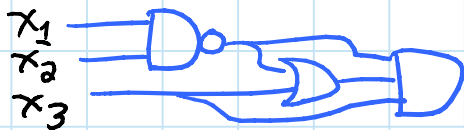


The most important NP-complete (logic) problem family!

SAT = Satisfiability: [Cook 1971; Levin 1973]

- given a Boolean formula (AND, OR, NOT) over n variables x_1, x_2, \dots, x_n
- can you set x_i 's to make formula true?

Circuit SAT: formula expressed as circuit of gates (allows re-use)



CNF SAT: formula = AND of clauses

[Cook 1971]

↓
Conjunctive
Normal Form

clause = OR of literals

literal $\in \{x_i, \text{NOT } x_i\}$

- can view as bipartite graph: variables vs. clauses, positive/negative edges

3SAT: clause = OR of 3 literals

[Cook 1971]

i.e. clause degrees = 3 (but allow repeats)

→ 3SAT-3: each variable occurs in ≤ 3 clauses

- E3SAT-4 but E3SAT-3 \in P

[Tovey-DAM 1984]

↳ exactly 3 distinct literals per clause

$\neg x_1 \vee x_2$
 $\neg x_2 \vee x_3$
 \vdots
 $\neg x_k \vee x_1$

Monotone 3SAT: [Gold-IC 1978]

each clause all positive or all negative,

Beware polynomial-time variants!

2SAT: clause = OR of 2 literals

- polynomial

- $x \text{ OR } y \equiv \text{NOT } x \Rightarrow y \quad (\equiv \text{NOT } y \Rightarrow x)$

- guess x_i , follow all implication chains to check OK

BUT...

Max 2SAT: set variables to maximize # true clauses

- NP-complete [Garey, Johnson, Stockmeyer 1976]

Horn SAT: each clause has ≤ 1 positive literal

- NOT x OR NOT y OR NOT z OR w

$\equiv \text{NOT } (x \text{ AND } y \text{ AND } z) \text{ OR } w$

$\equiv (x \text{ AND } y \text{ AND } z) \Rightarrow w$

\Rightarrow polynomial like 2SAT

[Horn 1951]

Dual-Horn SAT: each clause has ≤ 1 negative literal

\hookrightarrow "weakly positive satisfiability" [Schaefer 1978]

- negate all variables \rightarrow Horn SAT

\Rightarrow polynomial

DNF SAT: formula = OR of clauses

\downarrow
Disjunctive
Normal
Form

clause = AND of literals

\Rightarrow satisfiable $\Leftrightarrow \geq 1$ clause

Alternative clauses for 3SAT:

1-in-3SAT = exactly-1 3SAT [Schaefer 1978]

- clause = exactly 1 of 3 literals is true
(\Rightarrow 2 false \sim TFF, FTF, FFT)

omitted by Schaefer

Positive 1-in-3SAT: no negations - all literals positive

BUT... \rightarrow sometimes called "monotone"

Positive not-exactly-1 3SAT: [Schaefer 1978]

- clause = 0, 2, or 3 variables are true
i.e. $x_i \Rightarrow (x_j \text{ OR } x_k) \rightarrow$ Dual Horn
- also require $x_1 = \text{TRUE}$ (else set all $x_i = \text{FALSE}$)
& $x_2 = \text{FALSE}$ (or allow $|clause| \leq 3$)
- polynomial

NAE 3SAT = not-all-equal 3SAT [Schaefer 1978]

- clause = 3 literals not all the same value
(forbid FFF & TTT \Rightarrow 1 or 2 true, 2 or 1 false
 \sim whereas 3SAT forbids just FFF)
- nice symmetry between TRUE & FALSE

omitted by Schaefer

Positive NAE 3SAT: no negations - all literals positive

Schaefer's Dichotomy:

[Schaefer - STOC 1978]

- formula = AND of clauses
- general clause = relation on variables
 - assume in CNF (unique if minimal)

use Karnaugh maps

⇒ SAT is polynomial if either:

- setting all variables true or all variables false satisfies all relations
- subclauses are all Horn or all Dual Horn
- relations are all 2-CNF (subclause sizes ≤ 2)
- every relation can be expressed as a system of linear equations over \mathbb{Z}_2 :

"XOR SAT" $\left\{ \begin{array}{l} x_i \oplus x_j \oplus x_k \oplus x_l = 0 \text{ or } 1 \\ \quad \quad \quad \downarrow \text{XOR} \quad \quad \quad \rightarrow \text{Gaussian elimination} \end{array} \right.$

& otherwise, SAT is NP-complete!

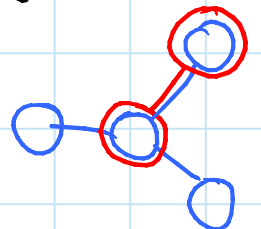
Another hard version of SAT - seldom used?

2-colorable perfect matching:

[Schaefer 1978]

- given a planar 3-regular graph
- 2-color the vertices such that every vertex has exactly 1 same-colored neighbor
- special case of 2-in-4 SAT

(planarity & 3-regular left as exercise)



Pushing blocks:

- 1×1 robot navigating grid of blocks
- goal: get robot from start to target
- Push- k : robot can push up to k blocks at once
- Push- $*$: infinite strength
- PushPush: blocks slide until they hit something
- PushPushPush: blocks slide other blocks (ice) in chain reaction, up to strength k
- Push...-F: some blocks are fixed
- Push...-X: robot path cannot self-intersect (tiles disappear after traversal)
- Sokoban = Push-1F but with goal of filling target squares with blocks

Push-*: reduction from 3SAT [Hoffmann 2000]

- variable: push right in x_i or \bar{x}_i row
→ fill in row of connection gadget
- connection: 1 free cell per occurrence of literal
- bridge: move up & block off leftward path
- clause: need a free spot below to traverse

PushPush-1 in 3D: reduction from 3SAT

[O'Rourke & Smith Problem Solving Group 1999]

(Push)Push-1 (in 2D): reduction from 3SAT

[Demaine, Demaine, O'Rourke 2000]

- clause gadget, block other, lock gadget
- XOR crossover: $N \rightarrow S$ xor $W \rightarrow E$
- unidir. crossover: optional $N \rightarrow S$, then $W \rightarrow E$

MIT OpenCourseWare
<http://ocw.mit.edu>

6.890 Algorithmic Lower Bounds: Fun with Hardness Proofs
Fall 2014

For information about citing these materials or our Terms of Use, visit: <http://ocw.mit.edu/terms>.