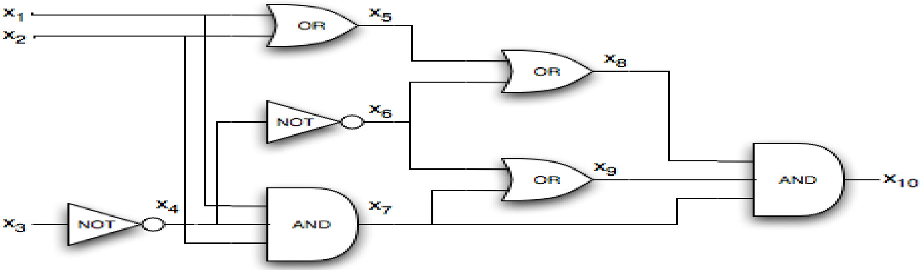
Practical SAT Solving

Lecture 8

Carsten Sinz, Tomáš Balyo | June 13, 2016

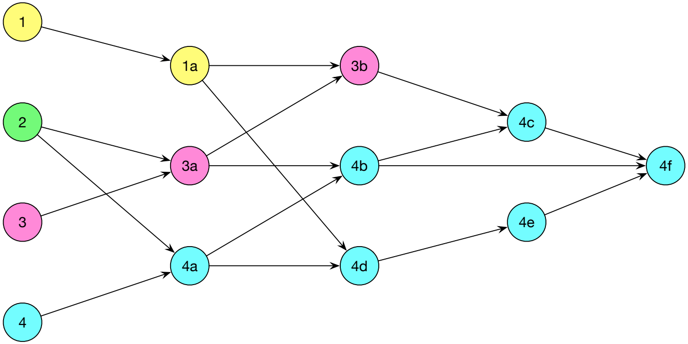
INSTITUTE FOR THEORETICAL COMPUTER SCIENCE



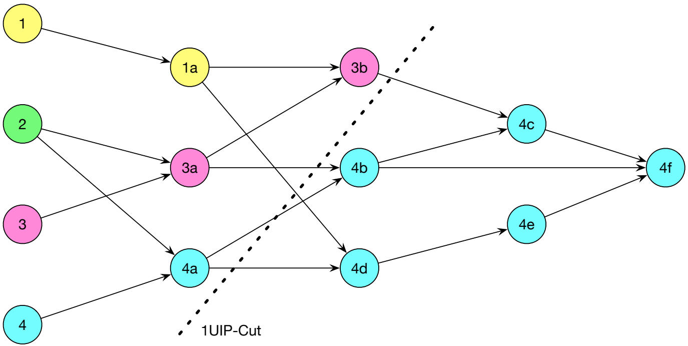
Lecture Outline

- Repetition: 1-UIP
- Preprocessing
- Inprocessing

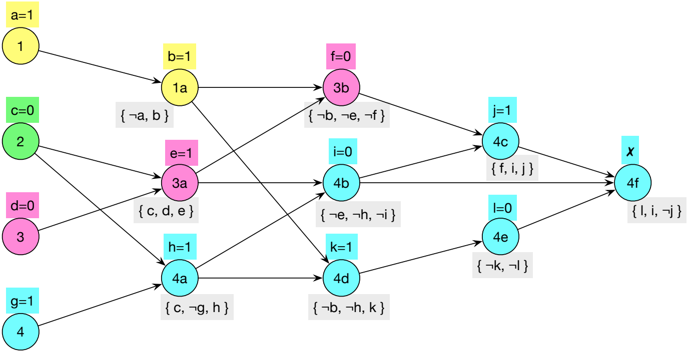
1-UIP Clause



1-UIP Clause

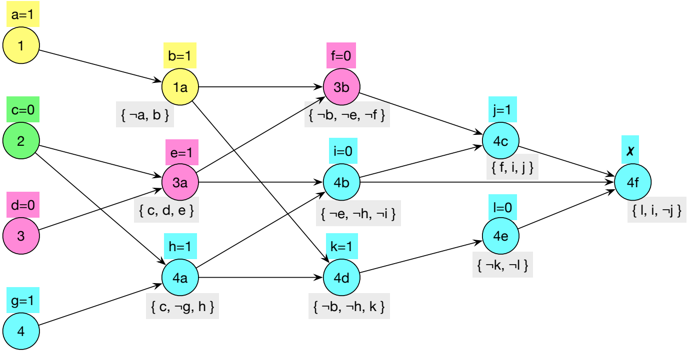


1-UIP Clause



a=1
1
b=1
1a
{ ¬a, b }
f=0
3b
{ ¬b, ¬e, ¬f }
c=0
2
e=1
3a
{ c, d, e }
i=0
4b
{ ¬e, ¬h, ¬i }
j=1
4c
{ f, i, j }
✗
4f
{ l, i, ¬j }
d=0
3
l=0
4e
{ ¬k, ¬l }
h=1
4a
{ c, ¬g, h }
k=1
4d
{ ¬b, ¬h, k }
g=1
4

1-UIP Clause



4f:		{ l, i, ¬j }	3x	
4e:	{ ¬k, ¬l }	{ i, ¬j, ¬k }	3x	
4d:	{ ¬b, ¬h, k }	{ ¬b, ¬h, i, ¬j }	3x	
4c:	{ f, i, j }	{ ¬b, f, ¬h, i }	2x	
4b:	{ ¬e, ¬h, ¬i }	{ ¬b, ¬e, f, ¬h }	1x	1UIP clause: { ¬b, ¬e, f, ¬h }

Self-Subsuming Resolution

Definition: Subsumption

A clause D subsumes a clause C iff $D \subseteq C$. We also say that clause C is subsumed by D.

To check satisfiability, subsumed clauses are irrelevant.

Definition: Self-Subsuming Resolution

Let C, D be clauses and $\otimes_x$ the resolution operator on variable x. If $C \otimes D \subseteq C$ then C is said to be *self-subsumed by D with respect to x*.

Example: $\{\neg b, \neg e, f, \neg h\}$ is self-subsumed by $\{\neg b, \neg e, \neg f\}$ w.r.t. f.
If C is self-subsumed by D, C can be replaced by $C \otimes D$. I.e. the learned clause in the 1UIP example can be strengthened to $\{\neg b, \neg e, \neg h\}$.

Preprocessing

- General idea: try to reduce the input formula by (polynomial time) simplification procedures.
- Assumption: smaller problems are easier to solve

Variable Elimination

Elimination by Clause Distribution

For a CNF F, let S_x and $S_{\overline{x}}$ be the sets of clauses containing x resp. $\overline{x}$, and $S = S_x \cup S_{\overline{x}}$. Let $S_x \otimes S_{\overline{x}} = \{C \otimes D \mid C \in S_x, D \in S_{\overline{x}}\}$. Replacing S by $S_x \otimes S_{\overline{x}}$ in F is called *elimination by clause distribution*.

The formulas F and F', where S is replaced by $S' = S_x \otimes S_{\overline{x}}$ are satisfiability equivalent.

Variable elimination procedure [1]:

- Apply "elimination by clause distribution" to all variables, but replace S by S' only if number of clauses decreases
- Do not count tautological resolvents

Variable elimination by clause distribution is also called "Bounded Variable Elimination" (BVE). It is the most important simplification technique today.

Variable Elimination and Gates

- Gates occur frequently when encoding hardware circuits into CNF.
- For example, the gate $x = AND(y, z)$ results in the clauses $\{\neg x, y\}, \{\neg x, z\}, \{x, \neg y, \neg z\}$.
- Resolving the clauses of a gate results in tautological clauses.
- Idea: detect a gate, split formula F into $F = G \cup R$, where G are the gate-clauses and R the remaining clauses.
- Apply elimination by clause distribution: $S = (G_x \cup R_x) \cup (G_{\overline{x}} \cup R_{\overline{x}})$.
- Clause distribution results in $S' = (G_x \otimes R_{\overline{x}}) \cup (R_x \otimes G_{\overline{x}}) \cup (R_x \otimes R_{\overline{x}})$
- Moreover, $(G_x \otimes R_{\overline{x}}) \cup (R_x \otimes G_{\overline{x}}) \models (R_x \otimes R_{\overline{x}})$. (Why?)
- Thus, we can replace S by the satisfiability-equivalent $(R_x \otimes G_{\overline{x}}) \cup (R_{\overline{x}} \otimes G_x)$.

Failed Literal Elimination

- If applying unit propagation on $F \wedge \{l\}$ derives UNSAT ($F \wedge \{l\} \vdash_{UP} \bot$), replace F by $F \wedge \{\neg l\}$.
- Generalization: If $F \setminus \{C\} \wedge \neg C \vdash_{UP} \bot$, remove C from F.

Blocked Clauses

Definition

A clause $(l \vee C)$ is blocked w.r.t. F by l if for every clause $(\neg l \vee D)$ in F the resolvent $(C \vee D)$ is a tautology.

Example: $F = (a \vee b) \wedge (a \vee \neg b \vee \neg c) \wedge (\neg a \vee c)$.
First clause is not blocked, second is blocked by both a and $\neg c$, third is blocked by c.

Removal of an arbitrary blocked clause preserves satisfiability. Blocked clause elimination (BCE) has a unique fixpoint.

Hidden Tautology Elimination

Karlsruhe Institute of Technology

Definition

The binary implication graph G_{BIG} of a formula F has litarals of F as vertices and for each clause $\{\neg a, b\}$ there is an edge between a and b.

Definition: Hidden Tautology Elimination (HTE)

Remove clauses that are subsumed by an (arbitrary step) implication in the Binary Implication Graph.

Autarkies

- Autarkies are a generalization of pure literals

Autarky

Given a partial assignment σ and a formula F, a clause (of F) is *touched by* σ if it contains the negation of a literal assigned in σ. A clause is *satisfied by* σ if it contains a literal assigned to *True* by σ. If all touched clauses are satisfied then σ is an *autarky*.

- All clauses touched by an autarky can be removed
- Example: $\{\neg a, b\}, \{\neg a, c\}, \{a, \neg b, \neg c\}, \{b, d\}, \ldots$ (more clauses without a and c)
- Then $\sigma = \{\neg a, \neg c\}$ is an autarky.

Preprocessing Techniques that do not Reduce the Problem Size

- There are techniques, such as Bounded Variable Addition (BVA), that increase problem size.
- BVA is mainly based on the Extended Resolution rule.

Extended Resolution

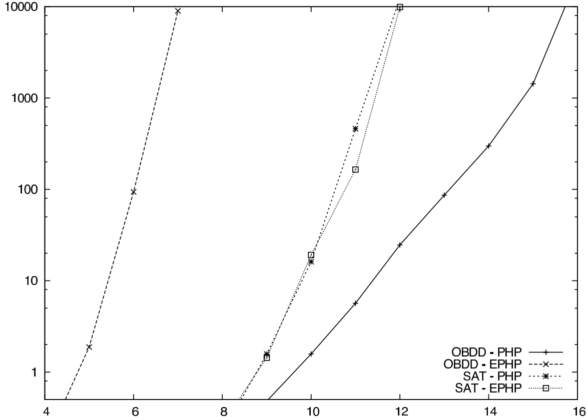
Extended resolution adds a second rule to the resolution calculus, the Extension Rule. The idea is to introduce new variables as conjunction of existing literals, $x_{\mathrm{new}} \leftrightarrow l_1 \wedge l_2$. As a rule for formulas in CNF:

$$\frac{}{(\neg x_{\mathrm{new}} \vee l_1) \wedge (\neg x_{\mathrm{new}} \vee l_2) \wedge (x_{\mathrm{new}} \vee \neg l_1 \vee \neg l_2)}$$

- There are proofs with exponential size by Resolution, but polynomial size by Extended Resolution, e.g. pigeonhole formulas [2, 3].

Ext. Res. and Pigeonhole Formulas

Karlsruhe Institute of Technology



From *Tveretina, Sinz, Zantema: Ordered Binary Decision Diagrams, Pigeonhole Formulas and Beyond. JSAT 7(1): 35-58 (2010)*.

Inprocessing: Idea [4]

- Idea: Interleave search and preprocessing
- Preprocessing can be extremely beneficial
  - Most solvers in the SAT competitions use variable elimination
  - Equivalence / XOR reasoing
  - Failed literal elimination
- Many preprocessing techniques, though polynomial, require considerable time

Inprocessing

- “Preempt” (interrupt) preprocessing techniques after some time
- Resume preprocessing between restarts
- Limit preprocessing time in relation to search

References I

- N. Eén, A. Biere, Effective preprocessing in SAT through variable and clause elimination, in: Proceedings of the 8th International Conference on Theory and Applications of Satisfiability Testing, Springer-Verlag, Berlin, Heidelberg, 2005, pp. 61–75. `doi:10.1007/11499107_5`. URL `http://dx.doi.org/10.1007/11499107_5`
- A. Haken, The intractability of resolution (complexity), Ph.D. thesis, University of Illinois at Urbana-Champaign, Champaign, IL, USA (1984).
- S. A. Cook, A short proof of the pigeon hole principle using extended resolution, SIGACT News 8 (4) (1976) 28–32. URL `http://doi.acm.org/10.1145/1008335.1008338`

References II

M. Järvisalo, M. J. H. Heule, A. Biere, Inprocessing rules, in: Automated Reasoning: 6th International Joint Conference, IJCAR 2012, Manchester, UK, June 26-29, 2012. Proceedings, Springer Berlin Heidelberg, 2012, pp. 355–370.
URL http://dx.doi.org/10.1007/978-3-642-31365-3_28