CS6170: Randomized Algorithms
Problem Set #3

NAME: Swathi Shree Narashiman
ROLL NO: EE22B149

MARKS: 35
DUE: March 31, 23:59

Problem 1 — 13 marks

In class we briefly mentione that w.h.p, the size of any connected component in the Cuckoo graph is $O(\log n)$. In this problem, we will work out a proof of the same using Cayley's formula that is given below.

Theorem 1 (Cayley's formula)**.** *The number of distinct trees on k vertices is k^{k-2}.*

Consider a random graph sampled from $G_{n,p}$ where $p = c/n$ for a constant $c < 1$.

(a) (2 marks) Let X_k be the number of tree components on exactly k vertices for a graph from $G_{n,p}$. A tree component on k vertices will be connected by $k-1$ edges and will be disconnected from the remaining $n-k$ vertices. Show that

$$\mathbb{E}[X_k] = \binom{n}{k} k^{k-2} p^{k-1} (1-p)^{kn - \frac{k(k+3)}{2} + 1}.$$

Solution: To find the expected number of tree components on k vertices, let us follow the following approach.

$$\textbf{\textit{The number of ways k vertices can be chosen}} = \binom{n}{k} \tag{1}$$

$$\textbf{\textit{The number of distinct trees on k vertices}} = k^{k-2} \tag{2}$$

Given k vertices, the probability that it is a tree component is computed as given:

$$\textbf{\textit{number of connected edges}} = k-1$$

$$\textit{number of disconnected edges} = \textit{number of edges with n-k vertices}$$
$$+\textit{number of disconnected edges within the component}$$

$$\textit{number of disconnected edges} = k(n-k) + \binom{k}{2} - (k-1)$$

$$\textbf{\textit{number of disconnected edges}} = nk - \frac{k(k+3)}{2} + 1$$

Therefore,

$$Pr(\textit{connected component on k vertices}) = p^{k-1}(1-p)^{nk - \frac{k(k+3)}{2} + 1} \tag{3}$$

Hence from (1), (2) and (3),

$$E[X_k] = \binom{n}{k} k^{k-2} p^{k-1} (1-p)^{nk - \frac{k(k+3)}{2} + 1}$$

(b) (3 marks) Show that for $1 \leq k \leq \sqrt{n}$

$$\mathbb{E}[X_k] \leq C \frac{n}{ck^2} e^{(1-c+\ln c)k},$$

for some constant C and large enough n.

Solution: From the results of the previous question we know that,

$$E[X_k] = \binom{n}{k} k^{k-2} p^{k-1} (1-p)^{nk-\frac{k(k+3)}{2}+1}$$

Since, $k \leq \sqrt{n}$

$$E[X_k] = \frac{n!}{k!(n-k)!} \frac{k^k}{k^2} p^{k-1} (1-p)^{nk-\frac{k(k+3)}{2}+1} \leq \frac{n!}{(n-k)!} \frac{k^k}{k!k^2} p^{k-1} (1-p)^{nk}$$

Now we will use the following inequalities:

$$\frac{k^k}{k!} \leq 1 + \frac{k}{1!} + \frac{k^2}{2!} + ... + \frac{k^k}{k!} + ... = e^k \tag{4}$$

$$\frac{n!}{(n-k)!} \leq n^k \tag{5}$$

Therefore, from (4) and (5) we get,

$$E[X_k] \leq \frac{e^k}{k^2} n^k p^{k-1} (1-p)^{nk}$$

Now, substituting $p = \frac{c}{n}$,

$$E[X_k] \leq \frac{e^k}{k^2} \frac{n}{c} (n \frac{c}{n} \left(1 - \frac{c}{n}\right)^n)^k$$

For a large n ,

$$\left(1 - \frac{c}{n}\right)^n \approx e^{-c}$$

Hence,

$$E[X_k] \leq \frac{e^k}{k^2} \frac{n}{c} c^k e^{-kc}$$

Rewriting $c^k = e^{k \ln c}$, we get:

$$E[X_k] \leq \frac{1}{k^2} \frac{n}{c} e^{k - ck + k \ln c} = \frac{n}{ck^2} e^{(1-c+\ln c)k}$$

Therefore, for some constant C we get

$$E[X_k] \leq C \frac{n}{ck^2} e^{(1-c+\ln c)k}$$

Hence proved.

(c) (2 marks) Using the same expression for $\mathbb{E}[X_k]$, show that

$$\frac{\mathbb{E}[X_{k+1}]}{\mathbb{E}[X_k]} = (n-k)\left(1+\frac{1}{k}\right)^{k-1} p(1-p)^{n-k-2},$$

and in turn that,

$$\frac{\mathbb{E}[X_{k+1}]}{\mathbb{E}[X_k]} \leq \left(1-\frac{k}{n}\right) c e^{1-c(1-k/n)} \left(1-\frac{c}{n}\right)^{-2}.$$

Solution: From the results of (a) we have,

$$E[X_k] = \binom{n}{k} k^{k-2} p^{k-1} (1-p)^{nk-\frac{k(k+3)}{2}+1}$$

This means that,

$$E[X_{k+1}] = \binom{n}{k+1} (k+1)^{k-1} p^k (1-p)^{nk+n-\frac{(k+1)(k+4)}{2}+1}$$

Therefore, dividing both we get,

$$\frac{E[X_{k+1}]}{E[X_k]} = \frac{n-k}{k+1} \frac{(k+1)^{k-1}}{k^{k-2}} p(1-p)^{n-k-2}$$

$$\frac{E[X_{k+1}]}{E[X_k]} = (n-k)\left(1+\frac{1}{k}\right)^{k-2} p(1-p)^{n-k-2}$$

Substituting $p = \frac{c}{n}$ we get,

$$\frac{E[X_{k+1}]}{E[X_k]} = (n-k)\left(1+\frac{1}{k}\right)^{k-2} \frac{c}{n} (1-\frac{c}{n})^{n-k-2}$$

We can use the following bounds, for large n

$$\left(1-\frac{c}{n}\right)^n \approx e^{-c}$$

$$\left(1-\frac{c}{n}\right)^{-k} = e^{-k ln(1-\frac{c}{n})} \approx e^{\frac{ck}{n}}$$

We use $\ln(1+x) \leq x$ for $x \geq 0$,

$$\left(1+\frac{1}{k}\right)^{k-2} = e^{k-2\ln\left(1+\frac{1}{k}\right)} \leq e^{(k-2)\frac{1}{k}} = e^{1-\frac{2}{k}} \leq e$$

Therefore,

$$\frac{E[X_{k+1}]}{E[X_k]} \leq c\left(1-\frac{k}{n}\right) e^{1-c+\frac{kc}{n}} \left(1-\frac{c}{n}\right)^{-2}$$

Hence,

$$\frac{E[X_{k+1}]}{E[X_k]} \leq c\left(1-\frac{k}{n}\right) e^{1-c(1-\frac{k}{n})} \left(1-\frac{c}{n}\right)^{-2}$$

Hence proved.

(d) (1 mark) Show that that $xe^{1-x} \leq 1$ for $x > 0$, and conclude that

$$\frac{\mathbb{E}[X_{k+1}]}{\mathbb{E}[X_k]} \leq \left(1 - \frac{c}{n}\right)^{-2}.$$

Solution: Consider,

$$f(x) = xe^{1-x} - 1$$

Taking derivative,

$$f'(x) = (x-1)e^{1-x} = 0$$

Therefore, x=1 is an extremum.

$$f''(1) = -1 < 0$$

Hence, x=1 is the maximum, this implies $\forall x$

$$f(x) \leq f(1) = 0$$

$$\implies xe^{1-x} - 1 \leq 0$$

$$\implies xe^{1-x} \leq 1$$

From the results of previous part, take $x = c\left(1 - \frac{k}{n}\right)$

$$\frac{E[X_{k+1}]}{E[X_k]} \leq \left(1 - \frac{c}{n}\right)^{-2}$$

(e) (5 marks) Using the above, argue that the maximum size of a tree component in G is $O(\log n)$ with probability $1 - o(1)$.

Solution: We know that, from part (b) we have

$$E[X_k] \leq C\frac{n}{ck^2}e^{(1-c+\ln c)k}$$

Substituting k=1 here,

$$E[X_1] \leq C\frac{n}{c}e^{1-c+\ln c}$$

From part (d) we have,

$$\frac{E[X_{k+1}]}{E[X_k]} \leq \left(1 - \frac{c}{n}\right)^{-2} = e^{\frac{2c}{n}}$$

This implies that, (by geometric series)

$$E[X_k] \leq C\frac{n}{c}e^{1-c+\ln c}e^{\frac{2c(k-1)}{n}} = C\frac{n}{c}e^{1-c+\ln c+\frac{2c(k-1)}{n}} \tag{6}$$

By Markov inequality,

$$Pr[X_k > 1] \leq E[X_k]$$

Using (6),

$$Pr[X_k > 1] \leq C\frac{n}{c}e^{1-c+\ln c+\frac{2c(k-1)}{n}}$$

$$\implies Pr[X_k > 1] \leq Cne^{1-c+\frac{2c(k-1)}{n}}$$

Here, $c = np$ is the expected number of edges, considering the example taught in class where the number of edges m>1, here c cant be <1.
Substituting, c = np in the above equation,

$$\implies Pr[X_k > 1] \leq Cne^{1-np+2p(k-1)}$$

$$\implies Pr[X_k > 1] \leq C'ne^{(2k-n)p}$$

Substituting $k = \log n$,

$$\implies Pr[X_k > 1] \leq C'ne^{(2\log n-n)p}$$

$$\implies Pr[X_k > 1] \leq C'n^{2p+1}e^{-np}$$

Since, n^{2p+1} grows polynomially and e^{-np} decays exponentially, we have ,

$$\implies Pr[X_{\log n} > 1] \leq o(1)$$

Therefore, that the maximum size of a tree component in G is $O(\log n)$ with probability $1 - o(1)$.

Problem 2 **8 marks**

You are at a ski resort and you want to go skiing. The cost of renting a ski for a day is 1 unit, and the cost of buying a ski is 3 units. If you buy a ski, then you can keep using it for all future days.

(a) (2 marks) Suppose you knew the number of days d that you are going to stay at the resort. What is the best strategy to minimize your cost for skiing.

Solution: If I knew the number of days d, then I will rent the ski if the number of days is less that 3 or buy otherwise.
Hence, my strategy is given as follows

$$Strategy(d) = \begin{cases} Rent\ the\ Ski, & d < 3 \\ Buy\ the\ Ski, & d \geq 3 \end{cases}$$

The above strategy will be the best to minimize the cost of skiing.
According to this strategy, my cost function is given as

$$cost = \begin{cases} 1, & d = 1 \\ 2, & d = 2 \\ 3, & d \geq 3 \end{cases}$$

(b) (3 marks) Suppose that you didn't know the number of days beforehand, give a deterministic online strategy that has a competitive ratio of 5/3.

Solution: If I didn't know the number of days beforehand, for each day I stay I make a choice according to the strategy given below.

Input: Let I_k denote the input for thek^{th} day.

$$I_k = \begin{cases} 1, & \textit{if staying} \\ 0, & \textit{if leaving} \end{cases}$$

For the $k+1^{th}$ day,

$$cost_k = \textit{Total amount till } k^{th} \textit{day}$$

$$Strategy(I_{k+1}) = \begin{cases} \textit{Rent the Ski}, & cost_k < 2 \\ \textit{Buy the Ski}, & otherwise \end{cases}$$

In other words, the first 2 days I will rent the ski, on the third day I will buy the ski if I am staying.

Therefore for this deterministic online strategy,

$$cost = \begin{cases} 1, & d = 1 \\ 2, & d = 2 \\ 5, & d \geq 3 \end{cases}$$

Looking at the worst case costs, for the given online deterministic algorithm and the offline algorithm we get ,

$$\textit{Competitive Ratio} = \frac{5}{3}$$

(c) (3 marks) Show that there is no deterministic online algorithm with competitive ratio better than 5/3.

Solution:

Consider any deterministic Online algorithm A as follows:

Rent the ski for k-1 days and buy it on k^{th} day if staying

Let us consider the worst case Adversary that worsens the performance of the algorithm by reducing it's advantage.

- **if k=1 then Adversary sends d=1**
  In this case, then OPT will rent it for 1 day whereas the algorithm A buys it on first day itself

$$Cost[OPT] = 1$$

$$Cost[A] = 3$$

$$\implies \textit{Competitive Ratio} = 3$$

- **if k=2 then Adversary sends d=2**
  In this case, then OPT will rent it for 2 days, whereas the algorithm A will rent it for 1 day and buy it on 2nd day.

$$Cost[OPT] = 2$$

$$Cost[A] = 4$$

$$\implies \textit{Competitive Ratio} = 2$$

- **if k≥3 then Adversary sends d=k**
  In this case, then OPT buy it on the first day itself whereas algorithm A rents the ski for k-1 days and buy's it on the k^{th} day.

$$Cost[OPT] = 3$$

$$Cost[A] = k - 1 + 3 = k + 2$$

  The best A can do is take k=3,

$$\implies Competitive\ Ratio = \frac{k+2}{3} \geq \frac{5}{3} = 1.67$$

Therefore, from all possible cases we have no deterministic online algorithm with competitive ratio better than 5/3

Collaborator: **Shri Prathaa M**

Problem 3 — 7 marks

An *edge coloring* of a graph is an assignment of colors to the edges of a graph such that no two edge that share a vertex are assigned the same color. Let us look at the online version of edge coloring where the number of vertices in the graph are fixed, and the edges arrive in an online fashion. We will assume that the degree of every vertex in the graph is at most Δ.

(a) (3 marks) Show that there exists an online algorithm that can edge color the graph with at most $2\Delta - 1$ colors.

The best that an online algorithm can be do is $\Delta + 1$, and follows from Vizing's theorem in graph theory.

Solution:

Let us define our online deterministic algorithm as follows

Algorithm

Inputs: $(u, v) \in E$
Color palette $= \{c_1, c_2, c_3, ...\}$

Let us give an index I_{V_k} corresponding to each $V_k \in V$ where $1 \leq k \leq n$.
Initialize $I_{V_k} = 0, \forall k$

For each input $e = (u, v)$
do
$I_e = max\{(I_u + 1) mod 2\Delta, (I_v + 1) mod 2\Delta\}$
$I_u \leftarrow I_e, I_v \leftarrow I_e$
$Colour(e) = c_{I_e}$
end

In this algorithm, we give the Index to any edge based on the maximum Index of either of the vertices. The colour is obtained from the corresponding index in the palette.

Analysis:
Consider $e = (v_1, v_2)$,

Given the degree to be Δ when e arrives,

$$max(I_{v_1}, I_{v_2}) \leq \Delta - 1$$

Considering the worst case scenario where v_1 occurs for the first time, let $I_{v_1} = 0, I_{v_2} = \Delta - 1$,

$$I_e = \Delta$$

Since degree is Δ, v_1 can occur $\Delta - 1$ more times, the maximum index of the colour picked will be,

$$I_{v_1} = \Delta + \Delta - 1 = 2\Delta - 1$$

Hence,

$$\textit{Number of colours required} \leq 2\Delta - 1$$

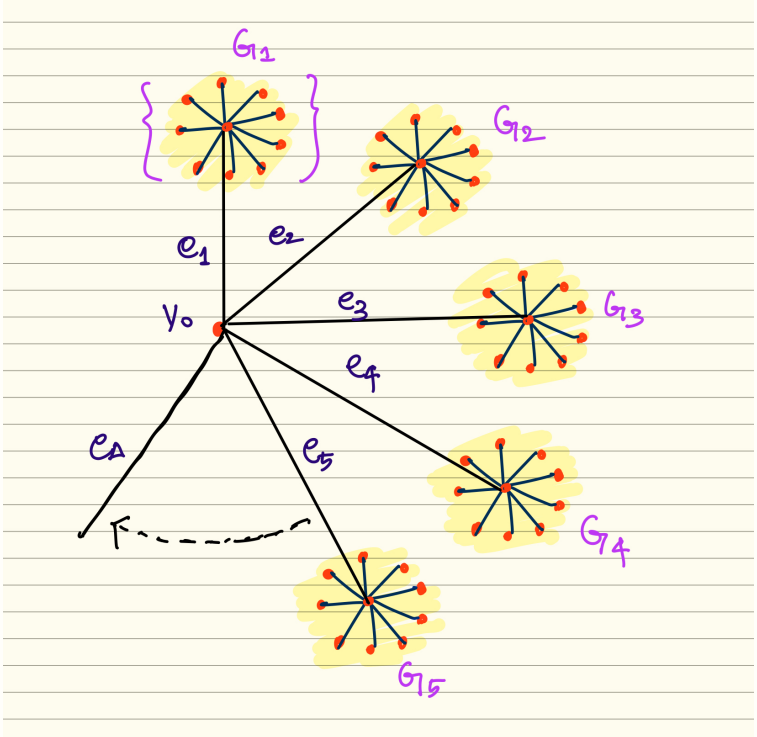
Threfore, the algorithm colours the graph in atmost $2\Delta - 1$ colours.
Hence proved.

(b) (4 marks) Show that there is no deterministic algorithm that uses fewer than $2\Delta - 1$ colors in the worst case.

To that end, consider a graph consisting of disjoint stars with Δ vertices, and edges connecting the center vertex of each of the stars to another fixed vertex. Show that for any deterministic algorithm, there is an adversarial order that can force $2\Delta - 1$ colors.

Solution:

Consider the image below for the given graph case.



- Let the edges corresponding to each star be grouped into $\{G_1, G_2, G_3 \ldots\ldots, \}$.
- To the common vertex V_0 a maximum of Δ centers can be connected.

- Let the edges corresponding to these be $\{e_1, e_2, ..., e_\Delta\}$.
- The number of colours given by any deterministic algorithm A depends on the order in which the edges $\{G_1, G_2, G_3, ..., e_1, e_2, ..., e_\Delta\}$ arrive.

Adversarial Attack:

The worst the adversary can do is keep quering the stars until atleast Δ stars have the same colour configuration . And then connect those stars to the common vertex.

Inputs $= \{G_1, , G_2, G_3, ..., e_1, e_2, e_3, ...e_\Delta\}$

Given the choice of these inputs, any deterministic algorithm will assign $\Delta - 1$ colours for edges in $G_i's$, where all of the stars have the same configuration. When e_1 arrives the algorithm chooses Δ^{th} new colour, then when e_2 arrives it chooses $\Delta + 1^{th}$ new colour and this process continues.

$$cost[A] = \Delta - 1 + \Delta = 2\Delta - 1$$

Hence, the most efficient deterministic algorithm also takes $2\Delta - 1$ colours.
Therefore, there exists no deterministic algorithm that uses fewer than $2\Delta - 1$ colors in the worst case.

Collaborator: **Nitin G**

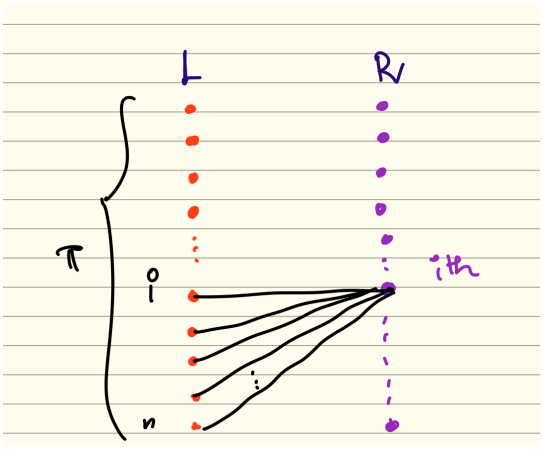
Problem 4 — 7 marks

Our goal in this problem is to show that there does not exist an online algorithm (randomized or deterministic) for the bipartite matching problem that gives a competitive ratio better than $1 - 1/e$. We will use Yao's minimax principle to achieve this.

To that end, let us construct a probability distribution over the input instances of the online bipartite matching problem. First, let us assume that the n vertices in L are known, and let π be a permutation of these vertices chosen uniformly at random. The n vertices in R arrive one-by-one and the i^{th} vertex in R has edges to the last $n - i + 1$ vertices in R according to the permutation π.

(a) (1 mark) Show that every bipartite graph sampled via the process above has a perfect matching.

Solution: For the proposed structure the graph looks like as follows:



If we assign the edges for a given index as the least number according to the permutation π amidst its neighbours, then we have a perfect matching.

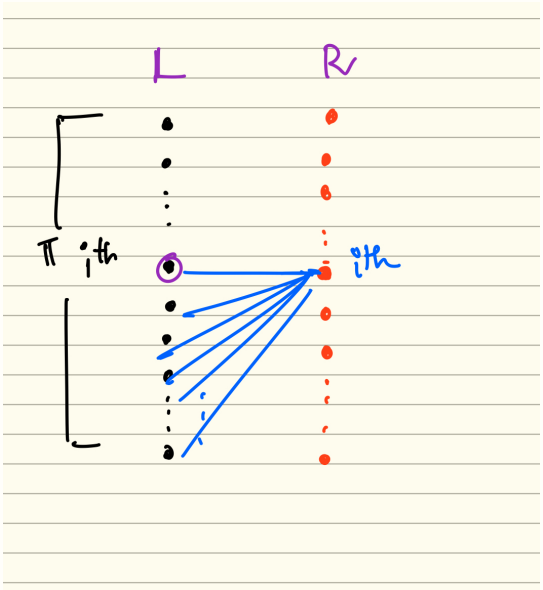
Simply connecting the i^{th} vertex in R to the i^{th} vertex in L according to the

permutation π we get a perfect bipartite matching.

(b) (3 marks) Let A be any deterministic online algorithm for bipartite matching. Prove that for every $i \in \{1, 2, \ldots, n\}$, the probability that A matches the i^{th} vertex of L to some vertex in R is at most

$$\min\left\{\sum_{j=1}^{i} \frac{1}{n-j+1}, 1\right\}.$$

Solution:



The i^{th} vertex in L will not have a matching is none of the first i vertices in R match to it.

Let A_k denote the probability that k^{th} vertex in R does not match to i. According to the given graph, k^{th} vertex matches to the least $n-k+1$ vertices in L.

$$\implies Pr(A_k) = \frac{n-k}{n-k+1}$$

$$\implies Pr(A_k^c) = \frac{1}{n-k+1}$$

$$Pr(\textit{No Matching}) = Pr(\bigcap_{k=1}^{i} A_k) = 1 - Pr(\bigcup_{k=1}^{i} A_k^c)$$

By union bound,

$$Pr(\textit{No Matching}) \geq 1 - \sum_{k=1}^{i} Pr(A_k^c)$$

$$\implies Pr(\textit{No Matching}) \geq 1 - \sum_{k=1}^{i} \frac{1}{n-k+1}$$

Hence,

$$Pr(\textit{Matching}) \leq \sum_{k=1}^{i} \frac{1}{n-k+1}$$

Therefore,

$$Pr(Matching) \leq \min\left\{\sum_{j=1}^{i} \frac{1}{n-j+1}, 1\right\}$$

Hence proved.

(c) (3 marks) Use the part above to conclude that for any deterministic online algorithm for bipartite matching, the expected size of the matching computed by it is at most $n(1-1/e)$.

Solution:

Let X_i denote the random variable that the i^{th} vertex of L is matched to some vertex in R.

By the result in part (b),

$$Pr[X_i] \leq \min\left\{\sum_{j=1}^{i} \frac{1}{n-j+1}, 1\right\}$$

By using the harmonic sum approximation using integrals,

$$\sum_{j=1}^{i} \frac{1}{n-j+1} = \sum_{k=1}^{n} \frac{1}{k} - \sum_{k=1}^{n-i} \frac{1}{k}$$

$$\sum_{j=1}^{i} \frac{1}{n-j+1} \approx \ln\left(\frac{n}{n-i}\right)$$

Comparing this with 1,

$$\ln\left(\frac{n}{n-i}\right) > 1$$

$$\implies i > n\left(1-\frac{1}{e}\right) \text{----------1}$$

Let X be the random variable denoting the size of matching

$$X = \sum_i X_i$$

By linearity of Expectations,

$$E[X] = \sum_i E[X_i]$$

Since, X_i is a Bernoulli random variable,

$$E[X] = \sum_i Pr[X_i] \leq \sum_i \min\left\{\sum_{j=1}^{i} \frac{1}{n-j+1}, 1\right\}$$

By using 1 we can simplify further as,

$$E[X] = \sum_i Pr[X_i] \leq \sum_{i=1}^{n(1-\frac{1}{e})} \sum_{j=1}^{i} \frac{1}{n-j+1} + \sum_{i=n(1-\frac{1}{e})}^{n} 1$$

By exchanging the sum we get,

$$E[X] \leq \sum_{j=1}^{n(1-\frac{1}{e})} \sum_{i=j}^{n(1-\frac{1}{e})} \frac{1}{n-j+1} + \frac{n}{e}$$

On simplifying we get,

$$E[X] \leq n(1-\frac{1}{e}) - \frac{n}{e} \sum_{j=1}^{n(1-\frac{1}{e})} \frac{1}{n-j+1} + \frac{n}{e}$$

Again by integral approximation,

$$\sum_{j=1}^{n(1-\frac{1}{e})} \frac{1}{n-j+1} = \sum_{k=1}^{n} \frac{1}{k} - \sum_{k=1}^{\frac{n}{e}} \frac{1}{k} \approx \ln n - \ln \frac{n}{e} \approx \ln e = 1$$

$$\implies E[X] \leq n(1-\frac{1}{e}) - \frac{n}{e} + \frac{n}{e}$$

$$\implies E[X] \leq n(1-\frac{1}{e})$$

Hence proved.