1 Bloom filters

a)

We define as F_x and F_y the Bloom filters X and Y respectively. Also, we define the indicator random variable E_i of the event that filter F_x and F_y are different at bit i. Moreover, we define as E the random variable of the event that depicts all the possible bits that could possibly those two filters differ so $E = \sum_{i=1}^{m} E_i$. Since both filters are used on similar sets with n elements we can use the notion of similarity in the following calculation of $Pr[E_i = 1]$:

$$\begin{aligned} Pr[E_i = 1] &= Pr[F_x^{(i)} = 1 \cap F_y^{(i)} = 0] + Pr[F_y^{(i)} = 1 \cap F_x^{(i)} = 0] \xrightarrow{similarity} \\ &= 2 \cdot Pr[F_x^{(i)} = 1 \cap F_y^{(i)} = 0] \Longrightarrow \\ &= 2 \cdot Pr[F_x^{(i)} = 1 | F_y^{(i)} = 0] \cdot Pr[F_y^{(i)} = 0] \end{aligned}$$

As 5.5.3 chapter on Bloom filters refers the probability of staying a specific bit still 0 is:

$$Pr[F_y^{(i)} = 0] = (1 - \frac{1}{m})^{kn} \approx e^{\frac{-kn}{m}} \tag{1}$$

About the $Pr[F_x^{(i)} = 1 | F_y^{(i)} = 0]$, we can reason about its complimentary probability $1 - Pr[F_x^{(i)} = 0 | F_y^{(i)} = 0]$. Conditioned on $F_y^{(i)} = 0$ we can conclude that elements $x \in X$ and $x \notin Y$ for each hash function $h_m(\cdot)$,$m \in k$, it is true that $h_m(x) \neq i$. So we should exclude all the common bits ($|X \cap Y|$) in the following calculation:

$$\begin{aligned} Pr[F_x^{(i)} = 1 | F_y^{(i)} = 0] &= 1 - Pr[F_x^{(i)} = 0 | F_y^{(i)} = 0] \Longrightarrow \\ &= 1 - (1 - \frac{1}{m})^{k \cdot (m - |X \cap Y|)} \Longrightarrow \\ &= 1 - e^{\frac{-k \cdot (m - |X \cap Y|}{m})} \end{aligned}$$

Finally from previous equation and equation (1) we get that the expected value of E is:

$$\begin{aligned} E[E] &= E[\sum_{i=1}^{m} E_i] \Longrightarrow \\ &= \sum_{i=1}^{m} E[E_i] \Longrightarrow \\ &= \sum_{i=1}^{m} Pr[E_i = 1] \Longrightarrow \\ &= \sum_{i=1}^{m} 2 \cdot Pr[F_x^{(i)} = 1 | F_y^{(i)} = 0] \cdot Pr[F_y^{(i)} = 0] \Longrightarrow \\ &= \sum_{i=1}^{m} 2 \cdot (1 - e^{\frac{-k \cdot (m - |X \cap Y|}{m})}) \cdot e^{\frac{-kn}{m}} = 2m \cdot (1 - e^{\frac{-k \cdot (m - |X \cap Y|}{m})}) \cdot e^{\frac{-kn}{m}} \end{aligned}$$

As we did in the previous homework in order to calculate the number of bits which those two sets are different you need linear time (O(m)). This could be used as a tool to find people with the same taste in music more easily, since we are looking for the common songs which is equal to $|X \cap Y|$ so we can calculate the $E[E]$ with $|X \cap Y|$ term as the only unknown parameter.

b)

Based on the 5.5.3 chapter and the guidelines of the exercise, the Bloom filter we initially built consists of an array of m bits, A[0] to $A[m-1] = A[2^b - 1]$, while the Bloom filter uses k independent random hash functions $h_1, ..., h_k$ with range $[0, ..., m-1] = [0, ..., 2^b - 1]$.

Since we have to use our Bloom filter, then the k independent random hash functions for the new Bloom filter are $\tilde{h_1}, ..., \tilde{h_k}$ with range $[0, ..., \tilde{m} - 1] = [0, ..., 2^{b-1} - 1]$. We define the new i^{th} hash function $\tilde{h_i}(x)$ as $\tilde{h_i}(x) = (x \mod 2^b) \mod 2^{b-1} = h_i(x) \mod 2^{b-1}, \forall i \in [m-1]$. With regards to new Bloom filter's array $\tilde{A}$ we have that it goes from $\tilde{A}[0]$ to $\tilde{A}[\tilde{m} - 1] = 2^{b-1\tilde{}} - 1$. As we notice, it sizes half of our Bloom filter array, so we can define it as $\tilde{A}[i] = A[i] or A[2^{b-1} + i], \forall i \in [m-1]$.

Last but not least, we should prove that the resulting hash functions are uniformly random and independent. In order to show uniformly randomness we should specify the sample space, which is 2^{b-1}, so the probability of i^{th} hash function $\tilde{h_i}(x)$ should be $p = \frac{1}{2^{b-1}}, \forall i \in [m-1]$. Hence if $a \in [0, 2^{b-1} - 1]$,

$$Pr(\tilde{h_i}(x) = a) = Pr(h_i(x) = a) + Pr(h_{2^{b-1}+i}(x) = a) = 2 \cdot \frac{1}{2^b} = \frac{1}{2^{b-1}}$$

For independence, since we pick initially k independent random hash functions $h_1, ..., h_k$ with range $[0, ..., m-1] = [0, ..., 2^b - 1]$ for our Bloom filter, then the new one which consists of the same functions with applied modulo 2^{b-1}, they should be respectively independent.

2 Expectations in random graphs

a)

We define a graph $G(V, E)$ generated from $G_{n,p}$ where V is the set of all vertices and E is the set of all edges. We define as C_i the indicator random variable of the event that a set i contains 4 nodes constructing a complete graph. By definition, a complete graph is a simple undirected graph in which every pair of distinct vertices is connected by a unique edge, so a complete graph of 4 nodes has 6 edges in total. Based on the fact that we generated our graph from $G_{n,p}$ the probability of having a 4-clique in the i^{th} set is $Pr(C_i = 1) = p^6$. We define as C the total possible sets of 4 nodes constructing a complete graph, so $C = \sum_{i \subseteq V} C_i$. Given the fact that all the possible ways to get 4 nodes from the total n is $\binom{n}{4}$ then the expected value of 4-cliques in G is the following:(we use linearity of expectation and the fact that $E[C]$ should be equal to 1)

$$\begin{aligned} E[C] = E[\sum_{i \subseteq V} C_i] &\Longrightarrow \\ = \sum_{i \subseteq V} E[C_i] &\Longrightarrow \\ = \binom{n}{4} \cdot p^6 &\Longrightarrow \\ \binom{n}{4} \cdot p^6 = 1 \Longrightarrow p = \sqrt[6]{\frac{1}{\binom{n}{4}}} = \Theta(\frac{1}{n^{\frac{2}{3}}}) & \end{aligned}$$

b)

We define a graph $G(V, E)$ generated from $G_{n,p}$ where V is the set of all vertices and E is the set of all edges. We define as X_i the indicator random variable of the event that a vertex i is isolated. By definition, an isolated vertex is a vertex with degree zero, so it is not connected with no other vertices. We define as X the total possible isolated vertices as $X = \sum_{i \subseteq V} X_i$. The expected value of X_i based on definition is $E[X_i] = (1-p)^{n-1}$, so the total expected value based on linearity of expectation is the following:

$$\begin{aligned} E[X] = E[\sum_{i \subseteq V} X_i] = \sum_{i \subseteq V} E[X_i] \Longrightarrow \\ = n \cdot (1-p)^{n-1} \Longrightarrow \\ n \cdot (1-p)^{n-1} = 1 \xrightarrow{(1-x) \leq e^{-x}} \\ n \cdot e^{(n-1) \cdot p} \leq 1 \Longrightarrow p \geq \frac{ln(n)}{n-1} = \Theta(\frac{ln(n)}{n}) \end{aligned}$$

c)

We define a graph $G(V, E)$ generated from $G_{n,p}$ where V is the set of all vertices and E is the set of all edges. We define as H_i the indicator random variable of the event that a Hamiltonian cycle i of n nodes take place in graph G. By definition, a Hamilton path, is a graph path between vertices of a graph that visits each vertex exactly once. Hence, the expected value of a Hamiltonian path i is $E[H_i] = p^n$. We define as H the total possible Hamiltonian paths as $H = \sum_{i \subseteq V} H_i$. Constructing a Hamiltonian path, from any vertex for the first node (n-1) choices are possible,for the second node (n-2) choices are possible and vice versa, so the total permutation is (n-1)!. However, we have double counted the reverse Hamiltonian paths as the hint expresses, so the total number of distinct Hamiltonian paths are $\frac{(n-1)!}{2}$. Therefore, the total expectation of H is given based on linearity of expectations as follows:

$$\begin{aligned} E[H] = E[\sum_{i \subseteq V} H_i] = \sum_{i \subseteq V} E[H_i] \Longrightarrow \\ = \frac{(n-1)!}{2} \cdot p^n \Longrightarrow \\ \frac{(n-1)!}{2} \cdot p^n = 1 \Longrightarrow \\ p = \sqrt[n]{\frac{2}{(n-1)!}} = \Theta(\sqrt[n]{\frac{n}{n!}}) \end{aligned}$$

d)

We define S as success event of the final creation of k connected components for the graph G. Based on that we imply that at the k^{th} epoch the addition of an edge connects two separated connected components into one and having in total k connected components. Also, we define as X the sum of all X_k random variable, which depicts the total number of edges needed at k^{th} epoch to have k connected components. Choosing a uniformly and random an edge that connects two nodes out of n of them is given by the probability $p = \frac{1}{\binom{n}{2}}$. Based on the coupons collector problem, the possible number of all those edges is calculated as follows: at

the start of k^{th} epoch epoch there are k-1 connected components so for each one node v out of n there are at least k-1 who are not connected to it, so the total number of edges to connect 2 disconnected components is $n \cdot (k-1)$. Nevertheless, we have double counted since an edge consists of two nodes so the total possible edges to be used at the k^{th} epoch are at least $\frac{n \cdot (k-1)}{2}$. So:

$$Pr(S) \geq \frac{n(k-1)}{2} \cdot \frac{1}{\binom{n}{2}} = \frac{n(k-1)}{2} \cdot \frac{2}{n(n-1)} = \frac{k-1}{n-1}$$

e)

Since X_k define the total number of edges we have to draw at the k^{th} epoch until we connect two disconnected components together, we can define $X_k \sim Geo(Pr(S)) = Geo(\frac{k-1}{n-1})$. So the expected value of X, $E[X]$ given that $X = \sum_{k=2} X_k, E[X_k] = \frac{n-1}{k-1}$ and the linearity of expectations is the following:

$$\begin{aligned} E[X] = E[\sum_{k=2}^{n} X_k] = \sum_{k=2}^{n} E[X_k] &\Longrightarrow \\ &\leq \sum_{k=2}^{n} \frac{n-1}{k-1} \Longrightarrow \\ &= (n-1) \cdot \sum_{k=2}^{n} \frac{1}{k-1} \xrightarrow{\textit{Harmonic number}} \\ &\leq nln(n) + O(n) \end{aligned}$$

3 Finding Hamiltonian cycles

a)

The algorithm in the book does not work for finding Hamiltonian cycles in directed graphs since it may not be correct to either reverse the path and make v_1 head or rotate the current path from v_k head to a previously used-edge or extend to an unused edges of head v_k since it may not exist a direct edge to them.

b)

Based on lecture, in order to generate a graph G from $G_{n,m}$ similar to $G_{n,p}$ we have to pick p such as $p = \frac{m}{\binom{n}{2}}$. Following the hint, a graph from $G_{n,p}$ can be generated by first choosing the number of edges X and then generate a graph from $G_{n,X}$. The X random variable which describes the number of edges we pick with probability p out of all possible edges $\binom{n}{2}$ follows a binomial distribution such as $X \sim Bin(\binom{n}{2}, p)$. So the algorithm based on theorem 5.17 will be the following:

We have as input a random graph model $G_{n,m}$ with $m \geq c \cdot nln(n)$. We pick $c = 40$ so $m \geq 40 \cdot nln(n)$

- Generate a graph G by $G_{n,m}$.
- Generate a random variable $X \sim Bin(\binom{n}{2}, p)$ with $p = \frac{m}{2 \cdot \binom{n}{2}}$ such that $p \geq \frac{m}{2 \cdot \binom{n}{2}} = \frac{m}{n(n-1)} = \frac{40 \cdot nln(n)}{n^2} = \frac{40 \cdot ln(n)}{n}$

- Generate a graph $\tilde{G}$ by $G_{n,X}$.
- If $X \leq m$, we have a graph $\tilde{G}$,which is a sub graph of G and we run randomized Hamiltonian algorithm using $\tilde{G}$ as input based on what we have seen in lecture 21.
- If $X > m$, we return "FAIL".

Since we pick $p \geq \frac{40 \cdot ln(n)}{n}$ we can use the main theorem from lecture describing that having such a probability p, the algorithm finds a Hamiltonian cycle in $\tilde{G}$ with probability $1 - O(\frac{1}{n})$ when $X \leq m$. The last thing to prove is that the our algorithm output a "FAIL" with probability $Pr(X > m)$ at most $O(\frac{1}{n})$. Given the fact that $X \sim Bin(\binom{n}{2}, p)$ and using the Chernoff bounds we have that

$$\begin{aligned} Pr(X > m) &= Pr(X > 2 \cdot \binom{n}{2} \cdot p) \xrightarrow{E[X]=\binom{n}{2} \cdot p} \\ &= Pr(X > 2 \cdot E[X]) \xrightarrow{\delta=1} \\ &\leq e^{-\frac{E[X] \cdot (\delta)^2}{3}} \xrightarrow{E[X]=\frac{m}{2}} \\ &\leq e^{-\frac{m}{6}} \leq e^{-\frac{n}{6}} \end{aligned}$$

At this point, let's define the event of error probability of the randomized Hamiltonian algorithm as E_1 and the event of "FAIL" in our proposed algorithm as $E_2 = X > m$. Given the fact that the error probability of the randomized Hamiltonian algorithm is $O(\frac{1}{n})$ and the error probability for the case of $X > m$ is $e^{-\frac{n}{6}}$ we can use the union bound for sufficient large n to calculate the total probability of our proposed algorithm as follows:

$$\begin{aligned} Pr(E_1) \cup Pr(E_2) &\leq Pr(E_1) + Pr(E_2) \Longrightarrow \\ &\leq O(\frac{1}{n}) + e^{-\frac{n}{6}} \approx O(\frac{1}{n}) \end{aligned}$$

c)

Having a random graph model $G_{n,p}$ where p is sufficient large we can alter the algorithm as follows:

- Generate a graph G by $G_{n,p}$. We define the number of edges as m.
- If $m \geq \frac{40 \cdot ln(n)}{n}$ then apply subquestion's (b) algorithm
- If $m < \frac{40 \cdot ln(n)}{n}$ then return FAIL.

As we show in sub question (b) the algorithm finds a Hamiltonian cycle in $\tilde{G}$ with probability $1 - O(\frac{1}{n})$. We need to point out that the bound of error probability for $m < \frac{40 \cdot ln(n)}{n}$ is also $O(\frac{1}{n})$. Hence, we take into consideration that edges m follow a binomial distribution so:

$$\begin{aligned} E[m] &= \binom{n}{2} \cdot p \xrightarrow{Pick\ p \geq \frac{100 \cdot ln(n)}{n}} \\ &\geq \binom{n}{2} \cdot \frac{100 \cdot ln(n)}{n} \Longrightarrow \\ &\geq \frac{n(n-1)}{2} \cdot \frac{100 \cdot ln(n)}{n} \Longrightarrow \\ &\geq 50 \cdot n \cdot ln(n) \end{aligned}$$

Therefore, using lower tail Chernoff bounds ($\delta \in (0,1)$) for the error probability $Pr(m < \frac{40 \cdot ln(n)}{n})$:

$$\begin{aligned} Pr(m < \frac{40 \cdot ln(n)}{n}) = Pr(m < (1 - \frac{1}{5})50 \cdot n \cdot ln(n)) = Pr(m < (1 - \frac{1}{5})E[m]) \Longrightarrow \\ \leq e^{-(\frac{1}{5})^2 \cdot 25 \cdot n \cdot ln(n)} \Longrightarrow \\ = e^{-(n \cdot ln(n))} \Longrightarrow \\ = O(\frac{1}{n}) \end{aligned}$$

Given that we can calculate the total error probability of the algorithm using the union bound, we define as $Pr(\textit{Error algo B})$ the error probability of subquestion (b) algorithm and we have:

$$\begin{aligned} Pr(m < \frac{40 \cdot ln(n)}{n}) \cup Pr(\textit{Error algo B}) \leq Pr(m < \frac{40 \cdot ln(n)}{n}) + Pr(\textit{Error algo B}) \Longrightarrow \\ = O(\frac{1}{n}) + O(\frac{1}{n}) = 2 \cdot O(\frac{1}{n}) \approx O(\frac{1}{n}) \end{aligned}$$

In conclusion, we proposed an algorithm that finds a Hamiltonian cycle with error probability $O(\frac{1}{n})$ given a random graph model $G_{n,p}$ with sufficient large p ($p \geq \frac{100 \cdot ln(n)}{n}$).