Construction des espaces de Eilenberg-MacLane en HoTT

Camil Champin

Encadré par Emile Oleon et Samuel Mimram

Juin-Juillet 2023

1 Introduction du problème

1.1 Espaces d'Eilenberg-MacLane

On dispose d'un groupe G et on cherche à construire un espace, dit de *Eilenberg-Maclane*, dont le groupe fondamental sera G, et dont tout les autres groupes d'homotopie seront triviaux. On dis qu'un tel espace est un $K(G,1)$. Cette opération, appelée "délooping", correspond a une construction courante en théorie des catégories. Plus généralement, on passe de tout monoïde à une catégorie à 1 objet dont les morphismes correspondent au éléments monoïde, et leur composition à la loi de composition interne :

INSERER IMAGE

Les opérations (ou foncteurs) intervenant ici sont le *looping* Ω qui "liste" les morphismes qui bouclent autour d'un point. On cherchera donc à réaliser l'opération inverse B : le *délooping*.

Nous commencerons par quelques rappels de HoTT, afin de détailler la constructions de groupes, ainsi que les notions associées (morphismes, actions, torseurs).

1.2 Rappels de HoTT

Looping : Un tuple $X :\equiv (A, a_0)$ est un *type-pointé* si A est un type et $a_0 : A$. On définit le *loop-space* de ce type pointé :

$$\Omega(X) = (a_0 =_A a_0)$$

n-types : On rappelle quelques notions sur la structure des égalités d'un type donné :

- *Proposition* : type possèdant au plus un habitant. ($\prod_{x,y:A}(x = y)$)
- *Ensemble* : type dont les égalités sont des propositions.
- *Groupoïde* : type dont les égalités sont des ensembles.

On note Prop et Set les types des propriétés et des ensembles dans un univers $\mathcal{U}$. On remarque qu'une proposition est toujours un ensemble, qu'un ensemble est un groupoïde... On parle aussi de (-1)-type pour les propositions, de 0-type pour les ensembles, de 1-types pour les groupoïdes et plus généralement de n-type.

Troncatures : On parle de la n-troncature de A, notée $\|A\|_n$, pour le type satisfaisant la propriété universelle suivante :

INSERER DIAGRAMME

On utilisera nottement la troncature propositionnelle $\|\cdot\|_{-1}$. Intuitivement, celle-ci affirme qu'un type est non-vide sans préciser d'habitant. Cependant si ce qu'on cherche à montrer est une proposition P, on à le droit de faire comme-si on se donnait un habitant :

$$\|A\|_{-1} \to P \simeq A \to P$$

Types connexes : Dans un type connexe, tout les types égalité sont non-vides. Pour autant il ne s'agit pas toujours de propositions :

$$\prod_{x,y:X} \|x = y\|_{-1} \not\simeq \prod_{x,y:X} (x = y)$$

En particulier, une proposition (droite) a des égalités triviales, ce qui n' est pas le cas pour tout type connexe : un exemple est $\mathbb{S}_1$.

1.3 Notion de Groupe en HoTT

Définition 1.1. *On appelle* groupe *un tuple :*

- $S : Set$
- $e : S$
- $\mu : \sum_{\star:S\to S\to S} \prod_{g_1,g_2,g_3:S} (g_1 \star e = e \star g_1 = g_1) \times (g_1 \star (g_2 \star g_3) = (g_1 \star g_2) \star g_3)$
- $\iota : \sum_{\mathrm{inv}:S\to S} \prod_{g:S} (f(g) \star g = g \star f(g) = e)$

Un groupe est donc un ensemble muni d'une loi de composition interne associative, d'un neutre pour cette loi, et d'une fonction qui a chaque élément associe un inverse. A noter que dans la bibliographie HoTT, on appelle ce genre d'objet "groupe abstrait", par opposition aux groupes "concrets" qui sont justement les déloopings des groupes abstraits. Pour l'instant nous ne manipuleront que des groupes abstraits. On notera Group le type des groupes abstraits.

1.4 Morphismes et Actions de Groupes

Définition 1.2. *Si* G, H : Group*, une fonction* $f : G \to H$ *est un morphisme si elle est munie d'un témoin :*

$$h : \prod_{x,y:G} f(x \star_G y) = f(x) \star_H f(y)$$

On note alors $(f, h) : \mathrm{Hom}(G, H)$*.*

Définition 1.3. *Une action de (abstraite)* $G \curvearrowright X$ *(où* X : Set*) est une fonction :*

$$\phi : G \to X \to X$$

et de témoins :

$$n : \prod_{x:X} \phi(1)(x) = x$$

$$h : \prod_{g_1,g_2:X} \phi(g_1 \star g_2)(x) = \phi(g_1) \circ \phi(g_2)(x)$$

Et on note G-Set *les ensembles de* Set *munis d'une action de* G*.*

Définition 1.4. *On note* G*-morphisme un morphisme entre* A, B : G-Set *respectant l'action de* G*. On note* $A \simeq_G B$ *lorsque il existe un* G*-morphisme ayant un inverse. (un triplet avec une fonction, et les identifications des deux composées à la fonction identité). On dit que deux tels ensembles sont* G*-isomorphes.*

Définition 1.5. *On appelle* P_G *le torseur principal de* G*, le* G-Set *de l'action (abstraite) de* G *sur lui-même par multiplication à gauche.*

2 Construction de $K(G,1)$ par torseurs

Définition 2.1. *Si (A, a_0) est un type pointé, on note* $\mathrm{BAut}(A) :\equiv \sum_{x:A} \|x = a_0\|_{-1}$ *la composante connexe de a_0 dans A.*

Lemme 2.2. *Le loop-space de la composante connexe de a_0 est le loop-space de a_0 :*

$$\Omega(\mathrm{BAut}(A)), a_0) = \Omega(A, a_0)$$

Démonstration. $(\mathrm{BAut}(A), a_0)$ est un type pointé, et soit code la fibration :

$$\begin{aligned} \mathrm{code} : \mathrm{BAut}(A)) &\to \mathcal{U} \\ (x, !) &\mapsto (x =_A x) \end{aligned}$$

On pose la fonction decode qui à $(x, !_x)$ et $p : (x = a_0)$ associe un élément de :

$$((x, !_x) = (a_0, !_{a_0})) \simeq \sum_{p:(x=a_0)} (p^*(!_x) =!_{a_0})$$

Il suffit de donner un élément de ce second type. C'est un tuple, le premier élément est p, le second est une preuve de $(p^*(!_x) =!_{a_0})$ qu'on tire de la contractibilité de $\|a_0 = a_0\|_{-1}$. On vérifie les hypothèses du lemme encode-décode pour les loop-spaces :

1. $c_0 :\equiv \mathrm{refl}_{a_0} : \mathrm{code}(a_0)$
2. $\mathrm{decode} : \prod_{x:\mathrm{BAut}(A)} \mathrm{code}(x) \to ((a_0, !) = x)$
3. Si $c : \mathrm{code}(a_0) \equiv (a_0 = a_0)$, alors $\mathrm{decode}_{a_0}(c) = ((a_0, !) = (a_0, !))$ et : $\mathrm{transport}^{\mathrm{code}}(\mathrm{decode}_{a_0}(c), c_0) = c_0 \equiv \mathrm{refl}_{a_0}$ par construction du transport.
4. $\mathrm{decode}_{a_0}(c_0) = c_0$

Le lemme permet alors de conclure que $\Omega(BAut(A), a_0) \simeq \mathrm{code}(a_0) :\equiv \Omega(A, a_0)$, ce qui conclut par univalence. □

Lemme 2.3. *Deux ensembles G-isomorphes sont égaux :*

$$(A \simeq_G B) \simeq (A = B)$$

Démonstration. On commence par montrer qu'un G-morphisme est un G-isomorphisme si et seulement si le morphisme assiocié est une équivalence (l'essentiel est de vérifier que si c'est une équivalence sa réciproque est aussi un morphisme). Nottamment ceci donne :

$$(G \simeq_G H) \simeq \sum_{e:G\simeq H} \prod_{g_1,g_2:G} e(g_1 \star g_2) = e(g_1) \circ e(g_2)$$

Par théorème, pour montrer que la famille indexée par H : Group :

$$(G \simeq H) \to (G = H)$$

Il suffit de montrer que la contractibilité de la famille :

$$\sum_{H:\mathrm{Group}} (G \simeq H) \text{ i.e. } \sum_{H:\mathrm{Group}} \sum_{e:G\simeq H} \prod_{g_1,g_2:G} e(g_1 \star g_2) = e(g_1) \circ e(g_2)$$

A FINIR □

Lemme 2.4. $G \simeq (P_G \simeq_G P_G)$

Démonstration. On construit une fonction de $(P_G \simeq P_G) \to G$ et on montre que c'est une équivalence. On associe à tout G-morphisme (μ, h, e) l'élément $\mu(1)$. La réciproque est donc de forme $(g : G) \mapsto \cdot \star g$, qui est clairement un G-isomorphisme. Enfaite, tout les G-isomorphismes sont de cette forme car (par h) : $\mu(g) = g\mu(1)$. □

On peut à présent donner une première construction d'un délooping de G :

Définition 2.5. *On pose le type des G-torseurs :*

$$BG :\equiv \mathrm{Torsor}_G :\equiv \sum_{X:\text{G-Set}} \|X = P_G\|$$

Théorème 2.1. *L'espace des G-torseurs est un délooping de G :*

$$\Omega(BG) \simeq G$$

Démonstration.

$$\begin{aligned} BG &\equiv BAut((\text{G-Set}, P_G)) \\ \Omega(BG) &\simeq \Omega((\text{G-Set}, P_G)) \\ &\simeq (P_G = P_G) \\ &\simeq (P_G \simeq_G P_G) \\ &\simeq G \end{aligned}$$

□

3 Construction minimale à partir de générateurs

Nous souhaitons à présent une construction plus simple de BG, on suppose qu'on dispose d'une famille génératrice $g_1, \cdots, g_n$ de G :

$$BG = \mathrm{Torsor}_{g_1 \cdots g_n} :\equiv \sum_{X:Set} \sum_{f_1,\cdots,f_n : X \simeq X} \|(X, f_1, \cdots, f_n) = (G, \lambda x.g_1 x, \cdots, \lambda x.g_n x)\|_{-1}$$

On va construire un inverse à la fonction :

$$\begin{aligned} eq_1 : \mathrm{Torsor}_{g_1 \cdots g_n} &\to \mathrm{Torsor}_G \\ ((X, f_1, \cdots, f_n), e) &\mapsto ([1], [2]) \end{aligned}$$

[1] On construit un G-set en posant une action ϕ sur X :
Pour tout $g = g_{i_1} \cdots g_{i_r}$ on pose $\phi(g) = f_{i_1} \circ \cdots \circ f_{i_r}$ Dont on montre rapidement qu' elle respecte la loi de groupe.

[2] On veut montrer $\|X = P_G\|_{-1}$ qui est une propriété donc on peut utiliser :

$$p : (X, f_1, \cdots, f_n) = (G, \lambda x.g_1 x, \cdots, \lambda x.g_n x)$$

C'est à dire qu'on a une équivalence $f : (X \simeq G)$ qui fait commuter pour tout i :
INSERER DIAGRAMME
Et on veut une équivalence $h : (X \simeq G)$ qui fasse commuter pour tout g :
INSERER DIAGRAMME
i.e.
INSERER DIAGRAMME
Ce que f vérifie :
INSERER DIAGRAMME