Why Probability in Computing?

- Almost any advance computer application today has some randomization/statistical components:
  - Network security
  - Cryptography
  - Web search and Web advertising
  - Spam filtering
  - Recommendation systems Amazon, Netfix,..
  - Machine learning
  - Communication protocols
  - Computational finance
  - System biology
  - DNA sequencing

Probability in Discrete Mathematics

- Random graphs approximate large graphs well
- Existence proofs
- Probabilistic proofs can often be "derandomized", leading to algorithms.
- Randomized rounding of LP solutions often leads to simple approximation algorithms.

Probability and Computing

- Randomized algorithms - random steps help!
- Probabilistic analysis of algorithms - average case, almost always case, worst case.
- Statistical inference - Machine learning, data mining...

Course Outline

- Basic (discrete) probability theory, moments, tail bounds.
- Randomized algorithms, probabilistic analysis, average and almost sure performance.
- Universal Hash functions.
- Random graphs
- The probabilistic method
- Generating random objects (e.g. spanning trees, permutations)
- Use of randomness in communication protocols.
- Exact algorithms based on derandomizing a randomized algorithm.
- Randomized approximation algorithms.
- Random walks - Markov chains.
- The Monte-Carlo method.
- Applications: sorting, selection, routing, graph algorithms,...
- Entropy, Randomness, and Information
- Dealing with dependencies: Martingales

Verifying polynomial equivalence

Given polynomials $F(x)$ and $G(x)$ both of degree d, where $F(x)$ is given as $F(x) = \Pi_{i=1}^{d}(x - a_i)$ and $G(x)$ is given by its canonical form $G(x) = \sum_{i=0}^{d} c_i x^i$, we want to verify

$$F(x) \equiv G(x)$$

Verifying polynomial equivalence

Given polynomials $F(x)$ and $G(x)$ both of degree d, where $F(x)$ is given as $F(x) = \Pi_{i=1}^{d}(x - a_i)$ and $G(x)$ is given by its canonical form $G(x) = \sum_{i=0}^{d} c_i x^i$, we want to verify

$$F(x) \equiv G(x)$$

standard method: convert $F(x)$ to its cannonical form $F(x) = \sum_{i=0}^{d} b_i x^i$ and check whether $b_i = c_i$ for $i = 0, 1, \ldots, d$. Takes $O(d^2)$operations.

Randomized approach:

1. Pick r uniformly at random in $\{1, 2, \ldots, 100d\}$.
2. Calculate $F(r)$ and $G(r)$ in $O(d)$ time.
3. If $F(r) \neq G(r)$ return 'No'; otherwise return 'Yes'

The probability that the algorithm fails is at most $\frac{d}{100d} = \frac{1}{100}$:
$F(x) - G(x)$ is a polynomium of degree at most d and thus, by the fundamental theorem of algebra, it has at most d distinct roots.

- Our algorithm may fail, but only if it (wrongly) returns 'Yes'.
- We can decrease the error probability in two ways:
  (a) By using a larger interval, say $\{1, 2, \ldots, 1000d\}$. Now the error probability is at most $\frac{1}{1000}$.
  (b) By running the algorithm k times (in time $O(kd)$. Since the out-come in one run is independent of the other runs, the probability that they all answer 'Yes' for an input with $F(x) \not\equiv G(x)$ is at most $\frac{1}{100^k}$.

Method (b) is often preferable as it requires only 5 runs to decrease the error probability to 10^{-10}. We could achieve the same probability by chooing r (sampling) from $\{1, 2, \ldots, 10000000000d\}$, but if d is large it may be difficult to work with integers from such a big range.

Verifying Matrix Multiplication

Given three $n \times n$ matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ in a Boolean field, we want to verify

$$\mathbf{AB} = \mathbf{C}.$$

Verifying Matrix Multiplication

Given three $n \times n$ matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ in a Boolean field, we want to verify

$$\mathbf{AB} \ = \ \mathbf{C}.$$

Standard method: Matrix multiplication - takes $\Theta(n^3)$ ($\Theta(n^{2.37})$) operations.

Randomized algorithm:

1. Chooses a random vector $\bar{r} = (r_1, r_2, \ldots, r_n) \in \{0, 1\}^n$.
2. Compute $\mathbf{B}\bar{r}$;
3. Compute $\mathbf{A}(\mathbf{B}\bar{r})$;
4. Computes $\mathbf{C}\bar{r}$;
5. If $\mathbf{A}(\mathbf{B}\bar{r}) \neq \mathbf{C}\bar{r}$ return $\mathbf{AB} \neq \mathbf{C}$, else return $\mathbf{AB} = \mathbf{C}$.

The algorithm takes $\Theta(n^2)$ time.

Theorem

If $\mathbf{AB} \neq \mathbf{C}$, and $\bar{r}$ is chosen uniformly at random from $\{0, 1\}^n$, then

$$\Pr(\mathbf{AB}\bar{r} = \mathbf{C}\bar{r}) \leq \frac{1}{2}.$$

Probability Space

Definition

A probability space has three components:

1. A sample space Ω, which is the set of all possible outcomes of the random process modeled by the probability space;
2. A family of sets $\mathcal{F}$ representing the allowable events, where each set in $\mathcal{F}$ is a subset of the sample space Ω;
3. A probability function $\Pr : \mathcal{F} \rightarrow \mathbf{R}$, satisfying the definition below.

An element of Ω is a simple event. In a discrete probability space we use $\mathcal{F} = 2^{\Omega}$.

Probability Function

Definition

A probability function is any function $\Pr : \mathcal{F} \to \mathbf{R}$ that satisfies the following conditions:

1. For any event E, $0 \leq \Pr(E) \leq 1$;
2. $\Pr(\Omega) = 1$;
3. For any finite or countably infinite sequence of pairwise mutually disjoint events $E_1, E_2, E_3, \ldots$

$$\Pr\left(\bigcup_{i\geq 1} E_i\right) = \sum_{i\geq 1} \Pr(E_i).$$

The probability of an event is the sum of the probabilities of its simple events.

Independent Events

Definition

Two events E and F are independent if and only if

$$\Pr(E \cap F) \;=\; \Pr(E) \cdot \Pr(F).$$

More generally, events $E_1, E_2, \ldots E_k$ are mutually independent if and only if for any subset $I \subseteq [1, k]$,

$$\Pr\left(\bigcap_{i \in I} E_i\right) \;=\; \prod_{i \in I} \Pr(E_i).$$

Conditional Probability

We have two coins, coin A is a fair coin, coin B has probability $2/3$ to come up HEAD. We chose a coin at random and got HEAD. What is the probability that we chose coin A?
$E_1 =$ the event "Chosen coin A".
$E_2 =$ the event "outcome is HEAD".
The conditional probability that we chose coin A given that the outcome is HEAD is denoted

$$Pr(E_1 \mid E_2).$$

Computing Conditional Probabilities

Definition

The conditional probability that event E_1 occurs given that event E_2 occurs is

$$\Pr(E_1 \mid E_2) \;=\; \frac{\Pr(E_1 \cap E_2)}{\Pr(E_2)}.$$

The conditional probability is only well-defined if $\Pr(E_2) > 0$.

By conditioning on E_2 we restrict the sample space to the set E_2. Thus we are interested in $Pr(E_1 \cap E_2)$ "normalized" by $Pr(E_2)$.

Example - a posteriori probability

We are given 2 coins. One is a fair coin A, the other coin, B has probability $2/3$ for HEAD.
We choose a coin at random, i.e. each coin is chosen with probability $1/2$.
Given that we got head, what is the probability that we chose the fair coin A???

Define a sample space of ordered pairs $(coin, outcome)$.
The sample space has four points

$$\{(A, h), (A, t), (B, h), (B, t)\}$$

$$Pr((A, h)) = Pr((A, t)) = 1/4$$

$$Pr((B, h)) = 1/2 * 2/3 = 1/3$$

$$Pr((B, t)) = 1/2 * 1/3 = 1/6$$

Define two events:
$E_1 =$ "Chose coin A".
$E_2 =$ "Outcome is head".

$$Pr(E_1 \mid E_2) = \frac{Pr(E_1 \cap E_2)}{Pr(E_2)} = \frac{1/4}{1/4 + 1/3} = 3/7.$$

Verifying Matrix Multiplication

Randomized algorithm:

1. Chooses a random vector $\bar{r} = (r_1, r_2, \ldots, r_n) \in \{0, 1\}^n$.
2. Compute $\mathbf{B}\bar{r}$;
3. Compute $\mathbf{A}(\mathbf{B}\bar{r})$;
4. Computes $\mathbf{C}\bar{r}$;
5. If $\mathbf{A}(\mathbf{B}\bar{r}) \neq \mathbf{C}\bar{r}$ return $\mathbf{AB} \neq \mathbf{C}$, else return $\mathbf{AB} = \mathbf{C}$.

The algorithm takes $\Theta(n^2)$ time.

Theorem

If $\mathbf{AB} \neq \mathbf{C}$, and $\bar{r}$ is chosen uniformly at random from $\{0, 1\}^n$, then

$$\Pr(\mathbf{AB}\bar{r} = \mathbf{C}\bar{r}) \leq \frac{1}{2}.$$

Lemma

Choosing $\bar{r} = (r_1, r_2, \ldots, r_n) \in \{0,1\}^n$ *uniformly at random is equivalent to choosing each* r_i *independently and uniformly from* $\{0,1\}$.

Proof.

If each r_i is chosen independently and uniformly at random, each of the 2^n possible vectors $\bar{r}$ is chosen with probability 2^{-n}, giving the lemma. $\square$

Proof:

Let $\mathbf{D} = \mathbf{AB} - \mathbf{C} \neq 0$.
$\mathbf{AB}\bar{r} = \mathbf{C}\bar{r}$ implies that $\mathbf{D}\bar{r} = 0$.
Since $\mathbf{D} \neq 0$ it has some non-zero entry; assume d_{11}.
For $\mathbf{D}\bar{r} = 0$, it must be the case that

$$\sum_{j=1}^{n} d_{1j} r_j = 0,$$

or equivalently

$$r_1 = -\frac{\sum_{j=2}^{n} d_{1j} r_j}{d_{11}}. \tag{1}$$

Here we use $d_{11} \neq 0$.

Principle of Deferred Decision

Assume that we fixed $r_2, \ldots, r_n$.
The RHS is already determined, the only variable is r_1.

$$r_1 \quad = \quad -\frac{\sum_{j=2}^{n} d_{1j} r_j}{d_{11}}. \tag{2}$$

Probability that $r_1 =$ RHS is no more than $1/2$.

More formally, summing over all collections of values $(x_2, x_3, x_4, \dots, x_n) \in \{0,1\}^{n-1}$, we have

$$\begin{aligned}
& \Pr(\mathbf{AB}\bar{r} = \mathbf{C}\bar{r}) \\
= & \sum_{(x_2,\dots,x_n)\in\{0,1\}^{n-1}} \Pr\left(\mathbf{AB}\bar{r} = \mathbf{C}\bar{r} \mid (r_2,\dots,r_n) = (x_2,\dots,x_n)\right) \\
& \qquad\qquad\qquad \cdot \Pr\left((r_2,\dots,r_n) = (x_2,\dots,x_n)\right) \\
= & \sum_{(x_2,\dots,x_n)\in\{0,1\}^{n-1}} \Pr\left((\mathbf{AB}\bar{r} = \mathbf{C}\bar{r}) \cap ((r_2,\dots,r_n) = (x_2,\dots,x_n))\right) \\
\le & \sum_{(x_2,\dots,x_n)\in\{0,1\}^{n-1}} \Pr\left(\left(r_1 = -\frac{\sum_{j=2}^{n} d_{1j} r_j}{d_{11}}\right) \cap ((r_2,\dots,r_n) = (x_2,\dots,x_n))\right) \\
= & \sum_{(x_2,\dots,x_n)\in\{0,1\}^{n-1}} \Pr\left(r_1 = -\frac{\sum_{j=2}^{n} d_{1j} r_j}{d_{11}}\right) \cdot \Pr\left((r_2,\dots,r_n) = (x_2,\dots,x_n)\right) \\
\le & \sum_{(x_2,\dots,x_n)\in\{0,1\}^{n-1}} \frac{1}{2} \Pr((r_2,\dots,r_n) = (x_2,\dots,x_n)) \\
= & \frac{1}{2}.
\end{aligned}$$

Theorem (Law of Total Probability)

Let $E_1, E_2, \ldots, E_n$ be mutually disjoint events in the sample space Ω, and $\cup_{i=1}^{n} E_i = \Omega$, then

$$\Pr(B) = \sum_{i=1}^{n} \Pr(B \cap E_i) = \sum_{i=1}^{n} \Pr(B \mid E_i) \Pr(E_i).$$

Proof.

Since the events E_i, $i = 1, \ldots, n$ are disjoint and cover the entire sample space Ω,

$$\Pr(B) = \sum_{i=1}^{n} \Pr(B \cap E_i) = \sum_{i=1}^{n} \Pr(B \mid E_i) \Pr(E_i).$$

$\square$

Smaller Error Probability

The test has a one side error, repeated tests are independent.

- Run the test k times.
- Accept $AB = C$ if it passed all k tests.

Theorem

The probability of making a mistake is $\leq (1/2)^k$.

Bayes' Law

Theorem (Bayes' Law)

Assume that $E_1, E_2, \ldots, E_n$ *are mutually disjoint sets such that* $\cup_{i=1}^{n} E_i = \Omega$, *then*

$$\Pr(E_j \mid B) = \frac{\Pr(E_j \cap B)}{\Pr(B)} = \frac{\Pr(B \mid E_j)\Pr(E_j)}{\sum_{i=1}^{n} \Pr(B \mid E_i)\Pr(E_i)}.$$

Application: Finding a Biased Coin

- We are given three coins, two of the coins are fair and the third coin is biased, landing heads with probability $2/3$. We need to identify the biased coin.
- We flip each of the coins. The first and second coins come up heads, and the third comes up tails.
- What is the probability that the first coin is the biased one?

Let E_i be the event that the i-th coin flipped is the biased one, and let B be the event that the three coin flips came up heads, heads, and tails.
Before we flip the coins we have $\Pr(E_i) = 1/3$ for $i = 1, \ldots, 3$, thus

$$\Pr(B \mid E_1) = \Pr(B \mid E_2) = \frac{2}{3} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{6},$$

and

$$\Pr(B \mid E_3) = \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{3} = \frac{1}{12}.$$

Applying Bayes' law we have

$$\Pr(E_1' \mid B) = \frac{\Pr(B \mid E_1)\Pr(E_1)}{\sum_{i=1}^{3} \Pr(B \mid E_i)\Pr(E_i)} = \frac{2}{5}.$$

The outcome of the three coin flips increases the probability that the first coin is the biased one from $1/3$ to $2/5$.

Bayesian approach

- Start with an *prior* model, giving some initial value to the model parameters.
- This model is then modified, by incorporating new observations, to obtain a *posterior* model that captures the new information.

Example: randomized matrix multiplication test

- We want to evaluate the increase in confidence through repeated tests.
- If we have no information about the process that generated the identity, a reasonable prior assumption is that the identity is correct with probability $1/2$.
- If we run the randomized test once and it returns that the matrix identity is correct, how does it change our confidence in the identity?

Let E be the event that the identity is correct, and let B be the event that the test returns that the identity is correct.
We start with $\Pr(E) = \Pr(\bar{E}) = 1/2$, and since the test has a one side error bounded by $1/2$, we have $\Pr(B \mid E) = 1$, and $\Pr(B \mid \bar{E}) \leq 1/2$.
Applying Bayes' law we have

$$\begin{aligned}
\Pr(E' \mid B) &= \frac{\Pr(B \mid E)\Pr(E)}{\Pr(B \mid E)\Pr(E) + \Pr(B \mid \bar{E})\Pr(\bar{E})} \\
&\geq \frac{1/2}{1/2 + 1/2 \cdot 1/2} = 2/3.
\end{aligned}$$

- Assume now that we run the randomized test again and it again returns that the identity is correct.
- After the first test, the prior model was revised, so $\Pr(E) \geq 2/3$, and $\Pr(\bar{E}) \leq 1/3$.
- $\Pr(B \mid E) = 1$ and $\Pr(B \mid \bar{E}) \leq 1/2$.

Applying Bayes' law we have

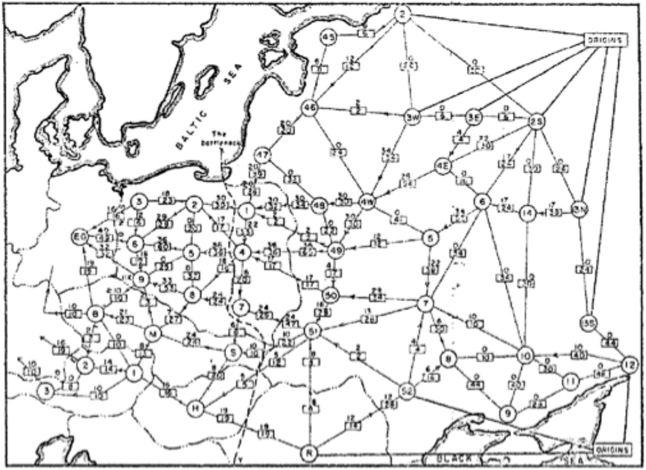
$$\Pr(E' \mid B) \geq \frac{2/3}{2/3 + 1/3 \cdot 1/2} = 4/5.$$

In general, if before running the test our prior model is that $\Pr(E) \geq 2^i/(2^i+1)$, and the test returns that the identity is correct (event B), then

$$\Pr(E' \mid B) \geq \frac{\frac{2^i}{2^i+1}}{\frac{2^i}{2^i+1} + \frac{1}{2}\frac{1}{2^i+1}} = \frac{2^{i+1}}{2^{i+1}+1} = 1 - \frac{1}{2^i+1}.$$

Thus, if all 100 calls to the matrix identity test return that the identity is correct, then our confidence in the correctness of this identity is at least $1 - \frac{1}{2^{100}+1}$.

Min-Cut



Source: *On the history of the transportation and maximum flow problems.*
Alexander Schrijver in Math Programming, 91: 3, 2002.

Min-Cut Algorithm

Input: An n-node graph G.

Output: A minimal set of edges that disconnects the graph.

1. **Repeat $n-2$ times:**
   1. Pick an edge uniformly at random.
   2. Contract the two vertices connected by that edge, eliminate all edges connecting the two vertices.
2. Output the set of edges connecting the two remaining vertices.

Theorem

The algorithm outputs a min-cut set of edges with probability $\geq \frac{2}{n(n-1)}$.

Lemma

Vertex contraction does not reduce the size of the min-cut set. (Contraction can only increase the size of the min-cut set.)

Proof.

Every cut set in the new graph is a cut set in the original graph. □

Analysis of the Algorithm

Assume that the graph has a min-cut set of k edges.
We compute the probability of finding one such set C.

Lemma

If the edge contracted does not belong to C, no other edge eliminated in that step belongs to C.

Proof.

A contraction eliminates a set of parallel edges (edges connecting one pair of vertices).
Parallel edges either all belong, or don't belong to C. □

Let $E_i =$ "the edge contracted in iteration i is not in C."
Let $F_i = \cap_{j=1}^{i} E_j =$ "no edge of C was contracted in the first i iterations".
We need to compute $Pr(F_{n-2})$

Since the minimum cut-set has k edges, all vertices have degree $\geq k$, and the graph has $\geq nk/2$ edges.
There are at least $nk/2$ edges in the graph, k edges are in C.
$Pr(E_1) = Pr(F_1) \geq 1 - \frac{2k}{nk} = 1 - \frac{2}{n}$.

Assume that the first contraction did not eliminate an edge of C (conditioning on the event $E_1 = F_1$).
After the first vertex contraction we are left with an $n-1$ node graph, with minimum cut set, and minimum degree $\geq k$.
The new graph has at least $k(n-1)/2$ edges.
$Pr(E_2 \mid F_1) \geq 1 - \frac{k}{k(n-1)/2} \geq 1 - \frac{2}{n-1}$.
Similarly,
$Pr(E_i \mid F_{i-1}) \geq 1 - \frac{k}{k(n-i+1)/2} = 1 - \frac{2}{n-i+1}$.

We need to compute

$$Pr(F_{n-2})$$

We use

$$Pr(A \cap B) = Pr(A \mid B)Pr(B)$$

$$Pr(F_{n-2}) =$$

$$Pr(E_{n-2} \cap F_{n-3}) = Pr(E_{n-2} \mid F_{n-3})Pr(F_{n-3}) =$$

$$Pr(E_{n-2} \mid F_{n-3})Pr(E_{n-3} \mid F_{n-4})....Pr(E_2 \mid F_1)Pr(F_1) \geq$$

$$\geq \Pi_{i=1}^{n-2}\left(1 - \frac{2}{n-i+1}\right) = \Pi_{i=1}^{n-2}\left(\frac{n-i-1}{n-i+1}\right)$$

$$= \left(\frac{n-2}{n}\right)\left(\frac{n-3}{n-1}\right)\left(\frac{n-4}{n-2}\right)\cdots\left(\frac{4}{6}\right)\left(\frac{3}{5}\right)\left(\frac{2}{4}\right)\left(\frac{1}{3}\right) = \frac{2}{n(n-1)}.$$

Useful identities:

$$Pr(A \mid B) = \frac{Pr(A \cap B)}{Pr(B)}$$

$$Pr(A \cap B) = Pr(A \mid B)Pr(B)$$

$$Pr(A \cap B \cap C) = Pr(A \mid B \cap C)Pr(B \cap C)$$

$$= Pr(A \mid B \cap C)Pr(B \mid C)Pr(C)$$

Let $A_1, ..., A_n$ be a sequence of events. Let $E_i = \bigcap_{j=1}^{i} A_i$

$$Pr(E_n) = Pr(A_n \mid E_{n-1})Pr(E_{n-1}) =$$

$$Pr(A_n \mid E_{n-1})Pr(A_{n-1} \mid E_{n-2})....P(A_2 \mid E_1)Pr(E_1)$$

Theorem

Assume that we run the randomized min-cut algorithm $n(n-1)\log n$ *times and output the minimum size cut-set found in all the iterations. The probability that the output is not a min-cut set is bounded by*

$$\left(1 - \frac{2}{n(n-1)}\right)^{n(n-1)\log n} \leq e^{-2\log n} = \frac{1}{n^2}.$$

Proof.

The algorithm has a one side error: the output is never smaller than the min-cut value. □

The Taylor series expansion of e^{-x} gives

$$e^{-x} = 1 - x + \frac{x^2}{2!} - \dots\dots$$

Thus, for $x < 1$,

$$1 - x \leq e^{-x}.$$