Probability and Random Variables

An **experiment** is procedure which yields one of a given set of possible outcomes.

The **sample space** of an experiment is the set of possible outcomes.

Consider the following example experiments.

Example 1: Flip a coin.

Our sample space is $S = \{Heads, Tails\}$

Example 2: Roll a single six-sided die.

Our sample space is $S = \{1, 2, 3, 4, 5, 6\}$

Example 3: Flip a coin three times.

(Let 'H' denote "heads" and let 'T' denote "tails") $\nearrow 2 \cdot 2 \cdot 2$

What is the sample space of this experiment? $2^3 = 8$ possible

$S = \{HHH, HHT, HTH, THH$
$TTT, TTH, THT, HTT\}$

An **event** E is a subset of the sample space S ($E \subseteq S$).

cardinality ↳ subset

The probability of $p(E) = \frac{|E|}{|S|}$ (Assuming uniform probability distribution)

→ each outcome has probability of $\frac{1}{|S|}$

With this definition $p(s) = \frac{1}{|S|}$ for all $s \in S$

// s is outcome

Example 4: Roll a die gives us the sample space $S = \{1, 2, 3, 4, 5, 6\}$

roll a 1 ↑ roll a 2 ↑

$p(1) = \frac{1}{|S|} = \frac{1}{6}$; $P(2) = \frac{1}{|S|} = \frac{1}{6}$

Let E denote the event that the die roll results in a prime number. // Events are collections of outcomes

$E = \{2, 3, 5\}$

$p(E) = \frac{|E|}{|S|} = \frac{3}{6} = \frac{1}{2}$

For probabilities in general the following holds:

certain event

(1) $0 \leq p(s) \leq 1$ for all $s \in S$

(2) $\sum_{s \in S} p(s) = 1$ – 100% $\Rightarrow$ sum of all possible outcomes in our set

(3) $p(E) = \sum_{s \in E} p(s)$

↳ $P(E)$ = sum of all possible outcomes in our event

Random Variables

A random variable (RV) is a real-valued function defined on the sample space S of an experiment. Let X be a RV defined on the sample space S. If $s \in S$, then $X(s)$ is a real number. takes outcome & spits out a number

Example 1:

Consider the experiment of 3 fair coin flips. Let X be a RV denoting the number of heads observed. X is looking for heads

$X(HHH) = 3$ $X(HTH) = 2$

$X(HHT) = 2$ $X(THT) = 1$

$X(TTH) = 1$ $X(THH) = 2$

$X(TTT) = 0$ $X(HTT) = 1$

//We want to think of the EV of these RVs

The **expected value** (also called the expectation or mean) of the random variable X on the sample space S is equal to RV = X

$$E(X) = \sum_{s \in S} p(s)X(s)$$

Not an event

For the above coin flipping example what is $E(X)$?

$E(X) = (p(HHH) \cdot X(HHH)) + (p(HHT) \cdot X(HHT)) + (p(HTH) \cdot X(HTH))$
$+ \ldots + p(TTT) \cdot X(TTT))$

//8 possible outcomes => all $p(s) = \frac{1}{8}$

$= (\frac{1}{8} \cdot 3) + (\frac{1}{8} \cdot 2) + (\frac{1}{8} \cdot 2) + \ldots + (\frac{1}{8} \cdot 0)$

$= \frac{12}{8} = 1.5$ heads ; $P(X = 1.5 H) = 0$ average

//can't get 1/2 a H

another way of calculating EV

from example above

We can also denote the expected value as $X(S) = \{3,2,1,0\}$

$$E(X) = \sum_{r \in X(S)} p(X = r)r$$

For the above coin flipping example what is $E(X)$ using this new definition?

$$E(X) = \sum_{r \in \{3,2,1,0\}} P(X = r)r$$

// fewer cases, but a/a harder to compute
↳ saves some space

$$= P(X=3)\cdot 3 + P(X=2)\cdot 2 + P(X=1)\cdot 1 + P(X=0)\cdot 0$$

$$= \frac{1}{8}\cdot 3 + \frac{3}{8}\cdot 2 + \frac{3}{8}\cdot 1 + 0 = \frac{12}{8} = 1.5 \text{ head}$$

Linearity of Expectation

If X and Y are random variables on S then

$$E(X + Y) = E(X) + E(Y)$$

Proof:

$$\Rightarrow E(X+Y) = \sum_{s \in S} (X(s) + Y(s)) \cdot P(s)$$

$$= \sum_{s \in S} (X(s)\cdot P(s) + Y(s)\cdot P(s))$$

$$= \sum_{s \in S} X(s)\cdot P(s) + \sum_{s \in S} Y(s)\cdot P(s)$$

$$= E(X) + E(Y)$$

Linearity of Expectation (cont.)

And in general, if X_i, $i = 1, 2, \ldots, n$ where n is a positive integer, are random variables on S then

$$E(X_1+X_2+\ldots+X_n) = E(X_1)+E(X_2)+\ldots+E(X_n)$$

Alternatively we can state this as:

$$E(\sum_{i=1}^{n} X_i) = \sum_{i=1}^{n} E(X_i)$$

Linearity of Expectation Example

Algorithm 1 int hireAssistant(int n)

```
best = 0;
i = 1;
while i ≤ n do
    if candidate i is better than candidate best then
        best=i;
        "hire candidate" //Very costly!   = C_H
    end if
    i = i + 1;
end while
```

$n = \#$ of applicants

$X = \#$ of people hired

runtime $= O(n + X \cdot C_H)$

Worst case: $X = n \Rightarrow O(n + \underbrace{n \cdot c_H}_{\text{dominant}}) = O(n \cdot c_H)$

=> Candidates sorted in ascending order based on skill. Must cycle through every candidate

Best case: $X = 1 \Rightarrow O(n + 1 \cdot c_H) = O(n + c_H)$

=> First candidate is best, no one better only pay high cost once

What about the average case?

Let X be the random variable denoting the number of candidates hired. Specifically:

$X(s)$ = # people hired when the candidates arrive in order s

indicator R.V.

Define $X_i(s) = \begin{cases} 1 \text{ (true)} & \text{if candidate } \# i \text{ is hired} \\ 0 \text{ (false)} & \text{otherwise} \end{cases}$

Clearly, $X(s) = \sum_{i=1}^{n} X_i(s) = 1+0+1+1+1+0+0 \ldots + n = X$

people we hire

We want to compute $E(X)$. From the Linearity of Expectation we know:

$$E(X) = E(\sum_{i=1}^{n} X_i) = \boxed{\sum_{i=1}^{n} E(X_i)}$$ looking for this

Best
A B C $\Rightarrow 3!$ ways to arrange candidates $= 6$

$\Rightarrow 2!$ ways to arrange everyone but best candidate

$= \frac{2!}{3!} = \frac{2}{6} = 1/3$

$i = 1$, $i = 2$, $i = 3$, $i = 4$, $i = n$

$\frac{3!}{4!} = \frac{1}{4}$

Chance hired

100 % $(\frac{1}{1})$
50 % $(1/2)$
1/3 %
1/4 %
1/n %

$\frac{(n-1)!}{n} = \frac{1}{n}$

What is $E(X_i)$?

$\Rightarrow 1 \cdot \frac{1}{i}$

$E(X_i) = 1 \cdot (P(X_i) = 1) + 0 \cdot (P(X_i = 0)$ $= 0$

$= \frac{1}{i} + 0$

So, $E(X) = \sum_{i=1}^{n} E(X_i) = \sum_{i=1}^{n} \frac{1}{i}$

We learned earlier that $\sum_{i=1}^{n} \frac{1}{i} \in O(\log n)$ (n-th harmonic number).

We can thus conclude that the average case runtime of hireAssistant is in $O(n + C_h \log n)$.

Deterministic Algorithms

A deterministic algorithm is one which, given the same input, will always follow the same sequence of logic to compute the output. Because it follows the same logic, it will also take the same runtime.

The algorithms we have seen so far in the class have all been deterministic.

Let's revisit insertion sort as an example of a deterministic algorithm.

Algorithm 2 int[] insertionSort(int $A[1 \ldots n]$)

```
j = 2;
while j <= n do
    key = A[j];
    i = j - 1;
    while (i > 0) and (A[i] > key) do
        A[i + 1] = A[i];
        i--;
    end while
    A[i + 1] = key;
    j++;
end while
return A;
```

Runtime for deterministic algorithms with input size n:

(1) Best case runtime
$\Rightarrow$ Attained by (at least) one specific input of size n

(2) Worst case runtime
$\Rightarrow$ Attained by (at least) one specific input of size n

(3) Average case runtime
$\Rightarrow$ Averaged over all possible inputs of size n

What disadvantages with deterministic algorithm such as the one above?

(1) .

(2) .

Alternative: Use a randomized algorithm