Imperial College London

COMPUTATIONAL Creativity Group

Computational Creativity Autumn School

Lecture 4

Building an Automated Mathematician

Simon Colton
Computational Creativity Group
Department of Computing
Imperial College, London

ccg.doc.ic.ac.uk

Talk Overview

- Automated theory formation (ATF) in pure mathematics
  - Overview, applications and quick demo
- Next generation approaches to ATF:
  - Completed PhDs
    - A computational model of Lakatos-style reasoning (Alison Pease)
    - A global workspace approach to combined reasoning (John Charnley)
- Future work

Automated Theory Formation (ATF) in Pure Maths

- Theory constituents:
  - Examples, concepts, conjectures, theorems and proofs
- Concepts categorise and describe examples, conjectures relate concepts and proofs demonstrate truth of true conjectures called theorems
  - E.g. integers, prime numbers, squares can't be primes, simple proof of this: number of divisors
- HR system
  - Fully automated construction methods for ATF

HR Project Overview

(Please ask for more details)

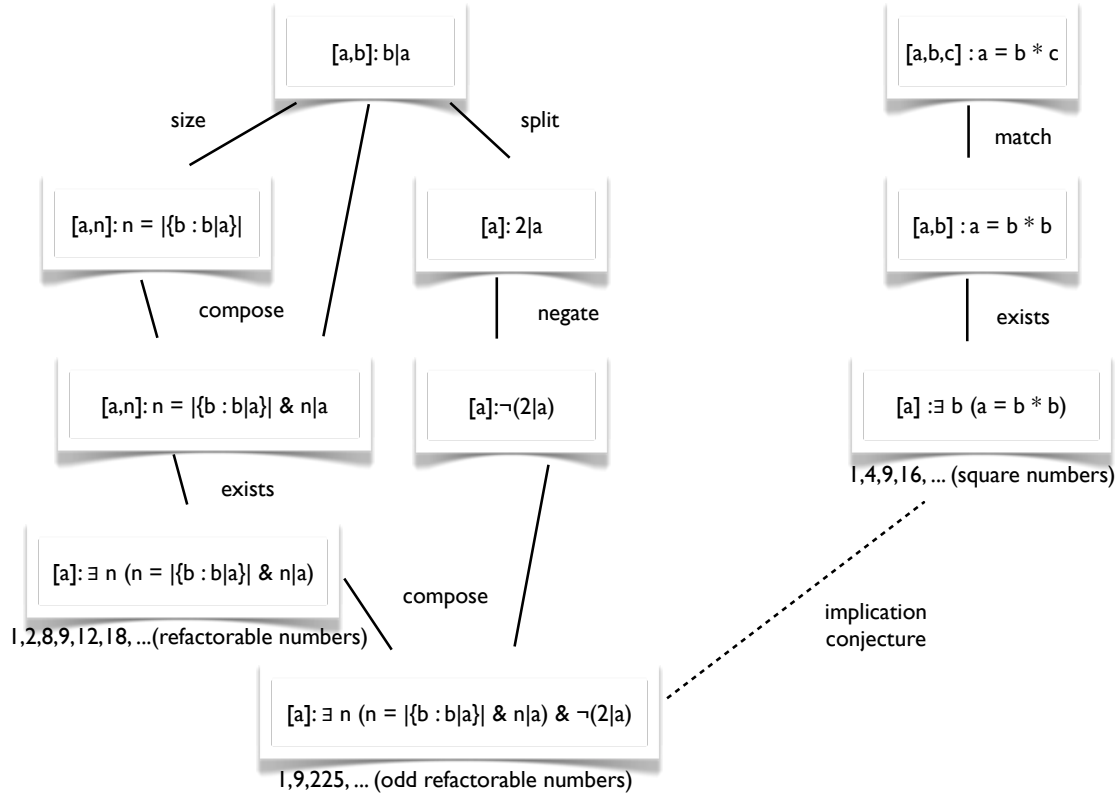
- 15 year project, with two major implementations (5 PhDs & ~10 MScs & ...)
- Mathematical theory formation
  - Works with little background knowledge, e.g., axioms, core concepts
- Uses production rules to form new concepts from old (17)
  - And measures of interestingness to drive various heuristic searches (20)
- Uses empirical conjecture making to find patterns between concepts (4)
  - And third party software to prove/disprove the conjectures
- With an emphasis on combining AI reasoning systems (ATP, constraint solving, machine learning, model generation, satisfiability solving, CAS, etc.)
- Applied to discovery tasks in number theory, finite algebras, graph theory
  - And to non-maths projects (vision, music, bioinformatics, graphic design)

Core Functionality

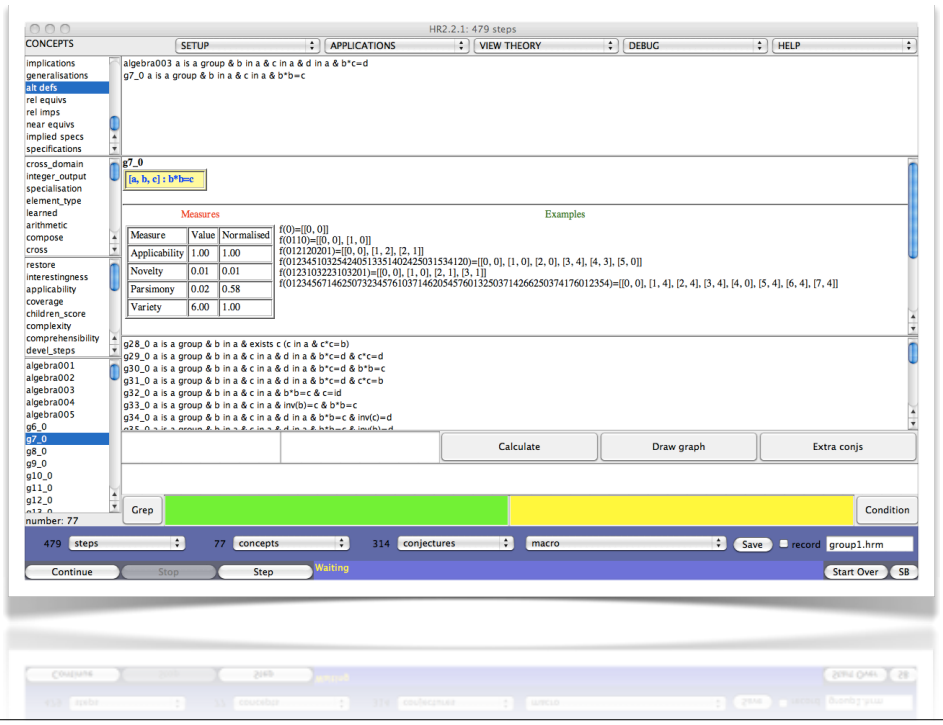
Definition: a refactorable number is such that its number of divisors is itself a divisor, i.e., τ(n) divides n

Odd refactorable numbers are perfect squares

Core Functionality



Demonstration



Application 1
Number Sequences

- Challenge: to invent sequences worthy of encyclopedia of integer sequences (novel and mathematically interesting)
- 3 series of experiments: with HR alone; with Maple; with encyclopedia
- Using Otter theorem prover as a filter for dull conjectures
- Refactorable numbers paper appeared in math. literature
- > 20 sequences in Encyclopedia
  - E.g., integers with $\tau(n)$ prime
- Some theorems
  - No triples of refactorables
  - $\Phi(n)$ square $\rightarrow$ $\tau(n)$ even
  - $\sigma(n)$ prime $\rightarrow$ $\tau(n)$ prime
  - $\tau(n)$ odd $\leftrightarrow$ n is square

Application 2
Algebraic Classification

- Work with Sorge, Meier, and McCasland
- HR part of a larger, bootstrapping system
  - Involving a record breaking(!) number of AI systems: ML, SAT, CAS, ATP
- Given the axioms of an algebraic domain, a particular size and an equivalence relation
  - It returns a fully proved classification theorem for that algebra and that size with respect to that relation
  - E.g., groups of size six are either Abelian or non-Abelian, up to isomorphism

Bootstrapping Algorithm Example

A_2	a	b	c
a	b	a	c
b	a	c	b
c	c	b	a

A_4	a	b	c
a	a	c	b
b	c	b	a
c	b	a	c

A_6	a	b	c
a	a	b	c
b	b	c	a
c	c	a	b

A_7	a	b	c
a	a	b	c
b	c	a	b
c	b	c	a

A_8	a	b	c
a	a	c	b
b	b	a	c
c	c	b	a

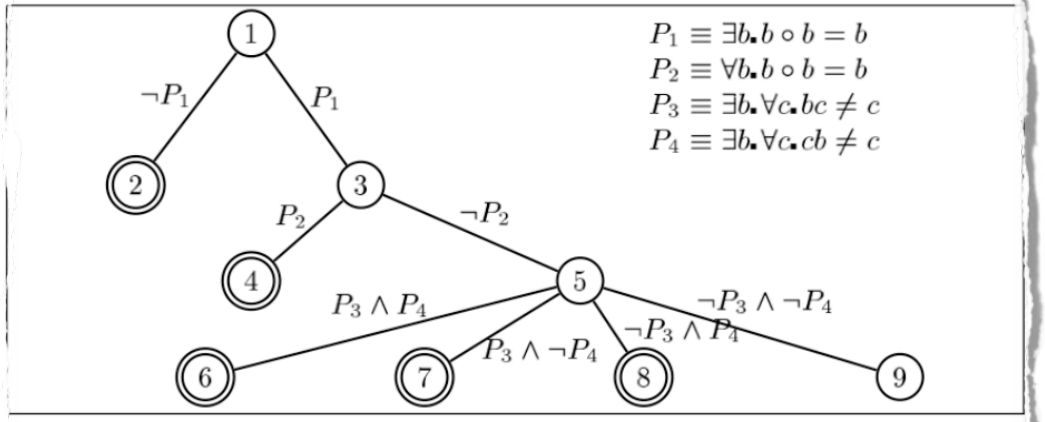


Fig. 3. Decision tree for the classification problem of order 3 quasigroups.

Recent Results

Isomorphism classification theorem for quasigroups of order 4

We are given the following properties of quasigroups:

P_1:$\forall b_\bullet \forall c_\bullet (b \circ (b \circ c)) = c$
P_2:$\forall b_\bullet \exists c_\bullet (c \circ c) = b$
P_3:$\forall b_\bullet ((b \circ b) \circ (b \circ b)) = (b \circ b)$
P_4:$\forall b_\bullet \forall c_\bullet (c \circ b) = (b \circ c)$
P_5:$\exists b_\bullet (b \circ b) = b$
P_6:$\forall b_\bullet \forall c_\bullet (c \circ b) = (b \circ c)$
P_7:$\forall b_\bullet \forall c_\bullet (c \circ b) = (b \circ c)$
P_8:$\forall b_\bullet ((b \circ b) \circ (b \circ b)) = (b \circ b)$
P_9:$\forall b_\bullet \exists c_\bullet ((c \circ b) \circ c) = b$
P_{10}:$\forall b_\bullet \exists c_\bullet (c \circ b) = c$
P_{11}:$\forall b_\bullet \exists c_\bullet (b \circ c) = c$
P_{12}:$\forall b_\bullet \exists c_\bullet ((c \circ b) \circ (c \circ b)) = c$
P_{13}:$\exists b_\bullet (b \circ b) = b$
P_{14}:$\forall b_\bullet \exists c_\bullet (c \circ b) = c$
P_{15}:$\exists b_\bullet (b \circ b) = b$
P_{16}:$\forall b_\bullet \exists c_\bullet (c \circ (b \circ c)) = b$
P_{17}:$\forall b_\bullet ((b \circ b) \circ b) = b$
P_{18}:$\forall b_\bullet (b \circ b) = b$
P_{19}:$\forall b_\bullet \exists c_\bullet (c \circ c) = b$
P_{20}:$\forall b_\bullet \exists c_\bullet \exists d_\bullet ((c \circ d) = b \wedge (\neg(d \circ d) = b))$
P_{21}:$\forall b_\bullet \forall c_\bullet ((\neg(b \circ c) = b) \vee (c \circ c) = c)$
P_{22}:$\forall b_\bullet \exists c_\bullet (c \circ b) = c$
P_{23}:$\exists b_\bullet (b \circ b) = b$
P_{24}:$\forall b_\bullet ((b \circ b) \circ (b \circ b)) = (b \circ b)$
P_{25}:$\exists b_\bullet (b \circ b) = b$
P_{26}:$\exists b_\bullet \exists c_\bullet ((b \circ c) = b \wedge ((\neg(b \circ b) = c) \wedge (c \circ b) = b))$
P_{27}:$\forall b_\bullet ((\neg(b \circ (b \circ b)) = b) \vee ((b \circ b) \circ b) = b)$
P_{28}:$\forall b_\bullet \forall c_\bullet ((\neg(b \circ c) = b) \vee ((b \circ b) = c \vee (\neg(c \circ b) = b)))$
P_{29}:$\forall b_\bullet \exists c_\bullet ((c \circ (c \circ b)) = b \wedge (\neg(b \circ c) = (c \circ b)))$
P_{30}:$\forall b_\bullet \exists c_\bullet ((\neg(c \circ (c \circ b)) = b) \wedge (b \circ c) = (c \circ b))$
P_{31}:$\forall b_\bullet \exists c_\bullet ((b \circ c) \circ (b \circ c)) = c$
P_{32}:$\forall b_\bullet \exists c_\bullet (c \circ b) = c$
P_{33}:$\forall b_\bullet ((b \circ b) \circ (b \circ b)) = (b \circ b)$
P_{34}:$\exists b_\bullet (b \circ b) = b$
P_{35}:$\exists b_\bullet \exists c_\bullet ((b \circ c) = c \wedge (\neg(b \circ b) = b))$
P_{36}:$\exists b_\bullet ?((b \circ (b \circ b)) = b \wedge (\neg((b \circ b) \circ b) = b)$

Then quasigroups of order 4 are characterised up to isomorphism by one of the following conjunction of properties (and their negations):

$(P_1 \wedge P_3 \wedge P_6)$
$(P_1 \wedge P_3 \wedge \neg P_6)$
$(P_1 \wedge \neg P_3 \wedge P_7)$
$(P_1 \wedge \neg P_3 \wedge \neg P_7 \wedge P_{12})$
$(P_1 \wedge \neg P_3 \wedge \neg P_7 \wedge \neg P_{12} \wedge \neg P_{19} \wedge P_{25})$
$(P_1 \wedge \neg P_3 \wedge \neg P_7 \wedge \neg P_{12} \wedge \neg P_{19} \wedge \neg P_{25})$
$(P_1 \wedge \neg P_3 \wedge \neg P_7 \wedge \neg P_{12} \wedge P_{19})$
$(\neg P_1 \wedge P_2 \wedge \neg P_5 \wedge P_{10})$
$(\neg P_1 \wedge P_2 \wedge \neg P_5 \wedge \neg P_{10})$
$(\neg P_1 \wedge P_2 \wedge P_5 \wedge \neg P_{11})$
$(\neg P_1 \wedge P_2 \wedge P_5 \wedge P_{11} \wedge \neg P_{18})$
$(\neg P_1 \wedge P_2 \wedge P_5 \wedge P_{11} \wedge P_{18})$
$(\neg P_1 \wedge \neg P_2 \wedge \neg P_4 \wedge P_9 \wedge P_{16} \wedge \neg P_{17})$
$(\neg P_1 \wedge \neg P_2 \wedge P_4 \wedge \neg P_8 \wedge \neg P_{13})$
$(\neg P_1 \wedge \neg P_2 \wedge P_4 \wedge \neg P_8 \wedge \neg P_{13})$
$(\neg P_1 \wedge \neg P_2 \wedge P_4 \wedge P_8 \wedge \neg P_{14} \wedge \neg P_{20})$
$(\neg P_1 \wedge \neg P_2 \wedge P_4 \wedge P_8 \wedge \neg P_{14} \wedge P_{20})$
$(\neg P_1 \wedge \neg P_2 \wedge P_4 \wedge P_8 \wedge P_{14})$
$(\neg P_1 \wedge \neg P_2 \wedge \neg P_4 \wedge \neg P_9 \wedge P_{15} \wedge \neg P_{21} \wedge \neg P_{26} \wedge \neg P_{32} \wedge \neg P_{35})$
$(\neg P_1 \wedge \neg P_2 \wedge \neg P_4 \wedge \neg P_9 \wedge P_{15} \wedge \neg P_{21} \wedge \neg P_{26} \wedge \neg P_{32} \wedge P_{35})$
$(\neg P_1 \wedge \neg P_2 \wedge \neg P_4 \wedge \neg P_9 \wedge P_{15} \wedge \neg P_{21} \wedge \neg P_{26} \wedge P_{32})$
$(\neg P_1 \wedge \neg P_2 \wedge \neg P_4 \wedge \neg P_9 \wedge P_{15} \wedge \neg P_{21} \wedge P_{26} \wedge \neg P_{33})$
$(\neg P_1 \wedge \neg P_2 \wedge \neg P_4 \wedge \neg P_9 \wedge P_{15} \wedge \neg P_{21} \wedge P_{26} \wedge P_{33} \wedge \neg P_{36})$
$(\neg P_1 \wedge \neg P_2 \wedge \neg P_4 \wedge \neg P_9 \wedge P_{15} \wedge \neg P_{21} \wedge P_{26} \wedge P_{33} \wedge P_{36})$

Better Results

We are given the following five functions which return coefficients of element types for quasigroups, Q:

$f_1(Q) = |\{b : b \in Q\}|$
$f_2(Q) = 0$ if $\nexists x_\bullet x * x = x$ or $f_2(Q) = |\{b : \exists d_\bullet (d * d = b)\}|$ otherwise
$f_3(Q) = |\{b : \exists c_\bullet \exists d_\bullet (c * b = b * c = d \wedge c * d \neq b \wedge b * d \neq c))\}|$
$f_4(Q) = |\{b : \exists c_\bullet \exists d_\bullet (c * d = d * c = b \wedge b * c \neq d)\ )\}|$
$f_5(Q) = |\{b : (b * b) * b = b\}|$

Then, quasigroups up to and including size 4 are characterised up to isomorphism by the following parameterisations:

1.1) $f_1 = 1$
2.1) $f_1 = 2$
3.1) $f_1 = 3 \wedge f_2 = 0$
3.2) $f_1 = 3 \wedge f_2 = 3 \wedge f_3 = 2$
3.3) $f_1 = 3 \wedge f_2 = 3 \wedge f_3 = 0$
3.4) $f_1 = 3 \wedge f_2 = 1 \wedge f_3 = 2$
3.5) $f_1 = 3 \wedge f_2 = 1 \wedge f_3 = 0$
4.1) $f_1 = 4 \wedge f_2 = 1 \wedge f_3 = 4$
4.2) $f_1 = 4 \wedge f_2 = 4 \wedge f_3 = 2$
4.3) $f_1 = 4 \wedge f_2 = 4 \wedge f_3 = 3$
4.4) $f_1 = 4 \wedge f_2 = 2 \wedge f_3 = 4$
4.5) $f_1 = 4 \wedge f_2 = 0 \wedge f_3 = 0$
4.6) $f_1 = 4 \wedge f_2 = 3 \wedge f_3 = 0$
4.7) $f_1 = 4 \wedge f_2 = 2 \wedge f_3 = 3$
4.8) $f_1 = 4 \wedge f_2 = 3 \wedge f_3 = 1$
4.9) $f_1 = 4 \wedge f_2 = 1 \wedge f_3 = 0 \wedge f_4 = 0$
4.10) $f_1 = 4 \wedge f_2 = 1 \wedge f_3 = 2 \wedge f_4 = 4$
4.11) $f_1 = 4 \wedge f_2 = 2 \wedge f_3 = 0 \wedge f_4 = 2$
4.12) $f_1 = 4 \wedge f_2 = 2 \wedge f_3 = 2 \wedge f_4 = 0$
4.13) $f_1 = 4 \wedge f_2 = 0 \wedge f_3 = 4 \wedge f_4 = 2$
4.14) $f_1 = 4 \wedge f_2 = 4 \wedge f_3 = 0 \wedge f_4 = 3$
4.15) $f_1 = 4 \wedge f_2 = 4 \wedge f_3 = 0 \wedge f_4 = 0$
4.16) $f_1 = 4 \wedge f_2 = 0 \wedge f_3 = 4 \wedge f_4 = 4$
4.17) $f_1 = 4 \wedge f_2 = 2 \wedge f_3 = 2 \wedge f_4 = 2$
4.18) $f_1 = 4 \wedge f_2 = 2 \wedge f_3 = 2 \wedge f_4 = 4$
4.19) $f_1 = 4 \wedge f_2 = 2 \wedge f_3 = 0 \wedge f_4 = 0$
4.20) $f_1 = 4 \wedge f_2 = 1 \wedge f_3 = 2 \wedge f_4 = 0$
4.21) $f_1 = 4 \wedge f_2 = 1 \wedge f_3 = 0 \wedge f_4 = 2$
4.22) $f_1 = 4 \wedge f_2 = 0 \wedge f_3 = 3 \wedge f_4 = 4$
4.23) $f_1 = 4 \wedge f_2 = 0 \wedge f_3 = 3 \wedge f_4 = 0$
4.24) $f_1 = 4 \wedge f_2 = 0 \wedge f_3 = 3 \wedge f_4 = 3$
4.25) $f_1 = 4 \wedge f_2 = 0 \wedge f_3 = 3 \wedge f_4 = 2$
4.26) $f_1 = 4 \wedge f_2 = 3 \wedge f_3 = 3 \wedge f_4 = 0$
4.27) $f_1 = 4 \wedge f_2 = 3 \wedge f_3 = 3 \wedge f_4 = 2$
4.28) $f_1 = 4 \wedge f_2 = 3 \wedge f_3 = 2 \wedge f_4 = 3$
4.29) $f_1 = 4 \wedge f_2 = 3 \wedge f_3 = 3 \wedge f_4 = 3$
4.30) $f_1 = 4 \wedge f_2 = 0 \wedge f_3 = 2 \wedge f_4 = 2 \wedge f_5 = 0$
4.31) $f_1 = 4 \wedge f_2 = 0 \wedge f_3 = 2 \wedge f_4 = 2 \wedge f_5 = 2$
4.32) $f_1 = 4 \wedge f_2 = 3 \wedge f_3 = 2 \wedge f_4 = 2 \wedge f_5 = 1$
4.33) $f_1 = 4 \wedge f_2 = 3 \wedge f_3 = 2 \wedge f_4 = 2 \wedge f_5 = 2$
4.34) $f_1 = 4 \wedge f_2 = 1 \wedge f_3 = 3 \wedge f_4 = 2 \wedge f_5 = 2$
4.35) $f_1 = 4 \wedge f_2 = 1 \wedge f_3 = 3 \wedge f_4 = 2 \wedge f_5 = 1$

Larger Systems

- TM = HR + Otter + MACE
  - Theorem modification for increased flexibility
- HOMER = HR + Otter + Maple
  - Discovery in number theory
- ICARUS = HR + Otter + CLPFD
  - Reformulation of constraint satisfaction problems
- Bootstrap = HR + lots of other systems
  - Discovery of algebraic theorems

Evaluation Methodologies

- Control over the search mechanism

  Is there an uncontrollable butterfly effect?
- Qualitative and quantitative comparison against other systems
  - AM, Eurisko, GT, Cyrano, etc.
- Re-invention of important results (inspiring and others)
  - Prime numbers, Abelian groups, complete graphs, etc.
- Invention of new mathematics
  - Number theory, finite algebras
- Application to improving AI techniques
  - Constraint solving, machine learning, theorem proving
- Application to non-mathematical domains
  - Software verification tasks, fitness function generation, bioinformatics, etc.

ATF: State of Play

- Core theoretical and practical work (Bundy, Walsh, Muggleton)
- Applications to mathematical discovery
  - Number theory (Huczynska, Dennis)
  - Algebra discovery (Torres, Cairns, Sorge, Pease)
  - Algebraic classification (Sorge, McCasland, Meier)
  - Graph theory (Wagner)
- More sophisticated models
  - Cross domain (Steel)
  - Integration into MathWeb (Zimmer, Sutcliffe)
  - Lakatos-style social reasoning (Pease, Smaill, Lee)
  - Cognitive models of axiomatisation (the wheelbarrow team)
  - Meta-theory formation (Torres)
  - Global Workspaces (Charnley, Shanahan)
  - Multi-agent theory formation
  - Puzzle generation
- AI problem solving and invention
  - "Proving" non-theorems (Pease)
  - Case-splitting theorems (Hoermann)
  - Reformulation of CSPs (Miguel, Charnley)
  - TPTP theorem invention (Sutcliffe, Zimmer, Gao)
  - Dynamic Investigation Problems (Ramezani)
- Applications to graphic design
  - Scene construction methods
    - Inventing fitness functions/ Inducing and using CSPs
  - Objet-trouve browsing methods (Torres, Rueger)
- Minor applications
  - Predictive toxicology, computer networks
  - Music (Vine, Marikar)
  - Vision - dice games (Santos, Magee)
  - Mixed reality games (Nika, Gow, Baumgarten)

A Computational Model of Lakatos-style Reasoning

- PhD project of Alison Pease, University of Edinburgh
  - Lakatos presents a fallibilist approach to maths - proofs, conjectures and concepts are fluid and open to negotiation
  - "Proofs and refutations" - using a classroom analogy
- He saw mathematics as an adventure in which conjectures and proofs are gradually refined – via patterns of analysis which he categorised into various methods
  - He highlighted the role of counterexamples in mathematics
- Alison's hypotheses: (i) it is possible to provide a computational reading of Lakatos's theory (ii) it is useful to do so

Aims of the Project

- Build a model of Lakatos's theory
- Show that doing so has enabled us to:
  - Clarify and extend Lakatos's theory
  - Investigate different domains to which his theory applies
  - Use our model to evaluate his methods
- Evaluate our implementation and extended theory of mathematical progress

The HRL System

- Extends the HR system
- Firstly, with an agency model
  - Teacher and student(s): copies of HR
  - Independent work + discussion phase
  - Communication of theory constituents
- Secondly, with Lakatos-style methods
  - As discussed in the next few slides

Lakatos-style Methods

- Given a conjecture C

  For all polyhedra
  V-E+F = 2

  - *Surrender:* reject the conjecture
  - *Monster-barring:* given a counter-example, modify concept definitions in C to exclude it
  - *Piecemeal exclusion:* generalise from counterexamples, and modify C by excluding that type of object
  - *Strategic withdrawal:* generalise from the positive examples, and modify C by limiting it to that type of object
  - *Lemma-incorporation:* Find which step of a faulty proof a counterexample violates and then make that step a condition of the conjecture

Lakatos-style Methods

- Given a conjecture C

  For all polyhedra
  V-E+F = 2

  - *Surrender:* reject the conjecture
  - *Monster-barring:* given a counter-example, modify concept definition to exclude it
  - *Piecemeal exclusion:* generalise from counterexamples, and modify C by excluding that type of object
  - *Strategic withdrawal:* generalise from the positive examples, and modify C by limiting to that type of object
  - *Lemma-incorporation:* Find which step of a faulty proof a counterexample violates and then make that step a condition of the conjecture

Monster Barring
Implementation

- Decide when to propose to bar an entity
  - If it breaks the conjecture under discussion; if it breaks over a user-defined percentage of conjectures in theory; if it forces other entities to become counterexamples
- React to a proposal to bar an entity
- How to perform monster-barring
  - Vague-to-vague, or vague-to-specific
- Teacher counts votes and tells the students to add entity to theory/ downgrade entity to pseudo entity
- Extended to monster-accepting

Monster Barring
HRL Examples

1) S1: 1-30; S2:1-30; S3: 2-30
S3 makes conj $-\exists n$ st $n * n = n$
S1,**S2** find counterexample 1
S3 finds 1 breaks 14% of its conjs, and votes to bar
S1, **S2** evaluate proposal but vote to keep (breaks 9 and 7%)
teacher requests S3 add 1 to its theory

2) S1: dog, dolphin, platypus, bat, trout, herring, shark, eel, lizard, crocodile, t_rex, turtle, snake, eagle, ostrich, penguin, cat; S2: as above - no platypus
S2 makes conj $-\exists$ an animal which produces milk and lays eggs
S1 sends platypus countereg
S2 finds platypus breaks 11% of its conjs and votes to bar
S1 finds it breaks 31% of its own conjs, and agrees
they both perform downgradeEntityToPseudoEntity

Lemma Incorporation

For all polyhedra
V-E+F = 2

- 'Proof':
  - Remove one face of the polyhedron and stretch it flat onto a blackboard
    (If C holds, we now have $V - E + F = 1$)
  - Cut all the polygons on the board into triangles by drawing more edges
    ($V - E + F = 1$ still holds)
  - Remove each triangle on the board one by one
    ($V - E + F = 1$ continues to hold)

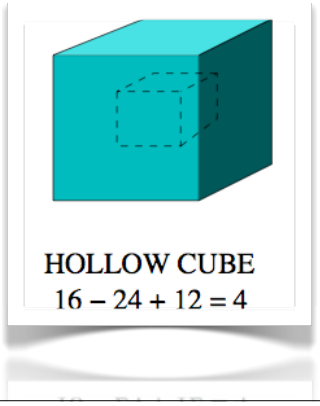
Lemma Incorporation
Implementation

- Represent faulty proof – an argument tree of conjectures
- Determine which type of lemma-incorporation to perform: global and local, hidden lemma-incorporation, local-only lemma-incorporation
- For global and local, with global conjecture $P \rightarrow Q$, and faulty lemma $R \rightarrow S$, form concept C = 'objects which satisfy faulty lemma' by merging the two concepts R and S. Modify the global conjecture to $C \rightarrow Q$
- Model of surprise:
  - An entity e is surprising with respect to a conjecture 'all x $P(x) \rightarrow Q(x)$' if there exists conjecture C' of form 'all x $P(x) \rightarrow (Q(x) \wedge R(x))$' which is satisfied by all x except e
  - An entity e is surprising with respect to conjecture $P(x) \rightarrow Q(x)$ if $\neg P(e)$

Lemma Incorporation
Lakatos Example

For all polyhedra
V-E+F = 2

- Given the hollow cube, C′ becomes ‘for any polyhedra which, by removing one face can be stretched flat onto a blackboard, V − E + F = 2’



Lemma Incorporation
HRL Example

Global conjecture: Let G be an algebra with one binary operator, and g, h, k be in G. Then $gh = gk \rightarrow h = k$.

Proof: lemma 1: $gh = gk \rightarrow g^{-1}(gh) = g^{-1}(gk)$

lemma 2: $g^{-1}(gh) = g^{-1}(gk) \rightarrow (g^{-1}g)h = (g^{-1}g)k$

lemma 3: $(g^{-1}g)h = (g^{-1}g)k \rightarrow eh = ek$

lemma 4: $eh = ek \rightarrow h = k$

Student found a counterexample to global conjecture, and lemma 2. It formed the concept:

algebras with tuples g, h and k such that $(g^{-1}g)h = (g^{-1}g)k$,

and from this modified the global conjecture to:

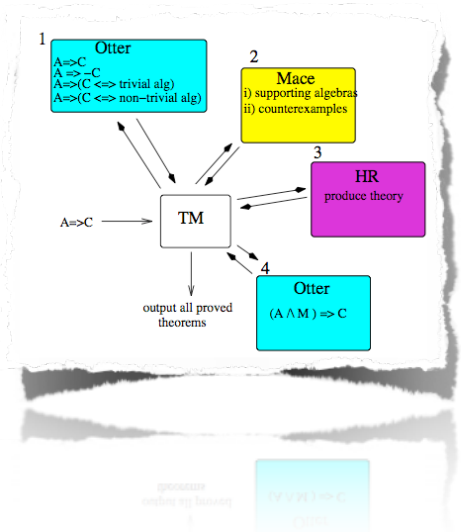
for all algebras with elements g, h and k,
$(g^{-1}g)h = (g^{-1}g)k \rightarrow (gh = gk \rightarrow h = k)$.

Evaluation: Aims

- Build a model of Lakatos's theory – HRL
- Show that doing so has enabled us to clarify and extend Lakatos's theory
  - Formal representations of methods and applications, when we might surrender a conjecture, how ambiguity may arise and how it can be resolved, application of methods to equivalence and non-existence conjectures, counterexample-barring and monster-accepting, ways in which we are surprised when faced with a counterexample in a proof...
- Investigate different domains to which his theory applies (number theory, group theory, animals dataset + more recently)
- Use our model to evaluate his methods (fine-tuning possible)

Evaluation: Axes

- From a philosophical standpoint:
  - Thagard's notions; Sloman's notions
- From an AI standpoint:
  - ATF can now generate new types of interesting conjectures
  - Discuss and reject controversial objects
  - Represent and improve proof plans
  - Applied to non-theorem handling system (TM)



A Global Workspace Approach to Combining Reasoning Systems

- PhD project of John Charnley
- Aims
  - Provide a generic framework within which combinations of AI reasoning systems can be fruitfully employed (e.g., automated theory formation + others)
  - Demonstrate improvements over ad-hoc combination approaches (e.g., ease of use, power, speed)
  - Throw light on cognitive science theory of global workspace architectures

ICARUS Case Study

- Constraint Satisfaction Problems (CSPs)
  - Very powerful + widely used AI technique
- Defining CSPs is an art form
  - Lots of projects to automatically reformulate CSPs
- Can use ATF to induce additional constraints for a given CSP (semi-automated approach)
  - John fully automated the process

ICARUS Case Study

- System overview
  - Use solver (CLPFD) to generate small example solutions to a given CSP
  - Form a theory about the solutions (HR)
    - Containing proved theorems (Otter)
  - Add individual theorems as additional constraints
    - Then try them in pairs, triples
    - See if the reformulated CSP is solved faster

ICARUS - Results

Algebra	No. ex.	Test order	HR time	Test time	Total time	Proportion (%)
QG7	12	6	9:31	6:29	16:00	8.6
QG6	2	6	11:35	1:37	13:12	12.2
Group	22	5	3:31	21:00	2	
QG3	8	5	5:32	1:55		
QG4	20	5	4:29	2:28		
Ring	25	5	9:17	19:48	2	
QG5	8	5	12:57	4:15	1	
Moufang	25	3	8:07	7:37	1	
QG1	16	4	15:26	7:01	2	
QG2	16	4	20:48	6:51	2	
Loop	16	5	6:42	3:07		
Medial	22	3	24:08	2:37	2	
Semigroup	23	3	4:15	1:39		
Monoid	32	4	4:32	4:55		

Table 1. Reformulation times (minutes:seconds) an model solving time for training (small) dor

Algebra	Domain size	Basic model	Reformulated model	Proportion (%)
QG3	7	3:09	1:24	44.5
	8	10:07:02	3:10:03	31.3
QG4	6	0:07	0:04	54.2
	7	11:19	5:18	46.8
QG5	7	1:24	1:17	91.7
	8	38:05	28:52	75.8
QG6	9	27:25	6:25	23.4
	10	24:21:00	5:53:03	24.2
QG7	8	19:12	3:33	18.5
	9	27:12:35	4:19:42	15.9
Group	8	16:37	4:15	25.6
	9	4:36:39	28:27	10.3
Moufang	4	0:11	0:08	72.3
	5	10:49	4:19	39.9
Ring	7	0:37	0:30	79.8
	8	4:22	2:09	49.5

Table 2. Solving times (hours:minutes:seconds) and proportion of basic model solving time for testing (larger) domain sizes. Note that only those algebras where a speed-up was identified in table 1 are shown

Global Workspaces

- Theory by Baars et. al
  - Cognitive science theory used to model massively parallel mammalian brain architectures and help explain attention
  - Used by Shanahan for modeling, in particular the Frame problem in AI
- Multiple processes attached to a global workspace
  - Processing occurs in rounds with only one process allowed to broadcast its results each time
- Rather a simple architecture
  - Research question: can we use for combining systems?

GC Framework and Toolkit

- Implementation of a global workspace architecture
  - Based on lessons from ICARUS
- Framework allowing the attachment of AI reasoners to a workspace
  - Not just entire systems like an ATP, but individual reasoning processes
- Toolkit of existing processes to enable rapid prototyping of new configurations

Configuring the Framework

- User needs to supply:
  - Information about the reasoning artefacts which each process will suggest for broadcast
  - Individual processes have to obey an interface
    - How they reason about a broadcast artefact
    - How they generate a new artefact for broadcast
    - How they score each new artefact
- Framework takes care of working out what/how to broadcast
  - And builds in parallelisation over a network

Example Configuration
Automated Theory Formation

- Broadcast artefacts:
  - Definitions, concept, background axioms and concepts, conjectures, explanations
- Processes:
  - AxiomProposer, MaceAxiomPopulator, Background, DefinitionCreator (PRs), DefinitionCreatorSpawner, RepeatProposer, DefinitionReviewer, ExampleFinder, EquivalenceReviewer, ImplicationMaker, EquivalenceSplitter, ConjectureForAll, ConjectureEquality, ConjectureApplicability, ConjectureReducer, TranslateOtter, Reporter, Control
- Scoring mechanisms
  - Explanation > Conjecture > Concept > Definition > Background

Other Configurations

- TM, HOMER and ICARUS
  - Different configurations, based on the nature of the task at hand
  - Wrappers to call external systems
    - Maple, Otter, Minion, CLPFD, Prover9
  - Use ATF toolkit processes

Evaluation

- Built a model of GWA which provides a generic framework for combining AI processes
  - Compared to other potential approaches
- Increased flexibility and ease of use than ad-hoc systems
  - Faster to implement and faster in execution (parallel)
  - Roughly the same quality of theories produced
- Identified extensions to GWA for engineering purposes
  - Ability of a process to remove itself from the workspace
  - Spawning of 'repeat processes' as a memory model

Conclusions

Thanks to Alison and John

- The automated theory formation odyssey
  - Provides the basis for various more sophisticated computational models of mathematical invention
  - With real applications to mathematical discovery
  - Stretching AI methods and methodologies
  - Making philosophical and cognitive theories more concrete
  - Provides fuel for thought for computational creativity issues (including philosophical aspects)

Future Work

- PhD project of Flaminia Cavallo:
  - Taking psychology results from books such as "The Big Book of Concepts" and implementing concept formation methods based on them
- Mathematics projects
  - User-driven theory exploration (Taxi)
- Artistic projects
  - Concept formation for the conception of subscene patterns in paintings
  - Automated theory formation for musical analysis
- Other applications
  - Formal software verification

Questions...?