What is type theory?

Astra Kolomatskaia

1

Logic

A Motivating Problem

SCENARIO: Imagine that you are in a room that has a vending machine that sells walnuts in shells and another vending machine that sells nutcrackers, and that you have a bag of coins. How would you get food?

A Motivating Problem

Scenario: Imagine that you are in a room that has a vending machine that sells walnuts in shells and another vending machine that sells nutcrackers, and that you have a bag of coins. How would you get food?

Solution: Suppose that you are in a room with these three things, then you would take a coin and put it in each of the machines to get a walnut and nutcracker. You would then put the walnut in the nutcracker and get food.

A Motivating Problem

SCENARIO: Imagine that you are in a room that has a vending machine that sells walnuts in shells and another vending machine that sells nutcrackers, and that you have a bag of coins. How would you get food?

SOLUTION: Suppose that you are in a room with these three things, then you would take a coin and put it in each of the machines to get a walnut and nutcracker. You would then put the walnut in the nutcracker and get food.

IN NOTATION:

$$\lambda\, v_w.\ \lambda\, v_{nc}.\ \lambda\, c.\ (v_{nc}\ c)\ (v_w\ c)$$

Parsing Conditional Statements

if it is raining,

then if I am going out,

then if I have an umbrella at home,

then I should take an umbrella with me

Parsing Conditional Statements

$$\textit{if it is raining, then } \left(\textit{if I am going out, then if I have an umbrella at home, then I should take an umbrella with me} \right)$$

Parsing Conditional Statements

if it is raining,

then (*if I am going out*,

then (*if I have an umbrella at home*,

then I should take an umbrella with me))

Parsing Conditional Statements

$$
\begin{aligned}
&\textit{if } \underbrace{\textit{it is raining}}_{\text{A}}, \\
&\quad \textit{then } \left(\textit{if } \underbrace{\textit{I am going out}}_{\text{B}}, \right. \\
&\qquad \textit{then } \big(\textit{if } \underbrace{\textit{I have an umbrella at home}}_{\text{C}}, \\
&\qquad\quad \left. \textit{then } \underbrace{\textit{I should take an umbrella with me}}_{\text{D}} \big) \right)
\end{aligned}
$$

Parsing Conditional Statements

$$\textit{if } \underbrace{\textit{it is raining}}_{A},$$
$$\textit{then } \Big(\textit{if } \underbrace{\textit{I am going out}}_{B},$$
$$\textit{then } \big(\textit{if } \underbrace{\textit{I have an umbrella at home}}_{C},$$
$$\textit{then } \underbrace{\textit{I should take an umbrella with me}}_{D}\big)\Big)$$

We write this as: $A \to B \to C \to D$ for $A \to (B \to (C \to D))$

The Walnut Example

Let:
A be the type of walnuts
B be the type of food
C be the type of coins

A walnut vending machine has type $C \to A$

A nutcracker vending machine has type $C \to A \to B$

A coin has type C

The scenario that we described has type:

$$(C \to A) \to (C \to A \to B) \to C \to B$$

Propositions in Minimal Logic

A proposition is either a logical atom or an arrow:

logical atom	X	$=$	A | B | C | ...	
proposition	T	$=$	X	*logical atom*
		$\mid$	$T \to T$	*arrow*

Propositions in Minimal Logic

A proposition is either a logical atom or an arrow:

$$\textit{logical atom} \quad X \quad = \quad \mathsf{A} \mid \mathsf{B} \mid \mathsf{C} \mid \ldots$$

$$\begin{array}{lccll} \textit{proposition} & T & = & X & \textit{logical atom} \\ & & \mid & T \rightarrow T & \textit{arrow} \end{array}$$

For example, we have:

$$\mathsf{A} \rightarrow ((\mathsf{B} \rightarrow (\mathsf{C} \rightarrow \mathsf{A})) \rightarrow \mathsf{B})$$

which we can more succinctly write as:

$$\mathsf{A} \rightarrow (\mathsf{B} \rightarrow \mathsf{C} \rightarrow \mathsf{A}) \rightarrow \mathsf{B}$$

Contexts and Judgments in Minimal Logic

Contexts are a list of propositions that we take as given

For example, we could assume $(A , A \to B)$

We typically denote variable contexts as Γ

Contexts and Judgments in Minimal Logic

Contexts are a list of propositions that we take as given

For example, we could assume $(\mathsf{A}\ ,\ \mathsf{A} \to \mathsf{B})$

We typically denote variable contexts as Γ

From a context and a proposition, we can form the judgment $\Gamma \vdash T$

This called a sequent and is read as: Γ *proves* T

We also consider the judgment $T \in \Gamma$

This is read as: T *is an assumption in* Γ

Contexts and Judgments in Minimal Logic *[cont.]*

$$
\begin{array}{llcll}
\textit{context} & \Gamma & = & (\,) & \textit{empty} \\
 & & \mid & (\Gamma\,,\,T) & \textit{extension} \\
\\
\textit{judgement} & \mathcal{J} & = & T \in \Gamma & \textit{lookup} \\
 & & \mid & \Gamma \vdash T & \textit{sequent}
\end{array}
$$

For example, we can form judgments like:

$$
\begin{array}{r}
\mathsf{A} \vdash \mathsf{A} \\
\mathsf{A}\,,\,\mathsf{A} \rightarrow \mathsf{B} \vdash \mathsf{B} \\
(\,) \vdash \mathsf{A} \rightarrow (\mathsf{A} \rightarrow \mathsf{B}) \rightarrow \mathsf{B} \\
(\,) \vdash \mathsf{A}
\end{array}
$$

Reasonability

The judgment:

$$() \vdash A \to (A \to B) \to B$$

Is more reasonable than the judgment:

$$() \vdash A$$

Since we know nothing about the atom A, so it should not follow from nothing

Reasonability

The judgment:

$$() \vdash A \to (A \to B) \to B$$

Is more reasonable than the judgment:

$$() \vdash A$$

Since we know nothing about the atom A, so it should not follow from nothing

How do we distinguish which statements are reasonable?

We will discuss two such notions: *truth* and *proof*

Truth

We will discuss a notion of *truth* known as the *Boolean interpretation*

This was originally introduced by Wittgenstein in the Tractatus

This notion of truth has to do with a semantics of *possible states of affairs*

Truth

We will discuss a notion of *truth* known as the *Boolean interpretation*

This was originally introduced by Wittgenstein in the Tractatus

This notion of truth has to do with a semantics of *possible states of affairs*

Denote the truth values by $\top$ *(true)* and $\bot$ *(false)*

Any atom A can either be $\top$ or $\bot$, and we have to account for all possibilities

For example, in $A \to A$, if A is true, then we get $\top \to \top$, which is true, and if A is false, we get $\bot \to \bot$, which is also true, so $A \to A$ is true independent of the state of affairs

Truth *[cont.]*

For atoms, we don't know their truth values, so we consider all possibilities

A *valuation* is a function $\nu : \text{Atom} \to \{\top, \bot\}$

Any valuation $\nu : \text{Atom} \to \{\top, \bot\}$ can be extended to $\langle - \rangle_\nu : \text{Prop} \to \{\top, \bot\}$

This is defined by:

$$\langle X \rangle_\nu \equiv \nu(X)$$
$$\langle T \to W \rangle_\nu \equiv \langle T \rangle_\nu \to \langle W \rangle_\nu$$

Where $\to$ on truth values is defined by the table:

	$T \to W$	$W = \top$	$W = \bot$
T	$\top$	$\top$	$\bot$
	$\bot$	$\top$	$\top$

Truth *[cont.]*

Given a context Γ, a valuation $\nu : \mathsf{Atom} \to \{\top, \bot\}$ is said to be *admissible* if for every $T \in \Gamma$, we have $\langle T \rangle_\nu \equiv \top$

Given a sequent $\Gamma \vdash T$, the sequent is said to be *true* if for every admissible valuation ν, we have $\langle T \rangle_\nu \equiv \top$

We only care about the values of ν on atoms that actually appear in the sequent

Truth *[cont.]*

For example, consider:

$$( ) \vdash A$$

Let ν be defined by $\nu(A) \equiv \bot$, then ν is vacuously admissible in the empty context and $\langle A \rangle_\nu \equiv \bot$

This judgment is therefore not true, as we have constructed a *countermodel*

On the other hand, if ν is admissible in the context (A), then $\nu(A) \equiv \top$

Therefore the following judgment is true:

$$A \vdash A$$

Truth *[cont.]*

For one last example, consider:

$$(\,) \vdash A \to (A \to B) \to B$$

Considering all four valuations defined on A and B, we get:

A	B	$\langle A \to (A \to B) \to B \rangle_\nu$
$\top$	$\top$	$\top \to ((\top \to \top) \to \top) \equiv \top \to (\top \to \top) \equiv \top \to \top \equiv \top$
$\top$	$\bot$	$\top \to ((\top \to \bot) \to \bot) \equiv \top \to (\bot \to \bot) \equiv \top \to \top \equiv \top$
$\bot$	$\top$	$\bot \to ((\bot \to \top) \to \top) \equiv \bot \to (\top \to \top) \equiv \bot \to \top \equiv \top$
$\bot$	$\bot$	$\bot \to ((\bot \to \bot) \to \bot) \equiv \bot \to (\top \to \bot) \equiv \bot \to \bot \equiv \top$

Since all valuations result in $\top$ on the formula, the judgment is true

Proof

Truth requires looking at an exponential number of valuations in terms of the number of atoms in the sequent

Is there a more efficient way to establish the validity of a sequent?

Yes, via *proof*!

Proof

Truth requires looking at an exponential number of valuations in terms of the number of atoms in the sequent

Is there a more efficient way to establish the validity of a sequent?

Yes, via *proof*!

We introduce proof rules:

$$\frac{}{T \in (\Gamma\ ,\ T)}\text{zv} \qquad \frac{T \in \Gamma}{T \in (\Gamma\ ,\ W)}\text{sv}$$

$$\frac{T \in \Gamma}{\Gamma \vdash T}\text{Var} \qquad \frac{\Gamma\ ,\ T \vdash W}{\Gamma \vdash T \to W}\to_{\text{I}} \qquad \frac{\Gamma \vdash T \to W \qquad \Gamma \vdash T}{\Gamma \vdash W}\to_{\text{E}}$$

Proof *[cont.]*

We are then able to chain proof rules together to form *proof trees*

For example:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A \to B \in (A\,,\, A \to B)}\,\mathsf{zv}}{A\,,\, A \to B \vdash A \to B}\,\mathsf{Var} \qquad \dfrac{\dfrac{\dfrac{}{A \in (A)}\,\mathsf{zv}}{A \in (A\,,\, A \to B)}\,\mathsf{sv}}{A\,,\, A \to B \vdash A}\,\mathsf{Var}}{A\,,\, A \to B \vdash B}\,{\to_{\mathsf{E}}}}{\dfrac{A \vdash (A \to B) \to B}{\vdash A \to (A \to B) \to B}\,{\to_{\mathsf{I}}}}\,{\to_{\mathsf{I}}}$$

BHK Interpretation

What does a proof of $T \to W$ mean?

From a constructive perspective, treat propositions as mathematical objects *naively* thought of as the set of their proofs

Write $\Gamma \vdash t : T$ for 'Γ proves T with proof t'

A proof of $T \to W$ is a *construction* that takes a proof of T and produces a proof of W, thus think of $T \to W$ as the *function space* between proofs of T and W

Given a formula with a last free variable $\Gamma\ , x : T \vdash t : W$, we can abstract over the variable to form a function $\Gamma \vdash \lambda\ (x : T) . t : T \to W$

Given a proof of $\Gamma \vdash f : T \to W$ and a proof $\Gamma \vdash t : T$, we can apply the function to form a proof $\Gamma \vdash f\ t : W$

Proof Terms

We define a language of functions:

$$\begin{array}{llcll} \textit{type} & T & = & X & \textit{base} \\ & & \mid & T \to T & \textit{arrow} \\ \\ \textit{term} & t & = & x & \textit{variable} \\ & & \mid & \lambda\ (x : T)\ .\ t & \textit{abstraction} \\ & & \mid & t\ t & \textit{application} \end{array}$$

Sample proof terms are:

$$\lambda\ (a : \mathsf{A})\ .\ a$$
$$\lambda\ (a : \mathsf{A})\ .\ \lambda\ (f : \mathsf{A} \to \mathsf{B})\ .\ f\ a$$

Proof Terms *[cont.]*

Applications are left associative and the scope of abstractions extend maximally to their right

Thus the term:

$$\lambda\ (f : \mathsf{A} \to \mathsf{B} \to \mathsf{C})\,.\ \lambda\ (a : \mathsf{A})\,.\ \lambda\ (b : \mathsf{B})\,.\ f\ a\ b$$

Denotes the fully parenthesised expression:

$$\lambda\ (f : \mathsf{A} \to (\mathsf{B} \to \mathsf{C}))\,.\ (\lambda\ (a : \mathsf{A})\,.\ (\lambda\ (b : \mathsf{B})\,.\ ((f\ a)\ \ b)))$$

Simply Typed Lambda Calculus

We adjust the judgments and proof rules from before to account for terms

Contexts Γ now become lists of variable bindings, such as $(a : \mathsf{A}\ ,\ f : \mathsf{A} \to \mathsf{B})$

Variable lookups assert that a certain binding is in the context $(x : T) \in \Gamma$

Sequents take the form $\Gamma \vdash t : T$, and are read as 'Γ proves T with proof t'

context	Γ	$=$	$(\,)$	*empty*
		$\mid$	$(\Gamma\ ,\ x : T)$	*extension*
judgement	$\mathcal{J}$	$=$	$(x : T) \in \Gamma$	*lookup*
		$\mid$	$\Gamma \vdash t : T$	*sequent*

Simply Typed Lambda Calculus *[cont.]*

The old proof rules for *minimal logic* were:

$$\frac{T \in \Gamma}{\Gamma \vdash T}\text{Var} \qquad \frac{\Gamma\,, T \vdash W}{\Gamma \vdash T \to W}\to_{\text{I}} \qquad \frac{\Gamma \vdash T \to W \qquad \Gamma \vdash T}{\Gamma \vdash W}\to_{\text{E}}$$

The new proof rules for *simply typed lambda calculus* are:

$$\frac{(x : T) \in \Gamma}{\Gamma \vdash x : T}\text{Var} \qquad \frac{\Gamma\,, x : T \vdash t : W}{\Gamma \vdash \lambda\ (x : T)\,.\, t : T \to W}\to_{\text{I}}$$

$$\frac{\Gamma \vdash t : T \to W \qquad \Gamma \vdash s : T}{\Gamma \vdash t\ s : W}\to_{\text{E}}$$

Simply Typed Lambda Calculus *[cont.]*

Our proof tree from before becomes:

$$\dfrac{\dfrac{}{a : \mathsf{A} \,,\, f : \mathsf{A} \to \mathsf{B} \vdash f : \mathsf{A} \to \mathsf{B}}\,\mathsf{Var} \qquad \dfrac{}{a : \mathsf{A} \,,\, f : \mathsf{A} \to \mathsf{B} \vdash a : \mathsf{A}}\,\mathsf{Var}}{\dfrac{a : \mathsf{A} \,,\, f : \mathsf{A} \to \mathsf{B} \vdash f\, a : \mathsf{B}}{\dfrac{a : \mathsf{A} \vdash \lambda\ (f : \mathsf{A} \to \mathsf{B})\,.\, f\, a : (\mathsf{A} \to \mathsf{B}) \to \mathsf{B}}{\vdash \lambda\ (a : \mathsf{A})\,.\,\lambda\ (f : \mathsf{A} \to \mathsf{B})\,.\, f\, a : \mathsf{A} \to (\mathsf{A} \to \mathsf{B}) \to \mathsf{B}}\,{\to_{\mathsf{I}}}}\,{\to_{\mathsf{I}}}}\,{\to_{\mathsf{E}}}$$

Note that at each step, the syntactic category in the conclusion of the rule tells us which rule was applied

The proof tree can thus be recovered uniquely from a well-typed term

$$\vdash \lambda\ (a : \mathsf{A})\,.\,\lambda\ (f : \mathsf{A} \to \mathsf{B})\,.\, f\, a : \mathsf{A} \to (\mathsf{A} \to \mathsf{B}) \to \mathsf{B}$$

Walnut Example

Let:

A be the type of walnuts

B be the type of food

C be the type of coins

Goal:

$$(\mathsf{C} \to \mathsf{A}) \to (\mathsf{C} \to \mathsf{A} \to \mathsf{B}) \to \mathsf{C} \to \mathsf{B}$$

Proof:

$$\lambda\ (v_w : \mathsf{C} \to \mathsf{A})\,.\ \lambda\ (v_{nc} : \mathsf{C} \to \mathsf{A} \to \mathsf{B})\,.\ \lambda\ (c : \mathsf{C})\,.\ (v_{nc}\ c)\ (v_w\ c)$$

Omitting type annotations:

$$\lambda\ v_w.\ \lambda\ v_{nc}.\ \lambda\ c.\ (v_{nc}\ c)\ (v_w\ c)$$

Computation

Functions defined by formulas are dynamic objects, and evaluating a formula on an input should result in computation

This leads to the β and η laws:

$$\frac{\Gamma, x : T \vdash t : W \qquad \Gamma \vdash s : T}{\Gamma \vdash (\lambda\, x.\, t)\ s \equiv t\ [x \mapsto s] : W}\,\beta \qquad\qquad \frac{\Gamma \vdash t : T \to W}{\Gamma \vdash t \equiv \lambda\, x.\, t\ x : T \to W}\,\eta$$

For example, six applications of the β law yield the following definitional equality:

$$(\lambda\, n.\, \lambda\, m.\, \lambda\, z.\, \lambda\, s.\, n\ (m\ z\ s)\ s)\ (\lambda\, z.\, \lambda\, s.\, s\ (s\ z))\ (\lambda\, z.\, \lambda\, s.\, s\ (s\ z))$$
$$\equiv \lambda\, z.\, \lambda\, s.\, s\ (s\ (s\ (s\ z)))$$

This is known as the computation that $2 + 2 = 4$ in *Church arithmetic*

2

Categories

Categories

A *category* $\mathcal{C}$ consists of a collection of object $\mathsf{ob}_{\mathcal{C}}$ and, for every two objects $A\ B : \mathsf{ob}_{\mathcal{C}}$, a collection of morphisms $\mathsf{mor}_{\mathcal{C}}(A, B)$

This is equipped with a composition operation

$$- \circ - : \mathsf{mor}_{\mathcal{C}}(B, C) \times \mathsf{mor}_{\mathcal{C}}(A, B) \to \mathsf{mor}_{\mathcal{C}}(A, C)$$

That is associative and has units $1_A : \mathsf{mor}_{\mathcal{C}}(A, A)$ satisfying the left and right identity laws

We write $f : A \to B$ for $f : \mathsf{mor}_{\mathcal{C}}(A, B)$

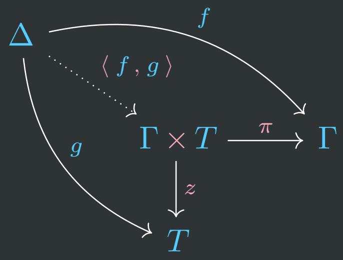
Terminal Objects and Products

A category $\mathcal{C}$ has a *terminal object* $\mathbb{1} : \mathrm{ob}_{\mathcal{C}}$ if for every object $\Gamma : \mathrm{ob}_{\mathcal{C}}$ there is a *unique* morphism $! : \Gamma \to \mathbb{1}$

Terminal Objects and Products

A category $\mathcal{C}$ has a *terminal object* $\mathbb{1} : \mathsf{ob}_{\mathcal{C}}$ if for every object $\Gamma : \mathsf{ob}_{\mathcal{C}}$ there is a *unique* morphism $! : \Gamma \to \mathbb{1}$

A category $\mathcal{C}$ has *products* if for every $\Gamma,\ T : \mathsf{ob}_{\mathcal{C}}$ there is an object $\Gamma \times T : \mathsf{ob}_{\mathcal{C}}$ along with projections $\pi : \Gamma \times T \to \Gamma$, $z : \Gamma \times T \to T$ such that for any $\Delta : \mathsf{ob}_{\mathcal{C}}$ along with $f : \Delta \to \Gamma$ and $g : \Delta \to T$, there is a *unique* $\langle\, f\,,\, g\,\rangle : \Delta \to \Gamma \times T$ satisfying $\pi \circ \langle\, f\,,\, g\,\rangle = f$ and $z \circ \langle\, f\,,\, g\,\rangle = g$



Cartesian Closed Categories

A category with products $\mathcal{C}$ is *cartesian closed* if for every T, W : $\mathsf{ob}_{\mathcal{C}}$, there is a natural family of bijections $\mathsf{mor}_{\mathcal{C}}(\Gamma \times T, W) \cong \mathsf{mor}_{\mathcal{C}}(\Gamma, T \Rightarrow W)$ for some representing object $T \Rightarrow W : \mathsf{ob}_{\mathcal{C}}$

Cartesian Closed Categories

A category with products $\mathcal{C}$ is *cartesian closed* if for every $T, W : \mathsf{ob}_{\mathcal{C}}$, there is a natural family of bijections $\mathsf{mor}_{\mathcal{C}}(\Gamma \times T, W) \cong \mathsf{mor}_{\mathcal{C}}(\Gamma, T \Rightarrow W)$ for some representing object $T \Rightarrow W : \mathsf{ob}_{\mathcal{C}}$

This means that there is $\Lambda : \mathsf{mor}_{\mathcal{C}}(\Gamma \times T, W) \to \mathsf{mor}_{\mathcal{C}}(\Gamma, T \Rightarrow W)$ and $\mathsf{App} : \mathsf{mor}_{\mathcal{C}}(\Gamma, T \Rightarrow W) \to \mathsf{mor}_{\mathcal{C}}(\Gamma \times T, W)$ that are mutually inverse, and naturality of App in Γ means that for $f : \Gamma \to T \Rightarrow W$ and $g : \Delta \to \Gamma$, then $\mathsf{App}\,(f \circ g) \equiv (\mathsf{App}\ f) \circ \langle\, g \circ \pi\ ,\ z \,\rangle$

Cartesian Closed Categories

A category with products $\mathcal{C}$ is *cartesian closed* if for every $T, W : \mathsf{ob}_{\mathcal{C}}$, there is a natural family of bijections $\mathsf{mor}_{\mathcal{C}}(\Gamma \times T, W) \cong \mathsf{mor}_{\mathcal{C}}(\Gamma, T \Rightarrow W)$ for some representing object $T \Rightarrow W : \mathsf{ob}_{\mathcal{C}}$

This means that there is $\Lambda : \mathsf{mor}_{\mathcal{C}}(\Gamma \times T, W) \to \mathsf{mor}_{\mathcal{C}}(\Gamma, T \Rightarrow W)$ and $\mathsf{App} : \mathsf{mor}_{\mathcal{C}}(\Gamma, T \Rightarrow W) \to \mathsf{mor}_{\mathcal{C}}(\Gamma \times T, W)$ that are mutually inverse, and naturality of App in Γ means that for $f : \Gamma \to T \Rightarrow W$ and $g : \Delta \to \Gamma$, then $\mathsf{App}\,(f \circ g) \equiv (\mathsf{App}\ f) \circ \langle\, g \circ \pi \,,\, z \,\rangle$

From this we can define:

$$\mathsf{app} : \mathsf{mor}_{\mathcal{C}}(\Gamma, T \Rightarrow W) \to \mathsf{mor}_{\mathcal{C}}(\Gamma, T) \to \mathsf{mor}_{\mathcal{C}}(\Gamma, W)$$

By:

$$\mathsf{app}\ f\ g = (\mathsf{App}\ f) \circ \langle\, 1_{\Gamma} \,,\, g \,\rangle$$

β and η laws in CCCs

$$\frac{\Gamma\, ,\, x : T \vdash t : W \qquad \Gamma \vdash s : T}{\Gamma \vdash (\lambda\, x.\, t)\ \ s \equiv t\ [x \mapsto s] : W}\,\beta$$

First, for $t : \Gamma \times T \to W$ and $s : \Gamma \to T$, we have:

$$\begin{aligned} &\mathsf{app}\ \ (\Lambda\ t)\ \ s \\ &\equiv (\mathsf{App}\ \ (\Lambda\ t)) \circ \langle\ 1_\Gamma\ ,\ s\ \rangle \\ &\equiv t \circ \langle\ 1_\Gamma\ ,\ s\ \rangle \end{aligned}$$

β and η laws in CCCs *[cont.]*

$$\frac{\Gamma \vdash t : T \to W}{\Gamma \vdash t \equiv \lambda\, x.\, t\; x : T \to W}\;\eta$$

Next, for $f : \Gamma \to T \Rightarrow W$, we have:

$$\begin{aligned}
&\Lambda\;(\mathsf{app}\;(f \circ \pi)\;\;z) \\
&\equiv \Lambda\;((\mathsf{App}\;(f \circ \pi)) \circ \langle\, 1_{\Gamma \times T}\,,\, z\,\rangle) \\
&\equiv \Lambda\;((\mathsf{App}\;f) \circ \langle\, \pi \circ \pi\,,\, z\,\rangle \circ \langle\, 1_{\Gamma \times T}\,,\, z\,\rangle) \\
&\equiv \Lambda\;((\mathsf{App}\;f) \circ \langle\, \pi \circ \pi \circ \langle\, 1_{\Gamma \times T}\,,\, z\,\rangle\,,\, z \circ \langle\, 1_{\Gamma \times T}\,,\, z\,\rangle\,\rangle) \\
&\equiv \Lambda\;((\mathsf{App}\;f) \circ \langle\, \pi\,,\, z\,\rangle) \\
&\equiv \Lambda\;((\mathsf{App}\;f) \circ 1_{\Gamma \times T}) \\
&\equiv \Lambda\;(\mathsf{App}\;f) \\
&\equiv f
\end{aligned}$$

Interpreting STLC

Suppose we consider STLC with only one logical atom A

Refer to the set of types as Ty, the set of contexts as Ctx, the set of variables of type T in context Γ as Var Γ T, and the set of terms of type T in context Γ as Tm Γ T

Now suppose that $\mathcal{C}$ is a cartesian closed category and that we choose an object Base : $\mathrm{ob}_{\mathcal{C}}$

We now define a collection of interpretations using `Agda`-esque pattern matching notation

Interpreting STLC *[cont.]*

TYPES:

$$\llbracket - \rrbracket : \mathsf{Ty} \to \mathsf{ob}_{\mathcal{C}}$$

$$\llbracket \mathsf{A} \rrbracket \equiv \mathsf{Base}$$

$$\llbracket T \to W \rrbracket \equiv \llbracket T \rrbracket \Rightarrow \llbracket W \rrbracket$$

CONTEXTS:

$$\llbracket - \rrbracket : \mathsf{Ctx} \to \mathsf{ob}_{\mathcal{C}}$$

$$\llbracket () \rrbracket \equiv \mathbb{1}$$

$$\llbracket (\Gamma , x : T) \rrbracket \equiv \llbracket \Gamma \rrbracket \times \llbracket T \rrbracket$$

Interpreting STLC *[cont.]*

Variables:

$$\llbracket - \rrbracket : \mathsf{Var}\ \Gamma\ T \to \mathsf{mor}_{\mathcal{C}}\,(\llbracket \Gamma \rrbracket, \llbracket T \rrbracket)$$
$$\llbracket \mathsf{zv} \rrbracket \equiv z$$
$$\llbracket \mathsf{sv}\ v \rrbracket \equiv \llbracket v \rrbracket \circ \pi$$

Terms:

$$\llbracket - \rrbracket : \mathsf{Tm}\ \Gamma\ T \to \mathsf{mor}_{\mathcal{C}}\,(\llbracket \Gamma \rrbracket, \llbracket T \rrbracket)$$
$$\llbracket \mathsf{var}\ v \rrbracket \equiv \llbracket v \rrbracket$$
$$\llbracket \lambda\, x.\, t \rrbracket \equiv \Lambda\, \llbracket t \rrbracket$$
$$\llbracket t\ s \rrbracket \equiv \mathsf{app}\ \llbracket t \rrbracket\ \llbracket s \rrbracket$$

Summary

The *categorical logic* of STLC is a generalisation of the Boolean semantics

The latter is a special case of the full subcategory of Set on the objects $\bot = \{\,\}$ $\top = \{\star\}$

Summary

The *categorical logic* of STLC is a generalisation of the Boolean semantics

The latter is a special case of the full subcategory of Set on the objects $\bot = \{\,\}$ $\top = \{\star\}$

STLC is not complete with respect to Boolean semantics

This is because Peirce's law $((A \to B) \to A) \to A$ is true but not provable as it is equivalent to LEM

But STLC is complete with respect to its categorical logic

Thank you for listening to my talk!