My HoTT notes

12 June 2020 - ongoing

1 Searchability

$$\texttt{Searchable}\ A :\equiv \prod p : A \to \mathbf{2}.\ \left(\sum x : A.p\ x = \texttt{tt}\right) + \prod x : A.p\ x = \texttt{ff}$$

In words: the type A is searchable if for every boolean predicate p we can find an element of A that satisfies p or else prove that no element of A satisfies p.

Theorem 1.1. The type `Searchable` A can be of any h-level.

Proof. $p\ x = \texttt{tt}$ and $p\ x = \texttt{ff}$ are propositions, the summands are disjoint, and dependent products preserve h-level (well, unless the domain is $\mathbf{0}$), so the h-level is determined by A.

If A is contractible, then `Searchable` A is contractible.

If A is a proposition, then `Searchable` A is a proposition.

If A is of h-level n, then so is `Searchable` A.

Hope my mental techniques for h-levels work :)

□

$$\texttt{MerelySearchable}\ A :\equiv \forall p : A \to \mathbf{2}.\ (\exists x : A.p\ x = \texttt{tt}) \vee \forall x : A.p\ x = \texttt{ff}$$

In words: a type is merely searchable if we can decide whether there merely exists an element that satisfies the given boolean predicate p, but in general there is no way to find such an element.

Theorem 1.2. `isProp` (`MerelySearchable`A)

Proof. We use the truncated logic. □

Theorem 1.3. Some instances of searchable types.

1. Searchable types include: $\mathbf{0}, \mathbf{1}, \mathbf{2}$, all finite types.

1.5. The interval is searchable.

2. If A and B are searchable, so are $A \times B$ and $A + B$.

3. If A is searchable and $B : A \to \mathcal{U}$ is a family of searchable types, $\sum x : A.B\ x$ is also searchable.

4. If A is searchable and $f : A \to B$, then the image of f is also searchable.

5. Very importantly, the type $\mathbb{N}_\infty$ is searchable.

6. There are arbitrarily big searchable ordinals.

Proof. 1. For finite types: we assume that "finite" means "there is a list of all the elements". So, use it to check every element, and poof – done.

1.5. The interval is contractible, thus equivalent to $\mathbf{1}$, so it's searchable.

2. To search $A + B$ first search A, then search B.

3. To search $A \times B$ and $\sum x : A.B\ x$ search for an a, and when looking for it, search for a b that can be paired with it.

4. To search the image of $f : A \to B$ using the predicate $p : B \to \mathbf{2}$, search the whole of A using the predicate fp (beware: I write composition of two functions f and g like fg!).

5. The searchability of $\mathbb{N}_\infty$ was established by Martin Escardó.

6. Proved by Escardó.

□

Theorem 1.4. Taboos

Searchability of $\mathbb{N}$ is taboo. In general, preservation of searchability for $\prod$, `W` and `M` and $=$ should be taboo.

Proof. $\mathbb{N}$ is the primordial searchability taboo – how can we search infinitely many numbers in a finite time (besides checking every next number twice as fast as the previous one)?

`W` can be used to define $\mathbb{N}$, so its searchability should be taboo.

What about `M`? Lists should in general not be searchable, so infinite lists shouldn't be too... but that's a dangerous line of reasoning.

As for Π, I'd guess that, even though $\mathbb{N}_\infty$ and $\mathbf{2}$ are searchable, the searchability of $\mathbb{N}_\infty \to \mathbf{2}$ is taboo.

TODO: check the two above paragraphs. □

A good question yet to be answered: does there purely exist a type that is not searchable? At first one might think about the universe $\mathcal{U}$, but at a second glance one may think the universe should satisfy some kind of Rice's

theorem, so maybe all boolean predicates on it are constant... and thus the universe should be searchable. To answer this question, let's take a look at some classical axioms.

(Remark by Escardó: if there is $p : \mathcal{U} \to \mathbf{2}$ such that $p\ X \neq p\ Y$ for some $X, Y : \mathcal{U}$, then WLPO holds)

Theorem 1.5. Relation to excluded middle

1. $\prod A : \mathcal{U}.\mathtt{Searchable}\ A \to A + \neg A$
2. Conclusion: $(\prod A : \mathcal{U}.\mathtt{Searchable}\ A) \to \prod A : \mathcal{U}.A + \neg A$
3. Conclusion 2: $\neg \prod A : \mathcal{U}.\mathtt{Searchable}\ A$
4. $\mathtt{LEM} \simeq \prod P : \mathtt{Prop}.\mathtt{Searchable}\ P$
5. $\mathtt{LEM} \simeq \prod A : \mathcal{U}.\mathtt{MerelySearchable}\ A$

Proof. 1. Set $p\ \_ \equiv \mathtt{tt}$. Searchability then reduces to $(\sum x : A.\mathtt{tt} = \mathtt{tt}) + \prod x : A.\mathtt{ff} = \mathtt{tt}$, which is equivalent to $A + \neg A$.

2. Trivial.

3. From the contraposition of (2) using the fact that, given univalence, excluded middle for all types does not hold (see chapter 2 of the HoTTbook for this; TODO – give precise theorem number).

4. Both sides of the equivalence are propositions, so it suffices to give two implications. Right-to-left is the same as in point (2). Left-to-right: for a proposition P the type $\sum x : P.p\ x = \mathtt{tt}$ is a proposition, so we can check if it's inhabited using $\mathtt{LEM}$. If it is, we found what we we're looking for. If not, we can dervie $\prod x : P.p\ x = \mathtt{ff}$.

5. Both sides are propositions, so we need two implications. Right-to-left: as in (4). Left-to-right: using $\mathtt{LEM}$ we can decide if $\exists x : A.p\ x = \mathtt{tt}$ and if not, we can conclude $\forall x : A.p\ x = \mathtt{ff}$ using classical logic. □

Having seen some instances of searchable types, we should be a bit dissatisfied: all types we considered we're sets and all type formers we saw preserved the property of being a set. What about types with more complicated path structure?

Theorem 1.6. The circle $\mathbb{S}^1$ is searchable.

Proof. We have $p : \mathbb{S}^1 \to \mathbf{2}$ and we need to find an $x : \mathbb{S}^1$ such that $p\ x = \mathtt{tt}$ or prove that no $x : \mathbb{S}^1$ satisfies p.

Let's check the point $\mathtt{base}$. If $p\ \mathtt{base} = \mathtt{tt}$, then we found our point. If not, it should be quite obvious that $\prod x : \mathbb{S}^1.p\ x = \mathtt{ff}$, because all points are connected to $\mathtt{base}$ with a path. This path is truncated (i.e. $\prod x : \mathbb{S}^1.||x = \mathtt{base}||$), but this doesn't matter – we can take it out because we're proving $p\ x = \mathtt{ff}$ which is a proposition. □

The above proof looks a bit suspicious: there is nothing particular about the circle in it. It could just as well work for any type that has a point connected to all other points... such types are called connected (HoTTBook, remark 3.11.2).

$\texttt{Connected}\ A :\equiv \sum x : A. \prod y : A.||x = y||$

Theorem 1.7. Searchability of connected types.

1. $\prod A : \mathcal{U}.\texttt{Connected}\ A \to \texttt{Searchable}\ A$

2. Conclusion: the circle $\mathbb{S}^1$, the sphere $\mathbb{S}^2$, the n-sphere $\mathbb{S}^n$ are searchable.

2.5. There is a searchable type that is not connected.

3. If A is searchable, then it may be the case that the searchability of $x =_A y$ is taboo.

4. If A is inhabited, then the suspension $\Sigma\ A$ is searchable. If not, then it's also searchable.

Proof. 1. A is connected so we have $x : A$ and $H : \prod y : A.||x = y||$. Given a predicate $p : A \to \mathbf{2}$, let's check $p\ x$. If it's $\texttt{tt}$, then we're done. If not, use the path unpacked from $H\ y$ to show that all $y : A$ satisfy $p\ y = p\ x$ and thus $p\ y = \texttt{ff}$.

2. These types are connected.

2.5. $\mathbf{2}$ is one such type.

3. $\mathbb{S}^1$ is searchable, but it's fundamental group is $\mathbb{Z}$, whose searchability is taboo (just like for naturals).

4. If A is inhabited, $\Sigma\ A$ is connected and thus searchable. If A is empty, then $\Sigma\ A \simeq \mathbf{2}$, so it's also searchable. □

A question: can we prove that the suspension of A is always searchable, no matter what A is? Maybe we can use truncations for that.

Theorem 1.8. Searchability of truncations.

1. If A is searchable, $||A||$ is searchable.

2. The converse need not hold.

Proof. 1. To search $||A||$ with $p : ||A|| \to \mathbf{2}$, search A with the predicate $q\ x :\equiv p\ |x|$.

2. $||\mathbb{N}||$ is trivially searchable, but searchability of $\mathbb{N}$ is taboo. □

The proof of theorem 1.8.1 should hint us at an obvious generalization (if the fact that images of searchable types are searchable didn't hint us at it yet).

$\texttt{isSurjective}\ f :\equiv \forall b : B.\exists a : A.f\ a = b$

If there's a surjection from A to B, it means A has "more" points than B. More geometrically we could say that the points of A cover the points of B.

Theorem 1.9. Searchability and surjections
If A is searchable and $f : A \to B$ is surjective, then B is searchable.

Proof. If we want to search B with $p : B \to \mathbf{2}$, we can search A with $q\ a :\equiv p\ (f\ a)$. If we found an a, then we know that $f\ a$ is the element of B we're looking for. Otherwise for any $b : B$ we have an a such that $f\ a = b$, so from $\forall a : A.p\ (f\ a) = \mathtt{ff}$ we can conclude $\forall b : B.p\ b = \mathtt{ff}$.

□

We are now ready to prove that suspension is always searchable.

Theorem 1.10. Suspension is searchable
1. There is a surjection $\mathbf{2} \to \Sigma\ A$.
2. $\Sigma\ A$ is searchable.

Proof. 1. Let's define a function:

$$\begin{aligned} &f : \mathbf{2} \to \Sigma\ A \\ &f\ \mathtt{tt} = \mathtt{N} \\ &f\ \mathtt{ff} = \mathtt{S} \end{aligned}$$

It remains to prove that $\forall y : \Sigma\ A.\exists x : \mathbf{2}.f\ x = y$. We do this by induction on y:
If y is `N`, we choose `tt`.
If y is `S`, we choose `ff`.
If y varies along `merid` a for some $a : A$, we need to give a path between proofs of $\exists$, but that's easy.
2. $\mathbf{2}$ is searchable and it covers $\Sigma\ A$, so $\Sigma\ A$ is searchable.

□

But there's more to it.

Theorem 1.11. Pushouts and pullbacks
1. If A and B are searchable, their pushout (for any $f : C \to A$ and $g : C \to B$) is searchable.
2. If A and B are searchable and C has decidable equality, then the pullback of $f : A \to C$ and $g : B \to C$ is searchable.

Proof. 1. The pushout is covered by the coproduct $A + B$.

2. To search the pushout with the predicate p, we may search $\prod AB$ with the predicate

$$q : A \times B \to \mathbf{2}$$
$$q\ (a, b) :\equiv p\ (a, b)\ \texttt{and}\ f\ a = g\ b$$

If we find some (a, b), we know it's ok, because if must satisfy $f\ a = g\ b$. If we didn't find anything, it means all (a, b) with $f\ a = g\ b$ fail to satisfy p, and these are all elements of the pullback.

□

As a closing, a trivial remark.

Theorem 1.12. Constructive logic
$\prod(A : \mathcal{U})(p : A \to \mathbf{2}).\neg\neg\,((\sum x : A.p\ x = \texttt{tt}) + \prod x : A.p\ x = \texttt{ff})$

Proof. Similar to the proof of doubly-negated excluded middle. □

One research question I was exploring, but that I couldn't answer in any way: how to turn any type A into a searchable type A' such that A embeds in A'?

2 Separatedness

Homotopy type theory is about paths – ways in which things can be (considered) "the same". But what about ways of differing? These are not as developed, but one manner of capturing them is the notion of separatedness.

S-`Separated` $A :\equiv \prod x\ y : A.\,(\prod p : A \to S.p\ x = p\ y) \to x = y$

In words: the type A is S-`Separated` (or, maybe, more colloquially: S separates the points of A) if any two points of $x, y : A$ are connected with a path as soon as they can't be distinguished using any function with codomain S.

The name "separated" comes from the contraposition of this statement: if the points $x, y : A$ are different, then they can be **separated** by a function with codomain S.

Theorem 2.1. Instances

1. If A and B are S-`Separated`, then $A + B$ is S-`Separated`.
2. If A and B are pointed and S-`Separated`, then $A \times B$ is S-`Separated`.
3. If A is pointed and S-`Separated` and $B : A \to \mathcal{U}$ is a family of pointed S-`Separated` types, then $\sum x : A.B\ x$ is S-`Separated`.

4. If $B : A \to \mathcal{U}$ is a family of S-`Separated` types, then $\prod x : A.B\ x$ is S-`Separated`.

5. A is A-`Separated`.

Proof. 1. Assume $x, y : A + B$ and $\prod p : A + B \to S.p\ x = p\ y$. We inspect the form of x and y. If they are of the form `inl` a and `inl` a', then we can get a path from the separatedness of A (if there was a predicate that can distinguish a and a', we could turn it into a predicate for distinguishing x and y, a contradiction). Similarly for `inr`. If x is `inl` and y is `inr` (or the other way around), then we get a contradiction, because we can make a predicate that distinguishes them.

2. Assume $x \equiv (a, b) : A \times B$ and $y \equiv (a', b') : A \times B$ and $\prod p : A \times B \to S.p\ x = p\ y$. We need to make a path $(a, b) = (a', b')$ for which we need paths $a = a'$ and $b = b'$. We can get $a = a'$ from the separatedness of A using the basepoint of B (and analogously for B).

3. Similar to (2).

4. Assume we have $f, g : \prod x : A.B\ x$ and $H : \prod p.p\ f = p\ g$. We need to make a path $f = g$ which amounts to $f\ a = g\ a$ for all $a : A$. We can use the separatedness of $B\ a$ for this. It remains to prove that given $p : B\ a \to S$, we have $p\ (f\ a) = p\ (g\ a)$. Let's define

$$\begin{aligned} &q : (\textstyle\prod x : A.B\ x) \to S \\ &q\ h :\equiv p\ (h\ a) \end{aligned}$$

Then $H\ q$ gives us $q\ f = q\ g$ which amounts to $p\ (f\ a) = p\ (g\ a)$ – exactly what we needed to prove. □

Having seen these instances we may start to wonder if there are any concrete types that are separable using some other concrete types. First we will establish which types are good for separating and then we'll see some separated types.

5. Given $x, y : A$ and $H : \prod p : A \to A.p\ x = p\ y$, we need $x = y$. We can get it easily using H `id`.

Theorem 2.2. Silly separatedness

1. **0**-`Separated` = `isProp`
2. **1**-`Separated` = `isProp`
3. `isProp` $S \to S$-`Separated` = `isProp`

Proof. 1. We have $(p\ x = p\ y) \simeq \mathbf{1}$, therefore $(\prod p : A \to S.p\ x = p\ y) \simeq \mathbf{1}$, so the premise is irrelevant and what remains is $x = y$.

2. Similar to (1).
3. Similar to (1). □

The above may look silly, but basically it says that propositions have "not enough proofs" to separate anything besides other propositions, which have at most one proof, so there's nothing to be separated...

The first nontrivial separator appears to be the type **2** of booleans. **2**-`Separated` types are also called totally separated (or at least Escardó says so) – this name comes from topology, where continuous maps into **2** represent clopen sets. Let's see what we can separate using **2**.

Theorem 2.3. **2**-`Separated` types
1. $\mathbb{N}$ is **2**-`Separated`.
2. If A has decidable equality, then it is **2**-`Separated`.
3. $\mathbb{N}_\infty$ is **2**-`Separated`.
4. There are **2**-`Separated` types whose decidable equality is taboo.

Proof. 1. We have $x, y : \mathbb{N}$ and for all $p : \mathbb{N} \to \mathbf{2}$ we have $p\ x = p\ y$. We need to prove that $x = y$. Let's define a p:

$$p : \mathbb{N} \to \mathbf{2}$$
$$p\ n :\equiv x =?n$$

From our assumption we have $p\ x = (x =?x) = \mathtt{tt}$ and also $p\ y = (x =?y)$, so $(x =?y) = \mathtt{tt}$ which means that x and y are equal.

2. Similar to (1).

3. For this one we use Cubical Type Theory/assume that bisimilarity is equality.

We have $n, m : \mathbb{N}_\infty$ and $\prod p : \mathbb{N}_\infty \to \mathbf{2}.p\ n = p\ m$ and we need to construct a proof of bisimilarity $n \sim m$.

We check the predecessors of n and m. If one of them is zero and the other is not, they are different and we can find an appropriate p to get a contradiction. If both are zero, they are bisimilar. If both have predecessors, we take a step in the bisimilarity and use our coinduction hypothesis. It remains to prove that $\prod p : \mathbb{N}_\infty \to \mathbf{2}.p\ n' = p\ m'$ where $n', m' : \mathbb{N}_\infty$ are the predecessors of n and m. Assume for contradiction that $p\ n' \neq p\ m'$ (this is constructively ok). This means we can make a predicate that distinguishes their successors n and m, so we can use our original assumption $\prod p : \mathbb{N}_\infty \to \mathbf{2}.p\ n = p\ m$ to get a contradiction.

4. Conclusion from (3).

□

It's quite intuitive that types with decidable equality are **2**-`Separated` – we can use an equality test to separate nonequal points and the specification of the equality test gives us a path in case the points are equal.

But it turns out that **2**-`Separated` is a more general concept than having decidable equality – $\mathbb{N}_\infty$ is **2**-`Separated`, but its decidable equality is taboo. Can we somehow constrain which types can be **2**-`Separated`?

Theorem 2.4. **2**-`Separated` types are sets

1. If A is **2**-`Separated`, then $\prod x\ y : A.\neg\neg x = y \to x = y$.
2. If A is **2**-`Separated`, then A is a set.
3. If we have `LEM`, then all sets are **2**-`Separated`.

Proof. 1. Assume all the needed stuff. We use hypothesis to get $x = y$, so we only need to prove $p\ x = p\ y$ for all $p : A \to \mathbf{2}$. Assume for contradiction that $p\ x \neq p\ y$. This gives us $\neg x = y$, because if $x = y$, then $p\ x = p\ y$ which is a contradiction. We therefore have both $\neg x = y$ and $\neg\neg x = y$, a contradiction.

2. From (1) we know that **2**-`Separated` A satisfies $\prod x\ y : A.\neg\neg x = y \to x = y$ and such types are sets by a theorem due to Kraus, Escardó, Coquand and Altenkirch.

3. With `LEM` all sets have decidable equality and all sets satisfy $\prod x\ y : A.\neg\neg x = y \to x = y$, so all sets are **2**-`Separated`. □

A question that arises instantly after seeing 2.4.3 is this: if we don't have `LEM`, what is the type that can separate all sets? If we recall that `LEM` is equivalent to saying that `Prop` $= \mathbf{2}$, we may conjecture that it's `Prop` that separates all sets.

Theorem 2.5. Separation of sets and n-types

1. All sets are `Prop`-`Separated`.
2. Every type A is separated by some universe $\mathcal{U}$.
3. If A is an (n + 1)−`Type`, then A is (n−`Type`)-`Separated`.
4. If S is an (n + 1)−`Type`, then S-`Separated` is an n−`Type`.
5. If S is an (n + 1)−`Type`, then $\prod x\ y : A.||x = y||_n \to \prod p : A \to S.p\ x = p\ y$.

Proof. 1. Assume that A is a set. Given $x, y : A$ and $H : \prod P : A \to$ `Prop`$.P\ x = P\ y$ we need to prove $x = y$. Let's define $P\ a :\equiv (x = a)$. From H we get $(x = x) = (x = y)$, and from this using `refl` x we can get $x = y$.

2. Just like (1).

3. From (2) using the fact that if A is (n + 1)−`Type` then each of its identity types is an n−`Type`.

4. This should be more than obvious.
5. $p\ x = p\ y$ is an n$-$`Type`, so we can unpack $||x = y||_n$ and use it.

□

Knowing this, we can now ask ourselves more questions: is `Prop` needed to separate sets, or can we do better (using something less "massive")? It should be obvious that we need a type with at least two elements, since one element won't be enough, and that without `LEM`, $\mathbf{2}$ is not enough. Are there any other 2-element types that could be of any help?

Well, there is – the type $\mathbf{1}_\bot$ of partial elements of unit. Functions of type $A \to B_\bot$ represent partial functions from A to B, i.e. things that take an element of A and may compute an element of B, but may also diverge (fail to terminate). The type $\mathbf{1}_\bot$ has two elements, one of which represents pure termination that doesn't carry any information, the other represents pure nontermination that also does not carry any other information.

(Fun fact: the type $\mathbf{1}_\bot$ is sometimes called the Sierpiński space, and functions $A \to \mathbf{1}_\bot$ represent the open sets of A, if we think of A as a topological space).

This type was described in the paper "Partiality revisited: The Partiality Monad as a Quotient Inductive-Inductive Type". I won't cover it in much more detail because I didn't think about it yet.

Let's try to get some profits from the facts relating separatedness to h-levels.

Theorem 2.6. $\mathbb{S}^1$ and $\mathbf{2}$
1. $\mathbb{S}^1$ is not $\mathbf{2}$-`Separated`.
2. $\mathbf{2}$ is $\mathbb{S}^1$-`Separated`

Proof. 1. $\mathbf{2}$-`Separated` types are necessarily sets, but $\mathbb{S}^1$ is not a set.

2. Given $x, y : \mathbf{2}$ and $H : \prod p : \mathbf{2} \to \mathbb{S}^1.p\ x = p\ y$ we need $x = y$. We do case analysis. If both x and y are `tt` (or `ff`), then we're done. If they are different, we can prove $\prod s\ t : \mathbb{S}^1.s = t$ by defining

$$\begin{aligned} &p : \mathbf{2} \to \mathbb{S}^1 \\ &p\ \texttt{tt} = s \\ &p\ \texttt{ff} = t \end{aligned}$$

and then $H\ p : s = t$. This is a contradiction, because we have just proved that $\mathbb{S}^1$ is a proposition, and it is not.

We have covered all four cases of x and y, so we're done.

□

We see that the notion of separatedness is weirder than we could have previously thought. $\mathbb{S}^1$ can separate the points of $\mathbf{2}$, but not because functions with codomain $\mathbb{S}^1$ can distinguish the points of $\mathbf{2}$ – rather it's just because the hypothesis $\prod p : \mathbf{2} \to \mathbb{S}^1.p\ x = p\ y$ in some cases implies that $\mathbb{S}^1$ is a proposition.

`Merely-`S`-Separated` $A :\equiv \prod x\ y\ :\ A.(\prod p : A \to S.||p\ x = p\ y||) \to ||x = y||$