Type and Propositions

Yu Zhang

Course web site: http://staff.ustc.edu.cn/~yuzhang/tpl

Outline

- Curry-Howard Isomorphism
  - Constructive Logic
  - Classical Logic
- Logic Programming
  - Datalog
- Hoare Logic

References

- PFPL
  - Chapter 12 Constructive Logic
  - Chapter 13 Classic Logic
- Concepts in Programming Languages
  - Chapter 15 The Logic Programming Paradigm and Prolog
- Datalog
  - 15-819K: Logic Programming Lecture 26

Types and Propositions

- Types and programs
  - Type: specify a behavior; Program: implement a behavior
  - Statics: relate a program to the *type* it implements
  - Dynamics: relate a program to its *simplification* by an execution step
- Propositions and proofs
  - Proposition: pose a problem; Proof: solve a problem
  - Formal logical system: relate a proof to the *proposition* it proves
  - Proof reduction: relate *equivalent* proofs

Curry-Howard Isomorphism

Propositions as types principle:

Identify **propositions** with **type** and **proofs** with **programs**

- A proposition is the type of its proofs, and a proof is a program of that type.
- Every theorem has computational content, its proof viewed as a program.
- Every program has mathematical content, i.e., the proof that the program represents.

Concepts in PLs ↔ Concepts in Logics

Logics

- Constructive logic 构造逻辑
  - Judgements: ϕ **prop** means ϕ is a proposition; ϕ **true** means the proposition ϕ is true
  - ϕ **true** exactly when ϕ has a proof. $\neg\phi$ **true** exactly when ϕ has a refutation.
    - $\phi \vee \neg\phi$ **true** is not universally valid.
- Classical logic 经典逻辑
  - Every proposition is either true or false
  - The law of the excluded middle (排中律)
    - $\phi \vee \neg\phi$ **true** is valid for all propositions ϕ.

Constructive Logic

- Structural properties of the hypothetical judgement
  - Γ *is a set of hypotheses.*

$$\frac{}{\Gamma, \phi \text{ true} \vdash \phi \text{ true}} \qquad \frac{\Gamma, \phi \text{ true}, \phi \text{ true} \vdash \theta \text{ true}}{\Gamma, \phi \text{ true} \vdash \theta \text{ true}}$$

$$\frac{\Gamma \vdash \phi \text{ true} \quad \Gamma, \phi \text{ true} \vdash \psi \text{ true}}{\Gamma \vdash \psi \text{ true}} \qquad \frac{\Gamma, \psi \text{ true}, \phi \text{ true}, \Gamma' \vdash \theta \text{ true}}{\Gamma, \phi \text{ true}, \psi \text{ true}, \Gamma' \vdash \theta \text{ true}}$$

$$\frac{\Gamma \vdash \psi \text{ true}}{\Gamma, \phi \text{ true} \vdash \psi \text{ true}}$$

- Propositional logic
  - Syntax $\phi ::= \top \mid \bot \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \phi_1 \Rightarrow \phi_2$
    - $\top$ true, $\bot$ false, $\wedge$ conjunction, $\vee$ disjunction, $\Rightarrow$ implication

Constructive Logic: Rules

- Rules
  - Introduction rules: a ***"direct" proof*** of a proposition formed from a given ***connective***
  - Elimination rules: exploit the existence of such a proof in an "indirect" proof of another proposition
- The principle of conservation of proof 证明守恒原理
  - These rules are inverse to one another.
    - 消去规则只能抽取引入规则所引入的信息（证明形式）
    - 可以使用引入规则构造证明，供消去形式使用。

Constructive Logic: Rules

- Truth (no elimination) $\frac{}{\Gamma \vdash \top \text{ true}}$
- Conjunction
  - Intro. $\frac{\Gamma \vdash \phi \text{ true} \quad \Gamma \vdash \psi \text{ true}}{\Gamma \vdash \phi \wedge \psi \text{ true}}$
  - Elim. $\frac{\Gamma \vdash \phi \wedge \psi \text{ true}}{\Gamma \vdash \phi \text{ true}}$ $\frac{\Gamma \vdash \phi \wedge \psi \text{ true}}{\Gamma \vdash \psi \text{ true}}$
- Implication
  - Intro. $\frac{\Gamma, \phi \text{ true} \vdash \psi \text{ true}}{\Gamma \vdash \phi \Rightarrow \psi \text{ true}}$
  - Elim. $\frac{\Gamma \vdash \phi \Rightarrow \psi \text{ true} \quad \Gamma \vdash \phi \text{ true}}{\Gamma \vdash \psi \text{ true}}$

Constructive Logic: Rules

- Falsehood(no intro.) $\frac{\Gamma \vdash \bot \text{ true}}{\Gamma \vdash \phi \text{ true}}$
- Disjunction
  - Intro. $\frac{\Gamma \vdash \phi \text{ true}}{\Gamma \vdash \phi \vee \psi \text{ tru}}$ $\frac{\Gamma \vdash \psi \text{ true}}{\Gamma \vdash \phi \vee \psi \text{ true}}$
  - Elim.

$$\frac{\Gamma \vdash \phi \vee \psi \text{ true} \quad \Gamma, \phi \text{ true} \vdash \theta \text{ true} \quad \Gamma, \psi \text{ true} \vdash \theta \text{ true}}{\Gamma \vdash \theta \text{ true}}$$

Rules of Proof

- Key to the ***propositions-as-types*** principle

  Make the forms of proof explicit
  - The basic judgement ϕ true, which states that ϕ has a proof, is replaced by the judgement p : ϕ, stating that p is a proof of ϕ. (Sometimes p is called a "proof term", but we will simply call p a "proof.")
  - Hypothetical judgement: $x_1 : \phi_1, \ldots, x_n : \phi_n \vdash p : \phi$
- The rules of constructive propositional logic may be restated using proof terms.

Rules of Proof

- Truth $\frac{}{\Gamma \vdash \texttt{trueI} : \top}$
- Conjunction

$$\frac{\Gamma \vdash p : \phi \quad \Gamma \vdash q : \psi}{\Gamma \vdash \mathbf{andI}(p, q) : \phi \wedge \psi} \qquad \frac{\Gamma \vdash p : \phi \wedge \psi}{\Gamma \vdash \mathbf{andE}[\mathrm{l}](p) : \phi} \qquad \frac{\Gamma \vdash p : \phi \wedge \psi}{\Gamma \vdash \mathbf{andE}[\mathrm{r}](p) : \psi}$$

- Implication

$$\frac{\Gamma, x : \phi \vdash p : \psi}{\Gamma \vdash \mathbf{impI}[\phi](x.p) : \phi \Rightarrow \psi} \qquad \frac{\Gamma \vdash p : \phi \Rightarrow \psi \quad \Gamma \vdash q : \phi}{\Gamma \vdash \mathbf{impE}(p, q) : \psi}$$

- Falsehood $\frac{\Gamma \vdash p : \bot}{\Gamma \vdash \texttt{falseE}[\phi](p) : \phi}$
- Disjunction

$$\frac{\Gamma \vdash p : \phi}{\Gamma \vdash \mathbf{orI}[\mathrm{l}][\psi](p) : \phi \vee \psi} \qquad \frac{\Gamma \vdash p : \psi}{\Gamma \vdash \mathbf{orI}[\mathrm{r}][\phi](p) : \phi \vee \psi}$$

$$\frac{\Gamma \vdash p : \phi \vee \psi \quad \Gamma, x : \phi \vdash q : \theta \quad \Gamma, y : \psi \vdash r : \theta}{\Gamma \vdash \mathbf{orE}[\phi; \psi](p, x.q, y.r) : \theta}$$

Propositions as Types

- Proposition ϕ and its type ϕ^*

Prop.	Type	
$\top$	unit	空积类型
$\bot$	void	空和类型
$\phi \wedge \psi$	$\phi^* \times \psi^*$	二元积类型
$\phi \vee \psi$	$\phi^* + \psi^*$	二元和类型
$\phi \Rightarrow \psi$	$\phi^* \rightarrow \psi^*$	函数类型

Proofs as Programs

证明	程序	
trueI	$*$	空积的引入
falseE[ϕ](p)	$Zero^{\phi^*}(p^*)$	空和的消去
andI(p, q)	$\langle p^*, q^* \rangle$	二元积的引入
andE[l](p)	$\mathrm{Proj}_1(p^*)$	二元积的消去
andE[r](p)	$\mathrm{Proj}_2(p^*)$	二元积的消去
impI[ϕ]($x.p$)	$\lambda x{:}\phi^*.p^*$	函数类型的引入
impE[ϕ](p, q)	$p^*\ q^*$	函数类型的消去
orI[l][ψ](p)	$\mathrm{Inleft}(p^*)$	二元和的引入
orI[r][ϕ](p)	$\mathrm{Inright}(p^*)$	二元和的引入
orE[ϕ;ψ](p, $x.q$,$y.t$)	Case $p^*(\lambda x{:}\phi^*.q^*)(\lambda y{:}\psi^*.r^*)$	二元和的消去

Curry-Howard Isomorphism

- Theorem
  1. If ϕ prop， then ϕ^* type；
  2. If $\Gamma \vdash p : \phi$, then $\Gamma^* \vdash p^* : \phi^*$ 。
  - 反映出命题和类型,以及证明和程序之间的**静态**对应关系
  - 进一步扩展得到**动态**对应关系：消去形式是引入形式的后逆

$\mathrm{andE[l](andI}(p, q)) \equiv p$

$\mathrm{andE[r](andI}(p, q)) \equiv q$

$\mathrm{impE}[\phi](\mathrm{impI}[\phi](x.p), q) \equiv [q/x]p$

$\mathrm{orE}[\phi;\psi](\mathrm{orI[l]}[\psi](p), x.q, y.r) \equiv [p/x]q$

$\mathrm{orE}[\phi;\psi](\mathrm{orI[r]}[\phi](p), x.q, y.r) \equiv [p/y]r$

Classical Logic

- Judgements
  - ϕ true：表示ϕ是一个真命题
  - ϕ false：表示ϕ是一个假命题
  - #：表示一个已经推导出的矛盾
  - Hypothetical judgement

    $\phi_1 \text{ false}, \cdots, \phi_m \text{ false}; \psi_1 \text{ true}, \cdots, \psi_n \text{ true} \vdash J$

  - J 是上述三种断言之一，用Γ表示为真的假设集，Δ表示为假的假设集

$$\frac{\Delta\Gamma \vdash \phi \text{ false} \qquad \Delta\Gamma \vdash \phi \text{ true}}{\Delta\Gamma \vdash \#}$$

$$\frac{}{\Delta, \phi \text{ false } \Gamma \vdash \phi \text{ false}}$$

$$\frac{}{\Delta\ \Gamma, \phi \text{ true} \vdash \phi \text{ true}}$$

Classical Logic

- Semantics (omitted in this class)
  - Learn by yourself if you are interested

Logic Programming

Logic Programming

- Logic program
  - Facts
  - Rules for deducing further facts

```
likes(john, mary).
likes(mary, bethany).
likes(X, Y) :- likes(Y, X).
likes(X, Z) :- likes(X, Y), likes(Y, Z).
```

- Datalog engine: http://abcdatalog.seas.harvard.edu/

Unification

- Query
  - likes(john, X)?
- Query engine
  - Check the query against every deducible fact to see if they match
- Unification
  - A procedure that attempts to produce a substitution that makes two terms equal.

    [X → mary] likes(john, X) = likes(john, mary)

Applications

- Family trees

```
parent(john, mary).
parent(bethany, john).
grandparent(X, Z) :- parent(X, Y), parent(Y, Z).
```

```
married(bethany, luke).
parent(X, Z) :- married(X, Y), parent(Y, Z)
```


Implementation

- Saturation
- Unification

Logic Programming

- About the assignment 4: Logic Engine
  1. Represent and implement *List* in both ML and Lua
  2. Do Part 1 of the assignment 4
  3. Learn and Practice Datalog
  4. Do Part 2 of the assignment 4