Back to k-SAT

Easier cases?

Thm Any k-CNF formula s.t.
  each var occurs \leq \frac{2^k}{ek^2}
  is satisfiable.

Pf Idea - probabilistic method
  i.e. take random assignment!
  but how to analyze?

Lovasz Local Lemma (’77)

Let \( E_1, \ldots, E_m \) be events s.t.
  each \( E_i \) depends on \leq d \) other events.
  Suppose \( \Pr(E_i) \leq \frac{1}{ed} \).

Then \( \Pr(\bigcap_{i=1}^m E_i) > 0 \).

(if completely independent)
( in general, if \( \Pr(E_i) < \frac{1}{d} \), can apply union bound)

Apply LLL with \( E_i = \{ i^{th} \text{ clause is satisfied} \} \)
  \( d \leq k \cdot \frac{2^k}{ek^2} = \frac{2^k}{e} \).
  \( \Pr(E_i) = \frac{1}{2^k} \leq \frac{1}{ed} \).
  \( \Pr(\bigcap_{i=1}^m E_i) > 0 \).

Pf: 
Claim \( \forall \text{ subset } S \subseteq \{1, \ldots, m\} \)
  \( \Pr(E_i^c \mid \bigcap_{j \in S} E_j) \leq \alpha \)
  (for some fixed \( \alpha \) with \( p \leq \alpha < 1 \))

LLL follows since
  \( \Pr(\bigcap_{i=1}^m E_i) = \Pr(E_1) \Pr(E_2 \mid E_1) \Pr(E_3 \mid E_1 \wedge E_2) \cdots \)
  \( \geq (1-\alpha) (1-\alpha) (1-\alpha) \cdots \)
  \( = (1-\alpha)^m \geq 0 \).

Proof - Claim: Pf. induction on \( |S| \)
\[ = (1 - \alpha)^m > 0 \rightarrow (1 - \frac{1}{d}) \]

Pf of Claim: By induction on \( |S| \).

\( S \neq \emptyset \): \( \Pr(E_i^c) \leq p \leq \alpha \checkmark \)

Fix \( i \). Let \( A = \{ j \in S : E_j \text{ depends on } E_i \} \)
\( B = S - A \).

\[ \Pr\left( E_i^c \mid \bigcap_{j \in B} E_j \right) = \frac{\Pr\left( \bigcap_{j \in A} E_j \cap \bigcap_{j \in B} E_j \right)}{\Pr\left( \bigcap_{j \in A} E_j \right)} \leq \frac{p}{(1 - \alpha)^k} \]

numerator \[ \leq \Pr\left( E_i^c \mid \bigcap_{j \in B} E_j \right) \]
\[ = \Pr(E_i^c) \leq p. \]

denominator = \[ \Pr\left( E_i \mid \bigcap_{j \in B} E_j \right) \cdot \Pr\left( E_i^c \mid \bigcap_{j \in B} E_j \right). \]

Write \( A = \{ i_1, \ldots, i_k \} \) \( k \leq d \).

\[ \Pr\left( E_{i_1} \mid E_{i_1} \cap E_{i_2} \cap \bigcap_{j \in B} E_j \right) \cdots \geq (1 - \alpha)(1 - \alpha) \cdots \text{ by ind. hyp.} \]
\[ = (1 - \alpha)^k \]
\[ \geq (1 - \alpha)^d \]

\[ \geq \Pr\left( E_i^c \mid \bigcap_{j \in S} E_j \right) \leq \frac{p}{(1 - \alpha)^d} = \alpha \]

by choosing \( p = \alpha (1 - \alpha)^d \).

Set \( \alpha \approx \frac{1}{d} \) and \( p = \frac{1}{d} (1 - \frac{1}{d})^d \rightarrow \frac{1}{e.d}. \)

\( \square \)

(General "asymmetric" form of LLL:
\( \forall \alpha_i \in (0,1) \) st.
Suppose $\exists \alpha_i \in (0, 1)$ s.t.
$$\Pr(E_i) = \alpha_i \prod (1-\alpha_j) .$$
\[ j: E_j \text{ depends on } E_i. \]

Then $$\Pr(\bigcap_{i=1}^{m} E_i) \geq \prod_{i=1}^{m} (1-\alpha_i) .$$

But above pf does not yield efficient algm
even with rand.

\(\text{(need } (\frac{1}{1-\nu_d})^m \text{ trials!)}\)
\(\text{worse than brute force!}\)

**Moser's Rand. Algm (08)**

Given $k$-CNF formula $F$ s.t.
each var occurs $\leq \frac{2^k}{c_0 k}$ times

\[ \text{Solve}(F): \]
1. pick rand. assignment
2. while $\exists$ unsatisfied clause $C$
3. $\text{Fix}(C)$

\[ \text{Fix}(C): \]
\[ \text{// make } C\text{ satisfied but keep all prev. satisfied clauses satisfied} \]
1. replace all $k$ vars in $C$ with new rand. values
2. while $\exists$ clause $C'$ that shares var with $C$ & is not satisfied
3. $\text{Fix}(C')$

if it terminates,
it finds sat. assignment
But will it terminate?

Analysis:

Idea - counting argument

Suppose \( \Pr[\text{alg needs } \geq t \text{ calls to Fix}] \geq \frac{1}{2} \).
Run algm & stop after \( t \) calls to Fix.

Define \( X = \) sequence of random bits used by algm

\[ = \begin{cases} \text{init. rand bits} & \text{assignment} \\ n + kt & \text{k random bits per call to Fix} \end{cases} \]

Define \( Y = \) sequence of the clauses fixed together with final assignment

\[ = \begin{cases} \text{init. rand bits} & \leq m((\log m + o(1)) + t(\log d + o(1)) + n \end{cases} \]

- from Solve, \( \leq m \) choices
- from Fix, \( \leq t \) choices
- extra bit to indicate end of recursive call

Obs: From \( Y \), can uniquely recover \( X \).

Pr: Just run algm backwards!

\[ = \begin{cases} \text{# possible } X \text{'s} & \leq \# possible } Y \text{'s} \end{cases} \]

\[ \leq \frac{1}{2} \cdot 2^{n + kt} \leq 2^{m((\log m + o(1)) + t(\log d + o(1)) + n} \]

\[ \leq m((\log m + o(1)) + t(\log d + o(1)) + n} \]

\[ \leq O(m \log m) \]
Set $d = \frac{2^k}{c_0}$

$(k - (\log d - O(1))) t \leq O(m \log m)$

$\Rightarrow t \leq O(m \log m)$

$\Rightarrow$ algorithm finds a satisfying assignment in polynomial time with probability $\geq \frac{1}{2}$. □