

# Polymorphism

- Outline
  - *Review*
  - *Curry-Howard Isomorphism (Propositions-as-types)*
  - *System F (Second-order lambda calculus, Girard 1972)*



# Propositional Logic Syntax

Propositions:

$$\begin{array}{l} e ::= \top \mid \perp \\ \mid P, Q, R, \dots \\ \mid \neg e \\ \mid e \wedge e \\ \mid e \vee e \\ \mid e \Rightarrow e \end{array}$$



# Sequents

- A *sequent* has the form  $\Gamma \vdash \Delta$
- $\Delta$  is a list of propositions  $\alpha_1, \dots, \alpha_n$
- $\Gamma$  is a *context* containing a list of propositions  $\beta_1, \dots, \beta_n$
- We can extend valuations to sequents, to get the following semantics:
  - A sequent  $\beta_1, \dots, \beta_n \vdash \alpha_1, \dots, \alpha_n$  is true if some  $\alpha_i$  is true whenever  $\beta_1, \dots, \beta_n$  are all true.



# Derivations

- There are two kinds of inference rules
  - *Introduction* rules operate on the right of the turnstile
  - *Elimination* rules operate on the left of the turnstile
- The base axiom

$$\frac{}{\Gamma_1, \alpha, \Gamma_2 \vdash \Delta_1, \alpha, \Delta_2} \text{ axiom}$$



# Rule table

$$\overline{\Gamma \vdash \Delta_1, \top, \Delta_2}$$

$$\overline{\Gamma_1, \perp, \Gamma_2 \vdash \Delta}$$

$$\frac{\Gamma \vdash \Delta_1, \alpha, \Delta_2 \quad \Gamma \vdash \Delta_1, \beta, \Delta_2}{\Gamma \vdash \Delta_1, \alpha \wedge \beta, \Delta_2}$$

$$\frac{\Gamma_1, \alpha, \beta, \Gamma_2 \vdash \Delta}{\Gamma_1, \alpha \wedge \beta, \Gamma_2 \vdash \Delta}$$

$$\frac{\Gamma \vdash \Delta_1, \alpha, \beta, \Delta_2}{\Gamma \vdash \Delta_1, \alpha \vee \beta, \Delta_2}$$

$$\frac{\Gamma_1, \alpha, \Gamma_2 \vdash \Delta \quad \Gamma_1, \beta, \Gamma_2 \vdash \Delta}{\Gamma_1, \alpha \vee \beta, \Gamma_2 \vdash \Delta}$$

$$\frac{\Gamma, \alpha \vdash \Delta_1, \beta, \Delta_2}{\Gamma \vdash \Delta_1, \alpha \Rightarrow \beta, \Delta_2}$$

$$\frac{\Gamma_1, \beta, \Gamma_2 \vdash \Delta \quad \Gamma_1, \Gamma_2 \vdash \alpha, \Delta}{\Gamma_1, \alpha \Rightarrow \beta, \Gamma_2 \vdash \Delta}$$



# Pierce's law

$$\frac{\frac{\frac{\frac{\overline{A \vdash B, A} \quad 4}{\vdash A \Rightarrow B, A} \quad 3}{(A \Rightarrow B) \Rightarrow A \vdash A} \quad 2}{\vdash ((A \Rightarrow B) \Rightarrow A) \Rightarrow A} \quad 1}{\vdash ((A \Rightarrow B) \Rightarrow A) \Rightarrow A} \quad 1$$



# First-order logic

- Add *atomic formulas*  $f(a_1, \dots, a_n)$  of various arities.
- Add *atomic predicates*  $P(a_1, \dots, a_m)$  of various arities.

$$\begin{array}{l} a ::= f(a_1, \dots, a_n) \\ e ::= \top \mid \perp \\ \quad | P(a_1, \dots, a_m) \\ \quad | \neg e \\ \quad | e \wedge e \\ \quad | e \vee e \\ \quad | e \Rightarrow e \\ \quad | \forall v. e \\ \quad | \exists v. e \end{array}$$



## Some examples

- $\forall x. \text{big}(x) \Rightarrow \text{heavy}(x)$
- $\forall i. (i + 1) > i$
- $\forall i. \exists j. j > i$
- $\exists i. \forall j. j \leq i$



# Adding rules for FOL

$$\frac{\Gamma \vdash \Delta_1, P(t), \Delta_2}{\Gamma \vdash \Delta_1, \exists v.P(v), \Delta_2} \text{ exists intro}$$

$$\frac{\Gamma \vdash \Delta_1, P(c), \Delta_2 \text{ (new } c\text{)}}{\Gamma \vdash \Delta_1, \forall v.P(v), \Delta_2} \text{ all intro}$$

$$\frac{\Gamma_1, P(c), \Gamma_2 \vdash \Delta \text{ (new } c\text{)}}{\Gamma_1, \exists v.P(v), \Gamma_2 \vdash \Delta} \text{ exists elim}$$

$$\frac{\Gamma_1, \forall v.P(v), \Gamma_2, P(t) \vdash \Delta}{\Gamma_1, \forall v.P(v), \Gamma_2 \vdash \Delta} \text{ all elim}$$



# A quantifier DeMorgan law

$$\frac{\frac{\frac{\frac{\frac{\frac{\frac{\overline{\varphi(c) \vdash \varphi(c)}^6}{\forall x.\varphi(x) \vdash \varphi(c)}^5}{\neg\varphi(c), \forall x.\varphi(x) \vdash \neg\varphi(c)}^4}{\neg\varphi(c) \vdash \neg(\forall x.\varphi(x))}^3}{\exists x.\neg\varphi(x) \vdash \neg(\forall x.\varphi(x))}^2}{\vdash (\exists x.\neg\varphi(x)) \Rightarrow \neg(\forall x.\varphi(x))}^1$$



# *Second-order logic*

- In FOL, the variables range over the elements of a structure
- Sometimes it is useful to consider all subsets of a structure
  - *“Each bounded non-empty set of reals has a supremum”*
  - *“Each non-empty set of natural numbers has a minimal element”*
- Second-order logic allows quantification over relations



# ***SOL language definition***

The language contains:

- individual variables  $x_0, x_1, \dots$
- constants  $c_0, c_1, \dots$  and  $n$ -ary function symbols  $f_0^n, f_1^n, \dots$
- $n$ -ary predicate variables  $X_0^n, X_1^n, \dots$
- $n$ -ary predicate symbols  $P_0^n, P_1^n, \dots$
- the usual connectives  $\neg, \wedge, \vee, \Rightarrow, \exists, \forall$



# Rules for predicate quantification

$$\frac{\Gamma \vdash \Delta_1, \varphi[\sigma/X], \Delta_2 \quad (\text{some predicate } \sigma)}{\Gamma \vdash \Delta_1, \exists X.\varphi, \Delta_2} \exists^2\text{intro}$$

$$\frac{\Gamma_1, \varphi[C/X], \Gamma_2 \vdash \Delta \quad (\text{new symbol } C)}{\Gamma_1, \exists X.\varphi, \Gamma_2 \vdash \Delta} \exists^2\text{elim}$$

$$\frac{\Gamma \vdash \Delta_1, \varphi[C/X], \Delta_2 \quad (\text{new symbol } C)}{\Gamma \vdash \Delta_1, \forall X.\varphi, \Delta_2} \forall^2\text{intro}$$

$$\frac{\Gamma_1, \forall X.\varphi, \Gamma_2, \varphi[\sigma/X] \vdash \Delta \quad (\text{some predicate } \sigma)}{\Gamma_1, \forall X.\varphi, \Gamma_2 \vdash \Delta} \forall^2\text{elim}$$



# Conjunction is definable

$$\frac{\frac{\frac{\varphi, \psi \vdash \varphi \quad \frac{\dots}{\varphi, \psi, \psi \Rightarrow C \vdash C}}{\varphi, \psi, \varphi \Rightarrow (\psi \Rightarrow C) \vdash C}}{\varphi, \psi \vdash (\varphi \Rightarrow (\psi \Rightarrow C)) \Rightarrow C}}{\varphi, \psi \vdash \forall X. ((\varphi \Rightarrow (\psi \Rightarrow X)) \Rightarrow X)}}{\vdash (\varphi \wedge \psi) \Rightarrow (\forall X. ((\varphi \Rightarrow (\psi \Rightarrow X)) \Rightarrow X))}$$



# Operator definitions in SOL

All operators can be defined in terms of  $\forall$  and  $\Rightarrow$

$$\begin{aligned}\perp &\Leftrightarrow \forall X.X \\ \varphi \wedge \psi &\Leftrightarrow \forall X.(\varphi \Rightarrow (\psi \Rightarrow X)) \Rightarrow X \\ \varphi \vee \psi &\Leftrightarrow \forall X.((\psi \Rightarrow X) \wedge (\varphi \Rightarrow X)) \Rightarrow X \\ \exists x.\varphi(x) &\Leftrightarrow \forall X.(\forall x.\varphi(x) \Rightarrow X) \Rightarrow X \\ \exists X.\varphi(X) &\Leftrightarrow \forall Y.(\forall X.(\varphi(X) \Rightarrow Y) \Rightarrow Y)\end{aligned}$$



# The polymorphic lambda calculus

$e$	$::=$	$v$	variables
		$\lambda x:t.e$	abstraction
		$e_1 e_2$	application
		$\Lambda X.e$	type abstraction
		$e[t]$	type application
$t$	$::=$	$X$	type variables
		$t_1 \rightarrow t_2$	function types
		$\forall X.t$	type quantification



# Identity program

Identity:

$$(\Lambda X. \lambda x: X. X) : (\forall X. X \rightarrow X)$$

Application of identity:

$$\begin{aligned} & (\Lambda X. \lambda x: X. X) [\mathbb{Z}] 1 \\ \rightarrow_{\beta} & (\lambda x: \mathbb{Z}. x) 1 \\ \rightarrow_{\beta} & 1 \end{aligned}$$



# ***But wait***

- Is the similarity between logic and programs an accident?
  - *If so, can we reason by analogy?*
  - *If not, what is the connection?*



# Curry-Howard isomorphism

- Propositions and types are isomorphic in a constructive logic
- Intuitionistic logic
  - *A proof of (A or B) must be a proof of A or a proof of B*
  - *An proof of an existential must provide a witness*



# Law of excluded middle

- In general, the proposition  $P \vee \neg P$  is not true in a constructive logic.
- An example classical proof: show there are two irrational numbers  $a, b$  s.t.  $a^b$  is rational.
  - - Proof: let  $c = \sqrt{2}^{\sqrt{2}}$ .
  - If  $c$  is rational, let  $a = \sqrt{2}, b = \sqrt{2}$
  - If  $c$  is irrational, choose  $a = c, b = \sqrt{2}$ , for which  $a^b = 2$ .



# Models

- “State of knowledge interpretation” (Kripke)
  - *A proposition is true only when we know it to be so, or we have proven it*
  - *At any moment we do not know what will be discovered*
  - *Knowledge is strictly increasing; nothing we know now will ever be contradicted*



# Derivations in constructive logic

- The only change: sequents can have only one conclusion
- A *sequent* has the form  $\Gamma \vdash \alpha$
- $\Gamma$  is a *context* containing a list of propositions  $\beta_1, \dots, \beta_n$
- We can extend valuations to sequents, to get the following semantics:
  - A sequent  $\beta_1, \dots, \beta_n \vdash \alpha$  is true if  $\alpha$  is true whenever  $\beta_1, \dots, \beta_n$  are all true.



# Intuitionistic logic

$\overline{\Gamma_1, \alpha, \Gamma_2 \vdash \alpha}$  axiom

$\overline{\Gamma_1, \perp, \Gamma_2 \vdash \alpha}$  false elim

$\overline{\Gamma \vdash \top}$  true intro

$\frac{\Gamma_1, \alpha, \beta, \Gamma_2 \vdash \delta}{\Gamma_1, \alpha \wedge \beta, \Gamma_2 \vdash \delta}$  and elim

$\frac{\Gamma \vdash \alpha \quad \Gamma \vdash \beta}{\Gamma \vdash \alpha \wedge \beta}$  and intro

$\frac{\Gamma_1, \alpha, \Gamma_2 \vdash \delta \quad \Gamma_1, \beta, \Gamma_2 \vdash \delta}{\Gamma_1, \alpha \vee \beta, \Gamma_2 \vdash \delta}$  or elim

$\frac{\Gamma \vdash \alpha}{\Gamma \vdash \alpha \vee \beta}$  or left     $\frac{\Gamma \vdash \beta}{\Gamma \vdash \alpha \vee \beta}$  or right

$\frac{\Gamma_1, \alpha \Rightarrow \beta, \beta, \Gamma_2 \vdash \delta \quad \Gamma_1, \Gamma_2 \vdash \alpha}{\Gamma_1, \alpha \Rightarrow \beta, \Gamma_2 \vdash \delta}$  implies elim

$\frac{\Gamma, \alpha \vdash \beta}{\Gamma \vdash \alpha \Rightarrow \beta}$  implies intro



# Curry-Howard Isomorphism

Curry-Howard isomorphism:

- The proposition  $\alpha \Rightarrow \beta$  corresponds to the type  $\alpha' \rightarrow \beta'$
- A proposition  $P$  is *true* i? there is a program that has type  $P$
- The  $\perp$  type is the “void” or empty type



# Program extraction

Automatic program extraction: annotate the proof rules with the programs that are the proofs.

$$\frac{\Gamma_1, f: \alpha \Rightarrow \beta, b: \beta, \Gamma_2 \vdash \delta \quad \mathbf{ext} \ t \quad \Gamma_1, f: \alpha \Rightarrow \beta, \Gamma_2 \vdash \alpha \quad \mathbf{ext} \ a}{\Gamma_1, f: \alpha \Rightarrow \beta, \Gamma_2 \vdash \delta \quad \mathbf{ext} \ t[f(a)/b]} \text{ implies elim}$$

$$\frac{\Gamma, x: \alpha \vdash \beta \quad \mathbf{ext} \ b}{\Gamma \vdash \alpha \Rightarrow \beta \quad \mathbf{ext} \ \lambda x. b} \text{ implies intro}$$



# Example extraction

$$\frac{\frac{\frac{}{x:A, y:B \vdash A}}{x:A \vdash B \Rightarrow A}}{\vdash A \Rightarrow B \Rightarrow A} \quad \begin{array}{l} 3 \\ 2 \\ 1 \end{array}$$



# Extraction annotations

$$\frac{\frac{\frac{x:A, y:B \vdash A \quad \text{ext } x}{x:A \vdash B \Rightarrow A} \quad \text{ext } \lambda y.x}{\vdash A \Rightarrow B \Rightarrow A} \quad \text{ext } \lambda x.\lambda y.x$$

3  
2  
1



## Another example

$$\frac{\frac{\frac{\frac{\frac{\frac{\frac{\frac{f:A \Rightarrow B \Rightarrow C, g:A \Rightarrow B, x:A, y:B, z:C \vdash C}{\text{ext } z}}{6}}{f:A \Rightarrow B \Rightarrow C, g:A \Rightarrow B, x:A, y:B \vdash C}{\text{ext } f \ x \ y}}{5}}{f:A \Rightarrow B \Rightarrow C, g:A \Rightarrow B, x:A \vdash C}{\text{ext } f \ x \ (g \ x)}}{4}}{f:A \Rightarrow B \Rightarrow C, g:A \Rightarrow B \vdash A \Rightarrow C}{\text{ext } \lambda x. f \ x \ (g \ x)}}{3}}{f:A \Rightarrow B \Rightarrow C \vdash (A \Rightarrow B) \Rightarrow (A \Rightarrow C)}{\text{ext } \lambda g. \lambda x. f \ x \ (g \ x)}}{2}}{\vdash (A \Rightarrow B \Rightarrow C) \Rightarrow (A \rightarrow B) \Rightarrow (A \rightarrow C)}{\text{ext } \lambda f. \lambda g. \lambda x. f \ x \ (g \ x)}}{1}}$$



# Hilbert's axioms for propositional logic

Constructive propositional logic has two axioms:

$$\mathbf{K}: A \Rightarrow B \Rightarrow A$$

$$\mathbf{S}: (A \Rightarrow B \Rightarrow C) \Rightarrow (A \Rightarrow B) \Rightarrow (A \Rightarrow C)$$

and the rule for modus-ponens (implication elimination).

**Theorem** The Hilbert system  $S$  and  $K$  is sound and complete.



# The polymorphic lambda calculus

$e$	$::=$	$v$	variables
		$\lambda x:t.e$	abstraction
		$e_1 e_2$	application
		$\Lambda X.e$	type abstraction
		$e[t]$	type application
$t$	$::=$	$X$	type variables
		$t_1 \rightarrow t_2$	function types
		$\forall X.t$	type quantification



# Identity program

Identity:

$$(\Lambda X. \lambda x: X. X) : (\forall X. X \rightarrow X)$$

Application of identity:

$$\begin{aligned} & (\Lambda X. \lambda x: X. X) [\mathbb{Z}] 1 \\ \rightarrow_{\beta} & (\lambda x: \mathbb{Z}. x) 1 \\ \rightarrow_{\beta} & 1 \end{aligned}$$



# *Operational semantics for System F*

Operational semantics for the simply-typed  $\lambda$  calculus.

$$(\lambda v:t.e_1) e_2 \rightarrow_{\beta} e_1[e_2/v]$$

Additional reductions for System F.

$$(\Lambda X.e) T \rightarrow_{\beta} e[T/X]$$



# Typing rules for System F

Typing rules for the simply-typed  $\lambda$  calculus.

$$\frac{}{\Gamma, x:t \vdash x : t} \text{ var}$$

$$\frac{\Gamma \vdash e_1 : (s \rightarrow t) \quad \Gamma \vdash e_2 : s}{\Gamma \vdash (e_1 e_2) : t} \text{ app}$$

$$\frac{\Gamma, x:s \vdash t}{\Gamma \vdash (\lambda x:s.e) : (s \rightarrow t)} \text{ abs}$$



# Typing rules for System F

Additional rules for System F.

$$\frac{\Gamma, X \vdash e : T}{\Gamma \vdash (\Lambda X.e) : (\forall X.T)} \quad \text{all intro}$$

$$\frac{\Gamma \vdash e : \forall X.T_2}{\Gamma \vdash e[T_1] : T_2[T_1/X]} \quad \text{all elim}$$

