Formal Topology in Univalent Foundations

Ayberk Tosun

February 27, 2020

Chalmers University of Technology

Motivation

Topology

$$\text{underststood} \Big\Downarrow \text{constructively}$$

Pointless topology

$$\text{understood} \Big\Downarrow \text{predicatively}$$

Formal topology

What locales are like

- Abstraction of open sets of a topology.
- Logic of *observable properties*.
- CS view: logic of *semidecidable properties*.

What locales are like

- Abstraction of open sets of a topology.
- Logic of *observable properties*.
- CS view: logic of *semidecidable properties*.
- "Junior-grade topos theory".

Locales

A poset $\mathcal{O}$ such that

- finite subsets of $\mathcal{O}$ have meets,
- all subsets of $\mathcal{O}$ have joins, and
- binary meets distribute over arbitrary joins:

$$a \wedge \left(\bigvee_{i \in I} b_i \right) = \bigvee_{i \in I} (a \wedge b_i),$$

for any $a \in \mathcal{O}$ and I-indexed family b over $\mathcal{O}$.

Locales of downward-closed subsets

Given a poset

$$\begin{aligned} A &: \mathsf{Type}_m \\ \sqsubseteq &: A \to A \to \mathsf{hProp}_m \end{aligned}$$

the type of downward-closed subsets of A is:

$$\sum_{(U\,:\,\mathcal{P}(A))} \prod_{(x\ y\,:\,A)} x \in U \to y \sqsubseteq x \to y \in U,$$

where

$$\mathcal{P} : \mathsf{Type}_m \to \mathsf{Type}_{m+1}$$
$$\mathcal{P}(A) :\equiv A \to \mathsf{hProp}_m.$$

This forms a locale:

$$
\begin{aligned}
\top &:\equiv \lambda\_.\ 1 \\
A \wedge B &:\equiv \lambda x.\ (x \in A) \times (x \in B) \\
\bigvee_{i\,:\,I} \mathbf{B}_i &:\equiv \lambda x.\ \left\| \sum_{(i\,:\,I)} x \in \mathbf{B}_i \right\|
\end{aligned}
$$

Nuclei for locales

Question: can we get all locales out of posets in this way?

One way is to employ the notion of a nucleus.

Let F be a locale. A nucleus on F is an endofunction $\mathbf{j} : |F| \to |F|$ such that

(1) $\prod_{(x \ : \ A)} x \sqsubseteq \mathbf{j}(x)$ [extensiveness],

(2) $\prod_{(x \ y \ : \ A)} \mathbf{j}(x \wedge y) = \mathbf{j}(x) \wedge \mathbf{j}(y)$ [meet preservation], and

(3) $\prod_{(x \ : \ A)} \mathbf{j}(\mathbf{j}(x)) \sqsubseteq \mathbf{j}(x)$ [idempotence].

Closure operators

In the particular case where F is the locale of downward-closed subsets for a poset $A : \text{Type}_m$, the nucleus can be seen as a closure operator—**if it can be shown to be propositional**.

- $\quad : \quad \underbrace{\mathcal{P}(A) \to \mathcal{P}(A)}_{\text{This is what we want.}}$

Closure operators

In the particular case where F is the locale of downward-closed subsets for a poset $A : \mathsf{Type}_m$, the nucleus can be seen as a closure operator—**if it can be shown to be propositional**.

- $\quad : \quad \underbrace{\mathcal{P}(A) \to A \to \mathsf{hProp}_m}_{\text{This is what we want.}}$

Closure operators

In the particular case where F is the locale of downward-closed subsets for a poset $A : \mathsf{Type}_m$, the nucleus can be seen as a closure operator—**if it can be shown to be propositional**.

- $$\quad : \quad \underbrace{\mathcal{P}\,(A) \to A \to \mathsf{Type}_m}_{\text{This is what we have.}}$$

Baire space ($\mathbb{N} \to \mathbb{N}$)

```
data 𝔻 : Type₀ where
  []   : 𝔻
  _⌢_  : 𝔻 → ℕ → 𝔻

IsDC : (𝔻 → Type₀) → Type₀
IsDC P = (σ : 𝔻) (n : ℕ) → P σ → P (σ ⌢ n)
```

Baire space ($\mathbb{N} \to \mathbb{N}$)

```
data _◂_ (σ : 𝔻) (P : 𝔻 → Type₀) : Type₀ where
  dir    : P σ → σ ◂ P
  branch : ((n : ℕ) → (σ ⌢ n) ◂ P) → σ ◂ P
  squash : (p q : σ ◂ P) → p ≡ q
```

We can now show that this defines a nucleus, without choice!

Baire space ($\mathbb{N} \to \mathbb{N}$)

Using the following, and then *truncating from the outside* does not work.

```
data _◂⋆_ (σ : 𝔻) (P : 𝔻 → Type₀) : Type₀ where
  dir    : P σ → σ ◂⋆ P
  branch : ((n : ℕ) → (σ ⌢ n) ◂⋆ P) → σ ◂⋆ P
  -- squash : (φ ψ : σ ◂ P) → φ ≡ ψ
```

Baire space ($\mathbb{N} \to \mathbb{N}$)

We can now prove the following idempotence law, without using countable choice ($\prod_{(i\ :\ I)} \|B_i\| \to \| \prod_{(i\ :\ I)} B_i\|$).

```
δ : σ ◂ P → ((v : 𝔻) → P v → v ◂ Q) → σ ◂ Q
δ (dir     uεP)         φ = φ _ uεP
δ (branch f)            φ = branch (λ n → δ (f n) φ) ⟶ problem
δ (squash u◂P₀ u◂P₁ i) φ = squash (δ u◂P₀ φ) (δ u◂P₁ φ) i

idempotence : σ ◂ (λ - → - ◂ P) → σ ◂ P
idempotence u◂◂P = δ u◂◂P (λ _ v◂P → v◂P)
```

Baire space ($\mathbb{N} \to \mathbb{N}$)

```
-- ζ inference à la Brouwer.
ζ : (n : ℕ) → IsDC P → σ ◂ P → (σ ⌢ n) ◂ P
ζ n dc (dir     σεP)         = dir (dc _ n σεP)
ζ n dc (branch f)            = branch λ m → ζ m dc (f n)
ζ n dc (squash σ◂P σ◂P′ i) = squash (ζ n dc σ◂P) (ζ n dc σ◂P′) i

ζ′ : IsDC P → IsDC (λ - → - ◂ P)
ζ′ P-dc σ n σ◂P = ζ n P-dc σ◂P
```

Baire space ($\mathbb{N} \to \mathbb{N}$)

This example can be accessed at:

`https://ayberkt.gitlab.io/msc-thesis/BaireSpace.html`