The Effect of Clause Elimination on SLS for SAT

Oliver Gableske[1]

[1]oliver.gableske@uni-ulm.de
Institute of Theoretical Computer Science
Ulm University
Germany

Pragmatics of SAT, 16.06.2012

Outline

Outline

Overview

- Work on stochastic local search (SLS) SAT Solvers
- Improving performance on structured formulas (CRAFTED)
- Several approaches possible:
  1. Develop new algorithm's heuristics, or
  2. Tune heuristic's parameters, or
  3. Alter the formulas to improve SLS performance (Preprocessing), or
  4. ...
- We stick to preprocessing!

Terminology

For this work, we understand

- basic concepts for altering propositional formulas in a satisfiability preserving way as *preprocessing techniques* (i.e. UP)
- the combination of preprocessing techniques run in a specific order until fixpoint of each technique as *preprocessors* (i.e. `SatELite`).

Solution Ratio

Let F be a CNF, $\mathcal{S}_F = \{\alpha | \alpha(F) = 1\}$,
$\mathcal{V}_F = \{v_1, \ldots, v_n | \exists c \in F : v_i \in c \vee \neg v_i \in c\}$.

Definition (Solution Ratio)

$\widehat{\mathcal{S}}_F := \frac{|\mathcal{S}_F|}{2^{|\mathcal{V}_F|}} \quad [0, 1] \subset \mathbb{R}$.

Assumption

Assumption

The more satisfying assignments a CNF formula has in relation to its search space size, the better the performance of an SLS solver should be in order to find such an assignment.

The task then is:

- Use a preprocessor such that $\widehat{\mathcal{S}}_F < \widehat{\mathcal{S}}_{\text{PREP}(F)}$.

Outline

Preprocessing Techniques

Preprocessing techniques are:

1. Unit Propagation (UP)
2. Failed Literal Detection (FLD)
3. Subsumption Elimination (SE)
4. Pure Literal Elimination (PLE)
5. Non-increasing variable elimination by Resolution (NiVER)
6. Gate Extraction (GE)
7. Asymmetric Tautology Elimination (ATE)
8. (Asymmetric) Blocked and Covered Clause Elimination (A/BCE, A/CCE)
9. ...

- We focus on technique 8 in this work
- We explain ABCE exemplarily

ABCE (1) – Asymmetric Literal Addition (ALA)

ALA is used to add literals to a clause while preserving logical equivalence.

Assume clause $(v_1 \vee v_2 \vee \neg v_3) = c \in F$, computing $c' = ALA(c)$ means:

- Perform UP, such that all $l_i \in c$ become false; in the example $\alpha = \{v_1 \mapsto 0, v_2 \mapsto 0, v_3 \mapsto 1\}$ (the ordering *does* matter).
- While propagating, ignore c.
- Any additionally propagated assignments that are enforced by $F \setminus c$, can be added to c with opposite sign.
  - Such an additional propagation could result from $(v_1 \vee v_2 \vee v_4) = d \in F$.
  - In this case α would be extended by $\{v_4 \mapsto 1\}$.
  - Then, $c' = c \cup \{\neg v_4\}$.
- Repeat this until UP stops
  - if UP ran into a conflict, then c is implied by $F \setminus c$. Drop c.
  - if UP did not run into a conflict, return c' as the result of $ALA(c)$.

ABCE (2) – Blocked Clause Elimination

BCE removes blocked clauses from a formula. A clause is *blocked*, if it contains a *blocking literal*.

A literal $l_i \in c$ is called blocking, if and only if all possible resolutions of c on l_i yield only tautologies.

Example, lets have $c, d_1, d_2 \in F$.

- Given $c = (v_1 \vee v_2 \vee v_3)$, as well as:
- $d_1 = (\neg v_1 \vee \neg v_2)$, and $d_2 = (\neg v_1 \vee \neg v_3)$ (being all clauses containing $\neg v_1$ in F).
- Then v_1 is blocking in c, and such c is blocked w.r.t. F.

Note

A/BCE and A/CCE can both increase $\mathcal{S}_F$ (number of satisfying assignments).

ABCE (3) – Combining ALA and BCE

The more literals a clause c contains, the higher its chances are to create only tautologies when resolving.
Therefore, the more literals a clause c contains, the higher its chances are to be blocked by one of its literals.

- Combining ALA and BCE yields ABCE, which has more impact than BCE alone.
- Instead of checking if c is blocked w.r.t. F, check if $c' = ALA(c)$ is blocked w.r.t. F.
- If so, c can be removed from F.
- In order to check if c' is blocked in F, it suffices to check the resolutions on the original literals of c' (those found in c).

Solution Ratio Increasing

Let F be a CNF.

Definition (Solution Ratio Increasing)

We call a preprocessing technique P solution ratio increasing, if and only if $\widehat{\mathcal{S}}_F \leq \widehat{\mathcal{S}}_{P(F)}$ (decreasing otherwise).

The following preprocessing techniques are solution ratio increasing: UP, FLD, PLE, SE, STR, GE, Ni/VER, ATE, A/BCE, A/CCE.

Strictly increasing, if $F \in SAT$: UP, FLD, Ni/VER, GE.

Can strictly increase, if $F \in SAT$: PLE, SE, ATE, A/BCE, A/CCE.

Preprocessors

The preprocessing techniques can be plugged together to yield preprocessors. We picked the following combinations.

- `SIMPLE`: Performs UP, then FLD, then SE, then PLE
- `SABCE`: Performs `SIMPLE`, then ABCE
- `SACCE`: Performs `SIMPLE`, then ACCE

They are all solution ratio increasing. But which one is the "strongest"?

Strength of Preprocessors

Definition (Strength)

Let F be a CNF and let $\mathtt{P}$,$\mathtt{Q}$ be preprocessors. We write $\mathtt{P} \succcurlyeq \mathtt{Q}$ ($\mathtt{P}$ is stronger than $\mathtt{Q}$), if

$$\forall F : \widehat{S}_{\mathtt{P(F)}} \geq \widehat{S}_{\mathtt{Q(F)}}$$

($\mathtt{P}$ raises the solution ratio at least as much as $\mathtt{Q}$).

It is $\mathtt{SACCE} \succcurlyeq \mathtt{SABCE} \succcurlyeq \mathtt{SIMPLE}$.

Outline

Conjecture

Conjecture

The larger the solution ratio for a given formula F, the easier it should be for SLS to find a satisfying assignment for F.

With P being a solution ratio increasing preprocessor:

- SLS performance should in general be better on $\texttt{P}(F)$ than on F
- performance improvement should increase with an increasing strength of P
- all together: SLS should perform best on $\texttt{SACCE}(F)$, second best on $\texttt{SABCE}(F)$, third best on $\texttt{SIMPLE}(F)$, worst on F.

Outline

Empirical Study – Solvers, Benchmark, Idea

SLS Solvers:

- `sparrow2011`, `sattime2011`, `EagleUP`, `adaptg2wsat2011`

Benchmark:

- 145 satisfiable CNF formulas from the SAT 2011 CRAFTED category
- *all* that were shown to be satisfiable during the competition
- Denote this set of formulas with F_o (original)
- Create additional sets of formulas from it:
  $F_s = \texttt{SIMPLE}(F_o)$, $F_b = \texttt{SABCE}(F_o)$, $F_c = \texttt{SACCE}(F_o)$,
  $F_e = \texttt{SatELite}(F_o)$

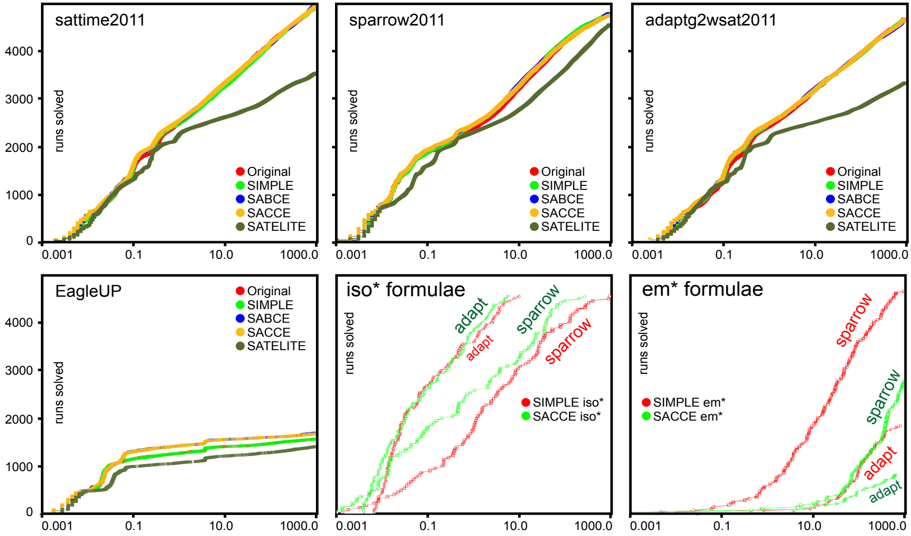
Idea:

- Check if, according to the conjecture, the runtime improves with F_o, F_s, F_b, F_c.
- Igonre the preprocessing time

Result-data:

- A lot

Empirical Study – Results (1)



Empirical Study – Results (2)

CNFs	Total **clause count** for set				
	F_o	F_s	F_b	F_c	F_e
em*	363446	344964 (5.09%)	214092 (37.94%)	214092 (0%)	363183
iso*	29253	4614 (84.23%)	3227 (30.07%)	3191 (1.12%)	3996

CNFs	Total **variable count** for set				
	F_o	F_s	F_b	F_c	F_e
em*	27610	27418 (0.7000%)	27418 (0%)	27418 (0%)	27477
iso*	2837	1370 (51.7100%)	1370 (0%)	1370 (0%)	977

Is dropping clauses without dropping variables deteriorating SLS performance?

Empirical Study – Results (3)

The paper contains more details.

You can get all the results, images, solvers etc. here:
`https://www.gableske.net/downloads/pos12.rar`

Outline

Conclusions and Future Work

Conclusions:

- The assumption that simply increasing the solution ratio of a formula results in an improved SLS performance is wrong.
- Even if a preprocessing aiming at this is beneficial for some formulas, one must not try to exploit this approach for *all* formulas.
- The general approach of speeding-up SLS by CE preprocessing has been rendered pretty worthless on the broad scale of the satisfiable CRAFTED formulae of the SAT 2011 Competition.

Future Work could deal with these questions:

- Why the runtime difference between `SACCE` and `SatELite` (runtime on formulas preprocessed with `SatELite` is much worse)? Is a specific technique (NiVER, STR, GE) responsible?
- When is CE based preprocessing helpful? Are there rules to detect this? An anonymous reviewer (nr. 3) gave the hint that it might only be useful if CE is also able to drop variables, not just clauses.
- Is *clause addition* able to improve the SLS performance?

Thank you

Thank you for your attention.

Details and References

For further details and references, check the paper
Oliver Gableske: "The Effect of Clause Elimination on SLS for SAT"
https://www.gableske.net/downloads/pos12.pdf

More Data (1)

Set	Solver	# succ. runs	% all runs	median cost	PAR10
F_o	sattime2011	4916	67.81	23.41	3264.65
	sparrow2011	4801	66.22	27.12	3413.52
	adaptg2wsat2011	4687	64.65	38.27	3579.53
	EagleUP	1567	21.61	1000.00	7843.79
F_s	sattime2011	4902	67.61 (-0.20)	27.35	3283.74
	sparrow2011	4818	66.46 (0.24)	22.44	3389.48
	adaptg2wsat2011	4671	64.43 (-0.22)	39.93	3601.50
	EagleUP	1573	21.70 (0.09)	1000.00	7836.15
F_b	sattime2011	4970	68.55 (0.74)	23.20	3195.64
	sparrow2011	4818	66.45 (0.23)	21.34	3392.29
	adaptg2wsat2011	4651	64.51 (-0.14)	38.16	3629.19
	EagleUP	1695	23.38 (1.77)	1000.00	7668.19
F_c	sattime2011	4935	68.07 (0.26)	23.20	3239.23
	sparrow2011	4773	65.83 (-0.39)	24.06	3450.61
	adaptg2wsat2011	4659	64.26 (-0.39)	39.50	3616.67
	EagleUP	1677	23.13 (1.52)	1000.00	7691.56
F_e	sattime2011	3537	48.79 (-19.02)	1000.00	5151.53
	sparrow2011	4568	63.01 (-3.21)	106.746	3754.44
	adaptg2wsat2011	3332	45.96 (-18.69)	1000.00	5434.21
	EagleUP	1411	19.46 (-2.15)	1000.00	8061.54

More Data (2)

CNFs	Total clause count for set				
	F_o	F_s	F_b	F_c	F_e
289*	172440	172440 (0%)	162840 (5.57%)	162840 (0%)	166040
cnf*	9780	9780 (0%)	9780 (0%)	9780 (0%)	9780
batt*	40713	40713 (0%)	40713 (0%)	40713 (0%)	40713
bqwh*	116566	116536 (0.03%)	116346 (0.17%)	116346 (0%)	116196
crn*	17346	17278 (0.40%)	17278 (0%)	17278 (0%)	15460
em*	363446	344964 (5.09%)	214092 (37.94%)	214092 (0%)	363183
frb*	271604	261910 (3.57%)	220002 (16.01%)	220002 (0%)	266097
Green*	176009	105123 (40.28%)	102536 (2.47%)	102504 (0.04%)	103152
inst*	712383	651511 (8.55%)	530577 (18.57%)	522577 (1.51%)	583452
iso*	29253	4614 (84.23%)	3227 (30.07%)	3191 (1.12%)	3996
lksat*	13634	13337 (2.18%)	12766 (4.29%)	12633 (1.05%)	12651
mod2*	16820	16820 (0%)	16820 (0%)	16820 (0%)	16820
rbsat*	1903455	1857153 (2.44%)	1573867 (15.26%)	1573867 (0%)	1874484
rnd*	56154	41312 (26.44%)	41122 (0.46%)	36200 (11.97%)	33491
sgen3*	5040	5040 (0%)	5040 (0%)	5040 (0%)	5040
srhd*	8826660	8826660 (0%)	7810905 (11.51%)	7810905 (0%)	8781846
Van*	209482	208488 (0.48%)	208466 (0.02%)	208466 (0%)	209482

More Data (3)

CNFs	Total variable count for set				
	F_o	F_s	F_b	F_c	F_e
289*	6400	6400 (0%)	6400 (0%)	6400 (0%)	4800
cnf*	500	500 (0%)	500 (0%)	500 (0%)	500
batt*	5013	5013 (0%)	5013 (0%)	5013 (0%)	5013
bqwh*	12513	12510 (0.03%)	12510 (0%)	12510 (0%)	11809
crn*	9242	9207 (0.38%)	9207 (0%)	9207 (0%)	4490
em*	27610	27418 (0.70%)	27418 (0%)	27418 (0%)	27477
frb*	4574	4574 (0%)	4574 (0%)	4574 (0%)	4315
Green*	3105	2668 (14.08%)	2668 (0%)	2668 (0%)	2243
inst*	55507	54230 (2.31%)	51214 (5.57%)	51214 (0%)	33633
iso*	2837	1370 (51.71%)	1370 (0%)	1370 (0%)	977
lksat*	2865	2799 (2.31%)	2771 (1.01%)	2770 (0.04%)	2461
mod2*	2369	2369 (0%)	2369 (0%)	2369 (0%)	2369
rbsat*	24955	24955 (0%)	24955 (0%)	24955 (0%)	23875
rnd*	15258	12352 (19.05%)	12328 (0.20%)	12274 (0.44%)	5432
sgen3*	2100	2100 (0%)	2100 (0%)	2100 (0%)	2100
srhd*	44814	44814 (0%)	44814 (0%)	44814 (0%)	43774
Van*	1645	1641 (0.25%)	1641 (0%)	1641 (0%)	1645