MathCheck: A Math Assistant Combining SAT with Computer Algebra Systems

Ed Zulkoski, Vijay Ganesh, Krzysztof Czarnecki

University of Waterloo

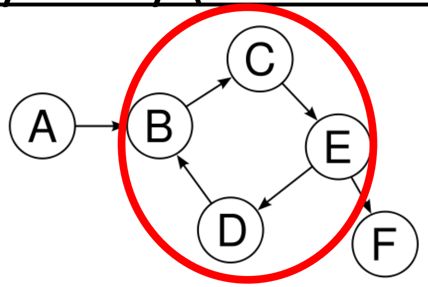
August 7, 2015

UNIVERSITY OF WATERLOO

Problem Statement

Many problems have an underlying Boolean structure, but are **not easily expressed** using standard SAT/SMT solvers.

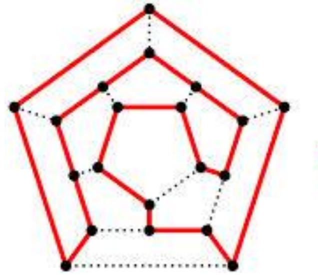
Acyclicity (Gebser'14)



Constrained Clustering (Métivier'12)

	A	B	C	D	E	F
t_1	×		×	×		
t_2		×			×	
t_3	×		×	×		
t_4		×	×	×		×
t_5		×			×	
t_6		×			×	
t_7		×		×		×

Hamiltonicity (Velev'09)

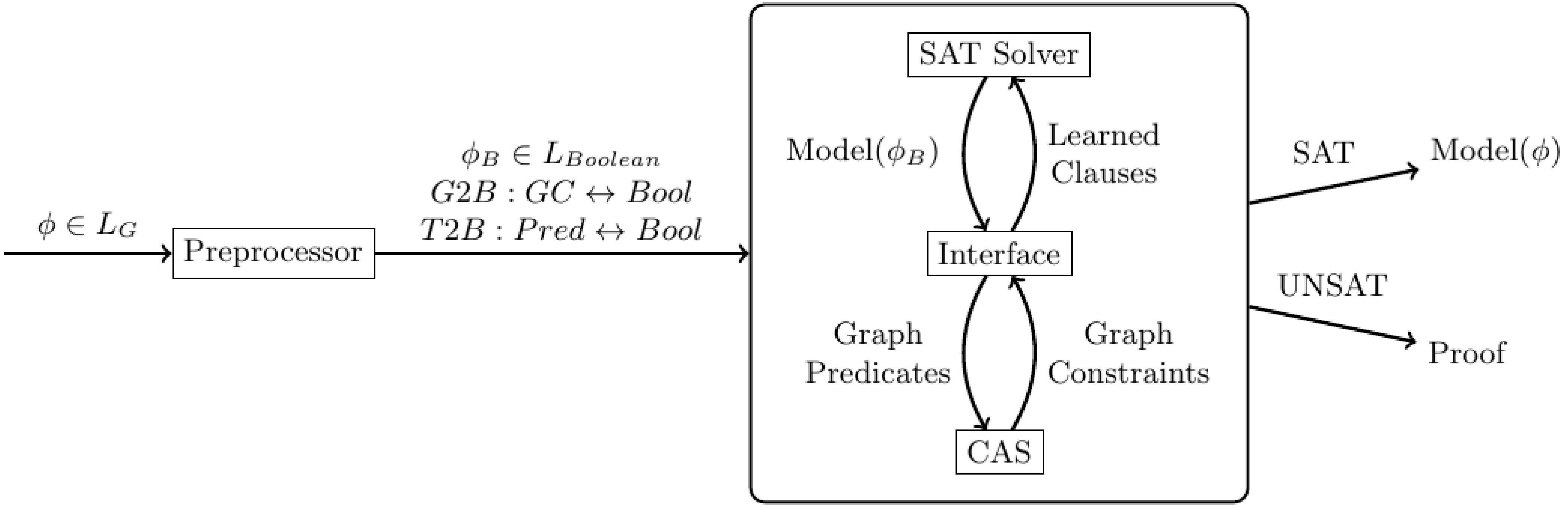


Finite domain search + complex predicates.

Goals

- Computer algebra systems (CAS) contain SOTA algorithms for solving complex properties
- SAT solvers are one of the best general approaches for finite domain search
- **Goal 1**: incorporate algorithms from a CAS with a SAT solver for:
  - Counterexample Construction for Math Conjectures
  - Bug Finding
- **Goal 2**: design an easily extensible language/API for such a system
  - Current focus is on graph theory

DPLL(CAS) Architecture

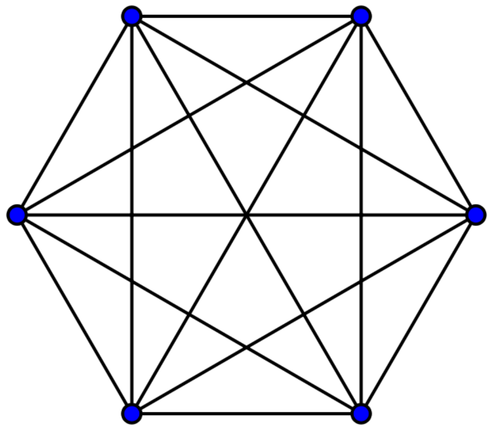


Extensibility preferred to a "one-algorithm-fits-all" approach.

Graph Variable Representation

graph x(6)

- One Boolean per each <u>potential</u> vertex
- One Boolean per each <u>potential</u> edge

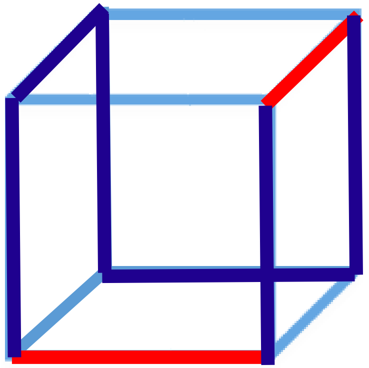


- Mapping between graph components and Booleans to facilitate defining SAT-based graph constraints

Case Study: Ruskey-Savage Conjecture

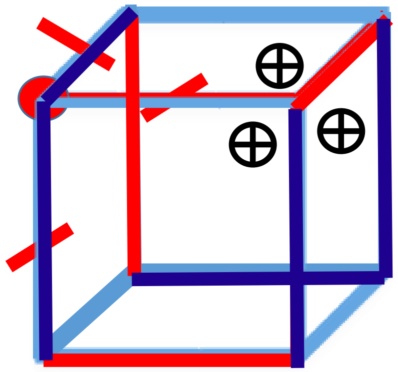
Conjecture: For every $d \geq 2$, any matching of the hypercube Q_d extends to a Hamiltonian cycle.

- Matching – independent set of edges that share no vertices
  - Maximal – cannot add edges without violating the matching property
  - Perfect – it covers all vertices
- Hamiltonian cycle – cycle that touches every vertex
- Previously shown true for $d \leq 4$



Q_3

Case Study Specification ($d = 5$)



graph x(32)

sage.CubeGraph G(5)

//$\forall x. matching(x, G) \Rightarrow extends\_to\_hamiltonian(x, G)$

assert(matching(x,G) ∧

imperfect_matching(x,G) ∧

maximal_matching(x,G)),

query(extends_to_Hamiltonian_cycle(x,G))

~10 LOC

~5 LOC

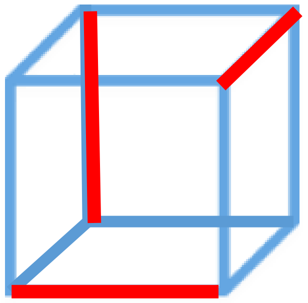
~5 LOC

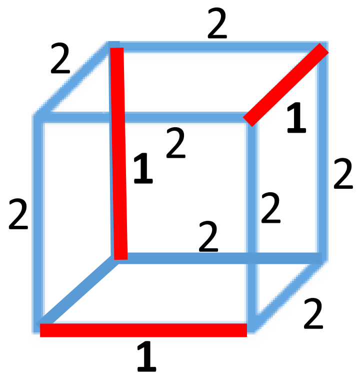
Blasted to SAT

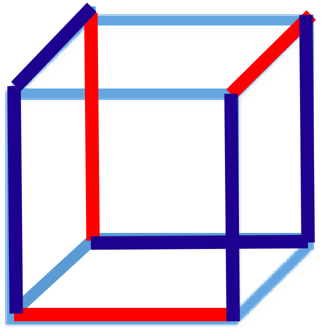
~25 LOC

Checked with SAGE

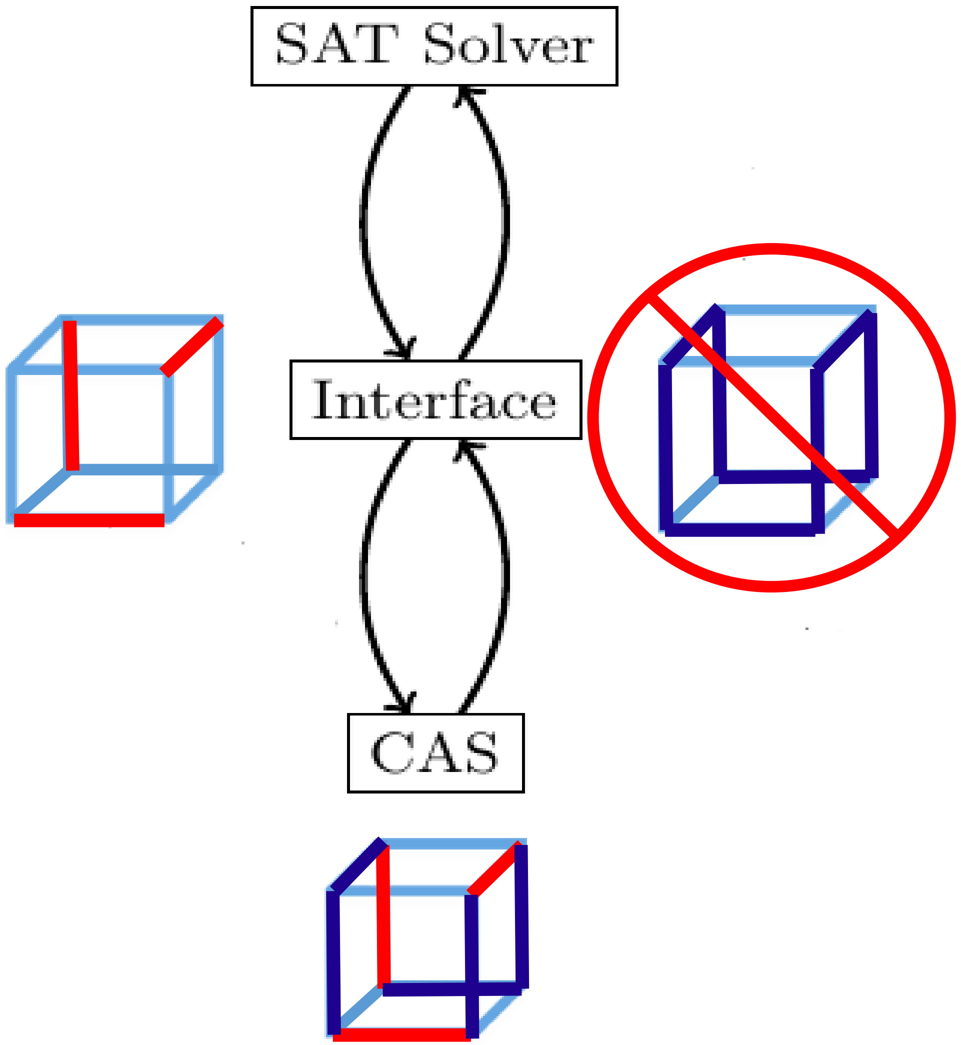
```
ExtendsToHamiltonian()
    g ← s.getGraph(G)
    q ← CubeGraph(5)
```







Case Study Approach



- Unsat after ~8 hours on laptop (Conjecture holds for $\boldsymbol{d = 5}$)
- For a pure SAT encoding, we need encode non-trivial Hamiltonicity constraints

A Sage-only approach…

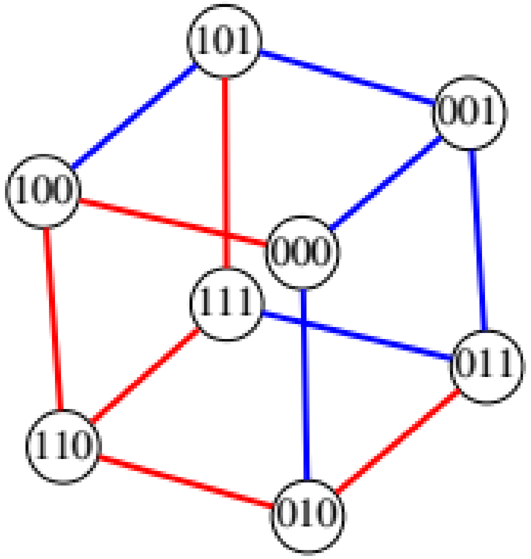
- Without SAT, we need a problem-specific search routine

	#Checks of extends_to_Hamiltonian_cycle
Matchings	13,803,794,944
Imperfect Matchings	4,619,529,024
Maximal Imperfect Matchings	6,911,604

- A Sage-only approach is:
  - Potentially less efficient
  - Potentially more error-prone

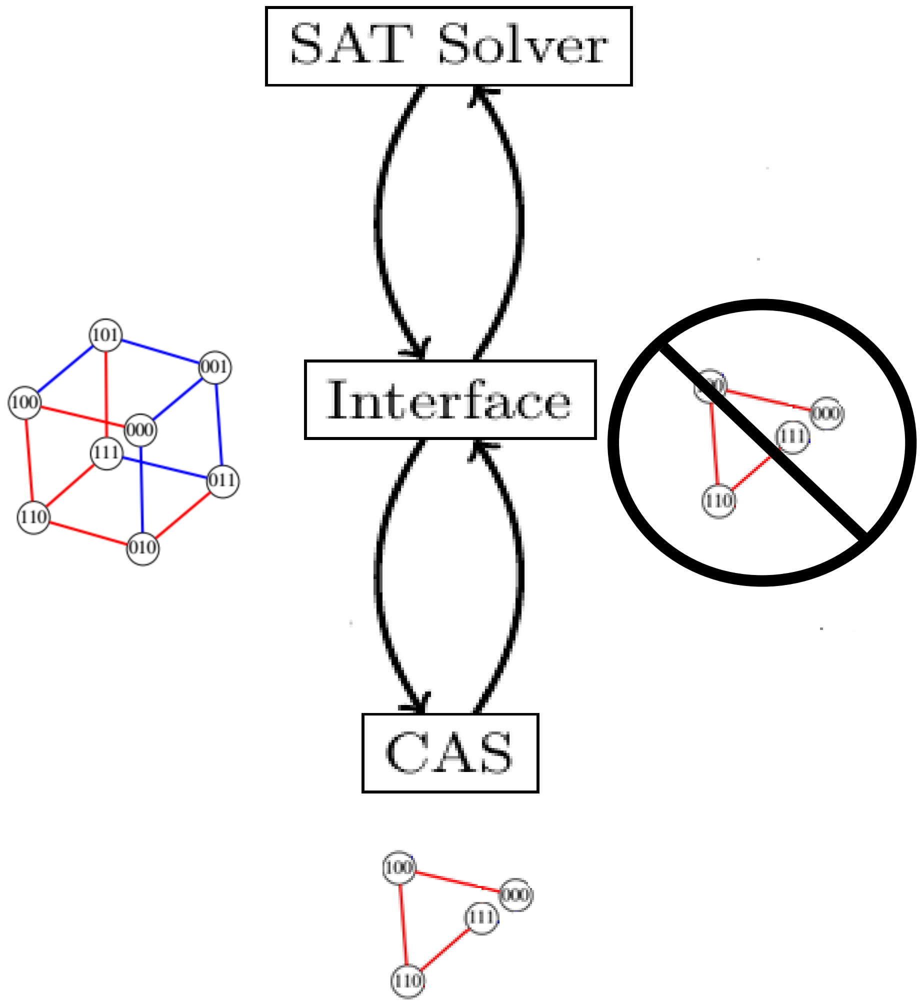
Case Study 2: Edge-antipodal colourings

- Conjecture: For every dimension $d \geq 2$, in every edge-antipodal 2-edge-coloring of Q_d, there exists a monochromatic path between two antipodal vertices.



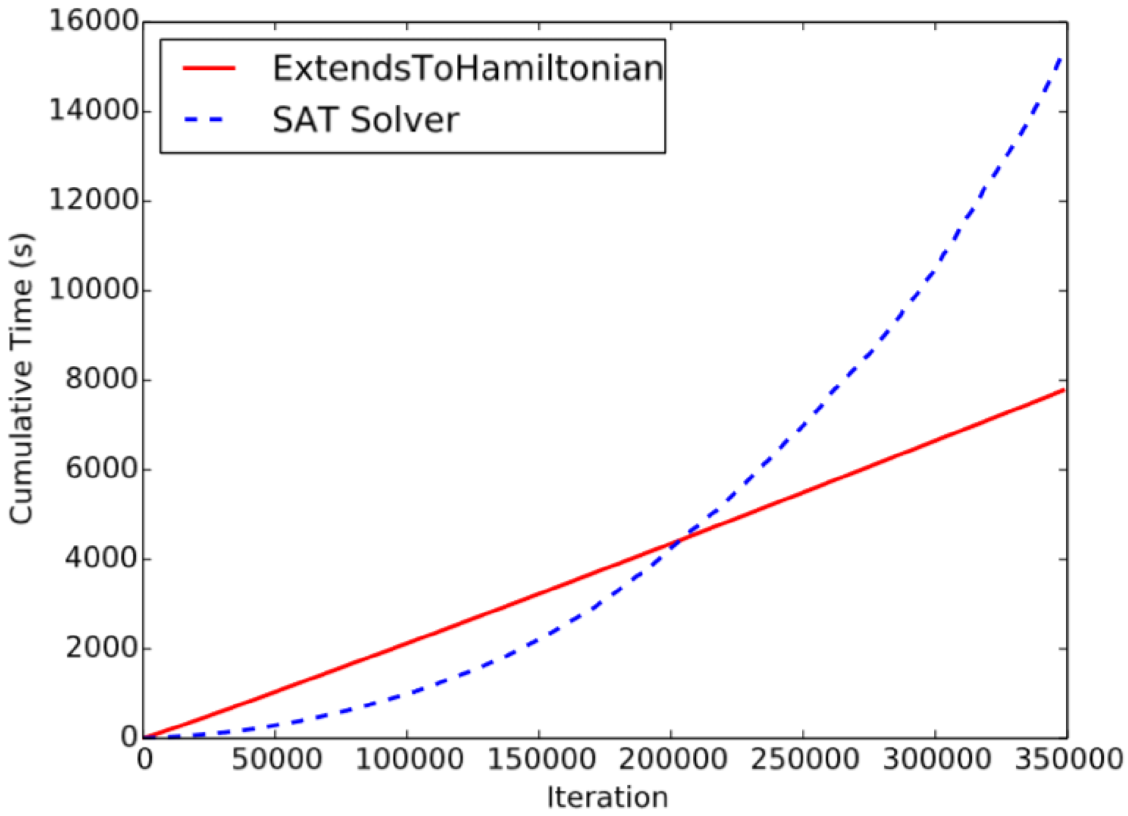
- For $d = 6$, search space of all colorings is: $2^{2^{\#edges/2}} = 2^{2^{96}}$.

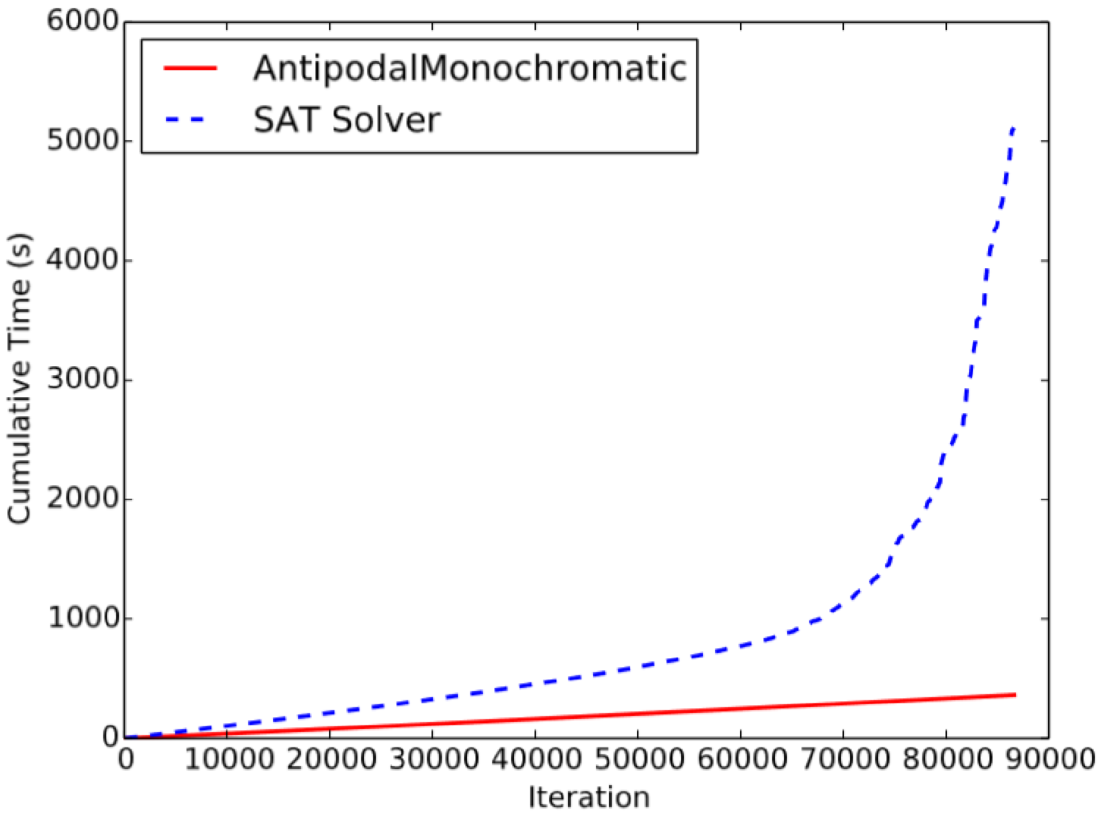
Case Study 2 Approach



- For a pure SAT encoding, we need to ensure **none** of the antipodal vertices are connected by a path
  - 32 connectivity constraints
- UNSAT after 1.5 hours ($d = 6$ holds)

What's the bottleneck?





Implementation Correctness

- SAT solver resolution proofs
  - Use Drup-trim
- SAGE computations
- Interactions between them

Future Work and Conclusions

- Moving to the SMT domain
  - Improved generation of proof objects / correctness checking
- Exploiting symmetry breaking capabilities
- Encoding complex predicates is facilitated by using off-the-shelf CAS algorithms
  - Promotes rapid extensibility/prototyping
- Demonstrated two case studies on hypercubes
  - “Fun case studies”