

“Balls Into Bins” Model — Occupancy Problems

Assume that m balls are thrown randomly into n bins, i.e., the location of each ball is independently and uniformly chosen at random from the n possibilities.

- How many randomly thrown balls are needed to fill all the bins? - **Coupon Collector** Problem.
- What is the maximum number of balls in any bin?
- How many bins are empty?
- What is the distribution of the balls in the bins?

What is the maximum number of balls in any box?

Let $E_{i, \geq k}$ be the event box i received $\geq k$ balls.

$$Pr(E_{i, \geq k}) \leq \binom{m}{k} \left(\frac{1}{n}\right)^k$$

The probability that any box received at least k balls is bounded by

$$Pr(\cup_{i=1}^n E_{i, \geq k}) \leq \sum_{i=1}^n Pr(E_{i, \geq k})$$

For $m = n$, and $k = \frac{e \log n}{\log \log n}$ we get

$$n \binom{n}{k} \left(\frac{1}{n}\right)^k \leq n \left(\frac{e}{k}\right)^k \leq$$

$$n e^{-2 \log n} = o(1).$$

For $m = 2n \ln n$, and $k = 4e \log n$, we get

$$n \binom{2n \log n}{k} \left(\frac{1}{n}\right)^k \leq n \left(\frac{2en \log n}{k}\right)^k \left(\frac{1}{n}\right)^k \leq$$

$$n e^{-2 \log n} = o(1).$$

Application to Geometric Random Graphs

Assume that n points are thrown randomly in a unit square.

The random geometric graph $G(n, r)$ has n nodes corresponding to the n points and two nodes are connected by an edge if they are within a distance of r of each other. We assume the L_∞ norm, i.e., for two points $u = (x_1, y_1)$, $v = (x_2, y_2)$, $d(u, v) = \max(|x_1 - x_2|, |y_1 - y_2|)$.

A popular model for Sensor Networks.

Connectivity of $G(n, r)$

Theorem 1. *If $r \geq \sqrt{\frac{c \log n}{n}}$, where c is a constant > 4 , then $G(n, r)$ is connected asymptotically almost surely, i.e., $\Pr(G(n, r) \text{ is connected}) \rightarrow 1$ as $n \rightarrow \infty$.*

Proof. Divide the unit square into bins of size $r/2 \times r/2$.

Number of bins is $4/r^2$.

$G(n, r)$ is connected if no bin is empty.

Let $r = \sqrt{\frac{c \log n}{n}}$.

The probability that a bin is empty is

$$(1 - r^2/4)^n \leq e^{-nr^2/4} = n^{-c/4}$$

Let X be the number of empty bins.

$E[X] = \frac{4}{r^2} n^{-c/4} = \frac{4n}{c \log n} n^{-c/4} \leq n^{1-c/4} \rightarrow 0$ if $c > 4$.

$\Pr(X > 0) \leq E[X] \rightarrow 0. \quad \square$

The Coupon Collector Problem

Assume that there are n distinct coupons. Every time you go to a shop you collect a random coupon i.e., every coupon is equally likely to show up. How many coupons you have to collect till you have all the n distinct coupons?

- Theorem 2.**
- 1. The expected number of coupons needed to collect all the n distinct coupons is $n \ln n + O(n)$.*
 - 2. With probability at least $1 - 1/n$, the number of coupons needed is at most $2n \ln n$.*
 - 3. Further, with probability at least $1 - 1/n$ no coupon is collected more than $9 \ln n$ number of times.*

Proof.

1. Let X = number of coupons needed.

Let X_i = number of coupons from the time $i - 1$ coupons are collected till a new coupon is got.

$$X = \sum_{i=1}^n X_i.$$

The probability of getting the i th new coupon, given that $i - 1$ coupons are already collected is

$$p_i = 1 - \frac{i - 1}{n}$$

$$E[X_i] = \frac{1}{p_i} = \frac{n}{n - i + 1}$$

$$E[X] = \sum_{i=1}^n \frac{n}{n - i + 1} = n \sum_{i=1}^n \frac{1}{i} = nH_n = n \ln n + O(n)$$

2. The probability that coupon i is not collected after $3n \ln n$ number of trials is:

$$\Pr(E_i) = \left(1 - \frac{1}{n}\right)^{3n \ln n} \leq e^{-3 \ln n} \leq 1/n^3$$

The probability that no coupon is collected is bounded by

$$\Pr(\cup_{i=1}^n E_i) \leq \sum_{i=1}^n \Pr(E_i) \leq 1/n^2 \leq 1/n$$

3. The expected number of times a particular coupon is collected in $3n \ln n$ trials is $3 \ln n$.

Using the Chernoff bound, with probability at least $1 - 1/n^3$, this coupon was not collected more than $9 \ln n$ times. Summing up the probabilities for n coupons we have the result.

□

Random Graphs

A probabilistic model of graphs for probabilistic analysis of graph algorithms.

$G(n, p)$ random graph model: Given a n -vertex labeled graph, an (undirected) edge occurs between a pair of vertices with probability p , independently of other pairs.

The probability of a graph G of k edges is given by

$$p^k(1 - p)^{\binom{n}{2} - k}$$

The probability that a graph $G \in G(n, p)$ contains k edges is binomially distributed:

$$\binom{\binom{n}{2}}{k} p^k (1 - p)^{\binom{n}{2} - k}$$

Algorithm for finding a Hamiltonian Path

1. Choose an arbitrary vertex x_0 to start the path.
 $HEAD = x_0$.
2. Repeat until all vertices are connected:
 - (a) Choose a random vertex (say u) from HEAD's adjacency list.
 - (b) remove $(HEAD, u)$ from HEAD's and u 's lists.
 - (c) **If** u is not in the path then $HEAD = u$;
else use the edge to ROTATE path and set HEAD to the successor of u in the path.

Algorithm for finding a Hamiltonian Path

1. Choose an arbitrary vertex x_0 to start the path.
 $HEAD = x_0$.
2. Repeat until all vertices are connected:
 - (a) Choose a random vertex (say u) from HEAD's adjacency list.
 - (b) remove $(HEAD, u)$ from HEAD's and u 's lists.
 - (c) **If** u is not in the path then $HEAD = u$;
else use the edge to ROTATE path and set HEAD to the successor of u in the path.

Rotation

Let

$$P = v_1, v_2, \dots, v_k$$

be a simple path in an undirected graph G . Let (v_k, v_i) be an edge of G . Then

$$P' = v_1, \dots, v_i, v_k, v_{k-1}, \dots, v_{i+2}, v_{i+1}$$

is a rotated simple path in G .

Theorem 3. *If no adjacency list is empty before termination, then whp:*

- 1. The algorithm finds a Hamiltonian path;*
- 2. The algorithm terminates after $O(n \log n)$ iterations of the while loop;*
- 3. No more than $a \log n$ edges (for some constant $a > 0$) are removed from any list.*

Proof. Consider a "less efficient" algorithm that for each vertex u keeps two lists:

1. $new_edges(u)$ - adjacent edges that were not used yet;
2. $old_edges(u)$ - edges that were already used.

When u is at the head of the path we choose:

1. A random element in $old_edges(u)$ with probability $\frac{|old_edges(u)|}{n}$.
2. u with probability $1/n$.
3. with probability $1 - \frac{1}{n} - \frac{|old_edges(u)|}{n}$, a random element in $new_edges(u)$ (and move it to $old_edges(u)$).

Make the successor (in the path) of the chosen node to be the new head by rotation. (In case u is chosen, then the successor is the last vertex in the path).

The number of iterations of the modified algorithm **stochastically dominates** the number of iterations of the original algorithm i.e.,

$$\forall x \quad \Pr(I_{mod} > x) \geq \Pr(I_{orig} > x)$$

The number of edges removed from a $new_edges(u)$ by the modified algorithm has the same distribution as in the original algorithm.

The probability that a given vertex becomes HEAD at a given iteration of the modified algorithm is $1/n$.

By the coupon collector analysis, if no list is empty before termination, then whp:

1. The algorithm terminates with an Hamiltonian path in $3n \ln n$ iterations.
2. No vertex is chosen more than $9 \ln n$ times.

□

Corollary 1. *If $p \geq \frac{36 \log n}{n}$ and $G \in G(n, p)$, then whp G has a Hamiltonian path and the algorithm will find it.*

Proof. What should be p so that whp every vertex has more than $18 \ln n$ neighbors?

The expected number of neighbors for a vertex is $(n - 1)p$; using the Chernoff bound and the union bound we can show that if $p \geq \frac{36 \ln n}{n}$, then every vertex has more than $18 \ln n$ neighbors. □

Stable Marriage

Consider a society of n men and n women. A marriage is a 1-1 function between men to women.

Each person (man or woman) has an ordered list of preferences.

A marriage is unstable if there are two couples x_1, y_1 and x_2, y_2 such that x_1 prefers y_2 to y_1 and y_2 prefers x_1 to x_2 . Else the marriage is **stable**.

Stable Marriage Algorithm:

1. Start with all men and women unmarried.
2. **Repeat** till all men are married.
 - (a) An unmarried man proposes to the first woman on his list that hasn't rejected him yet.
 - (b) The proposed woman accepts the proposal if she is currently unmarried, or if she is married but prefers the current proposer on her current mate (in that case the old mate becomes unmarried).

Theorem 4. *The algorithm always terminates in $O(n^2)$ steps with a stable marriage.*

Proof. Once a woman is married she remains married, when she changes mates her preference of the mates only improves.

If a man is not married there is an available woman in his list, since all the women he proposed to are married and there are n women. Thus, $O(n^2)$ time.

Assume that output marriage is not stable, i.e. x_1 married to y_1 and prefers y_2 and y_2 prefers x_1 on x_2 . We prove a contradiction.

If x_1 prefers y_2 he must have proposed y_2 before y_1 . Since y_2 rejected x_1 at some point in the algorithm it must be married to a more desirable mate.

□

Theorem 5. *Assuming that the lists of preferences are chosen uniformly and independently at random, the expected number of proposals made by the algorithm is $O(n \log n)$, the maximum number of proposal any woman receives is bounded by $O(\log n)$.*

Proof.

Instead of having a pre-selected preference lists we can assume that a man chooses a random woman that did not rejected him till now.

We simplify the analysis by assuming that each time a man proposed he proposed to a random woman chosen uniformly from among **all** women.

Since all the "added" proposals are rejected the number of proposals in the new algorithm **stochastically dominates** the number of proposal in the original algorithm.

$$\forall x \ Pr(T_{new} > x) \geq Pr(T > x).$$

The algorithm terminates once all women receive at least one proposal. \square