On Two Dimensional Packing *

Yossi Azar† *Leah Epstein*‡

April 29, 1996

Abstract

The paper considers *packing of rectangles* into an infinite bin. Similar to the *Tetris game*, the rectangles arrive from the top and, once placed, cannot be moved again. The rectangles are moved inside the bin to reach their place. For the case in which rotations are allowed, we design an algorithm whose performance ratio is constant. In contrast, if rotations are not allowed, we show that no algorithm of constant ratio exists. For this case we design an algorithm with performance ratio of $O(\log\frac{1}{\epsilon})$, where ϵ is the minimum width of any rectangle. We also show that no algorithm can achieve a better ratio than $\Omega(\sqrt{\log\frac{1}{\epsilon}})$ for this case.

1 Introduction

In this paper we consider the problem of orthogonal packing. We are given a bin with a fixed width (assuming equal to 1, without loss of generality) and an unbounded height, and also a set of open oriented rectangles. The rectangles have to be placed in the bin with the bottom of the rectangle parallel to the bottom of the bin. The spaces occupied by different rectangles may not overlap. The original rectangle packing problem was first proposed in 1980, by Baker et al. [1]. They present an approximation algorithm that achieves a performance ratio of 3 for rectangles and 2 for squares. Coffman et al. [8] present various algorithms with better performance ratio, and in particular split fit that has ratio of 1.5. We discuss the version of the problem, in which each rectangle has to be placed before the next one arrives.

There are a few models, which differ in the allowed movements inside the bin. In the original two-dimensional packing problem, a rectangle could be placed directly in any free space of the size of the rectangle, the concept of getting into a place was not discussed. In contrast, in our models, a rectangle arrives from the top as in the Tetris game, and should

*This work was submitted as part of the M.Sc. thesis of the second author

†Dept. of Computer Science, Tel-Aviv University. E-Mail: azar@math.tau.ac.il. Research supported in part by Allon Fellowship and by the Israel Science Foundation administered by the Israel Academy of Sciences.

‡Dept. of Computer Science, Tel-Aviv University. E-Mail: lea@math.tau.ac.il

be moved continuously around only in the free space until it reaches its place, (see figure 1), and then cannot be moved again. Moreover, it is possible either to allow rotations or not. If rotations are allowed, the rectangle may be rotated at any stage, and can be assigned on its side or on its bottom. If rotations are not allowed, the rectangle may only be moved with its bottom parallel to the bottom of the bin. It is also possible, to consider the integer case in which all widths and heights of rectangles are integers. In this case, the width of the bin is also an integer.

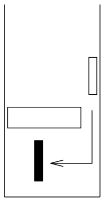

Figure 1: The narrow rectangle moves continuously in the unoccupied space to its place.

We note that our model in which the rectangles are not assigned directly, but should reach their place in a free space movement, has different applications. One possible application for this model is loading trucks with boxes, since while loading a truck, it is impossible to put anything under a large box that has already been packed (see figure 2). Also, the integer model with free space movement and rotations resembles the well known Tetris game. The main difference between the game, and our models is the goal. Our goal is to minimize the total height used, and in the Tetris game, the goal is, roughly, to maximize the number of full rows.

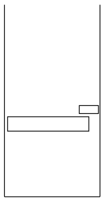

Figure 2: The big rectangle blocks the smaller rectangle

The cost of the algorithm is measured by the height of the packing, i.e. the maximum height, measured from the bottom of the bin, that is occupied by some rectangle. As usual, we try to compute the ratio between the cost of the on-line packing and the optimal off-line packing (the best packing that can be done for this sequence of rectangles).

The main results presented in this paper are as follows:

- A 4 competitive algorithm for the model with rotation.
- An $\Omega(\sqrt{\log \frac{1}{\epsilon}})$ lower bound for the model without rotation, for the case that the width is bounded below by ϵ or/and bounded above by $1-\epsilon$. This implies that there is no

competitive algorithm for the unbounded width case. Our lower bound holds for deterministic and randomised algorithms.

- An $O(\log \frac{1}{\epsilon})$ algorithm for the model without rotation for the case when widths are bounded below by ϵ or/and bounded above by $1-\epsilon$.
- A 4 competitive algorithm for squares in both models. This immediately follows from the first result and the fact that squares do not change their form when rotated.

It is interesting to note that although the two main algorithms that we present (one without rotation, and one with rotation) are quite similar, there is a substantial difference in their analysis and performance.

We note that, Karloff [17], independently, provided a lower bound of $\Omega(\sqrt{\log \frac{1}{\epsilon}})$ for deterministic algorithms for the case where only up and down movements are allowed. It does not seem obvious to extend his bound for randomised algorithms or for algorithms with arbitrary movements.

Other related work. A special case of the problem considered in the model without rotations is, when all widths are equal. In fact this is equivalent to the list scheduling problem, that was first studied by Graham [15] and later in [10, 2, 4, 14, 5, 18, 3].

In the case that all heights of the rectangles are equal it can be easily shown that the packing should consist of strips of that height. This problem is similar to the one-dimensional bin packing problem, since we can treat each strip as a bin. However the problems are not equivalent since full strips can block rectangles to reach other strips. Bin packing has been widely studied. Results on on-line bin packing appear in [16, 19, 22, 20, 21]

The problem in which the rectangles are packed directly into bounded bins has been also studied. Here the goal is to minimize the number of bins. For the on-line version constant competitive algorithms have been designed in [9]. Improvements and lower bounds on the constant appear in [9, 11, 7, 13, 21]. In both [13, 21] three dimensional packing of boxes into three dimensional bins is also considered.

Definitions and notations. We consider sequences σ of open rectangles r_i. The rectangles have to be packed into a bin of infinite height. Each rectangle, r_i has a width w_i (a real number between 0 and 1), and a height h_i (also real). The cost of the algorithm is measured by the maximum height, measured from the bottom of the bin, that is occupied by some rectangle of the packing. We denote the optimal off-line cost for the sequence by $C_{Opt}(\sigma)$, (or C_{Opt}, if the sequence is clear from the context). This is the smallest height that the sequence can occupy. The on-line algorithm's cost for σ is denoted by $C_{on}(\sigma)$, (or C_{on}). An on-line algorithm has competitive ratio cr if for all σ

$$C_{on} \leq cr \cdot C_{Opt} + c_1$$

for some constant c_1. We also use the notation of $A(\sigma)$, which is the total area of the rectangles in σ i.e.

$$A(\sigma) = \sum_i w_i h_i$$

2 The Model With Rotations

We first discuss the model in which rotations are allowed. Here we obviously assume that both the width and the height of each rectangle is bounded by 1. The most natural thing to do is to rotate the rectangles and assign them on their wide side. Unfortunately, it can be shown that this method does not work well. In fact, our algorithm assigns the rectangles on their narrow side. Our algorithm divides the bin into horizontal strips, one on top of the other. Each strip will be used for a certain range of heights of rectangles. Let W and α be real constants: $0 < \alpha < 1$ and $0 < W < \frac{1}{2}$.

The Algorithm A: When a rectangle arrives, it is first rotated on its narrow side and then is placed as follows:

1. If the rectangle is a buffer (i.e the width after rotation is at least W), a new strip of height of the buffer is open for it and the buffer is placed into the strip to the left.
2. If the rectangle is not a buffer, it is placed into a strip with the height equal to that of the rectangle rounded up to the closest power of α. The algorithm checks if there is such a strip that is reachable by the rectangle and the total width of all the rectangles in this strip (including the new rectangle) does not exceed $1 - W$. If there is such a strip, the rectangle is assigned there, otherwise, a new strip is opened on top of all previous strips, and the rectangle is placed there to the left.

If a piece can reach some open strip of the suitable height, and can be placed there, it is placed into the lowest strip of this type, next to the pieces that are already there to the left. We call rectangles of width between W and $1-W$ small buffers, and those of width between $1-W$ and 1, large buffers. Note that only large buffers might block some future rectangles from getting into a suitable place since we try to assign only non-buffers in previous strips.

Theorem 2.1 *If we choose $W = \frac{1}{4}$ and $\alpha = \frac{2}{3}$, then the algorithm A is 4-competitive.*

Proof: Since all heights are bounded (the width and the height of the rectangle are bounded by 1), we allow an additive constant. We are going to show that $C_{on} \leq 4 \cdot C_{Opt} + 3$. We define a strip as full, if the sum of the widths of the rectangles that were placed in it is at least $1 - 2W$, and non-full otherwise. We denote the subsequence of large buffers by $B_1, B_2, ..., B_k$. To prove the theorem it is enough to show that at least a fraction W of the area is occupied (except a height of $1/(1-\alpha)$) and thus

$$C_{Opt} \geq A(\sigma) \geq W \cdot (C_{on} - \frac{1}{1-\alpha}) .$$

We first consider strips in which small buffers were put. Since the width of buffers is at least W, those strips are at least W full. We ignore the first strip of each height that was ever opened, if it is non-full. The total height we ignore is bounded by $\sum_{i \geq 0} \alpha^i = 1/(1-\alpha)$. Consider the other non-full strips. We associate each non-full strip with a large buffer or full strip, depending on which of three ways a strip is opened:

- The first strip of a certain height (which we already considered).

- A large buffer blocks it from getting into a strip. We associate the strip with the highest buffer that blocked the rectangle, and caused the opening of the strip.
- A rectangle did not fit into the last strip of this height. The last strip must be full, otherwise any rectangle which is not a buffer would fit there. We associate the new strip with the last strip.

Note that for each buffer, there can be only one strip of each height associated with it, and for each full strip, only one non-full strip associated with it. Consider the full strips with no non-full strips associated with them. Those strips are at least α full in the height, and at least $1-2W$ full in the width. The occupied area is at least $\alpha(1-2W)$. We now compute the occupied area for the full strips with the non-full strips. If we put all the rectangles in one strip, the width would be at least $1-W$, since the rectangles did not fit into one strip together. The strip would be α full in the height, and the two strips together would be $\alpha(1-W)/2$ full, because of the second strip of the same height. As for the large buffers, each buffer was associated with a few strips, at most one of each height. Assume that the total height of those strips for a certain buffer is h, $0 \leq h \leq \frac{1}{1-\alpha}$ the height of the buffer is $h(B_i)$ and the width is $w(B_i)$. Since this is a large buffer, $1-W \leq w(B_i) \leq h(B_i) \leq 1$. The occupied height is $h+h(B_i)$ and the occupied area is at least $h(B_i)w(B_i)+\alpha \cdot h \cdot (1-w(B_i))$, since the strips are at least α full in the height, and $1-w(B_i)$ in the width (the rectangles in each one were blocked by the buffer). It is easy to check that for $W=\frac{1}{4}$ and $\alpha=\frac{2}{3}$ the occupied area is at least $\frac{1}{4}$ of the total area for each of the cases. This yields the competitive ratio of 4 which is actually the best ratio that can be achieved in this method. ■

3 The Lower Bound For The Model Without Rotations

In this section we provide a lower bound for the case of the problem in which the widths of the rectangles are bigger than ϵ, or all widths are less than $1-\epsilon$ (or both).

Note that we allow to place rectangles in any free space, and not only above other rectangles, so that there is no gravity. If we enforced gravity, no algorithm can achieve a good competitive ratio since we can prove the following claim:

Claim 3.1 Any algorithm for the model with gravity, no rotations, and when all rectangles are at least of width ϵ, is $\Omega(\frac{1}{\epsilon})$ competitive. This holds also if all widths are less than $1-\epsilon$.

Proof: The proof is in the appendix. ■

Thus if we enforced gravity, the trivial algorithm which places the rectangles one on top of the other, would be optimal, for the case with minimum width ϵ.

We show that there is no constant competitive algorithm for the case without rotations (and without gravity). More precisely, we prove the following theorem:

Theorem 3.1 Any on-line (deterministic or randomised) algorithm for this problem has a competitive ratio of at least $\Omega(\sqrt{\log \frac{1}{\epsilon}})$.

Proof: We assume that $\epsilon \leq \frac{1}{16}$, otherwise the lower bound is constant. We first prove the deterministic lower bound. Later we show how to modify it for the randomised case.

We use a sequence which consists of rectangles with width between ϵ and $1-\epsilon$ and height between δ and 1 where $\delta \leq 1/\log\frac{1}{\epsilon}$. We show that for that sequence the off-line cost is 1, and the on-line cost is $\Omega(\sqrt{\log\frac{1}{\epsilon}})$, we call it a base sequence. By multiplying the heights by a constant, we can get any off-line cost, and thus get the lower bound even if we allow an additive constant. Moreover, the bound is correct even if the heights of the rectangles are bounded by 1 (or any other constant). To build the sequence, we repeat the base sequence several iterations, separating the iterations by a rectangle with width $1-\frac{\epsilon}{2}$ and height δ. (Here we prove lower bound for $\frac{\epsilon}{2}$ instead of ϵ). Thus any rectangle that arrives after this rectangle can't be placed under this one, since the minimum width of rectangle is ϵ.

We can now introduce the base sequence. We define n such that $\epsilon = \frac{1}{2^n}$, which implies that $\delta \leq \frac{1}{n}$. To prove the lower bound consider the sequence of rectangles that consists of two types:

- Tall rectangles (with height $\frac{1}{4\sqrt{n}}$ and widths $2^{i-1}\epsilon$ for $i = 1$ to $\frac{n}{2}$)
- Buffers, (rectangles with height δ and widths $1 - 2^i\epsilon + \frac{\epsilon}{2}$)

The sequence consists of two types of phases. The $i^{\underline{th}}$ phase, consists of $\frac{\sqrt{n}}{2}$ tall pieces of width $2^{i-1}\epsilon$, and one buffer of width either $1-2^i\epsilon+\frac{\epsilon}{2}$ or $1-2^{i-1}\epsilon+\frac{\epsilon}{2}$, which corresponds to type 1 or type 2 phase. We run the sequence until the completion of $\frac{n}{2}$ phases, or $\sqrt{n}$ phases of type 2, whatever happens first. We show that the off-line packing uses a maximum height of 1, and that the height required for any on-line algorithms is at least $\sqrt{\log\frac{1}{\epsilon}}/8$, the available space of the off-line algorithm is called a box. First, the off-line algorithm uses an 1×1 (height 1 and width 1) box. All the rectangles and buffers must be packed in this box. After each phase the on-line algorithm will have a smaller box available, (see figure 4).

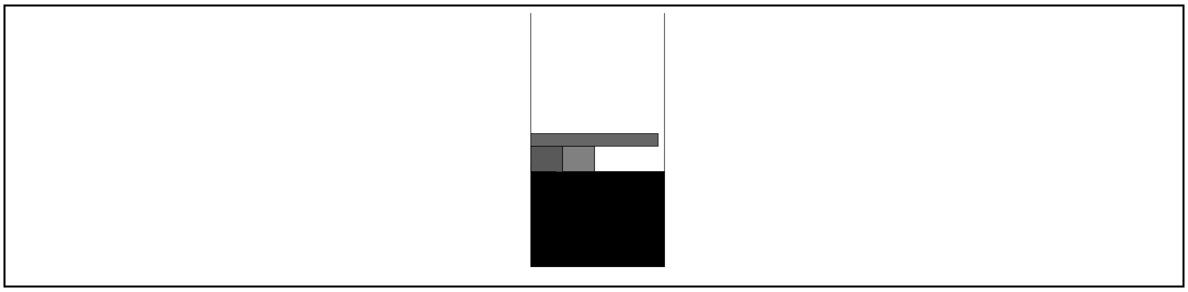

Figure 3: The on-line packing at state 1

The on-line algorithm is forced to put the buffer in such a way that no rectangle that arrives after this buffer can fit under the buffer hence for the on-line algorithm all the rectangles of one phase are above all the rectangles of the previous phase. Therefore the on-line algorithm may use at each phase width 1 to pack all rectangles of that phase. We call two rectangles overlapping, if it is possible to draw a horizontal line that intersects them both.

We build the sequence inductively. Assume we constructed the first i phases of the sequence, (i.e. chose the type of each phase). At phase i the $\frac{\sqrt{n}}{2}$ tall pieces arrive, and the type is chosen according to the way the on-line algorithm has placed them. There are two possible states.

1. If at least two current phase pieces overlap, (figure 3), then the phase becomes type 1 and a buffer of width of $1-2^i\epsilon+\frac{\epsilon}{2}$ arrives. The off-line places all the pieces one on

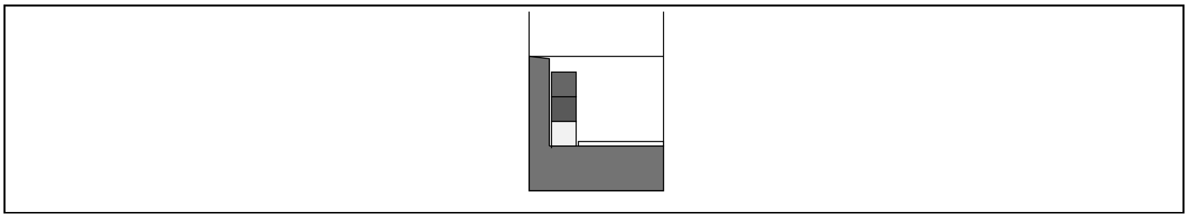

Figure 4: The off-line packing at state 1

   top of the other, and the buffer near them, as low as possible (see figure 4). Packing the rectangles like this, the off-line wastes height δ and width $2^{i-1}\epsilon$.

2. Otherwise, if no two current phase pieces overlap, then the phase becomes type 2 and a buffer of width $1-2^{i-1}\epsilon+\frac{\epsilon}{2}$ arrives. The off-line places all the pieces in one row on the bottom of the available rectangle. The buffer is placed on top of them, to the left. In this off-line packing the height wasted is $\frac{1}{4\sqrt{n}}+\delta$ and there is no wasted width.

Claim 3.2 Assume that up to (including) phase i, there are j phases of type 2. Then the on-line has wasted height of at least

$$(i-j)/(4\sqrt{n})+j/8 \tag{1}$$

and the off-line has a free box of height

$$1-\delta i-j/(4\sqrt{n}) \tag{2}$$

and width

$$1-(2^i-1)\epsilon \tag{3}$$

Proof: The proof is in the Appendix. ■

Recall that we run the sequence until the completion of $\frac{n}{2}$ phases or $\sqrt{n}$ type 2 phases, whatever happens first. Using Claim 3.2, it is easy to show that in both cases the on-line must have height of at least $\frac{\sqrt{n}}{8}$ and mxthe off-line algorithm can pack all the pieces in the box which completes the deterministic lower bound.

It is possible to modify the proof of the lower bound also for randomised algorithms. We omit the details. ■

4 Algorithms For The Model Without Rotations

As we proved in the last section, no constant competitive algorithm exists for the general problem, however we construct algorithms with competitive ratio which depends on the minimum width rectangle. In this section we present the algorithm for the case in which rotations are not allowed. The algorithm achieves competitive ratio of $O(\log \frac{1}{\epsilon})$ where ϵ is the minimum width of each rectangle. We also show how to change it for the case that the width of rectangles is bounded above by $1-\epsilon$ and not bounded below. Note that the heights of the rectangles are arbitrary and may be larger than 1. The algorithm uses horizontal strips, that are used for certain types of rectangles. Each strip is used for a certain range of heights, and a certain range of widths. An (i,j) strip is a strip that is used for rectangles of height h: $2^{j-1} < h \leq 2^j$ and width w: $2^{-i-1} < w \leq 2^{-i}$. (j is any integer, and i is a positive integer). The algorithm uses other separate strips for the rectangles of width at least $\frac{1}{4}$, those rectangles will be called buffers. Those strips are used only for buffers, one for each, and have the exact height of the buffer. We define a strip as available for a certain rectangle, if the rectangle can reach the strip (i.e. no buffer blocks the strip from the rectangle) and if after the rectangle is placed into the strip, the sum of all widths of the rectangles in the strip will not exceed $\frac{3}{4}$.

The Algorithm B: When a rectangle arrives it is classified and assigned as follows:

- If it is a buffer, a new strip of height of the buffer is opened for it, above all the previous strips, and the buffer is placed into this strip to the left.
- If it is a non-buffer, it is classified as an (i,j) rectangle for some i and j. If there exists an available (i,j) strip then we place it there to the left. Otherwise, we open a new strip of height 2^j, just above all previous strips, to be an (i,j) strip, and place the rectangle there, to the left.

Theorem 4.1 The algorithm B is $O(\log \frac{1}{\epsilon})$ competitive, where ϵ is the minimum width of any rectangle.

Corollary 4.1 For the integer case, the algorithm is $O(\log n)$ competitive, where n is the width of the bin.

Proof: Substitute $\epsilon = \frac{1}{n}$. ∎

Proof: (of theorem). We assume without loss of generality that $\epsilon \leq \frac{1}{16}$, (otherwise, the upper bound is constant) We define a strip as full, if the sum of widths of the rectangles placed there is at least $\frac{1}{2}$. We also define all buffer strips as full. We now can reduce the sequence σ to σ', so that there are no full strips for the on-line algorithm. We remove all rectangles that were placed into the full rows, but are not buffers. We also reduce the height of the buffers to zero, but we do not remove them. Those buffers remain in the sequence as one-dimensional buffers: they do not have height, but block the same rectangles as before the reduction from getting to strips.

Lemma 4.1 If the competitive ratio of the reduced sequence σ' is $O(\log \frac{1}{\epsilon})$, so is the competitive ratio of the original sequence.

Proof: The proof is in the Appendix. ■

Note that since our algorithm keeps the width of all non-buffer strips below $\frac{3}{4}$, rectangles of different ranges of width are packed independently. We reduce the sequence further. For each i, we define σ_i to be the subsequence of all the buffers, and rectangles of width w: $2^{-i-1} < w \leq 2^{-i}$. Note that the same strips are opened for the rectangles in σ_i as in σ'.

Lemma 4.2 If the algorithm is constant competitive on each such subsequence σ_i, then it is $O(\log \frac{1}{\epsilon})$ competitive on σ'.

Proof: The proof is in the Appendix. ■

Note that by the last reduction, the on-line algorithm has no full strips also for each σ_i.

Lemma 4.3 For a subsequence σ_i of rectangles of widths w: $2^{-i-1} < w \leq 2^{-i}$, and one-dimensional buffers, the algorithm is 10 competitive.

Proof: For this subsequence we can also omit all buffers of width $1 - 2^{-i}$ and less, since they cannot block any non-buffer rectangle from σ_i. We can also assume that there are no buffers wider than $1 - 2^{-i-1}$. If there are, we can treat all the rectangles between two wide buffers as the subsequence we are considering. Since those wide buffers block all subsequent rectangles, and force the off-line and the on-line begin a new packing, as if a new sequence has begun.

Consider the off-line packing for σ_i, and consider all remaining buffers: $Buf_1, ..., Buf_m$ (from the bottom of the packing to the top). (See figure 5). Recall that the buffers are one-dimensional and thus are assigned at some height in the bin. We denote by h_k the height that Buf_k was assigned by the off-line algorithm. Denote $h_0 = 0$, $h_{k+1} = C_{Opt}(\sigma_i)$, the bottom and the top are treated as buffers. Recall that two rectangles overlap, if it is possible to draw a horizontal line that intersects them both.

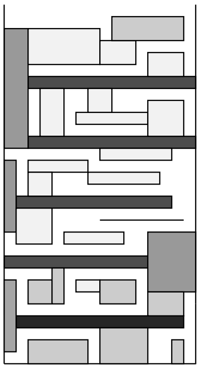

Figure 5: The off-line packing for σ_i

Consider all non-buffer rectangles, there are 3 types of possible packing for them at the off-line assignment:

1. A rectangle is considered "pioneer", if there was no *previous* rectangle that overlapped it.

2. The rectangle is not a pioneer, and was placed between two buffers, i.e. there exists k such that the top of the rectangle is not higher than h_{k+1}, and the bottom of the rectangle is not lower than h_k.

3. If the rectangle is not pioneer, and wasn't placed between buffers, then it must have been placed near some buffer.

We consider all strips that were opened for each type of rectangles by the on-line algorithm.

Claim 4.1 The strips used for type 1 rectangles, occupy at most height of $2C_{Opt}(\sigma_i)$.

Proof: The proof is in the Appendix. ∎

Claim 4.2 The strips opened for type 2 rectangles, occupy at most height of $4C_{Opt}(\sigma_i)$.

Proof: Denote by S_k the type 2 rectangles that were placed between Buf_k and Buf_{k+1} by the off-line. We first prove the following lemma.

Lemma 4.4 The on-line algorithm opened at most one strip of each height in order to place the rectangles in S_k.

Proof: By contradiction. Assume that two different strips of the same height were opened for the rectangles in S_k. Denote by r_1 and r_2 the two rectangles, that the two above strips were opened for. Since there are no full strips in the on-line packing, the second strip was opened since a buffer blocked r_2 from being placed at the same strip as r_1. This buffer arrived between r_1 and r_2 since all the buffers that arrived before r_1 were placed by the on-line algorithm below r_1.

We consider the place of this buffer in the off-line packing. Since r_1 is between two buffers, the buffer can not overlap r_1 in the packing and thus there are two possibilities for the off-line location of the buffer.

- The buffer is under r_1. The buffer should have passed near r_1 to reach its location. However, r_1 is not a pioneer and therefore any rectangle that passes near it must have width less than $1 - 2^{-i}$. The buffer has width of at least $1 - 2^{-i}$ which is a contradiction.

- The buffer is above r_1, The buffer is also above r_2, since both rectangles are between the same two buffers. r_2 arrived after the buffer and therefore its width is smaller than 1 minus the width of the buffer. Since the on-line algorithm assigns the buffers in a separate strip, it could not block r_2 which is a contradiction.

∎

We continue the proof of Claim 4.2. All the rectangles in S_k have height of at most $h_{k+1} - h_k$. Thus the highest strip that may be opened by the on-line algorithm for those rectangles is at most $2(h_{k+1} - h_k)$. There is at most one strip for each height, so the total height of strips opened for the rectangles in S_k is $4(h_{k+1} - h_k)$. Summing for all k we get that the total height is $4C_{Opt}$, since

$$4\sum_{k=0}^{m}(h_{k+1}-h_k) \leq 4C_{Opt} \ .$$

■

Claim 4.3 The strips used for type 3 rectangles, occupy at most height of $4C_{Opt}(\sigma_i)$.

Proof: First consider the off-line packing. Note that there is at most one rectangle near each buffer, since the width of the buffer is at least $1-2^{-i}$, and the width of any rectangle is larger than 2^{-i-1}. Moreover, consider the rectangles that were placed to the left of the buffers. Clearly, there are no rectangles to their left. All those rectangles can be moved so that they are just to the right of the left border of the bin. There is at most one rectangle at each height, and hence their total height is at most $C_{Opt}(\sigma_i)$. Since in the on-line packing the height of each strip is at most twice the height of the rectangle, the total height they might occupy is at most $2C_{Opt}(\sigma_i)$. The proof for the rectangles at the right of buffers is similar. ■

Now we conclude the proof of Lemma 4.3. Note that there might be strips that were used for more than one type of rectangles. Nevertheless, by the three Claims, the total height occupied by the σ_i rectangles is at most $10C_{Opt}(\sigma_i)$, and thus the algorithm is 10 competitive. ■

The proof of Theorem 4.1 follows immediately from Lemmas 4.1, 4.2 and 4.3. ■

Next, we claim that our analysis it tight.

Claim 4.4 The algorithm B is $\Omega(\log \frac{1}{\epsilon})$ competitive.

Proof: The proof is in the Appendix. ■

We can modify algorithm B to somewhat more natural algorithm B' as follows. The strips of a certain height are used for all widths of rectangles (instead of a separate strip for each width), in a way similar to the algorithm in section 2.

Theorem 4.2 The algorithm B' is $\Theta(\log \frac{1}{\epsilon})$ competitive.

Proof: The proof is in the Appendix. ■

Now we change the algorithm B to an algorithm C for the case that the width is not bounded below, but bounded above by $1-\epsilon$. When a rectangle of width less than ϵ arrives, we define how to place it, otherwise we place it in the same way as in algorithm B or B'. We open special strips for those rectangles. When a narrow rectangle arrives, its height is rounded up to a negative power of 2. If there is a non-full strip of narrow rectangles, for this height, the rectangle is placed there. Otherwise, a new strip of this height is opened, and the rectangle is placed there.

Theorem 4.3 The algorithm C is $O(\log \frac{1}{\epsilon})$ competitive, where $1-\epsilon$ is the maximum width of any rectangle.

Proof: The proof is in the Appendix. ■

Acknowledgment: We would like to thank Howard Karloff for his helpful discussion.

References

[1] B.S. Baker, E.G. Coffman,Jr , and R.L. Rivest. Orthogonal packings in two dimensions. In *The SIAM Journal of Computing*, pages 846–855, 1980.

[2] Y. Bartal, A. Fiat, H. Karloff, and R. Vohra. New algorithms for an ancient scheduling problem. In *Proc. 24$\underline{th}$ ACM Symp. on Theory of Computing*, pages 51-58, 1992.

[3] Y. Bartal, H. Karloff, Y. Rabani. A better lower bound for on-line scheduling. In *Information Processing Letters 50* pages 113-116, 1994.

[4] B. Chen, and A. van Vliet. On the on-line scheduling algorithm RLS Report 9325/A, Econometric Institue, Erasmus University, Rotterdam. 1993.

[5] B. Chen, A. van Vliet, and G.J. Woeginger. New lower and upper bounds for on-line scheduling. In *Operations Reserach Letters*, pages 222-230, 1994.

[6] F.R.K. Chung, M.R. Garey, and D.S. Johnson. On packing two-dimensional bins. *SIAM J. alg. Disc. Meth. 3(1)*, pages 66-76, 1982.

[7] J. Csirik, J.B.G. Frenk, and M. Labbe. Two-dimensional rectangle packing: on-line methods and results. In *Discrete applied Mathematics 45*, pages 197-204, 1993.

[8] E.G. Coffman,Jr , M.R. Garey, D.S. Johnson, and R.E. Tarjan. Performance bounds for level oriented two-dimensional packing Algorithms. In *The SIAM Journal of Computing*, pages 808-826, 1980.

[9] D. Coppersmith, and P. Raghavan. Multidimensional on-line bin packing: algorithms and worst-case analysis. In *Operation Research Letters*, pages 17-20, 1989.

[10] U. Faigle, W. Kern, and G. Turán. On the performance of on-line algorithms for partition problems. In *Acta Cybernetica 9*, pages 107-119, 1989.

[11] G. Galambos. A 1.6 lower bound for the two-dimensional on-line rectangle bin packing In *Acta Cybernetica*, pages 21-24, 1991.

[12] G. Galambos, A. van Vliet. An improved lower bound for On-Line Algorithms for the 2-Dimensional rectangle bin packing problem. In Report 9323/A, Econometric Institue, Erasmus University, Rotterdam. 1993.

[13] G. Galambos, A. van Vliet. Lower bounds for 1-,2- and 3-dimensional on-line bin packing algorithms. In *Computing* , pages 281-297, 1994.

[14] G. Galambos, and G.J. Woeginger. An on-line scheduling heuristic with better worst case than Graham's list scheduling. In *SIAM J. Comput. 22*, pages 349-355, 1993.

[15] R.L. Graham. Bounds for certain multiprocessing anomalies. In *Bell Sysem Tech J. 45*, pages 1563-1581, 1966.

[16] D.S. Johnson, A. Demers, J.D. Ullman, M.R. Garey, and R.L. Graham. Worst-case performance bounds for simple one-dimensional packing algorithms. In *The SIAM Journal of Computing*, pages 299-325, 1974.

[17] H. Karloff. Personal communication.

[18] D.R. Karger, S.J. Phillips, and E. Torng. A better algorithm for an ancient scheduling problem. In *Proc. 5 Ann. ACM-SIAM symposium on Discrete Algorithms*, pages 132-140, 1994.

[19] C.C. Lee, and D.T. Lee. A simple on-line bin-packing algorithm. In *Journal of the ACM*, pages 562-572, 1985.

[20] F. M. Liang. A lower bound for on-line bin packing. In *Information Processing letters*, pages 76-79, 1980.

[21] A. van Vliet. Lower and upper bounds for on-line bin packing and scheduling heuristics. Ph.D. thesis. Thesis publishers, Amsterdam, 1995.

[22] A. C. Yao. New algorithms for bin packing. In *Journal of the ACM*, pages 207-227, 1980.

5 Appendix

5.1 Proof of claim 3.1

To prove the claim we introduce the following sequence.

Fisrt at most $\frac{1}{\epsilon}$ pieces of height 1 and width 2ϵ arrive (we assume $\epsilon < \frac{1}{4}$). The pieces arrive until the on-line places two of the pieces in an overlapping way.

If the on-line did not place two pieces in an overlapping way, then $C_{on} = \frac{1}{\epsilon}$, the off-line places them in two rows, and thus $C_{Opt} = 2$.

Otherwise, if k pieces arrived, the height of the on-line is $k-1$, and the off-line will place them one on top of the other, yeilding the height of k.

Now for $i = 1$ to $\frac{1}{4\epsilon}$ we repeat the following sequence of two pieces:

- A buffer of small height δ, ($\delta < \epsilon$), and width $1 - 2\epsilon + \delta i$.
- One piece of height $k - 1 + \delta i$, and width $2\epsilon - \delta i$.

The off-line places all pieces under all buffers, since the total width of the rectangles is less than $2\epsilon + \frac{1}{4\epsilon} \cdot 2\epsilon < 1$. This is possible since the height of any rectangle is less than k, and each rectangle can be moved near any buffer that arrived before it. The total off line cost is $k + O(\frac{\delta}{\epsilon}) = O(k)$ for small δ.

When a new non-buffer rectangle arrives, the on-line must either create a new "shelf" for it, or to place it in a previous "shelf". The shelfs are formed exactly in the height of the rectangle in the phase, since the next buffer is too wide and must be placed above the rectangle, which closes the shelf from the top. There can be at most two rectangles in each shelf: the one that opened it, and another one may be placed near the buffer that closes this shelf. Thus the number of shelves, is at least half of the number of phases.

The cost of the on-line is at least $k - 1 + \frac{1}{8\epsilon}(k-1) = O(\frac{k-1}{\epsilon}) = O(\frac{k}{\epsilon})$. And thus $C_{on} \geq O(\frac{1}{\epsilon})C_{Opt}$.

5.2 Proof of Claim 3.2

We prove the claim by induction on the number of phases i. Before we begin ($i = j = 0$) the on-line lost no height, and the off-line has an 1×1 free box, as claimed. Suppose the claim is correct for the i first phases, which includes j type 2 phases. There are two possibilities:

1. Type 1 phase. There are two pieces that overlap in the on-line packing, since the sum of widths of the buffer and the two pieces is
$$(1 - 2^{i+1}\epsilon + \frac{\epsilon}{2}) + 2 \cdot 2^i\epsilon = 1 - 2^{i+1}\epsilon + \frac{\epsilon}{2} + 2^{i+1}\epsilon = 1 + \frac{\epsilon}{2} > 1 \ .$$
Then the buffer must be put above the line where they overlap. This adds the height of $\frac{1}{4\sqrt{n}} + \delta$, and proves (1). The off-line packs all the tall pieces one on top of the

other to the left of the free box. This can be done since the height of all $\frac{\sqrt{n}}{2}$ pieces together is $\frac{\sqrt{n}}{2} \cdot \frac{1}{4\sqrt{n}} = \frac{1}{8}$ and since the available height of the available box is

$$1 - \delta i - \frac{j}{4\sqrt{n}} \geq 1 - \delta \frac{n}{2} - \frac{\sqrt{n}}{4\sqrt{n}} \geq \frac{1}{4}$$

as $\delta \leq \frac{1}{n}$.

Now the width left is

$$(1 - (2^i - 1)\epsilon) - 2^i \epsilon = 1 - (2 \cdot 2^i - 1)\epsilon = 1 - (2^{i+1} - 1)\epsilon$$

Moreover the width of the buffer is $1 - (2^{i+1} - 1)\epsilon > 1 - 2^{i+1}\epsilon + \frac{\epsilon}{2}$. We can put it in the bottom of the available box, wasting the height of δ. Thus the height of the box is

$$1 - \delta i - \frac{j}{4\sqrt{n}} - \delta = 1 - \delta(i+1) - \frac{j}{4\sqrt{n}}$$

and the available width remains $1 - (2^{i+1} - 1)\epsilon$ which proves (2) and (3) .

2. Type 2 phase. The on-line algorithm did not put two tall overlapping pieces, (it is impossible to draw a horizontal line through two of the pieces), thus pieces must be put one on top of the other. Their total height is $\frac{1}{8}$ since there are $\frac{\sqrt{n}}{2}$ pieces of height $\frac{1}{4\sqrt{n}}$. The only way to place the buffer is on top of all the tall pieces, since a buffer and a tall piece can not be placed one next to the other, as the sum of their widths is

$$(1 - 2^i \epsilon + \frac{\epsilon}{2}) + (2^i \epsilon) = 1 + \frac{\epsilon}{2} > 1 \ .$$

The algorithm could not place any pieces under the buffers from previous phases, since the widths of buffers were at least $1 - 2^i \epsilon + \frac{\epsilon}{2}$. The height added is at least the height of big pieces, which is $\frac{1}{8}$, and this proves (1).

The off-line puts the big pieces in one row. This can be done since on one hand the available width is

$$1 - (2^i - 1)\epsilon \geq 1 - (2^{\frac{n}{2}-1})\epsilon = 1 - (2^{\frac{n}{2}-1})\frac{1}{2^n} = 1 - (2^{-\frac{n}{2}-1}) \geq 1 - 2^{-1} = \frac{1}{2} \ .$$

On the other hand there are $\frac{\sqrt{n}}{2}$ pieces of width $2^i \epsilon$ and the total width is

$$2^i \epsilon \cdot \frac{\sqrt{n}}{2} \leq 2^{\frac{n}{2}-1} \epsilon \cdot \frac{\sqrt{n}}{2} < 2^{\frac{n}{2}} \epsilon \cdot \frac{\sqrt{n}}{2} = \frac{1}{\sqrt{\epsilon}} \cdot \epsilon \cdot \frac{\sqrt{n}}{2} = \frac{\sqrt{\epsilon}\sqrt{n}}{2} = \frac{\sqrt{\frac{n}{2^n}}}{2} \leq \frac{1}{2}$$

since $\epsilon = \frac{1}{2^n}$ and $\frac{n}{2^n} \leq 1, \forall n \in N$. The buffer has width $1 - 2^i \epsilon + \frac{\epsilon}{2}$ and also fits into the width of $1 - (2^i - 1)\epsilon$. Thus the width of the available box does not change, but the height is reduced by $\frac{1}{4\sqrt{n}}$ by the big pieces and also δ by a buffer. By induction, the height of the available rectangle was $1 - \delta i - \frac{j}{4\sqrt{n}}$ at the beginning of the phase. Note that the number of type 2 phases has increased by 1 at the end of the phase. The height becomes

$$1 - \delta i - \frac{j}{4\sqrt{n}} - \delta - \frac{1}{4\sqrt{n}} = 1 - \delta(i+1) - \frac{j+1}{4\sqrt{n}}$$

and the available width is $1 - (2^i - 1)\epsilon \geq 1 - (2^{i+1} - 1)\epsilon$ as needed to prove.

5.3 Proof of Lemma 4.1

We denote the sum of the heights of full strips as H_{full} and the sum of heights of non-full strips as $H_{non-full}$. Clearly $C_{on} = H_{full} + H_{non-full}$. The full strips for non-buffers are at least $\frac{1}{2}$ full in the height, and at least $\frac{1}{2}$ full in the width. The buffer strips are full in the height, and at least $\frac{1}{4}$ full in the width. For the subsequent ϕ of rectangles in those strips: $C_{Opt}(\sigma) \geq A(\sigma) \geq A(\phi) \geq \frac{1}{4}H_{full}$, and thus $H_{full} \leq 4C_{Opt}(\sigma)$. The off-line cost of σ' may only be smaller than the off-line cost of σ. One can easily check that the behavior of the on-line algorithm on σ' is similar to its behavior on σ. More precisely, all buffers and rectangles that are not in a full strip are placed in the same way they were in σ, and the same strips are opened. (Since the buffers were left and they block the same rectangles they did before.) The new on-line cost is $C_{on}(\sigma') = H_{non-full}$, which satisfies:

$$H_{non-full} \leq O(\log\frac{1}{\epsilon})C_{Opt}(\sigma') \leq O(\log\frac{1}{\epsilon})C_{Opt}(\sigma) \ .$$

Thus

$$C_{on} \leq (O(\log\frac{1}{\epsilon}) + 4)C_{Opt} = O(\log\frac{1}{\epsilon})C_{Opt} \ .$$

5.4 Proof of Lemma 4.2

We denote the maximum competitive ratio over $2 \leq i \leq \lceil \log\frac{1}{\epsilon} \rceil$ as c. The on-line cost is actually the sum of costs for all the subsequences of various ranges of width. Since there are $O(\log\frac{1}{\epsilon})$ different ranges of width, and the off-line cost for each subsequence is bounded by the original off-line cost, we get:

$$C_{on}(\sigma') \leq c\sum_i C_{Opt}(\sigma_i)$$

$$\leq c \cdot O(\log\frac{1}{\epsilon})C_{Opt}(\sigma') = O(\log\frac{1}{\epsilon})C_{Opt}(\sigma') \ .$$

5.5 Proof of Claim 4.1

In the off-line packing, there is at most one pioneer rectangle at each height. (If there were two, one of them was packed earlier, and thus the second one is not pioneer). Thus the total height of pioneer rectangles is at most $C_{Opt}(\sigma_i)$. Since in the on-line packing, the height of each strip is at most twice the height of the rectangle in that strip. The total height those rectangles occupy is bounded by $2C_{Opt}(\sigma_i)$.

5.6 Proof of Claim 4.4

We introduce the following sequence: We define $n = \lceil \log\frac{1}{2\epsilon} \rceil$. For $i = 0$ to $n-1$ we repeat the following sequence of three pieces:

1. A rectangle of width $\epsilon 2^i$ and height 1.

2. Another rectangle of width $\epsilon 2^i$ and height 1.

3. A buffer of width $1 - \epsilon 2^{i+1} + \frac{\epsilon}{2}$, and height δ (small height).

The on-line algorithm places each pair in a strip, and the buffer above them. Each buffer blocks subsequent rectangles, as in the lower bound sequence. Thus the height of the on-line packing is $\Theta(n) = \Theta(\log \frac{1}{\epsilon})$. The off-line packs each pair one above the other, and the buffer near them as low as possible. The off-line cost is $1 + n \cdot \delta$. For a small δ, the cost is $O(1)$.

5.7 Proof of Theorem 4.2

We modify the proof of Algorithm B to prove this theorem. We eliminate full strips in the packing of B' and get a subsequence σ'. The full strips add a constant to the competitive ratio. We show that the height of B' on σ' is at most the height of B on σ'. We consider the set of first rectangles in each strip of B'. We claim that any two such rectangles are in different strips in B, which completes the proof. The claim is obvious If the two rectangles have different height ranges. If they have the same height range, there must be a buffer that arrived after the first rectangle, and before the second one, and blocks the second one to reach lower strips (since there are no full strips). Hence in B the same buffer blocks the second rectangle to be assigned in the same strip as the first rectangle.

5.8 Proof of Theorem 4.3

The narrow rectangles are not blocked by a buffer, since the width of buffers is bounded by $1 - \epsilon$. To prove the claim, we follow the proof of theorem 4.1. We need to bound the height wasted by the new strips, in the on-line packing. Since we reduce the sequence so that there are no full strips, we have at most one strip of each height. Consider the highest strip for narrow rectangles. Consider one of the rectangles inside the strip: r. The height of the strip is at most $2r$. The total height of the strips is at most $4r$. Since the rectangle is packed in the off-line packing too, $h(r) \leq C_{Opt}$, thus the wasted height is at most $4C_{Opt}$ and the algorithm is still $O(\log \frac{1}{\epsilon})$ competitive.