

Two simple algorithms for bin covering

J. Csirik * J. B. G. Frenk † M. Labbé ‡ S. Zhang §

Dedicated to Professor Ferenc Gécseg on the occasion of his 60th birthday

Abstract

We define two simple algorithms for the bin covering problem and give their asymptotic performance.

1 Introduction

In this chapter we consider the following version of bin packing sometimes called *dual bin packing* or *bin covering*: given a list

$$L = (a_1, a_2, \dots, a_n)$$

of items with size $s(a_i)$ for each item a_i , and a bin capacity C ,

$$C > \max_{1 \leq i \leq n} s(a_i),$$

pack the elements of L into a maximum number of bins so that the sum of sizes in any bin is at least C . This means, that we have to fill as many bins as possible. It is clear, that we can normalize the problem so that C is equal to 1 and $s(a_i) < 1$ for every $1 \leq i \leq n$. The above problem was investigated for the first time by Assmann (cf.[1]) and Assmann *et al.*(cf.[2]). In particular, they showed that the problem is *NP*-hard. Furthermore, they provided the first approximation algorithms and proved their worst-case performance. Some average-case analysis was also performed.

We denote by $OPT(L)$ the optimal, i.e. the maximal number of filled bins for a list

$$L = (a_1, a_2, \dots, a_n)$$

*Department of Informatics, József Attila University, Árpád tér 2, H-6720 Szeged, Hungary

†Erasmus University of Rotterdam, Faculty of Economics, Postbus 1738, 3000 DR Rotterdam, Netherlands

‡Université Libre de Bruxelles, Mathématiques du Triomphe, 1050 Bruxelles, Belgium

§Erasmus University of Rotterdam, Faculty of Economics, Postbus 1738, 3000 DR Rotterdam, Netherlands

and we define for every $k \geq 1$

$$R_A(k) := \min \left\{ \frac{A(L)}{k} \mid OPT(L) = k \right\}, \quad (1)$$

where $A(L)$ denotes the number of bins filled by algorithm A applied to the list L . The *performance ratio* or *asymptotic worst case ratio* of A , denoted by R_A , is now given by

$$R_A := \liminf_{k \rightarrow \infty} R_A(k). \quad (2)$$

Clearly, $R_A(k) \leq 1$ for every $k \geq 1$, and hence $R_A \leq 1$.

For an equivalent definition of R_A we observe that $R_A \geq K_1$ if there exist two constants K_1 and K_2 such that

$$A(L) \geq K_1 \cdot OPT(L) + K_2 \quad (3)$$

for every list L . Clearly the largest possible K_1 satisfying this inequality equals R_A . By this definition it is obvious that a heuristic A_1 is at least as good as heuristic A_2 (from a worst case point of view) if $R_{A_1} \geq R_{A_2}$.

2 Preliminary results

It is very natural to adapt classical bin packing heuristics to the dual problem. So the first heuristic is **Dual Next Fit** (DNF).

1. Put the first element into the first bin.
2. While there is an unpacked item, do the following: Let a_i be the first unpacked element, and let B_j be the bin that is not yet filled (if all bins are filled, we take a new empty bin as B_j). Place a_i in bin B_j .

This algorithm has the nice property that it uses only one bin at a time and that it works on-line. The algorithm runs in $O(n)$ time. However, the asymptotic worst case bound of DNF is not very good (cf.[2]).

Lemma 1 $R_{DNF} = 1/2$.

From the 'bad' lists for this algorithm it follows, that - contrary to classical bin packing - sorting the items in nonincreasing order does *not* improve the performance of the heuristic. This implies, that the heuristic **Next Fit Decreasing** (NFD) of classical bin packing adapted to the dual bin packing problem also has a performance ratio of $1/2$.

The next idea is to use First Fit type heuristics instead of Next Fit, i.e. to use all opened bins instead of the last one. However, it has no meaning in this case, because after filling a bin, it is useless to place further items into it. That means that neither **First Fit** nor **First Fit Decreasing** adapted to the dual bin packing problem have a larger performance ratio than Next Fit.

An improvement of the performance ratio of $1/2$ was achieved by Assmann *et al.* (cf.[2]) by defining an *artificial upper bound* on the sum of sizes of elements placed into the same bin. This upper bound can be regarded as the capacity of a bin and leads to some similarity with classical bin packing. However, after packing the items by a good heuristic for the classical bin packing problem, it might happen that in some of the bins the sum of item sizes is less than 1. Hence we can use a second step to fill these bins. The algorithm based on the above observation, proposed by Assman *et al.*, is called **First Fit Decreasing with parameter r** (FFD_r) and proceeds as follows:

Let $1 < r < 2$.

Phase I. ("Classical FFD ")

1. Presort the items in L so that

$$s(a_1) \geq s(a_2) \geq \dots \geq s(a_n).$$

2. While there is still an unpacked element, do the following: Let a_i be the first unpacked item and let B_j be the first (leftmost) unfilled bin with a current total content smaller than or equal to $r - s(a_i)$. If such a bin exists, place a_i in B_j , otherwise open a new empty bin and pack a_i into this bin.

Phase II. (Repacking unfilled bins)

1. While there is more than one open nonfilled bin, remove an item from the rightmost such bin and add it to the leftmost one.

The time complexity of FFD_r can be seen to be $O(n \log n)$. For this algorithm the following result holds (cf.[2]).

Lemma 2 $R_{FFD_r} = 2/3$ for $4/3 < r < 3/2$.

Assmann *et al.* also suggested a further improvement by defining a really sophisticated algorithm, called **Iterated Lowest Fit Decreasing** ($ILFD$). To define this heuristic we consider first the following problem: Given the list L and a fixed number N of bins, what is the *maximum* possible value for the minimum bin level in a packing of L into N bins? From a good heuristic A for this problem we can derive a good approximation algorithm for the bin covering problem by iteratively applying this algorithm A . We denote by $A(L, N)$ the minimum bin level in the packing of L generated by the heuristic A if the number of bins is fixed by N . Now the algorithm iteratively applying A proceeds as follows:

ITERATED "A"

1. Let $UB = \sum_{i=1}^n s(a_i)$, $LB = 1$. (Clearly $LB \leq OPT(L) \leq UB$.)
2. While $UB - LB > 1$ take $N = \lfloor (LB + UB)/2 \rfloor$ and apply heuristic A . If $A(L, N) > 1$ take $LB = N$, otherwise $UB = N$.

The resulting algorithm gives a feasible solution of the dual bin packing problem with LB bins.

Clearly, the performance of this method depends on the choice of A . While the problem to be solved by A is closely related to multiprocessor scheduling problems, it seems natural to use for the heuristic A the **Lowest Fit Decreasing** ($LF D$) algorithm, as studied by Graham (cf.[4]) and Deyermeyer *et al.*(cf.[3]). This algorithm proceeds as follows:

1. Order L so that $s(a_1) \geq s(a_2) \geq \dots \geq s(a_n)$ and start with N empty bins.
2. While there is an unpacked item in L do the following: let a_i be the first unpacked item and let B_j be the bin with minimum level (in case of ties, choose the rightmost). Put a_i into B_j .

It is not difficult to verify that the time complexity of $ILFD$ is $O(n \log^2 n)$. Furthermore, one can prove the following result (cf.[2]).

Lemma 3 $R_{ILFD} = 3/4$.

3 Two new simple algorithms

Now we will show that the same performance bounds can also be achieved by simpler algorithms too. First we discuss the heuristic **Simple** (SI). This algorithm proceeds as follows:

1. Sort the items of list L into nonincreasing order, i.e. from now on we assume that

$$s(a_1) \geq s(a_2) \geq \dots \geq s(a_n).$$

2. Let k_1 denote the index satisfying

$$\sum_{i=1}^{k_1} s(a_i) < 1 \quad \text{and} \quad \sum_{i=1}^{k_1+1} s(a_i) \geq 1.$$

Pack the elements a_1, a_2, \dots, a_{k_1} into the first bin. Fill the remaining space in the bin with items from the end of the list, i.e. with a_n, a_{n-1}, \dots until the sum of sizes of items in the bin is at least equal to one and remove the packed items from the list.

3. Renumber the indices of the remaining items and repeat step 2 until the list is empty.

Lemma 4 For all lists L ,

$$SI(L) \geq \frac{2}{3} \cdot OPT(L) - \frac{2}{3}.$$

Proof. Let us assume that the last item in the last filled bin is a_{last} and distinguish the cases $s(a_{\text{last}}) \leq 1/2$ and $s(a_{\text{last}}) > 1/2$.

If $s(a_{\text{last}}) \leq 1/2$ then the total sum of item sizes in the last filled bin is bounded above by $1 + s(a_{\text{last}}) \leq 3/2$. Moreover, since all the last items in the remaining filled bins are by the definition of *SI* always smaller than or equal to $s(a_{\text{last}})$ we obtain that the total sum of item sizes of all the filled bins is bounded above by $3/2$. As we have at most one non-filled bin, this implies

$$3/2 \cdot SI(L) + 1 \geq s(L).$$

Since $s(L) \geq OPT(L)$ we obtain

$$SI(L) \geq \frac{2}{3}(OPT(L) - 1)$$

for all lists L and so the lemma holds in this case.

If $s(a_{\text{last}}) > 1/2$, we only consider the case where all the opened bins are filled. If this does not hold (i.e. we have one non-filled bin) the proof can be easily adapted. Now it is clear that the *SI*-packing has the following structure:

$$\begin{array}{cccccccc} a_{n_1} & a_{n_2} & a_{n_3} & \cdots & a_{n_k} & & & \\ \vdots & \vdots & \vdots & & \vdots & & & \\ a_n & a_{n_1-1} & a_{n_2-1} & & a_{n_{k-1}-1} & a_{n_k-1} & a_{n_k-2} & a_{n_k-SI(L)+k} \\ a_1 & a_2 & a_3 & \cdots & a_k & a_{k+1} & a_{k+2} & \cdots & a_{SI(L)} \\ B_1 & B_2 & B_3 & \cdots & B_k & B_{k+1} & B_{k+2} & & B_{SI(L)} \end{array}$$

where $a_{n_k-SI(L)+k} = a_{\text{last}}$, $n_k - SI(L) + k = SI(L) + 1$, $s(a_{n_k-SI(L)+k}) > 1/2$ and $n > n_1 > n_2 > \dots > n_k$.

Call the elements $a_n, a_{n_1-1}, \dots, a_{n_1+1}, a_{n_1-1}, \dots, a_{n_2+1}, \dots, a_{n_{k-1}-1}, \dots, a_{n_k+1}$ type-*A* items and the remaining (i.e. the first and the last element in each bin) type-*B* items, and consider the optimal packing. Define

$$\begin{aligned} k_A &:= \# \text{ bins in the optimal packing with only type-}A \text{ items,} \\ k_{AB} &:= \# \text{ bins in the optimal packing with exactly one type-}B \text{ item,} \\ k_{BB} &:= \# \text{ bins in the optimal packing with more than one type-}B \text{ items.} \end{aligned}$$

Clearly

$$OPT(L) = k_A + k_{AB} + k_{BB}. \quad (4)$$

Moreover, by the definition of *SI* we observe that

$$\sum_{i \in \text{type-}A} s(a_i) + \sum_{i=1}^k s(a_i) < k$$

and this implies, since a_1, a_2, \dots, a_k are the biggest B -items, that $k_{AB} < k$. Applying the above inequality again and using $s(a_i) > 1/2$, for every $i \leq k$, we get:

$$\begin{aligned} k_A + k_{AB} &\leq \sum_{i \in \text{type-A}} s(a_i) + \sum_{i=1}^{k_{AB}} s(a_i) < \\ &< \frac{k + k_{AB}}{2} \leq \frac{SI(L) + k_{AB}}{2}. \end{aligned} \quad (5)$$

Finally, in order to obtain an upper bound for k_{BB} we note that the total number of type- B items in bins with more than one type- B item is given by $2 \cdot SI(L) - k_{AB}$ and hence

$$k_{BB} \leq \frac{2 \cdot SI(L) - k_{AB}}{2}. \quad (6)$$

By (4) and the upper bounds in (5) and (6) we obtain

$$OPT(L) \leq \frac{SI(L) + k_{AB}}{2} + \frac{2 \cdot SI(L) - k_{AB}}{2} = 3/2 \cdot SI(L).$$

and hence the desired result is proved. □

In the above lemma we proved that $R_{SI} \geq 2/3$. Furthermore, this lower bound can be achieved, as will be shown by the next result.

Theorem 5 $R_{SI} = 2/3$.

Proof. Let us define the lists $L_n, n \geq 1$ by the sizes of the elements and consider

$$L_n = \left(\frac{3}{4}, \underbrace{\frac{1}{2} - \varepsilon, \frac{1}{2} - \varepsilon, \dots, \frac{1}{2} - \varepsilon}_{6n+1 \text{ times}}, \underbrace{2\varepsilon, 2\varepsilon, \dots, 2\varepsilon}_{3n \text{ times}} \right).$$

If

$$\varepsilon < \frac{1}{24n}$$

then

$$OPT(L_n) = 3n + 1$$

and

$$SI(L_n) = 2n + 1.$$

Hence we obtain

$$\frac{SI(L_n)}{OPT(L_n)} = \frac{2}{3} + \frac{1}{9n+3}$$

and this implies $\lim_{n \rightarrow \infty} \frac{SI(L_n)}{OPT(L_n)} = \frac{2}{3}$. Applying Lemma 4 finally yields the desired result. \square

It is easy to see that similarly we can characterize the performance of the heuristic *SI* if $s(a_i) \leq 1/k$ for all a_i , where k is some positive integer. For this case the next result holds.

Theorem 6 *If $s(a_i) \leq 1/k$ for all items in $L = (a_1, a_2, \dots, a_n)$, where k is a positive integer, then the worst case performance of the heuristic *SI* is at least $\frac{k+1}{k+2}$.*

Proof. The proof directly follows that of Lemma 4 .

Finally, we consider an improved version of the *SI*-heuristic. Before introducing this so called **Improved simple heuristic** (*ISI*) we divide the list L into the following three parts.

- | | | |
|------|--|-------------|
| i) | $s(x_1) \geq s(x_2) \geq \dots \geq s(x_p) \geq 1/2$ | (X-sublist) |
| ii) | $1/2 > s(y_1) \geq s(y_2) \geq \dots \geq s(y_r) \geq 1/3$ | (Y-sublist) |
| iii) | $1/3 > s(z_1) \geq s(z_2) \geq \dots \geq s(z_m)$ | (Z-sublist) |

Clearly $p + r + m = n$. Now the *ISI*-heuristic is defined as follows.

Phase 1. If $s(x_1) \geq s(y_1) + s(y_2)$, then pack x_1 into an empty bin, otherwise pack y_1 and y_2 into an empty bin. Fill the just opened bin with elements from the end of the *Z*-sublist, i.e. with z_m, z_{m-1}, \dots until the bin is filled. Remove the packed elements from the corresponding sublists and repeat packing until either $X \cup Y$ or *Z* is empty.

Phase 2. If after phase 1, $X \cup Y$ is empty, pack the remaining elements in the *Z*-sublist according to the Next-Fit heuristic. Otherwise, if *Z* is empty, pack the remaining x -elements by two in a bin and the remaining y -elements by three.

For the above heuristic the next result holds.

Lemma 7 *The worst case performance ratio of the heuristic *ISI* is at least equal to $3/4$.*

Proof. To verify the above result it is sufficient to prove that $ISI(L) \geq 3/4(OPT(L) - 4)$ for all lists L . This is easy if $X \cup Y$ is empty after phase 1. Observe that in this case the last elements of all filled bins (after the execution of the heuristic) are elements from the *Z*-sublist and hence the sum of sizes in each filled bin is bounded from above by $4/3$. By an argument similar to that used in the first part of Lemma 4 the desired inequality follows.

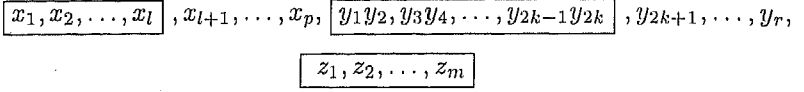


Figure 1: The packing by Improved Simple.

Subsequently we consider the case where the Z -sublist is empty after phase 1 and assume that x_1, x_2, \dots, x_l from the X -sublist and $y_1y_2, y_3y_4, \dots, y_{2k-1}y_{2k}$ from the Y -sublist are used in the first phase (cf. Figure 1).

By the definition of the heuristic we obtain

$$ISI(L) \geq l + k + \left\lfloor \frac{p-l}{2} \right\rfloor + \left\lfloor \frac{r-2k}{3} \right\rfloor - 1 \geq \frac{l}{2} + \frac{k}{3} + \frac{p}{2} + \frac{r}{3} - 3. \quad (7)$$

We now have to derive an upper bound on $OPT(L)$. In order to do so call the elements of the Z -sublist, which were packed last in a filled bin by the ISI -heuristic, B -items, and the remaining z -elements A -items and consider the optimal packing. We now rename A -items A^* -items if these A -items in the optimal packing are packed either with only A -items or with one or two y -elements or with exactly one x -element and define

$k_{A^*} := \#$ bins in the optimal packing with only A^* items,

$k_{A^*X} := \#$ bins in the optimal packing with only A^* items and exactly one element from the X -sublist,

$k_{A^*Y} := \#$ bins in the optimal packing with only A^* items and exactly one element from the Y -sublist,

$k_{A^*YY} := \#$ bins in the optimal packing with only A^* items and exactly two elements from the Y -sublist,

$k_{others} := \#$ of other bins in the optimal packing.

Clearly,

$$OPT(L) = k_{A^*} + k_{A^*X} + k_{A^*Y} + k_{A^*YY} + k_{others}. \quad (8)$$

To obtain an upper bound on the first four terms in (8) we note by the construction of the ISI -packing that

$$\sum_{i \in \text{type}-A^*} s(a_i) + \sum_{i=1}^l s(x_i) + \sum_{i=1}^{2k} s(y_i) < l + k \quad (9)$$

Furthermore, if $k_{A^*} + k_{A^*X} + k_{A^*Y} + k_{A^*YY} > l + k$, it is possible by the feasibility of the optimal packing and the definition of A^* items to pack more than $l + k$ bins in the first phase of the ISI heuristic and since this does not hold we get

$$k_{A^*} + k_{A^*X} + k_{A^*Y} + k_{A^*YY} \leq l + k. \quad (10)$$

To simplify notations, we now define $l' := k_{A^*X}$ and $2k' := k_{A^*Y} + 2k_{A^*YY}$. Then, by (10), it follows immediately that

$$l' + k' \leq k_{A^*} + k_{A^*X} + k_{A^*Y} + k_{A^*YY} \leq l + k. \quad (11)$$

It turns out that the inequality derived in (10) can be improved as follows. Clearly,

$$k_{A^*} + k_{A^*X} + k_{A^*Y} + k_{A^*YY} \leq \sum_{i \in \text{type}-A^*} s(a_i) + \sum_{i=1}^{l'} s(x_i) + \sum_{i=1}^{2k'} s(y_i). \quad (12)$$

By (9), it follows that the upper bound in (12) can be bounded from above by

$$U := l + k + \sum_{i=1}^{l'} s(x_i) - \sum_{i=1}^l s(x_i) + \sum_{i=1}^{2k'} s(y_i) - \sum_{i=1}^{2k} s(y_i). \quad (13)$$

In order to bound U we distinguish the following four cases:

- i) $l \geq l'$ and $2k \geq 2k'$,
- ii) $l < l'$ and $2k \geq 2k'$,
- iii) $l \geq l'$ and $2k < 2k'$,
- iv) $l < l'$ and $2k < 2k'$.

By (11) case (iv) will never occur and hence we only have to consider i), ii) and iii).

Clearly, if i) holds,

$$U = l + k - \sum_{i=l'+1}^l s(x_i) - \sum_{i=2k'+1}^{2k} s(y_i). \quad (14)$$

By the definition of *ISI* (cf. Figure 1), we now observe that

$$s(x_i) \geq s(x_l) \geq s(y_{2k+1}) + s(y_{2k+2}) > \frac{2}{3},$$

for every $i \leq l$, if $r \geq 2k + 2$.

Moreover, if $r \leq 2k + 1$, then $s(x_l)$ might be smaller than $2/3$ and hence the remaining elements in the X -sublist after phase 1 are always smaller than $2/3$. This observation implies that in phase 2 the sum of the sizes in a filled bin is bounded above by $4/3$ and together with the argument that in phase 1 the sum of sizes in a filled bin are also bounded above by $4/3$ the desired inequality follows. By this observation we may therefore assume that $s(x_i) \geq 2/3$ for every $i \leq l$ and this yields, by (14),

$$U \leq l + k - \frac{2}{3}(l - l') - \frac{1}{3}(2k - 2k') = \frac{1}{3}(l + k) + \frac{2}{3}(l' + k') \quad (15)$$

If ii) holds we obtain by (13) that

$$U = l + k + \sum_{i=l+1}^{l'} s(x_i) - \sum_{i=2k'+1}^{2k} s(y_i). \quad (16)$$

Observe by the definition of the first phase of the *ISI*-heuristic (cf. Figure 1) that the sum of sizes of any two y_i elements, $i = 2k' + 1, \dots, 2k$ is always bigger than the size of any x_i -element, $i = l + 1, \dots, l'$. This implies

$$\sum_{i=l+1}^{l'} s(x_i) \leq \sum_{i=2k'+1}^{2k'+2l'-2l} s(y_i)$$

and since by (11) $2k' + 2l' - 2l \leq 2k$, we obtain

$$\begin{aligned} U &\leq l + k - \sum_{i=2k'+2l'-2l+1}^{2k} s(y_i) \leq l + k - \frac{1}{3}(2k - (2k' + 2l' - 2l)) \\ &= \frac{l+k}{3} + \frac{2}{3}(l' + k') \end{aligned} \quad (17)$$

which is the same as inequality (15).

In order to bound U by the same upper bound as in (17), when iii) holds, we use a similar argument, i.e. we replace the remaining y -items by an x -item and using the lower bound for x_i yields the desired result.

Hence we have proved that in all cases the improved upper bound

$$U \leq \frac{l+k}{3} + \frac{2}{3}(l' + k') \quad (18)$$

holds.

By (8), we also need an upper bound on k_{others} . These remaining k_{others} bins contain $p - l'$ x -items, $r - 2k'$ y -items, $l + k$ B -items and some A -items. By the definition of k_A these A -items are always contained in a bin with some of the above elements and hence do not count in the computation of an upper bound for k_{others} . For this upper bound computation we consider four subcases.

$$\text{A) } l + k \geq p - l' + \frac{r-2k'}{2}.$$

If this holds, the best we can hope for is to pack one X -element with one B -item, two Y -elements with one B -item and the remaining B -items by four. Hence

$$k_{others} \leq p - l' + \frac{r - 2k'}{2} + \frac{l + k - (p - l' + \frac{r-2k'}{2})}{4} \quad (19)$$

and by (18) and (8) this implies

$$OPT(L) \leq \frac{7}{12}(l+k) + \frac{3}{4}p + \frac{3}{8}r - \frac{1}{12}(l'+k').$$

Since by A) $l' + k' \geq p + r/2 - l - k$ we conclude by the above inequality that

$$OPT(L) \leq \frac{2}{3}(l+k+p) + \frac{1}{3}r. \quad (20)$$

By (7),(20) and the inequality $r \geq 2k$ (cf.Figure 1) we finally obtain

$$\frac{4}{3}ISI(L) \geq \frac{2}{3}(l+p) + \frac{4}{9}(k+r) - 4 \geq OPT(L) - 4. \quad (21)$$

$$\text{B) } p - l' < l + k < p - l' + \frac{r-2k'}{2}.$$

If this holds, the best we can hope for is to pack one X -element with one B -item, the first part of the Y -elements by two with an additional B -item and the remaining Y -elements by three. Hence

$$\begin{aligned} k_{others} &\leq p - l' + ((l+k) - (p - l')) + \frac{1}{3}((r - 2k') - 2((l+k) - (p - l'))) \\ &= l + k + \frac{1}{3}((r - 2k') - 2((l+k) - (p - l'))) \end{aligned}$$

and by (18) and (8) this yields

$$OPT(L) \leq \frac{2}{3}(l+k+p) + \frac{r}{3}.$$

Clearly this is the same upper bound as discussed in (20) and so (21) also holds.

$$\text{C) } p - l' - (r - 2k') < l + k \leq p - l'.$$

If this holds, it follows that $r - 2k' - (p - l' - (l + k)) > 0$ and so the best we can hope for is to pack $l + k$ X -items with one additional B -item, the remaining part of the X -items, i.e. $p - l' - (l + k)$, with one additional Y -item and the rest of the Y -items, i.e. $r - 2k' - (p - l' - (l + k))$, by three. Hence

$$\begin{aligned} k_{others} &\leq l + k + p - l' - (l + k) + \frac{r - 2k' - (p - l' - (l + k))}{3} \\ &= p - l' + \frac{r - 2k' - (p - l' - (l + k))}{3} \end{aligned}$$

and by (18) and (8) this implies

$$OPT(L) \leq \frac{2}{3}(p+l+k) + \frac{r}{3}.$$

Clearly this is the same upper bound as discussed in (20) and so (21) also holds.

$$\text{D) } l + k \leq p - l' - (r - 2k').$$

If this holds, the best we can hope for is to pack $l + k$ X -items with one additional B -item, $r - 2k'$ X -items with one additional Y -item and the remaining part of the X -items, i.e. $p - l' - (l + k) - (r - 2k')$, by two. Hence

$$k_{\text{others}} \leq l + k + r - 2k' + \frac{p - l' - (l + k) - (r - 2k')}{2}$$

and by (18) and (8) this implies

$$OPT(L) \leq \frac{5}{6}(l + k) + \frac{r}{2} + \frac{p}{2} + \frac{1}{6}(l' - 2k'). \quad (22)$$

By D) it follows that $l' - 2k' \leq p - r - l - k$ and substituting this in (22) yields

$$OPT(L) \leq \frac{2}{3}(l + k + p) + \frac{r}{3}.$$

Clearly this is the same upper bound as discussed in (20) and so (21) also holds. This last subcase concludes the proof of our result. \square

In the above lemma we proved that $R_{ISI} \geq 3/4$. Furthermore, this lower bound can be achieved, as will be shown by the next result.

Theorem 8 $R_{ISI} = 3/4$.

Proof. Consider the lists L_n with

$$L_n = \left(\frac{1}{3} + \varepsilon_n, \frac{1}{3} + \varepsilon_n, \underbrace{\frac{1}{3} - 2\varepsilon_n, \frac{1}{3} - 2\varepsilon_n, \dots, \frac{1}{3} - 2\varepsilon_n}_{12n+1 \text{ times}}, \underbrace{6\varepsilon_n, 6\varepsilon_n, \dots, 6\varepsilon_n}_{4n \text{ times}} \right).$$

It is easy to verify that

$$OPT(L_n) = 4n + 1.$$

If ε_n is chosen in such a way that the first two items together with the last $4n$ items do not fill the first bin then

$$ISI(L_n) = 3n + 1.$$

Hence $\lim_{n \rightarrow \infty} \frac{ISI(L_n)}{OPT(L_n)} = \frac{3}{4}$ and by Lemma 2 the desired result follows. \square

As a last remark we note that for the above lists the ISI -packing is essentially the same as the simple packing.

4 Open question

It would be interesting to find a heuristic with a performance ratio greater than $3/4$.

References

- [1] Assmann, S.F.: Problems in Discrete Applied Mathematics, Ph.D. Thesis, Mathematics Department, MIT, Cambridge, MA, 1983.
- [2] Assmann S. F., Johnson D. S., Kleitman D. J., Leung J. Y.-T.: On a dual version of the one-dimensional bin packing problem, *J. of Algorithms* 5(1984), 502-525.
- [3] Deyermeyer, B.L., Friesen, D.K., Langston, M.A.: Scheduling to maximize the minimum processor finish time in a multiprocessor system, *SIAM J. Alg. Disc. Meth.* 3(1982), 190-196.
- [4] Graham, R.L.: Bounds on multiprocessing timing anomalies, *SIAM J. Appl. Math.* 17(1969), 263-269.