THE UNIVERSITY *of* EDINBURGH
informatics

AI4FM

The Use of Rippling to Automate Event-B Invariant Preservation Proofs

Yuhui Lin, Alan Bundy& Gudmund Grov

School of Informatics
University of Edinburgh

Background

- Proof automation is a bottleneck for industrial use of formal method
  - large number of proofs (e.g 43K/28K)
  - requires user expertise
  - High-proportion (e.g. 59%) are invariant preservation, which follows a particular pattern $f(x) \vdash f(\boxed{g(\boxed{x})})$
- Here, we show how an automatic proof technique called rippling is applicable to these POs
- Outline a novel approach combining rippling with scheme-based theory exploration to automate required lemma discovery

Event-B INV proofs

Consider the invariant & the event

$T = dom(R; f)$

any x, y
when $x \in T$
then $R := R \cup \{(x \mapsto y)\}$

$x \mapsto y$	$a\ pair$
$\mathrm{dom}(r)$	$\{x.\exists y.(x \mapsto y) \in r\}$
$p\,;q$	$\{(x \mapsto y).\exists z.(x \mapsto z) \in p \wedge (z \mapsto y) \in q\}$

AI4FM

Event-B INV proofs

Proof :

$x \in T$

$T = \mathrm{dom}(R \,;\, f)$

$\vdash$

$T = \mathrm{dom}((R \cup \{(x \mapsto y)\}) \,;\, f)$ $\quad$ $(A \cup B);C = A;C \cup B;C$

$T = \mathrm{dom}((R \,;\, f) \cup (\{(x \mapsto y)\} \,;\, f))$ $\quad$ $dom(A \cup B) = dom(A) \cup dom(B)$

$T = dom(R; f) \cup dom(\{(x \mapsto y)\}; f)$ Apply Assumption

$T = T \cup \mathrm{dom}(\{(x \mapsto y)\} \,;\, f)$

Rippling

- *Rippling is* developed for step cases of inductive proofs
  - guides searching by moving the goal towards the *induction hypothesis* (e.g. invariants in Event-B)
  - *skeleton* (embedding of the invariant) is intact
  - meta-level annotations called wave fronts only moves in certain desirable directions

$$f(x) \vdash f(\boxed{g(\boxed{x})}) \xrightarrow{f(g(x)) = h(f(x))} f(x) \vdash \boxed{h(\boxed{f(x)})}$$

- Allows rewrite rules in both directions with termination guaranteed (e.g. associative and distributive rules)
- Have strong expectation of the following proofs steps

Event-B invariant proofs by rippling

$$x \in T$$
$$T = \mathrm{dom}(R\,;\,f)$$
$$\vdash$$

$T = \mathrm{dom}(\ (R \cup \{(x \mapsto y)\})\ \ ;\,f)$ $\quad$ $(A \cup B);C = A;C \cup B;C$

$T = \mathrm{dom}(\ (R\,;\,f) \cup (\{(x \mapsto y)\}\,;\,f)\ )$ $\quad$ $dom(A \cup B) = dom(A) \cup dom(B)$

$T = \ \mathrm{dom}(R\,;\,f) \cup \mathrm{dom}(\{(x \mapsto y)\}\,;\,f)$ $\quad$ Apply Assumption

$$T = T \cup \mathrm{dom}(\{(x \mapsto y)\}\,;\,f)$$

Lemma discovery in rippling

- Proof can be blocked by lacks of lemmas
- Suppose our proof is blocked at:

$$T = \mathrm{dom}(\ R \cup S \ \ ; f)$$

- We can then follow a 4 step process which discovers the missing lemma

$$\boxed{(A \cup B);C = A;C \cup B;C}$$

Lemma discovery steps

1. **Generate the left hand side:** pick terms of blocked goals which are expected to change in the next rewriting step, e.g.

$$R \cup S \quad ; f$$

2. **Conjecture right hand side:** do it with strong expectation and patterns of scheme (e.g. distributive pattern)

$$?F_1((R ; f), (?F_2\ S\ f))$$

Where *?Fn* is a 2nd order meta-variables

- since skeleton must be preserved
- wave-front must move outwards.,

Lemma discovery steps

3. **Instantiate scheme**: then feed the scheme, i.e. $R \cup S \; ; f = \; ?F_1((R \, ; f), (?F_2 \; S \; f))$, together with a set of terms for instantiation to IsaScheme which
   - is a tool which discovers conjectures
   - with counter-examples checks
   - with proof attempts

4. **Prove conjecture**: one of the "sensible" instantiations is $(R \cup S) \, ; f = (R \, ; f) \cup (S \, ; f)$. But in more complex cases the process recurses or the user must provide a proof

THE UNIVERSITY of EDINBURGH
informatics

AI4FM

Evaluation & Further work

Num of POs	9
Rodin only or only Isabelle tactics	0
Rippling + Isabelle tactics	1
Rippling + IsaScheme + Isabelle tactics (R + I + I)	2
R + I + I with some interaction still required	6

- Our further works are:
  - dynamic scheme generation
  - proper set of terms for meta-variables to instantiate
  - conditional lemmas- generic
  - piecewise fertilisation