Ôn tập Mô hình và thuật toán Internet phổ biến

1. **Combinatorial analysis (Phân tích tổ hợp)**
    1.1. **Tổ hợp**
    1.2. **Chỉnh hợp**
    1.3. **Hoán vị**
    1.4. **Bài toán xếp Anten**

Bài toán: Giả sử có n cái anten trong đó có m cái bị hỏng. Các chiếc anten bình thường và bị hỏng giống nhau và không biệt được. Hỏi có bao nhiêu cách xếp n anten thành một hàng ngang sao cho không có hai cái hỏng nào xếp cạnh nhau.

Lời giải:

Giả sử m � $\frac{2n-1}{2}$

Ta xếp $n-m$ cái ăng ten bình thường vào một hàng ngang. Khi đó, để xếp các ăng ten hỏng vào hàng sao cho hai cái hỏng không cạnh nhau thì giữa hai cái ăng ten thường có nhiều nhất 1 chiếc ăng ten hỏng.

Do giữa $n-m$ ăng ten thường ở trên thì có $n-m-1$ vị trí ở giữa hai ăng ten bình thường và hai vị trí hai đầu nên có tổng cộng $n-m+1$ vị trí để xếp m chiếc ăng ten hỏng.

Do vậy, bài toán trở thành đếm số cách sắp xếp m chiếc ăng ten hỏng vào $n-m+1$ vị trí.

Theo bài toán tổ hợp thì ta có kết quả là: C_{n-m+1}^{m}.

2. **Axioms of probability (Tiên đoán xác suất)**
    2.1. **Matching problem**

Bài toán: Có n bức thư và n chiếc phong bì. Các lá thư được xếp ngẫu nhiên vào các phong bì khác nhau. Hỏi xác suất để có ít nhất 1 lá thư được xếp vào đúng chiếc phong bì có địa chỉ cần gửi?

Lời giải:

Không gian mẫu là hoán vị các cách xếp n lá thư vào n chiếc phong bì. Do đó không gian mẫu có $n!$ phần tử.

Gọi E_i là sự kiện lá thư thứ *i* được xếp vào đúng phong bì. Do đó, xác xuất cần tính chính là xác suất

$P\left(E_1 \cup E_2 \cup \ldots \cup E_n\right) = \sum_{k=1}^{n}(-1)^{k+1} \sum_{1 \le i_1 < \ldots < i_k \le n} P(E_{i_1}.E_{i_2}...E_{i_k})$ (theo nguyên lý thêm bớt)

Do $P(E_{i_1}.E_{i_2}...E_{i_k})$ là xác suất để có k bức thư đúng địa chỉ và $n-k$ bức thư không đúng địa chỉ nên ta dễ dàng tính được

$$P(E_{i_1}.E_{i_2}...E_{i_k}) = \frac{(n-k)!}{n!} = \frac{1}{k!C_n^k}$$

Suy ra: $P\left(E_1 \cup E_2 \cup \ldots \cup E_n\right) = \sum_{k=1}^{n}(-1)^{k+1}.C_n^k P(E_{i_1}.E_{i_2}...E_{i_k}) = \sum_{k=1}^{n}\frac{(-1)^{k+1}}{k!}$ (do Các giá trị $P(E_{i_1}.E_{i_2}...E_{i_k})$ không thay đổi với mỗi k và có C_n^k cách chọn k giá trị trong n giá trị).

2.2. Hiring problem

Thuật toán:

```
HIRE-ASSISTANT(n)
best←0
candidate 0 is a least-qualified dummy candidate
for i←1 to n
   do interview candidate i
      if candidate i is better than candidate best
         then best←i
              hire candidate i
```

Phân tích chi phí:

Chi phí phỏng vấn người thứ *i* (thấp) là c_i. Chi phí thuê người thứ *i* là c_h.

Giả sử cần thuê m người để tìm ra người tốt nhất. Như vậy, chi phí sẽ là $O\left(n.c_i + m.c_h\right)$. Trong đó, chi phí để phỏng vấn *n* ứng viên $n.c_i$ là cố định. Tất cả chi phí sẽ phụ thuộc vào số người được thuê.

Như vậy, trường hợp xấu nhất là phải thuê tất cả n người và chi phí lúc đó sẽ là $O\left(n.c_i + n.c_h\right)$.

2.3. Randomize Algorithm

2.4. Kiểm tra đa thức bằng nhau

Bài toán: Kiểm tra xem hai đa thức *F(x)* và *G(x)* có bằng nhau hay không?

Thuật toán:

Giả sử *F* và *G* lần lượt có bậc là d_F và d_G. Khi đó $d = \max\left(d_F, d_G\right)$.

Chọn một số ngẫu nhiên a từ tập các số từ 1 đến 100*d*.

Sau đó tính *F(a)* và *G(a).* Nếu kết quả $F\left(a\right) \neq G\left(a\right)$ ta kết luận $F \neq G$. Nếu không ta kết luận *F=G*

Phân tích thuật toán:

Thuật toán trên vẫn có thể sai trong trường hợp số a được chọn thỏa mãn phương trình $F\left(x\right) - G\left(x\right) = 0$.

Đa thức $F\left(x\right) - G\left(x\right)$ có bậc tối đa là d. Do đó, sô nghiệm tối đa của phương trình $F\left(x\right) - G\left(x\right) = 0$ là d.

Do đó, khi ta chọn một số ngẫu nhiên trong khoản giá trị từ 1 đến 100d thì có tối đa d cách để chúng ta chọn được các số là nghiệm của phương trình $F\left(x\right) - G\left(x\right) = 0$. Do vậy xác suất để đưa ra kết quả sai là dưới $\frac{d}{100d} = \frac{1}{100}$. Tức là phép thử trên cho ta xác suất chính xác hơn 99%.

2.5. Phân tích thuật toán thuê nhân viên

Tính kỳ vọng số người cần thuê:

Gọi *X* là biến ngẫu nhiên có giá trị bằng số người cần thuê.

Gọi X_i là sự kiện ứng viên thứ *i* được thuê.

Do đó, $X = X_1 + X_2 + \cdots + X_n$

Để người thứ *i* được thuê, thì trình độ của người thứ *i* phải cao hơn trình độ của người từ thứ nhất cho đến người thứ $i-1$. Ta có thể coi trình độ của *i* người là một dãy số ngẫu nhiên được xếp theo thứ tự tương ứng. Do đó, để người i có trình độ cao nhất thì số cuối cùng trong dãy trên phải là số lớn nhất trong dãy. Có *i* số, trong đó có một số là có giá trị lớn nhất. Giả sử số có giá trị lớn nhất trong dãy nằm ở vị trí thứ *j* (*j* < i), số vị trí *j* có thể có là *i*. Như vậy, xác suất để số lớn nhất nằm ở cuối dãy (tức là $j = i$) là $\frac{1}{i}$

Từ các điều trên, suy ra:

$$E(X) = E(X_1) + E(X_2) + \cdots + E(X_n)$$

$$= \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} \approx 1 + \ln n$$

3. Conditional probability and inference (Xác suất có điều kiện và suy luận)

3.1. Verification of Matrix Multiplication

Thuật toán:

Chọn một vector $r = (r_1, r_2, \ldots, r_n)$ trong đó các r_i là 0 hoặc 1.

Tính ma trận $A(Br)$ và Cr sau đó ta so sánh kết quả 2 ma trận tính được.

Nếu chúng bằng nhau thì kết luận $AB = C$, nếu không thì kết luận $AB \neq C$.

Phân tích thuật toán:

- Độ phức tạp: Độ phức tạp của thuật toán giảm từ $O\left(n^3\right)$ xuống còn $O\left(n^2\right)$
- Tính chính xác:

Tuy độ phức tạp của thuật toán giảm, nhưng thuật toán không chính xác hoàn toàn. Thuật toán sai khi chúng ta vô tình chọn vector *r* sao cho $AB \neq C$ nhưng $ABr = Cr$.

Ta sẽ tính xác suất: $\Pr(ABr = Cr)$

Ma trận $D = AB - C$

Khi đó, $\Pr(ABr = Cr) = \Pr\left(Dr = 0\right)$

Điều kiện $AB \neq C \Leftrightarrow D \neq 0$

Giả sử $d_{xy} \neq 0, \left(0 \le x, y \le n\right)$

Do $Dr = 0 \Leftrightarrow \sum_{j=1}^{n} d_{xj}.r_j = 0 \Rightarrow r_y = -\frac{\sum_{j \neq y} d_{xj}.r_j}{d_{xy}}$

Suy ra

$$\Pr\left(Dr = 0\right) \le \Pr\left(r_y = -\frac{\sum_{j \neq y} d_{xj}.r_j}{d_{xy}}\right)$$

$$\Rightarrow \Pr\left(ABr = Cr\right) \le \Pr\left(r_y = -\frac{\sum_{j \neq y} d_{xj}.r_j}{d_{xy}}\right)$$

$$\Rightarrow \Pr\left(ABr = Cr\right) \le \sum_{S} \Pr\left(\left(r_y = -\frac{\sum_{j \neq y} d_{xj}.r_j}{d_{xy}}\right) \cap S\right)$$

$$\Rightarrow \Pr\left(ABr = Cr\right) \le \sum_{S} \Pr\left(r_y = -\frac{\sum_{j \neq y} d_{xj}.r_j}{d_{xy}} \,\middle|\, S\right) \Pr\left(S\right)$$

Trong đó, S là một cách chọn vector con trong vector *r* sao cho không có phần tử r_y. Như vậy có tổng cộng 2^{n-1} cách chọn S.

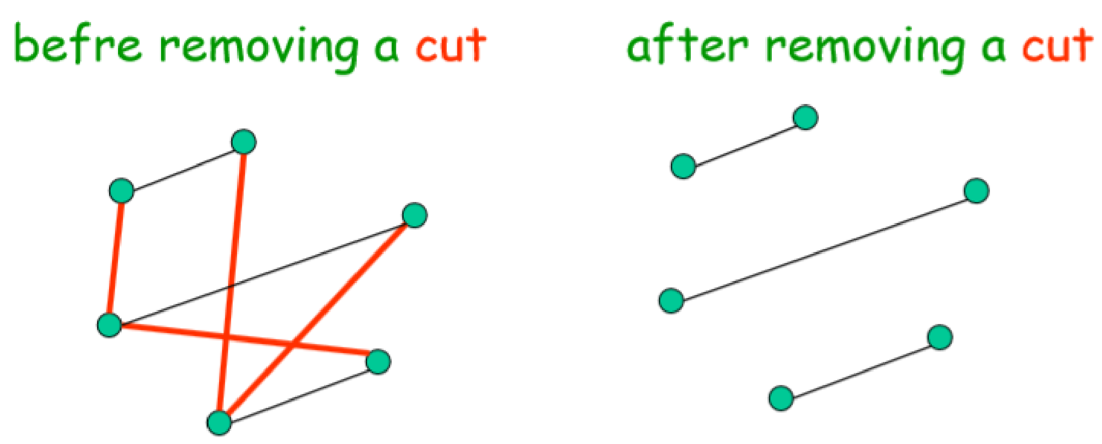
Suy ra $\Pr(ABr = Cr) \le \sum_S \frac{2^{n-1}}{2^n}\Pr(S) = \sum_S \frac{1}{2}\Pr(S) = \frac{1}{2}$

Như vậy, khi ta thực hiện lặp đi lặp lại phép thử trên k lần, xác suất sai số sẽ nhỏ hơn $\left(\frac{1}{2}\right)^k$. Nếu $k = 10$ thì xác suất sai số sẽ nhỏ hơn 0,1%.

3.2. Randomized Min-Cut

Bài toán: Cho *G* là một đồ thị vô hướng.

Một lát cắt của *G* là một tập hợp các cạnh mà sau khi bỏ chúng ra khỏi đồ thị thì đồ thị *G* trở thành một đồ thị có nhiều hơn 1 thành phần liên thông.



Bài toán tìm lát cắt nhỏ nhất (Min-cut problem) là tìm lát cắt cho đồ thị *G* sao cho có số cạnh nhỏ nhất.

Bài toán này rất hay sử dụng trong các vấn đề liên quan đến độ tin cậy trên Internet.

Thuật toán:

Gọi số đỉnh của đồ thị *G* là *n*, số cạnh là *m*.

- Đặt đồ thị *G'* là đồ thị *G*.
- Trong khi *G'* có nhiều hơn 2 đỉnh.
  - o Chọn một cạnh e trong *G'* (chọn ngẫu nhiên trong tất cả các cạnh)

- o Xóa bỏ cạnh e và kết hợp hai đỉnh của cạnh e thành 1 siêu đỉnh.
- o Các cạnh khác nối đến các đỉnh trên được nối vào siêu đỉnh vừa tạo.

- Cứ tiếp tục làm như vậy cho đến khi chỉ còn lại 2 đỉnh cuối cùng.

Đồ thị *G’* chính là lát cắt cần tìm

Phân tích:

Giả sử *C* là một Min-cut của *G* và *k* là kích thước (số phần tử) của *C*.

Xác suất để thuật toán chính xác là *Pr.*

Khi đó: $\Pr \geq$ Pr(*C* là kết quả cuối cùng)

$\geq \Pr$ Pr(tất cả các cạnh của C đều không thể liên kết tạo thành 1 siêu đỉnh mới)

Đặt E_i là sự kiện cạnh được liên kết tại bước thứ i không nằm trong C.

Như vậy Pr(tất cả các cạnh của C đều không thể liên kết) =

$$\Pr\left(\bigcap_{i=1}^{n-2} E_i\right)$$

(*n*-2 là do khi còn lại hai đỉnh thì không thực hiện liên kết 2 đỉnh này nữa)

$$= \Pr(E_1)\cdot\Pr(E_2|E_1)\cdot\Pr(E_3|E_2\cap E_1)\cdots\Pr\left(E_{n-2}\,\middle|\,\bigcap_{i=1}^{n-3} E_i\right)$$

Khi bắt đầu mỗi bước, bậc của một siêu đỉnh nhỏ nhất là k bởi nếu bậc nhỏ hơn k thì 2 đỉnh này đã tạo thành một thành phần liên thông riêng của đồ thị G.

Do đó, $\Pr(E_1) \geq 1 - \frac{k}{nk/2} = 1 - \frac{2}{n} = \frac{n-2}{n}$

$$\Pr(E_2|E_1) \geq 1 - \frac{k}{(n-1)k/2} = \frac{n-3}{n-1}$$

$$\Pr\left(E_3 \mid E_1 \cap E_2\right) \ge 1 - \frac{k}{(n-2)k/2} = \frac{n-4}{n-2}$$

M

$$\Pr\left(E_{n-2} \mid \bigcap_{i=1}^{n-3} E_i\right) \ge 1 - \frac{k}{3k/2} = \frac{1}{3}$$

Suy ra, $\Pr \ge \frac{n-2}{n} \cdot \frac{n-3}{n-1} \cdot \frac{n-4}{n-2} \cdots \frac{1}{3} = \frac{2}{n(n-1)}$

Vậy thuật toán đúng với xác suất lớn hơn $\frac{2}{n(n-1)}$, xác suất thuật toán sai nhỏ hơn $1 - \frac{2}{n(n-1)}$.

Sau đó, ta chạy thuật toán t lần. Khi đó, xác suất thuật toán sai sẽ nhỏ hơn $\left(1 - \frac{2}{n(n-1)}\right)^t \le e^{-\frac{2}{n(n-1)}}$

Đặt $t = n(n-1)\ln n$ thì thuật toán sau khi chạy t lần sẽ có độ sai số nhỏ hơn $\frac{1}{n^2}$.

3.3. Geometric Distribution

Là phân phối của xác suất xuất hiện lần đầu tiên của sự kiện *A* trong phép thử Béc-nu-li. Phân phối hình học được kí hiệu là $X : Geo(p)$, trong đó tham số *p* là xác suất xuất hiện của sự kiện *A* trong mỗi phép thử.

Xác suất: $\Pr(x) = p(1-p)^x$

Kỳ vọng: $E[X] = \frac{1}{p}$

3.4. Coupon Collector’s Problem

3.4.1. Vấn đề thu thập Coupon

Vấn đề: Giả sử rằng mỗi hộp ngũ cốc chứa một trong n phiếu giảm giá khác nhau. Một khi bạn nhận được tất cả các loại coupon, bạn có thể nhận một giải thưởng.
Bài toán: Số lượng hộp ngũ cốc trung bình cần mua để có đủ n loại coupon.
Lời giải:
Gọi X là biến ngẫu nhiên số lượng hộp ngũ cốc cần mua để có đu loại coupon.
Như vậy, $E[X]$ chính là số lượng hộp ngũ cốc trung bình cần mua để có đủ n loại coupon khác nhau.
Gọi X_i là số lượng hộp ngũ cốc cần mua để có coupon thứ *i* (coupon khác *i*-1 coupon đã có)
Dễ thấy: $X = \sum_{i=1}^{n} X_i$
Suy ra: $E[X] = \sum_{i=1}^{n} E[X_i]$
Ta có, xác suất để khi mở một hộp ngũ cốc mới, ta có một coupon mới khác với *i*-1 coupon đã có là: $p = \frac{n-i+1}{n}$ (do còn $n-i+1$ coupon chưa có).
Theo phân phối hình học, ta tính được $E[X_i] = \frac{1}{p_i} = \frac{n}{n-i+1}$
Như vậy, $E[X] = \sum_{i=1}^{n} E[X_i] = \sum_{i=1}^{n} \frac{n}{n-i+1} = n\sum_{i=1}^{n} \frac{1}{n-i+1} \approx n\ln n$

3.4.2. Packet Sampling

Ứng dụng: Tìm kiếm kẻ tấn công DOS
Cách thực hiện: Có 2 cách là Node Apend và Node Sampling

- Node Apend: Qua mỗi nút mạng, ta sẽ thêm vào đầu gói tin thông tin của nút mạng đó để nếu có tấn công DOS xảy ra thì có thể sử dụng phần thêm vào các gói tin để tìm ra kẻ tấn công.
  Tuy nhiên, cách này tồn tại nhược điểm là phần thông tin thêm vào là quá lớn so với nội dung thông điệp được gửi đi dẫn đến hiệu quả truyền tin giảm và có thể gây tắc nghẽn trong mạng.
- Node Sampling: Khi qua mỗi một nút mạng, các nút sẽ quyết định điền thông tin của mình vào gói tin với xác suất

p = 0.51. Nếu quyết định ghi thông tin lên gói tin thì sẽ ghi đè vào thông tin của các nút trước đó.
Để truy tìm ra kẻ tấn công thì người bị tấn công cần thu thập một số lượng gói tin đủ để có thể tổng hợp chúng lại thành một dãy các địa chỉ và tìm ra kẻ tấn công. Trong trường hợp tồi nhất cũng có thể không tìm ra được kẻ tấn công.

Phân tích Node Sampling:

Tìm số gói tin trung bình cần thiết để tìm ra kẻ tấn công.

3.5. Analysis of Quick-Sort

Thuật toán:

Quicksort Algorithm:

Input: A list $S = \{x_1, \ldots, x_n\}$ of n distinct elements over a totally ordered universe.

Output: The elements of S in sorted order.

1. If S has one or zero elements, return S. Otherwise continue.
2. Choose an element of S as a pivot; call it x.
3. Compare every other element of S to x in order to divide the other elements into two sublists:
   (a) S_1 has all the elements of S that are less than x;
   (b) S_2 has all those that are greater than x.
4. Use Quicksort to sort S_1 and S_2.
5. Return the list S_1, x, S_2.

Phân tích:

- Trường hợp tồi nhất $O(n^2)$
- Phụ thuộc vào việc chọn "chốt": trường hợp tốt là trường hợp đưa về hai tập S_1 và S_2 có số lượng phần tử tương đương nhau.
- Trong trường hợp chọn "chốt" tốt, độ phức tạp của thuật toán sẽ chỉ còn $O(n \log n)$

Trong trường hợp Randomized Quicksort, kỳ vọng của việc so sánh là $2n \log n$

Gọi biến ngẫu nhiên X_{ij} là 1 nếu ta so sánh phần tử thứ *i* và phần tử thứ *j* hoặc là 0 nếu ngược lại.
Gọi *X* là tổng số các phép so sánh được tạo ra bởi thuật toán.
Do chúng ta không bao giờ so sánh 2 số 2 lần nên ta có:

$$X = \sum_{i=0}^{n} \sum_{j=i+1}^{n} X_{ij}$$

Từ đó suy ra: $E[X] = \sum_{i=0}^{n} \sum_{j=i+1}^{n} E[X_{ij}]$

Ta có, nếu ta chọn chốt nằm giữa hai giá trị x_i và x_j thì hai số x_i và x_j sẽ nằm ở hai tập khác nhau trong bước tiếp theo. Do đó $X_{ij} = 0$.

Nếu ta chọn chốt là x_i hoặc x_j thì chắc chắn hai số x_i và x_j sẽ được so sánh với nhau, khi đó $X_{ij} = 1$.

Nếu ta chọn chốt nhỏ hơn x_i hoặc lớn hơn x_j, x_i và x_j sẽ lại cùng vào một tập và chúng ta lại bắt đầu xét lại giá trị cho X_{ij} như các bước ở trên.

Do đó, chúng ta có thể coi việc tìm giá trị của X_{ij} như một trò chơi ném bóng: nếu ta ném bóng vào giữa hai giá trị x_i và x_j ta sẽ có $X_{ij} = 0$, nếu ta ném trúng các giá trị là x_i hoặc x_j ta sẽ có $X_{ij} = 1$. Còn nếu không thì chúng ta sẽ tiếp tục chơi ném bóng một lần nữa. Tại mỗi bước, xác suất của $X_{ij} = 1$ trong điều kiện trò chơi không tiếp tục (các giá trị được chọn nằm từ x_i tới x_j) là $\frac{2}{j-i+1}$. Do đó, về mặt tổng thể $\Pr(X_{ij} = 1) = \frac{2}{j-i+1}$.

Như vậy, $\Pr(X_{i(i+1)} = 1) = 1$, $\Pr(X_{i(i+2)} = 1) = \frac{2}{3}$, …, $\Pr(X_{in} = 1) = \frac{2}{n-i+1}$

Vậy $E[X] = 2\sum_{i=1}^{n} \left(\frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots + \frac{1}{n-i+1} \right)$

Đặt $H_n = 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n}$

Suy ra $E[X] \le 2n(H_n - 1) \le 2n \ln n$

3.6. Birthday Paradox

Bài toán: Xác suất để trong một phòng 30 người có 2 người trùng ngày sinh nhật.

Lời giải:

Số cách chọn 30 ngày sinh nhật mà trong đó không có hai ngày trùng nhau là C_{365}^{30}

Số cách chọn 30 ngày sinh nhật mà các ngày sinh nhật có thể trùng nhau là 365^{30}

Như vậy xác suất để 30 người trong phòng có ngày sinh nhật khác nhau là

$$\frac{C_{365}^{30}}{365^{30}}=\frac{365.364.363.\ldots.(365-30+1)}{365^{30}}$$

$$=\frac{365}{365}.\frac{364}{365}.\frac{363}{365}.\ldots.\frac{336}{365}$$

$$=1.\left(1-\frac{1}{365}\right)\left(1-\frac{2}{365}\right)\ldots\left(1-\frac{29}{365}\right)$$

Theo Taylor, ta có: $1+x\approx e^{x}$ khi $x\to 0$

Suy ra $\frac{C_{365}^{30}}{365^{30}}\approx 1.e^{\left(-\frac{1}{365}\right)}.e^{\left(-\frac{2}{365}\right)}.\ldots.e^{\left(-\frac{29}{365}\right)}$

$$=e^{-\left(\frac{1+2+3+\ldots+29}{365}\right)}=e^{-\frac{29.30}{365}}\approx 9,25\%$$

Vậy xác suất để có hai người trùng ngày sinh nhật xấp xỉ 91,75%

Ứng dụng: Digital Signature Using Hash Functions

3.7. The Bins and Balls Model

Bài toán: Chúng ta có m quả bóng được ném vào n thùng. Trong đó, xác suất để bóng ném trúng một rổ là đều nhau và bằng $\frac{1}{n}$. Tính:

a. Kỳ vọng số lượng rổ rỗng
b. Kỳ vọng số rổ có số bóng đúng bằng 2
c. Kỳ vọng số rổ có số bóng lớn hơn hoặc bằng 3

Lời giải:

a. Mỗi một lần ném bóng là độc lập với nhau.
   Theo phân phối Nhị thức, xác suất để 1 rổ không có quả bóng nào được ném vào là $C_m^0\left(\frac{1}{n}\right)^0\left(1-\frac{1}{n}\right)^m=\left(1-\frac{1}{n}\right)^m\approx e^{-\frac{m}{n}}$.
   Gọi biến ngẫu nhiên X_i bằng 1 nếu rổ thứ *i* không có quả bóng nào được

   ném vào và bằng 0 nếu có ít nhất 1 quả bóng ném vào rổ thứ *i*.

Như vậy, $\Pr(X_i) = \left(1-\frac{1}{n}\right)^m$ và $E(X_i) = \left(1-\frac{1}{n}\right)^m$

Gọi X là số lượng rổ không có quả bóng nào được ném vào.

Như vậy, ta có: $X = \sum_{i=1}^{n} X_i$

Do đó: $E[X] = E\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} E[X_i] = n.\left(1-\frac{1}{n}\right)^m \approx n.e^{-\frac{m}{n}}$

b. Theo phân phối Nhị thức, xác suất để 1 rổ có đúng 2 quả bóng nào được ném vào là

$$C_m^2\left(\frac{1}{n}\right)^2\left(1-\frac{1}{n}\right)^{m-2} = C_m^2.\frac{(n-1)^{m-2}}{n^m} = \frac{m(m-1)}{2}.\frac{1}{(n-1)^2}.\left(1-\frac{1}{n}\right)^m \approx \frac{e^{-\frac{m}{n}}.\left(\frac{m}{n}\right)^2}{2}.$$

Gọi biến ngẫu nhiên Y_i bằng 1 nếu rổ thứ i có đúng 2 quả bóng được ném vào và bằng 0 nếu ngược lại.

Như vậy, ta có $\Pr(Y_i) \approx \frac{e^{-\frac{m}{n}}.\left(\frac{m}{n}\right)^2}{2}$ và $E[Y_i] \approx \frac{e^{-\frac{m}{n}}.\left(\frac{m}{n}\right)^2}{2}$

Gọi Y là số lượng rổ có đúng 2 quả bóng được ném vào.

Khi đó $Y = \sum_{i=1}^{n} Y_i$

Kỳ vọng số lượng rổ có đúng 2 quả bóng được ném vào là:

$$E[Y] = E\left[\sum_{i=1}^{n} Y_i\right] = \sum_{i=1}^{n} E[Y_i] \approx n.\frac{e^{-\frac{m}{n}}.\left(\frac{m}{n}\right)^2}{2}$$

c. Tương tự hai câu trên, Gọi biến ngẫu nhiên Z_i bằng 1 nếu rổ thứ i có nhiều hơn 3 quả bóng được ném vào và bằng 0 nếu ngược lại.

Ta có: $\Pr(Z_i) = 1-\left(\left(1-\frac{1}{n}\right)^m + C_m^1.\frac{1}{n}.\left(1-\frac{1}{n}\right)^{m-1} + C_m^2.\left(\frac{1}{n}\right)^2.\left(1-\frac{1}{n}\right)^{m-2}\right)$

$$\approx 1-\left(e^{-\frac{m}{n}} + e^{-\frac{m}{n}}.\frac{m}{n} + \frac{1}{2}.e^{-\frac{m}{n}}.\left(\frac{m}{n}\right)^2\right)$$

$$E[Z_i] \approx 1-\left(e^{-\frac{m}{n}} + e^{-\frac{m}{n}}.\frac{m}{n} + \frac{1}{2}.e^{-\frac{m}{n}}.\left(\frac{m}{n}\right)^2\right)$$

Gọi Z là số lượng rổ có nhiều hơn 3 quả bóng được ném vảo rổ. Do đó:

$$Z = \sum_{i=1}^{n} Z_i$$

Kỳ vọng cần tính là:

$$E[Z] = E\left[\sum_{i=1}^{n} Z_i\right] = \sum_{i=1}^{n} E[Z_i] = n.E[Z_i] \approx n.\left(1 - e^{-\frac{m}{n}} + e^{-\frac{m}{n}}.\frac{m}{n} + \frac{1}{2}.e^{-\frac{m}{n}}.\left(\frac{m}{n}\right)^2\right)$$