# Removable Online Knapsack with Bounded Size Items

Laurent Gourvès, Aris Pagourtzis

# Removable Online Knapsack with Bounded Size Items

Laurent Gourvès[1][0000−0002−5076−1583] and Aris Pagourtzis[2,3⋆][0000−0002−6220−3722]

[1] Université Paris Dauphine-PSL, CNRS, LAMSADE, 75016, Paris, France
[2] National Technical University of Athens, 15780, Zografou, Greece
[3] Archimedes Research Unit, Athena RC, 15125, Marousi, Greece
laurent.gourves@dauphine.fr, pagour@cs.ntua.gr

**Abstract.** In the online unweighted knapsack problem, some items arrive in sequence and one has to decide to pack them or not into a knapsack of given capacity. The objective is to maximize the total size of packed items. In the traditional setting, decisions are irrevocable, and the problem cannot admit any $\rho$-competitive algorithm. The removable nature of the items allows to withdraw previously packed elements of the current solution. This feature makes the online knapsack problem amenable to competitive analysis under the ratio of $(\sqrt{5}-1)/2$, which is at the same time the best possible performance guarantee [12]. This article deals with refinements of the best possible competitive ratio of the online unweighted knapsack problem with removable items when either an upper or a lower bound on the size of the items is known.

**Keywords:** Online Algorithms · Knapsack · Competitive analysis.

## 1 Introduction

This article deals with online computation where an instance is revealed over time and an irrevocable decision has to be made each time a portion of the input is known [1]. The problem under study in this work is *knapsack* [14]. In its online version, we are given the capacity $C$ of a knapsack, and items are disclosed sequentially. Each item has a positive size and a positive weight. The goal is, as usual, to pack items into the knapsack so as to maximize their total weight, under the constraint that the total size of packed items does not exceed $C$. The online knapsack problem is appealing since it models many real life situations, and many articles have been devoted to it (see for example [16, 15, 12, 2, 13, 4, 7, 5, 3]). Unfortunately, one can rapidly observe that no deterministic algorithm can exhibit a bounded competitive ratio, which is a worst case performance guarantee of an online algorithm against an ideal procedure which always makes

⋆ Aris Pagourtzis has been partially supported for this work by project MIS 5154714 of the National Recovery and Resilience Plan Greece 2.0 funded by the European Union under the NextGeneration EU Program.

the optimal decision. This bad news holds even in the unweighted case (a.k.a. *subset sum*) where the weight of every item is equal to its size. Indeed, think of an unweighted instance where the first item is very small, say $\epsilon \cdot C$ with $1 \gg \epsilon > 0$. If the item is not packed, and no other item is disclosed, then the competitive ratio is $0/(\epsilon \cdot C) = 0$. If the item is packed, but a second item of size $C$ arrives, then the first item prevents the second one from being packed. The competitive ratio is $(\epsilon \cdot C)/C = \epsilon$ because the right decision would be to pack the second item. In any case, the competitive ratio is at most $\epsilon$, which can be arbitrarily small.

Then, how can we bypass this difficulty? Quoting the authors of [14], we need "*to find suitable restrictions on the pure online formulation which make sense from a real-world point of view and permit the construction and analysis of more successful algorithms.*" Many successful directions have been proposed for knapsack, including the notion of *removable items* introduced by Iwama and Taketomi [12]. An item is removable if its insertion in the knapsack is a revocable decision. In this setting, a deterministic $\frac{\sqrt{5}-1}{2}$-competitive online algorithm exists, along with a proof that $\frac{\sqrt{5}-1}{2}$ is the best possible competitive ratio [12].

In the present work, we aim at going further and combine item removability with an additional information on item sizes in order to devise a more accurate competitive analysis. The previously mentioned example comprises two "extreme" items whose size relative to the capacity is either very small or very big ($\epsilon \cdot C$ and $C$, respectively). Such an instance may poorly represent the full spectrum of concrete situations that require to solve an online knapsack problem. In practice, one can expect to deal with instances where the items' sizes are less heterogeneous. Moreover, especially when a decision maker has already faced several instances in the past, she or he may be knowledgeable of a bound on the size of every item. Therefore, we propose to exploit the existence and knowledge of such a bound towards a refined analysis of the best possible competitive ratio of deterministic algorithms for the online unweighted knapsack problem with removable items.

**Setting and contribution.** As in [12, 7, 3, 13], we consider the online unweighted knapsack (a.k.a. subset sum) problem. Consider a capacity $C$ (a.k.a. budget) of 1 and some removable items disclosed online $o_1, o_2, \ldots$. Each item $o_i$ has a size $|o_i| \in (0, 1]$ and an identical weight of $|o_i|$. Assuming that $C = 1$ and $|o_i| \in (0, 1]$ for all $i$ is without loss of generality since dividing everything by $C$ does not affect the competitive ratio. The number of items is not known in advance. Starting from an initial empty set $S$, an item $o_i$ is revealed at every time step $i$ (also called round), and one has to decide if $o_i$ is packed in $S$ or not. Elements inserted during previous rounds can be removed, but discarded items (i.e., items that were directly rejected or removed after their insertion) cannot be included afterwards. The value of $S$, denoted by $v(S)$ and defined as $\sum_{o_i \in S} |o_i|$, should never exceed the capacity. The objective is to maximize $v(S)$ when no more items are disclosed. The competitive ratio $\rho_{\mathcal{A}}$ of a deterministic online algorithm $\mathcal{A}$ is the worst case ratio between the value of the solution $S_{\mathcal{A}}$ built by $\mathcal{A}$ and the value of an optimal solution $S^*$ which maximizes $v(S^*)$:

$\rho_{\mathcal{A}} := v(S_{\mathcal{A}})/v(S^*)$. The online unweighted knapsack problem with removable items admits a $t$-competitive online algorithm where

$$t := \frac{\sqrt{5}-1}{2} \approx 0.618 \tag{1}$$

is the golden ratio conjugate, and no deterministic online algorithm can be $(t+\epsilon)$-competitive for any positive $\epsilon$ [12].[4] Our contribution is to go beyond this result and propose refined bounds on the best competitive ratio based on known bounds on the size of the items. All the results of this article apply to *deterministic* algorithms for the removable online unweighted knapsack problem. In Section 2, some parameter $u \in (0, 1]$ such that any item's size is *upper bounded* by $u$ is given, and we provide both lower and upper bounds on the best possible competitive ratio $\rho(u)$. In Section 3, a lower bound $\ell \in (0, 1]$ on the size of every item is known, and we characterize the best possible competitive ratio $\rho(\ell)$.

Due to space constraints, some technical elements are skipped and will be made available in an extended version of the article.

**Related work.** The online knapsack problem was first studied by Marchetti-Spaccamela and Vercellis who made an average case analysis of the expected difference between the optimal and the approximate value [16]. A follow-up article, with a similar approach, is authored by Luecker [15].

Iwama and Taketomi introduced the notion of removable items and proved that the unweighted online knapsack problem admits a deterministic $t$-competitive algorithm, where $t$ is the best possible performance guarantee [12]. Unfortunately, no competitive algorithm can exist for the online *weighted* knapsack with removable items [13]. Sometimes, removing a packed item comes with a cost which is equal to $f$ times the item's size where $f > 1$ is a given *buyback factor*. The objective is to maximize the worth of packed items minus the cost paid for items which were removed after been packed. In this case, Han et al. studied the unweighted online knapsack problem [7] whereas Babaioff et al. considered a more general setting including the online weighted knapsack problem [2]. Their results include deterministic and randomized algorithms whose competitive ratio depends on $f$.

Other randomized algorithms for the online knapsack problem can be found in [8, 5] when $f = 0$. For the weighted case (resp., unweighted case), the competitive ratio for removable items is between 0.5 and $\frac{e}{1+e} \approx 0.73$ (resp., between 0.7 and 0.8). When items cannot be removed, the best competitive ratio for the weighted and unweighted cases are 0 and 0.5, respectively.

In [13], the authors consider the online knapsack problem under the *resource augmentation* framework where the online knapsack is $R \geq 1$ times larger than its offline counterpart. Their results consist of bounds on the competitive ratio as a function of $R$ for removable or weighted items. In the same vein, an alternative model supposes that the items can be placed in a buffer of size $K > 1$ before a selection of them is put in a knapsack of size 1 [10].

[4] Every competitive ratio of this article is in $[0, 1]$.

Sometimes, the items can be split and partially included in the knapsack. Han and Makino exploited this opportunity to derive a $k/(k+1)$-competitive online algorithm for the removable weighted case and a matching upper bound [11]. Here, $k$ is the maximum number of times that an item can be cut.

In [4], Böckenhauer et al. explore the *advice complexity* of the online knapsack problem, where the goal is to evaluate the possible improvements on the competitive ratio provided by some additional information about the complete instance.

In a recent article by Böckenhauer et al. [3] on the online unweighted knapsack problem, the possibility to reserve an item (i.e., postpone the decision about it) is studied. Given $\alpha \in (0,1)$, reserving an item costs $\alpha$ times its value. This model is closely related to the one with removal cost [2, 7] but, as opposed to "bought back" items, reserved items are not temporarily put into the knapsack. Thus, there is no hard capacity constraint. The authors characterize the best competitive ratio for all possible value of $\alpha$ [3].

The present work combines removable items with an additional information about the range of the items size. Having upper bounds on the worth of objects has already been done for analysing fair allocations of indivisible goods [6]. Concerning the online weighted knapsack problem, Babaioff et al. have used a parameter $\gamma \in (0,1]$ which restricts the size of any item to $\gamma \cdot C$. They gave a deterministic algorithm for the case $\gamma < 1/2$ whose competitive ratio tends to $1-2\gamma$ when the buyback factor goes to 0. The work of Chakrabarty et al. on the weighted knapsack problem also makes assumptions about both the size of objects and their weight-to-size ratio [17].

## 2 Upper bounded item size

Given an upper bound $u \in (0,1]$ on the size of every item, we aim at determining the best possible competitive ratio $\rho(u)$ of deterministic online algorithms for the unweighted knapsack problem with removable items. Our results are depicted in Figure 1. The lower (solid) and upper (dotted) bounds on $\rho(u)$ are non increasing functions $u$. One can observe that the competitive ratio is always above $t$, and the performance guarantee increases (and tends to 1) as the size of the items reduces. The curves of Figure 1 meet for many values of $u$ all over the interval $(0,1]$ but some gaps remain to be filled.

In this section, we begin with lower bounds on $\rho(u)$ obtained with a single parametric algorithm (cf. Section 2.1), followed by upper bounds on $\rho(u)$ induced by a family of instances (cf. Section 2.2).

### 2.1 Lower bounds on the competitive ratio

Throughout our presentation we will say that an item is *rejected* if it is discarded immediately after its disclosure; if the item is first inserted in the solution and it is discarded at a later step then we say that it is *removed*. We consider a single

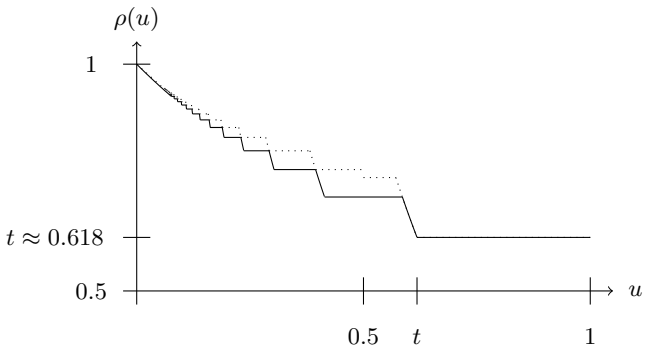


**Fig. 1.** Lower (solid) and upper (dotted) bounds on the competitive ratio $\rho$ as a function of $u$.

parametric online algorithm described in Algorithm 1. For every positive integer $k$, the competitive ratio of Algorithm 1 is either a constant $\gamma_k$ defined as

$$\gamma_k = \frac{k - 2 + \sqrt{k^2 + 4}}{2k}, \tag{2}$$

or a decreasing function of $u$ equal to $\frac{1-u}{(k-1)u}$.

Note that $\gamma_k$ belongs to $(0, 1]$ for all positive integers $k$. It increases with $k$, and $\gamma_1 = \frac{\sqrt{5}-1}{2} = t$. The rationale of $\gamma_k$ originates from the following equality which will be interpreted and exploited later on.

$$(k-1)(1-\gamma_k) + \frac{1-\gamma_k}{\gamma_k} = \gamma_k \tag{3}$$

Other useful technical properties of $\gamma_k$ (valid for all positive integers $k$) are the following four (their proof will appear in an extended version of this article).

$$0 < 1 - \gamma_k \leq \frac{1-\gamma_k}{\gamma_k} < 1/k \leq \frac{1-\gamma_k}{\gamma_k^2} \tag{4}$$

$$(k-1)\frac{1-\gamma_k}{\gamma_k^2} \leq 1 \tag{5}$$

$$(k+1)(1-\gamma_k) \geq \gamma_k \tag{6}$$

$$\frac{1 - \frac{1-\gamma_k}{\gamma_k^2}}{(k-1)\frac{1-\gamma_k}{\gamma_k^2}} = \gamma_k \tag{7}$$

Our analysis of the competitive ratio of Algorithm 1 is divided into two theorems for the sake of clarity. Theorems 1 and 2 correspond to intervals where the proposed lower bound on the competitive ratio is constant and decreasing with $u$, respectively. On Figure 1, the $i$-th constant part of the solid curve,

**Algorithm 1**

```
1: S ← ∅
2: while a new item o_i arrives do
3:    if v(S) ≥ γ_k then
4:       Reject o_i {S is not changed afterwards}
5:    else
6:       S ← S ∪ {o_i}
7:       if v(S) > 1 then
8:          Let B = {o ∈ S : |o| > 1 − γ_k}
9:          if v(B) > 1 then
10:            if B contains a subset Ŝ such that 1 ≥ v(Ŝ) ≥ γ_k then
11:               S ← Ŝ {S is not changed afterwards}
12:            else
13:               Remove the largest item of S
14:            end if
15:         else
16:            while v(S) > 1 do
17:               Remove from S one element of S \ B {chosen arbitrarily}
18:            end while
19:         end if
20:      end if
21:   end if
22: end while
```

when counting from the right, corresponds to Theorem 1 when $i = k$ and $u \leq (1 - \gamma_k)/\gamma_k^2$. The $i$-th decreasing part of the solid curve, still counting from the right, corresponds to Theorem 2 when $i = k - 1$ and $u \leq 1/(k - 1)$.

The solid curve depicted on Figure 1 corresponds, for every possible value of $u \in (0, 1]$, to the best (i.e., largest) lower bound offered by either Theorem 1 or Theorem 2, with an appropriate choice of the parameter $k$.

**Theorem 1.** *For all positive integers $k$, Algorithm 1 is $\gamma_k$-competitive when $u \in (0, \frac{1-\gamma_k}{\gamma_k^2}]$.*

*Proof.* Let $B_i$, $\hat{S}_i$, $S_i$, and $S_i^*$ denote $B$, $\hat{S}$, $S$ and an optimal solution at the end of round $i$ of Algorithm 1 (i.e., when $o_1$ to $o_i$ have been disclosed), respectively. We are going to prove that $v(S_i) \leq 1$ and $\frac{v(S_i)}{v(S_i^*)} \geq \gamma_k$ hold for all $i$.

An item $o$ is said to be *small* if $0 < |o| \leq 1-\gamma_k$, *medium* if $1-\gamma_k < |o| \leq \frac{1-\gamma_k}{\gamma_k}$, *large* if $\frac{1-\gamma_k}{\gamma_k} < |o| \leq k^{-1}$, and *extra-large* if $k^{-1} < |o| \leq \frac{1-\gamma_k}{\gamma_k^2}$. Note that the extra-large category does not exist when $k = 1$. The categories lead to the following useful interpretation of equation (3): $k-1$ medium items plus one large or extra-large item constitute a desirable set because its total size is between $(k-1)(1-\gamma_k) + \frac{1-\gamma_k}{\gamma_k} = \gamma_k$ and $(k-1)\frac{1-\gamma_k}{\gamma_k} + \frac{1-\gamma_k}{\gamma_k^2} = 1$, i.e., it is a feasible set satisfying the guarantee $\gamma_k$.

The proof is by induction and we begin with the base case ($i = 1$). Since $S_1 = S_1^* = \{o_1\}$, we have that $v(S_1) \leq 1$ and $\frac{v(S_1)}{v(S_1^*)} = 1 \geq \gamma_k$. In order to prove

$v(S_i) \leq 1$ and $\frac{v(S_i)}{v(S_i^*)} \geq \gamma_k$ when $i > 1$, we make the induction hypotheses that both $v(S_{i-1}) \leq 1$ and $\frac{v(S_{i-1})}{v(S_{i-1}^*)} \geq \gamma_k$ hold.

If $v(S_{i-1}) \geq \gamma_k$ at line 4 of the algorithm, then $S_i = S_{i-1}$ (item $o_i$ is rejected). Since $v(S_i^*) \leq 1$, the competitive ratio satisfies $\frac{v(S_i)}{v(S_i^*)} \geq \gamma_k$. The solution is not modified afterwards. Otherwise ($v(S_{i-1}) < \gamma_k$), the algorithm puts the new item $o_i$ in the current solution (cf. line 6): $S_i = S_{i-1} \cup \{o_i\}$. If $o_i$ fits (i.e., $v(S_i) \leq 1$ holds after the insertion of $o_i$), then $v(S_i) = v(S_{i-1}) + |o_i|$ and $v(S_i^*) \leq v(S_{i-1}^*) + |o_i|$. We get that $\frac{v(S_i)}{v(S_i^*)} \geq \frac{v(S_{i-1})+|o_i|}{v(S_{i-1}^*)+|o_i|} \geq \frac{v(S_{i-1})}{v(S_{i-1}^*)} \geq \gamma_k$ where the last inequality derives from the induction hypothesis.

From now on, suppose that $v(S_i) > 1$ holds after the insertion of $o_i$. We know from $v(S_{i-1}) < \gamma_k$ and $v(S_i) > 1$ that $|o_i| > 1 - \gamma_k$. In other words, $o_i$ is not a small item. By construction, $B_i$ is the subset of non-small items of $S_i$ and it contains $o_i$. If $v(B_i) \leq 1$, then the algorithm executes the while loop containing line 17. The while loop starts with $v(S_i) > 1$ and removes small items of $S_i$ until $v(S_i) \leq 1$. Since a small item has size at most $1 - \gamma_k$, we end up with a solution satisfying $\gamma_k \leq v(S_i) \leq 1$. Therefore, $\frac{v(S_i)}{v(S_i^*)} \geq \gamma_k$ and the solution is not modified afterwards. If $v(B_i) > 1$, then the algorithm tries to find in $B_i$ a subset of non-small items $\hat{S}_i$ such that $1 \geq v(\hat{S}_i) \geq \gamma_k$, and sets $S_i$ to $\hat{S}_i$ if $\hat{S}_i$ exists. In this case, the competitive ratio is reached and the solution is not modified afterwards.

Let us explain that verifying the existence of $\hat{S}_i$ is not difficult. $B_i$ contains at most $k+1$ items for the following reasons: $|B_i| - 1$ non-small items of $B_i$ were already present in $S_{i-1}$. Using $v(S_{i-1}) < \gamma_k$, and the fact that a non-small item has size at least $1 - \gamma_k$, we get that $|B_i| - 1 \leq k$ (indeed, Inequality (6) indicates that $k+1$ items of size $1 - \gamma_k$ have a total size of at least $\gamma_k$), which is equivalent to $|B_i| \leq k + 1$. Inequality (5) indicates that a set of $k - 1$ items of the largest possible size fits in the budget of 1. Together with $v(B_i) > 1$, we deduce that $B_i$ cannot contain $k - 1$ (or less) non-small items. Thus, $B_i$ contains $k$ or $k + 1$ non-small items. Taking $k - 1$ items of $B_i$ gives a feasible solution (cf. Inequality (5)). By taking the $k - 1$ biggest items of $B_i$, we can verify whether $\gamma_k$ can be reached. If not, $\hat{S}_i$ possibly requires $k$ items (when $|B_i| = k + 1$), which requires to test $k + 1$ possibilities. In all, at most $k + 2$ solutions are tested, where $k$ is upper bounded by the number of disclosed items.

So far we have considered the cases where the algorithm was able to find a subset of items whose total size is between $\gamma_k$ and 1. In these cases, the solution is not changed afterwards because the expected guarantee is reached, whichever item comes subsequently. Hereafter, we analyse the case where $\hat{S}_i \subseteq B_i$ such that $1 \geq v(\hat{S}_i) \geq \gamma_k$ does not exist. In this situation, the largest element is removed (cf. line 13). Note that line 13 is executed for round $i$, and line 13 was possibly executed during previous rounds. However, line 11 or the while loop were not previously executed because, so far, the guarantee has not been reached ($v(S_{i-1}) < \gamma_k$).

We have seen that $B_i$ contains either $k$ or $k + 1$ non-small items at line 8 of the algorithm. Let us make some observations which are valid for both cases.

(*i*) So far, the algorithm has not rejected or removed a small item. Only items of $B_i$ were removed by the possible execution of line 13 (by construction, $B_i$ is the set of non-small items).

(*ii*) The algorithm has not switched from one case to the other. If $|B_i| = k$, then it could not be $|B_j| = k+1$ in a previous round $j < i$. Similarly, if $|B_i| = k+1$, then it could not be $|B_j| = k$ in a previous round $j < i$. Indeed, suppose we have $|B_i| \neq |B_{i-1}|$.[5] Since we removed one item from $B_{i-1}$, and $o_i$ was inserted afterwards to yield $B_i$, we get that $|B_i| = |B_{i-1}|$, contradiction.

(*iii*) We have $v(S_i) \leq 1$ after the largest element of $B_i$ is removed (line 13). Indeed, $S_i$ is built as follows: take $S_{i-1}$, add an item $o_i$, and remove its largest item. We get that $v(S_i) \leq v(S_{i-1})$ and $v(S_{i-1}) \leq 1$ holds by induction hypothesis.

(*iv*) An optimal (offline) solution contains at most $|B_i|-1$ non-small items. Since the algorithm keeps the smallest non-small items in $B_i$, and until now $v(B_i)$ is always strictly larger than 1, it is not possible to find $|B_i|$ non-small items (within the set of non-small items already disclosed) whose total size is at most 1.

It remains to prove $\frac{v(S_i)}{v(S_i^*)} \geq \gamma_k$ in both cases, namely $|B_i| \in \{k, k+1\}$ at line 8 of the algorithm.

Case $|B_i| = k$. The current solution $S_i$ contains all the small items disclosed so far (cf. observation (*i*)). The total size of non-small items of $S_i$ is at least $1 - \frac{1-\gamma_k}{\gamma_k^2}$ because we had $v(B_i) > 1$ and one item of size at most $\frac{1-\gamma_k}{\gamma_k^2}$ has been removed. Meanwhile, observation (*iv*) says that the optimum contains at most $|B_i| - 1 = k - 1$ non-small items, each of which has size at most $\frac{1-\gamma_k}{\gamma_k^2}$, and possibly all the small items disclosed so far. Therefore, the competitive ratio is lower bounded by

$$\frac{v(\{o : 0 < |o| \leq 1-\gamma_k\}) + 1 - \frac{1-\gamma_k}{\gamma_k^2}}{v(\{o : 0 < |o| \leq 1-\gamma_k\}) + (k-1)\frac{1-\gamma_k}{\gamma_k^2}} \geq \frac{1 - \frac{1-\gamma_k}{\gamma_k^2}}{(k-1)\frac{1-\gamma_k}{\gamma_k^2}} = \gamma_k$$

where the last equality is due to (7).

Case $|B_i| = k+1$. Every item of $B_i$ is medium. Indeed, exactly $k$ items of $B_i$ come from $S_{i-1}$ and we know that $v(S_{i-1}) < \gamma_k$. If the non-small items of $S_{i-1}$ were not exclusively medium items, then its total size would be at least $(k-1)(1-\gamma_k) + \frac{1-\gamma_k}{\gamma_k}$, which is equal to $\gamma_k$ by (3), contradiction. The item of $B_i \setminus B_{i-1}$ (i.e., $o_i$) must be medium because the algorithm failed to find $\hat{S}_i$. Indeed, if $o_i$ were large or extra-large, then one could combine it with $k-1$ medium items of $B_i$ and create $\hat{S}_i$ whose total size is between $\gamma_k$ and 1 (cf. interpretation of (3)). No large or extra-large item was disclosed so far, because either this large or

[5] The assumption that the change in the size of $B$ occurs between rounds $i-1$ and $i$ is made without loss of generality because we can apply the arguments to the round (possibly prior to $i$) during which the size of $B$ is modified for the first time.

extra-large item could be used to produce $\hat{S}_i$ with $k-1$ medium items already present in the solution (contradiction with the non-existence of $\hat{S}_i$), or such a large or extra-large item was removed in a previous round $j$, but it corresponds to an incompatible situation where $|B_j| = k \neq |B_i|$, and we have seen that the algorithm does not switch from one case to the other (cf. observation $(ii)$).

The algorithm has kept all the small items disclosed so far and the $k$ smallest medium items. The optimum $S_i^*$ possibly contains all the small items disclosed so far, and at most $k$ non-small items (cf. observation $(iv)$) which are medium because no large or extra-large item has been disclosed. The size of a medium item being between $1-\gamma_k$ and $\frac{1-\gamma_k}{\gamma_k}$, the competitive ratio can be lower bounded as follows.

$$\frac{v(\{o : 0 < |o| \leq 1-\gamma_k\}) + k(1-\gamma_k)}{v(\{o : 0 < |o| \leq 1-\gamma_k\}) + k\frac{1-\gamma_k}{\gamma_k}} \geq \frac{k(1-\gamma_k)}{k(\frac{1-\gamma_k}{\gamma_k})} = \gamma_k \qquad \square$$

When $k = 1$, Theorem 1 indicates that Algorithm 1 is $t$-competitive for all $u \in (0, 1]$ because $\gamma_1 = t$ and $\frac{1-\gamma_1}{\gamma_1^2} = 1$, thus generalizing the result of [12].

When $k \geq 2$, Inequality (5) indicates that $\frac{1-\gamma_k}{\gamma_k^2} \leq \frac{1}{k-1}$ so we can consider in the following theorem how Algorithm 1 performs when $u \in (\frac{1-\gamma_k}{\gamma_k^2}, \frac{1}{k-1}]$.

**Theorem 2.** *For all integers $k \geq 2$, Algorithm 1 is $\frac{1-u}{(k-1)u}$-competitive when $u \in (\frac{1-\gamma_k}{\gamma_k^2}, \frac{1}{k-1}]$.*

*Proof.* We keep the same notations as in the proof of Theorem 1. Only the notion of extra-large item changes: $o$ is said to be extra-large if $k^{-1} < |o| \leq u$. All the cases of the proof of Theorem 1 remain unchanged and lead to a competitive ratio of $\gamma_k$ except when $\hat{S}_i$ such that $\gamma_k \leq v(\hat{S}_i) \leq 1$ does not exist, and $|B_i| = k$. In this case, the current solution $S_i$ contains all the small items disclosed so far. The total size of non-small items of $S_i$ is at least $1-u$ because we had $v(B_i) > 1$ and one item of size at most $u$ has been removed. Meanwhile, the optimum contains at most $|B_i| - 1 = k-1$ non-small items, each of which has size at most $u$, and possibly all the small items disclosed so far. Note that $k-1$ items of size at most $u$ fit in the budget since $u \leq \frac{1}{k-1}$ holds by assumption. The competitive ratio is lower bounded by

$$\frac{v(\{o : 0 < |o| \leq 1-\gamma_k\}) + 1 - u}{v(\{o : 0 < |o| \leq 1-\gamma_k\}) + (k-1)u} \geq \frac{1-u}{(k-1)u}.$$

Using (7) and $\frac{1-\gamma_k}{\gamma_k^2} \leq u$, we get that $\gamma_k = \frac{1-(1-\gamma_k)/\gamma_k^2}{(k-1)(1-\gamma_k)/\gamma_k^2} \geq \frac{1-u}{(k-1)u}$. Therefore, the competitive ratio is $\frac{1-u}{(k-1)u}$ in the worst case. $\square$

To conclude this part, the combination of Theorems 1 and 2 gives the lower bounds on the competitive ratio depicted in solid on Figure 1. Sometimes, the intervals of the theorems intersect for consecutive values of $k$. In this case, the best (i.e., largest) lower bound on the competitive ratio is retained.

### 2.2 Upper bounds on the competitive ratio

Our upper bounds, depicted in dotted on Figure 1, are obtained from instances showing that no deterministic online algorithm can have a competitive ratio larger than some specified value. We will often use the following simple observation: If the instances employed to show an upper bound of $r$ on the competitive ratio only contain items of size at most $x$, then the competitive ratio is at most $r$ for all $u \in [x, 1]$.

Let us first suppose that $u \in [\frac{4}{7}, 1]$, and consider the following instance.

**Instance 1** *The first item $o_1$ has size $u$. If $o_1$ is not taken, then the competitive ratio is zero. The second item $o_2$ has size $1 - u + \epsilon$. Note that there exists an $\epsilon$ such that $1 - u + \epsilon \leq u$ because $u > 0.5$. Since $|o_1| + |o_2| > 1$, either $o_1$ or $o_2$ is kept, but not both. If $o_2$ replaces $o_1$, then no more item is disclosed. The competitive ratio is $\frac{1-u+\epsilon}{u}$ which tends to $\frac{1-u}{u}$ when $\epsilon$ goes to zero. Otherwise, $o_1$ is kept and a last item $o_3$ of size $u - \epsilon$ is disclosed. Again, the two items $o_1$ and $o_3$ do not fit. Since $|o_1| \geq |o_3|$, we can consider that $o_1$ is kept. The competitive ratio in this case is $u$ because the optimum $\{o_2, o_3\}$ has value 1.*

**Proposition 1.** *Instance 1 gives an upper bound of $\frac{1-u}{u}$ on the competitive ratio when $u \in [\frac{4}{7}, t]$, and an upper bound of $t$ when $u \in [t, 1]$.*

*Proof.* The upper bound is $\frac{1-u}{u}$ when $u \in [\frac{4}{7}, t]$ because $\frac{1-u}{u} \geq u$ in that interval. Since $t = \frac{\sqrt{5}-1}{2}$ is a root of $\frac{1-u}{u} = u$, an upper bound of $t$ on the competitive ratio is derived from Instance 1 (fix $u$ to $t$ in the instance), and it is valid for all $u \in [t, 1]$ by the aforementioned observation. □

Note that the upper bound of $t$ corresponds to one given in [12, Theorem 2]. Now suppose that $u \in (\frac{1}{2}, \frac{4}{7})$ and consider the following instance.

**Instance 2** *The first 2 items $o_1, o_2$ have size $\frac{1}{4} - \epsilon$, where $\epsilon > 0$, and $\frac{1}{4}$, respectively. If any of them is not taken, then the competitive ratio tends to $\frac{1}{2}$. The next item $o_3$ has size $\frac{1}{2} + 2\epsilon$. If $o_3$ is kept, then one of $o_1, o_2$ must be removed. Then an item $o_4$ of size $\frac{1}{2} + \epsilon$ is disclosed, which does not fit in the current solution and can only replace $o_3$, leading to a solution of smaller value; hence, $o_4$ is not included, which gives a competitive ratio that tends to $\frac{3}{4}$ when $\epsilon$ goes to zero. Indeed, the total size of the current solution is at most $\frac{3}{4} + 2\epsilon$, while the optimal value is $\frac{1}{4} - \epsilon + \frac{1}{4} + \frac{1}{2} + \epsilon = 1$. If, on the other hand, $o_3$ is removed, then the instance is terminated, leading to a competitive ratio of $\frac{1/4-\epsilon+1/4}{1/4+1/2+2\epsilon} = \frac{1/2-\epsilon}{3/4+2\epsilon}$ that tends to $\frac{2}{3}$ as $\epsilon$ goes to zero.*

**Proposition 2.** *Instance 2 provides an upper bound of $\frac{3}{4}$ on the competitive ratio when $u \in (\frac{1}{2}, \frac{4}{7})$.*

*Proof.* The largest item of Instance 2 has size $1/2 + 2\epsilon$. The two cases lead to competitive ratios of $3/4 + 2\epsilon$ and $\frac{1/2-\epsilon}{3/4+2\epsilon}$, so we retain the largest one which tends to $3/4$ when $\epsilon$ goes to zero. By the above observation, the upper bound of $3/4$ is valid for $u > 1/2$. □

We finally consider a family of instances parameterized by an integer $k$ such that $k \geq 2$, and the interval covered by each individual instance is $(\frac{1}{k+1}, \frac{1}{k}]$. Thus, the family allows us to cover the case $u \in (0, \frac{1}{2}]$.

**Instance 3** *The first $k$ items $o_1, \ldots, o_k$ have size $\frac{1-u}{k} + \epsilon$ each where $\epsilon$ is a small positive real satisfying*

$$0 < \epsilon \leq \frac{1}{k}\left(u - \frac{1}{k+1}\right). \tag{8}$$

*If the first $k$ items are not all taken, then the competitive ratio is at most $\frac{k-1}{k}$. The next item $o_{k+1}$ has size $u$. Note that $u \geq \frac{1-u}{k} + \epsilon$ because of $u > \frac{1}{k+1}$ and (8). Since $\sum_{i=1}^{k+1} |o_i| > 1$, either $\{o_1, \ldots, o_k\}$ is kept (case A), or $o_{k+1}$ replaces $o_i$ for some $i \in \{1, \ldots, k\}$ (case B). If case A occurs, then keep disclosing items of size $u$ until $(i)$ either $k$ items of size $u$ have been disclosed but none of them entered the current solution, or $(ii)$ one item of size $u$ is put into the solution (hence, an item of size $\frac{1-u}{k} + \epsilon$ has been removed). If case A$(ii)$ or case B occurs, then disclose an item of size $\frac{1-u}{k} + \epsilon$. One cannot improve the value of the current solution with this last item (it does not fit, and its size is smaller than the size of any other item of the current solution). The competitive ratio under case A$(i)$ is $\frac{k(\frac{1-u}{k}+\epsilon)}{ku}$ which tends to $\frac{1-u}{ku}$ when $\epsilon$ goes to zero. The competitive ratio under cases A$(ii)$ and B is $\frac{(k-1)(\frac{1-u}{k}+\epsilon)+u}{(k+1)(\frac{1-u}{k}+\epsilon)}$ which tends to $\frac{(k-1)(\frac{1-u}{k})+u}{(k+1)(\frac{1-u}{k})} = \frac{k-1+u}{(k+1)(1-u)}$ when $\epsilon$ goes to zero.*

**Proposition 3.** *Instance 3 provides an upper bound of $\frac{1-u}{ku}$ when $u \in (\frac{1}{k+1}, c_k]$, and an upper bound of $\frac{1-c_k}{kc_k}$ for $u \geq c_k$, where*

$$c_k := \left(k^2 + k + 2 - \sqrt{(k^2+k+2)^2 - 4(k+1)}\right)/2. \text{ [6]}$$

*Proof.* The upper bound derived from Instance 3 is $\max\left(\frac{k-1}{k}, \frac{1-u}{ku}, \frac{k-1+u}{(k+1)(1-u)}\right)$. Since $u \leq \frac{1}{k}$, we know that $\frac{k-1}{k} \leq \frac{1-u}{ku}$. Thus, the upper bound is $\max(\frac{1-u}{ku}, \frac{k-1+u}{(k+1)(1-u)})$ where $\frac{1-u}{ku}$ is a decreasing function of $u$ while $\frac{k-1+u}{(k+1)(1-u)}$ is increasing. The cut point of these functions is $c_k \in (\frac{1}{k+1}, \frac{1}{k}]$. By the aforementioned observation, we have an upper bound on the competitive ratio of $\frac{1-c_k}{kc_k}$ when $u \geq c_k$. □

The combination of Propositions 1, 2 and 3 leads to the upper bounds depicted in dotted on Figure 1.

## 3 Lower bounded item size

Given a lower bound $\ell \in (0, 1]$ on the size of every item, we are going to show that the best possible competitive ratio $\rho(\ell)$ of deterministic online algorithms

[6] Note that $\frac{1-c_k}{kc_k} = \gamma_{k+1}$ and $c_k = \frac{1-\gamma_{k+1}}{\gamma_{k+1}^2}$.

for the unweighted knapsack problem with removable items is as follows.

$$\rho(\ell) = \begin{cases} t, & \text{if } 0 \leq \ell \leq 1-t \\ \sqrt{\ell}, & \text{if } 1-t < \ell \leq 1/2 \\ 1, & \text{if } 1/2 < \ell \leq 1 \end{cases} \tag{9}$$

See (1) for the definition of $t$. One can observe that the competitive ratio is always above $t$. As for Section 2 where an upper bound was known, we begin with lower bounds on $\rho(\ell)$ followed by upper bounds on $\rho(\ell)$. Together, they constitute a characterization of $\rho(\ell)$ since there is no gap.

### 3.1 Lower bounds on the competitive ratio

When $\ell \in [0, 1-t]$, the characterization of $\rho(\ell)$ given in (9) indicates a competitive ratio of $t$ which can be obtained with either Algorithm 1 ($k = 1$), or with the algorithm of Iwama and Taketomi [12]. If $\ell > 1/2$, then there exists a simple algorithm with competitive ratio 1: maintain in the current solution the largest item encountered thus far. Since any solution contains at most one item, this simple algorithm is optimal. It remains to provide a $\sqrt{\ell}$-competitive algorithm for the case $\ell \in [1-t, 1/2]$, cf. Algorithm 2 and Theorem 3. The proof of Theorem 3 is deferred to an extended version of this article.

**Algorithm 2** A $\sqrt{\ell}$-competitive algorithm for the case $1 - t \leq \ell \leq 1/2$

```
S ← ∅
while a new item o_i arrives do
    if v(S) ≥ √ℓ then
        Reject o_i {S is not changed afterwards}
    else if |o_i| ≥ √ℓ then
        S ← {o_i} {S is not changed afterwards}
    else if v(S) + |o_i| ≤ 1 then
        S ← S ∪ {o_i}
    else
        Let o be the unique item in S
        S gets the item of minimum size between o and o_i
    end if
end while
```

**Theorem 3.** *Algorithm 2 is $\sqrt{\ell}$-competitive when $\ell \in [1-t, 1/2]$.*

### 3.2 Upper bounds on the competitive ratio

Let us begin with $\ell \in (0, 1-t]$.

**Proposition 4.** *The competitive ratio of deterministic online algorithms is at most $t$ when $\ell \in (0, 1-t]$.*

*Proof.* We can reuse the bound provided by Iwama and Taketomi [12, Theorem 2]. Let us give the corresponding instance for the sake of readability. The first item $o_1$ has size $1-t$. If $o_1$ is not taken, then the competitive ratio is zero. The second item $o_2$ has size $t+\epsilon$ where $\epsilon$ is a tiny positive real. Items $o_1$ and $o_2$ cannot be both taken without exceeding the budget. If $o_2$ does not replace $o_1$, then stop. The competitive ratio tends to $\frac{1-t}{t}$ when $\epsilon$ goes to zero. Otherwise $o_2$ replaces $o_1$, and a new item $o_3$ of size $t$ is disclosed. The optimum $\{o_1, o_3\}$ has value 1 while the algorithm's solution has value at most $t+\epsilon$. The competitive ratio is either $\frac{1-t}{t}$ which is equal to $t$, or $t+\epsilon$, so we have an upper bound of $t$. In the instance, every item has size at least $1-t$ because $t+\epsilon \geq t \geq 1-t$, so the upper bound on the competitive ratio holds for all $\ell \in (0, 1-t]$. □

The next step concerns the interval $(1-t, \frac{1}{2}]$.

**Proposition 5.** *The competitive ratio of deterministic online algorithms is at most* $\sqrt{\ell}$ *when* $\ell \in (1-t, \frac{1}{2}]$.

*Proof.* Consider the following instance. The first item $o_1$ has size $\ell$. If $o_1$ is not taken, then the competitive ratio is zero. The second item $o_2$ has size $\sqrt{\ell}+\epsilon$. Since $\ell \leq 1$, we know that $\sqrt{\ell}+\epsilon \geq \ell$. Moreover, $\ell + \sqrt{\ell} + \epsilon > 1$ holds when $\ell \geq 1-t$, so no feasible solution can contain both items. If $o_2$ does not replace $o_1$, then stop. The competitive ratio tends to $\ell/\sqrt{\ell} = \sqrt{\ell}$ when $\epsilon$ goes to zero. Otherwise $o_2$ replaces $o_1$, and a new item $o_3$ of size $1-\ell$ is disclosed. The optimum $\{o_1, o_3\}$ has value 1 while the algorithm's solution has value at most $\sqrt{\ell}+\epsilon$ because $\sqrt{\ell}+\epsilon \geq 1-\ell$ holds when $\ell > 1-t$, leading to a competitive ratio which tends to $\sqrt{\ell}$ when $\epsilon$ goes to zero. Thus, both cases lead to the same upper bound of $\sqrt{\ell}$. □

No upper bound is needed when $\ell > 1/2$ because an optimal algorithm has been presented in the previous section.

## 4 Conclusion and directions for future work

We have considered the removable online unweighted knapsack problem with bounded size items (denoted by $u$ and $\ell$ for upper bound and lower bound, respectively). Our contribution consists of lower and upper bounds on the best competitive ratio for deterministic algorithms. The optimal ratio tends to 1 when the parameter $u$ goes to 0. A direct extension to our work would be to close the gap for all possible values of $u$.

The reader may wonder what the situation is when the items are not removable but their size is bounded. Given an upper bound $u$ (resp., lower bound $\ell$) on the item sizes, the best possible competitive ratio is $1-u$ (resp., $\ell$) and simple deterministic algorithms can achieve these ratios.

For the future, we believe that it would be interesting to combine bounded item sizes with other approaches such as resource augmentation [13], buffering [10], reservation [3], advice [4], or item splitting [11]. Other possible extensions of

the present work can be: exploring randomized online algorithms (as in [2, 7, 8, 5]) with bounded size items, exploiting a possible prediction of the total number of disclosed items, or dealing weighted items restricted to convex functions of the size (as in [9]).

## References


1. Albers, S.: Online algorithms: a survey. Math. Program. **97**(1-2), 3–26 (2003)
2. Babaioff, M., Hartline, J.D., Kleinberg, R.D.: Selling ad campaigns: online algorithms with cancellations. In: Chuang, J., Fortnow, L., Pu, P. (eds.) Proceedings 10th ACM Conference on Electronic Commerce (EC-2009), Stanford, California, USA, July 6–10, 2009. pp. 61–70. ACM (2009)
3. Böckenhauer, H., Burjons, E., Hromkovic, J., Lotze, H., Rossmanith, P.: Online simple knapsack with reservation costs. In: Bläser, M., Monmege, B. (eds.) STACS 2021, March 16-19, 2021, Saarbrücken, Germany (Virtual Conference). LIPIcs, vol. 187, pp. 16:1–16:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021)
4. Böckenhauer, H., Komm, D., Královic, R., Rossmanith, P.: The online knapsack problem: Advice and randomization. Theor. Comput. Sci. **527**, 61–72 (2014)
5. Cygan, M., Jez, L., Sgall, J.: Online knapsack revisited. Theory Comput. Syst. **58**(1), 153–190 (2016)
6. Demko, S., Hill, T.P.: Equitable distribution of indivisible objects. Mathematical Social Sciences **16**(2), 145–158 (1988)
7. Han, X., Kawase, Y., Makino, K.: Online unweighted knapsack problem with removal cost. Algorithmica **70**(1), 76–91 (2014)
8. Han, X., Kawase, Y., Makino, K.: Randomized algorithms for online knapsack problems. Theor. Comput. Sci. **562**, 395–405 (2015)
9. Han, X., Kawase, Y., Makino, K., Guo, H.: Online removable knapsack problem under convex function. Theor. Comput. Sci. **540**, 62–69 (2014)
10. Han, X., Kawase, Y., Makino, K., Yokomaku, H.: Online knapsack problems with a resource buffer. In: Lu, P., Zhang, G. (eds.) ISAAC 2019, December 8-11, 2019, Shanghai University of Finance and Economics, Shanghai, China. LIPIcs, vol. 149, pp. 28:1–28:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2019)
11. Han, X., Makino, K.: Online removable knapsack with limited cuts. Theor. Comput. Sci. **411**(44-46), 3956–3964 (2010)
12. Iwama, K., Taketomi, S.: Removable online knapsack problems. In: ICALP 2002, Malaga, Spain, July 8-13, 2002, Proceedings. pp. 293–305 (2002)
13. Iwama, K., Zhang, G.: Online knapsack with resource augmentation. Inf. Process. Lett. **110**(22), 1016–1020 (2010)
14. Kellerer, H., Pferschy, U., Pisinger, D.: Knapsack Problems. Springer, Berlin, Germany (2004)
15. Lueker, G.S.: Average-case analysis of off-line and on-line knapsack problems. J. Algorithms **29**(2), 277–305 (1998)
16. Marchetti-Spaccamela, A., Vercellis, C.: Stochastic on-line knapsack problems. Math. Program. **68**, 73–104 (1995)
17. Zhou, Y., Chakrabarty, D., Lukose, R.M.: Budget constrained bidding in keyword auctions and online knapsack problems. In: Papadimitriou, C.H., Zhang, S. (eds.) WINE 2008, Shanghai, China, December 17-20, 2008. Proceedings. LNCS, vol. 5385, pp. 566–576. Springer (2008)