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

ATX 2012

Outline

In this talk we

- discuss the needs to apply a meta-level reasoning technique to *Event-B Invariant (INV) proofs*
- show a proof technique called rippling is applicable
- outline a novel approach combining rippling with theory formation to automate lemma discovery

AI4FM

Event-B INV proofs

- Proof automation is a bottleneck for industrial use of formal method
  - large number of proofs (e.g 43,610 Roissy Airport Shuttle project; 27, 800 Paris Metro line 14 project)
  - requires expert experience

- We have observed that
  - the majority of proofs requiring automation are invariant proofs (e.g. 59% in one case study)
  - lacks of guides of high-level reasoning guide
  - INV proofs typically follow a pattern where one of the assumptions is embedded in the goal, i.e. $f(x) \vdash f(\ g(x)\ )$

AI4FM

Event-B INV proofs

Consider the invariant & the event

$T = dom(R; f)$

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

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

AI4FM

Event-B INV proofs

Proof :

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

$$(A \cup B);C = A;C \cup B;C$$

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

$$dom(A \cup B) = dom(A) \cup dom(B)$$

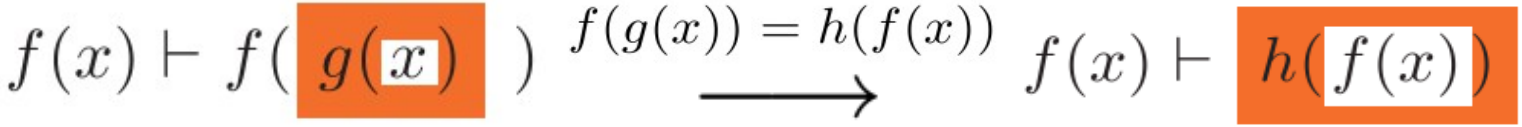
$$T = dom(R;f) \cup dom(\{(x \mapsto y)\};f)$$

$$T = dom(R;f) \text{ followed by cases on } y \in dom(f)$$

$$y \in dom(f) \vdash T = T \cup \{x\}$$
$$y \notin dom(f) \vdash T = T \cup \{\}$$

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



- 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

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

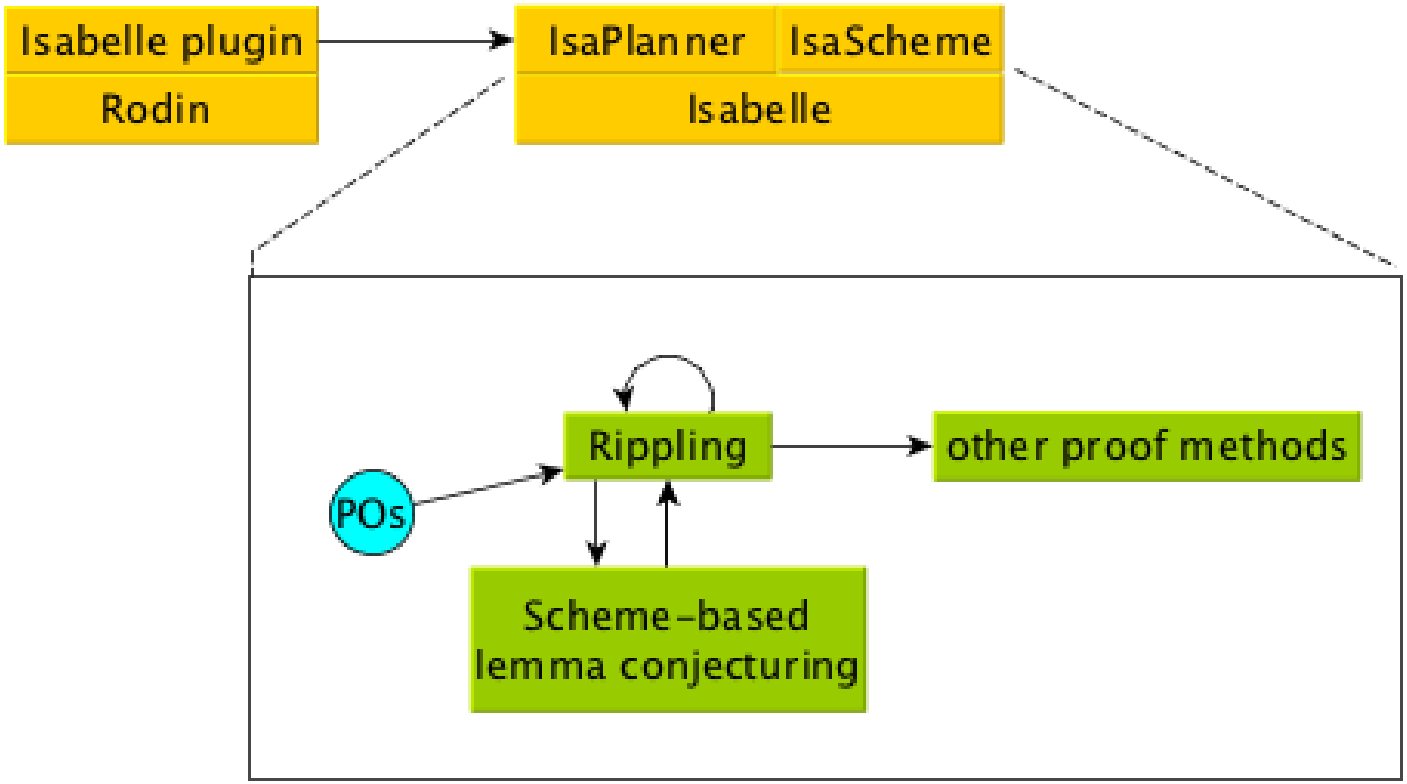
$$T = \mathrm{dom}(\,(R \cup \{(x \mapsto y)\})\;;f) \qquad (A \cup B);C = A;C \cup B;C$$

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

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

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

Approach


Isabelle plugin
Rodin
IsaPlanner
IsaScheme
Isabelle
Rippling
other proof methods
POs
Scheme-based
lemma conjecturing

Lemma discovery in rippling

- Suppose a simplified version of 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

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

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

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.

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

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. $R \cup S \;;f = ?F_1\ (R\,;f)\ (?F_2\ S\ f)$ ether 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

Evaluation

- The work is being implemented
  - in the Isabelle-based IsaPlanner tool
  - based upon Matthias Schmalz' Rodin/Isabelle integration

Num of POs	9
Proved automatically without rippling	0
Proved automatically with rippling	1
Proved automatically with rippling + IsaScheme	2
Need lemma conjecture	6

Conclusion and further work

- We have shown
  - that rippling is applicable to Event-B invariant POs

    a new technique to help discover missing lemmas
- We have implemented the automation of the lemma discovery process. Our further works are:
  - conditional lemmas
  - dynamic scheme generation
  - proper set of terms for meta-variables to instantiate