Lecture 4: Simply Typed Lambda Calculus

Lecturer: Max S. New
Scribe: Alexander Bandukwala

September 8th, 2025

1 Disjunction Property

For any set of propositional variables, but no axioms:

$$\text{If } \cdot \vdash A \vee B \text{ then either } \cdot \vdash A \text{ or } \cdot \vdash B$$

This is another distinguishing property of IPL from classical propositional logic, in that this is not true for classical propositional logic.

In classical propositional logic for any propositional variable we have the proof that the law of excluded middle holds:

$$\cdot \vdash x \vee \neg x$$

but we can't prove with no axioms which of the two cases is true.

This is why we say that IPL is *Constructive*: when you are proving a disjunction under no hypotheses you have to explictitly prove one case or the other. While classical propositional logic provides indirect non-constructive arguments.

Note that in the above definition the context is empty. This is not generally true of IPL in a non-empty context. e.g. $x \vee y \vdash x \vee y$ does not give a constructive proof of x or y.

This point of view says that maybe we should not really think of IPL as talking about truth at all, but as a stronger property of constructive proof or constructability.

1.1 Constructive Example

$$\cdot \vdash \top \vee \top$$

1.1.1 Obvious Proofs

$$\dfrac{\dfrac{}{\cdot \vdash \top}\ \top\text{I}}{\cdot \vdash \top \vee \top}\ \vee\text{I1} \qquad\qquad \dfrac{\dfrac{}{\cdot \vdash \top}\ \top\text{I}}{\cdot \vdash \top \vee \top}\ \vee\text{I2}$$

The above proofs give two different reasons the claim is true because you can prove the left hand side and it's true because you can prove the right hand side.

1.1.2 Circuitous Proof

$$\dfrac{\dfrac{\dfrac{}{\cdot \vdash \top}\,\top\mathrm{I}}{\cdot \vdash \top \vee \top}\,\vee\mathrm{I1} \qquad \dfrac{\dfrac{}{\top \vdash \top}\,\textsc{assum}}{\top \vdash \top \vee \top}\,\vee\mathrm{I1}}{\cdot \vdash \top \vee \top}\,\vee\mathrm{E}$$

This proof is not essentially different from the other two. It is indirect but still using $\vee$I1 to prove it, so it is a more complex version of the first proof.

We are going to use STLC to provide a theory of when two proofs are equivalent.

2 Simply Typed Lambda Calculus (STLC)

The Simply Typed Lambda Calculus is going to generalize directly IPL. The relationship is an instance of the *Curry-Howard* correspondence.

STLC is one of the simplest possible functional programing languages. It has types, terms (programs), and an equational theory for reasoning about those programs.

The terms constructros of STLC correspond directly to the rules of inference in IPL:

2.0.1 Judgments

IPL(Σ)	**STLC(Σ)**
Γ ctx	Γ ctx *
A Prop	A type
$\dfrac{\mathcal{D}}{\Gamma \vdash A}$	$\dfrac{\mathcal{D}}{\Gamma \vdash M : A}$
	$\dfrac{\mathcal{D}}{\Gamma \vdash M = N : A}$

M is a term summarizing the data of the derivation $\mathcal{D}$ as a syntactic object. So the proof derivation will be implicit in the structure of the term to be more concise than the proof trees. The last judgment is our judgment proving equalities.

A context has form

$$\Gamma = x_1 : A_1,\ x_2 : A_2,\ \ldots \quad \text{with all } x_i \text{ distinct and new variables never clash.}$$

The distinct variable convention is called the *Barendregt* convention.

2.0.2 Types

The types will in STLC are going to map directly to the proposition constructors in IPL.

IPL	**STLC**
$\top$	1 `Unit` $()$ `Void`
$A \wedge B$	$A \times B$
$A \supset B$	$A \Rightarrow B$
$\bot$	0
$A \vee B$	$A + B$
x	x

In STLC, each type connective comes with *introduction and elimination rules* (how to construct and deconstruct values), as well as *equational reasoning principles.* The latter are divided into:

- **Computational principles** (β-laws), which describe how programs simplify when you eliminate what was just introduced.
- **Extensionality principles** (η-laws), which describe when two terms are definitionally the same by their observable behavior.

2.0.3 $A \times B$

The introduction rule is the same as the IPL rule but with terms attached.

Introduction and elimination.

$$\frac{\Gamma \vdash M : A \qquad \Gamma \vdash N : B}{\Gamma \vdash (M, N) : A \times B}\,\times\mathrm{I} \qquad \frac{\Gamma \vdash M : A \times B}{\Gamma \vdash \pi_1 M : A}\,\times\mathrm{E1} \qquad \frac{\Gamma \vdash M : A \times B}{\Gamma \vdash \pi_2 M : B}\,\times\mathrm{E2}$$

Computational principles (β-laws).

$$\frac{\Gamma \vdash M : A \qquad \Gamma \vdash N : B}{\Gamma \vdash \pi_1(M, N) = M : A}\,\times\beta1 \qquad \frac{\Gamma \vdash M : A \qquad \Gamma \vdash N : B}{\Gamma \vdash \pi_2(M, N) = N : B}\,\times\beta2$$

Extensionality principles (η-laws).

$$\frac{\Gamma \vdash M : A \times B}{\Gamma \vdash (\pi_1 M, \pi_2 M) = M : A \times B}\,\times\eta \qquad \frac{\Gamma \vdash \pi_1 M = \pi_1 N : A \qquad \Gamma \vdash \pi_2 M = \pi_2 N : B}{\Gamma \vdash M = N : A \times B}\,\times\textsc{Ext}$$

The extensionality principle is not for simplifying terms but used for reasoning about when two arbitrary terms of type $A \times B$ are equivalent or not. Both $\times\eta$ and $\times$Ext are equivalent using rules provided below.

2.0.4 Equational Reasoning

For a reasonable notion of equality we want an equivalence relation.

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash M = M : A}\ \text{REFL} \quad \frac{\Gamma \vdash N = M : A}{\Gamma \vdash M = N : A}\ \text{SYM} \quad \frac{\Gamma \vdash M = N : A \qquad \Gamma \vdash N = P : A}{\Gamma \vdash M = P : A}\ \text{TRANS}$$

Congruence rules tell us that our equivalence relation, equating different terms of the language, is a congruence with respect to all the operations. Whenever we add a new term constructor to the language we're also going to add a corresponding congruence rule to it. e.g.

$$\frac{\Gamma \vdash M_1 = M_2 : A \qquad \Gamma \vdash N_1 = N_2 : B}{\Gamma \vdash (M_1, N_1) = (M_2, N_2) : A \times B}\ \times\text{ICong}$$

$$\frac{\Gamma \vdash M = M' : A \times B}{\Gamma \vdash \pi_1 M = \pi_1 M' : A}\ \times\text{E1Cong} \quad \frac{\Gamma \vdash M = M' : A \times B}{\Gamma \vdash \pi_2 M = \pi_2 M' : B}\ \times\text{E2Cong}$$

$$\frac{}{\Gamma, A, \Delta \vdash A}\ \text{Assumption} \quad \frac{}{\Gamma, x : A, \Delta \vdash x : A}\ \text{Var}$$

$$\frac{\Gamma \vdash A \qquad \Gamma, A \vdash B}{\Gamma \vdash B}\ \text{Subst} \quad \frac{\Gamma \vdash N : A \qquad \Gamma, x : A \vdash M : B}{\Gamma \vdash M[N/x] : B}\ \text{Admissable Rule}$$

where $M[N/x]$ is replacing all of the occurrences of the variable x with N in M.

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

2.0.5 1

$$\frac{}{\Gamma \vdash () : 1}\ 1\text{I} \qquad \frac{\Gamma \vdash M : 1}{\Gamma \vdash M = () : 1}\ 1\eta \qquad \frac{\Gamma \vdash M : 1 \qquad \Gamma \vdash N : 1}{\Gamma \vdash M = N : 1}\ 1\text{Ext}$$

There's no elimination principles, and no congruence principles because there's no assumptions in the rule. There's also no β rule because there's no elimination principle.

2.0.6 0

$$\frac{\Gamma \vdash M : 0}{\Gamma \vdash \mathtt{case} M\{\} : A}\ 0\text{E} \qquad \frac{\Gamma \vdash M : 0}{\Gamma \vdash N = P : A}\ 0\text{Ext} \quad \frac{\Gamma \vdash N : 0 \qquad \Gamma, x : 0, \Delta \vdash M : A}{\Gamma, \Delta \vdash M[N/x] = \mathtt{case} N\{\} : A}\ 0\eta$$

$$\frac{\Gamma \vdash M = N : 0}{\Gamma \vdash \mathtt{case} M\{\} = \mathtt{case} N\{\} : A}\ 0\text{Cong}$$

Dual to 1 there's no introduction rule, but there is elimination.

2.0.7 $+$

Result from Rust, Either from Haskell.

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash \sigma_1 M : A + B}\,{+}\mathrm{I1} \qquad\qquad \frac{\Gamma \vdash M : B}{\Gamma \vdash \sigma_2 M : A + B}\,{+}\mathrm{I2}$$

$$\frac{\Gamma \vdash M : A + B \qquad \Gamma, x : A \vdash N_1 : C \qquad \Gamma, y : B \vdash N_2 : C}{\Gamma \vdash \mathtt{case} M\{\sigma_1 x.N_1 \mid \sigma_2 y.N_2\} : C}\,{+}\mathrm{E}$$

$$\frac{\Gamma \vdash M : A + B \qquad \Gamma, x : A \vdash N_1 : C \qquad \Gamma, y : B \vdash N_2 : C}{\Gamma \vdash \mathtt{case}(\sigma_1 M)\{\sigma_1 x.N_1 \mid \sigma_2 y.N_2\} = N_1[M/x] : C}\,{+}\beta 1$$

$$\frac{\Gamma \vdash M : A + B \qquad \Gamma, x : A \vdash N_1 : C \qquad \Gamma, y : B \vdash N_2 : C}{\Gamma \vdash \mathtt{case}(\sigma_2 M)\{\sigma_1 x.N_1 \mid \sigma_2 y.N_2\} = N_2[M/y] : C}\,{+}\beta 2$$

$$\frac{\Gamma, z : A + B, \Delta \vdash M : C \qquad \Gamma \vdash N : A + B}{\Gamma \vdash M[N/z] = \mathtt{case} N\{\sigma_1 x.M[\sigma_1 x/z] \mid \sigma_2 y.M[\sigma_2 y/z]\} : C}\,{+}\eta$$

$$\frac{\Gamma, x : A \vdash M[\sigma_1 x/z] = N[\sigma_1 x/z] : C \qquad \Gamma, y : B \vdash M[\sigma_2 y/z] = N[\sigma_2 y/z] : C}{\Gamma, z : A + B \vdash M = N : C}\,{+}\textsc{Ext}$$

2.0.8 $\Rightarrow$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \Rightarrow B}\,{\Rightarrow}\mathrm{I} \qquad\qquad \frac{\Gamma \vdash M : A \Rightarrow B \qquad \Gamma \vdash N : A}{\Gamma \vdash MN : B}\,{\Rightarrow}\mathrm{E}$$

$$\frac{\Gamma, x : A \vdash M : B \qquad \Gamma \vdash N : A}{\Gamma \vdash (\lambda x.M)N = M[N/x] : B} \Rightarrow \beta \qquad\qquad \frac{\Gamma \vdash M : A \Rightarrow B}{\Gamma \vdash M = \lambda x.Mx : A \Rightarrow B} \Rightarrow \eta$$

$$\frac{\Gamma, x : A \vdash Mx = Nx : B}{\Gamma \vdash M = N : A \Rightarrow B}\,{\Rightarrow}\textsc{Ext}$$

2.0.9 Invertible Rules

In IPL every type constructor had an invertible rule and that corresponded in the semantics to their definition by the universal property:

$$\begin{array}{c} \Gamma, A \vdash B \\ \hline\hline \Gamma \vdash A \supset B \end{array}$$

$$\downarrow (A \supset B) = \{z | z \wedge A \leq B\}$$

In STLC:

So if we have the admissable property of weakening we can invert the rule. But for STLC we also care about equality of terms.

We want this rule to be bijective.

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \Rightarrow B}$$

$$\dfrac{\dfrac{\Gamma \vdash M : A \Rightarrow B}{\Gamma, x : A \vdash M : A \Rightarrow B}\,\text{Weak} \qquad \dfrac{}{\Gamma, x : A \vdash x : A}\,\text{Var}}{\Gamma, x : A \vdash Mx : B}\,{\Rightarrow}\text{E}$$

$$\dfrac{\dfrac{\Gamma, y : A \vdash M : B}{\Gamma, x : A, y : A \vdash M : B}\,\text{Weakening} \qquad \dfrac{}{\Gamma, x : A \vdash x : A}\,\text{var}}{\Gamma, x : A \vdash (\lambda y.M)x = M[x/y] : B}\Rightarrow \beta$$

So we pick our β and η rules to make our rules logically invertible and actually bijective on the sets of terms.

$\times$

$$\dfrac{\dfrac{\Gamma \vdash M : A \qquad \Gamma \vdash N : B}{\Gamma \vdash (M, N) : A \times B}}{\dfrac{\Gamma \vdash \pi_1(M, N) : A}{\Gamma \vdash \pi_1(M, N) = M : A}\times\beta 1 \qquad \dfrac{\Gamma \vdash \pi_2(M, N) : A}{\Gamma \vdash \pi_2(M, N) = N : B}\times\beta 2}$$

$$\dfrac{\dfrac{\Gamma \vdash P : A \times B}{\Gamma \vdash \pi_1 P : A \qquad \Gamma \vdash \pi_2 P : B}}{\Gamma \vdash (\pi_1 P, \pi_2 P) = P : A \times B}\times\eta$$

$+$

$$\frac{\Gamma, x : A \vdash M : C \qquad \Gamma, y : B \vdash N : C}{\Gamma, z : A + B \vdash \texttt{case}z\{\sigma_1 x.M \mid \sigma_1 y.N\} : C}$$

$$\frac{\Gamma, z : A + B, \vdash P : C}{\Gamma, x : A \vdash P[\sigma_1 x/z] \qquad \Gamma, y : B \vdash P[\sigma_2 y/z]}$$

2.0.10 Signatures

A signature for IPL was

$$\Sigma = (\Sigma_0, \Sigma_1)$$

Σ_0 a set of propositional variables

Σ_1 a set of axioms of the form $A_1, \ldots, A_n \implies B$

and we had rules

$$\frac{x \in \Sigma_0}{x \text{ prop}} \frac{A_1, \ldots \implies B \in \Sigma_1}{A_1, \ldots \vdash B}$$

Signatures in STLC will be of the form

$$\Sigma = (\Sigma_0, \Sigma_1, \Sigma_2)$$

Σ_0 a set of base types

Σ_1 a set of function symbols of the form $A_1, \ldots, A_n \to B$ where $A_1, \ldots, A_n$ is the domain and B is the codomain, together called the arity.

Σ_2 Set of pairs (M, N)
$\Gamma \vdash M : A, \Gamma \vdash N : A$

with rules

$$\frac{x \in \Sigma_0}{x \text{ type}} \qquad \frac{f \in \Sigma_1(A_1, \ldots, B) \qquad \Gamma \vdash M_1 : A_1 \ldots}{\Gamma \vdash f(M_1, \ldots) : B} \textsc{FunSymb}$$

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

Example Signature

$$\begin{aligned} \Sigma_0 &= X, S \\ \texttt{empty} &: \cdot \to S \\ \texttt{push} &: X, S \to S \\ \texttt{pop} &: S \to 1 + (X \times S) \end{aligned}$$

$$\begin{array}{lll} \Sigma_2 & & \\ \cdot & \vdash \texttt{pop}(\texttt{emp}()) = \sigma_1 & : 1 + (X \times S) \\ x : X, s : S & \vdash \texttt{pop}(\texttt{push}(x, s)) = \sigma_2(x, s) & : 1 + (X \times S) \end{array}$$