Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/091L23

Applying an Enhanced Heuristic Algorithm to a Constrained Two-Dimensional Cutting Stock Problem

Tzu-Yi Yu[1,*], *Jiann-Cherng Yang*[2], *Yeong-Lin Lai*[3], *Chih-Cheng Chen*[3] *and Han-Yu Chang*[1]

[1] Department of Information Management, National Chi Nan University, 470, University Rd., Puli, Taiwan, R.O.C.
[2] Department of Applied Mathematics, Feng Chia University, 100, Wenhwa Rd., Taichung, Taiwan, R.O.C.
[3] Department of Mechatronics Engineering, National Changhua University of Education, Changhua, Taiwan, R.O.C.

Received: 24 Nov. 2013, Revised: 25 Mar. 2014, Accepted: 26 Mar. 2014
Published online: 1 Feb. 2015

Abstract: This study focuses on developing fully functional cutting stock software by utilizing an improved multi-phase heuristic algorithm to find the optimal solution for the cutting material. Issues which commonly appear in real-world cutting stock applications are also discussed

Keywords: Cutting Stock, Guillotine, Genetic Algorithm

1 Introduction

The classic optimization issue known as the cutting stock problem (CSP) is common in industrial manufacturing. It is often referred to as a nesting or packing problem and typical scenarios include how to most efficiently cut sheets of raw materials into pieces or rolls of different sizes. Applications and methodologies designed to optimize efficiency by minimizing cutting waste through have been researched by programmers and businesses for decades. Extensions of CSP have been utilized in industrial applications including packing, loading and assortment composition. This issue has gained even more attention in the recent global economic downturn as companies seek to reduce labor costs and material waste in order to remain competitive. CSP generally contains many variables and leads to many feasible solutions. Recent research shows that heuristic algorithms can obtain the optimal solution in the solution domain. In CSP, the material to be cut is referred as a sheet or plate, and the patterns which need to be cut are called pieces. In this study, the major focus is placed on a two-dimensional constrained guillotine cut with multiple stock sheets. Commonly used in panel saw machines, a guillotine cut requires a cut on a regular sheet to begin at one edge and run continuously to another edge.

Most theoretical 2D guillotine problems use heuristic algorithms which focus on single sheets in attempting minimize cutting waste. However, in real-world industrial applications, cutting is commonly performed on multiple sheets to raise manufacturing efficiency. Therefore, it is necessary to modify and enhance the traditional objective function used for single sheets. In order to reach the maximum cutting ratio, available algorithms require heavy computational resources. Heuristic algorithms require both a large population and number of generations to reach the optimal result, resulting in a large amount of computation. However, many factories require minimal delays in cutting sequences, particularly when the material being cut is inexpensive and manufacturing time is the primary concern. Under this scenario, the tradeoff between the efficiency and accuracy of a given algorithm becomes important. If the cutting ratios are very close, many applications tend to favor solution speed over maximum optimization. This study presents an improved heuristic algorithm to handle a 2D multiple-sheet constrained guillotine cutting problem, with consideration given to real-world application..

2 Literature Review

When cutting a large stock sheet into smaller pieces, two issues appear: (1) an assortment problem and (2) a trim loss problem. The first issue is associated with a classification problem, while the second is related to

[*] Corresponding author e-mail: tyyu@ncnu.edu.tw

© 2015 NSP
Natural Sciences Publishing Cor.

reducing waste material [1]. These issues make CSP an NP-hard problem. The cutting domain can vary from one to n dimensions, yet most studies have focused on 1to3-dimension problems [2,?,?]. In 1964, Gilmore and Gomory applied a linear programming approach to associated 1D and 2D cutting problems [5,?]. Their approach not only solved the 1D problem by treating it as a knapsack problem, but also treated the 2D problem as a collection of 1D cases which could also be solved as a knapsack function. Since then, many 2D applications have been studied. Based on cutting constraints, 2D CSP can be roughly classified as guillotine and non-guillotine cuts. A guillotine cut refers to patterns with uninterrupted cuts running from one side of the sheet or from previously cut fragments to an opposite side. In a guillotine cut, an n-stage cutting implies a cutting pattern is cut in n phases. Each stage consists of guillotine cuts of the same direction, with two adjacent stages running in perpendicular directions. For example, in two-stage cutting, strips (strip patterns) produced in the first stage are cut into pieces in the second stage. The term non-stage refers to the opposite of the n-stage. It does not specify any stage number in which to cut. If the number of pieces to be cut from the plate is not constrained, it is called *unconstrained cutting*, otherwise, it is considered *constrained cutting*.

In 1985, Beasley presented a dynamic programming recursion method to solve staged cutting and general guillotine CSP [7]. While this approach obtains several optimal solutions using a recursion algorithm, it worked for small-scale cutting problems only. Christofides and Whitlock applied a tree-search method to solve the 2D guillotine cutting problem [8]. It overcame the disadvantages of recursive algorithms and worked for small scale problems. Following this method, Hifi Ouafi [9] and Viswanathan Bagchi [10] improved the tree-search approach by adding a Best-First strategy to improve efficiency.

Wang presented two combinatorial methods to create different guillotine cutting patterns and simultaneously assemble pieces vertically or horizontally to create a larger rectangular cutting area to form final cutting patterns [11]. Vasko [12] and Oliberia Ferreira [13] later presented an improved method based on Wangs approach, yet efficiency and accuracy issues remained. Following that, Valdes et al. presented the tabu search algorithm to solve general cutting problems [14]. A tabu search is a heuristic search algorithm commonly used for job shop and path finding problems. They further utilized the meta-heuristic algorithm known as Greedy Randomized Adaptive Search Procedure (GRASP) to improve optimal solutions and reduce the waste ratio over two-stage guillotine cutting problems [15]. Even though this algorithm demonstrated better performance, it required more computation time, again highlighting the tradeoff between CSP performance and efficiency.

Linear or dynamic programming attempts to search all possible solutions in feasible search domains. This approach is capable of finding the best optimal solution for small-scale problems. However, real-world application is generally complicated and typically large in scale. Additionally, these types of algorithms tend to be too time-consuming for practical use. As for the best-first search algorithm, there is no guarantee that it can find the best optimal solution. Thus, heuristic algorithms such as SA (simulated annealing) or GA (genetic algorithms) provide feasible candidate approaches and have recently become commonly adopted for CSP. Much CSP research has focused on the application of heuristic algorithms after the dramatic improvements in the computational capability of personal computers. For example, Lai and Chan used a simulated annealing algorithm to create a specific cutting pattern for a non-guillotine cutting problem [16]. Faina utilized a simulated annealing algorithm for guillotine and non-guillotine cutting problems [17]. Parada et al. [18] constructed a binary tree for particular cutting patterns for the constrained guillotine cutting problem with a simulated annealing approach. Since genetic algorithms are particularly efficient for integer-based problems, they are commonly applied to cutting stock applications. Wu [19], Goncalves [20], Pal [21], Liu Teng [22], Jakobs [23], and Babu Babu [24] provide notable examples.

Research publications often have a different focus than real-world application. For example, most literature presents algorithms which pursue the best cutting measure over a single sheet. However, actual industrial applications also value manufacturing efficiency. If a raw material is not particularly expensive, then quickly implementing a cutting layout becomes particularly important. In other words, if the cutting ratio is close to optimal, it is preferable to have as few cuts as possible. Many applications prefer to have different patterns tailored over different types of sheets simultaneously. The multi-sheet CSP is very common in real-world applications, yet it doubles the complexity and complicates the computation. For multi-sheet issues, Gilmore Gomory [5,?] and Beasley [7] presented an approach which collected all the different single sheets being used together and treated them as one final large sheet, then cut the resulting sheet. Unfortunately this approach is not practical, since different sheets may not be fully aligned, or because their size does not guarantee that the final sheet will remain a regular rectangle, and pieces may be cut over the common borders of two aligned sheets. Babu Babu presented an improved algorithm for multi-sheet CSP [24]. GA was used to handle a chromosome composed of sheets and pieces, resulting in more feasible solutions. In this study, an improved multi-phase GA algorithm is used to solve multi-sheet 2D CSP for constrained guillotine cutting with a modified objective function. The merits of the GA heuristic algorithm have been widely discussed in the literature. This population-based algorithm consists of all the solutions in the problem domain, and each solution is referred to as a chromosome. Through GA operations such as reproduction, crossover and mutation, new offspring are generated through multiple generations. Since this algorithm adopts the concept of natural selection sometimes referred to as the survival of the fittest concept, chromosomes which better fit the objective function of the problem have a higher survival probability. The mechanism of this algorithm can override the local optimum and converge to a global optimal solution. A modified multi-phase GA is used for optimum solution searching in a 2D constrained guillotine CSP. Research results confirm that this modified algorithm produces efficient results and can reach the optimum for other benchmark problems. Literature has also shown that the multi-phase approach can improve diversification for global search and enhance convergence [25]. The search procedure starts from a population in the specified domain. Each chromosome contains the required evolutionary information. Offspring evolve through reproduction, crossover and mutation procedures. The objective function is evaluated during each generation, and global searching obtained the possible optimum solution for the simulation.

3 Models

As this study focuses on a 2D CSP with constrained and guillotine cutting, the most common mathematic model is to minimize the waste area. Parada et al [18] presented a model for a single sheet case. Let the sheet have a dimension of L by W, with pieces $P_1,\ P_2,\ \ldots,\ P_n$ with dimensions of $l_1 \times w_1,\ l_2 \times w_2,\ l_n \times w_n$. Each Pi can be cut from the sheet for x_i pieces, but no more than bi pieces (the constraint condition). Thus, the mathematic model (fitness function) is shown as follows:

$$\begin{aligned}&\text{Minimize}\quad LW - \sum_{i=1}^{n} l_i w_i x_i.\\&\text{subject to}\quad 0 \le x_i \le b_i, \qquad (1)\\&0 \le i \le n \text{ integer},\ \forall i.\end{aligned}$$

This model can also be presented as a normalization form, divided by the area of the sheet to the original equation so that the fitness value can be between 0 and 1 (or represented as a percentage) as shown in the following:

$$\text{Minimize}\quad \frac{LW - \sum_{i=1}^{n} l_i w_i x_i}{LW}. \qquad (2)$$

The formula presented is very straightforward, since it represents trim loss. However, Goncalves [20] pointed out that this model poorly captures the potential improvement for a solution. In his research, two cases with the same trim loss were presented; his original figures are shown as Figure 1. From the figure above, one can tell these two different patterns have the same cutting ratio (both values

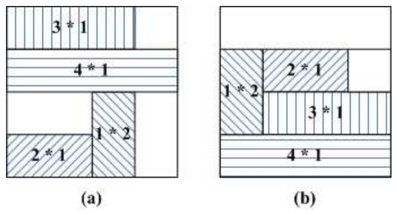


Figure 1: Different cutting patterns with the same cutting ratio.

are 5 according to equation (1), or 0.3125 according to equation (2)). However, the cutting patterns are closer to each other in Figure 1b than they are in Figure 1a, and the uncut space sits together. Clearly, if the unused spaces are widely dispersed, it would be difficult to continue cutting or for further utilization, even though the trim/loss ratio is the same. Thus, Goncalves presented a modified trim loss as follows:

$$\begin{aligned}&\text{Minimize}\quad \frac{LW - \sum_{i=1}^{n} l_i w_i x_i}{LW} - K \times \frac{small\_piece}{LW} \times \frac{largest\_ERS}{LW},\\&\text{subject to}\quad 0 \le x_i \le b_i, \qquad (3)\\&0 \le i \le n \text{ integer},\ \forall i.\end{aligned}$$

ERS represents the Empty Rectangular Space, and K is a constant between $[0.01, 0.1]$. This model takes the layout of the cutting pattern into consideration, and can better evaluate continuous unused space. This formula is thus a better model than the traditional one. However, since real-world applications generally consist of multiple sheets, we extended and modified the model to minimize the following equation:

$$\begin{aligned}&\sum_{i=1}^{S}\Bigg(\frac{L_iW_i - \sum j = 1^P l_j w_j x_j}{TotalArea}\\&-K \times \frac{small\_piece_i}{TotalArea} \times \frac{largest\_ERS_i}{TotalArea}\Bigg)\\&\text{where } TotalArea = \sum_{i=1}^{S} L_i W_i\\&\text{Subject to}\quad 0 \le x_i \le b_i, \qquad (4)\\&0 \le i \le n \text{ integer},\ \forall i\end{aligned}$$

In this extended model, the value of K is set as 0.03 as Goncalves suggested and $small\_piece_i$ is the smallest piece after rectangle sheets are cut. The largest ERS after rectangle sheets are cut is $largest\_ERS_i$, while $TotalArea$ is the sum of all the rectangular sheets. This model will also serve as the objective function in our multi-phase heuristic algorithm.

4 Heuristic Search Algorithm

Research has shown the heuristic search approach is the best approach for CSP. Heuristic searches are global in their approach, most commonly consisting of genetic and evolutionary algorithms. The parameters used for the algorithms include population size, number of generations, solution space, and number of decision variables. Nevertheless, as stated earlier, a global search algorithm is usually computationally expensive, requiring a high population and a large number of generations for convergence to global optimization. The major weakness of heuristic algorithms is efficiency, and research into this issue has become widespread. Many approaches in genetic algorithms have attempted to improve convergence and efficiency by providing modified mutation and crossover operators [26]. Yet it remains clear that no matter which different operators are utilized, the essential computation still relies on an evaluation of an objective function based on a large population size. For the single sheet case, the algorithm only needs to try different orders of the pieces for an optimal solution. However, the cutting order of the sheet needs to be considered for the multi-sheet case. The combination of considering the order of sheets and pieces makes this type of CSP more complex. This is especially true when the number of sheets and pieces increase, and the complexity of the search increases accordingly. Obtaining the best cutting ratio within a reasonable computation time is vital to many real-world applications. The CSP process outlined in this research includes the following procedures: encoding, population initialization, the guillotine cut processing, fitness function evaluation, reproduction, crossover and mutation. Each procedure is described as follows:

A. Encoding

Let set S and P represent the collection of sheets and pieces respectively, i.e., $S = S_i\{w,h,c_i\}, P = P_j\{w,h,c_i\}$. For example, let $S = \{\{200,\ 150,\ 1\},\ \{200,\ 200,\ 1\}\}$ indicate a set that has one sheet 200 by 150, and another sheet which is 200 by 200.

$$P = \{\{100,\ 50,\ 2\},\ \{65,\ 50,\ 2\},\ \{70,\ 50,\ 1\},\ \{50,\ 50,\ 4\},\ \{50,\ 100,\ 4\},\ \{75,\ 40,\ 5\},\ \{25,\ 50,\ 4\}\}$$

indicating that there are 7 different patterns (2 pieces of P_1, 2 pieces of P_2, 1 piece of P_3, 4 pieces of P_4, 4 pieces of P_5, 5 pieces of P_6 and 4 pieces of P_7 which need to be trimmed from sheets S_1 and S_2). This is demonstrated in Figure 2. Before the algorithm begins, the encoding procedure will assign code to sheets and pieces to form an individual chromosome. The first part of the chromosome consists of the codes of sheets, and the second part consists of the codes of the pieces. The code for the initial sheet or piece begins from 0 and consequently increases by 1 for the next sheet or piece. The following two tables illustrate the details. The possible chromosomes can be illustrated in Figure 3 after the encoding process. Each different chromosome will yield a different fitness value accordingly.

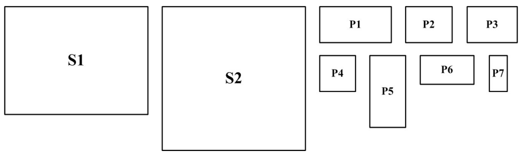


Figure 2: Encoding sample for the sheets and pieces.

Table 1: Encoding for the sheets

S	Dimension	count	Code
S_1	200×150	1	0
S_2	200×200	1	1

Table 2: Encoding for the pieces.

P	Dimension	count	Code
$P1$	100×50	2	0, 1
$P2$	65×50	2	2, 3
$P3$	70×50	1	4
$P4$	50×50	4	5, 6, 7, 8
$P5$	50×100	4	9, 10, 11, 12
$P6$	75×40	5	13, 14, 15, 16, 17
$P7$	25×50	4	18, 19, 20, 21

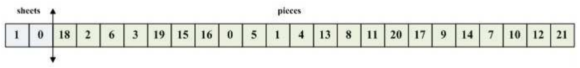


Figure 3: A chromosome consisting of sheets and pieces for simulation.

B. Initial Population

This modified algorithm is derived from the population-based heuristic approach of the original genetic algorithm. All the chromosomes consist of the entire population. Each chromosome is encoded randomly. Since population size plays an important role in determining solution quality as well as computational efficiency, the population size is set as 30∼40 of the length of a chromosome in this study.

C. Guillotine Cut Process

When a piece is guillotine cut from a sheet, one or two possible empty spaces will remain. Each empty space is referred as ERS (Empty Rectangular Space). Based on different cutting directions (horizontal or vertical), the newly created ERS can be either vertical or horizontal if the piece is smaller than the original sheet. If the width (or length) of the piece is the same as the original sheet, then the cutting result would be opposite [20].

All the ERS are stored in an ERS list, and the algorithm will search for suitable ERSs to continue cutting. If there is more than one candidate ERS to choose from, priority is determined by which of the candidates is closer to area of the cut piece and closer to the lower-left corner of the sheet (or the desired location the user has predefined). When the content of a chromosome is determined, applying this guillotine cut process to the chromosome will yield a cutting pattern. Figure 4 details the optimum cutting pattern for the chromosome listed in Figure 3 as an example.

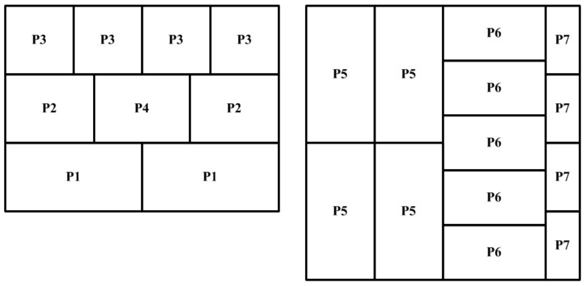


Figure 4: Optimum cutting pattern.

D. Heuristic Placement Method for Multiple Sheets

The algorithm searches the ERS list to find a proper ERS for a particular piece which needs to be cut. After the cut, the ERS list will be updated. Each cut will result in one or two ERS additions to the list. This process will continue until all the required pieces are cut or no space remains. The guillotine cutting algorithm is described in the following pseudo-code:

```
Let Si be the sheet to be cut.
Let NS be the number of sheets

FOR i = 1 TO NS REPEAT {
   Let Pj be the piece which wants to cut from Si.
  Let NP be the number of pieces.

  FOR j = 1 to NP REPEAT {
    IF Pj doesnt belong to any sheet &
        number of ERSlists > 0
   {
       Let ERS*k be the available ERS of ERSlists,
        which Pj fits and the ERS*k is closer to
        the Bottom-Left corner of Si.

        IF ERS*k is not empty {
            Place Pj at the bottom left corner of ERS*k.
         Update ERSlists of available ERS using
                     Guillotine Cut Process.
        }
    }
  }
}
```

E. Reproduction

The population is evaluated to the objective function (the mathematic model) after the first initialization. The GA algorithm will produce a new population by generating new offspring in each iterations. This study uses the elitist strategy and selects the top 15

F. Crossover

The crossover selects two chromosomes to produce a new chromosome which inherits the genes from the selected chromosomes. One sets a probability to decide whether the crossover procedure should take place. This probability is referred to as the crossover rate, and is generally set between 0.6∼0.9 [24]. Even though the one-point and two-point crossover are easy to implement, a uniform crossover operation is adopted in this study [27, ?]. The crossover procedure is individually applied to a given sheet and piece in the chromosome, and each bit in the chromosome is referred to as a gene. This crossover process is described as follows: Suppose there are two parents Chromosomes A and B are selected to create a new chromosome. Randomly pick one gene from A or B. If this selected gene has not been chosen before, put this gene to the new chromosome. If this gene has already appeared in the new chromosome, then select a gene from the other parent chromosome. If this selected gene does not appear in the crossover chromosome, then assign it to the crossover chromosome. If not, leave the position of the gene in the crossover chromosome blank to make sure the genes in the crossover chromosome will not be duplicated. This bitwise process continues until all the genes in the parent chromosomes are processed. Then the blank positions are filled with those unselected genes to form the final child chromosome. Figure 5 illustrates this process.

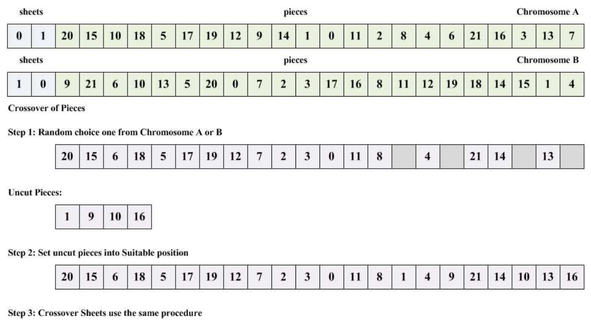


Figure 5: The uniform crossover procedure for a new chromosome

G. Mutation

A mutation operator is generally used to change the gene(s) of a chromosome, and is usually applied to some individuals to provide diversity for the simulation population. The probability of mutation is generally set to a small value. Instead of using a traditional gene-by-gene mutation approach with a very small probability for each generation, this study follows the mutation procedure described in Goncalves work 0. That is, when the probability reaches the mutation rate, then the worst 15% of the fitness value associated with the chromosome will be regenerated for the next generation.

H. Multi-Phase Genetic Algorithm (MPGA

The literature has discussed the limitations of heuristic algorithms for several decades. It identifies several critical factors for an algorithm to converge to the right solution [27, ?]. These include population size, number of generations, solution space, and number of decision variables. Because the complexity of the problems varies, no particular set of parameters can be applied to all the problems for optimal results every time. Poorly chosen parameters can lead the search to a local optimal solution. This problem compounds when the number of decision variables increases the solution space expands dramatically. The idea behind MPGA is to improve the diversification capability of the basic GA algorithm. It implements the basic GA algorithm *m* times (phases). The best φ solutions from the last generation in a phase will automatically become members in the initial population for the next phase, and the remaining members in the initial population will be generated by a GA crossover operator. This multi-phase design will not only improve the diversification capability of the basic GA algorithm, but also guide the search to good solution regions. The pseudo-code of the multi-phase GA algorithm is presented below.

```
For multi-phase=1 to m do
 if multi-phase =1
    Initialize all  populations
 else
Reset all parameters to initial conditions
parents = best  solutions from previous phase +
randomly generated (-) solutions
 For generation :=1 to maxgen do //perform the GA
 Evaluate the fitness values for the new population
and save the best  solutions
 Perform the Selection, Crossover and Mutation to yield new populations
 End do
End do
Output the best solution
```

5 Computational Result

This study presents the multi-phase Genetic Algorithm (MPGA) and applies it to a 2D guillotine-constrained CSP problem. The outer iteration number for the MPGA is set as 5, and the interior iteration is set as 50. The population size is set as 30 times the length of a chromosome. The crossover rate is 0.9, 15% of the top chromosomes are selected for the next generation, and 15% of the remaining chromosomes are selected for mutation. The stop criteria is set to either when all iterations are performed or if the best optimal solution is not updated for 2 outer MPGA iterations. The test cases presented by Cung are adopted as the test problems for this study. Cung et al. [29] used branch and bound methods to test 17 mid-scale and 9 large-scale problems. The test problems of A1s, A2s, A3, A4 and A5 were from Hifi [30]; STS2s and STS4s were from Tschoke [31]; CHL1s, CHL2s, CHL3s, CHL4s, CHL5, CHL6 and CHL7 were generated randomly, and Hchl3s, Hchl4s, Hchl5s, Hchl6s, Hchl7s and Hchl8s were more complicated and large-scale benchmarks. The computation results were compared with those of Valdes [14]. Valdes et al. utilized several approaches to test guillotine cutting problems. Their first approach was a constructive algorithm (CONS), followed by a GRASP (greedy adaptive search procedure) method. A path relinking approach was then added to GRASP, and the TABU search algorithm of 500 iterations was performed.

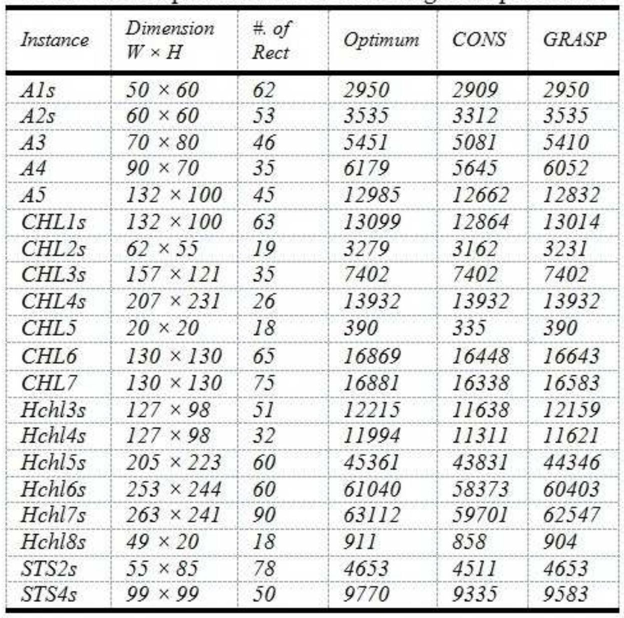
Table 3: Summary of computational results on Cung constrained instances

	Optima (out of 20)	Average ratio	Minimum ratio
CONS	2	0.956160123	0.858974359
GRASP	6	0.990184623	0.968901117
GRASP/PR	6	0.993078342	0.97873937
TABU500	11	0.998565177	0.992316136
Multi-phase GA	7	0.992984569	0.979446512

The computational results are shown in Table 4a and Table 5b.

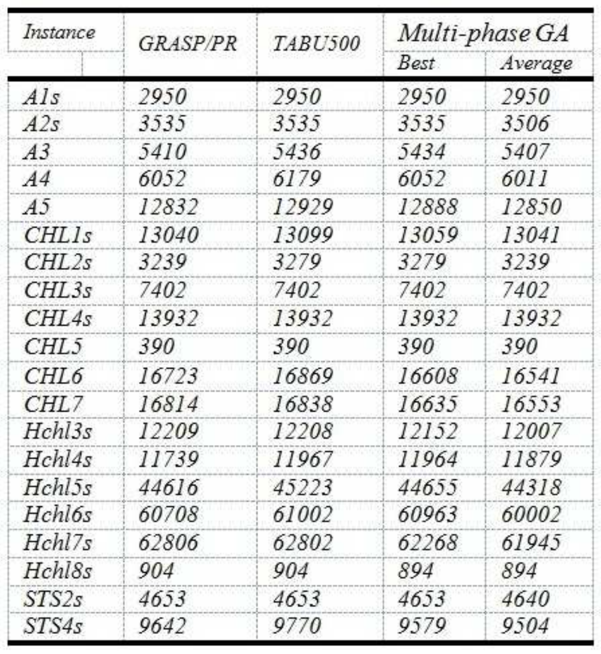
The first four solutions were presented by Valdes 0. The results from this study are listed under the multi-phase GA. Each benchmark was simulated 10 times, five times cutting vertically first, and five times cutting horizontally first. Best and average results are listed for each case. The computational results are summarized in Table 3. Notice that TABU500 had the best performance, reaching 11 optima simulations, and the best results for average and minimum ratios. In our results, multi-phase GPA is ranked as second with 7 optima, with the second best simulation on the average and minimum ratios, yet it is less computationally intensive than TABU500.

Table 4a. Computational results of Cung et al. problems.



Instance	*Dimension W × H*	*#. of Rect*	*Optimum*	*CONS*	*GRASP*
A1s	*50 × 60*	*62*	*2950*	*2909*	*2950*
A2s	*60 × 60*	*53*	*3535*	*3312*	*3535*
A3	*70 × 80*	*46*	*5451*	*5081*	*5410*
A4	*90 × 70*	*35*	*6179*	*5645*	*6052*
A5	*132 × 100*	*45*	*12985*	*12662*	*12832*
CHL1s	*132 × 100*	*63*	*13099*	*12864*	*13014*
CHL2s	*62 × 55*	*19*	*3279*	*3162*	*3231*
CHL3s	*157 × 121*	*35*	*7402*	*7402*	*7402*
CHL4s	*207 × 231*	*26*	*13932*	*13932*	*13932*
CHL5	*20 × 20*	*18*	*390*	*335*	*390*
CHL6	*130 × 130*	*65*	*16869*	*16448*	*16643*
CHL7	*130 × 130*	*75*	*16881*	*16338*	*16583*
Hchl3s	*127 × 98*	*51*	*12215*	*11638*	*12159*
Hchl4s	*127 × 98*	*32*	*11994*	*11311*	*11621*
Hchl5s	*205 × 223*	*60*	*45361*	*43831*	*44346*
Hchl6s	*253 × 244*	*60*	*61040*	*58373*	*60403*
Hchl7s	*263 × 241*	*90*	*63112*	*59701*	*62547*
Hchl8s	*49 × 20*	*18*	*911*	*858*	*904*
STS2s	*55 × 85*	*78*	*4653*	*4511*	*4653*
STS4s	*99 × 99*	*50*	*9770*	*9335*	*9583*

Table 5b. Computational results of Cung et al. problems.



Instance	GRASP/PR	TABU500	Multi-phase GA	
			Best	Average
A1s	2950	2950	2950	2950
A2s	3535	3535	3535	3506
A3	5410	5436	5434	5407
A4	6052	6179	6052	6011
A5	12832	12929	12888	12850
CHL1s	13040	13099	13059	13041
CHL2s	3239	3279	3279	3239
CHL3s	7402	7402	7402	7402
CHL4s	13932	13932	13932	13932
CHL5	390	390	390	390
CHL6	16723	16869	16608	16541
CHL7	16814	16838	16635	16553
Hchl3s	12209	12208	12152	12007
Hchl4s	11739	11967	11964	11879
Hchl5s	44616	45223	44655	44318
Hchl6s	60708	61002	60963	60002
Hchl7s	62806	62802	62268	61945
Hchl8s	904	904	894	894
STS2s	4653	4653	4653	4640
STS4s	9642	9770	9579	9504

6 Conclusions

In this study, we developed an efficient cutting stock algorithm to handle problems commonly seen in real-world industrial applications. Heuristic algorithms have been widely applied to cutting stock problems, and our study presents modified multiphase genetic algorithm. Traditional benchmark problems have been used to test validity and efficiency. The computational results have shown that this approach is superior to many other algorithms and requires fewer computational resources. Cutting stock has many applications; the panel saw machine is a typical example used in wood, glass or other industrial production. Since the cost of the stock sheet may not be extremely expensive, it is important to understand that long periods of time waiting for only modest improvement in a cutting ratio will not be tolerated. This paper presents an efficient algorithm to achieve fast convergence with high levels of quality. Additionally, this paper also presents an approach to handle multi-stock sheets common to real-world applications. This study has extended the traditional encoding process and adds processing for multiple sheets to the evolution process. While most literature uses the minimized waste area for the objective function for the heuristic simulation, research has shown that the traditional objective function for CSP is flawed, and a modified trim loss equation was presented. Our study also extended this equation to multi-sheet stock cases. Our algorithm has been tested with various benchmarks for constrained guillotine cutting. The results show this multiphase GA is a robust, accurate and efficient algorithm.

Acknowledgement

The authors are grateful to the National Science Council of Taiwan for its financial support (project number NSC 96-2622-E-260-001-CC3).

References

[1] Hinxman, A. I., The trim-loss and assortment problems: A survey, European Journal of Operational Research, 1979, **5**, 8-18.

[2] Kantorovich, L. V., Mathematical methods of organizing and planning production, Management Science, 1960, **6**, 366-422.

[3] Mahmoud, A. F., Samia, M., Eid, S. and Bahnasawi, A., Genetic algorithms for solving 2D cutting stock problem, Proceedings of the 46th International Midwest Symposium on Circuits and Systems, 2003, **2**, 956-960.

[4] Lin, C. C. and Yu, C. S., A heuristic algorithm for the three-dimensional container packing problem with zero unloading cost constraint, Systems, Man and Cybernetics, 2006. SMC '06, **6**, 4637-4642.

[5] Gilmore, P. C., Gomory, R. E., Multistage cutting stock problems of two and more dimensions, Operations Research, 1964, **13**, 94-120.

[6] Gilmore, P. C. and Gomory, R. E., The theory and computation of knapsack functions, Operations Research, 1966, **14**, 1045-1074.

[7] Beasley, J. E., Algorithms for unconstrained two-dimensional guillotine cutting, Journal of the Operational Research Society, 1985, **36**, 297-306.

[8] Christofides, N., Whitlock, C., An algorithm for two-dimensional cutting problems, Operations Research, 1977, Vol. **25**, 30-44.

[9] Hifi, M., Ouafi, R., Best-first search and dynamic programming methods for cutting problems: The cases of one or more stock plates, Computers & Industrial Engineering, 1996, **32**, 187-205.

[10] Viswanathan, K. V., Bagchi, A., Best-first search methods for constrained two-dimensional cutting stock problems, Operations Research, 1991, **41**, 768-776.

[11] Wang, P. Y., Two algorithms for constrained two-dimensional cutting stock problems, Operations Research, 1982, **32**, 573-586.

[12] Vasko, F. J., A computational improvement to Wangs two-dimensional cutting stock algorithm, Computers and Industrial Engineering, 1989, **16**, 109-115.

[13] Oliveira, J. F. and Ferreira, J. S., An improved version of Wangs algorithm for two-dimensional cutting problems, European Journal of Operational Research, 1990, **44**, 256-266.

[14] Valdes, R. A., Parajon, A. and Tamarit, J. M., A tabu search algorithm for large-scale guillotine (un)constrained two-dimensional cutting problems, Computers & Operations Research, 2000, **29**, 925-947.

[15] Valdes, R. A., Marti, R., Parajon, A., Tamarit, J. M., Grasp and path relinking for the two-dimensional two-staged cutting stock problem, INFORMS Journal on Computing, 2006, **19**, 1-12.
[16] Lai, K. K., Chan, J. W. M., Developing a simulated annealing algorithm for the cutting stock problem, Computers and Industrial Engineering, 1996, **32**, 115-127.
[17] Faina, L., An application of simulated annealing to the cutting stock problem, European Journal of Operational Research, 1999, **114**, 542-556.
[18] Parada, V., Sepulveda, M., Solar, M., Gomes, A., Solution for the constrained guillotine cutting problem by simulated annealing, Computers & Operations Research, 1996, **25**, 37-47.
[19] Wu Tai-Hsi, Wu Yi-Hua, Chang Chin-Ju. Solving Two-Dimensional Packing Problems by Using a Genetic Algorithm. Journal of Science and Engineering Technology, 2006, **2**, 75-83.
[20] Goncalves, J. F., A hybrid genetic algorithm-heuristic for a two-dimensional orthogonal packing problem, European Journal of Operational Research, 2007, **183**, 1212-1229.
[21] Pal, L., A genetic algorithm for the two-dimensional single large object placement problem, Proceedings of the 3rd Romanian-Hungarian Joint Symposium on Applied Computational Intelligence, 2006, 253-260.
[22] Liu, D. and Teng, H., An improved BL-algorithm for genetic algorithm of the orthogonal packing of rectangles, European Journal of Operational Research, 1997, **112**, 413-420.
[23] Jakobs, S., On Genetic algorithms for the packing of polygons, European Journal of Operational Research, 1993, **88**, 165-181.
[24] Babu, A. R., Babu, N. R., Effective nesting of rectangular parts in multiple rectangular sheets using genetic and heuristic algorithms, International Journal of Production Research, 1999, **37**, 1625-1643.
[25] Yu TY, Tsai C, Huang HT. Applying simulation optimization to the asset allocation of a propertycasualty insurer. 2010, European Journal of Operational Research, **207**, 499-507.
[26] Wang RL, Okazaki K. An improved genetic algorithm with conditional genetic operators and its application to set-covering problem, 2007, Soft Computing, **11**, 687694.
[27] Zbigniew, M., Genetic Algorithms + Data Structures = Evolution Programs. 1996, Springer, New York.
[28] Randy L. Haupt, Sue Ellen Haupt, Practical Genetic Algorithms, Second Edition, 2004, John Wiley & Sons, Inc.
[29] Cung, V. D., Hihi, M., Cun, B. L., Constrained two-dimensional cutting stock problems a best-first branch-and-bound algorithm, International Transactions in Operational Research, 1999, **7**, 185-210.
[30] Hifi, M., An improvement of Viswanathan and Bagchi's exact algorithm for constrained two-dimensional cutting stock, Computers & Operations Research, 1997, **24**, 727-736.
[31] Tschoke, S., Holthofer, N., A New Parallel Approach to the Constrained Two-Dimensional Cutting Stock Problem, in A. Ferreira and J. Rolim (Eds), Parallel Algorithms for Irregularly Structured Problems, Springer-Verlag, Berlin, Germany, 1995, 285300.

TzuYi Yu received his MS and PhD degrees in computational engineering from Mississippi State University, USA. He is currently an associate professor in the department of Information Management, National Chi Nan University in Taiwan. His research interests include computer simulation and optimization.

Jiann-Cherng Yang received his MS degree in computer science and PhD degree in computational engineering from Mississippi State University, USA. He is currently an associate professor in the department of Applied Mathematics, Feng Chia University in Taiwan. His research interests include Computer Geometry and Numerical Analysis.

Yeong-Lin Lai received his PhD degree from the Institute of Electronics, National Chiao Tung University, Taiwan. Dr. Lai is currently a full professor with the department of Mechatronics Engineering and the Graduate Institute of Integrated Circuit Design, National Changhua University of Education. His research interests include MEMS/NEMS, RFIC, RFID.

Chih-Cheng Chen received his PhD degree from the Department of Mechatronics Engineering, National Changhua University of Education in 2011. He is currently an associate professor in Department of Industrial Engineering and Management in National Chin-Yi Institute of Technology. His research interests include the RFID applications and system design.

Han-Yu Chang received his MS degree in department of Information Management, National Chi Nan University in Taiwan. He is currently a software engineer in a design company.