Proving ML Contextual Equivalences via "Logical" Relations

[OS & PE, section 5]

Proving ML Contextual Equivalences via (Kripke) "Logical" Relations

[OS & PE, section 5]

$p \triangleq$

```
let a = ref 0 in
fun(x : int) -> (a := !a + x ; !a)
```

$m \triangleq$

```
let b = ref 0 in
fun(y : int) -> (b := !b - y ; 0 - !b)
```

Are these Caml expressions (of type `int -> int`) contextually equivalent?

Yes! Recall the informal argument why $p =_{dx} m$ holds...

‘Logical’ simulation relation between ML programs, parameterised by state-relations

For each state-relation $r \in \mathbf{Rel}(w_1, w_2)$ we can define relations

$$e_1 \leq_r e_2 : ty \qquad (e_1 \in \mathbf{Prog}_{ty}(w_1), e_2 \in \mathbf{Prog}_{ty}(w_2))$$

(for each type ty), with the properties (I), (II) & (III) . . .

Kripke-style worlds: $w_1, w_2, \ldots$ are finite sets of locations.

States in world w: $\mathbf{St}(w) \triangleq \mathbb{Z}^w$. Programs in world w:

$$\mathbf{Prog}_{ty}(w) \triangleq \{\, e \in \mathbf{Prog}_{ty} \mid loc(e) \subseteq w \,\}.$$

State-relations: $r, r', \ldots \in \mathbf{Rel}(w_1, w_2)$ are non-empty* subsets of $\mathbf{St}(w_1) \times \mathbf{St}(w_2)$.

[* simplifies def? of $\leq_r$ at ground types]

Growth of the state

If $s, e \Rightarrow v, s'$

then $dom(s) \subseteq dom(s')$

E.g.

$\emptyset, p \Rightarrow (fun(x:int) \rightarrow l := !l+1; !l), \{l \mapsto 0\}$

$dom(\emptyset) = \emptyset \subseteq \{l\} = dom\ \{l \mapsto 0\}$

(Ⅲ) The relationship between $\leq_r$ and contextual equivalence

For all types ty, finite sets w of locations, and programs $e_1, e_2 \in \mathbf{Prog}_{ty}(w)$

$$e_1 \leq_{\mathbf{ctx}} e_2 : ty \quad \text{iff} \quad e_1 \leq_{id_w} e_2 : ty$$

where $id_w \in \mathbf{Rel}(w, w)$ is the identity state-relation for w:

$$id_w \triangleq \{\ (s, s) \mid\ s \in \mathbf{St}(w)\ \}.$$

Hence e_1 and e_2 are contextually equivalent iff both $e_1 \leq_{id_w} e_2 : ty$ and $e_2 \leq_{id_w} e_1 : ty$.

(I) The simulation property of $\leq_r$

To prove $e_1 \leq_r e_2 : ty$, it suffices to show that whenever

$$\begin{cases} (s_1, s_2) \in r \\ s_1, e_1 \Rightarrow v_1, s_1' \end{cases}$$

then there exists $r' \rhd r$ and v_2, s_2' such that

$$\begin{cases} s_2, e_2 \Rightarrow v_2, s_2' \\ (s_1', s_2') \in r' \end{cases}$$

and $v_1 \leq_{r'} v_2 : ty$.

This uses the notion of extension of state-relations:

$r' \rhd r$ holds iff $r' = r \otimes r''$ for some r''—see Definition 5.1.

Given $\begin{cases} r \in Rel(W_1, W_2) \\ r' \in Rel(W_1', W_2') \end{cases}$, $\boxed{r' \rhd r}$ means :

[see also Ex. B.5]

- $W_1' \supseteq W_1$
- $W_2' \supseteq W_2$
- $r' = r \otimes r''$ for some $r'' \in Rel(W_1' - W_1, W_2' - W_2)$

where $r \otimes r'' \triangleq \{(s_1 s_1'', s_2 s_2'') \mid (s_1, s_2) \in r \;\&\; (s_1'', s_2'') \in r''\}$

given states s & s', ss' is the state with
$dom(ss') = dom(s) \cup dom(s')$
and $(ss')(l) = \begin{cases} s'(l) & \text{if } l \in dom(s') \\ s(l) & \text{if } l \in dom(s) - dom(s') \end{cases}$

Recall:

Canonical forms (a.k.a. values) = closed expressions in the subset of all expressions generated by the grammar

$$
\begin{array}{lll}
v ::= & x & (x, f \in \mathit{Var}) \\
& f & \\
& \mathit{true} & \\
& \mathit{false} & \\
& n & (n \in \mathbb{Z}) \\
& () & \\
& v, v & \\
& \mathit{fun}\,(x : \mathit{ty}) \to e & \\
& \mathit{fun}\ f = (x : \mathit{ty}) \to e & \\
& l & (l \in \mathit{Loc})
\end{array}
$$

(II) The extensionality properties of $\leq_r$ on canonical forms

- For $ty \in \{\texttt{bool}, \texttt{int}, \texttt{unit}\}$, $v_1 \leq_r v_2 : ty$ iff $v_1 = v_2$.
- $v_1 \leq_r v_2 : \texttt{int ref}$ iff $\texttt{!}v_1 \leq_r \texttt{!}v_2 : \texttt{int}$ and for all $\texttt{n} \in \mathbb{Z}$, $(v_1 \texttt{ := n}) \leq_r (v_2 \texttt{ := n}) : \texttt{unit}$.
- $v_1 \leq_r v_2 : ty_1 * ty_2$ iff $\texttt{fst}\, v_1 \leq_r \texttt{fst}\, v_2 : ty_1$ and $\texttt{snd}\, v_1 \leq_r \texttt{snd}\, v_2 : ty_2$.
- $v_1 \leq_r v_2 : ty_1 \texttt{ -> } ty_2$ iff for all $r' \rhd r$ and all v_1', v_2'

$$v_1' \leq_{r'} v_2' : ty_1 \;\supset\; v_1\, v_1' \leq_{r'} v_2\, v_2' : ty_2$$

The last property is characteristic of (Kripke) logical relations (Plotkin 1973; O'Hearn and Riecke 1995).

Outline of the proof of $p =_{\mathtt{ctx}} m : \mathtt{int} \to \mathtt{int}$ (cf. Slide 2)

$$\emptyset, p \Rightarrow (\mathtt{fun}(x : \mathtt{int}) \to \ell_1 := !\ell_1 + x \;;\; !\ell_1), \{\ell_1 \mapsto 0\}$$

$$\emptyset, m \Rightarrow (\mathtt{fun}(y : \mathtt{int}) \to \ell_2 := !\ell_2 - x \;;\; 0 - !\ell_2), \{\ell_2 \mapsto 0\}$$

Define

$$r \triangleq \{\, (s_1, s_2) \mid s_1(\ell_1) = -s_2(\ell_2) \,\} \in \mathbf{Rel}(\{\ell_1\}, \{\ell_2\}).$$

Then $r \rhd id_\emptyset$, $(\{\ell_1 \mapsto 0\}, \{\ell_2 \mapsto 0\}) \in r$, and from Slide 20

$$(\mathtt{fun}(x : \mathtt{int}) \to \ell_1 := !\ell_1 + x \;;\; !\ell_1) \;\leq_r$$
$$(\mathtt{fun}(y : \mathtt{int}) \to \ell_2 := !\ell_2 - x \;;\; 0 - !\ell_2) : \mathtt{int} \to \mathtt{int}.$$

So by Slide 19, $p \leq_{id_\emptyset} m : \mathtt{int} \to \mathtt{int}$.

Hence by Slide 21, $p \leq_{\mathtt{ctx}} m : \mathtt{int} \to \mathtt{int}$.

Similarly $m \leq_{\mathtt{ctx}} p : \mathtt{int} \to \mathtt{int}$.

Outline of the proof of $p =_{\mathtt{ctx}} m : \mathtt{int} \to \mathtt{int}$ (cf. Slide 2)

$\emptyset, p \Rightarrow (\mathtt{fun}(x : \mathtt{int}) \to \ell_1 := !\ell_1 + x \;;\; !\ell_1), \{\ell_1 \mapsto 0\}$ S

$\emptyset, m \Rightarrow (\mathtt{fun}(y : \mathtt{int}) \to \ell_2 := !\ell_2 - x \;;\; 0 - !\ell_2), \{\ell_2 \mapsto 0\}$ S'

Define

$$r \triangleq \{\, (s_1, s_2) \mid s_1(\ell_1) = -s_2(\ell_2) \,\} \in \mathbf{Rel}(\{\ell_1\}, \{\ell_2\}).$$

Then $r \rhd id_\emptyset$, $(\{\ell_1 \mapsto 0\}, \{\ell_2 \mapsto 0\}) \in r$, and from Slide 20

V $(\mathtt{fun}(x : \mathtt{int}) \to \ell_1 := !\ell_1 + x \;;\; !\ell_1) \leq_r$

V' $(\mathtt{fun}(y : \mathtt{int}) \to \ell_2 := !\ell_2 - x \;;\; 0 - !\ell_2) : \mathtt{int} \to \mathtt{int}.$

So by Slide 19, $p \leq_{id_\emptyset} m : \mathtt{int} \to \mathtt{int}$.

Hence by Slide 21, $p \leq_{\mathtt{ctx}} m : \mathtt{int} \to \mathtt{int}$.

Similarly $m \leq_{\mathtt{ctx}} p : \mathtt{int} \to \mathtt{int}$.

$v \leq_r v' : int \to int$

iff {slide 20}

$\forall r' \rhd r,\ \forall n, n' \in \mathbb{Z}.\ n \leq_{r'} n' : int \supset v n \leq_{r'} v' n' : int$

iff {slide 20}

$\forall r' \rhd r,\ \forall n \in \mathbb{Z}.\ v n \leq_{r'} v' n : int$

prove this via property (I)

$$r \triangleq \{(s_1, s_2) \mid s_1(l_1) = -s_2(l_2)\}$$
$$r' \rhd r \supset \quad vn \leq_{r'} v'n : \text{int}$$

Proof: If $(s_1, s_2) \in r'$ then since $r' \rhd r$, we have

$s_1(l_1) = -s_2(l_2) = k$, say

So $\left.\begin{array}{l} s_1, vn \Rightarrow n', s_1[l_1 \mapsto n'] \\ s_2, v'n \Rightarrow n', s_2[l_2 \mapsto -n'] \end{array}\right\}$ where $n' \triangleq k + n$

and $n' \leq_{r'} n' : \text{int}$ {slide 20}

$(s_1[l_1 \mapsto n'], s_2[l_2 \mapsto -n']) \in r'$ $\{r' \rhd r\}$

So $vn \leq_{r'} v'n : \text{int}$ holds by the simulation property of $\leq_r$ (slide 19).

Summary

- To prove $e_1 \leq_{ctx} e_2 : ty$, prove $e_1 \leq_{id_w} e_2 : ty$ instead (where $w \supseteq locs(e_1, e_2)$) — relying on property (III) of $\leq_r$
- Try to prove $e_1 \leq_{id_w} e_2 : ty$ via the simulation property (I) of $\leq_r$ and the extensionality properties (II) of $\leq_r$ on canonical forms.

Exercise (B.7, p 410)

If $f \in \mathrm{Prog}_{\mathrm{int} \to \mathrm{int}}$ satisfies

- $\mathrm{loc}(f) = \emptyset$
- $\emptyset, f\,n \Downarrow$ for all $n \in \mathbb{Z}$

then $f \cong_{ctx} \mathrm{memo\_f} : \mathrm{int} \to \mathrm{int}$ where

$$\mathrm{memo\_f} \triangleq \begin{array}{l} \mathtt{let\ a = ref\ 0\ in} \\ \mathtt{let\ r = ref(f\ 0)\ in} \\ \mathtt{fun\,(x : int) \to} \\ \quad \mathtt{(if\ x = !a\ then\ ()\ else\ a := x\,;\, r := f\,x)\,;} \\ \quad \mathtt{!r} \end{array}$$

Summary

- To prove $e_1 \leq_{ctx} e_2 : ty$, prove $e_1 \leq_{id_w} e_2 : ty$ instead (where $w \supseteq locs(e_1, e_2)$) — relying on property (III) of $\leq_r$

this is only a sufficient, not a necessary condition for $e_1 \leq_{ctx} e_2 : ty$ to hold — for example...

A nasty contextual equivalence

$$\texttt{let } a = \texttt{ref } 0 \texttt{ in } \texttt{fun } (f : \texttt{unit} \to \texttt{unit}) \to (a := 1 ; f() ; !a)$$

$$\cong_{ctx}$$

$$\texttt{fun } (f : \texttt{unit} \to \texttt{unit}) \to f() ; 1$$

$$: (\texttt{unit} \to \texttt{unit}) \to \texttt{int}$$

Valid, but can't use the simulation property (I) to prove it. (See [15], Example 5.9.)