A new lower bound for sphere packing

Talk by Julian Sahasrabudhe
Notes by Sanjana Das
March 1, 2024

This is joint work with Marcelo Campos, Matt Jenssen, and Marcus Michelen.

§1 Introduction

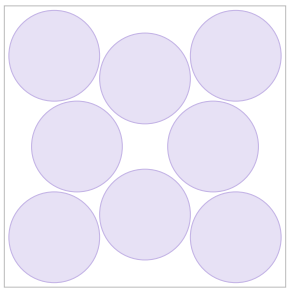
In the problem we're interested in, we take $\mathbb{R}^d$, and we want to cover the greatest possible fraction of it with disjoint, identical spheres. (We allow our spheres to be tangent, but their interiors shouldn't overlap.)

Definition 1.1. For $d \in \mathbb{N}$, we define $\theta(d)$ as the largest possible fraction of $\mathbb{R}^d$ that we can cover with disjoint, identical spheres.

The common size of the spheres doesn't matter; throughout the talk we'll assume they have volume 1.

Definition 1.2. We use r_d to denote the radius of the volume-1 sphere in $\mathbb{R}^d$.

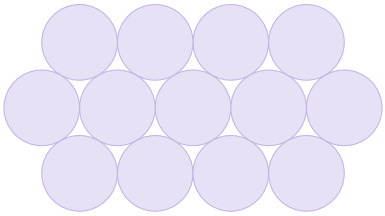
Then we imagine taking a very large box $[-L, L]^d$, and trying to pack disjoint spheres of radius r_d into it (and we want to know how many spheres we can fit).



§1.1 History for small d

This question has been around for a while. First, we have $\theta(1) = 1$ — in one dimension a ball is just an interval, and we can cover all of $\mathbb{R}$ with disjoint intervals.

But things already become interesting when we go to two dimensions. In this case, we do know the optimal packing — it comes from placing our balls at the centers of a hexagonal lattice. Then $\theta(2)$ is whatever the density of this packing is. This was proven by Thue quite a long time ago.



In three dimensions, finding $\theta(3)$ is already pretty difficult. It was long conjectured that the optimal packing is a lattice-like packing where we sort of stack hexagonally; this was resolved by Hales in the 1990s, and more recently computer-checked in the 2010s. This question has been around for a long time — it was first conjectured by Kepler in 1611, and there are interesting partial results of Gauss for the case where the centers of the spheres are on a lattice — and its proof was a tour de force using lots of computational power.

Beyond this, what we know comes from a few amazing achievements. First, $\theta(8)$ was determined by Viazovska in 2017 — she showed that the optimal packing is the one where we put balls centered at the points of the E_8 lattice. And building on this, Cohn, Kumar, Miller, Radchenko, and Viazovska (also in 2017) found $\theta(24)$ — again, here the optimal packing comes from a certain 24-dimensional lattice. Both of these results fit into a framework developed by Cohn and Elkies — they showed that if you can construct a certain *witness function* in $\mathbb{R}^n$ with certain symmetry properties (such that the function is positive in a certain region, and so is its Fourier transform), then it'll certify that these packings really are optimal.

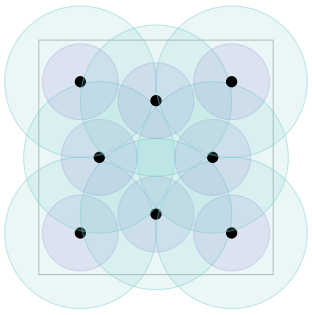
And these are the only dimensions for which we know the optimal packing.

§1.2 History for large d

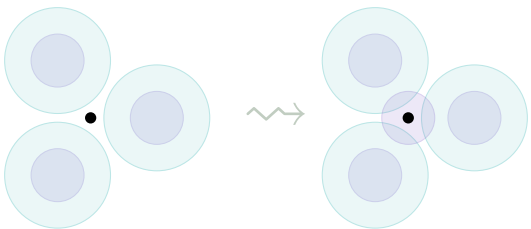
The main focus of our talk today will be about what happens when d is very large — so we'll think of taking d to ∞. Here it's less obvious how to even *construct* a natural sphere packing. But with a bit of thought, we can get a simple lower bound.

> **Claim 1.3 —** We have $\theta(d) \geq 2^{-d}$.

Proof. Take *any* sphere packing which is saturated (meaning that we can't add any other sphere to it), and imagine we blow up all the spheres by a factor of 2.



Then these larger spheres have to cover the entire space — since if there were some uncovered point, then we could have added a sphere at that point to the original packing.



And blowing up the radius of a sphere by 2 blows up its volume by 2^{-d}; so since the total volume of the blown-up spheres is at least that of the entire space, the original spheres occupy at least a 2^{-d}-fraction of the volume of the space. □

This bound was improved by Minkowski (1905), who showed that

$$\theta(d) \geq (2 + o(1))2^{-d}.$$

This remained the best-known lower bound until Rogers (1947), who improved the lower-order term from constant to linear, showing that

$$\theta(d) \geq cd2^{-d} \text{ with } c = 2/e.$$

Then there was a series of improvements to the constant c — Davenport–Rogers (1947) improved it to $c \approx 1.68$, and Ball (1992) improved it all the way to 2. This remained the best-known bound until Venkatesh (2013) improved it to 65963 (this is one of the more exciting numbers you'll see in a math paper). Furthermore, in the same paper he showed that we can get an improvement of $\log\log d$ on a certain sparse sequence of dimensions d — i.e., he proved that

$$\theta(d) \geq c(\log\log d)d2^{-d}$$

for a sequence of d's tending to ∞ (though quite a sparse one), for some constant $c > 0$.

§1.3 The main result

The main contribution of this paper is that there exist sphere packings that are even denser.

Theorem 1.4 (Campos–Jenssen–Michelen–Sahasrabudhe 2024)
For all sufficiently large d, we have

$$\theta(d) \geq \left(\frac{1}{2} - o(1)\right)\frac{d\log d}{2^d}.$$

There's some good news and some bad news about this bound. So far, we've only been discussing *lower* bounds for $\theta(d)$, but for context, what about upper bounds? Unfortunately, the best-known upper bounds are exponentially far from this lower bound. Kabatjanski–Levenstein (1978) showed that

$$\theta(d) \geq 2^{-0.599d+o(d)},$$

and since then, there's only been a constant factor improvement, due to Cohn–Zhao (2014). So we're happy because we've improved the best-known sphere packings, but it's a bit disappointing that we're still a long way from the best upper bound.

But there's actually something really interesting about getting up to this particular point (of $d(\log d)2^{-d}$). One interesting thing about our packings is that they don't look lattice-like at all. All the previous constructions come from picking a really clever lattice and putting the balls at the lattice points (with the better bounds coming from better ways of choosing the lattice). But here we use a totally different strategy.

Morally speaking, our strategy is roughly that we start by taking our big box, grabbing a random sphere, and throwing it out. And then we choose another sphere in the box, conditioned on not intersecting the first sphere, and we throw it out as well. And we keep doing this — choosing a random sphere, conditioned on not intersecting any of the previous ones, and throwing it out. This defines some random process; we can ask when it stops, and the answer is basically the bound in Theorem 1.4. (This isn't exactly how the construction works, but it's similar.)

And interestingly, this bound meets up with work of physicists who came to the same number using totally different methods — work by Parisi and Zamponi from the 2010s predicted that there exist *amorphous* sphere packings of density all the way up to

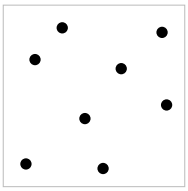
$$(1+o(1))\frac{d\log d}{2^d},$$

which is the same as the bound we get up to a factor of 2. It's not clear what exactly 'amorphous' means, but a sphere packing constructed in this way should satisfy most reasonable notions — for example, it'll have fairly rapid decay of correlations, it certainly doesn't look lattice-like, and so on.

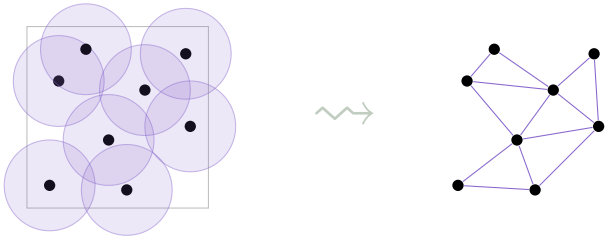
And the missing factor of 2 is also quite interesting — it actually meets up with one of the missing factors of 2 for off-diagonal Ramsey numbers.

§2 A connection to graphs

How is this problem connected to Ramsey numbers? The point is sort of that we can interpret constructing a sphere packing in graph theoretic terms. First, continuous space is confusing, so we'd like to discretize it in some way; we'll describe how to perform this discretization later, but for now imagine that we've taken our big box and discretized it to produce a finite set of points X (and we'll only treat the points in X as candidate centers for our spheres).



Then we can define a graph G_X on these points, where we join two points x and y if the spheres centered at x and y overlap — equivalently, if $|x-y|<2r_d$.



Then constructing a sphere packing (which only uses spheres centered at points in X) corresponds exactly to finding an independent set in G_X. So we're trying to bound the independence number of a graph G_X, which will have some confusing properties inherited from the way we constructed it (as a discretization of space). And that's what we're doing in off-diagonal Ramsey as well — we're trying to bound the independence number of a graph, given that it doesn't contain a small clique — which is roughly why they're connected.

§2.1 Finding independent sets in graphs

We'll now set aside sphere packing for a second, and talk about a few results about building independent sets in graphs — particularly graphs with a bit of structure.

Here, inspiration comes from the following theorem (which was first proved by Ajtai–Komlós–Szemerédi and then improved by Shearer), where the bit of structure we impose on our graph is triangle-free.

Definition 2.1. We use $\alpha(G)$ to denote the size of the largest independent set in G.

Theorem 2.2 (Ajtai–Komlós–Szemerédi, Shearer 1980s)
Let G be a n-vertex triangle-free graph with maximum degree Δ. Then

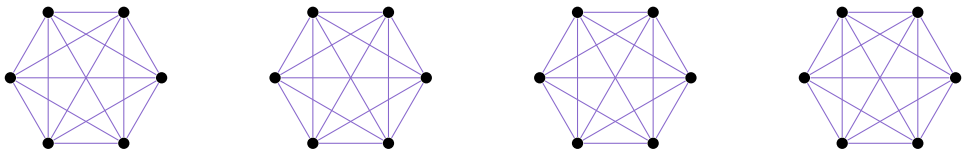
$$\alpha(G) \geq (1 + o(1))\frac{n}{\Delta}\log \Delta.$$

This is exactly what you'd use to prove an upper bound on the Ramsey number $R(3, k)$.

Is this a good result, and what does it mean? As a sanity check, let's see what happens if we *don't* impose any structure on G (i.e., we remove the triangle-free condition). Then the worst case is if G is the union of a bunch of disjoint copies of $K_{\Delta+1}$; then we'll have

$$\alpha(G) = \frac{n}{\Delta + 1},$$

since we can only take one vertex from each.



So Shearer's theorem says that we can get an independent set of size a $\log \Delta$ factor better than the trivial bound we'd get when there's no structure. It's also sharp up to a factor of 2 — there are graphs with

$$\alpha(G) \leq (2 + o(1))\frac{n \log \Delta}{\Delta}$$

(obtained from random d-regular graphs — there'll be a few triangles that appear and need to be dealt with, but that's not much of an issue). This factor of 2 is a mystery (we don't know what the correct constant should be, and figuring it out is a major open problem).

We'd like to weaken this hypothesis a bit. For example, Theorem 2.2 deals with graphs that have *no* triangles, but what about graphs that have only a *few* triangles? Specifically, every vertex is in at most Δ^2 triangles (its neighborhood has at most Δ vertices, so at most Δ^2 edges). So we're going to impose the condition that every vertex is in at most $\lambda\Delta^2$ triangles (where we think of λ as a small constant).

Corollary 2.3
Let G be a n-vertex graph with maximum degree Δ and with the property that every vertex is in at most $\lambda\Delta^2$ triangles. Then (for some constant $c > 0$) we have

$$\alpha(G) \geq c\frac{n}{\Delta}\log\frac{1}{\lambda}.$$

And this is sharp (we can get constructions from appropriate blowups).

This is the main inspiration for our main graph-theoretic result. In our main result, the condition we'll impose on G will capture just the right notion of structure that comes from taking a set of points in a big box. Instead of controlling *triangles*, we're going to control *codegrees* (the number of common neighbors of two vertices); we'll use $d(u, v)$ to denote the codegree of u and v.

Theorem 2.4 (Campos–Jenssen–Michelen–Sahasrabudhe 2024)
Let G be a n-vertex graph with maximum degree Δ such that for all $u \neq v$, we have

$$d(u,v) \leq c\frac{\Delta}{(\log \Delta)^8}$$

(where c is some constant). Then we have

$$\alpha(G) \geq (1-o(1))\frac{n}{\Delta}\log \Delta.$$

We definitely need to have *some* condition on G that drives us away from G being a disjoint union of copies of $K_{\Delta+1}$, and if we look at codegrees, it turns out that a $\mathsf{polylog}(\Delta)$ saving (compared to the trivial bound $d(u,v) \leq \Delta$) is enough.

And this theorem is sharp, up to the exponent of 8 (we can get an example in the other direction by putting down a bunch of cliques).

§3 Construction of the sphere packing

First we'll see how Theorem 2.4 is useful for constructing a sphere packing — in particular, how does the codegree condition arise in our graph G_X?

§3.1 The discretization

For this, we first need to construct an appropriate discretization X of our big box $[-L,L]^d$. The starting point is that we'll first sample X according to a Poisson point process with a certain intensity λ (the *intensity* of a Poisson point process is the expected number of points chosen from any open set of volume 1), which we'll take to be

$$\lambda = \left(\frac{d}{c\log d}\right)^{d/2} = d^{d/2+o(d)}$$

(for some constant c). And then we'll take this Poisson point process and modify it in a few basic ways, by deleting 'bad' vertices $x \in X$.

First, in order to apply Theorem 2.4, we need to ensure that none of our degrees are too large. To figure out what 'too large' should mean, what's the *expected* degree of a vertex? We're drawing an edge between two points if they're at most $2r_d$ apart, so for any fixed v, the number of neighbors of v is the number of points in the ball of radius $2r_d$ about v. And since we're choosing points according to a Poisson point process, this is controlled by the volume of that ball, which is 2^d (since the ball of radius r_d has volume 1). So we get

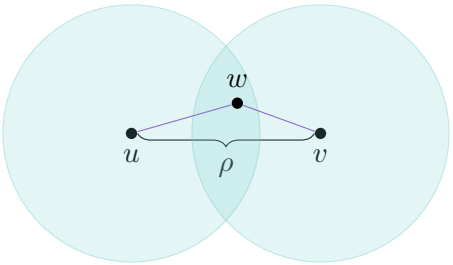
$$\mathbb{E}[d(v)] = \mathbb{E}\#(\text{points in } \mathbb{B}(v,2r_d)) = 2^d\lambda.$$

When choosing our target maximum degree we'll give ourselves a bit of elbow room, so we set

$$\Delta = 2^d\lambda\left(1+\frac{1}{d}\right).$$

Then there's the obvious first type of bad vertex — vertices whose degree exceeds Δ. We can use standard concentration estimates to show that the number of such vertices is a very small proportion of the total number of vertices, so we can just delete them — in other words, we delete all vertices v with $d(v) > \Delta$, which ensures that our graph has maximum degree Δ (and we lose just a tiny fraction of X by doing so).

The other thing we want to ensure is the condition on codegrees (i.e., common neighborhoods). Suppose we take two points u and v a distance ρ apart; how many points w do we expect to be joined to both u and v? (Interestingly, this step is basically the only geometric fact we need in the proof — the rest is pure combinatorics.) Suppose we draw two balls of radius $2r_d$ around u and v; then we want to find the expected number of points in their intersection, which corresponds to the *volume* of this intersection.



And we can show that $\mathrm{Vol}(\mathbb{B}(u) \cap \mathbb{B}(v)) \le e^{-\rho^2/4} \cdot 2^d$, so we get that

$$\mathbb{E}[d(u,v)] \le \lambda \cdot e^{-\rho^2/4} \cdot 2^d.$$

This brings us to our second type of bad point — we delete a point u if there is any point v very close to u, specifically with $|u - v| \le \log d$. Note that $r_d \approx \sqrt{d}$, so a distance of $\log d$ is very small in comparison; so we can calculate that there's very few such points u (in expectation), and throwing them away again loses just a negligible fraction of X.

And after deleting these points, every two remaining points have a distance of at least $\log d$, which means their expected codegree is

$$\mathbb{E}[d(u,v)] \le \lambda \cdot 2^d \cdot e^{-(\log d)^2/4}.$$

The term $\lambda \cdot 2^d$ is basically our value of Δ. Meanwhile, to deal with the extra factor, we defined $\Delta \approx \lambda \cdot 2^d$ where $\lambda \approx d^{d/2}$, which means $\log\log\Delta \approx \log d$ (asymptotically), which means

$$\mathbb{E}[d(u,v)] \le \Delta e^{-(\log\log\Delta)^2/4}.$$

And the point is that this extra factor goes to 0 much faster than any power of $\log\Delta$ (since the exponent of $(\log\log\Delta)^2$ grows faster than $\log\log\Delta$), so we'll get

$$\mathbb{E}[d(u,v)] \le \Delta e^{-(\log\log\Delta)^2/4} \le \frac{\Delta}{(\log\Delta)^c}$$

for *any* constant c (Theorem 2.4 asks for $c = 8$).

§3.2 Finding an independent set

Now we've controlled the degrees and codegrees of our graph G_X, which means we can use Theorem 2.4. Here $n = |X| \approx \lambda(2L)^d$, since we first sampled X with intensity λ from our big box (which has volume $(2L)^d$) and then deleted a negligible fraction of it when removing bad points. And meanwhile, we have $\Delta = 2^d\lambda$ with $\lambda \approx d^{d/2}$, so $\log\Delta \approx (d/2)\log d$, and we get

$$\theta(d) \ge \frac{\alpha(G)}{(2L)^d} \ge (1 - o(1)) \cdot \frac{\lambda(2L)^d}{2^d\lambda} \cdot \frac{d\log d}{2} \cdot \frac{1}{(2L)^d} = \left(\frac{1}{2} - o(1)\right)\frac{d\log d}{2^d}.$$

§4 Proof of Theorem 2.4

Now all that remains is to prove Theorem 2.4 (on finding independent sets in certain kinds of graphs).

§4.1 The main idea

The idea to approach this is an old one in combinatorics, called the *Rödl nibble* (though we'll take a twist on it). At the beginning of this talk, we said that we were going to construct our packing by removing a single point at a time; but for us it actually makes more sense to remove a *cluster* of points at a time.

Imagine we start with a n-vertex graph G with maximum degree Δ which satisfies our codegree condition. Our goal is to build a large independent set, and we'll construct this set in rounds (each of which is random).

We'll fix some positive constant $\gamma \ll 1$ (we'll actually take $\gamma = 1/(\log \Delta)^2$). Then in round 1, we'll let A_1 be a p_1-random set with $p_1 = \gamma/\Delta$ (i.e., we select every vertex independently with probability p_1).

Every vertex in G has degree at most Δ, which means its expected number of neighbors in A_1 is roughly $\gamma \ll 1$. This means most vertices in A_1 will have *no* neighbors in A_1, so almost all of A_1 is an independent set; this allows us to extract a large independent set from A_1, which we'll do later.

Now we remove all vertices with an edge to A_1 — i.e., the neighborhood of A_1, which we denote by $N(A_1)$.

This is the first step of our construction of the independent set, but it's not nearly enough on its own — this gives us an independent set of size roughly $\gamma n/\Delta$, and we want a set much bigger than this. So what we'd *like* to do is continue doing this on the remainder of the graph.

Why can we continue doing this? If G were a union of cliques, then we'd be deleting a bunch of these cliques, and the rest of the graph would remain the same (i.e., it'd still be a union of cliques), which would not be good for us (we'd only be able to iterate for $1/\gamma$ steps, to get an independent set of size n/Δ). So the codegree condition has to come into play somehow. And the way it comes into play is that we can use it to say that $N(A_1)$ looks somewhat like a *random* set. Specifically, we'd expect $N(A_1)$ to have size roughly γn (we have roughly $\gamma n/\Delta$ vertices in A_1, and each has roughly Δ neighbors). If we removed a *random* set of this size, then we'd expect to gobble up a γ-fraction of the neighbors of each vertex. And we can use the codegree condition to show that the same is true of $N(A_1)$ — specifically, when we delete it, most of the vertices left will have at least roughly a γ-fraction of their neighbors eaten (and we can also delete the few misbehaving vertices for which this isn't the case).

Then we're left with a graph on $n_2 \approx (1-\gamma)n$ vertices with maximum degree $\Delta_2 \approx (1-\gamma)\Delta$. And then we can do the same thing on this graph — in round 2, we choose A_2 to be p_2-random with $p_2 = \gamma/\Delta_2$, so

$$|A_2| \approx \frac{\gamma n_2}{\Delta_2} \approx \frac{\gamma(1-\gamma)n}{(1-\gamma)\Delta} = \frac{\gamma n}{\Delta}.$$

So this second bite A_2 we've taken is exactly the same size as the first bite A_1.

And we keep on iterating this as long as we can; when do we have to stop? At every step we're eating a $(1-\gamma)$-fraction of the world (when we remove the set A_t and its neighborhood), so after t rounds, the number of remaining vertices is roughly $(1-\gamma)^t n$. And we become screwed once this is smaller than $\gamma n/\Delta$ (then there's not enough room to take another step — $\gamma n/\Delta$ is the common size of each bite A_t), so the number of steps we can run for is roughly $t \approx (\log \Delta)/\gamma$. So we iterate for this many steps, and we track the process through those steps to show that it really works.

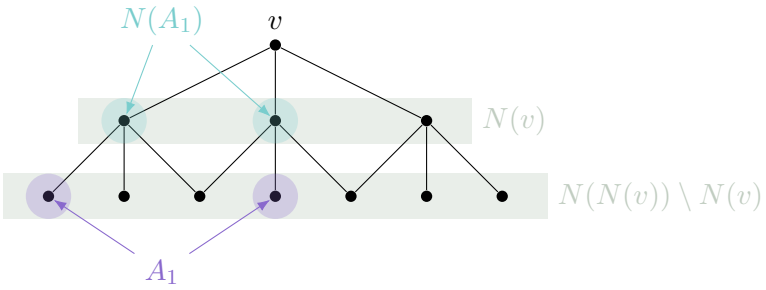
§4.2 Some difficulties

In the remaining time, we'll quickly talk about a few of the difficulties that pop up in this story, and what we do about them.

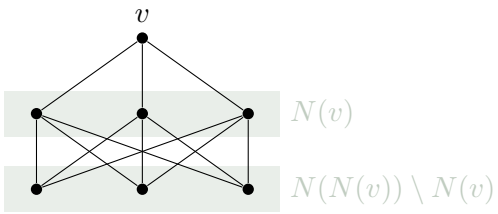
§4.3 Shrinking degrees

First, what's missing from what we've said so far is that we need to actually prove that $N(A_1)$ is 'quasirandom' (in the sense that removing it really eats up the right number of neighbors of all the other vertices).

To do this, let's consider some vertex v, and condition on v itself not getting eaten up in the first step (so neither v nor any of its Δ neighbors are in A_1). Then to see how the degree of v changes after the first step, we need to consider the neighbors of these neighbors — i.e., the set $N(N(v)) \setminus N(v)$. We need to show that enough vertices here get hit (i.e., included in A_1) that at least (roughly) a γ-fraction of the vertices in $N(v)$ get destroyed (meaning that one of their neighbors is placed in A_1).



If we didn't have the codegree condition, then there'd be an obvious problem — what if $N(v)$ and $N(N(v)) \setminus N(v)$ formed a complete bipartite graph with Δ vertices on each side?



Then the expected number of vertices in $N(N(v)) \setminus N(v)$ placed into A_1 would be just γ, which is tiny (since we're choosing each vertex with probability $p_1 = \gamma/\Delta$, and there are Δ vertices in it). So in this case, there would be lots of vertices v for which we don't choose *any* vertices of $N(N(v)) \setminus N(v)$ for A_1 (which means the degree of v doesn't shrink at all), which would be bad.

But the condition on codegrees prevents this from happening — in particular, it ensures that this second neighborhood actually has size at least $\Delta(\log \Delta)^8$ (which is much bigger than Δ). And it actually lets us run a martingale argument — we go through the vertices of $N(N(v)) \setminus N(v)$ one at a time and reveal each is in A_1 or not, and we track the random variable representing how many vertices in $N(v)$ get knocked out (i.e., have a neighbor placed into A_1). (We're considering the exposure martingale which keeps track of the expectation of this random variable, given the vertices we've revealed so far.) The codegree condition ensures this martingale is well-behaved, so we can use martingale concentration tools to get the statement we want. (We actually need a one-sided rather than two-sided bound, so the martingale tools are a bit different than usual, but this argument is still quite chill.)

We also have to preserve the codegree condition at each step of the algorithm (otherwise you could imagine that the algorithm first allows the codegrees to grow, and then we start getting clumpy bits and everything goes off the rails). But this can be done using the same argument — given two vertices u and v, we look at their common neighborhood $N(u) \cap N(v)$, and then the set of *its* neighbors. And we again define an exposure martingale corresponding to revealing those second neighbors one at a time.

§4.4 The regularization step

There's one final twist in this argument — what if v has large degree (e.g., Δ), but for some weird reason, all the vertices in $N(v)$ have much smaller degree? (So far we've sort of assumed all vertices have degree

exactly Δ, but this doesn't have to be the case.) Then the second neighborhood of v is going to be small, and this argument won't work.

You might think that in this situation we're ahead — it means we have lots of vertices with small degree, so we might hope to win for another reason. But it definitely complicates the analysis (since we wouldn't just be tracking concentration around a single value anymore). So instead, we deal with this in a very simple way — if we have vertices with small degree, we just add edges to the graph to increase their degrees. (Finding an independent set only becomes harder if we add more edges, so this is certainly acceptable.)

So after every nibble step (where we sample A_i and then delete it together with its neighbors), we take the remaining graph and regularize it (while preserving the codegree condition); and then everything is fine.

There's one other trick at this point — what if the graph is so small relative to Δ that we *can't* regularize (meaning that putting in an edge would force us to create a large codegree)? The trick is that in this case, we can just take a bunch of disjoint copies of the graph, and regularize by adding edges *between* these graphs; then the large independent set we find in the union of these copies will give us a large independent set in at least one of these copies.