Computers & Operations Research 39 (2012) 3031–3039

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Heuristic for constrained T-shape cutting patterns of rectangular pieces

Yaodong Cui*, Baixiong Huang

School of Computer, Electronics and Information, Guangxi University, Nanning, 530004, China

ARTICLE INFO

Available online 15 March 2012

Keywords:
Cutting stock
Two-dimensional cutting
T-shape patterns

ABSTRACT

T-shape patterns are often used in dividing stock plates into rectangular pieces, because they make good balance between plate cost and cutting complexity. A dividing cut separates the plate into two segments, each of which contains parallel strips, and the strip orientations of the two segments are perpendicular to each other. This paper presents a heuristic algorithm for constrained T-shape patterns, where the optimization objective is to maximize the pattern value, and the frequency of each piece type does not exceed the demand. The algorithm considers many dividing-cut positions, determines the pattern value associated to each position using a layout-generation procedure, and selects the one with the maximum pattern value as the solution. Pseudo upper bounds are used to skip some non-promising positions. The computational results show that the algorithm is fast and able to get solutions better than those of the optimal two-staged patterns in terms of material utilization.

1. Introduction

Manufacturing industries often use guillotine shears to divide stock plates into rectangular pieces to make various products, such as transport vehicles, household appliances, and fabricated metal products. It is necessary to solve the two-dimensional cutting stock problem (TDCS) to improve material utilization, where the optimization objective is to minimize plate cost, observing the constraint that the number of each piece type should be larger than or equal to the demand. The TDCS solution is a cutting plan that contains a set of specified cutting patterns with given frequencies. The sequential heuristic procedure (SHP) [1–3] is often used to generate the cutting plan. It generates the next pattern using the reduced demands, renews the reduced demands by deleting the pieces included in the pattern, and repeats the pattern-generation process until the reduced demands become zero. Initially the reduced demand of each piece type is equal to the original demand.

Assume that plate $L \otimes W$ (length$\otimes$width) is used to produce a cutting plan of m piece types, where piece type i has length l_i, width w_i, value c_i, and demand d_i, $i=1,\ldots,m$. Use P to denote a cutting pattern and N to denote the set of non-negative integers. The SHP generates each next pattern from solving the following constrained two-dimensional cutting problem (CTDC), where z_i is the frequency of piece type i in P:

$$\max z = \sum_{i=1}^{m} c_i z_i;\ P \text{ is a guillotine pattern};\ z_i \le d_i \text{ and } z_i \in \mathbf{N},\ i=1,\ldots,m$$

CTDC becomes unconstrained (UTDC) if the constraint $z_i \le d_i$ is ignored. The UTDC is a special case of the CTDC, where the number of times each piece type i can appear in the pattern is naturally constrained by $\lfloor L/l_i \rfloor \times \lfloor W/w_i \rfloor$. However, UTDC problems are generally easier to solve than CTDC.

Patterns of different types have been used in the literature, such as two-staged [4–5], three-staged [6], and T-shape patterns [7]. The pattern type must be determined before solving the CTDC. Unless dictated by technological constraints, choosing a specific type of patterns takes into account conflicting factors: material utilization and cutting complexity. Generally, simple cutting patterns yield low material utilization whereas patterns of high material utilization are complex to cut. Furthermore, generating patterns with high material utilization is computationally expensive.

T-shape patterns may make good balance between material utilization and cutting complexity. Although an UTDC algorithm for T-shape patterns has been reported [7], CTDC algorithms are not available.

A heuristic for generating constrained T-shape patterns is presented in this paper, where it is assumed that all piece types have fixed orientation, and the sizes of the plate and pieces are integers. The heuristic is adequate for being combined with the SHP to solve the TDCS because of its short computation time. It is also useful for improving material utilization. The computational experiments on 58 benchmark instances show that the heuristic gets solutions better than those of the optimal two-staged patterns [5] in 54 instances.

The contents of the next sections are arranged as follows: T-shape patterns are described in the next section, together with a brief literature review. The algorithm is presented in Section 3. The computational results are given in Section 4, followed by the conclusions in the last section.

* Corresponding author.
E-mail address: ydcui@263.net (Y. Cui).

0305-0548/$ - see front matter © 2012 Published by Elsevier Ltd.
doi:10.1016/j.cor.2012.03.001

2. T-shape patterns and literature review

Fig. 1 shows three strip types that can be used to fill a pattern. A general strip contains pieces of different types. A uniform strip consists of pieces of the same width. A homogenous strip includes pieces of the same type.

Fig. 2 shows two T-shape patterns. A dividing-cut (denoted by an arrow) separates the plate into two segments, one of which is an X-segment that contains horizontal strips, and the other is a Y-segment that consists of vertical strips. The pattern in Fig. 2(a) is a TX pattern, where the dividing-cut is vertical and the two segments are arranged horizontally from left to right. The pattern in Fig. 2(b) is a TY pattern, where the dividing-cut is horizontal and the two segments are arranged vertically from top to bottom. The algorithm will be presented only for TX patterns. It can also be used for TY patterns if the lengths and widths of both the plate and piece types are swapped.

T-shape patterns have the one-cut-one-strip (OCOS) feature that each cut on the plate produces just one strip of exact width, where exact means that the strip width should be equal to the maximum width of the pieces included. After the strips in the Y-segment are cut down (see Fig. 2(a)), the plate has to be rotated by 90° to cut the strips in the X-segment. Patterns of OCOS feature are welcomed when the two-phase cutting process is used to divide the plate into pieces, where a heavy machine cuts the plate into strips at the first phase, and light machines divide the strips into pieces at the second phase. Recall that the first strip cut down from the plate can be either vertical (Fig. 2(a)) or horizontal (Fig. 2(b)). The maximum length of a cut made at the first phase is max(L,W), and that made at the second phase is $\max_i(l_i,w_i)$. The later is usually shorter than the former. Hence light machines that are more economical can be used at the second phase.

Staged patterns have been intensively discussed [6,8–9]. A k-staged pattern can be cut into pieces in k stages, where the cuts at the same stage are in the same orientation, and the cut orientations of two adjacent stages are perpendicular to each other. A zero-staged pattern contains only one piece, and a one-staged pattern includes only one strip. Both zero- and one-staged patterns are rarely used in practice because they often yield low material utilization.

A two-staged pattern [5] contains parallel strips of the same length (Fig. 3). During the cutting process, first-stage cuts separate the plate into strips, and second-stage cuts divide the strips into pieces. Two-staged patterns also have the OCOS feature. They are simpler than T-shape patterns because rotation of the plate is not necessary to divide the plate into strips. T-shape patterns generalize two-staged patterns, because a two-staged pattern can be seen as a T-shape pattern that contains only one segment. For example, the two-staged pattern in Fig. 3 can be seen as a TX-pattern that contains only an X-segment.

T-shape patterns form a subset of three-staged patterns, because they can be cut into pieces in three stages. A general three-staged pattern is shown Fig. 4, where the arrows denote the cuts at the first stage. It is not a T-shape pattern because the first cut on the plate cannot produce a strip. Staged patterns with

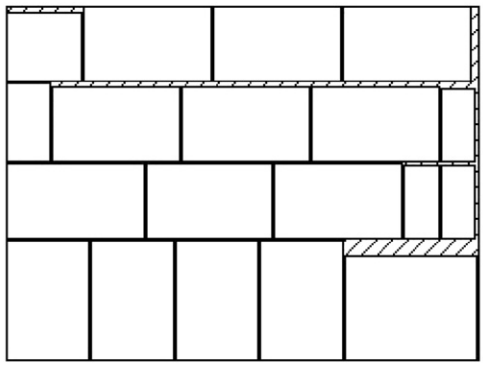
Fig. 3. Two-staged pattern.

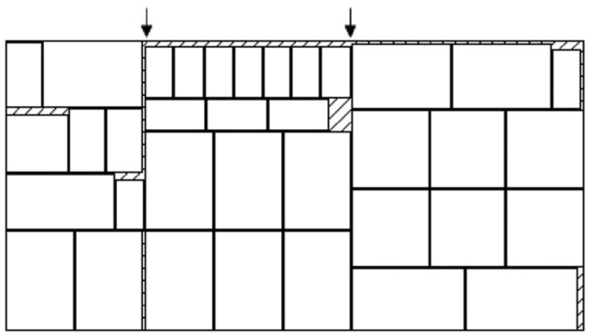
Fig. 4. Three-staged pattern.

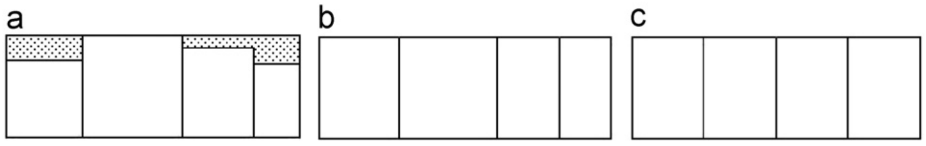

Fig. 1. Strip types. (a) General strip. (b) Uniform strip. (c) Homogenous strip.

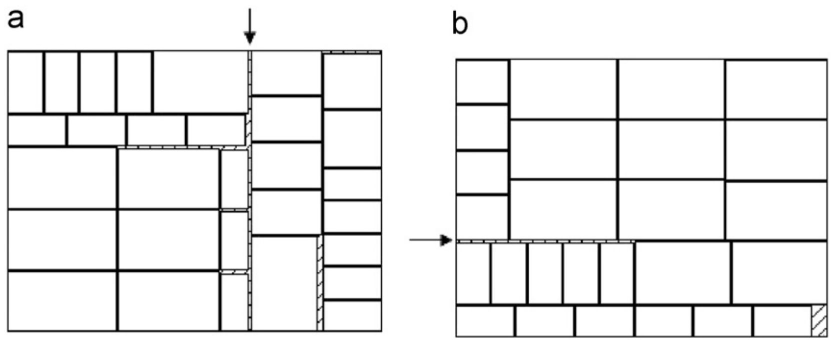

Fig. 2. T-shape patterns. (a) TX-pattern. (b) TY-pattern.

stage number larger than two do not have the OCOS feature. Although they generalize T-shape patterns, their cutting process may be more complex.

Multi-section patterns (referred to as general blocks in [10]) generalize all pattern types that have the OCOS feature. A k-section pattern contains k sections, each of which consists of strips of the same length and orientation. Fig. 5 shows a 4-section pattern, where the numbers denote the order of the cuts made on the plate. There are two vertical strips (produced by cuts 1 and 2) in Section 1, two horizontal strips (produced by cuts 3 and 4) in Section 2, one vertical strip (produced by cut 5) in Section 3, and two horizontal strips (produced by cuts 6 and 7) in Section 4. Cut 7 is necessary to make the last strip have exact width. During the cutting process of a k-section pattern, the plate must be rotated $k-1$ times to cut down the strips. Therefore, the k value can be constrained to simplify the cutting process. $k=2$ for general T-shape patterns ($k=1$ for the special case where a T-shape pattern contains only one segment), and $k=1$ for two-staged patterns.

When the number of sections increases, the material utilization improves at the expense of cutting complexity. Use δ_k to denote the percentage of improvement in pattern value when $(k-1)$-section patterns are replaced with k-section patterns. According to the law of diminishing return, it can be expected that $\delta_2 \geq \delta_3 \geq \cdots \geq \delta_K$ holds for multi-section patterns with section number not larger than K. This means that the improvement may the most significant when two-staged patterns ($k=1$) are replaced with T-shape patterns ($k=2$). Therefore, T-shape patterns may make good balance between material utilization and cutting complexity.

Although this paper focuses on T-shape patterns, it should be noted that T-shape patterns are to be used only when it is technologically feasible as there are industrial cases that prohibit their use. Take the two-phase cutting process as an example. A rolling shear can be used at the first phase to cut the plate into strips when the plate is thin. It divides the plate into strips in only one pass. T-shape patterns are not adequate for such case. Instead, two-staged patterns may be used.

UTDC algorithms for many pattern types have been reported [7,9,10–12]. They are usually fast and relatively easy to design because it is not necessary to consider the constraint on the frequencies of the piece types. The CTDC is much harder to solve because the constraint on the frequencies of the piece types must be satisfied. The bottom-up approach [13–17] has been used by several authors for generating optimal general CTDC patterns that are equivalent to staged patterns of infinite stage number. It is also used for generating optimal two-staged patterns [18],

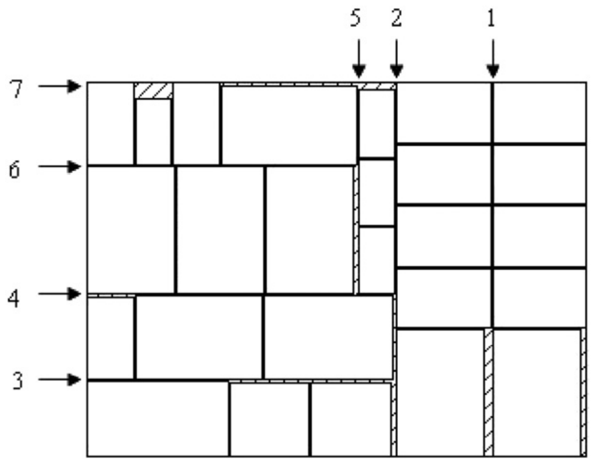

Fig. 5. Four-section pattern.

homogenous three-staged patterns [19], and homogenous T-shape patterns [20]. A recursive branch-and-bound algorithm for generating optimal constrained homogenous T-shape patterns is available in [21]. Exact CTDC algorithms [13–21] are not adequate for being combined with the SHP to solve medium- and large-scale TDCS instances because of their long computation time, whereas it may be necessary to generate thousands of patterns before the cutting plan is determined. Heuristic CTDC algorithms are useful to make the computation time affordable. Good heuristic algorithms for two-staged patterns are available in [5,22]. A fast heuristic for constrained homogenous T-shape patterns is described in [23]. Both exact and heuristic CTDC algorithms for general T-shape patterns have not been reported.

3. Algorithm

Four versions of the algorithm will be presented, where the techniques used to reduce the computation load are introduced gradually to facilitate presentation. All versions are based on two procedures: the strip-generation procedure and the layout-generation procedure. Section 3.1 describes the strip-generation procedure to generate candidate strips for a segment; Section 3.2 presents the layout-generation procedure to arrange the strips on the segment. Sections 3.3–3.6 depict various versions of the algorithm and the techniques used.

3.1. Strip-generation procedure

The strip-generation procedure generates candidate strips for a segment. Although the procedure will be described only for an X-segment, it can be used for a Y-segment by swapping the lengths and widths of both the segment and pieces.

Assume that the piece types have been arranged such that $w_1 \leq w_2 \leq \cdots \leq w_m$. For X-segment $x \otimes W$, let b_i be the upper bound on the frequency of piece type i, $i=1,\ldots,m$. $b_i \leq d_i$ holds because d_i is the maximum number of type i pieces that can be included in the pattern. The method for determining b_i will be described later. m strip types will be generated for the segment, where the jth ($j=1,\ldots,m$) strip type has width w_j, and is determined from the following bounded knapsack problem, where z_i is the frequency of pieces type i, and $f(j,x)$ is the maximum value of strip $x \otimes w_j$:

$$f(j,x) = \max \sum_{i=1}^{j} c_i z_i; \sum_{i=1}^{j} l_i z_i \leq x; z_i \leq b_i \text{ and } z_i \in \mathbf{N}, i=1,\ldots,j$$

It should be noted that some researchers [5] have used the constraint that the number of strip types to be generated should be equal to the number of different widths of the pieces. This paper does not use this constraint. The solution space of the strips may be larger if this constraint is ignored, because the strip types of the same width are generated using different sets of piece types.

This paper uses the dynamic programming algorithm [24] to generate the strips. It is sufficient to solve the following largest bounded knapsack problem to generate all strips $x \otimes w_j$ because of the all-capacity property:

$$f(m,L) = \max \sum_{i=1}^{m} c_i z_i; \sum_{i=1}^{m} l_i z_i \leq L; z_i \leq b_i \text{ and } z_i \in \mathbf{N}, i=1,\ldots,m \quad (1)$$

This means that if the upper bounds on the frequencies of the piece types do not vary, the strip types for any X-segment $x \otimes W$ ($x=1,2,\ldots,L$) can be obtained from solving problem (1) only once. The complexity is $O(L\sum_{i=1}^{m} b_i)$.

The strip-generation procedure generates only m strip types. Other strip-generation procedures may be used to generate far more strips, such as the procedure described in Section 2 of [5].

Actually the authors of this paper tested other strip-generation procedures and observed that using more strips does have positive effect on the solution quality of some segments, but has little effect on the solution quality of the T-shape pattern that is obtained from considering many dividing-cut positions.

3.2. Layout-generation procedure

Define the following notations:

$z(i,j)$	Number of pieces type i in strip $x\otimes w_j$, $i=1,\ldots,m$ and $j=1,\ldots,m$. It can be determined from the strips generated in Section 3.1.
$n(i,y)$	Number of pieces type i in the solution to X-segment $x\otimes y$, where $n(i,0)=0$, $i=1,\ldots,m$ and $y=0,\ldots,W$.

Given the m strip types for X-segment $x\otimes W$, the following dynamic programming recursion determines $F(y)$ the value of X-segment $x\otimes y$, $y=1,\ldots,W$:

$$F(y)=\max\{F(y-1),F(y-w_j)+v_j\,|\,y\geq w_j,j=1,\ldots,m\}$$

where $F(0)=0$; v_j denotes the incremental value obtained from packing strip $x\otimes w_j$ and is determined as follows:

$$v_j=\sum_{i=1}^{m}\{c_i\times\min[z(i,j),b_i-n(i,y-w_j)]\}$$

This means that surplus pieces are taken as waste. In case of a tie, the strip with smaller id is selected. Although the recursion is described for an X-segment, it can also be used for a Y-segment by using related strips (see Section 3.1) and swapping the length and width of the segment. The recursion is heuristic and greedy in nature. The layout-generation procedure is based on the recursion. Its complexity is $O(m^2W)$.

3.3. Version 1 of the algorithm

Version 1 is preliminary and easy to understand. It considers all dividing-cut positions to determine the solution. The dividing-cut at x separates the plate into two segments: X-segment $x\otimes W$ and Y-segment $(L-x)\otimes W$. One segment is referred to as the main segment and the other as the residual segment. The *constrained* (*or main*) *layout* on the main segment is determined using the original demand ($b_i=d_i$,$i=1,\ldots,m$); the *residual layout* on the residual segment is generated using the residual demand, that is, $b_i=d_i-z_i$, where z_i is the frequency of piece type i for the main segment, $i=1,\ldots,m$. Both layouts can be determined using the layout-generation procedure. Define the following notations:

$X_C(x)$	The main layout on X-segment $x\otimes W$.
$V_C(x)$	Value of layout $X_C(x)$.
$X_R(x)$	The residual layout on X-segment $x\otimes W$.
$V_R(x)$	Value of layout $X_R(x)$.
$Y_C(x)$	The main layout on Y-segment $x\otimes W$.
$U_C(x)$	Value of layout $Y_C(x)$.
$Y_R(x)$	The residual layout on Y-segment $x\otimes W$.
$U_R(x)$	Value of layout $Y_R(x)$.
$T_C(x)$	Pattern consisting of X-segment $x\otimes W$ and Y-segment $(L-x)\otimes W$.

Between the following two patterns, the one with larger value will be selected as $T_C(x)$:

$X_C(x)Y_R$	The main layout is on X-segment $x\otimes W$, and the residual layout on Y-segment $(L-x)\otimes W$. The pattern value is $V_C(x)+U_R(L-x)$.
$X_R(x)Y_C$	The residual layout is on X-segment $x\otimes W$, and the main layout on Y-segment $(L-x)\otimes W$. The pattern value is $V_R(x)+U_C(L-x)$.

Use $V(x)$ to denote the value of $T_C(x)$, $x=0,1,\ldots,L$, then

$$V(x)=\max\{V_C(x)+U_R(L-x),V_R(x)+U_C(L-x)\}$$

Let V_0 be the value of the best pattern. Box 1 shows Version 1 algorithm that is based on the description above, where *residual strips* denote those in the residual segment.

Step 1 uses the strip-generation procedure to generate horizontal strips for the main layouts on all X-segments; It is necessary to solve Problem (1) only once with $b_i=d_i$, $i=1,\ldots,m$. Similarly, Step 2 generates vertical strips of length W for the main layouts on all Y-segments. The complexity of Step 1 is $O(L\sum_{i=1}^{m}d_i)$, and that of Step 2 is $O(W\sum_{i=1}^{m}d_i)$.

Step 3 uses the layout-generation procedure to generate $Y_C(L)$ the main layout on Y-segment $L\otimes W$, and takes it as the initial best solution. Meanwhile, $Y_C(x)$ the main layout on Y-segment $x\otimes W$ ($x=1,\ldots,L-1$) is also obtained because of the all-capacity property. The complexity is $O(m^2L)$.

Steps 4 to 9 consider all dividing-cut positions one by one. For the dividing-cut at x, the strips for $x\otimes W$ have been determined in Step 1, and those for $(L-x)\otimes W$ in Step 2. Steps 5 to 7 consider pattern $X_C(x)Y_R$. Step 5 uses the layout-generation procedure to determine main layout $X_C(x)$; its complexity is $O(m^2W)$. Step 6 first generates the residual strips using the strip-generation procedure whose complexity is $O(W\sum_{i=1}^{m}d_i)$, then determines layout $Y_R(L-x)$ using the layout-generation procedure whose complexity is not larger than $O(m^2L)$. Step 7 renews the best pattern if an improvement is available. Steps 8 and 9 consider pattern $X_R(x)Y_C$ similarly. It is not necessary to generate $Y_C(L-x)$ that has been determined in Step 3. The complexity of Step 8 is not larger than $O(L\sum_{i=1}^{m}d_i)+O(m^2W)$. Considering that there are at most L dividing-cut positions, the complexity of Steps 4 to 9 can be estimated as:

$$O\{L\times[m^2W+(W\textstyle\sum_{i=1}^{m}d_i+m^2L)+(L\sum_{i=1}^{m}d_i+m^2W)]\}$$
$$=O\{L\times[m^2(L+2W)+(W+L)(\textstyle\sum_{i=1}^{m}d_i)]\}.$$

The total complexity of the algorithm is also the same as that of Steps 4 to 9, because the complexities of Steps 1 to 3 are relatively much smaller. This complexity is considered high (it depends on m^2, L, W and the sum of the demand). Some techniques are used in the following sub-sections to alleviate some of the computational burden. It should be noted that using

Box 1–Version 1 of the algorithm.

```
1 Generate horizontal strips $x\otimes w_j$ ($j=1\ldots m$) for the main
  layouts on X-segments $x\otimes W$, $x=1\ldots L$.
2 Generate vertical strips $W\otimes l_j$ ($j=1\ldots m$) for the main
  layouts on Y-segments $x\otimes W$, $x=1\ldots L$.
3 Generate Y$_C$(L) the main layout on Y-segment $L\otimes W$. Let
  $V_0=U_C(L)$.
4 For each dividing-cut position $x$ ($x$=1 to $L$)
5    Determine layout X$_C$(x).
6    Generate the vertical residual strips and determine
     layout $Y_R(L-x)$.
7    If $V_C(x)+U_R(L-x)>V_0$ then record $X_C(x)Y_R$ as the best
     pattern and le t $V_0=V_C(x)+U_R(L-x)$.
8    Generate the horizontal residual strips and determine
     layout $X_R(x)$.
9    If $V_R(x)+U_C(L-x)>V_0$ then record X$_R$(x)Y$_C$ as the best
     pattern and let $V_0=V_R(x)+U_C(L-x)$.
10 Output the best pattern.
```

these techniques may lead to slightly inferior solutions because of the heuristic nature of the algorithm.

3.4. Version 2 of the algorithm: Using pseudo upper bound and considering normal lengths

3.4.1. Using pseudo upper bound to skip some branches

The better pattern between $X_C(x)Y_R$ and $X_R(x)Y_C$ is selected as $T_C(x)$. The pseudo upper bound of $T_C(x)$ is defined as $V_C(x)+U_C(L-x)$. It is pseudo because the layout-generation procedure is heuristic. $T_C(x)$ is skipped without generating $Y_R(L-x)$ and $X_R(x)$ when $V_C(x)+U_C(L-x)\le V_0$.

The pseudo upper bound may be under-estimated because of the heuristic layout-generation procedure. Subsequently using this technique may lead to a solution inferior to that obtained from Version 1.

3.4.2. Considering only normal dividing-cut positions

The strip-generation procedure generates m strips for X-segment $x\otimes W$. The value of the jth strip $f(j,x)\ge f(j,x-1)$, $j=1,\ldots,m$, holds when $x>0$. X-Segment $x\otimes W$ is normal if there is at least one strip that makes $f(j,x)>f(j,x-1)$, $j=1,\ldots,m$. If X-segment $x\otimes W$ is normal, then it has a normal length; its dividing-cut position x is normal.

Normal lengths are usually defined as the combinations of the piece lengths [5,19]. Some of them may not have the feature described above. Therefore, the set of normal lengths used in this paper may be a subset of the normal lengths used in the literature.

The following analysis indicates that normal dividing-cut positions have the following property: assume that x is not a normal length and x_0 is the largest normal length not larger than x, $0<x_0<x$, then the value of pattern $X_C(x)Y_R$ is smaller than or equal to that of pattern $X_C(x_0)Y_R$. Let $V_1=V_C(x)+U_R(L-x)$and $V_2=V_C(x_0)+U_R(L-x_0)$. Thus $V_1-V_2=V_C(x)-V_C(x_0)+U_R(L-x)-U_R(L-x_0)$. From the strip-generation procedure and the definition of normal lengths, it is known that the candidate strips for $X_C(x)$ are the same as those for $X_C(x_0)$. This means that layout $X_C(x)$ is the same as layout $X_C(x_0)$. Consequently $V_C(x)=V_C(x_0)$. The candidate residual strips for $Y_R(L-x)$ is the same as those for $Y_R(L-x_0)$, because they are both generated using the same residual demands that are obtained from the same main layout $X_C(x)$. The layout-generation procedure guarantees that $U_R(L-x)\le U_R(L-x_0)$, where $L-x<L-x_0$. This means $V_1-V_2\le 0$, that is, the value of pattern $X_C(x)Y_R$ is smaller than or equal to that of pattern $X_C(x_0)Y_R$.

Recall that

$$V(x)=\max\{V_C(x)+U_R(L-x),V_R(x)+U_C(L-x)\}, \text{ and}$$

$$V(x_0)=\max\{V_C(x_0)+U_R(L-x_0),V_R(x_0)+U_C(L-x_0)\}.$$

Although $V_C(x_0)+U_R(L-x_0)\ge V_C(x)+U_R(L-x)$ according to the property of normal dividing-cut positions, $V_R(x_0)+U_C(L-x_0)\ge V_R(x)+U_C(L-x)$ may not hold. Considering that $L-x_0>L-x$ and x is not a normal length, the probability of $V_R(x_0)+U_C(L-x_0)\ge V_R(x)+U_C(L-x)$ is high. This indicates that considering only normal dividing-cut positions may lead to inferior solutions, but the probability is low.

Let Ω be the set of normal dividing-cut positions. Box 2 shows Version 2 of the algorithm that uses pseudo upper bound and considers only normal lengths. Compared with Version 1, Step 4 is changed to consider only normal dividing-cut positions; and Step 6 is inserted to use pseudo upper bound.

Box 2–Version 2 of the algorithm.

```
1 Generate horizontal strips $x\otimes w_j$ ($j=1\ldots m$) for the main layouts on $x\otimes W$, $x\in\Omega$.
2 Generate vertical strips $W\otimes l_j$ ($j=1\ldots m$) for the main layouts on Y-segments $x\otimes W$, $x=1\ldots L$.
3 Generate $Y_C(L)$ the main layout on Y-segment $L\otimes W$. Let $V_0=U_C(L)$.
4 For each x in $\Omega$
5     Determine layout $X_C(x)$.
6     Skip position x if $V_C(x)+U_C(L-x)\le V_0$.
7     Generate the vertical residual strips and determine layout $Y_R(L-x)$.
8     If $V_C(x)+U_R(L-x)>V_0$ then record $X_C(x)Y_R$ as the best pattern and let $V_0=V_C(x)+U_R(L-x)$.
9     Generate the horizontal residual strips and determine layout $X_R(x)$.
10    If $V_R(x)+U_C(L-x)>V_0$ then record $X_R(x)Y_C$ as the best pattern and let $V_0=V_R(x)+U_C(L-x)$.
11 Output the best pattern.
```

3.5. Version 3 of the algorithm: Using feasible loose solutions

Let $X_U(x)$ be the unconstrained layout and $V_U(x)$ be the value of X-segment $x\otimes W$, in which the layout is generated without considering the constraint on the frequencies of the piece types. According to the tightness of the constraint on the frequencies of the piece types, it is said that $X_U(x)$ is looser than $X_C(x)$, $X_C(x)$ is looser than $X_R(x)$, and $Y_C(x)$ is looser than $Y_R(x)$. $X_U(x)$ contains some strips obtained from the strip-generation procedure (Section 3.1), and is determined from the following dynamic programming recursion, where $F_U(y)$ is the maximum value of X-segment $x\otimes y$, $F_U(0)=0$, $f(j,x)$ is the value of strip $x\otimes w_j$, and $V_U(x)=F_U(W)$:

$$F_U(y)=\max\{F_U(y-1),F(y-w_j)+f(j,x)\,|\,y\ge w_j,j=1,\ldots,m\},y=1,\ldots,W$$

The complexity is $O(mW)$.

During the pattern generation process, it is not necessary to generate $X_C(x)$ if $X_U(x)$ is C-feasible, that is, the number of pieces type i included in $X_U(x)$ is not larger than d_i, $i=1,\ldots,m$. Using C-feasible $X_U(x)$ to replace $X_C(x)$ can shorten the computation time, because the complexity for generating $X_U(x)$ is much smaller than that for generating $X_C(x)$, that is, $O(mW)\ll O(m^2W)$.

For dividing-cut position x, it is known from Box 2 that $X_C(x)$ is determined before generating $X_R(x)$, and $Y_C(L-x)$ is obtained before generating $Y_R(L-x)$. Use $X_C(x)Y_C$ to denote the pattern whose left segment contains layout $X_C(x)$, and the right segment includes layout $Y_C(L-x)$. It is not necessary to generate $X_R(x)$ and $Y_R(L-x)$ if pattern $X_C(x)Y_C$ is C-feasible.

Version 3 of the algorithm uses loose solutions. Its contents are shown in Box 3. The explanations are given in the following paragraphs.

Steps 1 to 3 are the same as those of Version 2. Step 4 is added to obtain the unconstrained layout on each normal X-segment. Steps 5 to 6 consider all C-feasible $X_U(x)Y_C$ patterns to improve the solution quickly, where it is not necessary to generate any layout because $X_U(x)$ and $Y_C(L-x)$ have been determined in Steps 4 and 3, respectively.

Steps 7 to 15 are adapted from Steps 4 to 10 of Version 2, where Steps 12 to 15 are the same as Steps 7 to 10 of Version 2. Step 7 indicates that normal dividing-cut position x will be considered when $V_U(x)+U_C(L-x)>V_0$; this avoids non-promising positions. It should be noted that the dividing-cut positions related with C-feasible X_UY_C patterns are non-promising now because they have been considered in Steps 5 and 6 to improve the solution. Step 8 generates $X_C(x)$ only if $X_U(x)$ is not C-feasible.

Box 3–Version 3 of the algorithm.

```
1 Generate horizontal strips $x\otimes w_j$ ($j=1\ldots m$) for the main layouts on $x\otimes W$, $x\in\Omega$.
2 Generate vertical strips $W\otimes l_j$ ($j=1\ldots m$) for the main layouts on Y-segments $x\otimes W$, $x=1\ldots L$
3 Generate $Y_C(L)$ the main layout on Y-segment $L\otimes W$. Let $V_0=U_C(L)$.
4 Generate $X_U(x)$ for each normal dividing-cut positions x.
5 For each normal dividing-cut position with $V_U(x)+U_C(L-x)>V_0$
6   If pattern $X_U(x)Y_C$ is C-feasible then record it as the best pattern and let $V_0=V_U(x)+U_C(L-x)$.
7 For each normal dividing-cut position x with $V_U(x)+U_C(L-x)>V_0$
8   If $X_U(x)$ is C-feasible then take it as $X_C(x)$; otherwise generate $X_C(x)$.
9   Skip position x if $V_C(x)+U_C(L-x)\le V_0$.
10  If pattern $X_C(x)Y_C$ is C-feasible then record it as the best pattern and let $V_0=V_C(x)+U_C(L-x)$;
11  Else:
12     Generate the vertical residual strips and determine layout $Y_R(L-x)$.
13     If $V_C(x)+U_R(L-x)>V_0$ then record $X_C(x)Y_R$ as the best pattern and let $V_0=V_C(x)+U_R(L-x)$.
14     Generate the horizontal residual strips and determine layout $X_R(x)$.
15     If $V_R(x)+U_C(L-x)>V_0$ then record $X_R(x)Y_C$ as the best pattern and let $V_0=V_R(x)+U_C(L-x)$.
16 Output the best pattern.
```

Step 9 discards the current position if the pseudo upper bound is not larger than V_0. If pattern $X_C(x)Y_C$ is C-feasible, it replaces the best pattern (Step 10); otherwise, patterns $X_C(x)Y_R$ and $X_R(x)Y_C$ are considered to improve the solution (Steps 12 to 15).

3.6. *Version 4 of the algorithm: Considering strictly defined normal lengths*

Version 4 differs from Version 3 only in the definition of normal lengths. The normal lengths used in Version 3 are based on strip values (see Section 3.4). The normal lengths used in Version 4 are based on values of both the strips and segments. They are defined as follows: The length of X-segment $x\otimes W$ is normal if $x>0$, $V_U(x)>V_U(x-1)$, and there is at least one strip that makes $f(j,x)>f(j,x-1)$, $1\le j\le m$.

4. Computational results

As mentioned in Section 1, CTDC algorithms for general T-shape patterns have not been reported. This section contains three sub-sections. Section 4.1 compares various versions of the algorithm to show the effects of the techniques used. Section 4.2 describes the computational results of some benchmark instances to demonstrate that T-shape patterns can yield material utilization better than that of two-staged patterns. Section 4.3 compares general T-shape patterns with homogenous T-shape patterns to highlight the gain caused by the use of general strips.

The computations were performed on a personal computer (CPU 2.66 GHz and RAM 3.37 GB). Only TX patterns were generated. After the publication of the paper, the instances generated in this paper will be made available at http://www.gxnu.edu.cn/Personal/ydcui, such that they may be used by interested readers.

4.1. *Comparing different versions of the algorithm*

Six groups (Groups 1A, 1B, 2A, 2B, 3A and 3B) of random instances are used, each of which includes 10 instances. $m=50$ for Groups 1A and 1B; $m=100$ for Groups 2A and 2B; $m=150$ for Groups 3A and 3B. d_i ($i=1,\ldots,m$) is in [1,10] for Groups 1A, 2A, and 3A; it is in [1,50] for Groups 1B, 2B, and 3B. The other features of the instances are: plate length in [2000,3000] and width in [1000,1500]; for piece type i of an instance, l_i and w_i both in [50,500], and $c_i=\rho_i l_i w_i$, where ρ_i is a real number in [0.25, 0.75].

Table 1 shows the detailed computational results for each instance, where V_i denotes the pattern value obtained from Version i, $i=1,\ldots,4$; G_4 denotes the gap (in percent) between the pattern values of Versions 4 and 1, $G_4=(V_1-V_4)/V_1\times 100$. Table 2 shows the summarized computational results, where t_i denotes the average computation time (for an instance) of Version i, $i=1,\ldots,4$.

Table 1
Detailed computational results.

ID	V_1,V_2,V_3	V_4	G_4	ID	V_1,V_2,V_3	V_4	G_4
1A-1	2348411	2347562	0.0362	1B-1	2438507	2438507	0
1A-2	2709214	2706576	0.0974	1B-2	1911459	1911459	0
1A-3	1916956	1900926	0.8362	1B-3	2341637	2341637	0
1A-4	1941268	1941268	0	1B-4	2197234	2197234	0
1A-5	2417497	2411872	0.2327	1B-5	1645565	1645565	0
1A-6	2335733	2328631	0.3041	1B-6	2565656	2565656	0
1A-7	1733974	1733904	0.0040	1B-7	2520579	2520579	0
1A-8	2385263	2384477	0.0330	1B-8	2009580	2009580	0
1A-9	2587019	2573767	0.5123	1B-9	2378622	2378622	0
1A-10	1557472	1548349	0.5858	1B-10	1587621	1587621	0
Avg.			0.2642	Avg.			0
2A-1	1623066	1622186	0.0542	2B-1	2962308	2962308	0
2A-2	1654326	1654326	0	2B-2	2139455	2139455	0
2A-3	1708112	1708112	0	2B-3	2781639	2781639	0
2A-4	2077308	2077308	0	2B-4	2697175	2697175	0
2A-5	2842234	2838627	0.1269	2B-5	2038892	2038558	0.0164
2A-6	1843058	1842073	0.0534	2B-6	2117371	2115539	0.0865
2A-7	2256076	2249664	0.2842	2B-7	2012053	2012053	0
2A-8	2491653	2488885	0.1111	2B-8	2476999	2476999	0
2A-9	1960381	1960381	0	2B-9	2436651	2436651	0
2A-10	1526429	1526429	0	2B-10	1606148	1606148	0
Avg.			0.0630	Avg.			0.0103
3A-1	1657343	1648331	0.5438	3B-1	2101200	2101200	0
3A-2	1707680	1707010	0.0392	3B-2	2488753	2488753	0
3A-3	2973875	2973875	0	3B-3	2330206	2330206	0
3A-4	1473898	1473898	0	3B-4	2519139	2519139	0
3A-5	2822537	2818732	0.1348	3B-5	2330686	2330686	0
3A-6	2099802	2089657	0.4831	3B-6	1899947	1899947	0
3A-7	3028901	3028901	0	3B-7	1642788	1642788	0
3A-8	2029494	2029494	0	3B-8	2618019	2618019	0
3A-9	2561735	2553376	0.3263	3B-9	2147758	2144159	0.16757
3A-10	1603555	1603555	0	3B-10	1559129	1559129	0
Avg.			0.1527	Avg.			0.0168

Table 2
Summarized computational results.

Group	1A	2A	3A	1B	2B	3B
m	50	100	150	50	100	150
d_i	[1,10]	[1,10]	[1,10]	[1,50]	[1,50]	[1,50]
G_4	0.2642	0.0630	0.1527	0	0.0103	0.0168
t_1	7.813	15.705	27.999	8.541	21.059	32.042
t_2	1.673	4.464	6.498	0.822	3.355	5.414
t_3	1.319	2.898	2.484	0.169	0.266	0.695
t_4	0.220	0.541	0.666	0.042	0.097	0.175

The first three versions yield the same solution for each instance. Although Version 4 yields some inferior solutions, the average gap is only 0.0845 percent.

The computation time of Version 2 is much shorter than Version 1. t_2/t_1 is about 20 percent when averaged over all groups. This indicates that using pseudo upper bound and considering only normal lengths based on strip values can reduce the computation time drastically.

Recall that the average demand of Groups 2B to 3B is larger than that of Groups 2A to 2A. The computation time of Version 3 is much shorter than Version 2. The ratio of t_3/t_2 is 53 percent for groups 2A to 3A; it is 12 percent for groups 2B to 3B. This indicates that using loose solutions is efficient for reducing the computation time, especially when the average demand of the piece types is large. The reason may be that the number of loose solutions that are C-feasible increase with the average demand.

Version 4 considers normal lengths that are based on both strip values and segment values. The number of normal lengths considered is much smaller than that of Version 3. Correspondently, the computation time of Version 4 is much shorter that that of Version 3. t_4/t_3 is 22 percent when averaged over all groups. This indicates that using Version 4 can greatly reduce the computation time, although the solutions may be slightly inferior.

As mentioned in Section 1, thousands of patterns may be generated in solving the TDCS. When averaged over all groups, the computation time of Version 3 is 1.31 s, and that of Version 4 is 0.29 s. Both versions may be adequate for being used jointly with the SHP to solve the TDCS. The reason is that manufacturing industries often produce products in batches, in which case it may be not necessary to finish the working orders of the piece types instantly. It may be allowed to finish the working orders hours later.

4.2. Comparing T-shape and two-staged patterns

It is often necessary to consider material utilization, cutting complexity and computation time in selecting the pattern type to use. Compared with two-staged patterns, T-shape patterns may yield better material utilization but higher cutting complexity. Two-staged patterns may be adequate if the price of the plate is low and the cutting cost is relatively high. T-shape patterns may be adequate otherwise. This sub-section compares T-shape patterns with two-staged patterns to demonstrate the ability of T-shape patterns to improve material utilization. Two groups of benchmark instances are used.

The first group consists of the 38 small-scale benchmark instances used in [5]. Table 3 shows the computational results, where $V_{2\text{staged}}$ denotes the value of the two-staged pattern, $V_{\text{T-shape}}$ denotes that of the T-shape pattern, and G denotes the gap (in percent) between the values of the two-staged pattern and the T-shape pattern, $G=(V_{\text{T-shape}}-V_{2\text{staged}})/V_{\text{T-shape}}\times 100$. The values of the optimal two-staged patterns are obtained from Table 3 of [5]. The values of the T-shape patterns are obtained from the algorithm of this paper (Version 4).

Compared with those of two-staged patterns, the solutions of T-shape patterns are better in 27 instances, equal in 8 instances, and worse in 3 instances (Hch19, CHL7 and CU2). This indicates that using T-shape patterns are useful for improving material utilization.

Versions 1 to 3 also solved the instances. They yielded the same solution for each instance except for CW3, for which the solution value of Versions 1 to 3 is 5687, and that of Version 4 is 5468. The average computation time (of an instance) of the 4 versions are 0.014 s, 0.005 s, 0.002 s, and 0.001 s, respectively.

The second group contains the 20 benchmark instances extracted from Section 6.4 of [25]. The first ten instances are un-weighted where the piece value is equal to the piece area, and the last ten are weighted where the piece value may be not equal to the piece area. These instances were generated using the following variable ranges: Plate length and width both in [100,1000]; m in [25,60]; l_i in [0.05L, 0.4L] and w_i in [0.05W, 0.4W]; d_i in [1,9]; $c_i=l_iw_i$ for the un-weighted instances, and $c_i=\rho_il_iw_i$ for the weighted instances, where ρ_i is a real number in [0.25, 0.75]. Table 4 shows the computation results. The values of the optimal two-staged patterns are obtained from Table 1 of [22]. The values of the T-shape patterns are obtained from the algorithm of this paper (Versions 1–4), where all four versions yield the same solution value for each instance, and the average computation time (of an instance) is 0.457 s, 0.101 s, 0.012 s and 0.007 s, respectively.

It is seen that the solutions of two-staged patterns are generally inferior to those of T-shape patterns. The average gap is 0.451 percent for the first 10 instances, and 1.956 percent for the last 10 instances. This indicates that using T-shape patterns matches or improves material utilization.

4.3. Comparing general and homogenous T-shape patterns

Two sets of instances are used to compare general and homogenous T-shape patterns. The algorithm of this paper (Version 4) generated general T-shape patterns, and the exact algorithm in [21] generated homogenous T-shape patterns. It should be noted that the values of the homogenous T-shape patterns reported in this sub-section may be smaller than those in [21] because of the assumption of the oriented items.

Table 3
Results for the first group of benchmark instances.

ID	$V_{2\text{staged}}$	$V_{\text{T-shape}}$	G	ID	$V_{2\text{staged}}$	$V_{\text{T-shape}}$	G
HH	10,689	11,391	6.1628	STS4s	9,481	9,642	1.67
2	2,535	2,594	2.2745	OF1	2,713	2,713	0
3	1,720	1,740	1.1494	OF2	2,515	2,690	6.506
A1	1,820	1,820	0	W	2,623	2,721	3.602
A2	2,315	2,315	0	CHL1s	13,036	13,036	0
STS2	4,450	4,620	3.6797	CHL2s	3,162	3,236	2.287
STS4	9,409	9,700	3	A3	5,380	5,403	0.426
CHL1	8,360	8,474	1.3453	A4	5,885	6,014	2.145
CHL2	2,235	2,273	1.6718	A5	12,553	12,779	1.769
CW1	6,402	6,402	0	CHL5	363	363	0
CW2	5,354	5,354	0	CHL6	16,572	16,573	0.006
CW3	5,287	5,468	3.3102	CHL7	16,728	16,695	−0.2
Hchl2	9,630	9,735	1.0786	CU1	12,312	12,321	0.073
Hchl9	5,100	5,060	−0.791	CU2	26,100	25,934	−0.64
2s	2,430	2,604	6.682	Hchl3s	11,961	12,059	0.813
3s	2,599	2,623	0.915	Hchl4s	11,408	11,964	4.647
A1s	2,950	2,950	0	Hchl6s	60,170	60,666	0.818
A2s	3,423	3,451	0.8114	Hchl7s	62,459	62,845	0.614
STS2s	4,569	4,653	1.8053	Hchl8s	729	825	11.64

Table 4
Results for the second group of benchmark instances.

ID	$V_{2\text{staged}}$	$V_{\text{T-shape}}$	G	ID	$V_{2\text{staged}}$	$V_{\text{T-shape}}$	G
ATP30	140,168	140,461	0.2086	ATP40	63,945	66,032	3.1606
ATP31	820,260	822,417	0.2623	ATP41	202,305	206,190	1.8842
ATP32	37,880	38,015	0.3551	ATP42	32,589	33,289	2.1028
ATP33	235,580	235,580	0	ATP43	208,998	214,589	2.6054
ATP34	356,159	359,162	0.8361	ATP44	70,940	72,895	2.6819
ATP35	614,429	618,397	0.6417	ATP45	74,205	74,205	0
ATP36	129,262	130,156	0.6869	ATP46	146,402	149,658	2.1766
ATP37	384,478	385,811	0.3455	ATP47	144,317	146,789	1.6840
ATP38	259,070	260,622	0.5955	ATP48	165,428	165,640	0.1280
ATP39	266,135	267,684	0.5787	ATP49	206,965	213,667	3.1367
Avg.			0.451	Avg.			1.956

Table 5
Details for generating the piece types [21].

Group	m	l_i,w_i	d_i	Group	m	l_i,w_i	d_i
1A	25	[100,700]	[1,5]	4A	25	[50,500]	[1,5]
1B	25	[100,700]	[1,10]	4B	25	[50,500]	[1,10]
1C	25	[100,700]	[1,30]	4C	25	[50,500]	[1,30]
2A	50	[100,700]	[1,5]	5A	50	[50,500]	[1,5]
2B	50	[100,700]	[1,10]	5B	50	[50,500]	[1,10]
2C	50	[100,700]	[1,30]	5C	50	[50,500]	[1,30]
3A	200	[100,700]	[1,5]				
3B	200	[100,700]	[1,10]				
3C	200	[100,700]	[1,30]				

Table 6
Results for the instances of the first set.

Group	$V_{\text{HT-shape}}$	$V_{\text{T-shape}}$	G
1A	2353,302	2484,249	5.27
1B	2539,922	2582,253	1.64
1C	2647,864	2666,679	0.71
2A	2809,757	2921,086	3.81
2B	2669,282	2712,287	1.59
2C	2742,581	2759,587	0.62
3A	2830,123	2903,202	2.52
3B	2828,624	2871,215	1.48
3C	2836,208	2854,953	0.66
4A	2225,620	2450,691	9.18
4B	2643,745	2748,945	3.83
4C	2716,818	2735,386	0.68
5A	2435,402	2666,084	8.65
5B	2706,586	2804,997	3.51
5C	2706,278	2720,996	0.54
Avg.			2.98

Results for the instances of the second set

ID	$V_{\text{HT-shape}}$	$V_{\text{T-shape}}$	G	ID	$V_{\text{HT-shape}}$	$V_{\text{T-shape}}$	G
ATP30	138,904	140,461	1.11	ATP40	64,070	66,032	2.97
ATP31	816,042	822,417	0.78	ATP41	201,621	206,190	2.22
ATP32	37,881	38,015	0.35	ATP42	32,025	33,289	3.80
ATP33	232,839	235,580	1.16	ATP43	214,589	214,589	0
ATP34	352,613	359,162	1.82	ATP44	72,895	72,895	0
ATP35	614,132	618,397	0.69	ATP45	72,056	74,205	2.90
ATP36	126,928	130,156	2.48	ATP46	147,772	149,658	1.26
ATP37	380,964	385,811	1.26	ATP47	136,118	146,789	7.27
ATP38	257,134	260,622	1.34	ATP48	157,887	165,640	4.68
ATP39	266,211	267,684	0.55	ATP49	201,659	213,667	5.62
Avg.			1.15	Avg.			3.07

The first set includes the 15 groups of instances (20 in each group) generated in [21] using the following ranges: L in [2000,3000], W in [1000,1500]. Other details are listed in Table 5. For each instance, the value of an item type is $c_i = \rho_i l_i w_i$, where ρ_i is a real number in [0.5, 1], $i = 1,\ldots,m$. The computational results are shown in Table 6, where $V_{\text{HT-shape}}$ denotes the value of the homogenous T-shape pattern, $V_{\text{T-shape}}$ denotes that of the general T-shape pattern, and G denotes the gap (in percent) between the values of the two pattern types, $G = (V_{\text{T-shape}} - V_{\text{HT-shape}})/V_{\text{T-shape}} \times 100$. The average gap is 2.98 percent. This indicates that the material utilization of general T-shape patterns is much better than that of homogenous T-shape patterns.

The second set includes the 20 instances in Table 4. Table 6 shows the computational results. The average gap between the two pattern types is 1.15 percent for the first 10 instances, and 3.07 percent for the last 10 instances. This also indicates that the material utilization of general T-shape patterns is much better than that of homogenous T-shape patterns.

5. Conclusions

The heuristic for solving the CTDC with T-shape patterns has been described in detail. The computational results indicate that the heuristic is effective and efficient.

Although two-staged patterns are simpler than T-shape patterns, their material utilization is usually lower than that of T-shape patterns. Two-staged patterns may be selected when the plate is cheap or the cutting cost is high. T-shape patterns may be used otherwise. Therefore, T-shape patterns provide an alternative for being used jointly with the SHP to solve the TDCS. The CTDC algorithm presented in this paper can be used in designing the applications for the TDCS.

General T-shape patterns differ from the homogenous T-shape patterns only in the strip types used. The former consists of general strips and the later contains homogenous strips. The computational results indicate that using general strips to replace homogenous strips is useful to improve material utilization.

Four versions of the algorithm have been described. They are useful to facilitate understanding and demonstrate the effects of the techniques used. Although Versions 2 to 4 may yield solutions inferior to those of Version 1, they can greatly reduce computation time. The authors recommend Versions 3 and 4 for practical applications.

Future research work may include extending the algorithm to deal with the case where rotation of the piece types is allowed, and combine the algorithm with the SHP to solve the TDCS.

Acknowledgments

This research is part of Project 61063031 supported by National Natural Science Foundation of China.

References

[1] Suliman SMA. A sequential heuristic procedure for the two-dimensional cutting-stock problem. International Journal of Production Economics 2006;99:177–85.
[2] Song X, Chu CB, Nie YY, Bennell JA. An iterative sequential heuristic procedure to a real-life 1.5-dimensional cutting stock problem. European Journal of Operational Research 2006;175:1870–89.
[3] Haessler RW, Sweeney PE. Cutting stock problems and solution procedures. European Journal of Operational Research 1991;54:141–50.
[4] Belov G, Scheithauer G. A branch-and-cut-and-price algorithm for one-dimensional stock cutting and two-dimensional two-stage cutting. European Journal of Operational Research 2006;171:85–106.
[5] Hifi M, M'Hallah R. Strip generation algorithms for constrained two-dimensional two-staged cutting problems. European Journal of Operational Research 2006;172:515–27.
[6] Beasley JE. Algorithms for unconstrained two-dimensional guillotine cutting. Journal of the Operational Research Society 1985;36:297–306.
[7] Cui Y. Generating optimal T-shape cutting patterns for rectangular blanks. Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture 2004;218:857–66.
[8] Morabito R, Arenales MN. Staged and constrained two-dimensional guillotine cutting problems: an AND/OR-graph approach. European Journal of Operational Research 1996;94:548–56.
[9] Cui Y, Wang Z, Li J. Exact and heuristic algorithms for staged cutting problems. Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture 2005;219:201–8.
[10] Cui Y, Zhang X. Two-stage general block patterns for the two-dimensional cutting problem. Computers & Operations Research 2007;34:2882–93.
[11] Cintra GF, Miyazawa FK, Wakabayashi Y, Xavier EC. Algorithms for two-dimensional cutting stock and strip packing problems using dynamic programming and column generation. European Journal of Operational Research 2008;191:61–85.
[12] Fayard D, Zissimopoulos V. An approximation algorithm for solving unconstrained two-dimensional knapsack problems. European Journal of Operational Research 1995;84:618–32.
[13] Wang PY. Two algorithms for constrained two-dimensional cutting stock problems. Operations Research 1983;32:573–86.
[14] Viswanathan KV, Bagehi A. Best-first search methods for constrained two-dimensional cutting stock problems. Operations Research 1993;41:768–76.

[15] Cung V-D, Hifi M, Cun BL. Constrained two-dimensional cutting stock problems a best-first branch-and-bound algorithm. International Transactions in Operations Research 2000;7:185–210.
[16] Amaral ARS, Wright M. Efficient algorithm for the constrained two-dimensional cutting stock problem. International Transactions in Operations Research 2001;8:3–13.
[17] Vasko FJ, Bartkowski CL. Using Wang's two-dimensional cutting stock algorithm to optimally solve difficult problems. International Transactions in Operational Research 2009;16:829–38.
[18] Hifi M, M'Hallah R. An exact algorithm for constrained two-dimensional two-staged cutting problems. Operations Research 2005;53:140–50.
[19] Cui Y. Heuristic and exact algorithms for generating homogenous constrained three-staged cutting patterns. Computers & Operations Research 2008;35: 212–25.
[20] Cui Y. An exact algorithm for generating homogenous T-shape cutting patterns. Computers & Operations Research 2007;34:1107–20.
[21] Cui Y, Yang Y. A recursive branch-and-bound algorithm for constrained homogenous T-shape cutting patterns. Mathematical and Computer Modelling 2011;54:1320–33.
[22] Hifi M, M'Hallah R, Saadi T. Algorithms for the constrained two-staged two-dimensional cutting problem. INFORMS Journal on Computing 2008;20:212–21.
[23] Cui Y. Fast heuristic for constrained homogenous T-shape cutting patterns, Applied Mathematical Modelling; 2011 doi:10.1016/j.apm.2011.11.005.
[24] Kellerer H, Pferschy U, Pisinger D. Knapsack Problems. Berlin: Springer; 2004 190–191.
[25] Alvarez-Valdés R, Parajón A, Tamarit JM. A tabu search algorithm for large-scale guillotine (un)constrained two dimensional cutting problems. Computers & Operations Research 2002;29:925–47.