The Probabilistic Method

Janabel Xia and Tejas Gopalakrishna

MIT PRIMES Reading Group, mentors Gwen McKinley and Jake Wellens

December 7th, 2018

Introduction

What is the Probabilistic Method?

Introduction

What is the Probabilistic Method?

Basically, to show an object with a certain property exists, it suffices to show that an object drawn from a particular distribution over objects has the desired property with positive probability. This is often easier than explicitly constructing such an object (and sometimes the only way we know how to prove one exists!)

Basic Application: Turán's Theorem

- Consider a graph $G = (V, E)$.
- Let d_v be the degree of vertex v.
- Let $\alpha(G)$ be the size of the maximal independent set of vertices.

Basic Application: Turán's Theorem

- Consider a graph $G = (V, E)$.
- Let d_v be the degree of vertex v.
- Let $\alpha(G)$ be the size of the maximal independent set of vertices.

Turán's theorem gives a lower bound on $\alpha(G)$ for graphs with $|E|$ edges. Its proof is a classic example of the probabilistic method in action:

Basic Application: Turán's Theorem

- Consider a graph $G = (V, E)$.
- Let d_v be the degree of vertex v.
- Let $\alpha(G)$ be the size of the maximal independent set of vertices.

Turán's theorem gives a lower bound on $\alpha(G)$ for graphs with $|E|$ edges. Its proof is a classic example of the probabilistic method in action:

Theorem (Turán)

$$\alpha(G) \geq \sum_{v \in V} \frac{1}{d_v + 1} \geq \frac{n}{1 + \frac{2|E|}{n}}$$

Basic Application: Turán's Theorem

Theorem

$$\alpha(G) \geq \sum_{v \in V} \frac{1}{d_v + 1} \geq \frac{n}{1 + \frac{2|E|}{n}}$$

(second inequality is just convexity, we'll prove the first)

Proof:

Basic Application: Turán's Theorem

Theorem

$$\alpha(G) \geq \sum_{v \in V} \frac{1}{d_v + 1} \geq \frac{n}{1 + \frac{2|E|}{n}}$$

(second inequality is just convexity, we'll prove the first)

Proof:

- Let $<$ be a uniformly random linear order of V.

Basic Application: Turán's Theorem

Theorem

$$\alpha(G) \geq \sum_{v \in V} \frac{1}{d_v + 1} \geq \frac{n}{1 + \frac{2|E|}{n}}$$

(second inequality is just convexity, we'll prove the first)

Proof:

- Let $<$ be a uniformly random linear order of V.
- Define the independent set

$$I = I(<) := \{v \in V : \{v, w\} \in E \Rightarrow v < w\}.$$

(two neighbors cannot both be the "smallest" in their neighborhoods $\implies$ I is indep. set)

Basic Application: Turán's Theorem

Theorem

$$\alpha(G) \geq \sum_{v \in V} \frac{1}{d_v + 1} \geq \frac{n}{1 + \frac{2|E|}{n}}$$

(second inequality is just convexity, we'll prove the first)

Proof:

- Let $<$ be a uniformly random linear order of V.
- Define the independent set

$$I = I(<) := \{v \in V : \{v, w\} \in E \Rightarrow v < w\}.$$

  (two neighbors cannot both be the "smallest" in their neighborhoods $\implies$ I is indep. set)
- Let X_v be the indicator variable for the event $\{v \in I\}$, and set

$$X = \sum_{v \in V} X_v = |I|$$

Basic Application: Turán's Theorem

Proof: (cont.)

- For each v,

$$\mathrm{E}[X_v] = \Pr[v \in I] = \frac{1}{d_v + 1},$$

  because $v \in I$ iff v is least among v and its d_v neighbors.

Basic Application: Turán's Theorem

Proof: (cont.)

- For each v,

$$\mathrm{E}[X_v] = \Pr[v \in I] = \frac{1}{d_v + 1},$$

  because $v \in I$ iff v is least among v and its d_v neighbors.

- So

$$\mathrm{E}[X] = \sum_{v \in V} \frac{1}{d_v + 1}$$

  and therefore there exists an ordering $<$ with

$$|I(<)| \geq \sum_{v \in V} \frac{1}{d_v + 1}.$$

■

Another Basic Application: Increasing subsequences in a matrix

Problem

Determine the smallest $k = k(n)$ such that:

For any n by n matrix A with distinct entries, there is a permutation of the rows of A so that no column in the permuted matrix contains an increasing subsequence of length k.

Another Basic Application: Increasing subsequences in a matrix

Problem

Determine the smallest $k = k(n)$ such that:

For any n by n matrix A with distinct entries, there is a permutation of the rows of A so that no column in the permuted matrix contains an increasing subsequence of length k.

Lower bound: $k(n) \geq \sqrt{n}$.

Another Basic Application: Increasing subsequences in a matrix

Problem

Determine the smallest $k = k(n)$ such that:

For any n by n matrix A with distinct entries, there is a permutation of the rows of A so that no column in the permuted matrix contains an increasing subsequence of length k.

Lower bound: $k(n) \geq \sqrt{n}$.

Theorem (Erdös-Szekeres, 1935)

Any sequence of $n^2 + 1$ *distinct reals contains either an increasing or decreasing* $(n + 1)$*-subsequence.*

Another Basic Application: Increasing subsequences in a matrix

Problem

Determine the smallest $k = k(n)$ such that:

For any n by n matrix A with distinct entries, there is a permutation of the rows of A so that no column in the permuted matrix contains an increasing subsequence of length k.

Lower bound: $k(n) \geq \sqrt{n}$.

Theorem (Erdös-Szekeres, 1935)

Any sequence of $n^2 + 1$ distinct reals contains either an increasing or decreasing $(n+1)$-subsequence.

Consider a matrix whose first column is in the reverse relative order of the second column. Then for any permutation of rows, either the first or second column contains an increasing subsequence of length $\geq \sqrt{n}$.

Another Basic Application: Increasing subsequences in a matrix

Lower bound already holds for $n \times 2$ matrices – is it much harder to avoid increasing subsequences among n columns?

Another Basic Application: Increasing subsequences in a matrix

Lower bound already holds for $n \times 2$ matrices – is it much harder to avoid increasing subsequences among n columns?...Not really!

Another Basic Application: Increasing subsequences in a matrix

Lower bound already holds for $n \times 2$ matrices – is it much harder to avoid increasing subsequences among n columns?...Not really!
Upper bound: There exists $C > 0$ such that $k(n) \leq C\sqrt{n}$.

Another Basic Application: Increasing subsequences in a matrix

Lower bound already holds for $n \times 2$ matrices – is it much harder to avoid increasing subsequences among n columns?...Not really!

Upper bound: There exists $C > 0$ such that $k(n) \leq C\sqrt{n}$.

Proof: Consider a random permutation σ of the rows. Let $LIS(c)$ be the length of the largest increasing subsequence in the column vector c.

Consider each column separately:

Another Basic Application: Increasing subsequences in a matrix

Lower bound already holds for $n \times 2$ matrices – is it much harder to avoid increasing subsequences among n columns?...Not really!

Upper bound: There exists $C > 0$ such that $k(n) \leq C\sqrt{n}$.

Proof: Consider a random permutation σ of the rows. Let $LIS(c)$ be the length of the largest increasing subsequence in the column vector c.

Consider each column separately:

$$\Pr_{\sigma}[1 \ldots 2 \ldots 3 \ldots k] = \frac{1}{k!}$$

Another Basic Application: Increasing subsequences in a matrix

Lower bound already holds for $n \times 2$ matrices – is it much harder to avoid increasing subsequences among n columns?...Not really!
Upper bound: There exists $C > 0$ such that $k(n) \leq C\sqrt{n}$.
Proof: Consider a random permutation σ of the rows. Let $LIS(c)$ be the length of the largest increasing subsequence in the column vector c. Consider each column separately:

$$\Pr_{\sigma}[1 \ldots 2 \ldots 3 \ldots k] = \frac{1}{k!}$$

$$\implies \Pr_{\sigma}[LIS(c) \geq k] \leq \binom{n}{k} \frac{1}{k!}$$

Proof of upper bound

- $\Pr_\sigma[LIS(c) \geq k] \leq \binom{n}{k}\frac{1}{k!}$

Proof of upper bound

- $\Pr_\sigma[LIS(c) \geq k] \leq \binom{n}{k}\frac{1}{k!}$
- Use standard inequalities $\binom{n}{k} \leq \left(\frac{en}{k}\right)^k$ and $m! > (m/e)^m$

Proof of upper bound

- $\Pr_\sigma[LIS(c) \geq k] \leq \binom{n}{k}\frac{1}{k!}$
- Use standard inequalities $\binom{n}{k} \leq \left(\frac{en}{k}\right)^k$ and $m! > (m/e)^m$
- for $k = C\sqrt{n}$,

$$\Pr[LIS(c) \geq C\sqrt{n}\,] \leq \left(\frac{en}{C\sqrt{n}}\right)^{C\sqrt{n}} \frac{1}{(C\sqrt{n})!}$$

Proof of upper bound

- $\Pr_\sigma[LIS(c) \geq k] \leq \binom{n}{k}\frac{1}{k!}$
- Use standard inequalities $\binom{n}{k} \leq \left(\frac{en}{k}\right)^k$ and $m! > (m/e)^m$
- for $k = C\sqrt{n}$,

$$\Pr[LIS(c) \geq C\sqrt{n}\,] \leq \left(\frac{en}{C\sqrt{n}}\right)^{C\sqrt{n}} \frac{1}{(C\sqrt{n})!}$$

$$\leq \left(\frac{en}{C\sqrt{n}}\right)^{C\sqrt{n}} \left(\frac{e}{C\sqrt{n}}\right)^{C\sqrt{n}} = \left(\frac{e}{C}\right)^{2C\sqrt{n}}$$

Proof of upper bound

- $\Pr_\sigma[LIS(c) \geq k] \leq \binom{n}{k}\frac{1}{k!}$
- Use standard inequalities $\binom{n}{k} \leq \left(\frac{en}{k}\right)^k$ and $m! > (m/e)^m$
- for $k = C\sqrt{n}$,

$$\Pr[LIS(c) \geq C\sqrt{n}\,] \leq \left(\frac{en}{C\sqrt{n}}\right)^{C\sqrt{n}} \frac{1}{(C\sqrt{n})!}$$

$$\leq \left(\frac{en}{C\sqrt{n}}\right)^{C\sqrt{n}} \left(\frac{e}{C\sqrt{n}}\right)^{C\sqrt{n}} = \left(\frac{e}{C}\right)^{2C\sqrt{n}}$$

- Then by a union bound over all columns:

$$\Pr[LIS(c) \geq C\sqrt{n} \text{ for at least one column}] \leq n\left(\frac{e}{C}\right)^{2C\sqrt{n}} < 1$$

(for sufficiently large C).

Proof of upper bound

- $\Pr_\sigma[LIS(c) \geq k] \leq \binom{n}{k} \frac{1}{k!}$
- Use standard inequalities $\binom{n}{k} \leq \left(\frac{en}{k}\right)^k$ and $m! > (m/e)^m$
- for $k = C\sqrt{n}$,

$$\Pr[LIS(c) \geq C\sqrt{n}\,] \leq \left(\frac{en}{C\sqrt{n}}\right)^{C\sqrt{n}} \frac{1}{(C\sqrt{n})!}$$

$$\leq \left(\frac{en}{C\sqrt{n}}\right)^{C\sqrt{n}} \left(\frac{e}{C\sqrt{n}}\right)^{C\sqrt{n}} = \left(\frac{e}{C}\right)^{2C\sqrt{n}}$$

- Then by a union bound over all columns:

$$\Pr[LIS(c) \geq C\sqrt{n} \text{ for at least one column}] \leq n\left(\frac{e}{C}\right)^{2C\sqrt{n}} < 1$$

(for sufficiently large C). So with positive probability over σ, $LIS(c) \leq C\sqrt{n}$ for all columns. ■

Random Graphs

(Erdös-Renyi) Random graph $G(n, p)$:

- graph on n labeled vertices
- each edge appears independently with probability p.

Random Graphs

(Erdös-Renyi) Random graph $G(n,p)$:

- graph on n labeled vertices
- each edge appears independently with probability p.
- **Question:** How big does $p = p(n)$ have to be in order for a typical $G(n,p)$ to contain a clique of size 4?

Random Graphs

(Erdös-Renyi) Random graph $G(n, p)$:

- graph on n labeled vertices
- each edge appears independently with probability p.
- **Question:** How big does $p = p(n)$ have to be in order for a typical $G(n, p)$ to contain a clique of size 4?
- **First moment:** Expected number of cliques of size 4 is $\binom{n}{4} p^6$, so if $p \ll n^{-2/3}$, then

$$\Pr[G(n, p) \text{ has a 4-clique }] \leq \mathrm{E}[\text{number of 4-cliques}] \to 0$$

Random Graphs

(Erdös-Renyi) Random graph $G(n, p)$:

- graph on n labeled vertices
- each edge appears independently with probability p.
- **Question:** How big does $p = p(n)$ have to be in order for a typical $G(n, p)$ to contain a clique of size 4?
- **First moment:** Expected number of cliques of size 4 is $\binom{n}{4}p^6$, so if $p \ll n^{-2/3}$, then

$$\Pr[G(n,p) \text{ has a 4-clique }] \leq \mathrm{E}[\text{number of 4-cliques}] \rightarrow 0$$

- But is $p > n^{-2/3}$ enough to guarantee a 4-clique? Need to use the **second moment**.

The Second Moment Method

- Let X_i be indicator random variables for "symmetric" events A_i, and set $X = \sum_i X_i$.

The Second Moment Method

- Let X_i be indicator random variables for "symmetric" events A_i, and set $X = \sum_i X_i$.
- Write $i \sim j$ if A_i and A_j are not independent, and let

$$\Delta^* = \sum_{i \sim j} \Pr[A_j | A_i]$$

(which is independent of i by symmetry)

The Second Moment Method

- Let X_i be indicator random variables for "symmetric" events A_i, and set $X = \sum_i X_i$.
- Write $i \sim j$ if A_i and A_j are not independent, and let

$$\Delta^* = \sum_{i \sim j} \Pr[A_j | A_i]$$

(which is independent of i by symmetry)

Lemma

$\Pr[X = 0] \leq \frac{1+\Delta^*}{\mathrm{E}[X]}$.

(Proof is a fairly straightforward application of Chebyshev's inequality)

Cliques in $G(n, p)$

Theorem

If $p(n) \cdot n^{2/3} \to \infty$, *then* $\Pr[G(n,p) \text{ has a 4-clique }] \to 1$

Cliques in $G(n, p)$

Theorem

If $p(n) \cdot n^{2/3} \to \infty$, then $\Pr[G(n,p)$ has a 4-clique $] \to 1$

Proof:

- For each 4-set S of vertices in $G \sim G(n,p)$, let A_S be the event that S is a clique, let X_S be its indicator random variable, and set $X = \sum_{|S|=4} X_S$ to be the number of 4-cliques in G.

Cliques in $G(n, p)$

Theorem

If $p(n) \cdot n^{2/3} \to \infty$, then $\Pr[G(n, p)$ has a 4-clique $] \to 1$

Proof:

- For each 4-set S of vertices in $G \sim G(n, p)$, let A_S be the event that S is a clique, let X_S be its indicator random variable, and set $X = \sum_{|S|=4} X_S$ to be the number of 4-cliques in G.
- Then, $\mathrm{E}[X_S] = \Pr[A_S] = p^6$ and so

$$\mathrm{E}[X] = \sum_{|S|=4} \mathrm{E}[X_S] = \binom{n}{4} p^6 \sim \frac{n^4 p^6}{24} \to \infty$$

Cliques in $G(n, p)$

Theorem

If $p(n) \cdot n^{2/3} \to \infty$, then $\Pr[G(n,p)$ has a 4-clique $] \to 1$

Proof:

- For each 4-set S of vertices in $G \sim G(n,p)$, let A_S be the event that S is a clique, let X_S be its indicator random variable, and set $X = \sum_{|S|=4} X_S$ to be the number of 4-cliques in G.
- Then, $\mathrm{E}[X_S] = \Pr[A_S] = p^6$ and so

$$\mathrm{E}[X] = \sum_{|S|=4} \mathrm{E}[X_S] = \binom{n}{4} p^6 \sim \frac{n^4 p^6}{24} \to \infty$$

- By the lemma, it now suffices to show that $\Delta^* \ll n^4 p^6$.

Cliques in $G(n, p)$

Proof: (cont.)

- If S and T are 4-sets, then $S \sim T$ iff $S \neq T$ and S, T have common edges (i.e. $|S \cap T| = 2$ or 3).

Cliques in $G(n, p)$

Proof: (cont.)

- If S and T are 4-sets, then $S \sim T$ iff $S \neq T$ and S, T have common edges (i.e. $|S \cap T| = 2$ or 3).
- Fix S. There are $O(n^2)$ sets T with $|S \cap T| = 2$, and $O(n)$ with $|S \cap T| = 3$.

Cliques in $G(n, p)$

Proof: (cont.)

- If S and T are 4-sets, then $S \sim T$ iff $S \neq T$ and S, T have common edges (i.e. $|S \cap T| = 2$ or 3).
- Fix S. There are $O(n^2)$ sets T with $|S \cap T| = 2$, and $O(n)$ with $|S \cap T| = 3$.
- For each type of T, $\Pr[A_T | A_S] = p^5$ or p^3 respectively.

Cliques in $G(n, p)$

Proof: (cont.)

- If S and T are 4-sets, then $S \sim T$ iff $S \neq T$ and S, T have common edges (i.e. $|S \cap T| = 2$ or 3).
- Fix S. There are $O(n^2)$ sets T with $|S \cap T| = 2$, and $O(n)$ with $|S \cap T| = 3$.
- For each type of T, $\Pr[A_T | A_S] = p^5$ or p^3 respectively.
- So (since $p \gg n^{-2/3}$),

$$\Delta^* = O(n^2 p^5) + O(np^3) = o(n^4 p^6) = o(\mathrm{E}[X])$$

as needed. ■

k-SAT

- Suppose we have a k-CNF, i.e. an AND of n OR clauses on k Boolean variables each, e.g.

$$(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_5) \wedge (x_2 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee x_5 \vee x_6)$$

k-SAT

- Suppose we have a k-CNF, i.e. an AND of n OR clauses on k Boolean variables each, e.g.

  $$(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_5) \wedge (x_2 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee x_5 \vee x_6)$$

- Can we satisfy all clauses by assigning TRUE or FALSE to each x_i?

k-SAT

- Suppose we have a k-CNF, i.e. an AND of n OR clauses on k Boolean variables each, e.g.

$$(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_5) \wedge (x_2 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee x_5 \vee x_6)$$

- Can we satisfy all clauses by assigning TRUE or FALSE to each x_i?
- (Cook-Levin) Finding a satisfying assignment (or even deciding if one exists) for general k-CNFs is NP-complete (i.e. hopelessly hard)

k-SAT

- Suppose we have a k-CNF, i.e. an AND of n OR clauses on k Boolean variables each, e.g.

$$(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_5) \wedge (x_2 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee x_5 \vee x_6)$$

- Can we satisfy all clauses by assigning TRUE or FALSE to each x_i?
- (Cook-Levin) Finding a satisfying assignment (or even deciding if one exists) for general k-CNFs is NP-complete (i.e. hopelessly hard)
- What if each variable appears in a bounded number of clauses?

k-SAT

- Suppose we have a k-CNF, i.e. an AND of n OR clauses on k Boolean variables each, e.g.

$$(x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_5) \wedge (x_2 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee x_5 \vee x_6)$$

- Can we satisfy all clauses by assigning TRUE or FALSE to each x_i?
- (Cook-Levin) Finding a satisfying assignment (or even deciding if one exists) for general k-CNFs is NP-complete (i.e. hopelessly hard)
- What if each variable appears in a bounded number of clauses?
- The probabilistic tool we need is the **Lovász Local Lemma!**

The (Symmetric) Local Lemma

Theorem (Lovász, 1975)

Let $A_1, A_2, \ldots, A_n$ *be events in a probability space. Suppose* **each event is independent of all but at most** d **others**, *and that* $\Pr[A_i] \leq p$ *for all* $1 \leq i \leq n$. *If*

$$ep(d+1) \leq 1$$

then

$$\Pr\left[\bigwedge_{i=1}^{n} \overline{A_i}\right] > 0.$$

(i.e. with positive probability, no event A_i *holds).*

k-SAT with bounded occurrences

- Let

$$\phi = (x_1 \vee \cdots \vee x_3) \wedge (\neg x_{10} \vee \cdots \vee x_5) \wedge \cdots \wedge (x_{20} \vee \cdots \vee \neg x_{14})$$

be some k-CNF.

k-SAT with bounded occurrences

- Let

$$\phi = (x_1 \vee \cdots \vee x_3) \wedge (\neg x_{10} \vee \cdots \vee x_5) \wedge \cdots \wedge (x_{20} \vee \cdots \vee \neg x_{14})$$

  be some k-CNF.
- The probability that a random assignment leaves clause i unsatisfied is 2^{-k} (call this event A_i)

k-SAT with bounded occurrences

- Let

$$\phi = (x_1 \vee \cdots \vee x_3) \wedge (\neg x_{10} \vee \cdots \vee x_5) \wedge \cdots \wedge (x_{20} \vee \cdots \vee \neg x_{14})$$

  be some k-CNF.
- The probability that a random assignment leaves clause i unsatisfied is 2^{-k} (call this event A_i)
- Suppose each variable in ϕ appears in at most ℓ clauses.

k-SAT with bounded occurrences

- Let

$$\phi = (x_1 \vee \cdots \vee x_3) \wedge (\neg x_{10} \vee \cdots \vee x_5) \wedge \cdots \wedge (x_{20} \vee \cdots \vee \neg x_{14})$$

  be some k-CNF.
- The probability that a random assignment leaves clause i unsatisfied is 2^{-k} (call this event A_i)
- Suppose each variable in ϕ appears in at most ℓ clauses.
- Then each A_i is dependent on at most $k(\ell - 1)$ other A_j.

k-SAT with bounded occurrences

- Let

$$\phi = (x_1 \vee \cdots \vee x_3) \wedge (\neg x_{10} \vee \cdots \vee x_5) \wedge \cdots \wedge (x_{20} \vee \cdots \vee \neg x_{14})$$

  be some k-CNF.
- The probability that a random assignment leaves clause i unsatisfied is 2^{-k} (call this event A_i)
- Suppose each variable in ϕ appears in at most ℓ clauses.
- Then each A_i is dependent on at most $k(\ell - 1)$ other A_j.
- If

$$\ell \leq \frac{2^k}{ek}$$

  then $e2^{-k}(k(\ell - 1) + 1) < 1$ and hence the local lemma says that ϕ is satisfiable!

.

k-SAT with bounded occurrences

We've just shown

Theorem

If ϕ is a k-CNF in which each variable shows up at most $\frac{2^k}{ek}$ times, then ϕ has a satisfying assignment.

...how tight is this?

k-SAT with bounded occurrences

We've just shown

Theorem

If ϕ is a k-CNF in which each variable shows up at most $\frac{2^k}{ek}$ times, then ϕ has a satisfying assignment.

...how tight is this?

- consider the k-CNF on k variables with each of the 2^k possible clauses

k-SAT with bounded occurrences

We've just shown

Theorem

If ϕ is a k-CNF in which each variable shows up at most $\frac{2^k}{ek}$ times, then ϕ has a satisfying assignment.

...how tight is this?

- consider the k-CNF on k variables with each of the 2^k possible clauses
- unsatisfiable $\implies$ cannot replace $\frac{2^k}{ek}$ with 2^k

k-SAT with bounded occurrences

We've just shown

Theorem

If ϕ is a k-CNF in which each variable shows up at most $\frac{2^k}{ek}$ times, then ϕ has a satisfying assignment.

...how tight is this?

- consider the k-CNF on k variables with each of the 2^k possible clauses
- unsatisfiable $\implies$ cannot replace $\frac{2^k}{ek}$ with 2^k
- a more involved construction of Gebauer, Szabó and Tardos (2016) shows that $\frac{2^k}{ek}$ cannot be replaced with $(2 + o_k(1))\frac{2^k}{ek}$

k-SAT with bounded occurrences

We've just shown

Theorem

If ϕ is a k-CNF in which each variable shows up at most $\frac{2^k}{ek}$ times, then ϕ has a satisfying assignment.

...how tight is this?

- consider the k-CNF on k variables with each of the 2^k possible clauses
- unsatisfiable $\implies$ cannot replace $\frac{2^k}{ek}$ with 2^k
- a more involved construction of Gebauer, Szabó and Tardos (2016) shows that $\frac{2^k}{ek}$ cannot be replaced with $(2 + o_k(1))\frac{2^k}{ek}$
- can actually be improved to $2 \cdot \frac{2^k}{ek}$ using lopsided local lemma

Finding a satisfying assignment

- Let ϕ be a k-CNF with n clauses in which each variable shows up at most $\frac{2^k}{ek}$ times, which we now know is satisfiable... how can we find a satisfying assignment?

Finding a satisfying assignment

- Let ϕ be a k-CNF with n clauses in which each variable shows up at most $\frac{2^k}{ek}$ times, which we now know is satisfiable... how can we find a satisfying assignment?
- Brute force: try all possible assignments to the variables (could be as many as 2^{nk} of these to try!)

Finding a satisfying assignment

- Let ϕ be a k-CNF with n clauses in which each variable shows up at most $\frac{2^k}{ek}$ times, which we now know is satisfiable... how can we find a satisfying assignment?
- Brute force: try all possible assignments to the variables (could be as many as 2^{nk} of these to try!)
- A slightly (but not much) more intelligent algorithm:

Finding a satisfying assignment

- Let ϕ be a k-CNF with n clauses in which each variable shows up at most $\frac{2^k}{ek}$ times, which we now know is satisfiable... how can we find a satisfying assignment?
- Brute force: try all possible assignments to the variables (could be as many as 2^{nk} of these to try!)
- A slightly (but not much) more intelligent algorithm:
  - start with uniformly random truth assignment of all variables
  - pick at random any unsatisfied clause C
  - give all x_i in C new random assignments
  - repeat until all clauses are satisfied
- is this efficient?

Finding a satisfying assignment

- Let ϕ be a k-CNF with n clauses in which each variable shows up at most $\frac{2^k}{ek}$ times, which we now know is satisfiable... how can we find a satisfying assignment?
- Brute force: try all possible assignments to the variables (could be as many as 2^{nk} of these to try!)
- A slightly (but not much) more intelligent algorithm:
  - start with uniformly random truth assignment of all variables
  - pick at random any unsatisfied clause C
  - give all x_i in C new random assignments
  - repeat until all clauses are satisfied
- is this efficient?

Theorem (Moser, Tardos 2010)

The expected number of times this algorithm has to loop before finding a satisfying assignment is $\lesssim \frac{n}{2^k}$.

Acknowledgements

We would like to thank:

- Gwen McKinley and Jake Wellens, our mentors
- Dr. Tanya Khovanova
- Dr. Slava Gerovitch
- MIT PRIMES
- Noga Alon and Joel H. Spencer, for writing *The Probabilistic Method*
- Our families, for all their support