# CS5314
# Randomized Algorithms

## Lecture 6: Discrete Random Variables and Expectation

## (Coupon Collection, Quicksort)

# Objectives

- Discuss the Coupon Collector's problem
- Analyze expected runtime of Quicksort
- Before that, we define Harmonic number, and give a close bound for that

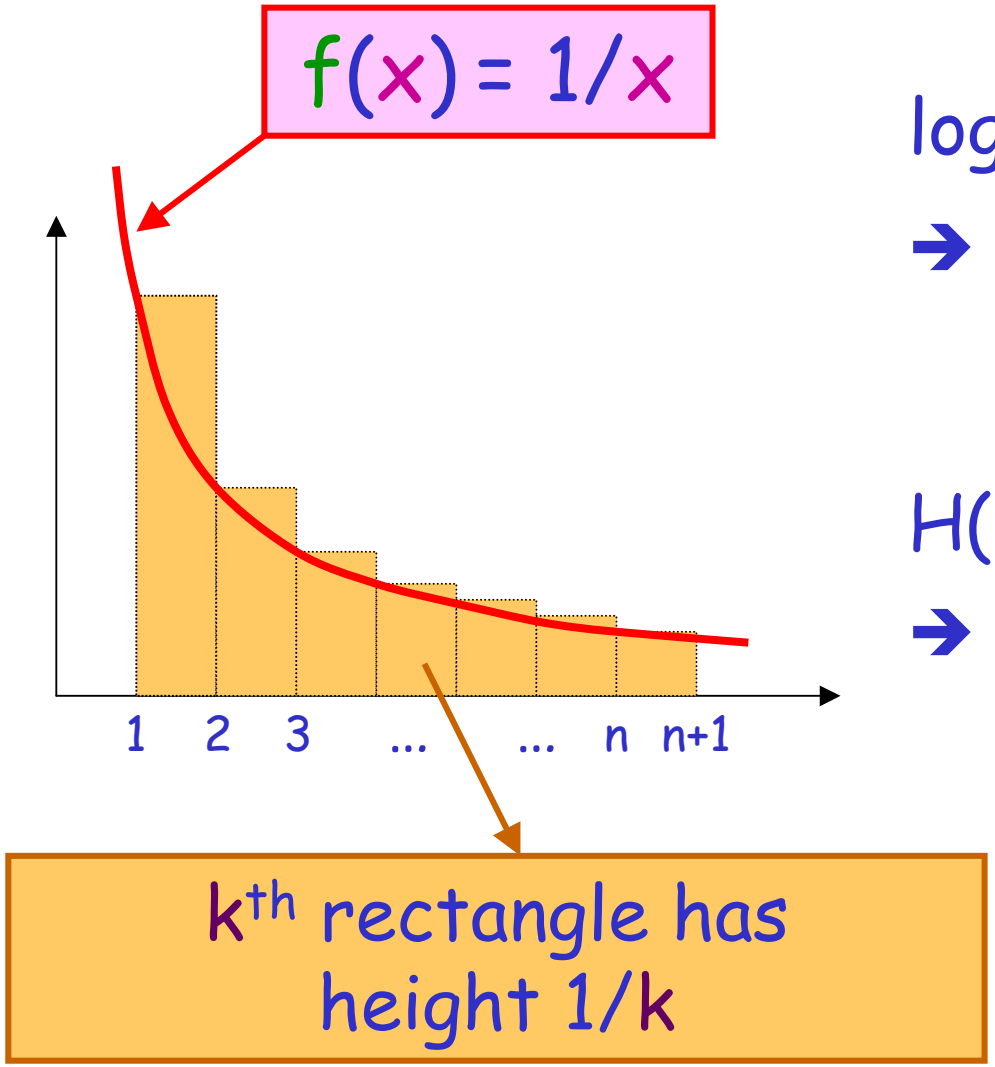
# Harmonic Number

Definition: For a positive integer $n$, the Harmonic number $H(n) = \sum_{k=1 \text{ to } n} 1/k$

Lemma: $\log_e (n+1) \leq H(n) \leq \log_e n + 1$

How to prove?

# Proof (Left Inequality)




$\log_e (n+1) = \int_1^{n+1} (1/x)\, dx$

➔ Area under red curve from x=1 to x=n+1

$H(n) = \sum_{k=1 \text{ to } n} (1/k)$

➔ Area of the first n rectangles

# Proof (Left Inequality)

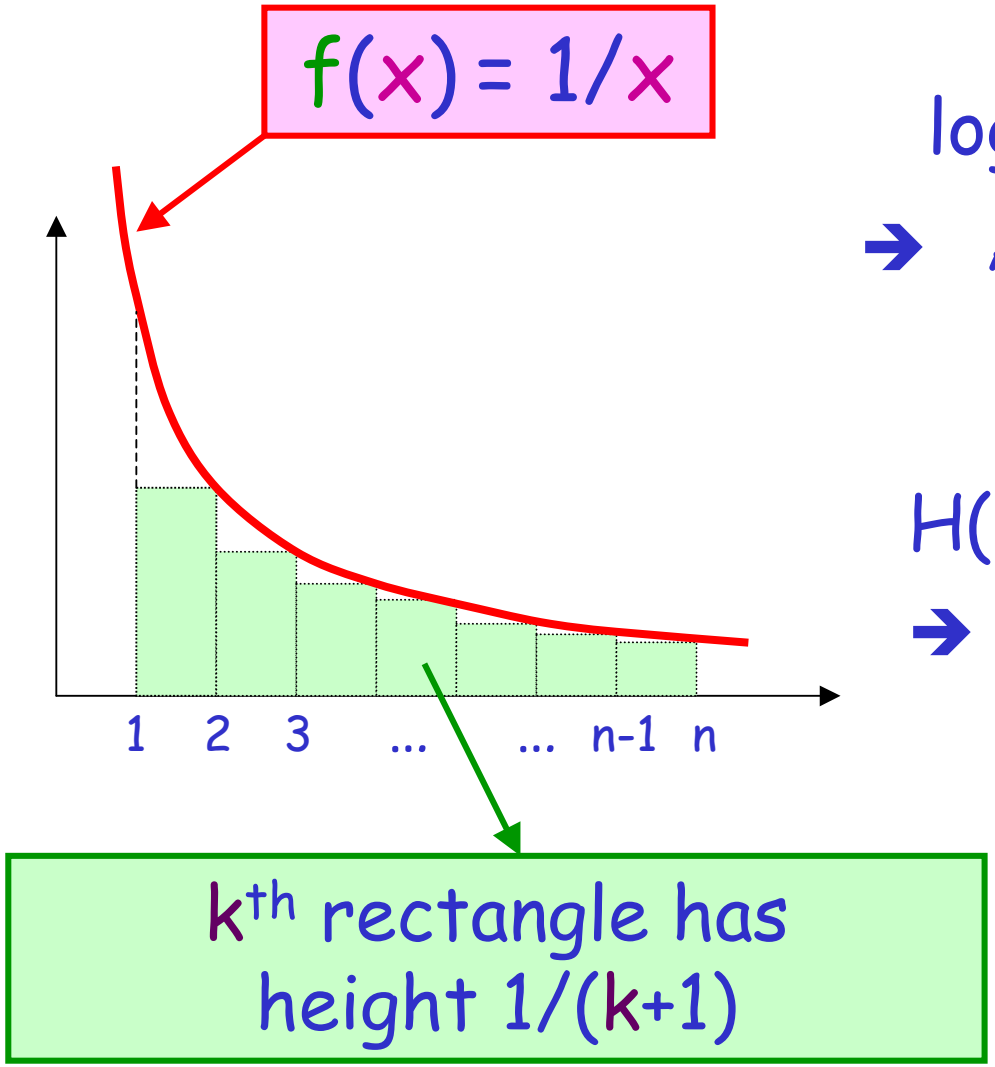
$$\log_e (n+1) = \int_1^{n+1} (1/x)\, dx$$

$$= \int_1^2 (1/x)dx + \int_2^3 (1/x)dx + \ldots + \int_n^{n+1} (1/x)dx$$

$$\leq \int_1^2 (1/1)dx + \int_2^3 (1/2)dx + \ldots + \int_n^{n+1} (1/n)dx$$

$$= 1 + (1/2) + \cdots + 1/n = H(n)$$

# Proof (Right Inequality)




$\log_e n = \int_1^n (1/x)\, dx$

➔ Area under red curve from x=1 to x=n

$H(n) - 1 = \sum_{k=2 \text{ to } n} (1/k)$

➔ Area of the first n-1 rectangles

# Proof (Right Inequality)

$$1 + \log_e n = 1 + \int_1^n (1/x)\, dx$$

$$= 1 + \int_1^2 (1/x)dx + \int_2^3 (1/x)dx + \ldots + \int_{n-1}^n (1/x)dx$$

$$\geq 1 + \int_1^2 (1/2)dx + \int_2^3 (1/3)dx + \ldots + \int_{n-1}^n (1/n)dx$$

$$= 1 + (1/2) + \cdots + 1/n = H(n)$$

# Coupon Collector's Problem

Suppose that if we buy $660 of items from Family Mart, we can pick one of the 10 different deities (uniformly at random)


You are thinking about collecting all these. How much money do you expect to pay?

# Coupon Collector's Problem (2)

Let us solve a more general problem:

- Suppose there are $n$ different cards.
- Each time, the card you obtain is chosen independently and uniformly at random from the $n$ cards

What is the expected number of cards bought in order to get a full collection?

# Coupon Collector's Problem (3)

Let $X$ = #cards bought to get full collection

➔ we are interested in $E[X]$

Let $X_i$ = #cards bought to get a new card, after we have just collected exactly $i-1$ distinct cards

So, $X = X_1 + X_2 + \ldots + X_n$ ➔

$$E[X] = E[X_1] + E[X_2] + \ldots + E[X_n]$$

# Coupon Collector's Problem (4)

What is $E[X_k]$ ?

After we have just collected k-1 cards, let p be the probability that the next card is a new one ➔ $p = (n-k+1)/n$

Note that $X_k$ is a geometric random variable, so $E[X_k] = 1/p = n/(n-k+1)$

# Coupon Collector's Problem (5)

Therefore,

$E[X]$

$= E[X_1] + E[X_2] + \ldots + E[X_n]$

$= n/n + n/(n-1) + n/(n-2) + \ldots + n/1$

$= n\,H(n)$

$= n \log_e n + \Theta(n)$

# Quicksort

- Quicksort is an algorithm for sorting a set of numbers, where the operations are based on comparison
- It is very efficient in practice
- input = a list of numbers
- output = a sorted list of input numbers

```
Quicksort(S) {
 1. If |S| ≤ 1, return S
 2. Else, pick an item, say x, from S
 3. Divide S into S1 and S2 such that
        S1 = a list of all items smaller than x
        S2 = a list of all items greater than x
 4. List1 = Quicksort(S1)
 5. List2 = Quicksort(S2)
 6. return List1, x, List2
}
// Step 3 is done by comparing each item with x
```

# Quicksort (2)

Suppose S contains n numbers.

- In the worst case, how many comparison operations are performed?

  Ans. $n(n-1)/2$

- Suppose each call of Quicksort chooses the median of S as x. How many comparisons are performed?

  Ans. $O( n \log n )$

# Quicksort (3)

One way to guarantee the median is picked is to run the Median-Finding algorithm, which takes $O(|R|)$ extra comparison when we are calling Quicksort(R)

➔ worst-case $O( n \log n )$ time

One drawback: need to write codes for the Median-Finding algorithm…

Suppose we are lazy, what can we do?

# Randomized Quicksort

Let us use randomization to help…

When we call Quicksort(R), suppose:

In Step 2, we choose x by picking an item uniformly at random from R

➔ Let's call this: Randomized Quicksort

Can we bound expected # of comparisons of Randomized Quicksort?

# Randomized Quicksort (analysis)

Observation: In (Randomized) Quicksort, two items can be compared at most once

Let $X$ = number of comparisons

Let $X_{ij}$ = random variable with:

$X_{ij} = 1$ if $i^{th}$ smallest item is compared with $j^{th}$ smallest item

$X_{ij} = 0$ otherwise

So, $X = \sum_{i<j} X_{ij}$ ➔ $E[X] = \sum_{i<j} E[X_{ij}]$

# Randomized Quicksort (analysis)

Note: $X_{ij}$ is an indicator random variable !!

Thus,

$$E[X_{ij}] = Pr(X_{ij} = 1)$$

$$= Pr(\text{ } i^{th} \text{ smallest item is compared with } j^{th} \text{ smallest item })$$

What is this probability?

# Randomized Quicksort (analysis)

Let $y_k$ = $k^{th}$ smallest item in $S$

Observation: $y_i$ is compared with $y_j$ if and only if among all items in

$$\{ y_i, y_{i+1}, y_{i+2}, \ldots, y_j \},$$

either $y_i$ or $y_j$ is picked by Step 2 before the others [why?]

Thus, $E[X_{ij}] = Pr(X_{ij} = 1) = 2/(j-i+1)$

So, $E[X] = \sum_{i<j} E[X_{ij}]$

$= \sum_{i=1 \text{ to } n-1} \sum_{j=i+1 \text{ to } n} 2/(j-i+1)$

$= \sum_{i=1 \text{ to } n-1} \sum_{k=2 \text{ to } n-i+1} 2/k$

changing role of $i$ and $k$

$= \sum_{k=2 \text{ to } n} \sum_{i=1 \text{ to } n-k+1} 2/k$

$= \sum_{k=2 \text{ to } n} (n-k+1)\, 2/k$

pulling out $-k$ term

$= [(n+1) \sum_{k=2 \text{ to } n} 2/k] - 2(n-1)$

$= (2n+2)H(n) - 4n = 2n \log_e n + \Theta(n)$

# Randomized Quicksort (analysis)

Conclusion:

For any input list of numbers,

Expected # of comparisons in Randomized Quicksort

$= 2n \log_e n + \Theta(n)$

# Deterministic Quicksort
## (on random input)

Another related problem is as follows:

- Suppose that each time when we call Quicksort(R), at Step 2, we pick the leftmost item in the list R
- The worst-case # of comparisons in this deterministic algorithm is $O(n^2)$
  - Interesting, if input list is sorted, it becomes a worst-case here !

# Deterministic Quicksort (2)
## (on random input)

Now, suppose that given a set of n number to be sorted, each permutation these numbers are equally likely to be input list

- What is the expected # of comparisons for this deterministic algorithm?

Note: Expectation is now over all input, instead of over all choice of x picked in Step 2

# Deterministic Quicksort (2)
## (on random input)

The expected number of comparisons is

$$2n \log_e n + \Theta(n)$$

To obtain this, we essentially use the same idea as we analyze Randomized Quicksort. Again, the probability that $i^{th}$ smallest item is compared with $j^{th}$ smallest item = $2/(j-i+1)$... [why?]

➔ Thus, we get the same bound

# Quick Quiz

A permutation $\pi$: [1,n] → [1,n] can be represented by a graph with n vertices, labeled by 1, 2, …, n, as follows:

- If $\pi(j) = k$, draw a directed edge from vertex j to vertex k

If each permutation is equally likely, what is the expected # of cycles in the graph?