4
Applications of Concentration Inequalities

4.1 Power of Two Choices: A Random Graphs Proof

Another way to show that the maximum load is $O(\log \log n)$—note that the constant is worse—is to use an first-priciples analysis based on properties of random graphs. We build a random graph G as follows: the n vertices of G correspond to the n bins, and the edges correspond to balls—each time we probe two bins we connect them with an edge in G. For technical reasons, we'll just consider what happens if we throw fewer balls (only $m = n/C$ balls) into n bins—also, let's imagine that each ball chooses two distinct bins each time.

The constants can be optimized a bit further, but I am keeping it simple here.

Theorem 4.1. *Let $C \geq 2e^2$. If we throw n/C balls into n bins using the best-of-two-bins method, the maximum load of any bin is $O(\log \log n)$ whp.*

Hence for n balls and n bins, the maximum load should be at most C times as much, whp. (It's as though after every n/C balls, we forget about the current loads and zero out our counters—not zeroing out these counters can only give us a more evenly balanced allocation; I'll try to put in a formal proof later.)

To prove the theorem, we need two results about the random graph G obtained by throwing in n/C random edges into n vertices. Both the proofs are simple but surprisingly effective counting arguments, they appear at the end.

Lemma 4.2. *The size of G's largest connected component is $O(\log n)$ whp.*

Lemma 4.3. *For all subsets S of the vertex set, the induced graph $G[S]$ contains at most $3|S|$ edges, and hence has average degree at most 6, whp.*

Given the graph G, suppose we repeatedly perform the following operation in rounds:

> In each round, remove all vertices of degree ≤ 12 in the current graph.

We stop when there are no more vertices of small degree.

Lemma 4.4. *This process ends after $O(\log\log n)$ rounds whp.*

Proof. Condition on the events in the two previous lemmas. Any component C in the current graph has average degree at most 5; by Markov at least half the vertices have degree at most 12 and will be removed, and we halve the component size. But the size of each component was $O(\log n)$ to begin, so this takes $O(\log\log n)$ rounds. □

Lemma 4.5. *If a node/bin survives i rounds before it is deleted, its load due to edges that have already been deleted is at most $12i$. If a node/bin is never deleted, its load is at most $12i^*$, where i^* is the total number of rounds.*

Proof. Consider the nodes removed in round 1: their degree was at most 12, so even if all those balls went to such nodes, their final load would be at most 12. Now, consider any node x that survived this round. While many edges incident to it might have been removed in this round, we claim that at most 12 of those would have contributed to x's load. Indeed, the each of the other endpoints of those edges went to bins with final load at most 12. So at most 12 of them would choose x as their less loaded bin before it is better for them to go elsewhere.

Now, suppose y is deleted in round 2: then again its load can be at most 24: twelve because it survived the previous round, and 12 from its own degree in this round. OTOH, if y survives, then consider all the edges incident to y that were deleted in previous rounds. Each of them went to nodes that were deleted in rounds 1 or 2, and hence had maximum load at most 24. Thus at most 24 of these edges could contribute to y's load before it was better for them to go to the other endpoint. The same inductive argument holds for any round $i \le i^*$. Finally, the process ends when each component is a singleton, and hence there are no more balls to assign. □

By Lemma 4.4, the number of rounds is $i^* = O(\log\log n)$ whp, so by Lemma 4.5 the maximum load is also $O(\log\log n)$ whp.

4.1.1 Missing Proofs of Lemmas

Lemma 4.6. *The size of G's largest connected component is $O(\log n)$ whp.*

This fact actually holds for a random graph with n nodes and any $m < \frac{1}{2}n$ edges; see the Frieze and Karoński book.

Proof. We have a graph with n vertices and $m = n/c$ edges where we connect vertices at random. If there is a component of at least k vertices, there must be a spanning tree with at least $k-1$ edges, and hence some subset S of k vertices must have had some $k-1$ edges fall into it. For any of the $\binom{n}{k}$ choices of S, and $\binom{m}{k-1}$ choices of the edge set, the probability of these edges choosing both endpoints in that set is $[(k/n)^2]^{k-1}$. (For $k \ll m$, we can upper bound $\binom{m}{k-1}$ by $\binom{m}{k}$—this

simplifies some calculations.) Now a union bound says that the "bad event" of some component of size k is at most

$$\binom{n}{k} \cdot \binom{m}{k} \cdot \left(\frac{k}{n}\right)^{2(k-1)} \leq n^2 \cdot \left(\frac{e^2}{C}\right)^k \leq 1/\operatorname{poly}(n),$$

for any $k = c \log n$ with a large enough constant c; here we used the standard approximation that $\binom{n}{k} \leq (\frac{ne}{k})^k$, and also our assumption that $C \geq 2e^2$. □

Lemma 4.7. *For all subsets S of the vertex set, the induced graph $G[S]$ contains at most $3|S|$ edges, and hence has average degree at most 6, whp.*

Proof. This proof is very similar in spirit to the one above, with a couple more steps. Recall that we have $m = n/C$ edges and n nodes. The probability that some set S of size k gets some $3k$ edges falling into it is again

$$\binom{n}{k} \cdot \binom{m}{3k} \cdot \left(\frac{k}{n}\right)^{6k} \leq \left(\frac{n\,e}{k}\right)^k \cdot \left(\frac{n\,e}{3Ck}\right)^{3k} \cdot \left(\frac{k}{n}\right)^{6k} = \left(\frac{e^4}{(3C)^3} \cdot \frac{k^2}{n^2}\right)^k.$$

Again, we used the approximations for the binomial, and that C is large enough. Now a union bound over all sizes $k \geq 2$ completes the argument. □

Bibliographic Notes: The original Balanced Allocations paper of Azar, Broder, Karlin, and Upfal uses a delicate layered induction argument via Chernoff bounds. The random graph analysis is in the paper Efficient PRAM Simulation on a Distributed Memory Machine by Karp, Luby, and Meyer auf der Heide; I learned it from Satish Rao; here are his notes.

One can get slightly better constaints than the naïve power-of-two-choices using using the brilliant ***Always-go-left*** algorithm due to How Asymmetry Helps Load Balancing by Berthold Vöcking. Here's a survey on the various proof techniques by Mitzenmacher, Sitaraman and Richa.