Crash Course of Separation Logic

崩溃课程的分离逻辑（浅入浅出版）

Jiaqi Wu

SIGMATH, SAST, NJUPT

March 22, 2025

First-order logic

proposition	P
conjunction	$P \land Q$
disjunction	$P \lor Q$
universal quantification	$\forall x\, F\, x$
existential quantification	$\exists x\, F\, x$

Example 1

Let R be a binary relation, we can define its transitivity as the following proposition:

$$\forall x\, \forall y\, \forall z\ (R\, x\, y \land R\, y\, z \to R\, x\, z)$$

Second-order logic

universal quantification over properties $\forall F\, F\, x$

existential quantification over properties $\exists F\, F\, x$

Example 2

Some random proposition:

$$\forall R \forall a \forall b\ (R\, a\, b \to R\, b\, a) \to (T\, a\, b \to T\, b\, a)$$

Ex. Is there any even higher order logic?

Expressive power

Example 3

Let R be a binary relation. We define the ancestral of R, $R^\star$, such that for every two elements x and y, $R^\star\,x\,y$ holds iff any of the following holds:

- $R\,x\,y$,
- $\exists a\ (R\,x\,a \wedge R\,a\,y)$,
- $\exists a\,\exists b\ (R\,x\,a \wedge R\,a\,b \wedge R\,b\,y)$,
- $\cdots$

We can write out the definition of $R^\star\,a\,b$ with second-order logic:

$$\forall F\ ((\forall x\ (R\,a\,x \rightarrow F\,x) \wedge \forall x\,\forall y\ ((F\,x \wedge R\,x\,y) \rightarrow F\,y)) \rightarrow F\,b)$$

But there's no way to express $R^\star$ in first-order logic! (w/o set theory)
(Ex. Define $R^\star$ using first-order logic and the symbol $\in$ from set theory.)

Classical logic and constructive logic

Proposition 1 (Law of excluded middle)

$$\forall P\ (P \vee (\neg P))$$

Example 4

There exists irrational numbers a and b such that a^b is rational.

Proof.

It is not difficult to prove that $\sqrt{2}$ is irrational.

If ${\sqrt{2}}^{\sqrt{2}}$ is rational, then let $a = b = \sqrt{2}$ therefore $a^b = {\sqrt{2}}^{\sqrt{2}}$ is rational.

Otherwise let $a = {\sqrt{2}}^{\sqrt{2}}$ and $b = \sqrt{2}$, then $a^b = \left({\sqrt{2}}^{\sqrt{2}}\right)^{\sqrt{2}} = 2$ is rational. □

Review: Simply typed λ-calculus ($\lambda_\rightarrow$)

term	t
abstraction	$\lambda x : T.t$
application	$t_1 \; t_2$
type of functions	$T_1 \rightarrow T_2$
typing context	Γ

Example 5

$$\dfrac{\dfrac{\Gamma, x : T_1 \vdash t_1 : T_2}{\Gamma \vdash \lambda x : T_1.t_1 : T_1 \rightarrow T_2} \qquad \Gamma \vdash t_2 : T_1}{\Gamma \vdash (\lambda x : T_1.t_1) \;\; t_2 : T_2}$$

Curry-Howard correspondence (simpl.)

Logic	Computation
Proposition	Type
Proof	Term
Conjunction $\wedge$	Product Type $\times$
Disjunction $\vee$	Sum Type $+$
Implication $\rightarrow$	Function Type $\rightarrow$
True $\top$	Unit (Single element type)
False $\perp$	Never (Zero element type)

Dependent type

or intuitionistic type theory, or constructive type theory, or Per Matin-Löf type theory, or intuitionistic logic, or constructive logic, or what ever

We treat types and terms equally and types can now be defined dependently on terms!

pi type	$\Pi x : \mathcal{U}.t$
sigma type	$\Sigma x : \mathcal{U}.t$
identity type	$t_1 = t_2$

Example 6

$$\text{head} : \Pi T : \mathcal{U}.\Pi n : \text{nat}.\text{Vec}\, T\ (n+1) \to T$$
$$\text{concat} : \Pi T : \mathcal{U}.\Pi n, m : \text{nat}.\text{Vec}\, T\, n \to \text{Vec}\, T\, m \to \text{Vec}\, T\ (n+m)$$

Ex. A common library function in the C programming language is merely dependently typed, what is it?

Universe

Russell's paradox

$$\{x \in \text{set of all sets} \mid x \notin x\}$$

Universe hierarchy: $x \in \mathcal{U}_0 \in \mathcal{U}_1 \in \mathcal{U}_2 \in \cdots$

$\mathcal{U}_i$ is *small* in $\mathcal{U}_{i+1}$.

Not Russell's paradox

$$\{x \in \mathcal{U}_i \mid x \notin x\} \notin \mathcal{U}_i \text{ but } \in \mathcal{U}_{i+1}$$

Curry-Howard correspondence

Logic	Computation
Proposition	Type
Proof	Term
Conjunction $\wedge$	Product Type $\times$
Disjunction $\vee$	Sum Type $+$
Implication $\rightarrow$	Function Type $\rightarrow$
True $\top$	Unit (Single element type)
False $\bot$	Never (Zero element type)
Universal Quantification $\forall$	Pi Type Π
Existential Quantification $\exists$	Sigma Type Σ

Function type is a specialized form of pi type, therefore implication is actually universal quantification where elements range over proofs!

Hoare logic

command	t
precondition	P
postcondition	Q
hoare triple	$\{P\}\, t\, \{Q\}$

Example 7

$$\{\top\}\, a := 114\, \{a = 114\}$$
$$\{b = 514\}\, \textsf{skip}\, \{b = 514\}$$
$$\{x = 2\}\, x := x + 1\, \{x = 3\}$$
$$\{\top\}\, \textsf{while true do skip end}\, \{\bot\}$$

Rules of Hoare logic

$$\frac{\{P\}\, t_1\, \{Q\} \quad \{Q\}\, t_2\, \{R\}}{\{P\}\, t_1; t_2\, \{R\}}\ \textsc{Hoare-Sequence}$$

$$\frac{P \vdash P' \quad \{P'\}\, t\, \{Q'\} \quad Q' \vdash Q}{\{P\}\, t\, \{Q\}}\ \textsc{Hoare-Consequence}$$

$$\frac{\{P \wedge c\}\, t_1\, \{Q\} \quad \{P \wedge \neg c\}\, t_2\, \{Q\}}{\{P\}\, \mathsf{if}\ c\ \mathsf{then}\ t_1\ \mathsf{else}\ t_2\ \mathsf{end}\, \{Q\}}\ \textsc{Hoare-If}$$

$$\frac{\{P \wedge c\}\, t\, \{P\}}{\{P\}\, \mathsf{while}\ c\ \mathsf{do}\ t\ \mathsf{end}\, \{P \wedge \neg c\}}\ \textsc{Hoare-While}$$

Separation logic

Questions?

Thanks!