Khoury College of Computer Sciences
Northeastern University
CS 7870: Seminar in Theoretical Computer Science

Fall 2023
1 December 2023
Scribe: Shashank Manjunath

Lecture 22 Outline:

- Online Algorithms
- Randomized Algorithm for Ski Rental
- Lower Bound for Randomized Ski Rental
- Online Covering Problems

We continue our study of online algorithms with a randomized algorithm for ski rental, a lower bound for randomized ski rental illustrating a general approach, and then close with introducing online covering problems. Some of the material is drawn from notes by Viswanathan Nagarajan [Nag21].

1 Online Algorithms

We begin with an overview of frameworks for online optimization. There are two main frameworks for online optimization: online learning and online algorithms. In online learning, we calculate Regret, which is the optimal static cost (e.g., the cost of a solution that is fixed, such as picking a single expert) subtracted from the online loss incurred by a given online algorithm. We typically try to analyze Regret as a function of T, the number of rounds we run our algorithm for, and the goal is to achieve regret sublinear in T. In online algorithms, we analyze the competitive ratio, given as follows:

$$\text{Competitive Ratio} = \max_{\sigma} \frac{\text{Alg}(\sigma)}{\text{OPT}(\sigma)}$$

Note that in the regret framework, the "optimal" being compared with is a static solution which is a fixed policy for the whole sequence, while in the competitive analysis framework, the optimal offline solution is the best possible for the given sequence of inputs. On the other hand, regret is an additive guarantee (we are taking the difference between the online algorithm and the static optimal), while competitive ratio is a multiplicative guarantee.

1.1 Randomized Algorithms for Ski Rental

Consider the ski rental problem. Consider rounds $t = 1, \cdots, T$. At each step t, we are asked to decide whether to rent skis or buy skis. Renting skis incurs cost 1, and buying skis incurs cost B. In the previous lecture, we showed that the best deterministic competitive ratio achievable for ski rental is $2 - 1/B$. In particular, we considered the following simple deterministic algorithm: for any length sequence to rent for $B - 1$ steps, then buy on the Bth step. This yields a competitive ratio of $2 - 1/B$. Indeed, no deterministic algorithm can attain a competitive ratio of less than $2 - 1/B$.

	I_1	I_2	I_3	I_4	I_5	$\cdots$	I_{B-1}	I_B	I_∞
Alg_0	$B/1$	$B/2$	$B/3$	$B/4$	$B/5$	$\cdots$	$B/(B-1)$	B/B	B/B
Alg_1	$1/1$	$(B+1)/2$	$(B+1)/3$	$(B+1)/4$	$(B+1)/5$	$\cdots$	$(B+1)/(B-1)$	$(B+1)/B$	$(B+1)/B$
Alg_2	$1/1$	$2/2$	$(B+2)/3$	$(B+2)/4$	$(B+2)/5$	$\cdots$	$(B+2)/(B-1)$	$(B+2)/B$	$(B+2)/B$
Alg_3	$1/1$	$2/2$	$3/3$	$(B+3)/4$	$(B+3)/5$	$\cdots$	$(B+3)/(B-1)$	$(B+3)/B$	$(B+3)/B$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ldots$	$\vdots$	$\vdots$	$\vdots$
Alg_{B-1}	$1/1$	$2/2$	$3/3$	$4/4$	$5/5$	$\cdots$	$(B-1)/(B-1)$	$(2B-1)/B$	$(2B-1)/B$

Table 1: Comparison of determinisitic algorithms and sequence lengths

We now present a randomized algorithm for ski rentals. The competitive ratio analyzes the price of uncertainty. In a randomized algorithm, we provide a probability distribution over deterministic algorithms, i.e. with probability ρ_i we pick Alg_i. Since we must have a probability distribution, we require that $\sum_i \rho_i = 1$. This provides a competitive ratio as follows:

$$\text{Competitive Ratio} = \max_\sigma \mathbb{E}\left[\frac{\text{Alg}(\sigma)}{\text{Opt}(\sigma)}\right]$$

For analyzing randomized algorithms, it is common to assume an *oblivious adversary*, or an adversary that does not know the random choices of algorithm. Consider the number of possible instances, and denote by I_j all sequences of length j. We present potential competitive ratios associated with instance lengths and algorithms in Table 1. We can analyze the expected cost of each instance as follows.

$$\mathbb{E}[I_1] = \rho_0 B + \rho_1 + \rho_2 + \cdots \tag{1}$$

$$\mathbb{E}[I_2] = \rho_0 \frac{B}{2} + \rho_1 \left(\frac{B+1}{2}\right) + \rho_2 + \cdots \tag{2}$$

$$\mathbb{E}[I_3] = \rho_0 \frac{B}{3} + \rho_1 \left(\frac{B+1}{3}\right) + \rho_2 \left(\frac{B+2}{3}\right) + \rho_3 + \cdots \tag{3}$$

$$\vdots$$

$$\mathbb{E}[I_B] = \rho_0 + \rho_1 \left(\frac{B+1}{B}\right) + \cdots + \rho_{B-1} \left(\frac{2B-1}{B}\right) \tag{4}$$

We can now express the selection of the best randomized algorithm as one obtained by the solution to the following linear program.

$$\begin{aligned} &\min c \text{ such that:} \\ &(1) \leq c \\ &(2) \leq c \\ &(3) \leq c \\ &\vdots \\ &(4) \leq c \\ &\sum_i \rho_i = 1 \end{aligned}$$

Here, c refers to the competitive ratio, which we want to minimize, and each of the constraints indicates that the expected cost of the algorithm on a given instance I_j is at most c times the optimum cost for that instance (as required). One potential solution to this linear program is to set all equations equal to c.

$$\begin{aligned}
\rho_0 B + \rho_1 + \rho_2 + \cdots \rho_{B-1} &= c \\
\rho_0 \frac{B}{2} + \rho_1 \left(\frac{B+1}{2}\right) + \rho_2 + \cdots &= c \\
\rho_0 \frac{B}{3} + \rho_1 \left(\frac{B+1}{3}\right) + \rho_2 \left(\frac{B+2}{3}\right) + \rho_3 + \cdots &= c \\
&\vdots \\
\rho_0 + \rho_1 \left(\frac{B+1}{B}\right) + \cdots + \rho_{B-1} \left(\frac{2B-1}{B}\right) &= c
\end{aligned}$$

Solving this system of equations, we obtain $\rho_i = \rho_0 \left(\frac{B}{B-1}\right)^i$. Furthermore, since the ρ_i form a probability distribution, we know that:

$$\sum_i \rho_i = 1$$

Therefore,

$$\begin{aligned}
\sum_{i=1}^{B-1} \rho_0 \left(\frac{B}{B-1}\right)^i &= 1 \\
\rho_0 \left(\frac{\left(\frac{B}{B-1}\right)^B - 1}{\frac{B}{B-1} - 1}\right) &= 1 \\
\implies \rho_0 &= \frac{1}{B-1} \cdot \frac{1}{\left(\frac{B}{B-1}\right)^B - 1}
\end{aligned}$$

Recall that we set the inequality constraints to be equalities. Therefore,

$$\begin{aligned}
c &= \rho_0(B-1) + \rho_0 + \rho_1 + \cdots + \rho_{B-1} = \rho_0(B-1) + 1 \\
c &= 1 + \frac{1}{\left(\frac{B}{B-1}\right)^B - 1} \\
c &\approx 1 + \frac{1}{e-1} \text{ for sufficiently large } B
\end{aligned}$$

Therefore, $c = \frac{e}{e-1}$ is the competitive ratio.

1.2 Lower Bound for Online Randomized Algorithm

Recall the linear program developed in Section 1.1. The randomized algorithm has ρ_i assigned to Alg_i. We rewrite this linear program as follows:

$$\forall j : \min c \text{ such that } -\sum_i \rho_i \frac{\text{Alg}_i(I_j)}{\text{Opt}(I_j)} + c \geq 0$$
$$\sum_i \rho_i = 1$$

Let μ_j denote each constraint, and r denote that $\sum_i \rho_i = 1$. We can write the dual problem as follows:

$$\max_r \text{ such that } \forall i : -\sum_j \mu_j \frac{\text{Alg}(I_j)}{\text{Opt}(I_j)} + r \leq 0$$
$$\sum_j \mu_j \leq 1$$

Note that μ_j is a distribution over instances, and we calculate the expected performance of Alg_i over all instances according to this distribution.

Lemma 1 (Yao's Lemma)**.** Consider an online problem Π. Suppose there is a distribution μ over instances such that every deterministic algorithm Alg has expected competitive ratio $\geq r$. Then there exists no randomized algorithm with competitive ratio less than r.

Proof. By linear program duality. □

Yao's lemma allows us to develop a lower bound for randomized algorithms by analyzing the expected performance of deterministic algorithms. The task for us is to select a suitable distribution over instances. It turns out that if we choose instance I_j with probability $\frac{1}{B}\left(1 - \frac{1}{B}\right)^j$, then every deterministic algorithm has expected competitive ratio at least $e/(e-1)$.

2 Online Covering Problems

For this problem, at the start, we have sets $S_1, \cdots, S_i, \cdots, S_n \subseteq \mathcal{U}$. We additionally have a cost c_i associated with each S_i. We assume that $c_i = 1$. In this problem, at each online step, an element of $i \in \mathcal{U}$ arrives. Our required action is to select a set S_i that covers the element i. All elements that have already arrived must be covered. A set that has been picked cannot be removed. The overall goal is to minimize the cost of the sets.

We present a potential algorithm using linear programming as follows:

$$\begin{aligned} \min \sum_i c_i x_i \text{ such that} \\ \sum_{i:j \in S_i} x_i \geq 1 \forall j \text{ arrived} \\ x_i \geq 0 \\ x_i \in 0, 1 \end{aligned}$$

Our approach for covering is to solve the linear relaxation of this linear program in an online manner, thus developing a competitive online algorithm for the fractional problems, then apply randomized rounding to get a randomized online algorithm for the (integral) covering problem.

References

[Nag21] Viswanath Nagarajan. Introduction to online optimization. Lecture notes, available from `https://viswa.engin.umich.edu/wp-content/uploads/sites/169/2021/02/online-basics.pdf`, 2021.