Theoretical Computer Science 508 (2013) 35–40

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

2D knapsack: Packing squares☆

Yan Lan [a], György Dósa [b], Xin Han [c,*], Chenyang Zhou [c], Attila Benko [b]

[a] *Dalian Neusoft Institute of Information, China*

[b] *Department of Mathematics, University of Pannonia, Veszprém, Hungary*

[c] *School of Software of Dalian University of Technology, China*

ABSTRACT

In this paper, we study a two-dimensional knapsack problem: packing squares as many as possible into a unit square. Our results are the following:

(i) we propose an algorithm called IHS (Increasing Height Shelf), and prove that the packing is optimal if in an optimal packing there are at most 5 squares, and this upper bound is sharp;
(ii) if all the squares have side length at most $\frac{1}{k}$, we propose a simple and fast algorithm with an approximation ratio $\frac{k^2+3k+2}{k^2}$ in time $O(n \log n)$;
(iii) we give an EPTAS for the problem, where the previous result in Jansen and Solis-Oba (2008) [16] is a PTAS, not an EPTAS. However our approach does not work on the previous model of Jansen and Solis-Oba (2008) [16], where each square has an arbitrary weight.

1. Introduction

The knapsack problem is one of the most classical and well studied problems in the combinatorial optimization field and has a lot of applications in the real world [17]. The (classical) knapsack problem is given a knapsack and a set of items with weights and sizes, to maximize the total weight of selected items in the knapsack satisfying the capacity constraint. In this paper, we study a geometric version of the 2D knapsack problem, where items are squares with weight 1 and side at most 1 and the knapsack is a unit size square and the objective is to maximize the total number of squares packed in the knapsack. In the packing, the sides of the items should be parallel to the corresponding sides of the knapsack and overlapping is not allowed. The problem was first studied by Baker et al. [2]. They gave an approximation algorithm with an asymptotic ratio 4/3. As mentioned in [16], this geometric packing problem has received a lot of attention recently [15,16,14,9,4], and has its applications in stock cutting, advertisement placement, image processing, and VLSI design [15,9].

Related work: It is well-known that the 1D knapsack problem is NP-hard and admits fully polynomial time approximation schemes (FPTAS) and the corresponding fractional problems can be solved by a greedy algorithm [1,5,11,17]. For the 2D geometric knapsack, in [3] Caprara and Monaci gave a simple algorithm with an approximation ratio $3+\epsilon$. Jansen and

☆ Partially supported by the NSFC (11101065) and "the Fundamental Research Funds for the Central Universities". The last author is supported in part by Project TAMOP-4.2.2/B-10/1-2010-0025.

* Corresponding author. Tel.: +86 411 87571630.
E-mail addresses: lanyan@neusoft.edu.cn (Y. Lan), dosagy@almos.vein.hu (G. Dósa), hanxin.mail@gmail.com (X. Han), zcy1988@gmail.com (C. Zhou), benko.attila@almos.vein.hu (A. Benko).

0304-3975/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2012.07.035

Zhang [14] improved the ratio to $(2 + \epsilon)$, where $\epsilon > 0$ can be arbitrarily small and they [15] also gave a simple $(2 + \epsilon)$-approximation algorithm for another version of the problem to maximize the total number of rectangles packed in a rectangular box.

For packing squares with arbitrary weights, Harren [9] gave a $(\frac{5}{4} + \epsilon)$ approximation algorithm. Then Jansen and Solis-Oba [16] proposed a $(1 + \epsilon)$ approximation algorithm, i.e., PTAS, not an EPTAS. If each item (square) has its weight equal to its area, Han et al. [6], Fishkin et al. [4] gave a PTAS independently. As for the online version of the knapsack problem, refer to the papers [10,7,8,19,18,12,13,20].

Our contributions: We first propose a simple and fast algorithm called IHS (Increasing Height Shelf), in which we sort all the items in ascending order of side length at first, then divide the knapsack into several layers and put as many items as possible into each layer, and prove that the packing by IHS is optimal if there are at most 5 squares packed in an optimal packing, and this upper bound of 5 is sharp; secondly we propose a modified IHS algorithm and prove that its approximation ratio is $\frac{k^2+3k+2}{k^2}$ if all the items have size (side length) at most $\frac{1}{k}$, where $k \geq 1$ is an integer; finally we give an EPTAS for the problem, which is simpler and more efficient than the previous result [16]. However our approach does not work on the previous model in [16], where each square has an arbitrary weight.

2. Preliminaries and models

In this section, we formally define our problem.

Packing Squares into a Knapsack
Input: a square knapsack with a unit size, and a set of square items $L = \{a_1, \ldots, a_n\}$.
Output: select a set of items $F \subseteq L$ with a maximum number of squares which can be packed into the knapsack subject to the following constraints:

1. there is no overlapping between any two items;
2. the sides of items are parallel to the corresponding sides of the knapsack.

We analyze algorithms by using one of the standards: the approximation ratio. Given an input sequence L, the *approximation ratio* of an approximation algorithm A is defined as follows:

$$R_A = \sup_L \frac{OPT(L)}{A(L)},$$

where $OPT(L)$ is the optimal value and $A(L)$ denotes the number of items packed by algorithm A. In this paper, we denote an item by a_i; we also use a_i to denote the side length of the item.

We can use the following two lemmas to estimate the optimal solution.

Lemma 1. *Assume $a_1 \leq a_2 \leq \cdots \leq a_n$ in the input L. If $\alpha \leq \sum_{i=1}^{k} a_i^2 \leq 1$, then $OPT(L) \leq k \cdot \frac{1}{\alpha}$.*

Proof. If $OPT(L) \leq k$ then the lemma holds. Otherwise assume $OPT(L) = j > k$. It is not difficult to see that the smallest j items $a_1, \ldots, a_j$ can be packed in the knapsack, where $j > k$. Then we have $\sum_{i=1}^{j} a_i^2 \leq 1$. Due to $a_1 \leq a_2 \leq \cdots \leq a_j$, and $\sum_{i=1}^{k} a_i^2 \geq \alpha$, then we have $\frac{\alpha}{k} \leq \frac{1}{j}$, since the average area of the first k items is not larger than the average area of the first j items. Hence this lemma holds. □

Lemma 2. *Assume $a_1 \leq a_2 \leq \cdots \leq a_n$ in the input L. If $\sum_{i=1}^{k+1} a_i^2 > 1$, then $OPT(L) \leq k$.*

Proof. If $OPT(L) \geq k + 1$ then we must have that: the smallest $k + 1$ squares can be packed together, i.e., $\sum_{i=1}^{k+1} a_i^2 \leq 1$, which causes a contradiction with the assumption. So, $OPT(L) \leq k$. □

3. A simple algorithm IHS and its applications

In this section, we first propose a simple algorithm called increasing height shelf (IHS), which runs in time $O(n \log n)$, where n is the number of items in the input L. An example of IHS packing is given in Fig. 1. Then we prove that (i) IHS is an optimal packing if $OPT(L) \leq 5$, and (ii) IHS is not optimal for some instance L if $OPT(L) \geq 6$. Finally, we analyze algorithm IHS and a modified IHS algorithm, and prove that both algorithms work very well when all the items are small; based on the Modified IHS we obtain a polynomial time approximation algorithm for the problem.

3.1. Increasing height shelf

If a rectangular box has length 1, we call it a *shelf*. The main ideas of shelf packing are (i) we cut the square knapsack into a set of boxes with length 1, i.e., a set of shelves, and (ii) in each shelf we pack square items in a greedy way, i.e., the bottoms of all the squares lie on the bottom of the shelf. The key point is how to select the heights of shelves. Our objective is to maximize the number of items packed in the knapsack, so it is very natural to consider small items first.

In the algorithm IHS, we first sort squares in ascending order of their side length. Then pack squares into shelves from small to large, then pack shelves into the knapsack until all the squares are packed or there is no room for a shelf in the knapsack. The details are given in Table 1.

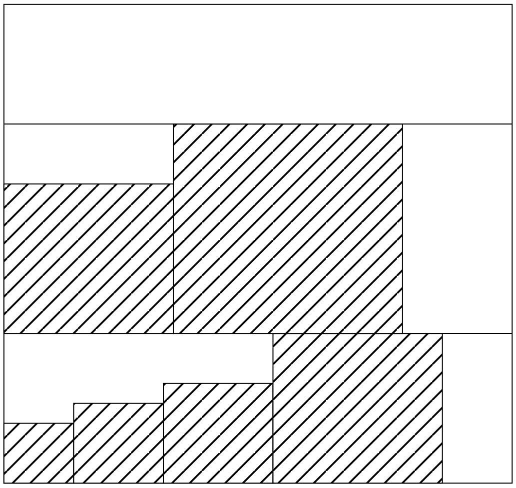

Fig. 1. Increasing height shelf.

Table 1
Algorithm: increasing height shelf.

1	Sort all the items in ascending order of their side length. Let $a_1 \le a_2 \le \cdots \le a_n$ be the items sorted.
2	Find a *largest* index i such that $\sum_{j=1}^{i} a_j \le 1$, pack these items into a shelf with width 1 and height a_i.
3	Pack the shelf into the knapsack: if there is no shelf in the knapsack, then pack the shelf on the bottom of the knapsack, else pack the shelf on the top of the last shelf in the knapsack, see Fig. 1. If the shelf exceeds over the top of the knapsack, then cancel packing the shelf and stop, else goto next step.
4	Delete all the squares just packed from the input list. If there are some squares unpacked, then rename all unpacked squares as $a_1 \le a_2 \le \cdots$ and goto step 2. Else stop.

Theorem 1. *Algorithm IHS is an optimal packing if $OPT(L) \le 5$, and IHS is not optimal for some instance L if $OPT(L) \ge 6$.*

Proof. For the positive result, we only prove the case: $OPT(L) = 5$ since the proof is similar for the other cases $OPT(L) \le 4$. Let $a_1 \le a_2 \le \cdots \le a_n$ be the input squares. When $OPT(L) = 5$, it is not difficult to see that the smallest squares a_1, $a_2, \ldots, a_5$ can be packed in the knapsack. Then for any $1 \le i < j \le 5$, we have

$$a_i + a_j \le 1, \tag{1}$$

otherwise squares a_i and a_j cannot be packed together. Due to $OPT(L) = 5$, we have the following result:

$$a_1 + a_2 + a_3 \le 1. \tag{2}$$

This result can be observed as below: given an optimal packing with $OPT(L) = 5$, by (1) we can repack the optimal packing like this, at each corner of the knapsack there is one square packed, and we do not change the positions of other squares in the optimal packing. Then for some $1 \le i < j < k \le 5$ we have $a_i + a_j + a_k \le 1$. Otherwise $OPT(L) \le 4$.

If $\sum_{i=1}^{5} a_i \le 1$ then we are done, since all the five items can be packed in a shelf of size $(1, a_5)$. Otherwise if $\sum_{i=1}^{4} a_i \le 1$ then the five items are packed into two shelves, that is, $a_1, \ldots, a_4$ are packed into a shelf of size $(1, a_4)$, and item a_5 is packed into a shelf of size $(1, a_5)$. By (1) we have $a_4 + a_5 \le 1$, then the two shelves can be packed into the knapsack. Otherwise if $\sum_{i=1}^{3} a_i \le 1$ then the five items are also packed into two shelves too, that is, $a_1, \ldots, a_3$ are packed into a shelf of size $(1, a_3)$, items a_4, a_5 are packed into a shelf of size $(1, a_5)$ due to $a_4 + a_5 \le 1$. By (1) we have $a_3 + a_5 \le 1$, then the two shelves can be packed into the knapsack.

Next we prove the negative result: when $OPT(L) \ge 6$, IHS may not produce an optimal packing. Consider the following input $L = \{a_1, \ldots, a_6\}$. Items $a_1, \ldots, a_5$ have size 1/3 and item a_6 has size 2/3. It is not difficult to see that all the items in L can be packed in the knapsack, whereas IHS only packs $a_1, \ldots, a_5$ into the knapsack. □

3.2. IHS algorithm for packing small items

In this subsection, we adopt algorithm IHS for packing small items and analyze its approximation ratio. In particular, if all the small items have size at most $1/k$, where integer $k \ge 3$, then the approximation ratio of IHS is at most $\frac{k^2}{(k-1)(k-2)}$.

Theorem 2. *When all the items have side length at most $1/k$, the approximation ratio of IHS is $\frac{k^2}{(k-1)(k-2)}$.*

Table 2
A modified IHS algorithm.

1. Sort all the items in ascending order of their side length, let $L = \{a_1, \ldots, a_n\}$ be items sorted, where $a_1 \le a_2 \le \cdots \le a_n$.
2. Construct an infeasible packing:
   (a) Find a *smallest* index u such that $\sum_{j=1}^{u} a_j \ge 1$, pack these items into a shelf with width $\sum_{j=1}^{u} a_j$ and height a_u;
   (b) Remove the items just packed from the input list, repeat the above packing, i.e., packing items into a shelf with width equal to or larger than 1 until all the items are packed into shelves.
3. Trim the infeasible packing:
   (a) Let S_k be the shelf generated in the k-th round. Assume the height of S_k is h_k.
   (b) Find a largest index i such that $\sum_{j=1}^{i} h_j \le 1$.
   (c) Shrink the width of each shelf to 1, if the last item is over-packed, then just remove the item. All the remaining items in shelves S_j for $1 \le j \le i$ form a feasible packing.

Proof. After applying IHS for packing items, if there is no item unpacked, then we are done and the approximation ratio of IHS is 1. Otherwise, assume there are j shelves packed in the knapsack, say $S_1, S_2, \ldots, S_j$, where shelf S_i has height h_i. According to algorithm IHS, we have

$$h_1 \le h_2 \le \cdots \le h_j \le \frac{1}{k}. \tag{3}$$

Then we have

$$\sum_{i=1}^{j} h_i > 1 - \frac{1}{k}, \qquad \sum_{i=1}^{j-1} h_i > 1 - \frac{2}{k} \tag{4}$$

otherwise shelf S_{j+1} would have been packed. Observe that for $2 \le i \le j$ in shelf S_i each square has side at least h_{i-1} and in the horizontal dimension the total size of side lengths is at least $1 - 1/k$, hence the total area of squares packed in shelf S_i is as least

$$h_{i-1} \cdot \left(1 - \frac{1}{k}\right). \tag{5}$$

Then by (5), (4) and (3), the total area of squares packed in $S_2, \ldots, S_k$ is at least

$$\left(1 - \frac{1}{k}\right) \sum_{i=1}^{j-1} h_i \ge \left(1 - \frac{1}{k}\right)\left(1 - \frac{2}{k}\right).$$

By Lemma 1, we have the approximation ratio of IHS is $\frac{k^2}{(k-1)(k-2)}$. Hence, the theorem holds. □

According to Theorem 2, IHS gets a pretty good result for small items. For instance, when $k = 10$, i.e. there is no item bigger than $\frac{1}{10}$, the approximation ratio is 1.389. But we cannot estimate the approximation ratio for $k = 1$ or $k = 2$.

3.3. A modified IHS algorithm for packing small items

In this subsection we propose a modified IHS with an upper bound $1 + \frac{3}{k} + \frac{2}{k^2}$, which is better than the upper bound of algorithm IHS. Also we can estimate the approximation ratio for $k = 1$ and $k = 2$. In the algorithm, there are two phases: in phase 1, we construct an infeasible packing based on the algorithm IHS (the infeasible packing helps us to estimate the upper bound of the optimal solution); then in phase 2, we discard some items to get a feasible packing such that the number of items discarded is small. The details are given in Table 2.

Theorem 3. *When all the items have side length at most $1/k$, the approximation ratio of Modified IHS is $1 + \frac{3}{k} + \frac{2}{k^2}$.*

Proof. Let $S_1, S_2, \ldots, S_m$ be the shelves generated in the phase of constructing an infeasible solution. Remember at Step 3(b), i is the maximal index such that $\sum_{j=1}^{i} h_j \le 1$, where $i \ge k$. Observe that in the vertical dimension we select the smallest i shelves in our solution, so the approximation factor in the vertical dimension is

$$\frac{m}{i}. \tag{6}$$

Let us consider the approximation factor in the horizontal dimension. When we construct a feasible solution from the infeasible solution, in each shelf we remove at most one item, i.e., in a shelf with j items packed, the approximation factor in the horizontal dimension is at most

$$\frac{j+1}{j} \le \frac{k+1}{k}, \tag{7}$$

where $j \ge k$ since each item has width at most $\frac{1}{k}$.

To prove this theorem, we have two cases.

Case 1 $m \le i+1$: The optimal value $OPT(L)$ is upper bounded by $\sum_{j=1}^{m} |S_j|$, where $|S_j|$ is the number of squares packed in S_j after the second phase. Based on the above observations, by (6) and (7) the approximation ratio is

$$\frac{m}{i} \times \frac{k+1}{k} \le \frac{i+1}{i} \times \frac{k+1}{k} \le \frac{(k+1)^2}{k^2} = 1 + \frac{2}{k} + \frac{1}{k^2},$$

since $i \ge k$.

Case 2 $m \ge i+2$: we can prove that, before we delete some squares from shelves in S_j, the total area of all the squares in $\cup_{j=1}^{i+2} S_j$ is at least 1. According to algorithm IHS, we have

$$h_1 \le h_2 \le \cdots \le h_{i+2} \le \frac{1}{k}. \tag{8}$$

And we have

$$\sum_{j=1}^{i} h_j \le 1, \qquad \sum_{j=1}^{i+1} h_j > 1, \tag{9}$$

otherwise shelf S_{i+1} would have been selected. Observe that for $2 \le j \le i+2$ in shelf S_j each square has side at least h_{j-1} and in the horizontal dimension the total size of side lengths is at least 1, hence the total area of squares packed in shelf S_j is as least

$$h_{j-1} \cdot 1. \tag{10}$$

Then by (10), (9) and (8), before we shrink the shelves, the total area of squares packed in $S_2, \ldots, S_{i+2}$ is at least

$$\sum_{j=1}^{i+1} h_j > 1.$$

So the total area of squares in $\cup_{j=1}^{i+2} S_j$ is larger than 1. By Lemma 2, $OPT(L)$ is upper bounded by the total number of squares packed in $\cup_{j=1}^{i+2} S_j$. Imagine that all the squares in $\cup_{j=1}^{i+2} S_j$ are packed into $i+2$ shelves. Of course this is an infeasible solution. Observe that in the vertical dimension we select the smallest i shelves in our solution, so the approximation factor in the vertical dimension is $\frac{i+2}{i} \le \frac{k+2}{k}$, where $i \ge k$. Let us consider the approximation factor in the horizontal dimension. When we construct a feasible solution from the infeasible solution, in each shelf we remove at most one item, i.e., in a shelf with j items packed, the approximation factor in the horizontal dimension is at most $\frac{j+1}{j} \le \frac{k+1}{k}$, where $j \ge k$ since each item has width at most $\frac{1}{k}$. Hence the approximation ratio of our algorithm is

$$\frac{k+2}{k} \times \frac{k+1}{k} = 1 + \frac{3}{k} + \frac{2}{k^2}.$$

Hence, the theorem holds. □

3.4. An efficient polynomial time approximation scheme

Though algorithms IHS and Modified IHS are good for packing small items, it is not enough to get an efficient polynomial time approximation scheme (EPTAS) for the general case. So we need other techniques which are described here.

Lemma 3 (*[14]*)**.** *Given a set S with c squares, we can verify whether all the squares in S can be packed into the knapsack in time* 4^{c^2}.

Proof. The following proof is from [14] at the bottom of page 331. For the sake of completeness, we give the proof again. The most naive way to find a feasible packing of c items (or determine that no such packing exists) is probably the following. Observe that if a packing exists, then there exists a packing in which each item is immediately adjacent to some item to its left (or to the bins' left border) and to some item below (or to the bottom of the bin). In such a packing, the x-coordinate (within the bin) of the bottom left corner of every item is the sum of widths of a subset of items, and similarly, the y-coordinate is the sum of heights of a subset of items. Thus there are at most $2^c \cdot 2^c$ potential positions for each item, and therefore at most 4^{c^2} possibilities to consider. □

The main ideas to produce an EPTAS are below: we first guess whether $OPT(L)$ is larger than a constant c; if $OPT(L) \le c$ then enumerate all the cases to get an optimal packing by Lemma 3; else remove large items then apply the Modified IHS algorithm for the remaining items. Next we give the details. Take a sufficiently small constant $0 < \epsilon \le 1/7$ such that $\frac{1}{\epsilon}$ is an integer. Let $k = \frac{1}{\epsilon}$.

Polynomial time approximation scheme
1. Guess whether $OPT(L) \geq k^3$ or not. This can be done by the following: take the smallest k^3-th squares, then verify all the k^3 squares can be packed together or not. 2. If $OPT(L) < k^3$ then find a maximal i such that the smallest i-th squares can be packed together. Output the packing and i. 3. Else $OPT(L) \geq k^3$, then apply the modified IHS for all the items with side at most ϵ.

Theorem 4. *The approximation ratio of our algorithm is* $1 + 5\epsilon$. *And the time complexity is* $O(n \log n + 4^{\epsilon^{-6}})$.

Proof. It is not difficult to see that if $OPT(L) < k^3$ we can get an optimal solution, i.e., the approximation ratio is 1.

Next we consider the case $OPT(L) \geq k^3$. If an item has side larger than $\epsilon = \frac{1}{k}$, then we call it *large* else *small*. Then the input L can be divided into sublists L_s for small items and L_b for large items. Then we have

$$k^3 \leq OPT(L) \leq OPT(L_s) + OPT(L_b) \leq OPT(L_s) + k^2.$$

Then we have

$$OPT(L_s) \geq OPT(L) - k^2 \geq (1 - \epsilon)OPT(L). \tag{11}$$

By Theorem 3, the solution by the modified IHS is at least $OPT(L_s)/(1 + 3\epsilon + 2\epsilon^2)$. By Eq. (11), the approximation ratio of our algorithm is $\frac{(1+3\epsilon+2\epsilon^2)}{1-\epsilon} \leq 1 + 5\epsilon$, where $\epsilon \leq 1/7$.

As the time complexity, in step 1 of our algorithm, by Lemma 3 the time complexity is $O(4^{k^6}) = O(4^{\epsilon^{-6}})$, where $k = \epsilon^{-1}$, in step 2, the time complexity is at most $O(k^3 \cdot 4^{k^6}) = O(4^{\epsilon^{-6}})$ by Lemma 3, and in step 3, the time complexity is $O(n \log n)$. □

Remarks. It seems that the upper bounds of both algorithms IHS and Modified IHS are not tight. So, it is possible to improve the upper bounds of the two algorithms further. We leave these problems as open questions.

Acknowledgments

The authors wish to thank the referees for their useful comments on the earlier draft of the paper.

References

[1] L.G. Babat, Linear functions on the N-dimensional unit cube, Doklady Akademii Nauk SSSR 222 (1975) 761–762. (Russian).
[2] B.S. Baker, A.R. Calderbank, E.G. Coffman Jr., J.C. Lagarias, Approximation algorithms for maximizing the number of squares packed into a rectangle, SIAM Journal on Algebraic and Discrete Methods 4 (3) (1983) 383–397.
[3] A. Caprara, M. Monaci, On the two-dimensional knapsack problem, Operations Research Letters 32 (2004) 5–14.
[4] A.V. Fishkin, O. Gerber, K. Jansen, R. Solis-Oba, Packing weighted rectangles into a square, MFCS (2005) 352–363.
[5] G. Gene, E. Levner, Complexity of approximation algorithms for combinatorial problems: a survey, ACM SIGACT News 12 (3) (1980) 52–65.
[6] X. Han, K. Iwama, G. Zhang, Online removable square packing, Theory of Computing Systems 43 (1) (2008) 38–55.
[7] X. Han, K. Makino, Online minimization knapsack problem, WAOA (2009) 182–193.
[8] X. Han, K. Makino, Online removable knapsack with limited cuts, Theoretical Computer Science 411 (44–46) (2010) 3956–3964.
[9] R. Harren, Approximation algorithms for orthogonal packing problems for hypercubes, Theoretical Computer Science 410 (44) (2009) 4504–4532.
[10] T. Horiyama, K. Iwama, J. Kawahara, Finite-state online algorithms and their automated competitive analysis, in: ISAAC, in: LNCS, vol. 4288, 2006, pp. 71–80.
[11] O.H. Ibarra, C.E. Kim, Fast approximation algorithms for the knapsack and sum of subset problems, Journal of the ACM 22 (1975) 463–468.
[12] K. Iwama, S. Taketomi, Removable online knapsack problems, in: Proc. ICALP, in: LNCS, 2380, 2002, pp. 293–305.
[13] K. Iwama, G. Zhang, Optimal resource augmentations for online knapsack, in: APPROX-RANDOM, 2007, pp. 180–188.
[14] K. Jansen, G. Zhang, Maximizing the total profit of rectangles packed into a rectangle, Algorithmica 47 (3) (2007) 323–342.
[15] K. Jansen, G. Zhang, Maximizing the number of packed rectangles, SWAT (2004) 362–371.
[16] K. Jansen, R. Solis-Oba, A polynomial time approximation scheme for the square packing problem, IPCO (2008) 184–198.
[17] H. Kellerer, U. Pferschy, D. Pisinger, Knapsack Problems, Springer, 2004.
[18] G.S. Lueker, Average-case analysis of off-line and on-line knapsack problems, in: Proc. Sixth Annual ACM-SIAM SODA, 1995, pp. 179–188.
[19] A. Marchetti-Spaccamela, C. Vercellis, Stochastic on-line knapsack problems, Mathematical Programming 68 (1, Ser. A) (1995) 73–104.
[20] J. Noga, V. Sarbua, An online partially fractional knapsack problem, in: ISPAN, 2005, pp. 108–112.