ISSN 1009 – 3095 *Journal of Zhejiang University*(*SCIENCE*) V.2, No.3, P.241 – 246, July – Sep., 2001
http://www.chinainfo.gov.cn/periodical; http://www.zju.edu.cn;
http://lib.zju.edu.cn/English; http://www.zjupress.com

EXACT ALGORITHM FOR BIN COVERING*

CHEN Feng(陈　峰)†, YAO En-yu(姚恩瑜)

(*Department of Applied Mathematics, Zhejiang University, Hangzhou 310027, China*)
†E-mail: chenfeng@math.zju.edu.cn

Received Oct. 24, 2000; revision accepted Feb.20, 2001

Abstract: This paper presents a new arc flow model for the one-dimensional bin covering problem and an algorithm to solve the problem exactly through a branch-and-bound procedure and the technique of column generation. The subproblems occuring in the procedure of branch-and-bound have the same structure and therefore can be solved by the same algorithm. In order to solve effectively the subproblems which are generally large scale, a column generation algorithm is employed. Many rules found in this paper can improve the performance of the methods.

Key words: bin covering, column generation, branch-and-bound algorithm
Document code: A　　**CLC number:** O221.7

INTRODUCTION

The bin covering problem can be stated as follows. Given n items of weights $l_1 \cdots, l_n$ and an unlimited number of identical bins of capacity C, pack the items in as many bins as possible so that the total weight of each bin is no less than its capacity. The bin covering problem was first studied by Assmann et al. (1984) as a dual problem of bin packing and was shown to belong to the class of NP-hard problems. It is known that the existence of a polynomial-time algorithm to solve the NP-hard problem optimally is unlikely. As a result, much more research has been conducted on the heuristics that solve NP-hard problem to near optimality. For detail discussion of the bin-covering problem in the field, see the papers of Assmann (1984); Su et al. (1999); Csirik et al. (1988). In contrast, to the author's knowledge, except for Labbé et al. (1995), little research has been done to solve the bin covering problem exactly.

We can assume, without loss of generality, that

$$l_1 \geqslant l_2 \geqslant \cdots \geqslant l_n .$$

and

$$l_j < C, j = 1, \cdots, n .$$

since any item with $l_j \geqslant C$ can be assigned alone to a bin, and

$$\sum_{j=1}^{n} l_j \geqslant 2C .$$

because otherwise a solution of value 0 or 1 can trivially be determined. Note that some of the items may not belong to the solution as it is not required to pack in a bin more items than necessary to fill it.

In general, the bin covering problem can be modeled as follows:

$$\mathrm{BC}\begin{cases} \max \quad z = \sum_{i=1}^{n} y_i . \\ s.t. \ \sum_{j=1}^{n} l_j x_{ij} \geqslant c y_i , \quad \forall\, i = 1, \cdots, n . \\ \quad \sum_{i=1}^{n} x_{ij} \leqslant 1, \quad \forall\, j = 1, \cdots, n . \\ \quad y_i = 0 \text{ or } 1 , \quad \forall\, i = 1, \cdots, n . \\ \quad x_{ij} = 0 \text{ or } 1 , \quad \forall\, i, j = 1, \cdots, n . \end{cases}$$

where,

$$y_i = \begin{cases} 1, & \text{if bin } i \text{ is used.} \\ 0, & \text{otherwise.} \end{cases}$$

$$x_{ij} = \begin{cases} 1, & \text{if item } j \text{ is assigned to bin } i . \\ 0, & \text{otherwise.} \end{cases}$$

Let

$$t = \min\left\{ s : \sum_{k=s}^{n} l_k < C \right\} .$$

$$P(j) = \min\left\{ p : \sum_{k=j}^{j+p} l_k \geqslant C \right\}$$

$$\text{for } j = 1, \cdots, \tau = \min(\lfloor \tfrac{1}{2} n \rfloor, t - 1) .$$

* Project(19971078) supported by Natural Science Foundation of China(NSFC).

$\alpha = (0), \alpha(k) = \sum_{j=1}^{k} p(j)$ for $k = 1, \cdots, \tau$.

$U_2(K) = k + \left\lceil \sum_{j=k+1}^{n} l_j - \alpha(k) / C \right\rceil$.

$U_2 = \min_{0 \leqslant k \leqslant \tau} \{U_2(K)\}$.

$\beta(j) = \min\left\{\beta: l_j + \sum_{k=1, k \neq j}^{\beta} l_k \geqslant C\right\}$.

$q(j) = \begin{cases} \beta(j) & \text{if } \beta(j) \geqslant j. \\ \beta(j) + 1 & \text{otherwise.} \end{cases}$

$U_3 = \left\lceil \sum_{j=1}^{n} \frac{1}{q(j)} \right\rceil$.

$U = \min(U_2, U_3)$.

We have the following proposition which was proved by Labbé, et al. (1995).

Proposition 1 U is a valid upper bound on z^* and $U \leqslant 2z^*$.

Let $W = C + \max_i \{l_i\} - 1$.

Proposition 2 The lower bound of BC is

$$L = \left\lceil \frac{\sum_{i=1}^{n} l_i}{W} \right\rceil.$$

A classical and widely used method for solving hard combinatorial problems is based on formulating the problem as a set-partitioning problem. This is done commonly for crew scheduling problems (Hoffman et al., 1993), vehicle routing problems (Desrochers, et al., 1992), bin-packing problems (Vllerio de Carvalho, 1999; Chan, et al., 1998) and many others.

For bin covering problem, let F be the set of feasible columns, that is,

$$F = \{a = (a_1, \cdots, a_n)^{\mathrm{T}} \in R^n : \sum_{i=1}^{n} l_i a_i \geqslant C, a_i \in \{0, 1\}\}. \quad (1)$$

Let $|F|$ be the cardinality of F. For simplicity, we denote

$$F = \{a^i : i = 1, \cdots, |F|\}.$$

It is easy to see that the bin covering problem can be also formulated as

$$\text{BC} \begin{cases} \max \quad \sum_{i=1}^{|F|} y_i \\ s.t. \quad \sum_{i=1}^{|F|} \boldsymbol{a}^i y_i \leqslant e \\ \quad \sum_{i=1}^{|F|} y_i \leqslant U, \\ \quad \sum_{i=1}^{|F|} y_i \geqslant L, \\ \quad y_i \in \{0, 1\}, i = 1, \cdots, |F| \end{cases}$$

where $e = (1, \cdots, 1)^{\mathrm{T}} \in R^n$.

Francois (Vanderbeck et al., 1996) developed an exact column generation algorithm for the above integer programming with a large number of columns.

In general, the implementation of procedures that combine branch-and-bound and column generatio has to overcome a crucial difficulty: the problem loses its "structure" as the branching constraints are added to the restricted master problem, and the type of subproblem that has to be solved at each iteration may change. Here we explore a flow model for the bin-covering and present an exact algorithm which can overcome the above difficulty partly. It was largely motivated by Valério (1999).

The contents of the paper are as follows. In section 2, we introduce the flow formulation of BC and present some criteria to reduce the number of variables. In section 3, we describe the branch and bound procedure for the flow formulation. In section 4, a column generation procedure is given to solve the subproblems at each node of branch-and-bound and the algorithm BCFE is also presented.

MATHEMATICAL MODEL

Because there may be more than one item with the same size, let P be the set of items with different size in L. Without loss of generality, suppose $P = \{l_1, l_2, \cdots, l_m\}$ and $l_1 \geqslant l_2 \geqslant \cdots \geqslant l_m$. The number of each size is denoted as $b_1, b_2, \cdots, b_m$ respectively.

We construct a graph $G = (V, A)$ with $V = \{0, 1, 2, \cdots W\}$ and $A = \{(i, j) | 0 \leqslant i \leqslant j \leqslant W$ and $j - i = l_k$, for every $k \leqslant m\}$. The number of arcs is $O(mW)$.

Consider additional arcs between $(i, i+1)$, $i = C, C+1, \cdots, W-1$ corresponding to unoccupied portions out of the bin. It is easy to prove that if there is a feasible column iff, there is a path between vertices 0 and W. The length of arcs that constitute the path define the size of an item in a cover.

If a solution covering a single bin corresponds to the flow of one unit between vertices 0 and W, a path carrying a larger flow will correspond to using the same cover solution in multiple bins.

By the flow decomposition properties, any non-negative flow of the above graph can be represented by paths connecting the only excess node 0 to the only deficit node W. According to this, the bin covering problem can be formulated as the problem of determining the maximum flow between vertex 0 and vertex W with additional constraints enforcing that every item will be used at most once. Considering decision binary variables x_{ij} associated with arcs defined above, which correspond to the number of the size j-i placed in any bin at a distance of i units from the beginning of the bin. The variable z can be denoted as x_{W0}. The model is as follows:

$$\text{BCF}\begin{cases}\max & z.\\ s.t. & \sum_{(i,j)\in A} x_{ij} - \sum_{(j,k)\in A} x_{jk} = \begin{cases} -z, & \text{if } j=0.\\ 0, & \text{if } j=1,2,\cdots,W-1.\\ z, & \text{if } j=W.\end{cases}\\ & \sum_{(k,k+p_d)\in A} x_{k,k+p_d} \leqslant b_d \quad d=1,2,\cdots,m.\\ & x_{ij}\geqslant 0 \text{ and integer.} \quad \forall (i,j)\in A.\end{cases}$$

Using the variables presented so far, there are many alternative solutions with exactly the same items in each bin. We try to reduce both the symmetry of the solution space and the size of the model by considering only a subset of arcs from A. The following three criteria are presented in order to meet this need.

If we search a solution in which the items are ordered in decreasing values of length, the following criteria may be used to reduce the number of arcs that are taken into account.

Criterion 1 An arc of size l_e, designed by $x_{k,k+l_e}$, can only have its tail at a node k that is the head of another arc of size l_d, $x_{k-l_d,k}$ for $l_d \geqslant l_e$, or, else, from node 0, i.e., the left border of the bin.

Example 1 Given an instance with bins of capacity $C=7$ and $P=\{6,5,3,2,1\}$ $B=(b_1,b_2,\cdots,b_m)=(1,1,3,1)$.

Appling the above criterion to A, we have

$\tilde{A}=\{(0,1),(0,2),(0,3),(0,5),(0,6),(1,2),(2,3),(2,4),(3,4),(3,5),(3,6),(4,5),(4,6),(5,6),(5,7),(5,8),(5,10),(6,7),(6,8),(6,9),(6,11),(6,12),(7,8),(7,9),(8,9),(8,10),(8,11),(9,10),(9,11),(9,12),(10,11),(10,12),(11,12)\}$.

An item is not needed to be put into a bin which is already filled. In the flow model, a criterion can be given as follows.

Criterion 2 The tail of any arc is less than C. According to the criterion, $\tilde{A}$ is modified by

$\tilde{A}=\{(0,1),(0,2),(0,3),(0,5),(0,6),(1,2),(2,3),(2,4),(3,4),(3,5),(3,6),(4,5),(4,6),(5,6),(5,7),(5,8),(5,10),(6,7),(6,8),(6,9),(6,11),(6,12)\}$.

In general, we want to limit the number of items which have the same size in a covering. i.e., no more than $\min\{b_d, \lfloor \frac{W}{l_d} \rfloor\}$. Although it is difficult to satisfy the desire, the following criterion is useful to meet the demand approximately.

Criterion 3 Given node k that is the head of another arc of size l_d ($l_d > l_e$) or $k=0$, the only valid arcs for size l_e are those that start at nodes $k+il_e$, $i=0,1,\cdots,b_e-1$ and $k+il_e \leqslant W$.

By this criterion, $\tilde{A}$ is changed as

$\tilde{A}=\{(0,1),(0,2),(0,3),(0,5),(0,6),(2,3),(2,4),(3,4),(3,5),(4,5),(4,6),(5,6),(5,7),(5,8),(6,7),(6,8),(6,9),(6,11)\}$.

On the condition of the above criterion, an arc is invalid if there are not enough arcs which can be used to construct a flow with the arc. Let

$$T_i=\max\{0, C-\sum_{j=i+1}^{m} b_j l_j\}, \quad i=1,\cdots m-1.$$

Criterion 4 The head of any arc which has length l_i is no less than T_i.

After applying the above criterion to $\tilde{A}$, $\tilde{A}$ is as follows,

$\tilde{A}=\{(0,2),(0,3),(0,5),(0,6),(2,4),(3,5),(4,6),(5,7),(5,8),(6,7),(6,8),(6,9),(6,11)\}$.

Let us denote $\bar{A}\subset A$ as the set of arcs that remain after applying the above criteria.

BRANCH AND BOUND PROCESS

In order to obtain an exact solution of BCF, a branch and bound algorithm is employed. At each node, a depth-first search is performed, using branching constraints of the following type:

$$x_{ij} \leqslant \lfloor x_{ij} \rfloor . \tag{2}$$

$$x_{ij} \geqslant \lceil x_{ij} \rceil \tag{3}$$

Where, x_{ij}^{l} is a single fractional variable of the solution at the current node. Thus, we solve the following programming which is characterized by cardinality constraints arising from branching:

$$\mathrm{BCF}^{u}\begin{cases}\max \quad z.\\ s.t. \quad +\sum_{(i,j)\in\bar{A}} x_{ij} - \sum_{(j,k)\in\bar{A}} x_{jk} = \begin{cases} -z, & \text{if } j=0.\\ 0, & \text{if } j=1,2,\cdots,W-1.\\ z, & \text{if } j=W.\end{cases}\\ \quad \sum_{(k,k+p_d)\in\bar{A}} x_{k,k+p_d} \leqslant b_d \qquad d=1,2,\cdots,m.\\ \quad x_{ij} \leqslant \lfloor x_{ij}^{l} \rfloor \qquad \forall\, l\in G^{u},\\ \quad x_{ij} \geqslant \lceil x_{ij}^{l} \rceil \qquad \forall\, l\in H^{u},\\ \quad x_{ij} \geqslant 0 \qquad \forall\,(i,j)\in\bar{A}.\end{cases}$$

Where, G^{u} and H^{u} are the index sets of branching constaints at node u of type (2) (3) repectively.

Remark 1 At the root node, we have

$$G^{u} = H^{u} = \phi .$$

After a branching constraint is added, the branching problem is reoptimized, and one of the following cases occurs:

1. The solution is an integer, with a value equal to U, which means that it is optimal;

2. The solution is fractional, with a value large than U, making it necessary to introduce new branching constraints;

3. The solution has a value that is strictly less than U. The node is fathomed.

Remark 2 In fact, if we obtain a feasible solution and the objective value is larger or equal to U, we can perform the depth-first search without finding the optimal solution of the current node.

The first upper bound of objective value is U, and if an integer solution is not found in the search tree, the upper bound has to be decreased by one unit, i. e., $U = U - 1$, and the procedure has to be repeated. Clearly, the first solution is optimal.

COLUMN GENERATION PROCESS

A crucial problem in the above branching and bound procedure is how to solve BCF^{u} effectively at node u of the search tree (branching tree). As discussed by Vanderbeck et al. (1996), a column generation can be used, i. e., we consider a sequence of programming with a subset of variables by increasing the subset of variables, which can be formulated as follows:

$$\mathrm{SBCF}^{u}\begin{cases}\max \quad z.\\ s.t. \quad +\sum_{(i,j)\in\bar{A}} x_{ij} - \sum_{(j,k)\in\bar{A}} x_{jk} = \begin{cases} -z, & \text{if } j=0.\\ 0, & \text{if } j\in\tilde{J}.\\ z, & \text{if } j=W.\end{cases}\\ \quad \sum_{(k,k+p_d)\in\bar{A}} x_{k,k+p_d} \leqslant b_d \qquad d=1,2,\cdots,m.\\ \quad x_{ij} \leqslant \lfloor x_{ij}^{l} \rfloor \qquad \forall\, l\in G^{u}\cap\tilde{A},\\ \quad x_{ij} \geqslant \lceil x_{ij}^{l} \rceil \qquad \forall\, l\in H^{u}\cap\tilde{A},\\ \quad x_{ij} \geqslant 0 \qquad \forall\,(i,j)\in\bar{A}.\end{cases}$$

Where $\tilde{A}$ is the current set of variables which have been considered. $\tilde{J}$ is the cardinate set of the vertex that are considered in the subproblem.

Any $x_{ij} \in \tilde{A}$ corresponds to a column $A^{ij} \in R^{W}$ with $A_i^{ij} = -1$; $A_j^{ij} = 1$ and 0 for other compents.

Following the discussion above, after adding the branching constraints and reoptimizing the programming (SBCF^u), suppose that the optimal solution is strictly less than U. Let $\pi = (u, v)$ be the vector of dual variables associated with the flow conservation and the constraints of numberof items and μ, ν the vectors of dual variables associated with the branching constraints of type (2) and (3), respectively.

From dual theory, we know that the increased cost of variable x_{ij} at node u is $\tilde{C}_{ij} = 0 - (u_i + u_j + v_d) - \sum_{l \in G^u_{(i,j)}} \mu_l + \sum_{l \in H^u_{(i,j)}} \nu_l$.

Any solution can be expressed as a nonnegative linear combination of paths, therefore, instead of generating single arcs, sets of arcs that can be associated with a single path can be generated. Column A^{ij} is attractive if its increased cost $\tilde{c}_{ij}$ is larger than zero. To determine the most attractive path, a subproblem is solved that corresponds to the longest path in an acyclic digraph with arc costs that depend on the value of the dual variables. The programming that is used to price out attractive paths, has the integrality property, and takes the following form:

$$\text{ABCF}^u \begin{cases} \max & + \sum_{(i,j)\in A} \bar{C}_{ij} x_{ij}. \\ s.t. & + \sum_{(i,j)\in A} x_{ij} - \sum_{(j,k)\in A} x_{jk} = \begin{cases} -1, & \text{if } j = 0. \\ 0, & \text{if } j = 1, 2, \cdots, W-1 \\ 1, & \text{if } j = W. \end{cases} \\ & x_{ij} \geqslant 0 \text{ and integer.} \qquad \forall (i,j) \in A. \end{cases}$$

Since all the arcs in SBCF^u have nonpositive reduced costs, we have the following necessary conditions.

Proposition 3 There is an attractive path only if there is an arc with a positive increased cost out of the restricted master problem.

Proposition 4 The reduce cost of a path p is equal to

$$\bar{c}_p = 1 - \sum_{(i,j)\in p} v_d - \sum_{l \in G^u_{(i,j)}} \mu_l + \sum_{l \in H^u_{(i,j)}} \nu_l \qquad (4)$$

Proof The reduced cost of a path p is equal to the sum of its arcs $\bar{c}_p = \sum_{(i,j)\in p} \bar{c}_{ij} = -\sum_{(i,j)\in p} (-u_i + u_j + v_d) - \sum_{l \in G^u_{(i,j)}} \mu_l + \sum_{l \in G^u_{(i,j)}} \nu_l$. The flow conservation dual variables cancel out for all nodes, except for the two terminal nodes. Therefore, $\bar{c}_p = -(-u_0 + u_w) - \sum_{(i,j)\in p} v_d - \sum_{l \in G^u_{(i,j)}} \mu_l + \sum_{l \in H^u_{(i,j)}} \nu_l$. The $(-u_0 + u_w)$ corresponds to the reduced cost with a value of -1.

Corollary 1 The set of arcs that correspond to path p is attractive if

$$\sum_{(i,j)\in p} v_d + \sum_{l \in G^u_{(i,j)}} \mu_l - \sum_{l \in H^u_{(i,j)}} \nu_l < 1. \qquad (5)$$

Remark 3 As a rule, the branching problem SBCF^u and the subproblem ABCF^u must be compatible. At a given node u of the branch-and-bound tree, there is a set of branching constraints that are enforced in the restricted master problem. To ensure compatibility, the variables introduced in the branching problem SBCF^u using column generation should not make feasible a point of the solution space of the master problem that violates the branching constraints imposed before-hand. Therefore, the path column to be introduced in the restricted master problem should not enter freely as a nonnegative variable.

Criterion 5 A variable (arc) that was set to zero in SBCF^u might be regenerated by the sub-problem ABCF^u as part of an attractive path. To prevent regeneration, the arc can be removed from the subproblem ABCF^u. This can be enforced, for instance, by setting its cost to $-\infty$.

Criterion 6 Let $A_d = \{(i,j) \mid j - i = l_d\}$, $d = 1, 2, \cdots, m$, be the set of valid arcs for order l_d. Suppose there is a set $\hat{A}_d \subset A_d$ of arcs such that the sum of the lower bounds imposed on them sum up the required demand, that is, $\sum_{l \in H^u_d} x_{ij} = b_d$ where $H^u_d \subset H^u$ is the set of branching constraints imposed on arcs that belong to $\hat{A}_d$. There can be no optimal solution with overproduction. Therefore, all the remaining arcs $(i,j) \in A_d / \hat{A}_d$ can be removed from the subproblem ABCF^u.

The above criterion 6 is equivalent to observation 8 in Vanderbeck et al. (1996).

Proposition 5 If the objective value of SBCF^u is less than U and there are no more attractive paths, the generation process may be aban-

doned, and the corresponding branch-and-bound node u fathomed, even if there are still attractive arcs, with a positive reduced cost, out of the restricted master problem.

It is well known that there are optimal degenerate extreme points that have bases that are optimal and bases that still have attractive variables. Intuitively, if we ignore the attractive arcs, we may spare some degenerate pivots before concluding that the extreme point is optimal.

ALGORITHM BCFE

step 1 According to criteria 1 – 4, we construct the valid set of arcs $\bar{A}$, let $g^u = H^u = \phi$. $u = 1$, $k(u) = 0$.

Step 2 Choose a subset $\tilde{A} \subset A$.

Step 3 Solve the programming SBCF^u and obtain the solution $\tilde{x}$ and the objective value $\tilde{z}^u$.

Step 4 If $\tilde{z}^u \geqslant U$.
then if $\tilde{x}_{ij}^u \in Z^1$ for any $A_{ij} \in \tilde{A}$.
then $\tilde{x}^u$ is the optimal solution , stop and output the solution.

Else if there exists $\tilde{x}_{ij}^u$ which is a fractional, let
$d = 2^{k(u)} + 2(u-1) + 1$.
$k(d) = k(u) + 1$.
$G^d = G^u \cup \{\tilde{x}_{ij}^u\}$.
$H^d = H^u$.
$d = 2^k + 2(u-1) + 1$.
$G^d = G^u$.
$H^d = H^u \cup \{\tilde{x}_{ij}^u\}$.

let $G^u = H^u = \phi$ and choose a node p satisfying G^p or $H^p \neq \phi$, let $u = p$ go to Step 3.
else go to step 5.

step 5 Solve ABCF^u and get the solution $\tilde{x}^a$, which corresponds to a path from 0 to vertex W. If the objective value is larger than 0, then Let $\tilde{A} = \tilde{A} \cup \{\tilde{x}_{ij}^a : \tilde{x}_{ij}^a \neq 0\}$ go to step 3.

step 6 Let $G^u = H^u = \phi$ and if there exists a node p satisfying G^p or $H^p \neq \phi$, let $u = p$ goto step 2.

step 7 let $U = U - 1$, goto Step 1.

According to the research of Kojima et al. (1989), SBCF^u can be solved in $0(|A|^3 L)$ time. From simple calculation, we can see that the BCFE algorithm will take $(U - z^*)(2^{|A|+1} - 2 + |A|)$ iterations, and that every iteration will take no less than $0(|A|^3 L)$ time. In fact, nearly all the computation time is spent in solving the mixed integer subproblem ABCF^u. It is obvious that the above algorithm can be terminated in finite steps. The computational test will be given in a future article.

ACKNOWLEDGEMENTS

We are grateful to Professor Y. Y. Ye and He Y. for several helpful discussions.

References

Assmann, S.F., Johnson, D.S., Kleitman, D.J. et al., 1984. On a dual version of the one-dimensional Bin packing problem. *J. of Algorithms*, **5**: 502-525.

Chan, L., Simchi-Levi, D., Bramel, J., 1998. Worst case analyses, linear programming and the bin-packing problem. *Mathematical Programming*, **83**: 213 – 227.

Csirik, J., Frenk, J.B.G., Galambos, G. et al., 1988. Online algorithmsfor a dual version of bin packing. *Disc. Appl. Math.*, **21**: 163 – 167.

Desrochers, M., Desrosiers, J., Solomon, M., 1992. A new optimization algorithm for thevehicle routing problem with time windows. *Operations Research*, **40**: 342 – 354.

Hooffman, K.L., Padberg, M., 1993. Solving airline crew scheduling problems by branch-and-cut. *Management Science*, **39**: 657 – 682.

Kojima, M., Mizuno, S., Yoshise, A., 1989. A primal-dual interior point method for linear programming, *In*: N. Meggido(ed.), Progress in Mathematical Programming. Springer-Verlag, New York, p. 29 – 47.

Labbé, M., Laporte, G., Martello, S., 1995. An exact algorithm for the dual bin packing problem. *Operations Research Letters*, **17**: 9 – 18.

Simchi-Levi, D., 1994. New worst-case results for the bin-packing problem. *Naval Research Logistics*, **41**: 579 – 585.

Su, C.J., Yao, E.Y., 1999. On-line bin-covering problem with kernels. *OR Transactions*, **3**(4): 71 – 78 (in Chinese, with English abstract).

Vanderbeck, F., Laurence Wolsey, A., 1996. An exact algorithm for IP column generation. *Operations Research Letters*, **19**: 151 – 159.

Vllerio de Carvalho, J.M., 1999. Exact solution of bin-packing problems using column gener ation and branch-and-bound. *Annals of Operations Research* , **86**: 629 – 659.