SMT Unsat Core Minimization

OFER GUTHMANN,

OFER STRICHMAN, ANNA TROSTANETSKI

FMCAD2016

Technion
Israel Institute of Technology

Satisfiability Modulo Theories

Satisfiability Modulo Theories (SMT): decides satisfiability of formulas over first order theories, by combining

- a SAT solver, and
- decision procedures for conjunctions of first order literals.

SMT solvers use Boolean Abstraction

Let φ be an SMT formula

φ's Boolean Abstraction, $e(\varphi)$, assigns a Boolean variable to every theory literal in φ.

Example:

- $\varphi = \big(\overset{e_1}{(x = 0)}\big) \wedge \big(\overset{e_2}{(x = 1)} \vee \neg\overset{e_3}{(x = 2)}\big)$
- $e(\varphi) = (e_1) \wedge (e_2 \vee \neg e_3)$
- Boolean structure unchanged.

Decoding: $d(e_1) \coloneqq (x = 0)$, $d(e_2) \coloneqq (x = 1)$, etc.

The Minimal Unsat Core Problem (MUC)

Let φ be an unsat SMT formula (in CNF).

Find a minimal (i.e., irreducible) unsat core of φ's clauses.

$\varphi = a \wedge (\neg a \vee b) \wedge (\neg a \vee \neg b) \wedge (b \vee c)$

$C = \{a, (\neg a \vee b), (\neg a \vee \neg b)\}$

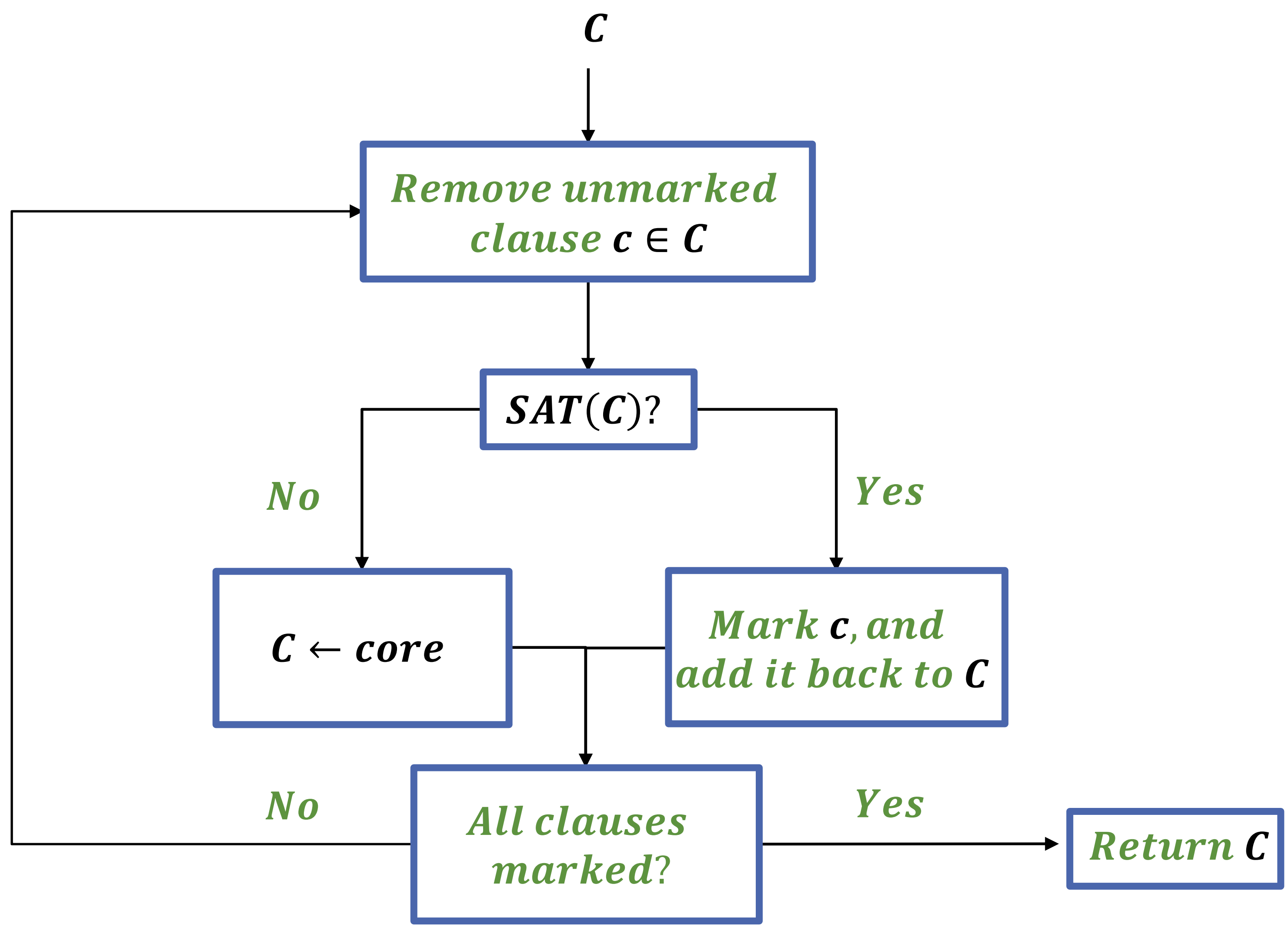
C is a minimal unsat core.

Many applications may benefit from finding a MUC:

- Abstraction refinement.
- Formal equivalence verification.
- Decision procedures.
- Etc.

We know of no SMT MUC extractors in the public domain

Deletion-based MUC Extraction (propositional case)



Z3 and Cores

Z3 is an open-source competitive SMT solver:

- Developed by Microsoft Research.
- Emits an unsat core (set of clauses used in proof).
- Uses high-level proof rules

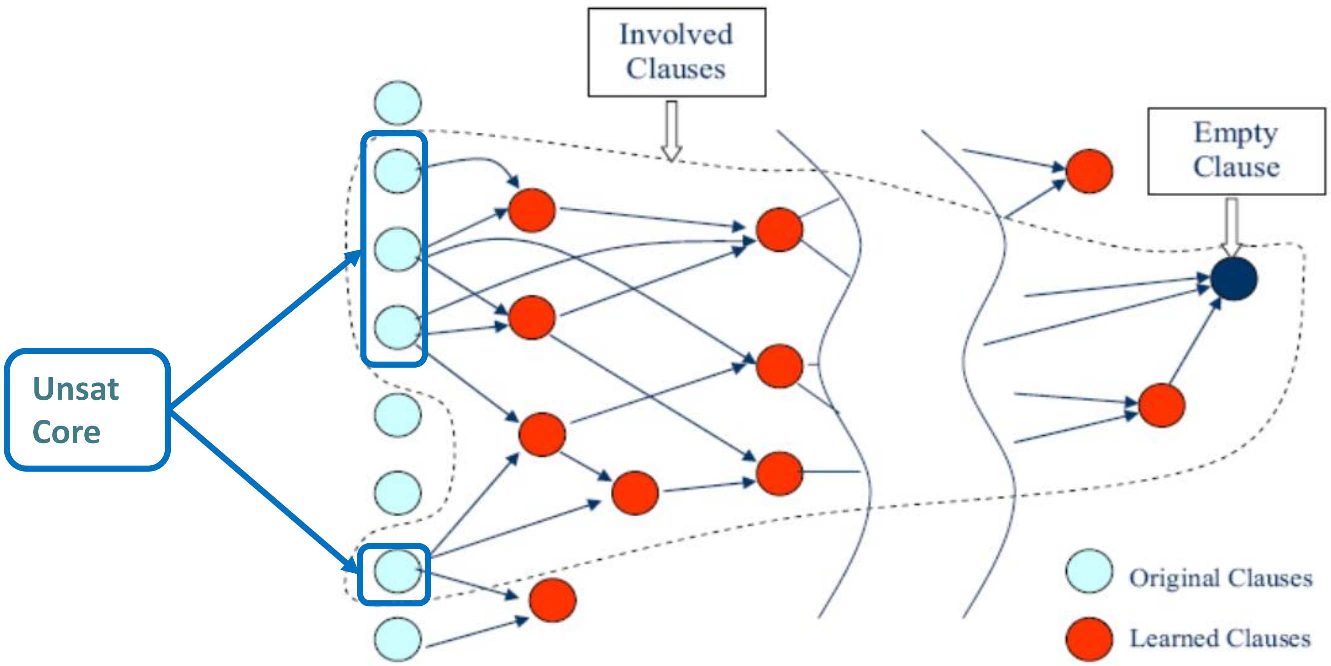
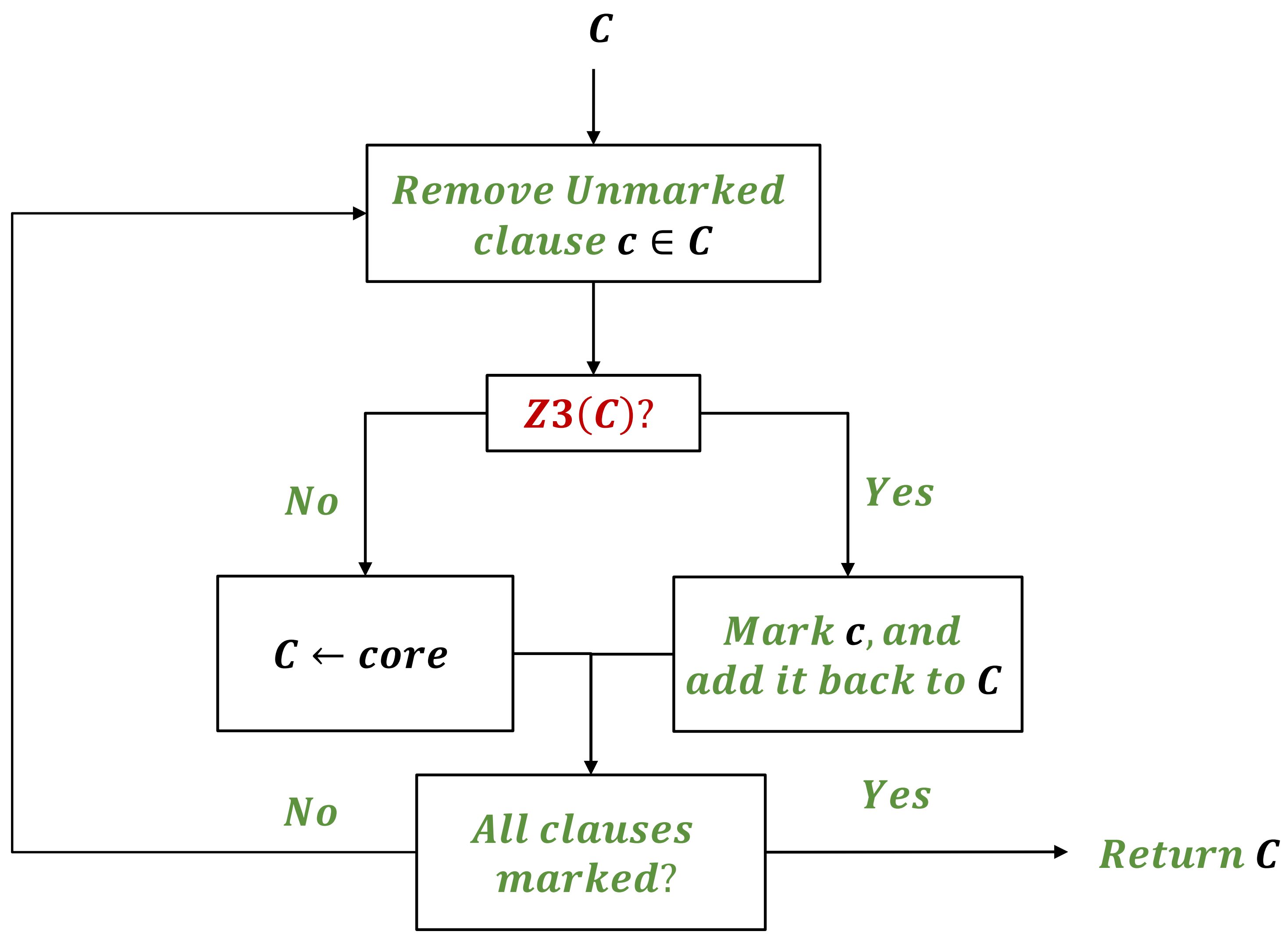


*Diagram taken from L. Zhang and S. Malik: *Validating SAT Solvers Using an Independent Resolution-Based Checker: Practical Implementations and Other Applications*. 2003.

HSmtMuc
A Deletion-based SMT MUC Extractor



Optimization: Rotation

* A. Belov and J. Marques-Silva. *Accelerating MUS extraction with recursive model rotation*. 2011.

Let c be a marked clause.

- $\varphi \setminus \{c\}$ is satisfiable.
- $\alpha \vDash \varphi \setminus \{c\}$.

Rotate(c, α)

- Find $\alpha' \neq \alpha$ and $c' \neq c$, s.t. $\alpha' \vDash \varphi \setminus \{c'\}$
  - By flipping variables in α that appear in c.
- If such c' was found:
  - Mark c'
  - Rotate(c', α')

Now in SMT: Theory Rotation

Let c be a marked clause.

- $\varphi \setminus \{c\}$ is satisfiable
- $\alpha \vDash e(\varphi \setminus \{c\})$.

Recall: e applies boolean abstraction

Rotate(c, α)

- Find $\alpha' \neq \alpha$ and $c' \neq c$, s.t. $\alpha' \vDash e(\varphi \setminus \{c'\})$:
  - By flipping variables in α that appear in c.
- If such c' was found:
  - Mark c'
  - Rotate(c', α')

The problem: the new assignment may not be T-consistent

Theory Rotation – Contradiction Example

$$\varphi = \underbrace{\big((x = 0)\big)}_{c} \wedge \big(\neg(x = 0) \vee (x = 1)\big) \wedge \big(\neg(x = 0) \vee (x = 2)\big)$$

$$e(\varphi) = \underbrace{(e_1)}_{e(c)} \wedge (\neg e_1 \vee e_2) \wedge (\neg e_1 \vee e_3)$$

For a model\interpretation where $x \longmapsto 1$ we have:

$$\alpha \coloneqq \big\{\{e_1, e_3\} \longmapsto F, \{e_2\} \longmapsto T\big\}$$

Theory Rotation – Contradiction Example

$$\varphi = \underbrace{\big((x=0)\big)}_{c} \land \big(\neg(x=0) \lor (x=1)\big) \land \big(\neg(x=0) \lor (x=2)\big)$$

$$e(\varphi) = \underbrace{(e_1)}_{e(c)} \land (\neg e_1 \lor e_2) \land (\neg e_1 \lor e_3)$$

For a model\interpretation where $x \longmapsto 1$ we have:

$$\alpha \coloneqq \big\{\{e_1, e_3\} \longmapsto F, \{e_2\} \longmapsto T\big\}$$

$$\alpha \vDash e(\varphi \setminus \{c\})$$

Flipping e_1 in α results in a T−contradiction.

- both $\mathrm{e}_1 \rightarrow (x = 0)$ and $\mathrm{e}_2 \rightarrow (x = 1)$ now hold.

Theory Rotation - Solution

After finding (c', α'), check if α' is T-consistent.

If it is T-consistent use Rotate (c', α') as before.

If it's not...

- One possibility is to give up and stop the recursion.
- Let's try and do better.

Theory Rotation – Fixing a T-Contradiction

Try and find more variables to flip in α'.

Variables to flip: choose from $core(\alpha')$.

- If resulting α'' still contradictory, recursively flip more vars.
- Recursion depth is determined heuristically.

$\alpha'' \vDash \varphi \setminus \{c''\}$ and is T-consistent $\Rightarrow$

- mark c'', and
- Rotate (c'', α'').

Adaptive Activation of Theory Rotation

Failed Theory Rotation can be costly.

Determine at runtime whether rotations is be continued:

First option:

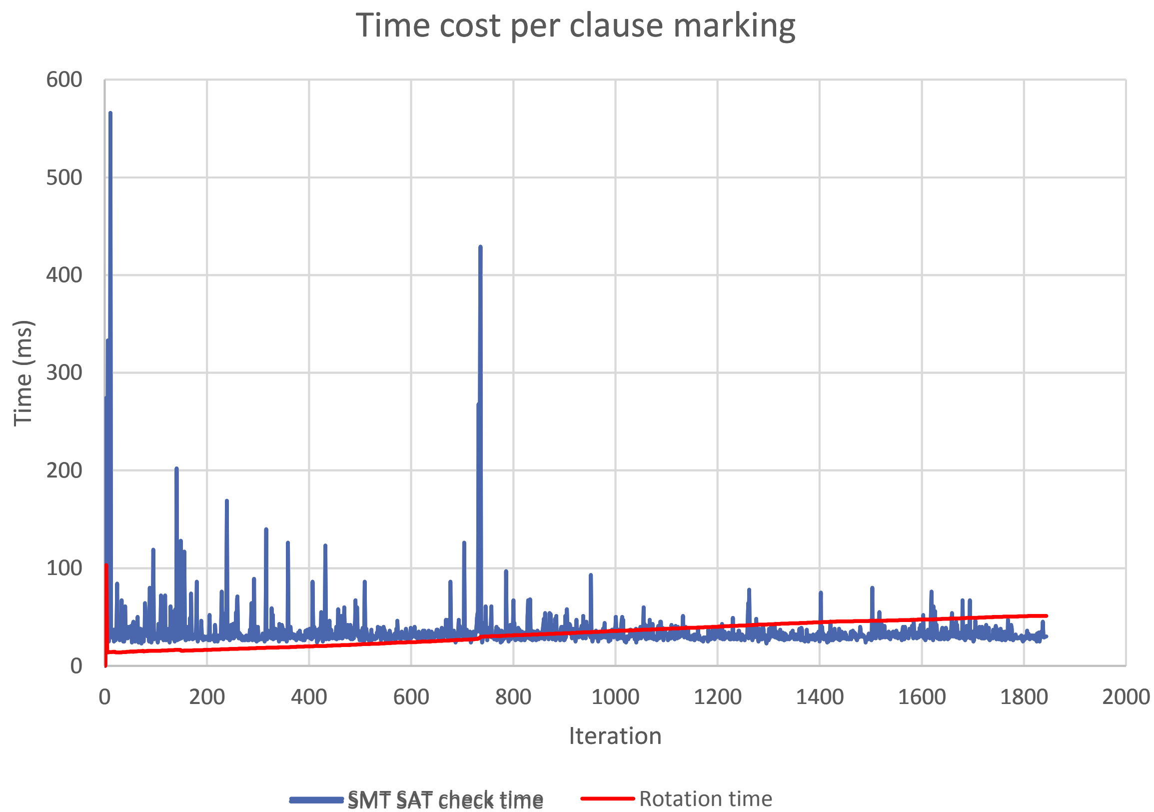
- Fail Bound: stop after x consecutive failures.
  - Failure: no clauses were marked.

Observation: Rotation success-rate declines through time.

Adaptive Activation of Theory Rotation

Another option

- Dynamic Measurement: estimate $t_{smt} < \frac{t_r}{n_r}$ to stop rotation.
  - Problem: measurement is non-monotonic.



Adaptive Activation of Theory Rotation

Exponential smoothing: Given a stream of measurements $\left\{\left(t_{smt}^i, t_{rot}^i, n_{rot}^i\right)\right\}_{i=1}^n$ define:

$$\begin{cases} T_{smt}^0 = t_{smt}^0 \\ T_{smt}^i = \alpha \cdot t_{smt}^i + (1-\alpha) \cdot T_{smt}^{i-1}, \qquad 0 \le \alpha \le 1 \end{cases}$$

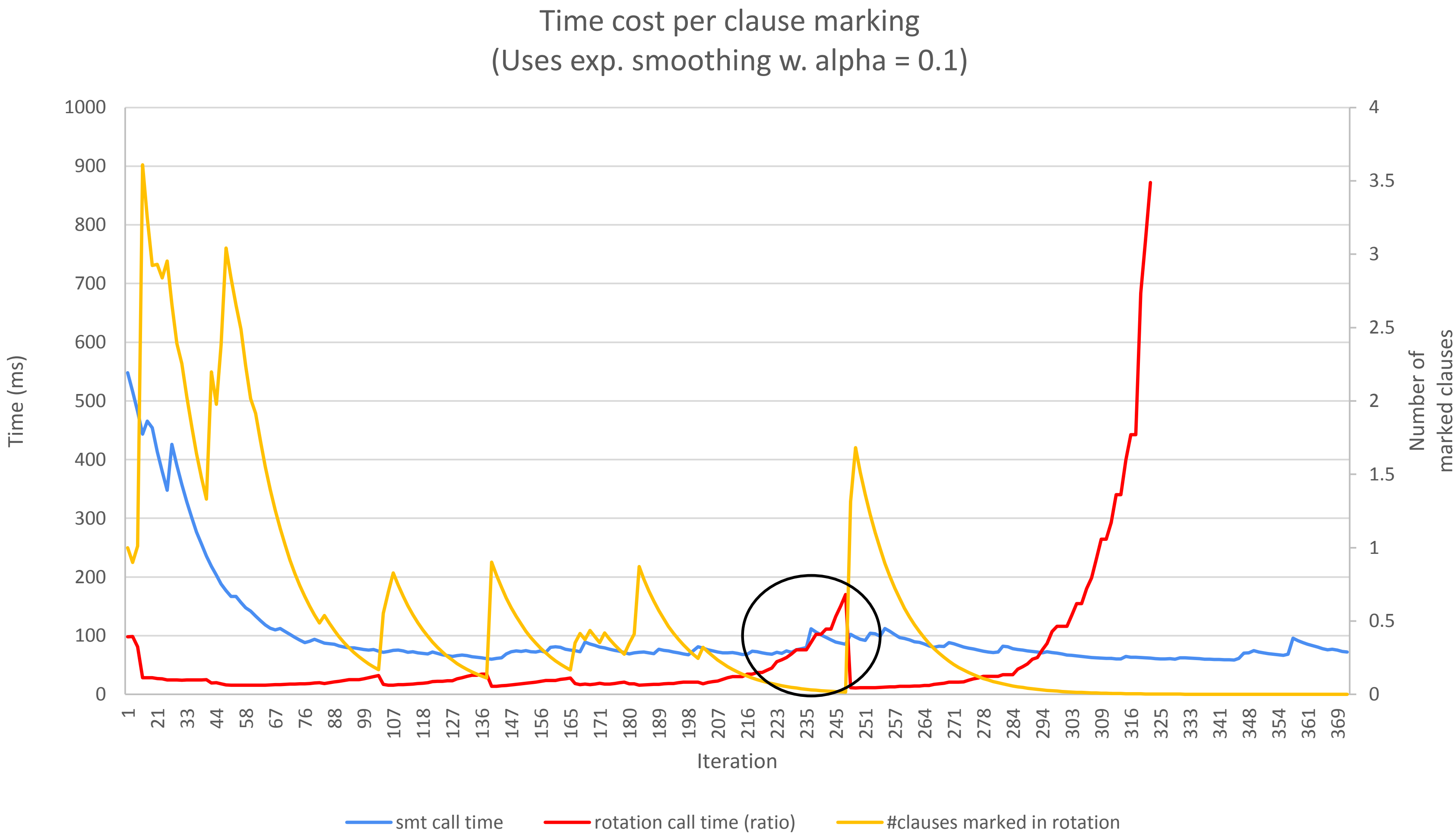
- Do the same for T_{rot}^i and N_{rot}^i

Stop rotation when $T_{smt}^i < \frac{T_{rot}^i}{N_{rot}^i}$ holds.

α chosen heuristically.

Adaptive Activation of Theory Rotation

Back to the example, now with exponential smoothing:

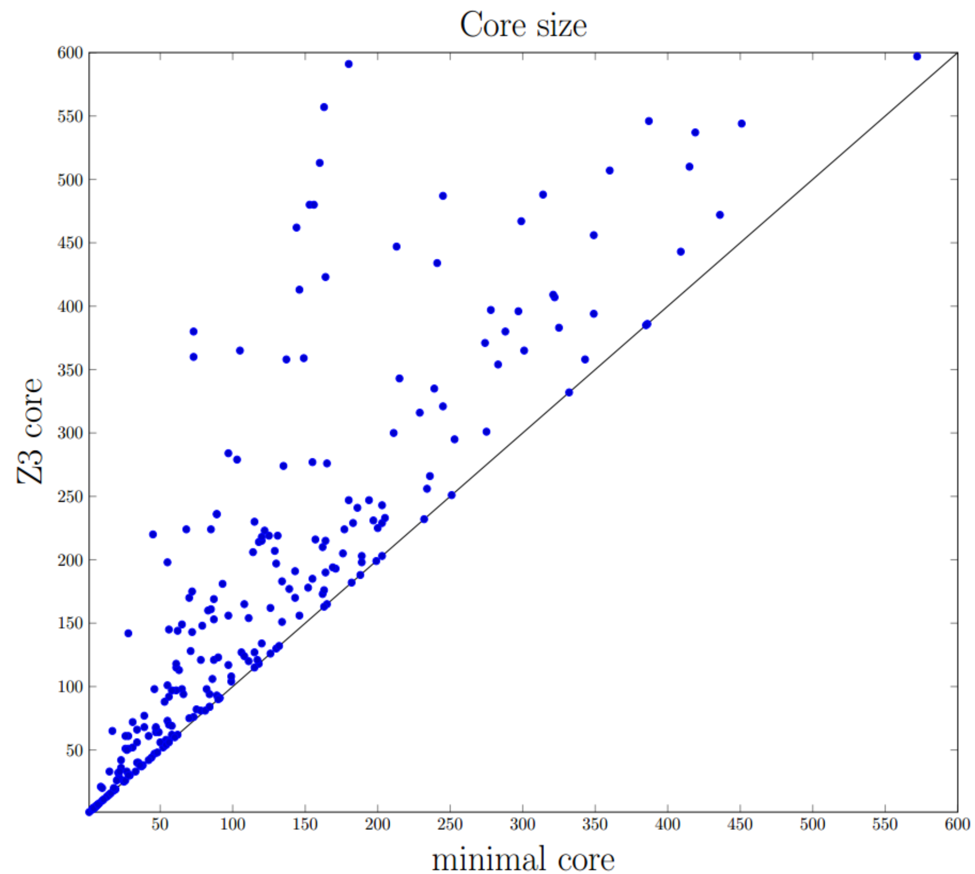


Experimental Results – Avg. core size reduction

561 unsat SMT-LIB instances*

Avg. core size:

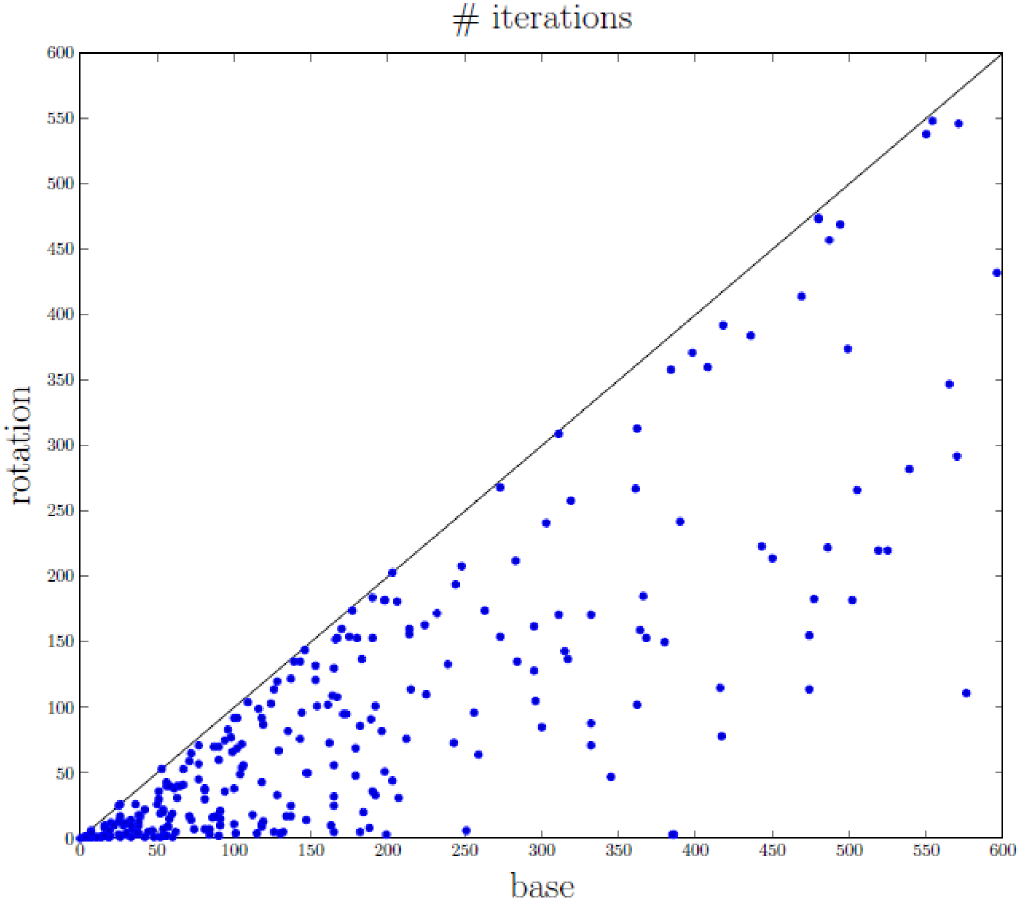
- Z3:   820 clauses.
- Min:454 clauses.



*Same instances seleScted in A. Cimatti, A. Griggio, and R. Sebastiani: *Computing small unsatisfiable cores in satisfiability modulo theories*. 2011.

Experimental Results – Theory Rotation

Reduces the number of (deletion) iterations.



Experimental Results – Theory Rotation

Translates to a modest run-time improvement (~6%-10%)

$Config.$	Time (sec.)	T-check Time (sec.)	T-Conflicts Resolved
(base)	30.5	0.0	0.0
T-Rotate	29.7	1.4	20.8
T-Rotate b 5	28.9	1.0	10.2
T-Rotate b 7	29.2	1.2	12.3
T-Rotate exp	29.6	1.2	11.2

Can be attributed to time spent on failed rotations, T-contradiction checks and additional var. flipping.

Best configuration is for Theory Rotation w. fail bound = 5

And now… Small Unsatisfiable Core (SUC)

[1] suggested an algorithm that finds a small (not necessarily minimal) SMT core

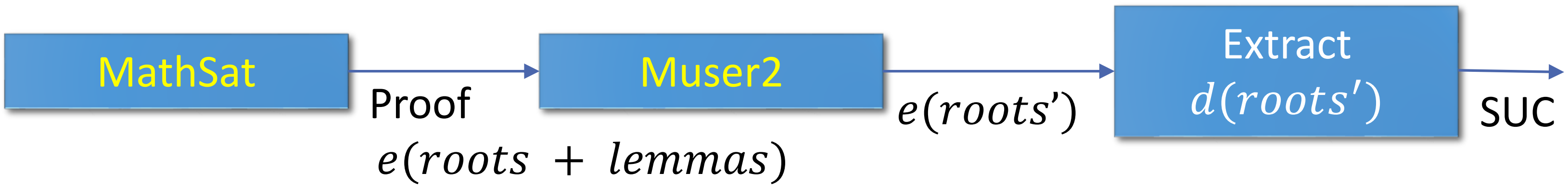
- Based on MathSat and the propos. MUC extractor Muser2

We re-implemented [1] based on Z3 + HaifaMuc

We also tested a hybrid approach in which we find a small core and then minimize it with HSmtMuc

[1] A. Cimatti, A. Griggio, and R. Sebastiani. *Computing small unsatisfiable cores in satisfiability modulo theories* (2011).

Small Unsatisfiable Core (SUC)



Our re-implementation with Z3 and HaifaMUC:

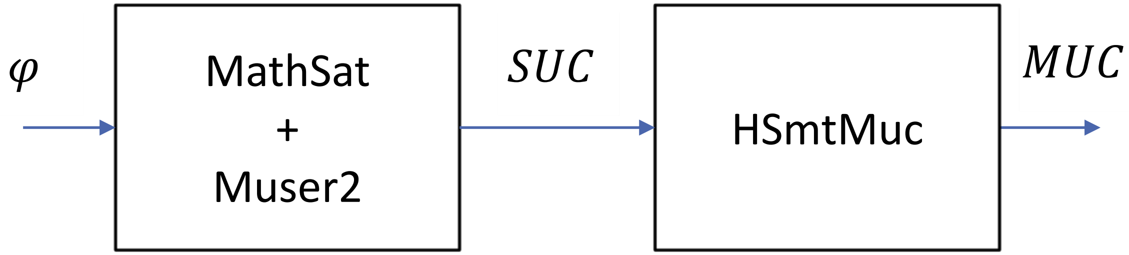
- Requires proof logging (slows Z3 a lot).
- Requires a propositional encoding of Z3's proof objects.
- Produces much larger proofs on avg. comparing to MathSat.

- Turned-out to be slower

We also tried a hybrid approach

MathSat-based SUC + minimization with HSmtMuc.

- Result is minimal.



The overall winner.

Less time-outs (HSmtMuc alone: 171 vs. Hybrid: 138).

- (but higher runtime than HSmtMuc on instances that completed, HSmtMuc: 22.9 sec. vs. Hybrid: 27.9 sec.).

Summary

HSmtMuc is the first SMT-MUC extractor in the public domain.

- Based on Z3.

Best observed results:

MUC: the Hybrid algorithm

- MathSat SUC extraction, followed by HSmtMuc.

SUC:

- MathSat SUC extraction.

More information & our implementation is available at http://strichman.net.technion.ac.il/

Questions?

Thank you!

- ofers@ie.technion.ac.il
- ofer.guthmann@cs.technion.aci.il
- annat@cs.technion.ac.il