대한산업공학회/한국경영과학회 2000 춘계공동학술대회 논문집. 2000년4월21-22일. 경남대학교
Session OB3.3

크기가 다양한 bin-packing 문제에 대한 algorithm

강장하, 박성수
한국과학기술원 산업공학과

Abstract

In this paper, we consider variable sized bin packing problem, where the objective is not to minimize the total space used in the packing but to minimize the total cost of the packing when the cost of unit size of each bin does not increase as the bin size increases. A heuristic algorithm is described, and analyzed in two special cases: 1) $b_m|\cdots|b_1$ and $w_n|\cdots|w_1$, and 2) $b_m|\cdots|b_1$, where b_i denotes the size of i-th type of bin and w_j denotes the size of j-th item. In the case 1), the algorithm guarantees optimality, and in the case 2), it guarantees asymptotic worst-case performance bounds of 11/9.

Consider a finite collection of bin sizes and an inexhaustible supply of bins of each size. Let L be a list of n items that are sorted by unincreasing order of their sizes, and let $w_j \in Z_+$ denote the size of j-th item. Let T be a list of m types of bins that are sorted by unincreasing order of their sizes, and let $c_i \in R_+$ and $b_i \in Z_+$ denote the cost and the size of i-th type of bin, respectively.

The problem we investigate is that of packing a list of items into bins so as not to minimize the total space used in the packing, but to minimize the total cost of the packing used when the cost of unit size of each bin does not increase as the bin size increases, that is,

$$\frac{c_{i_1}}{b_{i_1}} \le \frac{c_{i_2}}{b_{i_2}}, \quad \forall 1 \le i_1 \le i_2 \le m.$$

When the general L and T are considered, this problem is NP-hard. But if L and T have some divisibility constraints, this problem can be solved more easily. In this paper, an heuristic algorithm is described, and analyzed in two special cases: 1) $b_m|\cdots|b_1$ and $w_n|\cdots|w_1$, and 2) $b_m|\cdots|b_1$. In the case 1), the algorithm guarantees optimality, and in the case 2), it guarantees asymptotic worst-case performance bounds of 11/9.

First, starting with some notations, we introduce a new algorithm named by *"Marked First-Fit Decreasing Algorithm"*, **MFFD** in short. Here, "First-Fit Decreasing Algorithm", **FFD** in short, is a heuristic algorithm that considers the items that are sorted by unincreasing order of their sizes according to increasing indices. Also, the algorithm assigns each item to the lowest indexed initialized bin into which it fits; only when the current item cannot fit into any initialized bin, a new bin is introduced.

We let $B=\langle B_1, B_2, \cdots, B_k \rangle$ denote the ordered set of bins used by an algorithm, and we let $C(B)$ and $C(B_i)$, for all $1 \le i \le k$ denote the total cost of bins in B and the cost of $C(B_i)$, respectively.

Algorithm : Marked First-Fit Decreasing(MFFD)

(1) allocate all the items L into the first type of bin by the first-fit decreasing manner. Let $B^1=\langle B_1^1, B_2^1, \cdots, B_{k_1}^1 \rangle$ denote the ordered set of bins used in this step, and C^1 denote $C(B^1)$.

(2) allocate all the items allocated in $B_{k_1}^1$ to the second type of bin by the first-fit decreasing manner. Let $B^2=\langle B_1^2, B_2^2, \cdots, B_{k_2}^2 \rangle$ denote the ordered set of bins used in this step, and C^2 denote $C(B^1-\{B_{k_1}^1\})+C(B^2)$.

(3) recursively apply (2) to the remaining types of bins, and define $B^3, B^4, \cdots, B^m$ and $C^3, C^4, \cdots, C^m$, where $C^i=\sum_{j=1}^{i-1} C(B^j-\{B_{k_j}^j\})+C(B^i)$, for all $3 \le i \le m$.

(4) let $j^* = \arg\min_{1 \le j \le m} C^j$.

(5) for each used bin, if the total size of the contents is small enough to move into a smaller new bin, repack them into it, and then let B^* denote the result set of bins.

Here, B^* is the result of this algorithm and $C(B^*)$ is the objective value of it.