Computational Optimization and Applications, 33, 287–301, 2006

DOI: 10.1007/s10589-005-3067-y

Dynamic Programming Algorithms for Generating Optimal Strip Layouts

YAODONG CUI* ydcui@263.net
LING HUANG
Computer Science Department, Guangxi Normal University, Guilin, Guangxi, 541004, China

Received December 1, 2003; Revised March 17, 2005; Accepted April 7, 2005

Published online: 18 October 2005

Abstract. This paper presents dynamic programming algorithms for generating optimal strip layouts of equal blanks processed by shearing and punching. The shearing and punching process includes two stages. The sheet is cut into strips using orthogonal guillotine cuts at the first stage. The blanks are punched from the strips at the second stage. The algorithms are applicable in solving the unconstrained problem where the blank demand is unconstrained, the constrained problem where the demand is exact, the unconstrained problem with blade length constraint, and the constrained problem with blade length constraint. When the sheet length is longer than the blade length of the guillotine shear used, the dynamic programming algorithm is applied to generate optimal layouts on segments of lengths not longer than the blade length, and the knapsack algorithm is employed to find the optimal layout of the segments on the sheet. Experimental computations show that the algorithms are efficient.

Keywords: two-dimensional cutting, strip layout, guillotine cuts, CAD

1. Introduction

In the manufacturing industry, the shearing and punching process is often applied to the processing of parts made of metal sheet. First the sheet is cut into strips by a guillotine shear. Then the parts or blanks demanded are cut from the strips by a stamping press. Typically, a strip is fed into a stamping press that cuts a blank from the strip with each stroke, moves the strip forward, and cuts the next blank. Usually the blanks are of irregular type, and only identical blanks can appear in a strip. Where the strip area must be less than or equal to the sheet area, some trim loss is expected. Some customers may need blanks of different size. Those involved in mass production may want a large number of blanks of the same size. The strips of the same blank size are called identical strips. We refer to the two-dimensional guillotine-cutting problem of identical strips as the GCPIS.

As shown in figure 1, there are two types of layouts for irregular blanks processed by shearing and punching, namely the blank layout and the strip layout. A strip layout consists of one or more sections, with the strips in each section being of the same length and direction. The blank layout is determined at the tooling design stage. To improve raw material utilization, the orientation of the blank to be cut from the strip

*Author to whom correspondence should be addressed.

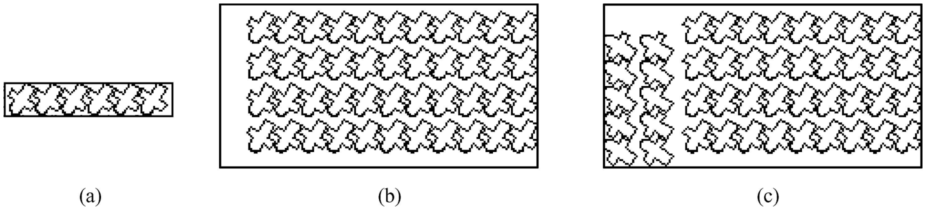


Figure 1. Blank layout and strip layout. (a) A blank layout, (b) A strip layout of 1-section, (c) A strip layout of 2-section.

and the strip width must be considered carefully at this stage. Although a large number of techniques are available for generating good blank layout [1–5], it is meaningful to develop algorithms for the GCPIS because of the two reasons below, which were originally describe in [7].

The first reason is that applying optimal strip layout may improve material usage at the tooling design stage. The object of the blank layout operation is to determine the blank orientation and strip width that maximizes the utilization of the raw material. As a blank is rotated on the strip, both the pitch between blanks changes, and the width of the strip necessarily changes. Knowing the pitch and the strip width, the number of blanks yielded by a sheet may be determined from the strip layout. The techniques for determining blank layout often assume that the strips are of the same direction and length (figure 1b). It is obvious that applying the layout with more sections and optimizing the strip layout may improve material usage.

The second reason is that applying optimal strip layout may improve material usage at the cutting stage. The sheet size is given when generating optimal blank layout at the tooling design stage. In practice, the sheet size used for real cutting may differ from that adopted at the tooling design stage. A partial sheet generated during previous cutting processes may also be used. Because of the investment required to build the tooling, and the great difficulty in changing blank orientation once the tooling is built, it is not practical to change the blank layout when the sheet size changes. In this case it is important to generate optimal strip layout to improve material usage.

A large number of techniques are available to minimize trim loss in cutting rectangular blanks [5]. There are also a lot of papers published on generating better layouts for the flame or laser cutting of irregular blanks [2, 6, 9, 13]. Several papers presented algorithms to generate optimal layouts for rectangular blanks of a single size [4, 8, 10, 14]. To our knowledge, research focused on generating optimal strip layout for the GCPIS has not been reported.

One of the authors has presented algorithms for generating optimal strip layouts [7, 15], where strips of different blank types are allowed to appear in a cutting pattern. The algorithm uses dynamic programming techniques to generate optimal T-shape patterns. A T-shape pattern includes two sections, which may be seen as the special case of the multi-section patterns applied in this paper. T-shape patterns are appropriate for the case where strips of different widths are demanded, for that the sheet may be occupied more fully through nesting of strips of various widths. For the GCPIS, only strips of the same blank type are concerned. All strips have the same width. If the number of sections is not allowed to exceed two, the material usage will be relatively low. This conclusion is confirmed by the example presented later.

The GCPIS discussed in this paper includes four types of problems: the unconstrained problem where the blank demand is unconstrained, the constrained problem where the blank demand must be met exactly, the unconstrained problem with blade length constraint, and the constrained problem with blade length constraint.

This paper presents dynamic programming algorithms to solve the four types of problems mentioned above. First the dynamic programming algorithm for the unconstrained problem is constructed. Then the algorithm is extended to solve the constrained problem. Another two algorithms are introduced to deal with the blade length constraint, one for the unconstrained problem, and the other for the constrained problem. Computations were performed to test the time efficiencies of the algorithms. The results indicate that the algorithms are very efficient.

It should be noted that this paper doesn't deal with irregular blanks and the problem can be transformed into a rectangular placement problem where, in each strip, one rectangle is different.

2. Normal multi-section patterns

2.1. Notation and functions

Some notation and functions to be used are listed in Table 1. 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 definitions in the table than in the text.

2.2. Some definitions

Definition 1. Initial step and succeeding step: As shown in figure 2, the feeding distance of the strip between two adjacent punch strokes is called a step. There may be one or more blanks in a step. A step is either an initial step or a succeeding step. The initial step is larger than or equal to the succeeding step. The strip length direction is the same as the strip feeding direction. In generating strip layouts, the steps and the strip width determined by the blank layout are given and the strip length is variable. Without loss of generality we assume one blank per step. The algorithms obtained under this assumption are applicable when there are more than one blank in a step.

Definition 2. Strips and sections: A strip includes one or more blanks. Figure 3(a) shows a strip in Y-direction. The strip direction is in the strip length direction. A section consists of one or several strips of the same length. Figure 3(b) shows a section in X-direction.

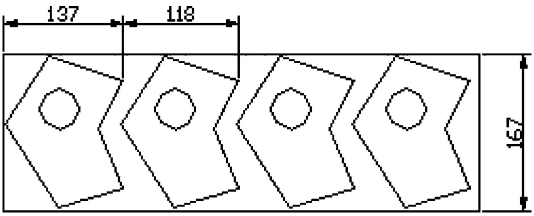
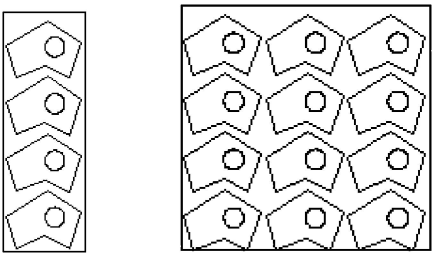


Figure 2. Initial step, succeeding step and strip width.

Table 1. Notation and functions.

l_0, l_1, w	Initial step, succeeding step, and strip width. $l_0 \geq l_1$
d	The blank demand
L, W, L_g	Sheet length, sheet width, and the maximum blade length. $W \leq L$, $W \leq L_g$
M	Number of normal lengths between 0 and L
A	The vector of normal lengths. $A = \{a_1, \ldots, a_M\}$, where $a_1 = 0$ and $a_{i+1} > a_i$ for $1 \leq i < M$
$p(x)$	Return the maximum normal length not larger than x
L_0, W_0	Normalized sheet length and width, $L_0 = p(L)$, $W_0 = p(W)$
s_n	The number of full sheets used to meet the blank demand (not including the partial sheet)
d_p	The number of blanks to be cut from the partial sheet
N	Number of the segment lengths to be considered
$F(x, y)$	The maximum number of blanks yielded by rectangle $x \times y$
$H = \{h_1, h_2, \ldots, h_N\}$	The set of segment lengths between 0 and L_g . Its elements are arranged in ascending order. In a division scheme the length of any segment must be an element of H
$V = \{v_1, v_2, \ldots, v_N\}$	v_i is the maximum number of blanks in rectangle $h_i \times W$, $v_i = F(h_i, W)$, i= 1,2, ..., N
int(x)	Return the maximum integer not larger than x
max(x, y)	Return the larger one in x and y
BinStrip(x)	Return the maximum number of blanks included in a strip of length x. BinStrip(x) = 0, if $x < l_0$; Otherwise, $\text{BinStrip}(x) = 1 + \text{int}\left((x - l_0)/l_1\right)$
SecLen(n)	Return the minimum length occupied by n blanks SecLen(n) = 0, if $n = 0$ $\text{SecLen}(n) = l_0 + (n-1)\,l_1$, if $n > 0$

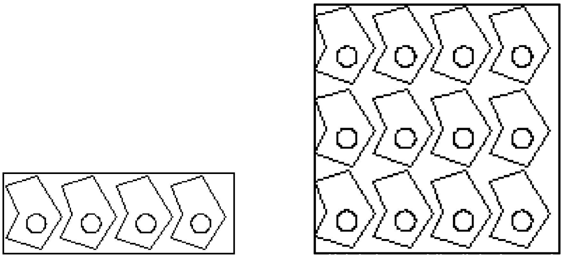


(a) (b)

Figure 3. A strip in *Y*-direction and a section in *X*-direction. (a) A strip in *Y*-direction, (b) A section in *X*-direction.

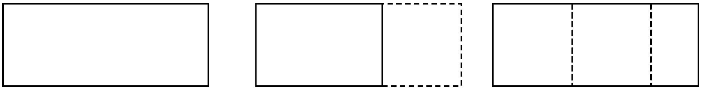
The section length is the same as the strip length. The section width is the product of the number of strips and the strip width. The section direction is the same as that of the strip width. In the cutting process it is assumed that the sheet must be divided into sections before the cutting of any section to produce strips. Figure 4 shows a strip in X-direction and a section in Y-direction.

Definition 3. Full sheet, partial sheet, and segment: Figure 5(a) shows a full sheet, which is an original stock sheet. As shown in figure 5(b), if a full sheet is divided into



(a) (b)

Figure 4. A strip in X-direction and a section in Y-direction. (a) A strip in X-direction, (b) A section in Y-direction.



(a) (b) (c)

Figure 5. Full sheet, partial sheet, and segment. (a) A full sheet, (b) A partial sheet, (c) Three segments.

two rectangles by a cut line in Y-direction, the left one is called a partial sheet. When the length of a full sheet is longer than the blade length, the sheet must be divided into several segments, with the length of each segment being not longer than the blade length. The sheet in figure 5(c) is divided into three segments.

Definition 4. Unconstrained and constrained problems: A GCPIS is either unconstrained or constrained. The former has no constraint on the blank demand, the objective being to include as many blanks as possible in a layout. The latter has the constraint that the blank demand must be exact. Generally some blanks must be arranged in a partial sheet. Assume that L is the sheet length, W is the sheet width, d is the blank demand, n^* is the maximum number of blanks yielded by a full sheet, s_n is the number of full sheets used to meet the blank demand (not including the partial sheet). The number of blanks to be cut from the partial sheet is d_p, which may be determined as follows:

$$n^* = F(L, W), \quad s_n = \text{int}(d/n^*), \quad d_p = d - n^* s_n \tag{1}$$

Here function $F(x, y)$ returns the maximum number of blanks included in rectangle $x \times y$.

We give the definition of the normal lengths below, which will be used to develop the algorithms.

Definition 5. Assume that l_0, l_1, and w are the initial step, succeeding step, and the width of the strip. x is a normal length, if

$$x = iw + \text{SecLen}(j), \quad i, j \text{ non-negative integers}, 0 \leq x \leq L$$

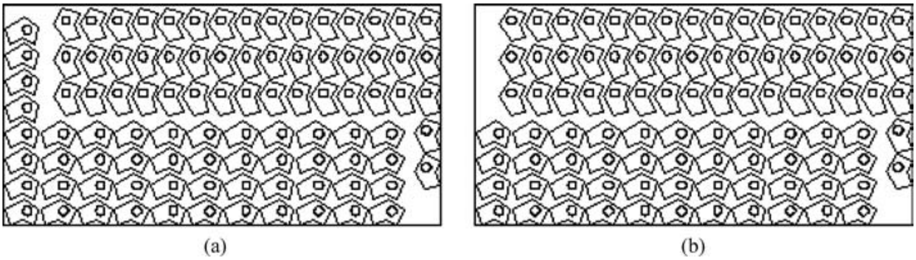


Figure 6. Normal multi-section patterns. (a) Section 1 is an X-section, (b) Section 1 is a Y-section.

SecLen(j) is the minimum length occupied by j blanks:

$$\text{SecLen}(j) = 0, \quad \text{if } j = 0; \quad \text{SecLen}(j) = l_0 + (j-1)l_1, \quad \text{if } j > 0$$

Namely, normal lengths are the combinations of strip width and the length occupied by integral steps. Below we denote the vector of normal lengths as $A = \{a_1, \ldots, a_M\}$, where

$$a_1 = 0,\ a_{i+1} > a_i \quad \text{for } 1 \le i < M.$$

2.3. *Normal multi-section patterns*

To standardize cutting patterns and develop algorithms, we introduce some definitions and theorems below.

Definition 6. Normal multi-section patterns: Figure 6 shows two normal multi-section layouts. Number the sections according to their cutting order. Thus all odd sections are in the same direction. All even sections also have a common direction perpendicular to that of the odd sections. Section 1 may be an X-section (in X direction, as shown in figure 6(a)) or a Y-section (figure 6(b)). A normal multi-section pattern is defined as a pattern with all X-sections being bottom-left justified and all Y-sections being top-right justified. A normal multi-section layout can be completely determined, if all section widths are known. As shown in figure 6(a), the length of the first section is equal to the sheet width. The length of the second section is equal to the difference between the sheet length and the width of the first section.

The following theorems hold for multi-section patterns:

Theorem 1. *Any multi-section layout can be transformed to a normal multi-section layout, without decreasing the number of blanks or increasing the number of sections in the layout.*

The proof is similar to that given in [8]. From this theorem, the optimal pattern for the GCPIS can be found by searching among normal multi-section patterns. From here on, only normal multi-section patterns will be discussed.

Theorem 2. *Let x_0 be the maximum normal length not larger than x and y_0 be the maximum normal length not larger than y. Then $F(x_0, y_0) = F(x, y)$.*

Proof: Suppose that the layout in rectangle $x \times y$ is optimal (figure 6). Push the right side of the rectangle and all Y-sections towards the left until any of the following events occurs. (1) At least one Y-section touches the left side of the rectangle. This is possible when the first section is a Y-section figure 6(b). (2) Y-sections get in touch with X-sections and no sections overlap. (3) The right side touches with an X-section. The blank number in the layout does not decrease and the length of the rectangle may be shortened to x_1 through the above pushing action. If any of the three events occurs, x_1 is a normal length from Definition 5. $x_1 \leq x_0 \leq x$ holds because that x_0 is the maximum normal length not larger than x. Since $F(x_1, y) = F(x, y)$ and $x_1 \leq x_0 \leq x$, $F(x_0, y) = F(x, y)$. This conclusion also holds for the rectangle width, so $F(x_0, y_0) = F(x, y)$.

From Theorem 2, segment $x \times W$ may be trimmed to segment $x_0 \times W$, without decreasing the number of blanks in it.

Definition 7. Optimal patterns and the simplest optimal pattern: An optimal pattern yields the maximum number of blanks among all patterns. If an optimal pattern is of the minimum number of sections among all optimal patterns, it is referred to as the simplest optimal pattern.

3. The algorithm for the unconstrained problem

3.1. The scheme of the algorithm

The normal multi-section layout may be constructed by building strips in X or Y-direction. It gives the recursion of dynamic programming:

$$F(x, y) = \max\{F(x, y - w) + \text{BinStrip}(x), F(x - w, y) + \text{BinStrip}(y)\} \qquad (2)$$

Formula (2) means that two paths may lead to the pattern on rectangle $x \times y$:

1. As shown in figure 7(a), lay an X-direction strip along the upper side of rectangle $x \times (y - w)$. The strip is of length x and includes BinStrip (x) blanks. The new rectangle $x \times y$ contains $F(x, y - w)$ + BinStrip (x) blanks.
2. As shown in figure 7(b), lay a Y-direction strip along the right side of rectangle $(x - w) \times y$. The strip is of length y and includes BinStrip (y) blanks. The new rectangle $x \times y$ contains $F(x - w, y)$ + BinStrip (y) blanks.

Although applying formula (2) directly can find the optimal pattern, it may not be the simplest optimal pattern. The following procedure may be applied to obtain the simplest optimal pattern. Let

$$n_x = F(x, y - w) + \text{BinStrip}(x), n_y = F(x - w, y) + \text{BinStrip}(y),$$

and $G(x, y)$ be the number of sections of the optimal pattern in rectangle $x \times y$. Use $Q(x, y)$ to record the path leads to $F(x, y)$. Let $F(x, y) = \max(n_x, n_y)$. Determine $Q(x, y)$ and $G(x, y)$ as follows:

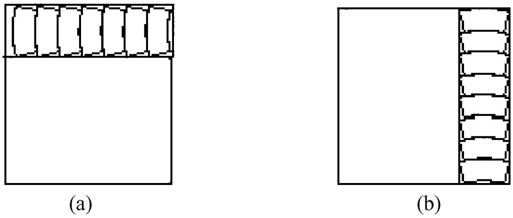


Figure 7. Two paths lead to rectangle $x \times y$. (a) From $x \times (y - w)$, (b) From $(x - w) \times y$..

(1) When $n_x > n_y$, the current strip is in X-direction, let $Q(x, y) = 1$. If $Q(x, y - w) = 2$, the last strip in rectangle $x \times (y - w)$ is in Y-direction, let $G(x, y) = G(x, y - w) + 1$; otherwise, let $G(x, y) = G(x, y - w)$.
(2) When $n_x < n_y$, the current strip is in Y-direction, let $Q(x, y) = 2$. If $Q(x - w, y) = 1$, the last strip in rectangle $(x - w) \times y$ is in X-direction, let $G(x, y) = G(x - w, y) + 1$; otherwise, let $G(x, y) = G(x - w, y)$.
(3) When $n_x = n_y$, the direction of the current strip does not affect the number of blanks in rectangle $x \times y$. If the current strip is in X-direction, determine g_x, the number of sections in rectangle $x \times y$ as follows:

$$g_x = G(x, y - w) + 1, \text{ if } Q(x, y - w) = 2; \text{ otherwise } g_x = G(x, y - w)$$

If the current strip is in Y-direction, determine g_y, the number of sections in rectangle $x \times y$ as follows:

$$g_y = G(x - w, y) + 1, \text{ if } Q(x - w, y) = 1; \text{ otherwise } g_y = G(x - w, y)$$

Let $G(x, y) = \min(g_x, g_y)$. If $g_x < g_y$, the current strip should be in X-direction, let $Q(x, y) = 1$. If $g_x > g_y$, the current strip should be in Y-direction, let $Q(x, y) = 2$. If $g_x = g_y$, the number of sections in rectangle $x \times y$ does not depend on the direction of the current strip, let $Q(x, y) = 3$.

3.2. *The algorithm for the unconstrained problem*

Assume that $p(x)$ is the maximum normal length not larger than x. L_0 is the normalized sheet length, $L_0 = p(L) = a_M$. W_0 is the normalized sheet width, $W_0 = p(W) = a_K$. The set of normal lengths $A = \{a_1, \ldots, a_M\}$ has been obtained from Definition 5. According to Theorem 2 and the scheme described above, the algorithm is as follows (Algorithm A):

Step 1. Let $F(x, y) = G(x, y) = Q(x, y) = 0$ for $x = a_1, \ldots, a_M$, $y = a_1, \ldots, a_K$. Let $i = 0$.
Step 2. Let $i = i + 1$. Go to Step 9 if $i > M$. Let $x = a_i$ and $x_0 = p[\max(0, x - w)]$. Let $j = 0$.
Step 3. Let $j = j + 1$. Go to Step 2 if $j > K$. Let $y = a_j$ and $y_0 = p[\max(0, y - w)]$.

Step 4. Let $n_x = n_y = 0$. If $y \geq w$, let $n_x = F(x, y_0) + \text{BinStrip}(x)$. If $x \geq w$, let $n_y = F(x_0, y) + \text{BinStrip}(y)$. Let $F(x, y) = \max(n_x, n_y)$. Go to Step 3 if $F(x, y) = 0$. Go to Step 5 if $n_x > n_y$. Go to Step 6 if $n_x < n_y$. Go to Step 7 if $n_x = n_y$.
Step 5. $n_x > n_y$. Let $Q(x, y) = 1$. Let $G(x, y) = \max[G(x, y_0), 1]$. Let $G(x, y) = G(x, y) + 1$ if $Q(x, y_0) = 2$. Go to Step 3.
Step 6. $n_x < n_y$. Let $Q(x, y) = 2$. Let $G(x, y) = \max[G(x_0, y), 1]$. Let $G(x, y) = G(x, y) + 1$ if $Q(x_0, y) = 1$. Go to Step 3.
Step 7. $n_x = n_y$. Let $g_x = G(x, y_0)$ and $g_y = G(x_0, y)$. Let $g_x = g_x + 1$ if $Q(x, y_0) = 2$. Let $g_y = g_y + 1$ if $Q(x_0, y) = 1$. Let $g_x = \max(g_x, 1)$ and $g_y = \max(g_y, 1)$.
Step 8. Let $G(x, y) = \min(g_x, g_y)$. Let $Q(x, y) = 1$ if $g_x < g_y$, let $Q(x, y) = 2$ if $g_x > g_y$, otherwise let $Q(x, y) = 3$. Go to Step 3.
Step 9. The maximum number of blanks is $F(L_0, W_0)$. Perform section-back-tracking to obtain the section layout of the optimal pattern.

The function of the section-back-tracking is to return the direction and number of strips of each section. The steps of the section-back-tracking are as follows:

Step 1. Let $x = L_0$ and $y = W_0$. Let m_i be the number of strips and q_i be the direction of the i-th section. Let $i = 1$ and $m_1 = 0$. Let $q_1 = 2$ if $Q(x, y) = 2$, other wise let $q_1 = 1$.
Step 2. Let $m_i = m_i + 1$ if $Q(x, y) = q_i$ or $Q(x, y) = 3$. Otherwise let $i = i+1$, $m_i = 1$, and $q_i = Q(x, y)$.
Step 3. Let $y = p(y - w)$ if $q_i = 1$, otherwise let $x = p(x - w)$.
Step 4. Go to Step 5 If $F(x, y) = 0$. Otherwise go to Step 2.
Step 5. The number of strips and the direction of each section have been found. Output the optimal pattern.

4. The other algorithms

4.1. The algorithm for the constrained problem

The blank demand must be met exactly for the constrained problem. The algorithm is as follows (Algorithm B):

Step 1. Determine $F(L_0, W_0)$ by Algorithm A.
Step 2. Let $n^* = F(L_0, W_0)$ in Eq. (1) and determine d_p, the number of blanks to be cut from the partial sheet. Go to Step 4 if $d_p = 0$.
Step 3. Find n, $2 \leq n \leq M$, so that $F(a_{n-1}, W_0) < d_p$ and $F(a_n, W_0) \geq d_p$. The length of the partial sheet is a_n.
Step 4. Obtain the optimal pattern through section-back-tracking.

Usually two runs of the section-back-tracking are needed. The first finds the section layout on the full sheet, and the second returns the section layout on the partial sheet.

4.2. The algorithm for the unconstrained problem with blade length constraint

In practice the length of the sheet in stock may be longer than the blade length of the guillotine shear used. In a factory that produces airplanes, the length of the aluminum alloy sheets may be between 5–7 m and the blade length is within 2 m. In the cutting process the sheet must be divided into several segments before any strip is cut. All segment lengths must be shorter than or equal to the blade length. This problem is referred to as the blade length problem.

Assume that L_g is the blade length, $L_g < L$ and $L_g \geq W$. Let $H = \{h_1, h_2, \dots, h_N\}$ be the set of segment lengths considered, $0 \leq h_i \leq L_g$, $i = 1, 2, \dots, N$. From Theorem 2 we know that a segment length can be trimmed to the maximum normal length not larger than itself, without decreasing the number of blanks in the segment. Therefore, any element of H may be set to one of the normal lengths between 0 and L_g. The following model should be solved to find y_i ($i = 1, 2, \dots, N$), the number of segments of length h_i in the optimal pattern:

$$\text{Max} \quad z = \sum_{i=1}^{N} v_i y_i$$
$$\text{s.t.} \quad \sum_{i=1}^{N} h_i y_i \leq L, \quad y_i \text{ is a nonnegative integer}, \quad i = 1, 2, \dots, N \tag{3}$$

Where v_i is the maximum number of blanks yielded by segment $h_i \times W$. Model (3) is a knapsack problem, the related knapsack functions are:

$$f(x) = \max \left\{ \sum_{i=1}^{N} v_i y_i;\ y_i \text{ nonnegative integers, and } \sum_{i=1}^{N} h_i y_i \leq x \right\} \tag{4}$$

For any given x, $q(x)$ is to be used to determine how $f(x)$ is achieved. The partition of x achieving $f(x)$ is given by $x_1, x_2, \dots, x_{\mathrm{r}} = x$, with $x_{j-1} = x_j - h_k$ and $k = q(x_j)$ for $j \geq 2$. So we can obtain the segment lengths in the optimal scheme by back tracking in $q(x)$. Furthermore, the number of blanks yielded by the optimal pattern is $n^* = f(L)$. The reader may refer to reference [3, 11] to find the method to solve model (4).

The algorithm for the optimal division of the long sheet may be simply described as follows (Algorithm C):

Step 1. Find N so that $a_N = p(L_g)$. Let $h_i = a_i$ for $1 \leq i \leq N$.
Step 2. Perform algorithm A only once to find $v_i = F(h_i, W_0)$, $i = 1, \dots, N$.
Step 3. Solve model (4) to obtain $f(x)$ and $q(x)$, $x = a_1, \dots, a_M$.
Step 4. Find the segment layout of the sheet by segment-back-tracking. Find the section layout of each segment appearing in the sheet by section-back-tracking. Output the optimal pattern.

The process of section-back-tracking is the same as that of algorithm A. To find the segment layout on a sheet of length L_s, the steps are as follows:

Step 1. Let $x = p(L_s)$. Let $y_i = 0$, $i = 1, 2, \dots, N$.
Step 2. Go to Step 4 if $f(x) = 0$.
Step 4. Let $i = q(x)$, $y_i = y_i + 1$, and $x = p(x - h_i)$. Go to Step 2.
Step 5. The number of each segment length has been found. Output the segment layout.

4.3. The algorithm for the constrained problem with blade length constraint

The algorithm is as follows (Algorithm D):

Step 1. Perform Algorithm C to solve the unconstrained problem with blade length constraint.
Step 2. From Eq. (1), let $n^* = f(p(L)) = f(a_M)$, $s_n = \text{int}(d/n^*)$, and $d_p = d - n^* s_n$. Go to Step 4 if $d_p = 0$.
Step 3. Find n, $1 < n \leq M$, so that $f(a_{n-1}) < d_p$ and $f(a_n) \geq d_p$. The length of the partial sheet is a_n.
Step 4. Find the segment layouts of the full sheet and the partial sheet through segment-back-tracking. Find the section layout of each segment appearing in the optimal pattern by section-back-tracking. Output the optimal pattern.

5. The experimental results

5.1. Test of the algorithms

The computations were performed on a computer with clock rate 1.9 GHz and inner memory 256 MB. 300 test problems were generated randomly according to the variable ranges below:

Item	Initial step and strip width	Sheet length	Sheet width	Blank demand
Range	100–300	3000–5000	1000–1500	1000–20000

The step and width of the strip, and the length and width of the sheet are all restricted to integers. In all problems generated, the succeeding step was determined as $l_1 = \text{int}(cl_0)$, where c is a random value between 0.6 and 1.0. The blank demands were only used for the constrained problems. They are not necessary for the unconstrained problems. The tests below are all on the above 300 test problems.

We use material usage to measure the quality of a pattern, which is determined by the following equation for the unconstrained problem:

$$u = [nl_1 w/(L \times W)] \times 100(\%)$$

Here u is the material usage, and n is the number of blanks in the pattern. For the constrained problem the material usage u is determined as follows:

$$u = \{dl_1 w/[(s_n L + L_f) \times W]\} \times 100(\%)$$

Here L_f is the partial sheet length.

(1) The unconstrained and constrained problems without blade length constraint

Algorithms A and B were used to solve the unconstrained and constrained problems respectively. For the unconstrained problems, the average computation time of one problem is 0.012 seconds, and the average material usage is 93.59%. The average computation time is also 0.012 seconds, and the material usage is 93.57% for the constrained problems.

20 test problems and the solutions given by algorithm A are listed in Table 2, where n_b is the number of blanks in the optimal pattern, n_s is the number of sections. These problems may be useful for other practitioners to compare against the approach of this paper.

(2) The unconstrained and constrained problems with blade length constraint

The blade length was set to 2000 for all problems. Algorithms C and D were applied to solve the unconstrained and constrained problems respectively. The average computation time of one problem is the same for the two algorithms, which is 0.007 seconds. The average material usage is 92.88% for the unconstrained problems, and 92.86% for the constrained problems.

5.2. *Effect of the blade length on material usage*

To analyze the effect of the blade length on material usage, Algorithm C and D were applied to the test problems. All problems were solved for each blade length. The computational results are as follows:
Here u_C is the average material usage for Algorithm C, and u_D is that for Algorithm D. The two algorithms have the same average computation time (when rounded off to 3 decimal places), which is denoted as t. The material usage increases with the blade length.

L_g	1500	2000	2500	3000	3500	4000	4500	≥5000
u_C (%)	92.49	92.88	93.12	93.24	93.36	93.46	93.56	93.59
u_D (%)	92.48	92.86	93.11	93.22	93.34	93.44	93.54	93.57
t	0.007	0.007	0.009	0.010	0.011	0.012	0.012	0.012

5.3. *Potential of the multi-section patterns in saving material*

1-section patterns are often used in industry. If only 1-section patterns are allowed. The average material usage is 91.15% for the unconstrained problems without blade length constraint. From Section 5.1 we know that the average material usage is 93.59% for multi-section patterns, which is 2.44% above that of the 1-section patterns.

Applying multi-section patterns can improve material usage. The improvement is stronger for smaller sheet sizes. For the strip data of the 300 test problems, the computation results of algorithm A for different sheet sizes are listed below, where u_1 is the average material usage of the 1-section patterns, and u_m is that of the multi-section patterns.

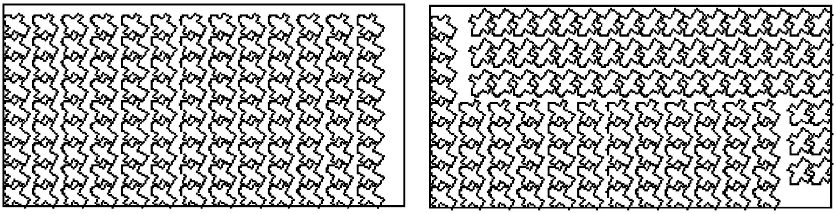
Sheet size	1500 × 750	2000 × 1000	2500 × 1250	3000 × 1500	4000 × 2000
u_1 (%)	83.19	87.35	89.99	91.92	93.49
u_m (%)	86.78	90.49	92.72	94.31	95.80
$(u_m - u_1)$ (%)	3.59	3.14	2.73	2.39	2.31

Table 2. Some test problems and their solutions.

No	w	l_0	l_1	L	W	n_b	n_s
1	100	114	72	3019	1009	410	1
2	268	115	72	3201	1139	177	4
3	229	225	159	4246	1389	156	1
4	130	114	81	4395	1129	447	4
5	246	218	178	4941	1464	160	1
6	235	204	194	4810	1465	149	3
7	160	209	195	4227	1176	154	4
8	100	250	199	3247	1452	224	1
9	272	223	145	4323	1009	93	2
10	274	106	84	3846	1080	174	2
11	125	286	277	4562	1183	150	3
12	274	264	246	4951	1049	74	2
13	193	261	244	3979	1323	104	2
14	212	114	95	3889	1450	272	3
15	115	214	180	3449	1265	206	3
16	161	284	228	3096	1363	108	2
17	155	243	163	3587	1294	168	1
18	297	118	108	3567	1007	108	1
19	241	152	117	3015	1073	100	1
20	169	225	206	4249	1336	156	3

5.4. Solution to an example

For the strip shown in figure 1(a), $l_0 = 160$, $l_1 = 132$ and $w = 183$ respectively. Assume that the sheet size is 2500×1250. Figure 8(a) shows the optimal 1-section pattern that contains 117 blanks. Figure 8(b) shows the optimal multi-section pattern generated by algorithm A, which contains 121 blanks in four sections. The algorithm of this paper guarantees that the optimal pattern is the simplest one. If the maximum number of sections is not allowed to exceed two or three, the number of blanks in the pattern must



(a) (b)

Figure 8. Cutting patterns of the example. (a) The 1-section pattern (117 blanks), (b) The multi-section pattern (121 blanks).

be less than 121. That is to say, the material usage may be relatively low if there exists constraint on the number of sections.

6. Conclusions

The algorithms for generating optimal cutting patterns of identical strips may be applied both at the tooling design stage, and at the cutting stock stage, to improve material usage.

Four algorithms based on dynamic programming are presented, which are cable of dealing with four types of problems often encountered in practice: the unconstrained problem, the constrained problem, the unconstrained problem with blade length constraint, the constrained problem with blade length constraint. A dynamic programming based algorithm was used to obtain the simplest optimal patterns. These patterns simplify the cutting process, which is an important consideration in industry.

Another advantage of dynamic programming algorithms is the easiness to design and understand, which is welcomed by practitioners.

Practical computational complexity of the dynamic programming algorithms is acceptable, however from the point of view of the theory of computational complexity, these algorithms have non-polynomial character. A linear algorithm exists for equal rectangular blanks [4]. Does it exist for identical strips? The existence of a polynomial algorithm for identical strips is a topic left for future research.

Acknowledgments

This paper is part of the project supported by Guangxi Science Foundation (Grant 0236017). The authors wish to express their appreciation to the supporter. The authors are also grateful to the anonymous referees for their helpful comments, which improved an early version of this paper.

References

1. P.K. Agrawal, “Minimizing trim loss in cutting rectangular blanks of a single size form a rectangular sheet using orthogonal guillotine cuts,” European Journal of Operational Research, vol. 64, no. 3, pp. 410–422, 1993.
2. S. Anand, C. Mccord, and R. Sharma et al., “An integrated machine vision based system for solving the nonconvex cutting stock problem using genetic algorithms,” Journal of Manufacturing Systems, vol. 18, no. 6, pp. 396–415, 1999.
3. R. Andonov, V. Poirrez, and S. Rajopadhye, “Unbounded knapsack problem: Dynamic programming revisited,” European Journal of Operational Research, vol. 123, pp. 394–407, 2000.
4. M.Z. Arslanov, “Continued fractions in optimal cutting of a rectangular sheet into equal small rectangles,” European Journal of Operational Research, vol. 125, pp. 239–248, 2000.
5. C.H. Cheng, B.R. Feiring, and T.C.E. Cheng, “Cutting stock problem—a survey,” International Journal of Production Economics, vol. 36, no. 3, pp. 291–305, 1994.
6. S.K. Cheng and K.P. Rao, “Large-scale nesting of irregular patterns using compact neighborhood algorithm,” Journal of Materials Processing Technology, vol. 103, pp. 135–140, 2000.
7. Y. Cui, “A cutting stock problem and its solution in the manufacturing industry of large electric generators,” Computers & Operations Research, vol. 32, no. 7, pp. 1709–1721, 2005.
8. Y. Cui and R. Zhou, “Generating optimal cutting patterns for rectangular blanks of a single size,” Journal of the Operational Research Society, vol. 53, pp. 1338–1346, 2002.

9. A.M. Gomes and J.F. Oliveira, “A 2-exchange heuristic for nesting problems,” European Journal of Operational Research, vol. 141, pp. 359–370, 2002.
10. N. Lindecrantz, “Method for optimum cutting of rectangular sheets,” Nordisk Tidskr. Informationsbehandling vol. I, pp. 30–35, 1964.
11. S. Martello and P. Toth, Knapsack problems: Algorithms and Computer Implementations. Wiley: New York, 1990.
12. T.J. Nye, “Optimal nesting of irregular convex blanks in strips via an exact algorithm,” International Journal of Machine Tools & Manufacture, vol. 41, pp. 991–1002, 2001.
13. A. Ramesh and N. Ramesh, “A generic approach for nesting of 2-D parts in 2-D sheets using genetic and heuristic,” Computer-Aided Design, vol. 33, pp. 879–891, 2001.
14. A.G. Tarnowski, J. Terno, and G. Scheithauer, “A polynomial time algorithm for the guillotine pallet-loading problem,” INFOR, vol. 32, pp. 275–287, 1994.
15. Y. Cui, “Generating optimal T-shape cutting patterns for circular blanks,” Computers & Operations Research, vol. 32, no. 1, pp. 143–152, 2005.