6 Lovász Local Lemma

The Lovász local lemma (LLL) was introduced in the paper of Erdős and Lovász (1975). It is a powerful tool in the probabilistic method.

In many problems, we wish to avoid a certain set of "bad events." Here are two easy to handle scenarios:

- (Complete independence) All the bad events are independent and have probability less than 1.
- (Union bound) The sum of the bad event probabilities is less than 1.

The local lemma deals with an intermediate situation where there is a small amount of local dependencies.

We saw an application of the Lovász local lemma back in Section 1.1, where we used it to lower bound Ramsey numbers. This chapter explores the local lemma and its applications in depth.

6.1 Statement and proof

Definition 6.1.1 (Independence from a set of events)

Here we say that an event A_0 is ***independent*** from events $A_1, \dots, A_m$ if A_0 is independent of every event of the form $B_1 \wedge \cdots \wedge B_m$ (we sometimes omit the "logical and" symbol $\wedge$) where each B_i is either A_i or $\overline{A_i}$, i.e.,

$$\mathbb{P}(A_0 B_1 \cdots B_m) = \mathbb{P}(A_0)\mathbb{P}(B_1 \cdots B_m),$$

or, equivalently, using Bayes's rule:

$$\mathbb{P}(A_0 | B_1 \cdots B_m) = \mathbb{P}(A_0).$$

Given a collection of events, we can associate to it a dependency graph. This is a slightly subtle notion, as we will explain. Technically speaking, the graph can be a directed graph (=digraph), but for most applications, it will be sufficient (and easier) to use undirected graphs.

Definition 6.1.2 (Dependency (di)graph)
Let $A_1, \dots, A_n$ be events (the "bad events" we wish to avoid). Let G be a (directed) graph with vertex set $[n]$. We say that G is a ***dependency (di)graph*** for the events $A_1, \dots, A_n$ if, for for every i, A_i is independent from all $\{A_j : j \notin N(i) \cup \{i\}\}$ ($N(i)$ is the set of (out)neighbors of i in G).

***Remark* 6.1.3** (Non-uniqueness)*.* Given a collection of events, there can be more than one valid dependency graphs. For example, the complete graph is always a valid dependency graph.

***Remark* 6.1.4** (Important!)*.* **Independence ≠ pairwise independence**
The dependency graph is *not* made by joining $i \sim j$ whenever A_i and A_j are not independent (i.e., $\mathbb{P}(A_i A_j) \neq \mathbb{P}(A_i)\mathbb{P}(A_j)$).

Example: suppose one picks $x_1, x_2, x_3 \in \mathbb{Z}/2\mathbb{Z}$ uniformly and independently at random and set, for each $i = 1, 2, 3$ (indices taken mod 3), A_i the event that $x_{i+1} + x_{i+2} = 0$. Then these events are pairwise independent but not independent. So the empty graph on three vertices is not a valid dependency graph (on the other hand, having at least two edges makes it a valid dependency graph).

In practice, it is not too hard to construct a valid dependency graph, since most applications of the Lovász local lemma use the following setup (which we saw in Section 1.1).

Setup 6.1.5 (Random variable model / hypergraph coloring)
Let $\{x_i : i \in I\}$ be a collection of independent random variables. Let $E_1, \dots, E_n$ be events where each E_i depends only on the variables indexed by some subset $B_i \subseteq I$ of variables. A ***canonical dependency graph*** for the events $E_1, \dots, E_n$ has vertex set $[n]$ and an edge ij whenever $B_i \cap B_j \neq \emptyset$.

It is easy to check that the canonical dependency graph above is indeed a valid dependency graph.

Example* 6.1.6** (Boolean satisfiability problem (SAT))*.* Given a ***CNF formula (conjunctive normal form, i.e., *and-of-or*'s), e.g., ($\wedge$ = and; $\vee$ = or)

$$(x_1 \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee x_2 \vee x_4) \wedge (\overline{x_2} \vee x_4 \vee x_5) \wedge \cdots$$

the problem is to find a satisfying assignment with boolean variables $x_1, x_2, \dots$. Many problems in computer science can be modeled using this way. This problem can be viewed as in Setup 6.1.5, where A_i is the event that the i-th clause is violated.

The following formulation of the local lemma is easiest to apply and is the most commonly used. It applies to settings where the dependency graph has small maximum degree.

Theorem 6.1.7 (Lovász local lemma; symmetric form)
Let $A_1, \ldots, A_n$ be events, with $\mathbb{P}[A_i] \le p$ for all i. Suppose that each A_i is independent from a set of all other A_j except for at most d of them. If

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

then with some positive probability, none of the events A_i occur.

***Remark* 6.1.8.** The constant e is best possible (Shearer 1985). In most applications, the precise value of the constant is unimportant.

Theorem 6.1.9 (Lovász local lemma; general form)
Let $A_1, \ldots, A_n$ be events. For each $i \in [n]$, let $N(i)$ be such that A_i is independent from $\{A_j : j \notin \{i\} \cup N(i)\}$. If $x_1, \ldots, x_n \in [0,1)$ satisfy

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

then

$$\mathbb{P}(\text{none of the events } A_i \text{ occur}) \ge \prod_{i=1}^{n} (1 - x_i).$$

Proof that the general form implies the symmetric form. Set $x_i = 1/(d+1) < 1$ for all i. Then

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

so the hypothesis of general local lemma holds. □

Here is another corollary of the general form local lemma, which applies if the total probability of any neighborhood in a dependency graph is small.

Corollary 6.1.10
In the setup of Theorem 6.1.9, if $\mathbb{P}(A_i) < 1/2$ and $\sum_{j \in N(i)} \mathbb{P}(A_j) \le 1/4$ for all i, then with positive probability none of the events A_i occur.

Proof. In Theorem 6.1.9, set $x_i = 2\mathbb{P}(A_i)$ for each i. Then

$$x_i \prod_{j\in N(i)} (1-x_j) \geq x_i \left(1 - \sum_{j\in N(i)} x_j\right) = 2\mathbb{P}(A_i)\left(1 - \sum_{j\in N(i)} 2\mathbb{P}(A_i)\right) \geq \mathbb{P}(A_i).$$

(The first inequality is by "union bound.") □

In some applications, one may need to apply the general form local lemma with carefully chosen values for x_i.

Proof of Lovász local lemma (general case). We will prove that

$$\mathbb{P}\left(A_i \,\middle|\, \bigwedge_{j\in S} \overline{A}_j\right) \leq x_i \quad \text{whenever } i \notin S \subseteq [n]. \tag{6.1}$$

Once (6.1) has been established, we then deduce that

$$\begin{aligned}\mathbb{P}(\overline{A}_1 \cdots \overline{A}_n) &= \mathbb{P}(\overline{A}_1)\mathbb{P}\left(\overline{A}_2 \,\middle|\, \overline{A}_1\right)\mathbb{P}\left(\overline{A}_3 \,\middle|\, \overline{A}_1\overline{A}_2\right)\cdots\mathbb{P}\left(\overline{A}_n \,\middle|\, \overline{A}_1\cdots\overline{A}_{n-1}\right)\\ &\geq (1-x_1)(1-x_2)\cdots(1-x_n),\end{aligned}$$

which is the conclusion of the local lemma.

Now we prove (6.1) by induction on $|S|$. The base case $|S| = 0$ is trivial.

Let $i \notin S$. Let $S_1 = S \cap N(i)$ and $S_2 = S \setminus S_1$. We have

$$\mathbb{P}\left(A_i \,\middle|\, \bigwedge_{j\in S} \overline{A}_j\right) = \frac{\mathbb{P}\left(A_i \bigwedge_{j\in S_1}\overline{A}_j \,\middle|\, \bigwedge_{j\in S_2}\overline{A}_j\right)}{\mathbb{P}\left(\bigwedge_{j\in S_1}\overline{A}_j \,\middle|\, \bigwedge_{j\in S_2}\overline{A}_j\right)} \tag{6.2}$$

For the RHS of (6.2), using that A_i is independent of $\left\{j \in S_2 : A_j\right\}$,

$$\text{numerator} \leq \mathbb{P}\left(A_i \,\middle|\, \bigwedge_{j\in S_2} \overline{A}_j\right) = \mathbb{P}(A_i) \leq x_i \prod_{j\in N(i)} (1-x_i), \tag{6.3}$$

and, denoting the elements of S_1 by $S_1 = \{j_1, \dots, j_r\}$,

$$\begin{aligned}\text{denominator} &= \mathbb{P}\left(\overline{A}_{j_1} \,\middle|\, \bigwedge_{j\in S_2} \overline{A}_j\right)\mathbb{P}\left(\overline{A}_{j_2} \,\middle|\, \overline{A}_{j_1} \bigwedge_{j\in S_2} \overline{A}_j\right)\cdots\mathbb{P}\left(\overline{A}_{j_r} \,\middle|\, \overline{A}_{j_1}\cdots\overline{A}_{j_{r-1}} \bigwedge_{j\in S_2} \overline{A}_j\right)\\ &\geq (1-x_{j_1})\cdots(1-x_{j_r}) \qquad \text{[by induction hypothesis]}\\ &\geq \prod_{j\in N(i)}(1-x_i)\end{aligned}$$

Thus (6.2) $\leq x_i$, thereby finishing the induction proof of (6.1). □

***Remark* 6.1.11.** We used the independence assumption only at step (6.3) of the proof. Upon a closer examination, we see that we only need to know correlation inequalities of the form $\mathbb{P}\left(A_i \,\middle|\, \bigwedge_{j\in S_2} \overline{A}_j\right) \leq \mathbb{P}(A_i)$ for $S_2 \subseteq N(i)$, rather than independence. This observation allows a strengthening of the local lemma, known as a lopsided local lemma, that we will explore later in the chapter.

6.2 Coloring hypergraphs

Previously, in Theorem 1.3.1, we saw that every k-uniform hypergraph with fewer than 2^{k-1} edges is 2-colorable. The next theorem gives a sufficient local condition for 2-colorability.

Theorem 6.2.1

A k-uniform hypergraph is 2-colorable if every edge intersects at most $e^{-1}2^{k-1} - 1$ other edges

Proof. For each edge f, let A_f be the event that f is monochromatic. Then $\mathbb{P}(A_f) = p := 2^{-k+1}$. Each A_f is independent from all $A_{f'}$ where f' is disjoint from f. Since at most $d := e^{-1}2^{k-1} - 1$ edges intersect every edge, and $e(d+1)p \leq 1$, so the local lemma implies that with positive probability, none of the events A_f occur. □

Corollary 6.2.2

For $k \geq 9$, every k-uniform k-regular hypergraph is 2-colorable.
(Here *k-regular* means that every vertex lies in exactly k edges.)

Proof. Every edge intersects $\leq d = k(k-1)$ other edges. And $e(k(k-1)+1)2^{-k+1} < 1$ for $k \geq 9$. □

***Remark* 6.2.3.** The statement is false for $k = 2$ (triangle) and $k = 3$ (Fano plane) but actually true for all $k \geq 4$ (Thomassen 1992).

Here is an example where the symmetric form of the local lemma is insufficient (why?).

Theorem 6.2.4

Let H be a (non-uniform) hypergraph where every edge has size ≥ 3. Suppose

$$\sum_{f \in E(H) \setminus \{e\} : e \cap f \neq \emptyset} 2^{-|f|} \leq \frac{1}{8}, \qquad \text{for each edge } e,$$

then H is 2-colorable.

Proof. Consider a uniform random 2-coloring of the vertices. Let A_e be the event that edge e is monochromatic. Then $\mathbb{P}(A_e) = 2^{-|e|+1} \leq 1/4$ since $|e| \geq 3$. Also,

$$\sum_{f \in E(H) \setminus \{e\} : e \cap f \neq \emptyset} \mathbb{P}(A_f) = \sum_{f \in E(H) \setminus \{e\} : e \cap f \neq \emptyset} 2^{-|f|+1} \leq 1/4.$$

Thus by Corollary 6.1.10 one can avoid all events A_e, and hence H is 2-colorable. □

***Remark* 6.2.5.** A sign to look beyond the symmetric local lemma is when there are bad events of very different nature (e.g., having very different probabilities).

Compactness argument

Now we highlight an important ***compactness argument*** that allows us to deduce the existence of an infinite object, even though the local lemma itself is only applicable to finite systems.

Theorem 6.2.6

Let H be a (non-uniform) hypergraph on a possibly infinite vertex set, such that each edge is finite, has at least k vertices, and intersect at most d other edges. If $e2^{-k+1}(d+1) \leq 1$, then H has a proper 2-coloring.

Proof. From a vanilla application of the symmetric local lemma, we deduce that for any finite subset X of vertices, there exists an 2-coloring X so that no edge contained in X is monochromatic (color each vertex iid uniformly, and consider the bad event A_e that the edge $e \subseteq X$ is monochromatic).

Next we extend the coloring to the entire vertex set V by a compactness argument. The set of all colorings is $[2]^V$. By Tikhonov's theorem (which says a product of a possibly infinite collection of compact topological spaces is compact), $[2]^V$ is compact under the product topology.

For each finite subset X, let $C_X \subseteq [2]^V$ be the subset of colorings where no edge contained in X is monochromatic. Earlier from the local lemma we saw that $C_X \neq \emptyset$. If $Y \subseteq X$, then $C_Y \supseteq C_X$. Thus

$$C_{X_1} \cap \cdots \cap C_{X_\ell} \supseteq C_{X_1 \cup \cdots \cup X_\ell},$$

so $\{C_X : |X| < \infty\}$ is a collection of closed subsets of $[2]^V$ with the finite intersection property (i.e., the intersection of any finite subcollection is nonempty).

Recall from point-set topology the following basic fact (a defining property): a space is compact if and only if every family of closed subsets having the finite intersection property has non-empty intersection.

Hence by compactness of $[2]^V$, the intersection of C_X taken over all finite X is non-empty. Any element of this intersection corresponds to a valid coloring of the hypergraph. □

More generally, the above compactness argument yields the following.

Lemma 6.2.7 (Compactness argument)
Consider a variation of the random variable model (Setup 6.1.5) where each variable has only finitely many choices but there can be possibly infinitely many events (each event depends on a finite subset of variables). If it is possible to avoid any finite subset of events, then it is possible to avoid all the events. □

***Remark* 6.2.8.** Note the conclusion may be false if we do not assume the random variable model (why?).

The next application appears in the paper of Erdős and Lovász (1975) where the local lemma originally appears.

Consider k-coloring the real numbers, i.e., a function $c\colon \mathbb{R} \to [k]$. We say that $T \subseteq \mathbb{R}$ is ***multicolored*** with respect to c if all k colors appear in T.

Question 6.2.9
For each k is there an m so that for every $S \subseteq \mathbb{R}$ with $|S| = m$, one can k-color $\mathbb{R}$ so that every translate of S is multicolored?

The following theorem shows that this can be done whenever $m > (3+\varepsilon)k \log k$ and $k > k_0(\varepsilon)$ sufficiently large.

Theorem 6.2.10

The answer to the above equation is yes if

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

Proof. Each translate of S is not multicolored with probability $p \le k(1-1/k)^m$, and each translate of S intersects at most $m(m-1)$ other translates. Consider a bad event for each translate of S contained in X. The symmetric local lemma tells us that it is possible to avoid any finite collection of bad events. By the compactness argument, it is possible to avoid all the bad events. □

Coloring arithmetic progressions

Here is an application where we need to apply the asymmetric local lemma.

Theorem 6.2.11 (Beck 1980)

For every $\varepsilon > 0$, there exists k_0 and a 2-coloring of $\mathbb{Z}$ with no monochromatic k-term arithmetic progressions with $k \ge k_0$ and common difference less than $2^{(1-\varepsilon)k}$.

Proof. We pick a uniform random color for each element of $\mathbb{Z}$. For each k-term arithmetic progression in $\mathbb{Z}$ with $k \ge k_0$ and common difference less than $2^{(1-\varepsilon)k}$, consider the event that this k-AP is monochromatic. By the compactness argument, it suffices to check that we can avoid any finite subset of events.

The event that a particular k-AP is monochromatic has probability exactly 2^{-k+1}. (Since this number depends on k, we should use the asymmetric local lemma.)

Recall that in the asymmetric local lemma (Theorem 6.1.9), we need to select $x_i \in [0,1)$ for each bad event A_i so that

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

It is usually a good idea to select x_i to be somewhat similar to $\mathbb{P}(A_i)$. In this case, if A_i is the event corresponding to a k-AP, then we take

$$x_i = 2^{-(1-\varepsilon/2)k} = \left(\frac{\mathbb{P}(A_i)}{2}\right)^{1-\varepsilon/2}$$

(with the same ε as in the statement of the theorem).

Fix a k-AP P in $\mathbb{Z}$ with $k \geq k_0$. The number of ℓ-APs with $\ell \geq k_0$ and common difference less than $2^{(1-\varepsilon)\ell}$ that intersects P is at most $k\ell 2^{(1-\varepsilon)\ell}$ (one choice for the element of k, a choice of the position of the ℓ-AP, and at most $2^{(1-\varepsilon)\ell}$ choices for the common difference). So to apply the local lemma, it suffices to check that

$$2^{-\varepsilon k/2+1} \leq \prod_{\ell \geq k_0} \left(1 - 2^{-(1-\varepsilon/2)\ell}\right)^{k\ell 2^{(1-\varepsilon)\ell}}.$$

Note that $1 - x \geq e^{-2x}$ for $x \in [0, 1/2]$. So

$$RHS \geq \exp\left(-\sum_{\ell \geq k_0} 2^{1-(1-\varepsilon/2)\ell} \cdot k\ell 2^{(1-\varepsilon)\ell}\right) = \exp\left(-k \sum_{\ell \geq k_0} \ell 2^{1-\varepsilon\ell/2}\right)$$

By making $k_0 = k_0(\varepsilon)$ large enough, we can ensure that $\sum_{\ell \geq k_0} \ell 2^{1-\varepsilon\ell/2} < \varepsilon/4$, and so continuing,

$$\cdots \geq e^{-\varepsilon k/4} \geq 2^{-\varepsilon k/2+1}$$

provided that $k \geq k_0(\varepsilon)$. So we can apply the local lemma to conclude. □

Decomposing coverings

We say that a collection of disks in $\mathbb{R}^d$ is a ***covering*** if their union is $\mathbb{R}^d$. We say that it is a ***k*-*fold covering*** if every point of $\mathbb{R}^d$ is contained in at least k disks (so 1-fold covering is the same as a covering).

We say that a k-fold covering is ***decomposable*** if it can be partitioned into two coverings.

In $\mathbb{R}^d$, is a every k-fold covering by unit balls decomposable if k is sufficiently large?

A fun exercise: in $\mathbb{R}^1$, every k-fold covering by intervals can be partitioned into k coverings.

Mani-Levitska and Pach (1986) showed that every 33-fold covering of $\mathbb{R}^2$ is decomposable.

What about higher dimensions?

Surprising, they also showed that for every k, there exists a k-fold indecomposable covering of $\mathbb{R}^3$ (and similarly for $\mathbb{R}^d$ for $d \geq 3$).

However, it turns out that indecomposable coverings must cover the space quite unevenly:

Theorem 6.2.12 (Mani-Levitska and Pach 1986)

Every k-fold nondecomposable covering of $\mathbb{R}^3$ by open unit balls must cover some point $\gtrsim 2^{k/3}$ times.

***Remark* 6.2.13.** In $\mathbb{R}^d$, the same proof gives $\geq c_d 2^{k/d}$.

We will need the following combinatorial geometric fact:

Lemma 6.2.14

A set of $n \geq 2$ spheres in $\mathbb{R}^3$ cut $\mathbb{R}^3$ into at most n^3 connected components.

Proof. Let us first consider the problem in one dimension lower. Let $f(m)$ be the maximum number of connected regions that m circles on a sphere in $\mathbb{R}^3$ can cut the sphere into.

We have $f(m+1) \leq f(m) + 2m$ for all $m \geq 1$ since adding a new circle to a set of m circles creates at most $2m$ intersection points, so that the new circle is divided into at most $2m$ arcs, and hence its addition creates at most $2m$ new regions.

Combined with $f(1) = 2$, we deduce $f(m) \leq m(m-1) + 2$ for all $m \geq 1$.

Now let $g(m)$ be the maximum number of connected regions that m spheres in $\mathbb{R}^3$ can cut $\mathbb{R}^3$ into. We have $g(1) = 2$, and $g(m+1) \leq g(m) + f(m) \leq g(m)$ by a similar argument as earlier. So $g(m) \leq f(m-1) + f(m-2) + \cdots + f(1) + g(0) \leq m^3$. □

Proof. Suppose for contradiction that every point in $\mathbb{R}^3$ is covered by at most $t \leq c2^{k/3}$ unit balls from F (for some sufficiently small c that we will pick later).

Construct an infinite hypergraph H with vertex set being the set of balls and edges having the form $E_x = \{\text{balls containing } x\}$ for some $x \in \mathbb{R}^3$. Note that $|E_x| \geq k$ since we have a k-fold covering.

Also, note that if $x, y \in \mathbb{R}^3$ lie in the same connected component in the complement of the union of all the unit spheres, then $E_x = E_y$ (i.e., the same edge).

Claim: every edge of intersects at most $d = O(t^3)$ other edges

Proof of claim: Let $x \in \mathbb{R}^3$. If $E_x \cap E_y \neq \emptyset$, then $|x - y| \leq 2$, so all the balls in E_y lie in the radius-4 ball centered at x. The volume of the radius-4 ball is 4^3 times the unit ball. Since every point lies in at most t balls, there are at most $4^3 t$ balls appearing among those E_y intersecting x, and these balls cut the radius-2 centered at x into $O(t^3)$ connected regions by the earlier lemma, and two different y's in the same region produce the same E_y. So E_x intersects $O(t^3)$ other edges. ■

With $t \leq c2^{k/3}$ and c sufficiently small, and knowing $d = O(t^3)$ from the claim, we have $e2^{-k+1}(d+1) \leq 1$. It then follows by Theorem 6.2.6 (local lemma + compactness argument) that this hypergraph is 2-colorable, which corresponds to a decomposition of the covering, a contradiction. □

6.3 Independent transversal

The application of the local lemma in this section is instructive in that it is not obvious at first what to choose as bad events (even if you are already told to apply the local lemma). It is worth trying different possibilities.

Every graph with maximum degree Δ contains an independent set of size $\geq |V|/(\Delta+1)$ (choose the independent set greedily). The following lemma shows that by decreasing the desired size of the independent set by a constant factor, we can guarantee an independent set that is also a transversal to a vertex set partition.

Theorem 6.3.1

Let $G = (V, E)$ be a graph with maximum degree Δ and let $V = V_1 \cup \cdots \cup V_r$ be a partition, where each $|V_i| \geq 2e\Delta$. Then there is an independent set in G containing one vertex from each V_i.

Proof. The first step in the proof is simple yet subtle: we may assume that $|V_i| = k := \lceil 2e\Delta \rceil$ for each i, or else we can remove some vertices from V_i (if we do not trim the vertex sets now, we will run into difficulties later).

Pick $v_i \in V_i$ uniformly at random, independently for each i.

This is an instance of the random variable model (Setup 6.1.5), where $v_1, \ldots, v_r$ are the random variables.

We would like to design a collection of "bad events" so that if we avoid all of them, then $\{v_1, \ldots, v_r\}$ is guaranteed to be independent set.

What do we choose as bad events? It turns out that some choices work better than others.

Attempt 1:

For each $1 \leq i < j \leq r$ where there exists an edge between V_i and V_j, let $A_{i,j}$ be the event that v_i is adjacent to v_j.

We find that $\mathbb{P}(A_{i,j}) \leq \Delta/k$.

The canonical dependency graph has $A_{i,j} \sim A_{i',j'}$ if and only if the two sets $\{i, j\}$ and $\{i', j'\}$ intersect. This dependency graph has max degree $\leq 2\Delta k$ (starting from (i, j), look at the neighbors of all vertices in $V_i \cup V_j$). The max degree is too large compared to the bad event probabilities.

Attempt 2:

For each edge $e \in E$, let A_e be the event that both endpoints of e are picked.

We have $\mathbb{P}(A_e) = 1/k^2$.

The canonical dependency graph has $A_e \sim A_f$ if some V_i intersects both e and f.

This dependency graph has max degree $\le 2k\Delta$ (if e is between V_i and V_j, then f must be incident to $V_i \cup V_j$).

We have $e(1/k^2)(2k\Delta + 1) \le 1$, so the local lemma implies the with probability no bad event occurs, in which case $\{v_1, \dots, v_r\}$ is an independent set. □

Remark 6.3.2. Alon (1988) introduced the above result as lemma in his near resolution of the still-open *linear arboricity conjecture* (see the Alon–Spencer textbook §5.5). Alon's approach makes heavy use of the local lemma.

Haxell (1995, 2001) relaxed the hypothesis to $|V_i| \ge 2\Delta$ for each i. The statement becomes false if 2Δ is replaced by $2\Delta - 1$ (Szabó and Tardos 2006).

6.4 Directed cycles of length divisible by k

A directed graph is ***d*-regular** if every vertex has in-degree d and out-degree d.

Theorem 6.4.1 (Alon and Linial 1989)

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

Corollary 6.4.2

For every k there exists d so that every $2d$-regular graph has a cycle of length divisible by k.

Proof. Every $2d$-regular graph can be made into a d-regular digraph by orientating its edges according to an Eulerian tour. And then we can apply the previous theorem. □

We will prove the following more general statement.

Theorem 6.4.3 (Alon and Linial 1989)

Every directed graph with min out-degree δ and max in-degree Δ contains a cycle of length divisible by $k \in \mathbb{N}$ as long as

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

Proof. By deleting edges, can assume that every vertex has out-degree exactly δ.

Assign every vertex v an element $x_v \in \mathbb{Z}/k\mathbb{Z}$ iid uniformly at random.

We will look for directed cycles where the labels increase by $1 \pmod k$ at each step. These cycles all have length divisible by k.

For each vertex v, let A_v be the event that there is nowhere to go from v (i.e., if no outneighbor is labeled $x_v + 1 \pmod k$). We have

$$\mathbb{P}(A_v) = (1 - 1/k)^\delta \le e^{-\delta/k}.$$

Since A_v depends only on $\{x_w : w \in \{v\} \cup N^+(v)\}$, where $N^+(v)$ denotes the out-neighbors of v and $N^-(v)$ the in-neighbors of v, the canonical dependency graph has

$A_v \sim A_w$ if $\{v\} \cup N^+(v)$ intersects $\{w\} \cup N^+(w)$.

The maximum degree in the dependency graph is at most $\Delta + \delta\Delta$ (starting from v, there are

(1) at most Δ choices stepping backward

(2) δ choices stepping forward, and

(3) at most $\delta(\Delta - 1)$ choices stepping forward and then backward to land somewhere other than v).

So an application of the local lemma shows that we are done as long as $e^{1-\delta/k}(1+\Delta+\delta\Delta)$, i.e.,

$$k \le \delta/(1 + \log(1 + \Delta + \delta\Delta)).$$

This is almost, but not quite the result (though, for most applications, we would be perfectly happy with such a bound).

The final trick is to notice that we actually have an even smaller valid dependency digraph:

A_v is independent of all A_w where $N^+(v)$ is disjoint from $N^+(w) \cup \{w\}$.

Indeed, even if we fix the colors of all vertices outside $N^+(v)$, the conditional probability that A_v is still $(1 - 1/k)^\delta$.

The number of w such that $N^+(v)$ intersects $N^+(w) \cup \{w\}$ is at most $\delta\Delta$ (no longer need to consider (1) in the previous count). And we have

$$ep(\delta\Delta + 1) \le e^{1-\delta/k}(\delta\Delta + 1) \le 1.$$

So we are done by the local lemma. □

6.5 Lopsided local lemma

Let us move beyond the random variable model, and consider a collection of bad events in the general setup of the local lemma. Instead of requiring that each event is independent of its non-neighbors (in the dependency graph), what if we assume that avoiding some bad events make it easier to avoid some others? Intuitively, it seems that it would only make it easier to avoid bad events.

We can make this notion precise by re-examining the proof of the local lemma. Where did we actually use the independence assumption in the hypothesis of the local lemma? It was in the following step, Equation (6.3):

$$\text{numerator} \le \mathbb{P}\left(A_i \,\middle|\, \bigwedge_{j \in S_2} \overline{A}_j\right) = \mathbb{P}(A_i) \le x_i \prod_{j \in N(i)} (1 - x_i).$$

If we had changed the middle = to ≤, the whole proof would remain valid. This observation allows us to weaken the independence assumption. Therefore we have the following theorem, which was used by Erdős and Spencer (1991) to give an application to Latin transversals that we will see shortly.

Theorem 6.5.1 (Lopsided local lemma)
Let $A_1, \dots, A_n$ be events. For each i, let $N(i) \subseteq [n]$ be such that

$$\mathbb{P}\left(A_i \,\middle|\, \bigwedge_{j \in S} \overline{A}_j\right) \le \mathbb{P}(A_i) \quad \text{for all } i \in [n] \text{ and } S \subseteq [n] \setminus (N(i) \cup \{i\}) \tag{6.1}$$

Suppose there exist $x_1, \dots, x_n \in [0, 1)$ such that

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

Then

$$\mathbb{P}(\text{none of the events } A_i \text{ occur}) \ge \prod_{i=1}^{n} (1 - x_i).$$

Like earlier, by setting $x_i = 1/(d + 1)$, we deduce a symmetric version that is easier to apply.

Corollary 6.5.2 (Lopsided local lemma; symmetric version)
In the previous theorem, if $|N(i)| \le d$ and $\mathbb{P}(A_i) \le p$ for every $i \in [n]$, and $ep(d+1) \le 1$, then with positive probability none of the events A_i occur.

The (di)graph where $N(i)$ is the set of (out-)neighbors of i is called a ***negative dependency (di)graph***.

***Remark* 6.5.3** (Important!)*.* Just as with the usual local lemma, the negative dependency graph is **not** constructed by simply checking pairs of events.

The hypothesis of Theorem 6.5.1 seems annoying to check. Fortunately, many applications of lopsided local lemma fall within a model that we will soon describe, where there is a canonical negative dependency graph that is straightforward to construct. This is analogous to the random variable model for the usual local lemma, where the canonical dependence graph has two events adjacency if they share variables.

Random injection model

We describe a random injection model where there is an easy-to-construct canonical negative dependency graph (Lu and Székely 2007).

Recall that a ***matching*** in a graph is a subset of edges with no two sharing a vertex. In a bipartite graph with vertex parts X and Y, a ***complete matching*** from X to Y is a matching where every vertex of X belongs to an edge of the matching.

Setup 6.5.4 (Random injection model)
Let X and Y be finite sets with $|X| \le |Y|$.

Let $f\colon X \to Y$ be an injection chosen uniformly at random. We can also represent f by a complete matching M from X to Y in $K_{X,Y}$ (the complete bipartite graph between X and Y). We will speak interchangeably of the injection f and matching M.

For a given matching F (not necessarily complete) in $K_{X,Y}$, let A_F denote the event that $F \subseteq M$.

Let $F_1, \dots, F_n$ be matchings in $K_{X,Y}$. The ***canonical negative dependency graph*** for the vents $A_{F_1}, \dots, A_{F_n}$ has one vertex for each event, and an edge between the events A_{F_i} and A_{F_j} $(i \neq j)$ if F_i and F_j are not vertex disjoint.

The following result shows that the above canonical negative dependency graph is a valid for the lopsided local lemma (Theorem 6.5.1).

Theorem 6.5.5 (Nonnegative dependence for random injections)
In Setup 6.5.4, let F_0 be a matching in $K_{X,Y}$ such that F_0 is vertex disjoint from $F_1 \cup \cdots \cup F_k$. Then

$$\mathbb{P}\left(A_{F_0} \,\middle|\, \overline{A}_{F_1} \cdots \overline{A}_{F_k}\right) \le \mathbb{P}(A_{F_0}).$$

Proof. Let $X_0 \subseteq X$ and $Y_0 \subseteq Y$ be the set of endpoints of F_0.

For each matching T in $K_{X,Y}$, let

$$\mathcal{M}_T = \{\text{complete matchings from } X \text{ to } Y \text{ containing } T \text{ but not containing any of } F_1, \dots, F_k\}.$$

For the desired inequality, note that

$$LHS = \mathbb{P}\left(A_{F_0} \,\middle|\, \overline{A}_{F_1} \cdots \overline{A}_{F_k}\right) = \frac{\left|\mathcal{M}_{F_0}\right|}{|\mathcal{M}_\varnothing|} = \frac{\left|\mathcal{M}_{F_0}\right|}{\sum_{T\colon X_0 \hookrightarrow Y} |\mathcal{M}_T|}$$

where the sum is taken over all $|Y|\,(|Y|-1)\cdots(|Y|-|X|+1)$ complete matchings T from X_0 to Y (which we denote by $T\colon X_0 \hookrightarrow Y$), and

$$RHS = \mathbb{P}(A_{F_0}) = \frac{1}{|\{T\colon X_0 \hookrightarrow Y\}|}.$$

Thus to show that $LHS \le RHS$, it suffices to prove

$$\left|\mathcal{M}_{F_0}\right| \le |\mathcal{M}_T| \quad \text{for every } T\colon X_0 \hookrightarrow Y.$$

It suffices to construct an injection $\mathcal{M}_{F_0} \hookrightarrow \mathcal{M}_T$. Let Y_1 be the set of endpoints of T in Y. Fix a permutation σ of Y such that

- σ fixes all elements of Y outside $Y_0 \cup Y_1$; and
- σ sends F_0 to T.

Then σ induces a permutation on the set of complete matchings from X to Y. It remains to show that if we start with a matching in $\mathcal{M}_{F_0}$, so that it avoids F_i for all $i \geq 1$, then it is sent to a matching that also avoids F_i for all $i \geq 1$ (and hence lies in $\mathcal{M}_T$). Indeed, this follows from the hypothesis that none of the edges in F_i use any vertex from X_0 or Y_0. □

As an example, here is a quick application.

Corollary 6.5.6 (Derangement lower bound)

The probability that a uniform random permutation of $[n]$ has no fixed points is at least $(1 - 1/n)^n$.

Proof. In the random injection model, let $X = Y = [n]$. Let $f \colon X \to Y$ be a uniform random permutation. For each $i \in [n]$, let F_i be the single edge (i, i), i.e., A_{F_i} is the even that $f(i) = i$. Note that the canonical negative dependency graph is empty since no two F_i's share a vertex. Since $\mathbb{P}(A_i) = 1 - 1/n$, we can set $x_i = 1 - 1/n$ for each i in the lopsided local lemma to obtain the conclusion

$$\mathbb{P}(f \text{ has no fixed points}) = \mathbb{P}(\overline{A_1} \cdots \overline{A_n}) \geq \left(1 - \frac{1}{n}\right)^n. \qquad \square$$

Remark* 6.5.7.** A fixed-point free permutation is called a ***derangement. Using inclusion-exclusion, one can deduce an exact answer to the above question: $\sum_{i=0}^{n} (-1)^i / i!$. This quantity converges to $1/e$ as $k \to \infty$, and the above lower bound $(1 - 1/n)^n$ also converges to $1/e$ and so is asymptotically optimal.

Latin transversal

A ***Latin square*** of order n is an $n \times n$ array filled with n symbols so that every symbol appears exactly once in every row and column. Example:

$$\begin{array}{ccc} 1 & 2 & 3 \\ 2 & 3 & 1 \\ 3 & 1 & 2 \end{array}$$

These objects are called Latin squares because they were studied by Euler (1707–1783) who used Latin symbols to fill the arrays.

Given an $n \times n$ array, a ***transversal*** is a set of n entries with one in every row and column. A ***Latin transversal*** is a transversal with distinct entries. Example:

$$\begin{array}{ccc} \mathbf{1} & 2 & 3 \\ 2 & \mathbf{3} & 1 \\ 3 & 1 & \mathbf{2} \end{array}$$

Here are is a famous open conjecture about Latin transversals.[1] (Do you see why the hypothesis on parity is necessary?)

Conjecture 6.5.8 (Ryser 1967)

Every odd order Latin square has a transversal.

The conjecture should be modified for even order Latin squares.

Conjecture 6.5.9 (Ryser-Brualdi-Stein conjecture)

Every even order Latin square has a transversal containing all but at most one symbol.

***Remark* 6.5.10.** Keevash, Pokrovskiy, Sudakov and Yepremyan (2022) proved that every order n Latin square contains a transversal containing all but $O(\log n/\log\log n)$ symbols, improving an earlier bound of $O(\log^2 n)$ by Hatami and Shor (2008).

Recently, Montgomery announced a proof of the conjecture for all sufficiently large even n. The proof uses sophisticated techniques combining the semi-random method and the absorption method.

The next result is the original application of the lopsided local lemma.

Theorem 6.5.11 (Erdős and Spencer 1991)

Every $n \times n$ array where every entry appears at most $n/(4e)$ times has a Latin transversal.

Proof. Pick a transversal uniformly at random. This is the same as picking a permutation $f\colon [n] \to [n]$ uniformly at random. In Setup 6.5.4, the random injection model, transversals correspond to perfect matchings.

For each pair of equal entries in the array not both lying in the same row or column, consider the bad event that the transversal contains both entries.

The canonical negative dependency graph is obtained by joining an edge between two bad events if the four entries involved share some row or column.

[1] Not to be confused with another conjecture also known as Ryser's conjecture concerning an inequality between the covering number and the matching number of multipartite hypergraphs, as a generalization of König's theorem. See Best and Wanless (2018) for a historical commentary and a translation of Ryser's 1967 paper.

Let us count neighbors in this negative dependency graph. Fix a pair of equal entries in the array. Their rows and columns span fewer than $4n$ entries, and for each such entry z, there are at most $n/(4e) - 1$ choices for another entry equal to z. Thus the maximum degree in the canonical negative dependence graph is

$$\leq (4n-4)\left(\frac{n}{4e} - 1\right) \leq \frac{n(n-1)}{e} - 1.$$

We can now apply the symmetric lopsided local lemma to conclude that with positive probability, none of the bad events occur. □

6.6 Algorithmic local lemma

Consider an instance of a problem in the random variable setting (e.g., k-CNF) for which the local lemma guarantees a solution. Can one find a satisfying assignment efficiently?

The local lemma tells you that some good configuration exists, but the proof is nonconstructive. The probability that a random sample avoids all the bad events is often very small (usually exponentially small, e.g., in the case of a set of independent bad events). It had been an open problem for a long time whether the local lemma can be made algorithmic.

Moser (2009), during his PhD, achieved a breakthrough by coming up with the first efficient algorithmic version of the local lemma for finding a satisfying assignment for k-CNF formulas. Moser and Tardos (2010) later extended the algorithm for the general local lemma in the random variable model.

***Remark* 6.6.1** (Too hard in general)**.** The Moser–Tardos algorithm works in the random variable model (there are subsequent work that concern other models such as the random injection model). Some assumption on the model is necessary since the problem can be computationally hard in general.

For example, let $q = 2^k$, and $f\colon [q] \to [q]$ be some fixed bijection (with an explicit description and easy to compute). Consider the computational task of inverting f: given $y \in [q]$, find x such that $f(x) = y$ (we would like an algorithm with running time polynomial in k).

If $x \in [q]$ is chosen uniformly, then $f(x) \in [q]$ is also uniform. For each $i \in [k]$, let A_i be the event that $f(x)$ and y disagree on i-th bit. Then $A_1, \dots, A_k$ are independent events. Also, $f(x) = y$ if and only if no event A_i occurs. So a trivial version of the local lemma (with empty dependency graph) implies the existence of some x such that $f(x) = y$.

On the other hand, it is believed that there exist functions f that is easy to compute but hard to invert. Such functions are called ***one-way functions***, and they are a fundamental building block in cryptography. For example, let g be a multiplicative generator of $\mathbb{F}_q$, and let $f\colon \mathbb{F}_q \to \mathbb{F}_q$ be given by $f(0) = 0$ and $f(x) = g^x$ and for $x \neq 0$. Then inverting f is the ***discrete logarithm problem***, which is believed to be computationally difficult. The computational difficulty of this problem is the basis for the security of important public key cryptography schemes, such as the Diffie–Hellman key exchange.

Moser–Tardos algorithm

The algorithm is surprisingly simple.

Algorithm 6.6.2 (Moser–Tardos "fix-it")

```
input  : a set of variables and events in the random variable model
output : an assignment of variables avoiding all bad events

Initialize by setting all variables to arbitrary values;
while there is some violated event do
    Pick an arbitrary violated event and uniformly resample its variables;
```

(We can make the algorithm more precise by specifying a way to pick an "arbitrary" choice, e.g., the lexicographically first choice.)

Theorem 6.6.3 (Moser and Tardos 2010)

In Algorithm 6.6.2, letting $A_1, \dots, A_n$ denote the bad events, suppose there are $x_1, \dots, x_n \in [0, 1)$ such that

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

then for each i,

$$\mathbb{E}[\text{number of times that } A_i \text{ is chosen for resampling}] \le \frac{x_i}{1 - x_i}.$$

We won't prove the general theorem here. The proof in Moser and Tardos (2010) is beautifully written and not too long. I highly recommend it reading it. In the next subsection, we will prove the correctness of the algorithm in a special case using a neat idea known as entropy compression.

***Remark* 6.6.4** (Las Vegas versus Monte Carlo). Here are some important classes of randomized algorithms:

- ***Monte Carlo algorithm*** (MC): a randomized algorithm that terminates with an output, but there is a small probability that the output is incorrect;
- ***Las Vegas algorithm*** (LV): a randomized algorithm that always returns a correct answer, but may run for a long time (or possibly forever).

The Moser–Tardos algorithm is a LV algorithm whose expected runtime is bounded by $\sum_i x_i/(1-x_i)$, which is usually at most polynomial in the parameters of the problem.

We are usually interested in randomized algorithms whose running time is small (e.g., at most a polynomial of the input size).

We can convert an efficient LV algorithm into an efficient MC algorithm as follows: suppose the LV algorithm has expected running time T, and now we run the algorithm but if it takes more than CT time, then halt and declare a failure. Markov's inequality then shows that the algorithm fails with probability $\leq 1/C$.

However, it is not always possible to convert an efficient MC algorithm into an efficient LV algorithm. Starting with an MC algorithm, one might hope to repeatedly run it until a correct answer has been found. However, there might not be an efficient way to **check the answer**.

For example, consider the problem of finding a **Ramsey coloring**, specifically, 2-edge-coloring of K_n without a monochromatic clique of size $\geq 100\log_2 n$. A uniform random coloring works with overwhelming probability, as can be checked by a simple union bound (see Theorem 1.1.2). However, we do not have an efficient way to check whether the random edge-coloring indeed has the desired property. It is a major open problem to find an LV algorithm for finding such an edge-coloring.

Entropy compression argument

We now give a simple and elegant proof for a special case of the above algorithm, due to Moser (2009). Actually, the argument in his paper is quite a bit more complicated. Moser presented a version of the proof below in a conference, and his ideas were popularized by Fortnow and Tao. (Fortnow called Moser's talk "one of the best STOC talks ever"). Tao introduced the phase ***entropy compression argument*** to describe Moser's influential idea. (We won't use the language of entropy here, and instead use a more elementary argument involving counting and the pigeonhole principle. We will discuss entropy in Chapter 10.)

To keep the argument simple, we work in the setting of k-CNFs. Recall from Example 6.1.6 that a ***k-CNF formula*** (conjunctive normal form) consist of a logical conjunction (i.e., and, $\wedge$) of clauses, where each ***clause*** is a disjunction (i.e., or, $\vee$) of exactly k literals. We shall require that the k literals of each clause use distinct variables $(x_1, \dots, x_N)$, and each variable appears either in its positive x_i or negative form $\overline{x_i}$. For example, here is a 3-CNF with 4 clauses on 6 variables:

$$(x_1 \lor x_2 \lor x_3) \land (\overline{x_1} \lor x_2 \lor x_4) \land (\overline{x_2} \lor x_4 \lor x_5) \land (x_3 \lor \overline{x_5} \lor \overline{x_6}).$$

The problem is to find a satisfying assignment with boolean variables so that the expression output to TRUE.

Algorithm 6.6.5 (Moser "fix-it")

```
   input  : a $k$-CNF
   output : a satisfying assignment
1  Initialize by setting all variables to arbitrary values;
2  while there is some violated clause $C$ do
3      fix ($C$);
4  Subroutine fix (clause $C$) :
5      Resample the variables in $C$ uniformly at random;
6      while there is some violated clause $D$ that shares a variable with $C$ do
7          fix ($D$);
```

(We can make the algorithm more well defined by specifying a way to pick an "arbitrary" choice, e.g., the lexicographically first choice. Also, in Line 6, we allow taking $D = C$.)

Theorem 6.6.6 (Correctness of Moser's algorithm)

Given a k-CNF where every clause shares variables with at most 2^{k-3} other clauses, Algorithm 6.6.5 output a satisfying assignment with expected running time at most polynomial in the number of variables and clauses.

Note that the Lovász local lemma guarantees the existence of a solution if each clause shares variables with at most $2^k/e - 1$ clauses (each clause is violated with probability exactly 2^{-k} in a uniform random assignment of variables). So the theorem above is tight up to an unimportant constant factor.

Lemma 6.6.7 (Outer while loop)

Each clause of the k-CNF appears at most once as a violated clause in the outer while loop (Line 2).

Proof. Given an assignment of variables, by calling `fix`(C) for any clause C, any clause that was previously satisfied remains satisfied after the completion of the execution of `fix`(C). Furthermore, C becomes satisfied after the function call. Thus, once `fix`(C) is called, C can never show up again as a violated clause in Line 2. □

Lemma 6.6.8 (The number of recursive calls to fix)
Fix a k-CNF on n variables where every clause shares variables with at most 2^{k-3} other clauses. Also fix a clause C_0 and some assignment of variables. Then, in an execution of $\texttt{fix}(C_0)$, for any positive integer ℓ,

$$\mathbb{P}(\text{there are at least } \ell \text{ recursive calls to } \texttt{fix} \text{ in Line 7}) \le 2^{-\ell+n+1}.$$

It follows that the expected number of recursive calls to `fix` is $n + O(1)$. Thus, in the Moser algorithm (Algorithm 6.6.5), the expected total number of calls to `fix` is $mn + O(m)$, where n is the number of variables and m is the number of clauses. This proves the correctness of the algorithm (Theorem 6.6.6).

Proof. Let us formalize the randomness in the algorithm by first initializing a random string of bits. Specifically, let $x \in \{0,1\}^{k\ell}$ be generated uniformly at random. Whenever the a clause in resampled in Line 5, one replaces the variables in the clause by the next k bits from x. Furthermore, if the line Line 7 is called for the ℓ-th time, we halt the algorithm and declare a failure (as we would have run out of random bits to resample had we continued).

At the same time, we keep an ***execution trace*** which keeps track of which clauses got called `fix`, and also when the inner while loop Line 6 ends. Note that the very first call to $\texttt{fix}(C_0)$ is not included in the execution trace since it is already given as fixed and so we don't need to include this information. Here is an example of an execution trace, writing C7 for the 7th clause in the k-CNF:

```
fix(C7) called
fix(C4) called
fix(C7) called
while loop ended
fix(C2) called
while loop ended
while loop ended
...
```

For illustration, here is the example of how clause variables could intersect:

```
C2: ****
C4:   ****
C7:      ****
```

It is straightforward to deduce which `while loop ended` corresponds to which `fix` call by reading the execution trace and keeping track of a first-in-first-out stack.

Encoding the execution trace as a bit string. We fix at the beginning some canonical order of all clauses (e.g., lexicographic). It would be too expensive to refer to each clause in its absolute position in this order (this is an important point!). Instead, we note that every clause shares variables with at most 2^{k-3} other clauses, and only these $\leq 2^{k-3}$ could be called in the inner while loop in Line 6. So we can record which one got called using a $k-3$ bit string.

- `fix`(D) `called`: suppose this was called inside an execution of `fix`(C), and D is the j-th clause among all clauses sharing a variable with C, then record in the execution trace bit string `0` followed by exactly $\ell - 3$ bits giving the binary representation of j (prepended by zeros to get exactly $\ell - 3$ bits).
- `while loop ended`: record 1 in the execution trace bit string.

Note that one can recover the execution trace from the above bit string encoding.

Now, suppose the algorithm terminates as a failure due to `fix` being called the ℓ-th time. Here is the key claim.

Key claim (recovering randomness). At the moment right before the ℓ-th recursive call to `fix` on Line 7, we can completely recover x from the current variable assignments and the execution trace.

Note that all ℓk random bits in x have been used up at this point.

To see the key claim, note that from the execution trace, we can determine which clauses were resampled and in what order. Furthermore, if `fix`(D) was called on Line 7, then D must have been violated right before the call, and there is a unique possibility for the violating assignment to D right before the call (e.g., if $D = x_1 \vee x_2 \vee \overline{x_3}$, then the only violating assignment is $(x_1, x_2, x_3) = (0, 0, 1)$). We can then rewind history, and put the reassigned values to D back into the random bit string x to complete recover x.

How long can the execution bit string be? It has length $\leq \ell(k-1)$. Indeed, each of the $\leq \ell$ recursive calls to `fix` produces $k-2$ bits for the call to `fix` and 1 bit for ending the while loop. So the total number of possible execution strings is $\leq 2^{\ell(k-1)+1}$ (the $+1$ accounts for variable lengths, though it can removed with a more careful analysis).

Thus, the key claim implies that each $x \in \{0,1\}^{\ell k}$ that leads to a failed execution produces a unique pair (variable assignment, execution bit string). Thus

$$\mathbb{P}(\geq \ell \text{ recursive calls to } \texttt{fix})\, 2^{\ell k} = |\{x \in \{0,1\}^n \text{ leading to failure}\}| \leq 2^n 2^{\ell(k-1)+1}.$$

Therefore, the failure probability is $\leq 2^{-\ell+n+1}$. □

***Remark* 6.6.9** (Entropy compression)**.** Tao use the phrase “entropy compression” to describe this argument. The intuition is that the recoverability of the random bit string

x means that we are somehow "compressing" a ℓk-bit random string into a shorter length losslessly, but that would be impossible. Each call to `fix` uses up k random bits and converts it to $k-1$ bits to the execute trace (plus at most n bits of extra information, namely the current variables assignment, and this is viewed as a constant amount of information), and this conversion is reversible. So we are "compressing entropy." The conservation of information tells us that we cannot losslessly compress k random bits to $k-1$ bits for very long.

***Remark* 6.6.10** (Relationship between the two proofs of the local lemma?)**.** The above proof, along with extensions of these ideas in Moser and Tardos (2010), seems to give a completely different proof of the local lemma than the one we saw at the beginning of the chapter. Is there some way to relate these seemingly completely different proofs? Are they secretly the same proof? We do not know. This is an interesting open-ended research problem.

Exercises

1. Show that it is possible to color the edges of K_n with at most $3\sqrt{n}$ colors so that there are no monochromatic triangles.
2. Prove that it is possible to color the vertices of every k-uniform k-regular hypergraph using at most $k/\log k$ colors so that every color appears at most $O(\log k)$ times on each edge.
3. $\star$ *Hitting thin rectangles.* Prove that there is a constant $C > 0$ so that for every sufficiently small $\varepsilon > 0$, one can choose exactly one point inside each grid square $[n, n+1) \times [m, m+1) \subset \mathbb{R}^2$, $m, n \in \mathbb{Z}$, so that every rectangle of dimensions ε by $(C/\varepsilon)\log(1/\varepsilon)$ in the plane (not necessarily axis-aligned) contains at least one chosen point.
4. *List coloring.* Prove that there is some constant $c > 0$ so that given a graph and a set of k acceptable colors for each vertex such that every color is acceptable for at most ck neighbors of each vertex, there is always a proper coloring where every vertex is assigned one of its acceptable colors.
5. Prove that, for every $\varepsilon > 0$, there exist ℓ_0 and some $(a_1, a_2, \dots) \in \{0,1\}^{\mathbb{N}}$ such that for every $\ell > \ell_0$ and every $i > 1$, the vectors $(a_i, a_{i+1}, \dots, a_{i+\ell-1})$ and $(a_{i+\ell}, a_{i+\ell+1}, \dots, a_{i+2\ell-1})$ differ in at least $(\frac{1}{2} - \varepsilon)\ell$ coordinates.
6. *Avoiding periodically colored paths.* Prove that for every Δ, there exists k so that every graph with maximum degree at most Δ has a vertex-coloring using k colors so that there is no path of the form $v_1 v_2 \dots v_{2\ell}$ (for any positive integer

ℓ) where v_i has the same color as $v_{i+\ell}$ for each $i \in [\ell]$. (Note that vertices on a path must be distinct.)

7. Prove that every graph with maximum degree Δ can be properly edge-colored using $O(\Delta)$ colors so that every cycle contains at least three colors.

   (An edge-coloring is *proper* if it never assigns the same color to two edges sharing a vertex.)

8. ⋆ Prove that for every Δ, there exists g so that every bipartite graph with maximum degree Δ and girth at least g can be properly edge-colored using $\Delta + 1$ colors so that every cycle contains at least three colors.

9. ⋆ Prove that for every positive integer r, there exists C_r so that every graph with maximum degree Δ has a *proper* vertex coloring using at most $C_r\Delta^{1+1/r}$ colors so that every vertex has at most r neighbors of each color.

10. *Vertex-disjoint cycles in digraphs.* (Recall that a directed graph is k*-regular* if all vertices have in-degree and out-degree both equal to k. Also, cycles cannot repeat vertices.)

    a) Prove that every k-regular directed graph has at least $ck/\log k$ vertex-disjoint directed cycles, where $c > 0$ is some constant.

    b) ⋆ Prove that every k-regular directed graph has at least ck vertex-disjoint directed cycles, where $c > 0$ is some constant.

    Hint: split in two and iterate

11. a) *Generalization of Cayley's formula.* Using Prüfer codes, prove the identity

    $$x_1 x_2 \cdots x_n (x_1 + \cdots + x_n)^{n-2} = \sum_T x_1^{d_T(1)} x_2^{d_T(2)} \cdots x_n^{d_T(n)}$$

    where the sum is over all trees T on n vertices labeled by $[n]$ and $d_T(i)$ is the degree of vertex i in T.

    b) Let F be a forest with vertex set $[n]$, with components having $f_1, \dots, f_s$ vertices so that $f_1 + \cdots + f_s = n$. Prove that the number of trees on the vertex set $[n]$ that contain F is exactly $n^{n-2} \prod_{i=1}^{s} (f_i / n^{f_i - 1})$.

    c) *Independence property for uniform spanning tree of K_n.* Show that if H_1 and H_2 are vertex-disjoint subgraphs of K_n, then for a uniformly random spanning tree T of K_n, the events $H_1 \subseteq T$ and $H_2 \subseteq T$ are independent.

    d) ⋆ *Packing rainbow spanning trees.* Prove that there is a constant $c > 0$ so that for every edge-coloring of K_n where each color appears at most cn times, there exist at least cn edge-disjoint spanning trees, where each spanning tree has all its edges colored differently.

(In your submission, you may assume previous parts without proof.)

The next two problems use the lopsided local lemma.

12. *Packing two copies of a graph.* Prove that there is a constant $c > 0$ so that if H is an n-vertex m-edge graph with maximum degree at most cn^2/m, then one can find two edge-disjoint copies of H in the complete graph K_n.

13. ⋆ *Packing Latin transversals.* Prove that there is a constant $c > 0$ so that every $n \times n$ matrix where no entry appears more than cn times contains cn disjoint Latin transversals.