Lecture 5: Signatures for STT, Set-theoretic Semantics

Lecturer: Max S. New
Scribe: Chris Jiang

Jan. 25, 2023

1 More Simple Type Theory

In STT, we have the following admissible rule, similar to the principal of substitution from IPL:

$$\frac{\Gamma \vdash M : A \qquad \Gamma, x : A \vdash N : B}{\Gamma \vdash N[M/x] : B} \text{ subst}$$

Since STT is an equational theory, we also want the corresponding congruence rule for substitution, which says that substitution preserves equality:

$$\frac{\Gamma \vdash M = M' : A \qquad \Gamma, x : A \vdash N = N' : B}{\Gamma \vdash N[M/x] = N'[M'/x] : B} \text{ subst cong}$$

If we put more effort into developing this theory, we could ensure that this congruence rule is admissible, but for the purposes of this class, we will treat this rule as a primitive rule.

This rule is useful in combination with the η rules for the 0 and + types. For example, consider the 0η rule:

$$\frac{x : 0 \in \Gamma \qquad \Gamma \vdash M : C}{\Gamma \vdash M = \textbf{case}\ x\{\} : C}\ 0\eta$$

This rule intuitively says that if there is a variable of type 0 in our context, then intuitively, our context is inconsistent, so any term M is just an empty case split, and so all terms are equal. However, it turns out that we don't actually need a variable of type 0 in our context to conclude this. Rather, it is sufficient to show that it is possible to derive a term of type 0 from our context. To be precise, we have the following rule:

$$\frac{\Gamma \vdash N : 0 \qquad \Gamma \vdash M : C}{\Gamma \vdash M = \textbf{case}\ N\{\} : C}$$

To prove the above rule, we can use substitution congruence:

$$\dfrac{\dfrac{\Gamma \vdash N : 0 \quad \Gamma \vdash N : 0}{\Gamma \vdash N = N : 0}\ \text{refl} \qquad \dfrac{\dfrac{}{x : 0 \in \Gamma, x : 0}\ \text{Var} \qquad \Gamma \vdash M : C}{\Gamma, x : 0 \vdash M = \textbf{case}\ x\{\} : C}\ 0\eta}{\Gamma \vdash M = \textbf{case}\ N\{\} : C}\ \text{subst cong}$$

On the final line of the above proof, we are using the facts $M = M[N/x]$ and $\textbf{case}\ N\{\} = \textbf{case}\ x\{\}[N/x]$, where x is a variable not contained in M.

2 Signatures for STT

Recall that for IPL, we had two layers: propositions and proofs such as $\Gamma \vdash A$. We then extended our propositions using a set of propositional variables which we called Σ_0. We also added a set Σ_1 of axioms of the form $\Gamma \Rightarrow A$ to our proof system. Note that such axioms are generated by our propositional symbols. Combined, we call $\Sigma = (\Sigma_0, \Sigma_1)$ a signature for IPL.

We will do something similar for STT. At the first layer, IPL had propositions, extended by propositional variables. Similarly, at the first layer, STT has types, which we will extend by adding base types. At the second later, IPL had proofs, extended by axioms. Similarly, at the second later, STT has terms (such as $\Gamma \vdash M : A$), which we will extend by adding function symbols. Finally, STT has a third layer consisting of equality judgements (such as $\Gamma \vdash M = N : A$), which we will extend by adding equational axioms.

2.1 Base Types

Currently, we can only create types in STT using the basic connectives we have defined $(0, 1, +, \times, \Rightarrow)$. However, typical programming languages also have built-in types such as Integers, Natural Numbers, Strings, Arrays, Lists, Reals, Floats, etc. Therefore, we will parameterize STT with a set Σ_0 of base types. For example, we could let $\Sigma_0 = \{\mathbb{N}, \mathbb{R}\}$

We also add following rule so that these base types are indeed considered types in our system:

$$\frac{X \in \Sigma_0}{X\text{type}}$$

2.2 Function Symbols

Currently, our base types are just labels, and we can't do anything interesting with them. For example, we might let $\mathbb{N}$ be a base type representing the natural numbers, but nothing in our system indicates that $\mathbb{N}$ behaves like the natural numbers. To give our base types more structure, we add function symbols to our system. We define Σ_1 to be a set of function symbols of the form:

$$f : A_0, A_1, A_2, ... \to A'$$

where $A_0, A_1, ...$ and A' are types. We call the pair of the list $A_0, A_1, A_2, ...$ and the output type A' the *arity* of f.

Here are some examples of function symbols:

$$\text{zero} : \cdot \to \mathbb{N}$$

$$\text{one} : \cdot \to \mathbb{N}$$

$$\text{add} : \mathbb{N}, \mathbb{N} \to \mathbb{N}$$

$$i : \mathbb{N} \to \mathbb{R}$$

$$\sin : \mathbb{R} \to \mathbb{R}$$

To embed these function symbols into STT, we add a new rule for function symbol application so that our terms can include function symbols:

$$\frac{f : A_0, ... \to A' \qquad M_0 : A_0 \qquad ...}{f(M_0, ...) : A'}$$

So then we can write terms that involve these things such as

$$x : \mathbb{N} \vdash \text{add}(x, x) : \mathbb{N}$$

or

$$\cdot \vdash \lambda x : \mathbb{N}.\text{add}(x, \text{one}()) : \mathbb{N} \Rightarrow \mathbb{N}$$

A function symbol with no inputs such as zero or one above is sometimes called a *constant*, and the parentheses are often elided when using these as terms but we will stick to the very pedantic notation one().

2.3 Equational Axioms

Now that we have function symbols such as zero, one, and add, our base type $\mathbb{N}$ now looks more like the natural numbers we are familiar with. However, we are still missing equational logic. For example, zero and one both have the exact same arity, and our system doesn't include any logic to tell us that one of these symbols acts like the number 0 and the other acts like the number 1. For example, our intuition says that we would like some equalities to hold. For example:

$$\Gamma, x : \mathbb{N} \vdash \text{add}(\text{zero}(), x) = x : \mathbb{N}$$

$$\Gamma, x : \mathbb{N}, y : \mathbb{N}, z : \mathbb{N} \vdash \text{add}(\text{add}(x, y), z) = \text{add}(x, \text{add}(y, z)) : \mathbb{N}$$

$$\Gamma, x : \mathbb{N}, y : \mathbb{N} \vdash \text{add}(x, y) = \text{add}(y, x) : \mathbb{N}$$

Note that these previous three axioms are the commutative monoid axioms, and we can use a system with these axioms to prove results about commutative monoids.

Formally, we define a set of axioms Σ_2 to be a set of 4-tuples (Γ, A, M, N), where Γ is a context, A is a type, and M and N are terms such that $\Gamma \vdash M : A$, $\Gamma \vdash N : A$. To add these axioms to our system, we add the following rule:

$$\frac{(\Gamma, A, M, N) \in \Sigma_2}{\Gamma \vdash M = N : A}$$

We then define $\Sigma = (\Sigma_0, \Sigma_1, \Sigma_2)$ to be a signature, and we let $\mathrm{STT}(\Sigma)$ be STT parameterized by Σ. Note that each of these layers builds on top of one another: Σ_2 is built from terms, which include function symbols, and Σ_1 is build from types, which include base types.

3 Consistency

3.1 Consistency as a Logic

Now that we have built the theory of STT, we would like to know about its consistency. For simplicity, we will just consider $\mathrm{STT}(\varnothing)$ with the empty signature. In IPL, we said that the system is consistent if we cannot prove $\cdot \vdash \bot$. Similarly, we can define a notion of consistency where STT is consistent as a logic if we cannot prove $\cdot \vdash M : 0$. To prove this, we note that the terms of STT look like proofs in IPL, and the rules of STT are generalizations of rules in IPL, so we will use the fact that IPL is consistent to prove consistency of STT. We define a map from STT to IPL as follows:

$$\begin{aligned}
[\![\cdot]\!] &: \mathrm{STT}_{type} \to \mathrm{IPL}_{prop} \\
[\![0]\!] &= \bot \\
[\![1]\!] &= \top \\
[\![A + B]\!] &= [\![A]\!] \vee [\![B]\!] \\
[\![A \times B]\!] &= [\![A]\!] \wedge [\![B]\!] \\
[\![A \Rightarrow B]\!] &= [\![A]\!] \supset [\![B]\!] \\
[\![\{x : A, y : B, ...\}]\!] &= \{[\![A]\!], [\![B]\!], ...\} \\
[\![\Gamma \vdash M : A]\!] &= [\![\Gamma]\!] \vdash [\![A]\!]
\end{aligned}$$

Then it can be proven that if $\Gamma \vdash M : A$ in STT, then $[\![\Gamma]\!] \vdash [\![A]\!]$ in IPL. Then since $\cdot \vdash M : 0$ maps to $\cdot \vdash \bot$, we conclude that since $\cdot \vdash \bot$ cannot be proven in IPL, we have that $\cdot \vdash M : 0$ cannot be proven in STT.

3.2 Set-theoretic semantics

Note that this previous notion of consistency ignores the equational theory of STT. Therefore, we want another notion of consistency: we say that STT is consistent provided that it is not the case that $\Gamma \vdash M = N : A$ for all $\Gamma \vdash M : A$ and $\Gamma \vdash N : A$. As discussed in a previous lecture, it is enough to show that we cannot prove $\cdot \vdash i_1() = i_2() : 1 + 1$ in STT.

To prove this, we will employ technique similar to the proof of consistency of IPL. To prove consistency of IPL, we mapped IPL to the Boolean model. Similarly, to

prove consistency of STT, we will map STT to set theory using a denotation that maps contexts and types to sets, terms to functions between sets, and equality to set-theoretic equality:

$$\frac{A \text{ type}}{[\![A]\!] \text{ set}}$$

$$\frac{\{x : A_x, ...\} \text{ ctx}}{[\![\{x : A_x, ...\}]\!] = \prod_{x:\in\Gamma} [\![A_x]\!]}$$

$$\frac{\Gamma \vdash M : A}{[\![M]\!] : [\![\Gamma]\!] \to [\![A]\!]}$$

$$\frac{\Gamma \vdash M = N : A}{[\![M]\!] = [\![N]\!]}$$

Note that the equality $[\![M]\!] = [\![N]\!]$ is equivalent to $[\![M]\!]\gamma = [\![N]\!]\gamma$ for all $\gamma \in [\![\Gamma]\!]$ as this is the usual set-theoretic equality of functions.

Using this denotation, we will prove that $[\![1 + 1]\!] = a, b$ for some $a \neq b$, and $[\![i_1()]\!] = a$, and $[\![i_2()]\!] = b$. So we have by contraposition that it is not the case that $\cdot \vdash i_1() = i_2() : 1 + 1$ because $a \neq b$.

Note that to extend this denotation to a signature Σ, we must include an interpretation σ mapping Σ to sets, where base types X are mapped to sets and function symbols $f : A_0, ... \to A'$ are mapped to functions $\sigma(f) : \prod_i [\![A_i]\!] \to [\![A']\!]$. For example:

$$\begin{aligned} \sigma(\mathbb{N}) &= \{0, 1, 2, ...\} \\ \sigma(\text{zero}) &= () \mapsto 0 \\ \sigma(\text{add}) &= (x, y) \mapsto x + y \end{aligned}$$

We must then verify that our equational axioms are preserved under σ. However, for simplicity we will use an empty signature for our proof of consistency. We map our types as follows:

$$\begin{aligned} [\![0]\!] &= \varnothing \\ [\![1]\!] &= \{*\} \text{ for some arbitrary element } * \\ [\![A \times B]\!] &= [\![A]\!] \times [\![B]\!] \\ [\![A + B]\!] &= [\![A]\!] \uplus [\![B]\!] \\ [\![A \Rightarrow B]\!] &= \{f : [\![A]\!] \to [\![B]\!]\} \end{aligned}$$

Note that the mapping $[\![0]\!] = \varnothing$ gives rise to another proof that we cannot prove $\cdot \vdash M : 0$ in STT, since $[\![\cdot \vdash M : 0]\!] = [\![M]\!] : \prod_{x:\in\varnothing} \cdot \to \varnothing = [\![M]\!] : \{*\} \to \varnothing$, and so $\cdot \vdash M : 0$ maps to a function from a 1 point set to the empty set, which cannot exist. Furthermore, note that in this denotation we make many arbitrary choices: we map

1 to $\{*\}$ where $*$ is arbitrary. We map $A + B$ to the disjoint union $[\![A]\!] \uplus [\![B]\!]$, which is arbitrarily defined in set theory to be the set $\{(0, x)|x \in [\![A]\!]\} \cup \{(1, y)|y \in [\![B]\!]\}$. And we map $A \times B$ to the Cartesian product $[\![A]\!] \times [\![B]\!]$, which is a set of pairs (a, b), where the ordered pair (a, b) is arbitrarily defined in set theory to be the set $\{a, \{a, b\}\}$. In fact, there are many ways in which we can interpret STT into set theory, which is unlike how there was only one way to interpret IPL into the Boolean algebra. Therefore, we do not have the same uniqueness result for interpreting STT into set theory as we do for interpreting IPL into Heyting algebras[1].

If we had interpreted the sum $A + B$ as the union $[\![A]\!] \cup [\![B]\!]$, then we wouldn't be able to prove our desired theorem that $[\![i_1()]\!] \neq [\![i_2()]\!]$, as $[\![1 + 1]\!] = \{*\} \cup \{*\} = \{*\}$ and so both denotations would be $*$.

Now, we must prove that all of our rules hold. First, we examine $1I$:

$$\frac{}{\Gamma \vdash () : 1}\ 1I$$

We have:

$$[\![\Gamma \vdash () : 1]\!] = [\![()]\!] : [\![\Gamma]\!] \to [\![1]\!] = [\![()]\!] : [\![\Gamma]\!] \to \{*\}$$

So we have that the only choice for denotation of $()$ is the unique map from $[\![\Gamma]\!]$ to $\{*\}$ that sends everything to the point $*$. Then with this denotation, we prove that 1η holds:

$$\frac{\Gamma \vdash M : 1}{\Gamma \vdash M = () : 1}\ 1\eta$$

We have:

$$[\![\Gamma \vdash M : 1]\!] = [\![M]\!] : [\![\Gamma]\!] \to [\![1]\!] = [\![M]\!] : [\![\Gamma]\!] \to \{*\}$$

So similar to before, we have that $[\![M]\!]$ must be the map sending everything to $*$. So for all $\gamma \in [\![\Gamma]\!]$, we have that $[\![M]\!]\gamma = * = [\![()]\!]\gamma$, and so $[\![M]\!] = [\![()]\!]$. Then since the denotation of $\Gamma \vdash M = () : 1$ is $[\![M]\!] = [\![()]\!]$, we have that this denotation preserves 1η.

Next, we will prove that the rules for $+$ types hold. First, we examine $+I$ to determine what our denotation for a term of type $A + B$ should be:

$$\frac{\Gamma \vdash M : A_1}{\Gamma \vdash i_1 M : A_1 + A_2}\ +I_1 \qquad \frac{\Gamma \vdash M : A_2}{\Gamma \vdash i_2 M : A_1 + A_2}\ +I_2$$

We see that

$$[\![\Gamma \vdash i_1 M : A_1 + A_2]\!] = [\![i_1 M]\!] : [\![\Gamma]\!] \to [\![A_1]\!] \uplus [\![A_2]\!]$$

So $[\![i_1 M]\!]$ must map into $[\![A_1]\!] \uplus [\![A_2]\!]$. To construct such a map, we will make use of our hypothesis $\Gamma \vdash M : A_1$. We see that

$$[\![\Gamma \vdash M : A_1]\!] = [\![M]\!] : [\![\Gamma]\!] \to [\![A_1]\!]$$

So $[\![M]\!]$ maps into $[\![A_1]\!]$. Therefore, we can compose this map with the inclusion $[\![A_1]\!] \hookrightarrow [\![A_1]\!] \uplus [\![A_2]\!]$ to obtain the map $[\![i_1 M]\!] = (\gamma \mapsto (0, [\![M]\!]\gamma))$ for our denotation for $[\![i_1 M]\!]$. Similarly, we let $[\![i_2 M]\!] = (\gamma \mapsto (1, [\![M]\!]\gamma))$ be our denotation for $[\![i_2 M]\!]$.

[1] But we will see there is a weakened version of this uniqueness principle.

Using these denotations, we will examine $+E$ to construct a denotation for case splits:

$$\frac{\Gamma \vdash M : A_1 + A_2 \qquad \Gamma, x_1 : A_1 \vdash N_1 : C \qquad \Gamma, x_2 : A_2 \vdash N_2 : C}{\Gamma \vdash \textbf{case}\ M\{i_1 x_1 \to N_1 | i_2 x_2 \to N_2\} : C}\ +E$$

We have:

$$[\![\Gamma \vdash \textbf{case}\ M\{i_1 x_1 \to N_1 | i_2 x_2 \to N_2\} : C]\!] = [\![\textbf{case}\ M\{i_1 x_1 \to N_1 | i_2 x_2 \to N_2\}]\!] : [\![\Gamma]\!] \to [\![C]\!]$$

So we want to use our hypotheses to construct a map from $[\![\Gamma]\!]$ to $[\![C]\!]$. Taking the denotations of our hypotheses:

$$\begin{aligned}
[\![\Gamma \vdash M : A_1 + A_2]\!] &= [\![M]\!] : [\![\Gamma]\!] \to [\![A_1]\!] \uplus [\![A_2]\!] \\
[\![\Gamma, x_1 : A_1 \vdash N_1 : C]\!] &= [\![N_1]\!] : [\![\Gamma]\!] \times [\![A_1]\!] \to [\![C]\!] \\
[\![\Gamma, x_2 : A_2 \vdash N_2 : C]\!] &= [\![N_2]\!] : [\![\Gamma]\!] \times [\![A_2]\!] \to [\![C]\!]
\end{aligned}$$

Looking at the domains and codomains of these three functions, we intuitively see that evaluating $[\![N_1]\!]$ and $[\![N_2]\!]$ at some $\gamma \in \Gamma$ (using currying) gives us functions $[\![A_1]\!] \to [\![C]\!]$ and $[\![A_2]\!] \to [\![C]\!]$, which combine to a map $[\![A_1]\!] \uplus [\![A_2]\!] \to [\![C]\!]$. We can then compose such map with $[\![M]\!]$ to get a map from $[\![\Gamma]\!]$ to $[\![C]\!]$. Concretely, we define $[\![\textbf{case}\ M\{i_1 x_1 \to N_1 | i_2 x_2 \to N_2\}]\!]$ as the following map:

$$\gamma \mapsto \begin{cases} [\![N_1]\!](\gamma, \pi_2([\![M]\!]\gamma)) & \text{if } \pi_1([\![M]\!]\gamma) = 0 \\ [\![N_2]\!](\gamma, \pi_2([\![M]\!]\gamma)) & \text{if } \pi_1([\![M]\!]\gamma) = 1 \end{cases}$$

where π_1 and π_2 are the coordinate projection maps.

Now that we have defined the denotations corresponding to the introduction and elimination rules for terms of $+$ types, we must show that they are compatible with the β and η rules. First, we examine $+\beta_1$:

$$\frac{}{\Gamma \vdash \textbf{case}\ i_1 M\{i_1 x_1 \to N_1 | i_2 x_2 \to N_2\} = N_1[M/x_1]}\ +\beta_1$$

So we have to prove:

$$[\![\Gamma \vdash \textbf{case}\ i_1 M\{i_1 x_1 \to N_1 | i_2 x_2 \to N_2\}]\!] = [\![N_1[M/x_1]]\!]$$

We first evaluate $[\![\Gamma \vdash \textbf{case}\ i_1 M\{i_1 x_1 \to N_1 | i_2 x_2 \to N_2\}]\!]$ at some arbitrary $\gamma \in \Gamma$:

$$\begin{aligned}
&[\![\Gamma \vdash \textbf{case}\ i_1 M\{i_1 x_1 \to N_1 | i_2 x_2 \to N_2\}]\!]\gamma \\
=&\begin{cases} [\![N_1]\!](\gamma, \pi_2([\![i_1 M]\!]\gamma)) & \text{if } \pi_1([\![i_1 M]\!]\gamma) = 0 \\ [\![N_2]\!](\gamma, \pi_2([\![i_1 M]\!]\gamma)) & \text{if } \pi_1([\![i_1 M]\!]\gamma) = 1 \end{cases} \\
=&[\![N_1]\!](\gamma, \pi_2([\![i_1 M]\!]\gamma)) && (\because \pi_1([\![i_1 M]\!]\gamma) = \pi_1(0, [\![M]\!]\gamma) = 0) \\
=&[\![N_1]\!](\gamma, [\![M]\!]\gamma)) && (\because \pi_2([\![i_1 M]\!]\gamma) = \pi_2(0, [\![M]\!]\gamma) = [\![M]\!]\gamma)
\end{aligned}$$

So to prove that

$$[\![\Gamma \vdash \textbf{case } i_1 M\{i_1 x_1 \to N_1 | i_2 x_2 \to N_2\}]\!]\gamma = [\![N_1[M/x_1]]\!]\gamma$$

we must prove that

$$[\![N_1]\!](\gamma, [\![M]\!]\gamma)) = [\![N_1[M/x_1]]\!]\gamma$$

So this will have to be an additional fact that we prove about our semantics function. We will continue this proof with the next lecture.