Theoretical Computer Science 527 (2014) 61–72

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

The online knapsack problem: Advice and randomization ☆,☆☆

Hans-Joachim Böckenhauer [a], Dennis Komm [a,*], Richard Královič [a], Peter Rossmanith [b]

[a] Department of Computer Science, ETH Zurich, Switzerland
[b] Department of Computer Science, RWTH Aachen, Germany

ARTICLE INFO

Article history:
Received 8 February 2012
Received in revised form 18 September 2013
Accepted 23 January 2014
Communicated by T. Erlebach

Keywords:
Online computation
Advice complexity
Randomization
Resource augmentation
Knapsack problem

ABSTRACT

We study the advice complexity and the random bit complexity of the online knapsack problem. Given a knapsack of unit capacity, and n items that arrive in successive time steps, an online algorithm has to decide for every item whether it gets packed into the knapsack or not. The goal is to maximize the value of the items in the knapsack without exceeding its capacity. In the model of advice complexity of online problems, one asks how many bits of advice about the unknown parts of the input are both necessary and sufficient to achieve a specific competitive ratio. It is well-known that even the unweighted online knapsack problem does not admit any competitive deterministic online algorithm.

For this problem, we show that a single bit of advice helps a deterministic online algorithm to become 2-competitive, but that $\Omega(\log n)$ advice bits are necessary to further improve the deterministic competitive ratio. This is the first time that such a phase transition for the number of advice bits has been observed for any problem. Additionally, we show that, surprisingly, instead of an advice bit, a single random bit allows for a competitive ratio of 2, and any further amount of randomness does not improve this. Moreover, we prove that, in a resource augmentation model, i.e., when allowing the online algorithm to overpack the knapsack by some small amount, a constant number of advice bits suffices to achieve a near-optimal competitive ratio.

We also study the weighted version of the problem proving that, with $\mathcal{O}(\log n)$ bits of advice, we can get arbitrarily close to an optimal solution and, using asymptotically fewer bits, we are not competitive. Furthermore, we show that an arbitrary number of random bits does not permit a constant competitive ratio.

1. Introduction

Online problems are an important class of computing problems where the input is not known to the algorithm in advance, but is revealed stepwise, and where, in each step, a piece of output has to be produced irrevocably. The standard way to analyze the quality of an online algorithm is via the so-called *competitive analysis*. Here, the quality of the solution as produced by the online algorithm is compared to the quality of an offline algorithm that knows the complete input in advance. An introduction to the theory and applications of competitive analysis is given in [1,5,12].

☆ This work was partially supported by ETH grant TH 18 07-3 and SNF grants 200020-120073 and 200021-141089.

☆☆ An extended abstract of this paper was presented at LATIN 2012 [4].

* Corresponding author.
E-mail addresses: hjb@inf.ethz.ch (H.-J. Böckenhauer), dennis.komm@inf.ethz.ch (D. Komm), richard.kralovic@inf.ethz.ch (R. Královič), rossmani@cs.rwth-aachen.de (P. Rossmanith).

0304-3975/$ – see front matter © 2014 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.tcs.2014.01.027

Comparing an algorithm having no knowledge about the forthcoming parts of the input with an algorithm having full knowledge of the future might only give a rough estimate of the real quality of an algorithm facing an online situation. To enable a more fine-grained analysis of the hardness of online problems, the *advice complexity* of online problems has been introduced recently [3,7,10,18]. The idea behind this concept is to measure the amount of information about the forthcoming parts of the input an online algorithm needs to be optimal or to achieve a certain competitive ratio. More precisely, in this model, the online algorithm has access to some tape containing advice bits produced by an oracle knowing the complete input, and its advice complexity is the number of bits it reads from this advice tape, i.e., the amount of information about the yet unknown input parts it needs to know for its computation.

In this paper, we deal with an online version of the well-known knapsack problem. Here, an input consists of a set of items with specified weights and values, and a knapsack capacity. The goal is to choose a set of items with maximum value such that the total sum of their weights does not exceed the knapsack's capacity. The knapsack problem is a very well-studied hard optimization problem, for an introduction, see [17,25]. In the online version of the knapsack problem, the items arrive one by one and the algorithm has to decide for each item whether it will pack it into the knapsack or not. These decisions must not be withdrawn at a later stage, i.e., no items can be removed from the knapsack. It is easy to see that no deterministic online algorithm can achieve any bounded competitive ratio [28]. Thus, the existing literature on the online knapsack problem mainly considers restricted variants of the problem [14,20,34] or an average-case analysis of randomized algorithms [28].

Our results are as follows. For the unweighted version of the problem, we prove that, with a single advice bit, a competitive ratio of 2 is achievable. Moreover, for an instance of n items, any number b, $2 < b < \log(n-1)$, of advice bits cannot improve the competitive ratio. However, for every constant $\varepsilon > 0$, a competitive ratio of $1+\varepsilon$ is achievable using $\mathcal{O}(\log n)$ advice bits. For computing an optimal solution, a linear number of advice bits is necessary. At first glance, these results fit well into the picture as given by the advice complexity results for other problems like paging, job shop scheduling, or disjoint path allocation [3]: Linear advice is needed for optimality, logarithmic advice for beating the best randomized algorithm, and very few bits suffice to beat a deterministic algorithm. But having a closer look, one sees that the situation is pretty much different for the knapsack problem compared to the other above-mentioned problems: This problem is the first one for which a sharp phase transition in the number of advice bits can be shown in the following sense. Even $\log(n-2)$ advice bits are exactly as helpful as one bit, but $\mathcal{O}(\log n)$ bits already allow for an almost optimal solution.

A second line of research in this paper considers the random bit complexity of randomized online algorithms (without advice) for the knapsack problem. For the unweighted case, it turns out that, surprisingly, a single random bit is as powerful as an advice bit, i.e., a single random bit can be used to achieve an expected competitive ratio of 2. Moreover, we prove that an arbitrary amount of additional randomness does not help at all, no randomized algorithm can achieve an expected competitive ratio better than $2-\varepsilon$, for any $\varepsilon > 0$.

We analyze the behavior of online algorithms with advice that are allowed to overpack the knapsack by some small constant amount of δ. In contrast to the original model, we show that, in this case, a constant number of advice bits is already sufficient to achieve a near-optimal competitive ratio when dealing with the unweighted knapsack problem.

In the second part of the paper, we study the weighted version of the problem; obviously, all lower bounds carry over immediately from the unweighted one. We show that, with less than $\log n$ advice bits, no online algorithm is competitive and that we can be arbitrarily close to the optimum when using $\mathcal{O}(\log n)$ advice bits. Conversely, randomized online algorithms are not competitive independent of the number of random bits used.

1.1. Related work

The decision version of the offline knapsack problem is one of Karp's famous 21 $\mathcal{NP}$-hard problems [23] and thus one of the first computing problems that were considered solvable, but intractable. Regarding the optimization variant, the knapsack problem admits a *pseudo-polynomial-time algorithm* using dynamic programming. Ibarra and Kim used this approach to design a *fully polynomial-time approximation scheme* [19]. An online version of the knapsack problem was first analyzed by Marchetti-Spaccamela and Vercellis [28]; they gave the following result.

Theorem 1 *(Marchetti-Spaccamela and Vercellis). (See [28].) No deterministic online algorithm for* SimpleKnapsack *(and thus* Knapsack*) without advice is competitive.* □

Due to this negative result, relaxed formulations were investigated. Iwama and Taketomi introduced the online removable knapsack problem; this variant allows to discard items that are already packed retrospectively [20]. A two-dimensional version of the problem was introduced by Han et al. [13].

The classical tool to analyze online algorithms, competitive analysis, was introduced by Sleator and Tarjan in the mid 80s [31]. Dobrev et al. were the first to equip online algorithms with an oracle that helps them to compute a high-quality solution [7]. Their model was then revised and applied to a number of different online problems; for a detailed introduction to the advice complexity of online problems, see [3,18]. More results on the advice complexity of specific problems are found in [2,10,27], the relationship between advice complexity and randomized algorithms is discussed in [2,26].

It was, however, pointed out earlier that competitive analysis, i.e., comparing online algorithms to an optimal offline algorithm, may be inaccurate to analyze the performance of algorithms that work under uncertainty, see, e.g., [1]. The textbook