Lean in new research

Neil Strickland

January 7, 2020

Using Lean as a tool for new research

- I formalised part of the new paper *Iterated chromatic localisation* (Bellumat and Strickland) in Lean.
- We should think about infrastructure/ecosystem issues that might arise when many people are trying to do this sort of thing, with varying levels of skill and experience.
- Ideally:
  - We should minimise the learning curve if possible.
  - We should have conventions about where and how to store resulting Lean code.
  - There should be support for an immutable reference version of the code plus compatible versions of Lean and Mathlib.
  - There should also be support for a mutable version that can be updated and extended and synchronised with developments in Lean and Mathlib. There should be a standard migration path for code to move into Mathlib where appropriate.
  - There should be conventions and software infrastructure for cross-references between Lean code and LaTeX/PDF. This should allow for the fact that the natural structure of the human-readable development may be quite different from the Lean structure.
  - There should be conventions for trusting and referring to the literature for material that is not yet formalised.

Using Lean as a tool for new research

- I formalised part of the new paper *Iterated chromatic localisation* (Bellumat and Strickland) in Lean.
- We should think about infrastructure/ecosystem issues that might arise when many people are trying to do this sort of thing, with varying levels of skill and experience.
- Ideally:
  - We should minimise the learning curve if possible.
  - We should have conventions about where and how to store resulting Lean code.
  - There should be support for an immutable reference version of the code plus compatible versions of Lean and Mathlib.
  - There should also be support for a mutable version that can be updated and extended and synchronised with developments in Lean and Mathlib. There should be a standard migration path for code to move into Mathlib where appropriate.
  - There should be conventions and software infrastructure for cross-references between Lean code and LaTeX/PDF. This should allow for the fact that the natural structure of the human-readable development may be quite different from the Lean structure.
  - There should be conventions for trusting and referring to the literature for material that is not yet formalised.

Using Lean as a tool for new research

- I formalised part of the new paper *Iterated chromatic localisation* (Bellumat and Strickland) in Lean.
- We should think about infrastructure/ecosystem issues that might arise when many people are trying to do this sort of thing, with varying levels of skill and experience.
- Ideally:
  - We should minimise the learning curve if possible.
  - We should have conventions about where and how to store resulting Lean code.
  - There should be support for an immutable reference version of the code plus compatible versions of Lean and Mathlib.
  - There should also be support for a mutable version that can be updated and extended and synchronised with developments in Lean and Mathlib. There should be a standard migration path for code to move into Mathlib where appropriate.
  - There should be conventions and software infrastructure for cross-references between Lean code and LaTeX/PDF. This should allow for the fact that the natural structure of the human-readable development may be quite different from the Lean structure.
  - There should be conventions for trusting and referring to the literature for material that is not yet formalised.

Using Lean as a tool for new research

- I formalised part of the new paper *Iterated chromatic localisation* (Bellumat and Strickland) in Lean.
- We should think about infrastructure/ecosystem issues that might arise when many people are trying to do this sort of thing, with varying levels of skill and experience.
- Ideally:
  - We should minimise the learning curve if possible.
  - We should have conventions about where and how to store resulting Lean code.
  - There should be support for an immutable reference version of the code plus compatible versions of Lean and Mathlib.
  - There should also be support for a mutable version that can be updated and extended and synchronised with developments in Lean and Mathlib. There should be a standard migration path for code to move into Mathlib where appropriate.
  - There should be conventions and software infrastructure for cross-references between Lean code and LaTeX/PDF. This should allow for the fact that the natural structure of the human-readable development may be quite different from the Lean structure.
  - There should be conventions for trusting and referring to the literature for material that is not yet formalised.

Using Lean as a tool for new research

- I formalised part of the new paper *Iterated chromatic localisation* (Bellumat and Strickland) in Lean.
- We should think about infrastructure/ecosystem issues that might arise when many people are trying to do this sort of thing, with varying levels of skill and experience.
- Ideally:
  - We should minimise the learning curve if possible.
  - We should have conventions about where and how to store resulting Lean code.
  - There should be support for an immutable reference version of the code plus compatible versions of Lean and Mathlib.
  - There should also be support for a mutable version that can be updated and extended and synchronised with developments in Lean and Mathlib. There should be a standard migration path for code to move into Mathlib where appropriate.
  - There should be conventions and software infrastructure for cross-references between Lean code and LaTeX/PDF. This should allow for the fact that the natural structure of the human-readable development may be quite different from the Lean structure.
  - There should be conventions for trusting and referring to the literature for material that is not yet formalised.

Using Lean as a tool for new research

- I formalised part of the new paper *Iterated chromatic localisation* (Bellumat and Strickland) in Lean.
- We should think about infrastructure/ecosystem issues that might arise when many people are trying to do this sort of thing, with varying levels of skill and experience.
- Ideally:
  - We should minimise the learning curve if possible.
  - We should have conventions about where and how to store resulting Lean code.
  - There should be support for an immutable reference version of the code plus compatible versions of Lean and Mathlib.
  - There should also be support for a mutable version that can be updated and extended and synchronised with developments in Lean and Mathlib. There should be a standard migration path for code to move into Mathlib where appropriate.
  - There should be conventions and software infrastructure for cross-references between Lean code and LaTeX/PDF. This should allow for the fact that the natural structure of the human-readable development may be quite different from the Lean structure.
  - There should be conventions for trusting and referring to the literature for material that is not yet formalised.

Using Lean as a tool for new research

- I formalised part of the new paper *Iterated chromatic localisation* (Bellumat and Strickland) in Lean.
- We should think about infrastructure/ecosystem issues that might arise when many people are trying to do this sort of thing, with varying levels of skill and experience.
- Ideally:
  - We should minimise the learning curve if possible.
  - We should have conventions about where and how to store resulting Lean code.
  - There should be support for an immutable reference version of the code plus compatible versions of Lean and Mathlib.
  - There should also be support for a mutable version that can be updated and extended and synchronised with developments in Lean and Mathlib. There should be a standard migration path for code to move into Mathlib where appropriate.
  - There should be conventions and software infrastructure for cross-references between Lean code and LaTeX/PDF. This should allow for the fact that the natural structure of the human-readable development may be quite different from the Lean structure.
  - There should be conventions for trusting and referring to the literature for material that is not yet formalised.

Using Lean as a tool for new research

- I formalised part of the new paper *Iterated chromatic localisation* (Bellumat and Strickland) in Lean.
- We should think about infrastructure/ecosystem issues that might arise when many people are trying to do this sort of thing, with varying levels of skill and experience.
- Ideally:
  - We should minimise the learning curve if possible.
  - We should have conventions about where and how to store resulting Lean code.
  - There should be support for an immutable reference version of the code plus compatible versions of Lean and Mathlib.
  - There should also be support for a mutable version that can be updated and extended and synchronised with developments in Lean and Mathlib. There should be a standard migration path for code to move into Mathlib where appropriate.
  - There should be conventions and software infrastructure for cross-references between Lean code and LaTeX/PDF. This should allow for the fact that the natural structure of the human-readable development may be quite different from the Lean structure.
  - There should be conventions for trusting and referring to the literature for material that is not yet formalised.

Using Lean as a tool for new research

- I formalised part of the new paper *Iterated chromatic localisation* (Bellumat and Strickland) in Lean.
- We should think about infrastructure/ecosystem issues that might arise when many people are trying to do this sort of thing, with varying levels of skill and experience.
- Ideally:
  - We should minimise the learning curve if possible.
  - We should have conventions about where and how to store resulting Lean code.
  - There should be support for an immutable reference version of the code plus compatible versions of Lean and Mathlib.
  - There should also be support for a mutable version that can be updated and extended and synchronised with developments in Lean and Mathlib. There should be a standard migration path for code to move into Mathlib where appropriate.
  - There should be conventions and software infrastructure for cross-references between Lean code and LaTeX/PDF. This should allow for the fact that the natural structure of the human-readable development may be quite different from the Lean structure.
  - There should be conventions for trusting and referring to the literature for material that is not yet formalised.

Using Lean as a tool for new research

- I formalised part of the new paper *Iterated chromatic localisation* (Bellumat and Strickland) in Lean.
- We should think about infrastructure/ecosystem issues that might arise when many people are trying to do this sort of thing, with varying levels of skill and experience.
- Ideally:
  - We should minimise the learning curve if possible.
  - We should have conventions about where and how to store resulting Lean code.
  - There should be support for an immutable reference version of the code plus compatible versions of Lean and Mathlib.
  - There should also be support for a mutable version that can be updated and extended and synchronised with developments in Lean and Mathlib. There should be a standard migration path for code to move into Mathlib where appropriate.
  - There should be conventions and software infrastructure for cross-references between Lean code and LaTeX/PDF. This should allow for the fact that the natural structure of the human-readable development may be quite different from the Lean structure.
  - There should be conventions for trusting and referring to the literature for material that is not yet formalised.

What's in the paper?

(a) General homotopy theory and category theory of finite posets

(b) Properties of some specific finite posets and maps between them.

(c) Theory of stable derivators. (An example of a derivator is the strict 2-functor $P \mapsto \mathrm{Ho}([P, \mathrm{Top}])$ assigning to each finite poset P the homotopy category of P-shaped diagrams of topological spaces. In general, a derivator is a functor from finite posets to categories subject to axioms inspired by this example.)

(d) Some specific results from chromatic homotopy theory: properties of Morava K-theory and associated Bousfield localisations.

Formalisation status:

(a) Fully formalised, but not using Mathlib category theory.

(b) Fully formalised.

(c) Not formalised. Not clear whether one would need a general library for bicategories/2-categories, or whether one could get away with an *ad hoc* approach.

(d) Not formalised, but clearly separated: the point is to provide examples of an axiomatically described situation under (c).

What's in the paper?

(a) General homotopy theory and category theory of finite posets
(b) Properties of some specific finite posets and maps between them.
(c) Theory of stable derivators. (An example of a derivator is the strict 2-functor $P \mapsto \mathrm{Ho}([P, \mathrm{Top}])$ assigning to each finite poset P the homotopy category of P-shaped diagrams of topological spaces. In general, a derivator is a functor from finite posets to categories subject to axioms inspired by this example.)
(d) Some specific results from chromatic homotopy theory: properties of Morava K-theory and associated Bousfield localisations.

Formalisation status:

(a) Fully formalised, but not using Mathlib category theory.
(b) Fully formalised.
(c) Not formalised. Not clear whether one would need a general library for bicategories/2-categories, or whether one could get away with an *ad hoc* approach.
(d) Not formalised, but clearly separated: the point is to provide examples of an axiomatically described situation under (c).

What's in the paper?

(a) General homotopy theory and category theory of finite posets
(b) Properties of some specific finite posets and maps between them.
(c) Theory of stable derivators. (An example of a derivator is the strict 2-functor $P \mapsto \mathrm{Ho}([P, \mathrm{Top}])$ assigning to each finite poset P the homotopy category of P-shaped diagrams of topological spaces. In general, a derivator is a functor from finite posets to categories subject to axioms inspired by this example.)
(d) Some specific results from chromatic homotopy theory: properties of Morava K-theory and associated Bousfield localisations.

Formalisation status:

(a) Fully formalised, but not using Mathlib category theory.
(b) Fully formalised.
(c) Not formalised. Not clear whether one would need a general library for bicategories/2-categories, or whether one could get away with an *ad hoc* approach.
(d) Not formalised, but clearly separated: the point is to provide examples of an axiomatically described situation under (c).

What's in the paper?

(a) General homotopy theory and category theory of finite posets

(b) Properties of some specific finite posets and maps between them.

(c) Theory of stable derivators. (An example of a derivator is the strict 2-functor $P \mapsto \mathrm{Ho}([P, \mathrm{Top}])$ assigning to each finite poset P the homotopy category of P-shaped diagrams of topological spaces. In general, a derivator is a functor from finite posets to categories subject to axioms inspired by this example.)

(d) Some specific results from chromatic homotopy theory: properties of Morava K-theory and associated Bousfield localisations.

Formalisation status:

(a) Fully formalised, but not using Mathlib category theory.

(b) Fully formalised.

(c) Not formalised. Not clear whether one would need a general library for bicategories/2-categories, or whether one could get away with an *ad hoc* approach.

(d) Not formalised, but clearly separated: the point is to provide examples of an axiomatically described situation under (c).

What's in the paper?

(a) General homotopy theory and category theory of finite posets
(b) Properties of some specific finite posets and maps between them.
(c) Theory of stable derivators. (An example of a derivator is the strict 2-functor $P \mapsto \mathrm{Ho}([P, \mathrm{Top}])$ assigning to each finite poset P the homotopy category of P-shaped diagrams of topological spaces. In general, a derivator is a functor from finite posets to categories subject to axioms inspired by this example.)
(d) Some specific results from chromatic homotopy theory: properties of Morava K-theory and associated Bousfield localisations.

Formalisation status:

(a) Fully formalised, but not using Mathlib category theory.
(b) Fully formalised.
(c) Not formalised. Not clear whether one would need a general library for bicategories/2-categories, or whether one could get away with an *ad hoc* approach.
(d) Not formalised, but clearly separated: the point is to provide examples of an axiomatically described situation under (c).

What's in the paper?

(a) General homotopy theory and category theory of finite posets
(b) Properties of some specific finite posets and maps between them.
(c) Theory of stable derivators. (An example of a derivator is the strict 2-functor $P \mapsto \mathrm{Ho}([P, \mathrm{Top}])$ assigning to each finite poset P the homotopy category of P-shaped diagrams of topological spaces. In general, a derivator is a functor from finite posets to categories subject to axioms inspired by this example.)
(d) Some specific results from chromatic homotopy theory: properties of Morava K-theory and associated Bousfield localisations.

Formalisation status:

(a) Fully formalised, but not using Mathlib category theory.
(b) Fully formalised.
(c) Not formalised. Not clear whether one would need a general library for bicategories/2-categories, or whether one could get away with an *ad hoc* approach.
(d) Not formalised, but clearly separated: the point is to provide examples of an axiomatically described situation under (c).

What's in the paper?

(a) General homotopy theory and category theory of finite posets

(b) Properties of some specific finite posets and maps between them.

(c) Theory of stable derivators. (An example of a derivator is the strict 2-functor $P \mapsto \mathrm{Ho}([P, \mathrm{Top}])$ assigning to each finite poset P the homotopy category of P-shaped diagrams of topological spaces. In general, a derivator is a functor from finite posets to categories subject to axioms inspired by this example.)

(d) Some specific results from chromatic homotopy theory: properties of Morava K-theory and associated Bousfield localisations.

Formalisation status:

(a) Fully formalised, but not using Mathlib category theory.

(b) Fully formalised.

(c) Not formalised. Not clear whether one would need a general library for bicategories/2-categories, or whether one could get away with an *ad hoc* approach.

(d) Not formalised, but clearly separated: the point is to provide examples of an axiomatically described situation under (c).

What's in the paper?

(a) General homotopy theory and category theory of finite posets

(b) Properties of some specific finite posets and maps between them.

(c) Theory of stable derivators. (An example of a derivator is the strict 2-functor $P \mapsto \mathrm{Ho}([P, \mathrm{Top}])$ assigning to each finite poset P the homotopy category of P-shaped diagrams of topological spaces. In general, a derivator is a functor from finite posets to categories subject to axioms inspired by this example.)

(d) Some specific results from chromatic homotopy theory: properties of Morava K-theory and associated Bousfield localisations.

Formalisation status:

(a) Fully formalised, but not using Mathlib category theory.

(b) Fully formalised.

(c) Not formalised. Not clear whether one would need a general library for bicategories/2-categories, or whether one could get away with an *ad hoc* approach.

(d) Not formalised, but clearly separated: the point is to provide examples of an axiomatically described situation under (c).

What's in the paper?

(a) General homotopy theory and category theory of finite posets
(b) Properties of some specific finite posets and maps between them.
(c) Theory of stable derivators. (An example of a derivator is the strict 2-functor $P \mapsto \mathrm{Ho}([P, \mathrm{Top}])$ assigning to each finite poset P the homotopy category of P-shaped diagrams of topological spaces. In general, a derivator is a functor from finite posets to categories subject to axioms inspired by this example.)
(d) Some specific results from chromatic homotopy theory: properties of Morava K-theory and associated Bousfield localisations.

Formalisation status:

(a) Fully formalised, but not using Mathlib category theory.
(b) Fully formalised.
(c) Not formalised. Not clear whether one would need a general library for bicategories/2-categories, or whether one could get away with an *ad hoc* approach.
(d) Not formalised, but clearly separated: the point is to provide examples of an axiomatically described situation under (c).

Some mathematical details

- $\mathbb{P} = \{$ subsets of $\{0, \dots, n-1\}\}$
- $\mathbb{Q} = \{$ subsets of $\mathbb{P}$ that are closed upwards $\}$
- $A \angle B$ means $a \leq b$ for all $a \in A$ and $b \in B$.
- $\mu\colon \mathbb{Q} \times \mathbb{Q} \to \mathbb{Q}$ is given by

  $$\mu(U, V) = A * B = \{A \cup B \mid A \in U,\ B \in V,\ A \angle B\}.$$

  Together with the union operation, this makes $\mathbb{Q}$ into a noncommutative semiring.
- A finite poset P is *strongly contractible* if there are monotone maps $f_i\colon P \to P$ with $f_0 = \mathrm{id}$ and $f_0 \leq f_1 \geq f_2 \leq f_3 \cdots f_n$ and f_n constant.
- A map $f\colon P \to Q$ is homotopy final if the poset $f/q = \{p \mid f(p) \leq q\}$ is strongly contractible for all q.
- If $U \boxtimes V = \{(A, B) \mid A \in U,\ B \in V,\ A \angle B\}$ then the union map $U \boxtimes V \to U * V$ is both homotopy final and homotopy cofinal.

Some mathematical details

- $\mathbb{P} = \{$ subsets of $\{0, \dots, n-1\}\}$
- $\mathbb{Q} = \{$ subsets of $\mathbb{P}$ that are closed upwards $\}$
- $A \angle B$ means $a \leq b$ for all $a \in A$ and $b \in B$.
- $\mu \colon \mathbb{Q} \times \mathbb{Q} \to \mathbb{Q}$ is given by

  $$\mu(U, V) = A * B = \{A \cup B \mid A \in U,\ B \in V,\ A \angle B\}.$$

  Together with the union operation, this makes $\mathbb{Q}$ into a noncommutative semiring.
- A finite poset P is *strongly contractible* if there are monotone maps $f_i \colon P \to P$ with $f_0 = \mathrm{id}$ and $f_0 \leq f_1 \geq f_2 \leq f_3 \cdots f_n$ and f_n constant.
- A map $f \colon P \to Q$ is homotopy final if the poset $f/q = \{p \mid f(p) \leq q\}$ is strongly contractible for all q.
- If $U \boxtimes V = \{(A, B) \mid A \in U,\ B \in V,\ A \angle B\}$ then the union map $U \boxtimes V \to U * V$ is both homotopy final and homotopy cofinal.

Some mathematical details

- $\mathbb{P} = \{$ subsets of $\{0, \ldots, n-1\}\}$
- $\mathbb{Q} = \{$ subsets of $\mathbb{P}$ that are closed upwards $\}$
- $A \angle B$ means $a \leq b$ for all $a \in A$ and $b \in B$.
- $\mu \colon \mathbb{Q} \times \mathbb{Q} \to \mathbb{Q}$ is given by

$$\mu(U, V) = A * B = \{A \cup B \mid A \in U,\ B \in V,\ A \angle B\}.$$

  Together with the union operation, this makes $\mathbb{Q}$ into a noncommutative semiring.
- A finite poset P is *strongly contractible* if there are monotone maps $f_i \colon P \to P$ with $f_0 = \mathrm{id}$ and $f_0 \leq f_1 \geq f_2 \leq f_3 \cdots f_n$ and f_n constant.
- A map $f \colon P \to Q$ is homotopy final if the poset $f/q = \{p \mid f(p) \leq q\}$ is strongly contractible for all q.
- If $U \boxtimes V = \{(A, B) \mid A \in U,\ B \in V,\ A \angle B\}$ then the union map $U \boxtimes V \to U * V$ is both homotopy final and homotopy cofinal.

Some mathematical details

- $\mathbb{P} = \{$ subsets of $\{0, \ldots, n-1\}\}$
- $\mathbb{Q} = \{$ subsets of $\mathbb{P}$ that are closed upwards $\}$
- $A \angle B$ means $a \leq b$ for all $a \in A$ and $b \in B$.
- $\mu \colon \mathbb{Q} \times \mathbb{Q} \to \mathbb{Q}$ is given by

$$\mu(U, V) = A * B = \{A \cup B \mid A \in U,\ B \in V,\ A \angle B\}.$$

  Together with the union operation, this makes $\mathbb{Q}$ into a noncommutative semiring.
- A finite poset P is *strongly contractible* if there are monotone maps $f_i \colon P \to P$ with $f_0 = \mathrm{id}$ and $f_0 \leq f_1 \geq f_2 \leq f_3 \cdots f_n$ and f_n constant.
- A map $f \colon P \to Q$ is homotopy final if the poset $f/q = \{p \mid f(p) \leq q\}$ is strongly contractible for all q.
- If $U \boxtimes V = \{(A, B) \mid A \in U,\ B \in V,\ A \angle B\}$ then the union map $U \boxtimes V \to U * V$ is both homotopy final and homotopy cofinal.

Some mathematical details

- $\mathbb{P} = \{$ subsets of $\{0, \dots, n-1\}\}$
- $\mathbb{Q} = \{$ subsets of $\mathbb{P}$ that are closed upwards $\}$
- $A \angle B$ means $a \leq b$ for all $a \in A$ and $b \in B$.
- $\mu \colon \mathbb{Q} \times \mathbb{Q} \to \mathbb{Q}$ is given by

  $$\mu(U, V) = A * B = \{A \cup B \mid A \in U,\ B \in V,\ A \angle B\}.$$

  Together with the union operation, this makes $\mathbb{Q}$ into a noncommutative semiring.
- A finite poset P is *strongly contractible* if there are monotone maps $f_i \colon P \to P$ with $f_0 = \mathrm{id}$ and $f_0 \leq f_1 \geq f_2 \leq f_3 \cdots f_n$ and f_n constant.
- A map $f \colon P \to Q$ is homotopy final if the poset $f/q = \{p \mid f(p) \leq q\}$ is strongly contractible for all q.
- If $U \boxtimes V = \{(A, B) \mid A \in U,\ B \in V,\ A \angle B\}$ then the union map $U \boxtimes V \to U * V$ is both homotopy final and homotopy cofinal.

Some mathematical details

- $\mathbb{P} = \{$ subsets of $\{0, \ldots, n-1\}\}$
- $\mathbb{Q} = \{$ subsets of $\mathbb{P}$ that are closed upwards $\}$
- $A \angle B$ means $a \leq b$ for all $a \in A$ and $b \in B$.
- $\mu\colon \mathbb{Q} \times \mathbb{Q} \to \mathbb{Q}$ is given by

  $$\mu(U, V) = A * B = \{A \cup B \mid A \in U,\ B \in V,\ A \angle B\}.$$

  Together with the union operation, this makes $\mathbb{Q}$ into a noncommutative semiring.
- A finite poset P is *strongly contractible* if there are monotone maps $f_i\colon P \to P$ with $f_0 = \mathrm{id}$ and $f_0 \leq f_1 \geq f_2 \leq f_3 \cdots f_n$ and f_n constant.
- A map $f\colon P \to Q$ is homotopy final if the poset $f/q = \{p \mid f(p) \leq q\}$ is strongly contractible for all q.
- If $U \boxtimes V = \{(A, B) \mid A \in U,\ B \in V,\ A \angle B\}$ then the union map $U \boxtimes V \to U * V$ is both homotopy final and homotopy cofinal.

Some mathematical details

- $\mathbb{P} = \{$ subsets of $\{0, \ldots, n-1\}\}$
- $\mathbb{Q} = \{$ subsets of $\mathbb{P}$ that are closed upwards $\}$
- $A \angle B$ means $a \leq b$ for all $a \in A$ and $b \in B$.
- $\mu \colon \mathbb{Q} \times \mathbb{Q} \to \mathbb{Q}$ is given by

  $$\mu(U, V) = A * B = \{A \cup B \mid A \in U,\ B \in V,\ A \angle B\}.$$

  Together with the union operation, this makes $\mathbb{Q}$ into a noncommutative semiring.
- A finite poset P is *strongly contractible* if there are monotone maps $f_i \colon P \to P$ with $f_0 = \mathrm{id}$ and $f_0 \leq f_1 \geq f_2 \leq f_3 \cdots f_n$ and f_n constant.
- A map $f \colon P \to Q$ is homotopy final if the poset $f/q = \{p \mid f(p) \leq q\}$ is strongly contractible for all q.
- If $U \boxtimes V = \{(A, B) \mid A \in U,\ B \in V,\ A \angle B\}$ then the union map $U \boxtimes V \to U * V$ is both homotopy final and homotopy cofinal.

Some mathematical details

- $\mathbb{P} = \{$ subsets of $\{0, \dots, n-1\}\}$
- $\mathbb{Q} = \{$ subsets of $\mathbb{P}$ that are closed upwards $\}$
- $A \angle B$ means $a \leq b$ for all $a \in A$ and $b \in B$.
- $\mu \colon \mathbb{Q} \times \mathbb{Q} \to \mathbb{Q}$ is given by

  $$\mu(U, V) = A * B = \{A \cup B \mid A \in U,\ B \in V,\ A \angle B\}.$$

  Together with the union operation, this makes $\mathbb{Q}$ into a noncommutative semiring.
- A finite poset P is *strongly contractible* if there are monotone maps $f_i \colon P \to P$ with $f_0 = \mathrm{id}$ and $f_0 \leq f_1 \geq f_2 \leq f_3 \cdots f_n$ and f_n constant.
- A map $f \colon P \to Q$ is homotopy final if the poset $f/q = \{p \mid f(p) \leq q\}$ is strongly contractible for all q.
- If $U \boxtimes V = \{(A, B) \mid A \in U,\ B \in V,\ A \angle B\}$ then the union map $U \boxtimes V \to U * V$ is both homotopy final and homotopy cofinal.

What's in the Lean code?

(a) We have about 240 lines of code about `fin n`. Roughly half of this is about the largest or smallest element of a nonempty subset of `fin n`. The rest is just tiny lemmas.

(b) If α is a finite, totally ordered set of size n then there is a unique order-preserving bijection from α to `fin n`. We have about 620 lines related to this.

(c) We have about 500 lines setting up the basic category theory and strong homotopy theory of finite posets.

(d) We have about 590 lines about the operation of subdivision on finite posets, and its interaction with strong homotopy (which takes most of the work).

(e) We have about 170 lines about the lattice of upper sets in a finite poset.

(f) We have about 2100 lines about the specific finite posets and maps that are relevant for iterated localisation.

For comparison: the LaTeX file is 3500 lines, and the PDF paper is 42 pages. Only about 7 PDF pages are fully formalised.

What's in the Lean code?

(a) We have about 240 lines of code about `fin n`. Roughly half of this is about the largest or smallest element of a nonempty subset of `fin n`. The rest is just tiny lemmas.

(b) If α is a finite, totally ordered set of size n then there is a unique order-preserving bijection from α to `fin n`. We have about 620 lines related to this.

(c) We have about 500 lines setting up the basic category theory and strong homotopy theory of finite posets.

(d) We have about 590 lines about the operation of subdivision on finite posets, and its interaction with strong homotopy (which takes most of the work).

(e) We have about 170 lines about the lattice of upper sets in a finite poset.

(f) We have about 2100 lines about the specific finite posets and maps that are relevant for iterated localisation.

For comparison: the LaTeX file is 3500 lines, and the PDF paper is 42 pages. Only about 7 PDF pages are fully formalised.

What's in the Lean code?

(a) We have about 240 lines of code about `fin n`. Roughly half of this is about the largest or smallest element of a nonempty subset of `fin n`. The rest is just tiny lemmas.

(b) If α is a finite, totally ordered set of size n then there is a unique order-preserving bijection from α to `fin n`. We have about 620 lines related to this.

(c) We have about 500 lines setting up the basic category theory and strong homotopy theory of finite posets.

(d) We have about 590 lines about the operation of subdivision on finite posets, and its interaction with strong homotopy (which takes most of the work).

(e) We have about 170 lines about the lattice of upper sets in a finite poset.

(f) We have about 2100 lines about the specific finite posets and maps that are relevant for iterated localisation.

For comparison: the LaTeX file is 3500 lines, and the PDF paper is 42 pages. Only about 7 PDF pages are fully formalised.

What's in the Lean code?

(a) We have about 240 lines of code about `fin n`. Roughly half of this is about the largest or smallest element of a nonempty subset of `fin n`. The rest is just tiny lemmas.

(b) If α is a finite, totally ordered set of size n then there is a unique order-preserving bijection from α to `fin n`. We have about 620 lines related to this.

(c) We have about 500 lines setting up the basic category theory and strong homotopy theory of finite posets.

(d) We have about 590 lines about the operation of subdivision on finite posets, and its interaction with strong homotopy (which takes most of the work).

(e) We have about 170 lines about the lattice of upper sets in a finite poset.

(f) We have about 2100 lines about the specific finite posets and maps that are relevant for iterated localisation.

For comparison: the LaTeX file is 3500 lines, and the PDF paper is 42 pages. Only about 7 PDF pages are fully formalised.

What's in the Lean code?

(a) We have about 240 lines of code about `fin n`. Roughly half of this is about the largest or smallest element of a nonempty subset of `fin n`. The rest is just tiny lemmas.

(b) If α is a finite, totally ordered set of size n then there is a unique order-preserving bijection from α to `fin n`. We have about 620 lines related to this.

(c) We have about 500 lines setting up the basic category theory and strong homotopy theory of finite posets.

(d) We have about 590 lines about the operation of subdivision on finite posets, and its interaction with strong homotopy (which takes most of the work).

(e) We have about 170 lines about the lattice of upper sets in a finite poset.

(f) We have about 2100 lines about the specific finite posets and maps that are relevant for iterated localisation.

For comparison: the LaTeX file is 3500 lines, and the PDF paper is 42 pages. Only about 7 PDF pages are fully formalised.

What's in the Lean code?

(a) We have about 240 lines of code about `fin n`. Roughly half of this is about the largest or smallest element of a nonempty subset of `fin n`. The rest is just tiny lemmas.

(b) If α is a finite, totally ordered set of size n then there is a unique order-preserving bijection from α to `fin n`. We have about 620 lines related to this.

(c) We have about 500 lines setting up the basic category theory and strong homotopy theory of finite posets.

(d) We have about 590 lines about the operation of subdivision on finite posets, and its interaction with strong homotopy (which takes most of the work).

(e) We have about 170 lines about the lattice of upper sets in a finite poset.

(f) We have about 2100 lines about the specific finite posets and maps that are relevant for iterated localisation.

For comparison: the LaTeX file is 3500 lines, and the PDF paper is 42 pages. Only about 7 PDF pages are fully formalised.

What's in the Lean code?

(a) We have about 240 lines of code about `fin n`. Roughly half of this is about the largest or smallest element of a nonempty subset of `fin n`. The rest is just tiny lemmas.

(b) If α is a finite, totally ordered set of size n then there is a unique order-preserving bijection from α to `fin n`. We have about 620 lines related to this.

(c) We have about 500 lines setting up the basic category theory and strong homotopy theory of finite posets.

(d) We have about 590 lines about the operation of subdivision on finite posets, and its interaction with strong homotopy (which takes most of the work).

(e) We have about 170 lines about the lattice of upper sets in a finite poset.

(f) We have about 2100 lines about the specific finite posets and maps that are relevant for iterated localisation.

For comparison: the LaTeX file is 3500 lines, and the PDF paper is 42 pages. Only about 7 PDF pages are fully formalised.

Where is the Lean code?

(a) There are six ancillary files attached to the arxiv submission of the paper. I did not put detailed instructions on the arxiv about how to use these files, nor did I do anything to attach them to a particular version of Lean and Mathlib. They are no longer compatible with the current versions. I will update them when the paper has been refereed, but not after that. The ancillary file framework is not very convenient when the number of files is large.

(b) The files are also in a GitHub repository. I have kept them separate from my main Lean GitHub repository, even though they contain some of the same files. I recently updated the repository to make the code compatible with current Mathlib, but only after it was broken for a couple of months.

(c) Version control issues can probably be solved with the right configuration of elan/leanpkg.toml etc. But this needs to be documented and made more transparent for referees and other users.

Where is the Lean code?

(a) There are six ancillary files attached to the arxiv submission of the paper. I did not put detailed instructions on the arxiv about how to use these files, nor did I do anything to attach them to a particular version of Lean and Mathlib. They are no longer compatible with the current versions. I will update them when the paper has been refereed, but not after that. The ancillary file framework is not very convenient when the number of files is large.

(b) The files are also in a GitHub repository. I have kept them separate from my main Lean GitHub repository, even though they contain some of the same files. I recently updated the repository to make the code compatible with current Mathlib, but only after it was broken for a couple of months.

(c) Version control issues can probably be solved with the right configuration of elan/leanpkg.toml etc. But this needs to be documented and made more transparent for referees and other users.

Where is the Lean code?

(a) There are six ancillary files attached to the arxiv submission of the paper. I did not put detailed instructions on the arxiv about how to use these files, nor did I do anything to attach them to a particular version of Lean and Mathlib. They are no longer compatible with the current versions. I will update them when the paper has been refereed, but not after that. The ancillary file framework is not very convenient when the number of files is large.

(b) The files are also in a GitHub repository. I have kept them separate from my main Lean GitHub repository, even though they contain some of the same files. I recently updated the repository to make the code compatible with current Mathlib, but only after it was broken for a couple of months.

(c) Version control issues can probably be solved with the right configuration of elan/leanpkg.toml etc. But this needs to be documented and made more transparent for referees and other users.

Where is the Lean code?

(a) There are six ancillary files attached to the arxiv submission of the paper. I did not put detailed instructions on the arxiv about how to use these files, nor did I do anything to attach them to a particular version of Lean and Mathlib. They are no longer compatible with the current versions. I will update them when the paper has been refereed, but not after that. The ancillary file framework is not very convenient when the number of files is large.

(b) The files are also in a GitHub repository. I have kept them separate from my main Lean GitHub repository, even though they contain some of the same files. I recently updated the repository to make the code compatible with current Mathlib, but only after it was broken for a couple of months.

(c) Version control issues can probably be solved with the right configuration of elan/leanpkg.toml etc. But this needs to be documented and made more transparent for referees and other users.

Cross-referencing

- This project consists of a conventionally structured LaTeX paper, together with loosely coupled Lean files. Other projects might have tighter coupling and different issues.
- The Lean files have some comments like

  ```
  /-- LaTeX: prop-sg-cofinal -/
  ```

  These refer to LaTeX labels and so are only meaningful with reference to the `.tex` file or the generated `.aux` file. One could set up a postprocessor to parse the `.aux` file and add theorem numbers to the `.lean` file, but we have not done so.
- The correspondence between Lean and LaTeX is very ragged. Large parts of the Lean code are human-trivial and so are not mentioned in LaTeX. Large parts of the LaTeX code consist of expository remarks, or material that we have not yet formalised. In some cases the human-readable proofs have a different structure from the formalisations.

Cross-referencing

- This project consists of a conventionally structured LaTeX paper, together with loosely coupled Lean files. Other projects might have tighter coupling and different issues.
- The Lean files have some comments like

  ```
  /-- LaTeX: prop-sg-cofinal -/
  ```

  These refer to LaTeX labels and so are only meaningful with reference to the `.tex` file or the generated `.aux` file. One could set up a postprocessor to parse the `.aux` file and add theorem numbers to the `.lean` file, but we have not done so.
- The correspondence between Lean and LaTeX is very ragged. Large parts of the Lean code are human-trivial and so are not mentioned in LaTeX. Large parts of the LaTeX code consist of expository remarks, or material that we have not yet formalised. In some cases the human-readable proofs have a different structure from the formalisations.

Cross-referencing

- This project consists of a conventionally structured LaTeX paper, together with loosely coupled Lean files. Other projects might have tighter coupling and different issues.
- The Lean files have some comments like

  ```
  /-- LaTeX: prop-sg-cofinal -/
  ```

  These refer to LaTeX labels and so are only meaningful with reference to the `.tex` file or the generated `.aux` file. One could set up a postprocessor to parse the `.aux` file and add theorem numbers to the `.lean` file, but we have not done so.
- The correspondence between Lean and LaTeX is very ragged. Large parts of the Lean code are human-trivial and so are not mentioned in LaTeX. Large parts of the LaTeX code consist of expository remarks, or material that we have not yet formalised. In some cases the human-readable proofs have a different structure from the formalisations.

Cross-referencing

- This project consists of a conventionally structured LaTeX paper, together with loosely coupled Lean files. Other projects might have tighter coupling and different issues.
- The Lean files have some comments like

  ```
  /-- LaTeX: prop-sg-cofinal -/
  ```

  These refer to LaTeX labels and so are only meaningful with reference to the `.tex` file or the generated `.aux` file. One could set up a postprocessor to parse the `.aux` file and add theorem numbers to the `.lean` file, but we have not done so.
- The correspondence between Lean and LaTeX is very ragged. Large parts of the Lean code are human-trivial and so are not mentioned in LaTeX. Large parts of the LaTeX code consist of expository remarks, or material that we have not yet formalised. In some cases the human-readable proofs have a different structure from the formalisations.

Moving to Mathlib

- About half the code is material that should certainly be in Mathlib, either with the current code or with some other code that achieves a similar result.
- Many things could be done better or differently. If I put in some PRs to include material in Mathlib, there would probably be a long process of review and refactoring before they were accepted. It feels like this would become very unwieldy with even a small increase in people wanting to contribute new material.
- (arxiv has about 35000 maths papers and 35000 CS papers per year.)
- Maybe `mathlib-nursery` is part of the answer. But that seems to be largely inactive, and I have not tried it.
- About half of the code is quite specific to the *Iterated chromatic localisation* project. This might not be considered appropriate for Mathlib. However, if left outside Mathlib then it will certainly rot.
- It would be useful for Travis to attempt to build my repository against current Mathlib, and report to me if it fails. But I have not worked out how to do anything with Travis. Perhaps there could be instructions in the Mathlib `doc` directory.

Moving to Mathlib

- About half the code is material that should certainly be in Mathlib, either with the current code or with some other code that achieves a similar result.
- Many things could be done better or differently. If I put in some PRs to include material in Mathlib, there would probably be a long process of review and refactoring before they were accepted. It feels like this would become very unwieldy with even a small increase in people wanting to contribute new material.
- (arxiv has about 35000 maths papers and 35000 CS papers per year.)
- Maybe `mathlib-nursery` is part of the answer. But that seems to be largely inactive, and I have not tried it.
- About half of the code is quite specific to the *Iterated chromatic localisation* project. This might not be considered appropriate for Mathlib. However, if left outside Mathlib then it will certainly rot.
- It would be useful for Travis to attempt to build my repository against current Mathlib, and report to me if it fails. But I have not worked out how to do anything with Travis. Perhaps there could be instructions in the Mathlib `doc` directory.

Moving to Mathlib

- About half the code is material that should certainly be in Mathlib, either with the current code or with some other code that achieves a similar result.
- Many things could be done better or differently. If I put in some PRs to include material in Mathlib, there would probably be a long process of review and refactoring before they were accepted. It feels like this would become very unwieldy with even a small increase in people wanting to contribute new material.
- (arxiv has about 35000 maths papers and 35000 CS papers per year.)
- Maybe `mathlib-nursery` is part of the answer. But that seems to be largely inactive, and I have not tried it.
- About half of the code is quite specific to the *Iterated chromatic localisation* project. This might not be considered appropriate for Mathlib. However, if left outside Mathlib then it will certainly rot.
- It would be useful for Travis to attempt to build my repository against current Mathlib, and report to me if it fails. But I have not worked out how to do anything with Travis. Perhaps there could be instructions in the Mathlib doc directory.

Moving to Mathlib

- About half the code is material that should certainly be in Mathlib, either with the current code or with some other code that achieves a similar result.
- Many things could be done better or differently. If I put in some PRs to include material in Mathlib, there would probably be a long process of review and refactoring before they were accepted. It feels like this would become very unwieldy with even a small increase in people wanting to contribute new material.
- (arxiv has about 35000 maths papers and 35000 CS papers per year.)
- Maybe `mathlib-nursery` is part of the answer. But that seems to be largely inactive, and I have not tried it.
- About half of the code is quite specific to the *Iterated chromatic localisation* project. This might not be considered appropriate for Mathlib. However, if left outside Mathlib then it will certainly rot.
- It would be useful for Travis to attempt to build my repository against current Mathlib, and report to me if it fails. But I have not worked out how to do anything with Travis. Perhaps there could be instructions in the Mathlib `doc` directory.

Moving to Mathlib

- About half the code is material that should certainly be in Mathlib, either with the current code or with some other code that achieves a similar result.
- Many things could be done better or differently. If I put in some PRs to include material in Mathlib, there would probably be a long process of review and refactoring before they were accepted. It feels like this would become very unwieldy with even a small increase in people wanting to contribute new material.
- (arxiv has about 35000 maths papers and 35000 CS papers per year.)
- Maybe `mathlib-nursery` is part of the answer. But that seems to be largely inactive, and I have not tried it.
- About half of the code is quite specific to the *Iterated chromatic localisation* project. This might not be considered appropriate for Mathlib. However, if left outside Mathlib then it will certainly rot.
- It would be useful for Travis to attempt to build my repository against current Mathlib, and report to me if it fails. But I have not worked out how to do anything with Travis. Perhaps there could be instructions in the Mathlib doc directory.

Moving to Mathlib

- About half the code is material that should certainly be in Mathlib, either with the current code or with some other code that achieves a similar result.
- Many things could be done better or differently. If I put in some PRs to include material in Mathlib, there would probably be a long process of review and refactoring before they were accepted. It feels like this would become very unwieldy with even a small increase in people wanting to contribute new material.
- (arxiv has about 35000 maths papers and 35000 CS papers per year.)
- Maybe `mathlib-nursery` is part of the answer. But that seems to be largely inactive, and I have not tried it.
- About half of the code is quite specific to the *Iterated chromatic localisation* project. This might not be considered appropriate for Mathlib. However, if left outside Mathlib then it will certainly rot.
- It would be useful for Travis to attempt to build my repository against current Mathlib, and report to me if it fails. But I have not worked out how to do anything with Travis. Perhaps there could be instructions in the Mathlib doc directory.

Moving to Mathlib

- About half the code is material that should certainly be in Mathlib, either with the current code or with some other code that achieves a similar result.
- Many things could be done better or differently. If I put in some PRs to include material in Mathlib, there would probably be a long process of review and refactoring before they were accepted. It feels like this would become very unwieldy with even a small increase in people wanting to contribute new material.
- (arxiv has about 35000 maths papers and 35000 CS papers per year.)
- Maybe `mathlib-nursery` is part of the answer. But that seems to be largely inactive, and I have not tried it.
- About half of the code is quite specific to the *Iterated chromatic localisation* project. This might not be considered appropriate for Mathlib. However, if left outside Mathlib then it will certainly rot.
- It would be useful for Travis to attempt to build my repository against current Mathlib, and report to me if it fails. But I have not worked out how to do anything with Travis. Perhaps there could be instructions in the Mathlib `doc` directory.

Referring to the literature

- The Lean code for the current project does not rely on results quoted from the literature. But that would change if we tried to formalise more of the paper, or turned to other projects.
- For example, in Boardman's stable homotopy category $\mathcal{B}$ there is an important object called MU. How to refer to $\mathcal{B}$ and MU in Lean? Full formalisation would be a massive project.
- We need a name for the file containing our pseudodefinition of $\mathcal{B}$.
    - We could make up our own name but that might not be a scalable procedure.
    - In this case there is a clear choice of Mathematics Subject Classification (namely 55P42), and that could be used as a component of the name. But other work may not have a clear choice.
    - If we use the MSC as a general structural guide, but do not use the numeric codes, then we arrive at a name like

      `algebraic_topology/homotopy_theory/stable_homotopy/spectra.lean`
    - Boardman's own development was never published. The traditional reference is a book by Adams (0402720 in MathSciNet). There are later textbook treatments that have many technical advantages, but no canonical choice among these.
- Many more issues

Referring to the literature

- The Lean code for the current project does not rely on results quoted from the literature. But that would change if we tried to formalise more of the paper, or turned to other projects.
- For example, in Boardman's stable homotopy category $\mathcal{B}$ there is an important object called MU. How to refer to $\mathcal{B}$ and MU in Lean? Full formalisation would be a massive project.
- We need a name for the file containing our pseudodefinition of $\mathcal{B}$.
  - We could make up our own name but that might not be a scalable procedure.
  - In this case there is a clear choice of Mathematics Subject Classification (namely 55P42), and that could be used as a component of the name. But other work may not have a clear choice.
  - If we use the MSC as a general structural guide, but do not use the numeric codes, then we arrive at a name like

    `algebraic_topology/homotopy_theory/stable_homotopy/spectra.lean`
  - Boardman's own development was never published. The traditional reference is a book by Adams (0402720 in MathSciNet). There are later textbook treatments that have many technical advantages, but no canonical choice among these.
- Many more issues

Referring to the literature

- The Lean code for the current project does not rely on results quoted from the literature. But that would change if we tried to formalise more of the paper, or turned to other projects.
- For example, in Boardman's stable homotopy category $\mathcal{B}$ there is an important object called MU. How to refer to $\mathcal{B}$ and MU in Lean? Full formalisation would be a massive project.
- We need a name for the file containing our pseudodefinition of $\mathcal{B}$.
  - We could make up our own name but that might not be a scalable procedure.
  - In this case there is a clear choice of Mathematics Subject Classification (namely 55P42), and that could be used as a component of the name. But other work may not have a clear choice.
  - If we use the MSC as a general structural guide, but do not use the numeric codes, then we arrive at a name like

    `algebraic_topology/homotopy_theory/stable_homotopy/spectra.lean`
  - Boardman's own development was never published. The traditional reference is a book by Adams (0402720 in MathSciNet). There are later textbook treatments that have many technical advantages, but no canonical choice among these.
- Many more issues

Referring to the literature

- The Lean code for the current project does not rely on results quoted from the literature. But that would change if we tried to formalise more of the paper, or turned to other projects.
- For example, in Boardman's stable homotopy category $\mathcal{B}$ there is an important object called MU. How to refer to $\mathcal{B}$ and MU in Lean? Full formalisation would be a massive project.
- We need a name for the file containing our pseudodefinition of $\mathcal{B}$.
  - We could make up our own name but that might not be a scalable procedure.
  - In this case there is a clear choice of Mathematics Subject Classification (namely 55P42), and that could be used as a component of the name. But other work may not have a clear choice.
  - If we use the MSC as a general structural guide, but do not use the numeric codes, then we arrive at a name like

    `algebraic_topology/homotopy_theory/stable_homotopy/spectra.lean`
  - Boardman's own development was never published. The traditional reference is a book by Adams (0402720 in MathSciNet). There are later textbook treatments that have many technical advantages, but no canonical choice among these.
- Many more issues

Referring to the literature

- The Lean code for the current project does not rely on results quoted from the literature. But that would change if we tried to formalise more of the paper, or turned to other projects.
- For example, in Boardman's stable homotopy category $\mathcal{B}$ there is an important object called MU. How to refer to $\mathcal{B}$ and MU in Lean? Full formalisation would be a massive project.
- We need a name for the file containing our pseudodefinition of $\mathcal{B}$.
  - We could make up our own name but that might not be a scalable procedure.
  - In this case there is a clear choice of Mathematics Subject Classification (namely 55P42), and that could be used as a component of the name. But other work may not have a clear choice.
  - If we use the MSC as a general structural guide, but do not use the numeric codes, then we arrive at a name like

    `algebraic_topology/homotopy_theory/stable_homotopy/spectra.lean`
  - Boardman's own development was never published. The traditional reference is a book by Adams (0402720 in MathSciNet). There are later textbook treatments that have many technical advantages, but no canonical choice among these.
- Many more issues

Referring to the literature

- The Lean code for the current project does not rely on results quoted from the literature. But that would change if we tried to formalise more of the paper, or turned to other projects.
- For example, in Boardman's stable homotopy category $\mathcal{B}$ there is an important object called MU. How to refer to $\mathcal{B}$ and MU in Lean? Full formalisation would be a massive project.
- We need a name for the file containing our pseudodefinition of $\mathcal{B}$.
  - We could make up our own name but that might not be a scalable procedure.
  - In this case there is a clear choice of Mathematics Subject Classification (namely 55P42), and that could be used as a component of the name. But other work may not have a clear choice.
  - If we use the MSC as a general structural guide, but do not use the numeric codes, then we arrive at a name like

    `algebraic_topology/homotopy_theory/stable_homotopy/spectra.lean`
  - Boardman's own development was never published. The traditional reference is a book by Adams (0402720 in MathSciNet). There are later textbook treatments that have many technical advantages, but no canonical choice among these.
- Many more issues

Referring to the literature

- The Lean code for the current project does not rely on results quoted from the literature. But that would change if we tried to formalise more of the paper, or turned to other projects.
- For example, in Boardman's stable homotopy category $\mathcal{B}$ there is an important object called MU. How to refer to $\mathcal{B}$ and MU in Lean? Full formalisation would be a massive project.
- We need a name for the file containing our pseudodefinition of $\mathcal{B}$.
  - We could make up our own name but that might not be a scalable procedure.
  - In this case there is a clear choice of Mathematics Subject Classification (namely 55P42), and that could be used as a component of the name. But other work may not have a clear choice.
  - If we use the MSC as a general structural guide, but do not use the numeric codes, then we arrive at a name like

    `algebraic_topology/homotopy_theory/stable_homotopy/spectra.lean`
  - Boardman's own development was never published. The traditional reference is a book by Adams (0402720 in MathSciNet). There are later textbook treatments that have many technical advantages, but no canonical choice among these.
- Many more issues

Referring to the literature

- The Lean code for the current project does not rely on results quoted from the literature. But that would change if we tried to formalise more of the paper, or turned to other projects.
- For example, in Boardman's stable homotopy category $\mathcal{B}$ there is an important object called MU. How to refer to $\mathcal{B}$ and MU in Lean? Full formalisation would be a massive project.
- We need a name for the file containing our pseudodefinition of $\mathcal{B}$.
  - We could make up our own name but that might not be a scalable procedure.
  - In this case there is a clear choice of Mathematics Subject Classification (namely 55P42), and that could be used as a component of the name. But other work may not have a clear choice.
  - If we use the MSC as a general structural guide, but do not use the numeric codes, then we arrive at a name like

    `algebraic_topology/homotopy_theory/stable_homotopy/spectra.lean`
  - Boardman's own development was never published. The traditional reference is a book by Adams (0402720 in MathSciNet). There are later textbook treatments that have many technical advantages, but no canonical choice among these.
- Many more issues

Referring to the literature

- The Lean code for the current project does not rely on results quoted from the literature. But that would change if we tried to formalise more of the paper, or turned to other projects.
- For example, in Boardman's stable homotopy category $\mathcal{B}$ there is an important object called MU. How to refer to $\mathcal{B}$ and MU in Lean? Full formalisation would be a massive project.
- We need a name for the file containing our pseudodefinition of $\mathcal{B}$.
  - We could make up our own name but that might not be a scalable procedure.
  - In this case there is a clear choice of Mathematics Subject Classification (namely 55P42), and that could be used as a component of the name. But other work may not have a clear choice.
  - If we use the MSC as a general structural guide, but do not use the numeric codes, then we arrive at a name like

    `algebraic_topology/homotopy_theory/stable_homotopy/spectra.lean`
  - Boardman's own development was never published. The traditional reference is a book by Adams (0402720 in MathSciNet). There are later textbook treatments that have many technical advantages, but no canonical choice among these.
- Many more issues