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 $\mathbf{wp}_\forall$ from the original model.

This object is defined as such :

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

This object behaves in the same way as $\mathbf{wp}_\forall$ for most of the rules previously described. Those rules are written down for completeness. Most of the rules can be mirrored as is (just switching a $\mathbf{wp}_\forall$ for a $\mathbf{wp}_\exists$), but some needs more changes.

WP$_{exists}$-EXISTS mirrors WP$_\forall$-ALL, but uses existential quantifiers instead of universal ones. Without much surprise, $\mathbf{wp}_\exists$ works better with existentials. WP$_\exists$-WHILE$_I$ was modified to take into account the necessity of projectability for $\mathbf{wp}_\exists$ statements.

An interesting point is that predicates that couldn't be defined in the original theory using the base langugage, and thus needed to be explicitly defined to be used, can now be defined using this new $\mathbf{wp}_\exists$ object.

$$\begin{array}{rcl} \mathrm{proj}(t) & := & \mathbf{wp}_\exists\, \boldsymbol{t}\, \{\mathsf{True}\} \\ t_1 \preceq t_2 & \dashv\vdash & \mathbf{wp}_\forall\, [1 : t_1]\, \{\mathbf{wp}_\exists\, [2 : t_2]\, \{v(1) = v(2) \wedge s(1) = s(2)\}\} \end{array}$$

Thus certain rules from LHC can now be rewritten by combining $\mathbf{wp}_\forall$ and $\mathbf{wp}_\exists$

Author's address: Antoine Guilmin-Crépon, antoine.guilmin@ens-paris-saclay.fr, ENS Paris-Saclay, Gif-Sur-Yvette, France.

$$\textsc{wp}_\forall\textsc{-triv}\quad \vdash \mathbf{wp}_\forall\, \boldsymbol{t}\, \{\mathsf{True}\}$$

$$\textsc{wp}_\forall\textsc{-cons}\quad \frac{\forall \boldsymbol{v}.\ Q(\boldsymbol{v}) \vdash Q'(\boldsymbol{v})}{\mathbf{wp}_\forall\, \boldsymbol{t}\, \{Q\} \vdash \mathbf{wp}_\forall\, \boldsymbol{t}\, \{Q'\}}$$

$$\textsc{wp}_\forall\textsc{-all}\quad \forall x.\, \mathbf{wp}_\forall\, \boldsymbol{t}\, \{Q(x)\} \dashv\vdash \mathbf{wp}_\forall\, \boldsymbol{t}\, \{\forall x.\, Q(x)\}$$

$$\textsc{wp}_\forall\textsc{-frame}\quad \frac{\Gamma \vdash \mathbf{wp}_\forall\, \boldsymbol{t}\, \{Q\} \qquad \mathrm{pvar}(P) \cap \mathrm{mods}(\boldsymbol{t}) = \emptyset}{\Gamma, P \vdash \mathbf{wp}_\forall\, \boldsymbol{t}\, \{\lambda \boldsymbol{r}.\, P \wedge Q(\boldsymbol{r})\}}$$

$$\textsc{wp}_\forall\textsc{-impl-r}\quad \frac{\mathrm{pvar}(P) \cap \mathrm{mods}(\boldsymbol{t}) = \emptyset}{P \Rightarrow \mathbf{wp}_\forall\, \boldsymbol{t}\, \{Q\} \dashv\vdash \mathbf{wp}_\forall\, \boldsymbol{t}\, \{\lambda \boldsymbol{r}.\, P \Rightarrow Q(\boldsymbol{r})\}}$$

$$\textsc{wp}_\forall\textsc{-subst}\quad \frac{\mathsf{x} \notin \mathrm{mods}(t)}{\mathsf{x}(i) = v \wedge \mathbf{wp}_\forall\, \left(\left[i{:}\, t[v/\mathsf{x}]\right] \cdot \boldsymbol{t}'\right) \{Q\} \vdash \mathbf{wp}_\forall\, \left([i{:}\, t] \cdot \boldsymbol{t}'\right) \{Q\}}$$

$$\textsc{wp}_\forall\textsc{-idx}\quad \frac{\pi \text{ bijective}}{(\mathbf{wp}_\forall\, \boldsymbol{t}\, \{Q\})(\!|\pi|\!) \vdash \mathbf{wp}_\forall\, \boldsymbol{t}(\!|\pi|\!)\, \{Q(\!|\pi|\!)\}}$$

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

$$\textsc{wp}_\forall\textsc{-var}\quad \vdash \mathbf{wp}_\forall\, [i{:}\, \mathsf{x}]\, \{\lambda \boldsymbol{r}.\, \boldsymbol{r}(i) = \mathsf{x}(i)\}$$

$$\textsc{wp}_\forall\textsc{-val}\quad \vdash \mathbf{wp}_\forall\, [i{:}\, v]\, \{\lambda \boldsymbol{r}.\, \boldsymbol{r}(i) = v\}$$

$$\textsc{wp}_\forall\textsc{-skip}\quad \mathbf{wp}_\forall\, (\boldsymbol{t} \cdot [i{:}\, \mathbf{skip}])\, \{Q\} \dashv\vdash \mathbf{wp}_\forall\, \boldsymbol{t}\, \{Q\}$$

$$\textsc{wp}_\forall\textsc{-prim}_\oplus\quad \vdash \mathbf{wp}_\forall\, [i{:}\, v_1 \oplus v_2]\, \{\lambda \boldsymbol{r}.\, \boldsymbol{r}(i) = (v_1\ [\![\oplus]\!]\ v_2)\}$$

$$\textsc{wp}_\forall\textsc{-seq}_I\quad \mathbf{wp}_\forall\, [i{:}\, t_i \mid i \in I]\, \left\{\mathbf{wp}_\forall\, [i{:}\, t_i' \mid i \in I]\, \{Q\}\right\} \dashv\vdash \mathbf{wp}_\forall\, [i{:}\, (t_i;\ t_i') \mid i \in I]\, \{Q\}$$

$$\textsc{wp}_\forall\textsc{-assign}_I\quad \frac{\forall i \in I.\, (\mathsf{x}_i, i) \notin \mathrm{pvar}(Q)}{\mathbf{wp}_\forall\, [i{:}\, e_i \mid i \in I]\, \{Q\} \vdash \mathbf{wp}_\forall\, [i{:}\, \mathsf{x}_i \coloneqq e_i \mid i \in I]\, \{\lambda \boldsymbol{r}.\, Q(\boldsymbol{r}) \wedge \textstyle\bigwedge_{i \in I} \boldsymbol{r}(i) = \mathsf{x}_i(i)\}}$$

$$\textsc{wp}_\forall\textsc{-if}_I\quad \mathbf{wp}_\forall\, [i{:}\, g_i \mid i \in I]\, \left\{\lambda \boldsymbol{b}.\, \mathbf{wp}_\forall \begin{pmatrix}[i{:}\, t_i \mid i \in I, \boldsymbol{b}(i) \neq 0]\cdot \\ [i{:}\, t_i' \mid i \in I, \boldsymbol{b}(i) = 0]\end{pmatrix} \{Q\}\right\} \dashv\vdash \mathbf{wp}_\forall\, [i{:}\, \mathbf{if}\ g_i\ \mathbf{then}\ t_i\ \mathbf{else}\ t_i' \mid i \in I]\, \{Q\}$$

$$\textsc{wp}_\forall\textsc{-while}_I\quad \frac{P \vdash \mathbf{wp}_\forall\, [i{:}\, g_i \mid i \in I]\, \left\{\lambda \boldsymbol{b}.\, (\boldsymbol{b} =_I 0 \wedge R) \vee (\boldsymbol{b} \neq_I 0 \wedge \mathbf{wp}_\forall\, [i{:}\, t_i \mid i \in I]\, \{P\})\right\}}{P \vdash \mathbf{wp}_\forall\, [i{:}\, \mathbf{while}\ g_i\ \mathbf{do}\ t_i \mid i \in I]\, \{R\}}$$

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

SOUNDNESS PROOFS

- WP-TRIV:

$$\begin{aligned} \vdash \mathbf{wp}_\forall\, \boldsymbol{t}\, \{\mathbf{wp}_\exists\, \boldsymbol{t}\, \{\mathsf{True}\}\} \quad &:= \quad \forall s\ v_1\ s_1'.\ \exists v_2\ s_2'.\ \langle t, s\rangle \Downarrow \langle v_1, s_1'\rangle \Rightarrow \langle t, s\rangle \Downarrow \langle v_2, s_2'\rangle \\ &\Leftarrow \quad \forall s\ v\ s'.\ \langle t, s\rangle \Downarrow \langle v, s'\rangle \Rightarrow \langle t, s\rangle \Downarrow \langle v, s'\rangle \end{aligned}$$

- WP$_\exists$-CONS:
  Suppose $\forall v.\, Q(v) \vdash Q'(v)$.

$$\textsc{wp}_\forall\textsc{-nest}\quad \mathbf{wp}_\forall\, \boldsymbol{t}_1\, \{\lambda v.\, \mathbf{wp}_\forall\, \boldsymbol{t}_2\, \{\lambda w.\, Q(v \cdot w)\}\} \dashv\vdash \mathbf{wp}_\forall\, (\boldsymbol{t}_1 \cdot \boldsymbol{t}_2)\, \{Q\}$$

$$\textsc{wp}_\forall\textsc{-conj}\quad \frac{\mathrm{idx}(Q_1) \cap \mathrm{supp}(\boldsymbol{t}_2) \subseteq \mathrm{supp}(\boldsymbol{t}_1) \qquad \mathrm{idx}(Q_2) \cap \mathrm{supp}(\boldsymbol{t}_1) \subseteq \mathrm{supp}(\boldsymbol{t}_2)}{\mathbf{wp}_\forall\, \boldsymbol{t}_1\, \{Q_1\} \wedge \mathbf{wp}_\forall\, \boldsymbol{t}_2\, \{Q_2\} \vdash \mathbf{wp}_\forall\, (\boldsymbol{t}_1 + \boldsymbol{t}_2)\, \{Q_1 \wedge Q_2\}}$$

$$\textsc{wp}_\forall\textsc{-proj}\quad \frac{}{\Pi_I.\, \big(\mathrm{proj}(\boldsymbol{t}_2) \Rightarrow \mathrm{proj}(\boldsymbol{t}_1) \wedge \mathbf{wp}_\forall\, (\boldsymbol{t}_1 \cdot \boldsymbol{t}_2)\, \{Q\}\big) \vdash \mathbf{wp}_\forall\, \boldsymbol{t}_2\, \{\hat{\Pi}_I.\, Q\}}\; I = \mathrm{supp}(\boldsymbol{t}_1)$$

Fig. 3. Hyper-structure laws from LHC

$$\textsc{wp}_\forall\textsc{-idx-pass}\quad \frac{i, j \notin \mathrm{supp}(\boldsymbol{t})}{(\mathbf{wp}_\forall\, \boldsymbol{t}\, \{Q\})(\!|i/j|\!) \vdash \mathbf{wp}_\forall\, \boldsymbol{t}\, \{Q(\!|i/j|\!)\}} \qquad \textsc{wp}_\forall\textsc{-idx-swap}\quad \frac{i \notin \mathrm{idx}(Q)}{\big(\mathbf{wp}_\forall\, ([j\colon t] \cdot \boldsymbol{t}')\, \{Q\}\big)(\!|i/j|\!) \vdash \mathbf{wp}_\forall\, ([i\colon t] \cdot \boldsymbol{t}')\, \{Q(\!|i/j|\!)\}}$$

$$\textsc{wp}_\forall\textsc{-idx-merge}\quad \big(\mathbf{wp}_\forall\, ([i\colon t, j\colon t] \cdot \boldsymbol{t}')\, \{Q\}\big)(\!|i/j|\!) \vdash \mathbf{wp}_\forall\, ([i\colon t] \cdot \boldsymbol{t}')\, \{Q(\!|i/j|\!)\}$$

$$\textsc{wp}_\forall\textsc{-idx-post}\quad \frac{\Gamma \vdash \mathbf{wp}_\forall\, \boldsymbol{t}\, \{Q\} \qquad j \notin \mathrm{supp}(\boldsymbol{t}) \cup \mathrm{idx}(\Gamma)}{\Gamma \vdash \mathbf{wp}_\forall\, \boldsymbol{t}\, \{Q(\!|i/j|\!)\}}$$

Fig. 4. Reindexing rules from LHC

$\mathbf{wp}_\exists\, \boldsymbol{t}\, \{Q\}$ means there exists an 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 $\mathbf{wp}_\exists\, \boldsymbol{t}\, \{Q'\}$.

- WP$_\exists$-EXISTS:
  Given that $\boldsymbol{t}$ does not depend on x,

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

- WP$_\exists$-FRAME:
  Suppose $\Gamma \vdash \mathbf{wp}_\exists\, \boldsymbol{t}\, \{Q\}$ and $\mathrm{pvar}(P) \cap \mathrm{mods}(t) = \emptyset$. The first assumption implies a terminating trace for $\boldsymbol{t}$ verifying Q.
  Given that no variable of P is modified by $\boldsymbol{t}$, for every execution of the term, P will equally hold at both ends of the trace.
  Thus $\Gamma, P \vdash \mathbf{wp}_\exists\, \boldsymbol{t}\, \{\lambda r.P \wedge Q(r)\}$.
- WP$_\exists$-IMPL-R:
  Similar to WP$_\exists$-FRAME.
- WP$_\exists$-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$_\exists$-NEST:

Suppose $\mathbf{wp}_\exists\,(\boldsymbol{t}_1 \cdot \boldsymbol{t}_2)\,\{Q\}$. 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 $\boldsymbol{t}_1$ and $\boldsymbol{t}_2$. Thus, forall input state, there exists a state that coincides with the output state given by the assumption for the domain of $\boldsymbol{t}_1$, and with the input state for the domain of $\boldsymbol{t}_2$ (it is here of prime importance that $\boldsymbol{t}_1$ and $\boldsymbol{t}_2$ are disjoint). Therefore, $\mathbf{wp}_\exists\,\boldsymbol{t}_1\,\{\mathbf{wp}_\exists\,\boldsymbol{t}_2\,\{Q\}\}$

- WP$_\exists$-CONJ:
  Suppose $\mathbf{wp}_\exists\,\boldsymbol{t}_1\,\{Q_1\} \wedge \mathbf{wp}_\exists\,\boldsymbol{t}_2\,\{Q_2\}$. It gives a trace for $\boldsymbol{t}_1$ satisfying Q_1, and a trace for $\boldsymbol{t}_2$ satisfying Q_2.
  To prove the result, one should construct a trace of $\boldsymbol{t}_1 + \boldsymbol{t}_2$ that satisfies $Q_1 \wedge Q_2$. To do so, combining the traces given by the assumptions works, apart from the intersections of the domains of the two hyperterms.

WP∃-CONS
$$\frac{\exists \boldsymbol{v}.\ Q(\boldsymbol{v}) \vdash Q'(\boldsymbol{v})}{\mathbf{wp}_\exists\, \boldsymbol{t}\, \{Q\} \vdash \mathbf{wp}_\exists\, \boldsymbol{t}\, \{Q'\}}$$

WP∃-EXISTS
$$\exists x.\, \mathbf{wp}_\exists\, \boldsymbol{t}\, \{Q(x)\} \dashv\vdash \mathbf{wp}_\exists\, \boldsymbol{t}\, \{\exists x.\, Q(x)\}$$

WP∃-FRAME
$$\frac{\Gamma \vdash \mathbf{wp}_\exists\, \boldsymbol{t}\, \{Q\} \qquad \mathrm{pvar}(P) \cap \mathrm{mods}(\boldsymbol{t}) = \emptyset}{\Gamma, P \vdash \mathbf{wp}_\exists\, \boldsymbol{t}\, \{\lambda \boldsymbol{r}.\, P \wedge Q(\boldsymbol{r})\}}$$

WP∃-IMPL-R
$$\frac{\mathrm{pvar}(P) \cap \mathrm{mods}(\boldsymbol{t}) = \emptyset}{P \Rightarrow \mathbf{wp}_\exists\, \boldsymbol{t}\, \{Q\} \dashv\vdash \mathbf{wp}_\exists\, \boldsymbol{t}\, \{\lambda \boldsymbol{r}.\, P \Rightarrow Q(\boldsymbol{r})\}}$$

WP∃-SUBST
$$\frac{\mathsf{x} \notin \mathrm{mods}(t)}{\mathsf{x}(i) = v \wedge \mathbf{wp}_\exists\, \left(\left[i{:}\, t[v/\mathsf{x}]\right] \cdot \boldsymbol{t}'\right) \{Q\} \vdash \mathbf{wp}_\exists\, \left([i{:}\, t] \cdot \boldsymbol{t}'\right) \{Q\}}$$

WP∃-IDX
$$\frac{\pi \text{ bijective}}{(\mathbf{wp}_\exists\, \boldsymbol{t}\, \{Q\})(\!|\pi|\!) \vdash \mathbf{wp}_\exists\, \boldsymbol{t}(\!|\pi|\!)\, \{Q(\!|\pi|\!)\}}$$

WP∃-SEQ$_I$
$$\mathbf{wp}_\exists\, [i{:}\, t_i \mid i \in I]\, \left\{\mathbf{wp}_\exists\, [i{:}\, t'_i \mid i \in I]\, \{Q\}\right\} \dashv\vdash \mathbf{wp}_\exists\, [i{:}\, (t_i;\ t'_i) \mid i \in I]\, \{Q\}$$

WP∃-ASSIGN$_I$
$$\frac{\exists i \in I.\, (\mathsf{x}_i, i) \notin \mathrm{pvar}(Q)}{\mathbf{wp}_\exists\, [i{:}\, e_i \mid i \in I]\, \{Q\} \vdash \mathbf{wp}_\exists\, [i{:}\, \mathsf{x}_i := e_i \mid i \in I]\, \{\lambda \boldsymbol{r}.\, Q(\boldsymbol{r}) \wedge \textstyle\bigwedge_{i \in I} \boldsymbol{r}(i) = \mathsf{x}_i(i)\}}$$

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

WP∃-WHILE$_I$
$$\frac{\forall \alpha.\, P(\alpha) \vdash \mathbf{wp}_\exists\, [i{:}\, g_i \mid i \in I]\, \left\{\lambda \boldsymbol{b}.\, (\boldsymbol{b} =_I 0 \wedge R) \vee (\boldsymbol{b} \neq_I 0 \wedge \mathbf{wp}_\exists\, [i{:}\, t_i \mid i \in I]\, \{\exists \alpha'.\, P(\alpha') \wedge \alpha' < \alpha\})\right\}}{P(\alpha_0) \vdash \mathbf{wp}_\exists\, [i{:}\, \mathbf{while}\ g_i\ \mathbf{do}\ t_i \mid i \in I]\, \{R\}}$$

WP∃-NEST
$$\mathbf{wp}_\exists\, \boldsymbol{t}_1\, \{\lambda \boldsymbol{v}.\, \mathbf{wp}_\exists\, \boldsymbol{t}_2\, \{\lambda \boldsymbol{w}.\, Q(\boldsymbol{v} \cdot \boldsymbol{w})\}\} \dashv\vdash \mathbf{wp}_\exists\, (\boldsymbol{t}_1 \cdot \boldsymbol{t}_2)\, \{Q\}$$

WP∃-CONJ
$$\frac{\mathrm{idx}(Q_1) \cap \mathrm{supp}(\boldsymbol{t}_2) \subseteq \mathrm{supp}(\boldsymbol{t}_1) \qquad \mathrm{idx}(Q_2) \cap \mathrm{supp}(\boldsymbol{t}_1) \subseteq \mathrm{supp}(\boldsymbol{t}_2)}{\mathbf{wp}_\exists\, \boldsymbol{t}_1\, \{Q_1\} \wedge \mathbf{wp}_\exists\, \boldsymbol{t}_2\, \{Q_2\} \vdash \mathbf{wp}_\exists\, (\boldsymbol{t}_1 + \boldsymbol{t}_2)\, \{Q_1 \wedge Q_2\}}$$

WP∃-IDX-PASS
$$\frac{i, j \notin \mathrm{supp}(\boldsymbol{t})}{(\mathbf{wp}_\exists\, \boldsymbol{t}\, \{Q\})(\!|i/j|\!) \vdash \mathbf{wp}_\exists\, \boldsymbol{t}\, \{Q(\!|i/j|\!)\}}$$

WP∃-IDX-SWAP
$$\frac{i \notin \mathrm{idx}(Q)}{\left(\mathbf{wp}_\exists\, \left([j{:}\, t] \cdot \boldsymbol{t}'\right) \{Q\}\right)(\!|i/j|\!) \vdash \mathbf{wp}_\exists\, \left([i{:}\, t] \cdot \boldsymbol{t}'\right) \{Q(\!|i/j|\!)\}}$$

WP∃-IDX-MERGE
$$\left(\mathbf{wp}_\exists\, \left([i{:}\, t, j{:}\, t] \cdot \boldsymbol{t}'\right) \{Q\}\right)(\!|i/j|\!) \vdash \mathbf{wp}_\exists\, \left([i{:}\, t] \cdot \boldsymbol{t}'\right) \{Q(\!|i/j|\!)\}$$

WP∃-IDX-POST
$$\frac{\Gamma \vdash \mathbf{wp}_\exists\, \boldsymbol{t}\, \{Q\} \qquad j \notin \mathrm{supp}(\boldsymbol{t}) \cup \mathrm{idx}(\Gamma)}{\Gamma \vdash \mathbf{wp}_\exists\, \boldsymbol{t}\, \{Q(\!|i/j|\!)\}}$$

WP-REFINE

$$\mathbf{wp}_\forall\,[1:t_1]\,\{\mathbf{wp}_\exists\,[2:t_2]\,\{v(1)=v(2)\wedge s(1)=s(2)\}\},\mathbf{wp}_\forall\,([i{:}\,t_2]\cdot\boldsymbol{t})\,\{Q\}\vdash\mathbf{wp}_\forall\,([i{:}\,t_1]\cdot\boldsymbol{t})\,\{Q\}$$

WP$_\forall$-PROJ

$$\frac{}{\Pi_I.\,\left(\mathbf{wp}_\forall\,\boldsymbol{t}_2\,\{\mathbf{wp}_\exists\,\boldsymbol{t}_1\,\{\mathsf{True}\}\}\wedge\mathbf{wp}_\forall\,(\boldsymbol{t}_1\cdot\boldsymbol{t}_2)\,\{Q\}\right)\vdash\mathbf{wp}_\forall\,\boldsymbol{t}_2\,\{\hat{\Pi}_I.\,Q\}}\;I=\mathrm{supp}(\boldsymbol{t}_1)$$

Fig. 6. Rules rewritten using $\mathbf{wp}_\exists$

WP-TRIV

$$\vdash\mathbf{wp}_\forall\,\boldsymbol{t}\,\{\mathbf{wp}_\exists\,\boldsymbol{t}\,\{\mathsf{True}\}\}$$

WP$_\forall$-WP$_\exists$-SWAP

$$\frac{\forall i.\,\mathbf{wp}_\forall\,[1:\boldsymbol{t}_2(i)]\,\{\mathbf{wp}_\exists\,[2:\boldsymbol{t}_1(i)]\,\{Q\}\}}{\mathbf{wp}_\exists\,\boldsymbol{t}_2\,\{\mathbf{wp}_\forall\,\boldsymbol{t}_1\,\{Q\}\}\vdash\mathbf{wp}_\forall\,\boldsymbol{t}_1\,\{\mathbf{wp}_\exists\,\boldsymbol{t}_2\,\{Q\}\}}$$

Fig. 7. Rules combining $\mathbf{wp}_\forall$ and $\mathbf{wp}_\exists$ rules

Temporary page!

LaTeX was unable to guess the total number of pages correctly. As there was some unprocessed data that should have been added to the final page this extra page has been added to receive it.

If you rerun the document (without altering it) this surplus page will go away, because LaTeX now knows how many pages to expect for this document.