Formalization of Double Groupoids and Crossed Modules

Jakob von Raumer | July 5, 2016

University of Nottingham

The Fundamental Group

The *fundamental group* of a Type $A : \mathcal{U}$ at a point $x : A$ is defined by

$$\pi_1(A, x) :\equiv \|x =_A x\|_0 .$$

It is trivial for each $x : A$ if and only if A is a set.

$\rightsquigarrow$ Internalize algebraic structures and use formalized algebra to gain knowledge about higher types!

The Fundamental Group

The *fundamental group* of a Type $A : \mathcal{U}$ at a point $x : A$ is defined by

$$\pi_1(A, x) :\equiv \|x =_A x\|_0 \,.$$

It is trivial for each $x : A$ if and only if A is a set.
$\rightsquigarrow$ Internalize algebraic structures and use formalized algebra to gain knowledge about higher types!

Higher Homotopy Groups

How to generalize fundamental groups?

- The n-th homotopy group of A is defined by

$$\pi_n(A, x) = \left\|\mathbb{S}^n \to A\right\|_0$$

- Allow for multiple base points!

1. Double Groupoids and Crossed Modules

2. Formalization in Lean

Fundamental Groupoids

- Second generalization of fundamental groups:
- For a top. space X and $C \subseteq X$ we have

  $$\pi_1(X, C) := \{\text{paths with endpoints in } C\}/_{\text{homotopies fixing endpoints}}$$

- Algebraic structure of this construction: Groupoid on C.

Fundamental Groupoids

- Second generalization of fundamental groups:
- For a top. space X and $C \subseteq X$ we have

$$\pi_1(X, C) := \{\text{paths with endpoints in } C\}/_{\text{homotopies fixing endpoints}}$$

- Algebraic structure of this construction: Groupoid on C.

Fundamental Groupoids

- Second generalization of fundamental groups:
- For a top. space X and $C \subseteq X$ we have

$$\pi_1(X, C) := \{\text{paths with endpoints in } C\}/_{\text{homotopies fixing endpoints}}$$

- Algebraic structure of this construction: Groupoid on C.

Categories and Groupoids in HoTT

A category consists of

- A carrier type $A : \mathcal{U}$,
- morphism sets hom : $\prod_{a,b:A}$ Set, ...
- a composition operation $\circ : \prod_{a,b,c:A} \mathrm{hom}(b,c) \to \mathrm{hom}(a,b) \to \mathrm{hom}(a,c)$,
- and identity morphisms id : $\prod_{a:A} \mathrm{hom}(a,a)$,

as well as the properties

- idLeft : $\prod_{a,b:A} \prod_{f:\mathrm{hom}(a,b)} \mathrm{id}_b \circ f = f$,
- idRight : $\ldots$,
- assoc : $\prod_{a,b,c,d:A} \prod_{f,g,h:\ldots} h \circ (g \circ f) = (h \circ g) \circ f$.

Categories and Groupoids in HoTT

A category consists of

- A carrier type $A : \mathcal{U}$,
- morphism sets hom : $\prod_{a,b:A}$ Set, ...
- a composition operation $\circ : \prod_{a,b,c:A} \mathrm{hom}(b,c) \to \mathrm{hom}(a,b) \to \mathrm{hom}(a,c)$,
- and identity morphisms id : $\prod_{a:A} \mathrm{hom}(a,a)$,

as well as the properties

- idLeft : $\prod_{a,b:A} \prod_{f:\mathrm{hom}(a,b)} \mathrm{id}_b \circ f = f$,
- idRight : $\ldots$,
- assoc : $\prod_{a,b,c,d:A} \prod_{f,g,h:\ldots} h \circ (g \circ f) = (h \circ g) \circ f$.

Categories and Groupoids in HoTT

A category consists of

- A carrier type $A : \mathcal{U}$,
- morphism sets hom : $\prod_{a,b:A}$ Set, ...
- a composition operation
  $\circ : \prod_{a,b,c:A} \mathrm{hom}(b, c) \to \mathrm{hom}(a, b) \to \mathrm{hom}(a, c)$,
- and identity morphisms id : $\prod_{a:A} \mathrm{hom}(a, a)$,

as well as the properties

- idLeft : $\prod_{a,b:A} \prod_{f:\mathrm{hom}(a,b)} \mathrm{id}_b \circ f = f$,
- idRight : $\ldots$,
- assoc : $\prod_{a,b,c,d:A} \prod_{f,g,h:\ldots} h \circ (g \circ f) = (h \circ g) \circ f$.

Categories and Groupoids in HoTT

A category consists of

- A carrier type $A : \mathcal{U}$,
- morphism sets $\mathrm{hom} : \prod_{a,b:A} \mathrm{Set}$, ...
- a composition operation $\circ : \prod_{a,b,c:A} \mathrm{hom}(b,c) \to \mathrm{hom}(a,b) \to \mathrm{hom}(a,c)$,
- and identity morphisms $\mathrm{id} : \prod_{a:A} \mathrm{hom}(a,a)$,

as well as the properties

- $\mathrm{idLeft} : \prod_{a,b:A} \prod_{f:\mathrm{hom}(a,b)} \mathrm{id}_b \circ f = f$,
- $\mathrm{idRight} : \ldots$,
- $\mathrm{assoc} : \prod_{a,b,c,d:A} \prod_{f,g,h:\ldots} h \circ (g \circ f) = (h \circ g) \circ f$.

Categories and Groupoids in HoTT

A category consists of

- A carrier type $A : \mathcal{U}$,
- morphism sets $\mathrm{hom} : \prod_{a,b:A} \mathrm{Set}$, ...
- a composition operation $\circ : \prod_{a,b,c:A} \mathrm{hom}(b,c) \to \mathrm{hom}(a,b) \to \mathrm{hom}(a,c)$,
- and identity morphisms $\mathrm{id} : \prod_{a:A} \mathrm{hom}(a,a)$,

as well as the properties

- $\mathrm{idLeft} : \prod_{a,b:A} \prod_{f:\mathrm{hom}(a,b)} \mathrm{id}_b \circ f = f$,
- $\mathrm{idRight} : \ldots$,
- $\mathrm{assoc} : \prod_{a,b,c,d:A} \prod_{f,g,h:\ldots} h \circ (g \circ f) = (h \circ g) \circ f$.

Categories and Groupoids in HoTT

A category consists of

- A carrier type $A : \mathcal{U}$,
- morphism sets $\mathrm{hom} : \prod_{a,b:A} \mathrm{Set}$, ...
- a composition operation
  $\circ : \prod_{a,b,c:A} \mathrm{hom}(b,c) \to \mathrm{hom}(a,b) \to \mathrm{hom}(a,c)$,
- and identity morphisms $\mathrm{id} : \prod_{a:A} \mathrm{hom}(a,a)$,

as well as the properties

- $\mathrm{idLeft} : \prod_{a,b:A} \prod_{f:\mathrm{hom}(a,b)} \mathrm{id}_b \circ f = f$,
- $\mathrm{idRight} : \ldots$,
- $\mathrm{assoc} : \prod_{a,b,c,d:A} \prod_{f,g,h:\ldots} h \circ (g \circ f) = (h \circ g) \circ f$.

Fundamental Groupoid of a 1-type

Be $X : \mathcal{U}$ a 1-type, $C : \mathcal{U}$ a set and $\iota : C \to X$.

- Let $\hom(x, y) :\equiv (\iota(x) =_X \iota(y))$,
- composition is the transitivity of equality,
- identity morphisms are $\mathrm{id}(x) :\equiv \mathrm{refl}_{\iota(x)}$.

Double Groupoids

Questions:

- How can we extend the concept of fundamental groupoids to include the second homotopy group?
- Which algebraic structures can capture the information?

$\rightsquigarrow$ Double groupoids and crossed modules are to different structures, which turn out to be equivalent as categories.

Double Categories

A *double category* is a category in which there are, besides objects and morphisms 2-cells which are bounded by four morphisms:

$$\begin{tikzcd}
\partial^-(\partial_1^-(u)) = \partial^-(\partial_2^-(u))\ \bullet \arrow[r, dash, "\partial_1^-(u)"] \arrow[d, dash, "\partial_2^-(u)"'] \arrow[dr, phantom, "u"] & \bullet\ \partial^+(\partial_1^-(u)) = \partial^-(\partial_2^+(u)) \arrow[d, dash, "\partial_2^+(u)"] \\
\partial^-(\partial_1^-(u)) = \partial^-(\partial_2^-(u))\ \bullet \arrow[r, dash, "\partial_1^+(u)"'] & \bullet\ \partial^+(\partial_1^+(u)) = \partial^+(\partial_2^+(u))
\end{tikzcd}$$

As a type family (D_0 is the carrier, D_1 the morphisms):

$$D_2 : \prod_{a,b,c,d:D_0} \prod_{f:D_1(a,b)} \prod_{g:D_1(c,d)} \prod_{h:D_1(a,c)} \prod_{i:D_1(b,d)} \mathcal{U}$$

Double Categories

A *double category* is a category in which there are, besides objects and morphisms 2-cells which are bounded by four morphisms:

$$\begin{tikzcd} a \arrow[r, dash, "f"] \arrow[d, dash, "h"'] \arrow[dr, phantom, "u"] & b \arrow[d, dash, "i"] \\ c \arrow[r, dash, "g"'] & d \end{tikzcd}$$

As a type family (D_0 is the carrier, D_1 the morphisms):

$$D_2 : \prod_{a,b,c,d:D_0} \prod_{f:D_1(a,b)} \prod_{g:D_1(c,d)} \prod_{h:D_1(a,c)} \prod_{i:D_1(b,d)} \mathcal{U}$$

Double Categories

Squares have to form a category vertically and horizontally, i. e.

- We have a vertical composition $\circ_1$ and a horizontal composition $\circ_2$, ...
- ... as well as vertical and horizontal "identity squares" id_1 and id_2.

$$\begin{aligned} \circ_1 : \prod_{\dots} & D_2(g_1, g_2, h_2, i_2) \\ & \to D_2(f, g_1, h_1, i_1) \\ & \to D_2(f, g_2, h_2 \circ h_1, i_2 \circ i_1) \end{aligned}$$

$$\begin{tikzcd}
\bullet \arrow[r, dash, "f"] \arrow[d, dash, "h_1"'] \arrow[dr, phantom, "u"] & \bullet \arrow[d, dash, "i_1"] \\
\bullet \arrow[r, dash, "g_1"] \arrow[d, dash, "h_2"'] \arrow[dr, phantom, "w"] & \bullet \arrow[d, dash, "i_2"] \\
\bullet \arrow[r, dash, "g_2"'] & \bullet
\end{tikzcd}$$

Double Categories

Squares have to form a category vertically and horizontally, i. e.

- We have a vertical composition $\circ_1$ and a horizontal composition $\circ_2$, ...
- ... as well as vertical and horizontal ”identity squares” id_1 and id_2.

$$\mathrm{id}_1 : \prod_{a,b:D_0} \prod_{f:D_1(a,b)} D_2(f, f, \mathrm{id}_a, \mathrm{id}_b)$$

$$\begin{tikzcd} \bullet \arrow[r, dash, "f"] \arrow[d, equal, "\mathrm{id}_a"'] \arrow[dr, phantom, "\mathrm{id}_1(f)"] & \bullet \arrow[d, equal, "\mathrm{id}_b"] \\ \bullet \arrow[r, dash, "f"'] & \bullet \end{tikzcd}$$

Double Categories

Squares have to form a category vertically and horizontally, i. e.

- Identity squares have to act as a left and right unit
- Associativity of square composition

Problem: For squares w, v, u we have

$$(w \circ_1 (v \circ_1 u)) : D_2(f, g, h_3 \circ (h_2 \circ h_1), i_3 \circ (i_2 \circ i_1)), \text{ but}$$
$$((w \circ_1 v) \circ_1 u) : D_2(f, g, (h_3 \circ h_2) \circ h_1, (i_3 \circ i_2) \circ i_1).$$

$\rightsquigarrow$ We have to transport along the associativity witness in the 1-skeleton:

$$\mathrm{assoc}_1(\ldots, w, v, u) : \mathrm{assoc}(i_3, i_2, i_1)_*(\mathrm{assoc}(h_3, h_2, h_1)_*(w \circ_1 (v \circ_1 u))) = ((w \circ_1 v) \circ_1 u)$$

Double Categories

Squares have to form a category vertically and horizontally, i. e.

- Identity squares have to act as a left and right unit
- Associativity of square composition

Problem: For squares w, v, u we have

$$(w \circ_1 (v \circ_1 u)) : D_2(f, g, h_3 \circ (h_2 \circ h_1), i_3 \circ (i_2 \circ i_1)), \text{ but}$$
$$((w \circ_1 v) \circ_1 u) : D_2(f, g, (h_3 \circ h_2) \circ h_1, (i_3 \circ i_2) \circ i_1).$$

$\rightsquigarrow$ We have to transport along the associativity witness in the 1-skeleton:

$$\begin{aligned}\mathrm{assoc}_1(\ldots, w, v, u) : \mathrm{assoc}(i_3, i_2, i_1)_*(\mathrm{assoc}(h_3, h_2, h_1)_*(w \circ_1 (v \circ_1 u)) \\ = ((w \circ_1 v) \circ_1 u)\end{aligned}$$

Double Categories

Squares have to form a category vertically and horizontally, i. e.

- Identity squares have to act as a left and right unit
- Associativity of square composition

Problem: For squares w, v, u we have

$$(w \circ_1 (v \circ_1 u)) : D_2(f, g, h_3 \circ (h_2 \circ h_1), i_3 \circ (i_2 \circ i_1)), \text{ but}$$
$$((w \circ_1 v) \circ_1 u) : D_2(f, g, (h_3 \circ h_2) \circ h_1, (i_3 \circ i_2) \circ i_1).$$

$\rightsquigarrow$ We have to transport along the associativity witness in the 1-skeleton:

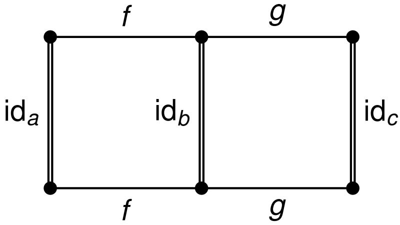
$$\begin{aligned}\mathrm{assoc}_1(\ldots, w, v, u) : \mathrm{assoc}(i_3, i_2, i_1)_*(\mathrm{assoc}(h_3, h_2, h_1)_*(w \circ_1 (v \circ_1 u))) \\ = ((w \circ_1 v) \circ_1 u)\end{aligned}$$

Double categories

Further axioms:

- Identity square must be distributive in the following sense (and the transposed case):

$$\prod_{a,b,c:D_0} \prod_{f:D_1(a,b)} \prod_{g:D_1(b,c)} \mathrm{id}_1(g \circ f) = \mathrm{id}_1(g) \circ_2 \mathrm{id}_1(f)$$



Double categories

Further axioms:

- Identity square must be distributive in the following sense (and the transposed case):

$$\prod_{a,b,c:D_0} \prod_{f:D_1(a,b)} \prod_{g:D_1(b,c)} \mathrm{id}_1(g \circ f) = \mathrm{id}_1(g) \circ_2 \mathrm{id}_1(f)$$

- Identity squares over identity morphisms are unique:

$$\prod_{a:D_0} \mathrm{id}_1(\mathrm{id}_a) =_{D_2(\mathrm{id}_a,\mathrm{id}_a,\mathrm{id}_a,\mathrm{id}_a)} \mathrm{id}_2(\mathrm{id}_a)$$

Thin Structures on Double Categories

Let $D = (D_0, D_1, D_2)$ be a double category. A *thin structure* is a subset of squares in D_2 such that:

- for each commuting boundary we give exactly one *thin* square:

$$\text{thin} : \prod_{a,b,c,d:D_0} \prod_{f,g,h,i:\ldots} (g \circ h = i \circ f) \to D_2(f, g, h, i),$$

- identity squares are thin and
- the composition of thin squares is thin.

Double groupoids

A *double groupoid* is a double category with thin structure such that all three categories involved are groupoids.

- Receive verically and horizontally ”mirrored” squares

$$\mathrm{inv}_1 : \prod_{\dots} D_2(f, g, h, i) \to D_2(g, f, h^{-1}, i^{-1})$$

- Double groupoids form a category **DGpd**, we call its morphisms *double functors*.

Double groupoids

A *double groupoid* is a double category with thin structure such that all three categories involved are groupoids.

- Receive verically and horizontally ”mirrored” squares

$$\mathrm{inv}_1 : \prod_{\ldots} D_2(f, g, h, i) \to D_2(g, f, h^{-1}, i^{-1})$$

- Double groupoids form a category **DGpd**, we call its morphisms *double functors*.

Double groupoids

A *double groupoid* is a double category with thin structure such that all three categories involved are groupoids.

- Receive verically and horizontally ”mirrored” squares

$$\mathrm{inv}_1 : \prod_{\dots} D_2(f, g, h, i) \to D_2(g, f, h^{-1}, i^{-1})$$

- Double groupoids form a category **DGpd**, we call its morphisms *double functors*.

Fundamental Double Groupoid

In Topology: Let $C \subseteq A \subseteq X$ be top. spaces. Define

$$
\begin{aligned}
R(X, A, C)_0 &:= C \\
R(X, A, C)_1 &:= \{\sigma : [0,1] \to A \mid \sigma \text{ cont.}, \sigma(0), \sigma(1) \in C\} \\
R(X, A, C)_2 &:= \Big\{\alpha : [0,1]^2 \to X \mid \alpha \text{ cont.}, \\
&\qquad \alpha(x, y) \in A, \text{ if } x \text{ or } y \in \{0,1\}, \\
&\qquad \alpha(x, y) \in C \text{ if } x \text{ and } y \in \{0,1\}\Big\}.
\end{aligned}
$$

Then, quotient by homotopies keeping corners fixed.

Fundamental Double Groupoid

Let $X, A, C : \mathcal{U}$ where isSet(C), is-1-type(A), and is-2-type(X).
Let $\iota : C \to A$ and $\iota' : A \to X$. Define

$$G_0 :\equiv C,$$

$$G_1 :\equiv \prod_{a,b:C} \iota(a) =_A \iota(b) \text{ and}$$

$$G_2 :\equiv \prod_{a,b,c,d:C} \prod_{f:G_1(a,b)} \cdots \prod_{i:G_1(b,d)} \mathrm{ap}_{\iota'}(h) \cdot \mathrm{ap}_{\iota'}(g) = \mathrm{ap}_{\iota'}(f) \cdot \mathrm{ap}_{\iota'}(i)$$

Crossed Modules

Let P be a groupoid. A *crossed module* on P is a family of groups $(M_p)_{p \in P}$ with group homomorphisms $\mu_p : M_p \to \hom_P(p, p)$ and a groupoid action $\phi : \hom(p, q) \times M_p \to M_q$, such that:

- $\mu_q(\phi(a, x)) = a \circ \mu_p(x) \circ a^{-1}$ for all $a \in \hom(p, q)$, $x \in M_p$ and
- $\phi(\mu_p(c), x) = c \cdot x \cdot c^{-1}$ for all $c, x : M_p$.

Crossed Modules

A *morphism of crossed modules* $(P, (M_p))$ and $(Q, (N_q))$ is a functor $F : P \to Q$ together with a family of group actions $\psi_p : M_p \to N_{F(p)}$, such that for all p the diagram

$$\begin{tikzcd} M_p \arrow[r, "\psi_p"] \arrow[d, "\mu_p"'] & N_{F(p)} \arrow[d, "\mu_{F(p)}"] \\ \hom(p,p) \arrow[r, "F"'] & \hom(F(p), F(p)) \end{tikzcd}$$

commutes and for all $a \in \hom(p, q)$ and $x \in M_p$ it holds that:

$$\psi_q(\phi(a, x)) = \phi(F(a), \psi_p(x)) \qquad .$$

The Equivalence

Crossed modules over groupoids, with the just defined morphisms, form a cateogry **XMod**.

Theorem (Ronald Brown)

The categories **XMod** *and* **DGpd** *are equivalent.*

The Equivalence

Define a functor γ : **DGpd** $\to$ **XMod** by defining $\gamma(G)$ as the crossed module with

$$\begin{aligned} P &:= (G_0, G_1), \\ M_p &:= \left\{ u \in G_2 \middle| \partial_1^+(u) = \partial_2^-(u) = \partial_2^+(u) = \epsilon(p) \right\} \text{ for } p \in G_0, \\ \mu &:= \partial_1^-, \text{ and} \\ \phi(a, u) &:= \mathrm{id}_1(a) \circ_2 u \circ_2 \mathrm{id}(a^{-1}). \end{aligned}$$

The Equivalence

In the other direction, consider λ : **XMod** $\to$ **DGpd**, defined, for a crossed module $(M_p)_{p \in P}$ by P on the 1-skeleton and by squares

$$G_2(f, g, h, i) = \left\{ m \in M_d \;\middle|\; \mu(m) = i \circ f \circ h^{-1} \circ g^{-1} \right\}.$$

The Lean theorem prover

- Project started in 2013 by Leonardo de Moura, Microsoft Research
- Emacs plugin: Soonho Kong, CMU
- Two modes: Proof irrelevance or HoTT
- Extensive libraries for both modes.

What did I formalize?

- Some necessary HoTT and category theory basics
- Double categories, thin structures, double groupoids and crossed modules
- Instantiation of the fundamental groupoid and double groupoid
- Large parts of the equivalence proof

Statistics

Theory	LoC	Compile time in s
`transport4`	49	0.333
`dbl_cat.`		
`basic`	224	5.272
`decl`	58	4.253
`dbl_gpd.`		
`basic`	199	6.375
`category_of`	41	1.314
`decl`	69	8.856
`functor`	582	47.997
`fundamental`	1270	21.091
`thin_structure.`		
`basic`	154	3.091
`decl`	33	0.766
`xmod.`		
`category_of`	25	0.327
`decl`	53	0.794
`morphism`	209	4.542

Statistics

Theory	LoC	Compile time in s
`equivalence.`		
`equivalence`	371	∗30.049
`gamma_functor`	51	6.109
`gamma`	164	6.054
`gamma_group`	229	5.973
`gamma_morphisms`	167	5.986
`gamma_mu_phi`	407	24.828
`lambda_functor`	27	4.086
`lambda`	569	16.148
`lambda_morphisms`	70	7.914

Double Categories in Lean

```
structure worm_precat {D₀ : Type} (C  : precategory D₀)
  (D₂ : Π ⦃a b c d : D₀⦄
    (f : hom a b) (g : hom c d) (h : hom a c) (i : hom b d), Type) :=
  (comp₁ : proof Π ⦃a b c₁ d₁ c₂ d₂ : D₀⦄
    ⦃f₁ : hom a b⦄ ⦃g₁ : hom c₁ d₁⦄ ⦃h₁ : hom a c₁⦄ ⦃i₁ : hom b d₁⦄
    ⦃g₂ : hom c₂ d₂⦄ ⦃h₂ : hom c₁ c₂⦄ ⦃i₂ : hom d₁ d₂⦄,
    (D₂ g₁ g₂ h₂ i₂) → (D₂ f₁ g₁ h₁ i₁)
    → (@D₂ a b c₂ d₂ f₁ g₂ (h₂ ∘ h₁) (i₂ ∘ i₁)) qed)
  (ID₁ : proof Π ⦃a b : D₀⦄ (f : hom a b), D₂ f f (ID a) (ID b) qed)
  (assoc₁ : proof Π ⦃a b c₁ d₁ c₂ d₂ c₃ d₃ : D₀⦄
    ⦃f  : hom a b⦄   ⦃g₁ : hom c₁ d₁⦄ ⦃h₁ : hom a c₁⦄ ⦃i₁ : hom b d₁⦄
    ⦃g₂ : hom c₂ d₂⦄ ⦃h₂ : hom c₁ c₂⦄ ⦃i₂ : hom d₁ d₂⦄
    ⦃g₃ : hom c₃ d₃⦄ ⦃h₃ : hom c₂ c₃⦄ ⦃i₃ : hom d₂ d₃⦄
    (w : D₂ g₂ g₃ h₃ i₃) (v : D₂ g₁ g₂ h₂ i₂) (u : D₂ f g₁ h₁ i₁),
    (assoc i₃ i₂ i₁) ▹ ((assoc h₃ h₂ h₁) ▹
        (comp₁ w (comp₁ v u))) = (comp₁ (comp₁ w v) u) qed)
  (id_left₁ : proof Π ⦃a b c d : D₀⦄
    ⦃f : hom a b⦄ ⦃g : hom c d⦄ ⦃h : hom a c⦄ ⦃i : hom b d⦄
    (u : D₂ f g h i),
    (id_left i) ▹ ((id_left h) ▹ (comp₁ (ID₁ g) u)) = u qed)
  (id_right₁ : proof Π ⦃a b c d : D₀⦄
    ⦃f : hom a b⦄ ⦃g : hom c d⦄ ⦃h : hom a c⦄ ⦃i : hom b d⦄
    (u : D₂ f g h i),
    (id_right i) ▹ ((id_right h) ▹ (comp₁ u (ID₁ f))) = u qed)
  (homH' : proof Π ⦃a b c d : D₀⦄
    ⦃f : hom a b⦄ ⦃g : hom c d⦄ ⦃h : hom a c⦄ ⦃i : hom b d⦄,
    is_hset (D₂ f g h i) qed)
```

Double Categories in Lean

```
structure dbl_precat {D₀ : Type} (C : precategory D₀)
  (D₂ : Π ⦃a b c d : D₀⦄
    (f : hom a b) (g : hom c d) (h : hom a c) (i : hom b d), Type)
  extends worm_precat C D₂,
    worm_precat C (λ ⦃a b c d : D₀⦄ f g h i, D₂ h i f g)
  renaming comp₁→comp₂ ID₁→ID₂ assoc₁→assoc₂
    id_left₁→id_left₂ id_right₁→id_right₂ homH'→homH'_dontuse :=
  (id_comp₁ : proof Π {a b c : D₀} (f : hom a b) (g : hom b c),
    ID₂ (g ∘ f) = comp₁ (ID₂ g) (ID₂ f) qed)
  (id_comp₂ : proof Π {a b c : D₀} (f : hom a b) (g : hom b c),
    ID₁ (g ∘ f) = comp₂ (ID₁ g) (ID₁ f) qed)
  (zero_unique : proof Π (a : D₀), ID₁ (ID a) = ID₂ (ID a) qed)
  (interchange : proof Π {a₀₀ a₀₁ a₀₂ a₁₀ a₁₁ a₁₂ a₂₀ a₂₁ a₂₂ : D₀}
    {f₀₀ : hom a₀₀ a₀₁} {f₀₁ : hom a₀₁ a₀₂} {f₁₀ : hom a₁₀ a₁₁}
    {f₁₁ : hom a₁₁ a₁₂} {f₂₀ : hom a₂₀ a₂₁} {f₂₁ : hom a₂₁ a₂₂}
    {g₀₀ : hom a₀₀ a₁₀} {g₀₁ : hom a₀₁ a₁₁} {g₀₂ : hom a₀₂ a₁₂}
    {g₁₀ : hom a₁₀ a₂₀} {g₁₁ : hom a₁₁ a₂₁} {g₁₂ : hom a₁₂ a₂₂}
    (x : D₂ f₁₁ f₂₁ g₁₁ g₁₂) (w : D₂ f₁₀ f₂₀ g₁₀ g₁₁)
    (v : D₂ f₀₁ f₁₁ g₀₁ g₀₂) (u : D₂ f₀₀ f₁₀ g₀₀ g₀₁),
    comp₁ (comp₂ x w) (comp₂ v u) = comp₂ (comp₁ x v) (comp₁ w u) qed)
```

Crossed Modules in Lean

```
structure xmod {P₀ : Type} [P : groupoid P₀] (M : P₀ → Group) :=
  (P₀_hset : is_hset P₀)
  (μ : Π ⦃p : P₀⦄, M p → hom p p)
  (μ_respect_comp : Π ⦃p : P₀⦄ (b a : M p), μ (b * a) = μ b ∘ μ a)
  (μ_respect_id : Π (p : P₀), μ 1 = ID p)
  (φ : Π ⦃p q : P₀⦄, hom p q → M p → M q)
  (φ_respect_id : Π ⦃p : P₀⦄ (x : M p), φ (ID p) x = x)
  (φ_respect_P_comp : Π ⦃p q r : P₀⦄ (b : hom q r) (a : hom p q) (x : M p),
    φ (b ∘ a) x = φ b (φ a x))
  (φ_respect_M_comp : Π ⦃p q : P₀⦄ (a : hom p q) (y x : M p),
    φ a (y * x) = (φ a y) * (φ a x))
  (CM1 : Π ⦃p q : P₀⦄ (a : hom p q) (x : M p), μ (φ a x) = a ∘ (μ x) ∘ a⁻¹)
  (CM2 : Π ⦃p : P₀⦄ (c x : M p), φ (μ c) x = c * (x * c⁻¹ᵍ))
```

Crossed Modules in Lean

Part of the proof that crossed modules form a category:

```
definition xmod_morphism_id_left :
  xmod_morphism_comp (xmod_morphism_id Y) f = f :=
begin
  cases f,
  fapply xmod_morphism_congr,
      apply idp,
    apply idp,
  repeat (apply eq_of_homotopy ; intros),
  apply idp,
end

universe variables l₁ l₂ l₃
definition cat_xmod :
  precategory.{(max l₁ l₂ l₃)+1 (max l₁ l₂ l₃)} Xmod.{l₁ l₂ l₃} :=
begin
  fapply precategory.mk,
    intros [X, Y], apply (xmod_morphism X Y),
    intros [X, Y], apply xmod_morphism_hset,
    intros [X, Y, Z, g, f], apply (xmod_morphism_comp g f),
    intro X, apply xmod_morphism_id,
    intros [X, Y, Z, W, h, g, f], apply xmod_morphism_assoc,
    intros [X, Y, f], apply xmod_morphism_id_left,
  intros [X, Y, f], apply xmod_morphism_id_right,
end
```

Conclusion

- Lean is suitable for formalization projects of this size
- Should have used pathovers etc.
- One should formalize the 2-dim. van Kampen theorem