Category Theory Scribe Notes

Lecturer: Max S. New
Scribe: Benjamin Kelly

March 8, 2023

1 A Recapped Question

Is the equational theory of STT consistent?

That is, is it possible that $\cdot \vdash i_1() = i_2() : 1 + 1$? Here, 1+1 represents a "boolean" type with $i_1()$ corresponding to "true" and $i_2()$ corresponding to "false." We want to show that it isn't possible a program can't be equal to both "true" and "false" simultaneously. We've previously shown that this is not possible by using a set theoretic model. We can now view that proof as an instance of our soundness theorem.

We constructed

I) A Bi-cartesian closed category of sets

II) A C-T structure (self Set)

III) $\mathcal{L}$ the "syntactic structure" with

$$\mathcal{L} \xrightarrow{[\![\cdot]\!]} \text{self Set}$$

where $[\![\cdot]\!]$ is a homomorphism that preserves all structures (products, exponentials, etc...) Then $i_1() \in \text{Tm}_{\mathcal{L}}(1+1)(\cdot)$ and $i_2() \in \text{Tm}_{\mathcal{L}}(1+1)(\cdot)$. So we get that

$$[\![i_1()]\!] \in \text{Tm}_{\text{self(Set)}}(\{(1,*),(2,*)\})(\{*\}) = \text{Set}(\{*\},\{(1,*),(2,*)\})$$

$$[\![i_2()]\!] \in \text{Set}(\{*\},\{(1,*),(2,*)\})$$

$$[\![i_1()]\!](*) = (1,*)$$

$$[\![i_2()]\!](*) = (2,*)$$

Since $[\![i_1()]\!](*) \neq [\![i_2()]\!](*)$, and by definition $[\![\cdot]\!]$ respects the equational theory, we know that the equivalence classes $[i_1()] \neq [i_2()]$ and therefore $i_1() = i_2()$ is not provable in the equational theory.

2 Mathematical Interpreters for the Language

We can think of the homomorphism $[\![\cdot]\!]$ defined above as a "mathemtical intrepreter" of STT, which we will call f, where f is of type

$$f : \{\cdot \vdash M : 1+1\} \to \{(1,*),(2,*)\}$$

where $f(M) = [\![M]\!](*)$. How should such an interpreter behave? We desire (and have already shown) that $M = M' \Rightarrow f(M) = f(M')$. However, we don't know if f made any arbitrary choices when evaluating M and M'. So, ideally we would desire a kind of converse:

$$f(M) = (1,*) \Rightarrow \cdot \vdash M = i_1() : 1+1$$
$$f(M) = (2,*) \Rightarrow \cdot \vdash M = i_2() : 1+1$$

That is, if the interpretation of a term is $(1,*)$, it is equivalent to "true" and the same with "false." That is, we desire that

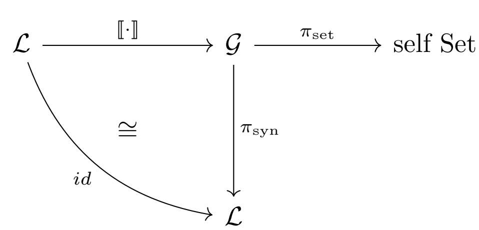
$$f(M) = (1,*) \Leftrightarrow \cdot \vdash M = i_1() : 1+1$$
$$f(M) = (2,*) \Leftrightarrow \cdot \vdash M = i_2() : 1+1$$

Such a result is called *canonicity* for STT because it says every closed boolean term is equal to a "canonical" one: true or false.

Think about what things would be like if this property weren't true: we would have that there is a term M that is not provably equivalent to true or false. So we wouldn't be able to predict how an interpreter would behave for this term. In a sense, this would be *undefined behavior* or at the very least *implementation-dependent*. So canonicity is a kind of ensures that defining an interpreter for STT is "fully specified" by the equational theory.

3 The Method of Logical Relations

To prove the desired result, we will use the method of "logical relations," which is also known by many other names: reducibility candidates, Tait's method of computability, and Artin gluing. Main idea is to construct a semantics where we co-construct evaluator and simultaneously draw relation between result of semantics. We will define a C-T structure $\mathcal{G}$ where the types have a set component and an STT semantic component. The idea is to construct $\mathcal{G}$ and functors of CT structures as fitting into the following diagram:



That is,

- The structure $\mathcal{G}$ will have projection functors $\pi_{\mathcal{L}}$ to $\mathcal{L}$ and π_{selfSet} to selfSet.
- $\mathcal{G}$ will model all types in STT, and the projection π_{syn} will preserve this type structure.
- Therefore by the soundness theorem, we will have a homomorphism $[\![\cdot]\!] : \mathcal{L} \to \mathcal{G}$ that preserves type structure.
- By the completeness theorem, since the composition of π_{syn} and $[\![\cdot]\!]$ preserves type structure, it is naturally isomorphic to the identity functor from $\mathcal{L}$ to $\mathcal{L}$.

4 Defining $\mathcal{G}$, the "Glued" CT Structure

The basic idea of $\mathcal{G}$ is that everything in it will consist of something in the set theoretic structure, something in the syntactic structure and a relation between the two. For instance,

4.1 Types of $\mathcal{G}$

A type $\hat{A} \in \mathcal{G}_T$ consists of

I) A set $\hat{A}_S$

II) An STT type $\hat{A}_{\text{ty}}$

III) A function $\hat{A}_P : \hat{A}_S \to \mathcal{L}_{\text{ty}}(\hat{A_{\text{ty}}})\cdot$. Also can call the codomain of this function the terms of type $\hat{A_{\text{ty}}}$ in the closed context $\text{Cl}(\hat{A_{\text{ty}}})$.

We think of the set $\hat{A}_S$ as an interpretation of a type, but we also include a function $\hat{A}_P$ which "reads back" a corresponding closed syntactic term of type $\hat{A}_{\text{ty}}$.

4.2 Contexts of $\mathcal{G}$

A context $\hat{\Gamma} \in \mathcal{G}_C$ consists of

I) A set $\hat{\Gamma_S}$

II) An STT context $\hat{\Gamma_C}$

III) A function $\hat{\Gamma_P} : \hat{\Gamma_S} \to \{\gamma : \cdot \to \hat{\Gamma_C}\}$. The codomain is the set of substitutions into the closed context, called $\text{Cl}(\hat{\Gamma_C})$.

4.3 Terms of $\mathcal{G}$

A term $\hat{M} \in \mathrm{Tm}_{\mathcal{G}}\ \hat{A}\ \hat{\Gamma}$ consists of

I) A function $\hat{M}_f : \hat{\Gamma_S} \to \hat{A_S}$

II) An STT term $\hat{M}_t$ such that $\hat{\Gamma_C} \vdash \hat{M}_t : \hat{A_{\mathrm{ty}}}$

III) And the following diagram commutes

$$\begin{array}{ccc} \hat{\Gamma_S} & \xrightarrow{\hat{M}_f} & \hat{A_S} \\ \big\downarrow{\scriptstyle \hat{\Gamma_P}} & & \big\downarrow{\scriptstyle \hat{A_P}} \\ \mathrm{Cl}(\hat{\Gamma_C}) & \xrightarrow{\hat{M_{\mathrm{ty}}}[\cdot]} & \mathrm{Cl}(\hat{A_{\mathrm{ty}}}) \end{array}$$

That is for any semantic context $\hat{\gamma} \in \hat{\Gamma}_S$, we get the same result if we first run the semantic function $\hat{M}_f(\hat{\gamma})$ and then read it back as a closed term $\hat{A}_P(\hat{M}_f(\hat{\gamma}))$ as if we first turn it into a closing substitution and then substitute it into the term $\hat{M}_t$: $\hat{M}_t[\hat{\Gamma}_P(\hat{\gamma})]$.

4.4 Substitutions of $\mathcal{G}$

A substitution $\hat{\gamma}$ of $\mathcal{G}\ \hat{\Delta}\ \hat{\Gamma}$ consists of

I) A function $\hat{\gamma_f} : \hat{\Delta_S} \to \hat{\Gamma_S}$.

II) An STT substitution $\hat{\gamma_S} : \hat{\Delta}_C \to \hat{\Gamma_C}$.

III) Such that the following diagram commutes:

$$\begin{array}{ccc} \hat{\Delta_S} & \xrightarrow{\hat{\gamma_f}} & \hat{\Gamma_S} \\ \big\downarrow{\scriptstyle \hat{\Delta_P}} & & \big\downarrow{\scriptstyle \hat{\Gamma_P}} \\ \mathrm{Cl}(\hat{\Delta_C}) & \xrightarrow{\hat{\gamma_S} \circ -} & \mathrm{Cl}(\hat{\Gamma_C}) \end{array}$$

4.5 Proving Properties of $\mathcal{G}$

Lastly, we need to prove some properties and make definitions about $\mathcal{G}$. Namely:

I) $\mathcal{G}_c$ is cartesian.

II) Define sole for $\mathcal{G}$.

III) Define all connective types (products, co-products, etc...)

Luckily, however, these definitions follow naturally from the universal properties of $\mathcal{G}$ and so their precise definitions and proofs are not fully written here.

We can also clearly define π_{Set} by projecting out the set theoretic (I) component of each structure, and similarly $\pi_{\mathcal{L}}$ by projecting out the STT component (II). This second projection $\pi_{\mathcal{L}}$ preserves all of the type structure as well (π_{Set} on the other hand, does not preserve function types).

4.6 Proving Canonicity

With just a few details of the construction of $\mathcal{G}$, we can prove the canonicity theorem.

First, $[\![\cdot]\!] : \mathcal{L} \to \mathcal{G}$ is a homomorphisms of CT structures that preserves all type structure of STT. Further, $\pi_{\mathcal{L}} \circ [\![\cdot]\!]$ preserves the type structure exactly, and so the isomorphism from the completeness theorem is the identity and we get that for any type A, $\pi_{\mathcal{L}}([\![A]\!]) = A$ and for any term M, $\pi_{\mathcal{L}}([\![M]\!]) = M$. In this sense, we get that each term M is mapped to a function M_f whose behavior tells us about M.

- The empty context $\cdot$ gets mapped to the unique glued context $(\{*\}, \{\cdot\}, f)$.
- The type $1 + 1$ gets mapped to $(\{(1, *), (2, *)\}, 1 + 1, g)$ where

$$g(1, *) = [i_1()]$$
$$g(2, *) = [i_2()]$$

- $[\![[i_1()]]\!] = (x \mapsto (1, *), [i_1()])$
- $[\![[i_2()]]\!] = (x \mapsto (2, *), [i_2()])$

Since $[i_1()]$ and $[i_2()]$ have different function components, this means they cannot be equal, and so $i_1() = i_2()$ is not provable in the equational theory, so we get another proof of consistency.

Next, for any $\cdot \vdash M : 1 + 1$, we get $\pi_{\text{Set}}[\![[M]]\!](*) \in \{(1, *), (2, *)\}$ and

$$[g(\pi_{\text{Set}}[\![[M]]\!](*))] = [M[\cdot]] = [M]$$

that is, that the equivalence class of M is the same as the equivalence class of $g(\pi_{\text{Set}}[\![[M]]\!](*))$. Since the only possible outputs of f are $[i_1()]$ and $[i_2()]$ this proves that either $M = i_1()$ or $M = i_2()$ is provable in the equational theory. Further, by canonicity this is an exclusive or. So we get

$$g(\pi_{\text{Set}}[\![[M]]\!](*)) = (1, *) \iff M = i_1() \text{ is provable}$$
$$g(\pi_{\text{Set}}[\![[M]]\!](*)) = (2, *) \iff M = i_2() \text{ is provable}$$

And so we have reduced the correctness of a mathematical evaluator and the canonicity result to showing the properties of $\mathcal{G}$ mentioned above.