Probability and Computing

Chapter 5: Balls, Bins, and Random Graphs

Zeng Yuxiang

The Hong Kong University of Science and Technology

July 16, 2018

Table of Contents

Table of Contents

Example: The Birthday Paradox

Example

You notice that there are 30 people in the room. What is the probability that no two people in the room share the same birthday?

Analysis

Example: The Birthday Paradox

Example

You notice that there are 30 people in the room. What is the probability that no two people in the room share the same birthday?

We assume the birthday of each person is a random day from a 365-day year, chosen independently and uniformly at random for each person.

Analysis

Example: The Birthday Paradox

Example

You notice that there are 30 people in the room. What is the probability that no two people in the room share the same birthday?

We assume the birthday of each person is a random day from a 365-day year, chosen independently and uniformly at random for each person.

Analysis

First, there are 365^{30} possible configurations in total.

Example: The Birthday Paradox

Example

You notice that there are 30 people in the room. What is the probability that no two people in the room share the same birthday?

We assume the birthday of each person is a random day from a 365-day year, chosen independently and uniformly at random for each person.

Analysis

First, there are 365^{30} possible configurations in total.
Second, there are $\binom{365}{30}$ possible confurations to choose 30 different days from 365 days. These 30 days can be assigned to the people in any of the 30! possible orders.

Example: The Birthday Paradox

Example

You notice that there are 30 people in the room. What is the probability that no two people in the room share the same birthday?

We assume the birthday of each person is a random day from a 365-day year, chosen independently and uniformly at random for each person.

Analysis

First, there are 365^{30} possible configurations in total.
Second, there are $\binom{365}{30}$ possible confurations to choose 30 different days from 365 days. These 30 days can be assigned to the people in any of the 30! possible orders.
Finally, there are $\binom{365}{30} \cdot 30!$ configurations where no two people share the same birthday, out of the 365^{30} ways the birthdays could occur. Thus, the probability is $\frac{\binom{365}{30} \cdot 30!}{365^{30}} = \frac{\frac{\prod_{i=0}^{29}(365-i)}{30!} \cdot 30!}{365^{30}} = \prod_{i=0}^{29}\left(1 - \frac{i}{365}\right) = \prod_{i=1}^{29}\left(1 - \frac{i}{365}\right)$.

Example: The Birthday Paradox

Example

You notice that there are 30 people in the room. What is the probability that no two people in the room share the same birthday?

Analysis (Another Version)

Consider the probability by one person at a time.
Specifically, the first person can choose any day.

Example: The Birthday Paradox

Example

You notice that there are 30 people in the room. What is the probability that no two people in the room share the same birthday?

Analysis (Another Version)

Consider the probability by one person at a time.
Specifically, the first person can choose any day.
The probability for the second person to choose a different day is $(1 - \frac{1}{365})$.

Example: The Birthday Paradox

Example

You notice that there are 30 people in the room. What is the probability that no two people in the room share the same birthday?

Analysis (Another Version)

Consider the probability by one person at a time.
Specifically, the first person can choose any day.
The probability for the second person to choose a different day is $(1 - \frac{1}{365})$.
The probability for the third person to choose a different day is $(1 - \frac{2}{365})$.
Similarly, the probability for the k-th person to choose a different day si $(1 - \frac{k-1}{365})$.

Example: The Birthday Paradox

Example

You notice that there are 30 people in the room. What is the probability that no two people in the room share the same birthday?

Analysis (Another Version)

Consider the probability by one person at a time.
Specifically, the first person can choose any day.
The probability for the second person to choose a different day is $(1-\frac{1}{365})$.
The probability for the third person to choose a different day is $(1-\frac{2}{365})$.
Similarly, the probability for the k-th person to choose a different day si $(1-\frac{k-1}{365})$.
Thus, the probability that 30 people all have different birthday is

$$(1-\frac{1}{365})(1-\frac{2}{365})\cdots(1-\frac{29}{365})=\prod_{i=1}^{29}(1-\frac{i}{365})\approx 0.2937$$

Example: The Birthday Paradox

Example

You notice that there are 30 people in the room. What is the probability that no two people in the room share the same birthday?

A similar calculation shows that only 23 people need to be in the room before it is more likely (with probability higher than 50%) than not that two people share a birthday.

Example: The Birthday Paradox

Example

More generally, if there are m people and n possible birthdays, then what is the probability that all m have different birthdays?

Analysis

Example: The Birthday Paradox

Example

More generally, if there are m people and n possible birthdays, then what is the probability that all m have different birthdays?

Analysis

Inductively,

$$(1-\frac{1}{n})(1-\frac{2}{n})\cdots(1-\frac{m-1}{n}) = \prod_{i=1}^{m-1}(1-\frac{i}{n})$$

Example: The Birthday Paradox

Example

More generally, if there are m people and n possible birthdays, then what is the probability that all m have different birthdays?

Analysis

Inductively,

$$(1-\frac{1}{n})(1-\frac{2}{n})\cdots(1-\frac{m-1}{n}) = \prod_{i=1}^{m-1}(1-\frac{i}{n})$$

Using that $1-i/n \approx e^{-i/n}$ when i is small compared to n, we see that if m is small compared to n then

$$\prod_{i=1}^{m-1}(1-\frac{i}{n}) \approx \prod_{i=1}^{m-1} e^{-i/n} = \exp\{-\prod_{i=1}^{m-1}\frac{i}{n}\}$$
$$= e^{-m(m-1)/2n} = e^{-m^2/2n}$$

Example: The Birthday Paradox

Example

More generally, if there are m people and n possible birthdays, then what is the probability that all m have different birthdays?

Analysis

The probability that all m have different birthdays is $e^{-m^2/2n}$. Hence, the value for m at which the probability that m people all have different birthdays is $1/2$, which is approximately given by the equation

$$\frac{m^2}{2n} = \ln 2,$$

or $m = \sqrt{2n \ln 2}$.

Example: The Birthday Paradox

Example

More generally, if there are m people and n possible birthdays, then what is the probability that all m have different birthdays?

Analysis

The probability that all m have different birthdays is $e^{-m^2/2n}$. Hence, the value for m at which the probability that m people all have different birthdays is $1/2$, which is approximately given by the equation

$$\frac{m^2}{2n} = \ln 2,$$

or $m = \sqrt{2n \ln 2}$.
For the case $n = 365$, this approximation gives $m = 22.49$ to two decimal places.

Example: The Birthday Paradox

Example

More generally, if there are m people and n possible birthdays, then what is the probability that all m have different birthdays?

Loose Bound Analysis

let E_k be the event that the kth person's birthday does not match any of the birthdays of the first $k-1$ people. Then the probability that the first k people fail to have distinct birthdays is

$$\Pr(\bar{E}_1 \cup \bar{E}_2 \cup \cdots \cup \bar{E}_k) \leq \sum_{i=1}^{k} \bar{E}_i \leq \sum_{i=1}^{k} \frac{i-1}{n} = \frac{k(k-1)}{2n}$$

Example: The Birthday Paradox

Example

More generally, if there are m people and n possible birthdays, then what is the probability that all m have different birthdays?

Loose Bound Analysis

let E_k be the event that the kth person's birthday does not match any of the birthdays of the first $k-1$ people. Then the probability that the first k people fail to have distinct birthdays is

$$\Pr(\bar{E}_1 \cup \bar{E}_2 \cup \cdots \cup \bar{E}_k) \leq \sum_{i=1}^{k} \bar{E}_i \leq \sum_{i=1}^{k} \frac{i-1}{n} = \frac{k(k-1)}{2n}$$

If $k \leq \sqrt{n}$ fo this probability is less than $1/2$, so with $\lfloor \sqrt{n} \rfloor$ people the probability is at least $1/2$ that all birthdays will be distinct.

Example: The Birthday Paradox

Example

More generally, if there are m people and n possible birthdays, then what is the probability that all m have different birthdays?

Loose Bound Analysis (Cont.)

Assume that the first $\lceil \sqrt{n} \rceil$ people all have distinct birthdays. Each person after has probability at least $\sqrt{n}/n = 1/\sqrt{n}$ of having the same birthday as one of these $\sqrt{n}$ people. Thus, the probability that the next $\lceil \sqrt{n} \rceil$ people all have different birthdays from the first $\lceil \sqrt{n} \rceil$ is at most

$$(1 - \frac{1}{\sqrt{n}})^{\lceil \sqrt{n} \rceil} < \frac{1}{e} < \frac{1}{2}.$$

Example: The Birthday Paradox

Example

More generally, if there are m people and n possible birthdays, then what is the probability that all m have different birthdays?

Loose Bound Analysis (Cont.)

Assume that the first $\lceil\sqrt{n}\rceil$ people all have distinct birthdays. Each person after has probability at least $\sqrt{n}/n = 1/\sqrt{n}$ of having the same birthday as one of these $\sqrt{n}$ people. Thus, the probability that the next $\lceil\sqrt{n}\rceil$ people all have different birthdays from the first $\lceil\sqrt{n}\rceil$ is at most

$$(1 - \frac{1}{\sqrt{n}})^{\lceil\sqrt{n}\rceil} < \frac{1}{e} < \frac{1}{2}.$$

Hence, once there are $\sqrt{n}$ people, the probability is at most $1/e$ that all birthdays will be distinct.

Table of Contents

Definition: The Balls-and-Bins Model

The balls-and-bins model

We have m balls that are thrown into n bins, with the location of each ball chosen independently and uniformly at random from the n possibilities. What does the distribution of the balls in the bins look like?

Definition: The Balls-and-Bins Model

The balls-and-bins model

We have m balls that are thrown into n bins, with the location of each ball chosen independently and uniformly at random from the n possibilities. What does the distribution of the balls in the bins look like?

- How many of the bins are empty?

Definition: The Balls-and-Bins Model

The balls-and-bins model

We have m balls that are thrown into n bins, with the location of each ball chosen independently and uniformly at random from the n possibilities. What does the distribution of the balls in the bins look like?

- How many of the bins are empty?
- How many balls are in the fullest bin?

Definition: The Balls-and-Bins Model

The balls-and-bins model

We have m balls that are thrown into n bins, with the location of each ball chosen independently and uniformly at random from the n possibilities. What does the distribution of the balls in the bins look like?

- How many of the bins are empty?
- How many balls are in the fullest bin?
- Whether there is a bin containing at least two balls? (Birthday Paradox)

Definition: The Balls-and-Bins Model

The balls-and-bins model

We have m balls that are thrown into n bins, with the location of each ball chosen independently and uniformly at random from the n possibilities. What does the distribution of the balls in the bins look like?

- How many of the bins are empty?
- How many balls are in the fullest bin?
- Whether there is a bin containing at least two balls? (Birthday Paradox)

Birthday Paradox

If m balls are randomly placed into n bins then, for some $m = \Omega(\sqrt{n})$, at least one of the bins is likely to have more than one ball in it.

Example: The Balls-and-Bins Model

Example

What is the maximum number of balls in a bin, or the maximum *load*.

Analysis

Example: The Balls-and-Bins Model

Example

What is the maximum number of balls in a bin, or the maximum *load*.

Analysis

Let us first consider the case $m = n$, so that the average *load* is 1. Of course the maximum *load* is m, but it is very unlikely that all m balls land in the same bin.

Example: The Balls-and-Bins Model

Example

What is the maximum number of balls in a bin, or the maximum *load*.

Analysis

Let us first consider the case $m = n$, so that the average *load* is 1. Of course the maximum *load* is m, but it is very unlikely that all m balls land in the same bin.
We seek an upper bound that holds with probability tending to 1 as n grows large.

Example: The Balls-and-Bins Model

Example

What is the maximum number of balls in a bin, or the maximum *load*.

Analysis

Let us first consider the case $m = n$, so that the average *load* is 1. Of course the maximum *load* is m, but it is very unlikely that all m balls land in the same bin.

We seek an upper bound that holds with probability tending to 1 as n grows large.

We can show that the maximum *load* is not more than $3 \ln n / \ln \ln n$ with probability at most $1/n$ for sufficiently large n via a direct calculation and a union bound.

Example: The Balls-and-Bins Model

Maximum Load

When n balls are thrown independently and uniformly at random into n bins, the probability that the maximum load is more than $3 \ln n / \ln \ln n$ is at most $1/n$ for n sufficiently large.

Example: The Balls-and-Bins Model

Maximum Load

When n balls are thrown independently and uniformly at random into n bins, the probability that the maximum load is more than $3 \ln n / \ln \ln n$ is at most $1/n$ for n sufficiently large.

Proof:

The probability that bin 1 receives at least M balls is at most

$$\binom{n}{M}\left(\frac{1}{n}\right)^M = \frac{\prod_{i=0}^{M-1}(n-i)}{M!} \cdot \frac{1}{n^M} \le \frac{1}{M!} \le \left(\frac{e}{M}\right)^M$$

The last inequality is because

$$\frac{k^k}{k!} < \sum_{i=0}^{\infty} \frac{k^i}{i!} = e^k \Rightarrow k! > (k/e)^k$$

Example: The Balls-and-Bins Model

Maximum Load

When n balls are thrown independently and uniformly at random into n bins, the probability that the maximum load is more than $3\ln n/\ln\ln n$ is at most $1/n$ for n sufficiently large.

Proof:

The probability that bin 1 receives at least M balls is at most $(\frac{e}{M})^M$. Applying a union bound again allows us to find that, for $M \geq 3\ln n/\ln\ln n$, the probability that any bin receives at least M balls is bounded above by

$$\begin{aligned} n\cdot(\frac{e}{M})^M &\leq n(\frac{e\ln\ln n}{3\ln n})^{3\ln n/\ln\ln n} \\ &< n(\frac{\ln\ln n}{\ln n})^{3\ln n/\ln\ln n} \\ &= e^{\ln n}(e^{\ln\ln\ln n-\ln\ln n})^{3\ln n/\ln\ln n} \\ &= e^{-2\ln n+3(\ln n)(\ln\ln\ln n)/\ln\ln n} \leq \frac{1}{n} \end{aligned}$$

Application: Bucket Sort

Definition: Bucket Sort

Suppose that we have a set of $n = 2^m$ elements to be sorted and that each element is an integer chosen independently and uniformly at random from the range $[0, 2^k)$, where $k \geq m$. Using Bucket sort, we can sort the numbers in expected time $O(n)$. Here, expectation is over the choice of the random input, since Bucket sort is a completely deterministic algorithm.

Basic Idea: Bucket Sort

Buck sort works in two stages. In the first stage, we place the elements into n buckets. The j-th bucket holds all elements whose first m binary digits correspond to the number j.

Application: Bucket Sort

Definition: Bucket Sort

Suppose that we have a set of $n = 2^m$ elements to be sorted and that each element is an integer chosen independently and uniformly at random from the range $[0, 2^k)$, where $k \geq m$. Using Bucket sort, we can sort the numbers in expected time $O(n)$. Here, expectation is over the choice of the random input, since Bucket sort is a completely deterministic algorithm.

Basic Idea: Bucket Sort

Buck sort works in two stages. In the first stage, we place the elements into n buckets. The j-th bucket holds all elements whose first m binary digits correspond to the number j. For example, if $n = 2^{10}$, bucket 3 contains all elements whose first 10 binary digits are 0000000011. When $i < \mathcal{L}$ the elements of the i-th bucket all come before the elements in the $\mathcal{L}$-th bucket in the sorted order.

Application: Bucket Sort

Definition: Bucket Sort

Suppose that we have a set of $n = 2^m$ elements to be sorted and that each element is an integer chosen independently and uniformly at random from the range $[0, 2^k)$, where $k \geq m$. Using Bucket sort, we can sort the numbers in expected time $O(n)$. Here, expectation is over the choice of the random input, since Bucket sort is a completely deterministic algorithm.

Basic Idea: Bucket Sort

Buck sort works in two stages. In the first stage, we place the elements into n buckets. The j-th bucket holds all elements whose first m binary digits correspond to the number j. Assuming that each element can be placed in the appropriate bucket in $O(1)$ time, this stage requires only $O(n)$ time. Because of the assumption that the elements to be sorted are chosen uniformly, the number of elements that land in a specific bucket follows a binomial distribution $B(n, 1/n)$.

Application: Bucket Sort

Basic Idea: Bucket Sort (Cont.)

In the second stage, each bucket is sorted using any standard quadratic time algorithm (such as Bubble sort or Insertion sort). Concatenating the sorted lists from each bucket in order gives us the sorted order for the elements. It remains to show that the expected time spent in the second stage is only $O(n)$.

Application: Bucket Sort

Basic Idea: Bucket Sort (Cont.)

In the second stage, each bucket is sorted using any standard quadratic time algorithm (such as Bubble sort or Insertion sort). Concatenating the sorted lists from each bucket in order gives us the sorted order for the elements. It remains to show that the expected time spent in the second stage is only $O(n)$.
The result relies on our assumption regarding the input distribution. Under the uniform distribution, Bucket sort falls naturally into the balls and bins model: the elements are balls, buckets are bins, and each ball falls uniformly at random into a bin.

Application: Bucket Sort

Basic Idea: Bucket Sort (Cont.)

Let X_j be the number of elements that land in the j-th bucket. The time to sort the j-th bucket is then at most $c(X_j)^2$ for some constant c. The expected time spent sorting in the second stage is at most

$$E\left[\sum_{j=1}^{n} c(X_j)^2\right] = c\sum_{j=1}^{n} E[X_j^2] = cnE[X_1^2].$$

Application: Bucket Sort

Basic Idea: Bucket Sort (Cont.)

Let X_j be the number of elements that land in the j-th bucket. The time to sort the j-th bucket is then at most $c(X_j)^2$ for some constant c. The expected time spent sorting in the second stage is at most

$$E\Big[\sum_{j=1}^{n} c(X_j)^2\Big] = c\sum_{j=1}^{n} E[X_j^2] = cnE[X_1^2].$$

Since X_1, is a binomial random variable $B(n, 1/n)$, we have

$$E[X_1^2] = n\frac{1}{n}(1-\frac{1}{n}) + (n\frac{1}{n})^2 = 2 - \frac{1}{n} < 2.$$

Hence the total expected time spent in the second stage is at most $2cn$, so Bucket sort runs in expected linear time.

Table of Contents

Background: The Poisson Distribution

Example

We now consider the probability that a given bin is empty in the balls and bins model. For the first bin to be empty, it must be missed by all m balls. Since each ball hits the first bin with probability $1/n$, the probability the first bin remains empty is $(1 - 1/n)^m \approx e^{-m/n}$.

Background: The Poisson Distribution

Example

We now consider the probability that a given bin is empty in the balls and bins model. For the first bin to be empty, it must be missed by all m balls. Since each ball hits the first bin with probability $1/n$, the probability the first bin remains empty is $(1-1/n)^m \approx e^{-m/n}$. If X_j is a random variable that is 1 when the j-th bin is empty and 0 otherwise, then $E[X_j] = (1-1/n)^m$. Let X be a random variable that represents the number of empty bins. Then, by the linearity of expectations

$$E[X] = E\left[\sum_{j=1}^{n} X_j\right] = \sum_{j=1}^{n} E[X_j] = n(1-\frac{1}{n})^m = ne^{-m/n}$$

Background: The Poisson Distribution

Example

We now consider the probability that a given bin is empty in the balls and bins model. For the first bin to be empty, it must be missed by all m balls. Since each ball hits the first bin with probability $1/n$, the probability the first bin remains empty is $(1-1/n)^m \approx e^{-m/n}$. If X_j is a random variable that is 1 when the j-th bin is empty and 0 otherwise, then $E[X_j] = (1-1/n)^m$. Let X be a random variable that represents the number of empty bins. Then, by the linearity of expectations

$$E[X] = E\Big[\sum_{j=1}^{n} X_j\Big] = \sum_{j=1}^{n} E[X_j] = n(1-\frac{1}{n})^m = ne^{-m/n}$$

Thus, the expected fraction of empty bins is approximately $e^{-m/n}$.

Background: The Poisson Distribution

Example

We can generalize the preceding argument to find the expected fraction of bins with r balls for any constant r. The probability that a given bin has r balls is

$$\binom{m}{r}(\frac{1}{n})^r(1-\frac{1}{n})^{m-r} = \frac{1}{r!}\frac{m(m-1)\cdots(m-r+1)}{n^r}(1-\frac{1}{n})^{m-r}$$

Background: The Poisson Distribution

Example

We can generalize the preceding argument to find the expected fraction of bins with r balls for any constant r. The probability that a given bin has r balls is

$$\binom{m}{r}(\frac{1}{n})^r(1-\frac{1}{n})^{m-r} = \frac{1}{r!}\frac{m(m-1)\cdots(m-r+1)}{n^r}(1-\frac{1}{n})^{m-r}$$

When m and n are large compared to r, the second factor on the right-hand side is approximately $(m/n)^r$, and the third factor is approximately $e^{-m/n}$.

Background: The Poisson Distribution

Example

We can generalize the preceding argument to find the expected fraction of bins with r balls for any constant r. The probability that a given bin has r balls is

$$\binom{m}{r}(\frac{1}{n})^r(1-\frac{1}{n})^{m-r} = \frac{1}{r!}\frac{m(m-1)\cdots(m-r+1)}{n^r}(1-\frac{1}{n})^{m-r}$$

When m and n are large compared to r, the second factor on the right-hand side is approximately $(m/n)^r$, and the third factor is approximately $e^{-m/n}$.
Hence, the probability p_r that a given bin has r balls is approximately $p_r = \frac{e^{-m/n}(m/n)^r}{r!}$ and the expected number of bins with exactly r balls is approximately np_r.

Definition: Poisson Random Variable

Discrete Poisson Random Variable

A discrete Poisson random variable X with parameter μ is given by the following probability distribution on $j = 0, 1, 2, \cdots$ $\Pr(X = j) = \frac{e^{-\mu}\mu^j}{j!}$.

Verfication

Let us verify that the definition gives a proper distribution in that the probabilities sum to 1:

$$\begin{aligned}\sum_{j=0}^{\infty} \Pr(X = j) &= \sum_{j=0}^{\infty} \frac{e^{-\mu}\mu^j}{j!} \\ &= e^{-\mu} \sum_{j=0}^{\infty} \frac{\mu^j}{j!} \\ &= 1\end{aligned}$$

where we use the Taylor expansion $e^x = \sum_{j=0}^{\infty} (x^j / j!)$.

Definition: Poisson Random Variable

Discrete Poisson Random Variable

A discrete Poisson random variable X with parameter μ is given by the following probability distribution on $j = 0, 1, 2, \cdots$ $\Pr(X = j) = \frac{e^{-\mu}\mu^j}{j!}$.

Expectation

Next we show that the expectation of this random variable is μ:

$$\begin{aligned} E[X] &= \sum_{j=0}^{\infty} j\Pr(X = j) = \sum_{j=1}^{\infty} j\frac{e^{-\mu}\mu^j}{j!} \\ &= \mu \sum_{j=1}^{\infty} \frac{e^{-\mu}\mu^{j-1}}{(j-1)!} = \mu \sum_{j=0}^{\infty} \frac{e^{-\mu}\mu^j}{j!} \\ &= \mu \end{aligned}$$

Definition: Poisson Random Variable

Discrete Poisson Random Variable

A discrete Poisson random variable X with parameter μ is given by the following probability distribution on $j = 0, 1, 2, \cdots$ $\Pr(X = j) = \frac{e^{-\mu}\mu^j}{j!}$.

Expectation

Next we show that the expectation of this random variable is μ:

$$\begin{aligned} E[X] &= \sum_{j=0}^{\infty} j \Pr(X = j) = \sum_{j=1}^{\infty} j \frac{e^{-\mu}\mu^j}{j!} \\ &= \mu \sum_{j=1}^{\infty} \frac{e^{-\mu}\mu^{j-1}}{(j-1)!} = \mu \sum_{j=0}^{\infty} \frac{e^{-\mu}\mu^j}{j!} \\ &= \mu \end{aligned}$$

where the distribution of the number of balls in a bin is approximately Poisson with $\mu = m/n$. It is exactly the average number of balls per bin.

Multiple Poisson Random Variable

Multiple Poisson Random Variable

The sum of a finite number of independent Poisson random variables is a Poisson random variable.

Proof:

Here, we consider independent two Poisson random variables X and Y with means μ_1 and μ_2.

$$\Pr(X+Y=j) = \sum_{k=0}^{j} \Pr\Big((X=k)\cup(Y=j-k)\Big)$$

$$= \sum_{k=0}^{j} \frac{e^{-\mu_1}\mu_1^k}{k!}\frac{e^{-\mu_2}\mu_2^{(j-k)}}{(j-k)!} = \frac{e^{-(\mu_1+\mu_2)}}{j!}\sum_{k=0}^{j}\frac{j!}{k!(j-k)!}\mu_1^k\mu_2^{(j-k)}$$

$$= \frac{e^{-(\mu_1+\mu_2)}}{j!}\sum_{k=0}^{j}\binom{j}{k}\mu_1^k\mu_2^{(j-k)} = \frac{e^{-(\mu_1+\mu_2)}(\mu_1+\mu_2)^j}{j!}$$

Multiple Poisson Random Variable

Moment generating function

The moment generating function of a Poisson random variable with parameter μ is $M_x(t) = e^{\mu(e^t - 1)}$.

Multiple Poisson Random Variable

Moment generating function

The moment generating function of a Poisson random variable with parameter μ is $M_x(t) = e^{\mu(e^t-1)}$.

Proof:

For any t,

$$E[e^{tX}] = \sum_{k=0}^{\infty} e^{tk} \frac{e^{-\mu}\mu^k}{k!} = e^{\mu(e^t-1)} \sum_{k=0}^{\infty} \frac{e^{-\mu e^t}(\mu e^t)^k}{k!} = e^{\mu(e^t-1)}.$$

Multiple Poisson Random Variable

Moment generating function

The moment generating function of a Poisson random variable with parameter μ is $M_x(t) = e^{\mu(e^t-1)}$.

Proof:

For any t,

$$E[e^{tX}] = \sum_{k=0}^{\infty} e^{tk} \frac{e^{-\mu}\mu^k}{k!} = e^{\mu(e^t-1)} \sum_{k=0}^{\infty} \frac{e^{-\mu e^t}(\mu e^t)^k}{k!} = e^{\mu(e^t-1)}.$$

Given two independent Poisson random variables X and Y with means μ_1 and μ_2, we have (apply Theorem 4.3 to prove)

$$M_{X+Y}(t) = M_X(t) \cdot M_Y(t) = e^{(\mu_1+\mu_2)(e^t-1)}.$$

which is the moment generating function of a Poisson random variable with mean $\mu_1 + \mu_2$.

Multiple Poisson Random Variable

Moment generating function

The moment generating function of a Poisson random variable with parameter μ is $M_x(t) = e^{\mu(e^t-1)}$.

Proof:

For any t,

$$E[e^{tX}] = \sum_{k=0}^{\infty} e^{tk} \frac{e^{-\mu}\mu^k}{k!} = e^{\mu(e^t-1)} \sum_{k=0}^{\infty} \frac{e^{-\mu e^t}(\mu e^t)^k}{k!} = e^{\mu(e^t-1)}.$$

Given two independent Poisson random variables X and Y with means μ_1 and μ_2, we have (apply Theorem 4.3 to prove)

$$M_{X+Y}(t) = M_X(t) \cdot M_Y(t) = e^{(\mu_1+\mu_2)(e^t-1)}.$$

which is the moment generating function of a Poisson random variable with mean $\mu_1 + \mu_2$. By Theorem 4.2, the moment generating function uniquely defines the distribution, and hence the sum $X + Y$ is a Poisson random variable with mean $\mu_1 + \mu_2$.

Chernoff Bound Revisit

Chernoff bound for Poisson random variable

Let X be a Poisson random variable with parameter μ,

1. If $x > \mu$, then $\Pr(X \geq x) \leq \frac{e^{-\mu}(e\mu)^x}{x^x}$;
2. If $x < \mu$, then $\Pr(X \leq x) \leq \frac{e^{-\mu}(e\mu)^x}{x^x}$.

Proof:

For any $t > 0$ and $x > \mu$, $\Pr(X \geq x) = \Pr(e^{tX} \geq e^{tx}) \leq \frac{E[e^{tX}]}{e^{tx}}$.
Plugging in the expression for the moment generating function of the Poisson distribution, we have $\Pr(X \geq x) \leq e^{\mu(e^t-1)-xt}$.
Choosing $t = \ln(x/\mu) > 0$ gives,

$$\Pr(X \geq x) \leq e^{x-\mu-x\ln(x/\mu)} = \frac{e^{-\mu}(e\mu)^x}{x^x}$$

Chernoff Bound Revisit

Chernoff bound for Poisson random variable

Let X be a Poisson random variable with parameter μ,

1. If $x > \mu$, then $\Pr(X \geq x) \leq \frac{e^{-\mu}(e\mu)^x}{x^x}$;
2. If $x < \mu$, then $\Pr(X \leq x) \leq \frac{e^{-\mu}(e\mu)^x}{x^x}$.

Proof:

For any $t < 0$ and $x < \mu$, $\Pr(X \leq x) = \Pr(e^{tX} \geq e^{tx}) \leq \frac{E[e^{tX}]}{e^{tx}}$.
Hence, $\Pr(X \leq x) \leq e^{\mu(e^t-1)-xt}$.
Choosing $t = \ln(x/\mu) < 0$, it follows that

$$\Pr(X \leq x) \leq e^{x-\mu-x\ln(x/\mu)} = \frac{e^{-\mu}(e\mu)^x}{x^x}$$

Limit of the Binomial Distribution

When throwing m balls randomly into b bins, the probability p_r that a bin has r balls is approximately the Poisson distribution with mean m/b. In general, the Poisson distribution is the limit distribution of the binomial distribution with parameters n and p.

Limit of the Binomial Distribution

Let X_n be a binomial random variable with parameters n and p, where p is a function of n and $\lim_{n\to\infty} np = \lambda$ is a constant that is independent of n. Then for any fixed k,

$$\lim_{n\to\infty} \Pr(X_n = k) = \frac{e^{-\lambda}\lambda^k}{k!}$$

Limit of the Binomial Distribution

When throwing m balls randomly into b bins, the probability p_r that a bin has r balls is approximately the Poisson distribution with mean m/b. In general, the Poisson distribution is the limit distribution of the binomial distribution with parameters n and p.

Limit of the Binomial Distribution

Let X_n be a binomial random variable with parameters n and p, where p is a function of n and $\lim_{n\to\infty} np = \lambda$ is a constant that is independent of n. Then for any fixed k,

$$\lim_{n\to\infty} \Pr(X_n = k) = \frac{e^{-\lambda}\lambda^k}{k!}$$

Proof:

We have $\Pr(X_n = k) = \binom{n}{k} p^k (1-p)^{n-k}$, Since $e^x(1-x^2) \le 1 + x \le e^x$.

$$\Pr(X_n = k) \le \frac{n^k}{k!} p^k \frac{(1-p)^n}{(1-p)^k} \le \frac{(np)^k}{k!} \frac{e^{-pn}}{1-pk} \le \frac{(np)^k e^{-pn}}{k!} \frac{1}{1-pk}$$

Limit of the Binomial Distribution

Limit of the Binomial Distribution

Let X_n be a binomial random variable with parameters n and p, where p is a function of n and $\lim_{n\to\infty} np = \lambda$ is a constant that is independent of n. Then for any fixed k,

$$\lim_{n\to\infty} \Pr(X_n = k) = \frac{e^{-\lambda}\lambda^k}{k!}$$

Proof:

$$\begin{aligned}\Pr(X_n = k) &\geq \frac{(n-k+1)^k}{k!} p^k (1-p)^n \geq \frac{((n-k+1)p)^k}{k!} e^{-pn}(1-p^2)^n \\ &\geq \frac{e^{-pn}((n-k+1)p)^k}{k!}(1-p^2 n)\end{aligned}$$

Limit of the Binomial Distribution

Limit of the Binomial Distribution

Let X_n be a binomial random variable with parameters n and p, where p is a function of n and $\lim_{n\to\infty} np = \lambda$ is a constant that is independent of n. Then for any fixed k,

$$\lim_{n\to\infty} \Pr(X_n = k) = \frac{e^{-\lambda}\lambda^k}{k!}$$

Proof:

$$\Pr(X_n = k) \geq \frac{(n-k+1)^k}{k!} p^k (1-p)^n \geq \frac{((n-k+1)p)^k}{k!} e^{-pn}(1-p^2)^n$$
$$\geq \frac{e^{-pn}((n-k+1)p)^k}{k!}(1-p^2 n)$$

Combining two inequalities, we have

$$\frac{e^{-pn}((n-k+1)p)^k}{k!}(1-p^2 n) \leq \Pr(X_n = k) \leq \frac{(np)^k e^{-pn}}{k!} \frac{1}{1-pk}$$

Limit of the Binomial Distribution

Limit of the Binomial Distribution

Let X_n be a binomial random variable with parameters n and p, where p is a function of n and $\lim_{n\to\infty} np = \lambda$ is a constant that is independent of n. Then for any fixed k,

$$\lim_{n\to\infty} \Pr(X_n = k) = \frac{e^{-\lambda}\lambda^k}{k!}$$

Proof:

In the limit, as n approaches ∞, p approaches 0 because the limiting value of np is the constant λ. Hence $1/(1-pk)$ approaches 1, $1-p^2n$ approaches 1, and the difference between $(n-k+1)p$ and np approaches 0. It follows that

$$\lim_{n\to\infty} \frac{(np)^k e^{-pn}}{k!}\frac{1}{1-pk} = \frac{e^{-\lambda}\lambda^k}{k!}$$

$$\lim_{n\to\infty} \frac{e^{-pn}((n-k+1)p)^k}{k!}(1-p^2n) = \frac{e^{-\lambda}\lambda^k}{k!}$$

Table of Contents

Background: The Poisson Approximation

The main difficulty in analyzing balls-and-bins problems is handling the dependencies that naturally arise in such systems. For example, if we throw m balls into n bins and find that bin 1 is empty, then it is less likely that bin 2 is empty because we know that the m balls must now be distributed among $n-1$ bins. More concretely: if we know, the number of balls in the first $n-1$ bins, then the number of balls in the last bin is completely determined.

Background: The Poisson Approximation

The main difficulty in analyzing balls-and-bins problems is handling the dependencies that naturally arise in such systems. For example, if we throw m balls into n bins and find that bin 1 is empty, then it is less likely that bin 2 is empty because we know that the m balls must now be distributed among $n-1$ bins. More concretely: if we know, the number of balls in the first $n-1$ bins, then the number of balls in the last bin is completely determined.

We have already shown that, after throwing m balls independently and uniformly at random into n bins, the distribution of the number of balls in a given bin is approximately Poisson with mean m/n. We would like to say that the joint distribution of the number of balls in all the bins is well approximated by assuming the load at each bin is an independent Poisson random variable with mean m/n.

Background: The Poisson Approximation

Suppose that m balls are thrown into n bins independently and uniformly at random, and let $X_i^{(m)}$ be the number of balls in the i-th bin, where $1 \leq i \leq n$. Let $Y_1^{(m)}, \cdots, Y_n^{(m)}$ be independent Poisson random variables with mean m/n.

Background: The Poisson Approximation

Suppose that m balls are thrown into n bins independently and uniformly at random, and let $X_i^{(m)}$ be the number of balls in the i-th bin, where $1 \leq i \leq n$. Let $Y_1^{(m)}, \cdots, Y_n^{(m)}$ be independent Poisson random variables with mean m/n.

The difference between throwing m balls randomly and assigning each bin a number of balls that is Poisson distributed with mean m/n is that, in the first case, we know there are m balls in total, whereas in the second case we know only that m is the expected number of balls in all of the bins.

Background: The Poisson Approximation

Suppose that m balls are thrown into n bins independently and uniformly at random, and let $X_i^{(m)}$ be the number of balls in the i-th bin, where $1 \le i \le n$. Let $Y_1^{(m)}, \cdots, Y_n^{(m)}$ be independent Poisson random variables with mean m/n.

The difference between throwing m balls randomly and assigning each bin a number of balls that is Poisson distributed with mean m/n is that, in the first case, we know there are m balls in total, whereas in the second case we know only that m is the expected number of balls in all of the bins.

But suppose when we use the Poisson distribution we end up with m balls. In this case, we do indeed have that the distribution is the same as if we threw m balls into n bins randomly.

Theorem: The Poisson Approximation

Relationships of Possion Approximation

The distribution $Y_1^{(m)}, \cdots, Y_n^{(m)}$ conditioned on $\sum_i Y_i^{(m)} = k$ is the same as $X_1^{(k)}, \cdots, X_n^{(k)}$, regardless of the value of m.

Proof:

When throwing k balls into n bins, the probability that $(X_1^{(k)}, \cdots, X_n^{(k)}) = (k_1, \cdots, k_n)$ for any $k_1, \cdots, k_n$ satisfying $\sum_i k_i = k$ is given by

$$\frac{\binom{k}{k_1, \cdots, k_n}}{n^k} = \frac{k!}{(k_1!)(k_2!)\cdots(k_n!)n^k}.$$

Theorem: The Poisson Approximation

Relationships of Possion Approximation

The distribution $Y_1^{(m)}, \cdots, Y_n^{(m)}$ conditioned on $\sum_i Y_i^{(m)} = k$ is the same as $X_1^{(k)}, \cdots, X_n^{(k)}$, regardless of the value of m.

Proof:

Now, for any $k_1, \cdots, k_n$ with $\sum_i k_i = k$, consider the probability that $(Y_1^{(m)}, \cdots, Y_n^{(m)}) = (k_1, \cdots, k_n)$ conditioned on $(Y_1^{(m)}, \cdots, Y_n^{(m)})$ satisfying $\sum_i Y_i^{(m)} = k$

$$\Pr\Big((Y_1^{(m)}, \cdots, Y_n^{(m)}) = (k_1, \cdots, k_n) \,|\, \sum_i Y_i^{(m)} = k\Big)$$

$$= \frac{\Pr\Big((Y_1^{(m)} = k_1) \cap \cdots \cap (Y_n^{(m)} = k_n)\Big)}{\Pr(\sum_i Y_i^{(m)} = k)}$$

Theorem: The Poisson Approximation

Proof:

The probability that $Y_i^{(m)} = k_i$ is $e^{-m/n}(m/n)^{k_i}/k_i!$, since the $Y_i^{(m)}$ are independent Poisson random variables with mean m/n. Also by Lemma 5.2 (Sum of Poisson), the sum of the $Y_i^{(m)}$ is itself a Poisson random variable with mean m. Hence,

$$\begin{aligned}\frac{\Pr\Big((Y_1^{(m)} = k_1) \cap \cdots \cap (Y_n^{(m)} = k_n)\Big)}{\Pr(\sum_i Y_i^{(m)} = k)} &= \frac{\prod_{i=1}^{n} e^{-m/n}(m/n)^{k_i}/k_i!}{e^{-m}m^k/k!} \\ &= \frac{k!}{(k_1!)(k_2!)\cdots(k_n!)n^k}\end{aligned}$$

Strong Theorem: The Poisson Approximation

Relationships of Possion Approximation

Let $f(X_1, \cdots, X_n)$ be a non-negative function. Then

$$E[f(X_1^{(m)}, \cdots, X_n^{(m)})] \le e\sqrt{m}E[f(Y_1^{(m)}, \cdots, Y_n^{(m)})]$$

Proof:

We have that

$$E[f(Y_1^{(m)}, \cdots, Y_n^{(m)})] = \sum_{k=0}^{\infty} E\left[f(Y_1^{(m)}, \cdots, Y_n^{(m)}) \mid \sum_{i=1}^{n} Y_i^{(m)} = k\right] \Pr(\sum_{i=1}^{n} Y_i^{(m)} = k)$$

$$\ge E\left[f(Y_1^{(m)}, \cdots, Y_n^{(m)}) \mid \sum_{i=1}^{n} Y_i^{(m)} = m\right] \Pr(\sum_{i=1}^{n} Y_i^{(m)} = m)$$

$$\ge E\left[f(X_1^{(m)}, \cdots, X_n^{(m)})\right] \Pr(\sum_{i=1}^{n} Y_i^{(m)} = m)$$

Strong Theorem: The Poisson Approximation

Relationships of Possion Approximation

Let $f(X_1, \cdots, X_n)$ be a non-negative function. Then

$$E[f(X_1^{(m)}, \cdots, X_n^{(m)})] \leq e\sqrt{m}E[f(Y_1^{(m)}, \cdots, Y_n^{(m)})]$$

Proof:

We have that

$$\begin{aligned} E[f(Y_1^{(m)}, \cdots, Y_n^{(m)})] &= \sum_{k=0}^{\infty} E\Big[f(Y_1^{(m)}, \cdots, Y_n^{(m)}) \mid \sum_{i=1}^{n} Y_i^{(m)} = k\Big] \Pr(\sum_{i=1}^{n} Y_i^{(m)} = k) \\ &\geq E\Big[f(Y_1^{(m)}, \cdots, Y_n^{(m)}) \mid \sum_{i=1}^{n} Y_i^{(m)} = m\Big] \Pr(\sum_{i=1}^{n} Y_i^{(m)} = m) \\ &\geq E\Big[f(X_1^{(m)}, \cdots, X_n^{(m)})\Big] \Pr(\sum_{i=1}^{n} Y_i^{(m)} = m) \end{aligned}$$

Since $\sum_{i=1}^{n} Y_i^{(m)} = m$ is Poisson distribution with mean m, we have

$$E[f(Y_1^{(m)}, \cdots, Y_n^{(m)})] \geq E\Big[f(X_1^{(m)}, \cdots, X_n^{(m)})\Big] \frac{m^m e^{-m}}{m!}$$

Strong Theorem: The Poisson Approximation

Relationships of Possion Approximation

Let $f(X_1, \cdots, X_n)$ be a non-negative function. Then

$$E[f(X_1^{(m)}, \cdots, X_n^{(m)})] \leq e\sqrt{m}E[f(Y_1^{(m)}, \cdots, Y_n^{(m)})]$$

Proof:

We use the following loose bound on $m!$, which proves later:

$$m! < e\sqrt{m}(\frac{m}{e})^m.$$

Strong Theorem: The Poisson Approximation

Relationships of Possion Approximation

Let $f(X_1, \cdots, X_n)$ be a non-negative function. Then

$$E[f(X_1^{(m)}, \cdots, X_n^{(m)})] \leq e\sqrt{m}E[f(Y_1^{(m)}, \cdots, Y_n^{(m)})]$$

Proof:

We use the following loose bound on $m!$, which proves later:

$$m! < e\sqrt{m}(\frac{m}{e})^m.$$

This yields,

$$E[f(Y_1^{(m)}, \cdots, Y_n^{(m)})] \geq E\left[f(X_1^{(m)}, \cdots, X_n^{(m)})\right]\frac{1}{e\sqrt{m}}$$

Lemma: Loose Bound on $m!$

Loose Bound on $m!$

$$n! \leq e\sqrt{n}(\frac{n}{e})^n$$

Proof:

We use the fact $\ln(n!) = \sum_{i=1}^{n} \ln i$. We first claim that, for $i \geq 2$,

$$\int_{i-1}^{i} \ln x dx \geq \frac{\ln(i-1) + \ln i}{2}.$$

Lemma: Loose Bound on $m!$

Loose Bound on $m!$

$$n! \leq e\sqrt{n}(\frac{n}{e})^n$$

Proof:

We use the fact $\ln(n!) = \sum_{i=1}^{n} \ln i$. We first claim that, for $i \geq 2$,

$$\int_{i-1}^{i} \ln x dx \geq \frac{\ln(i-1) + \ln i}{2}.$$

This follows from the fact that $\ln x$ is concave, since its second derivative is $-1/x^2$, which is always negative. Therefore,

$$\int_{1}^{n} \ln x dx \geq \sum_{i=1}^{n} \ln i - \frac{\ln n}{2}.$$

or, equivalently,

$$n \ln n - n + 1 \geq \ln(n!) - \frac{\ln n}{2}$$

Corollary: The Poisson Approximation

Theorem holds for any non-negative function on the number of balls in the bins. In particular, if the function is the indicator function that is 1 if some event occurs and 0 otherwise, then the theorem gives bounds on the probability of events.

Corollary: The Poisson Approximation

Theorem holds for any non-negative function on the number of balls in the bins. In particular, if the function is the indicator function that is 1 if some event occurs and 0 otherwise, then the theorem gives bounds on the probability of events.

Let us call the scenario in which the number of balls in the bins are taken to be independent Poisson random variables with mean m/n the Poisson case, and the scenario where m balls are thrown into n bins independently and uniformly at random the exact case.

Corollary

Any event that takes place with probability p in the Poisson case takes place with probability at most $e\sqrt{m} \cdot p$ in the exact case.

Strong Theorem on Special Case: The Poisson Approximation

Strong Theorem on Special Case

Let $f(x_1, \cdots, x_n)$ be a non-negative function such that $E[f(X_1^{(m)}, \cdots, X_n^{(m)})]$ is either monotonically increasing or monotonically decreasing in m. Then,

$$E[f(X_1^{(m)}, \cdots, X_n^{(m)})] \le 2E[f(Y_1^{(m)}, \cdots, Y_n^{(m)})]$$

Strong Theorem on Special Case: The Poisson Approximation

Strong Theorem on Special Case

Let $f(x_1, \cdots, x_n)$ be a non-negative function such that $E[f(X_1^{(m)}, \cdots, X_n^{(m)})]$ is either monotonically increasing or monotonically decreasing in m. Then,

$$E[f(X_1^{(m)}, \cdots, X_n^{(m)})] \leq 2E[f(Y_1^{(m)}, \cdots, Y_n^{(m)})]$$

Corollary

Let ε be an event whose probability is either monotonically increasing or monotonically decreasing in the number of balls. If ε has probability p in the Poisson case, then ε has probability at most $2p$ in the exact case.

Example: Usage of Corollary

Lemma

When n balls are thrown independently and uniformly at random into n bins, the maximum load is at least $\ln n / \ln \ln n$ with probability at least $1 - 1/n$ for n sufficiently large.

Example: Usage of Corollary

Lemma

When n balls are thrown independently and uniformly at random into n bins, the maximum load is at least $\ln n / \ln \ln n$ with probability at least $1 - 1/n$ for n sufficiently large.

Proof:

In the Poisson case, the probability that bin 1 has load at least $M = \ln n / \ln \ln n$ is at least $1/(eM!)$, which is the probability it has load exactly M. In the Poisson case, all bins are independent, so the probability that no bin has load at least M is at most $(1 - \frac{1}{eM!})^n \leq e^{-\frac{n}{eM!}}$.
We now need to choose M so that $e^{-n/(eM!)} < n^{-2}$, for then (by General Theorem) we will have that the probability that the maximum load is not at least M in the exact case is at most $e\sqrt{n}/n^2 < 1/n$.

Example: Usage of Corollary

Lemma

When n balls are thrown independently and uniformly at random into n bins, the maximum load is at least $\ln n/\ln\ln n$ with probability at least $1 - 1/n$ for n sufficiently large.

Proof:

We need to prove $M! \leq n/(2e\ln n)$, or equivalently that $\ln(M!) \leq \ln n - \ln\ln n - \ln(2e)$. According to the loose bound of $n!$,
$M! \leq e\sqrt{M}(\frac{M}{e})^M \leq M(\frac{M}{e})^M$.
when n (and hence $M = \ln n/\ln\ln n$) are suitably large,

$$\begin{aligned}\ln M! &\leq M\ln M - M + \ln M\\ &= \frac{\ln n}{\ln\ln n}(\ln\ln n - \ln\ln\ln n) - \frac{\ln n}{\ln\ln n} + (\ln\ln n - \ln\ln\ln n)\\ &\leq \ln n - \frac{\ln n}{\ln\ln n} \leq \ln n - \ln\ln n - \ln(2e)\end{aligned}$$

Table of Contents

Application: Chain Hashing

The balls-and-bins-model is also useful for modeling hashing.
A hash function f from a universe U into a range $[0, n-1]$ can be thought of as a way of placing items from the universe into n bins.

Application: Chain Hashing

The balls-and-bins-model is also useful for modeling hashing.
A hash function f from a universe U into a range $[0, n-1]$ can be thought of as a way of placing items from the universe into n bins.

Chain Hashing and Ball-and-Bin

We simply model the problem by assuming that hash functions are random. In other words, we assume that (a) for each $x \in U$, the probability that $f(x) = j$ is $1/n$ (for $0 \leq j \leq n-1$) and that (b) the values of $f(x)$ for each x are independent of each other.

Application: Chain Hashing

Chain Hashing and Ball-and-Bin

We simply model the problem by assuming that hash functions are random. In other words, we assume that (a) for each $x \in U$, the probability that $f(x) = j$ is $1/n$ (for $0 \leq j \leq n-1$) and that (b) the values of $f(x)$ for each x are independent of each other.

Example

Let us consider the search time when there are n bins and m items.

Application: Chain Hashing

Chain Hashing and Ball-and-Bin

We simply model the problem by assuming that hash functions are random. In other words, we assume that (a) for each $x \in U$, the probability that $f(x) = j$ is $1/n$ (for $0 \leq j \leq n-1$) and that (b) the values of $f(x)$ for each x are independent of each other.

Example

Let us consider the search time when there are n bins and m items.
If we search for a item that is not in our dictionary, the expected number of words the item hashes to is m/n.

Application: Chain Hashing

Chain Hashing and Ball-and-Bin

We simply model the problem by assuming that hash functions are random. In other words, we assume that (a) for each $x \in U$, the probability that $f(x) = j$ is $1/n$ (for $0 \leq j \leq n-1$) and that (b) the values of $f(x)$ for each x are independent of each other.

Example

Let us consider the search time when there are n bins and m items.
If we search for a item that is not in our dictionary, the expected number of words the item hashes to is m/n.
If we search for a item that is in our dictionary, the expected number of other words is $(m-1)/n$, so the expected number of words the item hashes to is $1 + (m - l)/n$.

Application: Chain Hashing

Chain Hashing and Ball-and-Bin

We simply model the problem by assuming that hash functions are random. In other words, we assume that (a) for each $x \in U$, the probability that $f(x) = j$ is $1/n$ (for $0 \leq j \leq n-1$) and that (b) the values of $f(x)$ for each x are independent of each other.

Example

Let us consider the search time when there are n bins and m items.
If we search for a item that is not in our dictionary, the expected number of words the item hashes to is m/n.
If we search for a item that is in our dictionary, the expected number of other words is $(m-1)/n$, so the expected number of words the item hashes to is $1 + (m-l)/n$.
If we choose $n = m$ bins for our hash table, then the expected number of words we must search through is constant. If the hashing takes constant time, then the total expected time for search is constant.

Application: Chain Hashing

Chain Hashing and Ball-and-Bin

We simply model the problem by assuming that hash functions are random. In other words, we assume that (a) for each $x \in U$, the probability that $f(x) = j$ is $1/n$ (for $0 \le j \le n-1$) and that (b) the values of $f(x)$ for each x are independent of each other.

Example

The maximum time to search for a word, however, is proportional to the maximum number of words in a bin.
We have shown that when $n = m$ this maximum load is $\Theta(\ln n / \ln \ln n)$ with probability close to 1, and hence with high probability this is the maximum search time in such a hash table.
While this is still faster than the required time for standard binary search $\Theta(\log m)$, it is much slower than the average, which can be a drawback for many applications.

Application: Hashing of Bit Strings

Search String in the Dictionary

Suppose we use a hash function to map each word into a 32-bit string. This string will serve as a short fingerprint for the word.
We keep the fingerprints in a sorted list. To search a string, we calculate its fingerprint and look for it on the list with binary search.

Example

In this case, it may not give the correct answer! The word is actually not in the dictionary while its fingerprint matches the fingerprint of another string. Hence there is some chance that hashing will yield a false positive: it may falsely declare a match when there is not an actual match.
The probability that a string has a fingerprint that is different from any specific words in dictionary S is $(1 - 1/2^b)$. It follows that if the set S has size m and if we use b bits for the fingerprint, then the probability of a false positive for a search string is $1 - (1 - 1/2^b)^m \geq 1 - e^{-m/2^b}$.

Application: Hashing of Bit Strings

Proof:

If we want this probability of a false positive to be less than a constant c, we need $e^{-m/2^b} \geq 1 - c$. which implies that $b \geq \log \frac{m}{\ln(1/(1-c))}$. That is, we need $b = \Omega(\log m)$ bits. On the other hand, if we use $b = 2\log m$ bits, then the probability of a false positive falls to

$$1 - (1 - 1/2^b)^m = 1 - (1 - \frac{1}{m^2})^m < \frac{1}{m}$$

Application: Hashing of Bit Strings

Proof:

If we want this probability of a false positive to be less than a constant c, we need $e^{-m/2^b} \geq 1 - c$. which implies that $b \geq \log \frac{m}{\ln(1/(1-c))}$. That is, we need $b = \Omega(\log m)$ bits. On the other hand, if we use $b = 2\log m$ bits, then the probability of a false positive falls to

$$1 - (1 - 1/2^b)^m = 1 - (1 - \frac{1}{m^2})^m < \frac{1}{m}$$

Example

In our example, if our dictionary has $2^{16} = 65536$ words, then using 32 bits when hashing yields a false positive probability of just less than $1/65536 \approx 0.0015\%$.

Application: Bloom Filters

Definition: Bloom Filters

A Bloom filter consists of an array of n bits, $A[0]$ to $A[n-1]$, initially all set to 0. A Bloom filter uses k independent random hash functions $h_1, \cdots, h_k$ with range $0, \cdots, n-1$. Suppose that we use a Bloom filter to represent a set $S = \{s_1, s_2, \cdots, s_n\}$ of n elements from a large universe U. For each element $s \in S$, the bits $A[h_i(s)]$ are set to 1 for $1 \leq i \leq k$. To check if an element x is in S, we check whether all array locations $A[h_i(x)]$ for $1 \leq i \leq k$ are set to 1. If not, then clearly x is not a member of S; If "yes", we assume that x is a member of S.

Application: Bloom Filters

Definition: Bloom Filters

A Bloom filter consists of an array of n bits, $A[0]$ to $A[n-1]$, initially all set to 0. A Bloom filter uses k independent random hash functions $h_1, \cdots, h_k$ with range $0, \cdots, n-1$. Suppose that we use a Bloom filter to represent a set $S = \{s_1, s_2, \cdots, s_n\}$ of n elements from a large universe U. For each element $s \in S$, the bits $A[h_i(s)]$ are set to 1 for $1 \leq i \leq k$. To check if an element x is in S, we check whether all array locations $A[h_i(x)]$ for $1 \leq i \leq k$ are set to 1. If not, then clearly x is not a member of S; If "yes", we assume that x is a member of S.

Probability of False Positive

After all elements in S are mapped, the probability that $A[i] = 0$ is $p = (1 - 1/n)^{km} \approx e^{-km/n}$. Thus, the probability of a false positive is no more than $f = (1-p)^k$.
We want to find the optimal k, i.e., the number of hash functions.

Application: Bloom Filters

The Optimal k

$p = (1 - 1/n)^{km} \approx e^{-km/n}$, $f = (1 - p)^k$.
Let $g = k \ln(1 - e^{-km/n})$, so that $f = e^g$ and minimizing the false positive probability f is equivalent to minimizing g w.r.t k.

$$\frac{dg}{dk} = \ln(1 - e^{-km/n}) + \frac{km}{n} \frac{e^{-km/n}}{1 - e^{-km/n}}$$

It is easy to check that the derivative is zero when $k = (\ln 2) \cdot (n/m)$ and that this point is a global minimum. In this case the false positive probability f is $(1/2)^k \approx (0.6185)^{n/m}$.

Application: Bloom Filters

The Fraction of Zero Entries

we reconsider our assumption that the fraction of entries that are still 0 after all of the elements of S are hashed into the Bloom filter is p. Each bit in the array can be thought of as a bin, and hashing an item is like throwing a ball. The fraction of entries that are still 0 after all of the elements of S are hashed is therefore equivalent to the fraction of empty bins after mk balls are thrown into n bins.

Application: Bloom Filters

The Fraction of Zero Entries

we reconsider our assumption that the fraction of entries that are still 0 after all of the elements of S are hashed into the Bloom filter is p. Each bit in the array can be thought of as a bin, and hashing an item is like throwing a ball. The fraction of entries that are still 0 after all of the elements of S are hashed is therefore equivalent to the fraction of empty bins after mk balls are thrown into n bins.

Let X be the number of such bins when mk balls are thrown. The expected fraction of such bins is $(1-1/n)^{km}$. The events of different bins being empty are not independent. but we can apply Corollary, along with Chernoff bound, to obtain

$$\Pr(|X-np'| \geq \varepsilon n) \leq 2e\sqrt{n}e^{-n\varepsilon^2/3p'}$$

Application: Bloom Filters

The Fraction of Zero Entries

we reconsider our assumption that the fraction of entries that are still 0 after all of the elements of S are hashed into the Bloom filter is p. Each bit in the array can be thought of as a bin, and hashing an item is like throwing a ball. The fraction of entries that are still 0 after all of the elements of S are hashed is therefore equivalent to the fraction of empty bins after mk balls are thrown into n bins.

Let X be the number of such bins when mk balls are thrown. The expected fraction of such bins is $(1 - 1/n)^{km}$. The events of different bins being empty are not independent. but we can apply Corollary, along with Chernoff bound, to obtain

$$\Pr(|X - np'| \geq \varepsilon n) \leq 2e\sqrt{n}e^{-n\varepsilon^2/3p'}$$

The bound tells us that the fraction of empty bins is close to p' (when n is reasonably large) and that p' is very close to p.

Table of Contents

Two Models of Random Graphs

Most of the work on random graphs has focused on two closely related models, $G_{n,p}$ and $G_{n,N}$.

Definition: Model $G_{n,p}$

In $G_{n,p}$ we consider all undirected graphs on n distinct vertices $v_1, v_2, \cdots, v_n$. A graph with a given set of m edges has probability $p^m(1-p)^{\binom{n}{2}-m}$.

Two Models of Random Graphs

Most of the work on random graphs has focused on two closely related models, $G_{n,p}$ and $G_{n,N}$.

Definition: Model $G_{n,p}$

In $G_{n,p}$ we consider all undirected graphs on n distinct vertices $v_1, v_2, \cdots, v_n$. A graph with a given set of m edges has probability $p^m(1-p)^{\binom{n}{2}-m}$.

Example

One way to generate a random graph in $G_{n,p}$ is to consider each of the $\binom{n}{2}$ possible edges in some order and then independently add each edge to the graph with probability p. The expected number of edges in the graph is therefore $\binom{n}{2}p$, and each vertex has expected degree $(n-1)p$.

Two Models of Random Graphs

Definition: Model $G_{n,N}$

In the $G_{n,N}$ model, we consider all undirected graphs on n vertices with exactly N edges. There are $\binom{\binom{n}{2}}{N}$ possible graphs, each selected with equal probability.

Example

Two Models of Random Graphs

Definition: Model $G_{n,N}$

In the $G_{n,N}$ model, we consider all undirected graphs on n vertices with exactly N edges. There are $\binom{\binom{n}{2}}{N}$ possible graphs, each selected with equal probability.

Example

- One way to generate a graph uniformly from the graphs in $G_{n,N}$ is to start with a graph with no edges.

Two Models of Random Graphs

Definition: Model $G_{n,N}$

In the $G_{n,N}$ model, we consider all undirected graphs on n vertices with exactly N edges. There are $\binom{\binom{n}{2}}{N}$ possible graphs, each selected with equal probability.

Example

- One way to generate a graph uniformly from the graphs in $G_{n,N}$ is to start with a graph with no edges.
- Choose one of the $\binom{n}{2}$ possible edges uniformly at random and add it to the edges.

Two Models of Random Graphs

Definition: Model $G_{n,N}$

In the $G_{n,N}$ model, we consider all undirected graphs on n vertices with exactly N edges. There are $\binom{\binom{n}{2}}{N}$ possible graphs, each selected with equal probability.

Example

- One way to generate a graph uniformly from the graphs in $G_{n,N}$ is to start with a graph with no edges.
- Choose one of the $\binom{n}{2}$ possible edges uniformly at random and add it to the edges.
- Now choose one of the remaining $\binom{n}{2} - 1$ possible edges independently and uniformly at random and add it to the graph.

Two Models of Random Graphs

Definition: Model $G_{n,N}$

In the $G_{n,N}$ model, we consider all undirected graphs on n vertices with exactly N edges. There are $\binom{\binom{n}{2}}{N}$ possible graphs, each selected with equal probability.

Example

- One way to generate a graph uniformly from the graphs in $G_{n,N}$ is to start with a graph with no edges.
- Choose one of the $\binom{n}{2}$ possible edges uniformly at random and add it to the edges.
- Now choose one of the remaining $\binom{n}{2} - 1$ possible edges independently and uniformly at random and add it to the graph.
- Similarly, continue choosing one of the remaining unchosen edges independently and uniformly at random until there are N edges.

Two Models of Random Graphs

Similarity Between Two Models

The $G_{n,p}$ and $G_{n,N}$ models are related: when $p = N/\binom{n}{2}$, the number of edges in a random graph in $G_{n,p}$ is concentrated around N, and conditioned on a graph from $G_{n,p}$ having N edges, that graph is uniform over all the graphs from $G_{n,N}$.

Two Models of Random Graphs

Similarity Between Two Models

The $G_{n,p}$ and $G_{n,N}$ models are related: when $p = N/\binom{n}{2}$, the number of edges in a random graph in $G_{n,p}$ is concentrated around N, and conditioned on a graph from $G_{n,p}$ having N edges, that graph is uniform over all the graphs from $G_{n,N}$.

The relationship is similar to the relationship between throwing m balls into n bins and having each bin have a Poisson distributed number of balls with mean m/n.

Random Graphs and Ball-and-Bin

Similarity Between Random Graphs and Ball-and-Bin

Throwing edges into the graph as in the $G_{n,N}$ model is like throwing balls into bins. However, since each edge has two endpoints, each edge is like throwing two balls at once into two different bins.

Random Graphs and Ball-and-Bin

Similarity Between Random Graphs and Ball-and-Bin

Throwing edges into the graph as in the $G_{n,N}$ model is like throwing balls into bins. However, since each edge has two endpoints, each edge is like throwing two balls at once into two different bins.

The pairing defined by the edges adds a rich structure that does not exist in the balls-and-bins model. Yet we can often utilize the relation between the two models to simplify analysis in random graph models. For example, in the coupon collector's problem we found that when we throw $n \ln n + cn$ balls, the probability that there are any empty bins converges to $e^{-e^{-c}}$ as n grows to infinity.

Random Graphs and Ball-and-Bin

Similarity Between Random Graphs and Ball-and-Bin

Throwing edges into the graph as in the $G_{n,N}$ model is like throwing balls into bins. However, since each edge has two endpoints, each edge is like throwing two balls at once into two different bins.

The pairing defined by the edges adds a rich structure that does not exist in the balls-and-bins model. Yet we can often utilize the relation between the two models to simplify analysis in random graph models. For example, in the coupon collector's problem we found that when we throw $n \ln n + cn$ balls, the probability that there are any empty bins converges to $e^{-e^{-c}}$ as n grows to infinity.

Probability of Zero Degree Vertices

Let $N = \frac{1}{2}(n \ln n + cn)$. Then the probability that there are any isolated vertices (vertices with degree 0) in $G_{n,V}$ converges to $e^{-e^{-c}}$ as n grows to infinity.

Application: Hamiltonian Cycles in Random Graphs

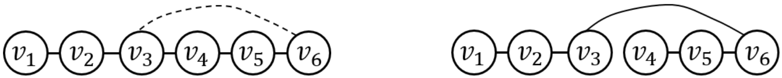
Fundamental Operation: Rotation

Let G be an undirected graph. Suppose that $P = v_1, v_2, \cdots, v_k$ is a simple path in G and that (v_k, v_i) is an edge of G. Then $P' = v_1, v_2, \cdots, v_i, v_k, v_{k-1}, \cdots, v_{i+2}, v_{i+1}$.

Application: Hamiltonian Cycles in Random Graphs

Fundamental Operation: Rotation

Let G be an undirected graph. Suppose that $P = v_1, v_2, \cdots, v_k$ is a simple path in G and that (v_k, v_i) is an edge of G. Then $P' = v_1, v_2, \cdots, v_i, v_k, v_{k-1}, \cdots, v_{i+2}, v_{i+1}$.



(a) Path P (b) Path P' after rotation

Figure 1: An example of rotation

Deterministic Algorithm: Hamiltonian Cycles

```
Algorithm 1: Hamiltonian Cycle Algorithm
input  : A graph $G = (V, E)$ with $n$ vertices.
output: A Hamiltonian cycle, or failure.
Start with a random vertex as the head of the path;
repeat
    Let the current path be $P = v_1, v_2, \cdots, v_k$, where $v_k$ is the head,
     and let $(v_k, u)$ be the first edge in the head's list;
    Remove $(v_k, u)$ from the head's list and u's list;
    if $u \neq v_i, \forall 1 \leq i \leq k$ then
        Add $u = v_{k+1}$ to the end of the path and make it the head.
    else if $u = v_i$ then
        Rotate current path with $(v_k, v_i)$ and set $v_{i+1}$ to be head.
until the rotation edge closes a Hamiltonian cycle or the unused-edges
 list of the head of the path is empty;
if No cycle was found then return Failure ;
else return the Hamiltonian cycle ;
```

Randomized Algorithm: Hamiltonian Cycles

Algorithm 2: Modified Hamiltonian Cycle Algorithm

```
repeat
    Let the current path be $P = v_1, v_2, \cdots, v_k$, where $v_k$ is head;
    Execute (1), (2), or (3) with probabilities $1/n$, $|\text{used-edges}(v_k)|/n$,
     and $1 - 1/n - |\text{used-edges}(v_k)|/n$, respectively;
    (1). Reverse the path and make $v_1$ the head;
    (2). Choose uniformly at random an edge from used-edges($v_k$);
    if the edge is $(v_k, v_i)$ then
        Rotate current path with $(v_k, v_i)$ and set $v_{i+1}$ to be head;
    (3). Select the first edge from unused-edges($v_k$), call it $(v_k, u)$;
    if $u \neq v_i, \forall 1 \leq i \leq k$ then
        Add $u = v_{k+1}$ to the end of the path and make it the head;
    else if $u = v_i$ then
        Rotate current path with $(v_k, v_i)$ and set $v_{i+1}$ to be head;
until $\cdots$;
```

A new model of random graphs

We use a new model for undirected graph that will be proved to be related to $G_{n,p}$ model.

A new model of random graphs

1. Assume that each of the $n-1$ possible edges connected to a vertex v is initially on the unused-edges list for vertex v independently with some probability q. We also assume these edges are in a random order

A new model of random graphs

We use a new model for undirected graph that will be proved to be related to $G_{n,p}$ model.

A new model of random graphs

1. Assume that each of the $n-1$ possible edges connected to a vertex v is initially on the unused-edges list for vertex v independently with some probability q. We also assume these edges are in a random order
2. Specifically, we create the unused-edges list for each vertex v by inserting each possible edge (v, u) with probability q;

A new model of random graphs

We use a new model for undirected graph that will be proved to be related to $G_{n,p}$ model.

A new model of random graphs

1. Assume that each of the $n-1$ possible edges connected to a vertex v is initially on the unused-edges list for vertex v independently with some probability q. We also assume these edges are in a random order
2. Specifically, we create the unused-edges list for each vertex v by inserting each possible edge (v,u) with probability q;
3. Notice that this means an edge (v,u) could initially be on the unused-edges list for v but not for u. Also, when an edge (v,u) is first used in the algorithm. if v is the head then it is removed just from the unused-edges list of v; if the edge is on the unused-edges list for u, it remains on this list.

A new model of random graphs

We use a new model for undirected graph that will be proved to be related to $G_{n,p}$ model.

A new model of random graphs

1. Assume that each of the $n-1$ possible edges connected to a vertex v is initially on the unused-edges list for vertex v independently with some probability q. We also assume these edges are in a random order
2. Specifically, we create the unused-edges list for each vertex v by inserting each possible edge (v, u) with probability q;
3. Notice that this means an edge (v, u) could initially be on the unused-edges list for v but not for u. Also, when an edge (v, u) is first used in the algorithm. if v is the head then it is removed just from the unused-edges list of v; if the edge is on the unused-edges list for u, it remains on this list.

We modify the rotation process so that the next head of the list is chosen uniformly at random from among all vertices of the graph.

Thereotical Analysis: Randomized Algorithm

Lemma

Suppose the modified Hamiltonian cycle algorithm is run on a graph chosen using the new random Graphs model. Let V_t be the head vertex after the t-th step. Then, for any vertex u, as long as at the t-th step there is at least one unused edge available at the head vertex,

$$\Pr(V_{t+1} = u \mid V_t = u_t, V_{t-1} = u_{t-1}, \cdots, V_0 = u_0) = 1/n.$$

That is, the head vertex can be thought of as being chosen uniformly at random from all vertices at each step, regardless of the history of the process.

Thereotical Analysis: Randomized Algorithm

Proof:

Consider the possible cases when the path is $P = v_1, v_2, \cdots, v_k$.

1. The only way v_1 can become the head is if the path is reversed, so $V_{t+1} = v_1$ with probability $1/n$.

Thereotical Analysis: Randomized Algorithm

Proof:

Consider the possible cases when the path is $P = v_1, v_2, \cdots, v_k$.

1. The only way v_1 can become the head is if the path is reversed, so $V_{t+1} = v_1$ with probability $1/n$.
2. If $u = v_{i+1}$ is a vertex that lies on the path and (v_k, v_i) is in used-edges(v_k), then the probability that $V_{t+1} = u$ is
$$\frac{|\text{used-edges}(v_k)|}{n} \frac{1}{|\text{used-edges}(v_k)|} = \frac{1}{n}.$$

Thereotical Analysis: Randomized Algorithm

Proof:

Consider the possible cases when the path is $P = v_1, v_2, \cdots, v_k$.

1. The only way v_1 can become the head is if the path is reversed, so $V_{t+1} = v_1$ with probability $1/n$.
2. If $u = v_{i+1}$ is a vertex that lies on the path and (v_k, v_i) is in used-edges(v_k), then the probability that $V_{t+1} = u$ is
$$\frac{|\text{used-edges}(v_k)|}{n} \frac{1}{|\text{used-edges}(v_k)|} = \frac{1}{n}.$$
3. If u is not covered by the first two cases then we use the fact that, when an edge is chosen from unused-edges(v_k), the adjacent vertex is uniform over all the $n - |\text{used-edges}(v_k)| - 1$ remaining vertices.

Thereotical Analysis: Randomized Algorithm

Proof:

Consider the possible cases when the path is $P = v_1, v_2, \cdots, v_k$.

1. The only way v_1 can become the head is if the path is reversed, so $V_{t+1} = v_1$ with probability $1/n$.
2. If $u = v_{i+1}$ is a vertex that lies on the path and (v_k, v_i) is in used-edges(v_k), then the probability that $V_{t+1} = u$ is
$$\frac{|\text{used-edges}(v_k)|}{n} \frac{1}{|\text{used-edges}(v_k)|} = \frac{1}{n}.$$
3. If u is not covered by the first two cases then we use the fact that, when an edge is chosen from unused-edges(v_k), the adjacent vertex is uniform over all the $n - |\text{used-edges}(v_k)| - 1$ remaining vertices. Because v_k's list was determined independently from the lists of the other vertices, the history of the algorithm tells us nothing about the remaining edges in unused-edges(v_k), and the principle of deferred decisions applies.

Thereotical Analysis: Randomized Algorithm

Proof:

(a) If $u = v_{i+1}$ is a vertex on the path but (v_k, v_i) is not in used-edges(v_k), then the probability that $v_{t+1} = u$ is the probability that the edge (v_k, v_i) is chosen from used-edges(v_k) as the next rotation edge, which is

$$\left(1 - \frac{1}{n} - \frac{|\text{used-edges}(v_k)|}{n}\right)\left(\frac{1}{n - 1 - |\text{used-edges}(v_k)|}\right) = \frac{1}{n} \tag{1}$$

Thereotical Analysis: Randomized Algorithm

Proof:

(a) If $u = v_{i+1}$ is a vertex on the path but (v_k, v_i) is not in used-edges(v_k), then the probability that $v_{t+1} = u$ is the probability that the edge (v_k, v_i) is chosen from used-edges(v_k) as the next rotation edge, which is

$$\left(1 - \frac{1}{n} - \frac{|\text{used-edges}(v_k)|}{n}\right)\left(\frac{1}{n - 1 - |\text{used-edges}(v_k)|}\right) = \frac{1}{n} \quad (1)$$

(b) Finally, if u is not on the path, then the probability that $V_{t+1} = u$ is the probability that the edge (v_{k+1}, u) is chosen from unused-edges(v_k). But this has the same probability as in Eq.(1).

Thereotical Analysis: Randomized Algorithm

Proof:

a. If $u = v_{i+1}$ is a vertex on the path but (v_k, v_i) is not in used-edges(v_k), then the probability that $v_{t+1} = u$ is the probability that the edge (v_k, v_i) is chosen from used-edges(v_k) as the next rotation edge, which is

$$\left(1 - \frac{1}{n} - \frac{|\text{used-edges}(v_k)|}{n}\right)\left(\frac{1}{n - 1 - |\text{used-edges}(v_k)|}\right) = \frac{1}{n} \qquad (1)$$

b. Finally, if u is not on the path, then the probability that $V_{t+1} = u$ is the probability that the edge (v_{k+1}, u) is chosen from unused-edges(v_k). But this has the same probability as in Eq.(1).

The problem of finding a Hamiltonian path looks exactly like the coupon collector's problem: the probability of finding a new vertex to add to the path when there are k vertices left to be added is k/n. Once all the vertices are on the path, the probability that a cycle is closed in each rotation is $1/n$.

Thereotical Analysis: Randomized Algorithm

Theorem

Suppose the input to the modified Hamiltonian cycle algorithm initially has unused-edge lists where each edge (v, u) with $u \neq v$ is placed on v's list independently with probability $q \geq 20 \ln n / n$. Then the algorithm successfully finds a Hamiltonian cycle in $O(n \ln n)$ iterations of the repeat loop with probability $1 - O(n^{-1})$.

Thereotical Analysis: Randomized Algorithm

Theorem

Suppose the input to the modified Hamiltonian cycle algorithm initially has unused-edge lists where each edge (v, u) with $u \neq v$ is placed on v's list independently with probability $q \geq 20 \ln n/n$. Then the algorithm succeessfully finds a Hamiltonian cycle in $O(n \ln n)$ iterations of the repeat loop with probability $1 - O(n^{-1})$.

Note that we did not assume that the input random graph has a Hamiltonian cycle. A corollary of the theorem is that, with high probability, a random graph chosen in this way has a Hamiltonian cycle.

Thereotical Analysis: Randomized Algorithm

Proof:

Consider the following two events:

ε_1: The algorithm ran for $3n \ln n$ steps with no unused-edges list becoming empty, but it failed to construct a Hamiltonian cycle.

ε_2: At least one unused-edges list became empty during the first $3n \ln n$ iterations of the loop.

For the algorithm to fail, either event ε_1 or ε_2 must occur.

Thereotical Analysis: Randomized Algorithm

Proof:

Consider the following two events:

ε_1: The algorithm ran for $3n \ln n$ steps with no unused-edges list becoming empty, but it failed to construct a Hamiltonian cycle.

ε_2: At least one unused-edges list became empty during the first $3n \ln n$ iterations of the loop.

For the algorithm to fail, either event ε_1 or ε_2 must occur.

- Previous Lemma implies that head is chosen uniformly at random each iteration. The probability that it takes more than $2n \ln n$ iterations to find a Hamiltonian path is exactly the probability that a coupon collector's problem on n types (*i.e.*, n different heads) requires more than $2n \ln n$ coupons. The probability that any specific coupon type has not been found among $2n \ln n$ coupons is

$$(1 - \frac{1}{n})^{2n \ln n} \leq e^{-2 \ln n} = \frac{1}{n^2}.$$

  By union bound, probability that any type is not found is at most $1/n$.

Thereotical Analysis: Randomized Algorithm

Proof:

ε_1: The algorithm ran for $3n \ln n$ steps with no unused-edges list becoming empty, but it failed to construct a Hamiltonian cycle.
ε_2: At least one unused-edges list became empty during the first $3n \ln n$ iterations of the loop.

- In order to complete a Hamiltonian path to a cycle the path must close, which it does at each step with probability $1/n$. Hence the probability that the path does not become a cycle within the next $n \ln n$ iterations is
$$(1 - \frac{1}{n})^{n \ln n} \leq e^{-\ln n} = \frac{1}{n}.$$
Thus, $\Pr(\varepsilon_1) \leq \frac{2}{n}$.

Thereotical Analysis: Randomized Algorithm

Proof:

ε_2: At least one unused-edges list became empty during the first $3n \ln n$ iterations of the loop.

ε_{2a}: At least $9 \ln n$ edges were removed from the unused-edges list of at least one vertex in the first $3n \ln n$ iterations of the loop.

ε_{2b}: At least one vertex had fewer than $10 \ln n$ edges initially in its unused-edges list.

For ε_2 to occur, either ε_{2a} or ε_{2b} must occur. Hence $\varepsilon_2 \leq \varepsilon_{2a} + \varepsilon_{2b}$.

- The number of times X that v is the head during the first $3n \ln n$ steps is a binomial random variable $B(3n \ln n, 1/n)$, and this dominates the number of edges taken from v's unused-edges list. Using the Chernoff bound
$$\Pr(X \geq (1+\delta)\mu) \leq (\frac{e^\delta}{(1+\delta)^{1+\delta}})^\mu$$
with $\delta = 2$ and $\mu = 3 \ln n$, we have $\Pr(X \geq 9 \ln n) \leq (\frac{e^2}{27})^{3 \ln n} \leq \frac{1}{n^2}$. By taking a union bound over all vertices, we have $\Pr(\varepsilon_{2a}) \leq 1/n$.

Thereotical Analysis: Randomized Algorithm

Proof:

ε_{2a}: At least $9 \ln n$ edges were removed from the unused-edges list of at least one vertex in the first $3n \ln n$ iterations of the loop.

ε_{2b}: At least one vertex had fewer than $10 \ln n$ edges initially in its unused-edges list.

- The expected number of edges Y initially in a vertex's unused-edges list is at least $(n-1)q \geq 20(n-1)\ln n/n \geq 19 \ln n$ for sufficiently large n. Using Chernoff bound with $\delta = 9/19$ and $\mu = 19 \ln n$,

$$\Pr(X \leq (1-\delta)\mu) \leq e^{-\mu\delta^2/2}.$$

  the probability that any vertex initially has fewer than $10 \ln n$ edgs on its list is at most

$$\Pr(Y \leq 10 \ln n) \leq e^{-19 \ln n (9/19)^2/2} \leq \frac{1}{n^2}$$

  By union bound, the probability that any vertex has fewer than $10 \ln n$ edgs is at most $1/n$. Hence, $\Pr(\varepsilon_{2b}) \leq \frac{1}{n}$.

Thereotical Analysis: Randomized Algorithm

Theorem

Suppose the input to the modified Hamiltonian cycle algorithm initially has unused-edge lists where each edge (v, u) with $u \neq v$ is placed on v's list independently with probability $q \geq 20 \ln n/n$. Then the algorithm successfully finds a Hamiltonian cycle in $O(n \ln n)$ iterations of the repeat loop with probability $1 - O(n^{-1})$.

Proof:

Consider the following two events:
ε_1: The algorithm ran for $3n \ln n$ steps with no unused-edges list becoming empty, but it failed to construct a Hamiltonian cycle.
ε_2: At least one unused-edges list became empty during the first $3n \ln n$ iterations of the loop.
We have $\Pr(\varepsilon_1) \leq \frac{2}{n}$ and $\Pr(\varepsilon_2) \leq \frac{2}{n}$.
Thus, $\Pr(\text{algorithm fails}) \leq \Pr(\varepsilon_1) + \Pr(\varepsilon_2) \leq \frac{4}{n}$.

Thereotical Analysis: Randomized Algorithm

We are left with showing how our algorithm can be applied to random graphs in $G_{n,p}$. We show that, as long as p is known, we can partition the edges of the graph into edge lists that satisfy the requirements of Theorem.

Thereotical Analysis: Randomized Algorithm

We are left with showing how our algorithm can be applied to random graphs in $G_{n,p}$. We show that, as long as p is known, we can partition the edges of the graph into edge lists that satisfy the requirements of Theorem.

Lemma

By initializing edges on the unused-edges lists appropriately, randomized algorithm will find a Hamiltonian cycle on a graph chosen randomly from $G_{n,p}$ with probability $1 - O(1/n)$ whenever $p \geq 40 \ln n/n$.

Thereotical Analysis: Randomized Algorithm

Proof:

We partition the edges of our input graph from $G_{n,p}$ as follows. Let $q \in [0, 1]$ be such that $p = 2q - q^2$. Consider any edge (u, v) in the input graph. We execute exactly one of the following three possibilities:

Thereotical Analysis: Randomized Algorithm

Proof:

We partition the edges of our input graph from $G_{n,p}$ as follows. Let $q \in [0,1]$ be such that $p = 2q - q^2$. Consider any edge (u, v) in the input graph. We execute exactly one of the following three possibilities:

- With probability $q(1-q)/(2q-q^2)$ we place the edge on u's unused-edges list but not on v's;

Thereotical Analysis: Randomized Algorithm

Proof:

We partition the edges of our input graph from $G_{n,p}$ as follows. Let $q \in [0, 1]$ be such that $p = 2q - q^2$. Consider any edge (u, v) in the input graph. We execute exactly one of the following three possibilities:

- With probability $q(1-q)/(2q-q^2)$ we place the edge on u's unused-edges list but not on v's;
- With probability $q(1-q)/(2q-q^2)$ we initially place the edge on v's unused-edges list but not on u's;

Thereotical Analysis: Randomized Algorithm

Proof:

We partition the edges of our input graph from $G_{n,p}$ as follows. Let $q \in [0, 1]$ be such that $p = 2q - q^2$. Consider any edge (u, v) in the input graph. We execute exactly one of the following three possibilities:

- With probability $q(1 - q)/(2q - q^2)$ we place the edge on u's unused-edges list but not on v's;
- With probability $q(1 - q)/(2q - q^2)$ we initially place the edge on v's unused-edges list but not on u's;
- With the remaining probability $q^2/(2q - q^2)$ the edge is placed on both unused-edges lists.

Thereotical Analysis: Randomized Algorithm

Proof:

We partition the edges of our input graph from $G_{n,p}$ as follows. Let $q \in [0,1]$ be such that $p = 2q - q^2$. Consider any edge (u, v) in the input graph. We execute exactly one of the following three possibilities:

- With probability $q(1-q)/(2q-q^2)$ we place the edge on u's unused-edges list but not on v's;
- With probability $q(1-q)/(2q-q^2)$ we initially place the edge on v's unused-edges list but not on u's;
- With the remaining probability $q^2/(2q-q^2)$ the edge is placed on both unused-edges lists.

For any possible edge (u, v), the probability that it is initially placed in the unused-edges list for v is

$$p \cdot \left(\frac{q(1-q)}{2q-q^2} + \frac{q^2}{2q-q^2}\right) = q.$$

Thereotical Analysis: Randomized Algorithm

Proof:

We partition the edges of our input graph from $G_{n,p}$ as follows. Let $q \in [0,1]$ be such that $p = 2q - q^2$. Consider any edge (u,v) in the input graph. We execute exactly one of the following three possibilities:

- With probability $q(1-q)/(2q-q^2)$ we place the edge on u's unused-edges list but not on v's;
- With probability $q(1-q)/(2q-q^2)$ we initially place the edge on v's unused-edges list but not on u's;
- With the remaining probability $q^2/(2q-q^2)$ the edge is placed on both unused-edges lists.

For any possible edge (u,v), the probability that it is initially placed in the unused-edges list for v is

$$p \cdot \left(\frac{q(1-q)}{2q-q^2} + \frac{q^2}{2q-q^2}\right) = q.$$

Moreover, the probability that an edge (u,v) is initially placed on the unused-edges list for both u and v is $p \cdot q^2/(2q-q^2) = q^2$.

Thereotical Analysis: Randomized Algorithm

Lemma

By initializing edges on the unused-edges lists appropriately, randomized algorithm will find a Hamiltonian cycle on a graph chosen randomly from $G_{n,p}$ with probability $1 - O(1/n)$ whenever $p \geq 40 \ln n/n$.

Thereotical Analysis: Randomized Algorithm

Lemma

By initializing edges on the unused-edges lists appropriately, randomized algorithm will find a Hamiltonian cycle on a graph chosen randomly from $G_{n,p}$ with probability $1 - O(1/n)$ whenever $p \geq 40 \ln n/n$.

Proof:

So the placements are indepdent events. Since each edge (u, v) is treated independently, this partitioning fulfills the requirements of Theorem provided the resulting q is at least $20 \ln n/n$. When $p \geq 40 \ln n/n$, we have $q \geq p/2 \geq 20 \ln n$, and the result follows.

Thereotical Analysis: Randomized Algorithm

Lemma

By initializing edges on the unused-edges lists appropriately, randomized algorithm will find a Hamiltonian cycle on a graph chosen randomly from $G_{n,p}$ with probability $1 - O(1/n)$ whenever $p \geq 40 \ln n/n$.

Theorem

Suppose the input to the modified Hamiltonian cycle algorithm initially has unused-edge lists where each edge (v, u) with $u \neq v$ is placed on v's list independently with probability $q \geq 20 \ln n/n$. Then the algorithm successfully finds a Hamiltonian cycle in $O(n \ln n)$ iterations of the repeat loop with probability $1 - O(n^{-1})$.

Q & A

Q & A

Thank you.