CS265/CME309: Randomized Algorithms and Probabilistic Analysis
Lecture #10: The Probabilistic Method

Gregory Valiant*

October 30, 2019

1 Introduction

The *probabilistic method* is a surprisingly effective approach for proving that certain combinatorial objects exist. The basic recipe for applying the probabilistic method is the following:

1. To show that object C exists, define some probability space and random variable X.
2. Show that $\Pr[X = C] > 0$.

At its core, the probabilistic method relies on the following trivial fact: for any object C, if you can define a random variable, X, such that $\Pr[X = C] > 0$, then C must exist! The surprising part is how useful this basic recipe is: there are many sorts of combinatorial objects for which clean instantiations of the probabilistic method give the best known bounds. We will begin by giving one of the first known instantiations of this approach, due to Paul Erdos from 1947 [1]. We will then begin considering some of the algorithmic aspects of this—namely if we know that the desired object exists, how do we actually find it?

2 Ramsey Numbers

Definition 1. *The kth Ramsey number, R_k is the smallest n such that for every way of coloring the edges of the complete graph on n vertices with 2 colors, there exists a monochromatic k-clique—i.e. a set of k vertices such that all the $\binom{k}{2}$ internal edges are the same color.*

Trivially, $R_1 = 1$, and $R_2 = 2$. $R_3 = 6$: showing that $R_3 > 5$ is easy, one just needs to provide a 2-coloring of the complete graph on 5 vertices such that there is no monochromatic triangle; showing that $R_3 \leq 6$ is a bit annoying, though can be accomplished via a brute-force enumeration over all possible 2-colorings, for example. $R_4 = 18$, which is quite annoying to show, and we don't even know R_5 exactly (the best we know is that it is between 43 and 48). The situation only gets worse

for larger k: we don't even know R_{10} up to a multiplicative factor of 30 (its somewhere between 798 and 23,556). Part of the challenge is that telling whether $R_k \leq n$ naively would require enumerating over all $2^{\binom{n}{2}}$ possible colorings, and for each of these, even telling whether there is a monochromatic k-clique would, naively, require searching through all $\binom{n}{k}$ possible subsets of k of the n vertices.

Theorem 1. $R_k \in (2^{k/2}, 2^{2k})$.

Proof. We first prove that $R_k > 2^{k/2}$ via the probabilistic method. Let $n = 2^{k/2}$, and consider coloring each of the $\binom{n}{2}$ edges red or blue according to the outcomes of independent coin flips. For a given set of k vertices, the probability they form a monochromatic k-clique is $(1/2)^{\binom{k}{2}-1} = 2^{-k^2/2+k/2+1}$, where this expression is because we can pick the first edge to be any color, and then we must color the remaining $\binom{k}{2} - 1$ edges with that color. Via a union bound over the $\binom{n}{k}$ possible sets of k vertices, we have

$$\Pr[\text{exists monochrome } k\text{-clique}] \leq \binom{n}{k} 2^{-k^2/2+k/2+1} \leq \frac{n^k}{k!} 2^{-k^2/2+k/2+1} = \frac{2^{k^2/2}}{k!} 2^{-k^2/2+k/2+1} = \frac{2^{k/2+1}}{k!} < 1.$$

Hence, the probability that this random edge coloring yields a coloring *without* a monochromatic k-clique is > 0, and hence there must exist such a coloring.

To show that $R_k < 2^{2k}$ we can proceed via an inductive argument. Define $R_{a,b}$ to be the minimal n such that any 2-coloring (say red and blue) of the complete graph on n vertices *either* has a monochromatic red clique of size at least a, or a monochromatic blue clique of size at least b. First observe that $R_{a,b} = R_{b,a}$, by symmetry, and $R_{1,k} = 1$, as all colorings have a red 1-clique (since that doesnt even involve any red edges).

Consider a 2-coloring of a graph on $n = 1 + R_{a-1,b} + R_{a,b-1}$ vertices. Fix a vertex v, and let S_r denote the subset of vertices that are connectd to v via red edges, and S_b denote the subset of vertices connected to v via blue edges. By construction, $|S_r| + |S_b| + 1 = n = 1 + R_{a-1,b} + R_{a,b-1}$, and hence either $|S_r| \geq R_{a-1,b}$ or $|S_b| \geq R_{a,b-1}$. In the case that $|S_r| \geq R_{a-1,b}$, either S_r has a blue clique of size b, or, a red clique of size $a-1$ all of whose vertices are connected to v via red edges, in which case the graph has a red clique of size a. An analogous statement holds in the case that $|S_b| \geq R_{a,b-1}$. In the case that $R_{a-1,b} + R_{a,b-1}$ is even we don't even need the extra $+1$ because any way of splitting $n = R_{a-1,b} + R_{a,b-1} - 1$ vertices into two sets will have the property that either one of the sets has size at least $R_{a-1,b}$ or the other has size at least $R_{a,b-1}$.

Hence we have shown the following:

$$R_{a,b} \leq 1 + R_{a-1,b} + R_{a,b-1} \text{ if } R_{a-1,b} + R_{a,b-1} \text{ is odd, and otherwise } R_{a,b} \leq R_{a-1,b} + R_{a,b-1}.$$

Inductively, we will argue that for any a, b, $R_{a,b} \leq 2^{a+b}$. For the base case, note that $R_{1,2} = R_{2,1} = 1$ and $R_{2,2} \leq R_{1,2} + R_{2,1} = 2 < 2^4$. For the inductive step, assuming $R_{a,b} \leq 2^{a+b}$, for all a, b that sum to at most c, then we claim that it holds for all a, b since $R_{a+1,b} \leq R_{a,b} + R_{a+1,b-1} \leq 2^{a+b} + 2^{a+1+b-1} \leq 2^{(a+1)+b}$, as desired. □

The proof of the above Theorem wasn't too difficult. Surprisingly, despite the huge gap between the upper and lower bounds, and the fact that people have studied Ramsey numbers for almost a century, we don't know how to improve significantly on either of them. Specifically, we don't know how to tighten the exponents by even a tiny constant factor: there is no constant $\epsilon > 0$ for which we can show that $R_k > 2^{(1+\epsilon)k/2}$ or that $R_k < 2^{(1-\epsilon)2k}$.

3 Independent Sets

In the Ramsey Number example, the distribution we came up with—coloring the edges uniformly at random—was a very simple/natural distribution. For some examples of the probabilistic method, we will need slightly more creative distributions.

Definition 2. *Given a graph, an* independent set *is a subset of the vertices such that no pair is connected via an edge.*

Computing the size of the largest independent set of a graph is NP-hard. The following theorem, however, guarantees the presence of a fairly large independent set, provided the number of edges in the graph is not too large:

Theorem 2. *For any graph with n vertices, and $m \geq n/2$ edges, there exists an independent set of size at least $\frac{n^2}{4m}$.*

Proof. We proceed via the probabilistic method. Consider the following randomized process for finding an independent set:

1. For each node, independently remove it and all the edges incident to it, with probability $1-\frac{n}{2m}$.
2. For each remaining edge, arbitrarily (it doesnt matter how) delete one of its two endpoints.
3. Return the set of remaining vertices.

The above process generates an independent set because of the second step—even without the first step, the second step ensures that there is no edge connecting a pair of returned vertices, because that edge would have removed one of those two endpoints. We now analyze the expected size of this returned set. Letting X denote the number of vertices that survive step 1, we have that by linearity of expectation,

$$\mathbf{E}[X] = n\frac{n}{2m} = \frac{n^2}{2m}.$$

Letting Y denote the number of edges surviving after the first step, we have that the probability an edge survives the first step is $(\frac{n}{2m})^2$, since it survives that step if and only if both its endpoints survive. Hence

$$\mathbf{E}[Y] = m(\frac{n}{2m})^2 = \frac{n^2}{4m}.$$

Finally, note that the number of remaining nodes is at least $X - Y$, because each of the Y edges can be responsible for the removal of at most 1 vertex in step 2, and hence

$$\mathbf{E}[\text{number returned vertices}] \geq \mathbf{E}[X - Y] = \mathbf{E}[X] - \mathbf{E}[Y] = \frac{n^2}{2m} - \frac{n^2}{4m} = \frac{n^2}{4m}.$$

To conclude, note that if the prescribed randomized process, in expectation, will return an independent set of size at least k, there must exist an independent set of size at least k. □

One comment: In the theorem statement, we assumed that $m \geq n/2$. If this were not true, then in step 1, we would be removing each vertex with probability $1 - n/(2m) < 0$, which is not valid (since probabilities cannot be less than 0).

4 Max-Cut, k-SAT, and De-randomization via Conditional Expectation

The Max-Cut problem is defined as follows: Given a graph $G = (V, E)$, partition V into two sets, A, B, so as to maximize the number of edges with one endpoint in A and one endpoint in B. This is a standard NP-hard problem, though, as we will see, an efficient greedy algorithm will always cut at least $|E|/2$ edges.

Rather than directly describing and analyzing the greedy algorithm, we will present it as the consequence of de-randomizing a simple randomized scheme.

Proposition 3. *Consider the randomized scheme that partitions V into A and B according to flips of independent fair coins for each vertex. The expected number of edges cut by this scheme is $|E|/2$.*

Proof. By linearity of expectation, the expected number of edges cut is the sum of the probabilities that each edge is cut. For a given edge (u, v), it will be cut with probability $1/2$, as this is the probability v and u are assigned different sets. □

The idea behind how to "de-randomize" this scheme is as follows: suppose we choose an ordering of the vertices, $v_1, \ldots, v_n$, and assign each vertex to either set A or B iteratively. Given our assignment for $v_1, \ldots, v_{t-1}$, we will assign v_t to whichever set maximizes the expected number of edges cut, *given the assignment of* $v_1, \ldots, v_{t-1}$ where the expectation is with respect to randomly assigning $v_{t+1}, \ldots, v_n$. The rational is as follows, where the expectations in the following equation are with respect to independently assigning each of $v_t, \ldots, v_n$ to A or B with probability $1/2$:

$$\begin{aligned}\mathbf{E}[\text{cut size}|\text{assignment to } v_1, \ldots, v_{t-1}] \;=\; & \frac{1}{2}\mathbf{E}[\text{cut size}|v_t \in A, \text{assignment to } v_1, \ldots, v_{t-1}] \\ & + \frac{1}{2}\mathbf{E}[\text{cut size}|v_t \in B, \text{assignment to } v_1, \ldots, v_{t-1}],\end{aligned}$$

hence at least one of the these two terms must be at least half the left hand side. Each of these two terms is straightforward to evaluate: $\mathbf{E}[\text{cut size}|v_t \in A, \text{assignment to } v_1, \ldots, v_{t-1}]$ is simply the sum of the number of edges between $v_1, \ldots, v_t$ that are cut in the prescribed assignment, plus $1/2$ times the number of edges with an endpoint in $v_{t+1}, \ldots, v_n$ (since these edges are cut with probability 1/2 over the randomness in assigning these remaining nodes to the two sets).

Before we have assigned any of the vertices, the expected number of edges cut is $|E|/2$, and after each successive assignment, the expectation conditioned on the assigned vertices is non-decreasing as we iteratively assign vertices. Hence, after all vertices are assigned, we have obtained a deterministic algorithm that cuts at least half the edges.

What is this algorithm actually doing? Well, at each step, we put v_t into whichever of A or B cuts more edges, which corresponds to asking whether node v_t has more neighbors among $v_1, \ldots, v_{t-1}$ in set A versus in set B. Hence this is simply the iterative greedy algorithm!

4.1 k-SAT

The same high-level arguments can also be applied to k-SAT formulas. For example, in the case of a 3-SAT formula over binary variables $x_1, \ldots, x_n$, if each clause contains exactly 3 variables, then a random assignment to $x_1, \ldots, x_n$ satisfies, in expectation, a $(1 - \frac{1}{2^3}) = 7/8$ fraction of clauses

(since a clause is not satisfied if each of the 3 variables gets "unlucky" in its random assignment. Hence, via the probabilistic method, there must exist a satisfying assignment that satisfies at least this fraction of clauses?

How can we make an efficient algorithm for finding such an assignment? Just as in the max-cut example, we can iteratively assign the variables $x_1, \ldots, x_n$ by sequentially assigning x_t so as to maximize the expected number of satisfied clauses given the assignment to $x_1, \ldots, x_{t-1}$, and with respect to the randomness of assigning $x_{t+1}, \ldots, x_n$. Computing these conditional expectations is straightforward, since for each clause, if the assignment to $x_1, \ldots, x_t$ doesnt already satisfy or falsify it, the probability it is satisfied is $1 - 1/2^k$ where k denotes the number of unassigned (i.e. randomized) variables that are left in that clause.

References

[1] P. Erdos. Some remarks on the theory of graphs. *Bull. Amer. Math. Soc.*, 53(4):292–294, 04 1947.