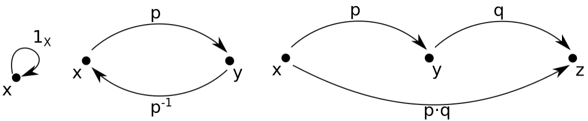
All derivations of groupoid laws are propositionally equal

Marc Lasson
INRIA – equipe projet $\pi r2$

May 15, 2014

The weak ω-groupoid structure of types

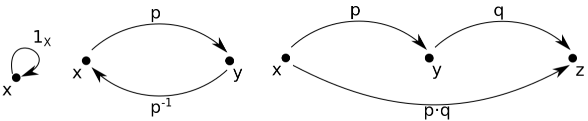
- The groupoid structure of types (Hofmann-Streicher).



  - `neutral` : $p \cdot 1 = p$,
  - `assoc` : $p \cdot (q \cdot r) = (p \cdot q) \cdot r$,
  - `involution` : $(p^{-1})^{-1} = p$

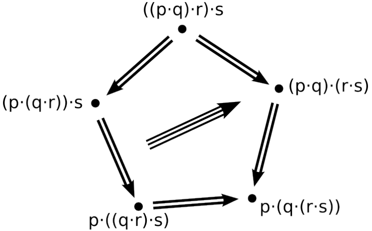
The weak ω-groupoid structure of types

- The groupoid structure of types (Hofmann-Streicher).



  - neutral : $p \cdot 1 = p$,
  - assoc : $p \cdot (q \cdot r) = (p \cdot q) \cdot r$,
  - involution : $(p^{-1})^{-1} = p$

- Weak ω-groupoid structure of types (Garner et al, Lumsdaine).



The synthetic approach

- Problem: Quite difficult to formalize ω-groupoids.
- Synthetic approach : Groupoid laws are proved when needed.

```
Definition left_action {X} {x y: X}
        (p : x = y) {z : X} {q : y = z}
        {r : y = z} (α : q = r)
         : p @ q  = p @ r.
induction α; induction q; induction p; reflexivity.
Defined.

Lemma assoc
   {X} {x y z u : X}
   (p : x = y) (q : y = z) (r : z = u) :
    (p @ q) @ r = p @ (q @ r).
induction r; induction q; induction p.
reflexivity.
Defined.

Lemma pentagon :
  forall X (x y z u v: X),
    forall (p : x = y)
           (q : y = z)
           (r : z = u)
           (s : u = v),
   (right_action (assoc p q r) s)
 @ (assoc p (q @ r) s)
 @ (left_action p (assoc q r s))
=  (assoc (p @ q) r s) @ (assoc p q (r @ s)).
intros.
induction s; induction r; induction q; induction p.
simpl.
reflexivity.
Defined.
```

Question: Do proof terms matter ?

Outline

1. Parametricity with identity types.
   Canonicity of identity functions:
   The identity function is the only term inhabiting $\forall X.X \to X$.
2. Polymorphic loop spaces.
   Canonicity of reflexivities in loop spaces:
   The reflexivity is the only term inhabiting a polymorphic loop space.
3. Syntactic approach of groupoid laws.
   Canonicity of groupoid laws:
   There is only one implementation of a given groupoid law.

Dependent parametricity theory in a nutshell

Logical predicates:

$$f \in |\forall x : A.B| \equiv \forall x : A, x_R : x \in |A|.(f\ x) \in |B|$$

Contexts:

$$[\![\Gamma, x : A]\!] \equiv [\![\Gamma]\!], x : A, x_R : x \in |A|$$

Abstraction theorem:

$$\frac{\Gamma \vdash M : A}{[\![\Gamma]\!] \vdash [\![M]\!] : M \in |A|}$$

Full definition:

$$\begin{array}{rcl}
[\![\lambda x : A.M]\!] & = & \lambda x : A, x_R : x \in |A|.[\![M]\!] \\
[\![M\ N]\!] & = & [\![M]\!]\ N\ [\![N]\!] \\
[\![x]\!] & = & x_R \\
[\![\forall x : A.B]\!] & = & \lambda f : \forall x : A.B.\forall x : A, x_R : x \in |A|.(f\ x) \in |B| \\
[\![\mathrm{Type}]\!] & = & \lambda \alpha : \mathrm{Type}\,.\alpha \to \mathrm{Type}
\end{array}$$

we have $M \in |A| \equiv [\![A]\!]\ M$.

The simplest "Theorem for free"

In standard type theories, the type `ID`

$$\forall X : \text{Type} . X \to X$$

is not provably uniquely inhabited. Ie. you cannot prove :

$$\forall f : \texttt{ID}, X : \text{Type}, x : X . f\ X\ x = x$$

But you can prove it is "unique in the syntax":

The simplest "Theorem for free"

In standard type theories, the type ID

$$\forall X : \mathrm{Type}\,.X \to X$$

is not provably uniquely inhabited. Ie. you cannot prove :

$$\forall f : \mathtt{ID}, X : \mathrm{Type}, x : X . f\ X\ x = x$$

But you can prove it is "unique in the syntax":

Theorem (Canonicity of the type of the identity)

If $\vdash M : \mathtt{ID}$, *then there exists :*

$$\vdash \pi_M : \forall X\ x . M\ X\ x = x$$

proof.

The abstraction theorem gives :

$$\vdash [\![M]\!] : M \in |\forall X : \mathrm{Type}\,.X \to X|$$

which undolds to a proof of :

$$\forall X : \mathrm{Type}, X_R : X \to \mathrm{Type}, x : X. \quad X_R\, x \to X_R\,(f\ X\ x)$$

We conclude by instantiating $X_R := \lambda y : X . y = x$. □

Identity types and parametricity

Type constructor:

$$\frac{M : A \qquad N : A}{M =_A N : \mathsf{Type}}$$

Introduction:

$$\frac{M : A}{1_M : M =_A M}$$

Elimination:

$$\frac{\begin{array}{c} x : A, p : M = x \vdash P : \mathsf{Type} \\ B : P[M/x, 1_M/p] \\ N : A \qquad U : M = N \end{array}}{\mathsf{J}(B, N, U) : P[N/x, U/p]}$$

Computation:

$$\mathsf{J}(B, M, 1) \equiv B$$

Transport:

$$\frac{U : M = N \qquad B : P[M/x]}{U_*(B) : P[N/x]}$$

Identity types and parametricity

Type constructor:

$$\frac{M : A \qquad N : A}{M =_A N : \text{Type}}$$

Introduction:

$$\frac{M : A}{1_M : M =_A M}$$

Elimination:

$$\frac{\begin{array}{c} x : A, p : M = x \vdash P : \text{Type} \\ B : P[M/x, 1_M/p] \\ N : A \qquad U : M = N \end{array}}{\mathsf{J}(B, N, U) : P[N/x, U/p]}$$

Computation:

$$\mathsf{J}(B, M, 1) \equiv B$$

Transport:

$$\frac{U : M = N \qquad B : P[M/x]}{U_*(B) : P[N/x]}$$

Translation of identity types:

$$U \in |M = N| \qquad \equiv \qquad U_*([\![M]\!]) = [\![N]\!]$$

Translation of reflexivity: $[\![1_M]\!] \equiv 1_{[\![M]\!]}$

$$\begin{aligned} [\![1_M]\!] \quad &: \quad 1_M \in |M = M| \\ &: \quad (1_M)_*([\![M]\!]) = [\![M]\!] \\ &: \quad [\![M]\!] = [\![M]\!] \end{aligned}$$

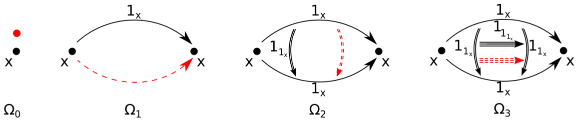
Translation of elimination:

$$[\![\mathsf{J}(B, N, U)]\!] = \mathsf{J}(\mathsf{J}([\![B]\!], N, U), [\![N]\!], [\![U]\!])$$

Translation of computations:

$$\begin{aligned} [\![\mathsf{J}(B, M, 1)]\!] \quad &\equiv \quad \mathsf{J}(\mathsf{J}([\![B]\!], M, 1), [\![M]\!], [\![1]\!]) \\ &\equiv \quad \mathsf{J}([\![B]\!], [\![M]\!], [\![1]\!]) \\ &\equiv \quad \mathsf{J}([\![B]\!], [\![M]\!], 1) \\ &\equiv \quad [\![B]\!] \end{aligned}$$

Loop spaces

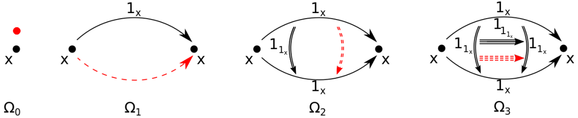


$$\begin{aligned}\Omega_n &: \forall X, X \to \text{Type}\\ \Omega_0(X,x) &:= X\\ \Omega_{n+1}(X,x) &:= \Omega_n(x = x, 1)\end{aligned}$$

$$\omega_n : \forall X\ x. \Omega_n(X,x)$$

$$\omega_n(X,x) \equiv 1_{\cdot_{\cdot_{\cdot_{1_x}}}} \Big\} \ n \text{ times}$$

Loop spaces



$$\begin{aligned} \Omega_n &: \forall X, X \to \text{Type} \\ \Omega_0(X,x) &:= X \\ \Omega_{n+1}(X,x) &:= \Omega_n(x = x, 1) \end{aligned}$$

$$\omega_n : \forall X\, x. \Omega_n(X,x)$$

$$\omega_n(X,x) \equiv 1_{\cdot_{\cdot_{\cdot_{1_x}}}} \Big\}\ n \text{ times}$$

Theorem (Canonicity of ω_n in Ω_n)

If $\vdash M : \forall X\, x. \Omega_n(X,x)$, *then there exists :*

$$\vdash \pi_M : \forall X\, x. M\, X\, x = \omega_n(X,x)$$

proof. We can prove :

$$p \in |\Omega_n(X,x)|[\lambda y : X.y = x / X_R] \to p = \omega_n(X,x)$$

by induction over n and by doing some algebra.
We conclude using $[\![M]\!] : \forall X\, X_R\, x\, x_R.(M\, X\, x) \in |\Omega_n(X,x)|$. □

Syntactic characterisation of groupoid laws

- Inspired by Guillaume Brunerie's notes.

Syntactic characterisation of groupoid laws

- Inspired by Guillaume Brunerie's notes.
- Contractible context:

  $X : \text{Type},\ x : X,\ x_1 : C_1,\ p_1 : M_1 = x_1,\ \dots,\ x_n : C_n,\ p_n : M_n = x_n$

  where x_i does not occur in M_i.

Syntactic characterisation of groupoid laws

- Inspired by Guillaume Brunerie's notes.
- Contractible context:

  $X : \mathrm{Type},\ x : X,\ x_1 : C_1,\ p_1 : M_1 = x_1,\ \ldots,\ x_n : C_n,\ p_n : M_n = x_n$

  where x_i does not occur in M_i.
- Let MLID be a minimal fragment of type theory, with:
  - Identity types (intro, elim, computation),
  - and restricted to contractible contexts.

  No function spaces, universes, sigma types, nor inductive families.

Syntactic characterisation of groupoid laws

- Inspired by Guillaume Brunerie's notes.
- Contractible context:

  $X : \text{Type},\ x : X,\ x_1 : C_1,\ p_1 : M_1 = x_1,\ \ldots,\ x_n : C_n,\ p_n : M_n = x_n$

  where x_i does not occur in M_i.
- Let MLID be a minimal fragment of type theory, with:
  - Identity types (intro, elim, computation),
  - and restricted to contractible contexts.

  No function spaces, universes, sigma types, nor inductive families.
- A groupoid law is a type $\forall \Gamma . C$ such that :

  $$\Gamma \vdash_{\mathrm{id}} C : \text{Type}$$

  with Γ a contractible context.

Syntactic characterisation of groupoid laws

- Inspired by Guillaume Brunerie's notes.
- Contractible context:

  $$X : \text{Type},\ x : X,\ x_1 : C_1,\ p_1 : M_1 = x_1,\ \dots,\ x_n : C_n,\ p_n : M_n = x_n$$

  where x_i does not occur in M_i.
- Let MLID be a minimal fragment of type theory, with:
  - Identity types (intro, elim, computation),
  - and restricted to contractible contexts.

  No function spaces, universes, sigma types, nor inductive families.
- A groupoid law is a type $\forall\Gamma.C$ such that :

  $$\Gamma \vdash_{\mathrm{id}} C : \text{Type}$$

  with Γ a contractible context.

Lemma (Groupoid laws in "$X : \text{Type}, x : X$" are loop spaces)

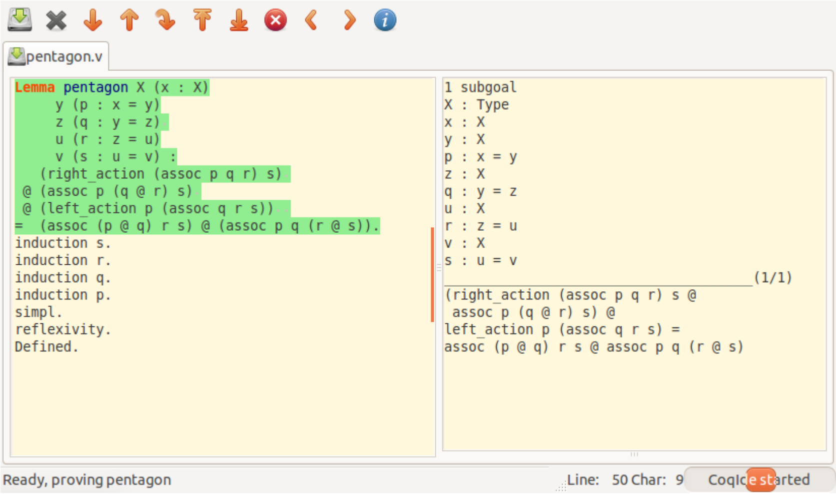
If $X : \text{Type}, x : X \vdash_{\mathrm{id}} M : C$, there exists n such that

$$M \equiv \omega_n(X, x) \qquad C \equiv \Omega_n(X, x)$$

Theorem (Groupoid laws are inhabited)

If $\Gamma \vdash_{\mathrm{id}} C : \mathsf{Type}$ *is a groupoid law, there exists* $\Gamma \vdash_{\mathrm{id}} \Theta_{\Gamma.C} : C$.

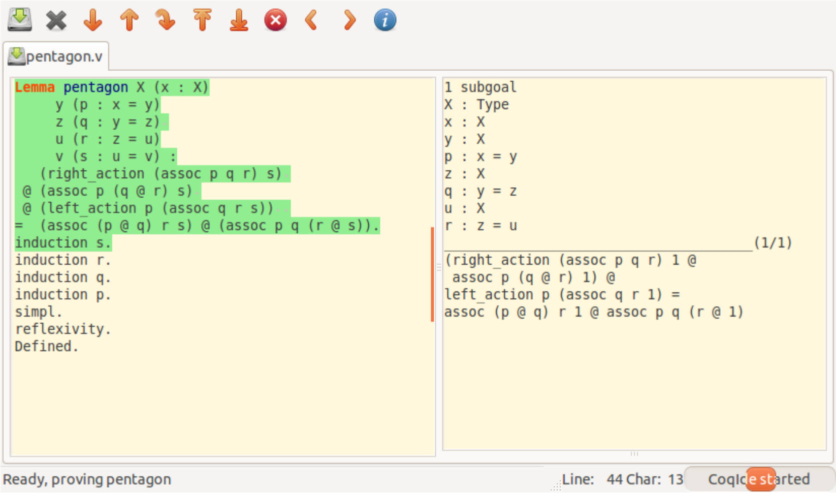
Example:



Theorem (Groupoid laws are inhabited)

If $\Gamma \vdash_{\mathrm{id}} C : \mathsf{Type}$ *is a groupoid law, there exists* $\Gamma \vdash_{\mathrm{id}} \Theta_{\Gamma.C} : C$*.*

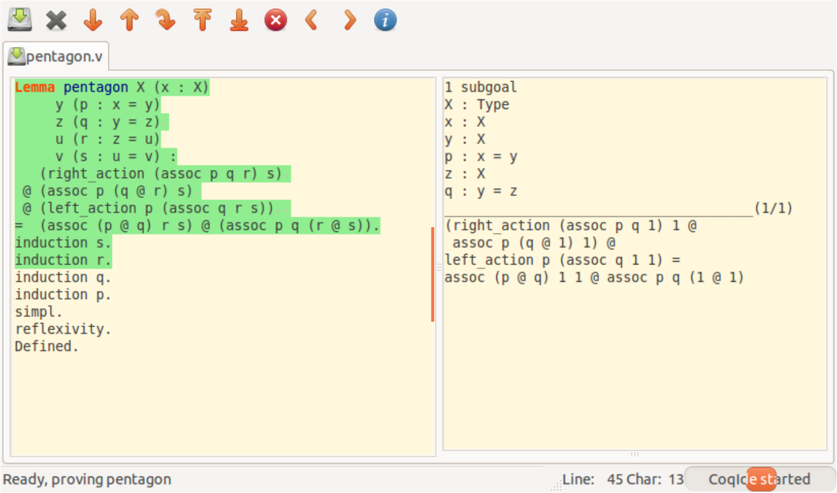
Example:



Theorem (Groupoid laws are inhabited)

If $\Gamma \vdash_{\mathrm{id}} C : \mathsf{Type}$ *is a groupoid law, there exists* $\Gamma \vdash_{\mathrm{id}} \Theta_{\Gamma.C} : C$.

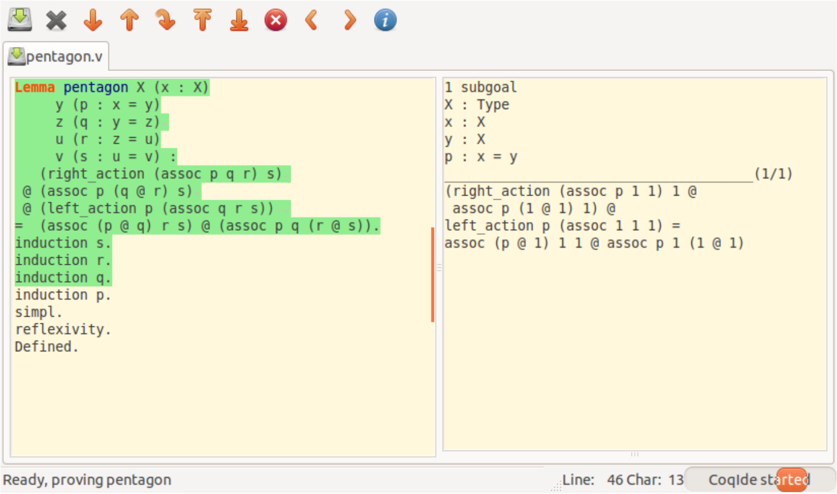
Example:



Theorem (Groupoid laws are inhabited)

If $\Gamma \vdash_{\mathrm{id}} C : \mathsf{Type}$ *is a groupoid law, there exists* $\Gamma \vdash_{\mathrm{id}} \Theta_{\Gamma.C} : C$.

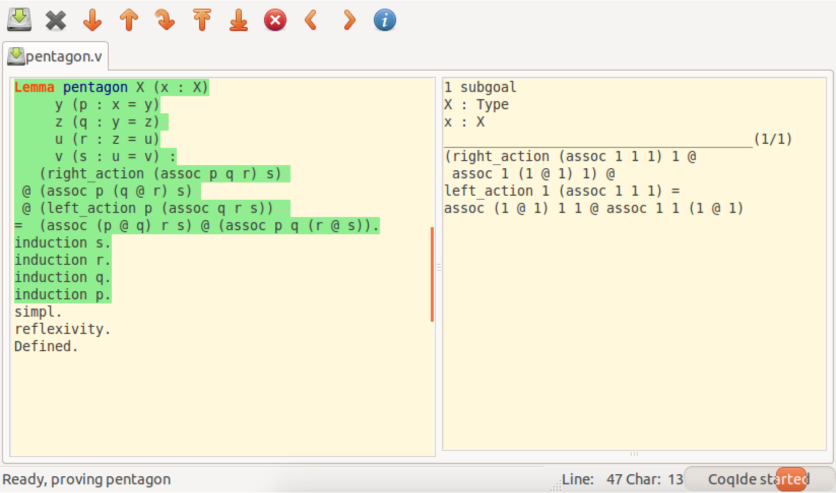
Example:



Theorem (Groupoid laws are inhabited)

If $\Gamma \vdash_{\mathrm{id}} C : \mathrm{Type}$ *is a groupoid law, there exists* $\Gamma \vdash_{\mathrm{id}} \Theta_{\Gamma.C} : C$.

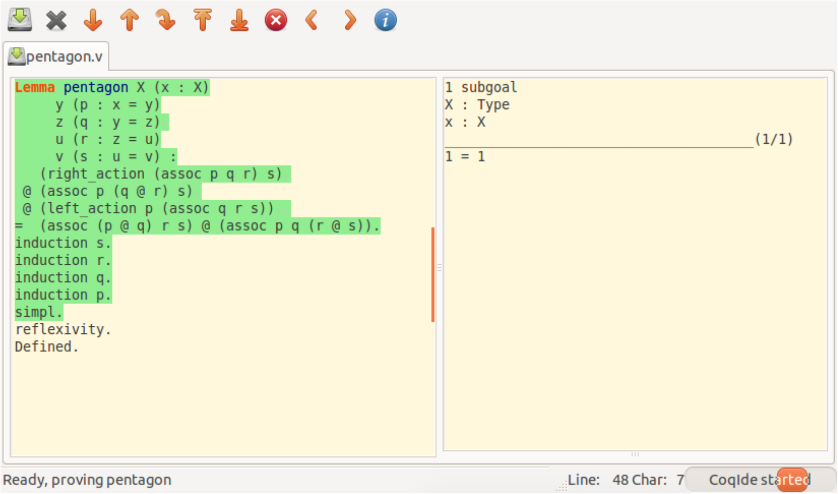
Example:



Theorem (Groupoid laws are inhabited)

If $\Gamma \vdash_{\mathrm{id}} C : \mathrm{Type}$ *is a groupoid law, there exists* $\Gamma \vdash_{\mathrm{id}} \Theta_{\Gamma.C} : C$.

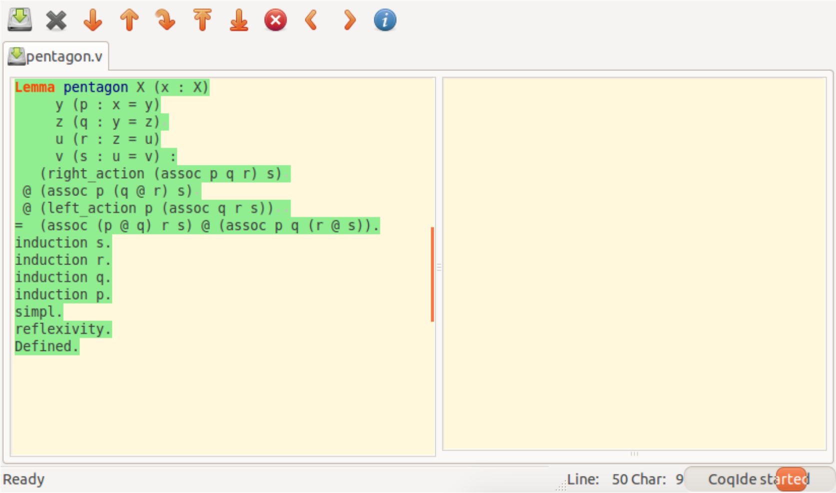
Example:



Theorem (Groupoid laws are inhabited)

If $\Gamma \vdash_{\mathrm{id}} C : \mathrm{Type}$ *is a groupoid law, there exists* $\Gamma \vdash_{\mathrm{id}} \Theta_{\Gamma.C} : C$.

Example:



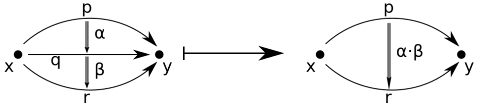
Unicity of the canonical inhabitant

Theorem (Canonicity of proofs of groupoid laws)

If $\forall \Gamma . C$ is a groupoid law and $\vdash M : \forall \Gamma . C$ and $\vdash M' : \forall \Gamma . C$ then there exists : $\vdash \pi_{M,M'} : \forall \vec{\gamma} : \Gamma . (M\,\vec{\gamma}) = (M'\,\vec{\gamma})$

Example vertical composition:

```
forall X (x y : X) (p : x = y) (q : x = y) (α : p = q)
         (z : X) (β : q = r), p = r.
```



We want to prove that

```
forall X x y p q α z β, M X x y p q α z β = M' X x y p q α z β
```

By successive inductions, it is enough to prove that :

```
forall X x, M X x x 1 1 1 x 1 = M' X x x 1 1 1 x 1
```

But LHS and RHS are an inhabitant of $\Omega_2(X, x)$!
Using the canonicity for loop spaces, they are equal to $\omega_2(X, x)$.

Conclusion

Open problems :

- Comparing models of MLID with definitions of groupoids.
- Compatibility with axioms:

UIP / K	✓
Proof-irrelevance	✓
Extensionality	✓
Excluded middle	✗
Univalence	???