infOnAlg-01a: Online Algorithms	**Lecture 01**	**Date: 24.10.2018**
Instructor: Sebastian Berndt		**Scribe: Sebastian Berndt**

1 Introduction

In this course, we will take at look at *online algorithms.* These algorithms do not have access to the complete input, but still need to produce a good solution (e. g. one with small costs or large profit). Some typical example where such a situation occurs can be found in logistics (e. g. an online shop putting its orders into a truck) or scheduling (e. g. the scheduler of the CPU does not know the next tasks to come).

2 Ski Rental

In the ski rental problem, we are given two integers $B, R \in \mathbb{N}$. You want to go skiing, but do not know the length of the season, i. e. how many days of snow you will have. You can either *r*ent skies for R euros per day or *b*uy them for B euros. Formally, we can model this problem as

Problem 1 (Ski Rental)

Given: *Binary string* $b = b_1, b_2, \ldots, b_n$ *with* $b_1 = b_2 = \ldots = b_t = 1$ *and* $b_{t+1} = b_{t+2} = \ldots = b_n = 0$, *two integers* $B, R \in \mathbb{N}$ *with* $R < B$.

Find: *Minimal costs* OPT *to go skiing.*

Fact 1 *The optimal offline costs are* $\text{OPT} = \min\{t \cdot R, B\}$ *if the season is* t *days long.*

Unfortunately, we do not know the length of the season in before. At day i, we only see the value of bit b_i indicating that there is still snow ($b_i = 1$) or that the season is over ($b_i = 0$). We will now take a look at two simple strategies:

- In the strategy $\mathsf{Strategy}_\infty$, we will always rent the skis. Clearly, the costs $c(\mathsf{Strategy}_\infty)$ of $\mathsf{Strategy}_\infty$ are always $t \cdot R$. If $\text{OPT} = B$, the ratio $\frac{t \cdot R}{B}$ might be arbitrary bad if the season is long enough.
- In the strategy $\mathsf{Strategy}_0$, we will always buy the skis on the first day. Clearly, the costs $c(\mathsf{Strategy}_0)$ of $\mathsf{Strategy}_0$ are always B. If $\text{OPT} = t \cdot R$, the ratio $\frac{B}{t \cdot R}$ might be arbitrary bad if the season is short enough (e. g. only a single day long).

We thus see that the ratio of the costs of both algorithms compared to the optimal costs might be arbitrary bad. We now want to design an online algorithm such that this ratio is bounded. Note that every sensible deterministic strategy can be expressed in the following way: "We rent for k days. If there is snow on day $k + 1$, buy." Let us denote this strategy as $\mathsf{Strategy}_k$. Note that $\mathsf{Strategy}_0$ and $\mathsf{Strategy}_\infty$ are special cases of this. In order to bound the ratio, we will now take a look at all possible cases that might occur:

Case 1: If $t \leq k$ and $t \cdot R \leq B$, we have $\text{OPT} = t \cdot R$ and $c(\mathsf{Strategy}_k) = t \cdot R$. Hence, the ratio $\frac{c(\mathsf{Strategy}_k)}{\text{OPT}} = \frac{t \cdot R}{t \cdot R}$ is always 1.

Case 2: If $t \leq k$ and $t \cdot R > B$, we have $\text{OPT} = B$ and $c(\mathsf{Strategy}_k) = t \cdot R$. Hence, we have the ratio $\frac{c(\mathsf{Strategy}_k)}{\text{OPT}} = \frac{t \cdot R}{B}$. As $t \leq k$, the worst case scenario here is $t = k$ and the ratio is then $\frac{k \cdot R}{B}$.

Case 3: If $t > k$ and $t \cdot R \leq B$, we have $\text{OPT} = t \cdot R$ and $c(\mathsf{Strategy}_k) = k \cdot R + B$. Hence, the ratio is $\frac{k \cdot R + B}{t \cdot R}$. As this ratio is monotonically decreasing in t and $t > k$, the worst case scenario here is $t = k + 1$ and the ratio is then $\frac{k \cdot R + B}{(k+1) \cdot R}$.

Case 4: If $t > k$ and $t \cdot R > B$, we have $\text{OPT} = B$ and $c(\mathsf{Strategy}_k) = k \cdot R + B$. Hence, the ratio is $\frac{k \cdot R + B}{B}$.

As we want to ensure that our algorithm is competitive in all of the above scenarios, we need to choose k such that the term

$$\max\{\frac{t \cdot R}{t \cdot R}, \frac{k \cdot R}{B}, \frac{k \cdot R + B}{(k+1) \cdot R}, \frac{k \cdot R + B}{B}\} \qquad (*)$$

is minimized. Note that the first term $\frac{t \cdot R}{t \cdot R}$ of (*) is always 1 and the second term $\frac{k \cdot R}{B}$ is always smaller than the fourth term $\frac{k \cdot R + B}{B}$. In order to minimize (*), it is thus sufficient to minimize

$$\max\{\frac{k \cdot R + B}{(k+1) \cdot R}, \frac{k \cdot R + B}{B}\}. \qquad (**)$$

As the first term is monotonically decreasing in k and the second term is monotonically increasing in k, the value at the point of intersection k^* minimizes (**). One can easily see that

$$\frac{k \cdot R + B}{(k+1) \cdot R} = \frac{k \cdot R + B}{B} \Leftrightarrow (k+1) \cdot R = B \Leftrightarrow k = \frac{B - R}{R} = \frac{B}{R} - 1.$$

Hence, the point of intersection k^* is $k^* = (B/R) - 1$. The value at this point is

$$\frac{k^* \cdot R + B}{B} = \frac{[(B/R) - 1] \cdot R + B}{B} = \frac{2B - R}{B} = 2 - (R/B).$$

If $k^* \notin \mathbb{N}$, we take a look at $\lceil k^* \rceil$ and $\lfloor k^* \rfloor$. Simple arithmetics shows that

$$\frac{\lceil k^* \rceil \cdot R + B}{B} \leq 2$$
$$\frac{\lfloor k^* \rfloor \cdot R + B}{B} \leq 2.$$

Hence, either $\mathsf{Strategy}_{\lceil k^* \rceil}$ or $\mathsf{Strategy}_{\lfloor k^* \rfloor}$ is optimal. We thus just proved the following result:

Lemma 1 *The costs of both strategies* $\mathsf{Strategy}_{\lceil k^* \rceil}$ *and* $\mathsf{Strategy}_{\lfloor k^* \rfloor}$ *are at most twice the costs of an optimal offline solution.*

Furthermore, as we made an exhausting case analysis, we also showed that no online algorithm can have lower costs:

Lemma 2 *No strategy* $\mathsf{Strategy}_k$ *has less costs than* $2 - (R/B)\,\text{OPT}$ *on all inputs.*

3 Competitive Analysis

We will now try to integrate our above reasoning into a more general framework. We only do this for minimization problems, where we try to minimize the costs of an algorithm. This can be simply adapted to maximization problems, where we try to maximize the profit of the algorithm.

In the *competitive analysis* of online algorithms, we compare the optimal offline value (typically denoted as OPT) with the costs of an online algorithm A that can not see in the future. For an input I of a corresponding problem, let $\text{OPT}(I)$ be the costs of an optimal offline algorithm and $\mathsf{A}(I)$ be the costs of the online algorithm A.

Definition 3 *The maximal ratio*

$$\max_{\text{instance } I} \{ \frac{\mathsf{A}(I)}{\text{OPT}(I)} \}$$

is the absolute competitive ratio *of* A*. If the absolute competitive ratio of* A *is at most* $\alpha \in \mathbb{R}_{\geq 1}$*, we say that* A *is* α-competitive.

Note that $\mathsf{A} \leq \alpha \, \text{OPT}(I)$ is an equivalent formulation to α-competitiveness. We can now restate Lemma 1 in this framework.

Lemma 4 *Both* $\mathsf{Strategy}_{\lceil k^* \rceil}$ *and* $\mathsf{Strategy}_{\lfloor k^* \rfloor}$ *are* 2*-competitive.*

Sometimes, the strict requirement of $\mathsf{A} \leq \alpha \, \text{OPT}(I)$ is too much to ask for, especially for small optima. A slightly weaker form is given by the *asymptotic competitive ratio.*

Definition 5 *If* $\mathsf{A} \leq \alpha \, \text{OPT}(I) + c$ *for a constant* c *(independent of the input* I*) holds for all inputs* I*, we say that* A *is* asymptotically α-competitive. *The smallest such value* α *is the* asymptotic competitive ratio *of* A.

The main lesson to be taken away is the following:

Competitive analysis gives worst-case guarantees against an unknown future.