Computational Proofs in Synthetic Homotopy Theory

Anders Mörtberg

Homotopy Type Theory and Computing – Classical and Quantum, April 19, 2024

Outline

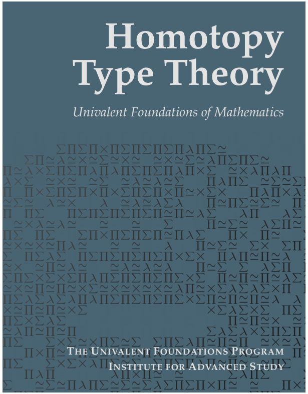
Outline

Synthetic algebraic topology in HoTT

By representing spaces as higher inductive types we can develop algebraic topology *synthetically* using univalence

This is well-suited for computer formalization and leads to very compact and elegant proofs

But, as univalence is added axiomatically in Book HoTT we cannot generally compute with these results in proof assistants...



Computational synthetic algebraic topology

Examples of computations that one might want to do in a formalization:

- Compute with functions

$$\phi : \pi_n(X) \to G \qquad\qquad \psi : H^n(X) \to G$$

  to compute winding numbers or prove that $\pi_n(X)$ and $H^n(X)$ are generated by particular elements

Computational synthetic algebraic topology

Examples of computations that one might want to do in a formalization:

- Compute with functions

$$\phi : \pi_n(X) \to G \qquad\qquad \psi : H^n(X) \to G$$

  to compute winding numbers or prove that $\pi_n(X)$ and $H^n(X)$ are generated by particular elements

- Compute with group/ring operations on $\pi_n(X)$, $H^n(X)$, $H^*(X)$, $H_n(X)$, ..., to distinguish spaces/types, e.g. $\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1 \neq \mathbb{T}$

Computational synthetic algebraic topology

Examples of computations that one might want to do in a formalization:

- Compute with functions

$$\phi : \pi_n(X) \to G \qquad\qquad \psi : H^n(X) \to G$$

  to compute winding numbers or prove that $\pi_n(X)$ and $H^n(X)$ are generated by particular elements

- Compute with group/ring operations on $\pi_n(X)$, $H^n(X)$, $H^*(X)$, $H_n(X)$, ..., to distinguish spaces/types, e.g. $\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1 \neq \mathbb{T}$

- Characterize $\pi_n(X)$, $H^n(X)$, $H^*(X)$, $H_n(X)$ by computation, e.g. the Brunerie number $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/\beta\mathbb{Z}$

Computational synthetic algebraic topology

Examples of computations that one might want to do in a formalization:

- Compute with functions

$$\phi : \pi_n(X) \to G \qquad\qquad \psi : H^n(X) \to G$$

  to compute winding numbers or prove that $\pi_n(X)$ and $H^n(X)$ are generated by particular elements

- Compute with group/ring operations on $\pi_n(X)$, $H^n(X)$, $H^*(X)$, $H_n(X)$, ..., to distinguish spaces/types, e.g. $\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1 \neq \mathbb{T}$

- Characterize $\pi_n(X)$, $H^n(X)$, $H^*(X)$, $H_n(X)$ by computation, e.g. the Brunerie number $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/\beta\mathbb{Z}$

- Prove tedious small proof steps by `refl` to shorten formal proofs

The cubical paradigm in HoTT

Theorem (Bezem-Coquand-Huber, 2013)

Univalent Type Theory has a constructive model in substructural Kan cubical sets ("BCH model").

This led to development of a variety of structural cubical set models and cubical type theories:

Theorem (Cohen-Coquand-Huber-M., 2015)

Univalent Type Theory has a constructive model in De Morgan Kan cubical sets ("CCHM model").

In cubical type theory we have a **univalence theorem** with computational content:

$$\mathsf{ua} : (A\ B : \mathcal{U}) \to (\mathsf{Path}_{\mathcal{U}}\ A\ B) \simeq (A \simeq B)$$

The many cubical models and type theories

	Structural	$\mathbb{I}$ operations	Kan operations	Diag. cofib.
BCH			$0 \to r, 1 \to r$	
CCHM	✓	$\wedge, \vee, \neg$ (DM alg.)	$0 \to 1$	
“Dedekind”	✓	$\wedge, \vee$ (dist. lattice)	$0 \to 1, 1 \to 0$	
Orton-Pitts	✓	$\wedge, \vee$ (conn. alg.)	$0 \to 1, 1 \to 0$	
Cartesian (A, AFH, ABCFHL)[1]	✓		$r \to s$	✓
Equivariant (ACCRS)[2]	✓		$\vec{r} \to \vec{s}$	✓
Cavallo-Sattler	✓	$\vee$ / $\wedge$	$0 \to r, 1 \to r$	✓

The last two are known to be equivalent to spaces

Comparison and unification of the Kan operations: *Unifying Cubical Models of Univalent Type Theory* (Cavallo-M.-Swan, CSL'20)

[1] Awodey, Angiuli-Favonia-Harper, Angiuli-Brunerie-Coquand-Favonia-Harper-Licata

[2] Awodey-Cavallo-Coquand-Riehl-Sattler

Cubical proof assistants

These different cubical type theories satisfy good metatheoretic properties: canonicity (Huber, AFH), normalization and decidable typechecking (Sterling-Angiuli)

There are also many cubical proof assistants: `cubical`, `cubicaltt`, `yacctt`, **RedPRL**, **redtt**, **cooltt**, `Cubical Agda`...

In `Cubical Agda` we have explored how to do synthetic proofs *computationally*, in particular by computing a Brunerie number

All results have been formalized in: `https://github.com/agda/cubical/`

Outline

Synthetic homotopy theory

The nth *homotopy group* of a pointed type X can be defined as:[3]

$$\pi_n(X) = \|\mathbb{S}^n \to_\star X\|_0$$

[3]Equivalently: $\pi_n(X) = \|\Omega^n(X)\|_0$

Synthetic homotopy theory

The *n*th *homotopy group* of a pointed type X can be defined as:[3]

$$\pi_n(X) = \|\mathbb{S}^n \to_\star X\|_0$$

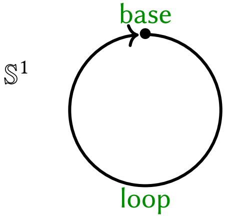
These groups constitute a topological invariant, making them a powerful tool for establishing whether two given spaces are homotopy equivalent

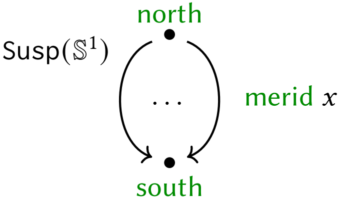
- $\pi_0(X)$ characterizes the connected components of X
- $\pi_1(X)$ characterizes equivalence classes the loops in X up to homotopy
- $\pi_n(X)$, for $n > 1$, characterizes of n-dimensional loops up to homotopy

[3] Equivalently: $\pi_n(X) = \|\Omega^n(X)\|_0$

Synthetic homotopy theory

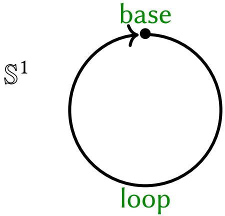
Using univalence we can prove properties of $\pi_n(X)$ for concrete spaces X represented using HITs

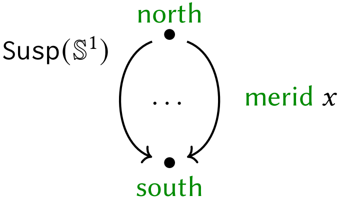




Synthetic homotopy theory

Using univalence we can prove properties of $\pi_n(X)$ for concrete spaces X represented using HITs





Example: $\pi_1(\mathbb{S}^1) \simeq \mathbb{Z}$ can be proved using the encode-decode method (Licata-Shulman '13)

Many other standard results allowing us to characterize homotopy groups of spheres can be found in the HoTT book: the Hopf fibration, Freudenthal suspension theorem, long exact sequence of homotopy groups, connectivity of spheres, ...

Homotopy groups of spheres synthetically

However, for many spaces, these groups tend to become increasingly esoteric and difficult to compute for large n

	π_1	π_2	π_3	π_4	π_5	π_6	π_7	π_8	π_9	π_{10}
$\mathbb{S}^1$	$\mathbb{Z}$	0	0	0	0	0	0	0	0	0
$\mathbb{S}^2$	0	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}_{12}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}_3$	$\mathbb{Z}_{15}$
$\mathbb{S}^3$	0	0	$\mathbb{Z}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}_{12}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z}_3$	$\mathbb{Z}_{15}$
$\mathbb{S}^4$	0	0	0	$\mathbb{Z}$	$\mathbb{Z}_2$	$\mathbb{Z}_2$	$\mathbb{Z} \times \mathbb{Z}_{12}$	$\mathbb{Z}_2 \times \mathbb{Z}_2$	$\mathbb{Z}_2 \times \mathbb{Z}_2$	$\mathbb{Z}_{24} \times \mathbb{Z}_3$

The fourth homotopy group of the 3-sphere in HoTT

Guillaume Brunerie's PhD thesis contains a synthetic proof in Book HoTT of:

Theorem (Brunerie, 2016)

The fourth homotopy group of the 3-sphere is $\mathbb{Z}/2\mathbb{Z}$, that is, $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$

The fourth homotopy group of the 3-sphere in HoTT

Guillaume Brunerie's PhD thesis contains a synthetic proof in Book HoTT of:

Theorem (Brunerie, 2016)

The fourth homotopy group of the 3-sphere is $\mathbb{Z}/2\mathbb{Z}$, that is, $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$

The proof is one of the most impressive pieces of synthetic homotopy theory to date and uses lots of advanced classical machinery developed synthetically in HoTT:

The fourth homotopy group of the 3-sphere in HoTT

Guillaume Brunerie's PhD thesis contains a synthetic proof in Book HoTT of:

Theorem (Brunerie, 2016)

The fourth homotopy group of the 3-sphere is $\mathbb{Z}/2\mathbb{Z}$*, that is,* $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$

The proof is one of the most impressive pieces of synthetic homotopy theory to date and uses lots of advanced classical machinery developed synthetically in HoTT: *symmetric monoidal structure of smash products, (integral) cohomology rings, the Mayer-Vietoris and Gysin sequences, Hopf invariant homomorphism, Whitehead products, the iterated Hopf construction, Blakers-Massey, ...*

The fourth homotopy group of the 3-sphere in HoTT

Guillaume Brunerie's PhD thesis contains a synthetic proof in Book HoTT of:

Theorem (Brunerie, 2016)

The fourth homotopy group of the 3-sphere is $\mathbb{Z}/2\mathbb{Z}$, that is, $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$

The proof is one of the most impressive pieces of synthetic homotopy theory to date and uses lots of advanced classical machinery developed synthetically in HoTT: *symmetric monoidal structure of smash products, (integral) cohomology rings, the Mayer-Vietoris and Gysin sequences, Hopf invariant homomorphism, Whitehead products, the iterated Hopf construction, Blakers-Massey, ...*

Furthermore, the proof is fully constructive!

The Brunerie number

The theorem can be phrased as:

$$\textit{there exists a number } \beta : \mathbb{Z} \textit{ such that } \pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/\beta\mathbb{Z}$$

[4] Using only 1-HITs

The Brunerie number

The theorem can be phrased as:

$$\textit{there exists a number } \beta : \mathbb{Z} \textit{ such that } \pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/\beta\mathbb{Z}$$

Appendix B of Brunerie's thesis contains a complete and concise definition of β as the image of 1 under a sequence of 12 maps:[4]

$$\mathbb{Z} \longrightarrow \Omega(\mathbb{S}^1) \longrightarrow \Omega^2(\mathbb{S}^2) \longrightarrow \Omega^3(\mathbb{S}^3)$$

$$\longrightarrow \Omega^3(\mathbb{S}^1 * \mathbb{S}^1) \xrightarrow{\Omega^3\alpha} \Omega^3(\mathbb{S}^2) \xrightarrow{h} \Omega^3(\mathbb{S}^1 * \mathbb{S}^1) \longrightarrow \Omega^3(\mathbb{S}^3)$$

$$\longrightarrow \Omega^2\|\mathbb{S}^2\|_2 \longrightarrow \Omega\|\Omega(\mathbb{S}^2)\|_1 \longrightarrow \|\Omega^2(\mathbb{S}^2)\|_0 \longrightarrow \Omega(\mathbb{S}^1) \longrightarrow \mathbb{Z}$$

[4] Using only 1-HITs

The Brunerie number

On page 85 Brunerie says (for $n := |\beta|$):

> *This result is quite remarkable in that even though it is a constructive proof, it is not at all obvious how to actually compute this n. At the time of writing, we still haven't managed to extract its value from its definition. A complete and concise definition of this number n is presented in appendix B, for the benefit of someone wanting to implement it in a prospective proof assistant. In the rest of this thesis, we give a mathematical proof in homotopy type theory that n = 2.*

Computing the Brunerie number

As the above cubical systems satisfy canonicity it should *in principle* be possible to use them to compute the Brunerie number...

Computing the Brunerie number

As the above cubical systems satisfy canonicity it should *in principle* be possible to use them to compute the Brunerie number... But this turned out to be **a lot** harder than expected!

Computing the Brunerie number

As the above cubical systems satisfy canonicity it should *in principle* be possible to use them to compute the Brunerie number... But this turned out to be **a lot** harder than expected!

First attempt in December 2014 when Guillaume visited Chalmers and tried to compute it with Thierry Coquand, Simon Huber, and myself using `cubical` (based on BCH model)

Computing the Brunerie number

As the above cubical systems satisfy canonicity it should *in principle* be possible to use them to compute the Brunerie number... But this turned out to be **a lot** harder than expected!

First attempt in December 2014 when Guillaume visited Chalmers and tried to compute it with Thierry Coquand, Simon Huber, and myself using `cubical` (based on BCH model)

Then many years of failed attempts by many experts and even using a supercomputer, see `https://mathstodon.xyz/@mortberg/111379612437759043` for a timeline

Computing the Brunerie number

As the above cubical systems satisfy canonicity it should *in principle* be possible to use them to compute the Brunerie number... But this turned out to be **a lot** harder than expected!

First attempt in December 2014 when Guillaume visited Chalmers and tried to compute it with Thierry Coquand, Simon Huber, and myself using `cubical` (based on BCH model)

Then many years of failed attempts by many experts and even using a supercomputer, see `https://mathstodon.xyz/@mortberg/111379612437759043` for a timeline

In 2022 Axel Ljungström and I finally had a breakthrough...

Computing the Brunerie number

As the above cubical systems satisfy canonicity it should *in principle* be possible to use them to compute the Brunerie number... But this turned out to be **a lot** harder than expected!

First attempt in December 2014 when Guillaume visited Chalmers and tried to compute it with Thierry Coquand, Simon Huber, and myself using `cubical` (based on BCH model)

Then many years of failed attempts by many experts and even using a supercomputer, see `https://mathstodon.xyz/@mortberg/111379612437759043` for a timeline

In 2022 Axel Ljungström and I finally had a breakthrough... A variation on the Brunerie number normalizes to -2 in just a few seconds in `Cubical Agda`!

Formalizing $\pi_4(\mathbb{S}^3) \cong \mathbb{Z}/2\mathbb{Z}$ and Computing a Brunerie Number in Cubical Agda

We have a LICS'23 paper about the details: `https://arxiv.org/abs/2302.00151`

The paper contains 3 fully formalized proofs:

1. Streamlined and complete version of Brunerie's original proof
2. Axel's new proof
3. A computational proof relying on normalization of the variation on β

Proofs 1 and 2 work in Book HoTT, proof 3 relies on cubical normalization

Brunerie's theorem: part 1 (chapters 1–3)

In the first half of the thesis (chapters 1–3) Guillaume constructs a map $g : \mathbb{S}^3 \to \mathbb{S}^2$

g is defined as the composition of a sequence of (pointed) maps $\mathbb{S}^3 \to \mathbb{S}^1 * \mathbb{S}^1 \to \mathbb{S}^2 \vee \mathbb{S}^2 \to \mathbb{S}^2$

Brunerie's theorem: part 1 (chapters 1–3)

In the first half of the thesis (chapters 1–3) Guillaume constructs a map $g : \mathbb{S}^3 \longrightarrow \mathbb{S}^2$

g is defined as the composition of a sequence of (pointed) maps $\mathbb{S}^3 \longrightarrow \mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^2 \vee \mathbb{S}^2 \longrightarrow \mathbb{S}^2$

Let $e : \pi_3(\mathbb{S}^2) \simeq \mathbb{Z}$ and define $\beta := e(|\ g\ |_0)$, the first main theorem is then that:

Theorem (Brunerie, Corollary 3.4.5)

We have $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/\beta\mathbb{Z}$

Contents

Brunerie's proof: part 2 (chapters 4–6)

- Symmetric monoidal structure of smash products

  $\implies$ The graded ring structure of the cup product $\smile : H^i(X) \times H^j(X) \to H^{i+j}(X)$
- The *Mayer-Vietoris* sequence
- The *Gysin Sequence*
- The *Hopf Invariant* homomorphism
- The *Iterated Hopf Construction*

Brunerie's proof: part 2 (chapters 4–6)

- **Symmetric monoidal structure of smash products**

  $\Longrightarrow$ The graded ring structure of the cup product
  $\smile : H^i(X) \times H^j(X) \to H^{i+j}(X)$
- The *Mayer-Vietoris* sequence
- The *Gysin Sequence*
- The *Hopf Invariant* homomorphism
- The *Iterated Hopf Construction*

Brunerie's proof: part 2 (chapters 4–6)

- $(A \wedge B \to_\star C) \simeq (A \to_\star (B \to_\star C))$ **workaround** (Brunerie-Ljungström-M. CSL'22)
  $\Longrightarrow$ The graded ring structure of the cup product $\smile : H^i(X) \times H^j(X) \to H^{i+j}(X)$
- The *Mayer-Vietoris* sequence
- The *Gysin Sequence*
- The *Hopf Invariant* homomorphism
- The *Iterated Hopf Construction*

Brunerie's proof: part 2 (chapters 4–6)

- **Symmetric monoidal structure of smash products** (Ljungström HoTT'23)
  $\Longrightarrow$ The graded ring structure of the cup product
  $\smile: H^i(X) \times H^j(X) \to H^{i+j}(X)$
- The *Mayer-Vietoris* sequence
- The *Gysin Sequence*
- The *Hopf Invariant* homomorphism
- The *Iterated Hopf Construction*

New proof

Having finished the formalization of chapters 4–6 Axel realized that one can actually simplify the proof a lot and completely avoid the second half of Brunerie's thesis

The new proof is very elementary – doesn't use any complicated theory!

New proof

Having finished the formalization of chapters 4–6 Axel realized that one can actually simplify the proof a lot and completely avoid the second half of Brunerie's thesis

The new proof is very elementary – doesn't use any complicated theory!

Idea: trace the maps by hand using clever tricks and choices

Sketch of new proof

Recall that $\beta := e(|g|_0)$ for $e : \pi_3(\mathbb{S}^2) \simeq \mathbb{Z}$ and $g : \mathbb{S}^3 \longrightarrow \mathbb{S}^2$. The goal is to show that $|\beta| = 2$

Sketch of new proof

Recall that $\beta := e(|g|_0)$ for $e : \pi_3(\mathbb{S}^2) \simeq \mathbb{Z}$ and $g : \mathbb{S}^3 \longrightarrow \mathbb{S}^2$. The goal is to show that $|\beta| = 2$

In fact, g is defined as the precomposition of a not very complicated map $\mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^2$ with the somewhat complicated equivalence $f : \mathbb{S}^3 \simeq \mathbb{S}^1 * \mathbb{S}^1$

Sketch of new proof

Recall that $\beta := e(|g|_0)$ for $e : \pi_3(\mathbb{S}^2) \simeq \mathbb{Z}$ and $g : \mathbb{S}^3 \longrightarrow \mathbb{S}^2$. The goal is to show that $|\beta| = 2$

In fact, g is defined as the precomposition of a not very complicated map $\mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^2$ with the somewhat complicated equivalence $f : \mathbb{S}^3 \simeq \mathbb{S}^1 * \mathbb{S}^1$

One of Axel's tricks in the proof is to define $\pi_3^*(A) := ||\mathbb{S}^1 * \mathbb{S}^1 \longrightarrow_\star A||_0$ and work with it instead so that f can be avoided

Sketch of new proof

We can now decompose $e : \pi_3(\mathbb{S}^2) \simeq \mathbb{Z}$ as:

$$\pi_3(\mathbb{S}^2) \overset{e_1}{\simeq} \pi_3^*(\mathbb{S}^2) \overset{e_2}{\simeq} \pi_3^*(\mathbb{S}^1 * \mathbb{S}^1) \overset{e_3}{\simeq} \pi_3^*(\mathbb{S}^3) \overset{e_4}{\simeq} \mathbb{Z}$$

Sketch of new proof

We can now decompose $e : \pi_3(\mathbb{S}^2) \simeq \mathbb{Z}$ as:

$$\pi_3(\mathbb{S}^2) \overset{e_1}{\simeq} \pi_3^*(\mathbb{S}^2) \overset{e_2}{\simeq} \pi_3^*(\mathbb{S}^1 * \mathbb{S}^1) \overset{e_3}{\simeq} \pi_3^*(\mathbb{S}^3) \overset{e_4}{\simeq} \mathbb{Z}$$

We can also give explicit definitions of

$$\eta_1 : \mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^2 \qquad \eta_2 : \mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^1 * \mathbb{S}^1 \qquad \eta_3 : \mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^3$$

such that

$$e_1(|g|_0) = |\eta_1|_0 \qquad e_2(|\eta_1|_0) = |\eta_2|_0 \qquad e_3(|\eta_2|_0) = |\eta_3|_0 \qquad e_4(|\eta_3|_0) = -2$$

Sketch of new proof

We can now decompose $e : \pi_3(\mathbb{S}^2) \simeq \mathbb{Z}$ as:

$$\pi_3(\mathbb{S}^2) \overset{e_1}{\simeq} \pi_3^*(\mathbb{S}^2) \overset{e_2}{\simeq} \pi_3^*(\mathbb{S}^1 * \mathbb{S}^1) \overset{e_3}{\simeq} \pi_3^*(\mathbb{S}^3) \overset{e_4}{\simeq} \mathbb{Z}$$

We can also give explicit definitions of

$$\eta_1 : \mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^2 \qquad \eta_2 : \mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^1 * \mathbb{S}^1 \qquad \eta_3 : \mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^3$$

such that

$$e_1(|g|_0) = |\eta_1|_0 \qquad e_2(|\eta_1|_0) = |\eta_2|_0 \qquad e_3(|\eta_2|_0) = |\eta_3|_0 \qquad e_4(|\eta_3|_0) = -2$$

The first 3 equalities are not definitional and requires some clever choices, but (surprisingly) a variation of the last one holds by `refl` in `Cubical Agda`!

New Brunerie numbers

$\eta_1 : \mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^2$
η_1 (inl x) = north
η_1 (inr y) = north
η_1 (push (x , y) i) = (σ y • σ x) i

$\eta_2 : \mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^1 * \mathbb{S}^1$
η_2 (inl x) = inr (- x)
η_2 (inr y) = inr y
η_2 (push (x , y) i) =
  (push (y - x , - x) $^{-1}$ • push (y - x , y)) i

$\eta_3 : \mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^3$
η_3 (inl x) = north
η_3 (inr y) = north
η_3 (push (x , y) i) =
  (σ ($x \smile_1 y$) $^{-1}$ • σ ($x \smile_1 y$) $^{-1}$) i

$\eta_3' : \mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^3$
η_3' (inl x) = north
η_3' (inr y) = north
η_3' (push (x , y) i) =
  (σ ($x \smile_1 y$) • σ ($x \smile_1 y$)) i

New Brunerie numbers

$\eta_1 : \mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^2$
$\eta_1\ (\text{inl}\ x) = \text{north}$
$\eta_1\ (\text{inr}\ y) = \text{north}$
$\eta_1\ (\text{push}\ (x\ ,\ y)\ i) = (\sigma\ y \cdot \sigma\ x)\ i$

$\eta_2 : \mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^1 * \mathbb{S}^1$
$\eta_2\ (\text{inl}\ x) = \text{inr}\ (-\ x)$
$\eta_2\ (\text{inr}\ y) = \text{inr}\ y$
$\eta_2\ (\text{push}\ (x\ ,\ y)\ i) =$
$\quad (\text{push}\ (y - x\ ,\ -\ x)^{-1} \cdot \text{push}\ (y - x\ ,\ y))\ i$

$\eta_3 : \mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^3$
$\eta_3\ (\text{inl}\ x) = \text{north}$
$\eta_3\ (\text{inr}\ y) = \text{north}$
$\eta_3\ (\text{push}\ (x\ ,\ y)\ i) =$
$\quad (\sigma\ (x \smile_1 y)^{-1} \cdot \sigma\ (x \smile_1 y)^{-1})\ i$

$\eta_3\text{'} : \mathbb{S}^1 * \mathbb{S}^1 \longrightarrow \mathbb{S}^3$
$\eta_3\text{'}\ (\text{inl}\ x) = \text{north}$
$\eta_3\text{'}\ (\text{inr}\ y) = \text{north}$
$\eta_3\text{'}\ (\text{push}\ (x\ ,\ y)\ i) =$
$\quad (\sigma\ (x \smile_1 y) \cdot \sigma\ (x \smile_1 y))\ i$

This gives a sequence of new Brunerie numbers:

$$\beta_1 := e_4(e_3(e_2(|\eta_1|_0))) \qquad \beta_2 := e_4(e_3(|\eta_2|_0)) \qquad \beta_3 := e_4(|\eta_3|_0) \qquad \beta' := e_4(|\eta_3'|_0)$$

File Edit Options Buffers Tools Agda Help

Save Undo

```
-- We also have a much more direct proof in Cubical.Homotopy.Group.Pi4S3.DirectProof,
-- not relying on any of the more advanced constructions in chapters
-- 4-6 in Brunerie's thesis (but still using chapters 1-3 for the
-- construction). For details see the header of that file.

π₄S³≃ℤ/2ℤ-direct : GroupEquiv (π 4 𝕊³) (ℤGroup/ 2)
π₄S³≃ℤ/2ℤ-direct = DirectProof.BrunerieGroupEquiv

-- This direct proof allows us to define a much simplified version of
-- the Brunerie number:
β' : ℤ
β' = fst DirectProof.computer η₃'

-- This number computes definitionally to -2 in a few seconds!
β'≡-2 : β' ≡ -2
β'≡-2 = refl

-- Combining all of this gives us the desired equivalence of groups by
-- computation as conjectured in Brunerie's thesis:
π₄S³≃ℤ/2ℤ-computation : GroupEquiv (π 4 𝕊³) (ℤGroup/ 2)
π₄S³≃ℤ/2ℤ-computation = DirectProof.BrunerieGroupEquiv''
```

U:--- **Summary.agda** Bot (112,0) Git:inducedstruct (Agda:Checked +2)

U:%*- *All Done* All (1,0) (AgdaInfo)

The three formalized proofs

We have three fully formalized synthetic proofs that $\pi_4(\mathbb{S}^3) \simeq \mathbb{Z}/2\mathbb{Z}$:

1. Streamlined and complete proof following Guillaume's thesis (17000 LOC + 8000 from library)
2. Axel's new direct elementary proof which avoids part 2 of the thesis completely (600 LOC)
3. The new computational proof by normalizing β' (400 LOC)

Common part to all proofs (Brunerie Chapters 1-3): 9000 LOC

The first two proofs are expressable in Book HoTT, while the third crucially relies on normalization of terms involving univalence and HITs (so expressable in cubical systems, and maybe H.O.T.T.)

Outline

1. Introduction: cubical methods in HoTT
2. Proofs by computation in synthetic homotopy theory
3. Proofs by computation in synthetic cohomology theory
4. Computational challenges and ongoing work

Synthetic cohomology theory

In HoTT we can define cohomology as:[5]

$$H^n(X, G) = \|X \to K(G, n)\|_0$$

In *Synthetic Integral Cohomology in Cubical Agda* (Brunerie-Ljungström-M., CSL'22) we equip $H^n(X, \mathbb{Z})$ with a very concrete group structure that computes quite well

[5] Buchholtz, Brunerie, Cavallo, Favonia, Finster, Licata, Shulman, van Doorn, ...

Synthetic cohomology theory

In HoTT we can define cohomology as:[5]

$$H^n(X, G) = \|X \to K(G, n)\|_0$$

In *Synthetic Integral Cohomology in Cubical Agda* (Brunerie-Ljungström-M., CSL'22) we equip $H^n(X, \mathbb{Z})$ with a very concrete group structure that computes quite well

We also compute cohomology groups for many classical spaces: spheres, torus, Klein bottle, wedge sums, real and complex projective planes

Many of these proofs are direct by analyzing function spaces, but some require more elaborate classical techniques (Eilenberg-Steenrod axioms, Mayer-Vietoris sequence)

[5] Buchholtz, Brunerie, Cavallo, Favonia, Finster, Licata, Shulman, van Doorn, ...

Side remark: relationship to homotopy groups of spheres

Integral cohomology gives a nice map $\pi_n(\mathbb{S}^n) \longrightarrow \mathbb{Z}$. Note the similarity in:

$$\pi_n(\mathbb{S}^n) = \|\mathbb{S}^n \longrightarrow_\star \mathbb{S}^n\|_0$$

$$H^n(\mathbb{S}^n, \mathbb{Z}) = \|\mathbb{S}^n \longrightarrow \|\mathbb{S}^n\|_n\|_0$$

This is used in the new Brunerie number computation: it is quite straightforward to prove that $H^3(\mathbb{S}^3, \mathbb{Z}) \simeq \mathbb{Z}$ and the maps have better computational behavior than the ones in $\pi_3(\mathbb{S}^3) \simeq \mathbb{Z}$ obtained by iterated Freudenthal suspension theorem

Computations in proofs of cohomology groups

Proofs by computation also pop up in synthetic cohomology theory:

Computations in proofs of cohomology groups

Proofs by computation also pop up in synthetic cohomology theory:

- Base cases when verifying the group laws for $H^n(X, \mathbb{Z})$ involve path algebra in loop spaces over the spheres which can typically be reduced to integer computations

Computations in proofs of cohomology groups

Proofs by computation also pop up in synthetic cohomology theory:

- Base cases when verifying the group laws for $H^n(X, \mathbb{Z})$ involve path algebra in loop spaces over the spheres which can typically be reduced to integer computations
- When showing that $H^n(X, G)$ or $\pi_n(X)$ is generated by a particular element e we can use that the group is equivalent to some nice group G (e.g. $\mathbb{Z}$) and check that e is mapped to a generator of G (e.g. ± 1))

Computations in proofs of cohomology groups

Proofs by computation also pop up in synthetic cohomology theory:

- Base cases when verifying the group laws for $H^n(X, \mathbb{Z})$ involve path algebra in loop spaces over the spheres which can typically be reduced to integer computations
- When showing that $H^n(X, G)$ or $\pi_n(X)$ is generated by a particular element e we can use that the group is equivalent to some nice group G (e.g. $\mathbb{Z}$) and check that e is mapped to a generator of G (e.g. ± 1))
- Various computations involving the group operations

Some of these are fast, some are slow, and some do not terminate in a reasonable amount of time (minutes on a normal laptop)

Cup product and cohomology ring

Cohomology allows us to distinguish many spaces, but it is sometimes a bit too coarse. We can equip cohomology groups also with a graded multiplication operations[6]

$$\smile : H^n(X, G_1) \to H^m(X, G_2) \to H^{n+m}(X, G_1 \otimes G_2)$$

This can be organized into a graded commutative ring $H^*(X, R)$ for commutative ring R

[6] The construction relies on working on the RHS of $(A \wedge B \to_\star C) \simeq (A \to_\star (B \to_\star C))$ which we used in the Brunerie-Ljungström-M. 2022 paper. The same trick has recently been used to work with cup products in [Wärn, 2023] and [Buchholtz-Christensen-Jaz Myers-Rijke, 2024]

Cup product and cohomology ring

Cohomology allows us to distinguish many spaces, but it is sometimes a bit too coarse. We can equip cohomology groups also with a graded multiplication operations[6]

$$\smile : H^n(X, G_1) \to H^m(X, G_2) \to H^{n+m}(X, G_1 \otimes G_2)$$

This can be organized into a graded commutative ring $H^*(X, R)$ for commutative ring R

These rings are often equivalent to quotients of multivariate polynomial rings and we computed some of these in *Computing Cohomology Rings in Cubical Agda* (Lamiaux-Ljungström-M., CPP'23)

[6]The construction relies on working on the RHS of $(A \wedge B \to_\star C) \simeq (A \to_\star (B \to_\star C))$ which we used in the Brunerie-Ljungström-M. 2022 paper. The same trick has recently been used to work with cup products in [Wärn, 2023] and [Buchholtz-Christensen-Jaz Myers-Rijke, 2024]

Cup product and cohomology ring

Cohomology allows us to distinguish many spaces, but it is sometimes a bit too coarse. We can equip cohomology groups also with a graded multiplication operations[6]

$$\smile : H^n(X, G_1) \to H^m(X, G_2) \to H^{n+m}(X, G_1 \otimes G_2)$$

This can be organized into a graded commutative ring $H^*(X, R)$ for commutative ring R

These rings are often equivalent to quotients of multivariate polynomial rings and we computed some of these in *Computing Cohomology Rings in Cubical Agda* (Lamiaux-Ljungström-M., CPP'23)

Application: $\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1$ has the same cohomology groups as $\mathbb{T}^2$, but they are not equivalent as the cohomology rings differ ($\smile$ trivial for $\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1$ but not $\mathbb{T}^2$)

[6] The construction relies on working on the RHS of $(A \wedge B \to_\star C) \simeq (A \to_\star (B \to_\star C))$ which we used in the Brunerie-Ljungström-M. 2022 paper. The same trick has recently been used to work with cup products in [Wärn, 2023] and [Buchholtz-Christensen-Jaz Myers-Rijke, 2024]

Further computations with cohomology rings

For a more ambitious computation involving $\smile$ consider Chapter 6 of Brunerie's PhD thesis. This chapter is devoted to proving that the generator $e : H^2(\mathbb{C}P^2)$ when multiplied with itself yields a generator of $H^4(\mathbb{C}P^2)$

Further computations with cohomology rings

For a more ambitious computation involving $\smile$ consider Chapter 6 of Brunerie's PhD thesis. This chapter is devoted to proving that the generator $e : H^2(\mathbb{C}P^2)$ when multiplied with itself yields a generator of $H^4(\mathbb{C}P^2)$

Let $g : \mathbb{Z} \to \mathbb{Z}$ be the map given by composing:

$$\mathbb{Z} \xrightarrow{\cong} H^2(\mathbb{C}P^2) \xrightarrow{\lambda\, x \to x \,\smile\, x} H^4(\mathbb{C}P^2) \xrightarrow{\cong} \mathbb{Z}$$

Further computations with cohomology rings

For a more ambitious computation involving $\smile$ consider Chapter 6 of Brunerie's PhD thesis. This chapter is devoted to proving that the generator $e : H^2(\mathbb{C}P^2)$ when multiplied with itself yields a generator of $H^4(\mathbb{C}P^2)$

Let $g : \mathbb{Z} \to \mathbb{Z}$ be the map given by composing:

$$\mathbb{Z} \xrightarrow{\cong} H^2(\mathbb{C}P^2) \xrightarrow{\lambda\, x \to x \smile x} H^4(\mathbb{C}P^2) \xrightarrow{\cong} \mathbb{Z}$$

The number $g(1)$ should reduce to ± 1 for $e \smile e$ to generate $H^4(\mathbb{C}P^2)$ and by evaluating it in `Cubical Agda` we should be able to reduce the whole chapter to a single computation...

Further computations with cohomology rings

For a more ambitious computation involving $\smile$ consider Chapter 6 of Brunerie's PhD thesis. This chapter is devoted to proving that the generator $e : H^2(\mathbb{C}P^2)$ when multiplied with itself yields a generator of $H^4(\mathbb{C}P^2)$

Let $g : \mathbb{Z} \to \mathbb{Z}$ be the map given by composing:

$$\mathbb{Z} \xrightarrow{\cong} H^2(\mathbb{C}P^2) \xrightarrow{\lambda\, x \to x \,\smile\, x} H^4(\mathbb{C}P^2) \xrightarrow{\cong} \mathbb{Z}$$

The number $g(1)$ should reduce to ± 1 for $e \smile e$ to generate $H^4(\mathbb{C}P^2)$ and by evaluating it in `Cubical Agda` we should be able to reduce the whole chapter to a single computation... However, `Cubical Agda` is currently stuck on computing $g(1)$

So this is yet another "Brunerie number"

Computations with cohomology rings

In *Computing Cohomology Rings in Cubical Agda* we have some more examples where it would be nice if things computed faster for characterizing $H^*(X, R)$ as quotients of polynomial rings

Computations with cohomology rings

In *Computing Cohomology Rings in Cubical Agda* we have some more examples where it would be nice if things computed faster for characterizing $H^*(X, R)$ as quotients of polynomial rings

For example, to show that $H^*(\mathbb{K}, \mathbb{Z}) \cong \mathbb{Z}[X, Y]/(X^2, XY, 2Y, Y^2)$ some computations are involved to show that the map $f : \mathbb{Z}[X, Y] \to H^*(\mathbb{K}, \mathbb{Z})$ is zero on the generators of $(X^2, XY, 2Y, Y^2)$

Computations with cohomology rings

In *Computing Cohomology Rings in Cubical Agda* we have some more examples where it would be nice if things computed faster for characterizing $H^*(X, R)$ as quotients of polynomial rings

For example, to show that $H^*(\mathbb{K}, \mathbb{Z}) \cong \mathbb{Z}[X, Y]/(X^2, XY, 2Y, Y^2)$ some computations are involved to show that the map $f : \mathbb{Z}[X, Y] \to H^*(\mathbb{K}, \mathbb{Z})$ is zero on the generators of $(X^2, XY, 2Y, Y^2)$

This gives even more examples of computations that are fast, slow, and some that don't terminate in a reasonable amount of time

Cohomology benchmarks

Axel and I have an extended version of the above papers called *Computational Synthetic Cohomology Theory in Homotopy Type Theory* (`https://arxiv.org/abs/2401.16336`) where we generalize to $H^n(X, G)$ and $H^*(X, R)$, and compute many benchmarks:

n	X	G	H	h	Results
1	$\mathbb{S}^1$	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$g^{(1)}$ $\quad g^{(1)} +_h g^{(1)}$ $\quad -_h g^{(1)}$ $g^{(1)}$ $\quad g^{(1)} +_h g^{(1)}$ $\quad -_h g^{(1)}$	✓ ✓ ✓ ✓ ✓ ✓
2	$\mathbb{S}^2$	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$g^{(2)}$ $\quad g^{(2)} +_h g^{(2)}$ $\quad -_h g^{(2)}$ $g^{(2)}$ $\quad g^{(2)} +_h g^{(2)}$ $\quad -_h g^{(2)}$	✓ ✗ ✗ ✓ ✗ ✗
3	$\mathbb{S}^3$	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$g^{(3)}$ $\quad g^{(3)} +_h g^{(3)}$ $\quad -_h g^{(3)}$ $g^{(3)}$ $\quad g^{(3)} +_h g^{(3)}$ $\quad -_h g^{(3)}$	✗ ✗ ✗ ✗ ✗ ✗
1	$\mathbb{T}^2$	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$\mathbb{Z}^2$ $\mathbb{Z}/2\mathbb{Z}^2$	$g_1^{(1)} +_h g_2^{(1)}$ $\quad g_1^{(1)} -_h g_2^{(1)}$ $g_1^{(1)} +_h g_2^{(1)}$ $\quad g_1^{(1)} -_h g_2^{(1)}$	✓ ✓ ✓ ✓
2	$\mathbb{T}^2$	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$g^{(2)}$ $\quad g_1^{(1)} \smile g_2^{(1)}$ $\quad \left(g_1^{(1)} +_h g_1^{(1)}\right) \smile g_2^{(1)}$ $g^{(2)}$ $\quad g_1^{(1)} \smile g_2^{(1)}$ $\quad \left(g_1^{(1)} +_h g_1^{(1)}\right) \smile g_2^{(1)}$	✗ ✗ ✗ ✗ ✗ ✗
1	$\bigvee_{i=2,1,1} \mathbb{S}^i$	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$\mathbb{Z}^2$ $\mathbb{Z}/2\mathbb{Z}^2$	$g_1^{(1)} +_h g_2^{(1)}$ $\quad g_1^{(1)} -_h g_2^{(1)}$ $g_1^{(1)} +_h g_2^{(1)}$ $\quad g_1^{(1)} -_h g_2^{(1)}$	✓ ✓ ✓ ✓
2	$\bigvee_{i=2,1,1} \mathbb{S}^i$	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$g^{(2)}$ $\quad g_1^{(1)} \smile g_2^{(1)}$ $\quad \left(g_1^{(1)} +_h g_1^{(1)}\right) \smile g_2^{(1)}$ $g^{(2)}$ $\quad g_1^{(1)} \smile g_2^{(1)}$ $\quad \left(g_1^{(1)} +_h g_1^{(1)}\right) \smile g_2^{(1)}$	✓ ✓ ✓ ✓ ✓ ✓
1	$\mathbb{R}P^2$	$\mathbb{Z}/2\mathbb{Z}$	$\mathbb{Z}/2\mathbb{Z}$	$g^{(1)}$ $\quad g^{(1)} +_h g^{(1)}$ $\quad -_h g^{(1)}$	✗ ✗ ✗
2	$\mathbb{R}P^2$	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$\mathbb{Z}/2\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$g^{(2)}$ $\quad g^{(2)} +_h g^{(2)}$ $\quad -_h g^{(2)}$ $g^{(2)}$ $\quad g_1^{(1)} \smile g_1^{(1)}$	✗ ✗ ✗ ✗ ✗
1	K^2	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}^2$	$g^{(1)}$ $\quad g^{(1)} +_h g^{(1)}$ $\quad -_h g^{(1)}$ $g_1^{(1)} +_h g_2^{(1)}$ $\quad g_1^{(1)} -_h g_2^{(1)}$	✗ ✗ ✗ ✗ ✗
2	K^2	$\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$\mathbb{Z}/2\mathbb{Z}$ $\mathbb{Z}/2\mathbb{Z}$	$g^{(2)}$ $\quad g_1^{(1)} \smile g_1^{(1)}$ $g^{(2)}$ $\quad g_1^{(1)} \smile g_2^{(1)}$ $\quad \left(g_1^{(1)} +_h g_1^{(1)}\right) \smile g_2^{(1)}$	✗ ✗ ✗ ✗ ✗
1	$\mathbb{R}P^\infty$	$\mathbb{Z}/2\mathbb{Z}$	$\mathbb{Z}/2\mathbb{Z}$	$g^{(1)}$ $\quad g^{(1)} +_h g^{(1)}$ $\quad -_h g^{(1)}$	✗ ✗ ✗

Outline

Computational challenges

We now have lots of computational challenges:

- The new Brunerie numbers: β_1, β_2, and β_3
- The original Brunerie number β (and various reformulations of it)
- Brunerie's $g(1)$ number
- Various cohomology ring computations
- The cohomology benchmarks
- ...

How can we make more things compute fast?

Making cubical (closed term) evaluation faster

Seems very promising, see András Kovács talk on `cctt`

For example, β_3 seems stuck in `Cubical Agda`, but computes instantly in `cctt`

Making cubical (closed term) evaluation faster

Seems very promising, see András Kovács talk on `cctt`

For example, β_3 seems stuck in `Cubical Agda`, but computes instantly in `cctt`

Compilation of cubical programs? (Current research project at Chalmers)

Make (co)homology theories compute better

It is of course natural to consider other (co)homology theories that might compute better

With Axel Ljungström and Loïc Pujet we have revisited Buchholtz-Favonia's synthetic approach to cellular (co)homology in HoTT

Make (co)homology theories compute better

It is of course natural to consider other (co)homology theories that might compute better

With Axel Ljungström and Loïc Pujet we have revisited Buchholtz-Favonia's synthetic approach to cellular (co)homology in HoTT

This gives a nice way to compute (co)homology of CW complexes, see Axel's talk for details

One major goal: formalize a cellular Hurewicz theorem and use in formalization of Barton-Campion's synthetic proof of Serre finiteness theorem for homotopy groups of spheres

Reducing (co)homology computations to linear algebra (WIP)

Another appealing aspect of cellular (co)homology is that computations of cellular (co)homology groups can be reduced to matrix computations

In particular: the boundary maps in cellular chain complexes can be represented by matrices with integer entries

Reducing (co)homology computations to linear algebra (WIP)

Another appealing aspect of cellular (co)homology is that computations of cellular (co)homology groups can be reduced to matrix computations

In particular: the boundary maps in cellular chain complexes can be represented by matrices with integer entries

By some elementary algebra one can prove that (co)homology groups can be computed by diagonalizing these matrices (i.e. putting them in Smith normal form) to get Betti numbers and the torsion part

Reducing (co)homology computations to linear algebra (WIP)

Another appealing aspect of cellular (co)homology is that computations of cellular (co)homology groups can be reduced to matrix computations

In particular: the boundary maps in cellular chain complexes can be represented by matrices with integer entries

By some elementary algebra one can prove that (co)homology groups can be computed by diagonalizing these matrices (i.e. putting them in Smith normal form) to get Betti numbers and the torsion part

This is still work in progress and it will be fun to see how efficient we can make it in Cubical Agda (formalization of Smith Normal Form computation already contributed to `Cubical Agda` by Kang Rongji, but currently quite slow)

Future work: other computational (co)homology theories

Reducing (co)homology computations to linear algebra and matrix manipulations is of course not new and widely used in computer algebra, e.g. to compute simplicial (co)homology

Future work: other computational (co)homology theories

Reducing (co)homology computations to linear algebra and matrix manipulations is of course not new and widely used in computer algebra, e.g. to compute simplicial (co)homology

It would be very interesting to formalize simplicial (co)homology in HoTT (how much can be made synthetic?)

Future work: other computational (co)homology theories

Reducing (co)homology computations to linear algebra and matrix manipulations is of course not new and widely used in computer algebra, e.g. to compute simplicial (co)homology

It would be very interesting to formalize simplicial (co)homology in HoTT (how much can be made synthetic?)

Can we then also compute persistent homology and do formally verified Topological Data Analysis in HoTT?

Thank you for your attention!

Questions?

Bonus slides

Computing with the cohomology ring

To distinguish $\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1$ and $\mathbb{T}^2$ we define a predicate $P : \mathsf{Type} \to \mathsf{Type}$:

$$P(A) := (x\ y : H^1(A)) \to x \smile y \equiv 0_h$$

Computing with the cohomology ring

To distinguish $\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1$ and $\mathbb{T}^2$ we define a predicate P : Type $\to$ Type:

$$P(A) := (x\ y : H^1(A)) \to x \smile y \equiv 0_h$$

We have the isomorphisms:

$$f_1 : H^1(\mathbb{T}^2) \cong \mathbb{Z} \times \mathbb{Z} \qquad g_1 : H^1(\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1) \cong \mathbb{Z} \times \mathbb{Z}$$

$$f_2 : H^2(\mathbb{T}^2) \cong \mathbb{Z} \qquad g_2 : H^2(\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1) \cong \mathbb{Z}$$

Computing with the cohomology ring

To distinguish $\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1$ and $\mathbb{T}^2$ we define a predicate $P : \mathsf{Type} \to \mathsf{Type}$:

$$P(A) := (x\ y : H^1(A)) \to x \smile y \equiv 0_h$$

We have the isomorphisms:

$$f_1 : H^1(\mathbb{T}^2) \cong \mathbb{Z} \times \mathbb{Z} \qquad g_1 : H^1(\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1) \cong \mathbb{Z} \times \mathbb{Z}$$

$$f_2 : H^2(\mathbb{T}^2) \cong \mathbb{Z} \qquad g_2 : H^2(\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1) \cong \mathbb{Z}$$

We will now disprove $P(\mathbb{T}^2)$ and prove $P(\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1)$, which establishes that they are not equivalent

Computing with the cohomology ring

To disprove $P(\mathbb{T}^2)$ we need $x, y : H^1(\mathbb{T}^2)$ such that $x \smile y \not\equiv 0_h$.

Computing with the cohomology ring

To disprove $P(\mathbb{T}^2)$ we need $x, y : H^1(\mathbb{T}^2)$ such that $x \smile y \not\equiv 0_h$. Let

$$x = f_1^{-1}(0, 1) \qquad\qquad y = f_1^{-1}(1, 0)$$

Computing with the cohomology ring

To disprove $P(\mathbb{T}^2)$ we need $x, y : H^1(\mathbb{T}^2)$ such that $x \smile y \not\equiv 0_h$. Let

$$x = f_1^{-1}(0, 1) \qquad\qquad y = f_1^{-1}(1, 0)$$

Now $f_2(x \smile y) \equiv 1$ holds by refl and thus $x \smile y \not\equiv 0_h$

Computing with the cohomology ring

To disprove $P(\mathbb{T}^2)$ we need $x, y : H^1(\mathbb{T}^2)$ such that $x \smile y \not\equiv 0_h$. Let

$$x = f_1^{-1}(0, 1) \qquad\qquad y = f_1^{-1}(1, 0)$$

Now $f_2(x \smile y) \equiv 1$ holds by refl and thus $x \smile y \not\equiv 0_h$

To prove $P(\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1)$ we let $x, y : H^1(\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1)$.

Computing with the cohomology ring

To disprove $P(\mathbb{T}^2)$ we need $x, y : H^1(\mathbb{T}^2)$ such that $x \smile y \not\equiv 0_h$. Let

$$x = f_1^{-1}(0, 1) \qquad\qquad y = f_1^{-1}(1, 0)$$

Now $f_2(x \smile y) \equiv 1$ holds by refl and thus $x \smile y \not\equiv 0_h$

To prove $P(\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1)$ we let $x, y : H^1(\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1)$. We have that $g_2(g_1^{-1}(g_1\ x) \smile g_1^{-1}(g_1\ y)) \equiv 0$, again by refl (modulo truncation elimination).

Computing with the cohomology ring

To disprove $P(\mathbb{T}^2)$ we need $x, y : H^1(\mathbb{T}^2)$ such that $x \smile y \not\equiv 0_h$. Let

$$x = f_1^{-1}(0, 1) \qquad\qquad y = f_1^{-1}(1, 0)$$

Now $f_2(x \smile y) \equiv 1$ holds by refl and thus $x \smile y \not\equiv 0_h$

To prove $P(\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1)$ we let $x, y : H^1(\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1)$. We have that $g_2(g_1^{-1}(g_1\ x) \smile g_1^{-1}(g_1\ y)) \equiv 0$, again by refl (modulo truncation elimination). Thus $g_1^{-1}(g_1\ x) \smile g_1^{-1}(g_1\ y) \equiv x \smile y \equiv 0_h$.

Computing with the cohomology ring

To disprove $P(\mathbb{T}^2)$ we need $x, y : H^1(\mathbb{T}^2)$ such that $x \smile y \not\equiv 0_h$. Let

$$x = f_1^{-1}(0, 1) \qquad\qquad y = f_1^{-1}(1, 0)$$

Now $f_2(x \smile y) \equiv 1$ holds by refl and thus $x \smile y \not\equiv 0_h$

To prove $P(\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1)$ we let $x, y : H^1(\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1)$. We have that $g_2(g_1^{-1}(g_1\ x) \smile g_1^{-1}(g_1\ y)) \equiv 0$, again by refl (modulo truncation elimination). Thus $g_1^{-1}(g_1\ x) \smile g_1^{-1}(g_1\ y) \equiv x \smile y \equiv 0_h$.

So $P(\mathbb{T}^2)$ does not hold while $P(\mathbb{S}^2 \vee \mathbb{S}^1 \vee \mathbb{S}^1)$ does, so these types are not equivalent