



# Learning from Expert's proof

Leo Freitas, Newcastle University  
UV10 @ MSR Redmond, 16<sup>th</sup> Nov. 2010

# The project

- Newcastle University
  - Cliff Jones, Leo Freitas, and Andrius Velykis
- University of Edinburgh
  - Alan Bundy, Gudmund Grov, and Yuhui Lin
- DEPLOY and other industrial partners
  - Bosch, Siemens, *etc.*
  - Praxis, US NRL
- 4 years – started April 2010

# Objectives

- Reduce verification bottlenecks
  - Avoid rework on (structurally) similar proofs
  - Not aiming at general mathematical problems
  - Focus on POs from development in industry
- Use machine learning for proof mining
  - Lemma suggestion and generalization
  - Inference of induction principles
  - Reasoned proof critics and plans
- Domain knowledge acquisition
  - Investigation of failed proof attempts
  - Proof family identification

# Context and Proof Support

- Source of proof obligations
  - Top-down : C by C, posit + prove
  - Bottom-up : post-facto verification
  - Both : abstraction guided documentation
- Decompose verification effort with key abstractions
  - Profit from structural / conceptual similarities
- Alan's learning scenarios
  - Proof chunking (*i.e.*, term taxonomy / hierarchy)
  - “n-proofs” (*i.e.*, proof versions: declarative, programmed, etc.)
  - Anti-unification (*i.e.*, find least-general common generalization)
  - Cut-rules (*i.e.*, lemmas suggestion and inference)
  - Meta-tagging (*i.e.*, defined function, constructor, type, etc.)

# AI4FM Approach

- Hypothesis: learning from proof processes of an expert
  - On a specific class of problems
  - Lemma suggestion and problem decomposition
  - Tool based learning from proof failures
- Rationale: proof influences modeling decisions
  - But through counter-examples / something new
  - Avoid model fiddling (just) to increase levels of proof automation
- Machine learning techniques envisaged
  - Proof planning and critics (e.g., IsaPlanner)
  - Top-down formal development (e.g., VDM, Z, B)
  - Bottom-up code-level verification (e.g., Boogie, Spark)
- Find “toy-problems”
  - Like lab mice in pharmaceutical research
  - Use strategy from toy to solve original problem

# AI4FM Approach

- **Proof expert role will still be key**
- Create strategy language
  - Beyond simply sequential tactical language
  - Take into account taxonomy of terms and their use
  - Specification method independent as much as possible
  - We are currently investigating AI algorithms/ideas to this effect
- Investigate industry-relevant proof data
- Proof data under consideration now / near future
  - Praxis' Tokeneer ID station (e.g., Ada, Spark)
  - Bosh cruise control (e.g., Event-B, Rodin, Pro-B)
  - NRL Xenon High-Assurance Hypervisor (e.g., CodeSonar, C++)
- Take inspiration from various sources
  - Event-B: various layers of abstraction and refinement
  - Boogie: targeted (to Z3/SMT) abstract intermediate language (ATP-like)
  - ACL2/Z-Eves: guess and prove, lemma generalization, and toy problems

# Finally

- We are at the beginning
  - More info at <http://www.ai4fm.org>
  - AI4FM mailing list is open: [ai4fm-info@jiscmail.ac.uk](mailto:ai4fm-info@jiscmail.ac.uk)
  - We have meetings planned for sharing results / ideas
- We would love to hear your feedback / criticisms
  - What do you like about the idea?
  - How would you do differently?
  - Goal: reduce residual / repetitive POs

## Questions?