Recall: Bloom filter

Space efficient data structure for *approximate* membership queries.

- Keep an array T of M bits
  - initially all entries are zero.
- k hash functions: \( h_1, h_2, \ldots, h_k : U \rightarrow [M] \)
  - Assume completely random hash functions for analysis

Adding a key:
- To add a key \( x \in S \subseteq U \), set bits \( T[h_1(x)], T[h_2(x)], \ldots, T[h_k(x)] \) to 1

Let’s analyze the probability of false positives.
Bloom filter

A false positive for a query occurs when all k bits in T corresponding to a query is set.

Let $p =$ probability that a bit in T is not set

$$p = \left( 1 - \frac{1}{M} \right)^{k N} = \left( 1 - \frac{1}{M} \right)^{m} \cdot \frac{k N}{M} \approx e^{-\frac{k N}{M}}$$

Prob. of false positive = all k bits set = $(1 - p)^k$

$$\left( 1 - e^{-\frac{k N}{M}} \right)^k$$
Bloom filter

Q: What value of $k$ minimizes prob. of false positives?

Differentiate and set to 0: Take $\ln \left( \left( 1 - e^{-\frac{kn}{M}} \right)^k \right)$

$$\frac{d}{dk} \left( k \ln \left( 1 - e^{-\frac{kn}{M}} \right) \right) = -\frac{kn}{M} \ln \left( 1 - e^{-\frac{kn}{M}} \right) + \frac{k e^{-\frac{kn}{M}}}{1 - e^{-\frac{kn}{M}}} \cdot \frac{n}{M}$$

$k = \frac{M}{N} \ln(2)$ is a minima

Let $\varepsilon$ denote the prob. of false positives. Then

$$\varepsilon = \left( \frac{1}{2} \right)^{\frac{M}{N} \ln 2} \frac{\frac{M}{N} \ln 2}{2^{\frac{M}{N} \ln 2} - 1} = \log \left( \frac{1}{\varepsilon} \right)$$

$$\Rightarrow M = 1.44 N \log \left( \frac{1}{\varepsilon} \right)$$
Bloom filter

Thus

\[ M = 1.44 N \log \left( \frac{1}{\epsilon} \right) \]

1.44 \log \left( \frac{1}{\epsilon} \right) \text{ bits per element}

E.g.: For 1% false positive probability, \( M \approx 10N \) bits and \( k = 7 \). Significantly smaller space than \( N \times \log(|U|) \) required to store the elements.
15-750: Graduate Algorithms

Hashing:
- Hash function basics and some constructions
- Hash tables:
  - Separate chaining
  - Open addressing
  - Cuckoo hashing
- Bloom filters

Load balancing (balls and bins)
- Concentration bounds
- Power-of-two choices
Applications: Load balancing

An important application of hashing.

N jobs, M machines.

Consider N = M.

Then, we can allocate jobs to machines such that load is 1 on each machine.

Can hash jobs onto machines.

Now there will be additional load on some machines due to randomness.

Question of interest: how high will be the load on maximally loaded machine?
Applications: Load balancing

The famous balls and bins problem:
N balls and N bins
Randomly put balls into bins

Q: Expected number of balls in each bin?
1

For studying load imbalance we need to understand the number of balls in the maximally loaded bin.
Load balancing

**Theorem:** The max-loaded bin has $O\left(\frac{\log N}{\log \log N}\right)$ balls with probability at least $1 - 1/N$.

**Proof.** High level steps:
1. We will first look at probability of any particular bin receiving more than $O\left(\frac{\log N}{\log \log N}\right)$ balls.
2. Then we will look at the probability of there being at least one bin with more than these many balls.

Q: What should the probability for Step 2 be?

At most $1/N$
Load balancing

**Theorem:** The max-loaded bin has $O\left( \frac{\log N}{\log \log N} \right)$ balls with probability at least $1 - \frac{1}{N}$.

**Proof.** **High level steps:**
1. First look at probability of any particular bin receiving more than $O\left( \frac{\log N}{\log \log N} \right)$ balls.
2. Then look at the probability of there being at least one bin with these many balls.
   (Want this to be at most $1/N$)

Q: What should the answer to Step 1 be?

**Hint:** Union bound
At most $1/N^2$ (Can use union bound over all bins)
Theorem: The max-loaded bin has $O\left(\frac{\log N}{\log \log N}\right)$ balls with probability at least $1 - \frac{1}{N}$.

Proof. High level steps:
1. First look at probability of any particular bin receiving more than $O\left(\frac{\log N}{\log \log N}\right)$ balls.
Assume fully random hash functions

$P(\text{bin } i \text{ has at least } k \text{ balls})$ is

\[
P \leq \binom{n}{k} \left(\frac{1}{n}\right)^k
\]

\[
= \frac{n!}{(n-k)!k!} \cdot \frac{1}{n^k}
\]

\[
\leq \frac{n^k}{k!} \cdot \frac{1}{n^k} \leq \frac{1}{k!}
\]
Load balancing

Theorem: The max-loaded bin has \( O\left(\frac{\log N}{\log \log N}\right) \) balls with probability at least \( 1 - 1/N \).

Proof.

\( P \) (bin \( i \) has at least \( k \) balls) is \( \leq \frac{1}{k!} \)

(Want this to be at most \( 1/N^2 \))

Using Sterling’s approximation:

\[
k! \sim \sqrt{2\pi k} \left(\frac{k}{e}\right)^k
\]

and choosing \( k = O\left(\frac{\log N}{\log \log N}\right) \) gives the desired result
Load balancing

Here we assumed fully random hash functions. But only thing used in the proof is that $k = \Theta\left(\frac{\log N}{\log \log N}\right)$ balls are independent.

So $k = \Theta\left(\frac{\log N}{\log \log N}\right)$ wise independence suffices.

But this is also quite expensive. What would happen if we use more relaxed hash families?

To analyze these it is useful to look at concentration bounds. A detour ..
Markov’s Inequality

The most basic concentration bound.

Let $X$ be a non-negative R.V. with mean $\mu$ then

$$P(X \geq \alpha) \leq \frac{\mu}{\alpha}$$

Proof: (Did last class)

In other terms,

$$P(X \geq k\mu) \leq \frac{1}{k}$$

Uses expectation only
Chebyshev’s Inequality

More powerful than Markov’s

Let $X$ be a R.V. with mean $\mu$ and variance $\sigma^2$

$$P(|X - \mu| \geq \varepsilon) \leq \frac{\sigma^2}{\varepsilon^2}$$

Proof: Ideas?

In other terms,

$$P(|X - \mu| \geq k\sigma) \leq \frac{1}{k^2}$$

Stronger since it uses variance information
Smaller the variance more concentrated the R.V. around mean
Using Chebyshev in load balancing

Suppose a 2-wise-independent hash family \( H \)

Same set up: map \( N \) balls into \( N \) bins

Lemma: maximum load over all bins is \( O(\sqrt{N}) \) w.p. at least \( \frac{1}{2} \)

\[ \text{Proof:} \]

Consider bin \( i \)

Let \( L_i = \text{load on bin } i \)

\[ L_i = \sum_{j=1}^{N} X_{ij} \]

where \( X_{ij} \in \{0,1\} \)

\[ = \begin{cases} 1 & \text{if } j^{\text{th}} \text{ ball mapped to bin } i \end{cases} \]

\[ \mu_i = E[L_i] = 1 \]

\[ \sigma_i^2 = \text{Var}[L_i] = \sum_{j=1}^{N} \text{Var}(X_{ij}) \]

\( (-1 \text{ of pairwise independence}) \)
Using Chebyshev in load balancing

\[ \begin{align*}
\text{Proof (cont.):} & \quad \frac{\sigma_i^2}{\sum_{j=1}^{N} \text{Var}(X_{ij})} = \frac{1}{N} \left( \frac{1}{N} - \frac{1}{N^2} \right) = 1 - \frac{1}{N} \\
& \\
By \text{Chebyshev,} & \quad P\left( |L_i - 1| > \sqrt{2N} \sigma_i \right) \leq \frac{1}{2N} \\
& \\
\text{Union bound gives the result.} & 
\end{align*} \]
Using Chebyshev in load balancing

Now what if we have p-wise-independent hash family H?

Higher-moment Chebyshev:

Let \( X \) be a R.V. with mean \( \mu \)

\[
P(|X - \mu| \geq \varepsilon) \leq \frac{E[(X - E[X])^p]}{\varepsilon^p}
\]
Chernoff Bound

For any R.V. X, for any $t > 0$

$$P(X \geq a) \leq \frac{E[e^{tx}]}{e^{ta}}$$

$$\Rightarrow P(X \geq a) \leq \min_{t > 0} \frac{E[e^{tx}]}{e^{ta}}$$

There are many different variants of Chernoff bounds applied to various different distributions
Hoeffding Bound

Hoeffding bound is a generalization of Chernoff bound

**Hoeffding bound:**
Let $X_i$’s be independent R.V.s taking values in $[0,1]$. Let $X = X_1 + X_2 + \ldots + X_n$
Let $\mu = E[X]$

\[
P(X > \mu + \lambda) \leq e^{-\frac{\lambda^2}{2\mu + \lambda}}
\]

\[
P(X < \mu - \lambda) \leq e^{-\frac{\lambda^2}{3\mu}}
\]
Hoeffding bound:
Let $X_i$’s be independent R.V.s taking values in $[0,1]$.
Let $X = X_1 + X_2 + \ldots + X_n$
Let $\mu = E[X]$

$$P(X > \mu + \lambda) \leq e^{-\frac{\lambda^2}{2\mu+\lambda}}$$

Q: Put $\lambda = c\mu$. How does it compare with Markov and Chebysev?
Exponential decay! Much much much stronger.
Using Hoeffding in load balancing

**Theorem:** The max-loaded bin has \( \tilde{O}\left(\frac{\log N}{\log \log N}\right) \) balls with probability at least \( 1 - \frac{1}{N} \).

**Proof 1.**

\( P(\text{bin } i \text{ has at least } k \text{ balls}) \leq \frac{1}{k!} \)

Using Sterling’s approximation and choosing

\( k = \tilde{O}\left(\frac{\log N}{\log \log N}\right) \) gives the desired result

**Proof 2.**

Can also prove this result using the Hoeffding bound.

Q: Ideas what R. V. will we use?
Using Hoeffding in load balancing

Recall $L_i = \sum_{j=1}^{N} x_{ij}$ where $x_{ij} \in (0, 1)$

$\mu = 1$

By Hoeffding,

$$P(L_i > \mu + \lambda) \leq e^{-\frac{\lambda^2}{2\mu + \lambda}}$$

Choosing $\lambda = c \log N$ gives desired result

(Think why $\log N$)
Load balancing

Another useful and interesting result. It turns out that the **bound is tight**!

**Theorem.** With high probability the max load is

\[
\Omega \left( \frac{\log n}{\log \log n} \right)
\]

Uniformly randomly placing balls into bins does not balance the load after all!
Case of having more balls than bins

\[ N > M \]

Interesting! Load imbalance reduces!

You will try this out in the programming assignment of the homework.
Load balancing: power-of-2-choice

When a ball comes in, pick two bins and place the ball in the bin with smaller number of balls.

Turns out with just checking two bins maximum number of balls drops to $O(\log \log n)$!

$=>$ called “power-of-2-choices”