Lecture 2

Concepts

- Conditional Probability, Independence
- Randomized Algorithms
- Random Variables, Expectation and its Linearity,
- Conditional Expectation, Law of Total Probability.

Examples

- Randomized Min-Cut
- Randomized Quick-Sort
- Randomized Approximation Algorithm for MAX-E3SAT
- Derandomization it using the conditional expectation method
- Expander code

Example 1: Randomized Min-Cut

Min-Cut Problem

Given a multigraph G, find a cut with minimum size.

RANDOMIZED MIN-CUT(G)

```
for i = 1 to n − 2 do
    Pick an edge e_i in G uniformly at random
    Contract two end points of e_i (remove loops)
end for
// At this point, two vertices u, v left
Output all remaining edges between u and v
```

Analysis

- Let C be a minimum cut, $k = |C|$
- If no edge in C is chosen by the algorithm, then C will be returned in the end, and vice versa
- For $i = 1..n-2$, let A_i be the event that $e_i \notin C$ and B_i be the event that $\{e_1, \dots, e_i\} \cap C = \emptyset$

$$\begin{aligned}
& \text{Prob}[C \text{ is returned}] \\
= \; & \text{Prob}[B_{n-2}] \\
= \; & \text{Prob}[A_{n-2} \cap B_{n-3}] \\
= \; & \text{Prob}[A_{n-2} \mid B_{n-3}] \, \text{Prob}[B_{n-3}] \\
= \; & \dots \\
= \; & \text{Prob}[A_{n-2} \mid B_{n-3}] \, \text{Prob}[A_{n-3} \mid B_{n-4}] \cdots \text{Prob}[A_2 \mid B_1] \, \text{Prob}[B_1]
\end{aligned}$$

Analysis

- At step 1, G has min-degree $\geq k$, hence $\geq kn/2$ edges
- Thus,

$$\text{Prob}[B_1] = \text{Prob}[A_1] \geq 1 - \frac{k}{kn/2} = 1 - \frac{2}{n}$$

- At step 2, the min cut is still at least k, hence $\geq k(n-1)/2$ edges. Thus, similar to step 1

$$\text{Prob}[A_2 \mid B_1] \geq 1 - \frac{2}{n-1}$$

- In general,

$$\text{Prob}[A_j \mid B_{j-1}] \geq 1 - \frac{2}{n-j+1}$$

- Consequently,

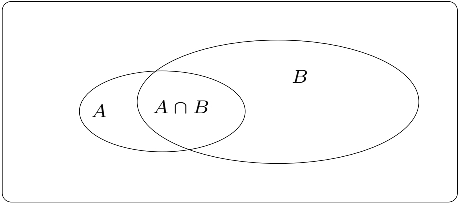
$$\text{Prob}[C \text{ is returned}] \geq \prod_{i=1}^{n-2} \left(1 - \frac{2}{n-i+1}\right) = \frac{2}{n(n-1)}$$

How to Reduce the Failure Probability

- The basic algorithm has failure probability at most $1 - \frac{2}{n(n-1)}$
- How do we lower it?
- Run the algorithm multiple times, say $m \cdot n(n-1)/2$ times, return the smallest cut found
- The failure probability is at most

$$\left(1 - \frac{2}{n(n-1)}\right)^{m \cdot n(n-1)/2} < \frac{1}{e^m}.$$

PTCF: Independence Events and Conditional Probabilities



- The conditional probability of A given B is

$$\text{Prob}[A \mid B] := \frac{\text{Prob}[A \cap B]}{\text{Prob}[B]}$$

- A and B are independent if and only if $\text{Prob}[A \mid B] = \text{Prob}[A]$
- Equivalently, A and B are independent if and only if

$$\text{Prob}[A \cap B] = \text{Prob}[A] \cdot \text{Prob}[B]$$

PTCF: Mutually Independence and Independent Trials

- A set $A_1, \dots, A_n$ of events are said to be independent or mutually independent if and only if, for any $k \leq n$ and $\{i_1, \dots, i_k\} \subseteq [n]$ we have
$$\text{Prob}[A_{i_1} \cap \dots \cap A_{i_k}] = \text{Prob}[A_{i_1}] \cdots \text{Prob}[A_{i_k}].$$

- If n independent experiments (or trials) are performed in a row, with the ith being "successful" with probability p_i, then
$$\text{Prob}[\text{all experiments are successful}] = p_1 \cdots p_n.$$

(Question: what is the sample space?)

Example 2: Randomized Quicksort

$\textsc{Randomized-Quicksort}(A)$

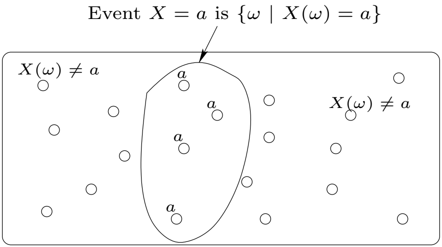
$n \leftarrow$ length(A)
if $n = 1$ **then**
    Return A
else
    Pick $i \in \{1, \dots, n\}$ uniformly at random, $A[i]$ is called the *pivot*
    $L \leftarrow$ elements $\leq A[i]$
    $R \leftarrow$ elements $> A[i]$
    // the above takes one pass through A
    $L \leftarrow \textsc{Randomized-Quicksort}(L)$
    $R \leftarrow \textsc{Randomized-Quicksort}(R)$
    Return $L \cdot A[i] \cdot R$
end if

Analysis of Randomized Quicksort

- The running time is proportional to the number of comparisons
- Let $b_1 \leq b_2 \leq \cdots \leq b_n$ be A sorted non-decreasingly
- For each $i < j$, let X_{ij} be the indicator random variable indicating if b_i was ever compared with b_j
- The expected number of comparisons is

$$\mathsf{E}\left[\sum_{i<j} X_{ij}\right] = \sum_{i<j} \mathsf{E}[X_{ij}] = \sum_{i<j} \mathsf{Prob}[b_i \;\&\; b_j \text{ were compared}]$$

- b_i was compared with b_j if and only if either b_i or b_j was chosen as a pivot before any other in the set $\{b_i, b_{i+1}, \dots, b_j\}$
- Hence, $\mathsf{Prob}[b_i \;\&\; b_j \text{ were compared}] = \frac{2}{j-i+1}$
- Thus, the expected running time is $\Theta(n \lg n)$



- A random variable is a function $X : \Omega \to \mathbb{R}$
- $p_X(a) = \text{Prob}[X = a]$ is called the probability mass function of X
- $P_X(a) = \text{Prob}[X \leq a]$ is called the (cumulative/probability) distribution function of X

PTCF: Expectation and its Linearity

- The expected value of X is defined as

$$\mathsf{E}[X] := \sum_a a\,\mathsf{Prob}[X = a].$$

- For any set $X_1, \dots, X_n$ of random variables, and any constants $c_1, \dots, c_n$

$$\mathsf{E}[c_1 X_1 + \cdots + c_n X_n] = c_1 \mathsf{E}[X_1] + \cdots + c_n \mathsf{E}[X_n]$$

This fact is called linearity of expectation

PTCF: Indicator/Bernoulli Random Variable

$$X : \Omega \to \{0, 1\}$$

$$p = \mathsf{Prob}[X = 1]$$

X is called a Bernoulli random variable with parameter p

If $X = 1$ only for outcomes ω belonging to some event A, then X is called an indicator variable for A

$$\begin{aligned} \mathsf{E}[X] &= p \\ \mathsf{Var}[X] &= p(1-p) \end{aligned}$$

Las Vegas and Monte Carlo Algorithms

Las Vegas Algorithm

A randomized algorithm which always gives the correct solution is called a Las Vegas algorithm.
Its running time is a random variable.

Monte Carlo Algorithm

A randomized algorithm which may give incorrect answers (with certain probability) is called a Monte Carlo algorithm.
Its running time may or may not be a random variable.

Example 3: Max-E3SAT

- An E3-CNF formula is a CNF formula φ in which each clause has *exactly* 3 literals. E.g.,

$$\varphi = \underbrace{(x_1 \vee \bar{x}_2 \vee x_4)}_{\text{Clause } 1} \wedge \underbrace{(x_1 \vee x_3 \vee \bar{x}_4)}_{\text{Clause } 2} \wedge \underbrace{(\bar{x}_2 \vee \bar{x}_3 \vee x_4)}_{\text{Clause } 3}$$

- Max-E3SAT Problem: given an E3-CNF formula φ, find a truth assignment satisfying as many clauses as possible

A Randomized Approximation Algorithm for Max-E3SAT

- Assign each variable to TRUE/FALSE with probability $1/2$

Analyzing the Randomized Approximation Algorithm

- Let X_C be the random variable indicating if clause C is satisfied
- Then, $\text{Prob}[X_C = 1] = 7/8$
- Let S_φ be the number of satisfied clauses. Then,

$$\mathsf{E}[S_\varphi] = \mathsf{E}\left[\sum_C X_C\right] = \sum_C \mathsf{E}[X_C] = 7m/8 \leq \frac{\text{OPT}}{8/7}$$

(m is the number of clauses)

- So this is a randomized approximation algorithm with ratio $8/7$

Derandomization with Conditional Expectation Method

- Derandomization is to turn a randomized algorithm into a deterministic algorithm
- By conditional expectation

$$\mathsf{E}[S_\varphi] = \frac{1}{2}\mathsf{E}[S_\varphi \mid x_1 = \text{ TRUE}] + \frac{1}{2}\mathsf{E}[S_\varphi \mid x_1 = \text{ FALSE}]$$

- Both $\mathsf{E}[S_\varphi \mid x_1 = \text{ TRUE}]$ and $\mathsf{E}[S_\varphi \mid x_1 = \text{ FALSE}]$ can be computed in polynomial time
- Suppose $\mathsf{E}[S_\varphi \mid x_1 = \text{ TRUE}] \geq \mathsf{E}[S_\varphi \mid x_1 = \text{ FALSE}]$, then

$$\mathsf{E}[S_\varphi \mid x_1 = \text{ TRUE}] \geq \mathsf{E}[S_\varphi] \geq 7m/8$$

- Set $x_1 = \text{TRUE}$, let φ' be φ with c clauses containing x_1 removed, and all instances of $x_1, \bar{x}_1$ removed.
- Recursively find value for x_2

PTCF: Law of Total Probabilities, Conditional Expectation

- Law of total probabilities: let $A_1, A_2, \dots$ be any partition of Ω, then
$$\mathsf{Prob}[A] = \sum_{i \geq 1} \mathsf{Prob}[A \mid A_i] \,\mathsf{Prob}[A_i]$$
(Strictly speaking, we also need "*and each A_i is measurable*," but that always holds for finite Ω.)
- The conditional expectation of X given A is defined by
$$\mathsf{E}[X \mid A] := \sum_a a \,\mathsf{Prob}[X = a \mid A].$$
- Let $A_1, A_2, \dots$ be any partition of Ω, then
$$\mathsf{E}[X] = \sum_{i \geq 1} \mathsf{E}[X \mid A_i] \,\mathsf{Prob}[A_i]$$
- In particular, let Y be any discrete random variable, then
$$\mathsf{E}[X] = \sum_y \mathsf{E}[X \mid Y = y] \,\mathsf{Prob}[Y = y]$$

Example 4: Error-Correcting Codes

- Message $\mathbf{x} \in \{0,1\}^k$
- Encoding $f(\mathbf{x}) \in \{0,1\}^n$, $n > k$, f an injection
- $C = \{f(\mathbf{x}) \mid \mathbf{x} \in \{0,1\}^k\}$: codewords
- $f(\mathbf{x})$ is sent over noisy channel, few bits altered
- $\mathbf{y}$ is received instead of $f(\mathbf{x})$
- Find codeword $\mathbf{z}$ "closest" to $\mathbf{y}$ in Hamming distance
- Decoding $\mathbf{x}' = f^{-1}(\mathbf{z})$
- Measure of utilization: relative rate of C

$$R(C) = \frac{\log |C|}{n}$$

- Measure of noise tolerance: relative distance of C

$$\delta(C) = \frac{\min_{\mathbf{c}_1,\mathbf{c}_2 \in C} \mathsf{Dist}(\mathbf{c}_1, \mathbf{c}_2)}{n}$$

Linear Codes

- For any $\mathbf{x} \in \mathbb{F}_2^n$, define

$$\textsc{weight}(\mathbf{x}) = \text{ number of 1-coordinates of } \mathbf{x}$$

- E.g., $\textsc{weight}(1001110) = 4$
- If C is a k-dimensional subspace of $\mathbb{F}_2^n$, then

$$\begin{aligned} |C| &= 2^k \\ \delta(C) &= \min\{\textsc{weight}(\mathbf{x}) \mid \mathbf{x} \in C\} \end{aligned}$$

- Every such C can be defined by a parity check matrix $\mathbf{A}$ of dimension $(n-k) \times n$:

$$C = \{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{0}\}$$

- Conversely, every $(n-k) \times n$ matrix $\mathbf{A}$ defines a code C of dimension $\geq k$

A Communication Problem

Large rate and large distance are conflicting goals

Problem

Does there exist a family of codes C_k, $|C_k| = 2^k$, for infinitely many k, such that

$$R(C_k) \geq R_0 > 0$$

and

$$\delta(C_k) \geq \delta_0 > 0$$

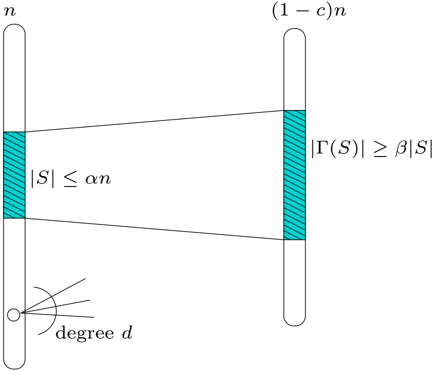
(Yes, using "magical graphs.")

Practicality

Design such a family explicitly, such that the codes are efficiently encodable and decodable.

Magical Graph

(n, c, d, α, β)-graph



c, d, α, β are constants, n varies.

From Magical Graphs to Code Family

- Suppose (n, c, d, α, β)-graphs exist for infinitely many n, and constants c, d, α, β such that $\beta > d/2$
- Consider such a $G = (L \cup R, E)$, $|L| = n, |R| = (1-c)n = m$
- Let $\mathbf{A} = (a_{ij})$ be the $m \times n$ 01-matrix, column indexed by L, and row-indexed by R, $a_{ij} = 1$ iff $(i,j) \in E$
- Define a linear code with $\mathbf{A}$ as parity check:

$$C = \{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{0}\}$$

- Then, $\dim(C) = n - \text{rank}(A) \geq cn$, and

$$|C| = 2^{\dim(C)} \geq 2^{cn} \Rightarrow R(C) \geq c$$

- For every $\mathbf{x} \in C$, $\text{WEIGHT}(\mathbf{x}) \geq \alpha n$, hence

$$\delta(C) = \frac{\min\{\text{WEIGHT}(\mathbf{x}) \mid \mathbf{x} \in C\}}{n} \geq \alpha$$

Existence of Magical Graph with $\beta > d/2$

- Determine n, c, d, α, β later
- Let $L = [n], R = [(1-c)n]$.
- Choose each of the d neighbors for $u \in L$ uniformly at random
- For $1 \leq s \leq \alpha n$, let B_s be the "bad" event that some subset S of size s has $|\Gamma(S)| < \beta|S|$
- For each $S \subset L$, $T \subset R$, $|S| = s, |T| = \beta s$, define

$$X_{S,T} = \begin{cases} 1 & \Gamma(S) \subseteq T \\ 0 & \Gamma(S) \not\subseteq T \end{cases}$$

- Then,

$$\text{Prob}[B_s] \leq \text{Prob}\left[\sum_{S,T} X_{S,T} > 0\right] \leq \sum_{S,T} \text{Prob}[X_{S,T} = 1]$$

Existence of Magical Graph with $\beta > d/2$

$$
\begin{aligned}
\text{Prob}[B_s] &\leq \binom{n}{s}\binom{(1-c)n}{\beta s}\left(\frac{\beta s}{(1-c)n}\right)^{sd} \\
&\leq \left(\frac{ne}{s}\right)^s \left(\frac{(1-c)ne}{\beta s}\right)^{\beta s} \left(\frac{\beta s}{(1-c)n}\right)^{sd} \\
&= \left[\left(\frac{s}{n}\right)^{d-\beta-1}\left(\frac{\beta}{1-c}\right)^{d-\beta} e^{\beta+1}\right]^s \\
&\leq \left[\left(\frac{\alpha\beta}{1-c}\right)^{d-\beta} \cdot \frac{e^{\beta+1}}{\alpha}\right]^s
\end{aligned}
$$

Choose $\alpha = 1/100$, $c = 1/10$, $d = 32$, $\beta = 17 > d/2$,

$$\text{Prob}[B_s] \leq 0.092^s$$

Existence of Magical Graph with $\beta > d/2$

The probability that such a randomly chosen graph is **not** an (n, c, d, α, β)-graph is at most

$$\sum_{s=1}^{\alpha n} \mathsf{Prob}[B_s] \leq \sum_{s=1}^{\infty} 0.092^s = \frac{0.092}{1 - 0.092} < 0.11$$

Not only such graphs exist, there are **a lot** of them!!!