Introduction

Marijn J.H. Heule
Warren A. Hunt Jr.

The University of Texas at Austin

Motivation satisfiability solving

From 100 variables, 200 clauses (early 90's)
to 1,000,000 vars. and 5,000,000 clauses in 15 years.

Motivation satisfiability solving

From 100 variables, 200 clauses (early 90's)
to 1,000,000 vars. and 5,000,000 clauses in 15 years.

Applications:

Hardware and Software Verification, Planning, Scheduling, Optimal Control, Protocol Design, Routing, Combinatorial problems, Equivalence Checking, etc.

Motivation satisfiability solving

From 100 variables, 200 clauses (early 90's)
to 1,000,000 vars. and 5,000,000 clauses in 15 years.

Applications:

Hardware and Software Verification, Planning, Scheduling, Optimal Control, Protocol Design, Routing, Combinatorial problems, Equivalence Checking, etc.

SAT used to solve many other problems!

Overview

- Introduction
- The Satisfiability problem
- Terminology
- SAT solving
- SAT benchmarks

"You are chief of protocol for the embassy ball. The crown prince instructs you either to invite *Peru* or to exclude *Qatar*. The queen asks you to invite either *Qatar* or *Romania* or both. The king, in a spiteful mood, wants to snub either *Romania* or *Peru* or both. Is there a guest list that will satisfy the whims of the entire royal family?"

"You are chief of protocol for the embassy ball. The crown prince instructs you either to invite *Peru* or to exclude *Qatar*. The queen asks you to invite either *Qatar* or *Romania* or both. The king, in a spiteful mood, wants to snub either *Romania* or *Peru* or both. Is there a guest list that will satisfy the whims of the entire royal family?"

$$(P \vee \neg Q) \wedge (Q \vee R) \wedge (\neg R \vee \neg P)$$

Truth Table

$$F := (P \lor \neg Q) \land (Q \lor R) \land (\neg R \lor \neg P)$$

P	Q	R	falsifies	$\varphi \circ F$
0	0	0	$(Q \lor R)$	0
0	0	1	—	1
0	1	0	$(P \lor \neg Q)$	0
0	1	1	$(P \lor \neg Q)$	0
1	0	0	$(Q \lor R)$	0
1	0	1	$(\neg R \lor \neg P)$	0
1	1	0	—	1
1	1	1	$(\neg R \lor \neg P)$	0

Slightly Harder Example

What are the solutions for the following formula?

$$(A \lor B \lor \neg C)$$
$$(\neg A \lor \neg B \lor C)$$
$$(B \lor C \lor \neg D)$$
$$(\neg B \lor \neg C \lor D)$$
$$(A \lor C \lor D)$$
$$(\neg A \lor \neg C \lor \neg D)$$
$$(\neg A \lor B \lor D)$$

Slightly Harder Example

What are the solutions for the following formula?

$$(A \lor B \lor \neg C)$$
$$(\neg A \lor \neg B \lor C)$$
$$(B \lor C \lor \neg D)$$
$$(\neg B \lor \neg C \lor D)$$
$$(A \lor C \lor D)$$
$$(\neg A \lor \neg C \lor \neg D)$$
$$(\neg A \lor B \lor D)$$

A	B	C	D	A	B	C	D
0	0	0	0	1	0	0	0
0	0	0	1	1	0	0	1
0	0	1	0	1	0	1	0
0	0	1	1	1	0	1	1
0	1	0	0	1	1	0	0
0	1	0	1	1	1	0	1
0	1	1	0	1	1	1	0
0	1	1	1	1	1	1	1

Given a *CNF formula*,
does there exist an *assignment*
to the *Boolean variables*
that satisfies all *clauses*?

Boolean variable

- can be assigned the Boolean values 0 or 1

Literal

- refers either to x_i or its complement $\neg x_i$
- literals x_i are satisfied if variable x_i is assigned to 1 (true)
- literals $\neg x_i$ are satisfied if variable x_i is assigned to 0 (false)

Terminology: Clauses

Clause

- Disjunction of literals: E.g. $C_j = (l_1 \lor l_2 \lor l_3)$
- Can be falsified with only *one* assignment to its literals: All literals assigned to false
- Can be satisfied with $2^k - 1$ assignment to its k literals
- One special clause - the empty clause (denoted by $\emptyset$) - which is always falsified

Terminology: Formulae

Formula

- Conjunction of clauses: E.g. $\mathcal{F} = C_1 \land C_2 \land C_3$
- Is *satisfiable* if there exists an assignment satisfying all clauses, otherwise *unsatisfiable*
- Formulae are defined in *Conjunction Normal Form* (CNF) and generally also stored as such - also learned information

Assignment

- Mapping of the values 0 and 1 to the variables
- $\varphi \circ \mathcal{F}$ results in a reduced formula $\mathcal{F}_{\text{reduced}}$:
  - all satisfied clauses are removed
  - all falsified literals are removed
- *satisfying assignment* $\leftrightarrow$ $\mathcal{F}_{\text{reduced}}$ is empty
- *falsifying assignment* $\leftrightarrow$ $\mathcal{F}_{\text{reduced}}$ contains $\emptyset$
- *partial assignment* versus *full assignment*

Resolution

Given two clauses $C_1 = (x \vee a_1 \vee \cdots \vee a_n)$ and $C_2 = (\neg x \vee b_1 \vee \cdots \vee b_m)$, the resolvent of C_1 and C_2 (denoted by $C_1 \bowtie C_2$) is $R = (a_1 \vee \cdots \vee a_n \vee b_1 \vee \cdots \vee b_m)$

Resolution

Given two clauses $C_1 = (x \vee a_1 \vee \cdots \vee a_n)$ and $C_2 = (\neg x \vee b_1 \vee \cdots \vee b_m)$, the resolvent of C_1 and C_2 (denoted by $C_1 \bowtie C_2$) is $R = (a_1 \vee \cdots \vee a_n \vee b_1 \vee \cdots \vee b_m)$

Examples for $F := (P \vee \neg Q) \wedge (Q \vee R) \wedge (\neg R \vee \neg P)$

- $(P \vee \neg Q) \bowtie (Q \vee R) = (P \vee R)$
- $(P \vee \neg Q) \bowtie (\neg R \vee \neg P) = (\neg Q \vee \neg R)$
- $(Q \vee R) \bowtie (\neg R \vee \neg P) = (Q \vee \neg P)$

Resolution

Given two clauses $C_1 = (x \vee a_1 \vee \dots \vee a_n)$ and $C_2 = (\neg x \vee b_1 \vee \dots \vee b_m)$, the resolvent of C_1 and C_2 (denoted by $C_1 \bowtie C_2$) is $R = (a_1 \vee \dots \vee a_n \vee b_1 \vee \dots \vee b_m)$

Examples for $F := (P \vee \neg Q) \wedge (Q \vee R) \wedge (\neg R \vee \neg P)$

- $(P \vee \neg Q) \bowtie (Q \vee R) = (P \vee R)$
- $(P \vee \neg Q) \bowtie (\neg R \vee \neg P) = (\neg Q \vee \neg R)$
- $(Q \vee R) \bowtie (\neg R \vee \neg P) = (Q \vee \neg P)$

Resolution, i.e., adding resolvents until fixpoint, is a complete proof procedure. It produces the empty clause if and only if the formula is unsatisfiable

A clause C is a tautology if it contains
for some variable x, both the literals x and $\neg x$.

Slightly Harder Example 2

Compute all non-tautological resolvents for:

$$(A \lor B \lor \neg C) \land (\neg A \lor \neg B \lor C) \land \\ (B \lor C \lor \neg D) \land (\neg B \lor \neg C \lor D) \land \\ (A \lor C \lor D) \land (\neg A \lor \neg C \lor \neg D) \land \\ (\neg A \lor B \lor D)$$

Which resolvents remain after removing the supersets?

SAT solving: Unit propagation

A *unit clause* is a clause of size 1

UnitPropagation $(\varphi, \mathcal{F})$:

while $\emptyset \notin \mathcal{F}$ **and** unit clause y exists **do**
    expand φ and simplify $\mathcal{F}$
end while
return $\varphi, \mathcal{F}$

Unit propagation: Example

$$\mathcal{F}_{\text{unit}} := (\neg x_1 \lor \neg x_3 \lor x_4) \land (\neg x_1 \lor \neg x_2 \lor x_3) \land (\neg x_1 \lor x_2) \land (x_1 \lor x_3 \lor x_6) \land (\neg x_1 \lor x_4 \lor \neg x_5) \land (x_1 \lor \neg x_6) \land (x_4 \lor x_5 \lor x_6) \land (x_5 \lor \neg x_6)$$

Unit propagation: Example

$$\mathcal{F}_{\text{unit}} := (\neg x_1 \vee \neg x_3 \vee x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (x_1 \vee x_3 \vee x_6) \wedge (\neg x_1 \vee x_4 \vee \neg x_5) \wedge (x_1 \vee \neg x_6) \wedge (x_4 \vee x_5 \vee x_6) \wedge (x_5 \vee \neg x_6)$$

$$\varphi = \{x_1{=}1\}$$

Unit propagation: Example

$$\mathcal{F}_{\text{unit}} := (\neg x_1 \lor \neg x_3 \lor x_4) \land (\neg x_1 \lor \neg x_2 \lor x_3) \land (\neg x_1 \lor x_2) \land (x_1 \lor x_3 \lor x_6) \land (\neg x_1 \lor x_4 \lor \neg x_5) \land (x_1 \lor \neg x_6) \land (x_4 \lor x_5 \lor x_6) \land (x_5 \lor \neg x_6)$$

$$\varphi = \{x_1{=}1, x_2{=}1\}$$

Unit propagation: Example

$$\mathcal{F}_{\text{unit}} := (\neg x_1 \vee \neg x_3 \vee x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2) \wedge (x_1 \vee x_3 \vee x_6) \wedge (\neg x_1 \vee x_4 \vee \neg x_5) \wedge (x_1 \vee \neg x_6) \wedge (x_4 \vee x_5 \vee x_6) \wedge (x_5 \vee \neg x_6)$$

$$\varphi = \{x_1{=}1, x_2{=}1, x_3{=}1\}$$

Unit propagation: Example

$$\mathcal{F}_{\text{unit}} := (\neg x_1 \lor \neg x_3 \lor x_4) \land (\neg x_1 \lor \neg x_2 \lor x_3) \land (\neg x_1 \lor x_2) \land (x_1 \lor x_3 \lor x_6) \land (\neg x_1 \lor x_4 \lor \neg x_5) \land (x_1 \lor \neg x_6) \land (x_4 \lor x_5 \lor x_6) \land (x_5 \lor \neg x_6)$$

$$\varphi = \{x_1{=}1, x_2{=}1, x_3{=}1, x_4{=}1\}$$

Davis Putnam Logemann Loveland [DP60,DLL62]

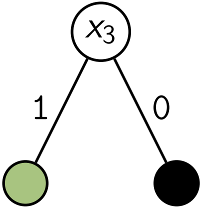
- Simplify (Unit Propagation)
- Split the formula
    - Variable Selection Heuristics
    - Direction heuristics

DPLL: Example

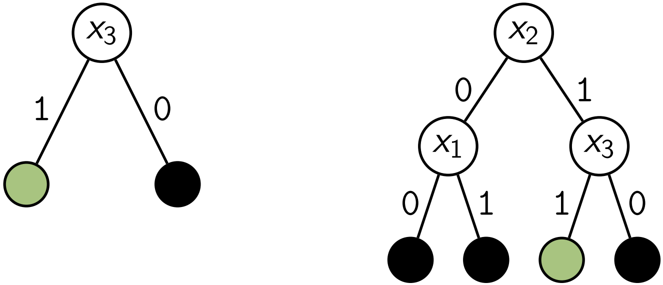
$$\mathcal{F}_{\text{DPLL}} := (x_1 \lor x_2 \lor \neg x_3) \land (\neg x_1 \lor x_2 \lor x_3) \land (\neg x_1 \lor \neg x_2 \lor x_3) \land (x_1 \lor x_3) \land (\neg x_1 \lor \neg x_3)$$

DPLL: Example

$$\mathcal{F}_{\text{DPLL}} := (x_1 \lor x_2 \lor \neg x_3) \land (\neg x_1 \lor x_2 \lor x_3) \land (\neg x_1 \lor \neg x_2 \lor x_3) \land (x_1 \lor x_3) \land (\neg x_1 \lor \neg x_3)$$



$$\mathcal{F}_{\text{DPLL}} := (x_1 \lor x_2 \lor \neg x_3) \land (\neg x_1 \lor x_2 \lor x_3) \land (\neg x_1 \lor \neg x_2 \lor x_3) \land (x_1 \lor x_3) \land (\neg x_1 \lor \neg x_3)$$



Slightly Harder Example 3

Construct a DPLL tree for:

$$(A \vee B \vee \neg C) \wedge (\neg A \vee \neg B \vee C) \wedge$$
$$(B \vee C \vee \neg D) \wedge (\neg B \vee \neg C \vee D) \wedge$$
$$(A \vee C \vee D) \wedge (\neg A \vee \neg C \vee \neg D) \wedge$$
$$(\neg A \vee B \vee D)$$

SAT solving: Decisions, implications

Decision variables

- Selected by the heuristics
- Play a crucial role in performance

Implied variables

- Assigned by reasoning (e.g. unit propagation)
- Maximizing the number of implied variables is an important aspect of look-ahead SAT solvers

SAT Solving: Clauses $\leftrightarrow$ assignments

- A clause C represents a set of falsified assignments, i.e. those assignments that falsify all literals in C
- A falsifying assignment φ for a given formula represents a set of clauses that follow from the formula
  - For instance with all decision variables
  - Important feature of conflict-driven SAT solvers

SAT solvers

Conflict-driven

- "brute-force", complete
- examples: zchaff, minisat, rsat

Look-ahead

- lots of reasoning, complete
- examples: march, OKsolver, kcnfs

Local search

- local optimazations, incomplete
- examples: WalkSAT, UnitWalk

Applications: Industrial

- Model Checking
  - Turing award '07 Clarke, Emerson, and Sifakis
- Software Verification
- Hardware Verification
- Equivalence Checking Problems

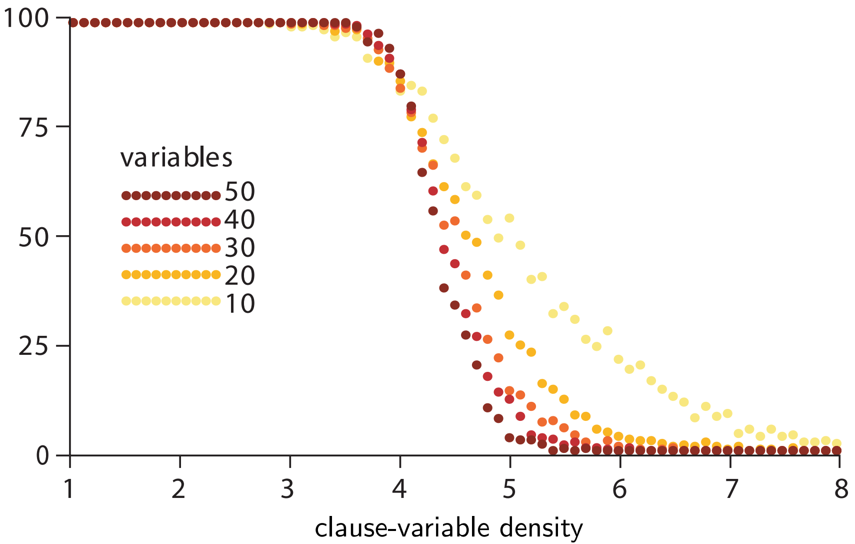
Applications: Crafted

- Combinatorial problems
- Sudoku
- Factorization problems

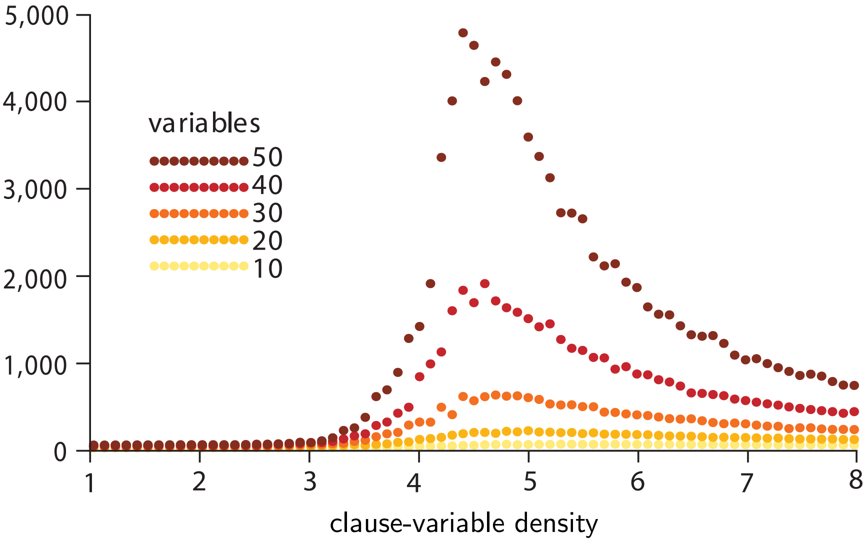
Random k-SAT: Introduction

- All clauses have length k
- Variables have the same probability to occur
- Each literal is negated with probability of 50%
- Density is ratio Clauses to Variables

Random 3-SAT: % satisfiable, the phase transition



Random 3-SAT: exponential runtime, the threshold



SAT game

by Olivier Roussel

http://www.cs.utexas.edu/~marijn/game/

Introduction

Marijn J.H. Heule
Warren A. Hunt Jr.

The University of Texas at Austin