AI4FM — how to say “why” — in proofs

Cliff Jones
Computing Science
Newcastle University

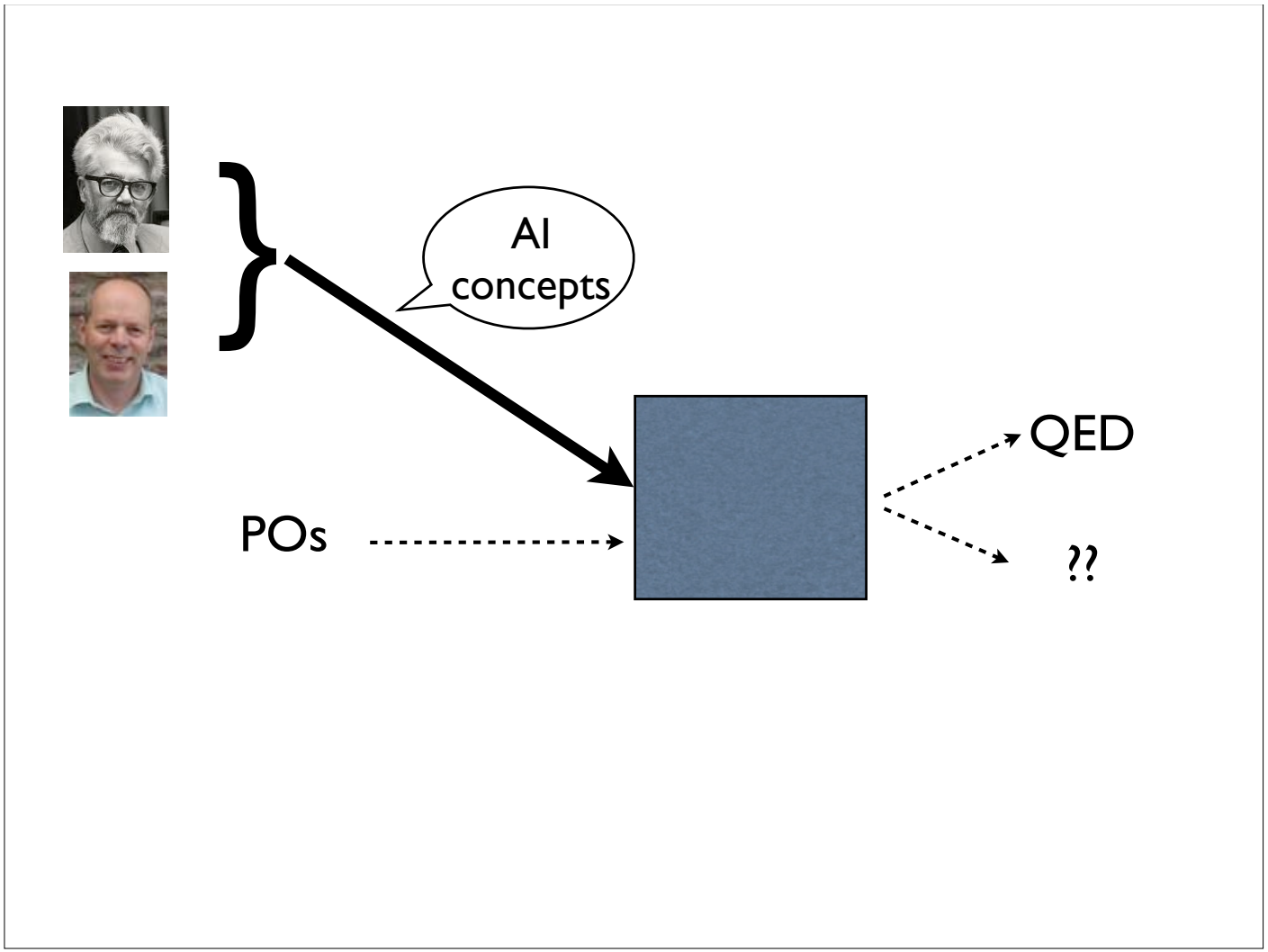
Contents

Observations

- (mine): "formal methods" pay off *during* development
  - *post facto* proof — not a smart way to debug
- (my belated) acceptance that tools are essential
- sadly, full formal reification rarely used in industry
  - (possibly imaginary) data:
  - a development step generates 500 proof obligations
  - of which, 400 discharged automatically
  - . . . search, clever heuristics, theories — courtesy of AI
  - discharging 100 POs by engineering staff is not acceptable
  - *but* 5 mini ideas cover the remaining proof tasks!
  - typically, these ideas are specific to the data structures etc.
  - (actually) non-automatic variation: 5%–40%
- idea:
  - learn the "5 ideas" from an expert; apply to other 95 POs
  - accept that this will/may not help with "next 100"

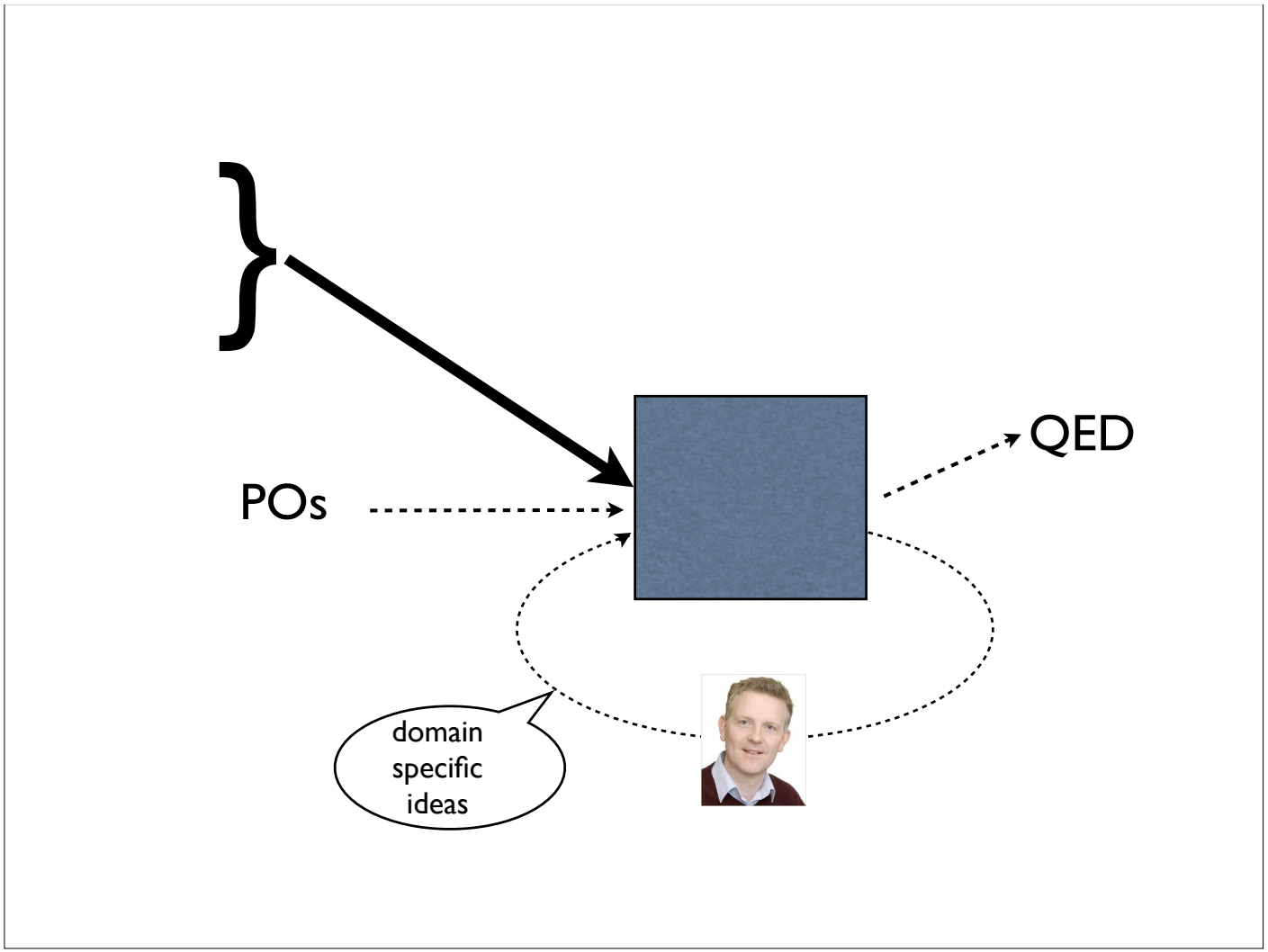
Approach

Show where "smarts" come in: traditional



Approach

Show where "smarts" come in: AI4FM



AI4FM project

Background

- Alan Bundy
  - noted AI expert — especially on proof
  - e.g. "Rippling" — [BBHI05]
  - J. Moore regularly in Edinburgh: "extracting toy problems"
- Andrew Ireland — Platform Grant with Alan
  - together with: Michael Butler
  - "revise models" proposal (one approach) — unfunded
  - Event-B; Rodin tools
  - complementary to "proof takes the strain" (AI4FM)
- CBJ
  - long-term "pusher" of FMs — posit&prove (e.g. VDM)
  - such methods themselves split the proof process!
  - FDSS; *mural* — [JJLM91]; Rodin tools

AI4FM project

- EPSRC funded
- research hypothesis:
  enough information can be learned from one proof within a family to discharge automatically the other POs of that family
- note: mine proof "process" (rather than finished proofs)
- Newcastle team: CBJ
  - Leo Freitas RA
  - Andrius Velykis PhD student
- Edinburgh team: Alan Bundy
  - Gudmund Grov RA
  - Yuhui Lin PhD student
- kindly supported by:
  - Andrew Ireland
  - Michael Butler
  - recall nice use of background proving in Rodin Tools

AI4FM project

Approach

- tactics/tacticals too "brittle"
  - anecdote (J-RA)
  - one major shortcoming: fail after minor changes
- striving for abstract *strategies*
  - must be re-usable over (related) tasks
  - we doubt they will be "sequential"
  - "forwards/backwards" dichotomy less of an issue
  - "splitting" is made difficult w/o a metric for "proof difficulty"
  - essence: cut down search space (cf. parallel activity)

AI4FM project

Report to date

- archaeology on proofs
- new proofs (with introspection)
- agreed to look at "How to say why"
  - record "intent" rather than details
  - more transferable than steps
  - capture "process"
- examples (next slides)
- theme taken up:
  - nice (internal) notes on this from Alan, Leo, . . .
  - these have certainly influenced what follows
- Leo reads!
  - Lenat's AM, . . . , Argunet
  - supports what follows

examples

- typically, want higher level than "case split"
  - "set up induction"
  - operator order mismatch (e.g. distribution needed)
  - try for a normal form
- choice of lemmas — some pro-actively generated
  - "shape" of lemmas — copied twixt theories
- J's "extract toy problem" as cutting down the search space
- Alan has a whole set from equation solving

examples of pro-activity (i)

(merging functions/operators)

1) spot similar operators in different theories

$subp : \mathbb{Z} \to \mathbb{Z} \to \mathbb{Z}$

$subp(i, j) \triangleq$ if $i = j$ then 0 else $subp(i, j + 1) + 1$ fi

2) generate induction rules

$$\boxed{\textit{subp-b}} \quad \frac{}{subp(i, i) = 0}$$

$$\boxed{\textit{subp-i}} \quad \frac{subp(i, j + 1) = k}{subp(i, j) = k + 1}$$

examples of pro-activity (ii)

Texbook 1: showing $rev(rev(s)) = s$ can get "stuck"

but the definition

$rev : X^* \to X^*$

$rev(s) \triangleq$ if $s = [\,]$ then $[\,]$ else $rev(\mathsf{tl}\, s) \curvearrowright \mathsf{hd}\, s$ fi

3) suggests that there is a need to relate rev to $\curvearrowright$

Textbook 2: recursive functions that use accumulators

4) for these (and others) being pro-active might be better

My "Σ approach"

- new name — been doing it for decades!
- sketch a state (Σ from habit)
  - quickly pinpoints questions
  - basis for discussion
- *mural* started life as a VDM model
  - . . . as have many more studies
- what follows is pretty much a first attempt
  - (although Leo has endured some even rougher versions)
  - it will evolve!
- what I'm searching for is a *framework* for ideas
- at this stage:
  - question is how to record these higher level proof views
  - come later to how to learn them

A basic model

Assume state contains various $Theory$s

$$\Sigma :: Id \xrightarrow{m} Theory$$

$$\begin{aligned} Theory :: \; & Operator \xrightarrow{m} OpDefn \\ & Id \xrightarrow{m} Conjecture \end{aligned}$$

$$\begin{aligned} OpDefn :: \; & Type \\ & Rest? \end{aligned}$$

$$\begin{aligned} Conjecture :: \; & from\colon Expr^* \\ & to\colon Expr \\ & justification\colon Tag \xrightarrow{m} \textsc{Axiom} \mid \textsc{Trusted} \mid Attempt \end{aligned}$$

$Attempt$ :: this is big – contains whole proofs – *mural* like

Some predicates

$linked : Attempt \times \Sigma \rightarrow \mathbb{B}$

$linked(a, \sigma) \triangleq$

are there links to all required hypotheses?

$complete : Attempt \times \Sigma \rightarrow \mathbb{B}$

$complete(a, \sigma) \triangleq$

do all of the required hypotheses have $justification$s?

Notes

Random decisions:

- $Conjecture$ was (could be) called $Lemma$
- we might separate AXIOMs from $Conjecture$
  but there are situations (in revising theories) where "axioms" can also have proofs
- I've merged "operators"/"functions" (and use former name)
- the final model should relate $Theory$s in a hierarchy; this could include Larch-like theories such as "collectors"
- $Attempt$ could be derived from *mural* book
  as could predicates that check whether a proof is $linked$ and $complete$
- it might be worth changing $linked$ to yield the gaps?

With Why

Overview of additions

$Theory :: \cdots$
$\quad Id \xrightarrow{m} Strategy$

$Conjecture :: \cdots$
$\quad shape: MetaType$
$\quad uses: Clue\text{-set}$

$Clue :: why: Why$
$\quad evidence: Test\text{-set}$

$Why = \cdots$

$Test :: \cdots$

$Strategy :: \cdots$

Combining the above for $Theory/Conjecture$:

$$
\begin{aligned}
Theory :: \; & Operator \xrightarrow{m} OpDefn \\
& Id \xrightarrow{m} Conjecture \\
& Id \xrightarrow{m} Strategy
\end{aligned}
$$

$$
\begin{aligned}
OpDefn :: \; & Type \\
& Rest?
\end{aligned}
$$

$$
\begin{aligned}
Conjecture :: \; & from\text{: } Expr^* \\
& to\text{: } Expr \\
& justification\text{: } Tag \xrightarrow{m} \textsc{Axiom} \mid \textsc{Trusted} \mid Attempt \\
& shape\text{: } MetaType \\
& uses\text{: } Clue\text{-set}
\end{aligned}
$$

Some detail

$$
\begin{array}{rcl}
Clue & :: & why\text{: } Why \\
 & & evidence\text{: } Test\text{-set}
\end{array}
$$

$$
Why = \textsc{NormalForm}, \textsc{CaseSplit}, \textsc{Induction}, \textsc{ReOrderOperands}, \textsc{DistributeOperators}, \textsc{MapToAnotherDatatype}, \textsc{EquationTransforms}, \textsc{LRrewr}, \cdots
$$

$$
\begin{array}{rcl}
Test & :: & predicate\text{: } Expr^* \times Expr \times Conjecture\text{-set} \to \mathbb{B} \\
 & & weight\text{: } \mathbb{Z}
\end{array}
$$

$$
\begin{array}{rcl}
Strategy & :: & needs\text{: } Why\text{-set} \\
 & & weight\text{: } \mathbb{N} \\
 & & split\text{: } Conjecture \to Conjecture\text{-set} \\
 & & combine\text{: } Conjecture\text{-set} \to Attempt \\
 & & recovery\text{: } FailureRecovery\text{-set}
\end{array}
$$

Some more functions

$analyse$: $Conjecture \rightarrow Diffn$-set

$Diffn =$ OPERATORORDER, OPERATORCOMBINATION,
DIFFERENTOPERATORS, GENERALISE, $\cdots$

$is\text{-}covered$: $Diffn \times Why \rightarrow \mathbb{B}$

$is\text{-}simpler$: $Conjecture \times Conjecture \rightarrow \mathbb{B}$

Possible scenarios (i)

for any new $OpDefn$ (pro-actively):

- combinations of operators will suggest lemmas
- if definition recursive:
  - generate induction rules
  - (simple structural and "course of values"?)
  - spot need for "accumulator" arguments for induction
- sibling $Theory$s could suggest lemmas by analogy
  - cf. $MetaType$
- decide whether to try automatic proof immediately
  - another role for learning
- even if proof fails, keep statement of the putative lemma
- using parallelism/concurrency

Possible scenarios (ii)

for any $Conjecture$, say c, (w/o a $complete$ proof $Attempt$):

- check if a $complete$ (say c') $Conjecture$ matches c
- check if any other (not yet proved) $Conjecture$ (c'') would discharge c
  - weigh up whether to try c''
- use $analyse$ to get a $Diffn$ set twixt $from/to$
- some $Diffn$s readily prompt searching for a $Conjecture$ (in the appropriate $Theory$) with a matching $Clue$
  - e.g. OPERATORORDER points to REORDEROPERANDS
- in cases such as GENERALISE, likely that a $Strategy$ from the appropriate $Theory$ will be of use
- compute $evidence$ to decide which avenue to try first
- generate a new $Attempt$ — split off new $Conjecture$s
  - . . . keeping track of where each needed

Possible scenarios (iii)

All of the above will be steered by the $weight$s; which, in turn, will be modified by experience

Notice that search is neither "forwards" nor "backwards"

when an $Attempt$ is $complete$:

- both trace back; and
- see if the result is of any use to any other $Attempt$

(of course) POs manifest themselves as (unproven) $Conjecture$s

Notes on "why"

- an example of a strategy would be SIMPLEINDUCTION:
  - it might generate three sub-conjectures whose uses are preparation, apply appropriate induction rule, clean up
- the $Clue$ set in a $Conjecture$ (say c) is about how c might be deployed
  - the $evidence$ fields in $Clue$ is a pros/cons list
  - we want to "learn" the weightings (which can be negative)
- the $Conjecture$-set (in the domain of $predicate$ in $Test$) is there because a strategy might rely on lemmas of a specific $shape$ (that might not be present for all $Theory$s)
- the list in Why is only indicative to show broad idea
  - it is certainly not a thought through list
  - furthermore, we almost certainly want to allow it to be extended dynamically

To be done

- the specific examples like RE-ORDER-OPERANDS could all be wrong and at least might be getting too concrete too early — they are there just to hint at what I'm trying to achieve
- *I think* that $Diffn$ is not identical to Why which is the reason we need $is\text{-}covered$
- "operator order"
  - can be expressed in a Lambda calculus style
  - Leo has a nice example that requires side conditions
  - would these fit better in $Conjecture$?
- we should find a place to put "counter example generators"
- figure out how to include "extract Toy problem"
  - J. Moore ACL2
  - also show way with (very) large records

What we are doing *now*

- "how to say why" + "models of why"
- remember — at this stage of AI4FM:
  - think about how to record high-level proof strategies
  - the issue of how to learn them comes next
  - $weight$s above are but a small nod to learning!
- *not* aiming at general maths proofs
  - precisely: FM POs
- understand role of "failure" in proof
  - I love the "proof critic" idea (Edinburgh)
  - . . . but do wonder if tracking failure is too sequential?
- inclusion of (parallel) "disproving"
- there are many other things that inhibit adoption of FMs:
  - tools themselves (mono-lingual)
  - no one universal method
  - "concept overload"

More information

- www.ai4fm.org
- open mailing list: ai4fm-info@jiscmail.ac.uk
- open meetings (next in Edinburgh 2010-04-28/29)
- Rippling — see [BBHI05]
- *mural* — see [JJLM91]
- AI4FM itself — see [JGB10, FJ10]


A. Bundy, D. Basin, D. Hutter, and A. Ireland.
Rippling: Meta-level Guidance for Mathematical Reasoning.
Cambridge University Press, 2005.


Leo Freitas and Cliff B. Jones.
Learning from an expert's proof: AI4FM.
In Tom Ball, Lenore Zuck, and Natarajan Shankar, editors, *UV10 (Usable Verification)*, November 2010.


Cliff B. Jones, Gudmund Grov, and Alan Bundy.
Some facets of a strategy language for proofs.
In *5th Automated Formal Methods workshop (AFM'10)*, July 2010.
Also available as Edinburgh University, School of Informatics technical report EDI-INF-RR-1377.


C. B. Jones, K. D. Jones, P. A. Lindsay, and R. Moore.
mural: A Formal Development Support System.
Springer-Verlag, 1991.