ELSEVIER

Int. J. Production Economics 74 (2001) 293–301

international journal of
production
economics

www.elsevier.com/locate/dsw

Pattern generating procedure for the cutting stock problem

Saad M.A. Suliman*

Department of Mechanical Engineering, University of Bahrain, P.O. Box. 32038, Isa Town, Bahrain

Abstract

One of the factors that add to the complexity of the cutting stock problem is the large number of the cutting patterns that may be encountered. When the cutting stock problem is expressed as an integer-programming problem, the large number of cutting patterns involved generally makes computation infeasible. However, if the linear programming formulation of the cutting stock problem is free of integer variables, then the effect of the number of cutting patterns will be mitigated. An auxiliary problem arises from the formulation where the columns of the linear programming constraint matrix need to be determined. In this work, a simple pattern generating procedure is developed for solving the auxiliary problem. It is based on an ad hoc solution method described in literature for the knapsack problem. A search tree is used to develop the pattern generation method. Examples are given to illustrate the procedure and its applications. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Cutting stock problem; Pattern generation; Column generation

1. Introduction

The cutting stock problem (CSP) is a business problem that arises in many industries such as textile, leather, paper, wood, glass and sheet metal. In all these industries, cutting of shapes from a piece of stock material is a recurring activity that requires proper scheduling so that the overall material needed by all cuts is minimized, i.e. minimum trim loss.

Different variants of CSP are available. An important variant of the CSP is the one-dimensional CSP. A collection of one-dimensional stock material objects (e.g. wooden rods, paper reels, iron slabs, etc) of a given length is divided into smaller pieces of desired lengths in order to satisfy specific customer demands. The objective of the CSP is to minimize the total amount of stock material or, equivalently, to minimize the trim loss.

Another variant of CSP is the two-dimensional CSP. In this variant, a set of order pieces is cut from a large supply of rectangular stock sheets of fixed size in a way that minimizes the total cost, such as the total number of stock sheets used. The two-dimensional CSP can be further classified into cutting of regular or irregular shapes, and the regular shapes can take guillotine or non-guillotine cut style [1].

In general, the CSP with all its extensions and variants has been classified as NP-hard [2]. Many researchers have applied linear programming (LP) models to CSPs [3–7]. The CSP is essentially an integer programming problem; however, a two-stage approach involving an LP relaxation of the CSP at the first stage followed by a rounding-up procedure at the second stage can be applied for many variants of the CSP [8]. This approach is frequently used for solving CSPs by applying the

* Tel.: +973-688314; fax: +973-684844.
E-mail address: ssuliman@eng.uob.bh (S.M.A. Suliman).

PII: S0925-5273(01)00134-7

column generation method of Gilmore and Gomory [9] and an appropriate rounding of the solution of the continuous relaxation problem [10].

An auxiliary problem arises from the LP formulation where the columns of the LP constraint matrix need to be determined. The columns of the LP constraint matrix represent all the cut patterns (i.e. the different ways of cutting the material) that can be produced from the available stock material. Columns may be generated in two ways: in advance or on-line. Advance column generation is used when all of the feasible patterns are being generated for small to medium problems. It is also used only when a representative subset is being generated for a large problem. The on-line pattern generation is used for solving large integer problems by using a column generation technique similar to that of the classic one-dimensional CSP [8].

For the one-dimensional CSP, Gilmore and Gomory [9,11] used an impressive column generation technique built into the frame of the simplex method. The next column in each simplex pivot step is generated when needed by solving the associated auxiliary problem (knapsack problem) using the shadow prices as the coefficients of the objective function. In the sequential heuristic procedure adopted by Haessler [12,13], cutting patterns are generated and used sequentially until all the requirements are met. The procedure limits the number of pattern changes by adding restrictions to the cutting patterns such that they have certain characteristics.

Christofides and Whitlock [14] designed for their n-stage solution approach of the constrained CSP an enumerative procedure to generate the cutting patterns (columns) without any duplication due to symmetry or cut ordering. Goulimis [15] approach to the one-dimensional CSP starts with the generation of all feasible cutting patterns, usually making provision for such constraints as the minimum size of the trim, the number of cuts in a pattern and the number of different lengths in a pattern.

Beside the column generation methods discussed above, further approaches have been considered in the literature. Among them are the approaches of Johnson et al. [8], Arbel et al. [16], Yanasse et al. [17], Savsar and Cogun [18], Chauny et al. [19], and Ferreira et al. [20].

One of the factors that add to the complexity of the CSP is the large number of cutting patterns that may be encountered. When the CSP is expressed as an integer-programming problem, the large number of cutting patterns involved generally makes computation infeasible. However, if the LP formulation of the CSP is free of integer variables, then the effect of the number of cutting patterns will be mitigated.

Within this work context, a simple heuristic is developed that generates in advance all the feasible cutting patterns.

2. The problem

Cutting pattern generation of the following CSP is investigated. Paper rolls or metallic coils of standard width w_k ($k = 1, \ldots, h$) are slit to n sizes with width and length specifications of w_i and l_i ($i = 1, \ldots, n$), respectively. There is no limit on the lengths of the standard rolls (coils) since, for practical purpose, limited-length rolls can be connected together to yield the required lengths.

It is required to determine the production schedule (cutting patterns) that minimizes the total waste (trim losses) while satisfying the given demand. This CSP can be formulated as follows:

$$\text{Minimize } x_o = \sum_{k=1}^{h} \sum_{j=1}^{m_k} c_{jk} x_{jk} + \sum_{i=1}^{n} w_i s_i, \tag{1}$$

subject to

$$\sum_{k=1}^{h} \sum_{j=1}^{m_k} a_{ijk} x_{jk} - s_i = l_i \quad \text{for all } i, \tag{2}$$

$$x_{jk}, S_i, c_{jk}, a_{ijk} \geqslant 0 \quad \text{for all } i,\ j \text{ and } k, \tag{3}$$

where a_{ijk} is the number of units of width w_i being cut according to the jth pattern from the kth roll ($i = 1, \ldots, n$; $j = 1, \ldots, m_k$; $k = 1, \ldots, h$), x_{jk} is the length of kth roll being cut according to the jth pattern, c_{jk} is the cut loss of the kth roll being cut according to the jth pattern, s_i is the surplus length being produced of the roll with width w_i, and m_k is the number of cut patterns that can be produced from the kth roll.

To gain some appreciation of the dimension of the cutting pattern generation problem, the

possible combinations are considered regardless of the feasibility issue. To enumerate all possible combinations requires at least $2^n - 1$ different combinations of the required widths (where n is the number of required widths), i.e. for six required widths, at least 63 combinations are involved (Ref. [16]).

2.1. Generation of feasible cutting patterns

The generation of the feasible cutting patterns is achieved through a search tree. The levels of the tree represent the required widths that are arranged in a non-increasing order with the largest size at the first level whereas the smallest size occupying the highest level of the tree. The starting node of level I shows the standard width of the kth roll (w'_k) that is used to generate the patterns. Thus, a separate search tree is used to generate patterns associated with each standard width. The branches of level i of a search tree show the product of the number of units of required width i that is cut according to the jth pattern times the required width w_i, (i.e. $a_{ijk}\, w_i$). This product represents the total width that is cut from roll k to satisfy the required width w_i. The starting node of level i represents the remaining width after satisfying the cuttings specified by the previous $i - 1$ branches. The end terminal nodes at the highest level of the tree show the cut losses resulting from the different cut patterns.

The search tree is constructed by traversing first from bottom root to top and then from left to right. The construction starts at the root (first level of the tree), and continues to move upward in the tree by adding additional sizes to the combinations already specified by the previous branches. While traversing along a specific path on the search tree, the feasibility of that path is maintained by ensuring a non-negative end terminal node which implies adequate material to satisfy the requirements of the path. The path from the root of the tree to an end terminal node represents a feasible cutting pattern whose components a_{ijk} indicate widths which are combined together and the number of units of each width.

As illustrated in Fig. 1, each node at any level has as many branches as the maximum number of units

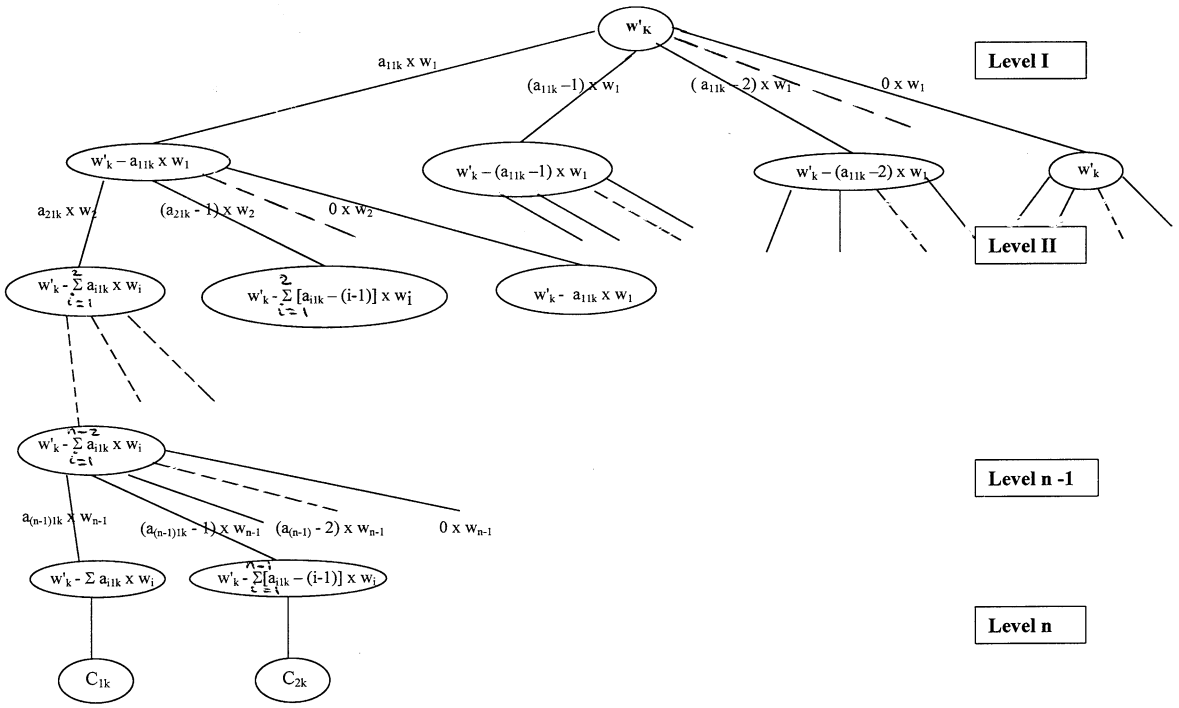

Fig. 1. General layout of the search tree.

that can be cut from the remaining width shown in that node, i.e. a_{ijk} determined by expression (7) which will be shown later, plus an additional branch. The left-most branch shows the total width assigned to a_{ijk}, whereas the subsequent branches from left to right show the total widths that are assigned to $a_{ijk}-1$, $a_{ijk}-2, \ldots, 2, 1$, respectively. The additional branch, placed at the right most of the branch group, shows the possibility of not choosing the width associated to that level in that particular combination. The path from the root of the tree to a terminal node represents a feasible cutting pattern (i.e. a column of matrix $[a_{ijk}]$).

3. Algorithmic approach

Let $w_1 > w_2 > w_3 > \cdots > w_n$. For each roll k, a matrix $[a_{ijk}]$ of $n \times m_k$ elements should be determined. The first element is given by

$$a_{11k} = \text{SmallestInt}\left(\frac{w'_k}{w_1}\right). \tag{4}$$

The second element in the first column is

$$a_{21k} = \text{SmallestInt}\left(\frac{w'_k - a_{21k}w_1}{w_2}\right). \tag{5}$$

The ith element in the first column is

$$a_{i1k} = \text{SmallestInt}\left[\left(w'_k - \sum_{z=1}^{i-1} a_{z1k}w_z\right)\Big/ w_i\right]. \tag{6}$$

Generally, any element in the matrix is given by

$$a_{ijk} = \text{SmallestInt}\left[\left(w'_k - \sum_{z=1}^{i-1} a_{zjk}w_z\right)\Big/ w_i\right]. \tag{7}$$

Dantzig [21] uses a formula similar to that of expression (7) in his ad hoc solution method to the knapsack problem formulation.

Having determined the elements of the matrix $[a_{ijk}]$, i.e. the cut patterns, the cut losses associated with these cut patterns can be determined by

$$c_{jk} = w'_k - \sum_{i=1}^{n} a_{ijk}w_i. \tag{8}$$

To determine the matrix $[a_{ijk}]$ of the kth roll ($k = 1$ for standard width w'_1, 2 for standard width $w'_2, \ldots, h$ for standard width w'_h), the following branch and bound algorithm is used.

1. Arrange the required width w_i ($i = 1, 2, \ldots, n$) in non-increasing order.
2. Apply expression (7) to fill the first column ($j = 1$) of the matrix.
3. Apply expression (8) to find the cut loss resulting from cut pattern 1.
4. Set the level index (row index) i to $n-1$.
5. Check the current node (cell) of level i, i.e. node (i,j). If the node has a value equal zero (i.e. $a_{ijk} = 0$), then proceed to Step 7. Otherwise, generate a new column $j = j + 1$ with the following elements:
   - $a_{zik} = a_{zi-1k}(z = 1, \ldots, i-1)$ elements to fill the nodes that precede the current node i,j.
   - $a_{ijk} = a_{ij-1k} - 1$ element to fill the current node i,j.
   - Fill the remaining nodes of the new column j, i.e. $a_{i+1jk}, a_{i+2jk}, \ldots, a_{njk}$, using expression (7).
6. Apply expression (8) to find the cut loss resulting from cut pattern j. Go to Step 4.
7. Decrement i, i.e. $i = i - 1$. If $i > 0$, then repeat Step 5. Otherwise, stop.

The steps of this algorithm are flowcharted as shown in Fig. 2. The CSP may be constrained by placing bounds on the trim losses. A lower bound may be specified when both sides of the rolls are trimmed to cut off the irregularities. Similarly, the management may also specify an upper bound on the trim losses. Such bounds reduce the number of patterns included in the LP model.

4. Illustrative example

To illustrate how the algorithm works, let us consider the following simple example. The Aluminum Rolling Mill Company (ARMCO) received four orders for aluminum coils with the following specifications:

Order number (i):	1	2	3	4
Required width w_i (cm):	50	40	30	20
Required length l_i (cm):	4000	5000	10 000	6000

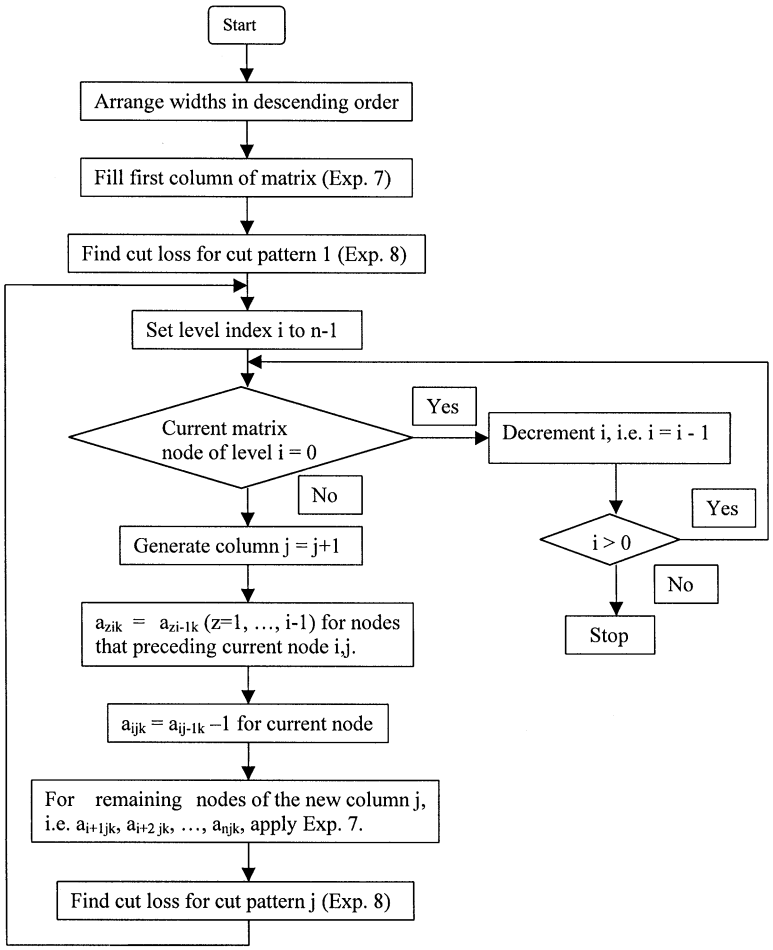

Fig. 2. Pattern generating algorithm flowchart.

Aluminum coils are produced by ARMCO in two standard widths: 130 and 100 cm that are slit to the sizes specified by the orders.

This problem has the form described earlier, and can be formulated by the model of expressions (1)–(3). The matrices $[a_{ijk}]$ ($k = 1$ for 130 cm and $k = 2$ for 100 cm) are prepared using the tree of Fig. 3.

Fig. 3 shows the search tree diagram that describes the different patterns produced from the 130 cm width roll (i.e. $w_1' = 130$ cm). The tree diagram is constructed from bottom root to top and from left to right. The branches of level I show the multiples of the largest required width (i.e. $w_1 = 50$) that can be produced from the 130 cm width, whereas the branches of level II show the multiples of the next largest required width (i.e. $w_2 = 40$), and so we branch along the tree as the required width decreases.

Starting with the left branch of level I, 2 units of width 50 is the maximum number of units that can be produced from the 130-width roll. The three branches of level I show respectively, from left to right 2, 1, and 0 units that can be cut from the 130-width roll. The terminal nodes of level IV show the cut losses resulting from the different cut patterns.

The elements of each column in Table 1 are corresponding to a sequence of connected branches in Fig. 3. For example, moving from top to bottom along the branches connecting the starting node of level I and the first terminal node at the bottom

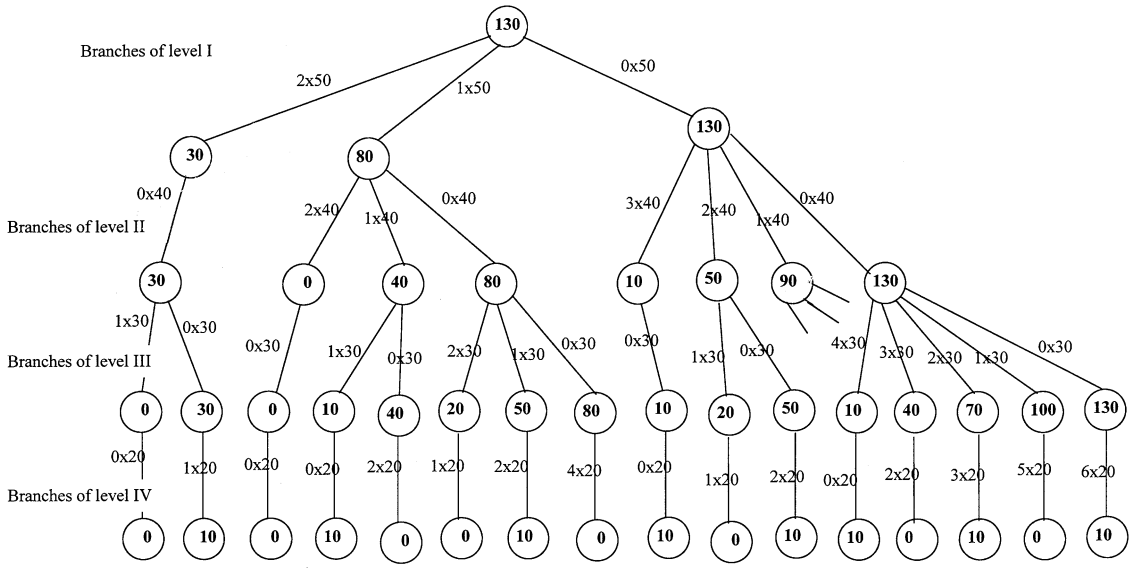

Fig. 3. Example search tree.

Table 1
Cutting patterns for 130 cm standard roll

Required width	Cutting patterns (columns)																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
50	2	2	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	2	1	1	0	0	0	3	2	2	1	1	1	1	0	0	0	0	0
30	1	0	0	1	0	2	1	0	0	1	0	3	2	1	0	4	3	2	1	0
20	0	1	0	0	2	1	2	4	0	1	2	0	1	3	4	0	2	3	5	6
Cut loss	0	10	0	10	0	0	10	0	10	0	10	0	10	0	10	10	0	10	0	10

level, i.e. level IV, one can identify the elements of column 1. Similarly, the elements of column j can be identified from the branches connecting the starting node of level I to the terminal node j of the last level.

Applying the algorithm to the example problem, the elements of the matrix a_{ij1}, i.e. for the 130 cm roll, are established in the following manner.

Step 1: Descending order of the required width: 50, 40, 30, 20.

Step 2: Elements of first column ($j = 1$):

- $a_{111} = \text{SmallestInt}[(130 - 0)/50] = 2$,
- $a_{211} = \text{SmallestInt}[(130 - 2 \times 50)/40] = 0$,
- $a_{311} = \text{SmallestInt}[(130 - 2 \times 50 - 0 \times 40)/30] = 1$,
- $a_{411} = \text{SmallestInt}[(130 - 2 \times 50 - 0 \times 40 - 1 \times 30)/20] = 0$.

Step 3: Cut loss of pattern 1, $C_{11} = 130 - (2 \times 50 + 0 \times 40 + 1 \times 30 + 0 \times 20) = 0$

Step 4: Set the level index i (row index) $= 4 - 1 = 3$

Step 5: Since the current node of level III (i.e. a_{311}) is greater than zero, a new column $j = j + 1 = 2$ is introduced with the following elements:

- $a_{121} = a_{111} = 2$ (elements of the nodes preceding the current node),
- $a_{221} = a_{211} = 0$,
- $a_{321} = a_{311} - 1 = 1 - 1 = 0$ (element to fill the current node i,j),
- $a_{421} = \text{SmallestInt}[(130 - 2 \times 50 - 0 \times 40 - 0 \times 30)/20]$
  $= 1$ (element of the remaining node).

Step 6: Cut loss of pattern $2\,C_{21} = 130 - (2 \times 50 - 0 \times 40 - 0 \times 30 - 1 \times 20) = 10$

Go back to Step 4.

Step 4: Set the level index $i = n - 1 = 3$.

Step 5: Since the current node of level III (i.e. a_{321}), is equal to zero, the algorithm branches to Step 7.

Step 7: the index i is decremented by 1, i.e. $i = i - 1 = 2$ which is greater than zero. Thus Step 5 is repeated.

Step 5: The current node of level II (i.e. a_{221}) is equal to zero, thus the algorithm branches again to Step 7.

Step 7: The index i becomes $i = 1$ which is greater than zero. Step 5 is repeated.

Step 5: The current node of level I (i.e. a_{121}) is greater than zero, a new column $j = j + 1 = 3$ is introduced with the following elements.

No preceding element since the current node is the first node in the column.

- $a_{131} = a_{121} - 1 = 1$ (element to fill the current node).
- $a_{231} = \text{SmallestInt}[(130 - 1 \times 50)/40]$
  $= 2$ (elements of the remaining nodes).
- $a_{331} = \text{SmallestInt}[(130 - 1 \times 50 - 2 \times 40)/30] = 0$.
- $a_{431} = \text{SmallestInt}[(130 - 1 \times 50 - 2 \times 40 - 0 \times 30)/20] = 0$.

Step 6: Cut loss of pattern $3\,C_{31} = 130 - (1 \times 50 + 2 \times 40 + 0 \times 30 + 0 \times 20) = 0$.

Go back to Step 4.

Step 4: Set the level of index $i = n - 1 = 3$

Step 5: The current node of level III (i.e. a_{331}) is equal to zero, thus we proceed to Step 7.

Step 7: The index $i = 2$ that is greater than zero. Step 5 is repeated.

Step 5: The current node of level II (i.e. a_{231}) is greater than zero, thus a new column $j = j + 1 = 4$ is introduced with the following elements:

- $a_{141} = a_{131} = 1$ (elements of the nodes preceding the current node).
- $a_{241} = a_{231} - 1 = 1$ (element to fill the current node).
- $a_{341} = \text{SmallestInt}[(130 - 1 \times 50 - 1 \times 40)/30]$
  $= 1$.
- $a_{441} = \text{SmallestInt}[(130 - 1 \times 50 - 1 \times 40 - 1 \times 30)/20]$
  $= 0$ (elements of the remaining nodes).

Step 6: Cut loss of pattern $4\,C_{41} = 130 - (1 \times 50 + 1 \times 40 + 1 \times 30 + 0 \times 20) = 10$.

Go back to Step 4.

The algorithm proceeds in the same manner to produce all the cut patterns shown in Table 1 for the 130 cm standard roll. The cut patterns of the 100 cm standard roll can be produced in a similar manner.

5. Further examples and applications

Based on the branch and bound algorithm described above, a special computer routine for generating cutting patterns is developed. The routine had been validated and used in determining the cutting patterns of the corrugated box factory model of Savsar and Cogun (Ref. [18]). Applying the upper and lower limits of the trim losses specified in the case study (trim loss range of 25–150 mm), 73 cutting patterns, typical to those produced by direct enumeration and reported in Ref. [18], were generated in a fraction of a second using a Pentium II 450 MHz (128 RAM).

An aluminium rolling mill plant of 100,000 tons capacity located in the Arabian Gulf region produces a wide series of aluminium alloys through hot and cold 4-high rolling mills, respectively. Aluminium coils of 100, 120, and 150 cm width are

produced by cold rolling. These coils are then send to the finishing department for slitting to the required width. In average, the mill works on 8–10 orders every week. Every time and other, there are 4–8 orders of different width requirements of the same series. The developed routine is currently being used by the plant to generate in advance all feasible cutting patterns, which are usually ranging between 20 and 150 patterns per coil. Having generated the cutting patterns, i.e. the entries of matrix a_{ijk}, the solution of the LP problem formulation of expressions (1)–(3) is achieved within a fraction of a second using standard LP package on a Pentium II 450 MHz (128 RAM) machine.

Although the pattern generation of the one-dimensional CSP is addressed in this work, however, the simple procedure developed can be used for the two- and three-dimensional CSPs by considering the area and volume of the required units, respectively, rather than their width. Consequently, the search tree will be modified to represent the dimensions of the problem. For example, each branch in the search tree of the two-dimensional problem will represent the total length and total width required for a specific number of units to be cut, and each intermediate node in the tree will show the unassigned length and width of the stock. In brief, the approach can be extended to generate cutting patterns for the two- and three-dimensional CSPs.

6. Conclusion

The CSP is frequently encountered in paper and sheet metalworking industries where paper rolls and metallic coils of standard width are slit to satisfy specific size requirement. In such situations, it's required to determine the production schedule (cutting patterns) that minimizes the total waste. The problem fits a LP formulation. An auxiliary problem arises from the LP formulation of the CSP. The auxiliary problem is manifested by the need to determine the columns of the LP constraint matrix. The columns represent the different cutting patterns that can be produced from the available stock material. A simple pattern generating procedure is developed for solving this auxiliary problem. The procedure is based on an ad hoc method of solution described in the literature for the knapsack problem. Examples are presented to demonstrate the procedure and its applications.

References

[1] C.H. Chenge, B.R. Feiring, T.C. Cheng, The cutting stock problem – A survey, International. Journal of Production Economics 36 (1994) 291–305.
[2] K.K. Lai, W.M. Chan, An evolutionary algorithm for the rectangular cutting stock problem, International Journal of Industrial Engineering 4 (2) (1997) 130–139.
[3] A.A. Farley, Mathematical programming models for cutting-stock problems in the clothing industry, Journal of Operational Research Society 39 (1988) 41–53.
[4] A.A. Farley, A note on bounding a class of linear programming problems, including cutting stock problems, Journal of Operations Research 38 (1990) 922–923.
[5] P.C. Gilmore, R.E. Gomory, A liner programming approach to the cutting-stock problem, Journal of Operations Research 9 (1961) 849–859.
[6] K.G. Murty, Linear Programming, Wiley, Chichester, 1983.
[7] T.A. Schultz, Application of linear programming in a gauze splitting operation, Journal of Operations Research 43 (5) (1995) 752–757.
[8] M.P. Johnson, C. Rennick, E. Zak, Skiving addition to the cutting stock problem in the paper industry, Journal of Siam Review 39 (3) (1997) 472–483.
[9] P.C. Gilmore, R.E. Gomory, Theory and computation of knapsack functions, Journal of Operations Research 14 (1966) 1045–1074.
[10] R.E. Johnston, Rounding algorithms for cutting stock problems, Asia–Pacific Journal of Operational Research 3 (1986) 166–171.
[11] P.C. Gilmore, R.E. Gomory, A linear programming approach to the cutting stock problem – part II, Journal of Operations Research 11 (1963) 863–888.
[12] R.W. Haessler, A heuristic programming solution to a nonlinear cutting stock problem, Journal of Management Science 17 (1971) B793–B802.
[13] R.W. Haessler, Controlling cutting pattern changes in one-dimensional trim problems, Journal of Operations Research 23 (1975) 483–493.
[14] N. Christofides, C. Whitlock, An algorithm for the two-dimensional cutting problem, Journal of Operations Research 25 (1977) 30–44.
[15] C. Goulimis, Optimal solutions for the cutting stock problem, European Journal of Operational Research 44 (1990) 197–208.
[16] A. Arbel, Large-scale optimization methods applied to the cutting stock problem of irregular shapes, International Journal of Production Research 31 (2) (1993) 483–500.

[17] H.H. Yanasse, A.S.I. Zinober, R.G. Harris, Two-dimensional cutting stock with multiple stock sizes, Journal of Operational Research Society 42 (1991) 673–683.
[18] M. Savsar, C. Cogun, Analysis and modelling of a production line in a corrugated box factory, International Journal of Production Research 32 (7) (1994) 1571–1589.
[19] F. Chauny, R. Loulou, S. Sadones, F. Soumis, A two-phase heuristic for the two-dimensional cutting-stock problem, Journal of the Operational Research Society 42 (1991) 1.
[20] J. Ferreira, M. Neves, P. Fonseca, A two-phase roll cutting problem, European Journal of Operational Research 44 (2) (1990) 185–196.
[21] G.G. Dantzig, P. Wolfe, Decomposition principle for linear programs, Journal of Operational Research 8 (1960) 101–111.