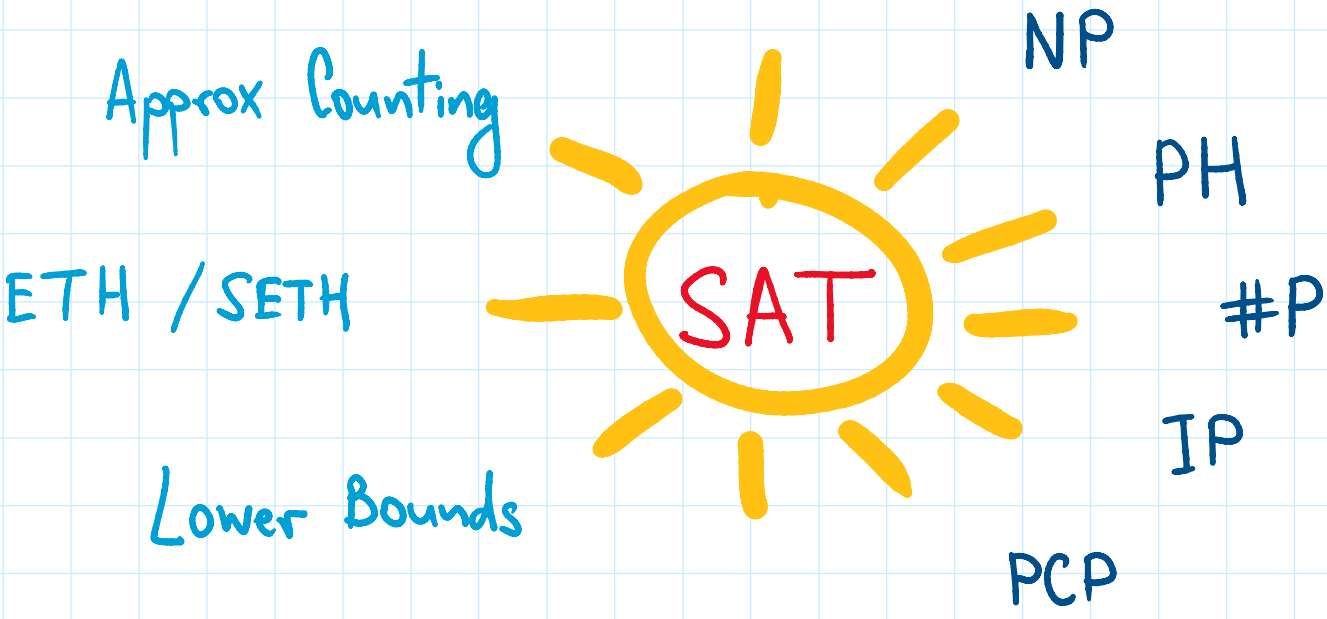


3-SAT : Is a given 3-CNF satisfiable?

$(x_3 \vee \bar{x}_7 \vee x_{13}) \wedge (x_1 \vee \bar{x}_3 \vee x_5) \wedge \ldots$

[k-CNF : each clause of size k]

Cook-Levin's Theorem ('71):

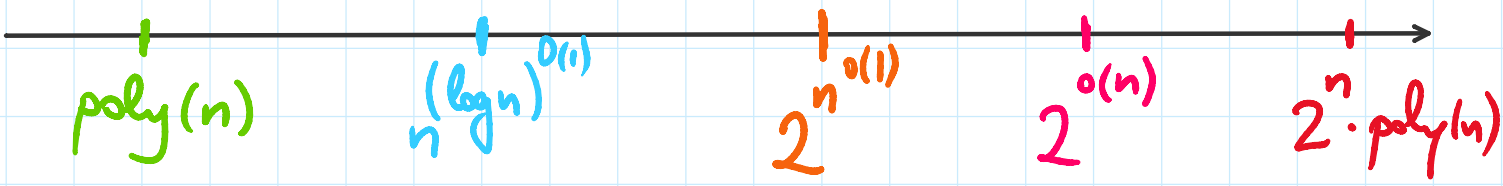
$3\text{-SAT} \in P \iff NP = P$

Million dollar question:

What is the complexity of 3-SAT?

K-SAT complexity

[n = # variables in a K-CNF]



P QP SUBEXP SUBE E

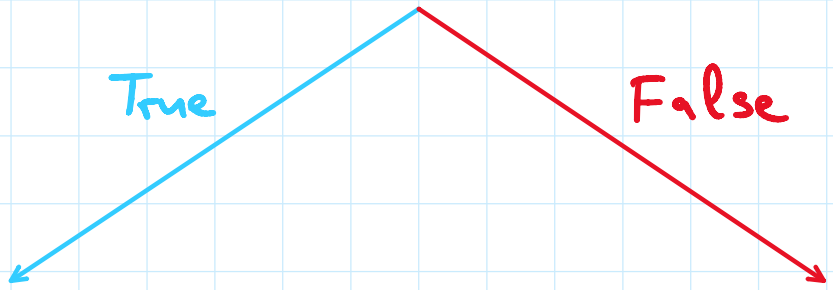
Fact: $k\text{-SAT} \in \text{RTime}\left(2^{n(1-\frac{1}{k})} \cdot \text{poly}(n)\right)$

[late 80s — now]

one such algo later...

ETH (Exponential Time Hypothesis) [Impagliazzo Paturi '99]

3-SAT cannot be solved in $\text{RTime}(2^{o(n)} \cdot \text{poly}(n))$



$P \neq NP$

& more

non-trivial algorithm for k-SAT & other problems

$$3\text{-SAT} \leq 4\text{-SAT} \leq 5\text{-SAT} \leq \dots$$

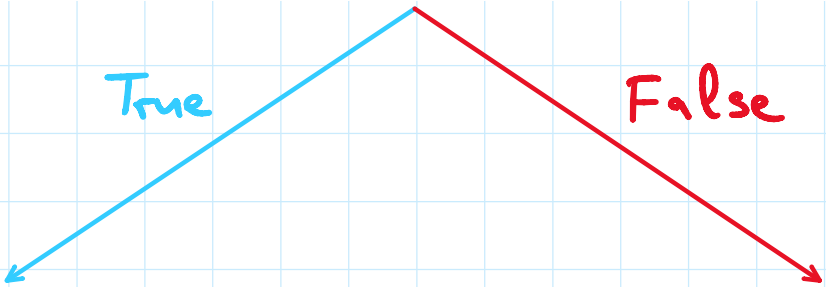
Is k-SAT getting strictly harder as k grows?

Yes! (assuming ETH) [IP '99]

Does k-SAT require time $2^n \cdot \text{poly}(n)$ as $k \to \infty$?

SETH (Strong ETH) [Calabro, Impagliazzo, Paturi '09]

$\forall\, 0 < \varepsilon < 1$, $\exists\, k$ k-SAT cannot be solved in $\text{RTime}\left(2^{(1-\varepsilon)\cdot n} \cdot \text{poly}(n)\right)$.



2^n-time lower bounds
"brute force is optimal"
for many NP-complete problems
eg. [Cygan et al. '12]
Exact Exponential Time Algos

optimality of many P-time algos (eg, n^2-time for Edit Distance [Backurs, Indyk '15])

Fine-Grained Complexity Theory

non-trivial algorithm for k-SAT & others

But, all known k-SAT algos are in $\text{RTime}(2^{n(1-\frac{c}{k})} \cdot \text{poly}(n))$

for constant $c > 0$.

Can we solve k-SAT in time $2^{n(1-\frac{\log^* k}{k})}$?

SSETH (Super-Strong ETH) [Williams '15, Vyas, Williams '19]

$\forall$ unbounded $f: \mathbb{N} \to \mathbb{N}$, $\exists\, k$ k-SAT cannot be solved in RTime$\left(2^{n(1-\frac{f(k)}{k})} \cdot \text{poly}(n)\right)$

- SSETH is false for random k-SAT
- SSETH is true for k-SAT $\Leftrightarrow$ it is true for Unique k-SAT

NSETH (Nondeterministic SETH)

[Carmosino et al. '16]

Important Tool for (S)ETH Results:

Sparsification Lemma

[Impagliazzo, Paturi, Zane '98]

Roughly speaking,
the hardest k-SAT instances are
k-CNFs on n variables
and $m = O(n)$ clauses
for $2^{o(n)}$-time algorithms.

Corollary [IPZ '98]: Fix any $k \geq 3$.

k-SAT $\in$ Time$(2^{o(m+n)} \cdot \text{poly}(n))$, $m = \#$ clauses, $n = \#$ variables

$\Rightarrow$

k-SAT $\in$ Time$(2^{o(n)} \cdot \text{poly}(n))$, $n = \#$ variables.

Application: $3\text{-SAT} \leq_p 3\text{-COL}$ [classical reduction]

3-CNF $\varphi(x_1, \ldots, x_n)$ on m clauses

$\xrightarrow{R}$

graph $G = (V, E)$, $|V| \in O(n+m)$

s.t.

φ is satisfiable $\iff$ G is 3-colourable

Hence, $3\text{-COL} \in P \Rightarrow 3\text{-SAT} \in P$.

But, $3\text{-COL} \in \text{Time}(2^{o(|V|)}) \overset{?}{\Rightarrow} 3\text{-SAT} \in \text{Time}(2^{o(n)})$

But, 3-COL $\in$ Time$(2^{o(|V|)}) \Rightarrow$ 3-SAT $\in$ Time$(2^{o(n+m)})$

3-COL $\in$ Time$(2^{o(|V|)})$

$\Rightarrow$ 3-SAT $\in$ Time$(2^{o(n+m)})$

$|V|$ = # vertices

n = # vars
m = # clauses

Sparsification $\Longrightarrow$ 3-SAT $\in$ Time$(2^{o(n)})$

$\Rightarrow$ $\neg$ETH

(same for many other NP-complete problems)

Sparsification Lemma [IPZ'98]: Fix any $k \geq 3$.

$\forall \varepsilon > 0$, $\exists$ $c = c(k, \varepsilon) \approx \frac{2^{2^k}}{\varepsilon}$

s.t. every k-CNF $\varphi(x_1, \ldots, x_n)$

$= \psi_1(x_1, \ldots, x_n) \vee \ldots \vee \psi_t(x_1, \ldots, x_n)$

for $t \leq 2^{\varepsilon n}$,

& each ψ_i: a k-CNF on $c \cdot n$ clauses.

Moreover, $\psi_1, \ldots, \psi_t$ computable in time $2^{\varepsilon n} \cdot \text{poly}(n)$.

Algorithm Sparsify (φ)

Input: k-clauses of a k-CNF $\varphi(x_1, \ldots, x_n)$

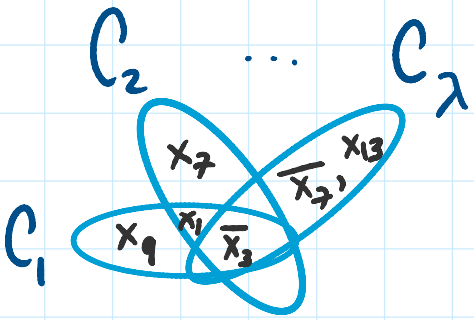
$x_1 \vee \bar{x}_3 \vee x_9 \quad \bar{x}_5 \vee \bar{x}_7 \vee x_{13} \quad \bar{x}_3 \vee x_5 \quad x_1 \vee \bar{x}_3 \vee x_7 \quad \bar{x}_3 \vee x_5 \vee x_7 \quad \ldots$

1. Eliminate every clause that contains some other clause.
   [clause subsumption]

$\{x_1, \bar{x}_3, x_9\} \quad \{\bar{x}_3, \bar{x}_7, x_{13}\} \quad \{\bar{x}_3, x_5\} \quad \{x_1, \bar{x}_3, x_7\} \quad \ldots$

2. Find a sunflower with

$$\text{Heart} := \bigcap_{i=1}^{\lambda} C_i$$



$\text{Petal}_i := C_i \setminus \text{Heart}$

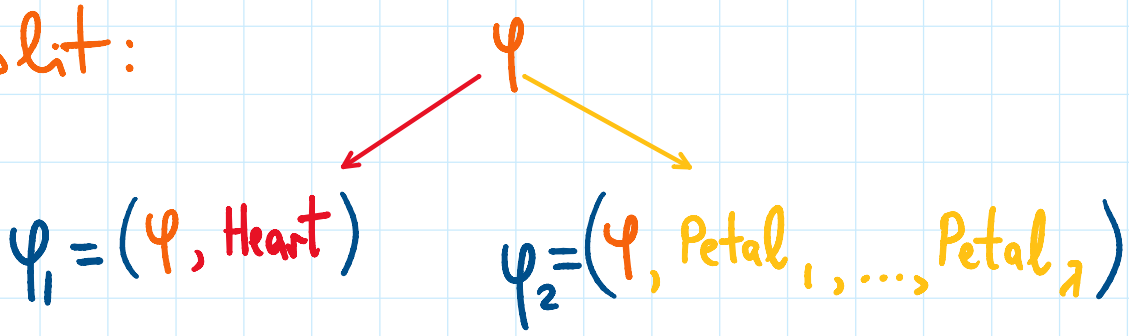
"large constant" λ.

$|C_1| = \ldots = |C_\lambda|$

favour
- short clauses
- large heart

If no sunflower, then output φ and halt.

3. Split:



4. Recurse: $\text{Sparsity}(\varphi_1)$; $\text{Sparsity}(\varphi_2)$

$$k\text{-SAT} \in \text{RTime}\left(2^{n(1-\frac{1}{k})} \cdot \text{poly}(n)\right)$$

Backtracking — Paturi, Pudlák, Zane '99 (PPZ)
Paturi, Pudlák, Saks, Zane '05 (PPSZ)

Local Search — Schöning '99

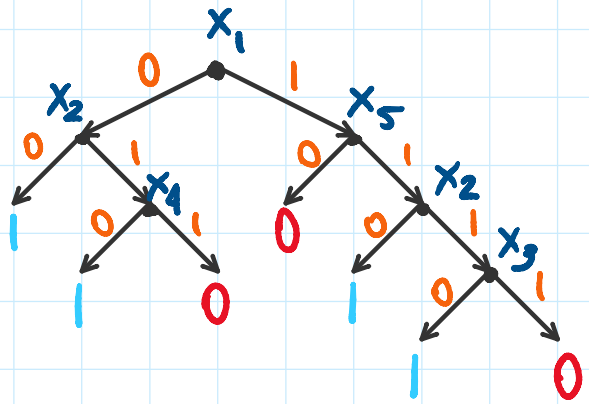
Polynomial Method — Chan, Williams '16

$\#k\text{-SAT} \in RTime\left(2^{n\left(1-\frac{1}{k}\right)} \cdot poly(n)\right)$

from the Switching Lemma

[Impagliazzo, Matthews, Paturi '12]

Decision Trees for $f(x_1, x_2, x_3, x_4, x_5)$



d-depth $\Delta T \Rightarrow$ d-DNF & d-CNF

Fact : $\#SAT(\text{d-depth } \Delta T) \in Time(2^d)$

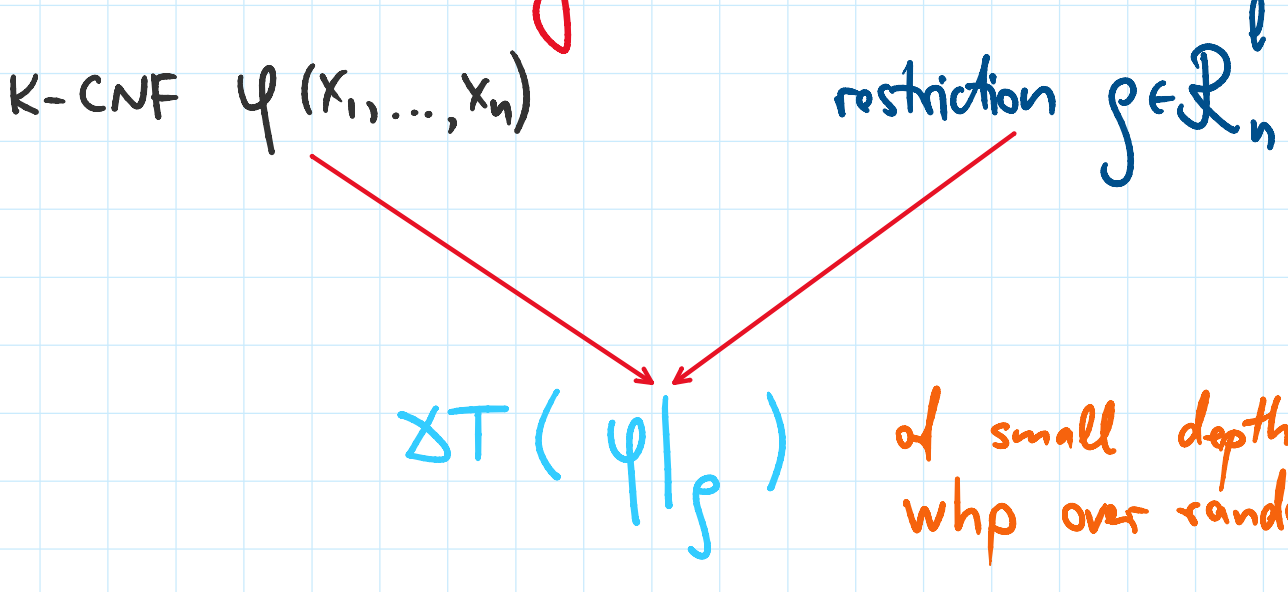
Proof: $\#SAT = \sum_{1\text{-branch}} 2^{n - |branch|}$. □

Random Restrictions $\rho \in R_n^\ell$

- pick a random $S \subseteq \{1, 2, \ldots, n\}$, $|S| = \ell$
- $\forall\ i \notin S$, pick a random $b_i \in \{0, 1\}$

- $\forall i \notin S$, pick a random $b_i \in \{0,1\}$
- define $\forall i,\ \rho(x_i) = \begin{cases} x_i & \text{if } i \in S \\ b_i & \text{if } i \notin S \end{cases}$

Switching Lemma



Håstad's Switching Lemma ('87): For any $0 < p < 1$,

for any k-CNF $\varphi(x_1, \ldots, x_n)$ $S > 0$

for any K-CNF $\varphi(x_1, \ldots, x_n)$, $s \geq 0$,

$$\Pr_{\rho \in R_n^{pn}} \left[\text{DT}(\varphi|_\rho) \text{ has depth } \geq s \right] \leq (7\rho K)^s$$

#K-SAT Algorithm

input: K-CNF $\varphi(x_1, \ldots, x_n)$

1. pick a random $S \subseteq \{1, \ldots, n\}$, $|S| = p \cdot n$ for $p = \frac{1}{15 \cdot K}$

2. compute

$$\sum_{b \in \{0,1\}^{n-pn}} \#\text{SAT}\left(\text{DT}(\varphi|_{S,b}) \right)$$

$$\varphi|_{S,b} = \varphi(x_{i \in S} ; x_{j \notin S} \leftarrow b)$$

Correctness: ✓

Expected Runtime:

$$\mathop{\mathrm{Exp}}_{\substack{S \subseteq \{1,\dots,n\} \\ |S| = \rho n}} \left[\sum_{b \in \{0,1\}^{n-\rho n}} \mathrm{Time}\left(\#\mathrm{SAT}\left(\Delta T\left(\varphi|_{S,b}\right)\right)\right) \right]$$

$$\leq 2^{n-\rho n} \cdot \mathop{\mathrm{Exp}}_{\rho \in \mathcal{R}_n^{\rho n}} \left[2^{\mathrm{depth}\left(\Delta T\left(\varphi|_\rho\right)\right)} \right]$$

$$= 2^{n-\rho n} \cdot \sum_{t \geq 1} \Pr_\rho \left[2^{\mathrm{depth}\left(\Delta T\left(\varphi|_\rho\right)\right)} \geq t \right]$$

$$\leq 2^{n-\rho n} \cdot \sum_{s \geq 0} 2^s \cdot \underbrace{\Pr_\rho \left[\mathrm{depth}\left(\Delta T\left(\varphi|_\rho\right)\right) \geq s \right]}_{\leq (7\rho k)^s = \left(\frac{7}{15}\right)^s}$$

$$\leq 2^{n-pn} \cdot \sum_{s \geq 0} \left(\frac{14}{15}\right)^s$$

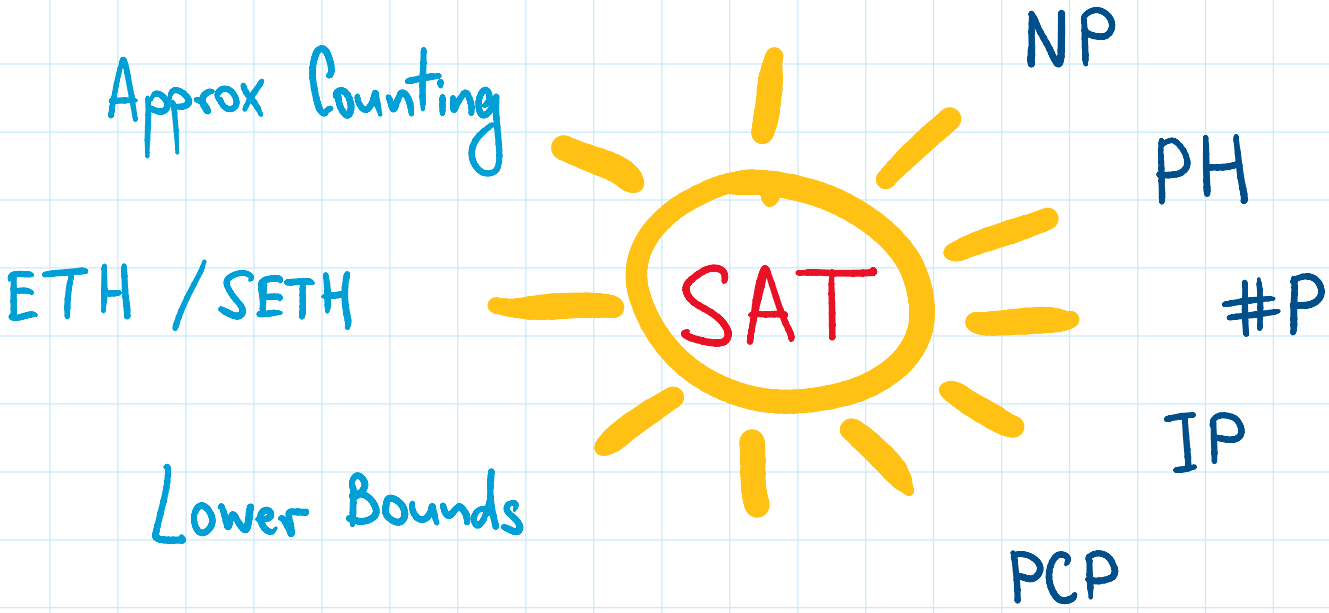
$$= 15 \cdot 2^{n\left(1-\frac{1}{15 \cdot k}\right)}.$$

Circuit Lower Bound Techniques $\Rightarrow$ SAT algorithms

"non-trivial" SAT algo for circuit class $\mathcal{C}$ $\Rightarrow$ $\mathcal{C}$-circuit lower bounds

$NQP \not\subseteq \mathcal{C}\text{-Size}(poly)$

$NQP \not\subseteq ACC^0$ [Murray, Williams '18]


NP
Approx Counting
PH
SAT
ETH / SETH
#P
IP
Lower Bounds
PCP