

Online algorithms

ICTS-RRI Maths Circle, Bengaluru

11, 25 April 2026, 10:00 am to 1:00 pm

Problems

In several real-life situations, we need to make a decision at some point based on the information we have then, but the consequences (in the form of costs or rewards) are determined by later events. In such situations, we cannot match what we could have achieved if we knew the future, but we would still like to devise strategies that make the best of the information we have at hand. Such strategies are called *online algorithm*; they are *online* because they make irrevocable commitments as they go along. We will learn about online algorithms using through some problems.



Picture: ChatGPT

1. (Bicycle rentals) Imagine you go to a resort for a long vacation. Each morning, you wake up and check the weather. If the weather is good, you can go bicycling, otherwise, if the weather turns bad, you have to cut short your vacation and come back home. Now to bicycle on a particular day, you have two choices: (i) rent a bicycle for that day for 1 rupee (it is unrealistically cheap, but the introductory example ought to be kept simple), or (ii) buy a bicycle for b rupees (then you own the bicycle and can use it for the rest of your trip, but you can't return them and get a refund if the weather turns bad the next day). Should you rent or should you buy? Maybe, you want to rent for few days, and then buy—how long should you wait before deciding to buy?
 - (a) Suppose the weather reports were 100% accurate, and you learn that there will k days of good weather (so you need to a bicycle for k days). What would be the best (optimal) strategy? Let us call the cost of this strategy $\text{Opt}(k)$.
 - (b) Now suppose you have no predictions for the weather available. A strategy might look like this: you decide to keep renting until day t (paying 1 rupee per day, total $t - 1$ rupees); now if the weather is still good on day t , you decide to spend b rupees and buy a bicycle on day t . If the good weather does not even last t days, you go home as soon as the weather turns bad. Let $C(t, k)$ be the cost incurred under this strategy, when there are k days of good weather. We would like to choose a value t so that for all $k \geq 1$, the ratio $C(t, k)/\text{Opt}(k)$ is as small as possible. E.g., if $b = 10$, $t = 5$ and $k = 12$, then $C(t, k) = 14$ and $\text{Opt}(k) = 5$, and the ratio is $14/5 = 2.8$ (you end up paying 2.8 times what

someone who could predict the future would have paid); we need to adjust t based on b (without knowing k in advance) so that this ratio is kept small no matter what k is.

(c) We will discuss what happens if you use a randomized strategy, where you determine the value of t based on the outcome of coin tosses.

2. (Prefix-free encoding) Imagine an infinite binary tree. We have a root, and every node has two children, a left child and a right child.

(a) Suppose the numbers $1, 2, \dots, n$ are placed at various nodes of the tree in such a way that no number is on the path from the root to another number. Suppose 1 is located at distance ℓ_1 from the root, 2 is located at distance ℓ_2 from the root, and so on. Show that

$$2^{-\ell_1} + 2^{-\ell_2} + \dots + 2^{-\ell_n} \leq 1. \quad (1)$$

[Hint: Try out some examples.]

(b) Now suppose ℓ_1, ℓ_2, \dots satisfy eq. (1). We will be given these numbers one by one: as soon as we receive ℓ_1 we have to place the number 1 on some node of the tree at distance ℓ_1 from the root, then we get ℓ_2 , and we must place 2 on a node of the tree at distance ℓ_2 from the root, and so on. At no point must a number appear on the path from the root to another number. Show a strategy to solve this *online* problem—it is online because as soon as ℓ_i is received, the location of i is fixed once and for all.

3. (Online caching) You are a radio jockey. You have a collection of n songs, all of the same size, which you have stored on the cloud. Your computer has memory that can hold k of the songs at a time. Whenever you receive a request for a song that is available on your computer, you can play it immediately. Otherwise, you must download the song from the cloud and then play it. If the computer already has k previously downloaded songs in its memory, you must replace one of the songs with the new song. Our goal is devise a strategy that minimizes the number of downloads.

(a) Suppose you know in advance all the requests that you are going to receive that evening. We need a strategy to decide which song in the memory to replace by the new song whenever the computer's memory is full. Show that the strategy that replaces the song that will be requested *farthest-in-the-future* with the new song is optimal, that is it makes the minimum number of downloads. [Hint: This might be tricky, but give it a try.]

- (b) A popular online strategy (when the future requests are not known) is to replace the *least recently used/requested* (LRU) song from the computer's memory to accommodate the new song (when the computer's memory is full). That is, when we need to evict a song in memory to accommodate the new song, we identify the song sitting in the computer's memory that has not been played for the longest time, and replace it with the song that has just been requested. Construct a (long) sequence of requests where the LRU strategy has almost k times more downloads than the optimum strategy.
4. (The theatre seating problem) Consider the following situation. You book a seat for a play, for which you receive your seat number on your phone. Just as you enter the theatre, your phone runs out of charge. You don't remember your seat number. There are n vacant seats, so you take one of the empty seats. Now, when the person who was assigned that seat shows up demands their seat, you get up (with due apologies) and occupy another empty seat. Assume all seats were booked and everybody shows up in the end.
- (a) In the worst case, how many times might you be asked to give up your seat before you finally get to your designated seat.
- (b) If at each step you picked one of the available seats *uniformly at random* (with equal probability), then *on average* how many times will you have to give up your seat? [Hint: not straightforward, but try some examples.]
- (c) Suppose you were the first person to enter the theatre and you occupied a uniformly chosen random seat from the n seats available ($n \geq 2$). Then on, whenever a person finds that their seat is already occupied, they too occupy a random seat; if their seat is available, of course, they take their designated seat. What is the probability that the last person to enter the theatre finds their designated seat occupied. [Hint: try some examples with $n = 2, 3, 4$, and try to guess the pattern.]