Introduction to Randomized Algorithms:

Probability Review

Coping with Intractability

- We saw a lot of optimization problems are NP complete/hard
- Sacrifice one of three desired features
    - Solve arbitrary instances of the problem
    - Solve problem to optimality
    - Solve problem in polynomial time
- Coping strategies
    - Design algorithms for special cases of the problem
    - Design approximation algorithms
    - Design faster exponential-time algorithms
    - Rely on heuristics

Approximation Algorithms

- Returns near-optimal solution to a minimization/maximization problem
- An algorithm is a factor α approximation or α-competitive for a problem iff for every instance of the problem it can find a solution within a factor α of the optimal solution
- Maximization problem: $\alpha > 1$ and the solution found by the algorithm is at most α times the optimal
- Minimization problem: $\alpha < 1$ and the approximate solution is at least α times the optimal
- Approximation solution for many NP hard problems:
    - Vertex cover, set cover, hamiltonian cycles
    - MAX-3-SAT, Max cut, etc.
- New algorithmic strategy for many approximation algorithms: randomization

Why Randomness

- Randomization. Allow fair coin flip in unit time.
- Why randomize?
    - Deterministic algorithms offer little flexibility
    - Randomization often leads to surprisingly simple & fast algorithms
- A big part of computer science:
    - Symmetry-breaking protocols
    - Contention resolution
    - Hashing
    - Load balancing
    - Cryptographic protocols, etc

Randomness in Algorithms

- Two ways in which randomness and algorithms can interact
  - The input to the algorithm could be random
    - Analyzing algorithms on random input is called *average-case analysis* (as we are analyzing the behavior of the algorithm on an "average" input, subject to underlying random process)
  - The algorithm itself behaves randomly
    - Inputs are worst-case but algorithm can flip some coins and make decisions based on that, we call these *randomized algorithms*

Where We're Going

- Randomized algorithms fall into two broad categories:
  - **Monte-Carlo algorithms**
    - Find the correct answer most of the time
    - Can usually amplify probability of success with repetitions
    - Example, Karger's min cut
  - **Las-Vegas algorithms**
    - Always find the correct answer, e.g. RandQuick sort
    - But the running time guarantees are not worst (but hold in expectation or with high probability depending on the randomness)
- Randomized data structures: hashing, search trees, filters, etc.

Outline for Coming Lectures

- Discrete probability review and warm up with randomization
    - Assignment 8 will give practice with this
- Randomized algorithms/ data structures (Chapter 13 in KT)
    - Min cut
    - Sorting, selection
    - Approximate Max-cut, MAX-3-SAT
    - Load balancing, balls and bins
    - Skip lists, Bloom filters, etc
- Approximation algorithms come next (Chapter 11 in KT)
    - Vertex cover, set cover, Ham cycle, etc.

Discrete Probability Review

- A discrete probability space consists of a non-empty countable set Ω, called the *sample space* with a probability mass function $\Pr : \Omega \to \mathbb{R}$ s.t.
  - $\Pr[\omega] \geq 0 \quad \forall \omega \in \Omega$ and $\sum_{\omega \in \Omega} \Pr[\omega] = 1$
- E.g.
  - A fair coin: $\Omega = \{\text{heads, tails}\}$ and $\Pr[\text{heads}] = \Pr[\text{tails}] = 1/2$
  - A fair six-sided die: $\Omega = \{1,2,3,4,5,6\}$ and $\Pr[\omega] = 1/6 \quad \forall \omega \in \Omega$

Events and Probability

- **Events.** A subset of Ω are usually called events that are usually a collection of outcomes that satisfy some condition
- Probably of an event A, is $\Pr[A] = \sum_{\omega \in A} \Pr[\omega]$ (extending definition: $\Pr : 2^{\Omega} \to \mathbb{R}$)
- Example: Getting a total of 6 when rolling two fair dice
  - $A = \{(1,5), (2,4), (3,3), (4,2), (5,1)\}$
  - $\Pr(A) = 5/36$
- Just like sets, events can be combined using set operations $\cap$, $\cup$, complement, etc.
  - Rolling two fair dice and getting two 5s
  - $\Pr[\text{two 5s}] = \Pr[\text{ red 5} \cap \text{blue 5}] = 1/36$

Conditional Prob and Union Bound

- Conditional probability and Bayes' Theorem. $\Pr(A \mid B)$ denotes the probability of event A, given that event B happens with non-zero probability
  - $\Pr[A \mid B] = \dfrac{\Pr[A \cap B]}{\Pr[B]}$
- Thus, if both A, B happen with non-zero probability we have:
  $\Pr[A \cap B] = \Pr[A \mid B] \cdot \Pr[B] = \Pr[B \mid A] \cdot \Pr[A]$
- Two events A and B are independent iff $\Pr[A \cap B] = \Pr[A] \cdot \Pr[B]$
- Thus, if two A and B events are independent, then $\Pr[A \mid B] = \Pr[A]$
- Union bound (Very Imp Tool in Randomized Algorithms).
  - Fix n arbitrary events $A_1, \ldots, A_n$ from some sample space Ω, then
    $\Pr[\cup_{i=1}^{n} A_i] \leq \sum_{i=1}^{n} \Pr[A_i]$

Random Variable an Expectation

- A random variable X is a function from a sample space Ω (with a probability measure) to some value set (e.g. real numbers, integers, etc.)
- A random variable from Ω to $\{0,1\}$ is called an indicator random variable
- The expectation of a random variable X is defined as:
  - $E[X] := \sum_x x \cdot P[X = x]$
- E.g. expected value of top face when rolling a dice $= \frac{1}{6} \cdot (1 + 2 + 3 + 4 + 5 + 6)$
- If A is an arbitrary event with $\Pr[A] > 0$, the conditional expectation of X given A is
  - $E[X|A] := \sum_x x \cdot \Pr[X = x|A]$

Random Variable an Expectation

- If A is an arbitrary event with $0 < \Pr[A] < 1$, we have
  - $E[X] = E[X \mid A] \cdot \Pr[A] + E[X \mid \overline{A}]\Pr[\overline{A}]$
- For random variables X and Y, $E[X] = \sum_y E[X \mid Y = y] \cdot \Pr[Y = y]$
- Linearity of expectation (Very Imp Tool in Randomized Algorithms).
  For any real-valued random variables $X_1, X_2, \ldots, X_n$ and any real coefficients $\alpha_1, \alpha_2, \ldots, \alpha_n$

$$E[\sum_{i=1}^{n} (\alpha_i \cdot X_i)] = \sum_{i=1}^{n} (\alpha_i \cdot E[X_i])$$

- Note. Linearity of expectation does not require independence of r.v.s

Common Probability Distributions

- A probability distribution assigns a probability to each possible value of a random variable
- **Uniform distribution** on set of outcomes S. e.g. fair die roll.
  - $\Pr[X = x] = 1/|S|$ and $E[X] = (\sum_{x \in S} x)/|S|$
- **Bernoulli.** Suppose you run an experiment with probability of success p and failure $1 - p$. Example, coin toss where head is success.
- Let X be a Bernoulli or indicator random variable that is 1 if we succeed, and 0 otherwise. Then,

$$E[X] = \sum_x x \cdot \Pr[X = x] = 0 \cdot \Pr[X = 0] + 1 \cdot \Pr[X = 1] = p$$

Binomial Distribution

- Consider now a sequence of n independent coin flips. What is the distribution of heads in the entire sequence?
- More generally consider n independent Bernoulli trials (with success probability p)
- Let X denote the number of successes then X has a Binomial distribution.
- $\Pr[X = j] = \binom{n}{j} p^j (1-p)^{n-j}$
- Let X_i denote the indicator variable that ith trial is a success, and $X = \sum_{i=1}^{n} X_i$
- Then $E[X] = E[\sum_{i=1}^{n} X_i] = \sum_{i=1}^{n} E[X_i] = np$ by linearity of expectation.

Geometric Distribution

- Suppose that we flip a coin until it lands on heads. What is the distribution of the number of coin flips?
- Let X be a random variable representing the number of independent Bernoulli trials (each with success probability p) until first success, then X is said to have a geometric distribution
- $\Pr[X = x] = (1-p)^{x-1}p$
- To calculate the expectation of X, we make use of the nice recursive structure of the process
    - Do one trial, if it is a success we are done, else we need to start over
    - $E[X] = p \cdot 1 + (1-p) \cdot (1 + E[X])$
    - Solving this gives us $E[X] = 1/p$
- That is, in 2 coin flips on average we expect to see a heads

Card Guessing Game: Memoryless

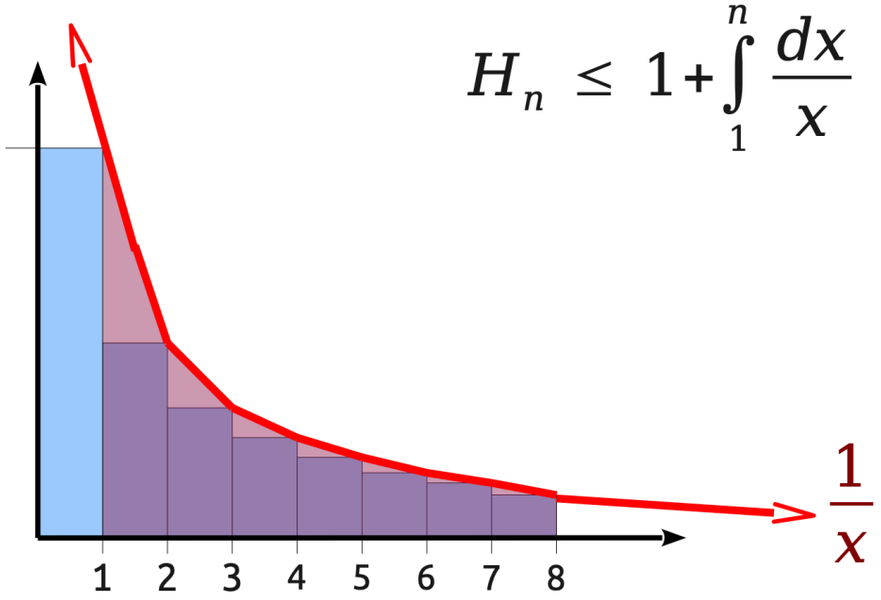
- **Game.** To amaze your friends you have them shuffle deck of n cards and then turn over one card at a time. Before each card is turned, you predict its identity. You have no psychic abilities or memory to remember cards
- Your strategy: guess uniformly at random
- How many predictions do you expect to be correct?
- Let X denote the r.v. equal to the number of correct predictions and X_i denote the indicator variable that the ith guess is correct
- Thus, $X = \sum_{i=1}^{n} X_i$ and $E[X] = E[\sum_{i=1}^{n} X_i] = \sum_{i=1}^{n} E[X_i]$
- $E[X_i] = 0 \cdot \Pr(X_i = 0) + 1 \cdot \Pr(X_i = 1) = \Pr(X_i = 1) = 1/n$
- Thus, $E[X] = 1$

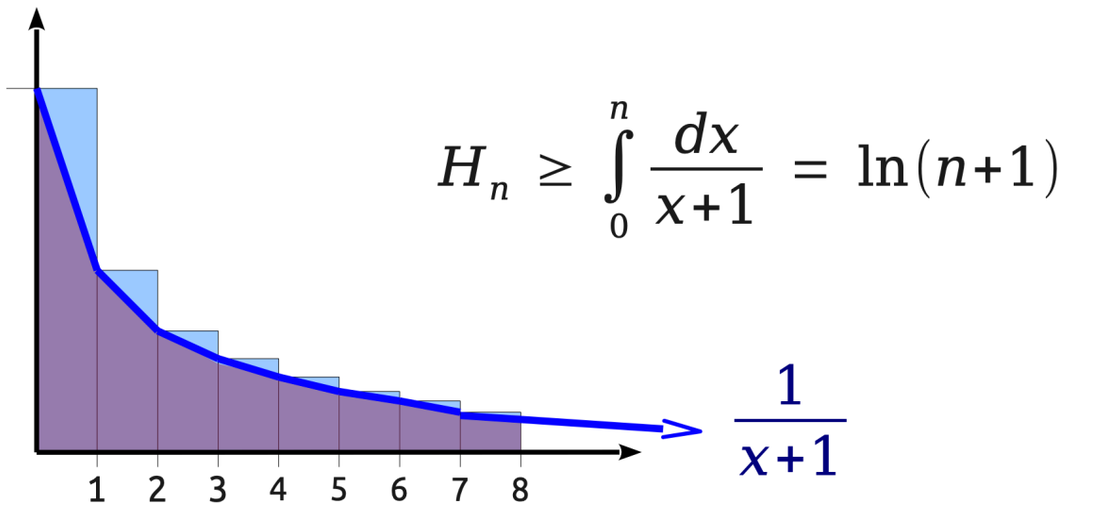
Card Guessing Game: Memoryfull

- Suppose we play the same game but now assume you have the ability to remember cards that have already been turned
- Your strategy: guess uniformly at random among cards that have not been turned over
- Let X denote the r.v. equal to the number of correct predictions and X_i denote the indicator variable that the ith guess is correct
- Thus, $X = \sum_{i=1}^{n} X_i$ and $E[X] = E[\sum_{i=1}^{n} X_i] = \sum_{i=1}^{n} E[X_i]$
- $E[X_i] = \Pr(X_i = 1) = \frac{1}{n - i + 1}$
- Thus, $E[X] = \sum_{i=1}^{n} \frac{1}{n - i + 1} = \sum_{i=1}^{n} \frac{1}{i}$

Harmonic Numbers

- The nth harmonic number, denoted H_n is defined as $H_n = \sum_{i=1}^{n} \frac{1}{i}$
- **Theorem.** $H_n = \Theta(\log n)$
- **Proof Idea.** Upper and lower bound area under the curve



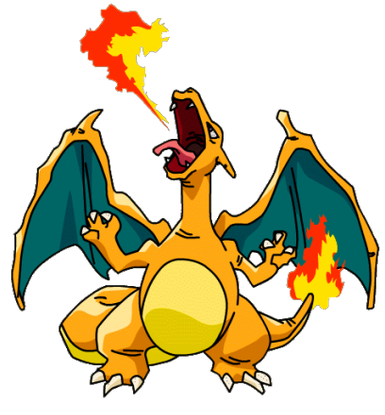


Card Guessing Game: Memoryfull

- Suppose we play the same game but now assume you have the ability to remember cards that have already been turned
- Your strategy: guess uniformly at random among cards that have not been turned over
- Let X denote the r.v. equal to the number of correct predictions and X_i denote the indicator variable that the ith guess is correct
- Thus, $X = \sum_{i=1}^{n} X_i$ and $E[X] = E[\sum_{i=1}^{n} X_i] = \sum_{i=1}^{n} E[X_i]$
- $E[X_i] = \Pr(X_i = 1) = \frac{1}{n - i + 1}$
- Thus, $E[X] = \sum_{i=1}^{n} \frac{1}{n - i + 1} = \sum_{i=1}^{n} \frac{1}{i} = \Theta(\log n)$

Coupon/Pokemon Collector Problem

- Suppose there are n different types of Pokemon cards
- In each trial we purchase a pack that contains a Pokemon card
- We repeat until we have at least one of each type of card, how long (how many packs) does it take in expectation to collect all?
- Let X be the r.v. equal to the numnber of boxes until you first have a a coupon of each type, we want $E[X]$
- We break X into smaller indicator variables as usual
- Idea: we make progress every time we get a card we don't already have



Coupon/Pokemon Collector Problem

- Let X_i denote the number of packs bought during the ith phase (ith phase ends as soon as we see the ith distinct card)
- We can think of each purchase as a biased coin flip: “heads” means we get a new Pokemon, “tails” means we got one we already have”
- For index i, the probability of heads in this step?
- $p = \frac{n - i + 1}{n}$ (each of the n Pokemons are equally likely and there are $n - i + 1$ Pokemon we don’t already have)
- $E[X_i] =$ Expected[number of flips until first heads] $= 1/p$
- $E[X] = E[\sum_{i=1}^{n} X_i] = \sum_{i=1}^{n} E[X_i] = \sum_{i=1}^{n} \frac{n}{n - i + 1} = \sum_{i=1}^{n} \frac{n}{i} = nH_n = \Theta(n \log n)$

Acknowledgments

- Some of the material in these slides are taken from
  - Kleinberg Tardos Slides by Kevin Wayne (https://www.cs.princeton.edu/~wayne/kleinberg-tardos/pdf/04GreedyAlgorithmsI.pdf)
  - Jeff Erickson's Algorithms Book (http://jeffe.cs.illinois.edu/teaching/algorithms/book/Algorithms-JeffE.pdf)
  - Lecture slides: https://web.stanford.edu/class/archive/cs/cs161/cs161.1138/