Theoretical Computer Science 440–441 (2012) 1–13

Contents lists available at SciVerse ScienceDirect

Theoretical Computer Science

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

New lower bounds for certain classes of bin packing algorithms☆

János Balogh, József Békési *, Gábor Galambos

Department of Applied Informatics, Gyula Juhász Faculty of Education, University of Szeged, H-6701 Szeged, Hungary

ARTICLE INFO

Article history:
Received 26 January 2011
Received in revised form 11 November 2011
Accepted 14 April 2012
Communicated by T. Erlebach

Keywords:
Bin packing
On-line algorithm
Worst-case behavior
Lower bound

ABSTRACT

On-line algorithms have been extensively studied for the one-dimensional bin packing problem. In this paper, we investigate two classes of one-dimensional bin packing algorithms, and we give better lower bounds for their asymptotic worst-case behavior. For on-line algorithms so far the best lower bound was given by van Vliet in (1992) [12]. He proved that there is no on-line bin packing algorithm with better asymptotic performance ratio than 1.54014 . . . . In this paper, we give an improvement on this bound to $\frac{248}{161} = 1.54037\ldots$ and we investigate the parametric case as well. For those lists where the elements are preprocessed according to their sizes in non-increasing order, Csirik et al. (1983) [1] proved that no on-line algorithm can have an asymptotic performance ratio smaller than $\frac{8}{7}$. We improve this result to $\frac{54}{47}$.

1. Introduction

The one-dimensional bin packing problem can be stated as follows. We are given a list L of n items – where the number of items is the length of the list – with sizes a_i, $i = 1, \ldots, n$, satisfying $0 < a_i \leq 1$. We need to pack these items into a minimal number of unit-capacity bins such that the total sum of the sizes in each bin is at most 1. The problem is known to be NP-hard [6]. So, substantial research has been focused on finding good approximation algorithms. One possibility to measure the performance of an algorithm A is to give its asymptotic performance ratio R_A. For a list L, let $\mathrm{OPT}(L)$ denote the number of bins in an optimal packing and let $A(L)$ denote the number of bins that algorithm A uses for packing L. If $R_A(l)$ denotes the supremum of the ratios $A(L)/\mathrm{OPT}(L)$ for all lists L with $\mathrm{OPT}(L) = l$, then the asymptotic performance ratio is defined as

$$R_A := \limsup_{l\to\infty} R_A(l).$$

If an algorithm belongs to the class of *on-line algorithms* then it packs items immediately when they appear without any knowledge of subsequent items of the list. After an item has been placed in a bin, it must not be moved again. This lack of knowledge is such a severe handicap that no on-line algorithm can have an asymptotic performance ratio close to 1. In the case of on-line algorithms, it is more fashionable to use the phrase asymptotic competitive ratio instead of asymptotic performance ratio. The best known on-line algorithm is due to Seiden [9] with asymptotic performance ratio at most 1.58889.., while van Vliet [12] gave a lower bound 1.54014 . . . for any on-line algorithm in 1992. He also investigated the parametric case, where for the sizes of the elements the inequality $0 < a_i \leq \frac{1}{r}$ is true for some $r > 1$ integer. To prove his result, van Vliet considered the solution of a special linear program. The proof is rather complicated and assumes a fair amount of knowledge about linear programming.

☆ This research was supported by HSC-DAAD Hungarian–German Research Exchange Program (project P-MÖB/837) and Gyula Juhász Faculty of Education, University of Szeged (project CS-001/2010).

* Corresponding author. Tel.: +36 62 546 057.
E-mail addresses: balogh@jgypk.u-szeged.hu (J. Balogh), bekesi@jgypk.u-szeged.hu (J. Békési), galambos@jgypk.u-szeged.hu (G. Galambos).

0304-3975/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2012.04.017

It was observed very early that the asymptotic performance ratio of on-line algorithms becomes significantly better if one can suppose that the elements arrive in decreasing order. For this case the best known on-line algorithm is *First Fit Decreasing* (FFD) given by Johnson [7] with $R_{\text{FFD}} = \frac{11}{9}$. For pre-ordered lists the best known lower bound is $\frac{8}{7}$. It was given by Csirik et al. [1]. So we have a very narrow gap [1.142857 . . . , 1.22 . . .] between the lower and upper bounds. In spite of great efforts, neither lower bound nor upper bound could be improved in the past 27 years.

This paper is organized as follows. In Section 2, we reformulate the packing pattern technique first introduced in [2]. In Section 3, we show that using this technique the 1.54014 . . . lower bound is also achievable with the right choice of the weights. Giving new sequences for the sizes of elements, in Section 4, we consider the parametric case and we slightly improve van Vliet's lower-bound to $\frac{248}{161} = 1.54037\ldots$. In Section 5, for pre-ordered lists we improve the $\frac{8}{7}$ lower bound to $\frac{54}{47} = 1.148936\ldots$. Some open problems conclude the paper.

2. Reformulated packing pattern technique

In this section, we reformulate the packing pattern technique which was first evaluated in [2]. Later the method was used by Galambos and Frenk in [3]. Both versions allowed only equal length lists in the construction of the proof. In his Ph.D. thesis, van Vliet [13] extended the technique for those constructions where one can use sublists with different sizes. Since we will use this basic theorem in our improvements we discuss the proof in detail. First, we need some preliminaries and we also introduce some notations.

For an arbitrary large integer n, we consider lists $L_1, L_2, \ldots, L_k$ of lengths $n_j = c_j \cdot n$ for certain integers c_j, $j = 1, 2, \ldots, k$. Sublist L_j contains equally sized elements. We assume that the size of an item does not depend on n. In the concatenated list $(L_1L_2 \ldots L_j)$ the elements of L_1 are followed by the elements of L_2 etc., and the list is terminated by the elements of L_j.

As a further notation, let $n \cdot U_j$ be an upper bound for the optimal packing of the concatenated list $(L_1L_2 \ldots L_j)$, i.e.,

$$U_j \geq \frac{\text{OPT}(L_1L_2 \ldots L_j)}{n}, \quad 1 \leq j \leq k.$$

Using the definition of the asymptotic performance ratio it is clear that for any on-line algorithm A

$$R_A \geq \max_{1\leq j\leq k} \limsup_{n\to\infty} \frac{A(L_1L_2 \ldots L_j)}{\text{OPT}(L_1L_2 \ldots L_j)} \geq \max_{1\leq j\leq k} \limsup_{n\to\infty} \frac{A(L_1L_2 \ldots L_j)}{n \cdot U_j}.$$

In order to establish the theorems, we introduce the definition of packing patterns (see [2]). Suppose that some algorithm A packs the elements of the concatenated list $L = (L_1L_2 \ldots L_k)$ into bins. A packing pattern $p = (p_1, p_2, \ldots, p_k)$ is a vector that denotes the number of elements from every list L_j, $j = 1, 2, \ldots, k$, while the algorithm places items into a bin according to that packing pattern. A packing pattern is feasible if $\sum_{i=1}^{k} a_ip_i \leq 1$, where a_i is the size of items in L_i. The set of all feasible packing patterns will be denoted by P. We define the subsets

$$P_i = \{p \in P \mid p_i > 0 \text{ and } p_j = 0, \text{ for } j < i\}, \quad i = 1, 2, \ldots k.$$

Clearly, $P_i \cap P_j = \emptyset$ if $i \neq j$, and $P = \cup_{i=1}^{k} P_i$.

While we pack the elements of the concatenated list $L = (L_1L_2 \ldots L_k)$, every bin must be filled according to one feasible packing pattern. For a given type $p = (p_1, p_2, \ldots, p_k)$ we denote the total number of bins which have been packed according to the packing pattern p by $n(p)$. The number of bins used by algorithm A while successively packing the lists is

$$A(L_1 \ldots L_j) = \sum_{i=1}^{j} \sum_{p\in P_i} n(p), \quad \text{for } j = 1, 2, \ldots, k, \tag{1}$$

and

$$n_j = \sum_{p\in P} p_jn(p), \quad \text{for } j = 1, 2, \ldots, k. \tag{2}$$

van Vliet stated the following theorem.

Theorem 2.1 ([13]). *Let w_j, $1 \leq j \leq k$, be some positive weights such that for every $p \in P_i$, $i = 1, 2, \ldots, k$*

$$\sum_{j=i}^{k} w_jp_j \leq k - i + 1 \tag{3}$$

holds. Then for every on-line algorithm A we have that

$$R_A \geq \frac{\sum_{j=1}^{k} w_jc_j}{\sum_{j=1}^{k} U_j}. \tag{4}$$

In this theorem, van Vliet considered k positive weights without any further condition, so if we apply this theorem for a special class of algorithms the weights can be arbitrary small. To avoid this inconvenience we can rescale the weights, and so we reformulate the above theorem as follows.

Theorem 2.2. *Let α_j and β_j be $2k$ positive integers such that for every $p \in P_i$, $i = 1, 2, \ldots, k$*

$$\sum_{j=i}^{k} \beta_j p_j \le \sum_{j=i}^{k} \alpha_j. \tag{5}$$

Then for every on-line algorithm A we have that

$$R_A \ge \max_{1\le j\le k} \limsup_{n\to\infty} \frac{A(L_1L_2\ldots L_j)}{OPT(L_1L_2\ldots L_j)} \ge \frac{\sum_{j=1}^{k}\beta_j c_j}{\sum_{j=1}^{k}\alpha_j U_j}. \tag{6}$$

Proof. If we multiply, for $j = 1, 2, \ldots, k$, Eqs. (1) and (2) by α_j and β_j, respectively, and sum all weighted equations, we get

$$\sum_{j=1}^{k} \alpha_j A(L_1\ldots L_j) = \sum_{j=1}^{k} \alpha_j \sum_{i=1}^{j} \sum_{p\in P_i} n(p) \tag{7}$$

and

$$\sum_{j=1}^{k} \beta_j n_j = \sum_{j=1}^{k} \beta_j \sum_{p\in P} p_j n(p). \tag{8}$$

Because of the property of the constants it follows that

$$\begin{aligned}
\sum_{j=1}^{k} \alpha_j \sum_{i=1}^{j} \sum_{p\in P_i} n(p) &= \sum_{p\in P_1} (\alpha_1+\alpha_2+\cdots+\alpha_k)\, n(p) \\
&\quad + \sum_{p\in P_2} (\alpha_2+\cdots+\alpha_k)\, n(p) \cdots + \sum_{p\in P_k} \alpha_k n(p) \\
&\ge \sum_{p\in P_1} (\beta_1 p_1+\beta_2 p_2+\cdots+\beta_k p_k)\, n(p) \\
&\quad + \sum_{p\in P_2} (\beta_2 p_2+\cdots+\beta_k p_k)\, n(p) + \cdots + \sum_{p\in P_k} \beta_k p_k n(p) \\
&= \sum_{j=1}^{k} \beta_j \sum_{p\in P} p_j n(p).
\end{aligned}$$

So – using (1) and (2) – we get that

$$\sum_{j=1}^{k} \alpha_j A(L_1\ldots L_j) \ge \sum_{j=1}^{k} \beta_j n_j. \tag{9}$$

Therefore

$$\begin{aligned}
R_A &\ge \max_{1\le j\le k} \limsup_{n\to\infty} \frac{\alpha_j A(L_1L_2\ldots L_j)}{\alpha_j \mathrm{OPT}(L_1L_2\ldots L_j)} \ge \limsup_{n\to\infty} \frac{\sum_{j=1}^{k}\alpha_j A(L_1\ldots L_j)}{\sum_{j=1}^{k}\alpha_j \mathrm{OPT}(L_1\ldots L_j)} \\
&\ge \limsup_{n\to\infty} \frac{\sum_{j=1}^{k}\beta_j n_j}{\sum_{j=1}^{k}\alpha_j \mathrm{OPT}(L_1\ldots L_j)} \ge \limsup_{n\to\infty} \frac{n\sum_{j=1}^{k}\beta_j c_j}{n\sum_{j=1}^{k}\alpha_j U_j} = \frac{\sum_{j=1}^{k}\beta_j c_j}{\sum_{j=1}^{k}\alpha_j U_j}. \quad \square
\end{aligned}$$

3. The right choice of the weights

In [3], Galambos and Frenk did not give an explicit discussion of the packing pattern technique, but – using the idea of the packing pattern – they were able to give a simpler proof for the 1.5363 . . . lower bound for on-line bin packing algorithms given by Liang [8]. They investigated the parametric case as well. In [13], van Vliet – using his generalization – improved the lower bound to 1.54014 . . . . Here, we will show that the right choice of the weights allows us to give the same lower bound using the packing pattern technique as van Vliet got with the help of the linear programming technique. During his proof he constructed a linear program, he solved it and defined two functions f_k and g_k, both of them depending on k. He received his result as a limit of a function in f_k and g_k for $k \to \infty$. Since van Vliet proved that with the help of the applied sequences there is no possibility to get a better lower bound, our procedure will also justify that our approach has the same power as the LP method has.

In all the papers mentioned above, a specific sequence – mostly called as *Salzer sequence* – was applied to construct lists with equal sizes of elements. This sequence was first introduced by Sylvester in 1880 [11], therefore, we refer to this sequence as *Sylvester sequence.* For integers $k > 1$ and $r \geq 1$, we define the Sylvester sequence $m_1, \ldots, m_k$ by setting

- $m_1 = r + 1$,
- $m_2 = r + 2$,
- $m_j = m_{j-1}(m_{j-1} - 1) + 1$, for $j = 3, \ldots, k$.

Now we define k lists as follows. Let $n = c(m_k - 1)$ for some positive integer c. Each list L_j, $j = 1, \ldots, k-1$, contains n elements, while L_k contains rn pieces of elements, i.e. $c_j = 1$, if $j = 1, 2, \ldots, k-1$, and $c_k = r$. The sizes of elements in L_j are $a_j = 1/m_{k-j+1} + \varepsilon$, where $0 < \varepsilon < 1/(r+k)(m_k(m_k - 1))$. The following Lemma was proved in [8].

Lemma 3.1. (i) *$OPT(L_1L_2 \ldots L_j) = \frac{n}{m_{k-j+1}-1}$, for all $j = 1, \ldots, k-1$.*
(ii) *$OPT(L_1L_2 \ldots L_k) = n$.*

So for a fixed k we set

$$U_j = \begin{cases} \dfrac{1}{m_{k-j+1} - 1}, & \text{if } 1 \leq j \leq k-1, \\ 1, & \text{if } j = k, \end{cases}$$

and we define the following constants.

$$\beta_j = \begin{cases} 1, & \text{if j=1} \\ (m_{k-j+1} - 1)\beta_{j-1}, & \text{if } 2 \leq j \leq k-1, \\ \beta_{k-1}, & \text{if } j = k. \end{cases}$$

$$\alpha_j = \begin{cases} \beta_{j+1}, & \text{if } 1 \leq j \leq k-1, \\ r\beta_k, & \text{if } j = k. \end{cases}$$

Comparing our weights to those given in [13] we can realize the difference between them. So, although the formula is almost the same, our result is better than that van Vliet has got with the help of the packing pattern technique. On the other side, it is also easy to check that our proof is much simpler than the LP technique.

Theorem 3.2 (*[12]*)**.** *Every one-dimensional on-line bin packing algorithm A has worst case ratio*

$$R_A \geq \lim_{k\to\infty} \frac{\sum_{j=1}^{k} c_j\beta_j}{\sum_{j=1}^{k} \alpha_j U_j}.$$

Proof. For the application of Theorem 2.2 we have to show that for every $i = 1, \ldots, k$,

$$\sum_{j=i}^{k} \beta_j p_j \leq \sum_{j=i}^{k} \alpha_j. \tag{10}$$

First, we investigate the left hand side of (10).

Lemma 3.3.

$$\sum_{j=i}^{k} \beta_j p_j \leq \beta_i(m_{k-i+1} - 1). \tag{11}$$

Proof. Let $p = (0, \ldots, 0, p_i, p_{i+1}, \ldots, p_k) \in P_i$ be a feasible packing-pattern. It has been proved several times (see e.g. [3], [5]) that if we replace each element of L_j by $m_{k-j+1} - 1$ elements of L_{j-1} for some $j = i+1, \ldots, k$ then the sum of the sizes of the elements in the bin does not increase, and so the new pattern – denoted by p' – remains feasible.

If we denote the left hand side of (11) by $W(p)$ and $W(p')$, respectively, then – using the definitions of the β-s – we get

$$W(p) = \sum_{l=i}^{k} \beta_l p_l = \sum_{l=i}^{j-1} \beta_l p_l + \underbrace{(m_{k-j+1} - 1)\beta_{j-1}}_{\beta_j} p_j + \sum_{l=j+1}^{k} \beta_l p_l = W(p'),$$

i.e. the left hand side of (11) does not change while doing this substitution. Repeating this replacement iteratively on p for $j = k, \ldots, i+1$ we will end up with a packing pattern containing only elements from L_i. Clearly for this pattern $p_i \le m_{k-i+1} - 1$ holds. From this we get that for every $p \in P_i$ pattern

$$\sum_{j=i}^{k} \beta_j p_j \le \beta_i (m_{k-i+1} - 1) \tag{12}$$

which completes the proof of the lemma. □

Now we concentrate on the right side of (10).

Lemma 3.4.

$$\beta_i (m_{k-i+1} - 1) = \sum_{j=i}^{k} \alpha_j. \tag{13}$$

Proof.

$$\begin{aligned}
\sum_{j=i}^{k} \alpha_j &= (\alpha_i + \cdots + \alpha_{k-1}) + \alpha_k = (\alpha_i + \cdots + \alpha_{k-1}) + r\beta_k \\
&= (\alpha_i + \cdots + \alpha_{k-1}) + (m_1 - 1)\beta_k \\
&= (\alpha_i + \cdots + \alpha_{k-2}) + \alpha_{k-1} + (m_1 - 1)\beta_{k-1} \\
&= (\alpha_i + \cdots + \alpha_{k-2}) + m_1 \beta_{k-1} \\
&= (\alpha_i + \cdots + \alpha_{k-3}) + \alpha_{k-2} + (m_2 - 1)\beta_{k-1} \\
&= (\alpha_i + \cdots + \alpha_{k-3}) + \beta_{k-1} + (m_2 - 1)\beta_{k-1} \\
&= (\alpha_i + \cdots + \alpha_{k-3}) + m_2 \beta_{k-1} = (\alpha_i + \cdots + \alpha_{k-3}) + m_2 (m_2 - 1)\beta_{k-2} \\
&= (\alpha_i + \cdots + \alpha_{k-3}) + (m_3 - 1)\beta_{k-2} = \cdots \\
&= \alpha_i + (m_{k-i} - 1)\beta_{i+1} = m_{k-i}\beta_{i+1} = m_{k-i}(m_{k-i} - 1)\beta_i \\
&= (m_{k-i+1} - 1)\beta_i
\end{aligned}$$

as it was stated in lemma. □

Combining the results of Lemmas 3.3 and 3.4 inequality (10) follows immediately. □

As an example we show the case $r = 1$, $k = 3$, where $m_1 = 2, m_2 = 3, m_3 = 7, \beta_1 = 1, \beta_2 = 2, \beta_3 = 2, \alpha_1 = 2, \alpha_2 = 2, \alpha_3 = 2, U_1 = \frac{1}{6}, U_2 = \frac{1}{2}, U_3 = 1$. So we get

$$R_A \ge \frac{\sum_{j=1}^{3} c_j \beta_j}{\sum_{j=1}^{3} \alpha_j U_j} = \frac{5}{\frac{1}{3} + 1 + 2} = \frac{3}{2}.$$

Table 2 displays van Vliet's lower bounds for the asymptotic performance ratio of on-line algorithms for some values of k and r, which where calculated by our formula.

4. The new parametric on-line lower bound

Proving his result van Vliet used the Sylvester sequence. This is a so-called double exponential sequence whose reciprocals tend very quickly to zero. That is the reason why constructing a lower bound for $k = 5$ the first five decimals have been reached by the appropriate choice of the sizes in the lists. During the past two decades a lot of efforts have been made to improve this result. It was already proved by van Vliet that his result was not improvable with the Sylvester sequence. Therefore we inquired to find other sequences which do not tend so quickly. Besides other approaches we attempted to give up the greedy choice of the next elements in the sequence. Among other – unsuccessful – shots we hit the following sequence. For any integer $r \ge 1$ let

- $b_{1,r} = r + 1$,
- $b_{2,r} = r + 2$,
- $b_{3,r} = b_{1,r} b_{2,r} + 1$,
- $b_{j,r} = b_{3,r}^{j-2}$, $4 \le j \le k - 1$,
- $b_{k,r} = b_{1,r} b_{2,r} b_{3,r}^{k-3} + 1$.

For the sake of simpler notation instead of $b_{i,r}$ we will use the notation b_i.

Table 1
The first few elements of the parametric Sylvester sequences if $k \geq 5$.

	$r = 1$	$r = 2$	$r = 3$	$r = 4$	$r = 5$
$m_1 = r + 1$	2	3	4	5	6
$m_2 = r + 2$	3	4	5	6	7
$m_3 = m_1 m_2 + 1$	7	13	21	31	43
$m_4 = m_3(m_3 - 1) + 1$	43	157	421	931	1 807
$m_5 = m_4(m_4 - 1) + 1$	1807	24 493	176 821	865 831	3 263 443

Table 2
van Vliet's lower bounds for on-line bin packing algorithms.

	$r = 1$	$r = 2$	$r = 3$	$r = 4$	$r = 5$
$k = 3$	1.5000000	1.3793103	1.2878787	1.2283464	1.1880733
$k = 4$	1.5390070	1.3895759	1.2914337	1.2298587	1.1888167
$k = 5$	1.5401467	1.3896489	1.2914427	1.2298604	1.1888172
$k = 6$	1.5401474	1.3896489	1.2914427	1.2298604	1.1888172
$k = 7$	1.5401474	1.3896489	1.2914427	1.2298604	1.1888172
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$k = \infty$	1.5401474	1.3896489	1.2914427	1.2298604	1.1888172

Table 3
The first few parametric values of the new sequence for $k = 6$.

	$r = 1$	$r = 2$	$r = 3$	$r = 4$	$r = 5$
$b_1 = r + 1$	2	3	4	5	6
$b_2 = r + 2$	3	4	5	6	7
$b_3 = b_1 b_2 + 1$	7	13	21	31	43
$b_4 = (b_1 b_2 + 1)^2$	49	169	441	961	1 849
$b_5 = (b_1 b_2 + 1)^3$	343	2 197	9 261	29 791	79 507
$b_6 = b_1 b_2 b_3^3 + 1$	2059	26 365	185 221	893 731	3 339 295

It is easy to prove that for any fixed integer $k < \infty$

$$r\frac{1}{b_1} + \sum_{i=2}^{k} \frac{1}{b_i} < 1.$$

If we compare the contents of Tables 1 and 3 it is conspicuous: we lose – in contrast to the greedy sequence – a bit at the fourth member, but – as we will see – our patience effects later improvement.

Using this new sequence we construct our lists as follows. Let A be an on-line algorithm. In the first step we consider a concatenated list with sublists $L_1, L_2, \ldots, L_k$ for $k \geq 4$ as follows.

(i) L_k contains nr elements of size $a_k = \frac{1}{b_1} + \varepsilon$,
(ii) L_{k-1} contains n elements of size $a_{k-1} = \frac{1}{b_2} + \varepsilon$,
(iii) L_j contains n elements of size $a_j = \frac{1}{b_{k-j+1}} + \varepsilon$, where $2 \leq j \leq k-2$
(iv) L_1 contains n elements of size $a_1 = \frac{1}{b_k} + \varepsilon$,

where $\varepsilon \leq \frac{1}{(k+r)b_k(b_k-1)}$, and $n = c(b_k - 1)$, for some integer $c \geq 1$. So, the constants that we apply while we use Theorem 2.2 are $c_j = 1$, if $j \leq k-1$, and $c_k = r$.

Note that for fixed $k \leq 4$ this definition gives the same lists, which are used in the proof of van Vliet's lower bound.

If one tries to prove that this sequence of the lists results in a better lower bound, of course the LP method established by van Vliet in [12] is adaptable. Indeed, we also constructed this LP. But – as we mentioned above – the proof of the cited paper seemed to be rather complicated, and so we tried to apply our packing pattern technique. To do that, the only question was whether we could find the correct values of α-s and β-s. (To find a good lower bound for the optimum was not difficult.) Before proving our main theorem we prove some lemmas.

Lemma 4.1. *For the optimum values of the concatenated lists the following relations hold*

(i) $OPT(L_1 \ldots L_j) \leq \frac{n}{b_1 b_2 b_3^{k-j-2}}$, *for* $1 \leq j \leq k-2$,
(ii) $OPT(L_1 \ldots L_{k-1}) \leq \frac{n}{b_1} = \frac{n}{r+1}$,
(iii) $OPT(L_1 \ldots L_k) \leq n$.

Proof. We will generate a feasible packing for each concatenated list.
Case (i): it is trivial that the items of L_1 can be packed into $\frac{n}{b_1b_2b_3^{k-3}}$ bins. Consider the list $(L_1L_2\dots L_j)$ for $2 \le j \le k-2$. Let $z = b_1b_2b_3^{k-3}$, then

$$\begin{aligned}
S_j &= \sum_{i=1}^{j} a_i \\
&= \frac{1}{b_1b_2b_3^{k-3}+1} + \frac{1}{b_3^{k-3}} + \dots + \frac{1}{b_3^{k-j-1}} + \frac{j}{(k+r)b_1b_2b_3^{k-3}(b_1b_2b_3^{k-3}+1)} \\
&< \frac{z + b_1b_2\cdot(z+1) + b_1b_2b_3(z+1) + \dots + b_1b_2b_3^{j-2}(z+1) + 1}{z(z+1)} \\
&= \frac{1 + b_1b_2(1 + b_3 + \dots + b_3^{j-2})}{z} = \frac{b_3^{j-1}}{b_1b_2b_3^{k-3}} = \frac{1}{b_1b_2b_3^{k-j-2}}.
\end{aligned}$$

This proves that the elements of the lists $(L_1\dots L_j)$ can be packed into $\frac{n}{b_1b_2b_3^{k-j-2}}$ bins if $1 \le j \le k-2$.
Case (ii):

$$\begin{aligned}
S_{k-1} &= \sum_{i=1}^{k-1} a_i \\
&= \frac{1}{b_1b_2b_3^{k-3}+1} + \frac{1}{b_3^{k-3}} + \dots + \frac{1}{b_3} + \frac{1}{b_2} + \frac{k-1}{(k+r)b_1b_2b_3^{k-3}(b_1b_2b_3^{k-3}+1)} \\
&< \frac{z + b_1b_2(z+1) + \dots + b_1b_2b_3^{k-4}(z+1) + b_1b_3^{k-3}(z+1) + 1}{z(z+1)} \\
&= \frac{b_1b_2(1 + b_3 + \dots + b_3^{k-4}) + b_1b_3k - 3}{z} = \frac{b_1b_2\frac{b_{k-3}-1}{b_3-1} + b_1b_3k - 3}{z} \\
&= \frac{1 + b_3^{k-3} - 1 + b_1b_3k - 3}{z} = \frac{b_3^{k-3}(b_1+1)}{b_1b_2b_3^{k-3}} = \frac{b_1+1}{b_1b_2} = \frac{1}{b_1}.
\end{aligned}$$

So, the elements of $(L_1\dots L_{k-1})$ can be packed into $\frac{n}{b_1}$ bins.
Case (iii):

$$\begin{aligned}
S_k &= \sum_{i=1}^{k-1} a_i + ra_k \\
&= \frac{1}{b_1b_2b_3^{k-3}+1} + \frac{1}{b_3^{k-3}} + \dots + \frac{1}{b_3} + \frac{1}{b_2} + \frac{r}{b_1} + \frac{k+r-1}{(k+r)b_1b_2b_3^{k-3}(b_1b_2b_3^{k-3}+1)} \\
&< \frac{(z+1)(1 + b_1b_2 + b_1b_2b_3 + \dots + b_1b_2b_3^{k-4} + b_1b_3^{k-3} + rb_2b_3^{k-3})}{z(z+1)} \\
&= \frac{1 + b_1b_2\left(1 + b_3 + \dots + b_3^{k-4}\right) + b_1b_3^{k-3} + rb_2b_3^{k-3}}{z} \\
&= \frac{1 + b_1b_2\frac{b_3^{k-3}-1}{b_3-1} + b_1b_3^{k-3} + rb_2b_3^{k-3}}{z} = \frac{1 + b_3^{k-3} - 1 + b_1b_3^{k-3} + rb_2b_3^{k-3}}{z} \\
&= \frac{b_3^{k-3}(1 + b_1 + rb_2)}{b_1b_2b_3^{k-3}} = \frac{b_2(1+r)}{b_1b_2} = 1.
\end{aligned}$$

Therefore, the elements of $(L_1\dots L_k)$ can be packed into n bins. □

Based on the above Lemma, we can choose the values of U_j^k as follows.

$$U_j^k = \begin{cases} \dfrac{1}{b_1b_2b_3^{k-j-2}}, & \text{if } j \le k-2, \\ \dfrac{1}{b_1} = \dfrac{1}{r+1}, & \text{if } j = k-1, \\ 1, & \text{if } j = k. \end{cases}$$

For a given $k \geq 4$ we define two k-dimensional vectors β^k and α^k, as follows.

$$\beta_j^k = \begin{cases} 1, & \text{if } j = 1, \\ b_1 b_2, & \text{if } j = 2, \\ b_3 \beta_{j-1}^k, & \text{if } 3 \leq j \leq k-2, \\ b_1 \beta_{k-2}^k, & \text{if } k-1 \leq j \leq k. \end{cases}$$

$$\alpha_j^k = \begin{cases} b_1 b_2, & \text{if } j = 1, \\ (b_1 b_2)^2, & \text{if } j = 2 \text{ and } k \geq 5, \\ b_3 \alpha_{j-1}^k, & \text{if } 3 \leq j \leq k-3, \\ \beta_{k-1}^k, & \text{if } k-2 \leq j \leq k-1, \\ r\beta_k^k, & \text{if } j = k. \end{cases}$$

Considering the above constants we need to prove for every feasible packing that inequality (5) holds. Let us suppose that the packing pattern $p = (0, \ldots, 0, p_i, \ldots, p_k)$ belongs to the subset P_i of the feasible packings. The packing pattern p is *dominant* in P_i if

$$a_t + \sum_{j=1}^{k} a_j p_j > 1,$$

for every $i \leq t \leq k$. We note that in our recent case $a_s < a_t$, if $s < t$, so during our proof we will use that p is *dominant* in P_i if

$$a_i + \sum_{j=1}^{k} a_j p_j > 1.$$

Let $D_i(p)$ be the set of those packing patterns for which p is dominant in P_i. So, it is enough to investigate the dominant patterns for each P_i. See for example [10].

Lemma 4.2. *Let $L = (L_1 \ldots L_k)$ be the above defined concatenated list for some $k \geq 4$. Then for every feasible dominant packing pattern $p \in P_i$*

$$\sum_{j=i}^{k} \beta_j^k p_j \leq \sum_{j=i}^{k} \alpha_j^k. \tag{14}$$

Proof. We prove by induction. Since the constants for the case $k = 4$ are the same as in the new proof of van Vliet's lower bound, for this case the statement of the Lemma holds. Suppose now that the statement holds for some $k \geq 4$.

We will distinguish two cases. First, we suppose that $p \in P_i$, $i \geq 3$. Let $p' \in P_j$ for some $i \leq j \leq k+1$. We say that the packing pattern p' is the *j-suffix* of p if

$$p'_l = \begin{cases} p_l, & \text{if } j \leq l \leq k+1, \\ 0, & \text{if } l < j. \end{cases}$$

Claim 4.3. *If $p \in P_i$, $i \geq 3$, and $p' \in P_j$ is its j-suffix, where $i \leq j$, then the packing pattern $p'' = (0, \ldots, 0, p_j, p_{j+1}, \ldots, p_{k+1})$ was already investigated during the case $k' = k - j + 2$ and it satisfies condition (14).*

Proof. Since $k' < k$, we can apply the induction hypothesis to the packing pattern p'', and the statement follows immediately from the definitions of the lists. □

Therefore, if $p \in P_i$, $i \geq 3$ then it satisfies condition (14).

So, we can suppose that $p \in P_i$, $i \leq 2$. Let us transform $p = (p_1, \ldots, p_{k+1})$ to a new packing pattern p^T as follows

$$p_j^T = \begin{cases} p_1 + b_1 b_2 p_2, & \text{if } j = 1, \\ 0, & \text{if } j = 2, \\ p_i, & \text{if } j > 2. \end{cases}$$

By the definition of β_j^{k+1} the equation

$$\sum_{j=1}^{k+1} \beta_j^{k+1} p_j^T = \sum_{j=1}^{k+1} \beta_j^{k+1} p_j$$

holds. Clearly, if p is dominant with respect to the set $D_i(p)$ then p^T is also feasible and dominant with respect to those packing patterns which we get with the same transformation from the elements of $D_i(p)$.

Claim 4.4. *For every dominant pattern p of the form $(p_1, 0, p_3, \ldots, p_{k+1})$, p_1 can be divided by b_3.*

Proof. Consider an arbitrary index j, $3 \le j \le k+1$. The packing pattern p contains exactly p_j items from L_j. By the definition of the items $q_j = \frac{b_{k+1}-1}{b_{k-j+2}}$ is an integer and can be divided by b_3. So we can substitute each element of L_j by q_j elements of L_1. We must prove that the pattern remains feasible after the substitutions. For this we show that $p_1 + \sum_{j=3}^{k+1} p_j q_j \le b_{k+1} - 1$.

The pattern is feasible before the substitutions, so

$$p_1 a_1 + \sum_{j=3}^{k+1} p_j a_j \le 1.$$

Since there is a positive p_j ($j = 1, 3, \dots, k+1$) and the previous sum contains at least one ε it follows that

$$p_1 \frac{1}{b_{k+1}} + \sum_{j=3}^{k+1} p_j \frac{1}{b_{k-j+2}} < 1,$$

so

$$\begin{aligned} 1 &> p_1 \frac{1}{b_{k+1}} + \sum_{j=3}^{k+1} p_j \frac{b_{k+1}-1}{(b_{k+1}-1)b_{k-j+2}} = p_1 \frac{1}{b_{k+1}} + \sum_{j=3}^{k+1} p_j q_j \frac{1}{b_{k+1}-1} \\ &\ge p_1 \frac{1}{b_{k+1}} + \sum_{j=3}^{k+1} p_j q_j \frac{1}{b_{k+1}}, \end{aligned}$$

i.e.

$$p_1 + \sum_{j=3}^{k+1} p_j q_j < b_{k+1},$$

and since p_1, p_j-s and q_j-s ($j = 3, \dots, k+1$) are integers,

$$p_1 + \sum_{j=3}^{k+1} p_j q_j \le b_{k+1} - 1. \tag{15}$$

Considering (15) and the definition of ε, we get that

$$\begin{aligned} \left(p_1 + \sum_{j=3}^{k+1} p_j q_j\right) a_1 &= \left(p_1 + \sum_{j=3}^{k+1} p_j q_j\right)\left(\frac{1}{b_{k+1}} + \varepsilon\right) \le (b_{k+1}-1)\left(\frac{1}{b_{k+1}} + \varepsilon\right) \\ &= \frac{b_{k+1}-1}{b_{k+1}} + (b_{k+1}-1)\varepsilon \le \frac{b_{k+1}-1}{b_{k+1}} + \frac{1}{b_{k+1}} = 1, \end{aligned}$$

which means that the pattern is feasible.

Having done this substitution for every j, we can calculate the maximal value of p_1 as

$$p_1 = b_1 b_2 b_3^{k-2} - \sum_{j=3}^{k+1} p_j q_j,$$

i.e. the difference between the maximal possible number of a_1 items in a bin, and the sum of the weighted q_j-s. Since each q_j can be divided by b_3, p_1 can also be divided by b_3. □

This fact proves that if p^T is a dominant pattern, then $p_1 + b_1 b_2 p_2$ can be divided by b_3 in p^T. Now we are ready to define a new packing pattern of $(L_1 L_2 \dots L_k)$. This will be

$$p^k = \left(\frac{p_1 + b_1 b_2 p_2}{b_3}, p_3, \dots, p_{k+1}\right).$$

Claim 4.5. *If $p = (p_1, p_2, \dots, p_{k+1})$ is a feasible dominant packing pattern for the list $L = (L_1 L_2 \dots L_{k+1})$ then $p^k = (\frac{p_1 + b_1 b_2 p_2}{b_3}, p_3, \dots, p_{k+1})$ is also feasible for $L' = (L_1 L_3 \dots L_{k+1})$.*

Proof. Let we remind the reader that the sizes of the elements are different if one considers L or L'. We need to prove that by substituting the p_1 and p_2 pieces of elements from L with $\frac{p_1 + b_1 b_2 p_2}{b_3}$ pieces of items from L' the occupied place will not increase.

$$\begin{aligned}\frac{p_1+b_1b_2p_2}{b_3}\left(\frac{1}{b_1b_2b_3^{k-3}+1}+\varepsilon\right) &< \frac{p_1}{b_1b_2b_3^{k-2}+b_3}+\frac{p_1}{b_3^{k-2}}\varepsilon+\frac{p_2}{b_3^{k-2}}+\frac{b_1b_2p_2}{b_3^{k-2}}\varepsilon \\ &< p_1\left(\frac{1}{b_1b_2b_3^{k-2}+1}+\frac{1}{b_3^{k-2}}\varepsilon\right)+p_2\left(\frac{1}{b_3^{k-2}}+\frac{b_1b_2b_3^{k-2}}{b_3^{k-2}}\varepsilon\right) \\ &< p_1\left(\frac{1}{b_1b_2b_3^{k-2}+1}+\varepsilon\right)+p_2\left(\frac{1}{b_3^{k-2}}+\varepsilon\right). \quad \square\end{aligned}$$

By the induction hypothesis for each feasible packing

$$\sum_{j=1}^{k}\beta_j^k p_j^k \le \sum_{j=1}^{k}\alpha_j^k. \tag{16}$$

Using the definition of α-s and β-s we can calculate both sides of (16) for a given packing pattern:

$$\begin{aligned}\sum_{j=1}^{k}\beta_j^k p_j^k &= \frac{p_1+b_1b_2p_2}{b_3}+\sum_{j=2}^{k}\beta_j^k p_{j+1} \\ &= \frac{p_1+b_1b_2p_2}{b_3}+\sum_{j=3}^{k+1}\frac{\beta_j^{k+1}}{b_3}p_j = \frac{1}{b_3}\sum_{j=1}^{k+1}\beta_j^{k+1}p_j,\end{aligned}$$

and

$$\sum_{j=1}^{k}\alpha_j^k = b_1b_2+\sum_{j=2}^{k}\alpha_j^k = b_1b_2+\sum_{j=3}^{k+1}\frac{\alpha_j^{k+1}}{b_3} = b_1b_2+\frac{1}{b_3}\sum_{j=3}^{k+1}\alpha_j^{k+1}.$$

So

$$\frac{1}{b_3}\sum_{j=1}^{k+1}\beta_j^{k+1}p_j \le b_1b_2+\frac{1}{b_3}\sum_{j=3}^{k+1}\alpha_j^{k+1},$$

therefore

$$\sum_{j=1}^{k+1}\beta_j^{k+1}p_j \le b_1b_2+(b_1b_2)^2+\sum_{j=3}^{k+1}\alpha_j^{k+1} = \sum_{j=1}^{k+1}\alpha_j^{k+1}.$$

So we completed the proof of Claim 4.5. $\square$

Now we are ready to prove the new lower bound.

Theorem 4.6. *Let r be a positive integer, and we consider the parametric bin packing problem, i.e. $a_i \le \frac{1}{r}$, if $a_i \in L$. Then there is no one-dimensional on-line bin packing algorithm A with an asymptotic performance ratio*

$$R_A < \frac{r^6+8r^5+29r^4+60r^3+75r^2+55r+20}{r^6+7r^5+22r^4+40r^3+45r^2+33r+13}.$$

Proof. Using Lemmas 4.1 and 4.2 and Theorem 2.2 the following inequality is true

$$R_A \ge \frac{\sum_{j=1}^{k} c_j\beta_j}{\sum_{j=1}^{k}\alpha_j U_j}.$$

Because

$$\begin{aligned}\sum_{j=1}^{k} c_j\beta_j &= 1+b_1b_2+(b_3-1)\sum_{j=3}^{k-2}b_3^{j-2}+b_3^{k-4}b_1^2b_2(1+r) \\ &= b_3+(b_3-1)b_3\frac{b_3^{k-4}-1}{b_3-1}+b_3^{k-4}b_1^3b_2 = b_3^{k-4}(b_3+b_1^3b_2)\end{aligned}$$

and

$$\begin{aligned}
\sum_{j=1}^{k}\alpha_j U_j &= \frac{1}{b_3^{k-3}} + \frac{(b_1b_2)^2}{b_1b_2b_3^{k-4}} + (b_1b_2)^2\sum_{j=3}^{k-3}\frac{b_3^{j-2}}{b_1b_2b_3^{k-j-2}} + \frac{1}{b_1b_2}b_1^2b_2b_3^{k-4} + \frac{1}{b_1}b_1^2b_2b_3^{k-4} + rb_1^2b_2b_3^{k-4}\\
&= \frac{1}{b_3^{k-3}} + \frac{b_1b_2}{b_3^{k-4}} + \frac{b_1b_2}{b_3^{k}}\sum_{j=3}^{k-3}b_3^{2j} + b_3^{k-4}b_1(1+b_2+rb_1b_2)\\
&= \frac{1}{b_3^{k-3}} + \frac{b_1b_2}{b_3^{k-4}} + \frac{b_1b_2b_3^6}{b_3^k}\frac{b_3^{2(k-5)}-1}{b_3^2-1} + b_3^{k-4}b_1(1+b_2+rb_1b_2)\\
&= \frac{1}{b_3^{k-3}} + \frac{b_1b_2}{b_3^{k-4}} + \frac{b_3^{k-4}}{b_3+1} - \frac{b_3^6}{b_3^k(b_3+1)} + b_3^{k-4}b_1(1+b_2+rb_1b_2)\\
&= \frac{\frac{b_3+1}{b_3} + b_1b_2(b_3+1) + b_3^{2(k-4)} - b_3^2}{b_3^{k-4}(b_3+1)} + \frac{b_3^{2(k-4)}b_1(1+b_2+rb_1b_2)(b_3+1)}{b_3^{k-4}(b_3+1)}\\
&= \frac{\frac{1}{b_3} + b_3^{2(k-4)}\big(1+b_1(1+b_2+rb_1b_2)(b_3+1)\big)}{b_3^{k-4}(b_3+1)},
\end{aligned}$$

we get that

$$\begin{aligned}
R_A &\geq \lim_{k\to\infty}\frac{b_3^{2(k-4)}(b_3+b_1^3b_2)(b_3+1)}{\frac{1}{b_3} + b_3^{2(k-4)}\big(1+b_1(1+b_2+rb_1b_2)(b_3+1)\big)}\\
&= \frac{(b_3+b_1^3b_2)(b_3+1)}{1+b_1(1+b_2+rb_1b_2)(b_3+1)}.
\end{aligned}$$

We know that $b_1 = r+1$, $b_2 = r+2$ and $b_3 = r^2+3r+3$, so we get

$$R_A \geq \frac{r^6+8r^5+29r^4+60r^3+75r^2+55r+20}{r^6+7r^5+22r^4+40r^3+45r^2+33r+13}. \quad \square$$

At the end of the section we give a table which displays the new lower bounds for the asymptotic competitive ratio of on-line algorithms for some values of r.

5. Improved lower bound for decreasing lists

For those lists where the elements are preprocessed according to their sizes in decreasing order, Csirik et al. [1] proved that there is no on-line algorithm with a better asymptotic performance ratio than $\frac{8}{7}$. Their construction is based on two lists which contain elements with sizes $\frac{1}{3}+\varepsilon$ and $\frac{1}{3}-2\varepsilon$. In the past three decades, attempts to obtain a better lower bound were not successful. The difficulty originates from the fact that the sizes of the elements of the last list in the concatenated list may not be too small, and hence they may fill up the opened bins, resulting in a better packing than in the earlier step. So, there is no point in investigating concatenated lists with k different sublists with $k\to\infty$, while the sizes of elements become progressively smaller and smaller. As a further application of Theorem 2.2, here we give a construction with three different lists. (Using 4 sublists, we were unsuccessful.) In our proof, we will use again the condition that the sublists must not have the same lengths.

Let A be an on-line algorithm. We consider a concatenated list with three sublists L_1, L_2 and L_3.

- L_1 contains n_1 elements of size $\frac{7}{24}-4\varepsilon$,
- L_2 contains n_2 elements of size $\frac{5}{24}+\varepsilon$,
- L_3 contains n_3 elements of size $\frac{4}{24}+\varepsilon$,

where $\varepsilon < \frac{1}{96}$, $n_1 = n_2 = 6n$ and $n_3 = 18n$. It means that $c_1 = c_2 = 6$ and $c_3 = 18$. It is easy to see that the following inequalities are true.

$$\mathrm{OPT}(L_1) \leq 2n, \qquad \mathrm{OPT}(L_1L_2) \leq 3n, \qquad \mathrm{OPT}(L_1L_2L_3) \leq 6n.$$

So, we can set the upper bounds to

$$U_1 = 2, \qquad U_2 = 3, \qquad U_3 = 6.$$

Table 4
The new lower bounds for on-line bin packing algorithms.

	$r=1$	$r=2$	$r=3$	$r=4$	$r=5$
$k=3$	1.5000000	1.3793103	1.2878787	1.2283464	1.1880733
$k=4$	1.5390070	1.3895759	1.2914337	1.2298587	1.1888167
$k=5$	1.5403448	1.3896631	1.2914442	1.2298607	1.1888172
$k=6$	1.5403721	1.3896636	1.2914443	1.2298607	1.1888172
$k=7$	1.5403726	1.3896636	1.2914443	1.2298607	1.1888172
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$k=\infty$	1.5403726	1.3896636	1.2914443	1.2298607	1.1888172
$k=\infty$	$\frac{248}{161}$	$\frac{1694}{1219}$	$\frac{7502}{5809}$	$\frac{24992}{20321}$	$\frac{68420}{57553}$

In fact, these upper bounds are sharp. Let us now consider the following constants.

$$\alpha_1 = 4, \qquad \alpha_2 = 3, \qquad \alpha_3 = 5$$

and

$$\beta_1 = 4, \qquad \beta_2 = 2, \qquad \beta_3 = 1.$$

Considering the above constants we need to prove that for every dominant packing pattern inequality (5) holds. Since the number of dominant patterns is small we can investigate all of them. Three cases have to be distinguished.

(i) For $i = 1$, we consider the dominant patterns in P_1. We need to prove that any feasible packing pattern $p \in P_1$ satisfies

$$12 \geq 4p_1 + 2p_2 + p_3.$$

The dominant patterns are (3, 0, 0), (2, 2, 0), (2, 1, 1), (2, 0, 2), (1, 3, 0), (1, 2, 1), (1, 1, 3) and (1, 0, 4). It is easy to check that the inequality holds for all of them.

(ii) For $i = 2$, the dominant patterns of bins in P_2 have to be considered. These are (0, 4, 0), (0, 3, 2), (0, 2, 3) and (0, 1, 4). All of them satisfy

$$8 \geq 2p_2 + p_3.$$

(iii) Finally, we have to address the packing patterns in P_3. The only dominant pattern is (0, 0, 5) and the inequality

$$5 \geq p_3$$

trivially holds.

So, the conditions of Theorem 2.2 hold and therefore

$$R_A \geq \frac{\sum_{j=1}^{3} c_j\beta_j}{\sum_{j=1}^{k} \alpha_j U_j} = \frac{24+12+18}{8+9+30} = \frac{54}{47}.$$

We can summarize our calculation in the following theorem.

Theorem 5.1. *No on-line algorithm for the one-dimensional bin packing problem which packs the elements in decreasing order and can have better asymptotic performance ratio than* $\frac{54}{47} = 1.1489361\ldots$.

6. Conclusions

In this paper, we improved two old lower-bound results for certain classes of one-dimensional bin packing algorithms. For on-line algorithms we considered the parametric case and the new lower bounds are summarized for some positive integers r in Table 4. For those semi-online bin packing algorithms, which allow pre-ordering, yielding the elements in decreasing order, our new lower bound is $\frac{54}{47} = 1.1489361\ldots$ vs. $\frac{8}{7} = 1.142857\ldots$. As a "byproduct" we gave a simple combinatorial proof for van Vliet's lower bound for the performance of on-line algorithms.

For the latter case we note that if the size of the largest elements is in the interval $(\frac{8}{29}, \frac{1}{2}]$ then First Fit Decreasing yields the upper bound $\frac{71}{60}$ (see [7]). However, our efforts to get a better result were not successful so far. It is true, that our improvements are very small in absolute values. However we did an exhaustive search for possible lists and we have not found worse ones. We strongly believe that the gap might be decreased only by defining better algorithms or one needs to find a new method for proving lower bounds.

The packing pattern technique was used for the two- and three-dimensional bin packing problems [5] and the best known lower bound for the on-line vector packing algorithms operates also with this technique (see [4]). Since in these cases the Sylvester sequence was used during the proof it is plausible that the application of the new series will also improve these lower bounds. We are convinced that this technique is usable for other classes of algorithms.

Acknowledgments

The authors are grateful to the reviewers' valuable comments that improved the manuscript.

References

[1] J. Csirik, G. Galambos, Gy. Turán, Some results on bin packing, in: Proceedings of EURO VI., Vienna, 1983.
[2] G. Galambos, A 1.6 lower bound for the two-dimensional on-line rectangle bin packing, Acta Cybernet. 10 (1991) 21–24.
[3] G. Galambos, J.B.G. Frenk, A simple proof of liang's lower bound for on-line bin packing and extension to the parametric case, Discrete Appl. Math. 41 (1993) 173–178.
[4] G Galambos, H. Kellerer, G. Woeginger, A lower bound for on-line vector-packing algorithms, Acta Cybernet. 11 (1993) 23–34.
[5] G. Galambos, A. van Vliet, Lower bounds for 1-, 2- and 3-dimensional on-line bin packing algorithms, Computing 52 (1994) 281–297.
[6] M.R. Garey, D.S. Johnson, Computer and Intractability: A Guide to the Theory of NP-completeness, W.H. Freeman, 1979.
[7] D.S. Johnson, Fast approximation algorithms for bin packing, J. Comput. System Sci. 8 (1974) 272–314.
[8] F.M. Liang, A lower bound for on-line bin packing, Inform. Process. Lett. 10 (1980) 76–79.
[9] S. Seiden, On the on-line bin packing problem, J. ACM 49 (2002) 640–671.
[10] S. Seiden, R. van Stee, L. Epstein, New bounds for variable-sized online bin packing, SIAM J. Comput. 32 (2003) 455–469.
[11] J. Sylvester, On a point in the theory of vulgar fractions, Amer. J. Math. 3 (1880) 332–335.
[12] A. van Vliet, An improved lower bound for on-line bin packing algorithms, Inform. Process. Lett. 43 (1992) 274–284.
[13] A. van Vliet, Lower bound and upper bounds for on-line bin packing and scheduling algorithms, Ph.D. Thesis, Tinbergen Institute Research Series no. 93.