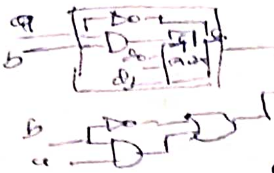
SAT (Satisfiability)

- $F(x_1, x_2 \cdots x_n) = F(\,) = 1$, satisfy → 1
- SAT → one satisfying assignment or no



∀ a, b, z [d0 d1, d2, d3] = 1 ?

equality? seen random repair network

Conjunctive Normal Form (CNF) = standard POS form

$\phi = (a+c)(b+c)(\neg a + \neg b + \neg c)$.

clause · positive literal · negative literal.

Suppose a=0, b=1, c&d unassigned

$\phi = (a+b')(a'+b+c')(a+b+d)(a'+b'+c')$.

conflicting clause=0 · clause=1 satisfied · unresolved · SAT

→ Recursively solve / Two big Ideas

Decision: simplify, assign & decide SAT

Deduction: Iteratively simplify, recurse & assign to other variables.

Boolean Constraint Propagation (BCP)

Deduction → BCP → propagating constraints.

★ Unit Clause Rule

→ one unassigned literal.

→ one way to be satisfied → pick polarity that makes clause = "1".

→ "Implication". assume a=1 & b=1

$\phi = (a'+c)(b'+c)(a'+b'+\bar{c})$.

c must be zero → SAT

$\phi = \omega_1 \omega_2 \omega_3 \omega_4 \omega_5 \omega_6 \omega_7 \omega_8 \omega_9$.

ω_1, ω_2 … ω_9 } make them all unit.

★ Boolean Constraint Propagation (BCP) for SAT

DPLL: Davis Putnam - Logemann Loveland Algo.

Smart BCP

Idea: systematic search of variable assignments. Useful CNF form for efficiency. BCP makes search stop earlier & resolving.

cover assignments w/o necessary more.

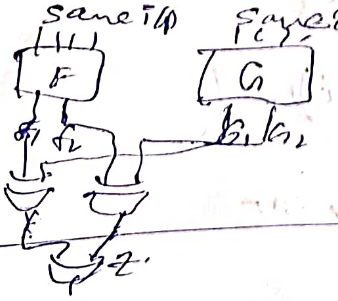
Good SAT: 50,000 vars, 25000,000 literals 50,000,000 clauses.

SAT solvers

- MiniSat
- CHAFF
- GRASP

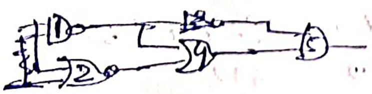
BDD	SAT
no guarantee can build it	no guarantee can build y/n on it.
can't build BDD (space) memory	can't find SAT run out of (time)
Full representation can build (∃xyz F & ∀xyz F)	do not represent full representation. quantified SAT solvers (∃xyz F), (∀xyz F) Only solve SAT

same i/p · same i/p



both same → 0 SAT, z=1; else → 1 UNSAT, z=0

Gates → CNF



CNF one gate at time

$\phi_d = [d == \overline{ab}] =$

$d \oplus (\overline{ab}) =$

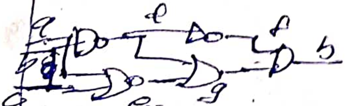
$(a+d)(b+d)(a'+b'+d') = \phi_d$.

XOR · XNOR
different · Equal

Gate consistency o/p that are "consistent"

a=0 b=0 d=1 → consistent d=1

a=1 b=1 d=1 → inconsistent d=0.



$\phi_g = g \oplus [d+e]$ $(d+g)(e+g)(d+e+\bar{g})$

SAT CNF:

ϕ = output var Π (x is gate output wire) ϕ_x

$\phi = h\,(a+d)(b+d)(a'+b'+d')$
$(b'+e)(c'+e)(b+c+e)$
$(d'+f)(e'+f)(d+e+f')$
$(d'+g)(e'+g)(d+e+g')$
$(f+h)(g+h)(f'+g'+h)$.

- Individual gate info
- And them get CNF

Gate consistency rules.

$z = OR$.

just wire

$[\bar{z} + z][x + \bar{z}]$

$z = NOR(x_1, x_2 \cdots x_n)$

$\left[\prod_{i=1}^{n} (\bar{x_i} + \bar{z})\right]\left[\left(\sum_{i=1}^{n} x_i\right) + z\right]$

product Sum

$z = OR(x_1, x_2, \cdots, x_n)$.

$\left[\prod_{i=1}^{n} (\bar{x_i} + z)\right]\left[\left(\sum_{i=1}^{n} x_i\right) + \bar{z}\right]$

$z = NOT(x)$.

$[x + z][\bar{x} + \bar{z}]$

$z = NAND(x_1, x_2, \cdots x_n)$.

$\left[\prod_{i=1}^{n} (x_i + z)\right]\left[\left(\sum_{i=1}^{n} \bar{x_i}\right) + \bar{z}\right]$

$z = AND(x_1, x_2, \cdots, x_n)$.

$\left[\prod_{i=1}^{n} (x_i + \bar{z})\right]\left[\left(\sum_{i=1}^{n} \bar{x_i}\right) + z\right]$

XOR/XNOR - unpleasant for SAT

$z = EXOR(a, b)$

$b_z = z \oplus (a \oplus b)$.

$= (z' + a' + b')(z' + a + b)(z + a' + b)$

$(z + a + b)$

#Idea: DPLL