ap}_G(F_2(a, b)) : G(F(a)) \bullet^P G(F(b)) = G(F(a \bullet^M b)),
\end{aligned}$$

for $F : \mathsf{MonGpd}(M, N)$, $G : \mathsf{MonGpd}(N, P)$ and a, $b : M$; we omit here the 2-paths $(G \circ F)_\alpha$, $(G \circ F)_\lambda$ and $(G \circ F)_\rho$ (appearing in the formalisation), while the corresponding ones for identity monoidal functors are trivial.

▶ **Definition 5.** *The type* $\mathsf{MonFun}_{M,N}(F, G)$ *of* ***monoidal natural isomorphisms*** *between monoidal functors* F, $G : \mathsf{MonGpd}(M, N)$ *is the* Σ*-type encoding a natural isomorphism* $\theta : F \sim G$ *between the underlying functors, together with a* 2*-path* θ_0 *and a family of* 2*-paths* θ_2*, as shown in Figure 3.*

We will use the same notation for a monoidal natural isomorphism $\theta : \mathsf{MonFun}_{M,N}(F, G)$ and its underlying homotopy. If $F : \mathsf{MonGpd}(M, N)$ and $G : \mathsf{MonGpd}(N, M)$, the underlying homotopies in $\eta : \mathsf{MonFun}_{M,M}(\mathsf{id}_M, G \circ F)$ and $\varepsilon : \mathsf{MonFun}_{N,N}(F \circ G, \mathsf{id}_N)$ prove that the functions underlying F and G are half adjoint in an equivalence between (the carriers of) M and N; this will be called a **monoidal equivalence** and denoted by $M \simeq N$.

$$\begin{array}{ccc} \mathsf{MonGpd}(F(X), M) & \xrightarrow{\phi_{X,M}} & (X \to M) \\ H \circ - \big\downarrow & & \big\downarrow H \circ - \\ \mathsf{MonGpd}(F(X), N) & \xrightarrow[\phi_{X,N}]{} & (X \to N) \end{array}$$

(a) Naturality of ϕ in M: the diagram commutes for every $H : \mathsf{MonGpd}(M, N)$, i.e. there is a homotopy $H \circ \phi_{X,M}(G) \sim \phi_{X,N}(H \circ G)$ for every $G : \mathsf{MonGpd}(F(X), M)$.

$$\begin{array}{ccc} (X \to M) & \xrightarrow{\psi_{X,M}} & \mathsf{MonGpd}(F(X), M) \\ - \circ h \big\downarrow & & \big\downarrow - \circ F(h) \\ (Y \to M) & \xrightarrow[\psi_{Y,M}]{} & \mathsf{MonGpd}(F(Y), M) \end{array}$$

(b) Naturality of ψ in X: the diagram commutes for every $h : Y \to X$, i.e. there is a term in $\mathsf{MonFun}_{F(Y),M}(\psi_{X,M}(g) \circ F(h), \psi_{Y,M}(g \circ h))$ for every $g : X \to M$.

Figure 4 Naturality conditions for ϕ and ψ in Definition 6.

▶ **Definition 6.** *A **functor** from the universe $\mathcal{U}$ of types to monoidal groupoids consists of a function term $F : \mathcal{U} \to \mathsf{MonGpd}$, together with a function between function types*

$$\overrightarrow{F} : \Pi(X, Y : \mathcal{U}).(X \to Y) \to \mathsf{MonGpd}(F(X), F(Y))$$

respecting identity and composition, i.e. terms

$$F_{\mathsf{id}} : \Pi(X : \mathcal{U}).\mathsf{MonFun}_{F(X),F(X)} \left(\overrightarrow{F}(\mathsf{id}_X), \mathsf{id}_{F(X)}\right) \text{ and}$$

$$F_{\circ} : \Pi(X, Y, Z : \mathcal{U},\ f : X \to Y,\ g : Y \to Z).\mathsf{MonFun}_{F(X),F(Z)} \left(\overrightarrow{F}(g) \circ \overrightarrow{F}(f), \overrightarrow{F}(g \circ f)\right).$$

*We will refer to a function $F : \mathcal{U} \to \mathsf{MonGpd}$ as a "functor" if the remaining data is implied. Such a functor is **free** if there are:*

- *a function $\phi : \Pi(X : \mathcal{U}).\Pi(M : \mathsf{MonGpd}).\mathsf{MonGpd}(F(X), M) \to X \to M$, natural in M, i.e. the diagram in Figure 4a commutes for every $H : \mathsf{MonGpd}(M, N)$;*
- *a function $\psi : \Pi(X : \mathcal{U}).\Pi(M : \mathsf{MonGpd}).(X \to M) \to \mathsf{MonGpd}(F(X), M)$, natural in X, i.e. the diagram in Figure 4b commutes for every $h : Y \to X$;*
- *a family of homotopies $\theta : \Pi(X : \mathcal{U}).\Pi(M : \mathsf{MonGpd}).\phi_{X,M} \circ \psi_{X,M} \sim \mathsf{id}_{X \to M}$;*
- *a family of monoidal natural isomorphisms*

$$\chi : \Pi(X : \mathcal{U}).\Pi(M : \mathsf{MonGpd}).\Pi(G : \mathsf{MonGpd}(F(X), M)).\ \mathsf{MonFun}_{F(X),M}(\psi_{X,M}(\phi_{X,M}(G)), G).$$

*If $X : \mathcal{U}$, the monoidal groupoid $F(X)$ is said to be **freely generated** by X.*

One can recognise, in the data listed above for the definition of a free functor, the requirements needed to verify that F is left adjoint to the forgetful functor to types, which in this case is the composition of the projections $\mathsf{MonGpd} \to \mathsf{Gpd}$ and $\mathsf{Gpd} \to \mathcal{U}$.

Focussing in Definition 6 on free monoidal groupoids generated by 0-types, Statement 1 can be then formulated as follows:

▶ **Theorem 7** (Coherence for monoidal groupoids)**.** *A free functor $\mathsf{FMG} : \mathcal{U} \to \mathsf{MonGpd}$ exists and, for every 0-type X, the carrier of $\mathsf{FMG}(X)$ is a 0-type.*

Indeed, Theorem 7 both ensures that the construction of a free monoidal groupoid (over a set) is possible, and shows that every diagram in such a groupoid commutes, i.e. that there is a 2-path between every two paths sharing endpoints.

Coherence will be achieved by means of two functors: one will be easily proved to be free (Section 3); the other one (list) to produce monoidal groupoids that are also 0-types. We will show that these two functors produce equivalent types (Section 4).

3 Free Monoidal Groupoids

Higher inductive types allow us to define a functor $\mathsf{FMG} : \mathcal{U} \to \mathsf{MonGpd}$ that contains the proof of its freeness in the elimination principle of the types it produces.

▶ **Definition 8** (FMG). *Given a type $X : \mathcal{U}$, the HIT $\mathsf{FMG}(X)$ is defined with the following constructors:*

$$\begin{aligned}\mathsf{FMG}(X) :\equiv\ & e : \mathsf{FMG}(X) \ \mid\ \iota : X \to \mathsf{FMG}(X) \ \mid\ \bullet : \mathsf{FMG}(X) \to \mathsf{FMG}(X) \to \mathsf{FMG}(X) \\ & \mid\ \alpha : \Pi(a, b, c : \mathsf{FMG}(X)).(a \bullet b) \bullet c = a \bullet (b \bullet c) \\ & \mid\ \lambda : \Pi(b : \mathsf{FMG}(X)).e \bullet b = b \ \mid\ \rho : \Pi(a : \mathsf{FMG}(X)).a \bullet e = a \\ & \mid\ \pentagon : \dots \ \mid\ \triangledown : \dots \ \mid\ T : \mathsf{IsTrunc}_1(\mathsf{FMG}(X)),\end{aligned}$$

where $\pentagon$ and $\triangledown$ are families of 2-path constructors corresponding to the coherence diagrams in Figures 1a and 1b.

For any type X, $\mathsf{FMG}(X)$ is a monoidal groupoid, with the monoidal structure provided by the constructors of the HIT.

▶ Remark 9. The elimination rule of $\mathsf{FMG}(X)$ states that, given a family $P : \mathsf{FMG}(X) \to \mathcal{U}$ together with terms:

- $e' : P(e)$; $\quad \iota' : \Pi(x : X).P(\iota(x))$; $\quad \bullet' : \Pi(a, b : \mathsf{FMG}(X)).P(a) \to P(b) \to P(a \bullet b)$ (infix; we will keep the arguments a and b implicit in the notation);
- a family α' witnessing, for every a, b, $c : \mathsf{FMG}(X)$ and $a' : P(a)$, $b' : P(b)$, $c' : P(c)$, a pathover $(\alpha_{a,b,c})^P_*\,((a' \bullet' b') \bullet' c') = a' \bullet' (b' \bullet' c')$; similarly defined families of pathovers λ' and ρ';
- appropriate families of 2-pathovers $\pentagon'$ and $\triangledown'$;
- $T' : \Pi(w : \mathsf{FMG}(X)).\mathsf{IsTrunc}_1(P(w))$,

there is a function $\mathsf{ind} :\equiv \mathsf{ind}_{\mathsf{FMG}}(e', \iota', \bullet', \dots) : \Pi(w : \mathsf{FMG}(X)).P(w)$, such that

$$\begin{aligned}\mathsf{ind}(e) &\equiv e', & \mathsf{apd}_{\mathsf{ind}}(\alpha_{a,b,c}) &= \alpha'_{\mathsf{ind}(a),\mathsf{ind}(b),\mathsf{ind}(c)}, \\ \mathsf{ind}(\iota(x)) &\equiv \iota'(x), & \mathsf{apd}_{\mathsf{ind}}(\lambda_b) &= \lambda'_{\mathsf{ind}(b)}, \\ \mathsf{ind}(a \bullet b) &\equiv \mathsf{ind}(a) \bullet' \mathsf{ind}(b), & \mathsf{apd}_{\mathsf{ind}}(\rho_a) &= \rho'_{\mathsf{ind}(a)},\end{aligned}$$

for all $x : X$ and a, b, $c : \mathsf{FMG}(X)$. When instantiated to constant families, it provides the following recursor: given a monoidal groupoid $\langle M, e', \bullet', \alpha', \dots\rangle$ and a function $\iota' : X \to M$, there is a function $\mathsf{rec} :\equiv \mathsf{rec}_{\mathsf{FMG}}(e', \iota', \bullet', \dots) : \mathsf{FMG}(X) \to M$ such that

$$\begin{aligned}\mathsf{rec}(e) &\equiv e', & \mathsf{ap}_{\mathsf{rec}}(\alpha_{a,b,c}) &= \alpha'_{\mathsf{rec}(a),\mathsf{rec}(b),\mathsf{rec}(c)}, \\ \mathsf{rec}(\iota(x)) &\equiv \iota'(x), & \mathsf{ap}_{\mathsf{rec}}(\lambda_b) &= \lambda'_{\mathsf{rec}(b)}, \\ \mathsf{rec}(a \bullet b) &\equiv \mathsf{rec}(a) \bullet' \mathsf{rec}(b), & \mathsf{ap}_{\mathsf{rec}}(\rho_a) &= \rho'_{\mathsf{rec}(a)},\end{aligned} \tag{3.1}$$

for all $x : X$ and a, b, $c : \mathsf{FMG}(X)$; this is also the underlying function of a monoidal functor, with rec_0 and $\mathsf{rec}_2(a, b)$ being identity paths for every a, $b : \mathsf{FMG}(X)$ and rec_α, rec_λ, rec_ρ given by the computation rules in (3.1).

The construction FMG is functorial, as shown in the following lemma.

▶ **Lemma 10.** *The function $\mathsf{FMG} : \mathcal{U} \to \mathsf{MonGpd}$, $X \mapsto \langle\langle \mathsf{FMG}(X), T\rangle, e, \bullet, \alpha, \lambda, \rho, \pentagon, \triangledown\rangle$ is a functor.*

Proof. Let $f : X \to Y$ be a function of types. By Remark 9, a function $\iota' : X \to \mathsf{FMG}(Y)$ is sufficient to define a monoidal functor $\mathsf{FMG}(f) : \mathsf{MonGpd}(\mathsf{FMG}(X), \mathsf{FMG}(Y))$; this can be given by $\iota' :\equiv \iota \circ f$. The proof that FMG respects identity and composition is given in detail in the Coq formalisation included as supplementary material for this paper, and it is also provided by the elimination rule of $\mathsf{FMG}(X)$. By way of example, given a type $X : \mathcal{U}$, a monoidal natural isomorphism

$$\mathsf{FMG}_{\mathsf{id}} : \mathsf{MonFun}_{\mathsf{FMG}(X),\mathsf{FMG}(X)}(\mathsf{FMG}(\mathsf{id}_X), \mathsf{id}_{\mathsf{FMG}(X)})$$

has as underlying homotopy

$$\mathsf{FMG}_{\mathsf{id}} : \Pi(w : \mathsf{FMG}(X)).\mathsf{rec}_{\mathsf{FMG}}(\mathsf{FMG}(X), e, \iota, \bullet, \ldots)(w) = w$$

the function $\mathsf{FMG}_{\mathsf{id}} :\equiv \mathsf{ind}_{\mathsf{FMG}}(e', \iota', \bullet', \ldots)$, with:

- $e' :\equiv \mathsf{refl} : e = e$;
- $\iota'(x) :\equiv \mathsf{refl} : \iota(x) = \iota(x)$ for every $x : X$;
- $a' \bullet' b' : \mathsf{rec}_{\mathsf{FMG}}(a) \bullet \mathsf{rec}_{\mathsf{FMG}}(b) = a \bullet b$, for a, $b : \mathsf{FMG}(X)$, $a' : \mathsf{rec}_{\mathsf{FMG}}(a) = a$ and $b' : \mathsf{rec}_{\mathsf{FMG}}(b) = b$, is given recursively by $a' \bullet b'$ (so that $\mathsf{FMG}_{\mathsf{id}}(a \bullet b)$ will compute to $\mathsf{FMG}_{\mathsf{id}}(a) \bullet \mathsf{FMG}_{\mathsf{id}}(b)$);
- the other required terms are obtained by naturality of associativity and unitality, together with the computation rules of $\mathsf{rec}_{\mathsf{FMG}}$.

With this definition, the diagrams in Figure 3 for $\mathsf{FMG}_{\mathsf{id}}$ commute trivially. ◀

As hinted by the universal property of $\mathsf{FMG}(X)$, given by its elimination rule, the functor FMG is free.

▶ **Proposition 11.** FMG *is a free functor, and hence the monoidal groupoid* $\mathsf{FMG}(X)$ *is freely generated by* X*, for every* $X : \mathcal{U}$*.*

Proof. We will proceed to fulfil the requirements listed in Definition 6 for a free functor.

- For $X : \mathcal{U}$ and $M : \mathsf{MonGpd}$, a function $\phi_{X,M} : \mathsf{MonGpd}(\mathsf{FMG}(X), M) \to X \to M$ is given by $\phi_{X,M}(G) :\equiv G \circ \iota$. Then, given a monoidal functor $H : \mathsf{MonGpd}(M, N)$, we have $H \circ \phi_{X,M}(G) \equiv \phi_{X,N}(H \circ G)$, so the diagram in Figure 4a commutes judgmentally (and hence pointwise) and ϕ is natural in M.
- Referring to Remark 9, for $X : \mathcal{U}$ and $M : \mathsf{MonGpd}$, a function $\psi_{X,M} : (X \to M) \to \mathsf{MonGpd}(\mathsf{FMG}(X), M)$ is immediately obtained, defining

  $$\psi_{X,M}(g) :\equiv \mathsf{rec}_{\mathsf{FMG}}(e^M, g, \bullet^M, \alpha^M, \ldots) : \mathsf{FMG}(X) \to M,$$

  where e^M, $\bullet^M$, α^M, … are the components of the monoidal structure of M, since the recursor of FMG is a monoidal functor. If $h : Y \to X$ is a function of types, a monoidal natural isomorphism

  $$\theta_\psi : \mathsf{MonFun}_{\mathsf{FMG}(Y),M}(\psi_{X,M}(g) \circ \mathsf{FMG}(h), \psi_{Y,M}(g \circ h))$$

  witnessing naturality of ψ in X can be given as follows. The natural transformation between the underlying monoidal functors

  $$\theta_\psi : \Pi(w : \mathsf{FMG}(Y)).\psi_{X,M}(g)(\mathsf{FMG}(h)(w)) = \psi_{Y,M}(g \circ h)(w)$$

  is defined as $\theta_\psi :\equiv \mathsf{ind}_{\mathsf{FMG}}(e', \iota', \bullet', \ldots)$, where:
  - $e' :\equiv \mathsf{refl} : \psi_{X,M}(g)(\mathsf{FMG}(h)(e)) = \psi_{Y,M}(g \circ h)(e)$, as both sides of the equality compute to e^M;

- $\iota'(y) :\equiv \mathsf{refl} : \psi_{X,M}(g)(\mathsf{FMG}(h)(\iota(y))) = \psi_{Y,M}(g \circ h)(\iota(y))$, for every $y : Y$, as both sides of the equality compute to $g(h(y))$;
- $a' \bullet' b' :\equiv a' \bullet^M b' : \psi_{X,M}(g)(\mathsf{FMG}(h)(a \bullet b)) = \psi_{Y,M}(g \circ h)(a \bullet b)$ for every a, $b : \mathsf{FMG}(Y)$, $a' : \psi_{X,M}(g)(\mathsf{FMG}(h)(a)) = \psi_{Y,M}(g \circ h)(a)$ and $b' : \psi_{X,M}(g)(\mathsf{FMG}(h)(b)) = \psi_{Y,M}(g \circ h)(b)$, as the equation computes to

$$\psi_{X,M}(g)(\mathsf{FMG}(h)(a)) \bullet^M \psi_{X,M}(g)(\mathsf{FMG}(h)(b)) = \psi_{Y,M}(g \circ h)(a) \bullet^M \psi_{Y,M}(g \circ h)(b);$$

  this way, $\theta_\psi(a \bullet b)$ will compute to $\theta_\psi(a) \bullet^M \theta_\psi(b)$;
- the families α', λ' and ρ' of pathovers are given by naturality of α^M, λ^M and ρ^M, together with the computation rules of $\mathsf{rec}_{\mathsf{FMG}}$ in (3.1); for example, α' corresponds to a 2-path filling the diagram in Figure 7;
- the families $\pentagon'$ and $\triangledown'$ of 2-pathovers are trivially given, since they correspond to 3-paths in a 1-type.

The 2-paths $(\theta_\psi)_0$ and $(\theta_\psi)_2(a, b)$ corresponding to the diagrams in Figure 3 for a, $b : \mathsf{FMG}(Y)$ are then trivial; hence, a monoidal natural isomorphism making the diagram in Figure 4b commute is provided and $\psi_{X,M}$ is natural in X.

- A homotopy $\theta : \phi_{X,M} \circ \psi_{X,M} \sim \mathsf{id}_{X \to M}$, for every $X : \mathcal{U}$ and $M : \mathsf{MonGpd}$ is trivially given, since, for every $g : X \to M$, we have $\phi_{X,M}(\psi_{X,M}(g)) \equiv \psi_{X,M}(g) \circ \iota \equiv g$.
- A monoidal natural isomorphism $\chi : \mathsf{MonFun}_{\mathsf{FMG}(X),M}(\psi_{X,M}(\phi_{X,M}(G)), G)$ for every $X : \mathcal{U}$, $M : \mathsf{MonGpd}$ and $G : \mathsf{MonGpd}(\mathsf{FMG}(X), M)$ is given as follows. The natural transformation between the underlying monoidal functors

$$\chi : \Pi(w : \mathsf{FMG}(X)).\psi_{X,M}(\phi_{X,M}(G))(w) = G(w)$$

  can be defined as $\chi :\equiv \mathsf{ind}_{\mathsf{FMG}}(e', \iota', \bullet', \ldots)$, where:
  - $e' :\equiv G_0 : \psi_{X,M}(\phi_{X,M}(G))(e) = G(e)$, since the left-hand side of the equality computes to e^M;
  - $\iota'(x) :\equiv \mathsf{refl} : \psi_{X,M}(\phi_{X,M}(G))(\iota(x)) = G(\iota(x))$ for $x : X$, as both sides of the equality are judgmentally equal to to $\phi_{X,M}(G)(x)$;
  - $a' \bullet' b' : \psi_{X,M}(\phi_{X,M}(G))(a \bullet b) = G(a \bullet b)$ is given recursively, for a, $b : \mathsf{FMG}(X)$, $a' : \psi_{X,M}(\phi_{X,M}(G))(a) = G(a)$ and $b' : \psi_{X,M}(\phi_{X,M}(G))(b) = G(b)$, by the concatenation:

$$\begin{aligned}
&\psi_{X,M}(\phi_{X,M}(G))(a \bullet b) \\
&\quad \equiv \psi_{X,M}(\phi_{X,M}(G))(a) \bullet^M \psi_{X,M}(\phi_{X,M}(G))(b) \\
&\quad = G(a) \bullet^M G(b) && \text{by } a' \bullet^M b' \\
&\quad = G(a \bullet b) && \text{by } G_2(a, b);
\end{aligned}$$

    this way, $\chi(a \bullet b)$ will compute to $(\chi(a) \bullet^M \chi(b)) \cdot G_2(a, b)$;
  - the families α', λ' and ρ' of pathovers are given by the computation rules of $\psi_{X,M}$, naturality of α^M, λ^M and ρ^M, and by G_α, G_λ and G_ρ; Figure 8 shows α', while the other families are obtained similarly;
  - again, the families $\pentagon'$ and $\triangledown'$ of 2-pathovers are trivially given.

With this definition of the underlying homotopy χ, there are trivial paths χ_0 and $\chi_2(a, b)$ corresponding to the diagrams in Figure 3, making χ into a monoidal natural isomorphism. ◀

4 Coherence

This section is devoted to the proof of Theorem 7; from here on, X is assumed to be a 0-type. Coherence will be proved by exhibiting functions $K : \mathsf{FMG}(X) \to \mathsf{list}(X)$ and $J : \mathsf{list}(X) \to \mathsf{FMG}(X)$, where the latter is a retraction of the former.

The type $\mathsf{list}(X)$ has a monoidal structure that sees the operation of list concatenation as the monoidal product, which by list-elimination can be proved associative and unital w.r.t. the unit given by the empty list. Families of coherence 2-paths $\pentagon^{\mathsf{list}}$ and $\triangledown^{\mathsf{list}}$ are provided by the truncation level of $\mathsf{list}(X)$, which is a 0-type (because X is); the ensuing construction $\mathsf{list} : \mathcal{U} \to \mathsf{MonGpd}$, when restricted to 0-types, satisfies the conditions to be a functor in the sense of Definition 6. As a matter of fact, $\mathsf{FMG}(X)$ and $\mathsf{list}(X)$ as monoidal groupoids can be shown to be in a monoidal equivalence, where the half adjoint monoidal functors are built upon the mentioned functions K and J. We will only make use of the monoidal components of J to prove the retraction; a complete proof of the monoidal equivalence is present in the Coq formalisation included to this paper as supplementary material.

▶ Remark 12. We will use the following notation and conventions. The constructors of $\mathsf{list}(X)$ are the empty list $\mathsf{nil} : \mathsf{list}(X)$ and $:: : X \to \mathsf{list}(X) \to \mathsf{list}(X)$ (infix). Concatenation of lists $+\!\!+$ (infix) is defined by list-elimination on the first argument, i.e. $\mathsf{nil} +\!\!+ l_2 :\equiv l_2$ and $(x :: l_1) +\!\!+ l_2 :\equiv x :: (l_1 +\!\!+ l_2)$ for every $x : X$ and $l_1, l_2 : \mathsf{list}(X)$. The components of the monoidal structure of $\mathsf{list}(X)$, besides nil and list concatenation, are $\lambda^{\mathsf{list}} : \Pi(l : \mathsf{list}(X)).\mathsf{nil} +\!\!+ l = l$, defined pointwise to be the identity path, and α^{list} and ρ^{list}, defined by list-elimination as follows for every l, l_1, l_2, $l_3 : \mathsf{list}(X)$ and $x : X$:

$$\begin{aligned}
\alpha^{\mathsf{list}}_{l_1,l_2,l_3} &: (l_1 +\!\!+ l_2) +\!\!+ l_3 = l_1 +\!\!+ (l_2 +\!\!+ l_3) & \qquad \rho^{\mathsf{list}}_{l} &: l +\!\!+ \mathsf{nil} = l \\
\alpha^{\mathsf{list}}_{\mathsf{nil},l_2,l_3} &:\equiv \mathsf{refl} & \rho^{\mathsf{list}}_{\mathsf{nil}} &:\equiv \mathsf{refl} \\
\alpha^{\mathsf{list}}_{x::\, l_1,l_2,l_3} &:\equiv \mathsf{ap}_{(x::\, -)}(\alpha^{\mathsf{list}}_{l_1,l_2,l_3}) & \rho^{\mathsf{list}}_{x::\, l} &:\equiv \mathsf{ap}_{(x::\, -)}(\rho^{\mathsf{list}}_{l}).
\end{aligned}$$

▶ **Definition 13.** We define a function $K : \mathsf{FMG}(X) \to \mathsf{list}(X)$ as

$$K :\equiv \mathsf{rec}_{\mathsf{FMG}}(\mathsf{nil}, (x \mapsto x :: \mathsf{nil}), +\!\!+, \alpha^{\mathsf{list}}, \lambda^{\mathsf{list}}, \rho^{\mathsf{list}}, \pentagon^{\mathsf{list}}, \triangledown^{\mathsf{list}});$$

that is, K is defined to map the monoidal structure of $\mathsf{FMG}(X)$ into that of $\mathsf{list}(X)$.

▶ **Definition 14.** We define a monoidal functor $J : \mathsf{MonGpd}(\mathsf{list}(X), \mathsf{FMG}(X))$ as follows. The underlying function $J : \mathsf{list}(X) \to \mathsf{FMG}(X)$ is defined by list-elimination, declaring $J(\mathsf{nil}) :\equiv e$ and, recursively, $J(x :: l) :\equiv \iota(x) \bullet J(l)$, for every $x : X$ and $l : \mathsf{list}(X)$. A path $J_0 : e = J(\mathsf{nil})$ is then given by refl, while, given l_1, $l_2 : \mathsf{list}(X)$, a path $J_2(l_1, l_2) : J(l_1) \bullet J(l_2) = J(l_1 +\!\!+ l_2)$ can be produced by induction on l_1:

$$\begin{aligned}
J_2(\mathsf{nil}, l_2) \quad &: J(\mathsf{nil}) \bullet J(l_2) \equiv e \bullet J(l_2) = J(l_2) \equiv J(\mathsf{nil} +\!\!+ l_2) && \text{by } \lambda_{J(l_2)}, \\
J_2(x :: l_1, l_2) &: J(x :: l) \bullet J(l_2) \equiv (\iota(x) \bullet J(l_1)) \bullet J(l_2) \\
&\qquad = \iota(x) \bullet (J(l_1) \bullet J(l_2)) && \text{by } \alpha_{\iota(x),J(l_1),J(l_2)} \\
&\qquad = \iota(x) \bullet J(l_1 +\!\!+ l_2) && \text{by } \mathsf{refl} \bullet J_2(l_1, l_2) \\
&\qquad \equiv J(x :: (l_1 +\!\!+ l_2)) \equiv J((x :: l_1) +\!\!+ l_2).
\end{aligned}$$

The construction of the families of 2-paths J_α, J_λ and J_ρ are shown in Figures 9 to 11. Since $+\!\!+$ satisfies left unitality judgmentally (Remark 12), we can easily find a 2-path $J_\lambda(l)$ for $l : \mathsf{list}(X)$, as the sought diagram (Figure 10) is trivial. Moreover, we have for every path $p : l_1 = l_2$ in $\mathsf{list}(X)$ and $x : X$, a 2-path

$$\mathsf{ap}_J(\mathsf{ap}_{(x::\, -)}(p)) = \mathsf{refl}_{\iota(x)} \bullet \mathsf{ap}_J(p) \tag{4.1}$$

by induction on p. This, together with the coherence diagrams in Figure 1 and naturality of associativity and unitality, allows us to define the families of 2-paths $J_\alpha(l_1, l_2, l_3)$ and $J_\rho(l)$ by list-elimination (on the first argument for J_α), as shown in Figure 9 and Figure 11, respectively.

▶ **Lemma 15.** *There is a homotopy $\eta : \mathsf{id}_{\mathsf{FMG}(X)} \sim J \circ K$, for K and J given in Definitions 13 and 14.*

Proof. A term $\eta : \Pi(w : \mathsf{FMG}(X)).w = J(K(w))$ is given by $\eta :\equiv \mathsf{ind}_{\mathsf{FMG}}(e', \iota', \bullet', \ldots)$, where:

- $e' :\equiv \mathsf{refl} : e = J(K(e))$, since the right-hand side of the equality computes to e;
- $\iota'(x) :\equiv \rho^{-1}_{\iota(x)} : \iota(x) = J(K(\iota(x)))$ for every $x : X$, the right-hand side of the equality computing to $\iota(x) \bullet e$;
- $a' \bullet' b' : a \bullet b = J(K(a \bullet b))$ for a, $b : \mathsf{FMG}(X)$, $a' : a = J(K(a))$ and $b' : b = J(K(b))$, is defined as the concatenation:

$$\begin{aligned} a \bullet b &= J(K(a)) \bullet J(K(b)) && \text{by } a' \bullet b' \\ &= J(K(a) +\!\!+ K(b)) \equiv J(K(a \bullet b)) && \text{by } J_2(K(a), K(b)), \end{aligned}$$

  so that $\eta(a \bullet b) \equiv (\eta(a) \bullet \eta(b)) \cdot J_2(K(a), K(b))$;
- α', λ' and ρ' correspond to the diagrams illustrated in Figures 12 to 14 and are proved using the monoidal components J_α, J_λ and J_ρ of J;
- the families $\pentagon'$ and $\triangledown'$ correspond to 3-paths in a 1-type, so they are obtained trivially. ◀

The proof of coherence is then immediately achieved.

Proof of Theorem 7. By Proposition 11, FMG is a free functor. By Lemma 15, $\mathsf{FMG}(X)$ is a retract of $\mathsf{list}(X)$; since X is a 0-type, so is $\mathsf{list}(X)$ and hence $\mathsf{FMG}(X)$, as shown in Figure 15. ◀

5 Discussion

The choice of employing the higher groupoid structure of types to represent categories, where paths take the role of arrows, leads to an important observation concerning the expressivity of the theory: the concept of strictness of a monoidal category cannot be formulated in the framework. Indeed, strictness is not homotopy invariant, and hence it cannot be detected by the theory. For this reason, we use instead a formulation of coherence (Statement 1) concerning a property that is preserved under equivalence of types. Its proof is, then, the presentation of a technique of *normalisation* of monoidal expressions over a set: a term $a : \mathsf{FMG}(X)$ has, as normal form, the term $J(K(a))$, for J and K given in Definitions 13 and 14.

The use of identity types and HITs in HoTT to describe monoidal structures largely simplifies the definition of the free monoidal groupoid, compared to [4, 5]; there, the free monoidal category over a set X is defined via:

- an inductive set of objects, whose terms correspond to the ones produced by the 0-constructors e, ι and $\bullet$ in $\mathsf{FMG}(X)$;
- inductive families of arrows, with identity arrows, $(- \circ -)$, $(- \bullet -)$, α, α^{-1}, λ, λ^{-1}, ρ and ρ^{-1} as constructors, on which induction is performed when proving coherence; in our implementation, the groupoid structure of identity types takes care of most of the inductive cases, whereas the cases for α, λ and ρ remain present in the application of the elimination principle of $\mathsf{FMG}(X)$;

- inductive families of equalities between arrows, with a sizeable number of constructors, including: reflexivity, symmetry and transitivity of equality; associativity and unitality of composition; substitution for composition and for the monoidal product; naturality of associativity and unitality of the monoidal product; the interchange law between composition and the monoidal product; composition of associativity and unitality arrows with their inverse; and the coherence diagrams. All but the latter is made redundant in our implementation, as path induction proves everything but the defining diagrams $\pentagon$ and $\triangledown$ of the monoidal structure.

As a result, the proof of coherence presented in this paper is considerably shorter than the one provided in [4, 5]; this is reflected in the rather compact computer verification (Section 6), which is included in this paper as supplementary material.

Another feature of our approach is the easiness in formulating and proving freeness of FMG (Proposition 11), as the proof is entirely contained in the elimination principle of the HIT itself. Although this proof was not present in [4, 5], our presentation shares the same strategy in defining *ad hoc* a free structure with inductively generated "objects" and "arrows" and normalising the ensuing monoidal expressions to a type of lists (via the functor K in Definition 13), where coherence is immediate, rather than directly show freeness for lists. This turns out to be the easiest option: indeed, for $X : \mathcal{U}$ and $M : \mathsf{MonGpd}$, the reasonable way of producing functions $\phi_{X,M}$ and $\psi_{X,M}$ as in Definition 6 for $F :\equiv \mathsf{list}$ would be by defining:

- $\phi_{X,M}(G) :\equiv G \circ (- :: \mathsf{nil}) : X \to M$ for every $G : \mathsf{MonGpd}(\mathsf{list}(X), M)$;
- the underlying function of $\psi_{X,M}(g) : \mathsf{MonGpd}(\mathsf{list}(X), M)$, for $g : X \to M$, by list-elimination, in a way analogous to J in Definition 14, i.e., declaring $\psi_{X,M}(g)(\mathsf{nil}) :\equiv e^M$ and $\psi_{X,M}(g)(x :: l) :\equiv g(x) \bullet^M \psi_{X,M}(g)(l)$ for every $x : X$ and $l : \mathsf{list}(X)$.

The term $(\psi_{X,M}(g))_2$, necessary to define a monoidal functor, and similarly $(\psi_{X,M}(g))_\alpha$, $(\psi_{X,M}(g))_\lambda$ and $(\psi_{X,M}(g))_\rho$, are not trivial and require to be also proved by list-elimination. Carrying convoluted proof terms in a proof of freeness of list is cumbersome and in stark contrast to the benefits given by the computation rules of $\mathsf{ind}_{\mathsf{FMG}}$, described in Remark 9 and used in Proposition 11. Moreover, in exhibiting a homotopy $\theta_{X,M} : \phi_{X,M} \circ \psi_{X,M} \sim \mathsf{id}_{X \to M}$, we would be forced to use function extensionality: indeed, for $g : X \to M$, we would have to provide a term

$$\theta_{X,M}(g) : \phi_{X,M}(\psi_{X,M}(g)) \equiv (x \mapsto g(x) \bullet^M e^M) = g,$$

whereas $\rho^M \circ g : \Pi(x : X).g(x) \bullet^M e^M = g(x)$ only ensures pointwise equality between the two functions. We believe it is worth observing that, contrary to what above, our proof of freeness of FMG does never employ function extensionality.

Alternative definitions for a monoidal functor $K : \mathsf{MonGpd}(\mathsf{FMG}(X), \mathsf{list}(X))$ are possible. Notably, the normalising functor described in [4, 5] could be replicated in our set-up as a monoidal functor having, as underlying function, the composition of

$$\mathsf{ev}_{\mathsf{nil}} : (\mathsf{list}(X) \to \mathsf{list}(X)) \to \mathsf{list}(X), \qquad \mathsf{ev}_{\mathsf{nil}}(f) :\equiv f(\mathsf{nil})$$

after a function $N : \mathsf{FMG}(X) \to (\mathsf{list}(X) \to \mathsf{list}(X))$, defined by the elimination principle of FMG, declaring $N(e) :\equiv \mathsf{id}_{\mathsf{list}(X)}$, $N(\iota(x)) :\equiv (x :: -)$ for every $x : X$, and $N(a \bullet b) :\equiv N(b) \circ N(a)$ for every $a, b : \mathsf{FMG}(X)$. The requirements relative to the higher constructors in the inductive definition of N are trivially fulfilled (since e.g. the composition of functions is judgmentally associative) and the composition $\mathsf{ev}_{\mathsf{nil}} \circ N$ can be shown to be a monoidal functor. While this definition is useful to normalise associativity and unitality, it does not

extend to more complex monoidal structures, for example when symmetry is present (see Section 7), as composition of functions is not symmetric; in that case, a functor such as the one presented in Definition 13 is more suitable to the task of normalisation.

6 Formalisation in Coq

The formalisation, included to this paper as supplementary material, has been written using the HoTT library [11, 2] for the Coq proof assistant[3] and is structured as follows:

- the categorical framework on monoidal groupoids is included in `monoidalgroupoid.v`; for the definitions of monoidal groupoids, monoidal functors, monoidal natural isomorphisms and free functors, we use classes instead of Σ-types for easy access to their components and for type coercions;
- in `FMG.v` we provide the definition, for a type X, of the HIT $\mathsf{FMG}(X)$ as a private inductive type, specifying the 0-constructors (on which Coq can perform pattern matching) and, separately, the higher constructors as axioms. The elimination principle $\mathsf{ind}_{\mathsf{FMG}}$ and its computation rules also need to be given as axioms, while the corresponding recursor $\mathsf{rec}_{\mathsf{FMG}}$ can be derived from $\mathsf{ind}_{\mathsf{FMG}}$; a specific (derived) version of the elimination principle for families of paths in a groupoid is also formalised;
- the proof of Proposition 11 appears in `FMG_free.v`;
- the proof that $\mathsf{list}(X)$ is a 0-type whenever X is a 0-type is included in `lists.v`; this is achieved by means of an "encode-decode" proof [16, 21] and is roughly based on the fact that, for every x_1, $x_2 : X$ and l_1, $l_2 : \mathsf{list}(X)$, there is an equivalence of identity types $(x_1 :: l_1 = x_2 :: l_2) \simeq (x_1 = x_2) \times (l_1 = l_2)$. The same file contains the definition of the monoidal structure of $\mathsf{list}(X)$;
- Theorem 7 is formalised in `FMG_coherence.v`;
- a library of lemmata about path algebra is included in a separate file (`hott_lemmas.v`).

7 Conclusions

The work presented in this paper serves as an example to highlight some of the features of HoTT that can be employed in the context of formalisation of mathematics. Identity types and higher inductive types can be used to give a concrete description of objects satisfying certain universal properties. This result opens the way to the formalisation of similar coherence theorems; for example, definitions analogue to those given in Sections 2 and 3 can be used to describe (free) *symmetric* monoidal groupoids, and symmetric monoidal expressions can be normalised to a HIT of lists with added paths and 2-paths encoding the action of symmetric groups, corresponding to transpositions of adjacent elements in a list and the relations they satisfy. The relevant formalisation in Coq is also present as part of the supplementary material included.

References

1 Sten Agerholm, Ilya Beylin, and Peter Dybjer. A comparison of HOL and ALF formalizations of a categorical coherence theorem. In Gerhard Goos, Juris Hartmanis, Jan van Leeuwen, Joakim von Wright, Jim Grundy, and John Harrison, editors, *Theorem Proving in Higher Order Logics*, pages 17–32, Berlin, Heidelberg, 1996. Springer Berlin Heidelberg. `doi:10.1007/BFb0105394`.

[3] Commit `68774877142adbd435ea5013c5f201f3ec6ff66a` (February 14th, 2020) of the HoTT library; Coq 8.10+alpha.

2 Andrej Bauer, Jason Gross, Peter LeFanu Lumsdaine, Mike Shulman, Matthieu Sozeau, and Bas Spitters. The HoTT library: A formalization of homotopy type theory in Coq, 2016. arXiv:1610.04591.

3 Benno van den Berg and Richard Garner. Types are weak ω-groupoids. *Proceedings of the London Mathematical Society*, 102(2):370–394, 2011. doi:10.1112/plms/pdq026.

4 Ilya Beylin. *An ALF Proof of Mac Lane's Coherence Theorem*. Licensiate thesis (revision: 5.26), Department of Computing Science, Chalmers / Göteborg University, 1997.

5 Ilya Beylin and Peter Dybjer. Extracting a proof of coherence for monoidal categories from a proof of normalization for monoids. In Stefano Berardi and Mario Coppo, editors, *Types for Proofs and Programs*, pages 47–61, Berlin, Heidelberg, 1996. Springer Berlin Heidelberg. doi:10.1007/3-540-61780-9_61.

6 Guillaume Brunerie. On the homotopy groups of spheres in homotopy type theory, June 2016. arXiv:1606.05916.

7 Guillaume Brunerie, Kuen-Bang Hou (Favonia), Evan Cavallo, Tim Baumann, Eric Finster, Jesper Cockx, Christian Sattler, Chris Jeris, Michael Shulman, et al. Homotopy Type Theory in Agda. URL: https://github.com/HoTT/HoTT-Agda.

8 Floris van Doorn. On the Formalization of Higher Inductive Types and Synthetic Homotopy Theory, 2018. arXiv:1808.10690.

9 Floris van Doorn, Jakob von Raumer, and Ulrik Buchholtz. Homotopy Type Theory in Lean. *Lecture Notes in Computer Science*, pages 479–495, 2017. doi:10.1007/978-3-319-66107-0_30.

10 Mike Gordon. Introduction to the HOL System. In *1991 International Workshop on the HOL Theorem Proving System and Its Applications*, pages 2–3, August 1991. doi:10.1109/HOL.1991.596265.

11 The HoTT library. URL: https://github.com/HoTT/HoTT.

12 INRIA – The Coq Proof Assistant. URL: https://coq.inria.fr/.

13 Gregory Maxwell Kelly. On MacLane's conditions for coherence of natural associativities, commutativities, etc. *Journal of Algebra*, 1(4):397–402, 1964. doi:10.1016/0021-8693(64)90018-3.

14 Tom Leinster. *Higher operads, higher categories*, volume 298 of *London Mathematical Society lecture note series*. Cambridge University Press, Cambridge, 2004. doi:10.1017/CBO9780511525896.

15 Daniel R. Licata and Guillaume Brunerie. A Cubical Approach to Synthetic Homotopy Theory. In *2015 30th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 92–103, 2015. doi:10.1109/LICS.2015.19.

16 Daniel R. Licata and Michael Shulman. Calculating the fundamental group of the circle in homotopy type theory. In *Proceedings of the 2013 28th Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '13, pages 223–232, Washington, DC, USA, 2013. IEEE Computer Society. doi:10.1109/LICS.2013.28.

17 Peter LeFanu Lumsdaine. Weak ω-categories from intensional type theory. *Logical Methods in Computer Science*, Volume 6, Issue 3, September 2010. doi:10.2168/LMCS-6(3:24)2010.

18 Peter LeFanu Lumsdaine and Michael Shulman. Semantics of higher inductive types. *Mathematical Proceedings of the Cambridge Philosophical Society*, pages 1–50, June 2019. doi:10.1017/s030500411900015x.

19 Saunders Mac Lane. *Categories for the working mathematician*, volume 5 of *Graduate texts in mathematics*. Springer, New York, 2nd ed. edition, 1998.

20 Lena Magnusson. *The Implementation of ALF – a Proof Editor based on Martin-Löf's Monomorphic Type Theory with Explicit Substitution*. PhD Thesis, Göteborg University and Chalmers University of Technology, 1995.

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

A Figures in Proofs

Derived Coherence Diagrams in Figure 1

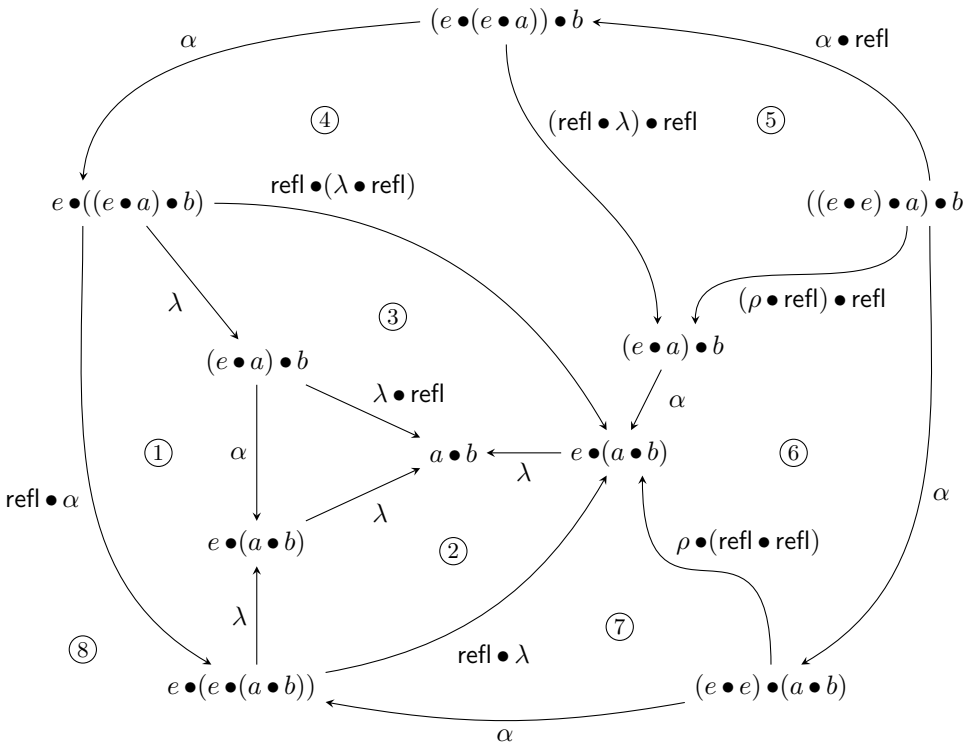


Figure 5 Derivation of the coherence diagram in Figure 1c, here appearing as the unmarked 2-path; the attribute $-^M$ has been omitted for clarity from e, $\bullet$, α, λ and ρ. The 2-paths (1), (2) and (3) are instances of naturality of λ^M; (4) and (6) are instances of naturality of α^M; (5) and (7) are instances of $\triangledown^M \bullet^M \mathsf{refl}$ and $\triangledown^M$ respectively; the outer pentagon (8) is an instance of $\pentagon^M$. The diagram in Figure 1d is obtained similarly.

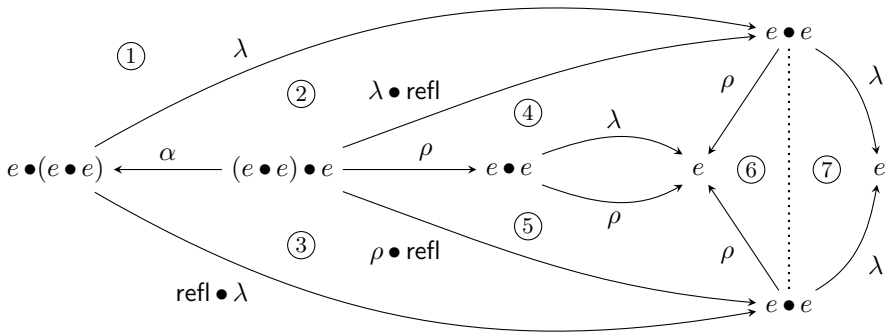


Figure 6 Derivation of the coherence diagram in Figure 1e, here appearing as the unmarked 2-path; again, $-^M$ has been omitted for clarity. The outer square (1) is an instance of naturality of λ^M; the 2-path (2) is the derived coherence diagram in Figure 1c; (3) is an instance of $\triangledown^M$; (4) and (5) are instances of naturality of ρ^M; (6) and (7) are trivial.

Figures in the Proof of Proposition 11 (Freeness of FMG)

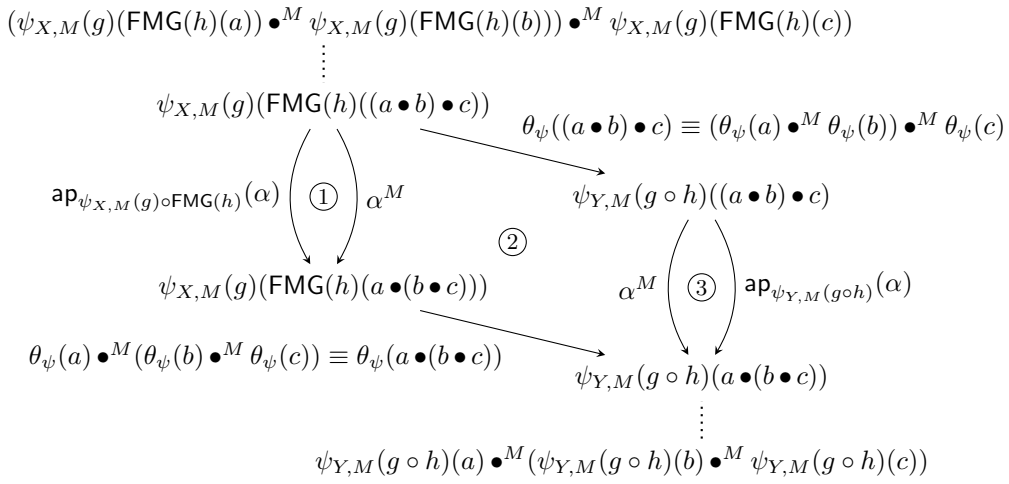


Figure 7 The underlying homotopy θ_ψ in the proof of naturality of $\psi_{X,M}$ in X is achieved via the elimination rules of $\psi_{X,M}$ and FMG; these require certain 2-paths in M to be provided, corresponding to the 1-path constructors of $\mathsf{FMG}(Y)$. This figure shows the 2-path α' for associativity. The 2-paths (1) and (3) are given by the computation rules of $\psi_{X,M}$ and $\mathsf{FMG}(h)$; (2) is filled by naturality of α^M. The 2-paths λ' and ρ' in M corresponding to the constructors for unitality are proved similarly.

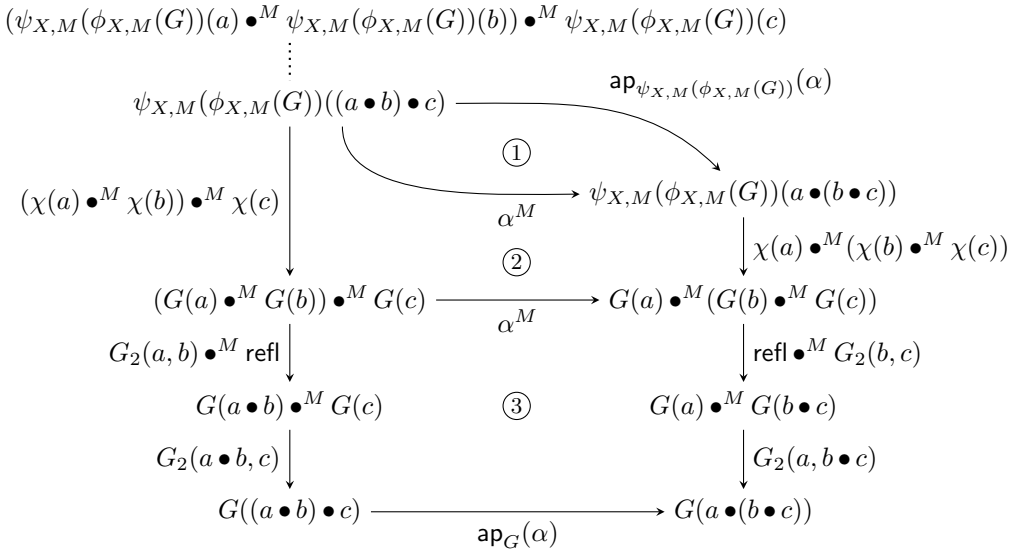
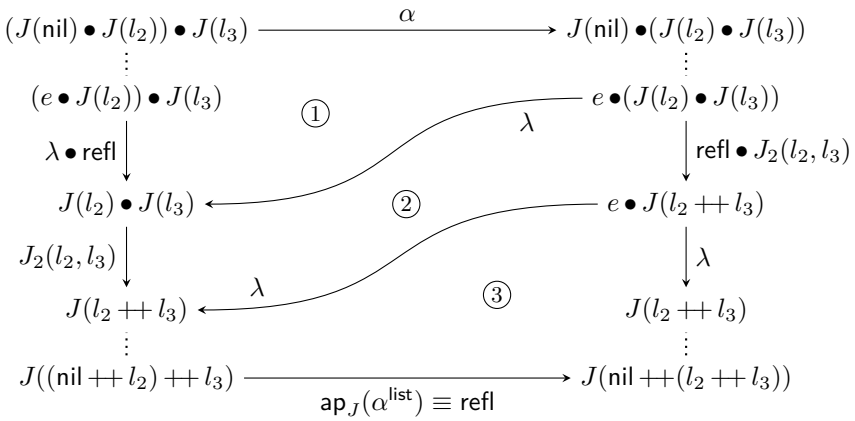
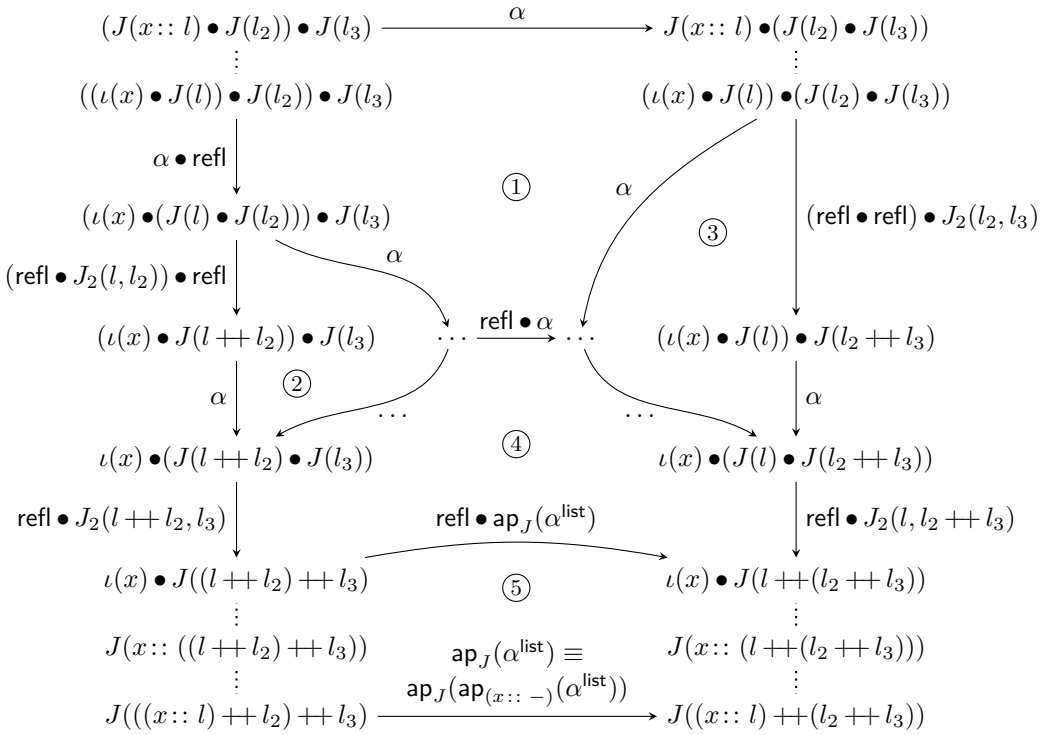


Figure 8 The 2-path in M providing α' in the definition of χ using the elimination principle of $\mathsf{FMG}(X)$. The 2-path (1) is given by a computation rule of $\psi_{X,M}$; (2) is an instance of naturality of α^M; (3) is an instance of G_α. The vertical paths correspond to the ones given by $\bullet'$, after application of the interchange law between path concatenation and the action of $\bullet^M$ on paths. The 2-paths for λ' and ρ' are obtained similarly.

Figures relevant to the Proof of Theorem 7 (Coherence for Monoidal Groupoids)



(a) The 2-path $J_\alpha(\mathsf{nil}, l_2, l_3)$ in the inductive definition of J_α. The 2-path (1) is an instance of the additional coherence diagram in Figure 1c; (2) is an instance of naturality of λ; (3) is trivial.



(b) $J_\alpha(x :: l, l_2, l_3)$ in the inductive definition of J_α. The 2-path (1) is an instance of ⬠; (2) and (3) are instances of naturality of α, where the omitted paths are $\mathsf{refl} \bullet (J_2(l, l_2) \bullet \mathsf{refl})$ and $\mathsf{refl} \bullet (\mathsf{refl} \bullet J_2(l_2, l_3))$ respectively; (4) is given by $\mathsf{refl}_{\iota(x)} \bullet J_\alpha(l, l_2, l_3)$; (5) is an instance of (4.1).

Figure 9 Construction of the 2-path $J_\alpha(l_1, l_2, l_3)$ by induction on l_1, after unfolding the definition of J_2 (appearing on the vertical sides) and some path algebra.

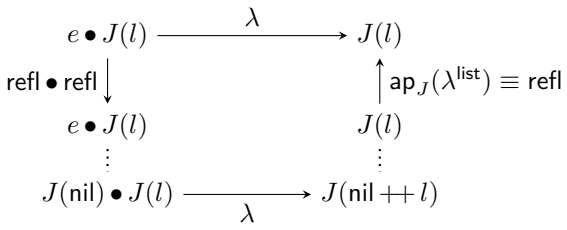
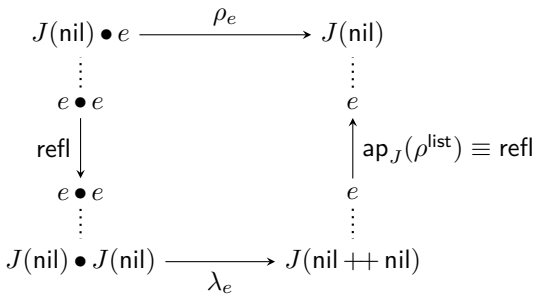
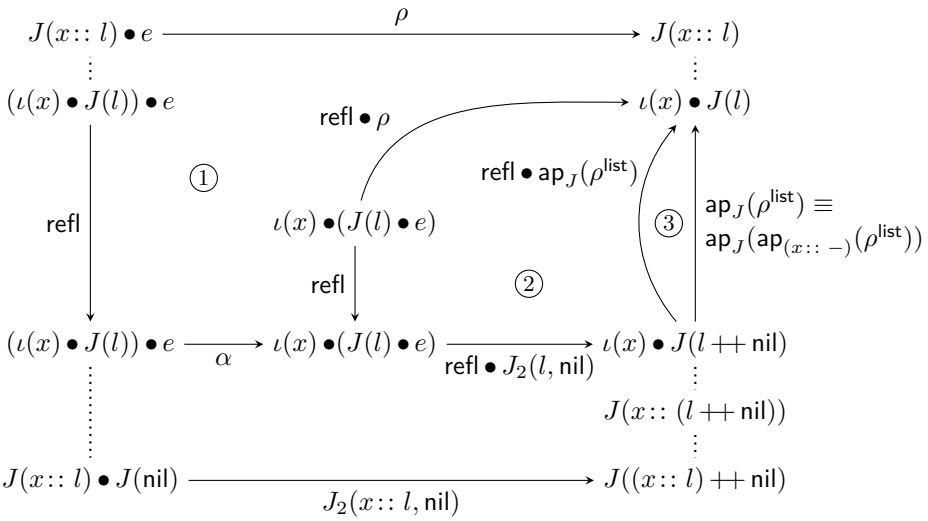


Figure 10 The 2-path $J_\lambda(l)$, after unfolding the definitions of J_0 (left side) and J_2 (bottom side), is trivial.



(a) The 2-path $J_\rho(\mathsf{nil})$ in the inductive definition of J_ρ can be obtained by the additional coherence diagram in Figure 1e.



(b) The 2-path $J_\rho(x :: l)$ in the inductive definition of J_ρ. The 2-path (1) is an instance of the additional coherence diagram in Figure 1d; (2) is given recursively by $\mathsf{refl}_{\iota(x)} \bullet J_\rho(l)$; (3) is an instance of (4.1).

Figure 11 The construction of the 2-path $J_\rho(l)$ by induction on l, after unfolding the definitions of J_0 (left side) and J_2 (bottom side) and some path algebra.

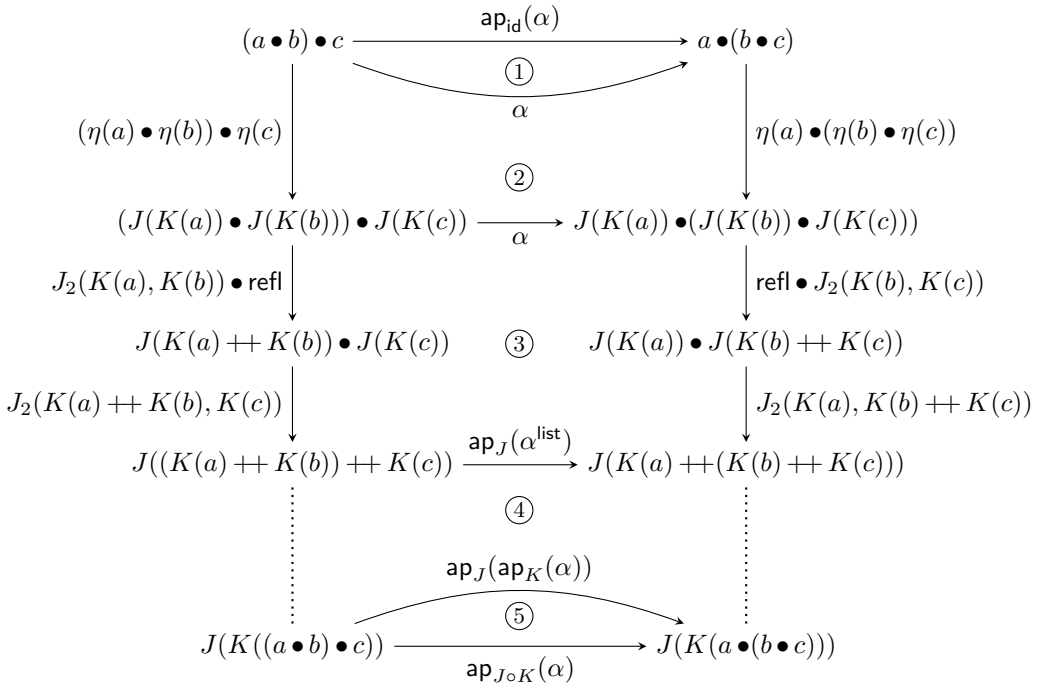


Figure 12 The 2-path α' in the definition of η. The vertical path on the left is equal to $\eta((a \bullet b) \bullet c)$ using the interchange law between path concatenation and the action of $\bullet$ on paths; similarly, the vertical path on the right is equal to $\eta(a \bullet (b \bullet c))$. The 2-paths (1) and (5) are given by path algebra; (2) is an instance of naturality of α; (3) is an instance of J_α; (4) is given by a computation rule of K.

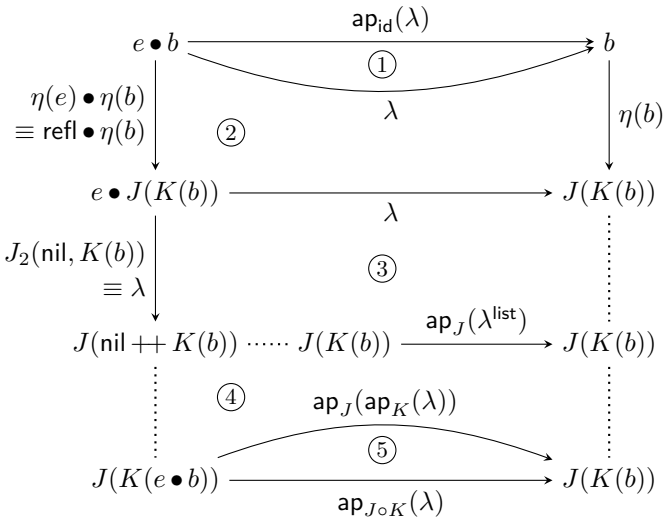


Figure 13 The 2-path λ' in the definition of η; the vertical path on the left is by definition $\eta(e \bullet b)$. The 2-paths (1) and (5) are given by path algebra; (2) is an instance of naturality of λ; (3) is trivial, as $\lambda^{\mathsf{list}}_{K(b)} \equiv \mathsf{refl}$; (4) is given by a computation rule of K.

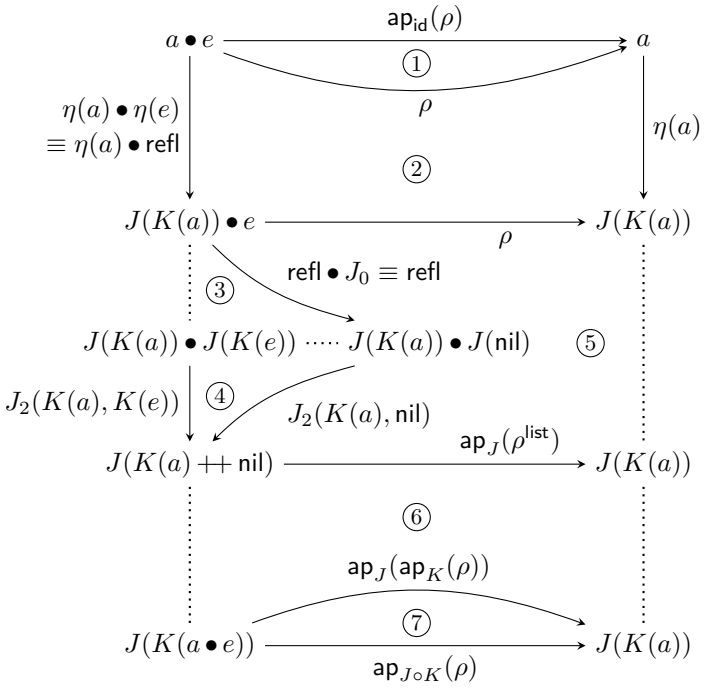


Figure 14 The 2-path ρ' in the definition of η; the vertical path on the left is by definition $\eta(a \bullet e)$. The 2-paths (1) and (7) are given by path algebra; (2) is an instance of naturality of ρ; (3) and (4) are trivial; (5) is an instance of J_ρ; (6) is given by a computation rule of K.

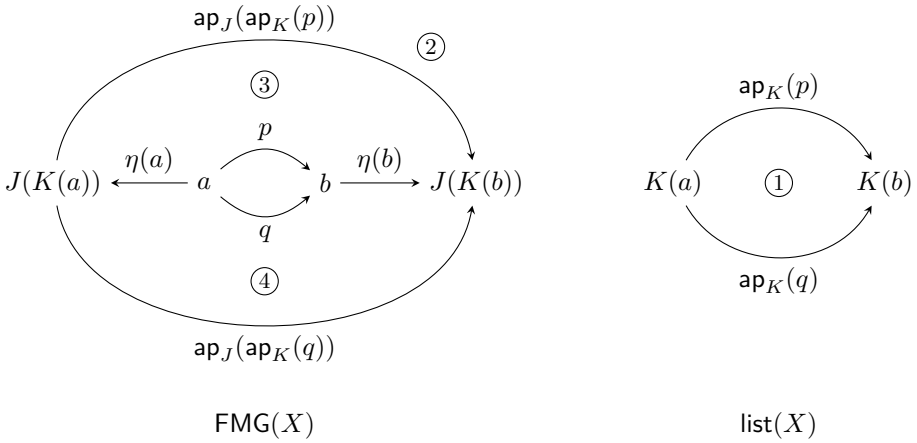


Figure 15 Proof of coherence. For any two paths $p, q : a = b$ in $\mathsf{FMG}(X)$, there is a 2-path (1) in $\mathsf{list}(X)$, since this is a 0-type. By functoriality, we obtain a 2-path in $\mathsf{FMG}(X)$ corresponding to the outer diagram (2). The 2-paths (3) and (4) are obtained by path induction (on p and q respectively), yielding a term in $p = q$ corresponding to the unmarked 2-path.