Formalising Mathematics – Coursework 3

In point-set topology, one considers a topological space to be made up of an underlying set X along with a collection of distinguished subsets of X. These distinguished sets are called the open subsets of X and they must satisfy some rules, namely they must include and X, and they must be closed under finite intersections and arbitrary unions. From here, continous maps can be defined as maps whose preimage sends open sets to open sets and the usual study of topology continues from there.

Point-free topology offers a different formulation for a space, where the underlying set of a space X is forgotten, and the focus moves to the structure given by the open sets. Ordering the open sets of a topological space by inclusions, and considering the above axioms for the topology on a set, we see that the opens of a topological space have the structure of a frame. Furthermore, given a continuous map between topological spaces, its inverse image turns out to be a homomorphism of frames between the relevant frames of opens. As a result of these observations, point-free topology forgets about topological spaces and continuous maps and focuses instead on frames and their homomorphisms.

As we will see later, the above idea of taking opens of a topological space gives rise to a functor from $\mathbf{Top}^{\mathrm{op}}$ to **Frame**. Going the other direction, there will be a functor taking a frame to the topological space of its points. The act of taking points is left adjoint to taking opens, and this adjunction will be the main goal of this project.

The Mathematics

Frames

A frame L can be defined as a partially ordered set which has all finite meets, arbitrary joins and it satisfies the following distributivity property for all $x \in L$ and $S \subseteq L$:

$$x \wedge \bigvee_{s \in S} s = \bigvee_{s \in S} (x \wedge s) \tag{1}$$

Here, all finite limits includes the empty meet, so in particular L has a top element $\top$.

A map $f : L \to M$ is a frame homomorphism if it respects finite meets and arbitrary joins, or equivalently, the following three conditions hold:

$$f(\top) = \top \qquad \forall x, y \in L, f(x \wedge y) = f(x) \wedge f(y) \qquad \forall S \subseteq L, f(\bigvee_{s \in S} s) = \bigvee_{s \in S} f(s)$$

From order theory we know that if a poset has arbitrary joins, then it must also have arbitrary meets, so frames can be seen as complete lattices satisfying (1). Despite this, it is important to note that our definition of frame homomorphisms does not require them to preserve arbitrary meets, so even though they exist, they will not be considered.

An important example of a frame is the initial frame $\mathbb{2} = \{\top, \bot\}$ of truth values, with meets given by logical conjunction and joins given by logical disjunction. This frame is initial since frame homomorphisms must preserve $\top$ and $\bot$, hence there is only one possible such map from $\mathbb{2}$ into any other frame L.

Points of a Frame

Although frames are the main topic of study in point-free topology, we now turn our attention to the points of a frame. For our purposes, given a frame L, we define a point as a frame homomorphism p from L to the initial frame $\mathbb{2}$. If we think of the elements of L as open sets, then for any $a \in L$, we take $p(a) = \top$ to mean that the point p lies in the open a and $p(a) = \bot$ to mean that it does not lie in a.

Fixing some lattice L, we now consider its set of points $\mathbf{Frame}(L, \mathbb{2})$. We can equip this set with a topology in an intuitive way: we call a subset $U \in \mathbf{Frame}(L, \mathbb{2})$ open if it is of the form $\{p \in \mathbf{Frame}(L, \mathbb{2}) \mid p(a) = \top\}$ for some $a \in L$. This takes the idea that we should view elements of L as open sets, and concretises it, by identifying elements $a \in L$ with the sets of points they contain. To see that this is in fact a topology consider that

- All points p must send $p(\top) = \top$, hence $\mathbf{Frame}(L, \mathbb{2})$ is the open set given by $\top$;
- Similarly, all points p must send $p(\bot) = \bot \neq \top$ hence is the open set given by $\bot$;
- Given two open sets $\{p \mid p(a) = \top\}$ and $\{p \mid p(b) = \top\}$ for $a, b \in L$, notice that their intersection is given by

$$\{p \mid p(a) = \top \text{ and } p(b) = \top\} = \{p \mid p(a) \wedge p(b) = \top\} = \{p \mid p(a \wedge b) = \top\}$$

- Given a collection of opens $\{\{p \mid p(s) = \top\}\}_{s \in S}$, then their union is given by

$$\{p \mid p(s) = \top \text{ for some } s \in S\} = \{p \mid \bigvee_{s \in S} p(s) = \top\} = \{p \mid p(\bigvee_{s \in S} s) = \top\}$$

Furthermore, given a frame homomorphism $f : L \to M$, we get a continuous map $\mathbf{Frame}(f, \mathbb{2}) : \mathbf{Frame}(M, \mathbb{2}) \to \mathbf{Frame}(L, \mathbb{2})$ which sends points p to points $p \circ f$. To see that this is continuous, consider some open set $\{p \mid p(a) = \top\} \subseteq \mathbf{Frame}(L, \mathbb{2})$ given by some $a \in L$. Then, the preimage under $\mathbf{Frame}(f, \mathbb{2})$ is given by $\{q| \ q \circ f(a) = \top\} = \{q| \ q(f(a)) = \top\}$, which is the open set associated with the element $f(a) \in M$.

It can be checked that this process sends identity maps to identity maps, and that it respects composition of functions. This turns $\mathbf{Frame}(-, \mathbb{2})$ into a functor from **Frame** to $\mathbf{Top}^{\mathrm{op}}$.

The Points-Opens Adjunction

So we have a functor $\mathbf{Frame}(-, \mathbb{2}) : \mathbf{Frame} \to \mathbf{Top}^{\mathrm{op}}$ which sends frames their spaces of points. As alluded to before, there is also a functor $\mathit{Opens} : \mathbf{Top}^{\mathrm{op}} \to \mathbf{Frame}$ which sends topological spaces to their frame of opens. It turns out that these two functors are very much related, in fact $\mathbf{Frame}(-, \mathbb{2})$ is the left adjoint of *Opens*. To show that this is the case, we define the unit and counit of this adjunction, and then show that they satisfy the triangle identities.

First, we focus on the unit of the adjunction. For this we need a frame homomorphism from L to $\mathit{Opens}(\mathbf{Frame}(L, \mathbb{2}))$ for every frame L, so given an element $a \in L$, we need to assign it an open set of $\mathbf{Frame}(L, \mathbb{2}))$. But this is essentially what we did when giving $\mathbf{Frame}(L, \mathbb{2})$ a topology, so we can simply send $a \mapsto \{q \in \mathbf{Frame}(L, \mathbb{2}) \mid q(a) = \top\}$. This map is then a frame homomorphism for the same reasons that the open sets of $\mathbf{Frame}(L, \mathbb{2})$ formed a topology.

Next, we look at the counit, which will be a map in $\mathbf{Top}^{\mathrm{op}}$. As such, we actually wish to find a continuous map from X to $\mathbf{Frame}(\mathit{Opens}(X), \mathbb{2})$ for each topological space X. So given a point $x \in X$, we would send it to the frame homomorphism $\lambda\, U, x \in U$, which returns $\top$ if $x \in U$ and $\bot$ otherwise. To see that this map is continuous, fix some open set $\{p \mid p(U) = \top\} \subseteq \mathbf{Frame}(\mathit{Opens}(X), \mathbb{2})$ with $U \subseteq X$ open. The preimage of this set is then $\{x \mid (\lambda\, V, x \in V)(U) = \top\} = \{x \mid x \in U\} = U$ which is open by assumption.

To ensure the above gives us an adjunction, one would still need to check that these are natural transformations and that the triangle identities are satisfied. These proofs are mostly routine and not very enlightening though, so we defer them to the Lean code.

Although this adjunction indicates some nice behaviour between these two functors, they don't always behave as well as we might have hoped. For example, consider a nonempty topological space X with the trivial topology: such a space has only two opens, so its frame of opens is $\mathbb{2}$. Furthermore, since $\mathbb{2}$ is initial in **Frame**, it only has one point; this means that all nonempty spaces with the trivial topology only have one "point-free point". Similarly, there also exist frames which don't have any points at all, which are all indistinguishable when viewed through point-set topology. By restricting to a specific subclass of frames and topological spaces though, this adjunction turns into an equivalence of categories, but this is outside the scope of this project.

The Formalisation

For this project I aimed to formalise this adjunction between **Frame** and $\mathbf{Top}^{\mathrm{op}}$. Some of the ground work for this had already been done, namely both categories already existed in `mathlib` and there was even the functor `Top_op_to_Frame` which assigned a topological space to its frame of opens.

The first step was then to define the functor taking a frame to its topological space of points. For this, `mathlib` already had the notion of a frame homomorphism through the type `frame_hom`. In addition, the type `Prop` also had an instance of the typeclass `frame`, so this was used for the initial frame $\mathbb{2}$. This was an interesting decision since the use of `Prop` in this way would not work in a constructive setting, where the law of the excluded middle is not assumed. In such a setting, we would be unable to prove that `Prop` has exactly two inhabitants and things could get messy very quick. Thankfully, Lean's type theory does support excluded middle, evidenced by the following theorems:

```
theorem em : p ∨ ¬p
theorem prop_complete (a : Prop) : a = true ∨ a = false
```

Overall, this decision was very helpful since it meant I did not have to spend time proving uninteresting, but useful, lemmas about the frame structure of propositions.

The `topological_space` instance for the points of a frame was defined in:

```
instance pts_topological_space {L : Type u} [frame L] :
  topological_space (frame_hom L Prop) :=
{ is_open        := λ U, ∃ (a : L), U = {p | p a},
  is_open_univ   := ...
  is_open_inter  := ...
  is_open_sUnion := ...
}
```

Here I take advantage of the fact that we are using `Prop` as the initial frame, and simply use `p a` instead of `p a = ⊤` in order to define the open sets. This is fine to do as they are actually the same proposition, by the following lemma:

```
lemma eq_true {a : Prop} : (a = true) = a
```

Next, I defined how this functor of points acts on frame morphisms and also showed it was a functor in the following two definitions:

```
def pts_map {L M : Type u} [frame L] [frame M] (f : frame_hom L M) :
  continuous_map (frame_hom M Prop) (frame_hom L Prop)
def points : Frame ⥤ Topᵒᵖ
```

With both functors defined, we can now move on to proving the desired adjunction. To do this, I first defined the individual components for the unit and counit natural transformations:

```
def points_opens_unit_map (L : Type u) [frame L] :
  frame_hom L (topological_space.opens (frame_hom L Prop)) :=
{ to_fun   := λ a, ⟨{p | p a}, a, refl {p | p a}⟩,
  map_inf' := ...
  map_top' := ...
  map_Sup' := ...
}

def points_opens_counit_map (X : Type u) [topological_space X] :
  continuous_map X (frame_hom (topological_space.opens X) Prop) :=
{ to_fun := λ x,
  { to_fun   := λ U, x ∈ U,
    map_inf' := ...
    map_top' := ...
    map_Sup' := ...
  },
  continuous_to_fun := ...
}
```

With these defined, I then defined the natural transformations and used `mk_of_unit_counit` to define the adjunction:

```
def points_opens_unit : 𝟭 Frame ⟶ points ⋙ Top_op_to_Frame :=
{ app := λ L, points_opens_unit_map L,
  naturality' := ...
}

def points_opens_counit : Top_op_to_Frame ⋙ points ⟶ 𝟭 (Topᵒᵖ) :=
{ app := λ Xop, (quiver.hom.op (points_opens_counit_map Xop.unop) :
    (opposite.op (bundled.of _)) ⟶ (opposite.op Xop.unop)),
  naturality' := ...
}

def points_opens_adjunction : points ⊣ Top_op_to_Frame
```

Defining the unit was simple and required no special attention. For the counit however, there were some difficulties due to the fact that I was defining maps on an opposite category. The main issue was due to the type checker not realising that `quiver.hom.op`'s return type was definitionally equal to what `app` needed. To solve this, initially I used tactic mode to rewrite the types until they matched, but this was not ideal since definitions arising from tactic mode can often become very unwieldy. Since the types were definitionally equal though, I was able to remove the rewrites and change the definition to term mode by explicitly giving the return type of `quiver.hom.op`.

Finally, I also made some progress towards showing that this adjunction resctricts to an equivalence of the appropriate subcategories. Namely, I defined the idea of a spatial frame and from there showed that the components of the adjunction unit associated with spatial frames were in fact frame isomorphisms:

```
class spatial (L : Type u) [frame L] : Prop :=
(enough_points : ∀ {a b : L}, ¬ a ≤ b → ∃ p : frame_hom L Prop, p a ∧ ¬ p b)

lemma spatial_imp_points_opens_unit_map_injective {L : Type u}
  [frame L] [spatial L] : function.injective (points_opens_unit_map L)

lemma points_opens_unit_map_surjective {L : Type u} [frame L] :
  function.surjective (points_opens_unit_map L)

lemma spatial_imp_points_opens_unit_map_bijective {L : Type u}
  [frame L] [spatial L] : function.bijective (points_opens_unit_map L)

noncomputable def spatial_imp_points_opens_unit_map_order_iso
  (L : Type u) [frame L] [spatial L] :
  L ≃o topological_space.opens (frame_hom L Prop)
```