Syntactic Logical Relations for Perfect Encryption, Higher-Order References and First-Class Channels

Eijiro Sumii
University of Tokyo

What is a Logical Relation?

- A relation ? $v \sim w : \tau$ between values v and w in a typed λ-calculus, defined according to their type τ

E.g.,

- ? $i \sim j : \text{int} \iff i = j$
- ? $f \sim g : \sigma \to \tau \iff$
  ? $\text{eval}(f\ v) \sim \text{eval}(g\ w) : \tau$ for any ? $v \sim w : \sigma$
- ? $(v_1, v_2) \sim (w_1, w_2) : \tau_1 \times \tau_2 \iff$
  ? $v_1 \sim w_1 : \tau_1$ and ? $v_2 \sim w_2 : \tau_2$

What is it Useful for?

- To show various forms of equivalence between programs
  - Correctness of optimization
    - ? $p_{opt} \sim p_{unopt} : \tau$
  - Secrecy as non-interference
    - ? $p_v \sim p_w : \tau$ for $v \neq w$
  - Correspondence between CPS and direct style
    - ? $p_{CPS} \sim p_{DS} : \tau$

etc

The "Fundamental Property" of Logical Relations

Theorem:

$$? v \sim v : \tau \quad \text{for any} \quad ? v : \tau$$

Corollary:

$? v_1 \sim v_2 : \tau \;\Rightarrow\; \text{eval}(f\ v_1) = \text{eval}(f\ v_2)$
for any $? f : \tau \to \text{bool}$

I.e., logical relations imply observational equivalence

This Talk

- Logical relations for wider range of programming constructs
  - Perfect encryption [Sumii-Pierce 01]

    Cf. Type abstraction [Reynolds 83]
  - Higher-order references [ongoing work]
  - First-class channels [ongoing work]

Everything is syntactic and operational

Perfect Encryption

$M ::= \ldots$ (standard λ-terms)
  | k (key)
  | new k in M (key generation)
  | $\{M\}_N$ (encryption)
  | let $\{x\}_{M1} = M_2$ in N_1 else N_2 (decryption)

- Useful for reasoning about information hiding by encryption (as in security protocols)

Example of Equivalence by Perfect Encryption

new k in
 $(\{3\}_k, \lambda c.\ \mathrm{let}\ \{i\}_k = c\ \mathrm{in}\ (i \bmod 2)\ \mathrm{else}\ {-1})$

$$\cong$$

new k in
 $(\{5\}_k, \lambda c.\ \mathrm{let}\ \{i\}_k = c\ \mathrm{in}\ (i \bmod 2)\ \mathrm{else}\ {-1})$

Cf. equivalence by type abstraction

$$\mathrm{pack\ int}, (3, \lambda i.\ i \bmod 2)\ \mathrm{as}\ \exists\alpha.\ \alpha \times (\alpha \to \mathrm{int})$$

$$\cong$$

$$\mathrm{pack\ int}, (5, \lambda i.\ i \bmod 2)\ \mathrm{as}\ \exists\alpha.\ \alpha \times (\alpha \to \mathrm{int})$$

Logical Relation for Perfect Encryption [Sumii-Pierce 01]

- Introduce <u>relation environment</u> φ to associate each key k with a relation $\varphi(k)$ between values encrypted by k
- φ ? $\{v\}_k \sim \{w\}_{k'}$: bits $\Leftrightarrow$
  k = k' and $(v, w) \in \varphi(k)$
- φ ? new k in M ~ new k in N : τ $\Leftrightarrow$
  φ, k $\mapsto$ r ? M ~ N : τ for some r

Cf. Logical Relation for Type Abstraction [Reynolds 83]

Associate each abstract type α with a relation $\varphi(\alpha)$ between values implementing α

φ ? $v \sim w : \alpha \iff (v, w) \in \varphi(\alpha)$

φ ? pack ρ, v as $\exists\alpha.\ \tau$
$\sim$ pack σ, w as $\exists\alpha.\ \tau : \exists\alpha.\ \tau \iff$
$\varphi, \alpha \mapsto r$? $v \sim w : \tau$
for some relation $r \subseteq \rho \times \sigma$

Extended Logical Relation: Motivating Example

new k_1 in new k_2 in
  $(\{k_2\}_{k1}, \lambda c.$ let $\{k_2'\}_{k1} = c$ in $\{3\}_{k2'}$ else ...)

$$\cong$$

new k_1 in new k_2 in
  $(\{k_2\}_{k1}, \lambda c.$ let $\{k_2'\}_{k1} = c$ in $\{5\}_{k2'}$ else ...)

What to take as $\varphi(k_1)$?

k_2 is yet to be generated!

Extended Logical Relation: Our Solution

Parameterize φ with respect to a relation environment ψ in the future

φ ? $\{v\}_k \sim \{w\}_{k'}$: bits $\Leftrightarrow$
$k = k'$ and $(v, w) \in \varphi_\varphi(k)$

E.g., take

$\varphi_\psi(k_1) = \{ (k_2, k_2) \mid \psi_\psi(k_2) = \{(3, 5)\} \}$

in the motivating example

References

M ::= ... (standard λ-terms)
| ℓ (location)
| let ℓ = ref M in N (cell allocation)
| M := N (update)
| !M (dereference)

Example of Equivalence by References

$$\text{let } \ell = \text{ref } 0 \text{ in } (\lambda x.\ !\ell,\ \lambda y.\ \ell := !\ell + 2)$$

$$\cong$$

$$\text{let } \ell = \text{ref } 0 \text{ in } (\lambda x.\ !\ell \times 2,\ \lambda y.\ \ell := !\ell + 1)$$

Logical Relation for First-Order References

Associate each location ℓ with a relation $\varphi(\ell)$ between values stored in ℓ

φ ? let ℓ = ref v in M
$\sim$ let ℓ = ref w in N : τ $\Leftrightarrow$
$\varphi, \ell \mapsto r$? M $\sim$ N : τ for some r $\ni$(v, w)

φ ? (ℓ := v) $\sim$ (ℓ' := w) : unit $\Leftrightarrow$
$\ell = \ell'$ and (v, w) $\in \varphi(\ell)$

Logical Relation for Higher-Order References

What about "references to references"?

— The same as "keys encrypting keys"!

(I don't have so interesting examples, though)

Channels

M ::= ... (standard λ-terms)
| c (channel)
| new c in M (channel creation)
| send M to N (output)
| recv x from M in N (input)

cf. π-calculus [Milner 89]

Example of Equivalence by Channels

new c in
  (send 3 to c, recv i from c in (i mod 2))

$$\cong$$

new c in
  (send 5 to c, recv i from c in (i mod 2))

Logical Relation for Second-Class Channels

Associate each channel c with a relation $\varphi(c)$ between values communicated through c

φ ? new c in M $\sim$ new c in N : τ $\Leftrightarrow$
φ, c $\mapsto$ r ? M $\sim$ N : τ for some r

φ ? send v to c $\sim$ send w to c' : unit $\Leftrightarrow$
c = c' and (v, w) $\in$ φ(c)

φ ? recv x from c in M
$\sim$ recv x from c' in N : τ $\Leftrightarrow$
c = c' and φ ? [v/x]M $\sim$ [w/x]N : τ
for any (v, w) $\in$ φ(c)

Logical Relation for First-Class Channels

- What about "channels passing channels"?
  — Again, the same as keys encrypting keys

- More interesting (than references to references) because first-class channels are essential in π-calculus

A Use of First-Class Channels: Client-Server System

```
new succserv in
  (recv (m, c) from succserv in
  send (m + 1) to c,
  new d in
     (send (2, d) to succserv,
      recv n from d in ...))
```

Or, Equivalently...

new idserv in
  (recv (m, c) from idserv in
  send m to c,
  new d in
    (send (3, d) to idserv,
    recv n from d in ...))

To show the equivalence, take
  $\varphi_\psi(\text{idserv}) =$
    $\{ ((2, c), (3, c)) \mid \psi_\psi(c) = \{(3, 3)\} \}$

Conclusion (1/2): Summary

- We have seen logical relations for
  - Perfect encryption

    Cf. type abstraction
  - Higher-order references
  - First-class channels

All of these are based on the same idea: associating each <u>generative name</u> n with a relation φ(n) between values involved in n

Conclusion (2/2): Future Work

- More applications (other than security protocols)
- Soundness proofs (except for logical relations for encryption)
- Completeness results
- Comparison with other methods (such as bisimulation)

Suggestions and discussions welcome!