MathCheck2: Combining SAT and CAS

Presentation at Applications of Computer Algebra 2016 – Kassel, Germany

Curtis Bright, Vijay Ganesh, Albert Heinle*, Ilias Kotsireas, Saeed Nejati, Krzysztof Czarnecki

University of Waterloo

2016–08–02

Motivation

The research areas of SMT [SAT-Modulo-Theories] solving and symbolic computation are quite disconnected. On the one hand, SMT solving has its strength in efficient techniques for exploring Boolean structures, learning, combining solving techniques, and developing dedicated heuristics, but its current focus lies on easier theories and it makes use of symbolic computation results only in a rather naive way.

– Erica Ábrahám[1]

[1] Building bridges between symbolic computation and satisfiability checking. *Proceedings of the 2015 International Symposium on Symbolic and Algebraic Computation.*

Who You Gonna Call?

- A mathematician has a conjecture and wants to check it for non-trivial examples.
- Computer algebra systems (CAS) provide large variety of libraries to check if certain structures have desired properties.
- However, finding these non-trivial examples is often challenging due to – but not limited to – an exponentially sized search space.
- CASs are lacking search capabilities. Satisfiability (SAT) solvers, on the other hand, utilize sophisticated methods (e.g. so-called conflict driven clause learning) to search in an exponentially sized space.

*"**Brute**-brute force has no hope. But clever, inspired brute force is the future."*

– Doron Zeilberger[2]

[2]From Doron Zeilberger's talk at the Fields institute in Toronto, December 2015 (`http://www.fields.utoronto.ca/video-archive/static/2015/12/379-5401/mergedvideo.ogv`, minute 44)

When One Could Use A SAT+CAS Combination...

- **Absolutely Necessary:** Elements that are examined can be represented using a finite boolean vector.
- The SAT+CAS combination is powerful when one tries to find examples up to a certain representation size.
- **Feasibility Condition:** Operations on the respective elements to check the desired property can be encoded as a SAT problem that has size polynomial in the representation of these elements.

This excludes e.g.

- Power series
- Structures that contain real/complex numbers.

Running Example: Hadamard matrices

- square matrix with ± 1 entries
- any two distinct rows are orthogonal

Running Example: Hadamard matrices

- square matrix with ± 1 entries
- any two distinct rows are orthogonal

Example

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ -1 & 1 & 1 & -1 \end{bmatrix}$$

Conjecture

An $n \times n$ Hadamard matrix exists for any n a multiple of 4.

Difficulty Levels – Images taken from pokemon.wikia.com I

2×2:

$$\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \leftrightarrow$$

4×4:

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ -1 & 1 & 1 & -1 \end{bmatrix} \leftrightarrow$$

Difficulty Levels – Images taken from pokemon.wikia.com II

8×8:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ -1 & 1 & 1 & -1 & 1 & -1 & -1 & 1 \end{bmatrix} \leftrightarrow$$

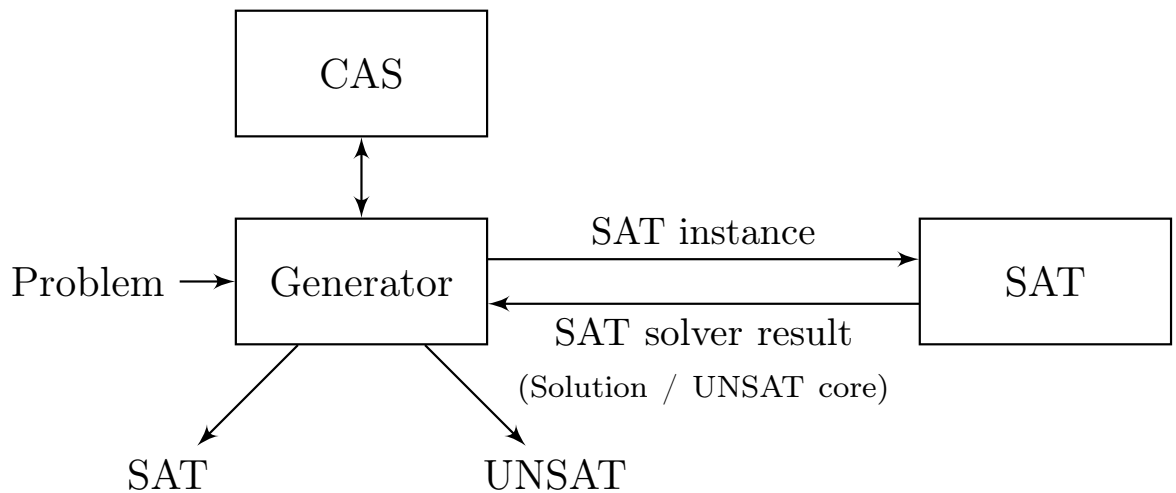
Difficulty Levels – Images taken from pokemon.wikia.com III

668×668:

MathCheck2 – Combining SAT+CAS

The MathCheck2 System

- Uses SAT and CAS functionality to finitely verify or counterexample conjectures in mathematics.
- Used to study conjectures in combinatorial design theory about the existence of Hadamard and Williamson matrices.



Experimental Results

MathCheck2 was able to show that...

- Williamson matrices of order 35 do not exist.
  - Used under 9 hours of computation time on SHARCNET[3].
  - First shown by Đoković[4], who requested an independent verification.
- Williamson matrices exist for all orders $n < 35$.
  - Even orders were mostly previously unstudied.
- Found over 500 Hadamard matrices, up to order 160, which had no equivalent entry in the database of the CAS Magma.

[3] 64-bit AMD Opteron processors running at 2.2 GHz

[4] Williamson matrices of order $4n$ for $n = 33, 35, 39$. *Discrete Mathematics*.

Some Techniques that Lead to Success

- Using special construction method
- Using CAS to partition the search space.
  - Creating a SAT instance for each partition.
  - Optimally, the CAS would be able to examine if one can exclude a whole partition from the search.
- Assist the SAT solver with a CAS for every SAT instance, using a feedback loop.
- Use a so-called UNSAT-core for all different SAT instances.

Challenges From a Usability Perspective

In decreasing order by difficulty:

- Find a way to feasibly encode the problem (including, but not limited to: objects of interests, their operations, and conditions).
- Find an easy to use interface to include custom CAS scripts.
- Study the problem well enough to be able to assist the SAT solver with CAS code to prune away branches in the search space.
- If possible, study the problem well enough to be able to partition the search space.

Challenges From a Usability Perspective

In decreasing order by difficulty:

- Find a way to feasibly encode the problem (including, but not limited to: objects of interests, their operations, and conditions).
- Find an easy to use interface to include custom CAS scripts.
- Study the problem well enough to be able to assist the SAT solver with CAS code to prune away branches in the search space.
- If possible, study the problem well enough to be able to partition the search space.

Example: Naive Hadamard Encoding

Each entry of H will be represented using a *Boolean variable* encoding with $\mathrm{BV}(1) = \text{true}$ and $\mathrm{BV}(-1) = \text{false}$.

Multiplication becomes XNOR under this encoding, i.e.,

$$\mathrm{BV}(\mathsf{x} \cdot \mathsf{y}) = \neg(\mathrm{BV}(\mathsf{x}) \oplus \mathrm{BV}(\mathsf{y})) \qquad \text{for } \mathsf{x}, \mathsf{y} \in \{\pm 1\}.$$

Example: Naive Hadamard Encoding Continued

Arithmetic formula encoding

$$\sum_{k=0}^{n-1} h_{ik} \cdot h_{jk} = 0 \qquad \text{for all } i \neq j.$$

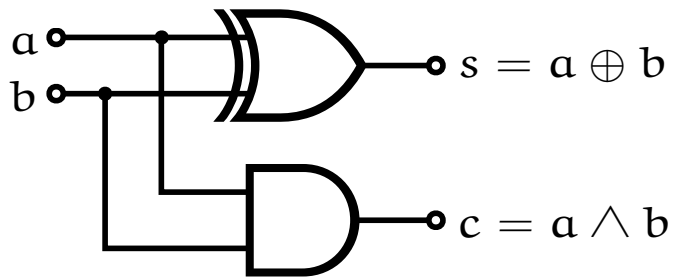
Boolean variable encoding

Using 'product' variables $p_{ijk} \coloneqq \mathrm{BV}(h_{ik} \cdot h_{jk})$ this becomes the cardinality constraints

$$\sum_{\substack{k=0 \\ p_{ijk}\ \mathrm{true}}}^{n-1} 1 = \frac{n}{2} \qquad \text{for all } i \neq j.$$

Example: Naive Hadamard Encoding Continued

A *binary adder* consumes Boolean values and produces Boolean values; when thought of as bits, the outputs contain the binary representation of how many inputs were true.



To encode the cardinality constraints we use a network of binary adders with:

- n inputs (the variables $\{p_{ijk}\}_{k=0}^{n-1}$)
- $\lfloor \log_2 n \rfloor + 1$ outputs (counting the number of input variables which are true)

Subcase: Williamson Matrices

- $n \times n$ matrices A, B, C, D
- entries ± 1
- symmetric, circulant
- $A^2 + B^2 + C^2 + D^2 = 4nI_n$

What Did We Learn From Looking at Hadamard Matrices in the Context of SAT?

- The mathematician needs to be CURRENTLY pretty desperate to be willing to encode his/her problem as SAT.
  $\Rightarrow$ Life needs to be made easier here.
- SAT is a great helper, allowing you to scale. But one cannot lean back and expect it to scale without help in form of theories.

The World Would Be a Nice Place if...

Pick one or more:

- CASs (or a separate system) would provide ways to encode certain finite structures and operations as SAT instances.
- CASs would use knowledge from the SAT-community to implement internal search capability.
- SAT-solvers would provide a clear interface with which domain-specific knowledge can be injected.

Remark: This great synergy leads to a another challenge: Verification of correctness.