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

\[ \text{SAT} = \text{Satisfiability:} \quad [\text{Cook 1971; Levin 1973}] \]
- given a Boolean formula (AND, OR, NOT)
  over \( n \) variables \( x_1, x_2, \ldots, x_n \)
- can you set \( x_i \)'s to make formula true?

\[ \text{Circuit SAT: formula expressed as circuit of gates} \quad (\text{allows re-use}) \]

\[ \text{CNF SAT: formula = AND of clauses} \quad [\text{Cook 1971}] \]
- clause = OR of literals
  \[ \text{literal } \in \{ x_i, \text{ NOT } x_i \} \]
- can view as bipartite graph:
  variables vs. clauses, positive/negative edges

\[ \text{3SAT: clause = OR of 3 literals} \quad [\text{Cook 1971}] \]
- i.e. clause degrees = 3 (but allow repeats)

\[ \text{3SAT-3: each variable occurs in } \leq 3 \text{ clauses} \]
- \( \text{E3SAT-4} \) but \( \text{E3SAT-3} \in \text{P} \)  
  \( \text{[Tovey - DAM 1984]} \)

\[ \text{Monotone 3SAT:} \quad [\text{Gold - I&C 1978}] \]
- each clause all positive or all negative
Beware polynomial-time variants!

\textbf{2SAT}: clause $= \text{OR of 2 literals}$
- \underline{polynomial}
- $x \text{ or } y \equiv \neg x \Rightarrow y \ (\equiv \neg y \Rightarrow x)$
- guess $x_i$, follow all implication chains to check ok

\textbf{Max 2SAT}: set variables to maximize \# true clauses
- NP-complete [Garey, Johnson, Stockmeyer 1976]

\textbf{Horn SAT}: each clause has $\leq 1$ positive literal
- $\neg x \text{ or } \neg y \text{ or } \neg z \text{ or } w$
- $\equiv \neg (x \text{ and } y \text{ and } z) \Rightarrow w$
- $\equiv (x \text{ and } y \text{ and } z) \Rightarrow w$
- $\Rightarrow$ \underline{polynomial} like 2SAT

\textbf{Dual-Horn SAT}: each clause has $\leq 1$ negative literal
- "weakly positive satisfiability" [Schaefer 1978]
- negate all variables $\Rightarrow$ Horn SAT
- $\Rightarrow$ \underline{polynomial}

\textbf{DNF SAT}: formula $= \text{OR of clauses}$
- clause $= \text{AND of literals}$
- $\Rightarrow$ satisfiable $\iff \geq 1$ clause
**Alternative clauses for 3SAT:**

\[ 1\text{-in-3SAT} = \text{exactly-1 3SAT} \quad \text{[Schaefer 1978]} \]

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

*omitted by Schaefer*

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

*But... sometimes called “monotone”*

**Positive not-exactly-1 3SAT:**

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

\[ \text{NAE 3SAT} = \text{not-all-equal 3SAT} \quad \text{[Schaefer 1978]} \]

- clause = 3 literals not all the same value
  (forbid FFF & TTT \( \Rightarrow 1 \text{ or 2 true, 2 or 1 false} \)
  \sim \text{whereas 3SAT forbids just FFF} \)
- nice symmetry between \text{TRUE} & \text{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)
  \[\Rightarrow\] AND of subclauses
- 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 \(\leq 2\))
  - every relation can be expressed as a system of linear equations over \(\mathbb{Z}_2\):
  \[
  \text{"XOR SAT" } \iff \begin{cases} 
  x_i \oplus x_j \oplus x_k \oplus x_l = 0 \text{ or } 1 \\
  \text{Gaussian elimination}
  \end{cases}
  \]
  
  & 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-4SAT (planarity & 3-regular left as exercise)
Pushing blocks:
- 1x1 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 \[\text{Hoffmann} \ 2000\]
- variable: push right in \(x_i\) or \(\overline{x_i}\) row
  \(\rightarrow\) 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
- applies to PushPush-\* \& PushPushPushPush-\* too

\underline{PushPush-1} \ \text{in 3D}: \ reduction \ from \ 3SAT
\[\text{O’Rourke \ & \ Smith \ Problem \ Solving \ Group \ 1999}\]

(Push)\underline{Push-1} \ (in \ 2D): \ reduction \ from \ 3SAT
\[\text{Demaine, \ Demaine, \ O’Rourke \ 2000}\]
- clause gadget, block other, lock gadget
- XOR crossover: \(N \rightarrow S\) xor \(W \rightarrow E\)
- unidir. crossover: optional \(N \rightarrow S\), then \(W \rightarrow E\)