Discussion Section #1

April 8, 2025

1. In this exercise, we show that the exponential distribution is the continuous-time analogue of the geometric distribution and can be approximated by geometric distributions.

   a) What is the geometric distribution? What does it model?

Answer:

For a random variable X with geometric distribution, characterized by the success probability p, it holds that $\Pr[X > n] = (1-p)^n$, i.e., the CDF (cumulative distribution function) of the geometric distribution is given by

$$\Pr[X \leq n] = 1 - (1-p)^n \tag{1}$$

Here, if p represents the probability of winning the lottery, X denotes the number of attempts to win the lottery.

---

   b) What is the exponential distribution? What does it mean for the exponential distribution to be memoryless?

Answer:

For a random variable X with exponential distribution, characterized by the rate λ, the CDF (cumulative distribution function) of the geometric distribution is given by

$$\Pr[X \leq t] = 1 - e^{-\lambda t} \tag{2}$$

A random variable X has a memoryless distribution, if $\Pr[X > t+s | X > s] = \Pr[X > t]$. This holds for the exponential distribution; since

$$\begin{aligned}\Pr[X > t+s | X > s] &= \frac{\Pr[X > t+s \text{ and } X > s]}{\Pr[X > s]} \\ &= \frac{\Pr[X > t+s]}{\Pr[X > s]} = \frac{e^{-\lambda(t+s)}}{e^{-\lambda s}} = e^{-\lambda t} = \Pr[X > t]\end{aligned}$$

Intuitively, previous failures to win the lottery or previous failures for an event to happen does not affect future trials or further wait time.

---

c) Approximate the exponential distribution via the geometric distributions.

Answer:

Suppose $p = \lambda\delta$, where λ is a positive constant, and δ is a small time interval. We are now interested in the probability of winning the lottery within a certain time window $[0, t]$, and each trial takes time δ. Let $\tilde{X}$ denote the time when we win the lottery. Then, we can write

$$\Pr[X \leq \lfloor t/\delta \rfloor] \leq \Pr[\tilde{X} \leq t] \leq \Pr[X \leq \lceil t/\delta \rceil]$$

Finally, let's take $\delta \to 0$. Define $k = \lfloor t/\delta \rfloor$. By equation (2) and $p = \lambda\delta$,

$$\lim_{\delta \to 0} \Pr[X \leq \lfloor t/\delta \rfloor] = \lim_{\delta \to 0} 1 - (1 - \lambda\delta)^{\lfloor t/\delta \rfloor} = 1 - e^{-\lambda t}.$$

Similarly,

$$\lim_{\delta \to 0} \Pr[X \leq \lceil t/\delta \rceil] \leq \lim_{\delta \to 0} 1 - (1 - \lambda\delta)^{1+\lceil t/\delta \rceil} = 1 - e^{-\lambda t}.$$

Hence, we recover the CDF of the exponential distribution, namely $F(t) = 1 - e^{-\lambda t}$.

---

d) **Numerical example for approximating the exponential distribution via geometric distribution:**

Consider an exponential distribution with rate parameter $\lambda = 0.1$ successes per hour. The probability of failure within the first $k = 5$ hours is $\Pr[\tilde{X} \leq k] = 1 - e^{-\lambda k} = 1 - e^{-0.1 \times 5} \approx 0.393$.

Now, for let's divide the time into discrete 1 hour-long intervals. Let p be the probability of success within one hour. Approximately $p = \lambda\delta = 0.1 \times 1 = 0.1$. Then, the probability of success within the first five hours becomes $\Pr[X \leq k] = 1 - (1 - p)^k = 1 - (1 - 0.1)^5 \approx 0.410$.

2. Construct a correct but insecure signature scheme. Write the full implementation of your signature scheme in pseudocode. Prove its correctness and show why it is insecure.

Answer:

The simplest example of a correct but insecure signature scheme is when the verification algorithm V is defined to always return 1. The choice of G and S won't matter. This scheme is correct trivially by the definition of correctness.

To show its insecurity, consider an adversary that returns any message-signature combination without use of the signing oracle. The probability such an adversary wins the forgery game will be 1.

Algorithm 1 Generation Algorithm G. Here, κ is the security parameter.

```
function G
    sk ← 1^κ
    pk ← 1^κ
    return (sk, pk)
end function
```

Algorithm 2 Signing Algorithm S

```
function S(sk, m)
    return sk
end function
```

Algorithm 3 Verification Algorithm V

```
function V(pk, m, σ)
    return true
end function
```

Correctness Proof: A digital signature scheme is said to be correct if for all key pairs $(sk, pk) \leftarrow \mathsf{G}()$ and all messages m, we have:

$$\mathsf{V}(pk, m, \mathsf{S}(sk, m)) = \texttt{true}$$

In this construction, $\mathsf{S}(sk, m)$ always returns sk, and V always returns `true`, regardless of the inputs. Thus, the correctness condition is trivially satisfied.

Insecurity: To demonstrate insecurity, consider an adversary $\mathcal{A}$ that, without querying the signing oracle, outputs any arbitrary message-signature pair (m^*, σ^*). Since V always returns `true`, we have:

$$\Pr[\mathsf{V}(pk, m^*, \sigma^*) = \texttt{true}] = 1$$

Therefore, the signature scheme is not secure.

---

3. Let us consider some simple solutions to the double spend problem and see if they work.

a) Does canceling both double spend transactions in retrospect work?

Answer:

No. Consider the scenario, where an adversary first buys an item in one transaction, and then goes on to double spend the bitcoin used in that transaction, but only after every honest node receives the first transaction. When both transactions are later canceled, the adversary still owns that item, so the seller loses money in that case.

---

b) Does canceling only the second transaction in the example above work? Stated more formally, every node considers only the first double spend transaction to be valid. After all, the merchant then keeps the bitcoin in the example above.

Answer:

No. Different nodes may receive the transactions in different orders if they are broadcast within the time span Δ, where Δ is the network delay (see Figure 1). Then, honest nodes will disagree about the order of transactions, as well as their validity, which implies that their ledgers also disagree in this case.

---

c) The third idea is to set up a time window. If both transactions are received within the window in the view of a node, then the node cancels both transactions. Otherwise, it accepts the first transaction and cancels the second transaction that lies outside of the time window. Ideally, imposing a time window on transactions ensures that either all honest nodes receive the same transaction first, or cancel both transactions.

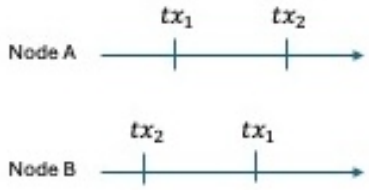

Figure 1: Arrows denote the direction of time. Points indicated by the transactions denote the times they were received by the respective nodes. Node A receives tx_1 first, whereas node B receives tx_2 first.

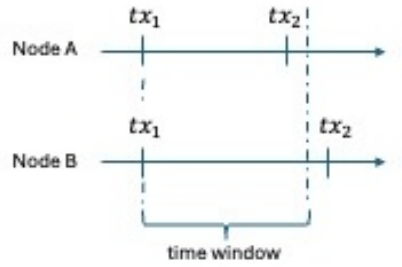

Figure 2: Node A receives tx_2 within the time window and cancels both transactions; whereas node B receives tx_2 outide the time window and accepts tx_1.

Answer:

No. Suppose our window starts at the time the first transaction is received. Consider the scenario, where all honest nodes first receive some transaction tx_1 at time t_1, and the adversary broadcasts tx_2 at some time t right before $t_1 + T_{\text{window}}$, i.e., right before the end of the time window; such that $t + \Delta > t_1 + T_{\text{window}}$, i.e., $t + \Delta$ lies outside of the time window (see Figure 2). In this case, some nodes would receive tx_2 within the time window, and the other nodes would receive it outside of the window. Hence, there will be disagreement once again.
