



THE UNIVERSITY of EDINBURGH
informatics



Productive use of failure in formal methods

Yuhui Lin

CIAO/Automatheo 2011

www.inf.ed.ac.uk



THE UNIVERSITY of EDINBURGH
informatics



Contents

- Background
- Proposed work
- Summary



Background – Formal methods

- Formal methods: mathematical approach
- How to apply FMs:
 - bottom-up: for existing system; debugging
 - top-down: towards codes; posit and prove; cost-effective
- Some techniques can be used in top-down e.g. Z, B, Event-B, VDM and so on ...



Background – Proof obligations

- Justifications generate proofs obligations (POs)
- POs are putative lemmas require proofs in theorem prover, e.g. Z/Eves for Z; Rodin for B/Event B
- 5 - 10% POs require human interaction
 - e.g. the Paris Metro line 14; 2,250 / 27,800 POs
- How to discharge them
 - by the modelling strategy
 - by the proof strategy
- These POs are expensive to discharge
- Proof automation becomes a bottleneck in industry.



Background – An example of POs

Telephone exchange in Z - states

Exchange

Free, Unavailable, Callers: P Subs -- Sets of Subscribe numbers

call: Subs \rightarrow SubRec -- Calls' num maps to (call's status, callee's num)

connected: Subs \rightsquigarrow Subs -- Call num maps to callee num

\langle Free, Unavailable, dom call \cup ran connected \rangle partition Subs

Callers = dom ((call \S st) \triangleright Connected) -- a set of status

connected = Callers \triangleleft (call \S num) e.g. ringing, speech



Background - An example of POs

Telephone exchange in Z – an operation

LiftFree

Δ *Exchange*

s?: *Subs*

s? \in *Free*

Free' = *Free* \setminus {*s?*}

Unavailable' = *Unavailable*

call' = *call* \cup {(*s?* \mapsto (*seize*, \diamond))}



Background - An example of POs

Telephone exchange in Z - a PO

$Callers = \text{dom} ((call \text{ ; } st) \triangleright Connected)$

$s? \in Free \quad call' = call \cup \{(s? \mapsto (seize, \langle \rangle))\}$

To prove:

$s? \in Free \Rightarrow Callers = \text{dom} (call' \text{ ; } st) \triangleright Connected$

Which is:

$s? \in Free \Rightarrow Callers = \text{dom} ((call \cup \{(s? \mapsto (seize, \langle \rangle))\}) \text{ ; } st) \triangleright Connected$



Proposed work - Outline

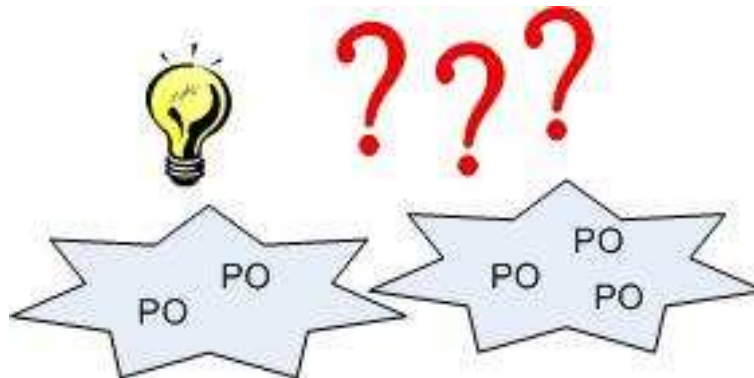
- Our hypothesis is
 - *It is possible to classify FM PO into families.*
 - *In each family there exist patterns of proof strategies.*
 - *The strategy can be used to guide the proof search of the blocked proofs from the same family.*
- Key words: *family, failure.*



Proposed work – key word family

- Definition for *family* from Merriam-Webster
a group of things related by common characteristics

Can POs in FMs be classified into families?





Proposed work - Families of POs

- Definition for *family* from Merriam-Webster

a group of things related by common characteristics

- Nature of POs in FMs:

- the number of POs is huge.

e.g. the Paris Metro line 14; 2,250 / 27,800 POs

- POs tends to exhibit a “similarity”

e.g. an industrial case-study using Event-B, 100 / 300 POs, can be handled by 5 strategies.



Proposed work – An example for *family*

- Proofs for PO1

PO1:

$$s? \in Free \Rightarrow Callers = \text{dom} ((call \cup \{(s? \mapsto (seize, \langle \rangle))\}) \S st) \triangleright Connected$$

From: $s? \in Free \Rightarrow s? \notin \text{dom} (call)$

Hence: $Callers = \text{dom} ((call \S st) \cup (\{(s? \mapsto (seize, \langle \rangle))\} \S st)) \triangleright Connected$

Hence:

$$Callers = \text{dom} (((call \S st) \triangleright Connected) \cup (\{(s? \mapsto (seize, \langle \rangle))\} \S st) \triangleright Connected)$$

Hence: $Callers = \text{dom} (((call \S st) \triangleright Connected) \cup \emptyset)$

Equals invariant: $Callers = \text{dom} ((call \S st) \triangleright Connected)$



Proposed work – An example for *family*

- Proofs for PO2

Inv: $connected = Callers \triangleleft (call \wp num)$

PO2: $s? \in Free \Rightarrow connected = Callers \triangleleft (call' \wp num)$

Which is: $connected = Callers \triangleleft ((call \cup \{(s? \mapsto (seize, \langle \rangle))\}) \wp num)$

Equals: $connected = Callers \triangleleft ((call \wp num) \cup (\{(s? \mapsto (seize, \langle \rangle))\} \wp num))$

Equals: $connected = (Callers \triangleleft (call \wp num)) \cup (Callers \triangleleft (\{(s? \mapsto (seize, \langle \rangle))\} \wp num))$

From: $(Callers \triangleleft (\{(s? \mapsto (seize, \langle \rangle))\} \wp num)) \Rightarrow Callers \triangleleft \{s? \mapsto \langle \rangle\} \Rightarrow \emptyset$

Done !



Proposed work – An example for *family*

- Similarities of proofs between PO1 and PO2

PO1: $s? \in Free \Rightarrow Callers = \text{dom}(call' \text{ \& } st) \triangleright Connected$

PO2: $s? \in Free \Rightarrow connected = Callers \triangleleft (call' \text{ \& } num)$

These two POs can be discharged in one strategy, which is to “push” changes to the outer level until it can be showed that the changes are empty.



Proposed work - key word *failure*



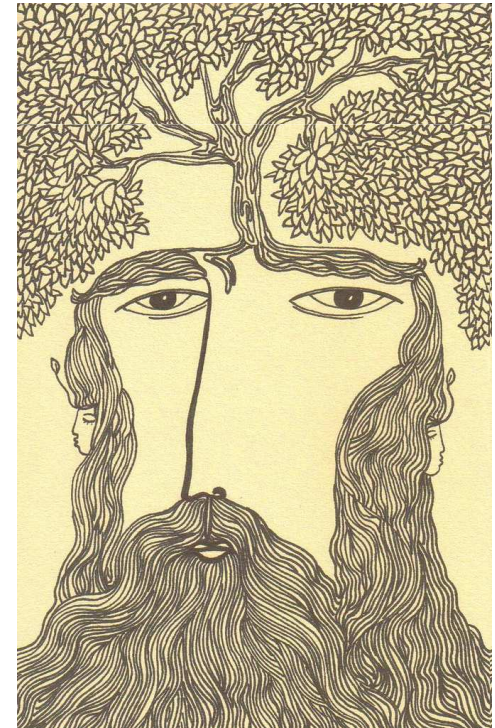


Proposed work - key word *failure*

Some encouraging sayings:

- Nothing fails like success because we don't learn from it.
We learn only from failure. ~Kenneth Boudling
- There is much to be said for failure. It is more interesting than success. ~Max Beerbohm
- Failure is only the opportunity to begin again more intelligently. ~Henry Ford

Well...but
they are not for mathematics.
We are dealing with proofs !!!





Proposed work – Use of failures in proofs

- Rippling: is a rewriting technique for the automation of mathematical reasoning which works when the goal is embedded in one of the givens.
 - not exhaustive rewriting, skeleton preserving and measure decreasing
 - rewriting in either direction
 - guarantee termination
- Proof critics : productive use of failure in rippling
 - from strong expectation of rippling, it can suggest a missing lemma, generalisation or a different induction rule.



Proposed work – proposed use of failure

- Gain value of failures by
 - analysing why they fail, how they fail.
 - extract meta-level strategies from patches to guide search of proof in similar case (family)
- Moreover, if a strategy fails in the families, critics can be developed to handle it as an “exception”





Proposed work – Approach

- Empirical process:
 - Collect POs and patches from existing industrial examples, e.g. Tokeneer
- Iterative development:
 - Strategies between families are supposed to be relatively independent
 - Implement strategies as methods (most likely in IsaPlanner)
 - Develop patches to failed strategies within families of POs as critics



Summary

- Undischarged POs are a bottleneck of development.
- We proposed to extract proof strategies for families of POs by analysing failures and patches.
- POs in FMs are suitable to be classified into families.
- Learning from failures can help us improve power to discharge POs automatically
- Our development is empirical and iterative.



THE UNIVERSITY of EDINBURGH
informatics



Thank you !!!

Q&A

- This work is supported by EPSRC grant EP/H024204/1
- AI4FM: using AI to aid automation of proof search in Formal Methods.
- For more details see <http://www.ai4fm.org>.

www.inf.ed.ac.uk