6 The Lovász local lemma

The local lemma is an extremely important tool, and we saw a basic use of it early on when finding bounds for diagonal Ramsey numbers. Recall that the vanilla way to do this (a union bound) doesn't use the fact that there are few local dependencies between our "bad events."

Most of our proofs in the previous few sections worked with high probability, but now we are shifting gears: this method is getting us a **small but nonzero** probability of success.

Theorem 6.1 (Local lemma, symmetric case)
Given events $A_1, \cdots, A_n$, where all $\Pr(A_i) \le p$, if all A_is are mutually independent from the set of all A_js except at most d of them, and

$$ep(d+1) \le 1,$$

then with positive probability, none of the events occur.

(e is the best constant we can have here.) If the ps are small, for instance if all $A_i \le \frac{1}{n}$, union bounding tells us that we have a scenario where none of the events happen. Alternatively, if the A_is were independent, we can just multiply the non-event probabilities. So this is a sort of happy medium between the two extremes!

Definition 6.2
An event A_0 is **mutually independent** of $\{A_1, \cdots, A_m\}$ if A_0 is independent of every event of the form $B_1 \cap B_2 \cap \cdots \cap B_m$, where all $B_i = A_i$ or $\overline{A_i}$.

In other words, any boolean information tells us nothing about A_0. As a warning, **this is different from saying that A_0 and A_i are independent for all i**, since events can be pairwise independent but not mutually independent. (For example, consider the three variables $X_1, X_2, X_1 + X_2 \bmod 2$, where X_1 and X_2 are uniform among $\{0, 1\}$.)

Almost all applications of the local lemma are of the following form: we have a set of variables Ω, and each A_i depends on a subset $S_i \subset \Omega$, so two events A_i and A_j are independent if $S_i \cap S_j \neq \varnothing$.

Let's start by looking at a few applications before returning to the proof of the theorem.

6.1 Coloring: hypergraphs and real numbers

Let's first consider a hypergraph: we wish to color all vertices red and blue so that no edge is monochromatic.

Theorem 6.3
A k-uniform hypergraph G is 2-colorable if every edge intersects at most $d = \frac{2^{k-1}}{e} - 1$ other edges.

Proof. Color the edges uniformly at random. For each edge f, let A_f be the event that f is monochromatic: each occurs with probability $2^{-k+1} \equiv p$.

Each bad event A_f is mutually independent of all other events $A_{f'}$ if f and f' do not share any vertices. (Note that here our probability space Ω is the vertices of our graph G.) By the Lovász local lemma, since $ep(d+1) \ge 1$, with positive probability, there is a graph whose coloring is proper. □

What are some consequences of this?

Question 6.4. *For which k is every k-uniform, k-regular hypergraph 2-colorable?*

A triangle is not 2-colorable, and the Fano plane is not 3-colorable, so our statement is false for $k = 2, 3$. Well, if we apply the theorem we've just proved, every edge intersects $k(k-1)$ other edges, so if

$$k(k-1) \geq \frac{2^{k-1}}{e} - 1,$$

then the statement is true. It turns out that is good enough for $k \geq 9$ - what about $4 \leq k \leq 8$? It's known that the statement is actually true for all $k \geq 4$, but that is much harder to prove.

Let's ask a related question now: say we're looking at k-colorings of the real numbers. Basically, we have some function that assigns one of the first k positive integers to each real number:

$$c : \mathbb{R} \to [k],$$

or we can think of our domain as $\mathbb{Z}$ instead if the Axiom of Choice is annoying to deal with. Say that a subset $T \subset \mathbb{R}$ is **multicolored** with respect to c if all k colors appear in $c(T)$.

Theorem 6.5

Fix k. Given a subset S of the reals with $|S| = m$, we can color the real numbers with k colors so that every translate of S is multicolored if

$$e(m(m-1)+1)k\left(1 - \frac{1}{k}\right)^m \leq 1.$$

Doing some calculations (omitted), this can be written as $m > (3+\varepsilon)k \log k$ for sufficiently large k. This is a hypergraph coloring problem, but we can still use the Lovász local lemma to solve it.

Proof. First, we'll show that we can color every finite subset $X \subset \mathbb{R}$ such that every $x + S \subset X$ is multicolored.

Color X uniformly at random among the k colors. Our "bad events" correspond to elements $x \in X$ where $x + S$ is not multicolored, so we can think of having a hypergraph where the vertices are elements of X and the edges correspond to translates. Each fixed translate $x + S \subset X$ is not multicolored with probability (union bound, pick a color to not include)

$$p \leq k\left(1 - \frac{1}{k}\right)^m.$$

Furthermore, each translate of S intersects at most $m(m-1)$ other translates (pick a pair $a \in S, a' \in S'$ to overlap). So now by the local lemma, there exists a good coloring for every finite set as long as the $ep(d+1) \leq 1$ condition is satisfied.

But how do we extend this to the reals? We use compactness: Tikhonov's theorem says that if we assign a discrete topology to k points, then $[k]^{\mathbb{R}}$ is compact. A point in $[k]^{\mathbb{R}}$ is a map from the reals to 1 through k, which is basically a coloring.

So for every $x \in \mathbb{R}$, let $C_x \subset [k]^{\mathbb{R}}$ be a subset of colorings so that $x + S$ is multicolored. Our goal is to show that we can make all of the translates multicolored at once. We've shown so far that every finite intersection of C_xs is nonempty by the local lemma. Under the product topology, C_x is closed for all $x \in \mathbb{R}$ (the set C_x is only limited by the finite set of reals in X), so the intersection of all closed sets

$$C = \bigcap_{x \in \mathbb{R}} C_x$$

is nonempty as well, and that's a coloring of all the real numbers. □

The logic here is "we have a bunch of closed sets, any finite intersection is nonempty, so the infinite intersection is nonempty." To elaborate on this, think of $\mathbb{Z}$ instead of $\mathbb{R}$. Then this is a diagonalization argument: if we can color every prefix of the positive integers, then we can write down something for each prefix, and find infinitely many ways of coloring 1, then 2, and so on.

6.2 Coverings of $\mathbb{R}^3$

Here's a motivating fact that we won't prove:

Fact 6.6 (Mani-Leviska, Pach)
For every $k \geq 1$, there exists a nondecomposable k-fold covering of $\mathbb{R}^3$ by open unit balls. Here, a k-fold covering means that every point in $\mathbb{R}^3$ is covered at least k times. A covering is **decomposable** if it can be partitioned into two coverings.

(This generalizes beyond 3 dimensions, by the way.) Maybe all the points are covered a uniform number of times: can we say that there aren't outliers in our covering? The answer is no:

Theorem 6.7
There exists an absolute constant $c > 0$ such that every nondecomposable k-fold covering of $\mathbb{R}^3$ by open unit balls must cover some point at least $c2^{k/3}$ times.

Proof. If we try to decompose a covering, we're coloring the unit balls red and blue so that each color forms a covering of $\mathbb{R}^3$ on its own. Our "bad events" here are where $x \in \mathbb{R}^3$ is only colored by one color - if no bad events occur, then we have a successful decomposition.

Construct an infinite hypergraph where the vertices F are the set of balls and the edges are

$$E_x = \text{ the set of balls in F containing } x.$$

So the edges are

$$E(H) = \{E_x : x \in \mathbb{R}^3\}$$

where two points in the same "cell" correspond to the same edge - this means decomposable is the same as 2-colorable in our hypergraph.

For the sake of contradiction, assume every point is covered at most t times, where t is to be determined. By a compactness argument, it suffices to show that every finite subgraph of H is 2-colorable.

We claim that every edge intersects $\lesssim t^3$ other edges.Two edges intersect if they share at least one ball in common: if we fix $E_x \cap E_y$ that are intersecting, since every point is covered at most t times, there are at most $4^3 t$ balls involved in E_y, since anything intersecting with x has to be within a ball of radius 4. It turns out m balls can cut $\mathbb{R}^3$ into at most $m^3 + 1$ regions, and each region corresponds to an edge. So there are at most $4^9 t^3 + 1$ other edges that any edge can intersect.

So now the rest is just applying the Lovász local lemma. Each edge is monochromatic with probability at most 2^{-k+1} (since it is covered at least k times), and in the dependency graph, the number of intersections is at most $\lesssim m^3$. So we can apply local lemma if

$$t^3 2^{-k+1} < c$$

for some sufficiently small constant c. So for $t \lesssim 2^{k/3}$, the graph is decomposable by Lovász, and therefore every nondecomposable k-fold covering must cover some point $\gtrsim 2^{k/3}$ times. □

By the way, to prove the claim that m balls can cut $\mathbb{R}^3$ in at most m^3+1 regions, we use induction. Adding a new ball creates regions if we have intersections with other balls – this is at most the number of regions on the sphere cut by m circles, and we can make arguments from there.

6.3 The general local lemma and proof

As previously stated, we have a bunch of bad events $A_1, \cdots, A_n$ that we are trying to avoid. We occasionally have that A_i is mutually independent of all other A_j except for some set $N(i)$ for each i. (Notice that $N(i)$ does not include i.)

Definition 6.8

The **dependency graph** is constructed by having a vertex for each bad event i and joining i to the set $N(i)$ (of dependencies to i).

This graph is sometimes directed, but in almost all applications, it's sufficient to make it undirected. For example, if we have a hypergraph coloring problem and we're coloring each vertex at random, the dependency connects edges that have nonzero vertex intersection. In such a setup, we have the following:

Theorem 6.9 (Local lemma, symmetric form)

If every node in the dependency graph has degree at most d, and every event has probability at most p, then there is a positive probability that no bad events occur as long as

$$ep(d+1) \le 1.$$

We're going to be proving a more general form of this:

Theorem 6.10 (Local lemma, general form)

If we have real numbers $x_1, \cdots, x_n \in [0,1)$ such that for all i,

$$\Pr(A_i) \le x_i \prod_{j \in N(i)} (1-x_j),$$

then with probability at least $\prod_{i=1}^n (1-x_i)$, no event A_i occurs.

Here, note that x_i is not the probability of A_i: it's something larger than x_i which may be weighted down by the other terms.

By the way, notice that if we can find values x_i to plug in, we can get the symmetric case from the general case:

Deducing the symmetric case. Set all $x_i = \frac{1}{d+1} < 1$. Then notice that

$$x_i \prod_{j \in N(i)} (1-x_j) = \frac{1}{d+1}\left(1-\frac{1}{d+1}\right)^{|N(i)|} \ge \frac{1}{d+1}\left(1-\frac{1}{d+1}\right)^{d} > \frac{1}{(d+1)e},$$

and if we have the hypothesis of the symmetric case of the lemma, then

$$x_i \prod_{j \in N(i)} (1-x_j) \ge p \ge \Pr(A_i),$$

and therefore the general case's hypothesis must hold as well. □

Here's another way to specialize the general form:

Corollary 6.11

If all events have probability $\Pr(A_i) < \frac{1}{2}$, and

$$\sum_{j \in N(i)} \Pr(A_j) \leq \frac{1}{4}$$

for all i, then there is a positive probability that no A_i holds.

Proof. Set $x_i = 2\Pr(A_i)$; this is always less than 1, and the product

$$x_i \prod_{j \in N(i)} (1 - x_j) = 2\Pr(A_i) \prod_{j \in N(i)} (1 - x_j) \geq \Pr(A_i),$$

so all probabilities are smaller than their corresponding products, and we can use the theorem in its general form. □

We'll now present the original proof of the local lemma from its publication – it uses induction.

Proof. Say we have n events. Let S be a subset of $[n]$ which indexes our bad events: we'll induct on $|S|$. The induction hypothesis is the following:

Proposition 6.12

If we have an event $i \notin S$, then

$$\Pr\left(A_i \mid \bigwedge_{j \in S} \overline{A_j}\right) \leq x_i.$$

Basically, this is the probability A_i occurs, conditioned on the fact that none of the events indexed by S occur.

If we prove this, then by Bayes' formula, the probability that none of the events occur is at least

$$\Pr(\overline{A_1}) \cdot \Pr(\overline{A_2} \mid \overline{A_1}) \cdot \cdots \cdot \Pr(\overline{A_n} \mid \overline{A_1 A_2 A_{n-1}}) \geq (1 - x_1)(1 - x_2) \cdots (1 - x_n),$$

where the last inequality comes from the inductive hypothesis. So this would imply the local lemma.

First of all, this proposition is easy to show when S is empty, since the probability that A_i occurs is

$$x_i \prod_{j \in N(i)} (1 - x_j) \leq x_i.$$

Now for the inductive step, we know i has some neighbors in S: call this set $S_1 = S \cap N(i)$, and call everything else $S_2 = S \setminus S_1$.

Let's understand the conditional probability $\Pr\left(A_i \mid \bigwedge_{j \in S} \overline{A_j}\right)$. We can separate this into contributions from S_1 and contributions from S_2 using Bayes' rule:

$$= \frac{\Pr\left(A_i \wedge \bigwedge_{j \in S_1} \overline{A_j} \mid \bigwedge_{j \in S_2} \overline{A_j}\right)}{\Pr\left(\bigwedge_{j \in S_1} \overline{A_j} \mid \bigwedge_{j \in S_2} \overline{A_j}\right)}$$

First, we can upper-bound the numerator by forgetting the dependencies that are hard to control: this is at most

$$\Pr\left(A_i \mid \bigwedge_{j \in S_2} \overline{A_j}\right)$$

since we're just removing some conditions on what needs to happen. But now since A_i is mutually independent from all of its neighbors, this is just $\Pr(A_i) \le x_i \prod_{j \in N(i)} (1 - x_j)$ by the assumed conditions.

Meanwhile, we can also lower bound the denominator. Label the elements of $S_1 = \{j_1, \cdots, j_r\}$; as before, we can write the denominator as a product of conditional probabilities

$$\Pr\left(\overline{A_{j_1}} \mid \bigwedge_{j \in S_2} \overline{A_j}\right) \Pr\left(\overline{A_{j_2}} \mid \overline{A_{j_1}} \wedge \bigwedge_{j \in S_2} \overline{A_j}\right) \cdots \Pr\left(\overline{A_{j_r}} \mid \overline{A_{j_1}} \wedge \cdots \wedge \overline{A_{j_{r-1}}} \bigwedge_{j \in S_2} \overline{A_j}\right)$$

S_2 is a smaller set than S (or else there are no dependencies in S at all to A_i, in which case this is easy). So by the induction hypothesis, each event $\overline{A_{j_k}}$ occurs with probability at most x_i, so this is at least

$$(1 - x_{j_1})(1 - x_{j_2}) \cdots (1 - x_{j_r}) \ge \prod_{j \in N(i)} (1 - x_j)$$

(since the LHS product is some subset of the RHS product). Putting the numerator and denominator together, we're done! The probability that A_i occurs given that none of the events in S occurs is at least

$$\frac{x_i \prod_{j \in N(i)} (1 - x_j)}{\prod_{j \in N(i)} (1 - x_j)} = x_i,$$

as desired, completing the inductive step. □

6.4 The Moser-Tardos algorithm

We now know that under certain circumstances, it is possible to avoid all of our bad events. But is there a nice way to algorithmically determine an example of this? It's not even obvious how to do a randomized algorithm, since the chance of success (avoiding all bad events) is generally so small.

Consider a **random variable model** to make our problem less abstract: we have a collection of independent random variables, where each event A_i depends only on some subset of variables. We have a dependency graph, where $A \sim B$ if A and B share common variables. Our goal is to find a way to flip the independent variables so that the A_is all do not occur.

It would seem like this algorithm could be very sophisticated, but here's one that isn't!

Algorithm 6.13

First, initialize all our random variables with some values. While some bad event A_i occurs (pick one arbitrarily), resample all of the variables that A_i depends on, since A_i only depends on some finite set of variables. Maybe this induces some other bad events: we just keep running the while loop.

We might be worried that this algorithm might never terminate, or on average, maybe it terminates after an exponential number of iterations. It turns out this never happens:

Theorem 6.14 (Moser–Tardos)

If the Lovász local lemma conditions hold, then the algorithm is expected to resample each A_i at most $\frac{x_i}{1 - x_i}$ times. In particular, the expected number of iterations of the while loop is at most

$$E = \sum_{i=1}^{n} \frac{x_i}{1 - x_i}.$$

Note that this algorithm is agnostic of the x_is: it doesn't change based the parameters that we're using.

There are two key concepts in this proof that are needed. We need an **execution log**, which is a list of the resampled events at each step (so we add on 1 event per run through the while loop).

We also need a **witness tree**, which is a finite rooted tree labeled by events such that the children of i are distinct from each other and are a subset of $N(i) \cup \{i\}$ (the neighbors of i and itself).

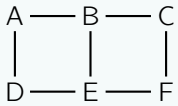
Why is this called a witness tree? We're going to use a prefix of our execution log to make a tree (though the lumberjack may say to go the other way).

Basically, given a prefix of the execution log, read the log **right to left**. The rightmost event is the root of the tree, and for each subsequent event A, if some node in the tree overlaps with A (in other words, it lies in $N(A) \cap \{A\}$, then add A as a child of a deepest such node (arbitrarily) If nothing is dependent with A, just discard it.

The key idea here is that this witness tree tracks some of the possible ways we could have progressed in our algorithm from bottom to top.

Example 6.15

Let's say we have the dependency graph



and we have an execution log with prefix $FBDEADBFDC$.

When we make our rooted tree, C is the root. D is next-to-last, but it couldn't have caused C to be sampled since C and D are not overlapping, so it is discarded. F is adjacent with C, though, so F is now a child of C. This represents the statement "C could have been sampled as a result of F."

Now B must be another child of C for the same reason the next D is discarded, A is a child of B, E is a child of B or F (arbitrarily choose F), and so on.

Events at the same depth are independent, because if they weren't, one would be forced to be a child of the other when it is inserted.

Lemma 6.16

For a given log, all prefixes produce distinct trees.

This is not too hard to show. Given any tree with root A, the number of times A appears in it is just the number of As up to that point in the prefix of the log. So all such trees must have a different number of As, and all trees with root A are different from trees of root B, and so on.

Lemma 6.17

For a given witness tree T, the probability that T appears as a witness tree of some prefix is at most

$$\prod_{v \in T} \Pr(A_{[v]}),$$

where $A_{[v]}$ is the event corresponding to the node v in the witness tree.

To show this, we'll introduce some randomness.

Definition 6.18

Let a **simulation** of T to be the following process: visit all nodes of T in reverse depth (the order doesn't matter for each given level). Then resample all the variables at each visited node. The simulation **succeeds** if all bad events encountered do occur.

Proof of lemma. How can we ensure that there's a level of consistency between the tree and simulation? We're going to do a **coupling** of the two, where we basically have two processes $X, Y \to (X, Y)$ that act in parallel. This way, if we look at just X or Y alone, it looks identical to an initial distribution, but X and Y aren't necessarily independent.

Our goal is to do this in such a way that **the simulation always succeeds if T appears as a witness tree**. Then, the probability that T appears must be at most the probability that the simulation succeeds, which gives us a bound we want.

What's the probability the simulation succeeds? This is easy: each step in the tree resampling is independent, so this is just

$$\prod_{v \in T} \Pr(A_{[v]}).$$

To make this useful, we need to find a common source of randomness. Let's say we have access to an infinite list of realizations of each random variable, and resampling is just taking the next realization of the list. The key idea is to use the same list to run both processes. We claim that if T appears as a witness tree, the simulation must succeed.

Why? During the execution step, there's an initialization of all variables. Since T appears as a witness tree, we know that looking at our execution log, each of the bottom nodes must have been "bad events" that were logged; since this is coupled directly to our witness tree, the same bad event must have occurred there. Now keep propagating up the tree; we'll notice that each event must have occurred because the prior resampling did not work. □

One last tool we're going to use in our proof is the **multitype Galton-Watson branching process.** Generate a tree in the following way: specify a root labeled by some event A_i, and for each possible child $A_j \neq A_i$, keep it with probability x_j (this is not the probablity of A_j - it is the corresponding real number from the statement of Lovász). Repeat the same process for each node; each child survives with some probability x_j or dies with some probability $1 - x_j$.

When this terminates, we get some finite or infinite tree.

Lemma 6.19

Then T is yielded with probability

$$p_T = \frac{1 - x_i}{x_i} \prod_{v \in T} x'_{[v]},$$

where

$$x'_{[v]} = x_{[v]} \prod_j (1 - x_{[j]}).$$

Basically, we're just multiplying the probabilities of each vertex working out, and the normalizing factor comes from the fact that our root is already formed for sure. We can think of this as a weighing of our trees in an information-theoretic manner.

It's time for us to prove the Moser-Tardos theorem:

Proof of Theorem 6.14. The number of times an event A_i is resampled is the number of times a witness tree is

generated, rooted at A_i. We expect this to be

$$\sum_{T \text{ rooted at } A_i} \Pr(T \text{ appears as a witness tree}).$$

From our construction by coupling, this is bounded by the expression

$$\sum_{T \text{ root at } A_i} \prod_{v \in T} \Pr(A_{[v]}).$$

Now recall that the x_is in the Lovász local lemma are at least their corresponding probabilities:

$$\leq \sum_{T \text{ root at } A_i} \prod_{v \in T} x'_{[v]}.$$

Now by the lemma above, this is just

$$= \frac{x_i}{1 - x_i} \sum_{T \text{ rooted at } A_i} P_T.$$

But summing over P_T gives the probability that some certain trees are our final result, so that sum can be at most 1. Therefore

$$\mathbb{E}[A_i \text{ is sampled}] \leq \frac{x_i}{1 - x_i},$$

and summing over all events A_i gives the desired result. □

6.5 A computationally hard example

Unfortunately, there are instances where we'd like to use the local lemma, but an algorithm may not quickly yield an answer. Here's a simple example of that:

Example 6.20

Let $q = 2^k$ be an integer, and let f be a bijection from $[q] \to [q]$. Let y ben an element of $[q]$; then let A_i be the "bad event" that $f(x)$ and y disagree on the ith bit, where we choose x uniformly on the domain $[q]$. (There are k events $A_1, \cdots, A_k$, since $q = 2^k$ has k bits.)

Notice that all events A_i are mutually independent, since the bits behave independently, by the "local lemma" (a very trivial use of it, aka just independent events), there exists an x such that $f(x) = y$.

But algorithmically, can we always find this efficiently? As a foundation of cryptography, the answer is (believed to be) no. A concrete example is the function $f : \mathbb{F}_q \to \mathbb{F}_q$ sending $f(x) = g^x$ for some generator g (except 0 at $x = 0$). Then we just have the **discrete logarithm** problem, which is believed to be computationally difficult.

6.6 Back to independent sets

We know that in a graph with maximum degree Δ, we have an independent set of size at least $\frac{|V|}{\Delta+1}$. (Take a vertex, include it and throw away all of its neighbors, and repeat.) But we can achieve something better with the local lemma:

Theorem 6.21

Let G be a graph with maximum degree Δ, and let the vertex set $V = V_1 \cup V_2 \cup \cdots \cup V_r$ be partitioned into sets of size $|V_i| \geq 2e\Delta$. Then there exists an independent set with one vertex in each V_i.

Then the size of this set is similar to what we have naively (we lose a constant), but we have much more control over what the set looks like!

Proof. First, we may assume all $|V_i| = k = \lceil 2e\Delta \rceil$ for simplicity: we can always toss away extra vertices that we just decide not to use.

For each V_i, pick a uniformly random v_i independently from all the others: our goal is to show that with positive probability, we get an independent set.

What are our bad events? We don't want our vertices to be adjacent, and there's a few ways we can try to approach this.

Attempt 1: Let's let A_{ij} be the events that v_i and v_j are connected by an edge. Then in the worst case, $\Pr(A_{ij}) \le \frac{\Delta}{k}$, since any given vertex in V_i can only be connected to Δ of the things in V_j.

But how large is our dependency set? The events A_{ij} and A_{rs} are dependent whenever some vertex in $V_i \cup V_j$ is adjacent to some vertex in $V_r \cup V_s$. The maximum degree of the dependency graph is therefore $2\Delta k$: each of the $2k$ vertices in $V_i \cup V_j$ has at most Δ neighbors. So the condition for local lemma requires

$$\frac{\Delta}{k} \cdot 2\Delta k$$

to be approximately constant, but this is too large and this method fails.

Attempt 2: So an alternative way to make bad events is to let A_e be the event that both endpoints of e are chosen, where $e \in E(G)$ is an edge between two different V_i, V_j. We know that

$$\Pr(A_e) \le \frac{1}{k^2},$$

since there are k vertices in each of V_i and V_j, and what do we know about the dependency graph? Again, A_e and A_f can be dependent if and only if one of the edges is in $V_i \cup V_j$: thus the maximum degree is $2\Delta k$. Thus by an application of the local lemma, as long as we have

$$\frac{1}{k^2}(1 + 2\Delta k) < \frac{1}{e},$$

we are good. □

So in many applications, it's important to pick the right events to think about.

6.7 Graphs containing large cycles

Theorem 6.22

For any k, there exists a d such that every d-regular directed graph has a directed cycle of length divisible by k.

Here, d-regular means that each vertex has d outgoing and d incoming edges, and a directed cycle is a cycle with all arrows pointing correctly. Also, note that if we have a (connected) $2d$-regular undirected graph, then we can find an Eulerian tour (since all degrees are even). So we can direct an undirected graph. If we have odd degree, we can just add a vertex (left as an exercise). Either way, this means that we can prove the undirected version of this theorem as well! We'll actually prove something stronger:

Theorem 6.23

Every directed graph with minimum out-degree δ and maximum in-degree Δ contains a cycle of length divisible by a constant k if

$$k \leq \frac{\delta}{1 + \log(1 + \delta\Delta)}.$$

Proof. We'll simplify the problem: we can delete some edges and assume that all outdegrees are exactly δ. Assign every element a uniform random element of $\mathbb{Z}/k\mathbb{Z}$: only look at those vertices where the label increases by 1 mod k along the arrow. In other words, we can split up the vertices into k groups, corresponding to the residues. Then if there exists a directed cycle in this new graph, it must have length divisible by k.

What can go wrong? It's bad to go along the cycle and get a dead end, so our goal is for every vertex to have at least 1 outgoing edge that remains. Then we can just keep going, and we must eventually terminate.

So let A_v be the event that v doesn't have any outgoing edges. The probability this happens is

$$\left(\frac{k-1}{k}\right)^{\Delta},$$

since all δ of the outgoing edges must not work. But the dependencies are a bit more subtle: when do we put an edge between A_v and A_w? Define $N^+(v)$ to be the out-neighbors of v and $N^+(w)$ be the out-neighbors of w: then we need to make sure

$$(v \cup N^+(v)) \cup (w \cup (N^+(w))) \neq \varnothing,$$

since the event A_v only depends on v and its out-neighbors. But we can do a little bit better:

Lemma 6.24

A_u is independent of all A_w such that

$$N^+(u) \cap (N^+(w) \cup \{w\}) = \varnothing.$$

Proof of lemma. To show this, notice that given a vertex u with some outgoing edges, if w points to u, A_u is independent of A_w because A_u has the same probability even when we fix the value of A_w. Specifically, if w points to u but not any of its out-neighbors, we can just pick whether w has outgoing edges first: this can't affect u. □

So now we can calculate the degrees of the dependency graph: for a fixed u, the number of w that are adjacent to u in our dependency digraph is at most $\delta\Delta$. Thus the conditions of the local lemma are satisfied as long as

$$ep(\delta\Delta + 1) = e\left(\frac{k-1}{k}\right)^{\delta}(\delta\Delta + 1) \leq e^{1-\delta/k}(\delta\Delta + 1) \leq 1,$$

and rearranging gives the result. □

Fact 6.25

We've often been saying "the dependency graph," but the way we should probably think of it as follows: specify a graph, and then say that our collection of events is consistent with the graph we've created. Basically, there are multiple options for our dependency graph for a single setup.

6.8 Bounds on the linear arboricity conjecture

Lemma 6.26

Let $k \leq d^{0.9}$. Then every d-regular directed graph can be vertex-colored (in any way) using k colors so that every vertex has $\frac{d}{k} + O\left(\sqrt{\frac{d \log d}{k}}\right)$ in-neighbors of each color (and also simultaneously the same number of out-neighbors).

So looking at our graph from every vertex, the graph looks fairly equidistributed.

Proof. Color every vertex uniformly at random. Bad events are of the form "a vertex has the wrong number of in-neighbors or out-neighbors of a color": denote $A^+_{v,c}$ to be the event "v doesn't have the correct number of out-neighbors of some color c," and analogously, let $A^-_{v,c}$ be the analogous bad event for in-neighbors.

The probability that each $A_{v,c}$ occurs is the probability we deviate too much from the mean:

$$\Pr\left(\left|\text{Binomial}\left(d, \frac{1}{k}\right) - \frac{d}{k}\right| > C\sqrt{\frac{d \log d}{k}}\right)$$

We do have to be careful about how we use our Chernoff bound, but we can pick C such that

$$\Pr(A_{v,c}) < \frac{1}{100d^3}$$

(to be safe in our calculations later). Now, when can $A^+_{v,c}$ and $A^+_{w,c'}$ be dependent? This only happens if they're reasonably close to each other in the graph:

$$(v \cup N^+(v)) \cap (w \cup N^+(w)) \neq \varnothing,$$

and the maximum degree that can occur here is $(2d)^2 k$. (Note that this is saying that if we condition on any event of the non-neighbors of v, the probability is still independent of those conditions.) Now by the Lovász Local Lemma, we can check that $ep(d+1) \leq 1$ and we're done. □

Now let's move on to an open problem:

Definition 6.27

A **linear forest** is a disjoint union of paths.

Conjecture 6.28 (Linear arboricity)

The edge-set of every graph with maximum degree Δ can be decomposed into $\lceil \frac{\Delta+1}{2} \rceil$ linear forests.

We can't really do any better than this for any given graph: any path only contributes 2 to the degree, so we need at least $\frac{\Delta}{2}$ forests. Notice that every graph of maximal degree Δ is a subgraph of a Δ-regular graph, so it suffices to consider Δ-regular graphs.

For a long time, there was a constant factor gap between the best-known bounds, until Alon showed in 1988 that $\frac{\Delta}{2} + o(\Delta)$ linear forests will work. Alon and Spencer found in 1992 that we can have $\frac{\Delta}{2} + \tilde{O}(n^{2/3})$ (which means there are some other log terms that are neglected). Finally, last year, Ferber, Fox, and Jain proved $\frac{\Delta}{2} + O(\Delta^{2/3-\alpha})$.

There's also a directed version of this conjecture:

Conjecture 6.29 (Directed linear arboricity or DLAC)
The edge-set of every Δ-regular directed graph can be decomposed into $\Delta + 1$ directed linear forests.

This implies the undirected version: take G to be a $2d$-regular graph, and now use an Eulerian tour to orient and get a d-regular digraph. If G starts as an odd-degree regular graph, we can just modify it a bit.

Also, we need at least $\Delta + 1$ directed linear forests for the same reason: every vertex can only contribute one to each vertex's indegree and outdegree, but we can't get 1 on everything (because of the endpoints of our paths).

Let's first outline some of the key steps before jumping into a somewhat weaker bound for DLAC. First, we show that the result is true if the girth of the graph is at least $8e\Delta$. The idea there is that (using Hall's theorem) we can decompose this graph into cycles. This gives Δ subgraphs, each of which is a disjoint union of cycles. To turn this into a covering of linear forests, we can just cut out an edge from each cycle, and our goal is to make sure the remaining edges form a directed linear forest as well.

After we prove this version with large girth, we can divide into subgraphs with large girth. That's also something we did: we found how to produce cycles with length divisible by k, and if $k \geq 8e\Delta$, we can get cycles of long length.

In our proof, we will use a few graph theory results:

Lemma 6.30 (A consequence of Hall's theorem)
A d-regular bipartite graph has a perfect matching.

Notice that removing a perfect matching from a d-regular bipartite graph gives a $(d-1)$-regular bipartite graph.

Corollary 6.31 (Konig's theorem)
Every d-regular bipartite graph can be decomposed into d matchings.

We can then use Konig's theorem to prove the following lemma:

Lemma 6.32
The edge-set of every Δ-regular digraph can be decomposed into Δ 1-regular spanning subgraphs.

For a directed graph, a 1-regular spanning subgraph is a collection of directed cycles using all vertices exactly once.

Proof. Construct two copies of the original vertex set. For each edge going from v_i to v_j, draw an edge from v_i in the first copy to v_j in the second copy: this gives a Δ-regular bipartite graph. Applying Konig, we get Δ perfect matchings: collapse each matching back to the original graph, and it becomes a collection of 1-regular directed graphs (think of v_i going to v_j as a permutation of the vertices), as desired. □

Lemma 6.33 (An easy bound on the DLAC)
The edgeset of every directed Δ-regular graph can be decomposed into at most 2Δ directed linear forests.

Proof. Use the lemma above to get Δ 1-regular spanning subgraphs: for each of these, split the cycles into two paths arbitrarily, and each 1-regular spanning subgraph becomes 2 directed linear forests. □

Theorem 6.34 (DLAC for large directed girth)
The directed linear arboricity conjecture is true if the directed girth is at least $8e\Delta$.

Proof. We can decompose the edgeset into Δ 1-regular spanning subgraphs, $F_1, F_2, \cdots, F_\Delta$, and now each F_i is a disjoint union of cycles. Our goal is to show that we can break up the cycles in a nice way.

Consider the **line graph** of G, $L(G)$, where the vertices are edges of G and $e_1 \to e_2$ is drawn if e_1 and e_2 are incident (share a vertex). Our goal is to select an independent set from this line graph that hits every cycle in the F_is above, because this would allow us to get $\Delta + 1$ directed linear forests.

By the first lemma, we know there exists a matching $M \subset E(G)$ that contains an edge from each cycle, since we have partitioned our graph into cycles have length at most $8e\Delta$, and each vertex in our line graph has degree at most 4Δ. Take $F_1 \setminus M, F_2 \setminus M, \cdots$: we've now broken up the cycles, so each of those is a linear forest. Thus M plus these gives the $\Delta + 1$ directed linear forests, as we want. □

Now we're ready to prove the main result of this section:

Theorem 6.35 (A better bound for DLAC)
Every Δ-regular directed graph can be decomposed into at most $\Delta + \tilde{O}(\Delta^{3/4})$ (possibly contains poly-log factors) linear forests.

Our goal is to produce a lot of cycles with long length. One thing we can do, as last time, is to label our vertices mod k, and make sure all cycles go from vertices i to $i + 1$.

Proof. Pick a prime k in the range $\left[10\sqrt{\Delta}, 20\sqrt{\Delta}\right]$ (this exists). By the second lemma, there exists a coloring of the vertex set by elements of $\mathbb{Z}/k\mathbb{Z}$, so that every vertex has $\frac{\Delta}{k} + \tilde{O}\left(\sqrt{\frac{\Delta}{k}}\right)$ neighbors of each element.

For each i, let D_i be the subgraph of edges whose color increases by i mod k from the start to end point. We know that Δ_i, the maximum degree of D_i, is at most $\frac{\Delta}{k} + \tilde{O}\left(\sqrt{\frac{\Delta}{k}}\right)$. For all $i \neq 0$, D_i has girth at least $k \geq 8e\Delta_i$ (this is where we need $k \gtrsim \sqrt{\Delta}$), so the DLAC for large girth tells us that we can decompose each D_i for nonzero i using $\Delta_i + 1$ linear forests. For D_0, use the easy bound of twice the degree from Lemma 6.33. This means the total number of linear forests is

$$\leq \left(\frac{\Delta}{k} + \tilde{O}\left(\sqrt{\frac{\Delta}{k}}\right)\right)(k+1) \leq \Delta + \tilde{O}(\Delta^{3/4}),$$

as desired. □

Notice that we needed k to be prime to ensure that all cycles go through all k colors.

How can we improve this? Notice that we can decompose K_4 into Hamiltonian paths: in general, we can decompose any K_{2n}. If we have $2n$ colors in our proof above, consider all the edges between the color groups: decompose into Δ_1 matchings for each "connection" between color groups, and this means we can decompose into Δ_1 paths, minus some edges. Applying the easy bound to the edges within the color groups again, this gives k Hamiltonian paths, and our bound is now

$$\leq \left(\frac{\Delta}{k} + \tilde{O}\left(\sqrt{\frac{\Delta}{k}}\right)\right)k + \frac{2\Delta}{k} + \tilde{O}\left(\sqrt{\frac{\Delta}{k}}\right)$$

and optimizing for the correct value of k, we minimize this at

$$\leq \Delta + \tilde{O}(n^{2/3}).$$

This was nice because we can pick a smaller value of k, and we don't have the girth requirement anymore!

6.9 The lopsided local lemma

Remember in the proof of the local lemma, we made a bound of the form

$$\text{numerator} \le \Pr\left(A_i \mid \bigwedge_{j \in S_2} \overline{A_j}\right),$$

where S_2 is elements outside of $N(i)$ [not connected]. By the definition of the dependency graph, this is just **equal to** $\Pr(A_i) \le x_i \prod_{j \in N(i)}(1 - x_i)$. This was the only place we used the independence hypothesis!

What if we instead just assumed an inequality in that equality? In other words, what if we always just had positive dependence, so bad events actually make each other less likely to occur?

Here's the formal setup. Let's say we have some bad events $A_1, \cdots, A_n$. The **negative dependence digraph** is one where (if $N(i)$ is the outneighbors of i), we have

$$\Pr\left(A_i \mid \bigwedge_{j \in S} \overline{A_j}\right) \le \Pr(A_i) \forall i, S \subset N(i)^C.$$

So this graph records potential negative dependence: $N(i)$ notes the bad events that can potentially be worrying for us (since the events are more likely to occur separately), and everything else either does nothing or actually helps us (because it means the events have positive dependence). This was called the **lopsidependency graph**.

Theorem 6.36 (Lopsided Lovász Local Lemma)
If there exist real numbers $x_1, \cdots, x_n \in [0, 1)$ such that

$$\Pr(A_i) \le x_i \prod_{j \in N(i)} (1 - x_j) \quad \text{for all } i \in [n],$$

then with probability at least $\prod_{i=1}^{n}(1 - x_i)$, no event A_i occurs.

The proof is exactly the same. Just change the boxes equals to an inequality. In fact, we can replace the $\le \Pr(A_i)$ condition with $\le x_i \prod_{j \in N(i)}(1 - x_j)$.

There's also a symmetric version, which is easier to use: if the negative dependency (di)graph has all neighborhoods size with size at most d, and $ep(d + 1) \le 1$, then with positive probability, no bad event occurs.

We showed using LLL that every k-uniform hypergraph is 2-colorable if every edge intersects at most $\frac{2^{k-1}}{e} - 1$ other edges: color all the vertices uniformly at random. What if we refine that argument a bit?

Proposition 6.37
Every k-uniform hypergraph is 2-colorable if every edge intersects at most $\frac{2^k}{e} - 2$ other edges.

Solution. For each edge f, we defined the bad event A_f if f was monochromatic, but now, let's let $A_{f,c}$ be the event that all vertices in f are colored with color c. It is clear that $\Pr(A_{f,c}) = 2^{-k}$.

Let our colors be 0 and 1. Consider the graph where $A_{f,c}$ and $A_{f',1-c}$ are adjacent if f and f' intersect. Edges of our dependency graph then come from two intersecting edges with opposite colors: we claim this is a negative dependency graph. The intuition is that compared to the vertex overlap graph, this one tells us a little more: if we

have an edge f, and we want to know the probability of f being colored all red, it's more likely if we know f' is not colored all blue.

So if the degree of the negative dependency graph is at most $d+1$, then $ep(d+2) \leq 1$ as long as $d \leq \frac{2^k}{e} - 2$, and applying the Lopsided Lovász Local Lemma yields the result. □

6.10 Latin squares

Let's now do an example where the lopsidedness matters:

Definition 6.38

A **Latin square** of order n is an $n \times n$ square filled with n symbols (usually 1 through n), such that every symbol appears exactly once in each row and column.

A **transversal** of an $n \times n$ array is a set of n entries with one per row and one per column. A **Latin transversal** has distinct entries in the Latin square.

Conjecture 6.39

All odd order Latin squares have a Latin transversal.

This is still open, so we're not going to prove it. Instead, we'll show the following weakening of that result:

Theorem 6.40 (Erdős, Spencer)

Every $n \times n$ array where every entry appears at most $\frac{n}{4e}$ times has a Latin transversal.

There's an open problem that every odd-n Latin square has a Latin transversal: this is a looser restriction of that. (It's not necessarily true for even n, but the conjecture is that we can always find $(n-1)$ different entries in that case in a "rook placement.")

Remark (Historical note)**.** *These objects are called Latin squares, because Euler started playing with them and wrote a paper where they had Latin entries instead of numbers.*

It's not really clear what the bad events can be here, but the point is that different permutations seem to involve every row and every column, so there's not that many disjoint supported variables. That makes it hard to apply the vanilla LLL. But some of the dependencies only help us, so we can apply the lopsided version.

Proof. Pick a transversal uniformly at random: this is equivalent to picking one of $n!$ permutations uniformly at random. Then we can have our bad events be of the form A_{ijkl}, where (i,j) and (k,l) are both picked, and they have the same entry written in them. (Note that the second part of this condition is not random: it is already determined by our initial array.) If we avoid all such bad events, we get a Latin transversal.

The probability of $0A_{ijkl}$ is 0 if the two squares (i,j) and (k,l) are in the same row or column, since we can't pick them both in our Latin transversal. For all others, the probability is $\frac{1}{n(n-1)}$, since there are $(n-2)!$ permutations that go through these two points.

If we try to do a dependency graph problem like the Lovász Local Lemma, we run into a problem: most bad events are dependent on each other, but what about the negative dependency graph?

Let G be the graph (V, E), where

- V consists of unordered pairs $\{(i,j),(k,l)\}$ that have the same entry and aren't in the same row or column.
- $\{(i,j),(k,l)\}$ and $\{(i',j'),(k',l')\}$ are connected by an edge if and only if there are any rows or columns that are shared: $\{i,k\} \cap \{i',k'\} \neq \varnothing$ or $\{j,l\} \cap \{j',l'\} \neq \varnothing$.

If we can show this is a negative dependency graph, then we can finish in the following way: given any pair of vertices, another pair is in the negative dependency graph if we pick one of the $(4n-4)$ squares in the same row or column, and then pick one of the $\frac{n}{4e}-1$ other squares with the same entry. Then the maximum degree is at most $\frac{n(n-1)}{e}-1$, and therefore $ep(d+1) \leq 1$, meaning we can apply the Lopsided Lovász Local Lemma.

So to show that we do have a negative dependency graph G, fix some i,j,k,l. Let J be a subset of the nonneighbors of the bad event $\{(i,j),(k,l)\}$. Our goal is to show that

$$\Pr\left(A_{ijkl} \mid \bigwedge_{i'j'k'l' \in J} \overline{A_{i'j'k'l'}}\right) \leq \Pr(A_{ijkl}).$$

Without loss of generality, we can assume $\{(i,j),(k,l)\} = (1,1,2,2)$ by permuting rows and columns. We need to show that the probability that the transversal goes through $(1,1)$ and $(2,2)$ is at least as large as it is conditioned on not going through any of the other events $A_{i'j'k'l'}$.

So we're not conditioning on any information through the first two columns or rows. How many transversals are there satisfying these conditions? Set $\mathcal{S}_{r,s}$ to be the number of transversals through $(1,r),(2,s)$ that avoid all bad events in J. Note that the $\mathcal{S}_{r,s}$s over distinct $r,s \in [n]$ partition $\bigwedge_{i'j'k'l'} \overline{A_{i'j'k'l'}}$ into $n(n-1)$ subsets. Our goal is to show that $|\mathcal{S}_{1,2}| \leq |\mathcal{S}_{r,s}|$ for all r,s: if we show this, it would follow that

$$\Pr\left(A_{ijkl} \mid \bigwedge_{i'j'k'l' \in J} \overline{A_{i'j'k'l'}}\right) = \frac{|\mathcal{S}_{1,2}|}{\sum_{r \neq s} |\mathcal{S}_{r,s}|} \leq \frac{1}{n(n-1)}.$$

So we're saying that not having any bad events in rows 3 through n only helps us in the $1,2$ case. To do this, we'll construct an injection: $\mathcal{S}_{1,2} \to \mathcal{S}_{r,s}$ by swapping the 1st and rth row, as well as the 2nd and sth row. This can't cause any bad events to occur, because all potential bad events occur in the bottom $(n-2)$ by $(n-2)$ square! This finishes the proof. □

Notice that the hardest part of the proof is finding the right bad events and dependency graph to work with: the rest is fairly straightforward.

18.218 Probabilistic Method in Combinatorics

Spring 2019