Discovery of Invariants through Automated Theory Formation

Maria Teresa Llano[1]
Joint work with:
Gudmund Grov[2] Andrew Ireland[1] Alison Pease[2]

[1]School of Mathematical and Computer Sciences
Heriot-Watt University

[2]School of Informatics
University of Edinburgh

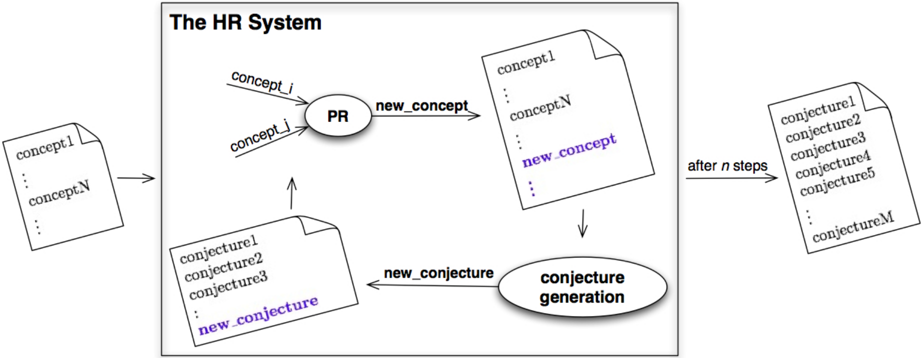
AI meets Formal Software Development
Dagstuhl, July 2012

Motivation

- The identification of invariants is a key aspect of the verification of formal models and the development of reliable systems.
- Discovering correct and meaningful invariants for a model represents a significant challenge.
- Our aim is to increase the level of automation in discovering invariants.
- We use Automated Theory Formation (ATF) techniques to reason about the behaviour of a specification and suggest candidate invariants.

Context: ATF and HR

- ATF is a machine learning technique that builds theories about objects of interest within a given domain.
- HR[1] is a system that implements ATF.



[1] http://www.doc.ic.ac.uk/~sgc/hr/

Theory formation example – Generating new concepts

Divisors	
1	1
2	1
2	2
3	1
3	3
.	.
.	.
10	1
10	2
10	5
10	10

size $< 1 >$

Number of divisors	
1	1
2	2
3	2
4	3
5	2
6	4
7	2
8	4
9	3
10	4

split $< 2 = 2 >$

Primes
2
3
5
7

- **size**$<$ *columns_set* $>$**:** counts the number of appearances of each element in the specified columns of the data table.
- **split**$<$ *columns_set* $=$ *values_set* $>$**:** filters a data table according to given values.

Theory formation example – Making conjectures

- Each time a new concept is generated HR checks to see whether a conjecture can be constructed.

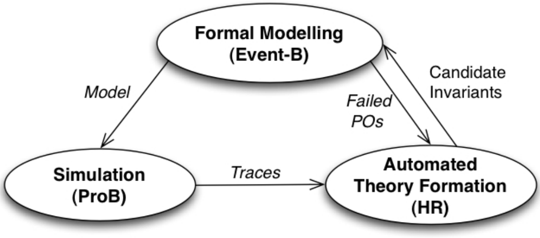
Prime numbers
2
3
5
7

$$\Rightarrow$$

Non-square numbers
2
3
5
6
7
8
10

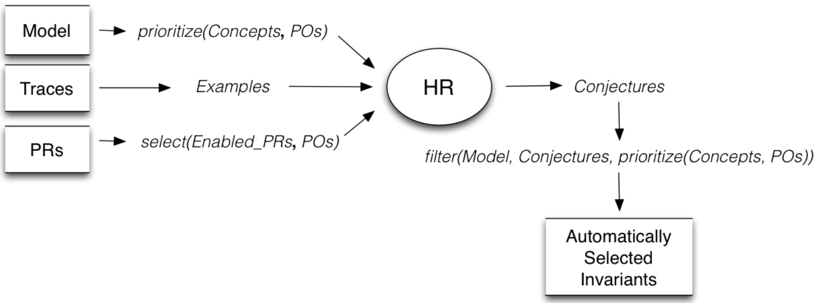
"All prime numbers are non-squares"

Our Approach



- Formal modelling that supports proof.
- Simulation is used to generate system traces.
- ATF is used to generate conjectures from the analysis of traces.
- We use proof failure analysis to focus the theory formation search.

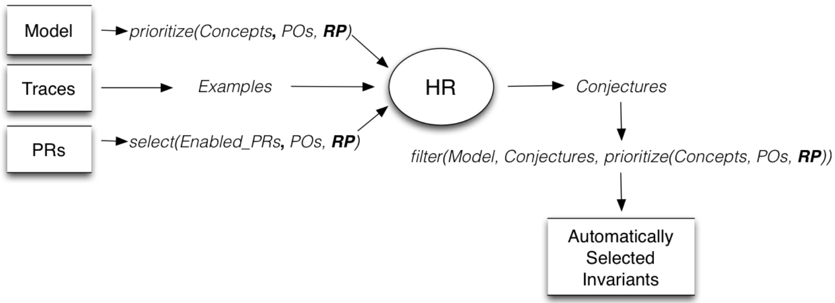
HREMO



Proof failure analysis guides:

- The prioritization of concepts.
- The selection of the appropriate PRs to be used during the search.
- The filtering of the conjectures we are interested in.

HREMO + Refinement Plans



- Refinement plans represent common patterns of refinement together with proofs (POs).
- To increase the flexibility of our plans we use schemas to represent invariant patterns.
- HREMO is used to instantiate such invariant patterns for a given refinement step.

Results

- We have extended the HR system by implementing our heuristics – we call this extension HREMO.
- We have performed a series of experiments to automatically discover invariants of various Event-B models through HREMO.
  - 7 case studies and a total of 23 refinement steps:
    - 15 steps for which all failed POs were discharged.
    - 5 steps where not all failed POs were discharged.
    - 3 steps were not applicable.
  - HREMO has proven to be very good at finding gluing invariants – these are invariants that relate the state of a refined model with the state of its abstraction.

Challenges and opportunities

- Increase the randomness of the simulation traces provided to HREMO.
- Add new PRs and/or conjecture making strategies suitable within formal modelling, e.g. a PR that allows the permutation of columns within a data table.
- Reduce the large number of concepts and conjectures generated by HR.
  - by adding measures that evaluate the interestingness of concepts and conjectures in the formal modelling context?
- Use HR to explore user scenarios and early design ideas to support formal modelling.