A Genetic Algorithm Approach for the Multiple Length Cutting Stock Problem

Yueh-Hong Chen
Department of Computer Science and Information Engineering
Far East University
Taiwan, R.O.C.
yuehhong@gmail.com

Hsiang-Cheh Huang
Department of Electrical Engineering
National University of Kaohsiung
Taiwan, R.O.C.
hch.nuk@gmail.com

Hong-Yi Cai
Department of Computer Science and Information Engineering
Far East University
Taiwan, R.O.C.
yuehhong@gmail.com

Pan-Fang Chen
Department of Computer Science and Information Engineering
Far East University
Taiwan, R.O.C.
yuehhong@gmail.com

***Abstract*—In this paper, we proposes a genetic algorithm approach to solving the one dimensional cutting stock problem with multiple stock lengths. The methods to solve one-dimensional cutting stock problems can be divided into two types: (1) exact algorithm, and (2) approximation algorithm. The execution time of an exact algorithm will increase to an unacceptable degree if the scale of the problem is larger. Since the one-dimensional cutting stock problem is a constrained optimization problem, the methods of genetic algorithm are suitable to solve this problem. The chromosome encodes the index of the stock material that demand pieces will be cut from. Then, the chromosome initialization process combining with random and heuristic scheme was proposed. Since infeasible solution may be generated after crossover and mutation operation were applied on chromosomes. A repair process was proposed to keep the offspring feasible. Experimentation shows that the proposed method actually can obtain an approximation solution in the constraint of acceptable time cost.**

***Keywords*—cutting stock problem, genetic algorithm, np-hard problem**

I. Introduction

In many industries, such as manufacturing, construction, textiles or transportation, it is often necessary to cut a fixed-size raw material into the required (or customer-specified) size under the constraint of minimizing costs, or minimizing the number of cutting knives. This type of problem is called the cutting Stock Problem(CSP). Depending on the dimensions of the raw materials, the cutting stock problem can be divided into one-dimensional (1-D), two-dimensional (2-D) and three-dimensional (3-D) cutting problems. Due to the high cost of raw materials in manufacturing, the 1-D CSP has always attracted the attention of the computer science researchers. However, the 1-D CSP had been proved to be an NP-Hard problem[1]. Although the method of finding the optimal solution exists, as the scale of the problem increases, the time required to find the optimal solution will still grow rapidly. Therefore, the methods to obtain the optimal solution of cutting stock problems have not been used in large scale problems. Since the 1-D CSP is a constrained optimization problem, the methods of evolutionary computation are suitable to solve this problem[2-5].

The methods to solve 1-D CSP can be divided into two types: (1) methods to obtain the optimal solution, and (2) methods to obtain acceptable solutions within acceptable time limits. These methods are briefly described as follows.

The column generation method proposed in [6, 7] is an old and commonly used method to solve cutting stock problems. In the column generation method, only a (usually small) subset of the variables is used initially. The method sequentially adds columns (i.e., variables), using information given by the dual variables for finding the appropriate variable to add. In [8], an exact hybrid solution procedure, called BISON, is proposed. BISON method favourably combines the well-known meta-strategy tabu search and a branch and bound procedure based on known and new bound arguments and a new branching scheme. Computational results shown in [8] indicate that BISON is effective. Since there have been several efforts to attack the 1-D CSP by linear programming based branch-and-bound with column generation (called branch-and-price) and by general-purpose Chvátal–Gomory cutting planes. A combination of both approaches was proposed in [9]. In [9], the LP relaxation at each branch-and-price node is strengthened by Chvátal–Gomory and Gomory mixed-integer cuts. The branching rule is that of branching on variables of the Gilmore–Gomory formulation. In [10], an exact method, based on an arc-flow formulation with side constraints was presented. This method solve cutting stock problems by simply representing all the patterns in a very compact graph. This formulation is equivalent to Gilmore and Gomory's, thus providing a very strong linear relaxation. However, instead of using column-generation in an iterative process, the method constructs a graph, where paths from the source to the target node represent each valid cutting pattern.

On the other hand, methods to find an acceptable solution within acceptable time cost, such as artificial neural network[11], genetic algorithm[12], simulated annealing[13], particle swarm optimization[14], or evolutionary approach [15] have emerged for solving intractable large scale cutting stock problems. These schemes have been identified as having the potential to solve combinatorial optimization problems with near optimal or optimal solutions. In [12], a method based on genetic algorithm is presented. This method uses a tree structure to represent the cutting pattern, and combines different patterns to generate patterns with higher performance. In [15], a novel evolutionary programming algorithm was proposed for cutting stock problems with and without contiguity, and two new mutation operators are proposed. In [14], a heuristic strategy that is based on the results of analysis of the optimal cutting pattern of particles with successful search processes is described, which process a global optimization problem of the cutting-stock as a sequential optimization problem by multiple stages. During every sequential stage, the best cutting pattern for the current situation is researched and processed. This strategy is repeated until all the required stocks have been generated.

Acknowledgments: This work is supported in part by Ministry of Science and Technology (formerly National Science Council) of Taiwan, R.O.C., under grants MOST 107-2622-E-269-011-CC3.

Two meta-heuristic algorithms, namely simulated annealing (SA) and tabu search (TS) were proposed in [13]. Moreover, several problems were solved using SA and TS, and then the experimental results were compared.

In real world applications, vendors may provide raw material with different sizes at the same time. However, the above works had not taken the cutting stock problem with multiple stock lengths into consideration. Thus, in this paper, we proposes a genetic algorithm approach to solve this problem. The experimental results show that the proposed method can exactly obtain an feasible and acceptable solution within acceptable time cost.

II. Cutting Stock Problem with multiple stock lengths

An 1-D CSP is the problem of cutting standard-sized pieces of one dimensional stock material, such as sheet metal, into pieces of specified sizes such that the wasted material is minimized. Precisely speaking, the cutting stock problem can be described as follows: supposed there are m types of standard-sized stock materials with different lengths. Given unlimited standard-sized stock materials with length L_1, L_2, L_3, …, L_m, and n types of demand pieces. The length of demand pieces of type i is l_i, and the demand number is d_i, $1 \leq i \leq n$. If the amount of stock materials of type j used to generate demand pieces is C_j, $1 \leq j \leq m$, total length of wasted materials, referred to as W can be compute with Eq. (1):

$$\mathrm{W} = \sum_{j=1}^{m} C_j \times L_j - \sum_{i=1}^{n} d_i \times l_i \qquad (1)$$

and the percentage P of wasted materials is:

$$\mathrm{P} = \frac{W}{\sum_{j=1}^{m} C_j \times L_j} \times 100\% \qquad (2)$$

An example of the cutting stock problem with multiple stock lengths is shown in Table 1.

Table 1 An example of the cutting stock problem with multiple stock lengths

Stock	Demand	
length	length	number
9,000	150	25
10,000	385	30
12,000	850	15
	1,000	20
	6,530	10

The example problem shown in Table 1 is a small-scale problem, but it is still not easy to find the optimal solution by hand. Because the demand type n is small, an exact algorithm described in Section I can find out the optimal solution in seconds. The optimal solution with minimal wasted material length is shown in Table 2.

In Table 2, the percentage of wasted materials of the optimal solution is extremely small, and it is rarely achieved in real world applications. In addition, it is also possible that more than one optimal solution exists. Taking the above problem as an example, the other optimal solution is shown in Table 3.

Table 2. optimal solution of the example problem shown in Table 1

Stock material		Cutting plan
length	number	
12,000	1	6530×1, 1000×2, 385×9
12,000	5	6530×1, 1000×3, 850×2, 385×2
12,000	1	6530×1, 1000×1, 850×1, 385×2, 150×19
12,000	1	6530×1, 1000×2, 385×5, 150×6
9,000	2	6530×1, 850×2, 385×2
length and % of wasted materials		650 (0.5%)

Table 3. another optimal solution of the example problem shown in Table 1

Stock material		Cutting plan
length	number	
10,000	1	6530×1, 1000×2, 385×3, 150×2
10,000	4	6530×1, 1000×1, 850×2, 385×2
10,000	1	6530×1, 385×2, 150×18
9,000	3	6530×1, 850×2, 385×2
9,000	1	1000×9
9,000	1	6530×1, 1000×1, 850×1, 150×4
9,000	1	1000×4, 385×11, 150×1
length and % of wasted materials		650 (0.5%)

The optimal cutting plan shown in Table 3 only use stock materials with length 9,000 and 10,000, and total length of wasted materials is also 610.

III. Genetic Algorithm

Genetic algorithm(GA), based on the concept of natural genetics, is a directed random search technique. The exceptional contribution of this method was developed by Holland [2] over the course of 1960s and 1970s, and finally popularized by Goldberg [3]. In the genetic algorithms, the parameters are represented by an encoded numeric string, called the ”chromosome”. And the elements in the numeric strings, or the ”genes”, are adjusted to minimize or maximize the fitness value. The fitness value, generated with a function, referred to as ”fitness function” is composed of multiple variables to be optimized by GA. For every iteration in GA, a pre-determined number of individuals (i.e., chromosomes) will correspondingly produce fitness values associated with the genes. Fig. 1 demonstrates the flow chart for a typical binary GA. It begins by defining the optimization parameters, the fitness function, and the fitness value, and it ends by testing for convergence. According to the applications for optimization, designers need to carefully define the necessary elements for training with GA. Then, we are able to evaluate the fitness function in addition to the terminating criteria with the natural selection, crossover, and mutation operations in a reasonable way[4, 5].

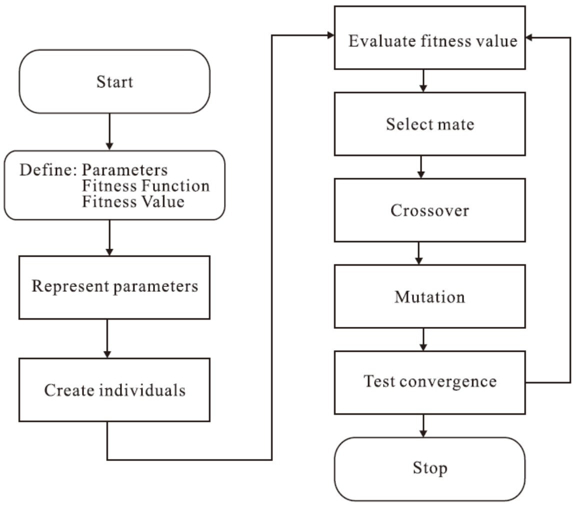

Fig. 1 The flowchart of genetic algorithm

Assuming that we employ GA to search for the largest fitness value with a given fitness function. In GA, as shown in Fig. 1, the core components are depicted as follows:

1) **Mate selection:** A large portion of individuals with low fitness value is discarded through this natural selection step. Of the *N* individuals in one iteration, only the top N_{good} individuals survive for mating, and the bottom $N_{\text{bad}} = N - N_{\text{good}}$ ones are discarded to make room for the new offspring in the next iteration. Therefore, the selection rate is N_{good}/N.
2) **Crossover:** Crossover is the first way that a GA explores a fitness surface. Two individuals are chosen from *N* individuals to produce two new offspring. Several crossover method can be adopted to generate offspring. For example, with single point crossover, a crossover point is selected between the first and last genes of the parents' individuals. Then, the fractions of each individual after the crossover point are exchanged, and two new offspring are produced.
3) **Mutation:** Mutation is the second way that a GA explores a fitness surface. It introduces traits not in the original individuals, and keeps GA from converging too fast. The pre-determined mutation rate should be low. Most mutations deteriorate the fitness of an individual; however, the occasional improvement of the fitness adds diversity and strengthens the individual.

A chromosome of the genetic algorithm used in the proposed scheme is a numeric string. The length of the numeric string is equal to

$$\mathrm{D} = \sum_{i=1}^{n} d_i \tag{3}$$

where *D* is the amount of demand pieces in cutting stock problem. In other words, each chromosome *G* consists of *D* numbers in the range of 1 to $D \times m$, and can be represented as:

$$G = g_1, g_2, \ldots, g_k, \ldots, g_D, \quad 1 \le k \le D \tag{4}$$

Each number g_k in a chromosome represents the stock material to which the demand piece *k* belongs. Because there are *D* demand pieces and *m* kinds of standard-sized stock materials, g_k is in the range of 1 to $D \times m$. If the value of g_k equals to $j \times D + i$, $0 \le j \le (m\text{-}1)$ and $1 \le i \le D$, then the demand piece corresponding to g_k will be cut from *i*-th stock material of the $(j+1)$-th type. Since there will be many unused stock materials, in practice, if (*i*-1)-th stock material is unused, as well as $i > 1$, all demand pieces belonging to *i*-th stock material will be moved to (*i*-1)-th stock material after the crossover and mutation operations are applied.

An example of the chromosome is as follows. Assuming that there are two types of standard-sized stock materials with length 3 and 5. Three pieces with length 1, two pieces with length 2 and two pieces with length 4 are going to be cut from the stock materials. A feasible chromosome is shown in Fig. 2.

Length of the demand pieces	1	1	1	2	2	4	4
Chromosome	1	8	9	1	2	8	9
Index *k*	1	2	3	4	5	6	7

Fig. 2 An example of the chromosome

In this example, stock materials indexed from 1 to 7 are with length 3; from 8 to 14 are with length 5. Since $g_1 = g_4 = 1$, the first and the forth demand piece will be obtained from stock material 1. Demand pieces corresponding to g_2 and g_6 are obtained from stock material 8; g_3 and g_7 from 9; g_5 from 2. Therefore, two stock materials with length 3 and two with length 5 are required.

In order to increase the convergence speed of the genetic algorithm, the individuals (i.e., chromosomes) are initialized with a method combining random and heuristic schemes. To form a chromosome, an unused stock material is randomly selected. Then, the demand pieces chosen randomly are assigned to the selected stock material until the rest of the material is not enough. The process is repeated until all demand pieces are assigned to a certain stock material.

The crossover operation used in the proposed method is to simply exchange the values of random chosen positons of two parent chromosomes. About 30% numbers will be exchanged in a crossover process. Because the crossover operation may generate offspring containing infeasible solutions, it is necessary to apply a repair process on two offspring. The infeasible solution generated by the crossover operation will contain infeasible cutting plans in which total length of demand pieces cutting from a single stock material is longer than the length of the material. The repair process will select a demand piece randomly, and then randomly move it to other cutting plan that still has enough length of unused material. If no cutting plan meets the requirement, a totally unused stock material will be chosen.

The mutation operation proposed in this paper is similar to the crossover operation. Several genes, g_k, are randomly chosen, and randomly generated values are assigned to these genes, respectively. The effect of the mutation operation is to find new efficient cutting plans, so these randomly generated values do not include the numbers corresponding to totally unused stock material. After the mutation operation is applied, the repair process will be used to deal with the infeasible solutions.

The fitness function used to evaluate the quality of solutions is similar to Eq.(1), but total length of wasted materials is multiplied by -1. So, the shorter the total length of wasted materials is, the higher the fitness value of the chromosome.

$$\text{fiteness} = \sum_{i=1}^{n} d_i \times l_i - \sum_{j=1}^{m} C_j \times L_j \qquad (5)$$

IV. Experimental Results

In this section, we present some experimental results to demonstrate the performance of the proposed method. Several randomly generated cutting stock problem with multiple stock lengths are used as test cases. Since number of types of the standard sized stock material is usually small, m is fixed to 3 among all test cases, and the lengths of stock materials are 9,000, 10,0000 and 12,000 in all test cases. The amount of types of demand pieces varies from 40 to 100 (i.e., $40 \le n \le 100$). Demand length and demand number of pieces are also generated randomly.

Parameters of the genetic algorithm used to solve above problems are described as follows. The population consists of 500 chromosome. The selection and mutation rates are set to 0.5 and 0.04, respectively, and the number of training iteration is 300. Experimental results are shown in Table 4.

Table 4 Experimental results obtained with the proposed method

Amount of types (n)	Amount of demand pieces	Length of wasted material	Percentage of wasted material
40	183	1,022,415	1.8%
50	224	1,563,184	2.2%
60	268	1,956,631	2.4%
70	310	2,484,713	2.5%
80	358	3,067,237	2.7%
90	401	3,706,347	2.9%
100	447	4,401,938	3.1%

As shown in Table 4, the percentage of wasted stock material increase with n. It is because a larger n will form a larger search space. Therefore, the number of chromosomes or the number of training iteration should increase correspondingly, if a solution with similar quality is desired.

The percentage of wasted stock material is lower than 3% until the amount of demand type increase to 100. Therefore, the proposed method actually can obtain an approximation solution in the constraint of acceptable time cost. In contrast with exact algorithms, although the solutions obtained with the proposed method are approximation solution, the proposed method can deal with a case in which n is 100. If n is larger than 60, the execution time of an exact algorithm will increase to an unacceptable degree.

V. Conclusion

In this paper, we proposes a genetic algorithm approach to solving the one dimensional cutting stock problem with multiple stock lengths. The chromosome encode the index of the stock material that demand pieces will be cut from. Then, the chromosome initialization process combining with random and heuristic scheme was proposed. Since infeasible solution may be generated after crossover and mutation operation were applied on chromosomes. A repair process was proposed to keep the offspring feasible. Experimentation shows that The percentage of wasted stock material is lower than 3% until the amount of demand type increase to 100. Therefore, the proposed method actually can obtain an approximation solution in the constraint of acceptable time cost.

References

[1] M. R. Garey and D. S. Johnson, Computers and Intractability; A Guide to the Theory of NP-Completeness, W. H. Freeman & Co., New York, NY, USA, 1990.

[2] J. H. Holland, Adaptation in Natural and Artificial Systems. The MIT Press, April 1992.

[3] D. E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Professional, Jan 1989.

[4] Y.-H. Chen and H.-C. Huang, ``Robust Watermarking for Wavelet-Based Stereoscopic Images with Genetic Algorithm," in Int'l Conf. on Robot, Vision and Signal Processing, pp. 155-158, Kaohsiung, Taiwan, R.O.C., 2015.

[5] Y.-H. Chen and H.-C. Huang, ``Reversible Image Watermarking Based on Genetic Algorithm," in Int'l Conf. on Intelligent Information Hiding and Multimedia Signal Processing, pp. 21-24, Kitakyushu, Japan, 2014.

[6] P. C. Gilmore and R. E. Gomory, "A linear programming approach to the cutting-stock problem," *Operations Research*, vol. 9, no. 6, pp. 849-859, Dec. 1961.

[7] P. C. Gilmore and R. E. Gomory, "A linear programming approach to the cutting stock problem–Part II," *Operations Research*, vol. 11, no. 6, pp. 863-888, Dec. 1963.

[8] A. Scholl, R. Klein and C. Jürgens, "Bison: A fast hybrid procedure for exactly solving the one-dimensional bin packing problem," *Computers & Operations Research*, vol. 24, issue 7, pp. 627-645, July 1997

[9] G. Belov and G. Scheithauer, "A branch-and-cut-and-price algorithm for one-dimensional stock cutting and two-dimensional two-stage cutting," *European Journal of Operational Research*, vol. 171, Issue 1, pp. 85-106, May 2006.

[10] F. Brandão and J. P. Pedroso, "Bin packing and related problems: General arc-flow formulation with graph compression," *Computers & Operations Research*, vol. 69, pp. 56-67, May 2016.

[11] S. Gu and T. Hao, "A pointer network based deep learning algorithm for 0–1 Knapsack Problem," *Tenth International Conference on Advanced Computational Intelligence (ICACI)*, pp. 357-361, 2018.

[12] P. Andras, A. Andras and Z. Szabo, "Genetic Solution for the Cutting Stock Problem," *1st Online Workshop on Soft Computing (WSC1)*, pp. 87-92, 1996.

[13] M. H. Jahromi, R. Tavakkoli-Moghaddam, A. Makui and A. Shamsi, "Solving an one-dimensional cutting stock problem by simulated annealing and tabu search," *Journal of Industrial Engineering International*, vol. 8, no. 24, Oct 2012.

[14] X. Shen, Y. Li, J. Yang and L. Yu, "A Heuristic Particle Swarm Optimization for Cutting Stock Problem Based on Cutting Pattern," *ICCS 2007, Lecture Notes in Computer Science*, vol. 4490. Springer, Berlin, Heidelberg, 2007.

[15] K.-H Liang, X. Yao, C. Newton and D. Hoffman, "A new evolutionary approach to cutting stock problems with and without contiguity," *Computers & Operations Research*, vol. 29, Issue 12, pp. 1641-1659, Oct 2002..