

SAT Solver Heuristics

SAT-solver History

- Started with David-Putnam-Logemann-Loveland (DPLL) (1962)
 - Able to solve 10-15 variable problems
- Satz (Chu Min Li, 1995)
 - Able to solve some 1000 variable problems
- Chaff (Malik et al., 2001)
 - Intelligently hacked DPLL , Won the 2004 competition
 - Able to solve some 10000 variable problems
- Current state-of-the-art
 - MiniSAT and SATELITEGTI (Chalmer's university, 2004-2006)
 - Jerusat and Haifasat (Intel Haifa, 2002)
 - Ace (UCLA, 2004-2006)

MiniSAT

- MiniSat is a **fast SAT solver** developed by Niklas Eén and Niklas Sörensson
 - MiniSat **won all industrial categories** in SAT 2005 competition
 - MiniSat **won SAT-Race 2006**
- MiniSat is simple and well-documented
 - **Well-defined interface** for general use
 - Helpful implementation **documents** and **comments**
 - **Minimal but efficient** heuristic
 - Main.C (344 lines)
 - Solver.C (741 lines)

Overview (1/2)

- A set of propositional variables and CNF clauses involving variables
 - $(x_1 \vee x_2' \vee x_3) \wedge (x_2 \vee x_1' \vee x_4)$
 - x_1, x_2, x_3 and x_4 are variables (true or false)
- Literals: Variable and its negation
 - x_1 and x_1'
- A clause is satisfied if one of the literals is true
 - $x_1 = \text{true}$ satisfies clause 1
 - $x_1 = \text{false}$ satisfies clause 2
- Solution: An assignment that satisfies all clauses

Overview (2/2)

- **Unit clause** is a clause in which **all but one of literals** is assigned to **False**

- **Unit literal** is the **unassigned literal** in a unit clause

.....

$$(x_0) \wedge$$

$$(-x_0 \vee x_1) \wedge$$

$$(-x_2 \vee -x_3 \vee -x_4) \wedge$$

.....

- (x_0) is a unit clause and 'x₀' is a unit literal
- $(-x_0 \vee x_1)$ is a unit clause since x₀ has to be True
- $(-x_2 \vee -x_3 \vee -x_4)$ can be a unit clause if the current assignment is that x₃ and x₄ are True

- **Boolean Constrain Propagation(BCP)** is the process of assigning the True value to all unit literals

DPLL Overview (1/3)

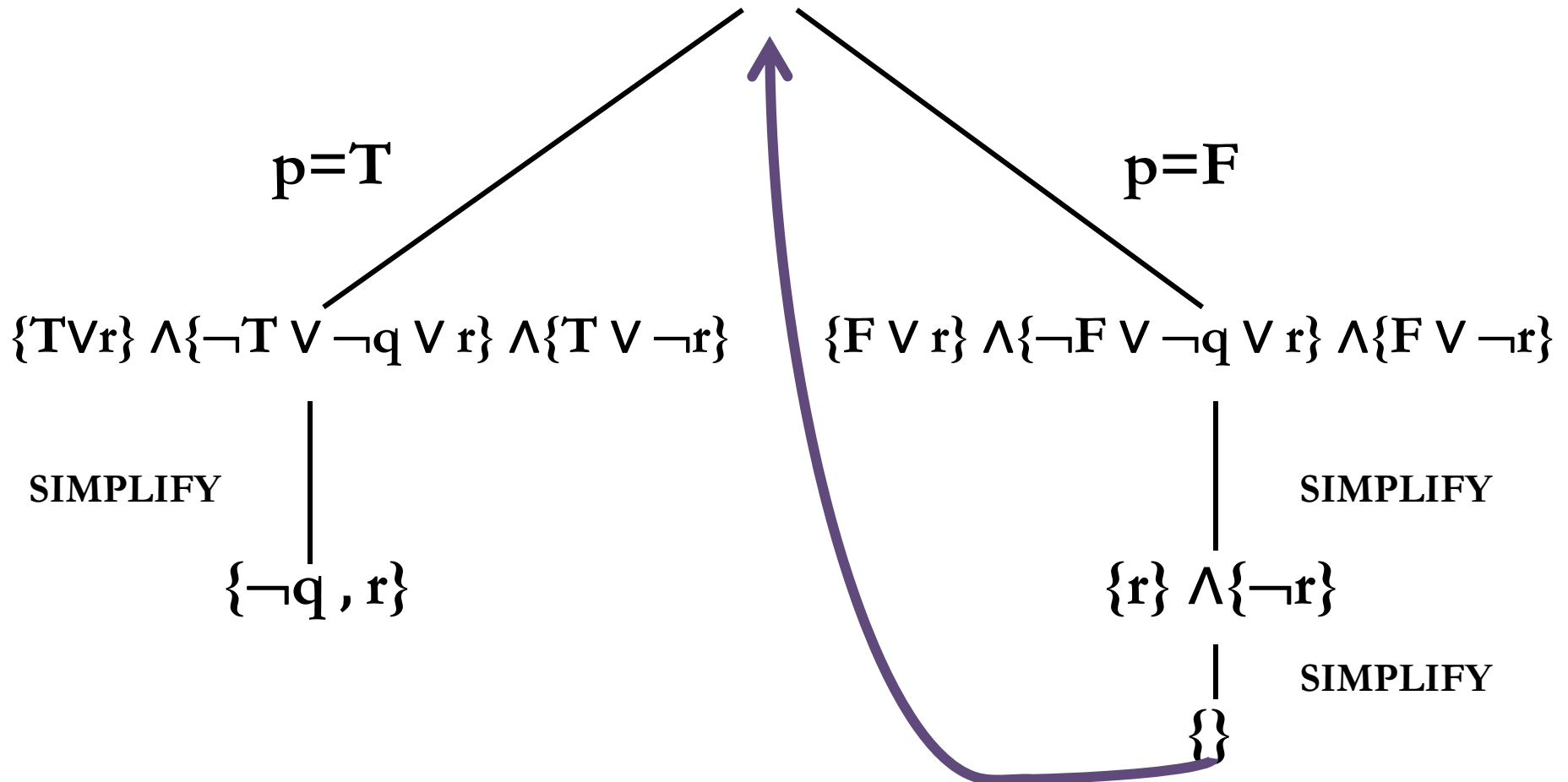
/* The Quest for Efficient Boolean Satisfiability Solvers

* by L.Zhang and S.Malik, Computer Aided Verification 2002 */

```
DPLL(a formula  $\phi$ , assignment) {
    necessary = deduction( $\phi$ , assignment);
    new_asgnment = union(necessary, assignment);
    if (is_satisfied( $\phi$ , new_asgnment))
        return SATISFIABLE;
    else if (is_conflicting( $\phi$ , new_asgnment))
        return UNSATISFIABLE;
    var = choose_free_variable( $\phi$ , new_asgnment);
    asgn1 = union(new_asgnment, assign(var, 1));
    if (DPLL( $\phi$ , asgn1) == SATISFIABLE)
        return SATISFIABLE;
    else {
        asgn2 = union (new_asgnment, assign(var,0));
        return DPLL ( $\phi$ , asgn2);
    }
}
```

DPLL Overview (2/3)

$$\{p \vee r\} \wedge \{\neg p \vee \neg q \vee r\} \wedge \{p \vee \neg r\}$$



DPLL Overview (3/3)

```
/* overall structure of Minisat solve procedure */
Solve(){
  while(true){
    boolean_constraint_propagation();
    if(no_conflict){
      if(no_unassigned_variable) return SAT;
      decision_level++;
      make_decision();
    }else{
      if (no_decisions_made) return UNSAT;
      analyze_conflict();
      undo_assignments();
      add_conflict_clause();
    }
  }
}
```


Conflict Clause Analysis (1/10)

- A conflict happens when one clause is falsified by unit propagation

Assume x_4 is False

$(x_1 \vee x_4) \wedge$

$(\neg x_1 \vee x_2) \wedge$

$(\neg x_2 \vee x_3) \wedge$

$(\neg x_3 \vee \neg x_2 \vee \neg x_1)$ **Falsified!**

Omitted clauses

- Analyze the **conflicting clause** to infer a clause
 - $(\neg x_3 \vee \neg x_2 \vee \neg x_1)$ is conflicting clause
- The inferred clause is a new knowledge
 - A new learnt clause is added to constraints

Conflict Clause Analysis (2/10)

- Learnt clauses are inferred by conflict analysis

$(x_1 \vee x_4) \wedge$
 $(\neg x_1 \vee x_2) \wedge$
 $(\neg x_2 \vee x_3) \wedge$
 $(\neg x_3 \vee \neg x_2 \vee \neg x_1) \wedge$
omitted clauses \wedge
 (x_4) learnt clause

- They help prune future parts of the search space
 - Assigning False to x_4 is the casual of conflict
 - Adding (x_4) to constraints prohibit conflict from $\neg x_4$
- Learnt clauses actually drive backtracking

Conflict Clause Analysis (3/10)

```
/* conflict analysis algorithm */
Analyze_conflict(){
    cl = find_conflicting_clause();
    /* Loop until cl is falsified and one literal whose value is determined in current
    decision level is remained */
    While(!stop_criterion_met(cl)){
        lit = choose_literal(cl); /* select the last propagated literal */
        Var = variable_of_literal(lit);
        ante = antecedent(var);
        cl = resolve(cl, ante, var);
    }
    add_clause_to_database(cl);
    /* backtrack level is the lowest decision level for which the learnt clause is
    unit clause */
    back_dl = clause_asserting_level(cl);
    return back_dl;
}
```

Algorithm from Lintao Zhang and Sharad malik
"The Quest for Efficient Boolean Satisfiability Solvers"

Conflict Clause Analysis (4/10)

- Example of conflict clause analysis
 - a, b, c, d, e, f, g, and h: 8 variables (2^8 cases)

$(\neg f \vee e) \wedge$
 $(\neg g \vee f) \wedge$
 $(b \vee a \vee e) \wedge$
 $(c \vee e \vee f \vee \neg b) \wedge$
 $(\neg h \vee g)$
 $(d \vee \neg b \vee h) \wedge$
 $(\neg b \vee \neg c \vee \neg d) \wedge$
 $(c \vee d)$

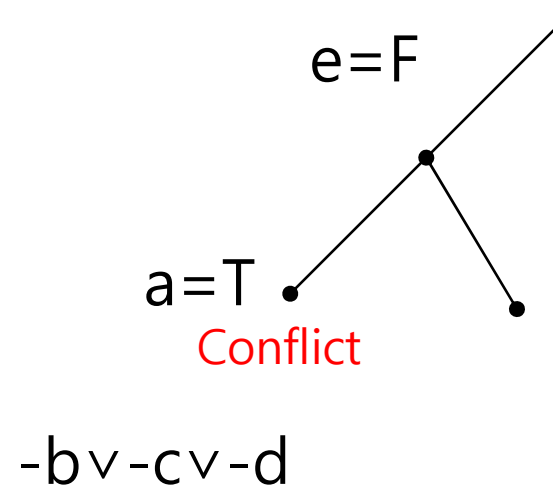
Satisfiable?

Unsatisfiable?

Conflict Clause Analysis (5/10)

Assignments	antecedent
e=F	assumption
f=F	-fve
g=F	-gvf
h=F	-hvg
a=F	assumption
b=T	bvave
c=T	cvevfv-b
d=T	dv-bvh

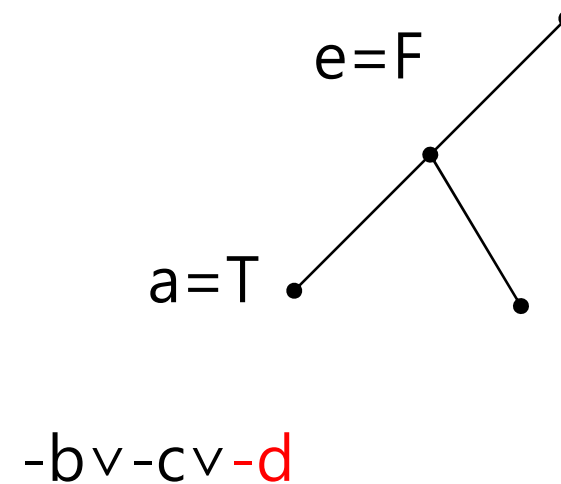
DLevel=1 (bracketed around f=F, g=F, h=F)
 DLevel=2 (bracketed around b=T, c=T, d=T)



Example slides are from CMU 15-414 course ppt

Conflict Clause Analysis (6/10)

Assignments	antecedent
e=F	assumption
f=F	-fve
g=F	-gvf
h=F	-hvg
a=F	assumption
b=T	bvave
c=T	cvevfv-b
d=T	d v-bvh



Resolution

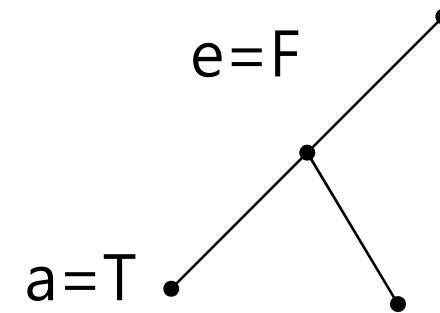
- Resolution is a process to generate a clause from two clauses
- Given two clauses $(x \vee y)$ and $(-y \vee z)$, the **resolvent** of these two clauses is $(x \vee z)$
 - $(x \vee y) \wedge (-y \vee z)$ is satisfiable iff $(x \vee y) \wedge (-y \vee z) \wedge (x \vee z)$ is satisfiable
 - The resolvent is redundant

Conflict Clause Analysis (7/10)

Assignments	antecedent
e=F	assumption
f=F	-fve
g=F	-gvf
h=F	-hvg
a=F	assumption
b=T	bvave
c=T	cvevfv-b
d=T	dv-bvh

DLevel=1

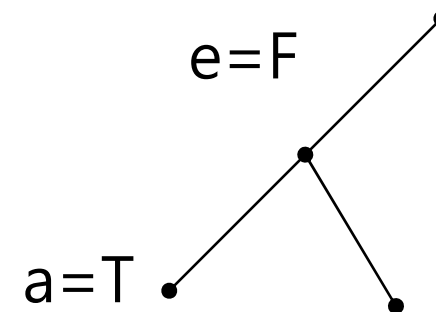
DLevel=2



-bv-cv-d
 -bv-cvh
 (a resolvent of
 -bv-cv-d
 and dv-bvh)

Conflict Clause Analysis (8/10)

Assignments	antecedent
e=F	assumption
f=F	-fve
g=F	-gvf
h=F	-hvg
a=F	assumption
b=T	bvave
c=T	cvevfv-b
d=T	dv-bvh

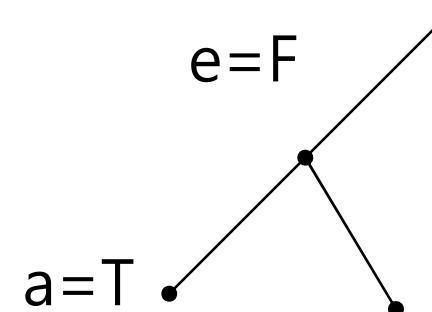


-bv-cv-d

-bv-cvh

Conflict Clause Analysis (9/10)

Assignments	antecedent
e=F	assumption
f=F	-fve
g=F	-gvf
h=F	-hvg
DLevel=1	
a=F	assumption
b=T	bvave
c=T	cvefv-b
d=T	dv-bvh
DLevel=2	



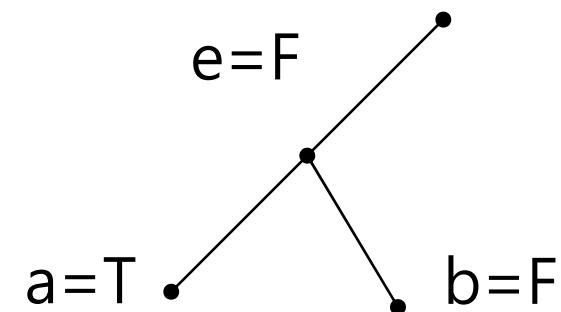
-bv-cv-d

-bv-cvh

-bvefvh **learnt clause**

Conflict Clause Analysis (10/10)

Assignments	antecedent
e=F	assumption
f=F	-fve
g=F	-gvf
h=F	-hvg
b=F	-bvevfvh
...	...



-bv-cv-d
 -bv-cvh
 -bvevfvh

Variable State Independent Decaying Sum(VSIDS)

- **Decision heuristic** to determine what variable will be assigned next
- Decision is **independent** from **the current assignment** of each variable
- VSIDS makes decisions based on **activity**
 - Activity is **a literal occurrence count** with higher weight on the more recently added clauses
 - MiniSAT does not consider any polarity in VSIDS
 - Each variable, not literal has score

activity description from Lintao Zhang and Sharad malik
"The Quest for Efficient Boolean Satisfiability Solvers"

VSIDS Decision Heuristic – MiniSAT style (1/8)

- Initially, the score for each variable is 0
- First make a decision $e = \text{False}$
 - The order between same score is unspecified.
 - MiniSAT always assigns False to variables.

Initial constraints

$(\neg f \vee e) \wedge$
 $(\neg g \vee f) \wedge$
 $(b \vee a \vee e) \wedge$
 $(c \vee e \vee f \vee \neg b) \wedge$
 $(\neg h \vee g) \wedge$
 $(d \vee \neg b \vee h) \wedge$
 $(\neg b \vee \neg c \vee \neg d) \wedge$
 $(c \vee d)$

Variable	Score	Value
a	0	
b	0	
c	0	
d	0	
e	0	F
f	0	
g	0	
h	0	

VSIDS Decision Heuristic (2/8)

- f, g, h are False after BCP

$(\neg f \vee e) \wedge$
 $(\neg g \vee f) \wedge$
 $(b \vee a \vee e) \wedge$
 $(c \vee e \vee f \vee \neg b) \wedge$
 $(\neg h \vee g) \wedge$
 $(d \vee \neg b \vee h) \wedge$
 $(\neg b \vee \neg c \vee \neg d) \wedge$
 $(c \vee d)$

Variable	Score	Value
a	0	
b	0	
c	0	
d	0	
e	0	F
f	0	F
g	0	F
h	0	F

VSIDS Decision Heuristic (3/8)

- a is next decision variable

$(-fve) \wedge$
 $(-gvf) \wedge$
 $(bva ve) \wedge$
 $(cvevf v-b) \wedge$
 $(-hvg) \wedge$
 $(dv-bvh) \wedge$
 $(-bv-cv-d) \wedge$
 (cvd)

Variable	Score	Value
a	0	F
b	0	
c	0	
d	0	
e	0	F
f	0	F
g	0	F
h	0	F

VSIDS Decision Heuristic (4/8)

- b, c are True after BCP
- Conflict occurs on variable d
 - Start conflict analysis

$(-fve) \wedge$
 $(-gvf) \wedge$
 $(bvave) \wedge$
 $(cvevfv-b) \wedge$
 $(-hvg) \wedge$
 $(dv-bvh) \wedge$
 $(-bv-cv-d) \wedge$
 (cvd)

Variable	Score	Value
a	0	F
b	0	T
c	0	T
d	0	T
e	0	F
f	0	F
g	0	F
h	0	F

VSIDS Decision Heuristic (5/8)

- The score of variable in resolvents is increased by 1
 - Even if a variable appears in resolvents two or more increase the score just once

$(-fve) \wedge$
 $(-gvf) \wedge$
 $(bvave) \wedge$
 $(cvevfv-b) \wedge$
 $(-hvg) \wedge$
 $(dv-bvh) \wedge$
 $(-bv-cv-d) \wedge$
 (cvd)

Resolvent on d
 $-bv-cvh$

Variable	Score	Value
a	0	F
b	1	T
c	1	T
d	0	T
e	0	F
f	0	F
g	0	F
h	1	F

VSIDS Decision Heuristic (6/8)

- The end of conflict analysis
- The scores are decaying 5% for next scoring

(-fve) \wedge
 (-gvf) \wedge
 (bvave) \wedge
 (cvevfv-b) \wedge
 (-hvg) \wedge
 (dv-bvh) \wedge
 (-bv-cv-d) \wedge
 (cvd)

Resolvents
 -bv-cvh
 -bvevfvh \leftarrow
learnt clause

Variable	Score	Value
a	0	F
b	0.95	T
c	0.95	T
d	0	T
e	0.95	F
f	0.95	F
g	0	F
h	0.95	F

VSIDS Decision Heuristic (7/8)

- b is now False and a is True after BCP
- Next decision variable is c with 0.95 score

Learnt clause $(c \vee d) \wedge$
 $(-b \vee e \vee f \vee h)$
 $(-f \vee e) \wedge$
 $(-g \vee f) \wedge$
 $(b \vee a \vee e) \wedge$
 $(c \vee e \vee f \vee -b) \wedge$
 $(-h \vee g) \wedge$
 $(d \vee -b \vee h) \wedge$
 $(-b \vee -c \vee -d) \wedge$

Variable	Score	Value
a	0	T
b	0.95	F
c	0.95	
d	0	
e	0.95	F
f	0.95	F
g	0	F
h	0.95	F

VSIDS Decision Heuristic (8/8)

- Finally we find a model!

$(\neg f \vee e) \wedge$
 $(\neg g \vee f) \wedge$
 $(b \vee a \vee e) \wedge$
 $(c \vee e \vee f \vee \neg b) \wedge$
 $(\neg h \vee g) \wedge$
 $(d \vee \neg b \vee h) \wedge$
 $(\neg b \vee \neg c \vee \neg d) \wedge$
Learnt clause $(c \vee d) \wedge$
 $(\neg b \vee e \vee f \vee h)$

Variable	Score	Value
a	0	T
b	0.95	F
c	0.95	F
d	0	T
e	0.95	F
f	0.95	F
g	0	F
h	0.95	F