Graph Packing over a Rooted Tree

by

Ze-Zeng ZHANG*, Shigeru MASUYAMA**,
Toshihide IBARAKI** and Hisashi MINE***

(Received December 26, 1986)

abstract

This paper investigates the computational complexity of the graph packing problem over a rooted tree (GPT) as a generalization of the one dimensional bin packing problem, where both the bins and the set of items to be packed are rooted trees. GPT is defined under two problem settings, edge GPT (EPT) and node GPT (NPT). In EPT, the items packed in a bin cannot share any edge but can share some node, while in NPT, the items can share neither node nor edge. We first prove that these problems are in general NP-complete, which strongly suggests that these problems are computationally intractable. However, for the case where the number k of different kinds of items is fixed, we derive a recursive formula of dynamic programming for the minimum number of bins required to pack all the items. This formula can be solved in polynomial time, if the bins and items are all uniform trees and/or comb-shaped trees in which each non-leaf node has the same number of sons. Furthermore, for GPT's with bins of uniform (d, H) trees and only one kind of item, of uniform (d, h) trees, we derive explicit formulas for the number of bins required.

1. Introduction

One dimensional bin packing problem is to minimize the number of bins of length B needed to pack all the given items of length p_i $(1 \leq i \leq n)$ into bins. This problem is known to be NP-complete[1),7)], and thus it is most unlikely that there exist some polynomial time algorithms for solving it. The main research stream has therefore been directed to obtain performance assurance of approximation algorithms (see e. g. 9) and a survey, 5). Recently, two dimensional bin packing problems are also studied, where both the items and the bins are

* Xibei Institute of Telecommunication Engineening, Xian, China
** Department of Applied Mathematics and Physics, Faculty of Engineering, Kyoto University, Kyoto, 606 Japan
*** Professor Emeritus, Kyoto University. Currently with the Department of Management Engineering, Kansai University, Suita, Osaka, 564 Japan

rectangles (see e. g., 2), 5)).

However, there are many practical problems like the VLSI design, which should be formulated as bin packing problems with some graph structures. As a first attempt to investigate bin packing problem with graph structures, we define, in this paper, the graph packing problems over a rooted tree (GPT), and study the computational comlexity of GPT's. GPT's are defined under two problem settings, edge GPT (EPT) and node GPT (NPT). In EPT, the items packed in a bin cannot share any edge but can share some node, while in NPT, the items can share neither edge nor node.

We first prove that these problems are NP-complete. This, however, does not prohibit the development of efficient algorithms for some special cases. In fact, we develop polynomial time algorithms for GPT's in which the number of different kinds of items is fixed. For this, we first derive a recursive formula of dynamic programming (see e. g., 4), 6) and 1)), which can be solved in polynomial time if the problem of deciding whether a given set of items can be packed into a bin is solved in polynomial time. We then develop polynomial time algorithms for such decision problems for the cases in which all the items and bins are uniform trees and/or comb-shaped trees such that each non-leaf node has the same number of sons.

Finally, for GPT's with only one kind of item, of uniform trees, in which each non-leaf node has the same number of sons as bins, we derive explicit formulas for the minimum number of bins required to pack all the given items.

2. Definitions

Terminologies for the subsequent discussion are briefly defined here (see e. g., 3), 8)).

A **rooted tree** $T=(V, E, r)$ is a directed graph, where V is the set of nodes, $E \subset V \times V$ is the set of edges and r is the **root**, satisfying the following three conditions;

1. There is no node $i \in V$ such that $(i, r) \in E$ (i. e., the root r has no parent where j is a **son** of i and i is a **parent** of j if $(i, j) \in E$).

2. For any $j \in V-\{r\}$, there exists just one $i \in V$ which satisfies $(i, j) \in E$ (i. e., any node except the root r has a unique parent).

3. For each $i \in V-\{r\}$, there exists a path from r to i (i. e., any node in T is reachable from the root through some path).

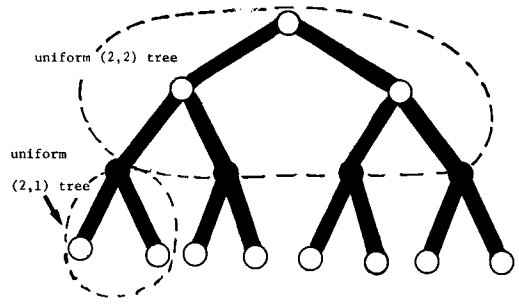
The number L of edges in a path P is the **length** of P. If there is a path from i to j, i is an **ancestor** of j and j is a **descendant** of i.

Any node of T with no son is a **leaf**. Note that the path from the root to any node is unique by the aforementioned condition 2. The **depth** of a node v in a rooted tree T is the length of the path from root r to v. The **height** of a rooted tree T is the largest depth among nodes in T. A **regular (d, h) tree** T is a rooted tree of height h in which each node except leaves has d sons. A **uniform (d, h) tree** T is a regular tree where every leaf v of T has depth h. Finally, a **comb-shaped (d, h) tree** T is a regular (d, h) tree in which each set S_i of nodes in depth i has at most one node that has some son.

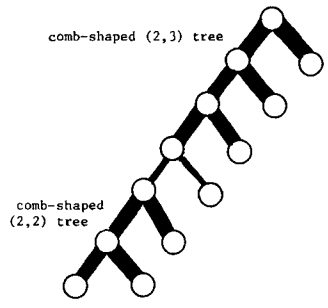
Given a rooted tree $T=(V, E, r)$ and a set $S=\{T_1, T_2, \ldots, T_n\}$ of rooted trees, S is an **edge packing** of T if, and only if, T contains subtrees $T'_i=(V'_i, E'_i, r'_i)$, $1 \leq i \leq n$, which are isomorphic to T_i and satisfy

$$E'_i \cap E'_j = \phi \quad for \ i \neq j. \tag{2.1}$$

Here T_i and T'_i are **isomorphic** if, and only if, they are identical under some renaming of nodes and edges. S is a **node packing** of T if it satisfies both



(a) A uniform (2,2) tree and four uniform (2,1) trees edge-packed into a uniform (2,3) tree (● denotes a node shared by two items).



(b) A comb-shaped (2,3) tree and a comb-shaped (2,2) tree is node-packed into a comb-shaped (2,6) tree.

Fig. 1 Definitions of GPT, uniform trees and comb-shaped trees.

(2. 1) and the following condition (2. 2):

$$V'_i \cap V'_j = \phi \text{ for } i \neq j. \qquad (2.\ 2)$$

Given identical rooted trees T (bins) and a set $S = \{T_1, \ldots, T_k\}$ of rooted trees (items), **the edge (node) packing problem over *T*, EPT (NPT)** is a problem of minimizing the number of bins needed to edge-pack (node-pack) all the items in S into bins. EPT and NPT are generally called **the graph packing problem over a rooted tree (GPT).**

Fig. 1 illustrates an edge packing and a node packing of uniform trees and comb-shaped trees, respectively.

3. Computational Complexity of GPT's

In this section, we prove the NP-completeness of GPT's by a polynomial time reduction from THREE PARTITION which is known to be NP-complete[1),7)].

Definition 3. 1. [1),7)] Given a set A of $3m$ elements, a positive number B and a size $s(a)$ for each $a \in A$ such that $B/4 < s(a) < B/2$, THREE PARTITION is to decide whether there exist m sets, each consisting of three elements, such that the sum of three elements in each set is B.

Theorem 3. 1. EPT is in general NP-complete.

(Proof) Obviously EPT$\in$NP holds. Thus we show that THREE PARTITION can be reduced to EPT in polynomial time. Let bins be comb-shaped (d, h) trees, and for each $a \in A$ we consider a comb-shaped $(d, s(a))$ tree. Then, THREE PARTITION has a solution if, and only if, all $3m$ items can be edge-packed into m bins. This shows that THREE PARTITION is polynomially reducible to EPT. Thus, we have proved the NP-completeness of EPT. (Q. E. D.)

Let the size of bins be $B+2$, then a similar discussion proves the NP-completeness of NPT although the details are omitted.

Theorem 3. 2. NPT is in general NP-complete. □

EPT and NPT are both NP-complete. Thus it is most unlikely that there exist efficient algorithms for solving these problems. However, when the number of different kinds of items is fixed, and the items are uniform trees of the same degree as that of the bins, we can develop efficient algorithms as we shall see in the next section.

4. Polynomial Time Algorithms for Some Special Cases of GPT's When the Number of Different Kinds of Items Is Fixed

Based on dynamic programming, we develop polynomial time algorithms for GPT when the number of different kinds of items are fixed, and both the items and the bins are uniform trees or comb-shaped trees.

If the problem of deciding whether a given set of items can be packed into a given bin is solvable in polynomial time, we can solve the bin packing problems, including GPT, in polynomial time as follows. Let $b\ (n_1, n_2, \ldots, n_k)$ be the minimum number of bins to pack n_j items j, $1 \leq j \leq k$. Then we have the following recursive formula of dynamic programming:

$$b\ (n_1,\ldots,n_k) = \begin{cases} 1: \text{ when all the items can be packed into a bin} \\ [\min\ b\ (n_1-x_1,\ n_2-x_2, \ldots,\ n_k-x_k)]\ +1: \text{ otherwise,} \end{cases} \quad (4.1)$$

where the minimum is taken over all x_j, $1 \leq j \leq k$, such that $0 \leq x_1 \leq n_1, \cdots, 0 \leq x_k \leq n_k$ and $(x_1, x_2, \ldots, x_k)$ can be packed into one bin.

(4.1) can be easily solved by computing b in the increasing order of n_i. The time complexity is $O\ (n^{2k}) \times$ (the time for deciding whether a set of given items can be packed into a bin).

Therefore, we concentrate on the problem of deciding whether all the given items can be packed into a given bin. In order to solve the above decision problem, we first consider the problem of finding the minimum height H_0 of a uniform tree (a bin) required to edge-pack (node-pack) all the given uniform trees (items), where all the non-leaf nodes of uniform trees, both the bins and the items, have the same number of sons. If this problem is solvable in polynomial time, then obviously we can check if the given set of uniform trees can be packed into the bin of a uniform tree of height H by comparing H with H_0.

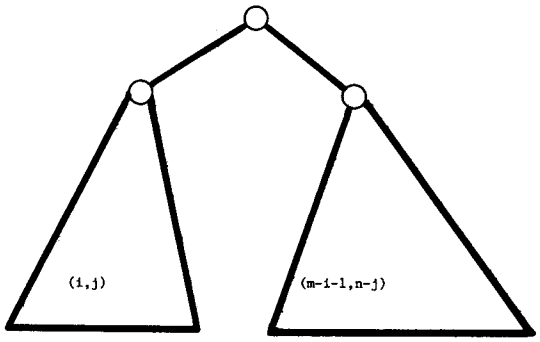
To clearly illustrate the idea, we consider the case where bins are uniform $(2, H)$ trees, and items are m uniform $(2, 1)$ trees and n uniform $(2, 2)$ trees.

Let $h\ (m, n)$ be the minimum height needed to edge-pack all the given items. Then, by dynamic programming,

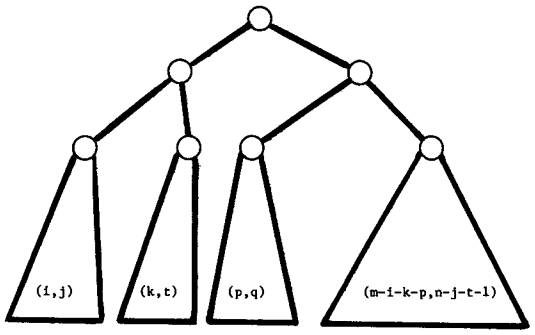
$$h(m, n) = \min \{ \min_{\substack{0 \leq i \leq m-1 \\ 0 \leq j \leq n}} \max \{h(i, j), h(m-i-1, n-j)\} + 1,\ \min_{\substack{0 \leq i \leq m \\ 0 \leq j \leq n-1 \\ 0 \leq k \leq m-i \\ 0 \leq t \leq n-j-1 \\ 0 \leq p \leq m-i-k \\ 0 \leq q \leq n-j-t-1}}$$

$$\max \{h\ (i,\ j),\ h\ (k,\ t),\ h\ (p,\ q),\ h\ (m-i-k-p,\ n-j-t-q)\} + 2\} \tag{4.2}$$

where the former (the latter) part of the right hand side of (4.2) is the minimum height, if the items of uniform (2, 1) tree (uniform (2.2) tree, respectively) are packed at the top of the bin as illustrated in Fig. 2. Note that if no item is edge-packed at the root, then we can pack a uniform (2.1) tree covering the root, and we may reduce the number of uniform (2.1) trees by one. Thus, such a case is excluded from consideration.



(a) A uniform (2,1) tree is packed into the top of the bin.



(b) A uniform (2,2) tree is packed into the top of the bin.

Fig. 2 Illustration of formula (4.2).

(4.2) can be easily computed by the standard solution method for dynamic programming formulas, i. e., by increasing $m+n$ by one at each stage.

The general case where the fixed number of different kinds of uniform trees are edge-packed (node-packed) into uniform trees can be similarily solved, although the details are omitted to avoid complication. We can also obtain a

similar formula as (4.2) of dynamic programming for GPT when the items are all comb-shaped trees and the number of different kinds of items is fixed, although the details are omitted.

5. Explicit Formulas for the Minimum Number of Bins for Some Special Cases

In this section, we derive explicit formulas for the minimum number of bins needed to pack when one kind of item, namely, uniform or comb-shaped trees are packed. We first consider the case where the bins are uniform (d, H) trees and the items are (d, h) trees.

Lemma 5.1. Let $f(H, h, d)$ be the maximum number of uniform (d, h) trees edge-packed into a uniform (d, H) tree. Then $f(H, h, d)$ satisfies the following recursive formula:

$$f(H, h, d) = d^{H-h} + f(H-h, h, d) \tag{5.1}$$

(Proof) We first show

$$f(H, h, d) \geq d^{H-h} + f(H-h, h, d) \tag{5.2}$$

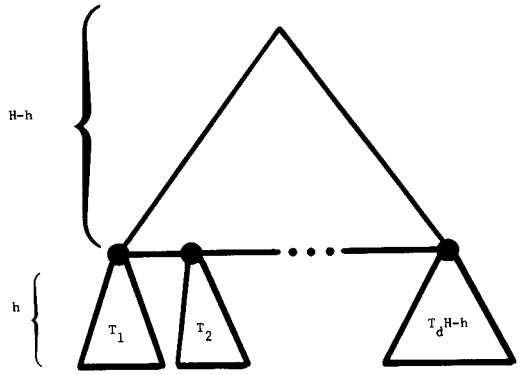
where $f(H, h, d) = 0$ for $H \leq h-1$. $f(H, h, d)$ is obviously 1 when $H=h$. When $H \geq h+1$, notice that the number of leaves of a uniform $(d, H-h)$ tree is d^{H-h}, *and* (5.2) holds (see Fig. 3 (a)).

We now show that

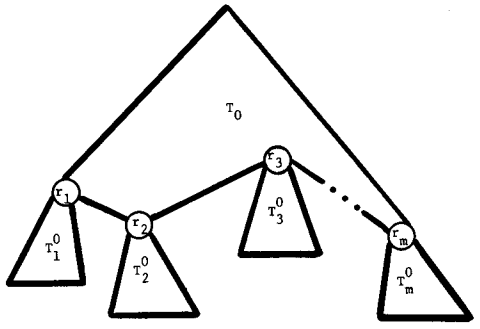
$$f(H, h, d) \leq d^{H-h} + f(H-h, h, d) \quad (H \geq h) \tag{5.3}$$

holds.

Consider a maximum edge-packing S of uniform (d, h) trees into a uniform (d, H) tree, and let $T^o_1, \ldots, T^o_m$ be the set of items in S where none of their leaves have nodes of other items as descendants. Consider a subtree T_o of T obtained by deleting all the edges of T^o_i, $i = 1, \ldots, m$, and let N be the number of uniform (d, h) trees packed in T_o. Let T' be a uniform $(d, H-h)$ tree, whose root coincides with that of T. By noting that all the distances of the roots of T^o_i, $i = 1, \ldots, m$, are not greater than $H-h$, we have $T_o \subset T'$. Thus



(a) Illustration of formula (5.1) in the proof of Lemma 5.1.



(b) Illustration of T_0 and T_i^0, i+1, ..., m, in the proof of Lemma 5.1.

Fig. 3 Proof of Lemma 5. 1.

$$N \leq f\ (H-h,\ h,\ d). \tag{5. 4}$$

Obviously,

$$m \leq 2^{H-h} \tag{5. 5}$$

also holds. And we have

$$f\ (H,\ h,\ d)\ = N+m \leq f\ (H-h,\ h,\ d)+\ 2^{H-h}. \tag{5. 6}$$

This proves (5. 3), and we have (5. 1). (Q. E. D.)

(5. 1) can be explicitly solved.

Theorem 5. 2.

$$f(H, h, d) = d^r (d^{rq} - 1) / (d^r - 1), \tag{5.7}$$

where $H = qh + r$, $0 \leq r \leq h - 1$. □

The number B of bins of uniform (d, H) trees needed to edge pack n items of a uniform (d, h) tree is

$$B = \lceil n/f(H, h, d) \rceil = \lceil n (d^r - 1) / d^r (d^{rq} - 1) \rceil, \tag{5.8}$$

where $\lceil x \rceil$ denotes the smallest integer greater than x.

Similarly, the maximum number $g(H, h, d)$ of items of uniform (d, h) trees and to be node-packed into a uniform (d, H) tree is given by

$$g(H, h, d) = d^{r'} (d^{r'q'} - 1) / (d^{r'} - 1), \tag{5.9}$$

where $H = q'(h+1) + r'$, $0 \leq r' \leq h$ and the number B of uniform (d, H) trees needed to edge pack n uniform (d, h) trees is

$$B = \lceil n (d^r - 1) / d^r (d^{r'q'} - 1) \rceil. \tag{5.10}$$

6. Concluding Remarks

The following topics seem to deserve the further research.

(1) Development and analysis of good approximation algorithms for the NP-complete GPT problems.

(2) To find other special cases solvable in polynomial time.

Along the line (2) above, we have already obtained some results. When identical chains are edge-packed (node-packed) into an undirected tree, there is an $O(n \log n)$ ($O(n)$) time algorithm[11] (n is the number of nodes of the tree). However the edge packing problem of identical chains of length greater than two into a general graph is NP-complete[11]. When identical rooted (or undirected) trees are packed into a rooted (or an undirected) tree we also developed polynomial time algorithms[12]. These recent results will be reported elsewhere[11, 12].

Acknowledgement

We would like to express our gratitude to Professor Toshiharu Hasegawa of Kyoto University, and all the members of Prof. Mine's and Prof. Hasegawa's laboratories, Department of Applied Mathematics and Physics, Faculty of Engineering, Kyoto University. This work was supported in part by Scientific Grant -in-Aid from the Ministry of Education, Science and Culture of Japan.

References

1) Aho, A. V., Hopcroft, J. E. and Ullman, J. D., *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, Mass., 1974.
2) Baker, B. S. Coffman, E. G., Jr. and Rivest, R. L., "Orthogonal packings in two dimensions", *SIAM J. Comput.* Vol. 9, pp. 846-855, 1980.
3) Berge, C., *Graphes et Hypergraphes*, Donod, Paris, 1970.
4) Bellman, R., *Dynamic Programming*, Princeton University Press, Princeton, 1957.
5) Coffman Jr., E. G., Garey, M. R. and Johnson, D. S., "Approximation algorithms for bin packing - an updated survey", *Working paper*, Bell Laboratories, 1984.
6) Denardo, E. V., *Dynamic Programming Models and Applications*, Prentice-Hall, Englewood Cliffs, New Jersey, 1982.
7) Garey, M. R. and Johnson, D. S., *Computers and Intractability: A Guide to the Theory of NP-completeness*, W. H., Freeman and Company, San Fransisco, 1979.
8) Harary, F., *Graph Theory*, Addison-Wesley, Reading, Mass., 1969.
9) Johnson, D. S., Demers, A., Ullman J. D., Garey, M. R. and Graham, R. L., "Worst-case performance bounds for simple one-dimensional packing algorithms", *SIAM J. Comput.*, Vol. 3, No. 4, pp. 299-325, 1974.
10) Kirkpatrick, D. G. and Hell, P., "On the complexity of general graph factor problems", *SIAM J. Comput.*, Vol. 12, No. 3, pp. 601-609, 1983.
11) Masuyama, S. and Ibaraki, T., "Computational complexity of chain packing problems", *Technical Report* #87001, Department of Applied Mathematics and Physics, Faculty of Engineering, Kyoto University, Fed. 1987.
12) Masuyama, S., "On the tree packing problem", forthcoming in the *Ko-Kyu-Roku (Lecture Note Series) of the Research Institute of Mathematical Science*, Kyoto University (in Japanese), 1987; Technical Report # 87014, Department of Applied Mathematics and Physics, Faculty of Engineering, Kyoto University, May, 1987.