Journal of Intelligent Manufacturing, 14, 209–218, 2003
© 2003 Kluwer Academic Publishers. Manufactured in The Netherlands.

A genetic algorithm approach for the cutting stock problem

GODFREY C. ONWUBOLU[1] and MICHAEL MUTINGI[2]

[1]*Department of Engineering, The University of the South Pacific, P.O. Box 1168, Suva, Fiji*
Email: onwubolu_g@usp.ac.fj
[2]*Olivine Industries, Zimbabwe*

Received December 2001 and accepted January 2002

In this paper, a genetic algorithm approach is developed for solving the rectangular cutting stock problem. The performance measure is the minimization of the waste. Simulation results obtained from the genetic algorithm-based approach are compared with one heuristic based on partial enumeration of all feasible patterns, and another heuristic based on a genetic neuro-nesting approach. Some test problems taken from the literature were used for the experimentation. Finally, the genetic algorithm approach was applied to test problems generated randomly. The simulation results of the proposed approach in terms of solution quality are encouraging when compared to the partial enumeration-based heuristic and the genetic neuro-nesting approach.

Keywords: Cutting stock, optimization, genetic algorithms

Introduction

The high cost of raw materials in manufacturing results in the need for minimization of wastes or scraps in using these materials. The problem of allocating two-dimensional patterns arises frequently in industrial applications that require determining how a set of these shapes will fit onto a large stock sheet of finite dimensions such that scraps are minimized. The cutting stock problem is frequently encountered in the flat metal sheet works in industry.

The basic attributes generally used to classify the cutting stock problem include dimensionality, the shape of patterns and the number of pieces. Dimensionality relates to whether the pieces being cut are either one-dimensional or two-dimensional. The shape of the patterns may be irregular or regular.

The two-dimensional rectangular pattern cutting stock problem is NP-hard (Garey and Johnson, 1979). This roughly means that the problem can be optimally solved by exact-algorithms, which have computation times that grow exponentially or may be far from the optimal solution when solved by heuristic methods. Over the years, two main approaches have emerged: heuristics and approaches based on linear programming relaxation. Most of the approaches proposed for the solution for the cutting stock problems before 1990 depended mainly on optimization algorithms and heuristic rules in order to reduce the search space required by the optimization model (Dagli and Poshyanonda, 1997). This explains the reason for scarcity of applications of optimization models in practice. Gilmore and Gomory (1963a, b; and 1966) gave one of the first formulations of the cutting stock problem. Later, various heuristics were proposed for solving the problem (Hains and Freeman, 1970; Garey and Johnson, 1979; Wang, 1983). Chauny *et al.* (1991) employed a two-phase technique. The strategic phase relaxes the global problem to a one-dimensional cutting stock problem and solves it using linear programming, while the tactical phase is a recursive algorithm based on repeated knapsack operations and other heuristics. For more information on several heuristics applied to the cutting stock problem, see Berkey and Wang (1987).

Since the 1990s, there has been a shift towards the application of search algorithms that are capable of solving combinatorial optimization problems. For example, the use of neural networks as the heuristics to reduce the search space of the optimization algorithm for NP-complete problems was proposed by Spears and De Jong (1990). Poshyanonda *et al.* (1992) proposed the integration of neural networks and optimization approaches to solve the rectangular cutting stock problem. Some other useful literature regarding the use of genetic algorithm for solving cutting and packing problems include Paradza Daza *et al.* (1995) and Kroger (1995). For more references to different approaches and cutting patterns for the cutting stock problems, interested readers may see Babu and Babu (1999).

As earlier mentioned, heuristic algorithms have solutions far from the optimal solution in most cases. Most heuristic algorithms are tailored to specific problems and are, therefore often not robust when applied to some other class of problems. On the other hand, adaptive memory procedures such as artificial neural networks, together with tabu search (Glover, 1989), genetic algorithms (Goldberg, 1989) and ant systems (Dorigo and Gambardella, 1997) are new techniques that have emerged for solving intractable problems. These techniques have been identified as having the potential to solve combinatorial optimization problems with near optimal or optimal solutions (Fleurent and Ferland, 1994). In this work, we employ genetic algorithms to the cutting stock problem. The results are compared with those obtained by the partial enumeration-based heuristic employed by Benati (1997) and the genetic neuro-nesting approach employed by Poshyanonda *et al.* (1992).

Problem formulation

The terminology used in the problem formulation of the cutting stock problem is defined as follows:

i	Each roll
j	Each piece
m	Number of rolls
n	Number of pieces
h_j	Height of each piece
w_j	Width of each piece
d_j	Demand of each piece
W_i	Width of roll
H	Height of the first guillotine cut from the roll base
w_1	Width of first item
h_1	Height of first item
$n^{\max}$	Maximum number of first item horizontally
$m^{\max}$	Maximum number of first item vertically
W^*	Width of waste rectangle
H^*	Height of waste rectangle
n^*	Number of first item that can be cut horizontally from the waste area
m^*	Number of first item that can be cut vertically from the waste area
n_w^*	Number of item j that can be cut horizontally from the waste area
m_h^*	Number of item j that can be cut vertically from the waste area
x_1	Binary variable that denote rotation non-rotation of first item (1 or 0)
y_j	Binary variable that denotes rotation/non-rotation of item j (1 or 0)
$\lfloor\ \rfloor$	Rounding down to the preceding integer number
$\lceil\ \rceil$	Rounding up to the succeeding integer number

The cutting stock problem is formulated as follows: a set of rectangular pieces must be cut from a set of sheets, so as to minimize total waste. The pieces are requested in large quantities, of rectangular shape and the set of sheets are long rolls of materials. Each piece j has a height h_j and a width w_j and is requested in a certain quantity d_j. Each of these pieces can be rotated and cut from a set of roll. Each roll i has a width w_i and virtually infinite height. The assumption made is that the size of the items is relatively large in comparison to the width of the rolls. Consequently, only cutting patterns made by at most two different items are considered, while the items could vary in number.

Consider first item, of dimensions (w_1, h_1). Piece 1 can be cut from the roll of width W_i obtaining a maximum number of pieces $n^{\max} = \lfloor W_i/w_1 \rfloor$ horizontally. Similarly, $m^{\max} = \lfloor H/h_1 \rfloor$ vertically. Therefore, horizontally the number n^* of first item that can be cut lies within the range $1 \leq n^* \leq n^{\max}$. Similarly vertically the number m^* of first item that can be cut lies within the range $1 \leq m^* \leq m^{\max}$.

For any n^* and m^*, a waste rectangle (see Fig. 1(a)) is obtained. The dimensions of the waste rectangle are W^* and H^* given as:

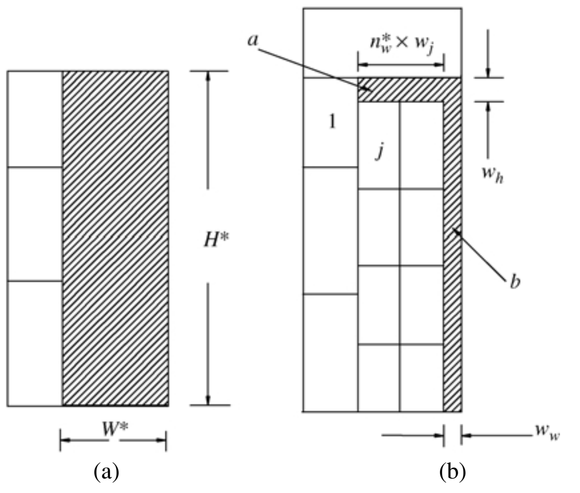

Fig. 1. Cutting stock pattern showing piece 1, piece j, and waste.

$$W^* = W_i - n^* \cdot w_i; \quad H^* = m^* \cdot h_1 \qquad (1)$$

Once first item is fixed, we attempt to fill the rectangular waste area whose dimensions are W^* and H^* with one item j. For each j there would be $n_w^* = \lfloor W^*/w_j \rfloor$ pieces vertically (see Fig. 1(b)).

Waste (a) has height $w_h = (H^* - m_h^* \cdot h_1)$ and width $n_w^* \cdot w_j$. Waste (b) has height H^* and width $w_w = (W^* - n_w^* \cdot w_j)$. The absolute waste is therefore given as

$$A(n^*, m^*, W_i, j) = \left(W^* - n_w^* \cdot w_j\right) \cdot H^* + (H^* - m_h^* \cdot h_j) \cdot \left(n_w^* \cdot w_j\right) \qquad (2)$$

We define

$$x_1 = \begin{cases} 1 & \text{if item 1 is rotated} \\ 0 & \text{if otherwise} \end{cases} \qquad (3)$$

$$y_j = \begin{cases} 1 & \text{if item } j \text{ is rotated} \\ 0 & \text{if otherwise} \end{cases} \qquad (4)$$

$$A(m^*, x_1, W_i) = \begin{cases} W_i h_1 m^* & \text{if item 1 is not rotated} \\ W_i w_1 m^* & \text{if otherwise} \end{cases} \qquad (5)$$

Using Equations 2 and 5, we define the objective function as the smallest percentage waste $A_{\min}$ given by

$$A_{\min} = \min_{n^*, m^*, x_1, i, j, y_j} \frac{A(n^*, m^*, x_1, W_i, j, y_j)}{A(m^*, x_1, W_i)} \qquad (6)$$

subject to

$$\min(\lceil d_1/n_1 \rceil, \ \lceil d_j/n_j \rceil) \qquad (7)$$

Equation 5 is the overall area of the rectangle from which the items are cut. Equation 6 is the objective function that minimizes percentage waste of the cutting stock problem. It is the ratio of the absolute waste to the overall area. Equation 7 is the condition to be met when $n_1 = n^* \cdot m^*$ pieces of first item and n_j pieces of item j for demands d_1 and d_j are cut as many times as necessary in order to satisfy the requests for pieces.

The genetic algorithm approach based on Equations 6 and 7 is effective because probabilistically, it is relative easy to find cutting patterns having a low percentage of waste by simply rotating and matching at best just two pieces at a time from all available rolls. Several other methods for the cutting stock problem employ two phases or include specific heuristic algorithms for the generation of a nested pattern. The genetic algorithm approach presented here is straightforward; it utilizes integer coded-strings and genetic operations for matching at best just two pieces at a time from all available rolls.

Genetic algorithm for the cutting stock problem

Genetic algorithm (GA) is a meta-heuristic method based on the mechanics of copying strings according to their objective function values and swapping partial strings to generate successive populations that improve over time. Its distinctive feature is the use of probabilistic genetic operators as tools to guide a search toward regions of the search space with likely improvement.

The robustness of the GA scheme is attributed to the fact that parameter-set of the optimisation problem is coded as a finite-length string over some finite alphabet, and solution search is carried out from a population of strings in the population. Consequently, climbing of many peaks is done in parallel, thereby reducing the probability of finding a false peak; there is no need for auxiliary information except for the objective function information; and probabilistic transition rules are used.

Methodology

The proposed method utilizes genetic algorithm to generate a nested pattern in the set of sheets for the considered parts with the objective of optimally utilizing the sheet material. Among the parts considered, the part having the maximum surface area is identified. This part is known as the ''first pattern'', which conceptually is then translated to the bottom-left corner of the first rectangular roll (sheet). The procedure seeks another pattern to be attached to the ``first pattern''. We refer to this pattern as an ``attached pattern''. The ``first pattern'' is labeled $s$ while the ``attached pattern'' is labeled r in the first and second elements of a typical coded string in genetic algorithm. If s is rotated, the third element of the coded genetic string is filled with a value 1, otherwise it is filled with a value 0. If r is rotated, the fourth position of the coded genetic string is filled with a value 1, otherwise it is filled with a value 0. The fifth element of the coded genetic string represents then the roll-number under consideration $([1, n \text{ rolls}])$. The sixth and seventh elements of the coded genetic string represent the number of the ``first pattern'' that can be cut horizontally from the waste area $(n^*)$, and the number of the ``first pattern'' that can be cut vertically from the waste area (m^*), respectively. A typical coded genetic string used in the work reported in this paper is shown as follows:

1	2	3	4	5	6	7
1	3	0	1	1	1	2

In this illustrative example, $s = 1$, and $r = 3$. This means that the ``first pattern'' is the first part or item on the list, while the ``attached pattern'' is the third part or item on the list. The ``first pattern'' is not rotated but the ``attached pattern'' is rotated. The first roll is used. The number of the ``first pattern'' that can be cut horizontally from the waste area, and the number of the ``first pattern'' that can be cut vertically from the waste area are 1 and 2 respectively.

An example of the set of sheets and parts considered in this study, along with their dimensions and coding are shown in Tables 1 and 2. There are two rolls; each having dimensions 11 mm × 7 mm. There are five parts to be cut from the two rolls and the demands or quantities are given in Table 2. A special class of problem exists when the number of rolls equal to one and the demand for each part equal to one. This class of problem is the one solved by Poshyanonda *et al.* (1992). Benati (1997) solved the problem for multiple number of rolls and demands for parts.

Table 1. Available rectangular rolls

Height (mm)	*Width (mm)*	*Quantity*	*Sheet serial number (ID)*
11	7	1	1
11	7	1	2

Table 2. Available rectangular parts

Width (mm)	*Height (mm)*	*Quantity*	*Part serial number (ID)*
4	3	10	1
2	5	10	2
2	3	10	3
2	2	10	4
2	2	10	5

The present genetic algorithm procedure first identifies the largest part, which is the first part in Table 2 and searches through the list of parts {1, 2, 3, 4, 5}, for the ``best'' part to pair up with the ``first (largest) part.'' This ``best'' part is known as the ``attached part,'' which in the illustration is the part with identity (ID) number = 3 or the third part. The new list for the unprocessed parts then becomes {2, 4, and 5}. The GA procedure re-initialises and the process is repeated, leading to the next pair of ``first part'' selected part being 2 and the ``attached part'' being 4. The new list for the unprocessed parts then becomes {5}. Again, the GA procedure re-initializes and the process is repeated, leading to the next pair of ``first part'' selected part being 5 and the ``attached part'' being none. The GA procedure then stops. Table 3 shows the optimum population report for part 3 attached to part 1. Table 4 shows the optimal solutions for the problem described in Tables 1 and 2. Figure 2 shows the nested pattern for the solution shown in Table 4. There are 3 patterns of parts 1 and 3 in pair, 2 patterns from roll 1 and 1 pattern from roll 2. There are 5 patterns of parts 2 and 4 in pair, 2 patterns from roll 1 and 3 pattern from roll 2. There is only one 1 pattern of part 5 from roll 2.

Although, the demand for each part is 10 as shown in Table 2, only 3 quantities of part types 1 and 3, 5 quantities of part types 2 and 4, and 1 quantity of part type 5 can be accommodated for the two rolls of size 11 mm by 7 mm. The total waste is then calculated as:

$$\begin{aligned}\sum \mathit{waste} &= (2 \cdot 7 \cdot 11) - \{3 \cdot [(4 \cdot 3) + (2 \cdot 3)] \\ &\quad + 5 \cdot [(2 \cdot 5) + (2 \cdot 2)] + 1 \cdot [(2 \cdot 2) + 0]\} \\ &= 154 - (54 + 70 + 4) \\ &= 26\,\text{mm}^2\end{aligned}$$

Table 3. Optimum population report for Part 3 attached to Part 1

String							*Objective function*	*Function score*	*Expected count*
1	3	0	1	2	1	2	0.0000	0.1429	5.0000
1	3	0	1	1	1	2	0.0000	0.1429	5.0000
1	3	0	1	2	1	1	0.1429	0.0000	0.0000
1	3	0	0	1	1	2	0.1429	0.0000	0.0000
1	3	1	1	1	1	2	0.1429	0.0000	0.0000
1	3	1	1	1	1	2	0.1429	0.0000	0.0000
1	3	0	1	2	1	1	0.1429	0.0000	0.0000
1	3	0	1	1	1	1	0.1429	0.0000	0.0000
1	3	0	0	1	1	2	0.1429	0.0000	0.0000
1	3	0	0	1	1	2	0.1429	0.0000	0.0000

Table 4. Optimal solutions for Tables 1 and 2 problem

	(s & r)				*(5 & 0)*
	(1 & 3)		*(2 & 4)*		
Part width	4	2	2	2	2
Part height	3	3	5	2	2
Rotation	0	1	1	0	1
(m^*, n^*)	1	2	1	2	3
Roll width		7		7	7
No. patterns		3		5	1

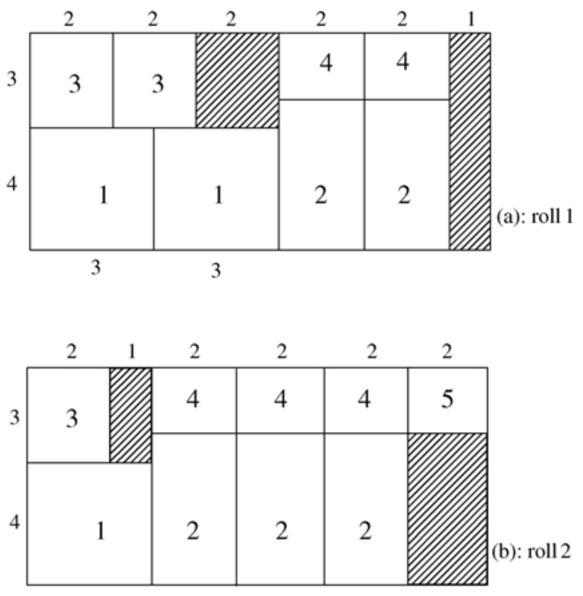

Fig. 2. Nested patterns for Table 4 solution.

therefore

$$\% \text{Waste} = 26/(2 \cdot 11 \cdot 7) = 16.883\%$$

Begin with an initial population

The initial population is a set of strings representing the code of the cutting stock optimization problem generated at random. Initially, a population of *popsize* (number of population) is generated randomly for elements 2, 3, 4, and 5 of each string in the population. For example in the first string of the population in Table 3, the integer values 3, 0, 1, and 2 are generated randomly. Elements 1, 6, and 7 of each string in the population are not generated randomly. Element 1 is the ''first pattern'', elements 6 and 7 of each string in the population are calculated. The definitions of these elements in each string in the population have been sufficiently discussed. As already discussed the GA procedure re-initializes after it finds the best ''attached pattern'' that pairs up with the ''first pattern''.

Selecting superior strings

The coded string of the individual solution is converted into objective function value. Copying strings according to their objective function values means that strings with a higher value have a higher probability of contributing one or subsequent strings in the next population. Liepins and Hilliard (1986) have suggested several strategies in the past. These include the following: probabilistic replacement, crowding strategy, and elitist strategy. In the study reported here, a combination of these strategies has been employed.

Partial bits exchange operator

In the partial bit exchange (PBE) operator, new strings are formed among strings of the temporary new population. PBE involves three steps of randomly choosing two strings from the temporary new population, randomly choosing exchanging sites along the length of the strings, and exchanging bits bounded by the exchanging sites. For a number of different problems solved using GA, the multiple-point PBE was found to yield better results than the single-point scheme (Onwubolu, 2001). The PBE operation results in new strings being formed and enables the GA to jump out of a local optimum into an entirely different feasible population (diversification), hence is a key operator in overcoming the deficiencies of local search algorithms. For the cutting stock problem, a more efficient PBE implemented in the present work is the bit-wise PBE. For each bit, a random number is generated between 0 and 1. For a true condition (random number ≥ 0.5), then a bit-crossover takes place. This is repeated for each bit in the whole length of the string. As an illustrative example, for a *true* condition for bit positions 2 and 4, the following strings are obtained:

Current string 1: 2 4 3 1 5 *Subsequent string* 1: 2 1 3 3 5
Current string 2: 5 1 2 3 4 *Subsequent string* 2: 5 4 2 1 4

The operation has resulted in duplication of 3 and 4, and loss of 4 and 3 in *subsequent string*-1 and *subsequent string*-2, respectively. The random bits alteration operation need to be invoked to rectify the situation, as described in the next subsection. After extensive experimentation, the PBE rates adopted are 0.788 for each operator.

Random bits alteration

Another genetic operator for forming new strings is the random bit alteration (RBA). The operator exchanges any two string-bits chosen at random. RBA alters the neighborhood of strings, thus achieving a local search within a close vicinity of each string in the current population. This operator is responsible for intensification of search for better solutions in the current region of the population. After extensive experimentation, the RBA rates adopted are 0.45 for each operator. For each string, a random number is generated between 0 and 1. If it is *true*, then a random bit alteration takes place. As an illustrative example, for a *true* condition for bit positions 4 and 5, the following strings are obtained:

string 1: 2 1 3 **3*** **5** *new string* 1: 2 1 3 **4** 5
string 2: 5 4 2 1 **4*** *new string* 2: 5 4 2 1 **3**

Two random integers r_1 and r_2 are selected from strings 1 and 2, respectively such that $1 \leq r_1, r_2 \leq n$ (length of the string) and $r_1 \neq r_2$. The elements with string bits designated as r_1 and r_2 are swapped. In this case $r_1 = 4$ and $r_2 = 5$.

Replacement strategy

In every PBE and RBA operation new strings are formed. The new strings may be superior or inferior to existing strings in the current population. This means that inferior strings are to be replaced using a replacement strategy. Several strategies, which have been suggested, include probabilistic replacement, crowding strategy and elitistic strategy (Liepins and Hilliard, 1986). However, it is possible to blend these strategies.

The strings in the temporary new population resulting from PBE operation are evaluated using the objective function values as a performance measure. The number of these strings is usually less than the number of strings in the preceding population. A close comparison of the objective function values is therefore made between the strings in these populations, and those performing better are advanced into the next new population. The difference in the number of strings required is selected randomly from the preceding population using a certain predetermined probability of acceptance.

Stopping criterion

The stopping criterion was set at 100 generations of the GA advancements, which gives optimal solution.

Experimental results

The results of the present work are compared with those obtained by the partial enumeration-based heuristic and also with those obtained by the genetic neuro-nesting approach in order to show the several implementation improvements that we have made on the current GA-version.

Table 5. Percentage waste for Benati (1997) and genetic algorithm heuristic

No.	*Code*	*n*	*m*	*BEN% waste (H + IC)*	*m*	*GA % waste*
1	P1-A1	17	1	10.84	1	7.843
2	P1-A2	17	2	6.33	2	5.171
3	P1-A3	17	3	3.48	3	3.700
4	P1-B1	17	1	6.67	1	7.973
5	P1-B2	17	2	2.06	2	3.235
6	P1-B3	17	3	2.06	3	2.880
7	P2-A1	12	1	2.59	1	3.817
8	P2-A2	12	2	1.46	2	1.886
9	P2-A3	12	3	1.56	3	1.360
10	P2-B1	12	1	3.72	1	5.051
11	P2-B2	12	2	2.04	2	1.917
12	P2-B3	12	3	1.63	3	0.760
13	P3-B1	22	1	9.58	1	9.476
14	P3-B2	22	2	4.84	2	4.390
15	P3-B3	22	3	3.78	3	3.649
16	P3-C1	22	1	12.14	1	9.713
17	P3-C2	22	2	5.83	2	5.151
18	P3-C3	22	3	3.06	3	2.969

Table 6. Improvement of the GA procedure over Benati (BEN)[1]

No.	*Code*	*Solution quality (% waste)*
1	P1-A1	27.65
2	P1-A2	18.31
3	P1-A3	− 6.32
4	P1-B1	− 19.53
5	P1-B2	− 57.04
6	P1-B3	− 39.81
7	P2-A1	− 47.37
8	P2-A2	− 29.18
9	P2-A3	12.82
10	P2-B1	− 35.78
11	P2-B2	6.03
12	P2-B3	53.37
13	P3-B1	1.09
14	P3-B2	9.30
15	P3-B3	3.47
16	P3-C1	20.00
17	P3-C2	11.65
18	P3-C3	29.74

[1] Improvement = (1 − GA/BEN)*100.

Table 7. Percentage waste for the present GA for randomly generated problems

No.	*Code*	*n*	*% waste*
1	R-1	20	1.515
2	R-2	20	2.470
3	R-3	20	1.511
4	R-4	25	1.271
5	R-5	25	1.402
6	R-6	50	1.122
7	R-7	50	1.066
8	R-8	100	0.461
9	R-9	100	0.487
10	R-10	100	0.427

The proposed heuristic was tested using three data sets. The first data set comprises of the eighteen published data in Benati (1997). The problem code, P_iL_j means that the problem with real data i has been solved with L_j as a set of widths.

The second data set is made of 10 randomly generated problems. The pattern height is 1500. The widths of the rolls are randomly generated from a uniform distribution $[1000, 2000]$. The widths and heights of the pieces of items are each randomly generated from a uniform distribution $[200, 1000]$. The demands for pieces are randomly generated from a uniform distribution $[100, 6000]$. The problems solved have the number of items within the range $[10, 200]$ and number of rolls within the range $[1, 10]$.

Table 8. Rectangular packing results for DP (genetic neuro-nesting approach)

Demand Set	*Figure set*										
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*Average*
1	14.43	2.00	1.14	8.44	2.14	9.67	14.12	7.64	10.19	12.80	8.26
2	15.02	2.64	1.04	9.44	3.26	10.63	8.38	11.65	5.98	11.88	7.99
3	12.23	5.15	0.70	8.62	2.95	8.25	4.44	6.07	4.64	11.43	6.45
4	14.67	4.04	1.19	11.06	4.32	10.39	9.70	11.50	6.81	10.48	8.42
5	12.16	3.64	0.88	10.93	4.50	8.13	11.86	12.88	7.62	12.75	8.54
6	13.81	4.13	0.98	9.74	3.34	9.27	8.73	9.22	6.68	11.64	7.75
7	11.72	4.01	0.77	10.43	4.60	7.90	7.43	12.86	5.33	13.98	7.90
8	13.59	2.92	0.92	8.83	2.88	8.84	9.53	8.36	7.72	11.99	7.56
9	14.22	3.62	1.05	9.92	3.84	8.64	11.10	9.62	8.20	14.39	8.46
10	12.80	3.33	0.90	9.50	3.54	9.36	6.46	10.95	5.14	10.97	7.30
11	13.07	2.39	0.82	8.63	2.88	8.82	8.88	9.68	7.10	12.48	7.48
12	15.42	2.70	1.19	9.84	4.09	10.02	10.08	11.30	7.66	15.05	8.74
13	15.35	4.54	1.11	8.94	3.11	9.40	7.51	6.60	6.60	14.84	7.80
14	14.87	4.76	1.30	11.40	4.34	9.59	13.59	9.18	9.23	11.24	8.95
15	15.27	1.47	1.04	7.71	2.23	11.54	6.62	10.41	5.44	10.29	7.20
16	12.91	3.95	0.95	10.13	3.91	8.51	10.46	11.08	7.32	13.46	8.27
17	16.42	3.14	1.33	9.83	3.68	11.86	6.92	11.59	5.19	12.02	8.20
18	14.82	4.12	1.08	9.23	2.78	10.00	8.17	7.49	6.52	10.59	7.48
19	12.83	2.71	0.84	8.95	3.31	8.37	10.07	10.08	7.54	14.34	7.90
20	15.04	1.49	1.02	7.99	2.02	11.17	8.22	8.92	6.48	7.45	6.98
Average	14.03	3.34	1.01	9.48	3.39	9.52	9.11	9.85	6.87	12.20	7.88

Table 9. Rectangular packing results for the present GA

Demand Set	*Figure set*										
	1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*Average*
1	2.85	0.00	2.97	3.62	6.39	2.12	1.91	3.66	3.46	8.73	3.57
2	4.95	0.00	6.86	2.82	7.88	2.34	3.79	3.99	2.35	9.23	4.42
3	3.97	0.00	3.37	5.25	6.16	2.09	2.19	3.99	2.39	8.37	3.78
4	4.78	0.00	6.19	3.38	6.46	1.94	2.32	4.10	2.77	9.12	4.11
5	6.80	0.00	6.40	3.01	5.95	2.44	3.00	4.03	2.11	8.87	4.26
6	4.72	0.00	4.02	2.78	5.94	1.94	4.26	4.07	2.22	6.33	3.63
7	3.53	0.00	3.24	5.72	7.19	2.09	2.86	3.99	2.59	5.14	3.64
8	4.21	0.00	4.22	4.96	6.51	2.55	3.14	3.36	0.83	7.03	3.68
9	3.97	0.00	2.47	6.29	5.39	2.34	3.31	3.36	0.40	4.19	3.17
10	5.54	0.00	2.11	3.03	6.26	2.09	2.21	5.05	0.79	4.81	3.19
11	3.53	0.00	3.68	3.03	5.34	2.33	2.59	3.45	0.46	3.82	2.82
12	5.00	0.00	2.30	3.35	5.39	2.34	2.07	1.97	0.56	4.52	2.75
13	5.97	0.00	7.00	2.82	5.73	1.94	3.05	3.36	0.44	7.87	3.82
14	5.00	0.00	6.48	5.67	7.62	1.94	1.97	3.75	0.85	5.76	3.90
15	5.54	0.00	5.65	3.24	5.30	2.12	2.56	2.96	2.64	6.16	3.62
16	3.79	0.00	6.33	3.24	4.71	2.55	2.66	4.24	2.55	4.24	3.43
17	4.38	0.00	3.98	4.69	4.57	2.45	3.63	4.57	0.71	9.52	3.85
18	3.10	0.00	3.89	4.62	3.04	2.12	3.79	2.90	0.88	3.82	2.82
19	6.03	0.00	1.75	3.62	4.93	2.55	3.24	4.03	0.96	9.90	3.70
20	2.78	0.00	9.59	4.85	4.62	2.23	2.52	3.30	0.83	9.14	3.97
Average	4.52	0.00	4.63	3.99	5.77	2.23	2.85	3.71	1.54	6.83	3.61

Table 10. Comparison between present GA over DP based on dimension of required parts in different figure sets

No.	*DP*	*Present GA*	*Improvement %*[1]
1	14.03	4.52	76.78
2	3.34	0.00	∇
3	1.01	4.63	− 358.42
4	9.48	3.99	57.91
5	3.39	5.77	70.21
6	9.52	2.23	76.57
7	9.11	2.85	68.72
8	9.85	3.71	62.24
9	6.87	1.54	77.58
10	12.20	6.83	44.01
Average	7.88	3.61	54.19

[1] Improvement of present GA over DP = (1 − GA/DP)*100.
∇ Excessive.

Values outside these ranges are very easily accommodated depending on the computer memory size.

The third data set comprises of 10 different rectangular patterns and 20 different demands taken from Dagli and Poshyanonda (1997) which was solved using their genetic neuro-nesting approach.

Tables 5 and 6 show the results of Benati (1997) otherwise referred to as BEN and the proposed genetic algorithm (GA) discussed in this paper (Table 5). Benati (1997) reported solutions with different starting lists ($H1$, $H2$, and $H3$), with the best one (H) improved by an interchange procedure (IC). Only ($H + \mathrm{IC}$) which represents the overall best quality of solutions for BEN are compared with GA.

Table 7 shows the results for GA proposed in this paper for the second data set randomly generated.

Tables 8 and 9, respectively show the results of Dagli and Poshyanonda (1997) otherwise referred to as DP and the proposed genetic algorithm (GA) discussed in this paper.

From Table 6 it can be observed that GA performed better than BEN in terms of the solution quality in 11 out of the 18 problems. In order words, GA performed better than BEN in 61% of the problems while BEN performed better in 39% of the problems.

Table 7 shows results for the GA procedure for 10 randomly generated problems for small, medium and large sizes. The number of rolls for problems 1–7 (small and medium) is 5, while the number of rolls for problems 8–10 (large) is 10. It can be observed from the results in Table 7 that smaller percentage wastes are obtained for greater number of rolls available. This is because there is a greater possibility of choices to be made from available rolls.

Table 11. Comparison between present GA over DP based on demands for each figure in a particular period

No.	*DP*	*Present GA*	*Improvement %*[1]
1	8.26	3.57	56.78
2	7.99	4.42	44.68
3	6.45	3.78	41.40
4	8.42	4.11	51.19
5	8.54	4.26	50.12
6	7.75	3.63	53.16
7	7.90	3.64	53.92
8	7.56	3.65	51.32
9	8.46	3.17	62.53
10	7.30	3.19	56.30
11	7.48	2.82	62.29
12	8.74	2.75	68.54
13	7.80	3.82	51.02
14	8.95	3.90	56.42
15	7.20	3.62	56.23
16	8.27	3.43	58.52
17	8.20	3.85	53.05
18	7.48	2.82	62.29
19	7.90	3.70	53.16
20	6.98	3.99	43.12
Average	7.88	3.61	54.19

[1] Improvement of present GA over DP = (1 − GA/DP)*100

From Tables 8 and 9 it can be observed that GA (the current genetic algorithm) performed better than DP (Dagli and Poshyanonda, 1997) in terms of the solution quality in all 20×10 problems. This is evident when the last row and column, which show the averages of all the problem sets are considered. Tables 10 and 11 compare the averages of Tables 8 and 9, respectively and simplify the reader's observation.

Conclusions

This paper proposes a genetic algorithm approach for solving the two-dimensional cutting stock problem. The procedure has been tested on a set of problems from the literature and randomly generated ones. Using the same set of problems, the results of the proposed procedure were compared with those of the partial enumeration-based approach by Benati. The experimentation shows that the proposed genetic algorithm approach yields better solutions in 60% of the instances compared to the partial enumeration-based approach. Again using the same 20×10 data set, the results of the proposed procedure were compared with those of the neuro-nesting approach by Dagli and Poshyanonda. The experimentation shows that the proposed genetic algorithm approach yields better solutions to the genetic neuro-nesting approach. It is anticipated that in further developments of this work, a mechanism will be devised for trapping the iteration or generation when the minimum waste is obtained for the first time before being repeated without changing.

Acknowledgment

The comments of the three unanimous referees were very constructive and have contributed immensely in improving the final version of the article.

References

Babu, A. R. and Babu, N. R. (1999) Effective nesting of rectangular parts in multiple rectangular sheets using genetic and heuristic algorithms. *International Journal of Production Research*, **37**(7), 1625–1643.

Benati, S. (1997) An algorithm for a cutting stock problem on a strip. *Journal of Operational Research Society*, **48**, 288–294.

Berkey, J. O. and Wang, P. Y. (1987) Two-dimensional final bin-packing algorithms. *Journal of Operational Research Society*, **38**, 423–429.

Chauny, F., Loulou, R., Sadones, S. and Soumis, F. (1991) A two-phase heuristic for the two-dimensional cutting—stock problem. *Journal of Operational Research Society*, **42**, 32–47.

Dagli, C. H. and Poshyanonda, P. (1997) New approaches to nesting rectangular patterns. *Journal of Intelligent Manufacturing*, **8**, 177–190.

Dorigo, M. and Gambardella, L. M. (1997) Ant colony system: a cooperative learning approach to the travelling salesman problem. *IEEE Transaction on Evolutionary Computation* **1**(10), 53–66.

Fleurent, C. and Ferland, J. (1994) Genetic hybrids for the quadratic assignment problem. DIMACS. *Series in Mathematical Theoretical Computer Science*, **16**, 190–206.

Garey, M. R. and Johnson, D. S. (1979) *Computers and intractability: a guide to the Theory of NP-Completeness*, W. H. Freeman and Company, San Francisco, CA, USA.

Gilmore, P. C. and Gomory, R. E. (1963a) A linear programming approach to the cutting stock problem, Part II. *Operations Research*, **11**, 863–888.

Gilmore, P. C. and Gomory, R. E. (1963b) The theory and the computation of knapsack functions. *Operational Research*, **14**, 1045–1074.

Gilmore, P. C. and Gomory, R. E. (1966) Multi-stage cutting stock problems of two and more dimensions. *Operational Research*, **13**, 94–120.

Glover, F. (1989) Tabu search—Part I, *ORSA Journal of Computing*, **1**, 190–206.

Goldberg, D. E. (1989) *Genetic Algorithms: In Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA.

Hains, M. J. and Freeman, H. (1970) A multistage solution of the template-layout problem. *IEEE Transactions on System, Science and Cybernetics*, **SSC-6**, 145–151.

Kroger, B. (1995) Guillotinable bin packing: a genetic approach. *European Journal of Operational Research*, **84**, 645–661.

Liepins, G. E. and Hilliard, M. R. (1986) Genetic algorithms as a paradigm for machine learning. *Paper Presented at the ORSA/IMS Joint National Meeting*, Miami, FL, USA.

Onwubolu, G. C. and Mutingi, M. (2001) Optimizing the multiple constrained resources product mix problem using genetic algorithms. *International Journal of Production Research*, **39**(9), 1897–1910.

Paradza Daza, V., Munoz, R. de-A. and Gomes, A. (1995) A hybrid genetic algorithm for the two-dimensional guillotine cutting problem, in Biethan, J. and Nissen, V. (eds.) *Evolutionary Algorithms in Management Applications*, Berlin, pp. 183–196.

Poshyanonda, P., Bahrami, A. and Dagli, C. (1992) Artificial neural networks in stock cutting problem, in *Neural Networks in Manufacturing and Robotics*, Shin, Y. C., Abodelmonen, A. H. and Kumara, S. (eds.) ASME Press, New York, pp. 143–153.

Spears, W. M. and De Jong, K. A. (1990) Using neural networks and genetic algorithms as heuristics for NP-complete problems, in *Proceedings of the International Joint Conference on Neural Networks IEEE*, **1**, 118–123.

Wang, P. Y. (1983) Two algorithms for constrained two-dimensional cutting stock problems. *Operations Research*, **31**, 573–586.