Verifying Propositional Unsatisfiability: Pitfalls to Avoid

Allen Van Gelder
Computer Science Department
University of California
Santa Cruz, CA, USA

avg@cs.ucsc.edu
http://www.cse.ucsc.edu/∼avg/Papers/

The Problem: Extract a Resolution Derivation from a Conflict Graph

Conflict Graph: Basic Data Structure in Many Modern SAT Solvers

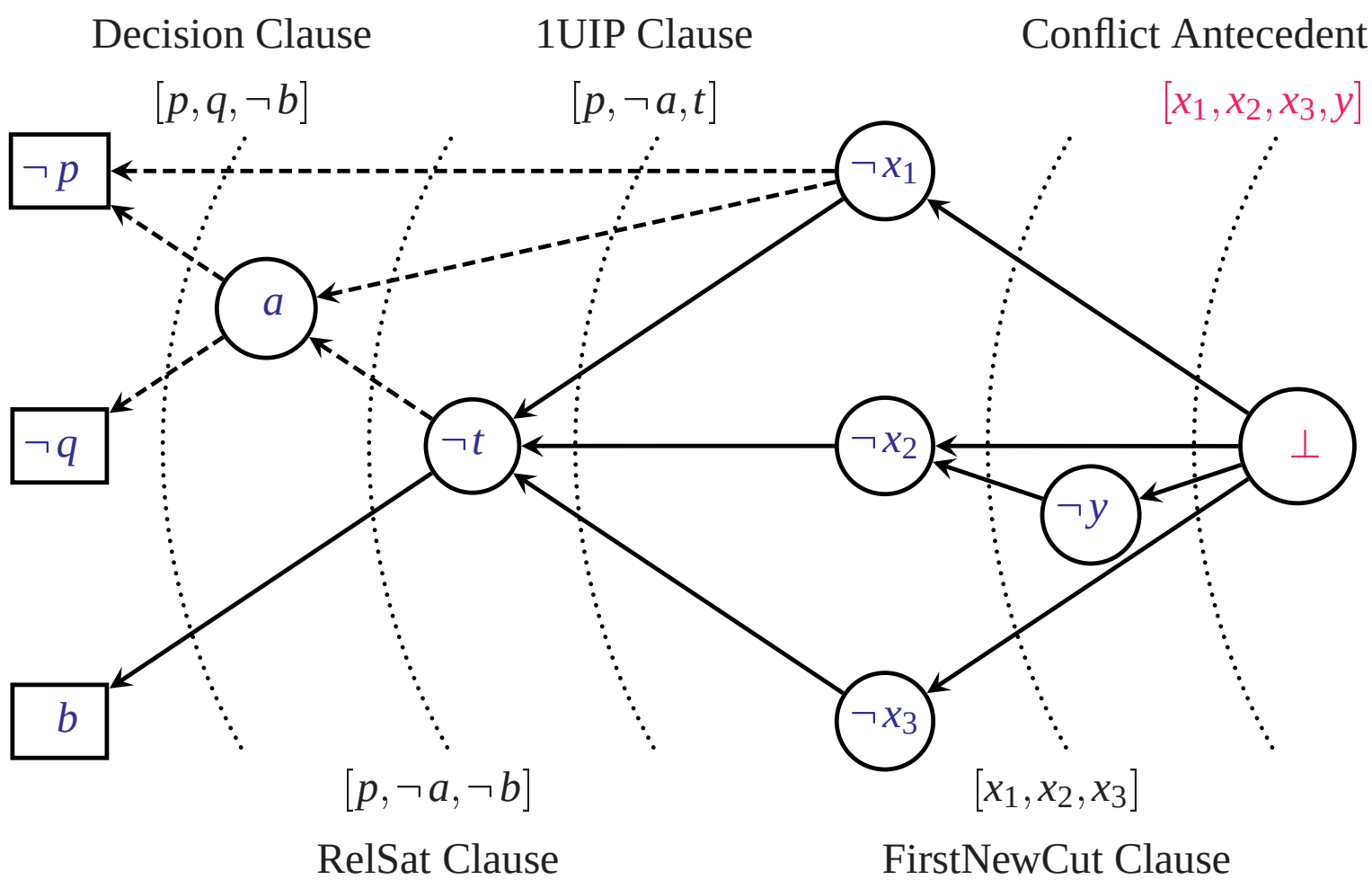
- Introduced in current form in GRASP by Marques-Silva and Sakallah [96,99]
- Adopted into Chaff by Moskewicz, Madigan, Zhao, L. Zhang, and Malik [01]
- Variations in zChaff by L. Zhang, Madigan, Moskewicz, and Malik [01]

If x is implied by unit-clause propagation, there is some clause $[x, \neg y_1, \neg y_2, \ldots]$ such that $y_1, y_2, \ldots$ were assumed or implied earlier.

Put arrows from vertex x to vertices $y_1, y_2, \ldots$.

$[x, \neg y_1, \neg y_2, \ldots]$ is called the *antecedent* of x.

Various Cuts in Conflict Graph Yield Different Conflict Clauses



$\bot$ denotes false.

Dashed lines go to vertices at lower (earlier) “decision levels”.

From Conflict Graph to Resolution Derivation of Conflict Clause

- Resolution procedure given by L. Zhang and Malik [DATE 03].
- *Conflict Clause Proof* proposed by Goldberg and Novikov [DATE 03].
- Theoretical analysis by Beame, Kautz, and Sabharwal [*JAIR* 2004]

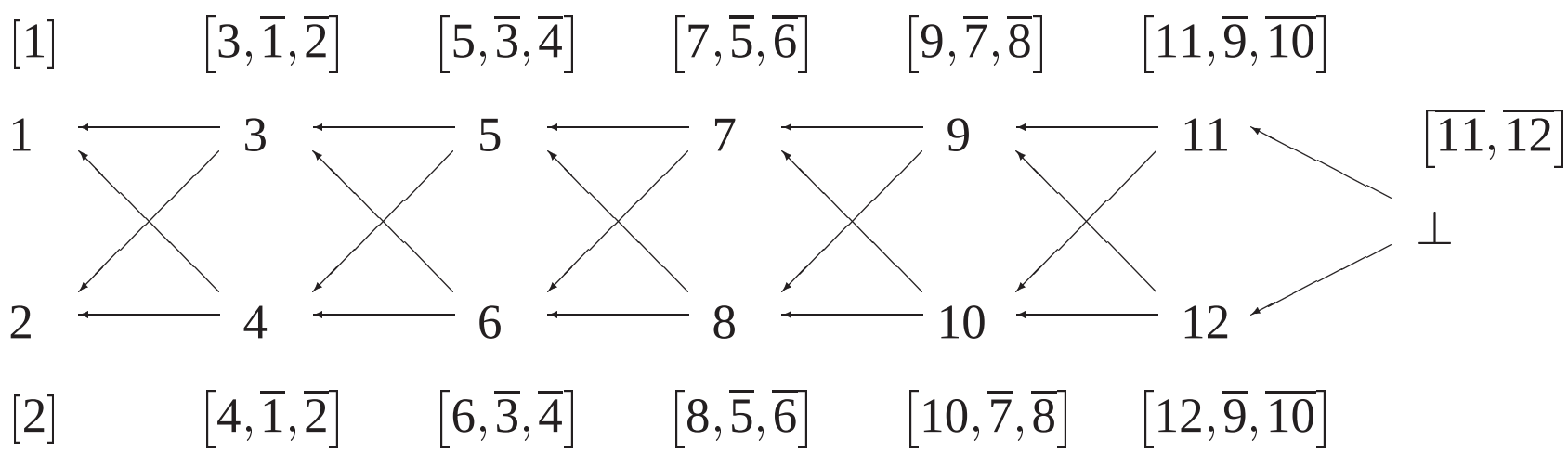
L. Zhang and Malik pseudocode to generate resolution proof of empty clause:

```
1.    cl = final_conflicting_clause;
2.    while (!is_empty_clause(cl)) {
3.        lit = choose_literal(cl);
4.        var = variable_of_literal(lit);
5.        ante_cl = antecedent(var);
6.        cl = resolve(cl, ante_cl); }
```

Note: On line 1, `final_conflicting_clause` is the antecedent of $\perp$, *not* a conflict clause.

DAG Family with Exponential Worst Case for Foregoing Procedure

- Diagram shows $h = 6$ instance.
- Antecedent clauses are shown in brackets; overbars denote negation.
- Grows as 2^h for "choose smallest variable number".
- Grows as 2^h for "choose smallest DAG height".



`zverify_df` version 2005.11.15 is exponential for reversed variables numbers (thanks to Zhaohui Fu for verifying this experimentally).

Experimental Manifestation of Inefficiency

- Sizes are mega-literals.
- "+?" indicates job was killed when size exceeded stated number.

Small Goldberg-Novikov 03

benchmark	2004.11.15	with fix
5pipe	153	15
5pipe_1_ooo	986	92
5pipe_5_ooo	2320	94
6pipe	169	97
6pipe_6_ooo	3072+?	486
7pipe	358	319
9vliw_bp_mc	308	58
barrel7	2754	5
barrel8	3072+?	73
barrel9	3072+?	80
c3540	393	78
c5315	162	9
c7552	2650	22

IBM_FV SAT 2005 Competition

benchmark	2004.11.15	with fix
01_SAT_dat.k10	3	0.134
07_SAT_dat.k30	3	2.582
07_SAT_dat.k35	3	2.859
18_SAT_dat.k10	174	1
18_SAT_dat.k15	102,520+?	14
1_11_SAT_dat.k10	13	0.338
1_11_SAT_dat.k15	128	2
20_SAT_dat.k10	13	0.208
23_SAT_dat.k10	3	0.062
23_SAT_dat.k15	15,259+?	4
26_SAT_dat.k10	0.036	0.036
2_14_SAT_dat.k10	70	1
2_14_SAT_dat.k15	3220	7

What Was the Fix?

Trivial Resolution (TVR) defined by Beame, Kautz, Subharwal

- *Linear* (must use previous resolvent), *Input* (must use an input clause)
- No repeated clashing literal

Their Proposition 4: Conflict clause can be derived by a TVR.

Caveat: Appropriate order must be found.

zchaff was already using TVR for all *nonempty* conflict clauses.

Fix: Do the same to derive the *empty* clause!