ON WEIGHTED RECTANGLE PACKING WITH LARGE RESOURCES*

Aleksei V. Fishkin,[1] Olga Gerber,[1] and Klaus Jansen[1]

[1]*University of Kiel*
Olshausenstr. 40, 24118 Kiel, Germany
{avf,oge,kj}@informatik.uni-kiel.de

Abstract We study the problem of packing a set of n rectangles with weights into a dedicated rectangle so that the weight of the packed rectangles is maximized. We consider the case of large resources, that is, the side length of all rectangles is at most 1 and the side lengths of the dedicated rectangle differ by a factor of at least $1/\varepsilon^4$, for a fixed positive $\varepsilon > 0$. We present an algorithm which finds a rectangle packing of weight at least $(1-\varepsilon)$ of the optimum in time polynomial in n. As an application we show a $(2+\varepsilon)$-approximation algorithm for packing weighted rectangles into k rectangular bins of size (a, b).

Keywords: Rectangle packing, approximation algorithms

Introduction

We address the following problem of packing rectangles with weights into a rectangle. We are given a dedicated rectangle R of width $a \geq 0$ and height $b \geq 0$, and a list L of n rectangles R_i $(i = 1, \dots, n)$ with widths $a_i \in (0, a]$, heights $b_i \in (0, b]$, and positive integral weights w_i. For a sublist $L' \subseteq L$ of rectangles, a *packing* of L' into the dedicated rectangle R is a positioning of the rectangles from L' within the area $[0, a] \times [0, b]$, so that all the rectangles of L' have disjoint interiors. Rectangles are not allowed to rotate. The goal is to find a sublist of rectangles $L' \subseteq L$ and its packing in R which maximizes the weight of packed rectangles, i.e., $\sum_{R_i \in L'} w_i$.

The above problem is a natural generalization of the knapsack problem to the two-dimensional version.

*Supported by EU-Projekt CRESCCO, Critical Resource Sharing for Cooperation in Complex Systems, IST-2001-33135, by EU Thematic Network APPOL, Approximation and Online Algorithms, IST-2001-30012 and by DFG-Graduiertenkolleg 357, Effiziente Algorithmen und Mehrskalenmethoden.

Related results. It is well-known that the knapsack problem is just weakly NP-hard [Garey and Johnson, 1979], and admits an FPTAS [Kellerer et al., 2004; Lawler, 1979]. In contrast, already the problem of packing squares with unit weights into a rectangle is strongly NP-hard [Baker et al., 1983]. So, the problem of packing rectangles with weights into a rectangle admits no FPTAS, unless $\mathrm{P} = \mathrm{NP}$.

From another side, one can also find a relation to strip packing: Given a list L of rectangles R_i $(i = 1, \ldots, n)$ with widths $a_i \in (0, 1]$ and positive heights $b_i \geq 0$ it is required to pack the rectangles of L into the vertical strip $[0, 1] \times [0, +\infty)$ so that the packing height is minimized. In particular, this also defines the problem of packing rectangles into a rectangle of fixed width and minimum height, or the well-known two-dimensional cutting stock problem [Gilmore and Gomory, 1965].

Of course, the strip packing problem is strongly NP-hard since it includes the bin packing problem as a special case. In fact many known simple strip packing ideas come from bin packing. The "Bottom-Left" heuristic has asymptotic performance ratio equal to 2 when the rectangles are sorted by decreasing widths [Baker et al., 1980]. In [Coffman et al., 1980] several simple algorithms were studied where the rectangles are placed on "shelves" using one-dimensional bin-packing heuristics. It was shown that the First-Fit shelf algorithm has asymptotic performance ratio of 1.7 when the rectangles are sorted by decreasing height (this defines the First-Fit-Decreasing-Height algorithm). The asymptotic performance ratio of the best heuristic was further reduced to $3/2$ [Sleator, 1980], then to $4/3$ [Golan, 1981] and to $5/4$ [Baker et al., 1981]. Finally, in [Kenyon and Rémila, 1996] it was shown that there exists an asymptotic FPTAS in the case when the side lengths of all rectangles in the list are at most 1. (In the above definition $a_i, b_i \in (0, 1]$ for all R_i.) For the absolute performance, the two best current algorithms have the same performance ratio 2 [Schiermeyer, 1994; Steinberg, 1997].

In contrast to knapsack and strip packing there are just few results known for packing rectangles into a rectangle. For a long time the only known result has been an asymptotic $(4/3)$-approximation algorithm for packing unweighted squares into a rectangle [Baker et al., 1983]. Only very recently in [Jansen and Zhang, 2004], several first approximability results have been presented for the packing rectangles with weights into a rectangle. The best one is a $(2 + \varepsilon)$-approximation algorithm.

Our results. In this paper we consider the case of so-called large resources, when the number of packed rectangles is relatively large. Formally, in the above formulation it is assumed that all rectangles R_i $(i = 1, \ldots, n)$ in the list L have widths and heights $a_i, b_i \in (0, 1]$, and the dedicated rectangle R has unit width $a = 1$ and quite a large height $b \geq 1/\varepsilon^4$, for a fixed positive

$\varepsilon > 0$. We present an algorithm which finds a sublist $L' \subseteq L$ of rectangles and its packing into the dedicated rectangle R with weight at least $(1-\varepsilon)\mathrm{OPT}$, where OPT is the optimum weight. The running time of the algorithm is polynomial in the number of rectangles n.

Our approach to approximation is as follows. At the beginning we take an optimal rectangle packing inside of the dedicated rectangle, considering it as a strip packing. We then perform several transformations that simplify the packing structure, without dramatically increasing the packing height and decreasing the packing weight, such that the final result is amenable to a fast enumeration. As soon as such a ”near-optimal” strip packing is found, we apply our shifting technique. This puts the packing into the dedicated rectangle by removing some less weighted piece of the packing.

Interestingly, by considering a weekly restricted case we are able to achieve the best possible approximation ratio. This makes a significant step in understanding of approximation properties of the problem. Furthermore, the difference in the side lengths of the dedicated rectangle and the rectangles in the list yields that the number of packed rectangles is large, that can be met quite often in practice. In order to be able to cope with the problem we also design several new approximation techniques, some of them are nice combinations of various classical techniques from knapsack and strip packing. This demonstrates quite a strong relation between several variants of packing.

Applications. There recently has been increasing interest in the advertisement placement problem for newspapers and the Internet [Adler et al., 1998; Freund and Naor, 2002]. In a basic version of the problem, we are given a list of n advertisements and k identical rectangular pages of fixed size (a, b), on which advertisements may be placed. Each ith advertisement appears as a small rectangle of size (a_i, b_i), and is associated with a profit p_i $(i = 1, \dots, n)$. Advertisements may not overlap. The goal is to maximize the total profit of the advertisements placed on all k pages.

This problem is also known as the problem of packing n weighted rectangles into k identical rectangular bins. Here, as an application of our algorithm, we provide a $(2+\varepsilon)$-approximation algorithm. The running time of the algorithm is polynomial in n for any fixed $\varepsilon > 0$.

Last notes. The paper is organized as follows. In section 1 we introduce notations and give some preliminary results. In Section 2, we present our shifting technique. In Section 3 we perform packing transformations. In Section 4 we outline the algorithm. Finally, in the last section we give an approximation algorithm to pack rectangles into k rectangular bins of size (a, b). Due to space limitations, some of the proofs are omitted.

1. Preliminaries

We are given a dedicated rectangle R of unit width $a = 1$ and height $b \geq 0$, and a list L of rectangles R_i $(i = 1, \ldots, n)$ with widths $a_i \in (0, 1]$, heights $b_i \in (0, 1]$, and positive integral weights w_i. The goal is to find a sublist of rectangles $L' \subseteq L$ and its packing in R which maximizes the weight of packed rectangles, i.e., $\sum_{R_i \in L'} w_i$.

We will use the following notations. For a sublist of rectangles $L' \subseteq L$, we will write $weight(L')$, $height(L')$, and $size(L')$ to denote the values of $\sum_{R_i \in L'} w_i$; $\sum_{R_i \in L'} b_i$, and $\sum_{R_i \in L'} a_i \cdot b_i$, respectively. Also, we will write $L^{opt} \subseteq L$ to denote an optimal sublist of rectangles, and OPT to denote the optimal objective value. Thus, $weight(L^{opt}) = \mathrm{OPT}$ and $size(L^{opt}) \leq a \cdot b = b$. Throughout of the paper we assume that $0 < \varepsilon < 1/2^{10}$, $1/\varepsilon' = (2 + \varepsilon)/\varepsilon$ is integral ($\varepsilon' = \varepsilon/(2 + \varepsilon)$), $m = 1/(\varepsilon')^2$, and the height value $b \geq 1/\varepsilon^4$.

1.1 Separating rectangles

Given a positive $\varepsilon' > 0$, we partition the list L of rectangles into two sublists: L_{narrow}, containing all the rectangles of width at most ε', and L_{wide}, containing all the rectangles of width larger than ε'.

1.2 Knapsack

In the knapsack problem we are given a knapsack capacity B and a set of items $I = \{1, 2, \ldots, n\}$, where each item $i \in I$ is associated with its size s_i and profit p_i. It is required to find a subset $I' \subseteq I$ which maximizes the profit of $\sum_{i \in I'} p_i$ subject to $\sum_{i \in I'} s_i \leq B$, i.e., it fits in a knapsack of size B.

The knapsack problem is NP-hard, but it admits an FPTAS [Garey and Johnson, 1979]. In particular, we can use any FPTAS version from [Kellerer et al., 2004; Lawler, 1979]. Given a precision $\delta > 0$, the algorithm outputs a subset $I(B) \subseteq I$ such that

$$\sum_{i \in I(B)} s_i \leq B \text{ and } \sum_{i \in I(B)} p_i \geq (1 - \delta)\mathrm{OPT}(I, B), \tag{1}$$

where $\mathrm{OPT}(I, B)$ is the maximum profit of I with respect to capacity B. For simplicity, we will write $KS(n, \delta)$ to denote the running time of the algorithm, which is polynomial in the number of items n and $1/\delta$.

1.3 Solving knapsacks with wide and narrow rectangles

We order all the wide rectangles in L_{wide} by non-increasing widths. W.l.o.g. we assume that there are n' wide rectangles $R_1 = (a_1, b_1)$, $R_2 = (a_2, b_2), \ldots$, $R_{n'} = (a_{n'}, b_{n'})$ with widths $a_1 \geq a_2 \geq \ldots \geq a_{n'} \geq \varepsilon'$. So, for any two $1 \leq k < \ell \leq n'$, let $L_{wide}(k, \ell)$ denote the list of all wide rectangles R_i in

L_{wide} with $\ell \geq i \geq k$. Here we work with rectangles as items. However, we treat narrow and wide rectangles differently. For wide rectangles, we only pay attention to the height values. For narrow rectangles, however, we only pay attention to the size values. By solving knapsack problems, we get the following result.

LEMMA 1 *Let H and S be some positive variables. Let $L_{wide}(k, \ell)$ be the list of wide rectangles between R_k and R_ℓ. Let L_{narrow} be the list of all narrow rectangles. Then, in $O(KS(n, \varepsilon^2)$ time we can find*

- *a sublist $L_{wide}(k, \ell, H) \subseteq L_{wide}(k, \ell)$ such that the height of $L_{wide}(k, \ell, H)$ is at most H and*

  $$weight(L_{wide}(k, \ell, H)) \geq (1 - \varepsilon^2/4)\mathrm{OPT}(L_{wide}(k, \ell), H),$$

  where $\mathrm{OPT}(L_{wide}(k, \ell), H)$ is the maximum weight of a subset of $L_{wide}(k, \ell)$ with a total height at most H.

- *a sublist $L_{narrow}(S) \subseteq L_{narrow}$ such that the size of $L_{narrow}(S)$ is at most S and*

  $$weight(L_{narrow}(S)) \geq (1 - \varepsilon^2/4)\mathrm{OPT}(L_{narrow}, S),$$

  where $\mathrm{OPT}(L_{narrow}, S)$ is the maximum weight of a subset of L_{narrow} with area at most S.

1.4 Packing narrow rectangles: NFDH

We consider the following strip-packing problem: Given a sublist $L' \subseteq L_{narrow}$ of narrow rectangles and a strip with fixed width $1 - c$ ($c \in [0, 1]$) and unbounded height, pack the rectangles of L' into the the strip such that the height to which the strip is filled is as small as possible.

First, we order the rectangles of L' by decreasing heights. Then, we put the narrow rectangles into the strip-packing by using Next-Fit-Decreasing-Height (NFDH): The rectangles are packed so as to form a sequence of sublevels. The first sublevel is just the bottom line of the strip. Each subsequent sublevel is defined by a horizontal line drawn through the top of the rectangle placed on the previous sublevel. Rectangles are packed in a left-justified greedy manner, until there is insufficient space to the right to place the next rectangle, at that point, the current sublevel is discontinued, the next sublevel is defined and packing proceeds on the new sublevel. For an illustration see Fig. 1.

We will use the following simple result.

LEMMA 2 *Let $L' \subseteq L_{narrow}$ be any sublist of narrow rectangles ordered by non-increasing heights. If the Next-Fit-Decreasing-Height (NFDH) heuristic outputs a packing of height $NFDH(L')$, then the area covered by the narrow rectangles $AREA \geq (1 - c - \varepsilon')(NFDH(L') - 1)$.*

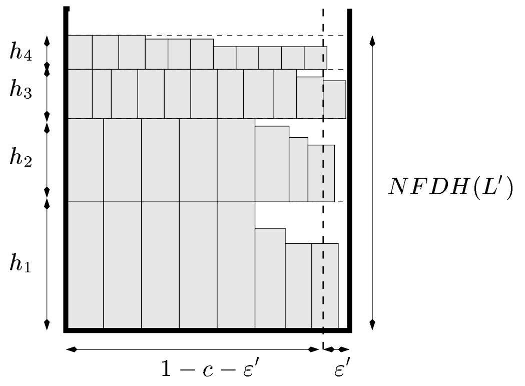

Figure 1. NFDH for narrow rectangles

1.5 Strip packing by KR-algorithm

We consider the following strip-packing problem: Given a sublist $L' \subseteq L$ of rectangles and a strip with unit width and unbounded height, pack the rectangles of L' into the the strip such that the height to which the strip is filled is as small as possible.

As we mentioned before the strip packing problem admits an asymptotic FPTAS. We will use the following result.

THEOREM 3 (KENYON AND RÉMILA, 1996) *There is an algorithm A which, given an accuracy $\varepsilon > 0$, a sublist $L' \subseteq L$ of rectangles and a strip with unit width 1 and unbounded height, packs the rectangles of L' into the the strip such that the height to which the strip is filled*

$$A(L') \leq (1+\varepsilon)strip(L') + O(1/\varepsilon^2), \tag{2}$$

where $strip(L')$ denotes the height of the optimal strip packing of L'. The running time of A is polynomial in n and $1/\varepsilon$.

For simplicity, we name such an algorithm in the theorem by the KR-algorithm. Also, we will write $KR(n, \varepsilon)$ to denote its running time. In Section 3 we will give more details on packing by the KR-algorithm.

2. Shifting

Assume that we are given a strip packing of height $(1 + O(\varepsilon))b$ for a list of rectangles whose weight is at least $(1 - O(\varepsilon))\mathrm{OPT}$. The idea of our shifting technique is to remove some less weighted piece of height $O(\varepsilon)b$. Then, the weight value remains $(1 - O(\varepsilon))\mathrm{OPT}$, but the height value reduces to b. This fits into the area of the dedicated rectangle $R = (a, b)$.

LEMMA 4 *Suppose we are given a strip packing of height $(1+\delta_2)b$ for a sublist $L' \subseteq L$ with weight at least $(1-\delta_1)OPT$, for some $\delta_1, \delta_2 \approx O(\varepsilon)$. If*

$$\left\lceil \frac{1}{\delta_1} \right\rceil \leq \left\lfloor \frac{(1+\delta_2)+2}{\delta_2 \cdot b + 2} \right\rfloor, \tag{3}$$

then in $O(n + 1/\varepsilon)$ time we can find a rectangle packing of a sublist of L' into the area of the dedicated rectangle $R = (a, b)$ with the weight value at least $(1-3\delta_1)\mathrm{OPT}$.

3. Transformations of optimal solution

Here we discuss some transformations which simplify the structure of the optimal solution L^{opt}. We start with transforming a packing of L^{opt} into a well structured packing. This introduces the lists L^{opt}_{wide} of wide rectangles, L^{opt}_{narrow} of narrow rectangles, and m optimal threshold rectangles. Next, assuming the m threshold rectangles and the m height capacity values are known, we perform a transformation of the optimal lists L^{opt}_{wide} and L^{opt}_{narrow} to some lists found by solving a series of knapsacks. Then, we perform a rounding transformation which turns all the m height capacity values to some discrete points. Each of these transformations may increases the height value by $O(\varepsilon b)$, and may decrease the weight value by $O(\varepsilon \mathrm{OPT})$. However, in the next section we show that L^{opt} can be still approximated with quite a good precision.

3.1 Well-structured packing

Here we describe a well structured packing of the optimal solution.

Separation. Let L^{opt} be the optimal solution. We define the lists of narrow and wide rectangles: $L^{opt}_{narrow} = L^{opt} \cap L_{narrow}$ and $L^{opt}_{wide} = L^{opt} \cap L_{wide}$. Clearly, $weight(L^{opt}_{wide}) + weight(L^{opt}_{narrow}) = \mathrm{OPT}$.

Threshold rectangles. Let $R_{k_1} = (a_{k_1}, b_{k_1}), R_{k_2} = (a_{k_2}, b_{k_2}), \ldots, R_{k_m} = (a_{k_m}, b_{k_m})$ be a sequence of optimal wide rectangles in L^{opt}_{wide} such that $1 \leq k_1 < k_2 < \ldots < k_m \leq n'$. Then, we call such rectangles as the threshold rectangles. As it is defined, widths $a_{k_1} \geq a_{k_2} \geq \ldots \geq a_{k_m} \geq \varepsilon'$.

Configurations. Now we can define configurations. A configuration is defined as a multi-set of widths chosen among the m threshold widths in $\{a_{k_i} | i = 1, \ldots, m\}$ which sum to at most 1, i.e. they may occur at the same level. Their sum is called the width of the configuration.

Layers. Let q be some positive integer. Let $C_1, C_2, \ldots, C_q$ be some distinct configurations, numbered by non-increasing widths, and let C_{q+1} be an empty configuration. Let α_{ij} denote the number of occurrences of width a_{k_i} in C_j. Then, the value of $c_j = \sum_{i=1}^{m} a_{k(ij)}\alpha_{ij}$ is called the width of C_j. Therefore, $c_1 \geq c_2 \geq \ldots \geq c_q \geq c_{q+1} = 0$.

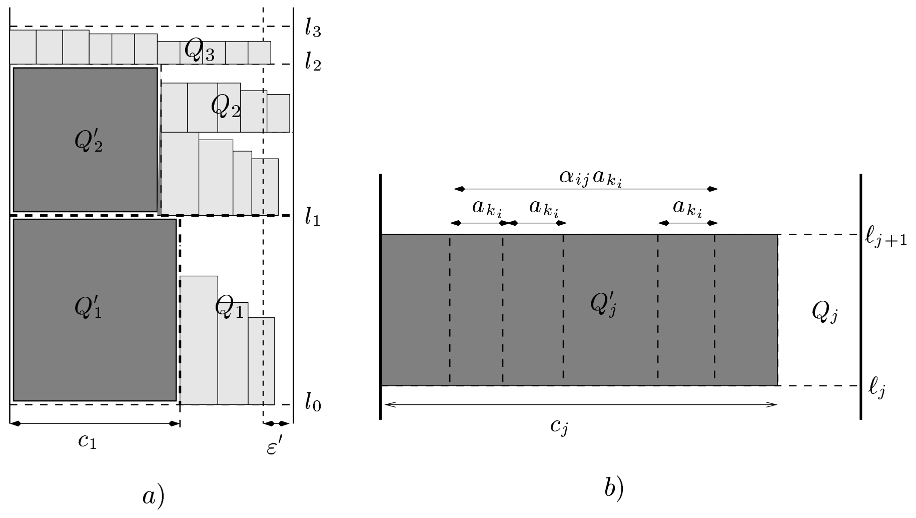

Figure 2. a) A well structured packing with 3 layers; b) Structure of layer $[0,1] \times [\ell_j, \ell_{j+1}]$

Let $0 = \ell_0 \leq \ell_1 \leq \ldots \leq \ell_q \leq \ell_{q+1} = h$ be some $q+1$ non-negative values. We define $q+1$ layers as follows. The layer $[0,1] \times [\ell_j, \ell_{j+1}]$ ($j = 0, \ldots, q+1$) corresponds to configuration C_j. It is divided into two rectangles: $Q_j = [c_j, 1] \times [\ell_j, \ell_{j+1}]$ and $Q'_j = [0, c_j] \times [\ell_j, \ell_{j+1}]$. (Notice that the last layer is $Q_{q+1} = [0,1] \times [\ell_q, \ell_{q+1}]$, as shown in Fig. 2a)

From one side, all Q_j ($j = 1, \ldots, q+1$) are empty. From another side, each Q'_j ($j = 1, \ldots, q$) consists of m vertical multi-slices, each ith of those with exactly α_{ij} identical slices of width a_{k_i}, as shown in Fig. 2b. The value of $(\ell_{j+1} - \ell_j)$ defines the height of configuration C_j, and the value of $h = \ell_{q+1}$ defines the packing height. The value of $H_i = \sum_{j=1}^{q} \alpha_{ij}(\ell_{j+1} - \ell_j)$ defines the total height of all slices of width a_{k_i}, and it is called the ith threshold capacity.

Well-structured packing. A strip packing of the optimal solution L^{opt} is called a well-structured strip packing with $q+1$ layers if all Q_j ($j = 1, \ldots, q+1$) are filled by narrow rectangles, and all the slices of width a_{k_i} ($i = 1, \ldots, m$) are greedily filled by the wide rectangles from $L^{opt} \cap L_{wide}(k_i, k_{i+1} - 1)$. (Here and further we assume w.l.o.g. that $k_{m+1} - 1 = n'$.) Now we are ready to give the following result.

THEOREM 5 (KENYON AND RÉMILA, 1996) *There exists a well-structured packing of L^{opt} with $2m+1$ layers such that its height $h \leq \max\{strip(L^{opt}_{wide})$*

$(1+1/(m\varepsilon'))+2m+1, size(L^{opt})(1+1/(m\varepsilon'))/(1-\varepsilon')+4m+1\}$*, where* $strip(L^{opt}_{wide})$ *is the height of the optimal strip packing of* L^{opt}_{wide}.

3.2 Augmentation

Now we can give the following simple result.

LEMMA 6 *If* $\varepsilon' = \varepsilon/(2+\varepsilon)$, $m = (1/\varepsilon')^2$, $\varepsilon < 1/2^{10}$ *and* $b \geq 1/\varepsilon^4$*, then there exists a well-structured packing with* $2m+1$ *layers of the optimal solution* L^{opt} *of height* $h \leq (1+2\varepsilon)b$.

3.3 Approximating wide rectangles

Our idea is to guess most profitable rectangles, knowing the optimal threshold rectangles and capacity values. Let R_{k_i} and H_i ($i = 1, \ldots, m$) be the optimal ith threshold rectangle and capacity, respectively. Then, by solving a series of knapsacks we can find the lists $L_{wide}(k_i, k_{i+1}-1, H_i)$ of wide rectangles. These are quite good approximations for lists $L_{wide}(k_i, k_{i+1}-1) \cap L^{opt}$, and hence all together they give a good approximation of the optimal list L^{opt}_{wide} of wide rectangles.

LEMMA 7 *The value of*

$$\sum_{i=1}^{m} weight(L_{wide}(k_i, k_{i+1}-1, H_i)) \geq (1-\varepsilon^2/4) weight(L^{opt}_{wide}). \qquad (4)$$

If the wide rectangles of L^{opt}_{wide} *are replaced by the rectangles of all lists* $L_{wide}(k_i, k_{i+1}-1, H_i)$ *(*$i = 1, \ldots, m$*), then the height* h *of the well-structured packing increases by at most* $\Delta_{wide} \leq \varepsilon b$.

Proof. As it was defined, $L_{wide}(k_i, k_{i+1}-1) \cap L^{opt} \subseteq L_{wide}(k_i, k_{i+1}-1)$. In the well structured packing, the rectangles of $L_{wide}(k_i, k_{i+1}-1) \cap L^{opt}$ are placed in the slices of width a_{k_i}. The total height of all these slices is exactly the value of H_i. So, $height(L_{wide}(k_i, k_{i+1}-1) \cap L^{opt}) \leq H_i$. Hence, by Lemma 1 solving m knapsack problems we can decrease the weight by at most some factor of $(1-\varepsilon^2/4)$.

Notice that both $L_{wide}(k_i, k_{i+1}-1, H_i)$ and $L_{wide}(k_i, k_{i+1}-1) \cap L^{opt}$ have quite similar characteristics. We use it as follows. We take the well-structured packing of L^{opt} and go over all the rectangles $Q'_1, Q'_2, \ldots, Q'_{2m}$ in the $2m$ layers. Inside all the slices of widths a_{k_i} ($i = 1, \ldots, m$) we replace the rectangles of $L_{wide}(k_i, k_{i+1}-1) \cap L^{opt}$ by the rectangles of $L_{wide}(k_i, k_{i+1}-1, H_i)$ in a greedy manner.

Since we greedily place rectangles, it may happen that some rectangles do not fit completely into the slices. We then increase the height of each layer by 1, that must create enough space for all rectangles. Since there are $2m$ layers, the height h increases by at most $\Delta_{wide} \leq 2m \leq \varepsilon b$, for $\varepsilon < 1/2^{10}$ and $b \geq 1/\varepsilon^4$. The result of lemma follows.

3.4 Approximating narrow rectangles

We use a similar idea to guess most profitable narrow rectangles, knowing the optimal configurations with heights and widths. Let c_j and ℓ_j ($i = 1, \ldots, 2m+1$) be the width and height of configuration C_j, respectively. Recall that the optimal narrow rectangles of L^{opt}_{narrow} are placed in rectangles $Q_1, Q_2, \ldots, Q_{2m}, Q_{2m+1}$. Hence we can bound the size value

$$size(L^{opt}_{narrow}) \leq \sum_{j=1}^{2m+1} (1 - c_j)(\ell_{i+1} - \ell_i). \tag{5}$$

So, by solving the knapsack problem we can find the list $L_{narrow}(S)$ of narrow rectangles, where the value of knapsack capacity

$$S = \sum_{j=1}^{2m+1} (1 - c_j)(\ell_{j+1} - \ell_j). \tag{6}$$

This is a good approximation of the optimal list L^{opt}_{narrow} of narrow rectangles.

LEMMA 8 *The value of $weight(L_{narrow}(S)) \geq (1-\varepsilon^2/4)weight(L^{opt}_{narrow})$. If the narrow rectangles of L^{opt}_{narrow} are replaced by the narrow rectangles $L_{narrow}(S)$, then the height h of the well-structured packing increases by at most $\Delta_{narrow} \leq 2\varepsilon b$.*

Proof. Clearly, the rectangles of L^{opt}_{narrow} must be in L_{narrow}. By (5), the area of L^{opt}_{narrow} is at most S. Hence, by Lemma 1 solving the knapsack problem can only decrease the weight by some factor of $(1 - \varepsilon^2/4)$. So, we get the weight at least $(1 - \varepsilon^2/4)weight(L^{opt}_{narrow})$.

Notice that both L^{opt}_{narrow} and $L_{narrow}(S)$ have quite similar characteristics. We use it as follows. We go over the rectangles $Q_1, Q_2, \ldots, Q_{2m}, Q_{2m+1}$ in the $2m+1$ layers, and place the rectangles of $L_{narrow}(S)$ by using NFDH. If not all rectangles are placed, then we work with a new layer of width 1 and height Δ_{narrow}.

The new rectangle has width 1 and height Δ_{narrow}. Similar to Lemma 2, the area covered by narrow rectangles in additional layer is at least $(1-\varepsilon')(\Delta_{narrow} - 1)$. Similarly, consider the narrow rectangles packed in rectangle Q_j ($j = 1, \ldots, 2m+1$). The height of this packing is at least $\ell_{j+1} - \ell_j - 1$. The width of Q_j is $1 - c_j$. Hence, the area covered by the narrow rectangles is at least $(1 - c_j - \varepsilon')(\ell_{j+1} - \ell_j - 2)$. Combining over all layers, the area covered is at least $\sum_{j=1}^{2m+1}(1 - c_j - \varepsilon')(\ell_{j+1} - \ell_j - 2) + (1 - \varepsilon')(\Delta_{narrow} - 1)$.

Recall that the area of $L^{opt}_{narrow}(S)$ is at most $S = \sum_{j=1}^{2m+1}(1-c_j)(\ell_{j+1}-\ell_j)$. We need an upper bound on the value of Δ_{narrow}. So, it is enough to require that this size value is equal to the above bound. So, $\sum_{j=1}^{2m+1}(1-c_j-\varepsilon')(\ell_{j+1}-\ell_j-2) + (1-\varepsilon')(\Delta_{narrow}-1) \leq \sum_{j=1}^{2m+1}(1-c_j)(\ell_{j+1}-\ell_j)$. Hence, $(1-\varepsilon')(\Delta_{narrow}-1) \leq 2\sum_{j=1}^{2m+1}(1-c_j-\varepsilon') + \varepsilon'\sum_{j=1}^{2m+1}(\ell_{j+1}-\ell_j)$ and from $\sum_{j=1}^{2m+1}(\ell_{j+1}-\ell_j) = h$

$$\Delta_{narrow} \leq 1 + [2\sum_{j=1}^{2m+1}(1-c_j-\varepsilon') + \varepsilon' \cdot h]/(1-\varepsilon') \leq 2\varepsilon b$$

for $\varepsilon < 1/2^{10}$, $m = 1/(\varepsilon')^2$, $\varepsilon' = \varepsilon/(2+\varepsilon)$ and $b \geq 1/\varepsilon^4$. The result of lemma follows.

3.5 Rounding

Finally, we round all values to some discrete points.

LEMMA 9 *If we round up each threshold capacity H_i ($i = 1, \ldots, m$) in $L_{wide}(k_i, k_{i+1}-1, H_i)$ to the the closest value in*

$$CAPACITY = \{t \cdot (\varepsilon')^4 \cdot b | t = 1, 2, \ldots, 1/(\varepsilon')^6\},$$

and the value of S in $L_{narrow}(S)$ to the closest value in

$$SIZE = \{t \cdot (\varepsilon')^4 \cdot b | t = 1, 2, \ldots, 1/(\varepsilon')^5\},$$

then the height h of the well-structured packing increases by at most $\Delta_{rounding} \leq \varepsilon b$.

Proof. Consider a well structured packing of all $L_{wide}(k_i, k_{i+1}-1, H_i)$ and $L_{narrow}(S)$ with $2m+1$ layers. Each layer is cut into slices which correspond to a configuration. The wide rectangles of $L_{wide}(k_i, k_{i+1}-1, H_i)$ are packed in the slices of width a_{k_i} in a greedy manner. The rectangles of $L_{narrow}(S)$ are packed by the NFDH heuristic. The height of the packing is

$$h + \Delta_{wide} + \Delta_{narrow} \leq (1+5\varepsilon)b.$$

By rounding, we increase the value of each H_i and S by at most $(\varepsilon')^4 b$. Hence, in solving knapsacks the height of $L_{wide}(k_i, k_{i+1}-1, H_i)$ increases by at most $(\varepsilon')^4 b$, and the area of $L_{narrow}(S)$ increases by at most $(\varepsilon')^4 b$. Next, we proceed as in approximating wide and narrow rectangles. We go over all slices of width a_{k_i} and replace all old wide rectangles by the new wide rectangles in $L_{wide}(k_i, k_{i+1}-1, H_i)$. Also, we go over all layers and replace all old narrow rectangles by the new narrow rectangles in $L_{narrow}(S)$.

In order to accommodate all of wide and narrow rectangles we need to increase the heights of some layers (configurations). We can estimate the total increase as follows. First, we increase the height value of each layer (configuration) by $(\varepsilon')^4 b$. Then, similar to approximating wide and narrow rectangles, we can pack all the rectangles, but cutting them if they do not fit into slices or layers. Since the height value of any rectangle is at most 1, we simply increase the height of each layer by 1. This eliminates cuts. In overall, we can estimate the total increase as

$$\Delta_{rounding} \leq (2m+1)[(\varepsilon')^4 b + 1] = O(\varepsilon^2 b) \leq \varepsilon b,$$

for $m = 1/(\varepsilon')^2$, $\varepsilon' = \varepsilon/(2+\varepsilon)$, $\varepsilon \leq 1/2^{10}$ and $b \geq 1/\varepsilon^4$.

The height of the final packing is at most $(1+5\varepsilon)b + \Delta_{rounding} = (1+6\varepsilon)b$. This means that the size of all $L_{wide}(k_i, k_{i+1}-1, H_i)$ and $L_{narrow}(S)$ is at most $(1+6\varepsilon)b$. Hence, after rounding the value of S is at most $(1+6\varepsilon)b \leq b/\varepsilon'$. Since the width value of the rectangles in $L_{wide}(k_i, k_{i+1}-1, H_i)$ is at least ε', after rounding the value of H_i can be at most $(1+6\varepsilon)b/\varepsilon' \leq b/(\varepsilon')^2$. Thus, the value of t in $CAPACITY$ and $SIZE$ can be at most $1/(\varepsilon')^5$ and $1/(\varepsilon')^6$, respectively. The result of lemma follows.

4. Overall algorithm

Here we outline our algorithm and summarize all above results. We simply enumerate all possible sequences of threshold rectangles and their capacity values. Then, we solve a series of knapsack problems to get several lists of wide and narrow rectangles, and find a packing for them by using the KR-algorithm. At the end, we select the most profitable packing and apply the shifting technique to it. The final packing fits into the dedicated rectangle and its weight is near-optimal.

Rectangle Packing (RP):
Input: List L, accuracy $\varepsilon > 0$, and $\varepsilon' = \varepsilon/(2+\varepsilon)$, $m = 1/(\varepsilon')^2$.

1 Split L into L_{narrow} and L_{wide} of narrow and wide rectangles, whose widths are at most ε' and larger than ε';

2 Sort the wide rectangles of L_{wide} according to their widths;

3 For each sequence of $m = (1/\varepsilon')$ wide threshold rectangles $R_{k_1}, R_{k_2}, \ldots,$ R_{k_m} from L_{wide}:

   (a) select m capacity values of $H_i \in CAPACITY$ and a value of $S \in SIZE$;

   (b) find m lists $L_{wide}(k_i, k_{i+1}-1, H_i)$ and list $L_{narrow}(S)$;

   (c) run the KR-algorithm and keep the solution (if it's height is at most $(1+16\varepsilon)b$).

4 Select a packing whose weight is maximum;

5 Apply the shifting technique.

We conclude with the following final result.

THEOREM 10 *The* RP*-algorithm outputs a rectangle packing of a sublist* $L' \subseteq L$ *in the area* $[0,a] \times [0,b]$ *of the dedicated rectangle* R*. The weight of the packing* $weight(L') \geq (1-\varepsilon)\mathrm{OPT}$*, where* OPT *is the optimal weight. The running time of the* RP*-algorithm is bounded by* $O(n^{1/\varepsilon^2}(1/\varepsilon^6)^{1/\varepsilon^2+1}[KS(n,\varepsilon) \cdot KR(n,\varepsilon)])$*, where* $KS(n,\varepsilon)$ *is the running time of a FPTAS for solving the knapsack problem, and* $KR(n,\varepsilon)$ *is the running time of the KR-algorithm.*

5. Packing into k rectangular bins

Here we consider the problem of packing weighted rectangles into k bins. Given k identical bins of size (a,b) and a list L of n rectangles R_i $(i = 1,\ldots,n)$ with widths $a_i \in (0,a]$, heights $b_i \in (0,b]$, and positive integral weights w_i. The goal is to find a sublist $L' \subseteq L$ of rectangles and its packing into k bins such that the total weight of packed rectangles is maximized. We present the following algorithm:
Algorithm k-Bins:
Input: List L, accuracy $\varepsilon > 0$, k bins of size (a,b).

Case 1. $k \leq O(1/\varepsilon^4)$. Use a $(2+\varepsilon)$-approximation algorithm, that generalizes an approximation algorithm for one bin [Jansen and Zhang, 2004] to a constant number of bins (for the details we refer to a full version of this paper).

Case 2. $k > O(1/\varepsilon^4)$.

1 Take all k bins together to get the rectangle (a,kb).

2 Apply our algorithm with the PTAS to pack a subset of rectangles into a larger rectangle (a,kb), that gives us a packing with the total profit $\geq (1-\varepsilon)\mathrm{OPT}$.

3 Take the current rectangle packing. Draw $(k-1)$ vertical lines which divide the packing into k bins.

4 Split this packing into 2 solutions:

   (a) solution, which contains all rectangles which lie inside of each bin.

   (b) solution, which contains all rectangles which intersect any dividing line between two bins.

5 Take the solution which has the highest profit.

We can conclude with the following result.

THEOREM 11 *The algorithm k-Bins is a $(2+\varepsilon)$-approximation algorithm. Its running time is polynomial in the number of rectangles n for any fixed $\varepsilon > 0$.*

References

Adler, M., Gibbons, P., and Matias, Y. (1998). Scheduling space-sharing for internet advertising. Journal of Scheduling (to appear).

Baker, B., Brownand, D., and Katseff, H. (1981). A 5/4 algorithm for two dimensional packing. *J. of Algorithms*, 2:348–368.

Baker, B., Calderbank, A., Coffman, E., and Lagarias, J. (1983). Approximation algorithms for maximizing the number of squares packed into a rectangle. *SIAM Journal on Algebraic and Discrete Methods*, 4:383–397.

Baker, B., Coffman, E., and Rivest, R. (1980). Orthogonal packings in two dimensions. *SIAM J. Comput.*, 9:846–855.

Coffman, E., Garey, M., Johnson, D., and Tarjan, R. (1980). Performance bounds for level-oriented two-dimensional packing algorithms. *SIAM J. Comput.*, 9:808–826.

Freund, A. and Naor, J. (2002). Approximating the advertisement placement problem. In *Proceedings of the 9th Conference on Integer Programming and Combinatorial Optimization (IPCO'02)*, LNCS 2337, pages 415–424.

Garey, M. R. and Johnson, D. S. (1979). *Computers and intractability: A guide to the theory of NP-completeness*. Freeman, San Francisco, CA.

Gilmore, P. and Gomory, R. (1965). Multistage cutting stock problems of two and more dimensions. *Operations Research*, 13:94–120.

Golan, I. (1981). Performance bounds for orthogonal, oriented two-dimensional packing algorithms. *SIAM J. Comput.*, 10:571–582.

Jansen, K. and Zhang, G. (2004). On rectangle packing: maximizing benefits. In *Fifteenth Annual Symposium on Discrete Algorithms*, pages 197–206.

Kellerer, H., Pferschy, U., and Pisinger, D. (2004). *Knapsack problems*. Springer.

Kenyon, C. and Remila, E. (1996). Approximate strip-packing. In *Thirty-Seventh Annual Symposium on Foundations of Computer Science*, pages 31–36.

Lawler, E. (1979). Fast approximation algorithms for knapsack problems. *Mathematics of Operations Research*, 4:339–356.

Schiermeyer, I. (1994). Reverse fit : a 2-optimal algorithm for packing rectangles. *Proceedings 2nd European Symposium on Algorithms*, pages 290–299.

Sleator, D. (1980). A 2.5 times optimal algorithm for bin packing in two dimensions. *IPL*, (10):37–40.

Steinberg, A. (1997). A strip-packing algorithm with absolute performance bound 2. *SIAM Journal on Computing*, 26(2):401–409.