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

E.g.,

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

What is it Useful for?

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

  etc.

The "Fundamental Property" of Logical Relations

Theorem:

$$\vdash v \sim v : \tau \text{ for any } \vdash v : \tau$$

Corollary:

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

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

$\cong$

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

Cf. equivalence by type abstraction

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

$\cong$

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

Logical Relation for Perfect Encryption [Sumii-Pierce 01]

Introduce relation environment φ to associate each key k with a relation $\varphi(k)$ between values encrypted by k

$\varphi \vdash \{v\}_k \sim \{w\}_{k'} : \text{bits} \iff$
$k = k'$ and $(v, w) \in \varphi(k)$

$\varphi \vdash \text{new } k \text{ in } M \sim \text{new } k \text{ in } N : \tau \iff$
$\varphi, k \mapsto r \vdash M \sim N : \tau$ 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

$\varphi \vdash v \sim w : \alpha \Leftrightarrow (v, w) \in \varphi(\alpha)$

$\varphi \vdash \mathsf{pack}\ \rho, v\ \mathsf{as}\ \exists\alpha.\ \tau$
$\sim \mathsf{pack}\ \sigma, w\ \mathsf{as}\ \exists\alpha.\ \tau : \exists\alpha.\ \tau \Leftrightarrow$
$\varphi, \alpha \mapsto r \vdash v \sim w : \tau$
for some relation $r \subseteq \rho \times \sigma$

Extended Logical Relation: Motivating Example

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

$\cong$

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

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

$\varphi \vdash \{v\}_k \sim \{w\}_{k'} : \text{bits} \Leftrightarrow$
$k = k'$ 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

$\varphi \vdash \text{let } \ell = \text{ref } v \text{ in } M$
$\sim \text{let } \ell = \text{ref } w \text{ in } N : \tau \Leftrightarrow$
$\varphi, \ell \mapsto r \vdash M \sim N : \tau$ for some $r \ni (v, w)$

$\varphi \vdash (\ell := v) \sim (\ell' := w) : \text{unit} \Leftrightarrow$
$\ell = \ell'$ 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

$\varphi \vdash \text{new } c \text{ in } M \sim \text{new } c \text{ in } N : \tau \iff$
$\varphi, c \mapsto r \vdash M \sim N : \tau$ for some r

$\varphi \vdash \text{send } v \text{ to } c \sim \text{send } w \text{ to } c' : \text{unit} \iff$
$c = c'$ and $(v, w) \in \varphi(c)$

$\varphi \vdash \text{recv } x \text{ from } c \text{ in } M$
$\sim \text{recv } x \text{ from } c' \text{ in } N : \tau \iff$
$c = c'$ and $\varphi \vdash [v/x]M \sim [w/x]N : \tau$
for any $(v, w) \in \varphi(c)$

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

φ_ψ(idserv) =

{ ((2, c), (3, c)) | ψ_ψ(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 generative name n with a relation $\varphi(n)$ 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!