Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
Computers & Operations Research 33 (2006) 1505–1520
computers & operations research
www.elsevier.com/locate/cor

Generating optimal two-section cutting patterns for rectangular blanks

Yaodong Cui*, Dongli He, Xiaoxia Song

Department of Computer Science, Guangxi Normal University, Guilin, Guangxi 541004, China

Abstract

This paper presents an algorithm for generating unconstrained guillotine-cutting patterns for rectangular blanks. A pattern includes at most two sections, each of which consists of strips of the same length and direction. The sizes and strip directions of the sections must be determined optimally to maximize the value of the blanks cut. The algorithm uses an implicit enumeration method to consider all possible section sizes, from which the optimal sizes are selected. It may solve all the benchmark problems listed in the OR-Library to optimality. The computational results indicate that the algorithm is efficient both in computation time and in material utilization. Finally, solutions to some problems are given.

Keywords: Guillotine; Optimization; Cutting stock problem; Two-dimensional cutting

1. Introduction

The unconstrained two-dimensional cutting problem is the problem of cutting from a single rectangular sheet a number of smaller rectangular blanks, each of which is of a given size and a given value, so as to maximize the value of the blanks cut. This problem appears in the cutting of steel sheets into required sizes, in the cutting of wood sheets to make furniture, and in several other industrial areas. The related problem of minimizing the amount of waste produced by the cutting can be converted into this problem by making the value of all blanks equal to their areas.

* Corresponding author. Fax: +86 773 5812383
E-mail address: ydcui@263.net (Y. Cui).

0305-0548/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2004.09.022

In practice it is often sufficient to consider only guillotine cuts. A guillotine cut on a rectangle is a cut from one edge of the rectangle to the opposite edge that is parallel to the two remaining edges. We refer to the unconstrained two-dimensional guillotine-cutting problem as the UTDGCP.

Young-Gun and Kang [1] pointed out that the UTDGCP should be distinguished from the two-dimensional guillotine-cutting stock problem (TDGCSP). The TDGCSP is the problem of cutting all required number of small rectangular blanks of different sizes from a set of large rectangular sheets at minimum sheet cost. A solution of the TDGCSP consists of several cutting patterns, each of which may be constructed by an algorithm for the UTDGCP. The linear programming (LP) approach is widely used to solve the TDGCSP [2]. It solves the TDGCSP through iteration. A large number of UTDGCP must be solved before the LP approach finds a solution close to optimal.

There are several exact algorithms for the UTDGCP, such as the dynamic-programming procedures in [3–4], the tree-search procedures in [5–7]. Because the problem is NP hard, these algorithms are not adequate for large-scale problems. They are also not adequate for constructing cutting patterns when the LP approach is used to solve the TDGCSP.

Restrictions on the cutting patterns are often used to speed up the solution process, such as the staged cutting patterns [3,4]. Hifi [8] presented two exact algorithms for large-scale unconstrained two and three staged cutting problems. Fayard and Zissimopoulos [9] proposed a heuristic that selects an optimal subset of optimal generated strips by solving a sequence of one-dimensional knapsack problems. Although restrictions may decrease the value of the cutting patterns, they may cut down the computation time drastically.

This paper puts forward a new restriction on the cutting patterns. It suggests the application of two-section patterns. A two-section pattern contains at most two sections. All strips in a section are of the same length and direction. A strip is either a uniform strip that includes only blanks of the same size, or a general strip consisting of blanks of different sizes. An exact algorithm (referred to as the TSEC algorithm) for generating optimal two-section patterns is constructed. It considers all possible section sizes implicitly and finds the optimal sizes through comparisons. To compare the algorithm with the algorithms for non-staged and staged cutting patterns, computations were performed on both benchmark problems and random problems. The results indicate that the TSEC algorithm is efficient both in computation time and in material utilization, and may simplify the cutting process.

It should be noted that the algorithm presented might be also seen as an improved version of that of Fayard and Zissimopoulos [9]. Their algorithm uses a series of one-dimensional knapsack problems for generating a set of optimal strips and then fills them in the sheet in an optimal way. Although not definitely pointed out, their algorithm generates only two-section patterns.

2. Two-section patterns

2.1. Notation and functions

Table 1 lists some notation and functions to be used. Most of them will be re-introduced where they are used for the first time. The reader may find it is more convenient to look for the notation definition in the table than in the text.

Table 1
Notation and functions

L, W	Length and width of the stock sheet
l_i, w_i, c_i	Length, width and value of the *i*th rectangular blank, $i = 1, 2, \ldots, m$
$F_X(x)$	Value of *X*-section $x \times W$ in an *X*-pattern
$F_Y(x)$	Value of *Y*-section $x \times W$ in an *X*-pattern
$f_X(y)$	Value of *X*-section $L \times y$ in a *Y*-pattern
$f_Y(y)$	Value of *Y*-section $L \times y$ in a *Y*-pattern
$\|L\|$	Number of *X* break points between 0 and *L*
$\|W\|$	Number of *Y* break points between 0 and *W*
$\boldsymbol{B_X}$	The set of *X* break points, $\boldsymbol{B_X} = \{b_{x,1}, b_{x,2}, \ldots, b_{x,\|L\|}\}$
$\boldsymbol{B_Y}$	The set of *Y* break points, $\boldsymbol{B_Y} = \{b_{y,1}, b_{y,2}, \ldots, b_{y,\|W\|}\}$
$u(i, x)$	Value of an *X*-strip $x \times w_i$, $0 \leqslant x \leqslant L$, $i = 1, 2, \ldots, m$
$v(i, y)$	Value of a *Y*-strip $y \times l_i$, $0 \leqslant y \leqslant W$, $i = 1, 2, \ldots, m$
$\mathrm{int}(x)$	Return the maximum integer not larger than *x*

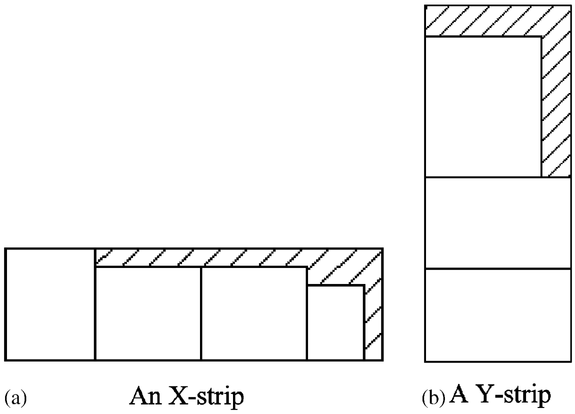

Fig. 1. Strips: (a) an *X*-strip, (b) a *Y*-strip.

2.2. Strips

Assume that the direction of the blanks is fixed. This restriction is not serious, since if a blank $l \times w$ can be cut with either direction, it may be treated as two different blanks $l \times w$ and $w \times l$, each of which has a fixed direction.

Both general and uniform strips will be considered in generating two-section patterns. A uniform strip contains only blanks of the same size and direction. As shown in Fig. 1, a strip may be either in *X* or *Y* direction. A strip is an *X*-strip if it is in *X* direction; otherwise it is a *Y*-strip. Blanks in *X*-strips are left justified, and those in *Y*-strips are bottom justified. The strip length of an *X*-strip is measured horizontally, and the strip length of a *Y*-strip is measured vertically.

Only strips of width equal to a blank width (horizontal strips) or length (vertical strips) will be considered. Assume that there are *m* blanks. The *i*th blank is of size $l_i \times w_i$, with l_i and w_i being positive integers, $i = 1, 2, \ldots, m$. Then the *i*th *X*-strip is of width w_i, and the *i*th *Y*-strip is of width l_i, $i = 1, 2, \ldots, m$.

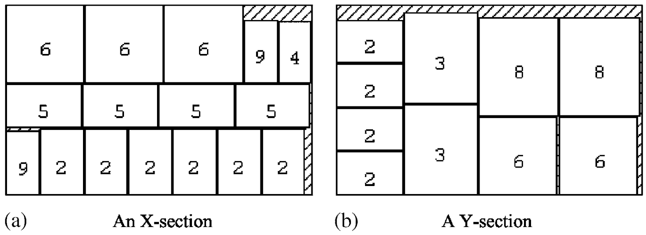

Fig. 2. Sections: (a) an X-section, (b) a Y-section.

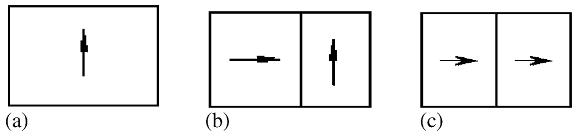

Fig. 3. X-patterns.

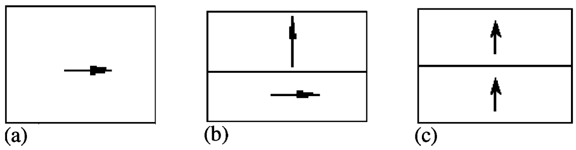

Fig. 4. Y-patterns.

2.3. Sections

Strips in a section are of the same direction and length. The direction of the strips is referred to as the section direction. A section is an X-section if the strips are in X direction, otherwise it is a Y-section. Fig. 2a shows an X-section and Fig. 2b shows a Y-section.

2.4. Two-section patterns

Figs. 3 and 4 show all possible patterns that should be considered. The arrow in each section indicates the direction of the section. The two sections may be justified either vertically (Y-pattern) or horizontally (X-pattern). The patterns of Figs. 3a and 4a are one-section patterns, which are the special cases of two-section patterns. The pattern of Fig. 3a may be seen as a two-section pattern containing two Y-sections, and the pattern of Fig. 4a may be taken as a two-section pattern consisting of two X-sections. Assume that the sheet size is $L \times W$. The width of an X-section in an X-pattern is W (Fig. 3b), and the length of a Y-section in a Y-pattern is L (Fig. 4b).

2.5. Break-points

Many authors [4–9] have used normal sets to develop algorithms for guillotine-cutting patterns. The elements of the normal sets are referred to as break points in this paper. They are defined as:

$$X \text{ break points}: \quad x = \sum_{i=1}^{m} z_i l_i \leqslant L, \quad z_i \geqslant 0 \quad \text{and integer},$$
$$Y \text{ break points}: \quad y = \sum_{i=1}^{m} z_i w_i \leqslant W, \quad z_i \geqslant 0 \quad \text{and integer}. \tag{1}$$

Denote the set of all X break points by $\boldsymbol{B_X}$, $\boldsymbol{B_X} = \{b_{x,1}, b_{x,2}, \ldots, b_{x,|L|}\}$, where $|L|$ is the number of X break points between 0 and L, $b_{x,i} < b_{x,i+1}$ for $1 \leqslant i < |L|$, $b_{x,1} = 0$ and $b_{x,|L|} = L$. Denote the set of all Y break points by $\boldsymbol{B}_Y$, $\boldsymbol{B}_Y = \{b_{y,1}, b_{y,2}, \ldots, b_{y,|W|}\}$, where $|W|$ is the number of Y break points between 0 and W, $b_{y,i} < b_{y,i+1}$ for $1 \leqslant i < |W|$, $b_{y,1} = 0$ and $b_{y,|W|} = W$. Break points have the following properties.

Property 1. For an X-section $x \times W$ in an X-pattern, assume that x_0 is the maximum X break point not larger than x, and $F_X(x)$ is the maximum value of the section, then $F_X(x) = F_X(x_0)$.

Property 2. For a Y-section $L \times y$ in a Y-pattern, assume that y_0 is the maximum Y break point not larger than y, and $f_Y(y)$ is the maximum value of the section, then $f_Y(y) = f_Y(y_0)$.

2.6. Strip values

Assume that c_i is the value of the ith blank, $u(i, x)$ is the maximum value of an X-strip $x \times w_i$, $S(i) = \{j | w_j \leqslant w_i, 1 \leqslant j \leqslant m\}$, $i = 1, \ldots, m$, then

$$u(i, x) = \max \sum_{j \in S(i)} c_j z_j, \quad \sum_{j \in S(i)} l_j z_j \leqslant x, \quad z_j \geqslant 0 \quad \text{and integer}, \tag{2}$$

where z_j is the number of blank j in strip $x \times w_i$. Similarly, assume that $v(i, y)$ is the maximum value of a Y-strip $l_i \times y$, $T(i) = \{j | l_j \leqslant l_i, 1 \leqslant j \leqslant m\}$, $i = 1, \ldots, m$, then

$$v(i, y) = \max \sum_{j \in T(i)} c_j z_j, \quad \sum_{j \in T(i)} w_j z_j \leqslant y, \quad z_j \geqslant 0 \quad \text{and integer}, \tag{3}$$

where z_j is the number of blank j in strip $l_i \times y$.

Models (2) and (3) are knapsack problems [10]. As pointed out by many authors [4,8,9], to obtain all X-strip values $u(i, x)$, $i = 1, \ldots, m$, $x = 0, 1, \ldots, L$, it is necessary only to apply dynamic programming to solve one knapsack problem related to the widest X-strip. Similarly, to obtain all Y-strip values $v(i, y)$, $i = 1, \ldots, m$, $y = 0, 1, \ldots, W$, it is necessary only to solve the knapsack problem related to the widest Y-strip.

2.7. Section values

The following model may determine the value of X-section $x \times W$ in an X-pattern:

$$F_X(x) = \sum_{i=1}^{m} u(i, x) z_i, \quad \sum_{i=1}^{m} w_i z_i \leqslant W, \quad z_i \text{ non-negative integers}, \quad i = 1, 2, \ldots, m \tag{4}$$

z_i is the number of the ith strip in the section. To obtain $F_x(x)$ for all x between 0 and L, it is sufficient to solve the above model $|L|$ times from Property 1.

The following model determines the value of Y-section $x \times W$ in an X-pattern:

$$F_Y(x) = \sum_{i=1}^{m} v(i, W) z_i, \quad \sum_{i=1}^{m} l_i z_i \leqslant x, \quad z_i \text{ non-negative integers}, \quad i = 1, 2, \ldots, m. \tag{5}$$

Solving the model only once is enough to obtain all $F_Y(x)$, $0 \leqslant x \leqslant L$.

The following model may determine the value of Y-section $L \times y$ in a Y-pattern.

$$f_Y(y) = \sum_{i=1}^{m} v(i, y) z_i, \quad \sum_{i=1}^{m} l_i z_i \leqslant L, \quad z_i \text{ non-negative integers}, \quad i = 1, 2, \ldots, m. \tag{6}$$

To obtain all $f_Y(y)$ for $0 \leqslant y \leqslant W$, it is sufficient to solve the above model $|W|$ times from Property 2.

The following model determines the value of X-section $L \times y$ in a Y-pattern.

$$f_X(y) = \sum_{i=1}^{m} u(i, L) z_i, \quad \sum_{i=1}^{m} w_i z_i \leqslant y, \quad z_i \text{ non-negative integers}, \quad i = 1, 2, \ldots, m. \tag{7}$$

Solving the model only once is enough to obtain all $f_X(y)$, $0 \leqslant y \leqslant W$.

Models (4)–(7) are all knapsack functions and the algorithms for them may be found in Refs. [3] and [10].

3. The TSEC Algorithm

3.1. The TSEC algorithm

Step 1. Get all X and Y break points from Eqs. (1).
Step 2. Determine strip values from models (2) and (3).
Step 3. Determine all Y-section values in an X-pattern, $F_Y(x)$, $0 \leqslant x \leqslant L$, from Model (5).
Step 4. Determine all X-section values in an X-pattern, $F_X(x)$, $0 \leqslant x \leqslant L$, from Model (4).
Step 5. Perform sub-algorithm A below to obtain the optimal X-pattern and its value V_X.
Step 6. Determine all X-section values in a Y-pattern, $f_X(y)$, $0 \leqslant y \leqslant W$, from Model (7).
Step 7. Determine all Y-section values in a Y-pattern, $f_Y(y)$, $0 \leqslant y \leqslant W$, from Model (6).
Step 8. Perform sub-algorithm B below to obtain the optimal Y-pattern and its value V_Y.
Step 9. The optimal value is $\max(V_X, V_Y)$.

3.2. Sub-algorithm A

Step 1. Let $V_X = 0$, $x_0 = 0$, and $i = 0$.
Step 2. Let $i = i + 1$. Go to step 5 if $i > |L|$.
Step 3. Let $x = b_{x,i}$, $V = F_X(x) + \max\{F_X(L - x), F_Y(L - x)\}$. Go to step 2 if $V < V_X$.
Step 4. Let $V_X = V$, $x_0 = x$. Go to Step 2.
Step 5. The best *X*-pattern has been found. The size of the *X*-section is $x_0 \times W$, and the size of the *Y*-section is $(L - x_0) \times W$.

3.3. Sub-algorithm B

Step 1. Let $V_Y = 0$, $y_0 = 0$, and $i = 0$.
Step 2. Let $i = i + 1$. Go to step 5 if $i > |W|$.
Step 3. Let $x = b_{y,i}$, $V = f_Y(y) + \max\{f_Y(W - y), f_X(W - y)\}$. Go to step 2 if $V < V_Y$.
Step 4. Let $V_Y = V$, $y_0 = y$. Go to Step 2.
Step 5. The best *Y*-pattern has been found. The size of the *Y*-section is $L \times y_0$, and the size of the *X*-section is $L \times (W - y_0)$.

3.4. The techniques used to improve the time efficiency of the algorithm

Below we describe three techniques to improve the time efficiency of the algorithm. Although only *X*-patterns are discussed, similar techniques apply to *Y*-patterns.

Technique 1: Use lower bounds.

In step 4 of the algorithm, $|L|$ knapsack problems must be solved to obtain all *X*-section values in an *X*-pattern, $F_X(x)$, $0 \leqslant x \leqslant L$. We know that the *X* break points have been arranged in ascending order in set $\boldsymbol{B_X} = \{b_{x,1}, b_{x,2}, \ldots, b_{x,|L|}\}$. Assume that x_1 and x_2 are two *X* break points and $x_1 < x_2$, then $F_X(x_1) \leqslant F_X(x_2)$. The optimal solution of section $x_1 \times W$ is a feasible solution of section $x_2 \times W$ because of $x_1 < x_2$. Therefore, it may be taken as the initial solution of section $x_2 \times W$. This will speed up the solution progress.

Technique 2: Skip some strips.

In the process of improving the solution of $x_2 \times W$, some strips may be skipped. Strip $x_2 \times w_i$ may be skipped if $u(x_2, i) = u(x_1, i)$, $1 \leqslant i \leqslant m$. In this case, strip $x_2 \times w_i$ may be shortened to $x_1 \times w_i$, without decreasing the value of the strip. The reason why this strip can be skipped is as follows. Introducing a strip of $x_1 \times w_i$ into the solution of section $x_1 \times W$ will not increase the value of the section, for that the solution is already optimal. Similarly, introducing a strip of $x_1 \times w_i$ into the initial solution of section $x_2 \times W$ will not improve the solution because that the initial solution is indeed the optimal solution of section $x_1 \times W$. Introducing a strip of $x_2 \times w_i$ will also not improve the solution because it has the same value and width as those of strip $x_1 \times w_i$. The computational results indicate that this technique may greatly decrease the number of strips to be considered.

Technique 3: Skip some sections.

Assume that we have obtained the *X*-section values for $x < L/2$. Let x_3 be the *X* break point next to x_2 and $x_2 \geqslant L/2$. Consider two *X*-patterns. One is the *X*-pattern related to division x_2 (the vertical cut at x_2 divides the sheet into two sections), and the other is the *X*-pattern related to division x_3. Their values are

$$V(x_2) = \max(F_X(x_2), F_Y(x_2)) + \max\{F_X(L - x_2), F_Y(L - x_2)\},$$
$$V(x_3) = \max(F_X(x_3), F_Y(x_3)) + \max\{F_X(L - x_3), F_Y(L - x_3)\}.$$

Both $F_X(L - x_2)$ and $F_X(L - x_3)$ are known because of $L - x_2 \leqslant L/2$ and $L - x_3 \leqslant L/2$. $F_Y(L - x_2)$ and $F_Y(L - x_3)$ have been determined in step 3. We have $\max(F_X(x_2), F_Y(x_2)) \leqslant \max(F_X(x_3), F_Y(x_3))$ because of $x_2 < x_3$. Division x_2 may be skipped if $\max\{F_X(L - x_2), F_Y(L - x_2)\} = \max\{F_X(L - x_3), F_Y(L - x_3)\}$. The reason is that the second X-pattern (related to division x_3) has a value not smaller than that of the first pattern, namely $V(x_3) \geqslant V(x_2)$.

3.5. The total number of knapsack problems to be solved

Two knapsack problems must be solved is two in step 2, one in step 3, at most $|L|$ in step 4, one in step 6, and at most $|W|$ in step 7. The total number of knapsack problems that should be solved is $4+|L|+|W|$. The computational results indicate that the real number of knapsack problems solved is less than half of that required by the algorithm of Fayard and Zissimopoulos [9].

3.6. Relationship between two-section patterns and staged patterns

Each section of a two-section pattern may be seen as a two-stage pattern. A two-section pattern may be a two-stage pattern, if it contains only one section. A two-section pattern may also be a three-stage pattern, if it consists of two sections.

4. The computational results

The computation was performed on a computer with clock rate 1.9 GHz and inner memory 256 MB. The test problems are grouped and identified by their group ID. The following notations are used to denote algorithms:

TSEC_G	The TSEC algorithm where general strips are used.
TSEC_U	The TSEC algorithm where only uniform strips are used. A uniform strip consists of blanks of the same size.
BOPT	Bealsy's [4] exact algorithm for the problem of general (non-staged) guillotine cutting.
B_3STG	Bealay's [4] algorithm for three-stage cutting.
G_2STG	Gilmore and Gomory's [11] algorithm for two-stage cutting.
H_2STG	Hiff's [8] exact algorithm for two-stage cutting.
H_3STG	Hiff's [8] exact algorithm for three-stage cutting.
FZ	The heuristic for guillotine cutting proposed by Fayard and Zissimopoulos [9].

4.1. Test on small-scale problems

Problems in Group 1 are the 13 test problems used in [4], which are available from the OR-Library. The optimal values of the first 12 problems are given in Table 1 of Ref. [4], and the optimal value of problem 13 is 8997780. Both the TSEC_G and the TSEC_U can solve all the problems to optimality,

Table 2
The computational results of groups 2–4

ID	Material utilization (%)				Number of problems solved to optimality			
	BOPT	TSEC_G	G_2STG	B_3STG	BOPT	TSEC_G	G_2STG	B_3STG
2	97.57	97.52	97.20	97.52	200	187	116	187
3	97.45	97.42	97.18	97.42	200	190	134	190
4	97.32	97.30	96.93	97.30	200	191	121	191

Table 3
The computational results of Group 5

No.	v_1	v_2	v_3	v_2/v_1 (%)	v_3/v_1 (%)	t_1	t_2	t_3
1	2997959	2993728	2996488	99.86	99.95	82.88	0.01	0.11
2	3000000	2999328	3000000	99.98	100.00	113.70	0.03	0.24
3	3000000	2999858	3000000	100.00	100.00	130.94	0.05	0.34
4	3000000	3000000	3000000	100.00	100.00	148.75	0.08	0.77
5	2992650	2992650	2992650	100.00	100.00	16.01	0.00	0.03
6	2997532	2997404	2997404	100.00	100.00	34.91	0.01	0.06
7	2997905	2997116	2997116	99.97	99.97	47.24	0.01	0.10
8	3000000	2998806	3000000	99.96	100.00	67.79	0.03	0.29
9	2960976	2960976	2960976	100.00	100.00	0.12	0.00	0.00
10	2962298	2961492	2961492	99.97	99.97	0.77	0.00	0.01
11	2986070	2986070	2986070	100.00	100.00	3.60	0.00	0.02
12	2983946	2978892	2982822	99.83	99.96	8.09	0.01	0.04
13	2988296	2972060	2988296	99.46	100.00	1.65	0.00	0.02

that is, the values are the same as those of the BOPT. The computation time for each one of the first 12 problems was less than 0.001 s, and could not be reported by the computer. The TSEC_G solved the 13 problem in 0.13 s, the TSEC_U in 0.01 s, and the BOPT in 92.24 s.

Group 2–4 are random problems, with 200 problems in each group. The sheet size $L \times W$ is 250×250 for Group 2, 500×500 for Group 3, and 1000×1000 for Group 4. Each problem includes 30 blanks. The length l_i of each blank was generated by sampling an integer from the uniform distribution $[L/4, 3L/4]$, with the width w_i of each blank being generated by sampling an integer from the uniform distribution $[W/4, 3W/4]$. The value of each blank is equal to its area.

We use material utilization to measure the quality of the solution, which is equal to s/s_0, where s is the area of all blanks yield by the cutting pattern, and s_0 is the sheet area. Table 2 gives the summary of the computational results. The computation time is omitted because the scale of the problems is too small for comparing time efficiency. For any problem in these groups, the value of the optimal two-section pattern is the same as that of the optimal three-stage pattern.

4.2. *Test on medium-scale problems*

The blank data of Group 5 are the same as the first group. The sheet size is 2000×1500 for all problems. The computational results are summarized is Table 3, where v_1 and t_1 denote the value and computation time of the BOPT, t_2 and v_2 denote those of the TSEC_U, t_3 and v_3 denote those of the TSEC_G.

Table 4
Six test problems in Group 6

P1	201 × 207, 277 × 450, 612 × 640, 244 × 407, 695 × 554, 278 × 648, 540 × 381, 522 × 528, 638 × 628, 259 × 221, 255 × 477, 638 × 221, 284 × 335, 491 × 546, 592 × 243, 411 × 451, 474 × 307, 318 × 399, 664 × 600, 308 × 460, 632 × 557, 229 × 285, 467 × 647, 200 × 366, 323 × 231, 251 × 506, 517 × 510, 505 × 427, 635 × 390, 358 × 617
P2	424 × 556, 546 × 216, 401 × 547, 370 × 640, 516 × 292, 220 × 235, 530 × 389, 523 × 201, 438 × 532, 279 × 221, 460 × 404, 680 × 460, 340 × 657, 321 × 682, 495 × 362, 370 × 490, 556 × 555, 318 × 400, 547 × 349,626 × 486, 417 × 547, 245 × 442, 552 × 308, 312 × 444, 576 × 410, 654 × 425, 562 × 296, 396 × 212, 502 × 295, 262 × 567
P3	294 × 403, 374 × 539, 526 × 633, 386 × 274, 640 × 544, 295 × 526, 570 × 360, 296 × 297, 413 × 562, 591 × 605, 470 × 523, 325 × 204, 336 × 370, 671 × 451, 518 × 617, 307 × 696, 547 × 556, 470 × 462, 615 × 211, 666 × 534, 631 × 678, 415 × 691, 687 × 206, 246 × 582, 441 × 693, 515 × 381, 237 × 296, 206 × 241, 242 × 214, 513 × 491
P4	691 × 454, 554 × 285, 414 × 567, 319 × 347, 416 × 543, 647 × 367, 255 × 336, 667 × 315, 558 × 349, 245 × 209, 643 × 356, 207 × 640,586 × 643, 439 × 242, 559 × 356, 476 × 328, 686 × 582, 531 × 210, 261 × 200, 458 × 229, 394 × 534, 202 × 512, 600 × 441, 683 × 698, 609 × 654, 529 × 283, 263 × 627, 573 × 478, 377 × 287, 204 × 492
P5	446 × 574, 477 × 205, 582 × 536, 511 × 569, 228 × 549, 615 × 660, 446 × 552, 400 × 320, 662 × 578, 365 × 668, 467 × 402, 233 × 663, 349 × 318, 612 × 288, 688 × 568, 536 × 427, 503 × 209, 603 × 676, 443 × 420, 320 × 248, 284 × 348, 517 × 262, 470 × 237, 318 × 321, 411 × 300, 692 × 643, 521 × 603, 206 × 527, 311 × 290, 233 × 318
P6	374 × 256, 635 × 686, 335 × 479, 429 × 666, 376 × 225,653 × 480, 297 × 285, 204 × 300, 421 × 318, 355 × 284, 353 × 442, 535 × 697, 486 × 304, 318 × 430, 487 × 262, 562 × 236, 631 × 366, 676 × 681, 616 × 290, 326 × 534, 593 × 638, 388 × 412, 592 × 315, 343 × 235, 376 × 422, 544 × 278, 594 × 434, 469 × 617, 465 × 409, 279 × 372

Group 6 includes 50 test problems. The sheet size is 3000 × 1500. The length or width of each blank was generated by sampling an integer from the uniform distribution [200, 700]. The value of each blank was set equal to its area. The computational results are summarized below, where *u* denotes the average material utilization, and *t* denotes the average computation time of one problem.

	BOPT	TSEC_G	TSEC_U	G_2STG	B_3STG
u (%)	99.74	99.64	99.56	99.44	99.66
t (seconds)	50.20	0.06	0.01	0.01	96.70

Table 4 lists the blank data of the first six test problems. Table 5 shows the related computational results. The meanings of t_1, t_2, t_3, v_1, v_2 and v_3 are the same as those in Table 3. Fig. 5 gives the two-section patterns generated by the TSEC_G.

4.3. Test on large-scale problems

The problems in Group 7 are the eight test problems in [8] and [12], namely problems U1, U2, U3, U4, W1, W2, W3 and W4. Hifi [8] solved these problems by the H_2STG and H_3STG algorithms (see

Table 5
The computational results of the six test problems in Table 4

No.	v_1	v_2	v_3	v_2/v_1 (%)	v_3/v_1 (%)	t_1	t_2	t_3
1	4490544	4480372	4480372	99.77	99.77	68.87	0.01	0.06
2	4488944	4484676	4484676	99.90	99.90	54.18	0.01	0.06
3	4489836	4484956	4489341	99.90	99.99	60.69	0.01	0.06
4	4487967	4475765	4483009	99.73	99.89	66.27	0.01	0.06
5	4485616	4480978	4482627	99.90	99.93	52.34	0.01	0.06
6	4494340	4491008	4491008	99.93	99.93	50.34	0.01	0.06

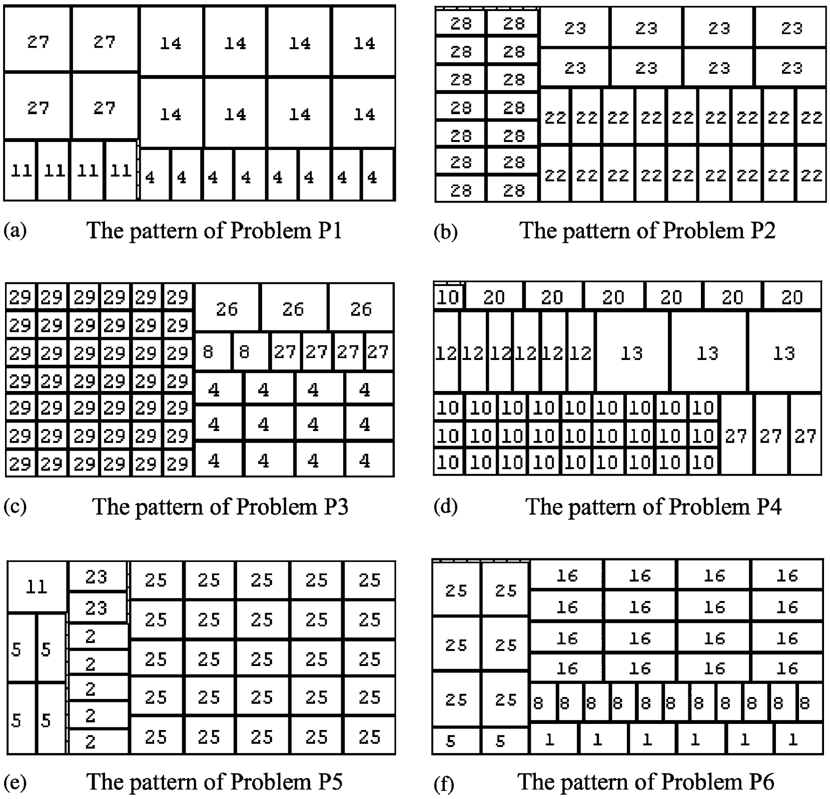

Fig. 5. The two-section patterns of the six test problems listed in Table 4: (a) The pattern of Problem P1, (b) The pattern of Problem P2, (c) The pattern of Problem P3, (d) The pattern of Problem P4, (e) The pattern of Problem P5, (f) The pattern of Problem P6.

Tables 2 and 3 in [8]). Table 6 shows the computational results, where v_1 and t_1 denote the value and computation time of the TSEC_U, t_2 and v_2 denote those of the TSEC_G, t_3 and v_3 denote those of the H_2STG, t_4 and v_4 denote those of the H_3STG. The clock rate of the computer used by Hifi [8] was 0.1 GHZ, and the readers should be aware of the low clock rate of this computer in interpreting the computational results (the computation of this paper was performed on a computer with clock rate 1.9 GHz).

Table 6
The computational results of Group 7

ID	v_1	v_2	v_3	v_4	t_1	t_2	t_3	t_4
U1	22397400	22416630	22338048	22397400	0.02	0.22	6.10	7693.57
U2	20355161	20382215	20355161	20355161	0.02	0.06	4.31	1652.90
U3	48171147	48239155	48144064	48239155	0.04	0.37	13.70	>10800
U4	48350130	48350130	48350130		0.04	1.66	32.56	
W1	167751	168289	162279	167751	0.02	0.18	8.91	8280.80
W2	37617	37621	37621	37621	0.01	0.03	4.59	1427.35
W3	253617	253617	250830	253617	0.06	0.36	23.08	>10800
W4	378366	378366	377910		0.14	0.71	57.29	

The H_3STG could not verify the optimality of problems U3 and W3 because that the computation time for each problem was not allowed to exceed 3 h. It also could not solve problems U4 and W4 for the large scale of the problem sizes.

These eight problems are all large-scale problems. The sheet size of them is 4500×5000, 5050×4070, 7350×6579, 7350×6579, 5000×5000, 3427×2769, 7500×7381, and 7500×7381, respectively. The numbers of blank sizes are 10, 10, 20, 40, 20, 20, 40, and 80 respectively.

As mentioned in Section 3.6, the TSEC algorithms actually generate two and three-stage patterns. v_4 should not be smaller than v_2 for any problem, for that it is the value of the optimal three-staged pattern. From Table 6 we know that $v_4 < v_2$ for problems U1, U2 and W1. Therefore, we can conclude that the author made some errors in presenting the computational results in [8]. To support this conclusion, the two-section patterns of these three problems are given in Figs. 6–8, and the blank data included are as follows (for problems U1 and U2: blank id × number of the blank × length × width; for problem W1: blank id × number of the blank × length × width × value):

U1: $1 \times 9 \times 437 \times 1490$, $3 \times 20 \times 237 \times 932$, $10 \times 20 \times 659 \times 921$,
U2: $3 \times 12 \times 932 \times 1107$, $4 \times 2 \times 598 \times 732$, $5 \times 7 \times 569 \times 1321$, $7 \times 2 \times 1248 \times 747$,
W1: $1 \times 75 \times 437 \times 731 \times 2223$, $9 \times 1 \times 598 \times 562 \times 1564$.

4.4. Comparison between the TSEC and the FZ algorithms

Both the TSEC and the FZ [9] algorithms will generate optimal two-section patterns. The TSEC may be seen as an improved version of the FZ. The three techniques mentioned in Section 3.4 are not used by the FZ. We used test problems of Group 8 to test the TSEC and FZ algorithms. The sheet size is 8000×6000. The blank data are the same as Group 6. The average material utilization is 99.98%, which is the same for the two algorithms, because both of them can generate optimal two-section patterns. The average computation time for one problem is 16.23 s for the FZ, 1.98 s for the TSEC_G. The average number of knapsack problems solved for one problem is 22,787 for the FZ, 9542 for the TSEC_G. The average number of strips considered in solving a knapsack problem related to a section is 30 for the FZ, 1.75 for the TSEC_G. When the TSEC_U is applied, the average computation time is 0.15 s. The average material utilization is 99.97%, which is nearly the same as that of the TSEC_G or FZ. Both the TSEC_G and the TSEC_U are much more time efficient than the FZ.

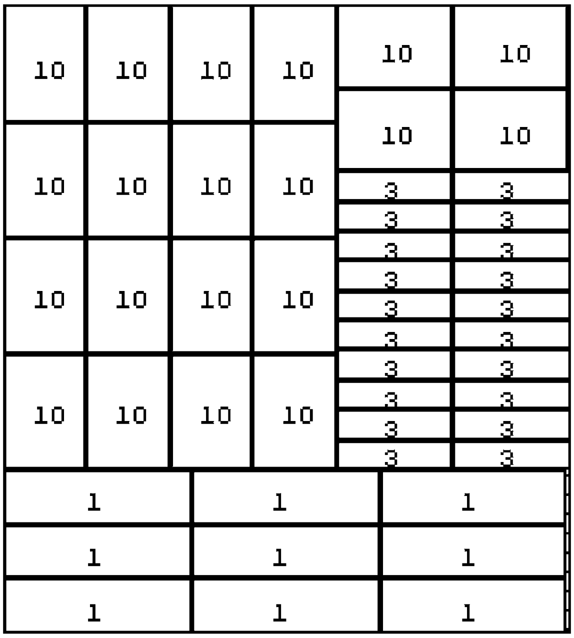

Fig. 6. The pattern of problem U1.

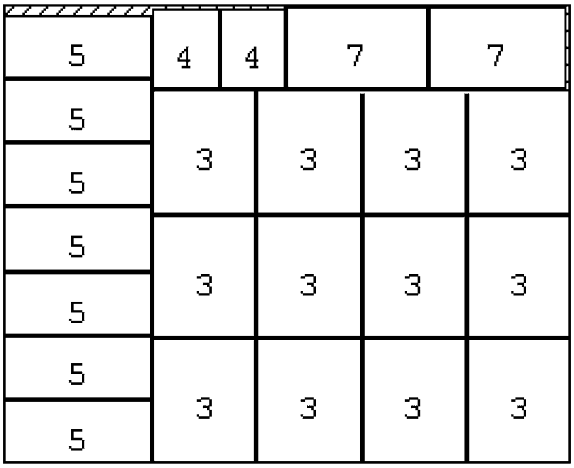

Fig. 7. The pattern of problem U2.

Below we give the solutions to the problems in Group 9, which may be used by future researchers or practitioners to test their algorithms. This group includes 12 problems. The sheet size is 8000 × 6000. The first six problems are those listed in Table 4. Problems P7–P12 are formed by combining the blank

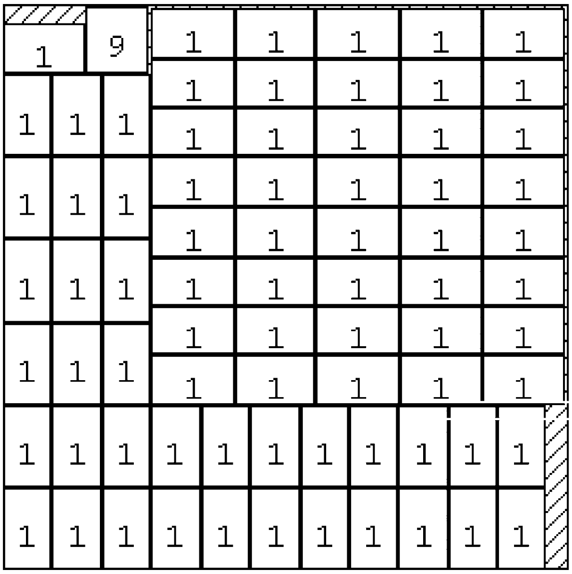

Fig. 8. The pattern of problem W1.

Table 7
The computational results of Group 9

ID	m	v_1	v_2	v_3	t_1	t_2	t_3
P1	30	47993491	47993491	47992398	21.17	2.01	0.16
P2	30	47991116	47991116	47991116	17.49	2.07	0.15
P3	30	47987624	47987624	47983659	14.56	1.80	0.16
P4	30	47993588	47993588	47993588	18.15	2.00	0.15
P5	30	48000000	48000000	48000000	13.87	2.16	0.16
P6	30	47997600	47997600	47997600	19.37	2.13	0.16
P7	60	48000000	48000000	48000000	38.41	5.91	0.36
P8	60	47998064	47998064	47998064	36.94	6.08	0.37
P9	60	48000000	48000000	48000000	39.72	6.06	0.35
P10	90	48000000	48000000	48000000	50.79	10.52	0.56
P11	90	48000000	48000000	48000000	56.15	10.50	0.54
P12	180	48000000	48000000	48000000	71.34	19.02	1.10

data of two or more problems from the first six. Namely,

Problem ID	P7	P8	P9	P10	P11	P12
Blank data	P1+P2	P3+P4	P5+P6	P1+P2+P3	P4+P5+P6	P1+P2+P3+ P4+P5+P6

The computational results are summarized in Table 7, where v_1 and t_1 are the value and computation time of the FZ, v_2 and t_2 are those of the TSEC_G, v_3 and t_3 are those of the TSEC_U.

5. Conclusions

The TSEC may generate optimal two-section patterns, which are either two-stage or three-stage patterns. For the small-scale problems tested, the TSEC can solve most of them to optimality, and the solutions are not inferior to those of the optimal three-stage patterns.

For medium-scale problems, the TSEC can give solutions very close to optimal. For the problems of Group 6, the average material utilization is very close to optimal (99.64% vs. 99.74%), superior to that of the optimal two-stage patterns (99.64% vs. 99.44%), and very close to that of the optimal three-stage patterns (99.64% vs. 99.66%). The average computation time is much more shorter than those of the B_OPT and B_3STG.

For the eight large-scale problems tested (Group 7), the TSEC gives solutions not inferior to those of the H_3STG. Although the TSEC was executed on a computer with a clock rate 19 times of that used by Hiff [8] (1.9 GHz vs. 0.1 GHz), we may still infer that it is more time efficient than the H_3STG from the huge difference in computation time shown in Table 6.

Applying the three techniques discussed in Section 3.4 makes the TSEC much more time efficient than the FZ. These techniques can also be used to improve the H_3STG. We have actually developed an algorithm based on these techniques, which is able to generate optimal three-stage patterns efficiently.

Acknowledgements

Guangxi Science Foundation supports this program under the grant No. 0236017. The authors wish to express their appreciation to the supporter. The authors are also grateful to the anonymous referee whose comments helped to improve an early version of this paper.

References

[1] Young-Gun G, Kang M-K. A new upper bound for unconstrained two-dimensional cutting and packing. Journal of the Operational Research Society 2002;53:587–91.
[2] Valerio de Carvalho JM. LP models for bin packing and cutting stock problems. European Journal of Operational Research 2002;14:253–73.
[3] Gilmore PC, Gomory RE. The theory and computation of knapsack functions. Operations Research 1966;14:1045–74.
[4] Beasley JE. Algorithms for unconstrained two-dimensional guillotine cutting. Journal of the Operational Research Society 1985;36:297–306.
[5] Herz JC. Recursive computational procedure for two-dimensional stock cutting. IBM Journal of Research and Development 1972;16:462–9.
[6] Christofides N, Whitlock C. An algorithm for two-dimensional cutting problems. Operations Research 1977;25:30–44.
[7] Hifi M, Zissimopoulos V. A recursive exact algorithm for weighted two-dimensional cutting. European Journal of Operational Research 1996;91:553–64.
[8] Hifi M. Exact algorithms for large-scale unconstrained two and three staged cutting problems. Computational Optimization and Applications 2001;18:63–88.
[9] Fayard D, Zissimopoulos V. An approximation algorithm for solving unconstrained two-dimensional knapsack problems. European Journal of Operational Research 1995;84:618–32.

[10] Andonov R, Poirrez V, Rajopadhye S. Unbounded knapsack problem: Dynamic programming revisited. European Journal of Operational Research 2000;123:394–407.
[11] Gilmore PC, Gomory RE. Multistage cutting stock problems of two and more dimensions. Operations Research 1965;13: 94–120.
[12] Hifi M. The DH/KD algorithm: a hybrid approach for unconstrained two-dimensional cutting problems. European Journal of Operational Research 1997;97:41–52.