Satisfiability

- SAT problem
- Standard representation
- DPLL algorithm
- Automata and SAT
- SMT

Def. Given a well-formed formula in propositional logic, determine whether there exists a satisfying solution (or valuation).

Boolean Satisfiability Problem SAT

Example: $\alpha(x_1, x_2, .. x_k) \equiv (x_1 \wedge x_2 \vee x_3) \wedge$
$(\underbrace{x_1 \wedge \neg x_3 \vee x_2}_{\text{Clause}})$

Set of variables $\{x_1, x_2, x_3\}$

type $(x_1) = \mathbb{B} = \{0, 1\}$

Boolean operators $\wedge \; \vee \; \neg \; \Rightarrow \; \Leftrightarrow$

Well-founded formula (informal) is a formula involving the variables and operators "properly"

Recall, a valuation x maps each $x_i \in X$ to $\{0, 1\}$

A valuation x satisfies α if each x_i in α replaced by $\mathsf{x} \lceil x_i$ evaluates to true.

We write this as $\mathsf{x} \models \alpha$

otherwise x does not satisfy α, $\mathsf{x} \not\models \alpha$

Example: $\langle x_1 \mapsto 1 \quad x_2 \mapsto 1 \quad x_3 \to 0 \rangle$ x

x satisfies α $\quad \mathsf{x} \models \alpha$

Def. Given a well-formed formula in propositional logic, determine whether there exists a satisfying solution (or valuation).

$$\exists x \in val(X) : x \models \alpha ?$$

if yes α is satisfiable
else α is unsatisfiable

if $\forall x \in val(x)$ $x \models \alpha$ then α is valid or a tautology.

if α is valid then $\neg\alpha$ is unsatisfiable.

α and α' are tautologically equivalent if they have the same truth tables

$$\forall x \in val(X) : x \models \alpha \Leftrightarrow x \models \alpha'$$

α and α' are equisatisfiable if
α is satisfiable iff α' is also satisfiable

How to solve SAT?

Build the truth table

k variable 2^k EXP

2-SAT is poly time

all clauses have at most 2 variables

3-SAT is NP-Complete [Cook '71, STOC]

- A solver for SAT can be used to solve any other problem in the NP-Class with only polynomial slow down
- Make sense to build SAT solvers
- Modern SAT solvers can solve problems with 10K+ variables and million + clauses

Z3, Yices, CVC4, maxSat, Chaff

How do SAT solvers work?

We assume α to be in Conjunctive normal form (CNF)

literals : variable or negation x_3 $\neg x_3$

Clause : disjunction (OR) of literals

$$x_1 \vee x_2 \vee \neg x_3$$

CNF : a formula Conjunction of Clauses

$$\alpha \quad (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_1)$$

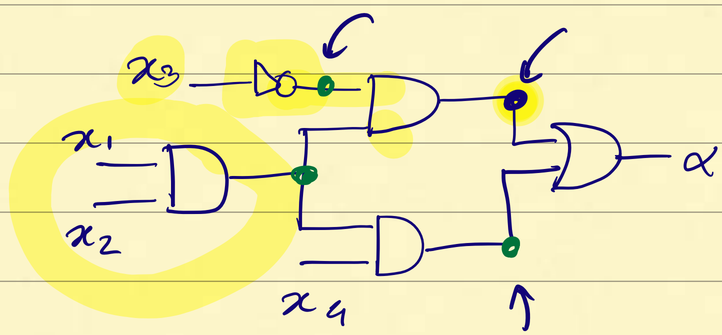
x_2 appears positively in the first clause
negatively in the second

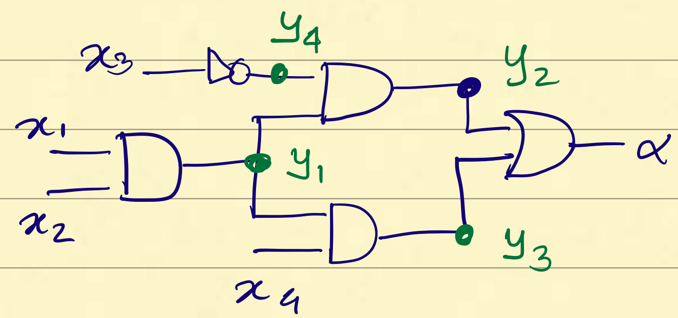
How to construct CNF?

Logic and Circuits

$$\alpha = (x_4 \wedge x_1 \wedge x_2) \vee (\neg x_3 \wedge x_1 \wedge x_2)$$

$$(x_4 \wedge (x_1 \wedge x_2)) \vee (\neg x_3 \wedge (x_1 \wedge x_2))$$





$y_4 \Leftrightarrow \neg x_3$

$y_4 \Rightarrow \neg x_3 \quad \wedge \quad \neg x_3 \Rightarrow y_4$

$(\neg y_4 \vee \neg x_3) \wedge (x_3 \vee y_4)$ — 1

$x_1 \wedge x_2 \Leftrightarrow y_1$

$(x_1 \wedge x_2 \Rightarrow y_1) \wedge (y_1 \Rightarrow x_1 \wedge x_2)$

$(\neg(x_1 \wedge x_2) \vee y_1) \wedge (\neg y_1 \vee (x_1 \wedge x_2))$

$(\neg x_1 \vee \neg x_2 \vee y_1) \wedge (\neg y_1 \vee x_1)$

$\wedge (\neg y_1 \vee x_2)$

— 2

$y_1 \wedge x_3 \Leftrightarrow y_3$

— 3

$\alpha \Leftrightarrow y_2 \wedge y_3$

— 4

—

$\alpha' \equiv 1 \quad \wedge \quad 2 \quad \wedge \quad 3 \quad \wedge \quad 4$

Standard representations of CNF

- $(\neg x_1 \vee \neg x_2 \vee x_5) \wedge (\neg x_5 \vee x_1)$
  $\wedge (\neg x_5 \vee x_2)$
- $(x_1' + x_2' + x_5)(x_5' + x_1)$
  $(x_5' + x_2)$
- (-1 -2 5)(-5 1)(-5 2)
  DIMACS
- SMTLib

GSAT

input : Clauses C over X

parameters $\underline{\text{max flips}}$, $\underline{\text{max-tries}}$

output : X satisfying α or ϕ

for $i = 1$ to max_tries

  $\underline{v} := $ random choice from val(X)

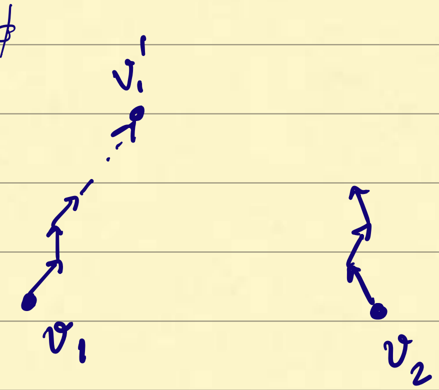
  for $j = 1$ to max flips

    if $v \models C$ return v

    $p :=$ variable in C s.t. flipping p gives the largest increase in # satisfied clauses

    $v := v$ with assignment to p flipped

return ϕ



Davis Putnam Logemann Loveland (DPLL) Algorithm 1962

- Transform the given formula α by applying a sequence of satisfiability preserving <u>rules</u>

- if final result has no literals $\rightarrow$ unsatisfiable
- if final result has no clauses $\rightarrow$ satisfiable

Davis Putnam Algorithm (DP) 1960

Rule 1. Unit propagation
Rule 2. Pure literal
Rule 3. Resolution

<u>Rule 1</u>. <u>Unit Prop</u>

if a clause has a single literal

$$\alpha \equiv \ldots (x_1 \vee \overset{0}{\neg p} \vee x_2) \wedge \overset{1}{\boxed{p}} \wedge \ldots \wedge (\neg x_3 \vee \overset{0}{\neg p} \vee x_1)$$

$\neg p$

$$\alpha' \equiv \ldots (x_1 \vee x_2) \wedge \ldots \wedge (\neg x_3 \vee x_1) \quad [p = 1]$$

α and α' are equisatisfiable.

Rule 2. Pure literal

A literal p appears only positively or negatively
set $p = 1$ (or 0) and remove all the
$\neg p$ occurrences

$$\alpha \equiv \cdots \wedge (x_1 \vee \neg p \vee x_2) \wedge (x_4 \vee \neg p) \wedge \cdots \wedge (\neg x_3 \vee x_1) \cdots$$

p does not appear anywhere

Makes sense to set $p = 0$

$$\alpha' \equiv \cdots \wedge \cdots \wedge \cdots (\neg x_3 \vee x_1) \cdots$$

Rule 3. Resolution

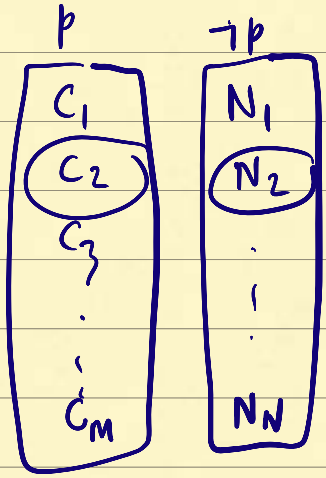
Choose a literal p that appears both +vely and
-vely : $(l_1 \vee l_2 \vee \cdots \vee p)$
$(k_1 \vee k_2 \vee \cdots \neg p)$

resolved clause $(l_1 \vee l_2 \vee \cdots \vee k_1 \vee k_2 \vee \cdots)$

Pairwise resolve each such pair

Take conjunction of all the resolve clauses.

- Why is the result equisatisfiable?



$$(C_1 \vee N_1) \wedge (C_1 \vee N_2) \wedge \dots$$
$$(C_2 \vee N_1) \wedge (C_2 \vee N_2) \dots$$
$$\vdots$$
$$(C_M, N_n)$$

What is the size of α after resolution?

$O(N^2)$

DPLL modifies resolution in DP
with DFS

Rule 3'. Let Δ be the current set of clauses
Choose a literal p in Δ
Check satisfiability of $\Delta \cup \{p\}$
(guess $p=1$)
if SAT then return True
else return result of Satisfiability
of $\Delta \cup \{\neg p\}$

Problem	tautology	dptaut	dplltaut
prime 3	0.00	0.00	0.00
prime 4	0.02	0.06	0.04
prime 9	18.94	2.98	0.51
prime 10	11.40	3.03	0.96
prime 11	28.11	2.98	0.51
prime 16	>1 hour	out of memory	9.15
prime 17	>1 hour	out of memory	3.87
ramsey 3 3 5	0.03	0.06	0.02
ramsey 3 3 6	5.13	8.28	0.31
mk_adder_test 3 2	>>1 hour	6.50	7.34
mk_adder_test 4 2	>>1 hour	22.95	46.86
mk_adder_test 5 2	>>1 hour	44.83	170.98
mk_adder_test 5 3	>>1 hour	38.27	250.16
mk_adder_test 6 3	>>1 hour	out of memory	1186.4
mk_adder_test 7 3	>>1 hour	out of memory	3759.9

From Slides of Clark Barrett's lecture.
Summer School on Verification Technology, Systems & Applications, September 17, 2008 – p. 42/98

Abstract DPLL as an automaton

States and transitions

$M \parallel F$

M: sequence of literals denoting partial assignment to variables

F: CNF formula being checked represented as list of clauses

initial state $\emptyset \parallel F$

final state : Fail (F is UNSAT)

$M \| G$ $\quad G$ equisatisfiable with F and $M \models G$

Transitions

Unit prop $\quad M \| F, C \vee \ell \longrightarrow M\ell \| F, C \vee \ell$

if ℓ is unassigned in M

and $\quad M \models \neg C$

Pure literal $\quad M \| F \longrightarrow M\ell \| F$

if ℓ is unassigned in M

ℓ occurs in some clause in F

$\neg \ell$ does not occur in F

Decide $\quad M \| F \longrightarrow M\ell^d \| F$

if ℓ is unassigned in M

both ℓ and $\neg\ell$ appear in some clauses in F

Backtrack $\quad M\ell^d N \| F, C \longrightarrow M \neg\ell \| F, C$

if $\quad M\ell^d N \models \neg C$

and $\quad N$ contains no decision literals

Fail $\quad M \| F, C \longrightarrow Fail$

if $\quad M \models \neg C \quad$ M satisfies $\neg C$

and M contains no decision literals

$x \models f$

x satisfies f

Example DPLL

$\phi \parallel 1 \vee \bar{2} \quad \bar{1} \vee \bar{2} \quad 2 \vee 3 \quad \bar{3} \vee 2 \quad 1 \vee 4$

Pure literal

$4 \parallel 1 \vee \bar{2} \quad \bar{1} \vee \bar{2} \quad 2 \vee 3 \quad \bar{3} \vee 2 \quad 1 \vee 4$

Decide

$4 1^d \parallel 1 \vee \bar{2} \quad \bar{1} \vee \bar{2} \quad 2 \vee 3 \quad \bar{3} \vee 2 \quad 1 \vee 4$

Unit prop

$4 1^d \bar{2} \parallel 1 \vee \bar{2} \quad \bar{1} \vee \bar{2} \quad 2 \vee 3 \quad \bar{3} \vee 2 \quad 1 \vee 4$

Unit prop

$4 1^d \bar{2} 3 \parallel 1 \vee \bar{2} \quad \bar{1} \vee \bar{2} \quad 2 \vee 3 \quad \bar{3} \vee 2 \quad 1 \vee 4$

$4 \bar{1} \parallel 1 \vee \bar{2} \quad \bar{1} \vee \bar{2} \quad 2 \vee 3 \quad \bar{3} \vee 2 \quad 1 \vee 4$

Unit prop

$4 \bar{1} \bar{2} \parallel 1 \vee \bar{2} \quad \bar{1} \vee \bar{2} \quad 2 \vee 3 \quad \bar{3} \vee 2 \quad 1 \vee 4$

Unit prop

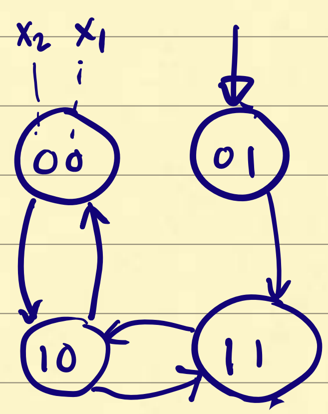
$4 \bar{1} \bar{2} 3 \parallel 1 \vee \bar{2} \quad \bar{1} \vee \bar{2} \quad 2 \vee 3 \quad \bar{3} \vee 2$

Fail

Automata, Reachability, and SAT.

Given an automaton $\mathcal{A} = \langle V, \Theta, A, \mathcal{D} \rangle$ and a candidate invariant $I \subseteq val(V)$ how to write the invariance check (Thm 7.1) as a satisfiability problem?

Example: Automaton



Variables

$x_1 : \mathbb{B}$

$x_2 : \mathbb{B}$ } To be filled

transitions

pre $x_1 \vee x_2$

eff $x_2 := 1$

pre $x_2 \wedge x_1$

eff $x_2 := 0$

Pre $\neg(x_1 \oplus x_2)$

eff $x_1 := \neg x_1$

$x_2 := \neg x_2$

Sample executions

$\langle 0,1\rangle \to \langle 1,1\rangle \to \langle 0,1\rangle \to \langle 0,1\rangle$
$\to \langle 1,1\rangle$

The transition relation of this automaton can be written a relation

$I(x_1,x_2)\ x_1 \oplus x_2$ $\qquad \mathcal{D} \subseteq Var(X) \times A \times Var(X)$
$\underline{\underline{Var(X) \times Var(X)}}$

$\Theta = x_1 \wedge \neg x_2$

$$F_D(x_1,x_2, x_1', x_2') \equiv (x_1 \vee x_2 \Rightarrow x_2' = 1 \wedge x_1' = x_1) \vee$$
$$(x_1 \wedge x_2 \Rightarrow x_2' = 0 \wedge x_1' = x_1) \vee$$
$$(\neg(x_1 \oplus x_2) \Rightarrow x_1' = \neg x_1 \wedge x_2' = \neg x_2)$$

Suppose I is an invraint
$$I(x_1, x_2) \equiv x_1 \oplus x_2$$

Checking the invariant can be stated as a SAT question.

$\forall x \ F_\Theta(x) \Rightarrow I(x) \wedge$ $\qquad$ (start condition)
$\forall x, x' \ F_D(x,x') \wedge I(x) \Rightarrow I(x')$
(transition cond)

To check validity of this statement we check the satisfiability of its negation

$$\exists x \quad F_\Theta(x) \wedge \neg I(x) \vee$$
$$\exists x, x' \quad F_D(x, x') \wedge I(x) \wedge \neg I(x') \quad \vee$$

SAT question:

$$\exists x, x' \left[F_\Theta(x) \wedge \neg I(x) \right] \vee$$
$$\left[F_D(x, x') \wedge I(x) \wedge \neg I(x') \right]$$