Iris: Higher-Order Concurrent Separation Logic

Lecture 7: Later Modality

Lars Birkedal

Aarhus University, Denmark

November 14, 2017

Overview

Earlier:

- Operational Semantics of $\lambda_{\mathrm{ref,conc}}$
  - e, $(h, e) \rightsquigarrow (h, e')$, and $(h, \mathcal{E}) \to (h', \mathcal{E}')$
- Basic Logic of Resources
  - $l \hookrightarrow v$, $P * Q$, $P \mathrel{-\!\!*} Q$, $\Gamma \mid P \vdash Q$
- Basic Separation Logic
  - $\{P\}\, e\, \{v.Q\}$: Prop, isList l xs, ADTs, foldr

Today:

- Later Modality: $\triangleright$
- Necessary for working with *invariants* (defined later in the course)
- Key Points:
  - Löb rule: $(\triangleright P \Rightarrow P) \Rightarrow P$
  - Guarded recursively defined predicates: $\mu r.\, P$

Later Modality

- Recall the recursion rule:

$$\frac{\Gamma, f : Val \mid S \wedge \forall y.\, \forall v.\, \{P\}\, f\ v\, \{u.Q\} \vdash \forall y.\, \forall v.\, \{P\}\, e[v/x]\, \{u.Q\}}{\Gamma \mid S \vdash \forall y.\, \forall v.\, \{P\}\, (\mathsf{rec}\ f(x) = e) v\, \{u.Q\}}\ \textsc{Ht-Rec}$$

- This rule involves a kind of recursive reasoning.
- We mentioned earlier that this rule is *sound* because function application involves reduction steps, *i.e.*, we will only use the recursive assumption after some reduction steps have taken place.
- Later in the course, when discussing *invariants*, we will want to have other forms of recursive reasoning, where the recursive reasoning steps are not directly tied to corresponding reduction steps.
- But soundness will still hinge on some reduction steps taking place.
- Thus to ensure soundness, we will need some way to express, in the logic, that a property is only supposed to hold *later*, after a reduction step has taken place.
- This is what the later modality $\triangleright$ achieves: intuitively, $\triangleright P$ holds if P holds after a reduction step has been taken.

Plan for today

Today

- Rules for reasoning about $\triangleright$, including strengthening of earlier Hoare rules.
- Example
  - Specification and proof of a fixed point combinator.
  - Proof relies on $\triangleright$.
  - The example is perhaps somewhat contrived — chosen to illustrate expressiveness without being too long.

Later on

- the rules we describe today will be important later on, especially when reasoning about invariants.

Löb Rule

- Typing for $\triangleright$:

$$\frac{\Gamma \vdash P : \mathsf{Prop}}{\Gamma \vdash \triangleright P : \mathsf{Prop}}$$

- Löb Rule:

$$\frac{\begin{array}{l}\textsc{Löb}\\ S \wedge \triangleright P \vdash P\end{array}}{S \vdash P}$$

- Akin to a coinduction proof rule: to show P, it suffices to show P *under the assumption that* P *holds later*.

Aside: semantics of propositions

- As suggested by the above, the meaning of Iris proposition is not just a set of resources.
- In more detail, an Iris proposition P is[1] a set of pairs (k, r), with k a natural number and r a resource.
- Think of k as a step-index, a natural number which expresses for how many reduction steps we know that r is in P.
- If $(k, r) \in P$ and $m \leq k$, then also $(m, r) \in P$.
- The step-indeces are used to interpret $\triangleright$:

$$\triangleright P = \{(m+1, r) \mid (m, r) \in P\} \cup \{(0, r) \mid r \in \mathcal{R}\}$$

- "later" means that the index number is smaller (there are fewer reduction steps left, after we have taken some reduction steps).
- The Löb Rule is proved sound by induction on these step-indeces.

[1] Not really, but closer to being. . .

Laws for Later Modality

$$\textsc{Later-Mono}\quad \frac{Q \vdash P}{\triangleright Q \vdash \triangleright P}$$

$$\textsc{Later-weak}\quad \frac{Q \vdash P}{Q \vdash \triangleright P}$$

$$\textsc{Löb}\quad \frac{Q \land \triangleright P \vdash P}{Q \vdash P}$$

Laws for Later Modality

$$\textsc{Later-conj}\quad \dfrac{R \vdash \triangleright(P \land Q)}{R \vdash \triangleright P \land \triangleright Q}$$

$$\textsc{Later-disj}\quad \dfrac{R \vdash \triangleright(P \lor Q)}{R \vdash \triangleright P \lor \triangleright Q}$$

$$\textsc{Later-all}\quad \dfrac{Q \vdash \triangleright \forall x.\, P}{Q \vdash \forall x.\, \triangleright P}$$

$$\textsc{Later-sep}\quad \dfrac{R \vdash \triangleright P * \triangleright Q}{R \vdash \triangleright(P * Q)}$$

$$\triangleright\text{-}\exists\quad \dfrac{\tau \text{ is inhabited} \qquad Q \vdash \triangleright \exists x : \tau.\, P}{Q \vdash \exists x : \tau.\, \triangleright P}$$

$$\triangleright\text{-}\exists\quad \dfrac{Q \vdash \exists x.\, \triangleright P}{Q \vdash \triangleright \exists x.\, P}$$

Stronger rules for Hoare triples

$$\textsc{Ht-beta}\quad \frac{S \vdash \{P\}\, e\,[v/x]\ \{u.Q\}}{S \vdash \{\triangleright P\}\,(\lambda x.e)v\,\{u.Q\}}$$

$$\textsc{Ht-Rec}\quad \frac{Q \vdash \{P\}\, e\,[(\mathsf{rec}\, f(x) = e)/f, v/x]\ \{\Phi\}}{Q \vdash \{\triangleright P\}\,(\mathsf{rec}\, f(x) = e)v\,\{\Phi\}}$$

$$\textsc{Ht-load}\quad \frac{}{S \vdash \{\triangleright \ell \hookrightarrow u\}\ !\,\ell\,\{v.v = u \wedge \ell \hookrightarrow u\}}$$

$$\textsc{Ht-store}\quad \frac{}{S \vdash \{\triangleright \exists u.\, \ell \hookrightarrow u\}\,\ell \leftarrow w\,\{v.v = () \wedge \ell \hookrightarrow w\}}$$

$$\textsc{Ht-Match}\quad \frac{S \vdash \{P\}\, e_i\,[u/x_i]\ \{v.Q\}}{S \vdash \{\triangleright P\}\,\mathsf{match}\ \mathsf{inj}_i\, u\ \mathsf{with}\ \mathsf{inj}_1\, x_1 \Rightarrow e_1 \mid \mathsf{inj}_2\, x_2 \Rightarrow e_2\ \mathsf{end}\,\{v.Q\}}$$

Remark on soundness

Why are the rules HT-LOAD and HT-STORE sound ?

- Difficult to explain intuitively.
- Relies on $\hookrightarrow$ being a timeless predicate together with the definition of Hoare triples (the fact that weakest precondition is "closed wrt. the fancy update modality").

Stronger derived Hoare triples

$$\textsc{Ht-let}\quad \frac{S \vdash \{P\}\, e_1\, \{x.\, \triangleright Q\} \qquad S \vdash \forall v.\, \{Q[v/x]\}\, e_2\, [v/x]\ \{u.R\}}{S \vdash \{P\}\, \textsf{let}\, x = e_1\, \textsf{in}\, e_2\, \{u.R\}}$$

$$\textsc{Ht-let-det}\quad \frac{S \vdash \{P\}\, e_1\, \{x.\, \triangleright(x = v) \wedge \triangleright Q\} \qquad S \vdash \{Q[v/x]\}\, e_2\, [v/x]\ \{u.R\}}{S \vdash \{P\}\, \textsf{let}\, x = e_1\, \textsf{in}\, e_2\, \{u.R\}}$$

$$\textsc{Ht-seq}\quad \frac{S \vdash \{P\}\, e_1\, \{\_.\, \triangleright Q\} \qquad S \vdash \{R\}\, e_2\, \{u.R\}}{S \vdash \{P\}\, e_1 ;\, e_2\, \{u.R\}}\,.$$

$$\textsc{Ht-If}\quad \frac{S \vdash \{P * v = \textsf{true}\}\, e_2\, \{u.Q\} \qquad S \vdash \{P * v = \textsf{false}\}\, e_3\, \{u.Q\}}{S \vdash \{\triangleright P\}\, \textsf{if}\, v\, \textsf{then}\, e_2\, \textsf{else}\, e_3\, \{u.Q\}}$$

Guarded recursively defined predicates

- We extend terms of the logic with

$$t ::= \cdots \mid \mu x : \tau .\, t$$

  with the side-condition that the recursive occurrences must be *guarded*: in $\mu x.\, t$, the variable x can only appear under the later $\triangleright$ modality.
- Fixed-point property expressed by the following rule:

$$\dfrac{\textsc{Mu-fixed}}{Q \vdash \mu x : \tau .\, t =_{\tau} t\,[\mu x : \tau .\, t / x]}$$

Guarded recursively defined predicates

- Example: using a stream (infinite list) as model of linked list:

$$\mu \,\mathsf{isStream} : \mathit{Val} \to \mathrm{stream}\ \mathit{Val} \to \mathsf{Prop}.\ \lambda l : \mathit{Val}.\ \lambda xs : \mathrm{stream}\ \mathit{Val}.$$
$$(xs = [] \wedge l = \mathsf{inj}_1())\vee$$
$$(\exists x, xs'.\ xs = x : xs' \wedge \exists hd, l'.\ l = \mathsf{inj}_2(hd) * hd \hookrightarrow (x, l') * \triangleright(\mathsf{isStream}\ l' xs))$$

- Note that xs is a stream (infinite list). Therefore we cannot define the predicate by induction on xs.
- Above, the recursion variable occurs positively.
- In Iris, Hoare triples are defined in terms of weakest-preconditions, which are defined by means of a guarded recursive definition for a positive definition to give a partial correctness interpretation.
- One can also define mixed-variance recursive predicates.

Guarded recursively defined predicates

- Mixed-variance guarded recursive predicates are useful for
  - Interpreting recursive types in a typed programming language by Iris predicates / relations (see ipm-paper).
  - Defining models of untyped / unityped languages (*e.g.*, for object capabilities).
  - Specifying and reasoning about libraries that can call themself recursively, *e.g.*, an event loop library (see iCap-paper).
    - M.Sc. Project idea: Formalize event loop library in Iris in Coq, if ambitious, consider library for asynchronous IO.

Example: Fixed-point combinator Θ_F

- Given a value F, the call-by-value Turing fixed-point combinator Θ_F is:

$$\begin{aligned} \Omega_F &= \lambda r.F(\lambda x.rrx) \\ \Theta_F &= \Omega_F \Omega_F \end{aligned}$$

- For any values F and v,

$$\Theta_F v \rightsquigarrow F(\lambda x.\Theta_F x)v$$

- Thus, if $F = \lambda fx.e$ then one should think of Θ_F as $\mathsf{rec}\, f(x) = e$.

Proof Rule for Θ_F

- Now we wish to derive proof rule for Θ_F, similar to the recursion rule.

$$\frac{\textsc{Ht-Turing-fp} \\ \Gamma \mid S \wedge \forall v.\, \{P\}\, \Theta_F v\, \{u.Q\} \vdash \forall v.\, \{P\}\, F(\lambda x.\Theta_F x) v\, \{u.Q\}}{\Gamma \mid S \vdash \forall v.\, \{P\}\, \Theta_F v\, \{u.Q\}}$$

- We will use the Löb rule.
- We will also use that if P is persistent, then $\triangleright P$ is persistent, which means that it can be moved in and out of preconditions.

Proof

- We proceed by the Löb rule and hence we assume

$$\triangleright \forall v. \{P\} \Theta_F v \{u.Q\} \qquad (*)$$

- and we are to show

$$\forall v. \{P\} \Theta_F v \{u.Q\}.$$

- Let v be a value.
- By Later-weak and the rule of consequence SFTS

$$\{\triangleright P\} \Theta_F v \{u.Q\}.$$

Proof

- Since Hoare triples are persistent, we can move our assumption $(*)$ into the precondition, and thus SFTS:

$$\{\triangleright(\forall v.\, \{P\}\, \Theta_F v\, \{u.Q\} \wedge P)\}\, \Theta_F v\, \{u.Q\}$$

- By the bind rule and the stronger rule $\textsc{Ht-beta}$ introduced above SFTS

$$\{\forall v.\, \{P\}\, \Theta_F v\, \{u.Q\} \wedge P\}\, F(\lambda x.\Theta_F x) v\, \{u.Q\}$$

- We again use persistence and move the triple $\forall v.\, \{P\}\, \Theta_F v\, \{u.Q\}$ into the context and then SFTS

$$\{P\}\, F(\lambda x.\Theta_F x) v\, \{u.Q\}$$

- But this is exactly the premise of the rule $\textsc{Ht-Turing-fp}$, and thus the proof is concluded.