Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT

Information Processing Letters 89 (2004) 215–221

Information Processing Letters

www.elsevier.com/locate/ipl

Approximation algorithms for a hierarchically structured bin packing problem [☆]

Bruno Codenotti [a,1], Gianluca De Marco [b], Mauro Leoncini [b,c], Manuela Montangero [b,*], Massimo Santini [b,c]

[a] *Department of Computer Science, The University of Iowa, Iowa City, IA, USA*
[b] *Istituto di Informatica e Telematica, Consiglio Nazionale delle Ricerche, Pisa, Italy*
[c] *Dipartimento di Scienze Sociali, Cognitive e Quantitative, Università di Modena e Reggio Emilia, Modena, Italy*

Received 10 October 2003; received in revised form 5 December 2003

Communicated by L. Boasson

Abstract

In this paper we study a variant of the bin packing problem in which the items to be packed are structured as the leaves of a tree. The problem is motivated by document organization and retrieval. We show that the problem is NP-hard and we give approximation algorithms for the general case and for the particular case in which all the items have the same size.
© 2003 Elsevier B.V. All rights reserved.

Keywords: Algorithms; Approximation algorithms; Bin packing; NP-hardness

1. Introduction

Bin packing is the problem of arranging (packing) a sequence $X = \{x_1, \dots, x_n\}$ of items, each of size $s(x_i) \in (0, 1]$, into the minimum possible number of containers (bins) of unit capacity. The sequence X is unstructured and the "geometry" of both items and bins is one-dimensional (i.e., a set $\overline{X} \subseteq X$ such that $\sum_{\overline{X}} s(x_i) \leqslant 1$ always fits in an empty bin) [2].

The problem investigated in this paper can be regarded as a variant of classical bin packing, in which the items are the leaves of a tree and the goal is packing while preserving some locality properties, i.e., leaves whose lowest common ancestor has low height should be possibly packed into the same bin. This problem originates in the area of document organization and retrieval, where one is confronted with searching issues on very large structured, tree-like ontologies, or clusters [1,4,5,9].

We identify the following algorithmic problem that, informally, can be defined as the problem of distributing hierarchically structured objects into different repositories in a way that the access to subsets of related objects involves as few repositories as possible.

[☆] Work partially supported by the European RTN Project under contract HPRN-CT-2002-00278, COMBSTRU.
[*] Corresponding author.
E-mail addresses: bcodenot@cs.uiowa.edu (B. Codenotti), gianluca.demarco@iit.cnr.it (G. De Marco), leoncini@acm.org (M. Leoncini), manuela.montangero@iit.cnr.it (M. Montangero), msantini@unimore.it (M. Santini).
[1] On leave from IIT–CNR.

The problem. Given a tree T and a vertex $v \in T$, $S_T(v)$ will denote the subtree rooted at v, $L_T(v)$ and $N_T(v)$ will denote the set of leaves and internal nodes of $S_T(v)$, respectively.

Definition 1 (*Node dispersal number*). Let $T = (V, E)$ be a tree and $\mathcal{P}$ a partition of L_T. The *node dispersal number* for a node v of T given $\mathcal{P}$ is

$$\rho(v, \mathcal{P}) = \bigl|\bigl\{A \in \mathcal{P} \mid L_T(v) \cap A \neq \emptyset\bigr\}\bigr|.$$

In other words, given a vertex v, $\rho(v, \mathcal{P})$ counts how many sets of $\mathcal{P}$ intersect $L_T(v)$.

Definition 2 (*Structured Bin Packing problem*). Given a tree T, a positive weight function $w : L_T \to N^+$ and a positive integer *capacity* $c \geqslant \max_{v \in T} w(v)$, find a partition $\mathcal{P}$ of L_T such that

(1) for every set $A \in \mathcal{P}$ the total weight of the leaves in A is bounded by c;
(2) the total node dispersal number $\rho(\mathcal{P}) = \sum_{v \in N_T} \rho(v, \mathcal{P})$ is minimum.

A *feasible solution* is a partition $\mathcal{P}$ satisfying only condition (1).

According to Definition 2, and identifying the problem of packing the items in L_v with that of packing v, we see that the measure $\rho(\mathcal{P})$ is strictly related to the average dispersal of internal nodes (say, categories or clusters) over repositories.

Our results. The paper is organized as follows. In Section 2 we prove that the Structured Bin Packing problem is NP-hard. In Section 3 we give a 2-approximation algorithm and we show that the analysis of the worst case performance ratio is tight. In Section 4 we study the case in which the weight function is constant. We first prove that the problem remains NP-hard also in this case and then we give an approximation algorithm whose worst case performance ratio is $3/2$.

2. Preliminaries

In order to simplify the notation, we will often drop the subscript when T is clear from the context, and we will simply use L to denote the leaves of T.

In the following we will denote with $W_T(v)$ the total weight of the leaves in subtree $S_T(v)$. Given a set of nodes $V' \subseteq V$ and a partition $\mathcal{P}$, we define $\rho(V', \mathcal{P}) = \sum_{v \in V'} \rho(v, \mathcal{P})$; given a subtree $T' = (V', E')$ and a partition $\mathcal{P}$, we define $\rho(T', \mathcal{P}) = \rho(V', \mathcal{P})$.

Lemma 1. *Let (T, w, c) be an instance of the Structured Bin Packing problem. Then, for every vertex v in T and for every partition $\mathcal{P}$ of L_T we have*

$$\frac{W_T(v)}{c} \leqslant \rho(v, \mathcal{P}) \leqslant \sum \rho(u, \mathcal{P}),$$

where the summation extends over the children of v.

Proof. On one hand, a set of the partition can not weight more than c, thus $\lceil W_T(v)/c \rceil$ sets are necessary. On the other hand, if a set $A \in \mathcal{P}$ intersects $L_T(v)$, then there exists at least one child u of v such that A intersects $L_T(u)$. □

The Structured Bin Packing problem is NP-hard by reduction from Minimum Bin Packing.

Definition 3 (*Minimum Bin Packing*). Given a finite set of items with positive integer sizes $s_1, \ldots, s_n$ and a positive integer capacity $c \geqslant \max s_i$, find a partition $\mathcal{Q}$ of $\{1, \ldots, n\}$ such that $\sum_{i \in A} s_i \leqslant c$ for every $A \in \mathcal{Q}$ and $|\mathcal{Q}|$ is minimum.

We shall refer to the sum $\sum_{i \in A} s_i$ as to the *level* of A.

The Minimum Bin Packing problem is NP-hard, approximable within $3/2$ [8] but not approximable within $3/2 - \varepsilon$ for any $\varepsilon > 0$ [6,3]. Nevertheless it admits a FPTAS$^\infty$ and is approximable within $1 + \varepsilon$ in time polynomial in $1/\varepsilon$, where $\varepsilon = \mathrm{O}(\log^2(\mathrm{opt})/\mathrm{opt})$ [7].

Theorem 1. *The Structured Bin Packing problem is* NP-*hard and it is not approximable within $3/2 - \varepsilon$ for any $\varepsilon > 0$.*

Proof. Reduction is from Minimum Bin Packing. Suppose we are given n items of sizes $s_1, \ldots, s_n$ and capacity c. Consider tree T having one internal node r and n leaves $l_1, \ldots, l_n$ such that $w(l_i) = s_i$, for each i. The value $\rho(\mathcal{P}) = \rho(r, \mathcal{P})$ is exactly the minimum number of bins needed to pack the n items.

Since the reduction is cost preserving, the inapproximability result holds. □

3. Arbitrary weights on leaves

A naive solution to the Structured Bin Packing problem consists of running an accurate off-line bin-packing algorithm on input the leaves and using the contents of the bin to determine the partition. However, this strategy can lead to high node dispersals. To see this, consider the following instance of the Structured Bin Packing problem (see Fig. 1): the root is connected with k internal nodes, each of which is connected with four leaves, with weights $c/4$, $c/4 - 1$, $c/4 - 2$ and $c/4 - 3$. It is easy to see that the optimal solution $\mathcal{P}_O$ has cost $\rho(\mathcal{P}_O) = 2k$ and consists of simply putting the leaves with same parent node into a single bin (or partition). Indeed, the dispersal of any internal node is one (which is the minimum possible) and the dispersal of the root is k (again the minimum, since k bins are required).

Consider now using First Fit Decreasing (FFD) for packing the leaves. It is again easy to see that using the partition produced by FFD each of the k internal node has dispersal 4 while the root has still dispersal k. This shows that the asymptotic performance ratio of this algorithm (derived from FFD) is at least $5/2$. However, by appropriately modifying the input instance, the ratio can be made much worse.

The example also shows that minimizing the number of bins (sets in the partition) is not a good idea, in general. In fact, even an exact (non-polynomial time) bin packing algorithm can be easily forced to put in a bin only items belonging to different subtrees. Note, in this respect, that FFD optimally places the items in the bins. Similar conclusions hold for the other accurate off-line or semi-online algorithm [2].

In the rest of this section we present a recursive 2-approximation algorithm to solve the Structured Bin Packing problem. We prove that the analysis of the worst case performance ratio is tight by giving a family of trees for which the approximation factor approaches two.

The input of the recursive call is a vertex v and the output is a partition of $L(v)$: if v is a leaf then the output is simply the singleton composed by v. Otherwise a recursive call is made on the children of v collecting the results in a partition $\mathcal{P}$ of $L(v)$. Considering the cardinality of each set in $\mathcal{P}$ as an item size, the algorithm computes a feasible solution to the Minimum Bin Packing problem. Finally, the recursive call returns a coarser partition obtained from $\mathcal{P}$ by merging the sets that have been packed together.

Let $\mathrm{BP}(c; s_1, \ldots, s_n)$ be an approximation algorithm for the Minimum Bin Packing problem. We say that BP is a *reasonable bin packer* if the levels $l_1 \geqslant l_2 \geqslant \cdots \geqslant l_m$ of the bins in the solution computed by BP are such that (1) $l_i > c/2$, for all $1 \leqslant i < m$ and (2) $l_{m-1} + l_m > c$. These conditions together simply say that the solution is minimal, in the sense that any two bins can not be merged together without violating the capacity constraint.

The recursive procedure of the approximation algorithm for the problem is the following:

```
Approx-SBP ($v$)
if $v$ is a leaf then return $\{v\}$;
for each child $u_i$ of $v$ ($i = 1, \ldots, n$)
    $P_i$ = Approx-SBP($u_i$);
$\mathcal{Q} = \mathrm{BP}(c; |P_1|, \ldots, |P_n|)$;
return $\mathcal{P} = \{A \mid A = \bigcup_{i \in Q} P_i,\ Q \in \mathcal{Q}\}$.
```

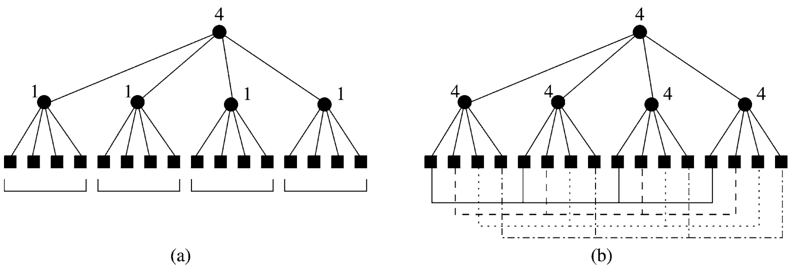

Fig. 1. Example of the use of FFD with $k = 4$. Leaves in each subtree have weight $c/4$, $c/4 - 1$, $c/4 - 2$ and $c/4 - 3$, going from left to right. (a) Optimal solution to the Structured Bin Packing problem. (b) Solution obtained using FFD on the set of leaves. Lines connect the leaves that are packed in the same bin.

Lemma 2. *Given a tree $T = (V, E)$ and a positive integer capacity c, let $\mathcal{P}_O$ be the optimal partition of the leaves and let $\mathcal{P}_A$ be the one found by* `Approx-SBP` *using a reasonable bin packer* `BP`. *For every vertex $v \in V$ we have*

$$2 \cdot \rho(v, \mathcal{P}_O) \geqslant \rho(v, \mathcal{P}_A).$$

Proof. Suppose that $\rho(v, \mathcal{P}_O) < \rho(v, \mathcal{P}_A) = m$, and let $l_1 \geqslant l_2 \geqslant \cdots \geqslant l_m$ be the levels of the solution returned by `Approx-SBP` (v) at node v.

Since `BP` is reasonable, observing that the sum of the levels is $W_T(v)$, we have

$$\begin{aligned} W_T(v) &= \sum_{i=1}^{m-2} l_i + (l_{m-1} + l_m) \\ &> (m-2)c/2 + c = mc/2, \end{aligned}$$

that implies $2 \cdot W_T(v)/c > m$. From Lemma 1 we have that $\rho(v, \mathcal{P}_O) \geqslant W_T(v)/c$ and hence

$$2 \cdot \rho(v, \mathcal{P}_O) \geqslant 2 \cdot \frac{W_T(v)}{c} > m = \rho(v, \mathcal{P}_A). \qquad \square$$

Theorem 2. `Approx-SBP` *is a 2-approximation algorithm.*

Proof. Using the same notation of Lemma 2, we have

$$\begin{aligned} 2 \cdot \rho(\mathcal{P}_O) &= 2 \cdot \sum_{v \in V} \rho(v, \mathcal{P}_O) \\ &\geqslant \sum_{v \in V} \rho(v, \mathcal{P}_A) = \rho(\mathcal{P}_A). \qquad \square \end{aligned}$$

The bound of Theorem 2 is tight. Indeed we have the following result.

Theorem 3. *There exists a family of trees for which the asymptotic worst case performance ratio of* `Approx-SBP` *approaches* 2.

Proof. Given a positive integer $n > 0$ consider the pair (T, c) where $c = 4n$ and tree $T = (N \cup L, E)$ is defined as follows:

$$N = \{v_k \mid 0 \leqslant k < 2n-1\} \cup \{z_k \mid 1 \leqslant k < 2n-1\}$$

is the set of internal nodes and

$$L = \big\{l_k \mid 0 \leqslant k < (2n-1)(2n+1)\big\}$$

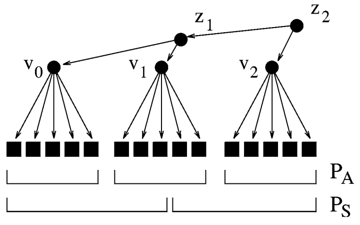

Fig. 2. Example of T for $n = 2$ and $c = 8$.

is the set of leaves. Edges are such that $L(v_k) = \{l_i \mid \lfloor i/(2n+1) \rfloor = k\}$ (for $0 \leqslant k < 2n-1$), and the children of the z_k's are $\{z_{k-1}, v_k\}$ (for $1 < k < 2n-1$), while the children of z_1 are $\{v_0, v_1\}$. Also, for each $l_k \in L_T$, we set $w(l_k) = 1$ (see Fig. 2 for an example).

First of all, consider the solution $\mathcal{P}_A$ of `Approx-SBP` on input (T, c). As it is easy to check $\mathcal{P}_A = \{L(v_k) \mid 0 \leqslant k < 2n-1\}$, since `Approx-SBP`$(v_k) = L(v_k)$ (for $0 \leqslant k < 2n-1$) and no such sets can be merged, since $W(v_k) = |L(v_k)| = 2n+1 > c/2$. Hence, $\rho(v_k, \mathcal{P}_A) = 1$ (for $0 \leqslant k < 2n-1$) and $\rho(z_k, \mathcal{P}_A) = k+1$ (for $1 \leqslant k < 2n-1$), therefore

$$\begin{aligned} \rho(\mathcal{P}_A) &= (2n-1) \times 1 + \sum_{1 \leqslant k < 2n-1} (k+1) \\ &= 2n^2 + \mathrm{O}(n). \end{aligned}$$

Now, consider the solution of an algorithm (called `Simple` from now onwards) that packs the leaves traversing the tree frontier from left to right in sets of size c. Let $S_k = \{l_i \mid \lfloor i/4n \rfloor = k\}$, for $0 \leqslant k < \lceil |L|/4n \rceil = n$, then the solution computed by `Simple` is $\mathcal{P}_S = \{S_k \mid 0 \leqslant k < n\}$ and

$$\begin{aligned} &L(v_{2k}) \subseteq S_k && 0 \leqslant k < n, \\ &L(v_{2k+1}) \subseteq S_k \cup S_{k+1} && 0 \leqslant k < n-1, \\ &L(z_k) \subseteq \bigcup_{0 \leqslant i < \lceil 1+k/2 \rceil} S_i && 0 \leqslant k < 2n-1 \end{aligned}$$

so that $\rho(v_k, \mathcal{P}_S) = 1 + (k \bmod 2) \leqslant 2$ (for $0 \leqslant k < 2n-1$) and $\rho(z_k, \mathcal{P}_S) \leqslant \lceil 1 + k/2 \rceil$ (for $1 \leqslant k < 2n-1$). Hence

$$\begin{aligned} \rho(\mathcal{P}_S) &\leqslant (2n-1) \times 2 + \sum_{1 \leqslant k < 2n-1} \lceil 1 + k/2 \rceil \\ &= 2 \sum_{2 \leqslant i \leqslant n} i = n^2 + \mathrm{O}(n). \end{aligned}$$

Finally, given the optimal solution $\mathcal{P}_O$, the asymptotic worst case performance ratio of `Approx-SBP` on input (T, c) can be bounded as follows:

$$\frac{\rho(\mathcal{P}_A)}{\rho(\mathcal{P}_O)} \geqslant \frac{\rho(\mathcal{P}_A)}{\rho(\mathcal{P}_S)} \geqslant \frac{2n^2 + \mathrm{O}(n)}{n^2 + \mathrm{O}(n)} = 2 + \mathrm{o}(1). \qquad \square$$

4. Unitary weights on leaves

In this section we study the case in which the weight function is constant. Without loss of generality, we can assume that $w(v) = 1$ for each leaf $v \in L_T$ and observe that, under this hypothesis, we always have $W_T(v) = |L_T(v)|$, for each v in the tree.

We first show that the problem is still NP-hard and then that in this case it is possible to use a simpler algorithm to obtain better performance ratio.

Lemma 3. *For every tree $T = (N \cup L, E)$ having all the leaves at distance 2 from the root r and for every partition $\mathcal{P}$ of the leaves, there exists a partition $\mathcal{P}'$ such that*

(1) $\rho(\mathcal{P}') \leqslant \rho(\mathcal{P})$,
(2) *for every $v \in N \setminus \{r\}$, there exists $A' \in \mathcal{P}'$ such that $L_T(v) \subseteq A'$; i.e., $\rho(v, \mathcal{P}') = 1$.*

Proof. We iterate the following argument over all nodes $\hat{v} \in N$ such that $\rho(\hat{v}, \mathcal{P}) > 1$.

Define $\mathcal{P}' = \{A' \neq \emptyset \mid \text{there exists } A \in \mathcal{P} \text{ s.t. } A' = A \setminus L_T(\hat{v})\} \cup L_T(\hat{v})$. Then, $\rho(\hat{v}, \mathcal{P}') = 1 < \rho(\hat{v}, \mathcal{P})$ and for each $v \neq \hat{v}$ that is not the root, we have $\rho(v, \mathcal{P}') = \rho(v, \mathcal{P})$. Finally, for the root r, $\rho(r, \mathcal{P}') = |\mathcal{P}'| \leqslant |\mathcal{P}| + 1 = \rho(r, \mathcal{P}) + 1$. Hence $\rho(\mathcal{P}') = \rho(r, \mathcal{P}') + \sum_{v \in N\setminus\{r\}} \rho(v, \mathcal{P}') \leqslant \rho(\mathcal{P})$. $\square$

Theorem 4. *The Structured Bin Packing problem with unitary weights is* NP-*hard.*

Proof. Reduction is from Minimum Bin Packing. Suppose we are given an instance of Minimum Bin Packing with capacity c and items of size $s_1, \dots, s_n$. We define tree T as in Fig. 3.

Let $\mathcal{P}$ be a feasible solution to the Structured Bin Packing problem on T with capacity c. By Lemma 3, we can assume that $\rho(i, \mathcal{P}) = 1$ for every $1 \leqslant i \leqslant n$. The partition $\mathcal{Q}$ of $\{1, \dots, n\}$ given by

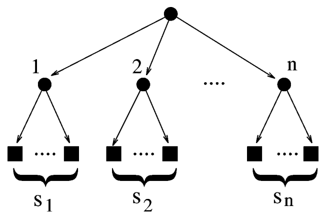

Fig. 3. Tree T constructed in the reduction from Min Bin Packing on input $(c; s_1, \dots, s_n)$.

the sets $\{i \mid L_T(i) \subseteq A \in \mathcal{P}\}$, for $1 \leqslant i \leqslant n$, has the same cardinality of $\mathcal{P}$ and gives immediately a feasible solution to the Minimum Bin Packing problem. Moreover, $\rho(\mathcal{P}) = |\mathcal{Q}| + n$. Whenever $\mathcal{P}$ is such that $\rho(\mathcal{P})$ is minimum, $|\mathcal{Q}|$ is also minimum. $\square$

Unfortunately, the reduction given in Theorem 4 is not cost preserving and so it does not lead to an inapproximability result.

4.1. *Algorithm* `Simple`

In this section we study the approximation factor of Algorithm `Simple` that packs the leaves traversing the tree frontier from left to right in sets of size c producing partition $\mathcal{P}_S$. We show that the cost of the solution given by `Simple` is not more than $3/2$ far away from optimum.

Lemma 4. *Given tree T rooted in r, $\rho(r, \mathcal{P}_O) \geqslant \rho(r, \mathcal{P}_S)$ and, for every vertex v in the tree, we have $\rho(v, \mathcal{P}_S) \leqslant \rho(v, \mathcal{P}_O) + 1$.*

Proof. Given partition $\mathcal{P}_S$, at the root of the tree we have $\rho(r, \mathcal{P}_S) = \lceil |L_T|/c \rceil$, by construction. For any partition $\mathcal{P}$ (and thus also for $\mathcal{P}_O$) we have $\rho(r, \mathcal{P}) \geqslant \lceil |L_T|/c \rceil$.

Given vertex $v \in T$ we have $\rho(v, \mathcal{P}_S) \leqslant \lceil |L_T(v)|/c \rceil + 1 \leqslant \rho(v, \mathcal{P}_O) + 1$. $\square$

Let $N_1 = \{v \in N \mid \rho(v, \mathcal{P}_O) = 1\}$ and $N_2 = N \setminus N_1 = \{v \in N \mid \rho(v, \mathcal{P}_O) \geqslant 2\}$. Observe that $\rho(N_2, \mathcal{P}_O) \geqslant 2|N_2|$ and that by Lemma 4 we have that $\rho(N_2, \mathcal{P}_S) \leqslant \rho(N_2, \mathcal{P}_O) + |N_2|$. Hence

$$\begin{aligned} \rho(N_2, \mathcal{P}_S)/\rho(N_2, \mathcal{P}_O) &\leqslant 1 + |N_2|/\rho(N_2, \mathcal{P}_O) \\ &\leqslant 3/2. \end{aligned}$$

Nodes in N_1 form a set $\mathcal{T}$ of at most $|N_1|$ distinct subtrees of T such that if the root of a subtree belongs to N_1 also all the internal nodes of the subtree do. Let $T' \in \mathcal{T}$ be a subtree with n internal nodes, height h, rooted in r'. Obviously, $\rho(T', \mathcal{P}_O) = n$ and the worst thing that algorithm `Simple` might do is to split the leaves of T' into two distinct elements of partition $\mathcal{P}_S$. Let z be the lowest common ancestor of two leaves of T' belonging to two different elements of $\mathcal{P}_S$ and consider Path(r', z), the unique path form node r' to z in T'. Then for each vertex $x \in$ Path(r', z) we have $\rho(x, \mathcal{P}_S) = 2$, while for any other vertex y not in the path we have $\rho(y, \mathcal{P}_S) = 1$. Hence, $\rho(T', \mathcal{P}_S) \leqslant 2 \cdot h + 1 \cdot (n - h) = n + h$. Whenever $n \geqslant 2h$, then $\rho(T', \mathcal{P}_S)/\rho(T', \mathcal{P}_O) \leqslant 3/2$.

So, let us finally concentrate on the set of disjoint subtrees $T_i \in \mathcal{T}$ (with n_i internal nodes, height h_i) such that $n_i < 2h_i$, for $i = 1, \ldots, m$ and some $m > 0$. As we assume that internal nodes have outdegree at least two, then it must be $n_i = 2h_i - 1$, for all i, and $\rho(T_i, \mathcal{P}_O) = 2h_i - 1$ while $\rho(T_i, \mathcal{P}_S) \leqslant 3h_i - 1$.

As the T_i are disjoint, if `Simple` splits the leaves of each T_i into two distinct elements of $\mathcal{P}_S$, then at the root r of T it must be $\rho(r, \mathcal{P}_S) \geqslant m$ and, thus, there must exist a positive integer K such that $\rho(r, \mathcal{P}_S) = K + m$. Remembering that $\rho(r, \mathcal{P}_O) \geqslant \rho(r, \mathcal{P}_S)$, we have

$$\begin{aligned}
\rho\Bigl(\{r\} \bigcup_i T_i, \mathcal{P}_O\Bigr) &= \rho(r, \mathcal{P}_O) + \sum_{i=1}^{m} \rho(T_i, \mathcal{P}_O) \\
&\geqslant m + K + \sum_{i=1}^{m} (2h_i - 1) \\
&= K + 2\sum_{i=1}^{m} h_i, \\
\rho\Bigl(\{r\} \bigcup_i T_i, \mathcal{P}_S\Bigr) &= \rho(r, \mathcal{P}_S) + \sum_{i=1}^{m} \rho(T_i, \mathcal{P}_S) \\
&\leqslant m + K + \sum_{i=1}^{m} (3h_i - 1) \\
&= K + 3\sum_{i=1}^{m} h_i,
\end{aligned}$$

and, as $K \geqslant 0$, we can conclude with the following theorem.

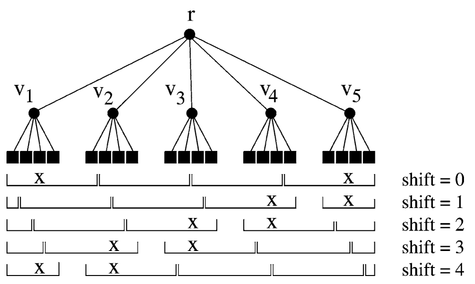

Fig. 4. In this example $c = 5$.

Theorem 5. *Let $\mathcal{P}_S$ and $\mathcal{P}_O$ be the partition found by* `Simple` *and by the optimal algorithm, respectively. Then $\rho(\mathcal{P}_S)/\rho(\mathcal{P}_O) \leqslant 3/2$.*

The analysis of Theorem 5 turns out to be accurate. Consider the class of trees $T(c) = (V, E)$ as in Fig. 4: $N = \{r, v_1, \ldots, v_c\}$ and $L = \{l_1, \ldots, l_{c(c-1)}\}$, where r is the root, the v_i's are the children of the root and each v_i has $c - 1$ leaves. It is easy to see that $\mathcal{P}_S$ has cardinality $c - 1$ and that only $L(v_1)$ and $L(v_c)$ are entirely contained in one element of $\mathcal{P}_S$. Hence we have, $\rho(r, \mathcal{P}_S) = c - 1$, $\rho(\bigcup_i v_i, \mathcal{P}_S) = 1 \cdot 2 + 2 \cdot (c - 2) = 2c - 2$ and $\rho(\mathcal{P}_S) = \rho(r, \mathcal{P}_S) + \rho(\bigcup_i v_i, \mathcal{P}_S) = 3c - 3$. Consider, now, partition $\mathcal{P}$ of cardinality c such that element $\mathcal{P}_i \in \mathcal{P}$ is $L_T(v_i)$, for $i = 1, \ldots, c$. We have that $\rho(r, \mathcal{P}) = c$, that $\rho(v_i, \mathcal{P}) = 1$ and therefore that $\rho(\mathcal{P}) = 2c$. The ratio $\rho(\mathcal{P}_S)/\rho(\mathcal{P}_O)$ tends to $3/2$ as c tends to infinity.

To conclude, observe that `Simple` does not work this well in the general case of Section 3: consider a tree with one single node r and $2c$ leaves whose weights are, alternatively, 1 and c. `Simple` finds a partition $\mathcal{P}_S$ such that $\rho(r, \mathcal{P}_S) = 2c$, while the optimal solution puts all the leaves with weight 1 together and finds a solution $\mathcal{P}_O$ such that $\rho(r, \mathcal{P}_O) = c + 1$.

Remark. One could object that algorithm `Simple` is too naive, in the sense that we could have possibly better solution if we did not require the leftmost element of the partition to have cardinality exactly c. Let `Smart-Simple` be an algorithm that computes this improved solution, say $\mathcal{P}_{SS}$. Observe that given $\mathcal{P}_S$ solution $\mathcal{P}_{SS}$ can be computed in O(cn) time.

Obviously the $3/2$ upper-bound is valid also for `Smart-Simple` but, unfortunately, also tight as there is a class of trees for which the approximation factor of `Smart-Simple` approaches $3/2$.

Consider again the example of Fig. 4. Any possible partition with shift greater than zero has c elements, while the partition with shift zero is the only one to have $c-1$ elements. For any partition $\mathcal{P}$ with any shift, there are exactly two distinct indices $i \neq j$ such that $L_T(v_i)$ and $L_T(v_j)$ are completely contained in one element of the partition (indicated with a cross in the example). Hence, $\rho(v_i, \mathcal{P}) = \rho(v_j, \mathcal{P}) = 1$ and $\rho(v_k, \mathcal{P}) = 2$ for any $k \neq i, j$. It is now clear that $\rho(\mathcal{P}_{SS}) = \rho(\mathcal{P}_S)$ and we already know that $\rho(\mathcal{P}_S)/\rho(\mathcal{P}_O)$ approaches $3/2$ when c goes to infinity.

An indicator for `Simple`. Even if we have shown a class of trees for which the performance of `Simple` approaches $3/2$ we observe that in many other cases the algorithm performs much better and might get very close to the optimal solution. Given tree $T = (N \cup L, E)$, by Lemmas 1 and 4 we have

$$\rho(\mathcal{P}_O) \geqslant \sum_{v \in N} \frac{|L(v)|}{c}$$

and

$$\rho(\mathcal{P}_S) \leqslant \rho(\mathcal{P}_O) + \sum_{v \in N} 1 = \rho(\mathcal{P}_O) + |N|,$$

respectively. Hence,

$$\frac{\rho(\mathcal{P}_S)}{\rho(\mathcal{P}_O)} \leqslant 1 + \frac{c|N|}{\sum_{v \in N} |L(v)|}. \qquad (1)$$

It is likely that for the majority of trees the ratio $R = c|N| / \sum_{v \in N} |L(v)|$ approaches zero. Nevertheless, this bound need not to be tight. Consider, for example, the tree in Fig. 4: R is $3/4$ (for the class of trees it is always greater than $1/2$) and from (1) we only get that, for this instance, the approximation factor is less than $7/4$.

5. Open problems

When the item sizes are relatively small there are simple approximation algorithms for the bin packing problem that achieve asymptotic performance ratio very close to one, e.g., first fit, space bounded next fit [2]. The hard instance for the algorithm given in Section 3, where the items are very small with respect to the bin size, seems to suggest that the requirement of locality, that this algorithm pursues by packing together subtrees to the maximum possible extent, is a serious obstacle to find packing schemes with ratios close to one. This is the natural question left open.

References

[1] S. Chakrabarti, B. Dom, R. Agrawal, P. Raghavan, Scalable feature selection, classification and signature generation for organizing large text databases into hierarchical topic taxonomies, VLDB J. 7 (1998) 163–178.

[2] E.G. Coffman Jr., M.R. Garey, D.S. Johnson, Approximation algorithms for bin packing: a survey, in: D. Hochbaum (Ed.), Approximation Algorithms for NP-hard Problems, PWS Publishing, Boston, 1996, pp. 46–93.

[3] P. Crescenzi, V. Kann, A compendium of NP optimization problems, http://www.nada.kth.se/theory/problemlist.html.

[4] I.S. Dhillon, J. Fan, Y. Guan, Efficient clustering of very large document collections, in: R. Grossman, G. Kamath, R. Naburu (Eds.), Data Mining for Scientific and Engineering Applications, Kluwer Academic Publishers, Dordrecht, 2001, pp. 357–382.

[5] G.W. Flake, K. Tsioutsiouliklis, R.E. Tarjan, Graph clustering techniques based on minimum cut trees, Technical Report 2002-06, NEC, Princeton, NJ, 2002.

[6] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman Company, San Francisco, 1979.

[7] N. Karmarkar, R.M. Karp, An efficient approximation scheme for the one-dimensional bin packing problem, in: Proc. 23rd Annual IEEE Symp. on Foundations of Computer Science, 1982, pp. 312–320.

[8] D. Simchi-Levi, New worst case results for the bin packing problem, Naval Res. Logistics 41 (1994) 579–585.

[9] C.L. Viles, J.C. French, Dissemination of collection wide information in a distributed information retrieval system, in: Proc. 18th Annual Internat. ACM Conf. on Research and Development in Information Retrieval (SIGIR), 1995, pp. 12–20.