CMPT 210: Probability and Computing

Lecture 10

Sharan Vaswani

February 3, 2023

Back to throwing dice - Independent Events

Q: Suppose we throw two standard dice one after the other. What is the probability that we get two 6's in a row?

E = We get a 6 in the second throw. F = We get a 6 in the first throw. $E \cap F$ = we get two 6's in a row. We are computing $\Pr[E \cap F]$. $\Pr[E] = \Pr[F] = \frac{1}{6}$.

$\Pr[E|F] = \frac{\Pr[E \cap F]}{\Pr[F]} \implies \Pr[E \cap F] = \Pr[E|F] \Pr[F]$.

Since the two dice are *independent*, knowing that we got a 6 in the first throw does not change the probability that we will get a 6 in the second throw. Hence, $\Pr[E|F] = \Pr[E]$ (conditioning does not change the probability of the event).

Hence, $\Pr[E \cap F] = \Pr[E|F] \Pr[F] = \Pr[E] \Pr[F] = \frac{1}{6} \frac{1}{6} = \frac{1}{36}$.

Independent Events

Independent Events: Events E and F are said to be independent, if knowledge that F has occurred does not change the probability that E occurs. Formally,

$$\Pr[E|F] = \Pr[E]; \quad ; \Pr[E \cap F] = \Pr[E] \Pr[F]$$

Q: I toss two independent, fair coins. What is the probability that I get the HT sequence?

Define E to be the event that I get a heads in the first toss, and F be the event that I get a tails in the second toss. Since the two coins are independent, events E and F are also independent. $\Pr[E \cap F] = \Pr[E] \Pr[F] = \frac{1}{2} \frac{1}{2} = \frac{1}{4}$.

Q: I randomly choose a number from $\{1, 2, \ldots, 10\}$. E is the event that the number I picked is a prime. F is the event that the number I picked is odd. Are E and F independent?

$\Pr[E] = \frac{2}{5}$, $\Pr[F] = \frac{1}{2}$, $\Pr[E \cap F] = \frac{3}{10}$. $\Pr[E \cap F] \neq \Pr[E] \Pr[F]$. Another way: $\Pr[E|F] = \frac{3}{5}$ and $\Pr[E] = \frac{2}{5}$, and hence $\Pr[E|F] \neq \Pr[E]$. Conditioning on F tell us that prime number cannot be 2, so it changes the probability of E.

Independent Events - Example

Q: We have a machine that has 2 independent components. The machine breaks if *each* of its 2 components break. Suppose each component can break with probability p, what is the probability that the machine does not break?

Let E_1 = Event that the first component breaks, E_2 = Event that the second component breaks. M = Event that the machine breaks = $E_1 \cap E_2$.

$\Pr[M] = \Pr[E_1 \cap E_2]$. Since the two components are independent, E_1 and E_2 are independent, meaning that $Pr[E_1 \cap E_2] = Pr[E_1] \Pr[E_2] = p^2$.

Probability that the machine does not break = $\Pr[M^c] = 1 - \Pr[M] = 1 - p^2$.

Independent Events - Examples

Q: We have a new machine that has 2 independent components. The machine breaks if *either* of its 2 components break. Suppose each component can break with probability p, what is the probability that the machine breaks?

For this machine, let M' be the event that it breaks. In this case, $\Pr[M'] = \Pr[E_1 \cup E_2]$.

Incorrect: By the union rule for mutually exclusive events, $\Pr[E_1 \cup E_2] = \Pr[E_1] + \Pr[E_2] = 2p$.

Mistake: *Independence does not imply mutual exclusivity* and we can not use the union rule. Independence implies that for any two events E and F, $\Pr[E \cap F] = \Pr[E] \Pr[F]$, while mutual exclusivity requires that $\Pr[E \cap F] = 0$.

Correct way 1:

$$\begin{aligned}
\Pr[E_1 \cup E_2] &= \Pr[E_1] + \Pr[E_2] - \Pr[E_1 \cap E_2] && \text{(By the inclusion-exclusion rule)} \\
&= \Pr[E_1] + \Pr[E_2] - \Pr[E_1] \Pr[E_2] = 2p - p^2 && \text{(Since E_1 and E_2 are independent.)}
\end{aligned}$$

Independent Events - Examples

Q: We have a new machine that has 2 independent components. The machine breaks if *either* of its 2 components break. Suppose each component can break with probability p, what is the probability that the machine breaks?

Correct way 2:

$$\Pr[E_1 \cup E_2] = 1 - \Pr[(E_1 \cup E_2)^c] = 1 - \Pr[E_1^c \cap E_2^c]$$

(Complement of union of sets is equal to the intersection of the complements of sets)

$$= 1 - \Pr[E_1^c]\Pr[E_2^c] = 1 - (1-p)^2 = 2p - p^2$$

(If E_1 and E_2 are independent, so are E_1^c and E_2^c (Proof on the next slide))

This implies that for the first machine, the probability of failure is p^2 while for the second one, it is $2p - p^2$. Since $p \leq 1$, $p^2 \leq 2p - p^2$, meaning that the first machine fails less often. This is intuitive since it fails only when *both* components fail.

Independent Events - Examples

Q: Prove that if E_1 and E_2 are independent, so are E_1^c and E_2^c. *Proof*:

$$\Pr[(E_1)^c \cap (E_2)^c] = \Pr[(E_1 \cup E_2)^c] = 1 - \Pr[E_1 \cup E_2] = 1 - \Pr[E_1] - \Pr[E_2] + \Pr[E_1 \cap E_2]$$

(By the inclusion-exclusion rule)

$$= 1 - \Pr[E_1] - \Pr[E_2] + \Pr[E_1] \Pr[E_2]$$

(Since E_1 and E_2 are independent)

$$\implies \Pr[(E_1)^c \cap (E_2)^c] = (1 - \Pr[E_1])\,(1 - \Pr[E_2]) = \Pr[E_1^c]\,\Pr[E_2^c]$$

Hence, events E_1^c and E_2^c are independent.

Questions?

Matrix Multiplication

Given two $n \times n$ matrices – A and B, if $C = AB$, then,

$$C_{i,j} = \sum_{k=1}^{n} A_{i,k} B_{k,j}$$

Hence, in the worst case, computing $C_{i,j}$ is an $O(n)$ operation. There are n^2 entries to fill in C and hence, in the absence of additional structure, matrix multiplication takes $O(n^3)$ time.

There are non-trivial algorithms for doing matrix multiplication more efficiently:

- (Strassen, 1969) Requires $O(n^{2.81})$ operations.
- (Coppersmith-Winograd, 1987) Requires $O(n^{2.376})$ operations.
- (Alman-Williams, 2020) Requires $O(n^{2.373})$ operations.
- Belief is that it can be done in time $O(n^{2+\epsilon})$ for $\epsilon > 0$.

Verifying Matrix Multiplication

For simplicity, we will focus on A, B being binary matrices (all entries are either 0 or 1), and matrix multiplication mod 2, i.e. $C_{i,j} = (\sum_{k=1}^{n} A_{i,k} B_{k,j}) \bmod 2$, implying that C is a binary matrix.

Example: $A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, $B = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ then $C = AB = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$

Objective: Verify whether a matrix multiplication operation is correct.

Trivial way: Do the matrix multiplication ourselves, and verify it using $O(n^3)$ (or $O(n^{2.373})$) operations.

Frievald's Algorithm: Randomized algorithm to verify matrix multiplication with high probability in $O(n^2)$ time.

(Basic) Frievald's Algorithm

Q: For $n \times n$ matrices A, B and D, is $D = AB \pmod 2$?

Algorithm:

1. Generate a random n-bit vector x, by making each bit x_i either 0 or 1 *independently* with probability $\frac{1}{2}$. E.g, for $n = 2$, toss a fair coin independently twice with the scheme – H is 0 and T is 1). If we get HT, then set $x = [0\,;\,1]$.
2. Compute $t = Bx \pmod 2$ and $y = At = A(Bx) \pmod 2$ and $z = Dx \pmod 2$.
3. Output "yes" if $y = z$ (all entries need to be equal), else output "no".

Computational complexity: Step 1 can be done in $O(n)$ time. Step 2 requires 3 matrix vector multiplications and can be done in $O(n^2)$ time. Step 3 requires comparing two n-dimensional vectors and can be done in $O(n)$ time. Hence, the total computational complexity is $O(n^2)$.

(Basic) Frievald's Algorithm

Let us run the algorithm on an example. Suppose we have generated $x = [1\,;\,0]$

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad ; \quad B = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \quad ; \quad D = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

$$Bx = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad ; \quad y = A\,(Bx) = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad ; \quad z = Dx = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Hence the algorithm will correctly output "no" since $D \neq AB(\text{mod } 2)$.

Q: Suppose we have generated $x = [1\,;\,1]$. What is y and z?

In this case, $y = z$ and the algorithm will incorrectly output "yes" even though $D \neq AB(\text{mod } 2)$.

(Basic) Frievald's Algorithm

Let us run the algorithm on an example. Suppose we have generated $x = [1\,;\,0]$.

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad ; \quad B = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \quad ; \quad C = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

$$Bx = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad ; \quad y = A\,(Bx) = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad ; \quad z = Cx = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Hence the algorithm will correctly output "yes" since $C = AB(\text{mod } 2)$.

Q: Suppose we have generated $x = [1\,;\,1]$. What is y and z?

In this case again, $y = z$ and the algorithm will correctly output "yes".

(Basic) Frievald's Algorithm

Let us analyze the algorithm for general matrix multiplication (not necessarily (mod 2)).

Case (i): If $D = AB$, does the algorithm always output "yes"? Yes! Since $D = AB$, for any vector x, $Dx = ABx$.

Case (ii) If $D \neq AB$, does the algorithm always output "no"?

Claim: For any input matrices A, B, D if $D \neq AB$, then the (Basic) Frievald's algorithm will output "no" with probability $\geq \frac{1}{2}$.

	Yes	No
$D = AB$	1	0
$D \neq AB$	$< \frac{1}{2}$	$\geq \frac{1}{2}$

Table 1: Probabilities for Basic Frievalds Algorithm

(Basic) Frievald's Algorithm

Proof: If $D \neq AB$, we wish to compute the probability that algorithm outputs "yes" and prove that it less than $\frac{1}{2}$.
Define $E := (AB - D)$ and $r := Ex = (AB - D)x = y - z$. If $D \neq AB$, then $\exists (i, j)$ s.t. $E_{i,j} \neq 0$.

$$\begin{aligned}
\Pr[\text{Algorithm outputs ``yes''}] &= \Pr[y = z] = \Pr[r = \mathbf{0}] \\
&= \Pr[(r_1 = 0) \cap (r_2 = 0) \cap \ldots \cap (r_i = 0) \cap \ldots] \\
&= \Pr[(r_i = 0)] \Pr[(r_1 = 0) \cap (r_2 = 0) \cap \ldots \cap (r_n = 0) | r_i = 0] \\
&\qquad\qquad \text{(By def. of conditional probability)} \\
\implies \Pr[\text{Algorithm outputs ``yes''}] &\leq \Pr[r_i = 0] \qquad\qquad \text{(Probabilities are in } [0, 1])
\end{aligned}$$

To complete the proof, on the next slide, we will prove that $\Pr[r_i = 0] \leq \frac{1}{2}$.

(Basic) Frievald's Algorithm

$$r_i = \sum_{k=1}^{n} E_{i,k}x_k = E_{i,j}x_j + \sum_{k\neq j} E_{i,k}x_k = E_{i,j}x_j + \omega \qquad (\omega := \textstyle\sum_{k\neq j} E_{i,k}x_k)$$

$$\Pr[r_i = 0] = \Pr[r_i = 0|\omega = 0]\Pr[\omega = 0] + \Pr[r_i = 0|\omega \neq 0]\Pr[\omega \neq 0]$$

(By the law of total probability)

$$\Pr[r_i = 0|\omega = 0] = \Pr[x_j = 0] = \frac{1}{2} \qquad (\text{Since } E_{i,j} \neq 0 \text{ and } \Pr[x_j = 1] = \tfrac{1}{2})$$

$$\Pr[r_i = 0|\omega \neq 0] = \Pr[(x_j = 1) \cap E_{i,j} = -\omega] = \Pr[(x_j = 1)]\Pr[E_{i,j} = -\omega|x_j = 1]$$

(By def. of conditional probability)

$$\implies \Pr[r_i = 0|\omega \neq 0] \leq \Pr[(x_j = 1)] = \frac{1}{2} \qquad (\text{Probabilities are in } [0,1],\ \Pr[x_j = 1] = \tfrac{1}{2})$$

$$\implies \Pr[r_i = 0] \leq \frac{1}{2}\Pr[\omega = 0] + \frac{1}{2}\Pr[\omega \neq 0] = \frac{1}{2}\Pr[\omega = 0] + \frac{1}{2}\left[1 - \Pr[\omega = 0]\right] = \frac{1}{2}$$

$(\Pr[E^c] = 1 - \Pr[E])$

$$\implies \Pr[\text{Algorithm outputs ``yes''}] \leq \Pr[r_i = 0] \leq \frac{1}{2}.$$

(Basic) Frievald's Algorithm

Hence, if $D \neq AB$, the Algorithm outputs "yes" with probability $\leq \frac{1}{2} \implies$ the Algorithm outputs "no" with probability $\geq \frac{1}{2}$.

In the worst case, the algorithm can be incorrect half the time! We promised the algorithm would return the correct answer with "high" probability close to 1.

A common trick in randomized algorithms is to have m independent trials of an algorithm and aggregate the answer in some way, reducing the probability of error, thus *amplifying the probability of success*.

Questions?

Frievald's Algorithm

By repeating the *Basic Frievald's Algorithm* (from slide 7) m times, we will amplify the probability of success. The resulting complete Frievald's Algorithm is given by:

1 Run the Basic Frievald's Algorithm for m independent runs.

2 If *any* run of the Basic Frievald's Algorithm outputs "no", output "no".

3 If *all* runs of the Basic Frievald's Algorithm output "yes", output "yes".

	Yes	No
$D = AB$	1	0
$D \neq AB$	$< \frac{1}{2^m}$	$\geq 1 - \frac{1}{2^m}$

Table 2: Probabilities for Frievald's Algorithm

If $m = 20$, then Frievald's algorithm will make mistake with probability $1/2^{20} \approx 10^{-6}$.

Computational Complexity: $O(mn^2)$

Probability Amplification

Consider a randomized algorithm $\mathcal{A}$ that is supposed to solve a binary decision problem i.e. it is supposed to answer either Yes or No. It has a one-sided error – (i) if the true answer is Yes, then the algorithm $\mathcal{A}$ correctly outputs Yes with probability 1, but (ii) if the true answer is No, the algorithm $\mathcal{A}$ incorrectly outputs Yes with probability $\leq \frac{1}{2}$.

Let us define a new algorithm $\mathcal{B}$ that runs algorithm $\mathcal{A}$ m times, and if *any* run of $\mathcal{A}$ outputs No, algorithm $\mathcal{B}$ outputs No. If *all* runs of $\mathcal{A}$ output Yes, algorithm $\mathcal{B}$ outputs Yes.

Q: What is the probability that algorithm $\mathcal{B}$ correctly outputs Yes if the true answer is Yes, and correctly outputs No if the true answer is No?

Probability Amplification - Analysis

$$
\begin{aligned}
&\Pr[\mathcal{B} \text{ outputs Yes} \mid \text{true answer is Yes }]\\
&= \Pr[\mathcal{A}_1 \text{ outputs Yes } \cap \mathcal{A}_2 \text{ outputs Yes } \cap \ldots \cap \mathcal{A}_m \text{ outputs Yes} \mid \text{true answer is Yes }]\\
&= \prod_{i=1}^{m} \Pr[\mathcal{A}_i \text{ outputs Yes} \mid \text{true answer is Yes }] = 1 \qquad \text{(Independence of runs)}\\
&\Pr[\mathcal{B} \text{ outputs No} \mid \text{true answer is No }]\\
&= 1 - \Pr[\mathcal{B} \text{ outputs Yes} \mid \text{true answer is No }]\\
&= 1 - \Pr[\mathcal{A}_1 \text{ outputs Yes } \cap \mathcal{A}_2 \text{ outputs Yes } \cap \ldots \cap \mathcal{A}_m \text{ outputs Yes} \mid \text{true answer is No }]\\
&= 1 - \prod_{i=1}^{m} \Pr[\mathcal{A}_i \text{ outputs Yes} \mid \text{true answer is No }] \geq 1 - \frac{1}{2^m}.
\end{aligned}
$$

When the true answer is Yes, both $\mathcal{B}$ and $\mathcal{A}$ correctly output Yes. When the true answer is No, $\mathcal{A}$ incorrectly outputs Yes with probability $< \frac{1}{2}$, but $\mathcal{B}$ incorrectly outputs Yes with probability $< \frac{1}{2^m} << \frac{1}{2}$. By repeating the experiment, we have "amplified" the probability of success.

Questions?