HoTT lecture 6 Q&A (Solved)

HoTTEST Summer School 2022

The HoTTEST TAs

25th July 2022

Q1 "So if you include the U and V types you have to also include other types? (In the successor of $U \sqcup V$)"

"There are embeddings from U and V, so you necessarily have other types than U and V in the successor of the join. Does that answer your question?"

"Then why are there necessarily embeddings from U and V, and what are embeddings?"

"Any type in U or V also has a code in the join $U \sqcup V$, and any type in a universe also has a code in the successor universe. That's what I mean by embedding. (We haven't defined what embeddings are yet.)"

"ok I'll look up what embeddings are in the book. Thx"

"Okay. You can think of the inclusion of path components in a space. Another name for them are (-1)-truncated maps, which is another way of saying "the fibres are propositions" i.e. the fibres have at most one element."

Q2 "universes don't a priori form a cumulative hierarchy, then?"

"We won't assume that codes in one universe are also codes in a higher universe judgmentally"

"So each universe contains a priori different codes for Nat, for example"

"Yes, this will be the case in Agda as well"

"maybe let me refine my question then. given universes U, V, we don't assume there is either an embedding $U \to V$ or an embedding $V \to U$, sending codes to codes in an appropriate way?"

"Egbert does not assume this in his book because the universes are not totally ordered, but if there is an inclusion, then such a map can be defined using inductive datatypes and is called Lift"

"hm ok, thanks for clarifying"

Q3 "What is the definition of a type family?"

"Just a type parametrized by variables of other types. It's explained very well in the beginning of the book."

"It may be explained well for those who understand, but I couldn't find a definition. And I couldn't find one in Wikipedia, either. What would be the definition?"

"pp 5. in the book, specifically defn 1.2.1 is a place to start. hth"

"*pp. 5"

"A type familly may be defined to be a function of several arguments that maps into the universe; thus, for every set of parameters, you get a type"

"Yes, exactly there. We can chat about this on the discord later"

"A is a type family in context Γ iff $\Gamma \vdash$ A type."

"It's also been discussed in Agda exercise 1."

"Thank you, I found the definition on page 3 of the 2019 version, it was "family of types""

Q5 "So "contractible" and "singleton" are synonyms in HoTT?"

"Essentially. "Contractible" is a technical concept representing the informal idea that a type is a singleton"

Q6 "If there is an element in is-contr(A), then shouldn't there be an element for every c:A? Why do we need to take the sum?"

"Not sure I'm understanding. The sigma is used to express "there exists a center c" which everything is equal to. We need to pick a particular center"

"The sum guarantees that A is inhabited, that it has at least one element (that is, the center c). If you just ask that for every x, y : A, x = y, then A could be empty. So contractibility is both existence and uniqueness"

"$\Sigma_{a:A}\Sigma_{x:A}a = x$ reads as "there is an element a in A such that every other element is equal to it" in set theory, or as "there is a *point* a in A such that for every other point there is a *path* from a to x" in HoTT"

Q7 "Can we define is-contr(A) := Pi x, Pi y, x = y?"

"This is is-prop(A)"

"A proposition may not be inhabited and that definition does not guarantee that the domain of quantification is non-empty"

"The empty type is an example of a type that satisfies this condition, but it has no center of contraction."

"Ah okay thanks! I forgot the edge case of empty types"

"so every proposition is contractible then?"

“Every inhabited proposition“

“ok“

Q8 ““Propositions” are just 0 or 1 element types, there is no any logic intuition behind it?”‘

“In the Curry-Howard interpretation they correspond to false and true“

“Non-casically, propositions form a lattice that may be more complicated than false,true“

“But the proposition $\Sigma_{x:A}\Pi_{y:A}x = y$ is not a proposition in this sense. And, vice versa, a set of all even prime numbers is a proposition.”

“No, we only have forall n, is-prop(is-prime(n))“

“Propositions are types that have at most one element. This is a way to say that they are “proof irrelevant”, i.e., that once they’re proven, there is no further content to them. The situation is the same in logic, where the rules of (first order) logic only give rules about provability, i.e., they don’t keep track of how propositions are proven, but only whether they are proven.“

“I get the idea, but it seems weird that the set of even prime numbers is a proposition but the set of odd prime numbers is not a proposition. It has nothing to do with logic in this case :(“

“Think also of things like “it’s raining” and “there is water on the sidewalk”. Neither is universally true, but the second follows from the first, but not vive-versa“

“Right, the set of even prime numbers would be a proposition“

“The set of even prime numbers is the proposition “There exists an even prime number” and this proposition is true“

"actually the set of even prime numbers is contractible, since it is an inhabited propostion. that aligns very well with how contractible types represent uniqueness: there is a unique even prime number"

"But the set of odd prime numbers is not a "proposition", while it corresponds to the proposition that there exists an odd prime number"

"in general, the *proposition* "there exists x such that P x" would be expressed as the propositional truncation $\|\Sigma_{x:X} Px\|$ and the non-truncated type $\Sigma_{p:\mathbb{N}} pprime \wedge peven$ happens to be a proposition, for reasons that indeed don't have much to do with logic"

"'@mateusz we might get there in a future lecture, but this is what propositional truncation is for. basically you take a type and forget about its inhabitants"

Q9 "what is the definition of $const_c$? and I feel in the observation, this $! = const_*$ should be an definition, but what is the symble of equality type"

"$const_c := \lambda x.c$ $! := const_*$"

"$const_c$ is the constant-c function: it has whatever domain you want, and returns c for every input"

"I think the ! is notating the unique (up to homotopy) such map — b1 is the terminal type, by its induction principle"

"Yep, I have used ! in my type theory fromalisations to reffer to things such as the substitution into the empty context (which is unique)"

"Yes, for any type A, ! is the unique map $A \to 1$. You can prove this using funext and the induction principle for 1"

Q10 "Why the circle from the Agda track not contractible? Taking only one point, base, and base = base by refl?"

"I gave a discussion of this at the end of my third Agda tutorial"

"Here is a brief summary:"

"Thanks, I will visit it then¡'

"Suppose you want to connect every point to base, how would you do this? Well first you could say "by the shortest path", but that definition is discontinuous as you move past the antipodal point"

"I'm talking about this topologically"

"In general, the induction principle does not say that every point of the circle is base"

"It just says that, in order to construct a map out of S1, that you need to say how that map acts on base and loop"

"(also one of the exercises for today's lecture will be proving that if A is contractible, then x=y is contractible for all x,y:A. The type base=base is not contractible)"

"Topologically, similarly, "go clockwise" is also discontinuous as you move the other point past base"

"In Cubical Agda, there are other points of the circle apart fom base; if i is an interval variable, then loop i : S1"

"I get it topologically but I need to understand it formally"

"Formally, just try to construct a term of type Pi(x : S1) base = x from the circle induction principle."

"Right, great. So when doing that induction, the case for base is easy (just loop or refl), but the case for loop is much harder (and we haven't discussed the dependent elimination principle for S1)"

"That second case makes it impossible, in fact"

Q12 "Could also give a proof of the right-to-left direction by choosing $B(a) := A$, with the second piece of data about singleton induction $(eval - at(a).const(a)\ 1_A)$ witnessing contractibility of A?"

"I think that we moved too far past this point to recall what the context is; sorry"

"Please ask this on discord, I'll address it later if noone has answered it by then"

Q13 "So if b is not actually in the image of f, $fib_f(b)$ will be empty?"

"Yes"

"How will we say b is not in the image of f? The only way I could think of is saying that $fib_f(b)$ maps to the empty type."

"Yep, that's how we would say that $fib_f(b)$ is not inhabited."

Q15 "What is tt-choice ?"

"It is the fact that Pi-types distribute over Sigma-types. When you read that distributivity by the Curry-Howard interpretation, it reads like the axiom of choice."

"(It's short for "the type-theoretic axiom of choice".)"

Q16 “is that the same as a half-adjoint equivalence?”

“is that the same as a half-adjoint equivalence?”

“Yep!”

Q17 “Does the K hypothesis mean that a coherently invertible map is more than an isomorphism?”‘

“We will show that being coherently invertible is a proposiiton equivalent to the proposition of being bi-invertible“

“Bi-invertibility doubles up the inverse on each side, while coherently invertible deals with the problem that quasi-invertibility is not a proposition by imposing a coherence“