Solving Pseudo-Boolean Constraints with SMT

A few first steps

Gereon Kremer

RWTH Aachen University, Germany
LuFG Theory of Hybrid Systems

March 2nd, 2018

What was already done?

- Bachelor thesis in 2017
- Pseudo-Boolean problems:
  - Linear
  - No objectives
  - Only conjunctions

What was already done?

- Bachelor thesis in 2017
- Pseudo-Boolean problems:
  - Linear
  - No objectives
  - Only conjunctions
- Different strategies in SMT-RAT
- Comparison with MiniSat+

Problem definition

Pseudo-Boolean constraint

Boolean variables x_i, integer coefficients a_i:

$$\sum_{i=1}^{n} a_i \cdot x_i \sim a_0$$

We assume $true = 1$ and $false = 0$.

Problem definition

Pseudo-Boolean constraint

Boolean variables x_i, integer coefficients a_i:

$$\sum_{i=1}^{n} a_i \cdot x_i \sim a_0$$

We assume $true = 1$ and $false = 0$.

Satisfiability

Given a set of pseudo-Boolean constraints C over variables x_i:
Find Boolean values for all x_i such that all $c \in C$ evaluate to $true$.

Standard approach

Encode in propositional logic:

- Bitvector-style encoding of arithmetic
- Size of bitvectors depends on the coefficients
- Regular SAT solver

Standard approach

Encode in propositional logic:

- Bitvector-style encoding of arithmetic
- Size of bitvectors depends on the coefficients
- Regular SAT solver

Properties:

- Encoding grows with the coefficients
- Efficient for small constraints
- Large arithmetic constraints can be a problem

Our SMT-based approach

- Encode easy constraints in propositional logic

$$
\begin{aligned}
ax_1 \geq b, a > b > 0 \quad &\Rightarrow \quad x_1 \\
x_1 - x_2 \geq 0 \quad &\Rightarrow \quad x_2 \to x_1 \\
\sum a_i x_i \geq b, \sum a_i = b \quad &\Rightarrow \quad \bigwedge x_i \\
\sum x_i \sim b \quad & \quad \text{cardinality constraints}
\end{aligned}
$$

Our SMT-based approach

- Encode easy constraints in propositional logic

$$
\begin{array}{rcl}
ax_1 \geq b, a > b > 0 & \Rightarrow & x_1 \\
x_1 - x_2 \geq 0 & \Rightarrow & x_2 \rightarrow x_1 \\
\sum a_i x_i \geq b, \sum a_i = b & \Rightarrow & \bigwedge x_i \\
\sum x_i \sim b & & \text{cardinality constraints}
\end{array}
$$

- Consider remaining constraints to be linear integer constraints
- Use (any) SMT solver for linear integer arithmetic
- Simplex and Branch&Bound

Experimental results

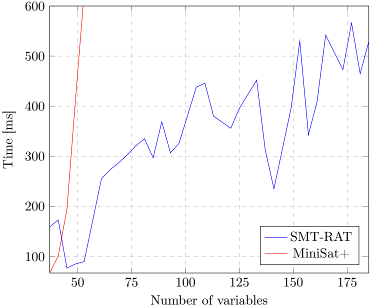
- Overall 4597 examples from PB evaluation 2015
- Ignore objective functions

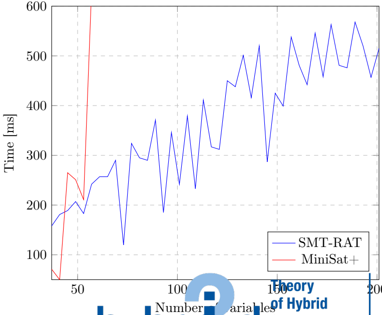
Experimental results

- Overall 4597 examples from PB evaluation 2015
- Ignore objective functions
- MiniSat$+$: $\approx 60\%$ solved
- SMT-RAT: $\approx 20\%$ solved

Experimental results

- Overall 4597 examples from PB evaluation 2015
- Ignore objective functions
- MiniSat+: $\approx 60\%$ solved
- SMT-RAT: $\approx 20\%$ solved
- But heavily depends on the structure of the benchmark





Preprocessing: Gauss

Use Gaussian elimination to simplify equations. Use equations to simplify inequations.

$$
\begin{aligned}
2x_1 + 1x_2 + 1x_3 &= 5 \\
1x_1 - 2x_2 + 1x_3 &= 4 \\
1x_1 + 1x_2 &= 2 \\
-5x_1 + 1x_3 &\geq 2 \\
4x_1 + 1x_2 + 4x_4 &\geq 1
\end{aligned}
\quad \Rightarrow \quad
\begin{aligned}
2x_1 + 1x_2 + 1x_3 &= 5 \\
-5x_1 + 1x_3 &= 2 \\
-2x_3 &= -1 \\
-1x_2 &\geq -1
\end{aligned}
$$

Preprocessing: Gauss

Use Gaussian elimination to simplify equations. Use equations to simplify inequations.

$$
\begin{aligned}
2x_1 + 1x_2 + 1x_3 &= 5 \\
1x_1 - 2x_2 + 1x_3 &= 4 \\
1x_1 + 1x_2 &= 2 \\
-5x_1 + 1x_3 &\geq 2 \\
4x_1 + 1x_2 + 4x_4 &\geq 1
\end{aligned}
\quad \Rightarrow \quad
\begin{aligned}
2x_1 + 1x_2 + 1x_3 &= 5 \\
-5x_1 + 1x_3 &= 2 \\
-2x_3 &= -1 \\
-1x_2 &\geq -1
\end{aligned}
$$

Less constraints, eliminate variables from individual constraints.

Preprocessing: Gauss

Use Gaussian elimination to simplify equations. Use equations to simplify inequations.

$$
\begin{aligned}
2x_1 + 1x_2 + 1x_3 &= 5 \\
1x_1 - 2x_2 + 1x_3 &= 4 \\
1x_1 + 1x_2 &= 2 \\
-5x_1 + 1x_3 &\geq 2 \\
4x_1 + 1x_2 + 4x_4 &\geq 1
\end{aligned}
\quad \Rightarrow \quad
\begin{aligned}
2x_1 + 1x_2 + 1x_3 &= 5 \\
-5x_1 + 1x_3 &= 2 \\
-2x_3 &= -1 \\
-1x_2 &\geq -1
\end{aligned}
$$

Less constraints, eliminate variables from individual constraints.

Detrimental. Our guess: input constraints are sparse but become dense.

Preprocessing: Residual Number Systems

Use an adaption of the Chinese Remainder Theorem to convert one large constraint to several easy constraints.

$$748x_1 + 936x_2 + 58x_3 + 493x_4 + 145x_5 + 85 + x_6 = 105$$

Choose primes in a clever way: $5, 17, 29$

$$\begin{aligned} 3x_1 + 3x_3 + 3x_4 &= 0 && \mod 5 \\ 7x_3 + 9x_5 &= 3 && \mod 17 \\ 23x_1 + 7x_2 + 27x_6 &= 18 && \mod 29 \end{aligned}$$

Preprocessing: Residual Number Systems

Use an adaption of the Chinese Remainder Theorem to convert one large constraint to several easy constraints.

$$748x_1 + 936x_2 + 58x_3 + 493x_4 + 145x_5 + 85 + x_6 = 105$$

Choose primes in a clever way: $5, 17, 29$

$$\begin{aligned} 3x_1 + 3x_3 + 3x_4 &= 0 && \mod 5 \\ 7x_3 + 9x_5 &= 3 && \mod 17 \\ 23x_1 + 7x_2 + 27x_6 &= 18 && \mod 29 \end{aligned}$$

Less terms per equation, smaller coefficients, encoding modulo is easy.

Preprocessing: Residual Number Systems

Use an adaption of the Chinese Remainder Theorem to convert one large constraint to several easy constraints.

$$748x_1 + 936x_2 + 58x_3 + 493x_4 + 145x_5 + 85 + x_6 = 105$$

Choose primes in a clever way: $5, 17, 29$

$$\begin{aligned} 3x_1 + 3x_3 + 3x_4 &= 0 && \mod 5 \\ 7x_3 + 9x_5 &= 3 && \mod 17 \\ 23x_1 + 7x_2 + 27x_6 &= 18 && \mod 29 \end{aligned}$$

Less terms per equation, smaller coefficients, encoding modulo is easy.

No effect. Our guess: Simplex performance depenends on total number of terms, smaller coefficients do not matter.

Future work

More Boolean encodings

Good Boolean encoding is usually better than arithmetic encoding
Identify good Boolean encodings for constraints from your problem

Future work

More Boolean encodings

Good Boolean encoding is usually better than arithmetic encoding
Identify good Boolean encodings for constraints from your problem

Optimization

Optimal solution with respect to an objective function
Growing support among SMT solvers (CVC4, SMT-RAT, z3)

Future work

More Boolean encodings

Good Boolean encoding is usually better than arithmetic encoding
Identify good Boolean encodings for constraints from your problem

Optimization

Optimal solution with respect to an objective function
Growing support among SMT solvers (CVC4, SMT-RAT, z3)

Nonlinear problems

Polynomial pseudo-Boolean constraints
Bitvector-based (CVC4, SMT-RAT, z3) or B&B-based (SMT-RAT, yices)

Future work

More Boolean encodings

Good Boolean encoding is usually better than arithmetic encoding
Identify good Boolean encodings for constraints from your problem

Optimization

Optimal solution with respect to an objective function
Growing support among SMT solvers (CVC4, SMT-RAT, z3)

Nonlinear problems

Polynomial pseudo-Boolean constraints
Bitvector-based (CVC4, SMT-RAT, z3) or B&B-based (SMT-RAT, yices)

Boolean combinations

Arbitrary Boolean combinations instead of pure conjunctions