Algorithms from (proofs of)
Circuit Lower Bounds

Thm \[ \ldots; \] Impagliazzo, Matthews, Paturi
\[ k\text{-SAT} \in \text{ZPTime}(2^{n(1-\frac{1}{40k})}) \]
\[ (\forall \ : k\text{-CNF on n variables}) \]

Moreover, in the same time, can count \# satisfying assignments!

Håstad's Switching Lemma

Before: \[ \forall \ k, t, p, \]
\[ \forall k\text{-CNF } \Phi \text{ on n vars.} \]
\[ \exists k > 1 \text{ \& } t = O(NF_k^t) \]
$A \text{ Learns}$

$Pr \left[ \frac{\mathcal{G}}{\mathcal{S}} \right]$ is not a $t$-$\Sigma_N F^{\leq} (5p^k)^t$.

Actually works for a particular Simplification Procedure: $\mathcal{G}, \mathcal{S} \rightarrow \mathcal{G}'/\mathcal{S}$ building a $t$-$\Sigma_N F$ for $\mathcal{G}'/\mathcal{S}$.

**Decision Trees**

$$f(x, y, z, w)$$

$f(0, 0, *, *) = 0$

$f(0, 1, 0, *) = 1$

$f(1, *, 1, *) = 1$
Observation: For a depth- $d$ decision tree for $f$, $f$ has $d$-DNF (and $d$-CNF).

Proof: OR of all branches with a leaf 1 AND all literals on the branch ($\leq d$)

Canonical Decision Tree for

a given $K$-CNF $\mathcal{F}(x_1, \ldots, x_n)$

$\mathcal{F}: (x \lor \overline{y} \lor v \lor w) \land (y \lor z) \land (\overline{x} \lor \overline{v} \lor \overline{z})$

$\mathcal{F}$

$X=0$
$y=1$
$w=0$

$X=0$
$y=1$
$w=0$

$\overline{x} \lor \overline{v} \lor \overline{z}$

build DT for $\mathcal{F}$

Query vars
in clause 1

Query vars
in clause 2
not guarded before

$\overline{z}$
The Canonical DT for a given $\psi: \text{CST} (\psi)$.

Håstad's Switching Lemma (actual statement)

$\forall k, t, p. \quad \forall \text{K-CNF } \psi \text{ on } n \text{ vars.}$

$\Pr_{\pi \sim \mathcal{R}_p}[\text{depth of } \text{CST} (\psi | \pi ) > t ] < (5^p k)^t.$

\textbf{K-SAT Algorithm} ($p = \frac{1}{20 \cdot k}$)

\underline{Input:} \quad \text{K-CNF } \psi (x_1, \ldots, x_n)

\underline{Form a set } \quad S \subseteq \{1, 2, \ldots, n\}$.
1. Form a set $S = \{1, 2, \ldots, n\}$:
   
   for $i = 1$ to $n$
   put $i$ into $S$ with prob $p$

2. For every partial assignment $\alpha$ that sets all vars not in $S$,
   Check if $\varphi |\alpha$ (in vars from $S$)
   is satisfiable: by computing $CST(\varphi |\alpha)$.
   If discover that $\varphi$ is satisfiable,
   output 'Yes'; otherwise, 'No'.

Observation: Algo is always correct.
In the worst case, it looks at all $2^n$
assignments to see if $\varphi$ is satisfiable.

BUT, actually, quite often $\varphi |\alpha$
is much simpler & hence is faster to
check for being SAT.
Claim: The described randomized algo runs in Expected Time
\[ n \cdot (1 - \frac{1}{40.K}) \]
2

Proof: (over simplified)
For a given input \( k \text{-CNF } \phi(x_1, ..., x_n) \), define a random variable
\[ T(S) = \text{runtime of our algo given a random set } S \subseteq \{x_1, ..., x_n\} \]
Want to upper-bound:
\[ \mathbb{E}_S \left[ T(S) \right] \]
\[ T(S) = \sum_{b: |b| = n - 181} \left| \text{\# of } \phi \text{-tuples} \right| \cdot \mathbb{P}(b) \]
b: \(|b| = n - |S|\) (assert to vars outside of \(S\))

SAT - check via CST construction

\[
\begin{align*}
\text{Assume } |S| &= n \cdot p \quad (\text{the expected size}) \quad \text{\text{]} } \\
T(S) &\leq 2^{n - np} \cdot \exp \left[ \frac{2 \cdot \text{depth-CST}(\psi | S, b)}{b: \quad |b| = n - np} \right] \cdot \text{poly}(n)
\end{align*}
\]

\[
\begin{align*}
\exp \left[ T(S) \right] &\leq S \quad \text{poly}(n) \cdot \exp \left[ 2^{n(1 - p)} \cdot \exp \left[ 2 \cdot \text{depth-CST}(\psi | S, b) \right] \right]
\end{align*}
\]

\[
\begin{align*}
&= \text{poly}(n) \cdot 2^{n(1-p)} \cdot \exp \left[ 2 \cdot \text{depth-CST}(\psi | S, b) \right] \\
&= \text{poly}(n) \cdot 2^{n(1-p)} \cdot \exp \left[ 2 \cdot \text{depth-CST}(\psi | b) \right]
\end{align*}
\]
\[ \text{Fact: For every random variable } X \text{ s.t. } (1) \; X \geq 0 \; \text{ and } (2) \; X \text{ is integer-valued,} \]

\[ \mathbb{E}[X] = \sum_{i=0}^{\infty} \Pr[X > i] . \]

(by Fact)

\[ = \text{poly}(n), \quad 2^{-u(1-\rho)} . \]

\[ = \sum_{i=1}^{\mathbb{N}^8} \Pr[\text{depth-}\text{CST}(y|_s) > \log_2 i] \]

\[ \leq (5^p \rho K)^{\log_2 i} \]

\[ \text{Set } \rho = \frac{1}{20 \cdot K} \text{. Then } (5^p \rho K) = \frac{1}{4} . \]

\[ \leq \frac{8 \log_2 i}{i^2} \leq O(1) \]
\[ \sum_{i=1}^{\pi^2} (\text{Eqn}) \]

Overall,
\[ \text{Exp} \left[ T(S) \right] \leq p^{-\left(\frac{4}{11}\right)} \cdot 2^{n(1-p)} \cdot O(1) \]
\[ = \text{poly}(n) \cdot 2^{\left(1 - \frac{1}{20.4} \right) n} \]

This concludes the proof, subject to our over-simplifying assumption that \( |S| = p \cdot n \).

Actually, by Chernoff, \( |S| \geq \frac{p \cdot n}{2} \), w.h.p. + extra (technical) work \( \Rightarrow \) rigorous proof.

Remark: We can modify the described \( k \)-SAT algorithm so that it counts \( \# \text{ sat assign.} \)
it sees!

The expected run time stays the same!

Last time we saw:

Algo: always correct, expected runtime $T$

$\downarrow$

Algo: zero-error, runtime $\leq 4T$

Corollary: $\#K$-SAT is in $ZPTIME(2^{\frac{n}{4}}(1-\frac{1}{40k})^n)$.

Musings
**K-SAT** is (randomized) time \(2^{n(1 - \frac{c}{k})}\) (const \(c > 0\))

with a number of different algorithms "whatever algo strategy you dry, you only get \(2^{n(1 - \frac{c}{k})}\) time"

Maybe it's a correct answer?

**Exponential-Time Hypothesis (ETH)**

K-SAT requires time 2:

\[
\forall k \geq 3, \exists \delta = \delta(k) > 0 \text{ s.t. } \text{K-SAT} \notin \text{BPTime}(2^{\delta n})
\]

**Strong ETH (SETH):**

\[
n(1 - o(1))
\]
S'kong ETH (SETH): \[ n(1-o(1)) \]

\[ k\text{-SAT requires Time } 2^{\cdot \cdot \cdot} \]
as \[ k \to \infty \]

\[ 8(k) \to 1 \]

ETH & SETH strengthen P\neq\text{NP}

assumption:

many implication on precise time complexity (even for problems in P).