

# Extending LHC to Hyperliveness Properties

Report for internship – MPI-SWS Saarbrücken

ANTOINE GUILMIN-CRÉPON, ENS Paris-Saclay, France

## 1 STATE OF THE ART

Hoare logic was one of the first attempt at creating a formal framework for proving statements on programs. However, it proved itself limited for proving certain properties. In particular, dealing with non-determinism can become tricky, since basic Hoare logic can only deal with a given execution at a time. More generally, hyperproperties are an issue for basic Hoare logic, as they require to reason on multiple executions of the same program, or the execution of different programs.

Thus, multiple frameworks have been designed over the years to deal with those hyperproperties, mainly by extending Hoare logic to directly work on so-called hyperterms, i.e finite sets of terms, instead of regular terms.

LHC (for Logic for Hypertriple Composition) is one of those frameworks,

## 2 PRESENTATION OF LHC

This section is inspired by the paper presenting LHC. The logic defines, upon an arbitrary toy language, hyperterms, hyperstores, hyperreturn values and hyperproperties.

## 3 DESCRIPTION OF THE EXTENSION

The main point of the extension is to add an object to the language complementary to the  $\mathbf{wp}_v$  from the original model.

This object is defined as such :

$$\mathbf{wp}_{\exists} t \{Q\} ::= \lambda s. \exists s'v. \langle t, s \rangle \Downarrow \langle v, s' \rangle \wedge Q(v)$$

This object behaves in the same way as  $\mathbf{wp}_v$  for most of the rules previously described. Those rules are written down for completeness.

An interesting point

$$\begin{array}{c}
\text{WP}_\forall\text{-TRIV} \\
\vdash \mathbf{wp}_\forall t \{ \text{True} \}
\end{array}
\quad
\begin{array}{c}
\text{WP}_\forall\text{-CONS} \\
\frac{\forall v. Q(v) \vdash Q'(v)}{\mathbf{wp}_\forall t \{Q\} \vdash \mathbf{wp}_\forall t \{Q'\}}
\end{array}
\quad
\begin{array}{c}
\text{WP}_\forall\text{-ALL} \\
\forall x. \mathbf{wp}_\forall t \{Q(x)\} \dashv\vdash \mathbf{wp}_\forall t \{ \forall x. Q(x) \}
\end{array}$$

$$\begin{array}{c}
\text{WP}_\forall\text{-FRAME} \\
\frac{\Gamma \vdash \mathbf{wp}_\forall t \{Q\} \quad \text{pvar}(P) \cap \text{mods}(t) = \emptyset}{\Gamma, P \vdash \mathbf{wp}_\forall t \{ \lambda r. P \wedge Q(r) \}}
\end{array}
\quad
\begin{array}{c}
\text{WP}_\forall\text{-IMPL-R} \\
\frac{\text{pvar}(P) \cap \text{mods}(t) = \emptyset}{P \Rightarrow \mathbf{wp}_\forall t \{Q\} \dashv\vdash \mathbf{wp}_\forall t \{ \lambda r. P \Rightarrow Q(r) \}}
\end{array}$$

$$\begin{array}{c}
\text{WP}_\forall\text{-SUBST} \\
\frac{x \notin \text{mods}(t)}{x(i) = v \wedge \mathbf{wp}_\forall ([i: t[v/x]] \cdot t') \{Q\} \vdash \mathbf{wp}_\forall ([i: t] \cdot t') \{Q\}}
\end{array}
\quad
\begin{array}{c}
\text{WP}_\forall\text{-IDX} \\
\frac{\pi \text{ bijective}}{(\mathbf{wp}_\forall t \{Q\}) \langle \pi \rangle \vdash \mathbf{wp}_\forall t \langle \pi \rangle \{Q \langle \pi \rangle\}}
\end{array}$$

Fig. 1. Base rules for  $\mathbf{wp}_\forall$  from LHC

$$\begin{array}{c}
\text{WP}_\forall\text{-VAR} \\
\vdash \mathbf{wp}_\forall [i: x] \{ \lambda r. r(i) = x(i) \}
\end{array}
\quad
\begin{array}{c}
\text{WP}_\forall\text{-VAL} \\
\vdash \mathbf{wp}_\forall [i: v] \{ \lambda r. r(i) = v \}
\end{array}$$

$$\begin{array}{c}
\text{WP}_\forall\text{-SKIP} \\
\mathbf{wp}_\forall (t \cdot [i: \text{skip}]) \{Q\} \dashv\vdash \mathbf{wp}_\forall t \{Q\}
\end{array}
\quad
\begin{array}{c}
\text{WP}_\forall\text{-PRIM}_\oplus \\
\vdash \mathbf{wp}_\forall [i: v_1 \oplus v_2] \{ \lambda r. r(i) = (v_1 \llbracket \oplus \rrbracket v_2) \}
\end{array}$$

$$\begin{array}{c}
\text{WP}_\forall\text{-SEQ}_I \\
\mathbf{wp}_\forall [i: t_i \mid i \in I] \{ \mathbf{wp}_\forall [i: t'_i \mid i \in I] \{Q\} \} \dashv\vdash \mathbf{wp}_\forall [i: (t_i; t'_i) \mid i \in I] \{Q\}
\end{array}$$

$$\begin{array}{c}
\text{WP}_\forall\text{-ASSIGN}_I \\
\frac{\forall i \in I. (x_i, i) \notin \text{pvar}(Q)}{\mathbf{wp}_\forall [i: e_i \mid i \in I] \{Q\} \vdash \mathbf{wp}_\forall [i: x_i := e_i \mid i \in I] \{ \lambda r. Q(r) \wedge \bigwedge_{i \in I} r(i) = x_i(i) \}}
\end{array}$$

$$\begin{array}{c}
\text{WP}_\forall\text{-IF}_I \\
\mathbf{wp}_\forall [i: g_i \mid i \in I] \left\{ \lambda b. \mathbf{wp}_\forall \left( \frac{[i: t_i \mid i \in I, b(i) \neq 0] \cdot [i: t'_i \mid i \in I, b(i) = 0]}{[i: t_i \mid i \in I, b(i) = 0]} \right) \{Q\} \right\} \dashv\vdash \mathbf{wp}_\forall [i: \text{if } g_i \text{ then } t_i \text{ else } t'_i \mid i \in I] \{Q\}
\end{array}$$

$$\begin{array}{c}
\text{WP}_\forall\text{-WHILE}_I \\
\frac{P \vdash \mathbf{wp}_\forall [i: g_i \mid i \in I] \{ \lambda b. (b =_I 0 \wedge R) \vee (b \neq_I 0 \wedge \mathbf{wp}_\forall [i: t_i \mid i \in I] \{P\}) \}}{P \vdash \mathbf{wp}_\forall [i: \text{while } g_i \text{ do } t_i \mid i \in I] \{R\}}
\end{array}$$

Fig. 2. Lockstep rules for  $\mathbf{wp}_\forall$  from LHC

$$\begin{array}{c}
 \text{WP}_\forall\text{-NEST} \\
 \mathbf{wp}_\forall t_1 \{\lambda v. \mathbf{wp}_\forall t_2 \{\lambda w. Q(v \cdot w)\}\} \dashv\vdash \mathbf{wp}_\forall (t_1 \cdot t_2) \{Q\} \\
 \\
 \text{WP}_\forall\text{-CONJ} \\
 \frac{\text{idx}(Q_1) \cap \text{supp}(t_2) \subseteq \text{supp}(t_1) \quad \text{idx}(Q_2) \cap \text{supp}(t_1) \subseteq \text{supp}(t_2)}{\mathbf{wp}_\forall t_1 \{Q_1\} \wedge \mathbf{wp}_\forall t_2 \{Q_2\} \vdash \mathbf{wp}_\forall (t_1 + t_2) \{Q_1 \wedge Q_2\}} \\
 \\
 \text{WP}_\forall\text{-PROJ} \\
 \frac{}{\Pi_I. (\text{proj}(t_2) \Rightarrow \text{proj}(t_1) \wedge \mathbf{wp}_\forall (t_1 \cdot t_2) \{Q\}) \vdash \mathbf{wp}_\forall t_2 \{\hat{\Pi}_I. Q\}} \quad I = \text{supp}(t_1)
 \end{array}$$

Fig. 3. Hyper-structure laws from LHC

$$\begin{array}{c}
 \text{WP}_\forall\text{-IDX-PASS} \qquad \qquad \qquad \text{WP}_\forall\text{-IDX-SWAP} \\
 \frac{i, j \notin \text{supp}(t)}{(\mathbf{wp}_\forall t \{Q\}) \langle i/j \rangle \vdash \mathbf{wp}_\forall t \{Q \langle i/j \rangle\}} \qquad \frac{i \notin \text{idx}(Q)}{(\mathbf{wp}_\forall ([j:t] \cdot t') \{Q\}) \langle i/j \rangle \vdash \mathbf{wp}_\forall ([i:t] \cdot t') \{Q \langle i/j \rangle\}} \\
 \\
 \text{WP}_\forall\text{-IDX-MERGE} \\
 (\mathbf{wp}_\forall ([i:t, j:t] \cdot t') \{Q\}) \langle i/j \rangle \vdash \mathbf{wp}_\forall ([i:t] \cdot t') \{Q \langle i/j \rangle\} \\
 \\
 \text{WP}_\forall\text{-IDX-POST} \\
 \frac{\Gamma \vdash \mathbf{wp}_\forall t \{Q\} \quad j \notin \text{supp}(t) \cup \text{idx}(\Gamma)}{\Gamma \vdash \mathbf{wp}_\forall t \{Q \langle i/j \rangle\}}
 \end{array}$$

Fig. 4. Reindexing rules from LHC

$$\begin{array}{c}
 \text{WP-TRIV} \qquad \qquad \qquad \text{WP}_\forall\text{-WP}_\exists\text{-SWAP} \\
 \vdash \mathbf{wp}_\forall t \{\mathbf{wp}_\exists t \{\text{True}\}\} \qquad \frac{\forall i. \mathbf{wp}_\forall [1 : t_2(i)] \{\mathbf{wp}_\exists [2 : t_1(i)] \{Q\}\}}{\mathbf{wp}_\exists t_2 \{\mathbf{wp}_\forall t_1 \{Q\}\} \vdash \mathbf{wp}_\forall t_1 \{\mathbf{wp}_\exists t_2 \{Q\}\}}
 \end{array}$$

 Fig. 5. New  $\mathbf{wp}_\exists$  rules

## SOUNDNESS PROOFS

- WP-TRIV:

$$\begin{aligned}
 \vdash \mathbf{wp}_\forall t \{\mathbf{wp}_\exists t \{\text{True}\}\} &:= \forall s \, v_1 \, s'_1. \exists v_2 \, s'_2. \langle t, s \rangle \Downarrow \langle v_1, s'_1 \rangle \Rightarrow \langle t, s \rangle \Downarrow \langle v_2, s'_2 \rangle \\
 &\Leftarrow \forall s \, v \, s'. \langle t, s \rangle \Downarrow \langle v, s' \rangle \Rightarrow \langle t, s \rangle \Downarrow \langle v, s' \rangle
 \end{aligned}$$

- WP<sub>∃</sub>-CONS:

Suppose  $\forall v. Q(v) \vdash Q'(v)$ .

$\mathbf{wp}_\exists t \{Q\}$  means there exists a return value  $v$  and a output store  $s'$  s.t  $\langle t, s \rangle \Downarrow \langle v, s' \rangle$  and  $Q(v)(s')$ . By the first assumption, one gets  $Q'(v)(s')$ , thus  $\mathbf{wp}_\exists t \{Q'\}$ .

- WP<sub>∃</sub>-EXISTS:

Given that  $t$  does not depend on  $x$ ,

$$\begin{aligned}
 &\exists x. \mathbf{wp}_\exists t \{Q(x)\} \dashv\vdash \mathbf{wp}_\exists t \{\exists x. Q(x)\} \\
 := &\forall s. (\exists x \, v \, s'. \langle t, s \rangle \Downarrow \langle v, s' \rangle \wedge Q(x)(v)(s')) \Leftrightarrow \exists v \, s'. \langle t, s \rangle \Downarrow \langle v, s' \rangle \wedge (\exists x. Q(x)(v)(s'))
 \end{aligned}$$

- **WP<sub>∃</sub>-FRAME:**

Suppose  $\Gamma \vdash \mathbf{wp}_{\exists} t \{Q\}$  and  $\text{pvar}(P) \cap \text{mods}(t) = \emptyset$ . The first assumption implies a terminating trace for  $t$  verifying  $Q$ .

Given that no variable of  $P$  is modified by  $t$ , for every execution of the term,  $P$  will equally hold at both ends of the trace.

Thus  $\Gamma, P \vdash \mathbf{wp}_{\exists} t \{\lambda r. P \wedge Q(r)\}$ .

- **WP<sub>∃</sub>-IMPL-R:**

Similar to WP<sub>∃</sub>-FRAME.

- **WP<sub>∃</sub>-IDX:**

The proof relies on the fact that the existence of some terminating trace for a certain hyperterm is unchanged by a bijective reindexing of said hyperterm.

- **WP<sub>∃</sub>-NEST:** Suppose  $\mathbf{wp}_{\exists} (t_1 \cdot t_2) \{Q\}$ . It says that for all input states, there exists an output state satisfying  $Q$ .

Given that there are no side effects between the terms of an hyperterm, one can "divide" the trace as two separate traces in  $t_1$  and  $t_2$ . Thus, for all input state, there exists a state that coincides with the output state given by the assumption for the domain of  $t_1$ , and with the input state for the domain of  $t_2$  (it is here of prime importance that  $t_1$  and  $t_2$  are disjoint).

Therefore,  $\mathbf{wp}_{\exists} t_1 \{\mathbf{wp}_{\exists} t_2 \{Q\}\}$