Load balancing.
Today

Load balancing.
Balls in Bins.
Today

Load balancing.
Balls in Bins.
Power of two choices.
Today

Load balancing.
Balls in Bins.
Power of two choices.
Cuckoo hashing.
\[
\left( \frac{n}{k} \right)^k \leq \binom{n}{k} \leq \frac{n^k}{k!} \leq \left( \frac{ne}{k} \right)^k
\]
\[
\left( \frac{n}{k} \right)^k \leq \binom{n}{k} \leq \frac{n^k}{k!} \leq \left( \frac{ne}{k} \right)^k
\]

\[
\binom{n}{k} = \frac{n(n-1)\cdots(n-k+1)}{k(k-1)\cdots1}
\]
\[
\left( \frac{n}{k} \right)^k \leq \binom{n}{k} \leq \frac{n^k}{k!} \leq \left( \frac{ne}{k} \right)^k
\]

\[
\binom{n}{k} = \frac{n(n-1)\ldots(n-k+1)}{k(k-1)\ldots1} = \frac{n}{k} \cdot \frac{n-1}{k-1} \ldots \frac{n-k+1}{1}
\]
\[
\left( \frac{n}{k} \right)^k \leq \binom{n}{k} \leq \frac{n^k}{k!} \leq \left( \frac{ne}{k} \right)^k
\]

\[
\binom{n}{k} = \frac{n(n-1)\cdots(n-k+1)}{k(k-1)\cdots1} = \frac{n}{k} \cdot \frac{n-1}{k-1} \cdots \frac{n-k+1}{1} \geq \frac{n}{k} \cdot \frac{n}{k} \cdots \frac{n}{k}
\]
\[
\left( \frac{n}{k} \right)^k \leq \binom{n}{k} \leq \frac{n^k}{k!} \leq \left( \frac{ne}{k} \right)^k
\]

\[
\binom{n}{k} = \frac{n(n-1)\cdots(n-k+1)}{k(k-1)\cdots1} = \frac{n}{k} \cdot \frac{n-1}{k-1} \cdots \frac{n-k+1}{1} \geq \frac{n}{k} \cdot \frac{n}{k} \cdots \frac{n}{k}
\]

\[
n(n-1)\cdots(n-k+1) \leq n^k
\]
\[
\left( \frac{n}{k} \right)^k \leq \binom{n}{k} \leq \frac{n^k}{k!} \leq \left( \frac{ne}{k} \right)^k
\]

\[
\binom{n}{k} = \frac{n(n-1)\cdots(n-k+1)}{k(k-1)\cdots1} = \frac{n}{k} \cdot \frac{n-1}{k-1} \cdots \frac{n-k+1}{1} \geq \frac{n}{k} \cdot \frac{n}{k} \cdots \frac{n}{k}
\]

\[
n(n-1)\cdots(n-k+1) \leq n^k
\]

\[
k! \geq \left( \frac{k}{e} \right)^k
\]
Simplest..

Load balance: $m$ balls in $n$ bins.
Simplest..

Load balance: $m$ balls in $n$ bins.
For simplicity: $n$ balls in $n$ bins.
Simplest..

Load balance: \( m \) balls in \( n \) bins.
For simplicity: \( n \) balls in \( n \) bins.
Round robin:
Simplest..

Load balance: \( m \) balls in \( n \) bins.

For simplicity: \( n \) balls in \( n \) bins.

Round robin: load 1
Load balance: $m$ balls in $n$ bins.
For simplicity: $n$ balls in $n$ bins.
Round robin: load 1!
Simplest..

Load balance: \( m \) balls in \( n \) bins.
For simplicity: \( n \) balls in \( n \) bins.
Round robin: load 1!
Centralized!
Load balance: $m$ balls in $n$ bins.
For simplicity: $n$ balls in $n$ bins.
Round robin: load 1!
Centralized! Not so good.
Load balance: \( m \) balls in \( n \) bins.

For simplicity: \( n \) balls in \( n \) bins.

Round robin: load 1!

Centralized! Not so good.

Uniformly at random?
Load balance: \( m \) balls in \( n \) bins.
For simplicity: \( n \) balls in \( n \) bins.
Round robin: load 1!
Centralized! Not so good.
Uniformly at random? Average load
Load balance: $m$ balls in $n$ bins.
For simplicity: $n$ balls in $n$ bins.
Round robin: load 1 !
Centralized! Not so good.
Uniformly at random? Average load 1.
Load balance: \( m \) balls in \( n \) bins.
For simplicity: \( n \) balls in \( n \) bins.
Round robin: load 1!
Centralized! Not so good.
Uniformly at random? Average load 1.
Max load?
Load balance: $m$ balls in $n$ bins.
For simplicity: $n$ balls in $n$ bins.
Round robin: load 1!
Centralized! Not so good.
Uniformly at random? Average load 1.
Max load?
$n$. 
Load balance: $m$ balls in $n$ bins.
For simplicity: $n$ balls in $n$ bins.
Round robin: load 1!
Centralized! Not so good.
Uniformly at random? Average load 1.
  Max load?
  $n$. Uh Oh!
Load balance: $m$ balls in $n$ bins.
For simplicity: $n$ balls in $n$ bins.
Round robin: load 1!
Centralized! Not so good.
Uniformly at random? Average load 1.
  Max load?
  $n$. Uh Oh!
Max load with probability $\geq 1 - \delta$?
Simplest..

Load balance: \( m \) balls in \( n \) bins.
For simplicity: \( n \) balls in \( n \) bins.
Round robin: load 1!
Centralized! Not so good.
Uniformly at random? Average load 1.
  Max load?
  \( n \). Uh Oh!
Max load with probability \( \geq 1 - \delta \)?:
  \( \delta = \frac{1}{n^c} \) for today.
Simplest..

Load balance: $m$ balls in $n$ bins.
For simplicity: $n$ balls in $n$ bins.
Round robin: load 1!
Centralized! Not so good.
Uniformly at random? Average load 1.
Max load?
$n$. Uh Oh!
Max load with probability $\geq 1 - \delta$?
$\delta = \frac{1}{n^c}$ for today. $c$ is 1 or 2.
Simplest..

Load balance: $m$ balls in $n$ bins.
For simplicity: $n$ balls in $n$ bins.
Round robin: load 1!
Centralized! Not so good.
Uniformly at random? Average load 1.
Max load?
$n$. Uh Oh!
Max load with probability $\geq 1 - \delta$?
$\delta = \frac{1}{n^c}$ for today. $c$ is 1 or 2.
Balls in bins.

For each of \( n \) balls, choose random bin:

\[
\Pr[X_i \geq k] \leq \sum_{S \subseteq [n], |S| = k} \Pr[\text{balls in } S \text{ chooses bin } i]
\]

From Union Bound:

\[
\Pr[\bigcup_i A_i] \leq \sum_i \Pr[A_i]
\]

\[
\Pr[\text{balls in } S \text{ chooses bin } i] = \left(\frac{1}{n}\right)^k \binom{n}{k}
\]

Choose \( k \), so that \( \Pr[X_i \geq k] \leq \frac{1}{n^2} \).

\[
\Pr[\text{any } X_i \geq k] \leq n \times \frac{1}{n^2} = \frac{1}{n} \rightarrow \max \text{ load} \leq k \text{ w.p. } \geq 1 - \frac{1}{n^k} \geq n^2 \text{ for } k = 2 e \log n \text{ (Recall } k! \geq (ke)^k).)
\]

Lemma: Max load is \( \Theta(\log n) \) with probability \( \geq 1 - \frac{1}{n^k} \).

Much better than \( n \).

Actually Max load is \( \Theta(\log n / \log \log n) \) w.h.p. (W.h.p. - means with probability at least 1 - \( O(1/n^c) \) for today.)
Balls in bins.

For each of $n$ balls, choose random bin: $X_i$ balls in bin $i$. 

Pr $[X_i \geq k] \leq \sum_{S \subseteq \{1, 2, \ldots, n\}, |S| = k} Pr[\text{balls in } S \text{ chooses bin } i] = \left(\frac{1}{n}\right)^k \binom{n}{k} \leq n \cdot \left(\frac{1}{n}\right)^2 = \frac{1}{n}$

Choose $k$, so that $Pr[X_i \geq k] \leq \frac{1}{n^2}$.

Pr any $X_i \geq k \leq n \times \frac{1}{n^2} = \frac{1}{n} \rightarrow \text{max load} \leq k$ w.p. $\geq 1 - \frac{1}{n}$.

Much better than $n$.

Actually max load is $\Theta(\log n)$ w.h.p.

(W.h.p. - means with probability at least $1 - O\left(\frac{1}{n^c}\right)$ for today.)
Balls in bins.

For each of $n$ balls, choose random bin: $X_i$ balls in bin $i$.

$$\Pr[X_i \geq k] \leq \sum_{S \subseteq [n], |S| = k} \Pr[\text{balls in } S \text{ chooses bin } i]$$
Balls in bins.

For each of $n$ balls, choose random bin: $X_i$ balls in bin $i$.

$Pr[X_i \geq k] \leq \sum_{S \subseteq [n], |S| = k} Pr[\text{balls in } S \text{ chooses bin } i]$  

From Union Bound: $Pr[\bigcup_i A_i] \leq \sum_i Pr[A_i]$
Balls in bins.

For each of \( n \) balls, choose random bin: \( X_i \) balls in bin \( i \).

\[
Pr[X_i \geq k] \leq \sum_{S \subseteq [n], |S| = k} Pr[\text{balls in } S \text{ chooses bin } i]
\]

From Union Bound: \( Pr[\bigcup_i A_i] \leq \sum_i Pr[A_i] \)

\[
Pr[\text{balls in } S \text{ chooses bin } i] = \left( \frac{1}{n} \right)^k
\]
Balls in bins.

For each of \( n \) balls, choose random bin: \( X_i \) balls in bin \( i \).

\[
Pr[X_i \geq k] \leq \sum_{S \subseteq [n], |S| = k} Pr[\text{balls in } S \text{ chooses bin } i]
\]

From Union Bound: \( Pr[\bigcup_i A_i] \leq \sum_i Pr[A_i] \)

\[
Pr[\text{balls in } S \text{ chooses bin } i] = \left( \frac{1}{n} \right)^k \quad \text{and} \quad \binom{n}{k} \text{ subsets } S.
\]
Balls in bins.

For each of \( n \) balls, choose random bin: \( X_i \) balls in bin \( i \).

\[
Pr[X_i \geq k] \leq \sum_{S \subseteq [n], |S| = k} Pr[\text{balls in } S \text{ chooses bin } i]
\]

From Union Bound: \( Pr[\bigcup_i A_i] \leq \sum_i Pr[A_i] \)

\[
Pr[\text{balls in } S \text{ chooses bin } i] = \left(\frac{1}{n}\right)^k \quad \text{and} \quad \binom{n}{k} \text{ subsets } S.
\]

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

\[
\leq \frac{n^k}{k!} \left(\frac{1}{n}\right)^k = \frac{1}{k!}
\]

Choose \( k \), so that \( Pr[X_i \geq k] \leq \frac{1}{n^2} \).

\[
Pr[\text{any } X_i \geq k] \leq n \times \frac{1}{n^2} = \frac{1}{n} \rightarrow \text{max load} \leq k \text{ w.p. } \geq 1 - \frac{1}{n}.
\]

Lemma: Max load is \( \Theta(\log n) \) with probability \( \geq 1 - \frac{1}{n} \).

Much better than \( n \).

Actually Max load is \( \Theta(\log n / \log \log n) \) w.h.p. (W.h.p. - means with probability at least \( 1 - O\left(\frac{1}{n^{c}}\right) \) for today.)
Balls in bins.

For each of $n$ balls, choose random bin: $X_i$ balls in bin $i$.

$Pr[X_i \geq k] \leq \sum_{S \subseteq [n], |S| = k} Pr[\text{balls in } S \text{ chooses bin } i]$  

From Union Bound: $Pr[\bigcup_i A_i] \leq \sum_i Pr[A_i]$  

$Pr[\text{balls in } S \text{ chooses bin } i] = \left(\frac{1}{n}\right)^k$ and $\binom{n}{k}$ subsets $S$.

$$Pr[X_i \geq k] \leq \binom{n}{k} \left(\frac{1}{n}\right)^k$$

$$\leq \frac{n^k}{k!} \left(\frac{1}{n}\right)^k = \frac{1}{k!}$$

Choose $k$, so that $Pr[X_i \geq k] \leq \frac{1}{n^2}$. 

Lemma: Max load is $\Theta(\log n)$ with probability $\geq 1 - \frac{1}{n}$.

Much better than $n$. Actually Max load is $\Theta(\log n / \log \log n)$ w.h.p. (W.h.p. - means with probability at least $1 - O\left(\frac{1}{n^c}\right)$ for today.)
Balls in bins.

For each of \( n \) balls, choose random bin: \( X_i \) balls in bin \( i \).

\[
\Pr[X_i \geq k] \leq \sum_{S \subseteq [n], |S| = k} \Pr[\text{balls in } S \text{ chooses bin } i]
\]

From Union Bound: \( \Pr[\bigcup_i A_i] \leq \sum_i \Pr[A_i] \)

\[
\Pr[\text{balls in } S \text{ chooses bin } i] = \left( \frac{1}{n} \right)^k \quad \text{and} \quad \binom{n}{k} \text{ subsets } S.
\]

\[
\Pr[X_i \geq k] \leq \binom{n}{k} \left( \frac{1}{n} \right)^k \\
\leq \frac{n^k}{k!} \left( \frac{1}{n} \right)^k = \frac{1}{k!}
\]

Choose \( k \), so that \( \Pr[X_i \geq k] \leq \frac{1}{n^2} \).

\[
\Pr[\text{any } X_i \geq k] \leq n \times \frac{1}{n^2}
\]
Balls in bins.

For each of $n$ balls, choose random bin: $X_i$ balls in bin $i$.

$$\Pr[X_i \geq k] \leq \sum_{S \subseteq [n], |S| = k} \Pr[\text{balls in } S \text{ chooses bin } i]$$

From Union Bound: $\Pr[\bigcup_i A_i] \leq \sum_i \Pr[A_i]$.

$\Pr[\text{balls in } S \text{ chooses bin } i] = \left(\frac{1}{n}\right)^k$ and $\binom{n}{k}$ subsets $S$.

$$\Pr[X_i \geq k] \leq \binom{n}{k} \left(\frac{1}{n}\right)^k \leq \frac{n^k}{k!} \left(\frac{1}{n}\right)^k = \frac{1}{k!}$$

Choose $k$, so that $\Pr[X_i \geq k] \leq \frac{1}{n^2}$.

$$\Pr[\text{any } X_i \geq k] \leq n \times \frac{1}{n^2} = \frac{1}{n}$$
Balls in bins.

For each of \( n \) balls, choose random bin: \( X_i \) balls in bin \( i \).

\[
Pr[X_i \geq k] \leq \sum_{S \subseteq [n], |S| = k} Pr[\text{balls in } S \text{ chooses bin } i]
\]

From Union Bound: \( Pr[\cup_i A_i] \leq \sum_i Pr[A_i] \)

\[
Pr[\text{balls in } S \text{ chooses bin } i] = \left( \frac{1}{n} \right)^k \quad \text{and} \quad \binom{n}{k} \text{ subsets } S.
\]

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

\[
\leq \frac{n^k}{k!} \left( \frac{1}{n} \right)^k = \frac{1}{k!}
\]

Choose \( k \), so that \( Pr[X_i \geq k] \leq \frac{1}{n^2} \).

\[
Pr[\text{any } X_i \geq k] \leq n \times \frac{1}{n^2} = \frac{1}{n} \rightarrow \text{max load } \leq k \text{ w.p. } \geq 1 - \frac{1}{n}
\]
Balls in bins.

For each of $n$ balls, choose random bin: $X_i$ balls in bin $i$.

$$\Pr[X_i \geq k] \leq \sum_{S \subseteq [n], |S| = k} \Pr[\text{balls in } S \text{ chooses bin } i]$$

From Union Bound: $\Pr[\bigcup_i A_i] \leq \sum_i \Pr[A_i]$.

$\Pr[\text{balls in } S \text{ chooses bin } i] = \left(\frac{1}{n}\right)^k$ and $\binom{n}{k}$ subsets $S$.

$$\Pr[X_i \geq k] \leq \binom{n}{k} \left(\frac{1}{n}\right)^k$$

$$\leq \frac{n^k}{k!} \left(\frac{1}{n}\right)^k = \frac{1}{k!}$$

Choose $k$, so that $\Pr[X_i \geq k] \leq \frac{1}{n^2}$.

$$\Pr[\text{any } X_i \geq k] \leq n \times \frac{1}{n^2} = \frac{1}{n} \rightarrow \text{max load } \leq k \text{ w.p. } \geq 1 - \frac{1}{n}$$

$k! \geq n^2$ for $k = 2e \log n$ (Recall $k! \geq \left(\frac{k}{e}\right)^k$.)

Lemma: Max load is $\Theta(\log n)$ with probability $\geq 1 - \frac{1}{n}$.

Much better than $n$.

Actually Max load is $\Theta(\log n / \log \log n)$ w.h.p. (W.h.p. - means with probability at least $1 - O\left(\frac{1}{n^c}\right)$ for today.)
Balls in bins.

For each of $n$ balls, choose random bin: $X_i$ balls in bin $i$.

$$Pr[X_i \geq k] \leq \sum_{S \subseteq [n], |S| = k} Pr[\text{balls in } S \text{ chooses bin } i]$$

From Union Bound: $Pr[\bigcup_i A_i] \leq \sum_i Pr[A_i]$

$$Pr[\text{balls in } S \text{ chooses bin } i] = \left(\frac{1}{n}\right)^k \text{ and } \binom{n}{k} \text{ subsets } S.$$ 

$$Pr[X_i \geq k] \leq \binom{n}{k} \left(\frac{1}{n}\right)^k$$

$$\leq \frac{n^k}{k!} \left(\frac{1}{n}\right)^k = \frac{1}{k!}$$

Choose $k$, so that $Pr[X_i \geq k] \leq \frac{1}{n^2}$.

$$Pr[\text{any } X_i \geq k] \leq n \times \frac{1}{n^2} = \frac{1}{n} \rightarrow \max \text{ load } \leq k \text{ w.p. } \geq 1 - \frac{1}{n}$$

$k! \geq n^2$ for $k = 2e \log n$ (Recall $k! \geq \left(\frac{k}{e}\right)^k$.)

Lemma: Max load is $\Theta(\log n)$ with probability $\geq 1 - \frac{1}{n}$. 

Much better than $n$.

Actually Max load is $\Theta(\log n / \log \log n)$ w.h.p. (W.h.p. - means with probability at least $1 - O\left(\frac{1}{n^{c}}\right)$ for today.)
Balls in bins.

For each of $n$ balls, choose random bin: $X_i$ balls in bin $i$.

$$
Pr[X_i \geq k] \leq \sum_{S \subseteq [n], |S|=k} Pr[\text{balls in } S \text{ chooses bin } i]
$$

From Union Bound: $Pr[\bigcup_i A_i] \leq \sum_i Pr[A_i]$

$$
Pr[\text{balls in } S \text{ chooses bin } i] = \left(\frac{1}{n}\right)^k \text{ and } \binom{n}{k} \text{ subsets } S.
$$

$$
Pr[X_i \geq k] \leq \binom{n}{k} \left(\frac{1}{n}\right)^k
\leq \frac{n^k}{k!} \left(\frac{1}{n}\right)^k = \frac{1}{k!}
$$

Choose $k$, so that $Pr[X_i \geq k] \leq \frac{1}{n^2}$.

$$
Pr[\text{any } X_i \geq k] \leq n \times \frac{1}{n^2} = \frac{1}{n} \rightarrow \text{max load } \leq k \text{ w.p. } \geq 1 - \frac{1}{n}
$$

$k! \geq n^2$ for $k = 2e \log n$ (Recall $k! \geq \left(\frac{k}{e}\right)^k$.)

**Lemma:** Max load is $\Theta(\log n)$ with probability $\geq 1 - \frac{1}{n}$.

Much better than $n$. 

Balls in bins.

For each of $n$ balls, choose random bin: $X_i$ balls in bin $i$.

$Pr[X_i \geq k] \leq \sum_{S \subseteq [n], |S| = k} Pr[\text{balls in } S \text{ chooses bin } i]$,

From Union Bound: $Pr[\cup_i A_i] \leq \sum_i Pr[A_i]$

$Pr[\text{balls in } S \text{ chooses bin } i] = \left(\frac{1}{n}\right)^k$ and $\binom{n}{k}$ subsets $S$.

$Pr[X_i \geq k] \leq \binom{n}{k} \left(\frac{1}{n}\right)^k$

$\leq \frac{n^k}{k!} \left(\frac{1}{n}\right)^k = \frac{1}{k!}$

Choose $k$, so that $Pr[X_i \geq k] \leq \frac{1}{n^2}$.

$Pr[\text{any } X_i \geq k] \leq n \times \frac{1}{n^2} = \frac{1}{n} \rightarrow \text{max load } \leq k \text{ w.p. } \geq 1 - \frac{1}{n}$

$k! \geq n^2$ for $k = 2e \log n$ (Recall $k! \geq (\frac{k}{e})^k$.)

**Lemma:** Max load is $\Theta(\log n)$ with probability $\geq 1 - \frac{1}{n}$.

Much better than $n$.

Actually Max load is $\Theta(\log n / \log \log n)$ w.h.p.
Balls in bins.

For each of $n$ balls, choose random bin: $X_i$ balls in bin $i$.

$$Pr[X_i \geq k] \leq \sum_{S \subseteq [n], |S|=k} Pr[\text{balls in } S \text{ chooses bin } i]$$

From Union Bound: $Pr[\bigcup_i A_i] \leq \sum_i Pr[A_i]$.

$$Pr[\text{balls in } S \text{ chooses bin } i] = \left(\frac{1}{n}\right)^k \text{ and } \left(\begin{array}{c} n \\ k \end{array}\right) \text{ subsets } S.$$ 

$$Pr[X_i \geq k] \leq \left(\begin{array}{c} n \\ k \end{array}\right) \left(\frac{1}{n}\right)^k$$

$$\leq \frac{n^k}{k!} \left(\frac{1}{n}\right)^k = \frac{1}{k!}$$

Choose $k$, so that $Pr[X_i \geq k] \leq \frac{1}{n^2}$.

$$Pr[\text{any } X_i \geq k] \leq n \times \frac{1}{n^2} = \frac{1}{n} \rightarrow \text{max load } \leq k \text{ w.p. } \geq 1 - \frac{1}{n}$$

$k! \geq n^2$ for $k = 2e\log n$ (Recall $k! \geq (\frac{k}{e})^k$.)

Lemma: Max load is $\Theta(\log n)$ with probability $\geq 1 - \frac{1}{n}$.

Much better than $n$.

Actually Max load is $\Theta(\log n / \log\log n)$ w.h.p.

(W.h.p. - means with probability at least $1 - O(1/n^c)$ for today.)
Power of two..

$n$ balls in $n$ bins.
$n$ balls in $n$ bins.
Choose two bins, pick least loaded.
$n$ balls in $n$ bins.

Choose two bins, pick least loaded.

still distributed, but a bit less than not looking.
Power of two..

$n$ balls in $n$ bins.

Choose two bins, pick least loaded.

still distributed, but a bit less than not looking.

Is max load lower?
Power of two..

$n$ balls in $n$ bins.

Choose two bins, pick least loaded.

still distributed, but a bit less than not looking.

Is max load lower? Yes?

$O(\log \log n)$!
$n$ balls in $n$ bins.
Choose two bins, pick least loaded.
still distributed, but a bit less than not looking.
Is max load lower? Yes? No?
Power of two..

$n$ balls in $n$ bins.
Choose two bins, pick least loaded.
    still distributed, but a bit less than not looking.
$n$ balls in $n$ bins.

Choose two bins, pick least loaded.
   still distributed, but a bit less than not looking.
   How much lower?

$\log n / 2$?
$\sqrt{\log n}$?
$O(\log \log n)$?

Exponentially better!
Old bound is exponential of new bound.
Power of two..

$n$ balls in $n$ bins.
Choose two bins, pick least loaded.
still distributed, but a bit less than not looking.
How much lower?
log $n/2$?
Power of two..

$n$ balls in $n$ bins.
Choose two bins, pick least loaded.
    still distributed, but a bit less than not looking.
    How much lower?
        $\log n/2$? $\sqrt{\log n}$?
$n$ balls in $n$ bins.
Choose two bins, pick least loaded.
still distributed, but a bit less than not looking.
How much lower?
$\log n/2$? $\sqrt{\log n}$? $O(\log \log n)$?
\( n \) balls in \( n \) bins.

Choose two bins, pick least loaded.

still distributed, but a bit less than not looking.


How much lower?

\( \log n/2? \sqrt{\log n}? O(\log \log n)? \)

\( O(\log \log n) \)
$n$ balls in $n$ bins.
Choose two bins, pick least loaded.

still distributed, but a bit less than not looking.


How much lower?

$\log n/2$? $\sqrt{\log n}$? $O(\log \log n)$?

$O(\log \log \log n)$!
$n$ balls in $n$ bins.
Choose two bins, pick least loaded.

still distributed, but a bit less than not looking.


How much lower?

$\log n/2$? $\sqrt{\log n}$? $O(\log \log n)$?

$O(\log \log n)$ !!
$n$ balls in $n$ bins.

Choose two bins, pick least loaded.

still distributed, but a bit less than not looking.


How much lower?

$\log n/2$? $\sqrt{\log n}$? $O(\log \log n)$?

$O(\log \log n)$ ! ! !
Power of two..

$n$ balls in $n$ bins.

Choose two bins, pick least loaded.

still distributed, but a bit less than not looking.


How much lower?

$log n/2$? $\sqrt{\log n}$? $O(\log \log n)$?

$O(\log \log n)$ !!!!
$n$ balls in $n$ bins.

Choose two bins, pick least loaded.

still distributed, but a bit less than not looking.


How much lower?

$\log n/2$? $\sqrt{\log n}$? $O(\log \log n)$?

$O(\log \log n)$ ! ! ! !

Exponentially better!
$n$ balls in $n$ bins.

Choose two bins, pick least loaded.

still distributed, but a bit less than not looking.


How much lower?

$\log n/2$? $\sqrt{\log n}$? $O(\log \log n)$?

$O(\log \log n)$ ! ! ! ! !

Exponentially better! Old bound is exponential of new bound.
Analysis.

$n/8$ balls in $n$ bins.
Analysis.

\(n/8\) balls in \(n\) bins.

Each ball chooses two bins at random.
Analysis.

\( \frac{n}{8} \) balls in \( n \) bins.

Each ball chooses two bins at random.
  picks least loaded.
Analysis.
n/8 balls in n bins.
Each ball chooses two bins at random.
   picks least loaded.
View as graph.
Analysis.

$n/8$ balls in $n$ bins.

Each ball chooses two bins at random.
    picks least loaded.

View as graph.
Bin is vertex.
Analysis.

$n/8$ balls in $n$ bins.

Each ball chooses two bins at random.
  picks least loaded.

View as graph.
Bin is vertex.
Each ball is edge.
Analysis.

$n/8$ balls in $n$ bins.

Each ball chooses two bins at random.
  picks least loaded.

View as graph.
  Bin is vertex.
  Each ball is edge.

Analysis Intuition:
Add edge, add one to lower endpoint's "count."
Max load is max vertices count.
If max count is $k$.
neighbors with counts $\geq k - 1$,
$k - 2$,
$k - 3$,...
and so on!
No cycles and max-load $k \rightarrow \geq 2k/2$ nodes in tree.
No connected component of size $X$ and no cycles $\Rightarrow$ max load $O(\log X)$.

Will show:
Max conn. comp is $O(\log n)$ w.h.p.
Average induced degree is small. (E.g.: cycle degree 2)
Extend tree intuition.
Analysis.

\[ \frac{n}{8} \text{ balls in } n \text{ bins.} \]

Each ball chooses two bins at random. 
- picks least loaded.

View as graph.
- Bin is vertex.
- Each ball is edge.

Analysis Intuition:
Add edge, add one to lower endpoint's "count."
Max load is max vertices count.
If max count is \( k \):
- neighbors with counts \( \geq k - 1 \), \( k - 2 \), \( k - 3 \),... and so on!

No cycles and max-load \( k \rightarrow \geq \frac{2k}{2} \) nodes in tree.

No connected component of size \( X \) and no cycles \( \Rightarrow \max \text{ load } O(\log X) \).

Will show:
- Max conn. comp is \( O(\log n) \) w.h.p.
- Average induced degree is small. (E.g.: cycle degree 2)

Extend tree intuition.
Analysis.

\( n/8 \) balls in \( n \) bins.

Each ball chooses two bins at random.

- picks least loaded.

View as graph.

- Bin is vertex.
- Each ball is edge.

Analysis Intuition:

Add edge, add one to lower endpoint's "count."

- Max load is max vertices count.

If max count is \( k \),

- neighbors with counts \( \geq k - 1 \),
- \( k - 2 \),
- \( k - 3 \),...

and so on!

No cycles and max-load \( k \rightarrow \geq 2 \frac{k}{2} \) nodes in tree.

No connected component of size \( X \) and no cycles \( \Rightarrow \) max load \( O(\log X) \).

Will show:

- Max conn. comp is \( O(\log n) \) w.h.p.

Average induced degree is small. (E.g.: cycle degree 2)

Extend tree intuition.
Analysis.

\( n/8 \) balls in \( n \) bins.

Each ball chooses two bins at random.
   picks least loaded.

View as graph.
  Bin is vertex.
  Each ball is edge.

Analysis Intuition:
Add edge, add one to lower endpoint’s “count.”
Analysis.

\( n/8 \) balls in \( n \) bins.

Each ball chooses two bins at random.
    picks least loaded.

View as graph.
    Bin is vertex.
    Each ball is edge.

Analysis Intuition:
Add edge, add one to lower endpoint’s “count.”

Max load is max vertices count.
Analysis.

\( n/8 \) balls in \( n \) bins.

Each ball chooses two bins at random.
   picks least loaded.

View as graph.
   Bin is vertex.
   Each ball is edge.

Analysis Intuition:
Add edge, add one to lower endpoint’s “count.”

Max load is max vertices count.
   If max count is \( k \).

\[ \text{Max load} \geq \frac{k}{2} \]

No connected component of size \( X \) and no cycles \( \Rightarrow \) max load \( O(\log n) \).

Will show:
Max connected comp is \( O(\log n) \) w.h.p.

Average induced degree is small. (E.g.: cycle degree 2)

Extend tree intuition.
Analysis.

$n/8$ balls in $n$ bins.
Each ball chooses two bins at random.
  picks least loaded.

View as graph.
Bin is vertex.
Each ball is edge.

Analysis Intuition:
Add edge, add one to lower endpoint’s “count.”

Max load is max vertices count.
If max count is $k$.
  neighbors with counts $\geq k-1, k-2, k-3, \ldots$. 

=\text{max load} \leq 2k / 2 = k.$$
Analysis.

$n/8$ balls in $n$ bins.

Each ball chooses two bins at random. picks least loaded.

View as graph.
Bin is vertex.
Each ball is edge.

Analysis Intuition:
Add edge, add one to lower endpoint’s “count.”

Max load is max vertices count.
If max count is $k$.
neighbors with counts $\geq k-1, k-2, k-3, \ldots$ and so on!
Analysis.

\( n/8 \) balls in \( n \) bins.

Each ball chooses two bins at random.
  picks least loaded.

View as graph.
  Bin is vertex.
  Each ball is edge.

Analysis Intuition:
Add edge, add one to lower endpoint’s “count.”

Max load is max vertices count.
  If max count is \( k \).
    neighbors with counts \( \geq k - 1, k - 2, k - 3, \ldots \)
    and so on!

No cycles and max-load \( k \rightarrow \geq 2^{k/2} \) nodes in tree.
Analysis.

\( n/8 \) balls in \( n \) bins.

Each ball chooses two bins at random.
- picks least loaded.

View as graph.
- Bin is vertex.
- Each ball is edge.

Analysis Intuition:
Add edge, add one to lower endpoint’s “count.”

Max load is max vertices count.
- If max count is \( k \).
  - neighbors with counts \( \geq k - 1, k - 2, k - 3, \ldots \) and so on!

No cycles and max-load \( k \to \geq 2^{k/2} \) nodes in tree.
No connected component of size \( X \) and no cycles
Analysis.

$n/8$ balls in $n$ bins.

Each ball chooses two bins at random.
  picks least loaded.

View as graph.
  Bin is vertex.
  Each ball is edge.

Analysis Intuition:
Add edge, add one to lower endpoint’s “count.”

Max load is max vertices count.
  If max count is $k$.
    neighbors with counts $\geq k - 1, k - 2, k - 3, \ldots$ and so on!

No cycles and max-load $k \rightarrow \geq 2^{k/2}$ nodes in tree.
No connected component of size $X$ and no cycles $\implies$ max load $O(\log X)$. 
Analysis.

$n/8$ balls in $n$ bins.
Each ball chooses two bins at random.
picks least loaded.

View as graph.
Bin is vertex.
Each ball is edge.

Analysis Intuition:
Add edge, add one to lower endpoint’s “count.”

Max load is max vertices count.
If max count is $k$.
neighbors with counts $\geq k - 1, k - 2, k - 3, \ldots$ and so on!

No cycles and max-load $k \rightarrow \geq 2^{k/2}$ nodes in tree.
No connected component of size $X$ and no cycles
$\implies$ max load $O(\log X)$.

Will show:
Analysis.

$n/8$ balls in $n$ bins.
Each ball chooses two bins at random.
picks least loaded.

View as graph.
Bin is vertex.
Each ball is edge.

Analysis Intuition:
Add edge, add one to lower endpoint’s “count.”

Max load is max vertices count.
If max count is $k$.
neighbors with counts $\geq k - 1, k - 2, k - 3, \ldots$ and so on!

No cycles and max-load $k \rightarrow \geq 2^{k/2}$ nodes in tree.
No connected component of size $X$ and no cycles\implies max load $O(\log X)$.

Will show:
Max conn. comp is $O(\log n)$ w.h.p.
Analysis.

\(\frac{n}{8}\) balls in \(n\) bins.

Each ball chooses two bins at random.
   picks least loaded.

View as graph.
   Bin is vertex.
   Each ball is edge.

Analysis Intuition:
Add edge, add one to lower endpoint’s “count.”

Max load is max vertices count.
   If max count is \(k\).
      neighbors with counts \(\geq k - 1, k - 2, k - 3, \ldots\)
         and so on!

No cycles and max-load \(k \rightarrow \geq 2^{k/2}\) nodes in tree.
No connected component of size \(X\) and no cycles
   \(\implies\) max load \(O(\log X)\).

Will show:
   Max conn. comp is \(O(\log n)\) w.h.p.
   Average induced degree is small. (E.g.: cycle degree 2)
Analysis.

\( n/8 \) balls in \( n \) bins.

Each ball chooses two bins at random.
   picks least loaded.

View as graph.
   Bin is vertex.
   Each ball is edge.

Analysis Intuition:
Add edge, add one to lower endpoint’s “count.”

Max load is max vertices count.
   If max count is \( k \).
       neighbors with counts \( \geq k - 1, k - 2, k - 3, \ldots \)
           and so on!

No cycles and max-load \( k \rightarrow \geq 2^{k/2} \) nodes in tree.
No connected component of size \( X \) and no cycles
\( \implies \) max load \( O(\log X) \).

Will show:
   Max conn. comp is \( O(\log n) \) w.h.p.
   Average induced degree is small. (E.g.: cycle degree 2)

Extend tree intuition.
**Claim:** Component size in $n$ vertex, $\frac{n}{8}$ edge random graph is $O(\log n)$ w/ prob. $\geq 1 - \frac{1}{nc}$.

**Proof:** Size $k$ component, $C$, contains $\geq k - 1$ edges.

$$\Pr[|C| \geq k] \leq \binom{n}{k} \left( \frac{n}{8} \right)^{k-1} \left( \frac{k}{n} \right)^{2(k-1)}$$  \hspace{1cm} (1)
**Claim:** Component size in $n$ vertex, $\frac{n}{8}$ edge random graph is $O(\log n)$ w/ prob. $\geq 1 - \frac{1}{n^c}$.

**Proof:** Size $k$ component, $C$, contains $\geq k - 1$ edges.

$$\Pr[|C| \geq k] \leq \binom{n}{k} \binom{n/8}{k-1} \left(\frac{k}{n}\right)^{2(k-1)} \quad (1)$$

Possible $C$. 

Connected Component.
Claim: Component size in $n$ vertex, $\frac{n}{8}$ edge random graph is $O(\log n)$ w/ prob. $\geq 1 - \frac{1}{nc}$.

Proof: Size $k$ component, $C$, contains $\geq k - 1$ edges.

$$\Pr[|C| \geq k] \leq \binom{n}{k} \left( \frac{n/8}{k-1} \right) \left( \frac{k}{n} \right)^{2(k-1)}$$

Possible $C$. Which edges.
Connected Component.

**Claim:** Component size in $n$ vertex, $\frac{n}{8}$ edge random graph is $O(\log n)$ w/ prob. $\geq 1 - \frac{1}{n^c}$.

Proof: Size $k$ component, $C$, contains $\geq k - 1$ edges.

\[
\Pr[|C| \geq k] \leq \binom{n}{k} \binom{n/8}{k-1} \left( \frac{k}{n} \right)^{2(k-1)}
\]  
(1)

Possible $C$. Which edges. Prob. both endpoints inside $C$. 

Connected Component.

**Claim:** Component size in \( n \) vertex, \( \frac{n}{8} \) edge random graph is \( O(\log n) \) w/ prob. \( \geq 1 - \frac{1}{n^c} \).

**Proof:** Size \( k \) component, \( C \), contains \( \geq k - 1 \) edges.

\[
\Pr[|C| \geq k] \leq \binom{n}{k} \binom{n/8}{k-1} \left( \frac{k}{n} \right)^{2(k-1)}
\]

Possible \( C \). Which edges. Prob. both endpoints inside \( C \).

\[
\Pr[|C| \geq k] \leq \frac{n}{k} \binom{n}{k} \binom{n/8}{k} \left( \frac{k}{n} \right)^{2k}
\]

pause
Connected Component.

**Claim:** Component size in $n$ vertex, $\frac{n}{8}$ edge random graph is $O(\log n)$ w/ prob. $\geq 1 - \frac{1}{n^c}$.

**Proof:** Size $k$ component, $C$, contains $\geq k - 1$ edges.

$$Pr[|C| \geq k] \leq \binom{n}{k} \binom{n/8}{k-1} \left(\frac{k}{n}\right)^{2(k-1)} \tag{1}$$

Possible $C$. Which edges. Prob. both endpoints inside $C$.

$$Pr[|C| \geq k] \leq \frac{n}{k} \binom{n}{k} \binom{n/8}{k} \left(\frac{k}{n}\right)^{2k}$$

$$\leq \frac{n}{k} \left(\frac{ne}{k}\right)^k \left(\frac{ne}{8k}\right)^k \left(\frac{k}{n}\right)^{2k}$$
Connected Component.

Claim: Component size in $n$ vertex, $\frac{n}{8}$ edge random graph is $O(\log n)$ w/ prob. $\geq 1 - \frac{1}{n^c}$.

Proof: Size $k$ component, $C$, contains $\geq k - 1$ edges.

\[
\Pr[|C| \geq k] \leq \binom{n}{k} \binom{n/8}{k-1} \left( \frac{k}{n} \right)^{2(k-1)}
\]

(1)

Possible $C$. Which edges. Prob. both endpoints inside $C$.

\[
\Pr[|C| \geq k] \leq \frac{n}{k} \binom{n}{k} \binom{n/8}{k-1} \left( \frac{k}{n} \right)^{2k}
\]

\[
\leq \frac{n}{k} (\frac{ne}{k})^k (\frac{ne}{8k})^k \left( \frac{k}{n} \right)^{2k} = \frac{n}{k} \left( \frac{e^2}{8} \right)^k
\]
Connected Component.

**Claim:** Component size in \( n \) vertex, \( \frac{n}{8} \) edge random graph is \( O(\log n) \) w/ prob. \( \geq 1 - \frac{1}{n^c} \).

**Proof:** Size \( k \) component, \( C \), contains \( \geq k - 1 \) edges.

\[
\Pr[|C| \geq k] \leq \binom{n}{k} \binom{n/8}{k-1} \left(\frac{k}{n}\right)^{2(k-1)}
\]

Possible \( C \). Which edges. Prob. both endpoints inside \( C \).

\[
\Pr[|C| \geq k] \leq \frac{n}{k} \binom{n}{k} \binom{n/8}{k} \left(\frac{k}{n}\right)^{2k}
\]
\[
\leq \frac{n}{k} \left(\frac{ne}{k}\right)^k \left(\frac{ne}{8k}\right)^k \left(\frac{k}{n}\right)^{2k} = \frac{n}{k} \left(\frac{e^2}{8}\right)^k \leq \frac{n}{k} (0.93)^k
\]

Choose \( k = -(c+1) \log_{0.93} n \) make probability \( \leq 1/n^c \).
Not dense.

Induced degree of node on subset, $S$, is degree of internal edges.
Not dense.

Induced degree of node on subset, $S$, is degree of internal edges.
Induced degree of node on subset, $S$, is degree of internal edges.

Induced degree of nodes in blue subset is 2,
Not dense.

Induced degree of node on subset, $S$, is degree of internal edges.

Induced degree of nodes in blue subset is 2, not 5!
Not dense.

Induced degree of node on subset, $S$, is degree of internal edges.

Induced degree of nodes in blue subset is 2, not 5!

**Claim:** Average induced degree on any subset of nodes is $\leq 8$ with probability $\geq 1 - O\left(\frac{1}{n^2}\right)$. 
Not dense.

Induced degree of node on subset, $S$, is degree of internal edges.

Induced degree of nodes in blue subset is 2, not 5!

**Claim:** Average induced degree on any subset of nodes is $\leq 8$ with probability $\geq 1 - O(\frac{1}{n^2})$.

**Proof:** Induced degree $\geq 8$
Not dense.

Induced degree of node on subset, $S$, is degree of internal edges.

Induced degree of nodes in blue subset is 2, not 5!

**Claim:** Average induced degree on any subset of nodes is $\leq 8$ with probability $\geq 1 - O\left(\frac{1}{n^2}\right)$.

**Proof:** Induced degree $\geq 8$
Not dense.

Induced degree of node on subset, $S$, is degree of internal edges.

Induced degree of nodes in blue subset is 2, not 5!

**Claim:** Average induced degree on any subset of nodes is $\leq 8$ with probability $\geq 1 - O\left(\frac{1}{n^2}\right)$.

**Proof:** Induced degree $\geq 8$

$\rightarrow 4k$ internal edges for subset of size $k$. 
Not dense.

Induced degree of node on subset, $S$, is degree of internal edges.

Induced degree of nodes in blue subset is 2, not 5!

**Claim:** Average induced degree on any subset of nodes is $\leq 8$ with probability $\geq 1 - O\left(\frac{1}{n^2}\right)$.

**Proof:** Induced degree $\geq 8$  
$\rightarrow 4k$ internal edges for subset of size $k$.

$$\Pr[\text{dense } S] \leq \binom{n}{k} \binom{n/8}{4k} \left(\frac{k}{n}\right)^{8k} \leq \left(\frac{e^{1.25}}{32}\right)^{4k} \left(\frac{k}{n}\right)^{3k} \leq \left(\frac{k}{n}\right)^{3k}$$
Not dense.

Induced degree of node on subset, $S$, is degree of internal edges.

Induced degree of nodes in blue subset is 2, not 5!

**Claim:** Average induced degree on any subset of nodes is $\leq 8$ with probability $\geq 1 - O\left(\frac{1}{n^2}\right)$.

**Proof:** Induced degree $\geq 8$

$\rightarrow 4k$ internal edges for subset of size $k$.

$$\Pr[\text{dense } S] \leq {n \choose k} \left(\frac{n}{4k}\right) \left(\frac{k}{n}\right)^{8k} \leq \left(\frac{e^{1.25}}{32}\right)^{4k} \left(\frac{k}{n}\right)^{3k} \leq \left(\frac{k}{n}\right)^{3k}$$

Starts at $1/n^3$, ...
Not dense.

Induced degree of node on subset, $S$, is degree of internal edges.

Induced degree of nodes in blue subset is 2, not 5!

**Claim:** Average induced degree on any subset of nodes is $\leq 8$ with probability $\geq 1 - O(\frac{1}{n^2})$.

**Proof:** Induced degree $\geq 8$  
$\rightarrow 4k$ internal edges for subset of size $k$.

$$\Pr[\text{dense } S] \leq \binom{n}{k} \left(\frac{n}{4k}\right)^8 \left(\frac{k}{n}\right) \leq \left(\frac{e^{1.25}}{32}\right)^{4k} \left(\frac{k}{n}\right)^{3k} \leq \left(\frac{k}{n}\right)^{3k}$$

Starts at $1/n^3$, decreasing till $k \leq n/8$ (at least)
Not dense.

Induced degree of node on subset, \( S \), is degree of internal edges.

Induced degree of nodes in blue subset is 2, not 5!

**Claim:** Average induced degree on any subset of nodes is \( \leq 8 \) with probability \( \geq 1 - O(\frac{1}{n^2}) \).

**Proof:** Induced degree \( \geq 8 \)
\( \rightarrow \) \( 4k \) internal edges for subset of size \( k \).

\[
\Pr[\text{dense } S] \leq \binom{n}{k} \left( \frac{n}{4k} \right)^{8k} \leq \left( \frac{e^{1.25}}{32} \right)^{4k} \left( \frac{k}{n} \right)^{3k} \leq \left( \frac{k}{n} \right)^{3k}
\]

Starts at \( 1/n^3 \), decreasing till \( k \leq n/8 \) (at least)
\( \rightarrow \) Total \( O(1/n^2) \).
**Removal Process!**

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:**
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
- Average induced degree 8 → half nodes with degree $\leq 16$.
- Half nodes removed in each iteration.
- $\log X$ iterations to remove all nodes.

Claim: Max load is $O(\log \log n)$ w.h.p.

Recall edge corresponds to ball.

Height of ball, $h_i$, is load of bin when it is placed in bin.

Corresponding edge removed in iteration $r_i$.

Property: $h_i \leq 16 r_i$.

Case $r_i = 1$ - only 16 balls incident to bin $\rightarrow h_i \leq 16$.

Induction: Previous removed edges (ball) induce load $\leq 16 (r_i - 1)$.

+ 16 edges/balls this iteration $\rightarrow h_i \leq 16 r_i$. 
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
- Average induced degree 8
Removal Process!

**Random Graph**: Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process**: Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
  - Average induced degree 8 $\rightarrow$ half nodes w/degree $\leq 16$.  

Recall edge corresponds to ball. Height of ball, $h_i$, is load of bin when it is placed in bin. Corresponding edge removed in iteration $r_i$.

Property: $h_i \leq 16$ $r_i$.

Case $r_i = 1$ - only 16 balls incident to bin $\rightarrow h_i \leq 16$.

Induction: Previous removed edges (ball) induce load $\leq 16$ ($r_i - 1$).+ 16 edges/balls this iteration. $\rightarrow h_i \leq 16$ $r_i$. 

Max load is $O(\log \log n)$ w.h.p.

Recall edge corresponds to ball. Height of ball, $h_i$, is load of bin when it is placed in bin. Corresponding edge removed in iteration $r_i$.

Property: $h_i \leq 16$ $r_i$.

Case $r_i = 1$ - only 16 balls incident to bin $\rightarrow h_i \leq 16$.

Induction: Previous removed edges (ball) induce load $\leq 16$ ($r_i - 1$).+ 16 edges/balls this iteration. $\rightarrow h_i \leq 16$ $r_i$. 

Max load is $O(\log \log n)$ w.h.p.
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
- Average induced degree 8 $\rightarrow$ half nodes w/degree $\leq 16$.
- $\rightarrow$ half nodes removed in each iteration.
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
- Average induced degree 8 $\rightarrow$ half nodes w/degree $\leq 16$.
- $\rightarrow$ half nodes removed in each iteration.
- $\rightarrow$ $\log X$ iterations to remove all nodes.
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
- Average induced degree 8 $\rightarrow$ half nodes w/degree $\leq 16$.
- $\rightarrow$ half nodes removed in each iteration.
- $\rightarrow$ log $X$ iterations to remove all nodes.

**Claim:** Max load is $O(\log \log n)$ w.h.p.
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
- Average induced degree 8 $\rightarrow$ half nodes w/degree $\leq 16$.
- $\rightarrow$ half nodes removed in each iteration.
- $\rightarrow$ log $X$ iterations to remove all nodes.

Claim: Max load is $O(\log \log n)$ w.h.p.

Recall edge corresponds to ball.
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
- Average induced degree 8 $\rightarrow$ half nodes w/degree $\leq 16$.
- $\rightarrow$ half nodes removed in each iteration.
- $\rightarrow$ log $X$ iterations to remove all nodes.

Claim: Max load is $O(\log \log n)$ w.h.p.

Recall edge corresponds to ball.
- Height of ball, $h_i$, is load of bin when it is placed in bin.
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
- Average induced degree 8 $\rightarrow$ half nodes w/degree $\leq 16$.
  $\rightarrow$ half nodes removed in each iteration.
  $\rightarrow$ $\log X$ iterations to remove all nodes.

**Claim:** Max load is $O(\log \log n)$ w.h.p.

Recall edge corresponds to ball.
- Height of ball, $h_i$, is load of bin when it is placed in bin.
- Corresponding edge removed in iteration $r_i$. 
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
- Average induced degree 8 $\rightarrow$ half nodes w/degree $\leq 16$.
- $\rightarrow$ half nodes removed in each iteration.
- $\rightarrow$ log $X$ iterations to remove all nodes.

Claim: Max load is $O(\log \log n)$ w.h.p.

Recall edge corresponds to ball.
- Height of ball, $h_i$, is load of bin when it is placed in bin.
- Corresponding edge removed in iteration $r_i$.

**Property:** $h_i \leq 16r_i$. 
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
- Average induced degree 8 $\rightarrow$ half nodes w/degree $\leq 16$.
- $\rightarrow$ half nodes removed in each iteration.
- $\rightarrow$ log $X$ iterations to remove all nodes.

**Claim:** Max load is $O(\log \log n)$ w.h.p.

Recall edge corresponds to ball.
- Height of ball, $h_i$, is load of bin when it is placed in bin.
- Corresponding edge removed in iteration $r_i$.

**Property:** $h_i \leq 16r_i$.
- Case $r_i = 1$
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
- Average induced degree 8 $\rightarrow$ half nodes w/degree $\leq 16$.
- $\rightarrow$ half nodes removed in each iteration.
- $\rightarrow$ log $X$ iterations to remove all nodes.

**Claim:** Max load is $O(\log \log n)$ w.h.p.

Recall edge corresponds to ball.
- Height of ball, $h_i$, is load of bin when it is placed in bin.
- Corresponding edge removed in iteration $r_i$.

**Property:** $h_i \leq 16r_i$.
- Case $r_i = 1$ - only 16 balls incident to bin.
Removal Process!

**Random Graph:** Component size is \( c \log n \) and max-induced degree is 8 w.h.p.

**Process:** Remove degree \( \leq 16 \) nodes and incident edges. Repeat.
Claim: \( O(\log X) \) iterations where \( X \) is max component size.

For any connected component:
  - Average induced degree 8 \( \rightarrow \) half nodes w/degree \( \leq 16 \).
  - \( \rightarrow \) half nodes removed in each iteration.
  - \( \rightarrow \) \( \log X \) iterations to remove all nodes.

**Claim:** Max load is \( O(\log \log n) \) w.h.p.

Recall edge corresponds to ball.
  - Height of ball, \( h_i \), is load of bin when it is placed in bin.
  - Corresponding edge removed in iteration \( r_i \).

**Property:** \( h_i \leq 16r_i \).
  - Case \( r_i = 1 \) - only 16 balls incident to bin \( \rightarrow h_i \leq 16 \).
  - Induction:
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
- Average induced degree 8 $\rightarrow$ half nodes w/degree $\leq 16$.
- $\rightarrow$ half nodes removed in each iteration.
- $\rightarrow \log X$ iterations to remove all nodes.

Claim: Max load is $O(\log \log n)$ w.h.p.

Recall edge corresponds to ball.
- Height of ball, $h_i$, is load of bin when it is placed in bin.
- Corresponding edge removed in iteration $r_i$.

**Property:** $h_i \leq 16r_i$.
- Case $r_i = 1$ - only 16 balls incident to bin $\rightarrow h_i \leq 16$.
- Induction: Previous removed edges(ball) induce load $\leq 16(r_i - 1)$.
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
- Average induced degree 8 $\rightarrow$ half nodes w/degree $\leq 16$.
  $\rightarrow$ half nodes removed in each iteration.
  $\rightarrow$ log $X$ iterations to remove all nodes.

Claim: Max load is $O(\log \log n)$ w.h.p.

Recall edge corresponds to ball.
- Height of ball, $h_i$, is load of bin when it is placed in bin.
  Corresponding edge removed in iteration $r_i$.

**Property:** $h_i \leq 16r_i$.
- Case $r_i = 1$ - only 16 balls incident to bin $\rightarrow h_i \leq 16$.
- Induction: Previous removed edges(ball) induce load $\leq 16(r_i - 1)$.
  $+16$ edges/balls this iteration.
Removal Process!

**Random Graph:** Component size is $c \log n$ and max-induced degree is 8 w.h.p.

**Process:** Remove degree $\leq 16$ nodes and incident edges. Repeat.

Claim: $O(\log X)$ iterations where $X$ is max component size.

For any connected component:
- Average induced degree 8 $\rightarrow$ half nodes w/degree $\leq 16$.
  $\rightarrow$ half nodes removed in each iteration.
  $\rightarrow$ log $X$ iterations to remove all nodes.

**Claim:** Max load is $O(\log \log n)$ w.h.p.

Recall edge corresponds to ball.
- Height of ball, $h_i$, is load of bin when it is placed in bin.
  Corresponding edge removed in iteration $r_i$.

**Property:** $h_i \leq 16r_i$.

Case $r_i = 1$ - only 16 balls incident to bin $\rightarrow$ $h_i \leq 16$.

Induction: Previous removed edges(ball) induce load $\leq 16(r_i - 1)$.
  $+16$ edges/balls this iteration.

$\rightarrow$ $h_i \leq 16r_i$. 
Power of two choices.

Max load: $\log X$ where $X$ is max component size.
Power of two choices.

Max load: $\log X$ where $X$ is max component size.

$X$ is $O(\log n)$ with high probability.
Power of two choices.

Max load: $\log X$ where $X$ is max component size.

$X$ is $O(\log n)$ with high probability.

Max load is $O(\log \log n)$. 
Cuckoo hashing.

Hashing with two choices: max load $O(\log \log n)$.  

Fails if cycle for insert.

C\ell - event of cycle of length \ell at a vertex.

$$\Pr[C_\ell] \leq (m^{\ell} n^{\ell})^{2(\ell n)} \leq (e^{2/8})^{\ell (3)}$$

Probability that an insert hits a cycle of length \ell

Rehash every $\Omega(n)$ inserts (if $\leq n/8$ items in table.)

$O(1)$ time on average.
Cuckoo hashing.

Hashing with two choices: max load $O(\log \log n)$.

Cuckoo hashing:
Cuckoo hashing.

Hashing with two choices: max load $O(\log \log n)$.

Cuckoo hashing:
Array.
Cuckoo hashing.

Hashing with two choices: max load $O(\log \log n)$.

Cuckoo hashing:
Array. Two hash functions $h_1$, $h_2$. 

---

Probability that an insert hits a cycle of length $\ell$: 
$\Pr[C_\ell] \leq (m \ell)(n \ell)(\ell n)^2 \leq (e^2 8)^{\ell/3}$

Rehash every $\Omega(n)$ inserts (if $\leq n/8$ items in table.)

$O(1)$ time on average.
Cuckoo hashing.

Hashing with two choices: max load $O(\log \log n)$.

Cuckoo hashing:
Array. Two hash functions $h_1, h_2$.
Insert $x$: place in $h_1(x)$ or $h_2(x)$ if space.
Cuckoo hashing.

Hashing with two choices: max load $O(\log \log n)$.

Cuckoo hashing:
Array. Two hash functions $h_1$, $h_2$.

Insert $x$: place in $h_1(x)$ or $h_2(x)$ if space.
   Else bump elt $y$ in $h_i(x)$ u.a.r. for $i \in [1,2]$. 
Cuckoo hashing.

Hashing with two choices: max load $O(\log\log n)$.

Cuckoo hashing:
Array. Two hash functions $h_1, h_2$.

Insert $x$: place in $h_1(x)$ or $h_2(x)$ if space.

Else bump elt $y$ in $h_i(x)$ u.a.r. for $i \in [1,2]$.

Bump $y, x$: place $y$ in $h_j(y)$ where $j \neq i$ if space.
Cuckoo hashing.

Hashing with two choices: max load $O(\log \log n)$.

Cuckoo hashing:
Array. Two hash functions $h_1$, $h_2$.

Insert $x$: place in $h_1(x)$ or $h_2(x)$ if space.
   Else bump elt $y$ in $h_i(x)$ u.a.r. for $i \in [1,2]$.
Bump $y, x$: place $y$ in $h_j(y)$ where $j \neq i$ if space.
   Else bump $y'$ in $h_i(y)$.
Cuckoo hashing.

Hashing with two choices: max load $O(\log \log n)$.

Cuckoo hashing:
Array. Two hash functions $h_1$, $h_2$.

Insert $x$: place in $h_1(x)$ or $h_2(x)$ if space.
   Else bump elt $y$ in $h_i(x)$ u.a.r. for $i \in [1, 2]$.
Bump $y, x$: place $y$ in $h_j(y)$ where $j \neq i$ if space.
   Else bump $y'$ in $h_i(y)$.

If go too long. Fail. Rehash entire hash table.
Fails if cycle for insert.

$C_\ell$ - event of cycle of length $\ell$ at a vertex.

$$\Pr[C_\ell] \leq (m^{\ell})(n^{\ell})(\ell^n)^2 \leq (e^{2/8})^{\ell}(3)$$

Probability that an insert hits a cycle of length $\ell \leq \ell n (e^{2/8})^{\ell}$
Rehash every $\Omega(n)$ inserts (if $\leq n/8$ items in table.)

$O(1)$ time on average.
Cuckoo hashing.

Hashing with two choices: max load $O(\log \log n)$.

Cuckoo hashing:
Array. Two hash functions $h_1, h_2$.

Insert $x$: place in $h_1(x)$ or $h_2(x)$ if space.
   Else bump elt $y$ in $h_i(x)$ u.a.r. for $i \in [1,2]$.
Bump $y, x$: place $y$ in $h_j(y)$ where $j \neq i$ if space.
   Else bump $y'$ in $h_i(y)$.

If go too long.
Cuckoo hashing.

Hashing with two choices: max load $O(\log \log n)$.

Cuckoo hashing:
Array. Two hash functions $h_1$, $h_2$.

Insert $x$: place in $h_1(x)$ or $h_2(x)$ if space.
Else bump elt $y$ in $h_i(x)$ u.a.r. for $i \in [1, 2]$.

Bump $y$, $x$: place $y$ in $h_j(y)$ where $j \neq i$ if space.
Else bump $y'$ in $h_i(y)$.

If go too long. Fail.
Cuckoo hashing.

Hashing with two choices: max load $O(\log \log n)$.

Cuckoo hashing:
Array. Two hash functions $h_1$, $h_2$.

Insert $x$: place in $h_1(x)$ or $h_2(x)$ if space.
  Else bump elt $y$ in $h_i(x)$ u.a.r. for $i \in [1, 2]$.
Bump $y$, $x$: place $y$ in $h_j(y)$ where $j \neq i$ if space.
  Else bump $y'$ in $h_i(y)$.

If go too long. Fail. Rehash entire hash table.
Cuckoo hashing.

Hashing with two choices: max load $O(\log \log n)$.

Cuckoo hashing:
Array. Two hash functions $h_1, h_2$.

Insert $x$: place in $h_1(x)$ or $h_2(x)$ if space.
   Else bump elt $y$ in $h_i(x)$ u.a.r. for $i \in [1, 2]$.
Bump $y, x$: place $y$ in $h_j(y)$ where $j \neq i$ if space.
   Else bump $y'$ in $h_i(y)$.

If go too long. Fail. Rehash entire hash table.
   Fails if cycle for insert.
Cuckoo hashing.

Hashing with two choices: max load $O(\log \log n)$.

Cuckoo hashing:
Array. Two hash functions $h_1$, $h_2$.

Insert $x$: place in $h_1(x)$ or $h_2(x)$ if space.
   Else bump elt $y$ in $h_i(x)$ u.a.r. for $i \in [1,2]$.
Bump $y$, $x$: place $y$ in $h_j(y)$ where $j \neq i$ if space.
   Else bump $y'$ in $h_i(y)$.

If go too long. Fail. Rehash entire hash table.
   Fails if cycle for insert.

$C_\ell$ - event of cycle of length $\ell$ at a vertex.
Cuckoo hashing.

Hashing with two choices: max load \( O(\log \log n) \).

Cuckoo hashing:
Array. Two hash functions \( h_1, h_2 \).

Insert \( x \): place in \( h_1(x) \) or \( h_2(x) \) if space.
   Else bump elt \( y \) in \( h_i(x) \) u.a.r. for \( i \in [1,2] \).
Bump \( y, x \): place \( y \) in \( h_j(y) \) where \( j \neq i \) if space.
   Else bump \( y' \) in \( h_i(y) \).

If go too long. Fail. Rehash entire hash table.
   Fails if cycle for insert.

\( C_\ell \) - event of cycle of length \( \ell \) at a vertex.

\[
\Pr[C_\ell] \leq \left( \frac{m}{\ell} \right) \left( \frac{n}{\ell} \right) \left( \frac{\ell}{n} \right)^{2\ell} \leq \left( \frac{e^2}{8} \right)^\ell
\]  (3)
Cuckoo hashing.

Hashing with two choices: max load $O(\log \log n)$.

Cuckoo hashing:
Array. Two hash functions $h_1, h_2$.

Insert $x$: place in $h_1(x)$ or $h_2(x)$ if space.
   Else bump elt $y$ in $h_i(x)$ u.a.r. for $i \in [1,2]$.

Bump $y, x$: place $y$ in $h_j(y)$ where $j \neq i$ if space.
   Else bump $y'$ in $h_i(y)$.

If go too long. Fail. Rehash entire hash table.
   Fails if cycle for insert.

$C_\ell$ - event of cycle of length $\ell$ at a vertex.

$$Pr[C_\ell] \leq \binom{m}{\ell} \binom{n}{\ell} \left( \frac{\ell}{n} \right)^{2(\ell)} \leq \left( \frac{e^2}{8} \right)^{\ell} \tag{3}$$

Probability that an insert hits a cycle of length $\ell \leq \frac{\ell}{n} \left( \frac{e^2}{8} \right)^{\ell}$
Cuckoo hashing.

Hashing with two choices: max load $O(\log \log n)$.

Cuckoo hashing:
Array. Two hash functions $h_1, h_2$.

Insert $x$: place in $h_1(x)$ or $h_2(x)$ if space.
Else bump elt $y$ in $h_i(x)$ u.a.r. for $i \in [1,2]$.  
Bump $y, x$: place $y$ in $h_j(y)$ where $j \neq i$ if space.  
Else bump $y'$ in $h_i(y)$.

If go too long. Fail. Rehash entire hash table.  
Fails if cycle for insert.

$C_\ell$ - event of cycle of length $\ell$ at a vertex.

$$\Pr[C_\ell] \leq \left( \binom{m}{\ell} \binom{n}{\ell} \left( \frac{\ell}{n} \right)^{2(\ell)} \leq \left( \frac{e^2}{8} \right)^\ell \right.$$

Probability that an insert hits a cycle of length $\ell \leq \frac{\ell}{n} \left( \frac{e^2}{8} \right)^\ell$

Rehash every $\Omega(n)$ inserts (if $\leq n/8$ items in table.)
Cuckoo hashing.

Hashing with two choices: max load \( O(\log \log n) \).

Cuckoo hashing:
Array. Two hash functions \( h_1, h_2 \).

Insert \( x \): place in \( h_1(x) \) or \( h_2(x) \) if space.
   Else bump elt \( y \) in \( h_i(x) \) u.a.r. for \( i \in [1,2] \).

Bump \( y, x \): place \( y \) in \( h_j(y) \) where \( j \neq i \) if space.
   Else bump \( y' \) in \( h_i(y) \).

If go too long. Fail. Rehash entire hash table.
   Fails if cycle for insert.

\( C_\ell \) - event of cycle of length \( \ell \) at a vertex.

\[
\Pr[C_\ell] \leq \binom{m}{\ell} \binom{n}{\ell} \left( \frac{\ell}{n} \right)^{2(\ell)} \leq \left( \frac{e^2}{8} \right)^\ell
\]

(3)

Probability that an insert hits a cycle of length \( \ell \leq \frac{\ell}{n} \left( \frac{e^2}{8} \right)^\ell \)

Rehash every \( \Omega(n) \) inserts (if \( \leq n/8 \) items in table.)
\( O(1) \) time on average.
Sum up

Balls in bins: $\Theta(\log n / \log \log n)$ load.
Sum up

Balls in bins: \( \Theta(\log n / \log \log n) \) load.

Power of two: \( \Theta(\log \log n) \).
Sum up

Balls in bins: $\Theta(\log n / \log \log n)$ load.
Power of two: $\Theta(\log \log n)$.
Cuckoo hashing.
See you on Thursday...