COMPSCI 530: Design and Analysis of Algorithms November 19, 2013

Lecture 23: Competitive ratio: rent or buy problems

Lecturer: Debmalya Panigrahi *Scribe: Hangjie Ji*

1 Overview

In this lecture, we focus on a kind of problem where the optimization needs to be performed without the input in advance. This kind of algorithm is called online algorithm. To evaluate the performance of the algorithm, we introduce the concept of ”competitive ratio”. Then we illustrate the concepts through the ski-rental problem and the online paging/caching problem. Finally we discuss the randomized online algorithm for the randomized ski-rental problem.

2 Online Algorithms

An **online algorithm** is one that processes its input which arrives in phases, without having the entire input information available from the beginning. On the contrary, an **offline algorithm** refers to the algorithm that has access to the entire input from the start and outputs its decision in this setting. Because of the lack of the whole input, the decision that an online algorithm makes may turn out not to be optimal later. Thus the quality evaluation of online algorithms has focused on comparing the performance of an online algorithm and its corresponding offline algorithm for the same problem instance, which is exactly the idea of **competitive analysis**. Following this idea, we define the **competitive ratio** of the online algorithm as follows:

Definition 1.

$$\textit{Competitive ratio} = \max_{\textit{over input sequences}} \left\{ \frac{\textit{cost of online ALGO}}{\textit{cost of offline OPT}} \right\}$$

2.1 Ski-rental problem

In this subsection, we analyze the online algorithm for the ski-rental problem, which is also called the rent-or-buy problem.

Suppose we need skis to go skiing in the winter. Everyday we have two options: either renting the skis for cost \$1 per day, or paying \$ B to buy the skis and use it for the rest of the season. However, we do not know for how long the season will last, thus we cannot make our decision by comparing the total cost of the two options at the beginning of the season.

Let x be the length of the season. Suppose we buy skis on the (k+1)-th day, then

$$\text{online ALGO cost} = \begin{cases} B+k & \text{if } x \geq k \\ x & \text{if } x < k \end{cases}$$

and as the offline optimal algorithm knows the length of the season in advance, we have

$$\text{offline OPT} = \begin{cases} B & \text{if } x \geq B \\ x & \text{if } x < B. \end{cases}$$

We want to minimize the competitive ratio ρ which is defined by

$$\text{Competitive ratio } \rho = \max_{\text{over input sequences}} \{\frac{\text{cost of online ALGO}}{\text{cost of offline OPT}}\}.$$

If we let $k = B$, i.e. we buy skis on the (B+1)-th day, then

- If $x \leq B$, then $\rho = \frac{x}{x} = 1$;
- If $x > B$, then $\rho = \frac{B+B}{B} = 2$.

We will show that $\rho = 2$ is the best competitive ratio we can get by the following online lower bounds argument.

Let us consider a game between an adversary (ADV) and the algorithm (ALGO), in which ADV wants the lower bound of the competitive ratio to be as large as possible while ALGO wants the lower bound to be as small as possible. In each round of the game, ADV gives an input to the problem and ALGO provides its optimal solution based on the available input information at this stage. The game continues until ADV decides to stop the game, which means the season ends in the ski-rental problem.

Since ADV is in favor of the larger competitive ratio, once ALGO decides to buy the skis, ADV's strategy should be ending the season at once; otherwise, the competitive ratio will become smaller, since the cost of ALGO keeps fixed and the cost of offline OPT increases after ALGO buys the skis. Therefore if we continue the game for k rounds, i.e. ALGO buys the skis at the $k+1$-th day and ADV stops at once, then we have the competitive ratio $\mathcal{L}_k$ being

$$\mathcal{L}_k = \begin{cases} \frac{k+B}{k} & \text{if } k \leq B \\ \frac{k+B}{B} & \text{if } k > B. \end{cases}$$

Then the lower bound of the competitive ratio would be

$$\text{lower bound } \mathcal{L} = \min_k \mathcal{L}_k = 2,$$

and the minimum is attained when $k = B$.

2.2 Online paging/caching problem

Next we turn to the online paging problem, which is also called caching problem. To access data from some page of the main memory, we need to bring the page into a smaller section of fast memory called *cache*. When the requested page is already in the cache, then it can be served directly. Otherwise, we have a cache *miss*, which means we need to go in the main memory to bring the requested page and this will slow down the speed of fetching data.

Assume that the cache has k slots (capacity) and that the main memory has a capacity of N pages. Requests (a sequence of pages) are served from the cache and many result in cache misses. The objective of a online paging/caching algorithm is to evict pages from the cache in a good way that minimizes the number of cache misses. This algorithm is an online algorithm because generally we do not know what the requests will be from the beginning.

First we introduce the concepts of Least Recently Used (LRU) and *Round*.

Definition 2. *Least Recently Used : LRU removes the page which was used farthest in the past. That is to say, the page that has been in the cache for the longest time without being used will be evicted by LRU rule.*

Definition 3. *Round(k): the maximal sequence of requests that ask for exactly k pages.*

Remark 1. *Note that in this sequence, each page can be asked for more than once.*

Example 1. Let $k = 3$. Suppose the request sequence is $[1,2,1,3,3,2,1,4]$. Then the round is $[1,2,1,3,3,2,1]$. In this example, value of ALGO $\leq k+1$ and offline OPT ≥ 1, thus giving us

$$\frac{\text{value of ALGO}}{\text{offline OPT}} \leq k+1.$$

Theorem 1. *LRU has a competitive ratio of k.*

Proof. In general, we can divide a request sequence into M rounds. According to the LRU rule, at most k cache misses can happen in each round, as once a specific page is used in a round, it will not be evicted out of cache until the end of this round. Therefore

$$\text{value of ALGO } \leq Mk.$$

Since the cache has only k slots, the offline optimal algorithm must have at least one cache miss after serving k distinctive pages without cache misses. Therefore the optimal algorithm leads to at least one cache miss in each round and

$$\text{offline OPT} \geq M.$$

Hence the competitive ratio $\rho \leq k$.

Next we show that LRU has $\rho \geq k$ by proving the Theorem 2 via example construction. This finishes our proof. □

Theorem 2. *Paging has a lower bound of k.*

Proof. We construct an example for this lower bound. Let the main memory has a capacity of $n = k+1$ pages, where the cache has k slots. Consider an adversary (ADV) vs. algorithm (ALGO) game similar to the game for the ski-rental problem. ADV wants to get a greater lower bound of the competitive ratio while ALGO wants a smaller lower bound. Clearly the best strategy of ADV is to request the page that is not in algorithm cache. Then the number of cache misses in ALGO equals the number of requests, which is greater than $k+1$, while the offline optimal algorithm is to evict the page that is farthest into the future and the value of OPT is 1. This example gives the lower bound of k for the paging problem. □

3 Randomized Online Algorithms

Now we consider an extension for the online algorithms. The idea of randomized online algorithms is that we toss a coin in the algorithm to make decisions. In this case, in the Adversary vs. Algorithm analysis, instead of knowing the output of ALGO, ADV knows only the strategy of ALGO but not its output.

3.1 Randomized ski-rental problem

Under the same settings of the above ski-rental problem, we change our strategy to be buying skis on the (x+1)-th day with probability P_x, where $\sum_{x=0}^{\infty} P_x = 1$ and the events of buying skis on specific dates are mutually independent. Let k be the length of the winter season, then apparently the optimal offline solution

$OPT(k) = \min(B,k)$, i.e. buying the skis on the first day if $k \geq B$ and renting for the whole season otherwise. In the online algorithm, suppose we plan to buy the skis on the (x+1)-th day, then we need to pay \$ $(x+B)$ if $x < k$ and \$ k if $x \geq k$. Hence we have

$$ALGO(x,P_x) = k\sum_{x\geq k} P_x + \sum_{x<k}(x+B)P_x \approx k\int_k^\infty P_x dx + \int_0^k (x+B)P_x dx. \tag{1}$$

The goal of this problem is to minimize $\max_k \frac{ALGO(k,Px)}{OPT(k)}$ over all Px distributions s.t. $\int_0^\infty P_x dx = 1$.

Remark 2. *Note that the online algorithm has control over P_x but has no control over k.*

Fact 3. *The minimax problem can be solved when $\frac{ALGO(k,Px)}{OPT(k)}$ is independent of k. Using variational analysis, it can be shown that the minimax goal can be attained when the functional $\frac{ALGO(k,Px)}{OPT(k)}$ is uniform with respect to k.*

Then we solve for the optimal distribution P_x. Let $\alpha = \frac{ALGO(k,Px)}{OPT(k)}$, where α is independent of k. So with eqn (1) and $OPT(k) = \min(B,k)$, we have

$$\int_k^\infty P_x dx + \int_0^k (x+B)P_x dx = \alpha \min(B,k). \tag{2}$$

- Case $k \geq B$: Taking derivatives with respect to k of both sides of eqn (2) yields

$$-kP_k + \int_k^\infty P_x dx + (k+B)P_k = 0, \tag{3}$$

$$BP_k + \int_k^\infty P_x dx = 0. \tag{4}$$

  Since $B > 0$ and $P_x \geq 0$, we have $P_k = 0$.

- Case $k < B$: Taking derivatives with respect to k of both sides of eqn (2) yields

$$-kP_k + \int_k^\infty P_x dx + (k+B)P_k = \alpha, \tag{5}$$

$$BP_k + \int_k^\infty P_x dx = \alpha. \tag{6}$$

  Differentiating both sides with respect to k again gives us

$$B\frac{dP_k}{dk} - P_k = 0,$$

  Hence

$$P_k = Ae^{k/B}, \text{where A is a constant to be determined.} \tag{7}$$

Combining the both cases and using $\int_0^\infty P_x dx = 1$, we have

$$\int_0^\infty P_x dx = \int_0^B Ae^{k/B} dk = 1,$$

thus

$$A = \frac{1}{B(e-1)}$$

and the competitive ratio

$$\alpha = \frac{e}{e-1}.$$

4 Summary

This lecture introduces the concepts of (randomized) online algorithms via examples of (randomized) ski-rental problems and caching/paging problems with an emphasis on their competitive analysis .

References

[1] Online algorithm. Wikipedia contributors. *Wikipedia, The Free Encyclopedia.*. `http://en.wikipedia.org/wiki/Online_algorithm`. Accessed November 20, 2013.

[2] Shuchi Chawla, Lecture notes for an Advanced Algorithms course at WISC available at `http://pages.cs.wisc.edu/~shuchi/courses/787-F09/scribe-notes/lec14.pdf`. Accessed November 20, 2013.