MODEL STRUCTURES FROM HIGHER INDUCTIVE TYPES

PETER LEFANU LUMSDAINE

Abstract. We show that for any dependent type theory with Martin-Löf identity types and *mapping cylinders* (defined as certain higher-dimensional inductive types), the category of contexts carries a *pre-model-structure*, i.e. a model structure minus the completeness conditions. The (trivial cofibrations,fibrations) are the Gambino-Garner weak factorisation system of [GG08], while the weak equivalences are equivalences in the sense of Voevodsky [Voe].

It follows that any categorical model of this type theory carries a pre-model-structure, and so, if it is additionally complete and co-complete, is a model category.

Contents

This note isn't intended for formal publication in its current form: I'd like to wait until some more background is available (eg [LS11a], [LS11b]) to give better context and motivation. However, at least from a purely formal point of view, this proof stands on its own; and I've talked a bit about the result publicly, so it seems right to make the proof available in some form.

Thanks as ever to Michael Warren, Mike Shulman, Chris Kapulkin, Steve Awodey, and Nicola Gambino for helpful feedback and suggestions!

1. Type-theoretic background

We will need a few basic definitions and deductions in the type theory. We work, for this section, in the setting of Martin-Löf Type Theory with at least identity types; we do not assume Π-, Σ-, or any base types.

To emphasise the intended homotopy-theoretic interpretation, we will write the identity types as $\mathsf{Paths}_A(x, x')$; and the syntax we use for their eliminator will be:

$$\frac{\begin{array}{c}\Gamma,\, x, x' {:} A,\, u {:} \mathsf{Paths}_A(x,x'),\, \vec{w} {:} \Delta(x,x',u) \;\vdash\; C(x,x',u,\vec{w})\ \mathsf{type} \\ \Gamma,\, x {:} A,\, \vec{w} {:} \Delta(x,x,\mathsf{refl}(x)) \;\vdash\; d(x,\vec{w}) : C(x,x,\mathsf{refl}(x),\vec{w})\end{array}}{\begin{array}{l}\Gamma,\, x, x' {:} A,\, u {:} \mathsf{Paths}_A(x,x'),\, \vec{w} {:} \Delta(x,x',u) \;\vdash \\ \quad \mathsf{path\text{-}elim}_{x,x',u,\vec{w}.\, C(x,x',u,\vec{w})}(x, \vec{w}.d(x,\vec{w});\, x, x', u, \vec{w}) : C(x,x',u,\vec{w})\end{array}}\ \mathsf{Id}\text{-elim}$$

Date: 7 December, 2011.

Where unambiguous, we will omit the subscripts on $\mathsf{path\text{-}elim}$.

From these identity types, we can derive *identity contexts* over arbitrary contexts ([Str93], [GG08]), with formation rule:

$$\frac{\Gamma \ \vdash \ \Delta \ \mathsf{cxt}}{\Gamma,\, \vec{x}, \vec{x}' {:} \Delta \ \vdash \ \mathsf{Paths}_{\Delta}(\vec{x}, \vec{x}') \ \mathsf{cxt}}$$

and satisfying intro, elim, and comp rules analogous to those for identity types. We will write the "eliminator" for these also as $\mathsf{path\text{-}elim}$.

We will need two derived rules involving paths: transport between fibers of a fibration along a path in the base; and the action of any map on paths, taking paths in its domain to paths in its codomain.

Lemma 1. *One may derive terms* $\mathsf{transport}$, $\mathsf{dep\text{-}cong}$[1], *witnessing the following rules:*

$$\frac{\Gamma,\, x{:}A \ \vdash \ B(x) \ \mathsf{type}}{\Gamma,\, x, x'{:}A,\, u{:}\mathsf{Paths}(x, x'),\, y{:}B(x) \ \vdash \ \mathsf{transport}_{x.\, B(x)}(u, y) : B(x')}$$

$$\frac{\Gamma,\, x{:}A \ \vdash \ f(x) : B(x)}{\Gamma,\, x, x'{:}A,\, u{:}\mathsf{Paths}(x, x') \ \vdash \ \mathsf{dep\text{-}cong}(x.f(x);\, u) : \mathsf{Paths}(\,\mathsf{transport}(u, f(x))\,,\, f(x')\,)}$$

and with computation rules

$$\mathsf{transport}\, y\, (\mathsf{refl}(x)) = y, \qquad \mathsf{dep\text{-}cong}\, (\mathsf{refl}(x)) = \mathsf{refl}(f(x)).$$

As with $\mathsf{path\text{-}elim}$, in practice we will elide the subscripts on these as often as possible, and use these terms also to mean their analogues for identity contexts.

Proof. The terms involved are each defined by a single instance of $\mathsf{path\text{-}elim}$:

$$\mathsf{transport}_{x.\, B(x)}(u, y) := \mathsf{path\text{-}elim}_{x, x', u, y.\, B(y)}(x, y.\, y;\, x, x', u, y)$$

$$\mathsf{dep\text{-}cong}(x.\, f(x);\, u) :=$$

$$\mathsf{path\text{-}elim}_{x, x', u.\, \mathsf{Paths}(\mathsf{transport}(u, f(x)), f(x'))}(x.\, \mathsf{refl}(f(x));\, x, x', u) \qquad \square$$

Given a universe or some other form of polymorphism, $\mathsf{transport}$ and $\mathsf{dep\text{-}cong}$ can be defined as polymorphic terms of the theory.

We will also make use of some derived judgements: contractibility of a dependent type, and equivalences between contexts.

Definition 2. When $\Gamma \ \vdash \ A \ \mathsf{type}$, write

$$\Gamma \ \vdash \ (a, \alpha) \ \mathsf{contract} \ A$$

to abbreviate the pair of judgements

$$x{:}\Gamma \ \vdash \ a(x) : A \qquad\qquad \vec{x}{:}\Gamma,\, y, y'{:}A \ \vdash \ \alpha(\vec{x}, y, y') : \mathsf{Paths}(y, y').$$

Similarly, we will write

$$\Gamma \ \vdash \ \mathsf{isContr} \ A$$

if there exist terms (a, α) such that $\Gamma \ \vdash \ (a, \alpha) \ \mathsf{contract} \ A$.

In a theory with more constructors, one can represent these internally by the type $\mathsf{isContr}(A) := A \times \Pi_{x, x' : A} \mathsf{Paths}(x, x')$.

[1]The name $\mathsf{dep\text{-}cong}$ stands for 'dependent congruence'; in type theory, the action of functions on paths is traditionally considered as the fact that *equality is a congruence for all functions.*

Definition 3. Next, if $\Gamma \vdash \Delta, \Theta\ \mathsf{cxt}$, we will write

$$\Gamma \vdash (\vec{f}, \vec{g}, \vec{g}', \vec{\eta}, \vec{\epsilon})\ \Delta\ \mathsf{equiv}\ \Theta$$

to abbreviate the five judgements

$$\Gamma, \vec{x}{:}\Delta \vdash \vec{f}(\vec{x}) : \Theta$$

$$\Gamma, \vec{y}{:}\Theta \vdash \vec{g}(\vec{y}) : \Delta \qquad \Gamma, \vec{x}{:}\Delta \vdash \vec{\eta}(\vec{x}) : \mathsf{Paths}_{\Delta}(\vec{g}(\vec{f}(\vec{x}), \vec{x}))$$

$$\Gamma, \vec{y}{:}\Theta \vdash \vec{g}'(\vec{y}) : \Delta \qquad \Gamma, \vec{y}{:}\Theta \vdash \vec{\epsilon}(\vec{x}) : \mathsf{Paths}_{\Theta}(\vec{f}(\vec{g}'(\vec{y})), \vec{y}).$$

In other words, $\vec{g}$ is a left up-to-homotopy inverse of $\vec{f}$, and $\vec{g}'$ a right. We will write

$$\Gamma \vdash \mathsf{isEquiv}\ \vec{f}$$

when there exist $(\vec{g}, \vec{g}', \vec{\eta}, \vec{\epsilon})$ as above. Again, with enough constructors in the theory, $\mathsf{isEquiv}\ f$ may be represented internally as a type.

(This definition is equivalent to Voevodsky's definition of equivalences in terms of contractible homotopy fibres; compare Lemma 17 below.)

2. Type-theoretic mapping cylinders

Mapping cylinders are familiar from topology; here we axiomatise them type-theoretically, as a special case of *higher inductive types*. See [Lum11], [LS11a] for a more general introduction to these types.

Definition 4. A type theory T with identity types has *mapping cylinders (for types)* if it has constructors Cyl, $\mathsf{in\text{-}base}$, $\mathsf{in\text{-}top}$, $\mathsf{in\text{-}cyl}$, satisfying the eight following rules. The pattern of these is familiar from ordinary inductive types: there is one formation rule; there are three introduction rules, for the three constructors; there is one elimination rule, with three "inductive premises"; and there are three computation rules. As usual, we suppress throughout an ambient context Γ, assumed to appear in all judgements involved.

The formation rule is straightforward:

$$\frac{x{:}X \vdash f(x){:}Y}{y{:}Y \vdash \mathsf{Cyl}_{x.f(x)}(y)\ \mathsf{type}}\ \mathsf{Cyl\text{-}FORM}$$

By abuse of notation, we will often write just Cyl_f for $\mathsf{Cyl}_{x.f(x)}$. In all the following rules, we will assume the hypothesis of $\mathsf{Cyl\text{-}form}$ among the premises.

The introduction rules are also straightforward:

$$\frac{}{y{:}Y \vdash \mathsf{in\text{-}base}(y){:}\mathsf{Cyl}_f(y)} \qquad \frac{}{x{:}X \vdash \mathsf{in\text{-}top}(x){:}\mathsf{Cyl}_f(f(x))}$$

$$\frac{}{x{:}X \vdash \mathsf{in\text{-}cyl}(x){:}\mathsf{Paths}_{\mathsf{Cyl}_f(f(x))}(\mathsf{in\text{-}top}(x), \mathsf{in\text{-}base}(f(x)))}$$

The third "inductive premise" of the eliminator is slightly subtle in form, since $\mathsf{in\text{-}cyl}$ is a path-constructor:

$$\frac{\begin{array}{c} y{:}Y, z{:}\mathsf{Cyl}_f(y) \vdash C(y,z)\ \mathsf{type} \\ y{:}Y \vdash d_{\mathsf{base}}(y){:}C(y, \mathsf{in\text{-}base}(y)) \\ x{:}X \vdash d_{\mathsf{top}}(x){:}C(f(x), \mathsf{in\text{-}top}(x)) \\ x{:}X \vdash d_{\mathsf{cyl}}(x) : \mathsf{Paths}_{C(f(x),\mathsf{in\text{-}base}(f(x)))}(\,(\mathsf{transport}\ \mathsf{in\text{-}cyl}(x)\ d_{\mathsf{top}}(x))\,,\, d_{\mathsf{base}}(f(x))\,) \end{array}}{y{:}Y, z{:}\mathsf{Cyl}_f(y) \vdash \mathsf{cyl\text{-}elim}_C(d_{\mathsf{base}}, d_{\mathsf{top}}, d_{\mathsf{cyl}}; y, z){:}C(y,z)}$$

The three computation rules assert, under the same premises as the elimination rule, that:

$$y{:}Y \ \vdash\ \mathsf{cyl\text{-}elim}_C(d_{\mathsf{base}}, d_{\mathsf{top}}, d_{\mathsf{cyl}}; y, \mathsf{in\text{-}base}(y)) = d_{\mathsf{base}}(y) : C(y, \mathsf{in\text{-}base}(y))$$

$$x{:}X \ \vdash\ \mathsf{cyl\text{-}elim}_C(d_{\mathsf{base}}, d_{\mathsf{top}}, d_{\mathsf{cyl}}; f(x), \mathsf{in\text{-}top}(x)) = d_{\mathsf{top}}(x) : C(f(x), \mathsf{in\text{-}top}(y))$$

$$x{:}X \ \vdash\ \mathsf{cyl\text{-}comp}(x)$$
$$: \mathsf{Paths}(\ \mathsf{dep\text{-}cong}(\ z.\, \mathsf{cyl\text{-}elim}_C(d_{\mathsf{base}}, d_{\mathsf{top}}, d_{\mathsf{cyl}}; f(x), z)\ ,\ \mathsf{in\text{-}cyl}(x)\ )\ ,\ d_{\mathsf{cyl}}(x)\ )$$

It may seem odd that this last "computation" rule in fact only posits a path, not a definitional equality. This is currently a point of uncertainty for higher inductive types in general. However, since the action of morphisms on paths (as given by dep-cong) is not generally well-behaved up to definitional equality, it seems perhaps incongruously strict to request one here. More pragmatically, the analogue of Proposition 7 seems difficult with the third computation rule definitional.

In the first two computation rules, however, or at least in the one for in-top, we really do require definitional equality, since we use cyl-elim to construct on-the-nose factorisations of maps in Lemma 15.

In the hypotheses needed for the main theorem, we will be able to weaken these rules in one respect:

Definition 5. T has *non-dependent mapping cylinders* if the above rules hold over the empty context. (That is, we remove the implicit convention that all rules should be read as including an ambient context of parameters.)

While this is in many ways a less natural definition, it suffices for the main theorem of the present notes; and it has the advantage of being (possibly) simpler to model, avoiding some of the potential coherence problems of the full dependent version.

In another direction, we will need to strengthen these rules:

Definition 6. T has *mapping cylinders for contexts* if for any dependent context morphism

$$\Gamma \ \vdash \Delta, \Xi\ \mathsf{cxt} \qquad \Gamma, \vec{x}{:}\Delta \ \vdash\ \vec{f}(\vec{x}) : \Xi$$

we are given a context

$$\Gamma,\, \vec{y}{:}\Xi \ \vdash\ \mathsf{Cyl}_{\vec{x}.\vec{f}(\vec{x})}(\vec{y})\ \mathsf{cxt}$$

with constructors and eliminators analogous to those of Definition 4.

However, this is not too much of a strengthening:

Proposition 7. *If* T *has* Id*-types,* Σ*-types, and all (resp. non-dependent) mapping cylinders for types, then it has such mapping cylinders also for contexts.*

Proof. Given any context map $\vec{f}$, we can use Σ-types to represent it as a map of types f; then the mapping cylinder of f provides one for $\vec{f}$. □

3. A pre-model-structure from mapping cylinders

Definition 8. A *weak factorisation system* on a category $\mathbf{C}$ is a pair $(\mathcal{L}, \mathcal{R})$ of classes of maps of $\mathbf{C}$, such that $\mathcal{L} = {}^{\boxslash}\mathcal{R}$, $\mathcal{R} = \mathcal{L}^{\boxslash}$, and every map in $\mathbf{C}$ admits some factorisation as a map in $\mathcal{L}$ followed by a map in $\mathcal{R}$.

Note that in any wfs, both classes of maps are automatically closed under retracts.

Definition 9. A *pre-model-structure* on a category $\mathbf{C}$ consists of three classes $(\mathcal{C}, \mathcal{F}, \mathcal{W})$ of maps of $\mathbf{C}$, such that $\mathcal{W}$ satisfies 3-for-2, and $(\mathcal{C}, \mathcal{F} \cap \mathcal{W})$ and $(\mathcal{C} \cap \mathcal{W}, \mathcal{F})$ are weak factorisation systems on $\mathbf{C}$.

Compare [Hov99, 1.1.3–4]. A pre-model-structure is just a model structure, minus the completeness conditions on $\mathbf{C}$ and functoriality of the factorisations.

Theorem 10. *Suppose* T *is a type theory with* Id- *and* Σ*-types and non-dependent mapping cylinders; or, more generally, with* Id*-types, and non-dependent mapping cylinders for contexts.*

Then $\mathcal{C}\ell(\mathsf{T})$ *carries a pre-model-structure, with classes of maps as defined in Definitions 11–13 below.*

By Proposition 7, it is sufficient to consider the case with mapping cylinders for contexts; so from here on, fix such T, and work within it.

Definition 11 (Gambino-Garner [GG08])**.** Take the *generating fibrations* $\mathcal{F}_0$ to be the dependent projections $\pi_{\Gamma,A}$, for all types $\Gamma \vdash A\ \mathsf{type}$. Set $\mathcal{TC} := {}^{\boxslash}(\mathcal{F}_0)$ (the *trivial cofibrations*), and then $\mathcal{F} := \mathcal{TC}^{\boxslash}$ (*fibrations*).

Definition 12. Take the *generating trivial fibrations* $\mathcal{TF}_0$ to be the dependent projections $\pi_{\Gamma,A}$ for types such that $\Gamma \vdash \mathsf{isContr}\ A$. Then as above, define $\mathcal{C} := {}^{\boxslash}(\mathcal{TF}_0)$ (*cofibrations*), $\mathcal{TF} := \mathcal{C}^{\boxslash}$ (*trivial fibrations*).

Definition 13. Take $\mathcal{W}$, the class of *weak equivalences*, to be the class of context morphisms f such that $\vdash \mathsf{isEquiv}\ f$.

This definition of the cofibrations and trivial cofibrations is rather indirect: can we describe them, or at least some large class of them, more explicitly? We will address this in Section 4 below; but the main intuition to have is that constructors of inductive types will be cofibrations, and that these will be trivial when they are the only constructor of the type.

The following two lemmas show special cases of that:

Lemma 14 (Gambino-Garner [GG08, 4.2.2])**.** *Every map factors as a map which is in both* $\mathcal{TC}$ *and* $\mathcal{W}$*, followed by a map in* $\mathcal{F}_0$*. Call the class of maps appearing in these factorisations* $\mathcal{TC}_0$*.*

Proof. Given a map $f : \Theta \to \Delta$, factor it through its "homotopy fiber" context:

$$\vec{y}{:}\Delta,\ \vec{x}{:}\Theta,\ \vec{u}{:}\mathsf{Paths}_\Delta(f(\vec{x}), y). \qquad \square$$

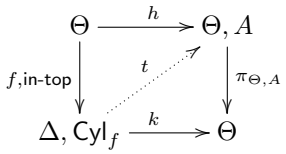
Lemma 15. *Every map factors as a* $\mathcal{C}$*, followed by a* $\mathcal{TF}_0$*.*

Proof. Given $f{:}\,\Theta \to \Delta$, factor it through its mapping cylinder:

$$\Theta \xrightarrow{f,\mathsf{in\text{-}top}} \Delta, \mathsf{Cyl}_f \xrightarrow{\pi_{\mathsf{Cyl}_f}} \Delta$$

We need to show that $(f, \mathsf{in\text{-}top})$ is a cofibration, and that $\vec{y}{:}\Delta \vdash \mathsf{isContr}\ \mathsf{Cyl}_f(\vec{y})$.

The first of these is a typical case of showing an inductive constructor to be in $\mathcal{C}$. We need to show that we can lift $(f, \mathsf{in\text{-}top})$ against any generating trivial fibration:



This amounts to finding a term

$$\vec{y}{:}\Delta,\ z{:}\mathsf{Cyl}_f\ \vdash\ t(\vec{y},z) : A(k(\vec{y})).$$

such that $t(\vec{x},\mathsf{in\text{-}top}(\vec{x})) = h(\vec{x})$. But we may obtain this just by an application of $\mathsf{cyl\text{-}elim}$, using h for the premise corresponding to $\mathsf{in\text{-}top}$, and supplying the other premises using the contraction on A. Specifically, if a, α is a contraction for A, then

$$t(\vec{y},z) := \mathsf{cyl\text{-}elim}_{y,z.A(k(\vec{y},z))}(\vec{x}.h(\vec{x}),\ \vec{y}.a(k(\vec{y},\mathsf{in\text{-}base}(\vec{y}))),\ \vec{x}.\alpha(\_,\ \_))$$

gives a filler as desired.

Secondly, to show that $\vec{y}{:}\Delta\ \vdash\ \mathsf{isContr}\ \mathsf{Cyl}_f(\vec{y})$, take $\mathsf{in\text{-}top}(\vec{y})$ as the canonical inhabitant. Then to construct the desired contraction $\vec{y} : \Delta,\ z, z' : \mathsf{Cyl}_f(\vec{y})\ \vdash\ \alpha(\vec{y},z,z') : \mathsf{Paths}(z,z')$, it is enough (by composition of paths) to construct a term

$$\vec{y}{:}\Delta,\ z{:}\mathsf{Cyl}_f(\vec{y})\ \vdash\ \beta(\vec{y},z) : \mathsf{Paths}(z,\mathsf{in\text{-}top}(\vec{y})).$$

This again we obtain by $\mathsf{cyl\text{-}elim}$. Two of the inductive premises are easy: in the case $z = \mathsf{in\text{-}base}(\vec{x})$, we use $\mathsf{in\text{-}cyl}(\vec{x})$, and in the case $z = \mathsf{in\text{-}top}(\vec{y})$, we use $\mathsf{refl}(\mathsf{in\text{-}top}(\vec{y}))$. It remains to provide the third inductive premise, for $\mathsf{in\text{-}cyl}$:

$$\vec{x}{:}\Theta\ \vdash\ d_{\mathsf{cyl}}(\vec{x}) \\ : \mathsf{Paths}(\,\mathsf{transport}_{z.\mathsf{Paths}(z,\mathsf{in\text{-}top}(f(\vec{x})))}(\mathsf{in\text{-}cyl}(\vec{x}),\mathsf{in\text{-}cyl}(\vec{x})),\ \mathsf{refl}(\mathsf{in\text{-}top}(f(\vec{x})))\,).$$

But this is an instance of a general fact about how paths transport along themselves, provable for any type X

$$x_0, x_1 : X,\ u : \mathsf{Paths}(x_0,x_1)\ \vdash \\ \mathsf{transport\text{-}lemma}(u) : \mathsf{Paths}(\mathsf{transport}_{x.\mathsf{Paths}(x_0,x_1)}(u,u),\mathsf{refl}(x_1))$$

by $\mathsf{path\text{-}elim}$: when u is $\mathsf{refl}(x)$, the goal reduces to $\mathsf{Paths}(\mathsf{refl}(x),\mathsf{refl}(x))$, so in this case we may use $\mathsf{refl}(\mathsf{refl}(x))$. □

Lemma 16. *$\mathcal{W}$ satisfies 3-for-2, and is closed under retracts.*

Proof. This follows purely formally, from the fact that homotopy of context morphisms is an equivalence relation, preserved by composition on either side. For instance, suppose

$$\begin{array}{ccc} \Gamma_0 & \underset{r_0}{\overset{s_0}{\rightleftarrows}} & \Delta_0 \\ \downarrow{\scriptstyle u} & & \downarrow{\scriptstyle w} \\ \Gamma_1 & \underset{r_1}{\overset{s_1}{\rightleftarrows}} & \Delta_1 \end{array}$$

is a retraction, with $w \in \mathcal{W}$.

Now if $i\colon \Delta_1 \to \Gamma_1$ is a left homotopy inverse for w, then $r_1 \cdot i \cdot s_0$ is a left homotopy inverse for u, since $r_1 \cdot i \cdot s_0 \cdot u = r_1 \cdot i \cdot w \cdot s_0 \simeq r_1 \cdot s_0 = 1$.

Similarly, if $j\colon \Delta_1 \to \Gamma_1$ is a right homotopy inverse for w, then $r_1 \cdot j \cdot s_0$ is a right homotopy inverse for u: $u \cdot r_1 \cdot i \cdot s_0 = r_0 \cdot w \cdot i \cdot s_0 \simeq r_0 \cdot s_0 = 1$.

The three parts of 3-for-2 all follow similarly. □

We need one last logical lemma:

Lemma 17. *For any type $\Gamma\ \vdash\ A\ \mathsf{type}$, the projection $\pi_{\Gamma,A}$ is an equivalence if and only if $\Gamma\ \vdash\ \mathsf{isContr}\ A$. In other words, $\mathcal{F}_0 \cap \mathcal{W} = \mathcal{TF}_0$.*

Proof. ($\Leftarrow$) Suppose $\Gamma \ \vdash \ (a, \alpha)$ $\mathsf{contract}$ A. Then $\vec{x} \mapsto (\vec{x}, a(\vec{x}))$ gives an on-the-nose right inverse for $\pi_{\Gamma,A}$, and $\vec{x}, y \mapsto \vec{x}, \alpha(y, a(\vec{x}))$ witnesses that it is moreover a left homotopy inverse.

($\Rightarrow$) Suppose $\pi_{\Gamma,A}$ is a weak equivalence, with right homotopy inverse $\vec{g}$. Then $\vec{g}$ is of the form $\vec{g}_\Gamma, a'$, where

$$\vec{x}{:}\Gamma \ \vdash \ \vec{\epsilon}(\vec{x}) : \mathsf{Paths}_\Gamma(\vec{g}_\Gamma(\vec{x}), \vec{x}),$$

$$\vec{x}{:}\Gamma \ \vdash \ a'(x) : A(\vec{g}_\Gamma(\vec{x})).$$

Then $a(\vec{x}) := \mathsf{transport}(\vec{\epsilon}(\vec{x}), a'(\vec{x}))$ gives us the desired inhabitant of $A(\vec{x})$, for any $x{:}\Gamma$.

To show $\Gamma \ \vdash \ \mathsf{isContr}A$, it remains to find some α such that

$$\vec{x}{:}\Gamma,\, y, y'{:}A \ \vdash \ \alpha(\vec{x}, y, y') : \mathsf{Paths}(y, y').$$

For this, it is sufficient to find β such that

$$\vec{x}{:}\Gamma,\, y{:}A \ \vdash \ \beta(\vec{x}, y) : \mathsf{Paths}(a(\vec{x}), y).$$

Since $\pi_{\Gamma,A}$ is a weak equivalence, any right homotopy inverse is also a left homotopy inverse. In particular, (π_A, a) is: we can derive $\vec{\eta}$ so that

$$\vec{x}{:}\Gamma,\ y{:}A \ \vdash \ \vec{\eta}(\vec{x}, y) : \mathsf{Paths}_{\Gamma,A}(\,(\vec{x}, a(\vec{x}))\,,\,(\vec{x}, y)\,).$$

This looks tantalisingly like the β we seek, but is not quite right: unwinding the definition of path contexts, $\vec{\eta}$ consists of $(\vec{\eta}^\Gamma, \eta^A)$, where

$$\vec{x}{:}\Gamma,\ y{:}A \ \vdash \ \vec{\eta}^\Gamma(\vec{x}, y) : \mathsf{Paths}_\Gamma(\vec{x}, \vec{x}),$$

$$\vec{x}{:}\Gamma,\ y{:}A \ \vdash \ \eta^A(\vec{x}, y) : \mathsf{Paths}_{A(\vec{x})}(\,\mathsf{transport}(\vec{\eta}^\Gamma(\vec{x}, y), a(\vec{x}))\,,\, y\,),$$

so the source of $\eta^A(\vec{x}, y)$ isn't exactly $a(\vec{x})$, as we'd like it to be. However, we can correct the difference, using $\mathsf{dep\text{-}cong}$:

$$\vec{x}{:}\Gamma,\ y{:}A \ \vdash \ \mathsf{dep\text{-}cong}(\vec{x}.a(\vec{x}); \vec{\eta}^\Gamma(\vec{x}, y)) : \mathsf{Paths}_{A(\vec{x})}(\,\mathsf{transport}(\vec{\eta}^\Gamma(\vec{x}, y), a(\vec{x}))\,,\, a(\vec{x})\,),$$

so composing the inverse of this with $\eta^A(\vec{x}, y)$, we have the desired term β. □

Let's recap what we now know:

- $(\mathcal{TC}, \mathcal{F})$ and $(\mathcal{C}, \mathcal{TF})$ are closed orthogonal pairs of classes, each admitting factorisations, with the factors lying in subclasses $\mathcal{TC}_0$, $\mathcal{F}_0$ etc.;
- $\mathcal{W}$ contains all identities, satisfies 3-for-2, and is closed under retracts;
- $\mathcal{TF}_0 = \mathcal{F}_0 \cap \mathcal{W}$;
- $\mathcal{TC}_0 \subseteq \mathcal{W}$.

To show that we have a pre-model-structure, it remains to show that $\mathcal{TF} = \mathcal{F} \cap \mathcal{W}$ and $\mathcal{TC} = \mathcal{C} \cap \mathcal{W}$. But this follows from the facts listed above, by a succession of purely formal factorisation/retract arguments.

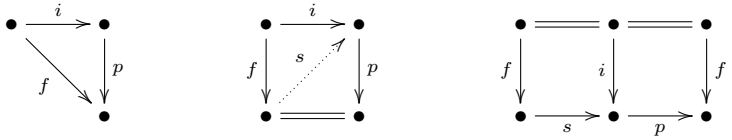
Lemma 18. *$\mathcal{TF} = \mathcal{F} \cap \mathcal{W}$, and $\mathcal{TC} = \mathcal{C} \cap \mathcal{W}$.*

Proof. As every first-year logic student knows, this reduces to showing six inclusions, which we tackle individually.

By definition, $\mathcal{TF}_0 \subseteq \mathcal{F}_0$; so by monotonicity, $\mathcal{TC} \subseteq \mathcal{C}$, and thence $\mathcal{TF} \subseteq \mathcal{F}$.

$\mathcal{TC} \subseteq \mathcal{W}$: given $f \in \mathcal{TC}$, factor it as $f = p \cdot i$, where $i \in \mathcal{TC}_0$ (and hence $i \in \mathcal{W}$), and $p \in \mathcal{F}_0$.

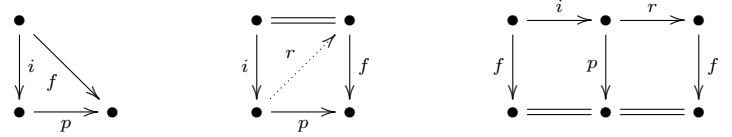
Then f lifts against p: there is some s with $i = s \cdot f$, $p \cdot s = 1$. But this expresses f as a retract of $i \in \mathcal{W}$, so f is itself in $\mathcal{W}$.



$\mathcal{W} \cap \mathcal{C} \subseteq \mathcal{TC}$: given any $f \in \mathcal{W} \cap \mathcal{C}$, again factor $f = p \cdot i$, where $i \in \mathcal{TC} \subseteq \mathcal{W}$, $p \in \mathcal{F}_0$. By 3-for-2, $p \in \mathcal{W}$, so in fact $p \in \mathcal{W} \cap \mathcal{F}_0$ and hence $p \in \mathcal{TF}_0$. So again f lifts against p, expressing f as a retract $i \in \mathcal{TC}$; so $f \in \mathcal{TC}$.

$\mathcal{TF} \subseteq \mathcal{W}$: exactly dual to the proof above that $\mathcal{TC} \subseteq \mathcal{W}$. Every $f \in \mathcal{TF}$ is a retract of some $p \in \mathcal{TF}_0 \subseteq \mathcal{W}$, so is itself in $\mathcal{W}$.

$\mathcal{W} \cap \mathcal{F} \subseteq \mathcal{TF}$: suppose $f \in \mathcal{W} \cap \mathcal{F}$. Similarly to before, factoring $f = p \cdot i$, where $p \in \mathcal{F}_0$ and $i \in \mathcal{TC}_0 \subseteq \mathcal{W}$, we can lift i against f to express f as a retract of p. But by 3-for-2, $p \in \mathcal{W}$, so $p \in \mathcal{W} \cap \mathcal{F}_0 \subseteq \mathcal{TF}_0$, and hence $f \in \mathcal{TF}$.



This completes the proof of the theorem. □

4. Characterisations of fibrations and cofibrations

Our definition of the factorisation systems explicitly provided natural classes of fibrations and trivial fibrations, large enough to include most of the examples one meets in practice. Can we give something similar for the cofibrations?

It turns out that in general yes, we can: constructor inclusions of inductive types. However, a precise statement and proof of this depend on which formulation(s) of inductive types one wants to consider; so for the sake of brevity, we will give just an informal statement and a sketch of how the proof typically goes.

Not-Quite-Lemma 19. *Suppose $A(\vec{y})$ is a (possibly higher) inductive family, varying over the context $\vec{y} : \Delta$ of indices, defined over an ambient context of parameters Γ; and suppose c is a 0-cell constructor of A, with type*

$$\Gamma,\, \vec{x} : \Theta \vdash \mathsf{c}(\vec{x}) : A(g(\vec{x}))$$

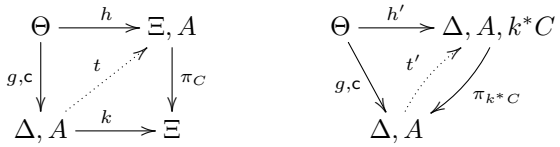
(for some $g : \Theta \to \Delta$). Then the context morphism $(1_\Gamma, g, \mathsf{c}) : \Gamma,\, \Theta \to \Gamma,\, \Delta,\, A$ is in $\mathcal{C}$; and indeed for any $f : \Gamma' \to \Gamma$, so is $(f, f^*g, f^*\mathsf{c})$.

If c is the only constructor of A, then it is moreover in $\mathcal{TC}$.

Finally, these morphisms form generating subclasses of $\mathcal{C}$, $\mathcal{TC}$.

Not-quite-proof. We essentially just repeat the proof of Lemma 15. For simplicity, we suppress the paramaters Γ throughout.

First, note that to fill squares it suffices (by pullback) to fill triangles:



Now note, for the single-constructor case, that the triangle gives exactly the premises of the elim rule: a predicate defined over the family A, and inhabitants of this predicate for all canonical elements $\mathsf{c}(\vec{x})$. So the elim and comp rule together give us an appropriate map, and show that it's a filler.

For the case where A has other constructors, the triangle doesn't directly provide all the inductive premises of the elim rule—we need to know that the predicate holds for all canonical elements, not just those of the form $\mathsf{c}(\vec{x})$. However, we have the additional assumption that the the predicate is contractible; and this contraction allows us to provide witnesses for the predicate over any other constructors (including higher ones); so now as before, the elim and comp rules give us the required filler.

Finally, since the factorisations provided by Lemmata 14 and 15 are all of this form, it follows that these are generating subclasses of $\mathcal{C}$, $\mathcal{TC}$. □

A very nice characterisation of trivial cofibrations is given in [GG08, 5.1.1]: they are exactly the context morphisms for which certain rules can be derived—essentially, the rules of a one-constructor inductive type. A similarly type-theoretic characterisation of cofibrations would be very nice, but seems elusive!

In lieu of that, one can at least give a precise characterisation by general retract arguments, albeit a less purely type-theoretic one:

Lemma 20. *A context morphism is a cofibration exactly if it is a retract of some constructor inclusion of an inductive type.*

Proof. The "if" direction follows from closure of $\mathcal{C}$ under retracts. For the "only if", note that any cofibration is a retract of the cofibration occurring in its $(\mathcal{C}, \mathcal{TF})$ factorisation; but by Lemma 15, this may always be taken to be the constructor inclusion $\mathsf{in\text{-}top}$ of a mapping cylinder. □

5. Model structures from mapping cylinders

The most obvious instance of the above construction is on snytactic type theories, giving pre-model-structures on syntactic categories of type theories. Such categories will typically not be (even finitely) complete or co-complete, nor will the factorisations in these cases generally be functorial; so will not be full model categories.

However, it also of course applies to semantic type theories those coming from categorical models of this type theory. (Precisely, to the internal language of any category with attributes admitting appropriate type-constructors; see [Car78] et seq., surveyed in [Lum10, 1.2]).

These examples are very often complete and co-complete; and so if moreover the factorisations are functorial (or may be modified to be so), then the pre-model structure constructed here underlies an actual model structure.

For instance, in the Hofmann-Streicher groupoid model ([HS98]), it is not hard to construct non-dependent mapping cylinders by hand, and show that these are functorial; so the results of this note give an alternate construction of the standard model structure on **Gpd**.

These results may thus be seen as partially converse to the forthcoming [LS11b], where we show that a large class of higher inductive types, including mapping cylinders, can be modelled in any locally presentable model category, modulo (as ever) some technical issues of coherence.

References

[Car78] John Cartmell, *Generalised algebraic theories and contextual categories*, Ph.D. thesis, Oxford, 1978.

[GG08] Nicola Gambino and Richard Garner, *The identity type weak factorisation system*, Theoret. Comput. Sci. **409** (2008), no. 1, 94–109, arXiv:0803.4349, doi:10.1016/j.tcs.2008.08.030. MR 2469279

[Hov99] Mark Hovey, *Model categories*, Mathematical Surveys and Monographs, vol. 63, American Mathematical Society, Providence, RI, 1999. MR 1650134 (99h:55031)

[HS98] Martin Hofmann and Thomas Streicher, *The groupoid interpretation of type theory*, Twenty-five years of constructive type theory (Venice, 1995), Oxford Logic Guides, vol. 36, Oxford Univ. Press, New York, 1998, pp. 83–111.

[LS11a] Peter LeFanu Lumsdaine and Michael Shulman, *Higher inductive types 0: Introduction*, in preparation, 2011.

[LS11b] ———, *Higher inductive types 1: models in locally presentable model categories*, in preparation, 2011.

[Lum10] Peter LeFanu Lumsdaine, *Higher categories from type theories*, Ph.D. thesis, Carnegie Mellon University, 2010.

[Lum11] ———, *Higher inductive types: a tour of the menagerie*, blog post, April 2011, http://homotopytypetheory.org/2011/04/24/higher-inductive-types-a-tour-of-the-menagerie.

[Str93] Thomas Streicher, *Investigations Into Intensional Type Theory*, Habilitationsschrift, Ludwig-Maximilians-Universität München, November 1993.

[Voe] Vladimir Voevodsky, *The equivalence axiom and univalent models of type theory*, Lecture delivered at Carnegie Mellon University, February 2010, and unpublished notes, http://www.math.ias.edu/~vladimir/Site3/Univalent_Foundations_files/CMU_talk.pdf.