Randomized Algorithms, Lecture 8

Jacob Holm (jaho@di.ku.dk)

May 20th 2019

Good morning.

Today's Lecture

- The probabilistic method
  - Overview
  - MAX-SAT
  - OR-concentrator
  - Probability Amplification

The probabilistic method: Core ideas

1. Any random variable $X \in \mathbb{R}$ takes some value $\leq \mathbb{E}[X]$ and some value $\geq \mathbb{E}[X]$.
2. If a random object taken from a universe U has nonzero probability of satisfying a property P, then there must be an object in U satisfying P.

The probabilistic method: Core ideas

1. Any random variable $X \in \mathbb{R}$ takes some value $\leq \mathbb{E}[X]$ and some value $\geq \mathbb{E}[X]$.
2. If a random object taken from a universe U has nonzero probability of satisfying a property P, then there must be an object in U satisfying P.

The probabilistic method: Core ideas

1. Any random variable $X \in \mathbb{R}$ takes some value $\leq \mathbb{E}[X]$ and some value $\geq \mathbb{E}[X]$.
2. If a random object taken from a universe U has nonzero probability of satisfying a property P, then there must be an object in U satisfying P.

The probabilistic method: Examples

- Since the *expected* size of the autopartition generated by RandAuto from Lecture 1 is $\mathcal{O}(n \log n)$, there *always exists* an autopartition of size $\mathcal{O}(n \log n)$.
- Since the *expected* number of leaves inspected by RandBoolGTE on any instance of $T_{2,k}$ is $3^k = n^{0.793\ldots}$ (where $n = 4^k$), there *always exists* a set of at most $3^k = n^{0.793\ldots}$ leaves that *certify* the value.

The probabilistic method: Examples

- Since the *expected* size of the autopartition generated by RANDAUTO from Lecture 1 is $\mathcal{O}(n \log n)$, there *always exists* an autopartition of size $\mathcal{O}(n \log n)$.
- Since the *expected* number of leaves inspected by RANDBOOLGTE on any instance of $T_{2,k}$ is $3^k = n^{0.793\ldots}$ (where $n = 4^k$), there *always exists* a set of at most $3^k = n^{0.793\ldots}$ leaves that *certify* the value.

The probabilistic method: Examples

- Since the *expected* size of the autopartition generated by $\textsc{RandAuto}$ from Lecture 1 is $\mathcal{O}(n \log n)$, there *always exists* an autopartition of size $\mathcal{O}(n \log n)$.
- Since the *expected* number of leaves inspected by $\textsc{RandBoolGTE}$ on any instance of $T_{2,k}$ is $3^k = n^{0.793\ldots}$ (where $n = 4^k$), there *always exists* a set of at most $3^k = n^{0.793\ldots}$ leaves that *certify* the value.

The probabilistic method: Max-cut

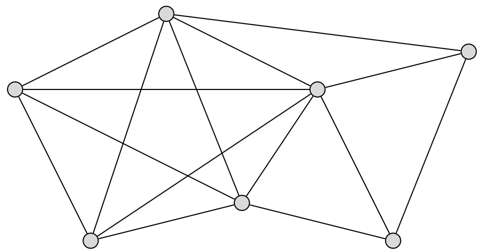
Theorem

For any undirected graph $G = (V, E)$, there is a partition of V into sets A, B such that the set C of edges with one end in A and one in B has $|C| \geq \frac{|E|}{2}$.

The probabilistic method: Max-cut

Theorem

For any undirected graph $G = (V, E)$, there is a partition of V into sets A, B such that the set C of edges with one end in A and one in B has $|C| \geq \frac{|E|}{2}$.

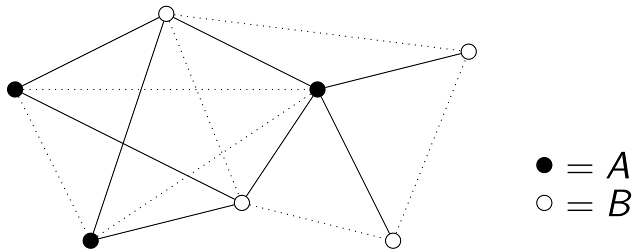


This graph has 15 edges. The theorem says that we can partition its vertices into sets A, B so that at least half (i.e. 8) of the edges go between the two sets.

The probabilistic method: Max-cut

Theorem

For any undirected graph $G = (V, E)$, there is a partition of V into sets A, B such that the set C of edges with one end in A and one in B has $|C| \geq \frac{|E|}{2}$.



Partitioning its vertices in to sets A and B as shown leaves $8 \geq \frac{15}{2}$ edges crossing the cut between A and B.

The probabilistic method: Max-cut

Theorem

For any undirected graph $G = (V, E)$, there is a partition of V into sets A, B such that the set C of edges with one end in A and one in B has $|C| \geq \frac{|E|}{2}$.

Proof.

Imagine that we have independently assigned each v in V to A or B with equal probability. Then for any edge $e \in E$, $\Pr[e \in C] = \frac{1}{2}$. So

$$\mathbb{E}[|C|] = \mathbb{E}\left[\sum_{e \in E} [e \in C]\right] = \sum_{e \in E} \mathbb{E}[[e \in C]] = \sum_{e \in E} \Pr[e \in C] = \frac{|E|}{2}$$

But then *some* partition of V into A and B has $|C| \geq \mathbb{E}[|C|] = \frac{|E|}{2}$. $\square$

The probabilistic method: Max-cut

Theorem

For any undirected graph $G = (V, E)$, there is a partition of V into sets A, B such that the set C of edges with one end in A and one in B has $|C| \geq \frac{|E|}{2}$.

Proof.

Imagine that we have independently assigned each v in V to A or B with equal probability. Then for any edge $e \in E$, $\Pr[e \in C] = \frac{1}{2}$. So

$$\mathbb{E}[|C|] = \mathbb{E}\left[\sum_{e \in E} [e \in C]\right] = \sum_{e \in E} \mathbb{E}[[e \in C]] = \sum_{e \in E} \Pr[e \in C] = \frac{|E|}{2}$$

But then *some* partition of V into A and B has $|C| \geq \mathbb{E}[|C|] = \frac{|E|}{2}$. □

Let $e = (u, v)$. No matter what set u is assigned to, since they were assigned independently the probability that v is in the other set is $\frac{1}{2}$.

The probabilistic method: Max-cut

Theorem

For any undirected graph $G = (V, E)$, there is a partition of V into sets A, B such that the set C of edges with one end in A and one in B has $|C| \geq \frac{|E|}{2}$.

Proof.

Imagine that we have independently assigned each v in V to A or B with equal probability. Then for any edge $e \in E$, $\Pr[e \in C] = \frac{1}{2}$. So

$$\mathbb{E}[|C|] = \mathbb{E}\Big[\sum_{e \in E} [e \in C]\Big] = \sum_{e \in E} \mathbb{E}[[e \in C]] = \sum_{e \in E} \Pr[e \in C] = \frac{|E|}{2}$$

But then *some* partition of V into A and B has $|C| \geq \mathbb{E}[|C|] = \frac{|E|}{2}$. □

The size of a set $X \subseteq Y$ can always be described as $|X| = \sum_{y \in Y} [y \in X]$, i.e. as the sum over all $y \in Y$ of the indicator variable $[y \in X]$. This is an extremely useful trick.

The probabilistic method: Max-cut

Theorem

For any undirected graph $G = (V, E)$, there is a partition of V into sets A, B such that the set C of edges with one end in A and one in B has $|C| \geq \frac{|E|}{2}$.

Proof.

Imagine that we have independently assigned each v in V to A or B with equal probability. Then for any edge $e \in E$, $\Pr[e \in C] = \frac{1}{2}$. So

$$\mathbb{E}[|C|] = \mathbb{E}\left[\sum_{e \in E} [e \in C]\right] = \sum_{e \in E} \mathbb{E}[[e \in C]] = \sum_{e \in E} \Pr[e \in C] = \frac{|E|}{2}$$

But then *some* partition of V into A and B has $|C| \geq \mathbb{E}[|C|] = \frac{|E|}{2}$. $\square$

Linearity of expectation.

The probabilistic method: Max-cut

Expectation of indicator variable.

Theorem

For any undirected graph $G = (V, E)$, there is a partition of V into sets A, B such that the set C of edges with one end in A and one in B has $|C| \geq \frac{|E|}{2}$.

Proof.

Imagine that we have independently assigned each v in V to A or B with equal probability. Then for any edge $e \in E$, $\Pr[e \in C] = \frac{1}{2}$. So

$$\mathbb{E}[|C|] = \mathbb{E}\left[\sum_{e \in E}[e \in C]\right] = \sum_{e \in E} \mathbb{E}[[e \in C]] = \sum_{e \in E} \Pr[e \in C] = \frac{|E|}{2}$$

But then *some* partition of V into A and B has $|C| \geq \mathbb{E}[|C|] = \frac{|E|}{2}$. $\square$

The probabilistic method: Max-cut

Theorem

For any undirected graph $G = (V, E)$, there is a partition of V into sets A, B such that the set C of edges with one end in A and one in B has $|C| \geq \frac{|E|}{2}$.

Proof.

Imagine that we have independently assigned each v in V to A or B with equal probability. Then for any edge $e \in E$, $\Pr[e \in C] = \frac{1}{2}$. So

$$\mathbb{E}[|C|] = \mathbb{E}\left[\sum_{e \in E} [e \in C]\right] = \sum_{e \in E} \mathbb{E}[[e \in C]] = \sum_{e \in E} \Pr[e \in C] = \frac{|E|}{2}$$

But then *some* partition of V into A and B has $|C| \geq \mathbb{E}[|C|] = \frac{|E|}{2}$. $\square$

The probabilistic method: Max-cut

First principle of the probabilistic method.

Theorem

For any undirected graph $G = (V, E)$, there is a partition of V into sets A, B such that the set C of edges with one end in A and one in B has $|C| \geq \frac{|E|}{2}$.

Proof.

Imagine that we have independently assigned each v in V to A or B with equal probability. Then for any edge $e \in E$, $\Pr[e \in C] = \frac{1}{2}$. So

$$\mathbb{E}[|C|] = \mathbb{E}\left[\sum_{e \in E} [e \in C]\right] = \sum_{e \in E} \mathbb{E}[[e \in C]] = \sum_{e \in E} \Pr[e \in C] = \frac{|E|}{2}$$

But then *some* partition of V into A and B has $|C| \geq \mathbb{E}[|C|] = \frac{|E|}{2}$. □

The probabilistic method

Note that the probabilistic method by itself does not by itself say anything about *finding* the object (e.g. the partition into A,B) that it proves exists.

In particular, we don't care about analyzing the running time or getting good bounds on the probability of success.

Once we have proved that some object exists, we can try to find it. In many cases this is hard!

Although in this example it is trivial to convert the proof into a Monte Carlo algorithm. Even better, in Assignment #6 next week you will be asked to derandomize this Monte Carlo algorithm.

The probabilistic method

Note that the probabilistic method by itself does not by itself say anything about *finding* the object (e.g. the partition into A,B) that it proves exists.

In particular, we don't care about analyzing the running time or getting good bounds on the probability of success.

Once we have proved that some object exists, we can try to find it. In many cases this is hard!

The probabilistic method

Note that the probabilistic method by itself does not by itself say anything about *finding* the object (e.g. the partition into A,B) that it proves exists.

In particular, we don't care about analyzing the running time or getting good bounds on the probability of success.

Once we have proved that some object exists, we can try to find it. In many cases this is hard!

... But we'll start with one more example where it is not.

MAX-SAT: Definition

Given a set of m clauses in conjunctive normal form over n variables, find a truth assignment that maximizes the number of satisfied clauses.

$$(x_1 \vee \bar{x}_2 \vee x_4) \wedge (\bar{x}_3 \vee \bar{x}_4 \vee x_5) \wedge (\bar{x}_1 \vee x_2 \vee x_4 \vee \bar{x}_5)$$

This is NP-complete in general, so we settle for approximation.

MAX-SAT: Definition

Given a set of m clauses in conjunctive normal form over n variables, find a truth assignment that maximizes the number of satisfied clauses.

$$(x_1 \vee \bar{x}_2 \vee x_4) \wedge (\bar{x}_3 \vee \bar{x}_4 \vee x_5) \wedge (\bar{x}_1 \vee x_2 \vee x_4 \vee \bar{x}_5)$$

This is NP-complete in general, so we settle for approximation.

MAX-SAT: Definition

Given a set of m clauses in conjunctive normal form over n variables, find a truth assignment that maximizes the number of satisfied clauses.

$$(x_1 \vee x_2) \wedge (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee x_2) \wedge (\bar{x}_1 \vee \bar{x}_2)$$

This is NP-complete in general, so we settle for approximation.

MAX-SAT: Definition

Given a set of m clauses in conjunctive normal form over n variables, find a truth assignment that maximizes the number of satisfied clauses.

$$(x_1 \vee x_2) \wedge (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee x_2) \wedge (\bar{x}_1 \vee \bar{x}_2)$$

This is NP-complete in general, so we settle for approximation.

MAX-SAT: Simple

Theorem

For any set of m clauses there is a truth assignmnent that satisfies at least $\frac{m}{2}$ clauses.

This is best possible, since $(x_1) \wedge (\bar{x}_1)$ can only have one clause satisfied out of two.

MAX-SAT: Simple

Theorem

For any set of m clauses there is a truth assignment that satisfies at least $\frac{m}{2}$ clauses.

This is best possible, since $(x_1) \wedge (\bar{x}_1)$ can only have one clause satisfied out of two.

MAX-SAT: Simple, Proof

Proof.

Set each variable to TRUE or FALSE independently and equiprobably. Let Z_i indicate that the i'th clause is satisfied. If Z_i has k_i distinct literals then $\mathbb{E}[Z_i] = \Pr[Z_i = 1] \geq 1 - 2^{-k_i} \geq \frac{1}{2}$. The expected number of satisfied clauses with this assignment is $\mathbb{E}[\sum_{i=1}^{m} Z_i] = \sum_{i=1}^{m} \mathbb{E}[Z_i] \geq \frac{m}{2}$. Therefore there *exists* an assignment with at least $\mathbb{E}[\sum_{i=1}^{m} Z_i] \geq \frac{m}{2}$ satisfied clauses. □

We'll call the algorithm implicit in this proof MAX-SAT-SIMPLE.

MAX-SAT: Simple, Proof

Proof.

Set each variable to TRUE or FALSE independently and equiprobably. Let Z_i indicate that the i'th clause is satisfied. If Z_i has k_i distinct literals then $\mathbb{E}[Z_i] = \Pr[Z_i = 1] \geq 1 - 2^{-k_i} \geq \frac{1}{2}$. The expected number of satisfied clauses with this assignment is $\mathbb{E}[\sum_{i=1}^{m} Z_i] = \sum_{i=1}^{m} \mathbb{E}[Z_i] \geq \frac{m}{2}$. Therefore there *exists* an assignment with at least $\mathbb{E}[\sum_{i=1}^{m} Z_i] \geq \frac{m}{2}$ satisfied clauses. □

We'll call the algorithm implicit in this proof MAX-SAT-SIMPLE.

MAX-SAT: Simple, Proof

Proof.

Set each variable to TRUE or FALSE independently and equiprobably. Let Z_i indicate that the i'th clause is satisfied. If Z_i has k_i distinct literals then $\mathbb{E}[Z_i] = \Pr[Z_i = 1] \geq 1 - 2^{-k_i} \geq \frac{1}{2}$. The expected number of satisfied clauses with this assignment is $\mathbb{E}[\sum_{i=1}^{m} Z_i] = \sum_{i=1}^{m} \mathbb{E}[Z_i] \geq \frac{m}{2}$. Therefore there *exists* an assignment with at least $\mathbb{E}[\sum_{i=1}^{m} Z_i] \geq \frac{m}{2}$ satisfied clauses. $\square$

We'll call the algorithm implicit in this proof MAX-SAT-SIMPLE.

If the same variable appear both negated and unnegated in clause i, then the probability is $1 > 1 - 2^{-k_i}$.
Otherwise, since each of the distinct k_i literals have independent probability $\frac{1}{2}$ of being true, the probability that all of them are false is 2^{-k_i}, so the probability that at least one is true is $1 - 2^{-k_i}$.

MAX-SAT: Simple, Proof

Because the clause has $k_i \geq 1$ variables.

Proof.

Set each variable to TRUE or FALSE independently and equiprobably. Let Z_i indicate that the i'th clause is satisfied. If Z_i has k_i distinct literals then $\mathbb{E}[Z_i] = \Pr[Z_i = 1] \geq 1 - 2^{-k_i} \geq \frac{1}{2}$. The expected number of satisfied clauses with this assignment is $\mathbb{E}[\sum_{i=1}^{m} Z_i] = \sum_{i=1}^{m} \mathbb{E}[Z_i] \geq \frac{m}{2}$. Therefore there *exists* an assignment with at least $\mathbb{E}[\sum_{i=1}^{m} Z_i] \geq \frac{m}{2}$ satisfied clauses. □

We'll call the algorithm implicit in this proof MAX-SAT-SIMPLE.

MAX-SAT: Simple, Proof

Proof.

Set each variable to TRUE or FALSE independently and equiprobably. Let Z_i indicate that the i'th clause is satisfied. If Z_i has k_i distinct literals then $\mathbb{E}[Z_i] = \Pr[Z_i = 1] \geq 1 - 2^{-k_i} \geq \frac{1}{2}$. The expected number of satisfied clauses with this assignment is $\mathbb{E}[\sum_{i=1}^m Z_i] = \sum_{i=1}^m \mathbb{E}[Z_i] \geq \frac{m}{2}$. Therefore there *exists* an assignment with at least $\mathbb{E}[\sum_{i=1}^m Z_i] \geq \frac{m}{2}$ satisfied clauses. $\square$

We'll call the algorithm implicit in this proof MAX-SAT-SIMPLE.

MAX-SAT: Simple, Proof

Proof.

Set each variable to TRUE or FALSE independently and equiprobably. Let Z_i indicate that the i'th clause is satisfied. If Z_i has k_i distinct literals then $\mathbb{E}[Z_i] = \Pr[Z_i = 1] \geq 1 - 2^{-k_i} \geq \frac{1}{2}$. The expected number of satisfied clauses with this assignment is $\mathbb{E}[\sum_{i=1}^{m} Z_i] = \sum_{i=1}^{m} \mathbb{E}[Z_i] \geq \frac{m}{2}$. Therefore there *exists* an assignment with at least $\mathbb{E}[\sum_{i=1}^{m} Z_i] \geq \frac{m}{2}$ satisfied clauses. □

We'll call the algorithm implicit in this proof MAX-SAT-SIMPLE.

MAX-SAT: Simple, Proof

Proof.

Set each variable to TRUE or FALSE independently and equiprobably. Let Z_i indicate that the i'th clause is satisfied. If Z_i has k_i distinct literals then $\mathbb{E}[Z_i] = \Pr[Z_i = 1] \geq 1 - 2^{-k_i} \geq \frac{1}{2}$. The expected number of satisfied clauses with this assignment is $\mathbb{E}[\sum_{i=1}^{m} Z_i] = \sum_{i=1}^{m} \mathbb{E}[Z_i] \geq \frac{m}{2}$. Therefore there *exists* an assignment with at least $\mathbb{E}[\sum_{i=1}^{m} Z_i] \geq \frac{m}{2}$ satisfied clauses. □

We'll call the algorithm implicit in this proof MAX-SAT-SIMPLE.

MAX-SAT: Simple, Proof

Proof.

Set each variable to TRUE or FALSE independently and equiprobably. Let Z_i indicate that the i'th clause is satisfied. If Z_i has k_i distinct literals then $\mathbb{E}[Z_i] = \Pr[Z_i = 1] \geq 1 - 2^{-k_i} \geq \frac{1}{2}$. The expected number of satisfied clauses with this assignment is $\mathbb{E}[\sum_{i=1}^{m} Z_i] = \sum_{i=1}^{m} \mathbb{E}[Z_i] \geq \frac{m}{2}$. Therefore there *exists* an assignment with at least $\mathbb{E}[\sum_{i=1}^{m} Z_i] \geq \frac{m}{2}$ satisfied clauses. $\square$

We'll call the algorithm implicit in this proof MAX-SAT-SIMPLE.

MAX-SAT: Simple, Proof

Proof.

Set each variable to TRUE or FALSE independently and equiprobably. Let Z_i indicate that the i'th clause is satisfied. If Z_i has k_i distinct literals then $\mathbb{E}[Z_i] = \Pr[Z_i = 1] \geq 1 - 2^{-k_i} \geq \frac{1}{2}$. The expected number of satisfied clauses with this assignment is $\mathbb{E}[\sum_{i=1}^{m} Z_i] = \sum_{i=1}^{m} \mathbb{E}[Z_i] \geq \frac{m}{2}$. Therefore there *exists* an assignment with at least $\mathbb{E}[\sum_{i=1}^{m} Z_i] \geq \frac{m}{2}$ satisfied clauses. □

We'll call the algorithm implicit in this proof MAX-SAT-SIMPLE.

MAX-SAT: Performance ratio

Given a MAX-SAT instance I, and algorithm A, let

$m_\star(I)$ be the maximal number of clauses that can be satisfied, and

$m_A(I)$ be the number of clauses satisfied when using algorithm A.

Let $\alpha = \inf_I \frac{\mathbb{E}[m_A(I)]}{m_\star(I)}$. Then α is called the *performance ratio* of A, and A is called an *α-approximation algorithm.*

MAX-SAT: Performance ratio

The performance ratio is also called the *approximation ratio*

Given a MAX-SAT instance I, and algorithm A, let

$m_\star(I)$ be the maximal number of clauses that can be satisfied, and

$m_A(I)$ be the number of clauses satisfied when using algorithm A.

Let $\alpha = \inf_I \frac{\mathbb{E}[m_A(I)]}{m_\star(I)}$. Then α is called the *performance ratio* of A, and A is called an α*-approximation algorithm*.

MAX-SAT: Simple, Summary

MAX-SAT-SIMPLE is a $\frac{1}{2}$-approximation algorithm.

In fact, if each clause has at least k distinct literals it is a $(1 - 2^{-k})$-approximation algorithm.

In particular, if each clause has at least 2 distinct literals MAX-SAT-SIMPLE is a $\frac{3}{4}$-approximation algorithm.

We'll now show a different algorithm that is good when many clauses have only one literal. Then we'll combine them to a $\frac{3}{4}$-approximation algorithm for all cases.

Because for any I with m clauses, $\mathbb{E}[m_{\text{MAX-SAT-SIMPLE}}(I)] \geq \frac{m}{2}$ and $m_\star(I) \leq m$, thus $\inf_I \frac{\mathbb{E}[m_{\text{MAX-SAT-SIMPLE}}(I)]}{m_\star(I)} = \frac{\frac{m}{2}}{m} = \frac{1}{2}$.

MAX-SAT: Simple, Summary

MAX-SAT-SIMPLE is a $\frac{1}{2}$-approximation algorithm.

In fact, if each clause has at least k distinct literals it is a $(1 - 2^{-k})$-approximation algorithm.

In particular, if each clause has at least 2 distinct literals MAX-SAT-SIMPLE is a $\frac{3}{4}$-approximation algorithm.

We'll now show a different algorithm that is good when many clauses have only one literal. Then we'll combine them to a $\frac{3}{4}$-approximation algorithm for all cases.

MAX-SAT: Simple, Summary

MAX-SAT-SIMPLE is a $\frac{1}{2}$-approximation algorithm.

In fact, if each clause has at least k distinct literals it is a $(1 - 2^{-k})$-approximation algorithm.

In particular, if each clause has at least 2 distinct literals MAX-SAT-SIMPLE is a $\frac{3}{4}$-approximation algorithm.

We'll now show a different algorithm that is good when many clauses have only one literal. Then we'll combine them to a $\frac{3}{4}$-approximation algorithm for all cases.

MAX-SAT: Simple, Summary

MAX-SAT-SIMPLE is a $\frac{1}{2}$-approximation algorithm.

In fact, if each clause has at least k distinct literals it is a $(1 - 2^{-k})$-approximation algorithm.

In particular, if each clause has at least 2 distinct literals MAX-SAT-SIMPLE is a $\frac{3}{4}$-approximation algorithm.

We'll now show a different algorithm that is good when many clauses have only one literal. Then we'll combine them to a $\frac{3}{4}$-approximation algorithm for all cases.

MAX-SAT: ILP+RR

For our second algorithm we'll use the idea of *randomized rounding*. Let

C_j^+	be the set of indices of variables that appear unnegated in clause j.
C_j^-	be the set of indices of variables that appear negated in clause j.

Consider the following ILP

$$\begin{array}{lll} \text{maximize} & \sum_{j=1}^{m} z_j & \\ \text{where} & y_i, z_j \in \{0,1\} & (\forall i, j) \\ \text{subject to} & \sum_{i \in C_j^+} y_i + \sum_{i \in C_j^-} (1 - y_i) \geq z_j & (\forall j) \end{array}$$

MAX-SAT: ILP+RR

For our second algorithm we'll use the idea of *randomized rounding*. Let

C_j^+ be the set of indices of variables that appear unnegated in clause j.

C_j^- be the set of indices of variables that appear negated in clause j.

Consider the following ILP

$$\begin{array}{lll} \text{maximize} & \displaystyle\sum_{j=1}^{m} z_j & \\ \text{where} & y_i, z_j \in \{0,1\} & (\forall i,j) \\ \text{subject to} & \displaystyle\sum_{i \in C_j^+} y_i + \sum_{i \in C_j^-} (1-y_i) \geq z_j & (\forall j) \end{array}$$

Note here that the left side of the last line is nonzero if and only if the assignment corresponding to y satisfies the clause. Thus z_j is allowed to be 1 only when C_j is satisfied. Since we are maximizing the sum of z_j's, that means that z_j will be 1 exactly when C_j is satisfied. And therefore the sum is actually counting the number of satisfied clauses.

MAX-SAT: ILP+RR

For our second algorithm we'll use the idea of *randomized rounding*. Let

C_j^+ be the set of indices of variables that appear unnegated in clause j.

C_j^- be the set of indices of variables that appear negated in clause j.

And its LP-relaxation

$$
\begin{array}{lll}
\text{maximize} & \displaystyle\sum_{j=1}^{m} \hat{z}_j & \\
\text{where} & \hat{y}_i, \hat{z}_j \in [0,1] & (\forall i, j) \\
\text{subject to} & \displaystyle\sum_{i \in C_j^+} \hat{y}_i + \sum_{i \in C_j^-} (1 - \hat{y}_i) \geq \hat{z}_j & (\forall j)
\end{array}
$$

MAX-SAT: RR

Let MAX-SAT-RR be the algorithm that first solves the LP-relaxation, and then independently sets each x_i to TRUE with probability $\hat{y}_i$ (and else FALSE).

We will show that MAX-SAT-RR is a $(1 - \frac{1}{e})$-approximation algorithm.

Let $\beta_k = 1 - (1 - \frac{1}{k})^k$. (Observe $\beta_k \geq 1 - \frac{1}{e}$, why?).

Lemma

The probability that clause C_j is satisfied by MAX-SAT-RR *is at least* $\beta_{k_j} \hat{z}_j$.

MAX-SAT: RR

Let MAX-SAT-RR be the algorithm that first solves the LP-relaxation, and then independently sets each x_i to TRUE with probability $\hat{y}_i$ (and else FALSE).

We will show that MAX-SAT-RR is a $(1-\frac{1}{e})$-approximation algorithm.

Let $\beta_k = 1-(1-\frac{1}{k})^k$. (Observe $\beta_k \geq 1-\frac{1}{e}$, why?).

Lemma

The probability that clause C_j is satisfied by MAX-SAT-RR *is at least $\beta_{k_j}\hat{z}_j$.*

$1-\frac{1}{e} \approx 0.632120559$ is already better than $\frac{1}{2}$.

MAX-SAT: RR

Let MAX-SAT-RR be the algorithm that first solves the LP-relaxation, and then independently sets each x_i to TRUE with probability $\hat{y}_i$ (and else FALSE).

We will show that MAX-SAT-RR is a $(1 - \frac{1}{e})$-approximation algorithm.

Let $\beta_k = 1 - (1 - \frac{1}{k})^k$. (Observe $\beta_k \geq 1 - \frac{1}{e}$, why?).

Lemma

The probability that clause C_j is satisfied by MAX-SAT-RR *is at least $\beta_{k_j} \hat{z}_j$.*

MAX-SAT: RR

$1-\frac{1}{k} \leq e^{-\frac{1}{k}} \implies (1-\frac{1}{k})^k \leq \frac{1}{e} \implies 1-\frac{1}{e} \leq 1-(1-\frac{1}{k})^k = \beta_k.$

Let MAX-SAT-RR be the algorithm that first solves the LP-relaxation, and then independently sets each x_i to TRUE with probability $\hat{y}_i$ (and else FALSE).

We will show that MAX-SAT-RR is a $(1 - \frac{1}{e})$-approximation algorithm.

Let $\beta_k = 1 - (1 - \frac{1}{k})^k$. (Observe $\beta_k \geq 1 - \frac{1}{e}$, why?).

Lemma

The probability that clause C_j is satisfied by MAX-SAT-RR *is at least $\beta_{k_j} \hat{z}_j$.*

MAX-SAT: RR

Let MAX-SAT-RR be the algorithm that first solves the LP-relaxation, and then independently sets each x_i to TRUE with probability $\hat{y}_i$ (and else FALSE).

We will show that MAX-SAT-RR is a $(1-\frac{1}{e})$-approximation algorithm.

Let $\beta_k = 1-(1-\frac{1}{k})^k$. (Observe $\beta_k \geq 1-\frac{1}{e}$, why?).

Lemma

The probability that clause C_j is satisfied by MAX-SAT-RR *is at least $\beta_{k_j}\hat{z}_j$.*

MAX-SAT: RR Lemma Proof

Proof.

We can assume without loss of generality that C_j has the form $x_1 \vee \cdots \vee x_{k_j}$. Since each x_i is independently set to FALSE with probability $1 - \hat{y}_i$, C_j is unsatisfied with probability $\prod_{i=1}^{k_j}(1 - \hat{y}_i)$. So C_j is satisfied with probability

$$1 - \prod_{i=1}^{k_j}(1 - \hat{y}_i) \geq 1 - \left(1 - \frac{\hat{z}_j}{k_j}\right)^{k_j}$$

For all $k \in \mathbb{N}$, the function $f_k(z) = 1 - (1 - \frac{z}{k})^k$ is *concave* for $z \in [0, 1]$, so $f_k(0) = 0$ and $f_k(1) = \beta_k$ implies that $f_k(z) \geq z \cdot \beta_k$ for $z \in [0, 1]$. In particular, $\Pr[C_j \text{ satisfied}] \geq f_{k_j}(\hat{z}_j) \geq \beta_{k_j} \hat{z}_j$. $\square$

This is without loss of generality because

- We can change every literal using a given variable to its opposite and get an equivalent instance.
- We can renumber variables as we like without changing the instance.

MAX-SAT: RR Lemma Proof

Proof.

We can assume without loss of generality that C_j has the form $x_1 \vee \cdots \vee x_{k_j}$. Since each x_i is independently set to FALSE with probability $1 - \hat{y}_i$, C_j is unsatisfied with probability $\prod_{i=1}^{k_j}(1 - \hat{y}_i)$. So C_j is satisfied with probability

$$1 - \prod_{i=1}^{k_j}(1 - \hat{y}_i) \geq 1 - \left(1 - \frac{\hat{z}_j}{k_j}\right)^{k_j}$$

For all $k \in \mathbb{N}$, the function $f_k(z) = 1 - (1 - \frac{z}{k})^k$ is *concave* for $z \in [0, 1]$, so $f_k(0) = 0$ and $f_k(1) = \beta_k$ implies that $f_k(z) \geq z \cdot \beta_k$ for $z \in [0, 1]$. In particular, $\Pr[C_j \text{ satisfied}] \geq f_{k_j}(\hat{z}_j) \geq \beta_{k_j}\hat{z}_j$. $\square$

MAX-SAT: RR Lemma Proof

Proof.

We can assume without loss of generality that C_j has the form $x_1 \vee \cdots \vee x_{k_j}$. Since each x_i is independently set to FALSE with probability $1 - \hat{y}_i$, C_j is unsatisfied with probability $\prod_{i=1}^{k_j}(1 - \hat{y}_i)$. So C_j is satisfied with probability

$$1 - \prod_{i=1}^{k_j}(1 - \hat{y}_i) \geq 1 - \left(1 - \frac{\hat{z}_j}{k_j}\right)^{k_j}$$

For all $k \in \mathbb{N}$, the function $f_k(z) = 1 - (1 - \frac{z}{k})^k$ is *concave* for $z \in [0,1]$, so $f_k(0) = 0$ and $f_k(1) = \beta_k$ implies that $f_k(z) \geq z \cdot \beta_k$ for $z \in [0,1]$. In particular, $\Pr[C_j \text{ satisfied}] \geq f_{k_j}(\hat{z}_j) \geq \beta_{k_j}\hat{z}_j$. □

MAX-SAT: RR Lemma Proof

Proof.

We can assume without loss of generality that C_j has the form $x_1 \vee \cdots \vee x_{k_j}$. Since each x_i is independently set to FALSE with probability $1 - \hat{y}_i$, C_j is unsatisfied with probability $\prod_{i=1}^{k_j}(1 - \hat{y}_i)$. So C_j is satisfied with probability

$$1 - \prod_{i=1}^{k_j}(1 - \hat{y}_i) \geq 1 - \left(1 - \frac{\hat{z}_j}{k_j}\right)^{k_j}$$

For all $k \in \mathbb{N}$, the function $f_k(z) = 1 - (1 - \frac{z}{k})^k$ is *concave* for $z \in [0,1]$, so $f_k(0) = 0$ and $f_k(1) = \beta_k$ implies that $f_k(z) \geq z \cdot \beta_k$ for $z \in [0,1]$. In particular, $\Pr[C_j \text{ satisfied}] \geq f_{k_j}(\hat{z}_j) \geq \beta_{k_j} \hat{z}_j$. □

Minimized when all $\hat{y}_i$ are equal. Since $\hat{z}_i = \sum_{i=1}^{k_j} \hat{y}_i$, this means $\hat{y}_i = \frac{\hat{z}_j}{k_j}$.

MAX-SAT: RR Lemma Proof

Proof.

We can assume without loss of generality that C_j has the form $x_1 \vee \cdots \vee x_{k_j}$. Since each x_i is independently set to FALSE with probability $1 - \hat{y}_i$, C_j is unsatisfied with probability $\prod_{i=1}^{k_j}(1 - \hat{y}_i)$. So C_j is satisfied with probability

$$1 - \prod_{i=1}^{k_j}(1 - \hat{y}_i) \geq 1 - \left(1 - \frac{\hat{z}_j}{k_j}\right)^{k_j}$$

For all $k \in \mathbb{N}$, the function $f_k(z) = 1 - (1 - \frac{z}{k})^k$ is *concave* for $z \in [0, 1]$, so $f_k(0) = 0$ and $f_k(1) = \beta_k$ implies that $f_k(z) \geq z \cdot \beta_k$ for $z \in [0, 1]$. In particular, $\Pr[C_j \text{ satisfied}] \geq f_{k_j}(\hat{z}_j) \geq \beta_{k_j}\hat{z}_j$. $\square$

By definition, a function $f(x)$ is *concave* on an interval $[a, b]$ if for all $\lambda \in [0, 1]$, $f((1 - \lambda)a + \lambda b) \geq (1 - \lambda)f(a) + \lambda f(b)$.

If f has a second derivative f'' on $[a, b]$ then this is equivalent to $f''(x) \leq 0$ for all $x \in [a, b]$.

For $k \in \mathbb{N}$ and $z \in [0, k]$

$$\frac{d^2}{dz^2}\left(1 - \left(1 - \frac{z}{k}\right)^k\right) = -\frac{(k - 1)(1 - \frac{z}{k})^{k-2}}{k} \leq 0$$

So f_k is concave on $[0, k]$, and in particular on $[0, 1]$.

If $a = f(a) = 0$ and $b = 1$, being concave simplifies to $f(\lambda) \geq \lambda f(1)$ for $\lambda \in [0, 1]$.

MAX-SAT: RR Lemma Proof

Proof.

We can assume without loss of generality that C_j has the form $x_1 \vee \cdots \vee x_{k_j}$. Since each x_i is independently set to FALSE with probability $1 - \hat{y}_i$, C_j is unsatisfied with probability $\prod_{i=1}^{k_j}(1 - \hat{y}_i)$. So C_j is satisfied with probability

$$1 - \prod_{i=1}^{k_j}(1 - \hat{y}_i) \geq 1 - \left(1 - \frac{\hat{z}_j}{k_j}\right)^{k_j}$$

For all $k \in \mathbb{N}$, the function $f_k(z) = 1 - (1 - \frac{z}{k})^k$ is *concave* for $z \in [0, 1]$, so $f_k(0) = 0$ and $f_k(1) = \beta_k$ implies that $f_k(z) \geq z \cdot \beta_k$ for $z \in [0, 1]$. In particular, $\Pr[C_j \text{ satisfied}] \geq f_{k_j}(\hat{z}_j) \geq \beta_{k_j}\hat{z}_j$. $\square$

MAX-SAT: RR Lemma Proof

Proof.

We can assume without loss of generality that C_j has the form $x_1 \vee \cdots \vee x_{k_j}$. Since each x_i is independently set to FALSE with probability $1 - \hat{y}_i$, C_j is unsatisfied with probability $\prod_{i=1}^{k_j}(1 - \hat{y}_i)$. So C_j is satisfied with probability

$$1 - \prod_{i=1}^{k_j}(1 - \hat{y}_i) \geq 1 - \left(1 - \frac{\hat{z}_j}{k_j}\right)^{k_j}$$

For all $k \in \mathbb{N}$, the function $f_k(z) = 1 - (1 - \frac{z}{k})^k$ is *concave* for $z \in [0, 1]$, so $f_k(0) = 0$ and $f_k(1) = \beta_k$ implies that $f_k(z) \geq z \cdot \beta_k$ for $z \in [0, 1]$. In particular, $\Pr[C_j \text{ satisfied}] \geq f_{k_j}(\hat{z}_j) \geq \beta_{k_j}\hat{z}_j$. $\square$

MAX-SAT: RR Theorem

Theorem

MAX-SAT-RR *is an* $(1 - \frac{1}{e})$*-approximation algorithm.*

Proof.

Let Z_j indicate that clause j is satisfied. By previous lemma, $\mathbb{E}[Z_j] \geq \beta_{k_j}\hat{z}_j \geq (1 - \frac{1}{e})\hat{z}_j$. The expected number of satisfied clauses is

$$\begin{aligned}\sum_{j=1}^{m} \mathbb{E}[Z_j] \geq \sum_{j=1}^{m} (1 - \tfrac{1}{e})\hat{z}_j &= (1 - \tfrac{1}{e}) \sum_{j=1}^{m} \hat{z}_j \\ &\geq (1 - \tfrac{1}{e}) \sum_{j=1}^{m} z_j\end{aligned}$$

$\square$

MAX-SAT: RR Theorem

Theorem

MAX-SAT-RR *is an* $(1 - \frac{1}{e})$*-approximation algorithm.*

Proof.

Let Z_j indicate that clause j is satisfied. By previous lemma, $\mathbb{E}[Z_j] \geq \beta_{k_j} \hat{z}_j \geq (1 - \frac{1}{e})\hat{z}_j$. The expected number of satisfied clauses is

$$\sum_{j=1}^{m} \mathbb{E}[Z_j] \geq \sum_{j=1}^{m} (1 - \tfrac{1}{e})\hat{z}_j = (1 - \tfrac{1}{e}) \sum_{j=1}^{m} \hat{z}_j$$

$$\geq (1 - \tfrac{1}{e}) \sum_{j=1}^{m} z_j \qquad \square$$

MAX-SAT: RR Theorem

Theorem

MAX-SAT-RR *is an* $(1 - \frac{1}{e})$*-approximation algorithm.*

Proof.

Let Z_j indicate that clause j is satisfied. By previous lemma, $\mathbb{E}[Z_j] \geq \beta_{k_j}\hat{z}_j \geq (1 - \frac{1}{e})\hat{z}_j$. The expected number of satisfied clauses is

$$\sum_{j=1}^{m} \mathbb{E}[Z_j] \geq \sum_{j=1}^{m}(1 - \tfrac{1}{e})\hat{z}_j = (1 - \tfrac{1}{e})\sum_{j=1}^{m}\hat{z}_j$$

$$\geq (1 - \tfrac{1}{e})\sum_{j=1}^{m} z_j \qquad \square$$

Expectation of indicator variable.

MAX-SAT: RR Theorem

Theorem

MAX-SAT-RR *is an* $(1 - \frac{1}{e})$*-approximation algorithm.*

Proof.

Let Z_j indicate that clause j is satisfied. By previous lemma, $\mathbb{E}[Z_j] \geq \beta_{k_j} \hat{z}_j \geq (1 - \frac{1}{e})\hat{z}_j$. The expected number of satisfied clauses is

$$\sum_{j=1}^{m} \mathbb{E}[Z_j] \geq \sum_{j=1}^{m} (1 - \tfrac{1}{e})\hat{z}_j = (1 - \tfrac{1}{e}) \sum_{j=1}^{m} \hat{z}_j$$

$$\geq (1 - \tfrac{1}{e}) \sum_{j=1}^{m} z_j \qquad \square$$

MAX-SAT: RR Theorem

Theorem

MAX-SAT-RR *is an* $(1 - \frac{1}{e})$*-approximation algorithm.*

Proof.

Let Z_j indicate that clause j is satisfied. By previous lemma, $\mathbb{E}[Z_j] \geq \beta_{k_j} \hat{z}_j \geq (1 - \frac{1}{e})\hat{z}_j$. The expected number of satisfied clauses is

$$\sum_{j=1}^{m} \mathbb{E}[Z_j] \geq \sum_{j=1}^{m} (1 - \tfrac{1}{e})\hat{z}_j = (1 - \tfrac{1}{e}) \sum_{j=1}^{m} \hat{z}_j$$

$$\geq (1 - \tfrac{1}{e}) \sum_{j=1}^{m} z_j \qquad \square$$

MAX-SAT: RR Theorem

Theorem

MAX-SAT-RR *is an* $(1 - \frac{1}{e})$*-approximation algorithm.*

Proof.

Let Z_j indicate that clause j is satisfied. By previous lemma, $\mathbb{E}[Z_j] \geq \beta_{k_j} \hat{z}_j \geq (1 - \frac{1}{e})\hat{z}_j$. The expected number of satisfied clauses is

$$\sum_{j=1}^{m} \mathbb{E}[Z_j] \geq \sum_{j=1}^{m} (1 - \tfrac{1}{e})\hat{z}_j = (1 - \tfrac{1}{e}) \sum_{j=1}^{m} \hat{z}_j$$

$$\geq (1 - \tfrac{1}{e}) \sum_{j=1}^{m} z_j \qquad \square$$

Because the objective value of the solution to the LP-relaxation of a maximization problem must be $\geq$ than the optimal objective value for the original ILP.

MAX-SAT: Combined

We can do better!

Theorem

For any instance I let $n_1 = \mathbb{E}[m_{\text{MAX-SAT-SIMPLE}}(I)]$ and $n_2 = \mathbb{E}[m_{\text{MAX-SAT-RR}}(I)]$, then

$$\max\{n_1, n_2\} \geq \frac{3}{4} \sum_{j=1}^{m} \hat{z}_j$$

So the algorithm MAX-SAT-COMBINED that runs both MAX-SAT-SIMPLE and MAX-SAT-RR and takes the best is a $\frac{3}{4}$-approximation algorithm.

MAX-SAT: Combined

We can do better!

Theorem

For any instance I let $n_1 = \mathbb{E}[m_{\text{MAX-SAT-SIMPLE}}(I)]$ and $n_2 = \mathbb{E}[m_{\text{MAX-SAT-RR}}(I)]$, then

$$\max\{n_1, n_2\} \geq \frac{3}{4} \sum_{j=1}^{m} \hat{z}_j$$

So the algorithm MAX-SAT-COMBINED that runs both MAX-SAT-SIMPLE and MAX-SAT-RR and takes the best is a $\frac{3}{4}$-approximation algorithm.

MAX-SAT: Combined

We can do better!

Theorem

For any instance I let $n_1 = \mathbb{E}[m_{\text{MAX-SAT-SIMPLE}}(I)]$ and $n_2 = \mathbb{E}[m_{\text{MAX-SAT-RR}}(I)]$, then

$$\max\{n_1, n_2\} \geq \frac{3}{4} \sum_{j=1}^{m} \hat{z}_j$$

So the algorithm MAX-SAT-COMBINED that runs both MAX-SAT-SIMPLE and MAX-SAT-RR and takes the best is a $\frac{3}{4}$-approximation algorithm.

MAX-SAT: Combined Proof

Proof.

Since $\max\{n_1, n_2\} \geq \frac{n_1+n_2}{2}$, it is sufficient to show that $\frac{n_1+n_2}{2} \geq \frac{3}{4}\sum_{j=1}^{m} \hat{z}_j$. Let S^k be the set of clauses with k distinct literals, then

$$n_1 = \sum_k \sum_{C_j \in S^k} (1 - 2^{-k}) \geq \sum_k \sum_{C_j \in S^k} (1 - 2^{-k}) \hat{z}_j$$

$$n_2 = \sum_k \sum_{C_j \in S^k} \beta_k \hat{z}_j$$

$$\frac{n_1 + n_2}{2} \geq \frac{1}{2} \sum_k \sum_{C_j \in S^k} ((1 - 2^{-k}) + \beta_k) \hat{z}_j$$

$$\geq \frac{1}{2} \sum_k \sum_{C_j \in S^k} \frac{3}{2} \hat{z}_j = \frac{3}{4} \sum_{j=1}^{m} \hat{z}_j$$

□

MAX-SAT: Combined Proof

Proof.

Since $\max\{n_1, n_2\} \geq \frac{n_1+n_2}{2}$, it is sufficient to show that $\frac{n_1+n_2}{2} \geq \frac{3}{4}\sum_{j=1}^{m} \hat{z}_j$. Let S^k be the set of clauses with k distinct literals, then

$$n_1 = \sum_k \sum_{C_j \in S^k} (1 - 2^{-k}) \geq \sum_k \sum_{C_j \in S^k} (1 - 2^{-k})\hat{z}_j$$

$$n_2 = \sum_k \sum_{C_j \in S^k} \beta_k \hat{z}_j$$

$$\frac{n_1 + n_2}{2} \geq \frac{1}{2} \sum_k \sum_{C_j \in S^k} ((1 - 2^{-k}) + \beta_k)\hat{z}_j$$

$$\geq \frac{1}{2} \sum_k \sum_{C_j \in S^k} \frac{3}{2}\hat{z}_j = \frac{3}{4}\sum_{j=1}^{m} \hat{z}_j$$

□

MAX-SAT: Combined Proof

Proof.

Since $\max\{n_1, n_2\} \geq \frac{n_1+n_2}{2}$, it is sufficient to show that $\frac{n_1+n_2}{2} \geq \frac{3}{4}\sum_{j=1}^{m} \hat{z}_j$. Let S^k be the set of clauses with k distinct literals, then

$$n_1 = \sum_k \sum_{C_j \in S^k} (1 - 2^{-k}) \geq \sum_k \sum_{C_j \in S^k} (1 - 2^{-k}) \hat{z}_j$$

$$n_2 = \sum_k \sum_{C_j \in S^k} \beta_k \hat{z}_j$$

$$\frac{n_1 + n_2}{2} \geq \frac{1}{2} \sum_k \sum_{C_j \in S^k} ((1 - 2^{-k}) + \beta_k) \hat{z}_j$$

$$\geq \frac{1}{2} \sum_k \sum_{C_j \in S^k} \frac{3}{2} \hat{z}_j = \frac{3}{4} \sum_{j=1}^{m} \hat{z}_j \qquad \square$$

Because $0 \le \hat{z}_j \le 1$.

MAX-SAT: Combined Proof

Proof.

Since $\max\{n_1, n_2\} \ge \frac{n_1+n_2}{2}$, it is sufficient to show that $\frac{n_1+n_2}{2} \ge \frac{3}{4}\sum_{j=1}^{m} \hat{z}_j$. Let S^k be the set of clauses with k distinct literals, then

$$n_1 = \sum_k \sum_{C_j \in S^k} (1 - 2^{-k}) \ge \sum_k \sum_{C_j \in S^k} (1 - 2^{-k})\hat{z}_j$$

$$n_2 = \sum_k \sum_{C_j \in S^k} \beta_k \hat{z}_j$$

$$\frac{n_1 + n_2}{2} \ge \frac{1}{2}\sum_k \sum_{C_j \in S^k} ((1 - 2^{-k}) + \beta_k)\hat{z}_j$$

$$\ge \frac{1}{2}\sum_k \sum_{C_j \in S^k} \frac{3}{2}\hat{z}_j = \frac{3}{4}\sum_{j=1}^{m} \hat{z}_j \qquad \square$$

MAX-SAT: Combined Proof

Proof.

Since $\max\{n_1, n_2\} \geq \frac{n_1+n_2}{2}$, it is sufficient to show that $\frac{n_1+n_2}{2} \geq \frac{3}{4}\sum_{j=1}^{m} \hat{z}_j$. Let S^k be the set of clauses with k distinct literals, then

$$n_1 = \sum_k \sum_{C_j \in S^k} (1 - 2^{-k}) \geq \sum_k \sum_{C_j \in S^k} (1 - 2^{-k})\hat{z}_j$$

$$n_2 = \sum_k \sum_{C_j \in S^k} \beta_k \hat{z}_j$$

$$\frac{n_1 + n_2}{2} \geq \frac{1}{2} \sum_k \sum_{C_j \in S^k} ((1 - 2^{-k}) + \beta_k)\hat{z}_j$$

$$\geq \frac{1}{2} \sum_k \sum_{C_j \in S^k} \frac{3}{2}\hat{z}_j = \frac{3}{4}\sum_{j=1}^{m} \hat{z}_j \qquad \square$$

MAX-SAT: Combined Proof

Proof.

Since $\max\{n_1, n_2\} \geq \frac{n_1+n_2}{2}$, it is sufficient to show that $\frac{n_1+n_2}{2} \geq \frac{3}{4}\sum_{j=1}^{m} \hat{z}_j$. Let S^k be the set of clauses with k distinct literals, then

$$n_1 = \sum_k \sum_{C_j \in S^k} (1 - 2^{-k}) \geq \sum_k \sum_{C_j \in S^k} (1 - 2^{-k})\hat{z}_j$$

$$n_2 = \sum_k \sum_{C_j \in S^k} \beta_k \hat{z}_j$$

$$\frac{n_1 + n_2}{2} \geq \frac{1}{2}\sum_k \sum_{C_j \in S^k} \left((1 - 2^{-k}) + \beta_k\right)\hat{z}_j$$

$$\geq \frac{1}{2}\sum_k \sum_{C_j \in S^k} \frac{3}{2}\hat{z}_j = \frac{3}{4}\sum_{j=1}^{m} \hat{z}_j \qquad \square$$

MAX-SAT: Combined Proof

Proof.

Since $\max\{n_1, n_2\} \geq \frac{n_1+n_2}{2}$, it is sufficient to show that $\frac{n_1+n_2}{2} \geq \frac{3}{4}\sum_{j=1}^{m} \hat{z}_j$. Let S^k be the set of clauses with k distinct literals, then

$$n_1 = \sum_k \sum_{C_j \in S^k} (1 - 2^{-k}) \geq \sum_k \sum_{C_j \in S^k} (1 - 2^{-k})\hat{z}_j$$

$$n_2 = \sum_k \sum_{C_j \in S^k} \beta_k \hat{z}_j$$

$$\frac{n_1 + n_2}{2} \geq \frac{1}{2} \sum_k \sum_{C_j \in S^k} \left((1 - 2^{-k}) + \beta_k\right)\hat{z}_j$$

$$\geq \frac{1}{2} \sum_k \sum_{C_j \in S^k} \frac{3}{2}\hat{z}_j = \frac{3}{4}\sum_{j=1}^{m} \hat{z}_j$$

□

For $k = 1$: $(1 - 2^{-k}) + \beta_k = (1 - \frac{1}{2}) + (1 - (1 - \frac{1}{1})^1) = \frac{3}{2}$.
For $k = 2$: $(1 - 2^{-k}) + \beta_k = (1 - \frac{1}{4}) + (1 - (1 - \frac{1}{2})^2) = \frac{3}{2}$.
For $k \geq 3$, $(1 - 2^{-k}) + \beta_k \geq (1 - 2^{-3}) + (1 - \frac{1}{e}) \approx 1.507 > \frac{3}{2}$

MAX-SAT: Combined Proof

Proof.

Since $\max\{n_1, n_2\} \geq \frac{n_1+n_2}{2}$, it is sufficient to show that $\frac{n_1+n_2}{2} \geq \frac{3}{4}\sum_{j=1}^{m} \hat{z}_j$. Let S^k be the set of clauses with k distinct literals, then

$$n_1 = \sum_k \sum_{C_j \in S^k} (1 - 2^{-k}) \geq \sum_k \sum_{C_j \in S^k} (1 - 2^{-k})\hat{z}_j$$

$$n_2 = \sum_k \sum_{C_j \in S^k} \beta_k \hat{z}_j$$

$$\frac{n_1 + n_2}{2} \geq \frac{1}{2} \sum_k \sum_{C_j \in S^k} \left((1 - 2^{-k}) + \beta_k\right)\hat{z}_j$$

$$\geq \frac{1}{2} \sum_k \sum_{C_j \in S^k} \frac{3}{2}\hat{z}_j = \frac{3}{4}\sum_{j=1}^{m} \hat{z}_j \qquad \square$$

MAX-SAT: Summary

We used the probabilistic method to prove that there is always a truth assigment that satisfies at least $\frac{m}{2}$ clauses.

The proof gave us the $\frac{1}{2}$-approximation algorithm MAX-SAT-SIMPLE.

Then we showed an ILP formulation of the MAX-SAT problem, and using randomized rounding on the LP-relaxation of this gave us the $(1 - \frac{1}{e})$-aproximation algorithm MAX-SAT-RR.

Finally, we combined the two algorithms into the $\frac{3}{4}$-approximation algorithm MAX-SAT-COMBINED.

MAX-SAT: Summary

We used the probabilistic method to prove that there is always a truth assigment that satisfies at least $\frac{m}{2}$ clauses.

The proof gave us the $\frac{1}{2}$-approximation algorithm MAX-SAT-SIMPLE.

Then we showed an ILP formulation of the MAX-SAT problem, and using randomized rounding on the LP-relaxation of this gave us the $(1 - \frac{1}{e})$-aproximation algorithm MAX-SAT-RR.

Finally, we combined the two algorithms into the $\frac{3}{4}$-approximation algorithm MAX-SAT-COMBINED.

MAX-SAT: Summary

We used the probabilistic method to prove that there is always a truth assigment that satisfies at least $\frac{m}{2}$ clauses.

The proof gave us the $\frac{1}{2}$-approximation algorithm MAX-SAT-SIMPLE.

Then we showed an ILP formulation of the MAX-SAT problem, and using randomized rounding on the LP-relaxation of this gave us the $(1 - \frac{1}{e})$-aproximation algorithm MAX-SAT-RR.

Finally, we combined the two algorithms into the $\frac{3}{4}$-approximation algorithm MAX-SAT-COMBINED.

MAX-SAT: Summary

We used the probabilistic method to prove that there is always a truth assigment that satisfies at least $\frac{m}{2}$ clauses.

The proof gave us the $\frac{1}{2}$-approximation algorithm $\textsc{max-sat-simple}$.

Then we showed an ILP formulation of the MAX-SAT problem, and using randomized rounding on the LP-relaxation of this gave us the $(1 - \frac{1}{e})$-aproximation algorithm $\textsc{max-sat-rr}$.

Finally, we combined the two algorithms into the $\frac{3}{4}$-approximation algorithm $\textsc{max-sat-combined}$.

OR-concentrator: Definition

Back to the probabilistic method...

Definition

An (n, d, α, c) OR-concentrator is a bipartite multigraph $G(L, R, E)$, with $|L| = |R| = n$, such that

1. Every vertex in L has degree at most d.
2. Every $S \subseteq L$ with $|S| \leq \alpha n$ has at least $c|S|$ neighbors in R.

This is an example of an *expanding graph*.

For most applications we want d small and c large.
Usually d, α, c are constants and $c > 1$.

OR-concentrator: Definition

Back to the probabilistic method...

Definition

An (n, d, α, c) OR-concentrator is a bipartite multigraph $G(L, R, E)$, with $|L| = |R| = n$, such that

1. Every vertex in L has degree at most d.
2. Every $S \subseteq L$ with $|S| \leq \alpha n$ has at least $c|S|$ neighbors in R.

This is an example of an *expanding graph*.

For most applications we want d small and c large.
Usually d, α, c are constants and $c > 1$.

This particular type of graph is (apparently) useful when designing phone networks.

OR-concentrator: Definition

Back to the probabilistic method...

Definition

An (n, d, α, c) OR-concentrator is a bipartite multigraph $G(L, R, E)$, with $|L| = |R| = n$, such that

1. Every vertex in L has degree at most d.
2. Every $S \subseteq L$ with $|S| \leq \alpha n$ has at least $c|S|$ neighbors in R.

This is an example of an *expanding graph*.

For most applications we want d small and c large.
Usually d, α, c are constants and $c > 1$.

OR-concentrator: Definition

Back to the probabilistic method...

Definition

An (n, d, α, c) OR-concentrator is a bipartite multigraph $G(L, R, E)$, with $|L| = |R| = n$, such that

1. Every vertex in L has degree at most d.
2. Every $S \subseteq L$ with $|S| \leq \alpha n$ has at least $c|S|$ neighbors in R.

This is an example of an *expanding graph*.

For most applications we want d small and c large.
Usually d, α, c are constants and $c > 1$.

OR-concentrator: Theorem

Theorem

There is an integer n_0 such that for all $n > n_0$, there exists an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.

Let each $v \in L$ choose d neighbors in R, independently and uniformly at random, with replacement.

Let $\mathcal{E}_s$ be the *bad* event that some $S \subseteq L$ with $|S| = s$ has less than cs neighbors in R.

We will first bound $\Pr[\mathcal{E}_s]$ and then show $\Pr[\cup_{s=1}^{\alpha n} \mathcal{E}_s] < 1$.

Since $\Pr[\text{good}] = 1 - \Pr[\cup_{s=1}^{\alpha n} \mathcal{E}_s] > 0$, the result follows.

OR-concentrator: Theorem

Theorem

There is an integer n_0 such that for all $n > n_0$, there exists an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.

Let each $v \in L$ choose d neighbors in R, independently and uniformly at random, with replacement.

Let $\mathcal{E}_s$ be the *bad* event that some $S \subseteq L$ with $|S| = s$ has less than cs neighbors in R.

We will first bound $\Pr[\mathcal{E}_s]$ and then show $\Pr[\cup_{s=1}^{\alpha n} \mathcal{E}_s] < 1$.

Since $\Pr[\text{good}] = 1 - \Pr[\cup_{s=1}^{\alpha n} \mathcal{E}_s] > 0$, the result follows.

OR-concentrator: Theorem

Theorem

There is an integer n_0 such that for all $n > n_0$, there exists an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.

Let each $v \in L$ choose d neighbors in R, independently and uniformly at random, with replacement.

Let $\mathcal{E}_s$ be the *bad* event that some $S \subseteq L$ with $|S| = s$ has less than cs neighbors in R.

We will first bound $\Pr[\mathcal{E}_s]$ and then show $\Pr[\cup_{s=1}^{\alpha n} \mathcal{E}_s] < 1$.

Since $\Pr[\text{good}] = 1 - \Pr[\cup_{s=1}^{\alpha n} \mathcal{E}_s] > 0$, the result follows.

OR-concentrator: Theorem

Theorem

There is an integer n_0 such that for all $n > n_0$, there exists an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.

Let each $v \in L$ choose d neighbors in R, independently and uniformly at random, with replacement.

Let $\mathcal{E}_s$ be the *bad* event that some $S \subseteq L$ with $|S| = s$ has less than cs neighbors in R.

We will first bound $\Pr[\mathcal{E}_s]$ and then show $\Pr[\cup_{s=1}^{\alpha n} \mathcal{E}_s] < 1$.

Since $\Pr[\text{good}] = 1 - \Pr[\cup_{s=1}^{\alpha n} \mathcal{E}_s] > 0$, the result follows.

OR-concentrator: Theorem

Theorem

There is an integer n_0 such that for all $n > n_0$, there exists an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.

Let each $v \in L$ choose d neighbors in R, independently and uniformly at random, with replacement.

Let $\mathcal{E}_s$ be the *bad* event that some $S \subseteq L$ with $|S| = s$ has less than cs neighbors in R.

We will first bound $\Pr[\mathcal{E}_s]$ and then show $\Pr[\cup_{s=1}^{\alpha n} \mathcal{E}_s] < 1$.

Since $\Pr[\text{good}] = 1 - \Pr[\cup_{s=1}^{\alpha n} \mathcal{E}_s] > 0$, the result follows.

OR-concentrator: Proof

$$
\begin{aligned}
\Pr[\mathcal{E}_s] &= \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } < cs \text{ neighbors in } R\right] \\
&\le \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } \le cs \text{ neighbors in } R\right] \\
&\le \binom{n}{s}\binom{n}{cs}\left(\frac{cs}{n}\right)^{ds} && \text{(By a union bound)} \\
&\le \left(\frac{ne}{s}\right)^{s}\left(\frac{ne}{cs}\right)^{cs}\left(\frac{cs}{n}\right)^{ds} && \left(\text{Using } \tbinom{n}{k} \le \left(\tfrac{ne}{k}\right)^{k}\right) \\
&= \left(\left(\frac{s}{n}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} \\
&\le \left(\left(\frac{1}{3}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} && \left(\text{Using } \alpha = \tfrac{1}{3} \text{ and } s \le \alpha n\right) \\
&= \left(\frac{c^{d-c}}{3^{d}}(3e)^{1+c}\right)^{s} \\
&\le \left(\left(\frac{c}{3}\right)^{d}(3e)^{1+c}\right)^{s} && \text{(Using } c \ge 1) \\
&= \left(\left(\frac{2}{3}\right)^{18}(3e)^{3}\right)^{s} && \text{(Using } c = 2 \text{ and } d = 18) \\
&= r^{s} && \left(\text{Where } r := \left(\tfrac{2}{3}\right)^{18}(3e)^{3} \approx 0.367 < \tfrac{1}{2}\right)
\end{aligned}
$$

By definition of $\mathcal{E}_s$.
The notation $\binom{L}{s}$ means all subsets of L of size s.

OR-concentrator: Proof

$$
\begin{aligned}
\Pr[\mathcal{E}_s] &= \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } < cs \text{ neighbors in } R\right] \\
&\leq \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } \leq cs \text{ neighbors in } R\right] \\
&\leq \tbinom{n}{s}\tbinom{n}{cs}\left(\tfrac{cs}{n}\right)^{ds} && \text{(By a union bound)} \\
&\leq \left(\tfrac{ne}{s}\right)^{s}\left(\tfrac{ne}{cs}\right)^{cs}\left(\tfrac{cs}{n}\right)^{ds} && \text{(Using } \tbinom{n}{k} \leq \left(\tfrac{ne}{k}\right)^{k}\text{)} \\
&= \left(\left(\tfrac{s}{n}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} \\
&\leq \left(\left(\tfrac{1}{3}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} && \text{(Using } \alpha = \tfrac{1}{3} \text{ and } s \leq \alpha n\text{)} \\
&= \left(\tfrac{c^{d-c}}{3^{d}}(3e)^{1+c}\right)^{s} \\
&\leq \left(\left(\tfrac{c}{3}\right)^{d}(3e)^{1+c}\right)^{s} && \text{(Using } c \geq 1\text{)} \\
&= \left(\left(\tfrac{2}{3}\right)^{18}(3e)^{3}\right)^{s} && \text{(Using } c = 2 \text{ and } d = 18\text{)} \\
&= r^{s} && \text{(Where } r := \left(\tfrac{2}{3}\right)^{18}(3e)^{3} \approx 0.367 < \tfrac{1}{2}\text{)}
\end{aligned}
$$

Since this event is implied by the first one.

OR-concentrator: Proof

$$\begin{aligned}
\Pr[\mathcal{E}_s] &= \Pr\big[\text{Some } S \in \tbinom{L}{s} \text{ has } < cs \text{ neighbors in } R\big] \\
&\le \Pr\big[\text{Some } S \in \tbinom{L}{s} \text{ has } \le cs \text{ neighbors in } R\big] \\
&\le \tbinom{n}{s}\tbinom{n}{cs}\left(\tfrac{cs}{n}\right)^{ds} && \text{(By a union bound)} \\
&\le \left(\tfrac{ne}{s}\right)^{s}\left(\tfrac{ne}{cs}\right)^{cs}\left(\tfrac{cs}{n}\right)^{ds} && \text{(Using } \tbinom{n}{k} \le \left(\tfrac{ne}{k}\right)^{k}\text{)} \\
&= \left(\left(\tfrac{s}{n}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} \\
&\le \left(\left(\tfrac{1}{3}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} && \text{(Using } \alpha = \tfrac{1}{3} \text{ and } s \le \alpha n\text{)} \\
&= \left(\tfrac{c^{d-c}}{3^{d}}(3e)^{1+c}\right)^{s} \\
&\le \left(\left(\tfrac{c}{3}\right)^{d}(3e)^{1+c}\right)^{s} && \text{(Using } c \ge 1\text{)} \\
&= \left(\left(\tfrac{2}{3}\right)^{18}(3e)^{3}\right)^{s} && \text{(Using } c = 2 \text{ and } d = 18\text{)} \\
&= r^{s} && \text{(Where } r := \left(\tfrac{2}{3}\right)^{18}(3e)^{3} \approx 0.367 < \tfrac{1}{2}\text{)}
\end{aligned}$$

There are $\binom{n}{s}$ ways of choosing a set $S \in \binom{L}{s}$, and $\binom{n}{cs}$ ways of choosing a set $T \in \binom{R}{cs}$.

For any such combination of S and T, the probability that all ds neighbors chosen by S are in T is $\left(\frac{cs}{n}\right)^{ds}$.

The probability that this happens for *some* such pair (S, T) is at most the sum of probabilities that it happens for each.

"The probability of a union of events is at most the sum of probabilities of the individual events."

This type of bound is called a *union bound*.

OR-concentrator: Proof

$$
\begin{aligned}
\Pr[\mathcal{E}_s] &= \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } < cs \text{ neighbors in } R\right] \\
&\le \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } \le cs \text{ neighbors in } R\right] \\
&\le \binom{n}{s}\binom{n}{cs}\left(\frac{cs}{n}\right)^{ds} && \text{(By a union bound)} \\
&\le \left(\frac{ne}{s}\right)^{s}\left(\frac{ne}{cs}\right)^{cs}\left(\frac{cs}{n}\right)^{ds} && \text{(Using } \tbinom{n}{k} \le \left(\tfrac{ne}{k}\right)^{k}\text{)} \\
&= \left(\left(\frac{s}{n}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} \\
&\le \left(\left(\frac{1}{3}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} && \text{(Using } \alpha = \tfrac{1}{3} \text{ and } s \le \alpha n\text{)} \\
&= \left(\frac{c^{d-c}}{3^{d}}(3e)^{1+c}\right)^{s} \\
&\le \left(\left(\frac{c}{3}\right)^{d}(3e)^{1+c}\right)^{s} && \text{(Using } c \ge 1\text{)} \\
&= \left(\left(\frac{2}{3}\right)^{18}(3e)^{3}\right)^{s} && \text{(Using } c = 2 \text{ and } d = 18\text{)} \\
&= r^{s} && \text{(Where } r := \left(\tfrac{2}{3}\right)^{18}(3e)^{3} \approx 0.367 < \tfrac{1}{2}\text{)}
\end{aligned}
$$

OR-concentrator: Proof

$$
\begin{aligned}
\Pr[\mathcal{E}_s] &= \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } < cs \text{ neighbors in } R\right] \\
&\leq \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } \leq cs \text{ neighbors in } R\right] \\
&\leq \tbinom{n}{s}\tbinom{n}{cs}\left(\tfrac{cs}{n}\right)^{ds} && \text{(By a union bound)} \\
&\leq \left(\tfrac{ne}{s}\right)^{s}\left(\tfrac{ne}{cs}\right)^{cs}\left(\tfrac{cs}{n}\right)^{ds} && \text{(Using } \tbinom{n}{k} \leq \left(\tfrac{ne}{k}\right)^{k}\text{)} \\
&= \left(\left(\tfrac{s}{n}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} \\
&\leq \left(\left(\tfrac{1}{3}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} && \text{(Using } \alpha = \tfrac{1}{3} \text{ and } s \leq \alpha n\text{)} \\
&= \left(\tfrac{c^{d-c}}{3^{d}}(3e)^{1+c}\right)^{s} \\
&\leq \left(\left(\tfrac{c}{3}\right)^{d}(3e)^{1+c}\right)^{s} && \text{(Using } c \geq 1\text{)} \\
&= \left(\left(\tfrac{2}{3}\right)^{18}(3e)^{3}\right)^{s} && \text{(Using } c = 2 \text{ and } d = 18\text{)} \\
&= r^{s} && \text{(Where } r := \left(\tfrac{2}{3}\right)^{18}(3e)^{3} \approx 0.367 < \tfrac{1}{2}\text{)}
\end{aligned}
$$

OR-concentrator: Proof

$$
\begin{aligned}
\Pr[\mathcal{E}_s] &= \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } < cs \text{ neighbors in } R\right] \\
&\leq \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } \leq cs \text{ neighbors in } R\right] \\
&\leq \tbinom{n}{s}\tbinom{n}{cs}\left(\tfrac{cs}{n}\right)^{ds} && \text{(By a union bound)} \\
&\leq \left(\tfrac{ne}{s}\right)^{s}\left(\tfrac{ne}{cs}\right)^{cs}\left(\tfrac{cs}{n}\right)^{ds} && \text{(Using } \tbinom{n}{k} \leq \left(\tfrac{ne}{k}\right)^{k}\text{)} \\
&= \left(\left(\tfrac{s}{n}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} \\
&\leq \left(\left(\tfrac{1}{3}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} && \text{(Using } \alpha = \tfrac{1}{3} \text{ and } s \leq \alpha n\text{)} \\
&= \left(\tfrac{c^{d-c}}{3^{d}}(3e)^{1+c}\right)^{s} \\
&\leq \left(\left(\tfrac{c}{3}\right)^{d}(3e)^{1+c}\right)^{s} && \text{(Using } c \geq 1\text{)} \\
&= \left(\left(\tfrac{2}{3}\right)^{18}(3e)^{3}\right)^{s} && \text{(Using } c = 2 \text{ and } d = 18\text{)} \\
&= r^{s} && \text{(Where } r := \left(\tfrac{2}{3}\right)^{18}(3e)^{3} \approx 0.367 < \tfrac{1}{2}\text{)}
\end{aligned}
$$

OR-concentrator: Proof

$$
\begin{aligned}
\Pr[\mathcal{E}_s] &= \Pr\left[\text{Some } S \in \binom{L}{s} \text{ has } < cs \text{ neighbors in } R\right] \\
&\le \Pr\left[\text{Some } S \in \binom{L}{s} \text{ has } \le cs \text{ neighbors in } R\right] \\
&\le \binom{n}{s}\binom{n}{cs}\left(\frac{cs}{n}\right)^{ds} && \text{(By a union bound)} \\
&\le \left(\frac{ne}{s}\right)^{s}\left(\frac{ne}{cs}\right)^{cs}\left(\frac{cs}{n}\right)^{ds} && \left(\text{Using } \binom{n}{k} \le \left(\frac{ne}{k}\right)^{k}\right) \\
&= \left(\left(\frac{s}{n}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} \\
&\le \left(\left(\frac{1}{3}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} && \left(\text{Using } \alpha = \frac{1}{3} \text{ and } s \le \alpha n\right) \\
&= \left(\frac{c^{d-c}}{3^{d}}(3e)^{1+c}\right)^{s} \\
&\le \left(\left(\frac{c}{3}\right)^{d}(3e)^{1+c}\right)^{s} && \text{(Using } c \ge 1) \\
&= \left(\left(\frac{2}{3}\right)^{18}(3e)^{3}\right)^{s} && \text{(Using } c = 2 \text{ and } d = 18) \\
&= r^{s} && \left(\text{Where } r := \left(\frac{2}{3}\right)^{18}(3e)^{3} \approx 0.367 < \frac{1}{2}\right)
\end{aligned}
$$

OR-concentrator: Proof

Thus is just sloppy. Dropping this step would allow you to reduce d to 14 and still get an $r \approx 0.464 < \frac{1}{2}$ in the last step.

$$
\begin{aligned}
\Pr[\mathcal{E}_s] &= \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } < cs \text{ neighbors in } R\right] \\
&\leq \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } \leq cs \text{ neighbors in } R\right] \\
&\leq \tbinom{n}{s}\tbinom{n}{cs}\left(\tfrac{cs}{n}\right)^{ds} && \text{(By a union bound)} \\
&\leq \left(\tfrac{ne}{s}\right)^{s}\left(\tfrac{ne}{cs}\right)^{cs}\left(\tfrac{cs}{n}\right)^{ds} && \text{(Using } \tbinom{n}{k} \leq \left(\tfrac{ne}{k}\right)^{k}\text{)} \\
&= \left(\left(\tfrac{s}{n}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} \\
&\leq \left(\left(\tfrac{1}{3}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} && \text{(Using } \alpha = \tfrac{1}{3} \text{ and } s \leq \alpha n\text{)} \\
&= \left(\tfrac{c^{d-c}}{3^{d}}(3e)^{1+c}\right)^{s} \\
&\leq \left(\left(\tfrac{c}{3}\right)^{d}(3e)^{1+c}\right)^{s} && \text{(Using } c \geq 1\text{)} \\
&= \left(\left(\tfrac{2}{3}\right)^{18}(3e)^{3}\right)^{s} && \text{(Using } c = 2 \text{ and } d = 18\text{)} \\
&= r^{s} && \text{(Where } r := \left(\tfrac{2}{3}\right)^{18}(3e)^{3} \approx 0.367 < \tfrac{1}{2}\text{)}
\end{aligned}
$$

OR-concentrator: Proof

$$\begin{aligned}
\Pr[\mathcal{E}_s] &= \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } < cs \text{ neighbors in } R\right] \\
&\le \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } \le cs \text{ neighbors in } R\right] \\
&\le \tbinom{n}{s}\tbinom{n}{cs}\left(\tfrac{cs}{n}\right)^{ds} && \text{(By a union bound)} \\
&\le \left(\tfrac{ne}{s}\right)^{s}\left(\tfrac{ne}{cs}\right)^{cs}\left(\tfrac{cs}{n}\right)^{ds} && \left(\text{Using } \tbinom{n}{k} \le \left(\tfrac{ne}{k}\right)^{k}\right) \\
&= \left(\left(\tfrac{s}{n}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} \\
&\le \left(\left(\tfrac{1}{3}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} && \left(\text{Using } \alpha = \tfrac{1}{3} \text{ and } s \le \alpha n\right) \\
&= \left(\tfrac{c^{d-c}}{3^{d}}(3e)^{1+c}\right)^{s} \\
&\le \left(\left(\tfrac{c}{3}\right)^{d}(3e)^{1+c}\right)^{s} && \text{(Using } c \ge 1) \\
&= \left(\left(\tfrac{2}{3}\right)^{18}(3e)^{3}\right)^{s} && \text{(Using } c = 2 \text{ and } d = 18) \\
&= r^{s} && \left(\text{Where } r := \left(\tfrac{2}{3}\right)^{18}(3e)^{3} \approx 0.367 < \tfrac{1}{2}\right)
\end{aligned}$$

OR-concentrator: Proof

$$
\begin{aligned}
\Pr[\mathcal{E}_s] &= \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } < cs \text{ neighbors in } R\right] \\
&\le \Pr\left[\text{Some } S \in \tbinom{L}{s} \text{ has } \le cs \text{ neighbors in } R\right] \\
&\le \binom{n}{s}\binom{n}{cs}\left(\frac{cs}{n}\right)^{ds} && \text{(By a union bound)} \\
&\le \left(\frac{ne}{s}\right)^{s}\left(\frac{ne}{cs}\right)^{cs}\left(\frac{cs}{n}\right)^{ds} && \text{(Using } \tbinom{n}{k} \le \left(\tfrac{ne}{k}\right)^{k}\text{)} \\
&= \left(\left(\frac{s}{n}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} \\
&\le \left(\left(\frac{1}{3}\right)^{d-c-1} e^{1+c} c^{d-c}\right)^{s} && \text{(Using } \alpha = \tfrac{1}{3} \text{ and } s \le \alpha n\text{)} \\
&= \left(\frac{c^{d-c}}{3^{d}}(3e)^{1+c}\right)^{s} \\
&\le \left(\left(\frac{c}{3}\right)^{d}(3e)^{1+c}\right)^{s} && \text{(Using } c \ge 1\text{)} \\
&= \left(\left(\frac{2}{3}\right)^{18}(3e)^{3}\right)^{s} && \text{(Using } c = 2 \text{ and } d = 18\text{)} \\
&= r^{s} && \text{(Where } r := \left(\tfrac{2}{3}\right)^{18}(3e)^{3} \approx 0.367 < \tfrac{1}{2}\text{)}
\end{aligned}
$$

OR-concentrator: Proof

$$\begin{aligned}
\Pr[\cup_{s=1}^{\alpha n} \mathcal{E}_s] &\le \sum_{s=1}^{\alpha n} \Pr[\mathcal{E}_s] && \text{(Union bound)} \\
&\le \sum_{s=1}^{\alpha n} r^s && \text{(Last slide, since } s \le \alpha n\text{)} \\
&= r \sum_{s=0}^{\alpha n - 1} r^s \\
&< r \sum_{s=0}^{\infty} r^s && \text{(Since } r > 0\text{)} \\
&= \frac{r}{1-r} < 1 && \text{(Since } r < \tfrac{1}{2}\text{)}
\end{aligned}$$

$\square$

OR-concentrator: Proof

$$\begin{aligned}
\Pr[\cup_{s=1}^{\alpha n}\mathcal{E}_s] &\le \sum_{s=1}^{\alpha n}\Pr[\mathcal{E}_s] && \text{(Union bound)}\\
&\le \sum_{s=1}^{\alpha n} r^s && \text{(Last slide, since } s \le \alpha n\text{)}\\
&= r\sum_{s=0}^{\alpha n-1} r^s \\
&< r\sum_{s=0}^{\infty} r^s && \text{(Since } r > 0\text{)}\\
&= \frac{r}{1-r} < 1 && \text{(Since } r < \tfrac{1}{2}\text{)}
\end{aligned}$$

$\square$

OR-concentrator: Proof

$$
\begin{aligned}
\Pr[\cup_{s=1}^{\alpha n} \mathcal{E}_s] &\le \sum_{s=1}^{\alpha n} \Pr[\mathcal{E}_s] && \text{(Union bound)} \\
&\le \sum_{s=1}^{\alpha n} r^s && \text{(Last slide, since } s \le \alpha n\text{)} \\
&= r \sum_{s=0}^{\alpha n - 1} r^s \\
&< r \sum_{s=0}^{\infty} r^s && \text{(Since } r > 0\text{)} \\
&= \frac{r}{1-r} < 1 && \text{(Since } r < \tfrac{1}{2}\text{)}
\end{aligned}
$$

□

OR-concentrator: Proof

$$
\begin{aligned}
\Pr[\cup_{s=1}^{\alpha n} \mathcal{E}_s] &\leq \sum_{s=1}^{\alpha n} \Pr[\mathcal{E}_s] && \text{(Union bound)} \\
&\leq \sum_{s=1}^{\alpha n} r^s && \text{(Last slide, since } s \leq \alpha n\text{)} \\
&= r \sum_{s=0}^{\alpha n - 1} r^s \\
&< r \sum_{s=0}^{\infty} r^s && \text{(Since } r > 0\text{)} \\
&= \frac{r}{1-r} < 1 && \text{(Since } r < \tfrac{1}{2}\text{)}
\end{aligned}
$$

$\square$

OR-concentrator: Proof

$$
\begin{aligned}
\Pr[\cup_{s=1}^{\alpha n} \mathcal{E}_s] &\leq \sum_{s=1}^{\alpha n} \Pr[\mathcal{E}_s] && \text{(Union bound)} \\
&\leq \sum_{s=1}^{\alpha n} r^s && \text{(Last slide, since } s \leq \alpha n\text{)} \\
&= r \sum_{s=0}^{\alpha n - 1} r^s \\
&< r \sum_{s=0}^{\infty} r^s && \text{(Since } r > 0\text{)} \\
&= \frac{r}{1-r} < 1 && \text{(Since } r < \tfrac{1}{2}\text{)}
\end{aligned}
$$

$\square$

OR-concentrator: Summary

We used the probabilistic method to show that an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator exists.

A more careful analysis shows that with "good" probability, the random graph described in the proof is an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.

Does this give us a Monte Carlo algorithm for finding one?

OR-concentrator: Summary

We used the probabilistic method to show that an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator exists.

A more careful analysis shows that with "good" probability, the random graph described in the proof is an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.

Does this give us a Monte Carlo algorithm for finding one?

OR-concentrator: Summary

We used the probabilistic method to show that an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator exists.

A more careful analysis shows that with "good" probability, the random graph described in the proof is an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.

Does this give us a Monte Carlo algorithm for finding one?

OR-concentrator: Summary

We used the probabilistic method to show that an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator exists.

A more careful analysis shows that with "good" probability, the random graph described in the proof is an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.

Does this give us a Monte Carlo algorithm for finding one? Yes

OR-concentrator: Summary

We used the probabilistic method to show that an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator exists.

A more careful analysis shows that with "good" probability, the random graph described in the proof is an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.

Does this give us a Monte Carlo algorithm for finding one? Yes

Can this be turned into a Las Vegas algorithm?

OR-concentrator: Summary

We used the probabilistic method to show that an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator exists.

A more careful analysis shows that with "good" probability, the random graph described in the proof is an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.

Does this give us a Monte Carlo algorithm for finding one? Yes

Can this be turned into a Las Vegas algorithm? No

OR-concentrator: Summary

We used the probabilistic method to show that an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator exists.

A more careful analysis shows that with "good" probability, the random graph described in the proof is an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.

Does this give us a Monte Carlo algorithm for finding one? Yes

Can this be turned into a Las Vegas algorithm? No

Why not?

OR-concentrator: Summary

We used the probabilistic method to show that an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator exists.

A more careful analysis shows that with "good" probability, the random graph described in the proof is an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.

Does this give us a Monte Carlo algorithm for finding one? Yes

Can this be turned into a Las Vegas algorithm? No

Why not? No fast way to tell if a given graph is one.

Probability Amplification

Recall from Lecture 3:

Definition

Let $\mathcal{L} \subseteq \Sigma^\star$ be some language, and let n be a prime.
A function $A : \Sigma^\star \times \mathbb{Z}_n \to \{0, 1\}$ is an **RP** algorithm for deciding $\mathcal{L}$, if it runs in polynomial time for all inputs, and

If $x \in \mathcal{L}$, then $A(x, r) = 1$ for at least half of all $r \in \mathbb{Z}_n$.

If $x \notin \mathcal{L}$, then $A(x, r) = 0$ for all $r \in \mathbb{Z}_n$.

Probability Amplification

Using $2\log_2 n$ random bits to sample two numbers from $[n]$ and applying two-point sampling, we can construct a sequence $r_0, \ldots, r_{t-1} \in [n]$ of $t \leq n$ samples, such that for any $x \in \mathcal{L}$,

$$\Pr\left[\forall i \in [t] : A(x, r_i) = 0\right] \leq \frac{1}{t}$$

We will show that there exists an algorithm using only $\log_2^2 n$ random bits that constructs a sequence $S \subseteq [n]$ of at most $12\log_2^2 n$ samples, such that for any $x \in \mathcal{L}$,

$$\Pr\left[\forall r \in S : A(x, r) = 0\right] \leq \frac{1}{n^{(\log_2 n)-1}}$$

Probability Amplification

Using $2\log_2 n$ random bits to sample two numbers from $[n]$ and applying two-point sampling, we can construct a sequence $r_0, \ldots, r_{t-1} \in [n]$ of $t \le n$ samples, such that for any $x \in \mathcal{L}$,

$$\Pr\left[\forall i \in [t] : A(x, r_i) = 0\right] \le \frac{1}{t}$$

We will show that there exists an algorithm using only $\log_2^2 n$ random bits that constructs a sequence $S \subseteq [n]$ of at most $12\log_2^2 n$ samples, such that for any $x \in \mathcal{L}$,

$$\Pr\left[\forall r \in S : A(x, r) = 0\right] \le \frac{1}{n^{(\log_2 n)-1}}$$

Note that the naive way to use $\log_2^2 n$ bits, which just picks $\log_2 n$ elements from $[n]$ independently and uniformly at random, only gives an error probability of $(\frac{1}{2})^{\log_2 n} = \frac{1}{n}$. Thus, in a sense this technique *amplifies* the probability of success.

Probability Amplification

We'll need the graph from this

Theorem

For n sufficiently large, there exists a bipartite graph $G(L, R, E)$ with $|L| = n$, $|R| = 2^{\log_2^2 n}$ such that:

1. *Every subset of $\frac{n}{2}$ vertices of L has at least $2^{\log_2^2 n} - n$ neigbors in R.*
2. *No vertex in R has more than $12 \log_2^2 n$ neighbors.*

Proof of Probability Amplification.

Let $x \in \mathcal{L}$ and let $S_x = \{r \in L \mid A(x, r) = 1\}$ be the *witnesses* for x. By definition, $|S_x| \geq \frac{n}{2}$, so S_x has at least $2^{\log_2^2 n} - n$ neighbors in R. Use the $\log_2^2 n$ random bits to pick a random node in R, and let $S \subseteq L$ be its neighbors.

$\Pr[\text{fail}] = \Pr[S \cap S_x = \emptyset] \leq 1 - \frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}} = \frac{n}{2^{\log_2^2 n}} = \frac{1}{n^{(\log_2 n) - 1}}$. □

The theorem is only interesting when $n \geq 12 \log_2^2 n$, which happens when $n \geq 1278$.

Note that the R side of this graph is *huge*. For $n = 1278$, it is larger than 10^{32}.

Thus, there is no hope of representing this graph explicitly. But we can hope for some *implicit* representation.

In particular, we'll assume we have a function $N_G : R \to 2^L$ that given a node $v \in R$ returns its neighbors in L.

Probability Amplification

We'll need the graph from this

Theorem

For n sufficiently large, there exists a bipartite graph $G(L, R, E)$ with $|L| = n$, $|R| = 2^{\log_2^2 n}$ such that:

1. *Every subset of $\frac{n}{2}$ vertices of L has at least $2^{\log_2^2 n} - n$ neigbors in R.*
2. *No vertex in R has more than $12 \log_2^2 n$ neighbors.*

Proof of Probability Amplification.

Let $x \in \mathcal{L}$ and let $S_x = \{r \in L \mid A(x, r) = 1\}$ be the *witnesses* for x. By definition, $|S_x| \geq \frac{n}{2}$, so S_x has at least $2^{\log_2^2 n} - n$ neighbors in R. Use the $\log_2^2 n$ random bits to pick a random node in R, and let $S \subseteq L$ be its neighbors.

$\Pr[\text{fail}] = \Pr[S \cap S_x = \emptyset] \leq 1 - \frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}} = \frac{n}{2^{\log_2^2 n}} = \frac{1}{n^{(\log_2 n) - 1}}$. $\square$

Probability Amplification

We'll need the graph from this

Theorem

For n sufficiently large, there exists a bipartite graph $G(L, R, E)$ with $|L| = n$, $|R| = 2^{\log_2^2 n}$ such that:

1. *Every subset of $\frac{n}{2}$ vertices of L has at least $2^{\log_2^2 n} - n$ neigbors in R.*
2. *No vertex in R has more than $12 \log_2^2 n$ neighbors.*

Proof of Probability Amplification.

Let $x \in \mathcal{L}$ and let $S_x = \{r \in L \mid A(x, r) = 1\}$ be the *witnesses* for x. By definition, $|S_x| \geq \frac{n}{2}$, so S_x has at least $2^{\log_2^2 n} - n$ neighbors in R. Use the $\log_2^2 n$ random bits to pick a random node in R, and let $S \subseteq L$ be its neighbors.

$\Pr[\text{fail}] = \Pr[S \cap S_x = \emptyset] \leq 1 - \frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}} = \frac{n}{2^{\log_2^2 n}} = \frac{1}{n^{(\log_2 n) - 1}}$. $\square$

Probability Amplification

We'll need the graph from this

Theorem

For n sufficiently large, there exists a bipartite graph $G(L, R, E)$ with $|L| = n$, $|R| = 2^{\log_2^2 n}$ such that:

1. *Every subset of $\frac{n}{2}$ vertices of L has at least $2^{\log_2^2 n} - n$ neigbors in R.*
2. *No vertex in R has more than $12 \log_2^2 n$ neighbors.*

Proof of Probability Amplification.

Let $x \in \mathcal{L}$ and let $S_x = \{r \in L \mid A(x, r) = 1\}$ be the *witnesses* for x. By definition, $|S_x| \geq \frac{n}{2}$, so S_x has at least $2^{\log_2^2 n} - n$ neighbors in R. Use the $\log_2^2 n$ random bits to pick a random node in R, and let $S \subseteq L$ be its neighbors.

$\Pr[\text{fail}] = \Pr[S \cap S_x = \emptyset] \leq 1 - \frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}} = \frac{n}{2^{\log_2^2 n}} = \frac{1}{n^{(\log_2 n) - 1}}$. □

Probability Amplification

We'll need the graph from this

Theorem

For n sufficiently large, there exists a bipartite graph $G(L, R, E)$ with $|L| = n$, $|R| = 2^{\log_2^2 n}$ such that:

1. *Every subset of $\frac{n}{2}$ vertices of L has at least $2^{\log_2^2 n} - n$ neigbors in R.*
2. *No vertex in R has more than $12 \log_2^2 n$ neighbors.*

Proof of Probability Amplification.

Let $x \in \mathcal{L}$ and let $S_x = \{r \in L \mid A(x, r) = 1\}$ be the *witnesses* for x. By definition, $|S_x| \geq \frac{n}{2}$, so S_x has at least $2^{\log_2^2 n} - n$ neighbors in R. Use the $\log_2^2 n$ random bits to pick a random node in R, and let $S \subseteq L$ be its neighbors.

$\Pr[\text{fail}] = \Pr[S \cap S_x = \emptyset] \leq 1 - \frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}} = \frac{n}{2^{\log_2^2 n}} = \frac{1}{n^{(\log_2 n) - 1}}$. □

Probability Amplification

We'll need the graph from this

Theorem

For n sufficiently large, there exists a bipartite graph $G(L, R, E)$ with $|L| = n$, $|R| = 2^{\log_2^2 n}$ such that:

1. *Every subset of $\frac{n}{2}$ vertices of L has at least $2^{\log_2^2 n} - n$ neigbors in R.*
2. *No vertex in R has more than $12 \log_2^2 n$ neighbors.*

Proof of Probability Amplification.

Let $x \in \mathcal{L}$ and let $S_x = \{r \in L \mid A(x, r) = 1\}$ be the *witnesses* for x. By definition, $|S_x| \geq \frac{n}{2}$, so S_x has at least $2^{\log_2^2 n} - n$ neighbors in R. Use the $\log_2^2 n$ random bits to pick a random node in R, and let $S \subseteq L$ be its neighbors.

$\Pr[\text{fail}] = \Pr[S \cap S_x = \emptyset] \leq 1 - \frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}} = \frac{n}{2^{\log_2^2 n}} = \frac{1}{n^{(\log_2 n) - 1}}$. □

This uses the function $N_G(v)$ that we assumed to exist.

Probability Amplification

We'll need the graph from this

Theorem

For n sufficiently large, there exists a bipartite graph $G(L, R, E)$ with $|L| = n$, $|R| = 2^{\log_2^2 n}$ such that:

1. *Every subset of $\frac{n}{2}$ vertices of L has at least $2^{\log_2^2 n} - n$ neigbors in R.*
2. *No vertex in R has more than $12 \log_2^2 n$ neighbors.*

Proof of Probability Amplification.

Let $x \in \mathcal{L}$ and let $S_x = \{r \in L \mid A(x, r) = 1\}$ be the *witnesses* for x. By definition, $|S_x| \geq \frac{n}{2}$, so S_x has at least $2^{\log_2^2 n} - n$ neighbors in R. Use the $\log_2^2 n$ random bits to pick a random node in R, and let $S \subseteq L$ be its neighbors.

$\Pr[\text{fail}] = \Pr[S \cap S_x = \emptyset] \leq 1 - \frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}} = \frac{n}{2^{\log_2^2 n}} = \frac{1}{n^{(\log_2 n) - 1}}$. $\square$

Probability Amplification

We'll need the graph from this

Theorem

For n sufficiently large, there exists a bipartite graph $G(L, R, E)$ with $|L| = n$, $|R| = 2^{\log_2^2 n}$ such that:

1. *Every subset of $\frac{n}{2}$ vertices of L has at least $2^{\log_2^2 n} - n$ neigbors in R.*
2. *No vertex in R has more than $12 \log_2^2 n$ neighbors.*

Proof of Probability Amplification.

Let $x \in \mathcal{L}$ and let $S_x = \{r \in L \mid A(x, r) = 1\}$ be the *witnesses* for x. By definition, $|S_x| \geq \frac{n}{2}$, so S_x has at least $2^{\log_2^2 n} - n$ neighbors in R. Use the $\log_2^2 n$ random bits to pick a random node in R, and let $S \subseteq L$ be its neighbors.

$\Pr[\text{fail}] = \Pr[S \cap S_x = \emptyset] \leq 1 - \frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}} = \frac{n}{2^{\log_2^2 n}} = \frac{1}{n^{(\log_2 n) - 1}}$. □

Probability Amplification

We'll need the graph from this

Theorem

For n sufficiently large, there exists a bipartite graph $G(L, R, E)$ with $|L| = n$, $|R| = 2^{\log_2^2 n}$ such that:

1. *Every subset of $\frac{n}{2}$ vertices of L has at least $2^{\log_2^2 n} - n$ neigbors in R.*
2. *No vertex in R has more than $12 \log_2^2 n$ neighbors.*

Proof of Probability Amplification.

Let $x \in \mathcal{L}$ and let $S_x = \{r \in L \mid A(x, r) = 1\}$ be the *witnesses* for x. By definition, $|S_x| \geq \frac{n}{2}$, so S_x has at least $2^{\log_2^2 n} - n$ neighbors in R. Use the $\log_2^2 n$ random bits to pick a random node in R, and let $S \subseteq L$ be its neighbors.

$\Pr[\text{fail}] = \Pr[S \cap S_x = \emptyset] \leq 1 - \frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}} = \frac{n}{2^{\log_2^2 n}} = \frac{1}{n^{(\log_2 n) - 1}}$. $\square$

Probability Amplification

We'll need the graph from this

Theorem

For n sufficiently large, there exists a bipartite graph $G(L, R, E)$ with $|L| = n$, $|R| = 2^{\log_2^2 n}$ such that:

1. *Every subset of $\frac{n}{2}$ vertices of L has at least $2^{\log_2^2 n} - n$ neigbors in R.*
2. *No vertex in R has more than $12 \log_2^2 n$ neighbors.*

Proof of Probability Amplification.

Let $x \in \mathcal{L}$ and let $S_x = \{r \in L \mid A(x, r) = 1\}$ be the *witnesses* for x. By definition, $|S_x| \geq \frac{n}{2}$, so S_x has at least $2^{\log_2^2 n} - n$ neighbors in R. Use the $\log_2^2 n$ random bits to pick a random node in R, and let $S \subseteq L$ be its neighbors.

$\Pr[\text{fail}] = \Pr[S \cap S_x = \emptyset] \leq 1 - \frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}} = \frac{n}{2^{\log_2^2 n}} = \frac{1}{n^{(\log_2 n) - 1}}$. $\square$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$\begin{aligned}
\Pr[\text{fail } 1] &= \Pr\left[\text{Some } S \in \binom{L}{\frac{n}{2}} \text{ has } < 2^{\log_2^2 n} - n \text{ neighbors in } R\right] \\
&\le \Pr\left[\text{Some } S \in \binom{L}{\frac{n}{2}} \text{ has } \le 2^{\log_2^2 n} - n \text{ neighbors in } R\right] \\
&\le \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{2^{\log_2^2 n} - n} \left(\frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}}\right)^{\frac{dn}{2}} \qquad \text{(By union bound)} \\
&= \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{n} \left(1 - \frac{n}{2^{\log_2^2 n}}\right)^{\frac{dn}{2}} \\
&\le \left(\frac{ne}{\frac{n}{2}}\right)^{\frac{n}{2}} \left(\frac{2^{\log_2^2 n} e}{n}\right)^{n} \left(e^{-\frac{n}{2^{\log_2^2 n}}}\right)^{\frac{dn}{2}} \\
&= \left((2e)^{\frac{1}{2}} \left(\frac{e}{n}\right) \left(\frac{2}{e^2}\right)^{\log_2^2 n}\right)^{n} < \frac{1}{2}
\end{aligned}$$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$\Pr[\text{fail } 1] = \Pr[\text{Some } S \in \binom{L}{\frac{n}{2}} \text{ has } < 2^{\log_2^2 n} - n \text{ neighbors in } R]$$

$$\le \Pr[\text{Some } S \in \binom{L}{\frac{n}{2}} \text{ has } \le 2^{\log_2^2 n} - n \text{ neighbors in } R]$$

$$\le \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{2^{\log_2^2 n} - n} \left(\frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}}\right)^{\frac{dn}{2}} \qquad \text{(By union bound)}$$

$$= \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{n} \left(1 - \frac{n}{2^{\log_2^2 n}}\right)^{\frac{dn}{2}}$$

$$\le \left(\frac{ne}{\frac{n}{2}}\right)^{\frac{n}{2}} \left(\frac{2^{\log_2^2 n} e}{n}\right)^{n} \left(e^{-\frac{n}{2^{\log_2^2 n}}}\right)^{\frac{dn}{2}}$$

$$= \left((2e)^{\frac{1}{2}} \left(\frac{e}{n}\right) \left(\frac{2}{e^2}\right)^{\log_2^2 n}\right)^{n} < \frac{1}{2}$$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$
\begin{aligned}
\Pr[\text{fail } 1] &= \Pr[\text{Some } S \in \tbinom{L}{\frac{n}{2}} \text{ has } < 2^{\log_2^2 n} - n \text{ neighbors in } R] \\
&\leq \Pr[\text{Some } S \in \tbinom{L}{\frac{n}{2}} \text{ has } \leq 2^{\log_2^2 n} - n \text{ neighbors in } R] \\
&\leq \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{2^{\log_2^2 n} - n} \left(\frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}}\right)^{\frac{dn}{2}} \qquad \text{(By union bound)} \\
&= \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{n} \left(1 - \frac{n}{2^{\log_2^2 n}}\right)^{\frac{dn}{2}} \\
&\leq \left(\frac{ne}{\frac{n}{2}}\right)^{\frac{n}{2}} \left(\frac{2^{\log_2^2 n} e}{n}\right)^{n} \left(e^{-\frac{n}{2^{\log_2^2 n}}}\right)^{\frac{dn}{2}} \\
&= \left((2e)^{\frac{1}{2}} \left(\frac{e}{n}\right) \left(\frac{2}{e^2}\right)^{\log_2^2 n}\right)^{n} < \frac{1}{2}
\end{aligned}
$$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$\begin{aligned}
\Pr[\text{fail } 1] &= \Pr\left[\text{Some } S \in \binom{L}{\frac{n}{2}} \text{ has } < 2^{\log_2^2 n} - n \text{ neighbors in } R\right] \\
&\le \Pr\left[\text{Some } S \in \binom{L}{\frac{n}{2}} \text{ has } \le 2^{\log_2^2 n} - n \text{ neighbors in } R\right] \\
&\le \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{2^{\log_2^2 n} - n} \left(\frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}}\right)^{\frac{dn}{2}} \qquad \text{(By union bound)} \\
&= \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{n} \left(1 - \frac{n}{2^{\log_2^2 n}}\right)^{\frac{dn}{2}} \\
&\le \left(\frac{ne}{\frac{n}{2}}\right)^{\frac{n}{2}} \left(\frac{2^{\log_2^2 n} e}{n}\right)^{n} \left(e^{-\frac{n}{2^{\log_2^2 n}}}\right)^{\frac{dn}{2}} \\
&= \left((2e)^{\frac{1}{2}} \left(\frac{e}{n}\right) \left(\frac{2}{e^2}\right)^{\log_2^2 n}\right)^{n} < \frac{1}{2}
\end{aligned}$$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$
\begin{aligned}
\Pr[\text{fail } 1] &= \Pr\left[\text{Some } S \in \binom{L}{\frac{n}{2}} \text{ has } < 2^{\log_2^2 n} - n \text{ neighbors in } R\right] \\
&\leq \Pr\left[\text{Some } S \in \binom{L}{\frac{n}{2}} \text{ has } \leq 2^{\log_2^2 n} - n \text{ neighbors in } R\right] \\
&\leq \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{2^{\log_2^2 n} - n} \left(\frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}}\right)^{\frac{dn}{2}} \qquad \text{(By union bound)} \\
&= \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{n} \left(1 - \frac{n}{2^{\log_2^2 n}}\right)^{\frac{dn}{2}} \\
&\leq \left(\frac{ne}{\frac{n}{2}}\right)^{\frac{n}{2}} \left(\frac{2^{\log_2^2 n} e}{n}\right)^{n} \left(e^{-\frac{n}{2^{\log_2^2 n}}}\right)^{\frac{dn}{2}} \\
&= \left((2e)^{\frac{1}{2}} \left(\frac{e}{n}\right) \left(\frac{2}{e^2}\right)^{\log_2^2 n}\right)^{n} < \frac{1}{2}
\end{aligned}
$$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$
\begin{aligned}
\Pr[\text{fail } 1] &= \Pr[\text{Some } S \in \tbinom{L}{\frac{n}{2}} \text{ has } < 2^{\log_2^2 n} - n \text{ neighbors in } R] \\
&\le \Pr[\text{Some } S \in \tbinom{L}{\frac{n}{2}} \text{ has } \le 2^{\log_2^2 n} - n \text{ neighbors in } R] \\
&\le \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{2^{\log_2^2 n} - n} \left(\frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}}\right)^{\frac{dn}{2}} \qquad \text{(By union bound)} \\
&= \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{n} \left(1 - \frac{n}{2^{\log_2^2 n}}\right)^{\frac{dn}{2}} \\
&\le \left(\frac{ne}{\frac{n}{2}}\right)^{\frac{n}{2}} \left(\frac{2^{\log_2^2 n} e}{n}\right)^{n} \left(e^{-\frac{n}{2^{\log_2^2 n}}}\right)^{\frac{dn}{2}} \\
&= \left((2e)^{\frac{1}{2}} \left(\frac{e}{n}\right) \left(\frac{2}{e^2}\right)^{\log_2^2 n}\right)^{n} < \frac{1}{2}
\end{aligned}
$$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$
\begin{aligned}
\Pr[\text{fail } 1] &= \Pr[\text{Some } S \in \tbinom{L}{\frac{n}{2}} \text{ has } < 2^{\log_2^2 n} - n \text{ neighbors in } R] \\
&\leq \Pr[\text{Some } S \in \tbinom{L}{\frac{n}{2}} \text{ has } \leq 2^{\log_2^2 n} - n \text{ neighbors in } R] \\
&\leq \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{2^{\log_2^2 n} - n} \left(\frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}} \right)^{\frac{dn}{2}} \qquad \text{(By union bound)} \\
&= \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{n} \left(1 - \frac{n}{2^{\log_2^2 n}} \right)^{\frac{dn}{2}} \\
&\leq \left(\frac{ne}{\frac{n}{2}} \right)^{\frac{n}{2}} \left(\frac{2^{\log_2^2 n} e}{n} \right)^{n} \left(e^{-\frac{n}{2^{\log_2^2 n}}} \right)^{\frac{dn}{2}} \\
&= \left((2e)^{\frac{1}{2}} \left(\frac{e}{n} \right) \left(\frac{2}{e^2} \right)^{\log_2^2 n} \right)^{n} < \frac{1}{2}
\end{aligned}
$$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$\begin{aligned}
\Pr[\text{fail } 1] &= \Pr[\text{Some } S \in \binom{L}{\frac{n}{2}} \text{ has } < 2^{\log_2^2 n} - n \text{ neighbors in } R] \\
&\leq \Pr[\text{Some } S \in \binom{L}{\frac{n}{2}} \text{ has } \leq 2^{\log_2^2 n} - n \text{ neighbors in } R] \\
&\leq \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{2^{\log_2^2 n} - n} \left(\frac{2^{\log_2^2 n} - n}{2^{\log_2^2 n}}\right)^{\frac{dn}{2}} \qquad \text{(By union bound)} \\
&= \binom{n}{\frac{n}{2}} \binom{2^{\log_2^2 n}}{n} \left(1 - \frac{n}{2^{\log_2^2 n}}\right)^{\frac{dn}{2}} \\
&\leq \left(\frac{ne}{\frac{n}{2}}\right)^{\frac{n}{2}} \left(\frac{2^{\log_2^2 n} e}{n}\right)^{n} \left(e^{-\frac{n}{2^{\log_2^2 n}}}\right)^{\frac{dn}{2}} \\
&= \left((2e)^{\frac{1}{2}} \left(\frac{e}{n}\right) \left(\frac{2}{e^2}\right)^{\log_2^2 n}\right)^{n} < \frac{1}{2}
\end{aligned}$$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$
\begin{aligned}
\Pr[\text{fail } 2] &= \Pr[\text{Some } v \in R \text{ has } X_v > 12\log_2^2 n \text{ neighbors in } L] \\
&\leq \sum_{v \in R} \Pr[X_v > 12\log_2^2 n] && \text{(Union bound)} \\
&= \sum_{v \in R} \Pr[X_v > (1+2)\mu] && \left(\mu = \tfrac{nd}{2^{\log_2^2 n}} = 4\log_2^2 n\right) \\
&< \tfrac{1}{2}
\end{aligned}
$$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$
\begin{aligned}
\Pr[\text{fail } 2] &= \Pr[\text{Some } v \in R \text{ has } X_v > 12\log_2^2 n \text{ neighbors in } L] \\
&\leq \sum_{v \in R} \Pr[X_v > 12\log_2^2 n] && \text{(Union bound)} \\
&= \sum_{v \in R} \Pr[X_v > (1+2)\mu] && \left(\mu = \tfrac{nd}{2^{\log_2^2 n}} = 4\log_2^2 n\right) \\
& < \tfrac{1}{2}
\end{aligned}
$$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$\begin{aligned}
\Pr[\text{fail } 2] &= \Pr[\text{Some } v \in R \text{ has } X_v > 12\log_2^2 n \text{ neighbors in } L] \\
&\leq \sum_{v \in R} \Pr[X_v > 12\log_2^2 n] && \text{(Union bound)} \\
&= \sum_{v \in R} \Pr[X_v > (1+2)\mu] && \left(\mu = \tfrac{nd}{2^{\log_2^2 n}} = 4\log_2^2 n\right)
\end{aligned}$$

$$< \frac{1}{2}$$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$\begin{aligned}
\Pr[\text{fail 2}] &= \Pr[\text{Some } v \in R \text{ has } X_v > 12\log_2^2 n \text{ neighbors in } L] \\
&\leq \sum_{v\in R} \Pr[X_v > 12\log_2^2 n] && \text{(Union bound)} \\
&= \sum_{v\in R} \Pr[X_v > (1+2)\mu] && \left(\mu = \frac{nd}{2^{\log_2^2 n}} = 4\log_2^2 n\right) \\
&< \sum_{v\in R} \left(\frac{e^2}{(1+2)^{(1+2)}}\right)^{\mu} && \text{(Chernoff bound)} \\
&= 2^{\log_2^2 n}\left(\frac{e^2}{3^3}\right)^{4\log_2^2 n} = \left(\frac{2e^8}{3^{12}}\right)^{\log_2^2 n} < \frac{1}{2}
\end{aligned}$$

We can use a Chernoff bound, since X_v can be seen as a sum of independent binary variables, one for each $u \in L$.

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$\begin{aligned}
\Pr[\text{fail } 2] &= \Pr[\text{Some } v \in R \text{ has } X_v > 12\log_2^2 n \text{ neighbors in } L] \\
&\leq \sum_{v \in R} \Pr[X_v > 12\log_2^2 n] && \text{(Union bound)} \\
&= \sum_{v \in R} \Pr[X_v > (1+2)\mu] && \left(\mu = \frac{nd}{2^{\log_2^2 n}} = 4\log_2^2 n\right) \\
&< \sum_{v \in R} \left(\frac{e^2}{(1+2)^{(1+2)}}\right)^{\mu} && \text{(Chernoff bound)} \\
&= 2^{\log_2^2 n}\left(\frac{e^2}{3^3}\right)^{4\log_2^2 n} = \left(\frac{2e^8}{3^{12}}\right)^{\log_2^2 n} < \frac{1}{2}
\end{aligned}$$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$\begin{aligned}
\Pr[\text{fail } 2] &= \Pr[\text{Some } v \in R \text{ has } X_v > 12\log_2^2 n \text{ neighbors in } L] \\
&\le \sum_{v \in R} \Pr[X_v > 12\log_2^2 n] && \text{(Union bound)} \\
&= \sum_{v \in R} \Pr[X_v > (1+2)\mu] && \left(\mu = \frac{nd}{2^{\log_2^2 n}} = 4\log_2^2 n\right) \\
&< \sum_{v \in R} \left(\frac{e^2}{(1+2)^{(1+2)}}\right)^{\mu} && \text{(Chernoff bound)} \\
&= 2^{\log_2^2 n}\left(\frac{e^2}{3^3}\right)^{4\log_2^2 n} = \left(\frac{2e^8}{3^{12}}\right)^{\log_2^2 n} < \frac{1}{2}
\end{aligned}$$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$\begin{aligned}
\Pr[\text{fail 2}] &= \Pr[\text{Some } v \in R \text{ has } X_v > 12\log_2^2 n \text{ neighbors in } L] \\
&\leq \sum_{v \in R} \Pr[X_v > 12\log_2^2 n] && \text{(Union bound)} \\
&= \sum_{v \in R} \Pr[X_v > (1+2)\mu] && \left(\mu = \tfrac{nd}{2^{\log_2^2 n}} = 4\log_2^2 n\right) \\
&< \sum_{v \in R} \left(\tfrac{e^2}{(1+2)^{(1+2)}}\right)^{\mu} && \text{(Chernoff bound)} \\
&= 2^{\log_2^2 n}\left(\tfrac{e^2}{3^3}\right)^{4\log_2^2 n} = \left(\tfrac{2e^8}{3^{12}}\right)^{\log_2^2 n} < \tfrac{1}{2}
\end{aligned}$$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4 \log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$\Pr[\text{fail}] \leq \Pr[\text{fail } 1] + \Pr[\text{fail } 2] \qquad \text{(By union bound)}$$

$$< \tfrac{1}{2} + \tfrac{1}{2} = 1$$

$$\Pr[\text{success}] = 1 - \Pr[\text{fail}] > 0$$

Since there is a nonzero probability that this randomly selected graph has the properties we want, the graph *exists*. □

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$\begin{aligned}\Pr[\text{fail}] &\leq \Pr[\text{fail } 1] + \Pr[\text{fail } 2] \qquad \text{(By union bound)} \\ &< \tfrac{1}{2} + \tfrac{1}{2} = 1\end{aligned}$$

$$\Pr[\text{success}] = 1 - \Pr[\text{fail}] > 0$$

Since there is a nonzero probability that this randomly selected graph has the properties we want, the graph *exists*. $\square$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$\begin{aligned}\Pr[\text{fail}] &\le \Pr[\text{fail 1}] + \Pr[\text{fail 2}] \qquad \text{(By union bound)}\\ &< \tfrac{1}{2} + \tfrac{1}{2} = 1\\ \Pr[\text{success}] &= 1 - \Pr[\text{fail}] > 0\end{aligned}$$

Since there is a nonzero probability that this randomly selected graph has the properties we want, the graph *exists*. $\square$

Probability Amplification

Proof of existence of $G(L, R, E)$.

Consider the random graph where each node in L chooses $d = \frac{4\log_2^2 n}{n} \cdot 2^{\log_2^2 n}$ neighbors in R independently and uniformly at random, with replacement.

$$\begin{aligned}
\Pr[\text{fail}] &\leq \Pr[\text{fail } 1] + \Pr[\text{fail } 2] \qquad \text{(By union bound)} \\
&< \tfrac{1}{2} + \tfrac{1}{2} = 1 \\
\Pr[\text{success}] &= 1 - \Pr[\text{fail}] > 0
\end{aligned}$$

Since there is a nonzero probability that this randomly selected graph has the properties we want, the graph *exists*. $\square$

Probability Amplification: Summary

We have shown that there *exists* an algorithm using only $\log_2^2 n$ random bits that constructs a sequence $S \subseteq [n]$ of at most $12\log_2^2 n$ samples, such that for any $x \in \mathcal{L}$,

$$\Pr\left[\forall r \in S : A(x,r) = 0\right] \leq \frac{1}{n^{(\log_2 n)-1}}$$

Unfortunately, we don't know of any (efficient) way of constructing the required graph. Also, while the algorithm works using the same graph for all $x \in \Sigma^\star$, it requires a new graph for each n, so it is *non-uniform*.

Probability Amplification: Summary

We have shown that there *exists* an algorithm using only $\log_2^2 n$ random bits that constructs a sequence $S \subseteq [n]$ of at most $12 \log_2^2 n$ samples, such that for any $x \in \mathcal{L}$,

$$\Pr\left[\forall r \in S : A(x, r) = 0\right] \leq \frac{1}{n^{(\log_2 n) - 1}}$$

Unfortunately, we don't know of any (efficient) way of constructing the required graph. Also, while the algorithm works using the same graph for all $x \in \Sigma^\star$, it requires a new graph for each n, so it is *non-uniform*.

Probability Amplification: Summary

We have shown that there *exists* an algorithm using only $\log_2^2 n$ random bits that constructs a sequence $S \subseteq [n]$ of at most $12 \log_2^2 n$ samples, such that for any $x \in \mathcal{L}$,

$$\Pr\left[\forall r \in S : A(x, r) = 0\right] \leq \frac{1}{n^{(\log_2 n) - 1}}$$

Unfortunately, we don't know of any (efficient) way of constructing the required graph. Also, while the algorithm works using the same graph for all $x \in \Sigma^\star$, it requires a new graph for each n, so it is *non-uniform*.

However, it turns out there *are* ways to construct (implicit) expander graphs, that can be used in a slightly different way to achieve probability amplification in practice.

Summary

- We have seen how the probabilistic method lets us determine the *existence* of certain objects. E.g. a large cut in a graph, and a truth assignment satisfying at least half the clauses in a MAX-SAT instance.
- We then detoured a bit and showed 3 algorithms for approximating MAX-SAT, called MAX-SAT-SIMPLE, MAX-SAT-RR, and MAX-SAT-COMBINED.
- Returning to the probabilistic method, we used it to prove the existence of an *expanding* graph called an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.
- Finally, we proved that an algorithm for *probability amplification* exists.
- Next time: More on randomized routing, "Lovasz Local Lemma", and a technique for derandomization.

Summary

- We have seen how the probabilistic method lets us determine the *existence* of certain objects. E.g. a large cut in a graph, and a truth assignment satisfying at least half the clauses in a MAX-SAT instance.
- We then detoured a bit and showed 3 algorithms for approximating MAX-SAT, called MAX-SAT-SIMPLE, MAX-SAT-RR, and MAX-SAT-COMBINED.
- Returning to the probabilistic method, we used it to prove the existence of an *expanding* graph called an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.
- Finally, we proved that an algorithm for *probability amplification* exists.
- Next time: More on randomized routing, "Lovasz Local Lemma", and a technique for derandomization.

Summary

- We have seen how the probabilistic method lets us determine the *existence* of certain objects. E.g. a large cut in a graph, and a truth assignment satisfying at least half the clauses in a MAX-SAT instance.
- We then detoured a bit and showed 3 algorithms for approximating MAX-SAT, called MAX-SAT-SIMPLE, MAX-SAT-RR, and MAX-SAT-COMBINED.
- Returning to the probabilistic method, we used it to prove the existence of an *expanding* graph called an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.
- Finally, we proved that an algorithm for *probability amplification* exists.
- Next time: More on randomized routing, "Lovasz Local Lemma", and a technique for derandomization.

Summary

- We have seen how the probabilistic method lets us determine the *existence* of certain objects. E.g. a large cut in a graph, and a truth assignment satisfying at least half the clauses in a MAX-SAT instance.
- We then detoured a bit and showed 3 algorithms for approximating MAX-SAT, called MAX-SAT-SIMPLE, MAX-SAT-RR, and MAX-SAT-COMBINED.
- Returning to the probabilistic method, we used it to prove the existence of an *expanding* graph called an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.
- Finally, we proved that an algorithm for *probability amplification* exists.
- Next time: More on randomized routing, "Lovasz Local Lemma", and a technique for derandomization.

Summary

- We have seen how the probabilistic method lets us determine the *existence* of certain objects. E.g. a large cut in a graph, and a truth assignment satisfying at least half the clauses in a MAX-SAT instance.
- We then detoured a bit and showed 3 algorithms for approximating MAX-SAT, called MAX-SAT-SIMPLE, MAX-SAT-RR, and MAX-SAT-COMBINED.
- Returning to the probabilistic method, we used it to prove the existence of an *expanding* graph called an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.
- Finally, we proved that an algorithm for *probability amplification* exists.
- Next time: More on randomized routing, "Lovasz Local Lemma", and a technique for derandomization.

Summary

- We have seen how the probabilistic method lets us determine the *existence* of certain objects. E.g. a large cut in a graph, and a truth assignment satisfying at least half the clauses in a MAX-SAT instance.
- We then detoured a bit and showed 3 algorithms for approximating MAX-SAT, called MAX-SAT-SIMPLE, MAX-SAT-RR, and MAX-SAT-COMBINED.
- Returning to the probabilistic method, we used it to prove the existence of an *expanding* graph called an $(n, 18, \frac{1}{3}, 2)$ OR-concentrator.
- Finally, we proved that an algorithm for *probability amplification* exists.
- Next time: More on randomized routing, "Lovasz Local Lemma", and a technique for derandomization.