CS5275 Lecture 3: The Probabilistic Method

Jonathan Scarlett

January 20, 2025

Acknowledgment. The first version of these notes was prepared by Eugene Lim and Ivona Martinović for a CS6235 assignment.

Useful References

- Chapter 6 of Mitzenmacher's and Upfal's book *Probability and Computing*
- Class 11 and 12 of Wooter's lecture *Randomized Algorithms*[1]
- Postle's lecture series *Probabilistic Methods*[2]
- Alon's and Spencer's book *The Probabilistic Method*
- Matoušek's and Vondrák's lecture notes *The Probabilistic Method*[3]
- Luke Postle's video lectures[4]

1 Introduction

The probabilistic method is a technique for proving the existence of certain objects (e.g., graphs, codes, algorithms) having certain properties, using probabilistic arguments (even when the object itself may have nothing to do with probability). The idea is to define a probability space and subsequently demonstrate that the probability that a randomly selected object has the desired properties is greater than zero. This, in turn, implies that the probability space must encompass such an object, thereby confirming the existence of such an object. More broadly, similar ideas of probabilistic reasoning can also be used to prove other kinds of results such as non-existence properties and bounds.

We will see several examples throughout the lecture, but mention a couple as motivation:

- A fundamental problem in Coding Theory (to be covered later) is to find a "large" collection of strings (say, binary of length n) that are "well-separated" (say, differing pairwise in at least δn positions for some $\delta \in (0,1)$). A concrete example would be: *Find a set $S \subseteq \{0,1\}^n$ of size $|S| = 2^{0.1n}$ such that any two of them differ in at least $0.1n$ positions.*
  - Note: If you are unfamiliar with coding theory, think of this as a company wanting to produce many serial numbers ($2^{0.1n}$ of them) under the requirement that no two of them can be too similar.

[1]`https://www.youtube.com/playlist?list=PLkvhuSoxwjI_JL7GYcJHK7-EK55tOKYGO`
[2]`https://www.youtube.com/playlist?list=PL2BdWtDKMS6nRF72s3TOGyBqXwMVHYiLU`
[3]`https://www.cs.cmu.edu/~15850/handouts/matousek-vondrak-prob-ln.pdf`
[4]`https://www.youtube.com/playlist?list=PL2BdWtDKMS6nRF72s3TOGyBqXwMVHYiLU`

A simple argument based on Hoeffding's inequality and the union bound shows that by generating $2^{0.1n}$ completely random strings, this "well-separated" property will hold with high probability when n is large enough. Thus, at least one such set S must exist.

- In areas such as Machine Learning, it is of interest to map "high-dimensional data" (say $x_1, \dots, x_N$ with $x_i \in \mathbb{R}^d$ for large d) to some lower-dimensional space (say $\mathbb{R}^m$ with $m \ll d$) while preserving some key structural properties of the original data. One important structural property is pairwise distances, and the resulting problem is: *Find a matrix $A \in \mathbb{R}^{m\times d}$ such that*

$$(1-\epsilon)\|x_i - x_j\| \le \|Ax_i - Ax_j\| \le (1+\epsilon)\|x_i - x_j\|, \quad \forall i,j.$$

  Finding such an A by hand is very tricky, but it turns out that by letting A have i.i.d. Gaussian entries, this property will hold with high probability under scaling of the form $m = O\big(\frac{1}{\epsilon^2}\log N\big)$. You can look up the *Johnson–Lindenstrauss Lemma* if you are interested to know more.

2 Basic Counting Arguments

The basic counting argument is an integral part of the probabilistic method. The idea of the counting method is as follows:

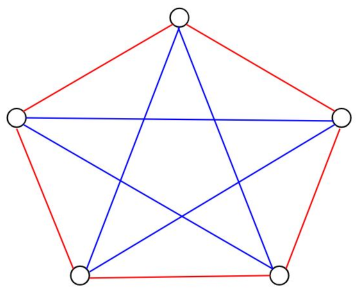
1. Set up a sampling scheme to sample an element (e.g., a random draw of S or A in the above examples) that has the potential to satisfy the desired property.
2. Design a set of "bad events" such that if none of the events hold, the sampled object x definitely has the desired property.
3. Count the number of such bad events. Let this number be m.
4. Compute the probability that each of these bad events occurs (or an upper bound on this probability). Let this probability be p.
5. Conclude that the object of interest must exist if $1 - mp > 0$. This is because, by union bound, we know the probability that none of the bad events holds is at least $1 - mp$.

2.1 Ramsey Number

Section 1.1 of *The Probabilistic Method*

The Ramsey number $R(k,l)$ is the smallest positive integer n such that, in every two-coloring (say blue and red) of the edges of a complete graph K_n of n vertices, we can always find either a red clique K_k of size k or a blue clique K_l of size l.

- Note: Another interpretation is to replace "red edge" by "edge is present" and "blue edge" by "'edge is absent", and to seek either a clique (i.e., fully connected subset) of size k or an independent set (i.e., subset with no connections) of size l. We'll stick with the red/blue terminology.
- It's fair to say that this is a fairly abstract concept without an immediately obvious application, but Ramsey numbers have had implications throughout theoretical computer science and beyond (e.g., see `https://www.cs.umd.edu/~gasarch/TOPICS/ramsey/ramsey.html`)


(a) K_5 with no monochromatic K_3.

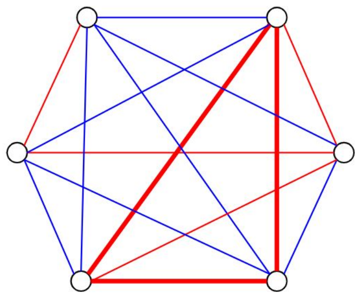

(b) K_6 with at least one monochromatic K_3.

Figure 1: Example showing $R(3,3) = 6$.

Let us consider a small example where $k = l = 3$. Figure 1a provides a coloring of K_5 with no monochromatic clique of size 3, implying that $R(3,3) > 5$. It turns out that $R(3,3) = 6$. To see this, note that each vertex v in K_6 is adjacent to five edges. By the pigeonhole principle, at least three such edges have the same color. Without loss of generality, we assume these edges are colored blue and are connected to vertices x, y, z. If any of the edges $(x,y), (y,z), (z,x)$ are colored blue, then we have a blue clique of size 3; otherwise, these edges are all colored red, thus forming a red clique of size 3. Figure 1b demonstrates an edge-coloring of K_6 with a red clique of size 3.

The Ramsey theorem shows that $R(k,l)$ is always finite, but calculating exact Ramsey numbers for larger values of k and l can be challenging and not many precise values of $R(k,l)$ are known. Instead, we can settle on knowing upper and lower bounds on $R(k,l)$ can be helpful. Here, we will focus on deriving lower bounds for the "diagonal" ($l = k$) Ramsey numbers $R(k,k)$.

Theorem 2.1 (Erdős, 1947)**.** *If $\binom{n}{k} \cdot 2^{1-\binom{k}{2}} < 1$ then $R(k,k) > n$. Thus for any $k \geq 3$, $R(k,k) > \lfloor 2^{k/2} \rfloor$.*

(Note: A similar upper bound dependent on $4^k = 2^{2k}$ is known, so while the bounds don't exactly match, they are both exponential and they mainly only differ in the exponent being $k/2$ vs. $2k$.)

Proof. Let us consider a random two-coloring of the edges of a complete graph K_n where every edge is colored independently either red or blue with probability 1/2. Consider an arbitrary ordering of all $\binom{n}{k}$ cliques $\{C_1, \ldots, C_{\binom{n}{k}}\}$ of size k. For each $i \in \{1, \ldots, \binom{n}{k}\}$, let A_i be the event that the clique C_i is monochromatic. For A_i to hold, the $\binom{k}{2}$ edges should be all blue or all red (holding with probability $2^{-\binom{k}{2}}$ each), and hence

$$\Pr[A_i] = 2^{1-\binom{k}{2}}.$$

As there are $\binom{n}{k}$ cliques, we can apply the union bound to calculate the probability that at least 1 such event occurs:

$$\Pr\left[\bigcup_{i=1}^{\binom{n}{k}} A_i\right] \leq \sum_{i}^{\binom{n}{k}} \Pr[A_i] = \binom{n}{k} \cdot 2^{1-\binom{k}{2}} < 1,$$

where the last inequality holds by assumption in the theorem. Therefore, the probability that no such event A_i occurs is $1 - \binom{n}{k} 2^{1-\binom{k}{2}} > 0$; or in other words, there is a positive probability of sampling a two-coloring of K_n without a monochromatic K_k. Thus, $R(k,k) > n$.

If $k \geq 3$ and we take $n = \lfloor 2^{k/2} \rfloor$ then:

$$\binom{n}{k} \cdot 2^{1-\binom{k}{2}} \overset{(i)}{<} \frac{n^k}{k!} \cdot \frac{2^{1+k/2}}{2^{k^2/2}} \overset{(ii)}{\leq} \frac{2^{k^2/2}}{k!} \cdot \frac{2^{1+k/2}}{2^{k^2/2}} = \frac{2^{1+k/2}}{k!} \overset{(iii)}{<} \frac{2^{1+k/2}}{2^k} = 2^{1-k/2} < 1, \tag{1}$$

where step (i) applies $\binom{n}{k<\frac{n^k}{k!}}$ and $\binom{k}{2} = \frac{k(k-1)}{2} = \frac{k^2-k}{2}$, step (ii) applies $n = \lfloor 2^{k/2} \rfloor \leq 2^{k/2}$, and step (iii) applies $k! \geq 2^k$. From (1), we deduce that $R(k,k) > \lfloor 2^{k/2} \rfloor$ as desired. $\square$

Converting into a randomized algorithm. The proof of existence hints at a strategy to sample a two-coloring of a complete graph with n vertices such that there is no monochromatic clique of size k. More generally, many proofs by the probabilistic method can be converted to an algorithm that uses random sampling to try to find an element with the desired property. Such algorithms can be categorized into *Monte Carlo algorithms* or *Las Vegas algorithms*, as we describe below.

Monte Carlo algorithm. A Monte Carlo algorithm is a randomized algorithm that may have some probability of failure, while often having a fixed (pre-specified) number of "rounds" of random sampling. In the simplest case, we simply run a single round of sampling and show that it succeeds with high probability. More generally, we might run the same random procedure multiple times and either (i) figure out which one is the "best" one, or (ii) aggregate all of the results. (For instance, if the randomized algorithm only "succeeds" with probability 0.01, we could run it $\gg 100$ times to be confident of at least one success.)

In the proof, we've already established the sampling algorithm as a means of coloring the graph by independently assigning each edge as either red or blue, selected at random. If the probability of obtaining a sample with the desired property is represented as p. If p is already very small (say $p = o(1)$), then simply running the random procedure once will succeed wit high probability.

For example, if we want to color K_{1000} in a way that there are no monochromatic K_{20}, we can first check if this is possible using Theorem 2.1. As $n = 1000 \leq 2^{10} = 2^{k/2}$, this means that there exists a coloring with no monochromatic K_{20}. If we simplify the first expression from the theorem, we can get:

$$\binom{n}{k} 2^{1-\binom{k}{2}} \leq \frac{n^k}{k!} 2^{1-(k(k-1)/2)} \leq \frac{2^{k/2+1}}{k!} < 1,$$

where the three inequalities follow from (i) $\binom{n}{k} \leq \frac{n^k}{k!}$ and $\binom{k}{2} = \frac{k(k-1)}{2}$; (ii) the inequality $n \leq 2^{k/2}$ established above; and (iii) $k! > 2^k > 2^{k/2+1}$ for any $k > 2$. Using this simplification, we know that the probability that a random coloring has a monochromatic K_{20} is maximally $\frac{2^{20/2+1}}{20!} < 8.5 \cdot 10^{-16}$. Thus, the suggested Monte Carlo algorithm has only a very small probability of giving an incorrect solution.

Some important caveats to this sort of argument are as follows:

- In situations where the probability of bad events is high, Monte Carlo methods may have a higher probability of providing incorrect solutions and performing poorly.
- Perhaps more importantly, *checking the desired property cannot always be done efficiently.* In fact, the Ramsey number example fits in this category, as checking with a clique of a certain size exists in a graph is a famously (NP-)hard problem. Another example would be the first motivating example in Section 1 – with $|S| = 2^{0.1n}$ strings, the number of pairwise checks would be $O\big((2^{0.1n})^2\big) = O\big(2^{0.2n}\big)$, which is exponentially large. In fact, even just storing the list of strings would be a problem (unless n is small), as there are $2^{0.1n}$ of them.

Las Vegas algorithm. Unlike a Monte Carlo algorithm, a Las Vegas is *guaranteed* to output the correct the solution; the idea is to repeatedly perform the random sampling and continue until it is verified to succeed. Thus, the guaranteed correctness may come at the expense of (having some small probability of) taking a long time.

To derive an upper limit on the expected running time of this Las Vegas algorithm, we simply multiply the time needed to generate and verify each sample by the expected number of samples, which is $1/p$. In the Ramsey number example, when the value k is constant, there exists a polynomial time verification algorithm: Just search over all $\binom{n}{k}$ cliques to ensure that they are not monochromatic. However, it is important to note that if k grows with n, this verification algorithm ceases to execute within polynomial time.

Weaknesses. While the basic counting method tend to be relatively straightforward, its efficacy depends on the design of sampling schemes and the accurate identification of bad events. The use of the union bound may also result in overly loose inequalities, especially in cases with a large number of bad events m, where condition $1 - mp > 0$ becomes unlikely to hold. We shall see some ways to get around this problem in Section 6.

3 Expectation Arguments

The expectation argument relies on the fact that a discrete random variable must, with positive probability, take on values both lower and higher than its expected value. For instance, if the average score of the class is 42 points, then at least one student scored is 42 or higher, and at least one student scored 42 or lower. This observation forms the basis of the expectation argument in probabilistic reasoning. Formally, we have the following lemma.

Lemma 3.1. *For any real-valued random variable X, we have* $\Pr[X \geq \mathbb{E}[X]] > 0$ *and* $\Pr[X \leq \mathbb{E}[X]] > 0$.

Proof. We present the proof for the case that X is a discrete random variable, but the argument for continuous variables is similar with sums replaced by integrals.

We apply a proof by contradiction: If we were to have $\Pr[X \geq \mathbb{E}[X]] = 0$, it would follow that

$$\mathbb{E}[X] = \sum_{x} x \Pr[X = x] = \sum_{x < \mathbb{E}[X]} x \Pr[X = x] < \sum_{x < \mathbb{E}[X]} \mathbb{E}[X] \Pr[X = x] = \mathbb{E}[X],$$

which is a contradiction. The other claim is proven similarly. □

3.1 Finding a Large Cut

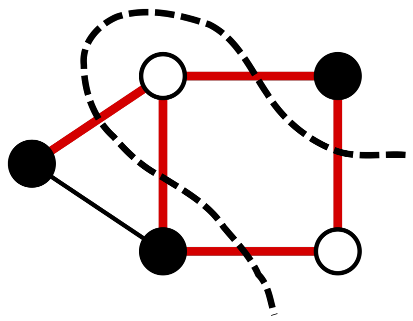
Section 6.2.1 of *Probability and Computing*

A cut (S_1, S_2) for an undirected graph $G = (V, E)$ is a partition of V into two disjoint sets S_1 and S_2. The value of the cut $C(S_1, S_2)$ is the number of edges that cross from S_1 to S_2. The following figure[5] shows an example of a large cut where S_1 is the set of black vertices and S_2 is the set of white vertices. The value of the cut $C(S_1, S_2) = 5$, which is also the maximum cut for the graph. More generally, finding the cut with the largest number of edges is known as the Maximum Cut (MAXCUT) problem.

- Note: Applications of MAXCUT include network design, statistical physics, and clustering problems.

[5]Source: Wikipedia page on maximum cut.

- It is an NP-hard problem, but a 0.878-approximation algorithm is known. We'll be less ambitious here and show that a very simple probabilistic argument gives a 0.5-approximation.



Theorem 3.2. *Given an undirected graph $G = (V, E)$ with $|E| = m$, there exist a cut (S_1, S_2) such that the value of the cut $C(S_1, S_2) \geq m/2$.*

Proof. For each vertex $v \in V$, assign v uniformly to either S_1 or S_2. Let $e_1, \ldots, e_m$ be an arbitrary arrangement of E and define $X_i = \mathbf{1}[e_i \text{ connects } S_1 \text{ to } S_2]$ for each $i \in [m]$ where $[m] = \{1, \ldots, m\}$. Since the probability that e_i crosses S_1 to S_2 is 0.5, we have $\mathbb{E}[X_i] = 0.5$. Then

$$\mathbb{E}[C(S_1, S_2)] = \mathbb{E}\left[\sum_{i=1}^{m} X_i\right] = \sum_{i=1}^{m} \mathbb{E}[X_i] = \frac{m}{2}.$$

Since $\mathbb{E}[C(S_1, S_2)] = m/2$, by Lemma 3.1, there exist a cut (S_1, S_2) such that $C(S_1, S_2) \geq m/2$. □

Tightness of the lower bound. It turns out that $m/2$ is an essentially tight lower bound for the size of the maximum cut. To see this, consider the complete graph K_{2n} with $2n$ vertices where $n \in \mathbb{N}$, and let (S_1, S_2) be a cut with $|S_1| = k$ and $|S_2| = 2n - k$. Note that the value of this cut is $k(2n - k)$, which is maximized when $k = n$. As such, the value of any maximum cut in K_{2n} is n^2. Since $m = \binom{2n}{2} = n(2n - 1) = 2n^2 - n$, some simple algebraic manipulation reveals that the value of the maximum cut is $m/(2 - 1/n)$, which approaches $m/2$ as $n \to \infty$.

Converting to a randomized algorithm. As before, we can convert this argument into a Las Vegas algorithm that finds a cut (S_1, S_2) with $C(S_1, S_2) \geq m/2$ in expected polynomial time: we just have to sample many random cuts $(S_1^{(1)}, S_2^{(1)}), (S_1^{(2)}, S_2^{(2)}), \ldots$ until we obtain an $(S_1^{(T)}, S_2^{(T)})$ with $C(S_1^{(T)}, S_2^{(T)}) \geq m/2$. Since checking if a given $S_1^{(j)}, S_2^{(j)}$ satisfy $C(S_1^{(j)}, S_2^{(j)}) \geq m/2$ takes polynomial time, we are left to show that $\mathbb{E}[T]$ is also polynomial in m. Let $p = \Pr[C(S_1, S_2) \geq m/2]$. Observe that

$$\frac{m}{2} = \mathbb{E}[C(S_1, S_2)] = \sum_{i < m/2} i \Pr[C(S_1, S_2) = i] + \sum_{i \geq m/2} i \Pr[C(S_1, S_2) = i] \leq (1 - p)\left(\frac{m}{2} - 1\right) + pm,$$

where the inequality follows since $i \leq \frac{m}{2} - 1$ in the first sum and $i \leq m$ in the second sum. By re-arranging the above inequality $\frac{m}{2} \leq (1 - p)\left(\frac{m}{2} - 1\right) + pm$ and solving for p, we obtain $p \geq \frac{1}{m/2+1}$. If we imagine each sampling of the cut as a (biased) coin flip, then we are interested in the following question: "Given a coin that flips a head with probability p, how many flips T do we need in expectation before we get a head?"

Since T follows the geometric distribution with parameter p, we have $\mathbb{E}[T] = 1/p \leq m/2 + 1$, which is linear in m. Note also that this is only an upper bound, and the actual value of $\mathbb{E}[T]$ may be much smaller.

Converting to a deterministic algorithms. This turns out to be a well-known example where the randomized algorithm can be *derandomized*, i.e., a deterministic strategy gives the same guarantee. See Appendix A for the details. Deterministic algorithms are often considered preferable, e.g., so that the practical performance can be more predictable or reliable. However, in general performing derandomization while maintaining computational efficiency can be difficult or impossible.

3.2 Maximum Satisfiability

Section 6.2.2 of *Probability and Computing*

Here we generalize the example given in the introduction section of this lecture.

In a logical formula, a literal is either a variable x or its complement $\bar{x}$, and a clause is a disjunction ($\vee$) of literals. A logical formula is in conjunctive normal form (CNF) if it is the conjunction ($\wedge$) of a set of clauses. For brevity, we shall call a formula that is in CNF form a *CNF formula*. For example, the CNF formula

$$\psi_0(x_1, x_2, x_3, x_4) = (x_1 \vee x_2 \vee \bar{x}_3 \vee \bar{x}_4) \wedge (\bar{x}_1 \vee x_3) \wedge (\bar{x}_1 \vee x_2 \vee x_4)$$

has four variables and three clauses.

Let ψ be a CNF formula with n variables. A truth assignment to ψ is an assignment of its variables $x_1, \ldots, x_n$ to values $\{\texttt{TRUE}, \texttt{FALSE}\}$ such that $\psi(x_1, \ldots, x_n) = \texttt{TRUE}$. For example, a truth assignment to ψ_0 as defined above is $x_1 = x_3 = \texttt{FALSE}$ and $x_2 = x_4 = \texttt{TRUE}$. The problem of finding a truth assignment to ψ is known as the Constraint Satisfaction (SAT) problem. Furthermore, if all clauses have exactly k literals, then the problem is known as k-SAT. We will discuss more about k-SAT in a later section.

For now, let us consider a variation of the problem where we want to find an assignment to the variables that satisfy as many clauses as possible. This is known as the Maximum Satisfiability (MAXSAT) problem. (Both SAT and MAXSAT are famously NP-hard problems, and SAT is the basis for countless reductions showing other problems to be NP-hard.)

Theorem 3.3. *Let ψ be a CNF formula with m clauses, and let k_i be the number of literals in the i-th clause. Let $k = \min_{i \in [m]} k_i$. Then there is an assignment whose number of satisfied clauses is at least*

$$\sum_{i=1}^{m} (1 - 2^{-k_i}) \geq m(1 - 2^{-k}).$$

Proof. Assign each variable independently and uniformly to either TRUE or FALSE. Let Y_i be the indicator random variable for which the i-th clause is satisfied. We have $\mathbb{E}[Y_i] = 1 - 2^{-k_i}$, and thus

$$\mathbb{E}\left[\sum_{i=1}^{m} Y_i\right] = \sum_{i=1}^{m} \mathbb{E}[Y_i] = \sum_{i=1}^{m} (1 - 2^{-k_i}) \geq m(1 - 2^{-k}).$$

By Lemma 3.1, there must exist an assignment with at least that many clauses satisfied. □

Converting to a randomized algorithm. Similarly to the MAXCUT example, we can convert such an argument into a Las Vegas algorithm that finds an assignment with at least $m(1 - 2^{-k})$ satisfied clauses.

The conversion scheme and proof are similar to before. The reader is encouraged to verify that we need to sample, on average, at most $m \cdot 2^{-k} + 1$ assignments to obtain the desired assignment.

Weaknesses. The expectation method relies on the expectation of the quantity of interest to be sufficiently large (or sufficiently small, depending on the specifics of the problem). However, in some cases, extreme outliers in the constructed probability space might skew this expectation to be way smaller (or larger) than what is required. In such cases, one can sometimes condition on suitably-chosen events that hold with high probability but 'remove" the outlier cases. The use of second moments in addition to the mean is also often useful; see Section 4.

4 Second Moment Methods

Just like expectation, the variance (a.k.a., second centered moment) is another useful statistic for a random variable X. In the context of the probabilistic method, *second moment methods* refer to methods that use both mean and variance information to prove existence properties. Perhaps the most common tool for this purpose of Chebyshev's inequality, which is stated as follows.

Lemma 4.1. *(Chebyshev's inequality). Let X be a random variable with a finite variance. Then for $t > 0$*

$$\Pr[|X - \mathbb{E}[X]| \geq t] \leq \frac{\mathrm{Var}[X]}{t^2}.$$

Proof. Observe that

$$\Pr[|X - \mathbb{E}[X]| \geq t] = \Pr[(X - \mathbb{E}[X])^2 \geq t^2] \leq \frac{\mathbb{E}[(X - \mathbb{E}[X])^2]}{t^2} = \frac{\mathrm{Var}[X]}{t^2},$$

where the inequality holds due to Markov's inequality. □

Notes: Computing or upper bounding $\mathrm{Var}[X]$ is not always straightforward; it is worth noting the following:

- If $X = \sum_{i=1}^n U_i$ and the U_i are independent, we simply have $\mathrm{Var}[X] = \sum_{i=1}^n \mathrm{Var}[U_i]$ (an easy case).
- If $X = \sum_{i=1}^n U_i$ but the U_i are not necessarily independent, then this generalizes to: (try as an exercise)

  $$\mathrm{Var}[X] = \sum_{i=1}^n \mathrm{Var}[U_i] + \sum_{(i,j)\,:\,i \neq j} \mathrm{Cov}[U_i, U_j],$$

  and bounding the second tern may be more difficult (but often possible).
- Alternatively, if $X = \sum_{i=1}^n U_i$ then we can decompose $\mathrm{Var}[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$ and use $\mathbb{E}[X^2] = \sum_{i=1}^n \sum_{j=1}^n \mathbb{E}[U_i U_j]$ (try as an exercise); see the tutorial for an example ('Counting Triangles').
- There are also more advanced tools for bounding variance (e.g., Efron-Stein inequality), but they are beyond our scope.

4.1 Bounding the Middle Binomial Coefficient

Section 5.2 of *Matoušek's and Vondrák's Lecture Notes*

The binomial coefficient $\binom{2m}{m}$ is the largest among the binomial coefficients $\binom{2m}{k}$ for $k = 0, 1, \dots, 2m$. It frequently appears in various mathematical formulas, so having access to simple bounds can be very useful.

A very simple upper bound is $\binom{2m}{m} \leq 2^{2m}$, which follows by interpreting $\binom{2m}{m}$ as the number of length-$2m$ sequences containing exactly m ones and m zeros (and the upper bound 2^{2m} is the *total* number of binary sequences). Using the second moment method, we can show that this simple upper bound is tight to within a $\Theta(\sqrt{m})$ factor.

Theorem 4.2. *For all $m \geq 1$, we have $\binom{2m}{m} \geq 2^{2m}/(4\sqrt{m}+2)$.*

Proof. Let us define a random variable $X = X_1 + X_2 + \cdots + X_{2m}$, where we independently sample each X_i as 0 or 1 with probability 1/2. That is, $X \sim \text{Binomial}(2m, 1/2)$We know that the expected value of each X_i is 1/2. Using the linearity of expectation, we can calculate the expected value of X as $\mathbb{E}[X] = m$. Similarly, we can calculate variance for each variable $\text{Var}[X_i] = \mathbb{E}[X_i^2] - \mathbb{E}[X_i]^2 = \frac{1}{2} - (\frac{1}{2})^2 = \frac{1}{4}$. Since $X_1, \dots, X_{2m}$ are independent, the variance of X is $\text{Var}[X] = (1/4) \cdot 2m = m/2$. Now, if we apply Chebyshev's inequality with $t = \sqrt{m}$, we get

$$\Pr[|X - m| < \sqrt{m}] \geq \frac{1}{2}.$$

We rewrite the probability as

$$\Pr[|X - m| < \sqrt{m}] = \sum_{x:|x-m|<\sqrt{m}} \Pr[X = x] = \sum_{k:|k|<\sqrt{m}} \Pr[X = m+k]$$

where in the last step, we replace $x - m$ by k. Since $X \sim \text{Binomial}(2m, 1/2)$, we know that the probability of X being equal to a specific value $m+k$ is $\binom{2m}{m+k} \cdot 2^{-2m} \leq \binom{2m}{m} \cdot 2^{-2m}$ (because $\binom{2m}{m}$ is the largest binomial coefficient). Therefore,

$$\frac{1}{2} \leq \sum_{k:|k|<\sqrt{m}} \Pr[X = m+k] \leq (2\sqrt{m}+1)\binom{2m}{m}2^{-2m},$$

where the $2\sqrt{m}+1$ term comes from upper bounding how many integers k satisfy $|k| < \sqrt{m}$. Simple re-arranging gives the desired bound, $\binom{2m}{m} \geq 2^{2m}/(4\sqrt{m}+2)$. □

4.2 Distinct Sum Problem

Section 2.1 of `https://cse.buffalo.edu/~hungngo/classes/2011/Spring-694/lectures/sm.pdf`

Consider the following problem: *Given an integer n, what's the largest-cardinality subset $S \subseteq \{1, \dots, n\}$ we can find such that every $S' \subseteq S$ has a different value of $\sum_{i \in S'} i$?* Let $f(n)$ denote this largest possible cardinality, and we say that the set S satisfies the *distinct sum property.*

It is easy to see that

$$f(n) \geq \lfloor \log_2 n \rfloor + 1$$

because we can choose powers of two $S = \{1, 2, 4, 8, \dots\}$. Any distinct combination of powers two will give a distinct value; this follows readily from the fact that the integers can be converted to binary in a one-to-one manner, and we can interpret $S' \subseteq S$ as indicating which bits are set to 1.

Using the probabilistic method, a useful upper bound on $f(n)$ can be proved. Note that unlike previous sections where we gave existence statements, this result is a *non-existence* statement, stating that no suitable sets S above a certain size can exist.

Theorem 4.3. *It holds that $f(n) \leq \log_2 n + \frac{1}{2}\log_2 \log_2 n + O(1)$.*

Proof. For intuition, we first prove a weaker result that has $\log_2 \log_2 n$ in place of $\frac{1}{2} \log_2 \log_2 n$. Let S be a set satisfying the distinct sum property, and let k be its cardinality. Since $S \subseteq \{1, \dots, n\}$, the sums can only take values below nk. But by definition there must be 2^k distinct sums, so it must hold that $2^k \le nk$. Performing some simple asymptotic manipulations on this inequality gives $k \le \log_2 n + \log \log n + O(1)$ (see the above link if you want details on this step).

We refine this argument using Chebyshev's inequality. This time we explicitly write $S = \{a_1, \dots, a_k\}$. We let S' be a *random* subset in which each element of S is included with probability $\frac{1}{2}$, and define the sum $X = \sum_{i \in S'} i$. Observe that for any integer i, there are only two possibilities:

- If i cannot be produced by summing any subset in S, then $\Pr[X = i] = 0$.
- If i can be produced by summing some subset in S, then $\Pr[X = i] = \frac{1}{2^k}$. This is because the 2^k subsets are equally likely and each give a distinct sum.

Letting $\mu = \mathbb{E}[X]$ and $\sigma^2 = \mathrm{Var}[X]$, Chebyshev's inequality gives

$$\Pr\big[|X - \mu| \le 2\sigma\big] \ge 1 - \frac{1}{4} = \frac{3}{4}.$$

On the other hand, using the above property $\Pr[X = i] \in \big\{0, \frac{1}{2^k}\big\}$ and the fact that there are at most $4\sigma + 1$ integers satisfying $|X - \mu| \le 2\sigma$, we have

$$\Pr\big[|X - \mu| \le 2\sigma\big] \le \frac{4\sigma + 1}{2^k}.$$

Combining the above inequalities gives $\frac{4\sigma+1}{2^k} \ge \frac{3}{4}$.

Furthermore, since X is an independent sum of variables of the form $a_i \times \mathrm{Bernoulli}(1/2)$ (for $i = 1, \dots, k$), and since the variance of Bernoulli(1/2) is 1/4, we get

$$\sigma^2 = \frac{\sum_{i=1}^k a_i^2}{4} \le \frac{n^2 k}{4},$$

which implies $\sigma \le n\sqrt{k}/2$. (Note that the last step above follows since $a_i \le n$.)

Combining the conclusions of the above two paragraphs gives $\frac{2n\sqrt{k}+1}{2^k} \ge \frac{3}{4}$, or equivalently

$$n \ge \frac{(3/4)2^k - 1}{2\sqrt{k}}.$$

From here, some asymptotic manipulations similar to those of the simpler argument (first paragraph of this proof) give $k \le \log_2 n + \frac{1}{2} \log \log n + O(1)$, as desired. □

4.3 (**Optional**) Other Applications and Inequalities

To highlight how diverse applications of the probabilistic method can be, we briefly mention an application to number theory. Informally, the result proved is that *except for a vanishingly small number of positive integers n, it holds that the number of unique prime factors is close to* $\log \log n$. See `https://www.youtube.com/watch?v=5pV_35vjVmU` for a precise statement and proof.

The second moment is also often used to derive threshold properties of random graphs; specifically, if each edge is independently included in the graph with probability p, then the graph satisfies a property of interest with probability approaching 1 for p above a suitable threshold, but with probability approach 0 for p below a similar threshold. See the tutorial for an example on the existence or non-existence of triangles depending on whether $p \gg \frac{1}{n}$ or $p \ll \frac{1}{n}$.

We also briefly mention that Chebyshev's inequality is not the only second moment based tool. To name one other example, the Paley-Zygmund Inequality (Notes 6 of *Measure-Theoretic Probability*) states that for any random variable $X \geq 0$ and constant $0 \leq \theta \leq 1$, it holds that

$$\Pr[X > \theta \mathbb{E}[X]] \geq (1-\theta)^2 \frac{\mathbb{E}[X]^2}{\mathbb{E}[X^2]}.$$

This is related to variance since $\mathrm{Var}[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$, so $\frac{\mathbb{E}[X]^2}{\mathbb{E}[X^2]}$ being high (e.g., close to 1) indicates lower variance. Specializing to $\theta = 0$ gives $\Pr[X > 0] \geq \frac{\mathbb{E}[X]^2}{\mathbb{E}[X^2]}$, which is particularly useful when we only need to show that a certain object exists (i.e., $X > 0$, where X counts occurrences of that object).

5 Sample and Modify Approaches

Sometimes building a random structure with the desired properties directly can be complicated. In such cases, a simpler solution may be to construct a random structure that lacks the desired properties and subsequently modifying it so that it has those properties. An illustration of this approach, known as the sample-and-modify technique, is given below.

5.1 Independent Sets

Section 6.4.1 of *Probability and Computing*

As a reminder, a subset S of vertices in a graph G is called an independent set if, for any pair of vertices in S, there does not exist an edge between them. Finding independent sets in graphs is of both theoretical interest, with applications including network design, data clustering, and resource allocation (e.g., an edge may indicate that two objects are "conflicting", and we want to find non-conflicting subsets).

Identifying the largest independent set within a graph is a computationally challenging problem. However, the following theorem illustrates how the probabilistic method can offer valuable bounds and insights into the size of the largest independent set in a graph.

Theorem 5.1. *Let $G = (V, E)$ be a connected graph on n vertices with $m \geq n/2$ edges. Then G has an independent set with at least $n^2/4m$ vertices.*

Proof. Let us define the average degree of the vertices as $d = 2m/n \geq 1$. We can define the following sample-and-modify algorithm:

1. **Sample.** Select a subset S of vertices from G that we "tentatively" intend to keep; specifically, each of the n vertices is independently placed in S with probability $1/d$. Keep only the edges that connect these selected vertices.

2. **Modify.** For each remaining edge (between two nodes in S), delete it, and also delete one of its neighboring vertices. The final independent set is the set of non-deleted vertices in S.

As we have deleted all of the edges, the remaining vertices form an independent set, which we constructed by first sampling the vertices and then modifying the remaining graph.

The analysis is as follows:

- Let X be the number of selected vertices in the first step. Given that the graph comprises n vertices, and considering that each vertex is selected with a probability $1/d$, it follows that $\mathbb{E}[X] = \frac{n}{d}$.
- Let Y be the number of edges that remain after the first step. Since there are $m = nd/2$ edges in the graph, and an edge remains if and only if its two adjacent vertices are selected, it follows that $\mathbb{E}[Y] = \frac{nd}{2}(\frac{1}{d})^2 = \frac{n}{2d}$.
- In the second step, we delete all of the edges and in the worst case, for each remaining edge we delete a different vertex, for a maximum total of Y vertices. At the end of the algorithm, the size of an independent set that we construct is at least $X - Y$, and thus $\mathbb{E}[X - Y] = \frac{n}{d} - \frac{n}{2d} = \frac{n}{2d}$. By the expectation argument, there exists a scenario in which $X - Y \geq \frac{n}{2d}$.

We have thus constructed a random independent set with at least $n/2d$ vertices on average, so by the expectation argument, there must exist an independent set of size at least $n/2d = n^2/4m$. □

6 Lovász Local Lemma

This section builds on the counting argument described in Section 2, which was based on the following general strategy to argue that entities with some property $\mathcal{P}$ exist:

1. Design a sampling scheme $\mathcal{S}$ and let $s \sim \mathcal{S}$ be an arbitrary sample.
2. Design a set of bad events $\mathcal{B}_1, \ldots, \mathcal{B}_m$ for which if none of these bad events holds, then s satisfy $\mathcal{P}$. Below we will let $\bar{\mathcal{B}}_i$ denote the complement of $\mathcal{B}_i$.
3. Show that there is a positive probability that none of the bad events occur.

In the third step, we need to find a lower bound for the probability that none of the bad events occur, which is equivalent to upper bounding the probability that least one of them occurs. In particular, if there is some $p \in [0, 1]$ such that $\Pr[\mathcal{B}_i] \leq p$ for all $i \in \{1, \ldots, m\}$, then by the union bound, we have $\Pr[\cap_{i=1}^m \bar{\mathcal{B}}_i] \geq 1 - mp$. If $mp \leq 1$, then we are done – this is essentially the approach used to prove Theorem 2.1.

The Lovász local lemma is a useful result to facilitate the final step of the argument if it is not possible to achieve $mp \leq 1$. To motivate the lemma, suppose (unrealistically but only momentarily) that the bad events are mutually independent. Then, it is easy to show that there is a positive probability that none of the bad events occur: We simply have $\Pr[\cap_{i=1}^m \bar{\mathcal{B}}_i] \geq (1 - p)^m$, which is positive if $p \neq 1$. Unfortunately, this argument is almost never suitable, as the bad events are almost always dependent.

Intuitively, the Lovász local lemma states that if $\mathcal{B}_1, \ldots, \mathcal{B}_m$ are "not too dependent", then we can treat them as "roughly independent". The vague notions of "not too dependent" and "roughly independent" are made formal in the following definition and theorem.

Definition 6.1. Let $S = \{\mathcal{B}_1, \ldots, \mathcal{B}_m\}$. For each $i \in [m]$, we say that an event $\mathcal{B}_i$ is mutually independent to all but at most d events if there exist a set $S_{\text{dep}} \subset S \backslash \{\mathcal{B}_i\}$ (possibly different for each i) of size at most d such that for any subset $S_{\text{ind}} \subset S \backslash (S_{\text{dep}} \cup \{\mathcal{B}_i\})$, the probability of event $\mathcal{B}_i$ is unchanged upon conditioning on the (complements of the) events indexed by S_{ind}:

$$\Pr\left[\mathcal{B}_i \middle| \bigcap\nolimits_{\mathcal{B} \in S_{\text{ind}}} \bar{\mathcal{B}}_i\right] = \Pr[\mathcal{B}_i].$$

(Recall that $\bar{\mathcal{B}}_i$ denotes the complement event of $\mathcal{B}_i$.)

Theorem 6.2 (Lovász local lemma)**.** *Let $\mathcal{B}_1, \dots, \mathcal{B}_m$ be a set of bad events such that for each $i \in [m]$, we have $\Pr[\mathcal{B}_i] \le p$ and $\mathcal{B}_i$ is mutually independent to all but at most d events. If $4pd \le 1$, then*

$$\Pr\left[\bigcap_{i=1}^{m} \bar{\mathcal{B}}_i\right] \ge (1-2p)^m > 0.$$

Intuitively, the condition $4pd \le 1$ states that the more dependence is permitted (higher d), the rarer the events have to be (smaller p) in order to maintain the "independence-like" property. To see the need for this kind of a condition, suppose that we were to have $pd = 1$. Then, a possible scenario would be that in which there are d events with probability $\frac{1}{d}$ each that are *disjoint* from each other (an extreme form of dependence), meaning exactly one of them always holds. Then we would have $\Pr\left[\bigcup_{i=1}^{m} \mathcal{B}_i\right] = 1$ or equivalently $\Pr\left[\bigcap_{i=1}^{m} \bar{\mathcal{B}}_i\right] = 0$, which contradicts our goal of showing that this probability is positive.

The (optional) proof of Theorem 6.2 is given in Appendix B. We proceed to look at an example application.

6.1 k-Satisfiability

Section 6.7.2 of *Probability and Computing*

We now return to constraint satisfaction problem introduced at the start of the lecture (as well as Section 3.2), in which there are n variables $x_1, \dots, x_n$, and m clauses $c_1, \dots, c_m$, with each clause being the OR of a subset of variables and/or negations of variables. Previously we discussed the problem of satisfying as many clauses as possible (the MAXSAT problem), whereas here we ask the question of whether or not it is possible to satisfy *all* clauses (the SAT problem).

Accordingly, let $\psi = c_1 \wedge \dots \wedge c_n$, which is called a *conjunctive normal form* (CNF) formula. We call ψ a k-CNF formula if all of its clauses have exactly k literals (i.e., variables or their negations). The problem of determining whether or not a satisfying assignment to a k-CNF formula exists is known as k-SAT. The following theorem uses the probabilistic method to give a general sufficient condition under which such an assignment is guaranteed to exist.

Theorem 6.3. *If none of the variables in a k-CNF formula ψ appear in more than $2^k/4k$ clauses, then there exists a satisfying assignment for ψ.*

- Note: Remarkably, the number of variables n and clauses m play no direct role! (Though the condition in the theorem does prevent m from growing arbitrarily large.)
- This result is vacuous for $k \le 4$ (since any such k gives $\frac{2^k}{4k} \le 1$), but becomes stronger as k increases.

Proof. Assign each variable independently and uniformly to either `TRUE` or `FALSE`. Let $\mathcal{B}_i$ be the bad event that the i-th clause is not satisfied. Thus, $p = \Pr[\mathcal{B}_i] = 2^{-k}$. Observe that $\mathcal{B}_i$ is mutually independent to all other events related to clauses that do not share variables with the i-th clause. Since each variable can appear in at most $2^k/4k$ clauses, $\mathcal{B}_i$ is mutually independent to all but $d \le k2^k/4k = 2^{k-2}$ events. Since

$$4dp \le 4 \times 2^{k-2} \times 2^{-k} = 1,$$

by Theorem 6.2, the probability of sampling an assignment with none of the bad events occurring is nonzero. □

Appendix

A Deterministic Algorithm for Finding a Large Cut

Section 6.3 of *Probability and Computing*

We have previously proved the existence of a cut (S_1, S_2) with value $C(S_1, S_2) \geq m/2$ by showing that a randomized strategy gives $\mathbb{E}[C(S_1, S_2)] \geq m/2$. We shall now see how derandomization can help us deterministically obtain a cut that always achieves $C(S_1, S_1) \geq m/2$.

Let $v_1, v_2, \ldots, v_n$ be an arbitrary order for the vertices. We will also designate ϵ_i as the set in which we place v_i, where ϵ_i can be either S_1 or S_2. We have already shown that if ϵ_i are independent with $\Pr(\epsilon_1 = S_1) = \Pr(\epsilon_1 = S_2) = \frac{1}{2}$, then $\mathbb{E}[C(S_1, S_2)] \geq m/2$. To "derandomize" this result, we will consider incrementally choosing each ϵ_i in a deterministic manner based on $\epsilon_1, \ldots, \epsilon_{i-1}$.

Let $\mathbb{E}[C(S_1, S_2)|\epsilon_1, \ldots, \epsilon_k]$ denote the average cut value given specific choices of $\epsilon_1, \ldots, \epsilon_k$ but with the remaining values $\epsilon_{k+1}, \ldots, \epsilon_n$ still being random. We seek to sequentially choose the ϵ_i values in manner that ensures the following:

$$\mathbb{E}[C(S_1, S_2)|\epsilon_1, \epsilon_2, \ldots, \epsilon_{k-1}] \leq \mathbb{E}[C(S_1, S_2)|\epsilon_1, \epsilon_2, \ldots, \epsilon_{k-1}, \epsilon_k]. \tag{2}$$

Before describing how to achieve this goal, we discuss its implications.

Recursively applying the inequality, we find that $\mathbb{E}[C(S_1, S_2)] \leq \mathbb{E}[C(S_1, S_2)|\epsilon_1, \epsilon_2, \ldots, \epsilon_n]$, where $\mathbb{E}[C(S_1, S_2)]$ represents the "purely random" scenario in which no vertices have been assigned to any set, and

$$\mathbb{E}[C(S_1, S_2)|\epsilon_1, \epsilon_2, \ldots, \epsilon_n]$$

represents the "final cut value" after having placed every vertex into either S_1 or S_2. (Hence, the $\mathbb{E}[\cdot|\cdot]$ operation is not actually "averaging" anything – by this point, $C(S_1, S_2)$ is fully specified.) Since $\mathbb{E}[C(S_1, S_2)] \geq m/2$, this implies that our deterministic algorithm will yield a cut with a value of at least $m/2$.

We establish the property (2) through induction. In the base case, when ϵ_1 is introduced, we have $\mathbb{E}[C(S_1, S_2)|\epsilon_1] = \mathbb{E}[C(S_1, S_2)]$. This base case holds by symmetry – it makes no difference whether we place the first node in S_1 or S_2, since the remaining ϵ_i all follow a 50/50 choice anyway. For the inductive step, fix the iteration index k and suppose that $\epsilon_1, \ldots, \epsilon_{k-1}$ have already been specified. At this stage, there are two options for placing v_k: either in S_1 or in S_2, each with a probability of 1/2. Consequently, we have

$$\mathbb{E}[C(S_1, S_2)|\epsilon_1, \epsilon_2, \ldots, \epsilon_{k-1}] = \frac{1}{2}\mathbb{E}\Big[C(S_1, S_2)\,\Big|\,(\epsilon_1, \epsilon_2, \ldots, \epsilon_{k-1}), \epsilon_k = S_1\Big] + \frac{1}{2}\mathbb{E}\Big[C(S_1, S_2)\,\Big|\,(\epsilon_1, \epsilon_2, \ldots, \epsilon_{k-1}), \epsilon_k = S_2\Big].$$

Since the maximum of two terms is at least as high as the average, it follows that

$$\begin{aligned}\max(\mathbb{E}[C(S_1, S_2)|\epsilon_1, \epsilon_2, \ldots, \epsilon_{k-1}, \epsilon_k = S_1], \mathbb{E}[C(S_1, S_2)|\epsilon_1, \epsilon_2, \ldots, \epsilon_{k-1}, \epsilon_k = S_2])& \\ \geq \mathbb{E}[C(S_1, S_2)|\epsilon_1, \epsilon_2, \ldots, \epsilon_{k-1}]&.\end{aligned}$$

Therefore, we know that we just need to determine the expectation with which set S_1 or S_2 is greater and put v_k in that set, and (2) will hold.

At this stage, it may not be obvious how to compute the two expectations in the $\max(\cdot, \cdot)$; this is detailed as follows. When calculating $\mathbb{E}[C(S_1, S_2)|\epsilon_1, \epsilon_2, \ldots, \epsilon_{k-1}, \epsilon_k = S_1]$, we already know from the k assignments of ϵ_i that certain edges contribute to the cut, and for the remaining ones, we know that the probability that its two endpoints will end up in different sets (thus incrementing the cut value by 1) is 1/2 Therefore, to calculate the expectation, using the linearity property, we can just sum the edges that already contribute and half of the remaining ones. This can be computed in linear time, and we can do the same for the other expectation.

Simplifying further, it is straightforward to show that choosing $\epsilon_k = S_1$ or $\epsilon_k = S_2$ is equivalent to placing the vertex v_k in the set where v_k has fewer neighbors. The edges that don't involve v_k contribute equally to both expectations. Therefore, by allocating v_k to the set with fewer neighbors, a greater number of its edges will contribute to the overall cut. With this, we can conclude the formulation of our deterministic algorithm:

- Begin by initially assigning the first vertex arbitrarily to either set S_1 or S_2.
- Subsequently, allocate each successive vertex to the set containing fewer neighbors. (If there is a tie, then either one can be chosen arbitrarily.)

We have demonstrated that this algorithm always gives a cut with minimally $m/2$ edges.

B (**Optional**) Proof and Extensions of the Lovász Local Lemma

Class 11.3 of *Randomized Algorithms*

Proof of Theorem 6.2:
We begin with the following useful lemma.

Lemma B.1. *Let $\mathcal{B}_1, \ldots, \mathcal{B}_m$ be a set of bad events such that for all $i \in [m]$, we have $\Pr[\mathcal{B}_i] \le p$ and $\mathcal{B}_i$ is mutually independent to all but at most d other events where $4dp \le 1$. For any set $S \subset \{\mathcal{B}_1, \ldots, \mathcal{B}_m\}$ and any $\mathcal{B}_i \notin S$,*

$$\Pr\left[\mathcal{B}_i \middle| \bigcap_{\mathcal{B}\in S} \bar{\mathcal{B}}\right] \le 2p.$$

Proof. We prove this by induction. Let $S \subset \{\mathcal{B}_1, \ldots, \mathcal{B}_m\}$ and $\mathcal{B}_i \notin S$. When $|S| = 0$, we have

$$\Pr\left[\mathcal{B}_i \middle| \bigcap_{\mathcal{B}\in S} \bar{\mathcal{B}}\right] = \Pr[\mathcal{B}_i] \le p \le 2p$$

where the first inequality holds by assumption in the lemma.

Now suppose the statement holds for all S' with $|S'| \le k$ and choose S with $|S| = k+1$. Partition S into two sets S_{ind} and S_{dep} such that $\mathcal{B}_i$ is mutually independent to S_{ind} and dependent to S_{dep}. Note that the choice of S_{ind} and S_{dep} might differ for different $\mathcal{B}_i$. If $|S_{\text{ind}}| = k+1$, then $S = S_{\text{ind}}$, which implies that

$$\Pr\left[\mathcal{B}_i \middle| \bigcap_{\mathcal{B}\in S} \bar{\mathcal{B}}\right] = \Pr\left[\mathcal{B}_i \middle| \bigcap_{\mathcal{B}\in S_{\text{ind}}} \bar{\mathcal{B}}\right] = \Pr[\mathcal{B}_i] \le p \le 2p.$$

Now suppose $|S_{\text{ind}}| \le k$. Then using the definition of conditional probability, we have

$$\Pr\left[\mathcal{B}_i \middle| \bigcap_{\mathcal{B}\in S} \bar{\mathcal{B}}\right] = \Pr\left[\mathcal{B}_i \middle| \left(\bigcap_{\mathcal{B}\in S_{\text{dep}}} \bar{\mathcal{B}}\right) \cap \left(\bigcap_{\mathcal{B}\in S_{\text{ind}}} \bar{\mathcal{B}}\right)\right] = \frac{\Pr\left[\mathcal{B}_i \cap \left(\bigcap_{\mathcal{B}\in S_{\text{dep}}} \bar{\mathcal{B}}\right) \middle| \left(\bigcap_{\mathcal{B}\in S_{\text{ind}}} \bar{\mathcal{B}}\right)\right]}{\Pr\left[\left(\bigcap_{\mathcal{B}\in S_{\text{dep}}} \bar{\mathcal{B}}\right) \middle| \left(\bigcap_{\mathcal{B}\in S_{\text{ind}}} \bar{\mathcal{B}}\right)\right]}.$$

The numerator term can be loosely bounded by

$$\Pr\left[\mathcal{B}_i \cap \left(\bigcap_{\mathcal{B}\in S_{\text{dep}}} \bar{\mathcal{B}}\right) \middle| \left(\bigcap_{\mathcal{B}\in S_{\text{ind}}} \bar{\mathcal{B}}\right)\right] \le \Pr\left[\mathcal{B}_i \middle| \left(\bigcap_{\mathcal{B}\in S_{\text{ind}}} \bar{\mathcal{B}}\right)\right] = \Pr[\mathcal{B}_i] \le p.$$

The denominator term can be bounded by first observing that the probability of its negation is

$$\begin{aligned}
\Pr\left[\left(\bigcup\nolimits_{\mathcal{B}\in S_{\text{dep}}} \mathcal{B}\right)\middle|\left(\bigcap\nolimits_{\mathcal{B}\in S_{\text{ind}}} \bar{\mathcal{B}}\right)\right] &\overset{(a)}{\le} \sum\nolimits_{\mathcal{B}\in S_{\text{dep}}} \Pr\left[\mathcal{B}\middle|\left(\bigcap\nolimits_{\mathcal{B}\in S_{\text{ind}}} \bar{\mathcal{B}}\right)\right] \\
&\overset{(b)}{\le} 2p \cdot |S_{\text{dep}}| \\
&\overset{(c)}{\le} 2pd \\
&\overset{(d)}{\le} \frac{1}{2},
\end{aligned}$$

where (a) holds due to the union bound, (b) holds due to the inductive hypothesis, (c) holds due to the assumption that $\mathcal{B}_i$ is mutually independent to all but at most d other events, and (d) holds due to the assumption that $4dp \le 1$. Thus, the denominator can be bounded by

$$\Pr\left[\left(\bigcap\nolimits_{\mathcal{B}\in S_{\text{dep}}} \bar{\mathcal{B}}\right)\middle|\left(\bigcap\nolimits_{\mathcal{B}\in S_{\text{ind}}} \bar{\mathcal{B}}\right)\right] = 1 - \Pr\left[\left(\bigcup\nolimits_{\mathcal{B}\in S_{\text{dep}}} \mathcal{B}\right)\middle|\left(\bigcap\nolimits_{\mathcal{B}\in S_{\text{ind}}} \bar{\mathcal{B}}\right)\right] \ge \frac{1}{2}.$$

Putting them together, we have

$$\Pr\left[\mathcal{B}_i\middle|\bigcap\nolimits_{\mathcal{B}\in S} \bar{\mathcal{B}}\right] = \frac{\Pr\left[\mathcal{B}_i \cap \left(\bigcap\nolimits_{\mathcal{B}\in S_{\text{dep}}} \bar{\mathcal{B}}\right)\middle|\left(\bigcap\nolimits_{\mathcal{B}\in S_{\text{ind}}} \bar{\mathcal{B}}\right)\right]}{\Pr\left[\left(\bigcap\nolimits_{\mathcal{B}\in S_{\text{dep}}} \bar{\mathcal{B}}\right)\middle|\left(\bigcap\nolimits_{\mathcal{B}\in S_{\text{ind}}} \bar{\mathcal{B}}\right)\right]} \le \frac{p}{1/2} = 2p. \qquad \square$$

We now prove Theorem 6.2 using Lemma B.1.

Proof of Theorem 6.2. By Lemma B.1, we have $\Pr[\bar{\mathcal{B}}_i|\bar{\mathcal{B}}_{i+1} \cap \cdots \cap \bar{\mathcal{B}}_m] \ge 1 - 2p$ for all $i \in [m]$. Thus,

$$\begin{aligned}
\Pr\left[\bigcap_{i=1}^{m} \bar{\mathcal{B}}_i\right] &= \Pr\left[\bar{\mathcal{B}}_1\middle|\bigcap_{i=2}^{m} \bar{\mathcal{B}}_i\right] \times \Pr\left[\bar{\mathcal{B}}_2\middle|\bigcap_{i=3}^{m} \bar{\mathcal{B}}_i\right] \times \cdots \times \Pr[\bar{\mathcal{B}}_{m-1}|\bar{\mathcal{B}}_m] \times \Pr[\bar{\mathcal{B}}_m] \\
&\ge (1-2p)^m \\
&> 0, \qquad \square
\end{aligned}$$

where the last step holds by assumption in the theorem.

Variations and Extensions:

Other variants of the Lovász local lemma also exist, one of which replaces $4pd \le 1$ by $ep(d+1) \le 1$, and more importantly, an *asymmetric version* in which the dependencies are encoded via a general graph in which different nodes (random variables) may have significantly different degrees (number of dependencies). We will not cover such variants.

Another natural follow-up question is whether it is possible to transform an existence argument relying on the Lovász local lemma into an efficient algorithm. In other words, we want a general algorithmic approach that allows us to avoid all the bad events $\mathcal{B}_1, \ldots, \mathcal{B}_m$. While we won't delve into this topic here, we direct interested readers to explore the relevant resources on *algorithmic Lovász Local Lemma* for more information (e.g., Class 12.1 of *Randomized Algorithms* linked on the first page).