PSL 1.1 in HOL

a deep embedding

Mike Gordon, Klaus Schneider, Thomas Tuerk
(with help from Cindy Eisner and Dana Fisman of IBM)

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

PSL is an industry standard property language (IEEE P1850)

- mainly developed by Cindy Eisner and Dana Fisman
- LTL based, but incorporates features from ITL (regular expressions) and CTL
  - $\{r_1\} \mapsto \{r_2\}$, $\{r_1 \;\&\&\; r_2\}$ (ITL), $X!f$, $[f_1 U f_2]$ (LTL), EXf, $E[f_1 \; U \; f_2]$ (CTL)
- supports infinite paths (for model checking) and finite paths (for simulation run checking)
- has both clocked and unclocked semantics
  - $\pi \models^c f$ — formula f holds for (finite or infinite) path π when clocked on c
- most constructs are defined from a small kernel
  - $b[=i] = \{\neg b[*]; b\}[*i]; \neg b[*]$, $r_1 \; \mathit{within} \; r_2 = \{[*];\; r_1;\; [*]\} \;\&\&\; \{r_2\}$
  - $[f_1 \; W \; f_2] = [f_1 \; U \; f_2] \vee Gf_1$, $\mathit{next\_event}(b)(f) = [\neg b \; W \; b \wedge f]$

Semantics in HOL's classical higher order logic is a straightforward deep embedding

PSL syntax $[f_1 \; U \; f_2]$	HOL representation F_UNTIL(f_1, f_2)
PSL semantics $\pi \models^c [f_1 \; U \; f_2] \iff$ there exist $k < \lvert v \rvert$ s.t. $\pi^k \models c$, $\pi^{k..} \models^c f_2$ and for every $j < k$ s.t. $\pi^j \models c$, $\pi^{j..} \models^c f_1$	HOL representation F_SEM v c (F_UNTIL(f1,f2)) = ∃k < LENGTH v. B_SEM (ELEM v k) c ∧ F_SEM (RESTN v k) c f2 ∧ ∀j ≤ k. B_SEM (ELEM (COMPLEMENT v) j) c ⟹ F_SEM (RESTN v j) c f1)

A verified translation of a subset of *PSL* to *LTL* and then to ω-automata was created

PSL	LTL	ω-automaton
$\pi \models^c [f_1 \; U \; f_2]$	$\pi \models$ $\big((\neg c \;\vee\; (\neg c \, U \, (c \;\wedge\; a))) \, U$ $(c \wedge (\neg c \, U \, (c \wedge b)))\big)$	$\pi \models \mathcal{A}_\exists\big(\{s_0, s_1, s_2\},\; s_2,$ $(s_0 \leftrightarrow (c \wedge a) \vee (\neg c \wedge X s_0)) \wedge$ $(s_1 \leftrightarrow (c \wedge b) \vee (\neg c \wedge X s_1)) \wedge$ $(s_2 \leftrightarrow (c \wedge s_1) \vee ((\neg c \vee s_0) \wedge X s_2)),$ $FG(s_0 \rightarrow (c \wedge a)) \wedge FG(s_1 \rightarrow (c \wedge b)) \wedge$ $FG(s_2 \rightarrow (c \wedge s_1))\big)$

Applications

- translation of *PSL* to ω-automata allows the translation of *PSL* model checking problems to *Fair-CTL* model checking problems
  - an interface to the model checker *SMV* allows *PSL* model checking
  - an interface to Amjad's *HOLCheck* tool allows verified PSL model checking
- tool created to check, if IBM's model checker *RuleBase* correctly handles *PSL*