Online algorithms

ICTS-RRI Maths Circle, Bengaluru

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

Problems

Picture: ChatGPT

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.

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 a bicycle for k days). What would be be the best (optimal) strategy? Let us call the cost of this strategy $\mathsf{Opt}(k)$.

   (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 $C(t,k)/\mathsf{Opt}(k)$ is as small as possible. E.g., if $b = 10$, $t = 5$ and $k = 12$, then $C(t,k) = 14$ and $\mathsf{Opt}(k) = 5$, 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, ..., 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

$$2^{-\ell_1} + 2^{-\ell_2} + \cdots + 2^{-\ell_n} \leq 1. \tag{1}$$

[Hint: Try out some examples.]

(b) Now suppose $\ell_1, \ell_2, \ldots$ 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.]

Summary of the 11th April session

We started with the bicycle rental problem. Everybody understood the dilemma introduced by the *online* nature of the problem, and quickly figured out that for k days of good weather, the optimum strategy was to buy the bicycle (on the first day) if and only if $k \geq b$, that is, $\mathsf{Opt}(k) = \min(b, k)$. The second part of the question was not

that obvious. I wrote down

$$\mathsf{ratio}(t) = \max_k \frac{C(t,k)}{\mathsf{Opt}(k)},$$

provoking the question: having decided to buy the bicycle on the t-th day of good weather, what would be the worst ratio we could suffer?

Sam suggested that we make a table, an excellent idea that I should have taken up immediately.

$b = 5$

Each entry of the table has the form c/d, where $c = C(t,k)$ and $d = \mathsf{Opt}(k)$

	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$	$k=8$	$k=9$	$k=10$
$t=1$	$\boxed{5/1}$	$5/2$	$5/3$	$5/4$	$5/5$	$5/5$	$5/5$	$5/5$	$5/5$	$5/5$
$t=2$	$1/1$	$\boxed{6/2}$	$6/3$	$6/4$	$6/5$	$6/5$	$6/5$	$6/5$	$6/5$	$6/5$
$t=3$	$1/1$	$2/2$	$\boxed{7/3}$	$7/4$	$7/5$	$7/5$	$7/5$	$7/5$	$7/5$	$7/5$
$t=4$	$1/1$	$2/2$	$3/3$	$\boxed{8/4}$	$8/5$	$8/5$	$8/5$	$8/5$	$8/5$	$8/5$
$t=5$	$1/1$	$2/2$	$3/3$	$4/4$	$\boxed{9/5}$	$9/5$	$9/5$	$9/5$	$9/5$	$9/5$
$t=6$	$1/1$	$2/2$	$3/3$	$4/4$	$5/5$	$\boxed{10/5}$	$10/5$	$10/5$	$10/5$	$10/5$
$t=7$	$1/1$	$2/2$	$3/3$	$4/4$	$5/5$	$6/5$	$\boxed{11/5}$	$11/5$	$11/5$	$11/5$
$t=8$										
$t=9$										
$t=10$	$1/1$	$2/2$	$3/3$	$4/4$	$5/5$	$6/5$	$7/5$	$8/5$	$9/5$	$14/5$

Based on our observations, we can write the following.

Claim 1. *Fix a value of $b \geq 1$.*

Justify Claim 1.

(a) For every t, the ratio $C(t,k)/\mathsf{Opt}(k)$ is maximum when $k = t$.

(b) $\mathsf{ratio}(t) = (2t-1)/\min(b,t)$.

Now, it only remains to determine the value of t, for which $\mathsf{ratio}(t)$ is minimum. Complete the following

Claim 2. *For every $b \geq 1$, $\mathsf{ratio}(t)$ is minimum when $t = b$, so*

Justify Claim 2.

$$\min_t \max_k \frac{C(t,k)}{\mathsf{Opt}(t)} = \min_t \ \mathsf{ratio}(t) = 2 - \frac{1}{b}.$$

Randomized strategies: I pointed out that we have only considered *deterministic* (or *pure*) strategies, where we need to settle on single value of t in advance. I suggested we consider what happens when we decide to buy on day $b/2$ (assume b is even) with probability p, and buy on day b with probability $1-p$. Such strategies that hedge between various possibilities are called *mixed* strategies. The cost we incur is not fixed: it depends on the random choice we make. We chose to compute how well the strategy does in expectation (or on average). The *expected* cost is $pC(b/2,k) + (1-p)C(b,k)$, and to find the best p, we consider

$$\min_{p\in[0,1]} \max_k \frac{pC(b/2,k) + (1-p)C(b,k)}{\mathsf{Opt}(k)}.$$

If we set $p = 1$, then we are reduced the deterministic strategy with $t = b/2$, and the expected ratio is $\mathsf{ratio}(b/2) = (b/2 + b - 1)/(b/2) = 3 - 2/b$. This is worse than the old ratio $2 - 1/b$ that we would get if we set $p = 0$. Would an intermediate value of p give an expectation better than $2 - 1/b$? I suggested that it would, but did not work out the details. Just to give them an idea of the continuous version of the problem, I asked what would happen if the rent was not charged per day but by minute. So if we rented a bike for x minutes, we pay x rupees. We can buy the bicycle, as before, for b rupees. If we had a mixed strategy that decides to buy the bicycle at a random time with probability (density) $p(t)$, then the expected cost would be

$$\mathbb{E}[C(p,x)] = \int_0^x p(t)(t+b)\,\mathrm{d}t + x\int_x^\infty p(t)\,\mathrm{d}t.$$

The goal still remains the same, to determine the competitive ration; we wish to determine such that

$$\min_p \max_x \mathbb{E}[C(p,x)]/\min(x,b).$$

It turns out that the optimum choice for p is given by $p(t) = Ae^{t/b}$ with $A = 1/(k(b-1))$, for $0 \le t \le b$ and $p(t) = 0$ for $t > b$. The above *competitive ratio* for this strategy is $e/(e-1) = 1.581967....$ A similar randomized strategy also yields approximately the same ratio in the discrete setting we considered earlier.

We moved on to the online caching problem. Suppose the songs are numbered, and we knew exactly in what order they will be requested, e.g.,

$$1, 52, 3, 4, 19, 1, 3, 4, 1, 19, 52, 26, \ldots$$

Suppose the system in the studio can hold only one song at a time: then there is not much to do. Each song will have to be downloaded and played. What if we can only hold three songs at any time? We

A bored professor once wrote this parody of a *quwwali* from an old Hindi movie: watch (warning: very long!).

```
Yeh risk risk hai, risk risk
Yeh risk risk hai, risk

Share market me badnaam hai risk
Climate change ka toofaan hai risk
Yeh bekabu belagaam hai risk
Har danishwar har pir se anjaan hai
risk

Yeh risk risk hai, risk risk
Yeh risk risk hai, risk

Risk ka anumaan hi hai risk ka fix
Large deviations ka toh ghulaam hai
risk
Tayyar hai mera decision analytics
Ab na lagega mujhe risk ka jinx
Kyun ki meri strategy mein hai mix
mix
Porfolio bhi meri mixed

Yeh risk risk hai, risk risk

Strategy mein hai mix mix

Risk risk
Mix mix
Risk
Mix
```

Supurna emphasized that the important message in this is that randomized decisions are often superior to deterministic decisions while dealing with risks.

1 (miss)
1, 52 (miss)
1, 52, 3 (miss)
1, 4, 3 (miss)
1, 19, 3 (miss)
1, 19, 3 (hit)
1, 19, 3 (hit)
1, 19, 4 (miss)
1, 19, 4 (hit)
52, 19, 4 (miss)
26, 19, 4 (miss)

Figure 1: Farthest-in-the-future in action

will download and play the first three songs: 1, 52, 3. When song 4 is requested, we need to *evict* one of the songs. Which one? One possible strategy is to evict the song that is going to be requested *farthest in the future* (FIF). We tried to develop some intuition as to why this might be a good strategy. Is this the best strategy? I asked them to think this through. For a sequence s of requests let $\mathsf{Opt}(s)$ be the minimum number of evictions performed by a strategy (designed with full advance knowledge of the sequence).

FIF is an *offline* strategy. To implement it we need to know something about the future. What could be a reasonable strategy? If we had some understanding of the likelihood of various songs being requested, e.g., if the requests were drawn from a probability distribution, then perhaps, we keep the songs most likely to be requested and evict the least likely. In our model, we do not have any such knowledge. We wish to design a strategy, that is competitive no matter what the sequence of requests. We discussed the *least recently used* strategy: evict the song currently in the system that has remained unused the longest. I wanted to lead up to the following claim.

Claim 3. *Suppose the system can hold k songs at a time.*

(a) There is a sequence for which LRU needs to download the song each time a song requested.

(b) There is a sequence of requests where $\mathsf{LRU}(s)/\mathsf{Opt}(s)$ can be made arbitrarily close to k, that is, $k - \epsilon$ for all $\epsilon > 0$.

(c) For all sequences of requests s, we have $\mathsf{LRU}(s)/\mathsf{Opt}(s) \leq k$.

How many misses does the LRU strategy suffer on the round robin sequence? How many misses does the FIF suffer?

We considered the *round robin* sequence:

$$1, 2, 3, \ldots, k, k+1, 1, 2, 3, \ldots, k-1, k, k+1, \ldots . (r \text{ times})$$

For r large, it appeared that LRU suffers approximately k-times the number of misses compared to FIF. Could LRU do even worse on some other sequence? I suggested we focus on the requests that caused LRU to evict some song.

True or false: Between two evictions of the same song by LRU there must be k evictions of some other songs.

Another strategy was suggested: evict the *most recently used* song; a similar strategy in the computer science literature is call last-in first-out (LIFO). This strategy could end up having to download a song each time a request is received. But then, so does LRU! How bad can LIFO be in comparison with the optimal strategy? We will discuss another popular strategy is *first in first out* (FIFO): evict the song that was has resided in the system the longest without considering whether it has been used since. Are LRU and FIFO different strategies? How competitive is FIFO? We did not consider randomness, . . . next time.

As we prepared to leave for the day, I reminded everybody about problem 4, the theatre seating problem.