第二次习题课

Simply-Typed Lambda Calculus 和 Hoare Logic

…还有之外的世界

复习：逻辑基础

- 课程目标：讨论程序行为和正确性
- 要说正确性的地方就要有逻辑
- 有逻辑就要有模型：形式语义
- 直觉主义逻辑：证明的可构造性（constructivity）
- 函数式代码与证明对象（proof object）

展望

类型系统（TYPE SYSTEM）

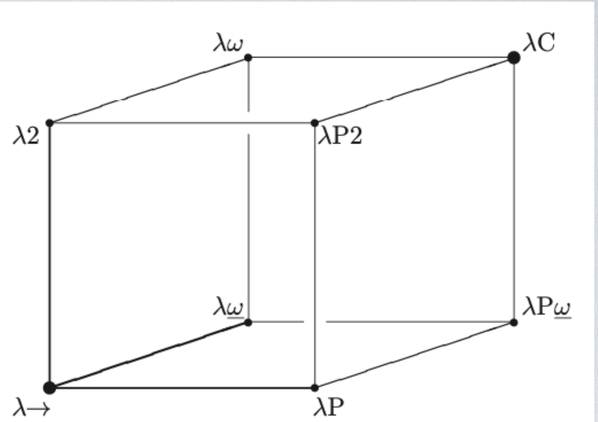
primitive values	b	::=	$n \mid \mathbf{true} \mid \mathbf{false} \mid \mathbf{null}$
terms	e	::=	$b \mid x \mid e_1\, e_2 \mid \lambda x{:}\tau.\, e$
primitive types	B	::=	$\mathbf{int} \mid \mathbf{bool} \mid 1$
types	τ	::=	$\tau_1 \to \tau_2 \mid B$

$$\Gamma \vdash n : \mathbf{int} \qquad \Gamma \vdash \mathbf{true} : \mathbf{bool} \qquad \Gamma \vdash \mathbf{false} : \mathbf{bool} \qquad \Gamma \vdash \mathbf{null} : 1 \qquad \frac{\Gamma(x) = \tau}{\Gamma \vdash x : \tau}$$

$$\frac{\Gamma \vdash e_0 : \tau \to \tau' \quad \Gamma \vdash e_1 : \tau}{\Gamma \vdash e_0\, e_1 : \tau'} \qquad \frac{\Gamma,\, x{:}\tau \vdash e : \tau'}{\Gamma \vdash (\lambda x{:}\tau.\, e) : \tau \to \tau'}$$

- STLC really is simple
- 类型：根据动态行为的近似给程序“大概”分个类
- “大概”：保守性（conservative）与正确性（soundness）

- 类型论（type theory）和类型系统（type system）
- 类型的逻辑对应：知识的双向传递
- 类型的语义解释：值（value）的“集合”
- 静态类型安全：语形法（syntactic approach）和语义法（logical approach）
- 类型与抽象（abstraction）：三种可能
- 类型的实际应用：丰富多彩；两个维度
- 类型检查的时机：静态（static）和动态（dynamic）
- 类型注释（type annotation）：显式（explicit）或隐式（implicit）



with terms depending on types: $\lambda 2$,
with types depending on types: $\lambda\underline{\omega}$,
with types depending on terms: λP.

程序验证（PROGRAM VERIFICATION）

- 程序：描述怎么做计算的东西（how）
- 验证：规约（specification）和证明（proof）
- 能验证什么：性质；应用范围
- 用什么验证

$$\frac{}{\{\ulcorner\urcorner\}\ v\ \{\lambda x.\ulcorner x = v\urcorner\}}$$

$$\frac{\{H\}\ t_1\ \{Q'\} \qquad \forall x.\ \{Q'\ x\}\ t_2\ \{Q\}}{\{H\}\ (\mathsf{let}\ x = t_1\ \mathsf{in}\ t_2)\ \{Q\}}$$

$$\frac{v = \mathsf{true} \Rightarrow \{H\}\ t_1\ \{Q\} \qquad v = \mathsf{false} \Rightarrow \{H\}\ t_2\ \{Q\}}{\{H\}\ (\mathsf{if}\ v\ \mathsf{then}\ t_1\ \mathsf{else}\ t_2)\ \{Q\}}$$

$$\frac{\begin{array}{c}\forall f.\ Pf \Rightarrow \{H\}\ t_2\ \{Q\} \\ Pf = (\forall x H' Q'.\ \{H'\}\ t_1\ \{Q'\} \Rightarrow \{H'\}\ (f\ x)\ \{Q'\})\end{array}}{\{H\}\ (\mathsf{let\ rec}\ f\ x = t_1\ \mathsf{in}\ t_2)\ \{Q\}}$$

- 和（简单）类型系统的区别与联系：更多人工指导和精确性
- 依赖类型语言（dependently-typed language）：证明和验证的结合
- 语言层面的验证支持（language-integrated）和嵌入后验证（embedding）
- 验证框架的信赖代码库大小（trusted code base）：如何验证验证的结果？证书（certificates）
- 断言逻辑（assertion logic）和程序逻辑（program logic）
- 正确性逻辑（correctness logic）和非正确性逻辑（incorrectness logic）
- 带来困难的语言特性：复杂控制流（non-local control flow）；可变性（mutability）；别名（aliasing）；高阶状态（higher-order store）；并发（concurrency）
- 验证的工程维度：理论上足够不等于实用；证明工程（proof engineering）
- 分离逻辑（separation logic）：还原推理的局部性（local reasoning）
- 证明合成（proof synthesis）：对比程序合成（program synthesis）

答疑

- Coq的底层基础：CoC + (co)inductive types + other extensions
- 类型参数（parameter）与类型指标（index）
- 高阶逻辑（higher-order logic）
- 其他?