Theoretical Computer Science 584 (2015) 42–52

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Non-additive two-option ski rental

Amir Levi, Boaz Patt-Shamir [1]

School of Electrical Engineering, Tel Aviv University, Tel Aviv 6997801, Israel

ARTICLE INFO

Article history:
Received 24 November 2013
Received in revised form 17 January 2015
Accepted 20 January 2015
Available online 23 January 2015

Keywords:
Competitive analysis
Ski rental
Randomized algorithms
Duration prediction
Buy or rent

ABSTRACT

We consider the following generalization of the classical problem of ski rental. There is a game that ends at an unknown time, and the algorithm needs to decide how to pay for the time until the game ends. In our generalization, there are two "payment plans" called "options," such that each option i (for $i = 1, 2$) consists of two kinds of costs: b_i is the (one time) cost to start using Option i, and a_i is the (ongoing) usage cost per unit of time for Option i. We assume w.l.o.g. that $a_1 > a_2$ and $b_1 < b_2$. Additionally, we assume the existence of a transition cost c, which is incurred if we switch from Option 1 to Option 2. (In the classical version, $b_1 = 0$, $a_2 = 0$ and $c = b_2$.)
We give deterministic and randomized algorithms for this general setting and analyze their competitive ratio. We also prove that the competitive ratios of our algorithms are the best possible by presenting matching lower bounds for both the deterministic and the randomized cases.

1. Introduction and summary

To buy or not to buy? This classical dilemma is formalized quantitatively by the "ski rental" problem [8]: one alternative is better for the long term, the other for the short term, and only future will tell which is the right choice. Ski rental is a fundamental on-line problem, as it allows us to better understand how to minimize the cost of predicting time duration. This abstraction is useful in many computer-related scenarios, e.g., snoopy caching and TCP acknowledgment batching (see [8,6]), it is a topic of interest in communication systems field, e.g., session management [2], and obviously it applies also to many real-life situations, e.g., payment plans [4,14].

The most basic setting is as follows. We are given two ways to pay for some resource we need. In the "buy" option there is a one-time fee and that's it, and in the "rent" option, we pay proportionally to the actual usage time. (These options are sometimes called "slopes".) The algorithm needs to decide how to pay for the usage, which boils down to decide if and when to switch from the rent to the buy option, and the challenge is that the duration of the time we need to pay for is unknown in advance. From the competitive analysis point of view [3], one would like to bound the *competitive ratio*, namely the worst-case ratio, over all possible instances, of the cost paid by the algorithm, to the optimal cost (which can be known only in hindsight). It is straightforward to see that the deterministic competitive ratio is 2 for this setting. A deeper result shows that the randomized competitive ratio[2] is $e/(e-1) \approx 1.58$ [7]. Intuitively, it turns out that it is a good idea to *guess* what is the distribution of the game; the key is the probability distribution over the guesses.

E-mail addresses: amirlevi@tau.ac.il (A. Levi), boaz@tau.ac.il (B. Patt-Shamir).

[1] Supported in part by the Israel Science Foundation (grant 1444/14) and by a grant from the Israel Ministry of Science, Technology and Space, Israel and the French Ministry of Higher Education and Research (Maimonide 31768XL).

[2] The competitive ratio of a randomized online algorithm is the least upper bound, over all instances, on the ratio between the *expected* cost incurred by the algorithm for a given instance to the best possible cost for that instance.

http://dx.doi.org/10.1016/j.tcs.2015.01.038
0304-3975/

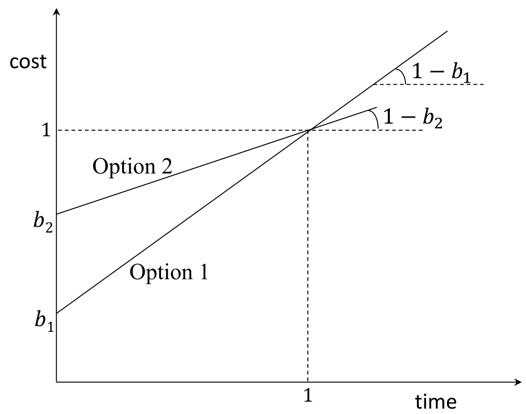

Fig. 1. A normalized graphic representation of the input. Option i is denoted by $b_i + (1 - b_i)t$, namely it intersects the y axis at b_i and its slope is $1 - b_i$. In addition there is a transition cost from Option 1 to Option 2 denoted by c and is independent of b_1 and b_2.

Recently there was some renewed interest in the problem, motivated by power-saving models: Augustin et al. [1] mapped the "buy" and "rent" options to different operational modes of a system where cost models energy consumption. Inspired by this correspondence, they generalized the problem to include options whose associated cost is a general linear function of time, i.e., any one-time fee and any ongoing payment rate. Formally, using an (a, b)-option for t time units in this model has cost $at + b$, where $a, b \geq 0$ are given constants.

To fully formalize generalized ski rental, one has also to state precisely what happens when the algorithm switches from one option to another. Augustin et al. [1] distinguish between two variants of this issue: in the *additive* model, the cost of just switching from an (a, b) option to an (a', b') option is $b' - b$, namely the one-time cost paid in the past is deducted from future one-time costs. In the *non-additive* model, by contrast, it is assumed that the cost of switching between any two options is arbitrary (usually represented by a "transition matrix").

In this work we present the first complete study of a non-trivial variant of non-additive ski rental. Specifically, the problem we study, called NTSR, is defined as follows.

*The non-additive two-slope ski rental problem (*NTSR*)* We are given five parameters: a_1, a_2, b_1, b_2 and c, all non-negative real numbers. We consider 2 payment options such that:

- Option 1 is characterized by a_1 and b_1.
- Option 2 is characterized by a_2 and b_2.

The total cost is determined by the following rules:

1. Using Option i for t time units costs $a_i t$.
2. Starting with Option i costs b_i.
3. Transition from Option 1 to Option 2 costs c.

For example, if we start with option 1 at time 0 and switch to option 2 at time $t_s > 0$, then the total cost at time $t_f \geq t_s$ is $b_1 + a_1 t_s + c + a_2(t_f - t_s)$. The algorithm may also start with Option 2, in which case it never switches to Option 1. In that case, if the game lasts t_f time units, the total cost would be $b_2 + a_2 t_f$.

Intuitively, this model formalizes simple situations in which changing one's mind, even instantly, has a cost. In the additive model, the cost of starting with Option 1 and immediately switching to Option 2 is the same as starting with Option 2. In the non-additive model this is not the case: If the algorithm starts with Option 1 and immediately switches to Option 2, the resulting cost is $b_1 + c$ while the cost of starting with Option 2 upfront is b_2.

1.1. Preliminary simplifications

We make a few assumptions to simplify notation and focus on the interesting cases. These assumption do not restrict the generality of our results.

- We assume w.l.o.g. that $a_1 > a_2$ and $b_1 < b_2$, so that Option 1 is closer to "rent" and Option 2 is closer to "buy". If this is not the case even after renaming, then one option is always better than the other and the question is trivial.
- To reduce the number of parameters, we rescale the time and cost units so that the graphs of the two options intersect at the point $(1, 1)$ (see Fig. 1). Formally, given the parameters as above using any "old" units, we define a "new" time