Cutting Stock Waste Reduction Using Genetic Algorithms

Y. Khalifa
Computer Engineering Department
State University of New York
New Paltz, NY 12561, USA
khalifay@newpaltz.edu

O. Salem
Civil and Environmental Engineering Department
The University of Cincinnati
Cincinnati, Ohio 45221, USA
o.salem@uc.edu

A. Shahin
Civil Engineering Department
University of Alberta
Edmonton, Alberta, T6G2G7, Canada
ashahin@ualberta.ca

ABSTRACT

A new model for the One-dimensional Cutting Stock problem using Genetic Algorithms (GA) is developed to optimize construction steel bars waste. One-dimensional construction stocks (i.e., steel rebars, steel sections, dimensional lumber, etc.) are one of the major contributors to the construction waste stream. Construction wastes account for a significant portion of municipal waste stream. Cutting one-dimensional stocks to suit needed project lengths results in trim losses, which are the main causes of one-dimensional stock wastes. The model developed and the results obtained were compared with real life case studies from local steel workshops. Cutting schedules produced by our new GA model were tested in the shop against the current cutting schedules. The comparisons show the superiority of this new GA model in terms of waste minimization.

Categories and Subject Descriptors
J.2 **[Computer Applications]**

General Terms: Management, Experimentation.

Keywords
Cutting Stock Problem, Waste Reduction, Genetic Algorithms.

1. INTRODUCTION

Construction wastes account for a significant portion of America’s municipal waste stream. In [1], it estimates the generated amounts of construction and demolition wastes to be about 23% of the total solid waste stream. This percentage of wastes translates to more than 100 million tons per year [2]. The percentages reported from other countries verify these figures from the United States.

A portion of the wastes generated from stock cutting is avoidable, meaning that it is generated due to inefficient use of materials. Using the materials in a more efficient way would reduce the amount of unnecessary purchased materials, unnecessary workmanship, overheads, wastes, and hauling and tipping fees needed to dump the waste.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
GECCO’06, July 8–12, 2006, Seattle, Washington, USA.
Copyright 2006 ACM 1-59593-186-4/06/0007…$5.00.

Ultimately, the cutting wastes that are generated on a construction project should only be unavoidable wastes, which could be achieved through better planning and scheduling for cutting the materials. The cutting stock problem (CSP) is a major source of one-dimensional stock wastes that are generated in the construction industry. The costs that are incurred due to inefficient stock cutting affect the prices of bids and lower the construction firms’ profit margins.

2. THE ONE-DIMENSIONAL CUTTING STOCK PROBLEM (CSP)

The one-dimensional cutting stock problem (CSP) addresses the practical issue of how to cut the required lengths of a material from an existing standard length stock with minimum trim losses. Generally, steel rebars and sections are produced in standard lengths (L_s). The lengths of units demanded by construction projects are normally only a fraction of the standard lengths. The process of cutting the standard units to supply the different lengths of the demanded units would normally result in losses known as trim, or cutting, losses. An unlimited supply of the standard units is generally assumed.

Throughout the rest of this paper, focus will be given to steel (rebars and sections) as an example of one-dimensional stocks. The same approaches adopted could be used to deal with other one-dimensional stocks. From the design drawings, the exact needs of reinforcement steel units are computed. The standard-length units need to be cut in a way that would supply the demanded lengths and minimize the cutting losses.

One of the most popular techniques for solving the CSP was introduced by [3]. The problem was solved by formulating it as an LP problem, where the columns of the constraint matrix represent the different ways of cutting the standard lengths. There are too many patterns, so too many columns could be generated, and it was impractical to list them all. They used an ingenious column generation technique, which has the advantage of not requiring the generation of all possible cutting patterns. The technique introduced by [3] is very efficient, but it might not reach optimal solutions due to the need of rounding the relaxed fractional solutions to integer values, which would result in additional wastes.

The solution reached using Linear Programming (LP) is normally not an integer solution. Applying rounding techniques to relaxed non-integer solution results in solutions departing from optimality, resulting in unnecessary wastes. According to [4], a hard problem is one in which the average value of L_s/l_j (Standard length of the stock to be cut / Demanded length) is less than 5; all the other problems are defined as easy problems. The terms that were used by [5] were hard versus soft problems. He used the analogy of filling cups with large pebbles as a hard problem and filling them with sand as a soft problem. The problem with hard problems is that the application of rounding procedures to the relaxed solutions or the use of a greedy heuristic like the sequential heuristic procedure (SHP) usually results in high trim losses.

Integer Programming (IP) approaches were used to reach optimal integer solutions. Reaching an integer solution could be very complex, and avoiding this problem is not an easy task. Integer Programming requires a lot of computational effort once the number of cutting patterns becomes too large. Because of the large number of possible cutting patterns that exist in the CSP, the result is a combinatorial problem [6]. Another approach was combining both the cutting planes technique, and the branch and bound technique [5]. Recently, there has been renewed interest in such an approach as it was quite successful in solving many problems. “Almost all the work done in the CSP has concentrated on the heuristics as the underlying integer program has been difficult to solve to optimality.”

3. DEVELOPMENT OF A GA MODEL FOR SOLVING THE ONE DIMENSIONAL CSP

In this section, the components of a GA model for solving the one-dimensional CSP are presented. The next sub-sections introduce the basic components of GA3, which was developed in the Visual Basic programming language.

3.1 Representation

Each chromosome was constructed of a series of pairs; the first number in each pair would give the pattern number to be used, and the second number would give the number of times this pattern would be used in the solution. Figure 1 shows a chromosome encoded in this way. The chromosome length is a GA parameter that should be set to a reasonable value. Previous studies show that the number of patterns used will be close to the number of different lengths needed [7], which means using the number of pairs in the chromosome equal to the number of different lengths demanded. To cover all possibilities, the chromosome length was chosen to be double the number of different lengths demanded (N).

10	12	12	9	26	11	59	29	58	16	67	11

Fig. 1. Chromosome Structure

3.2 Data Reading

The data input is dealt with in the opening screen of GA1D. As shown in Figure 2, the input screen of GA1D, the data input form consists mainly of two tables. In the first table, the user would enter the standard lengths of the stock to be cut (L_s). In the second table, the user would enter the demanded lengths (l_j) and the number of times each of those lengths is needed (d_j).

Generating the Cutting Patterns After reading the data, the next step is to generate the feasible cutting pattern. For a pattern to be feasible, the total length of the pattern (utilized length) must be less than or equal to the standard length of the standard units (L_s).

However, the feasible cutting patterns generated in GA1D were limited to the efficient cutting patterns only. According to [8], an efficient cutting pattern is defined as a pattern that would have an unused (wasted) length that is less than the smallest demanded unit length $(l_j)_{min}$. The procedure provided in Appendix 1, adapted from [9], can be used to generate all the efficient feasible cutting patterns.

3.3 Initialization

For generating the random numbers, the Multiplicative Linear Congruential Scheme was used throughout the model, which could be defined by the following recursive equation on the nth iteration:

$Z_n = a \,.\, Z_{n-1} \text{ MOD } m$, where

Z_0 = a user-defined starting integer value (referred to as a seed);

m = the modulus is a large integer number chosen to be $2^{31} - 1$; and

a = the multiplier, set at 7^5 (16,807).

The random number, R_n, on the ith iteration would be obtained by $R_n = \frac{Z_n}{m}$.

R_n generated by this equation is a number between 0 and 1. For generating the pattern numbers, R_n will be multiplied by the maximum pattern number, and the first integer number bigger than or equal to the resulting figure will be returned. For generating the patterns' repetitions, R_n will be multiplied by the maximum number of demanded units needed $(d_j)_{max}$, and the first integer number bigger than or equal to the resulting figure will be returned.

3.4 Evaluation

The objective of the evaluation function in this model is to minimize the waste while in the same time, meet the demand of each length. The proposed evaluation function is

$$\frac{\sum_{j=1}^{N} l_j \cdot d_j}{L_S \sum_{i=1}^{I} X_i + L_S \sum_{j=1}^{N} \#Un\sup_j}, \text{ where}$$

l_j = the length of demanded length number j;

d_j = the number of times length number j is demanded;

X_i = the number of standard units cut according to pattern number i;

$\#Unsup_j$ = the number of unsupplied units of length l_j; and

L_s = the length of the standard units

The numerator of the evaluation function expresses the total length of the demand. It is a summation of the demanded lengths, l_j, multiplied by the number of times each length is needed, d_j.

The first term in the denominator is a summation of the total length of all the standard units cut to satisfy the demand, which is calculated by multiplying the summation of all the patterns' repetitions by the standard length (L_s). The second term in the denominator is calculated on the assumption that, in the worst case scenario, the number of standard units needed to be cut in order to satisfy the remaining unsupplied units will not exceed the number of those unsupplied units. This scenario assumes that every standard unit would deliver only one unsupplied unit, so the length that would be used to satisfy the unsupplied units would be calculated by multiplying the number of unsupplied units by the standard length (L_s). Adding the two terms together gives the maximum length of standard units needed to satisfy the demand.

After first attempts of the algorithm, it was noticed that it had a tendency of supplying the demand of the shorter units and leaving a number of the longer units unsupplied. In the denominator, the second term (summation of the number of the unsupplied units multiplied by the standard length) did not differentiate between the shorter and longer units. At the same time, the shorter units would normally exist in more patterns, and their numbers in the patterns are higher than the longer units, which made it easier for the algorithm to supply the shorter units. In the same time, with respect to the goal of minimizing wastes, having longer unsupplied units usually results in generating more wastes than having shorter unsupplied units. Therefore, a new evaluation function was generated:

$$\frac{\sum_{j=1}^{N} l_j \cdot d_j}{\sum_{i=1}^{I} L_{si}.X_i + P\sum_{j=1}^{N} \#Un\sup_j \cdot l_j}, \text{ where}$$

l_j = the length of demanded length number j;

d_j = the number of times length number j is demanded;

X_i = the number of standard units cut according to pattern number i;

L_{si} = the standard length from which pattern number i is cut;

$\#Unsup_j$ = the number of unsupplied units of length l_j; and

P = penalty coefficient for not supplying the demanded units.

The numerator of the first evaluation function was left unchanged. The first part of the denominator was changed to allow for the use of multiple standard lengths. The second part of the denominator dealt with the total length of unsupplied units instead of the total number of unsupplied units, which gave more pressure for supplying the demand of the longer units. To increase this pressure, the penalty coefficient (P) was introduced. The higher the penalty coefficient, more pressure would be in favor of supplying the longer units.

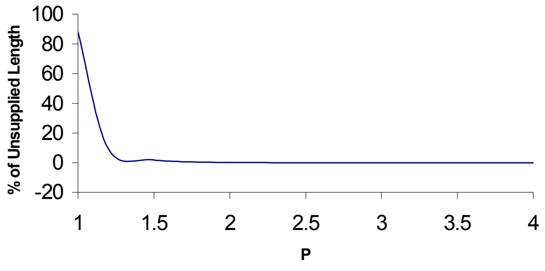

Fig. 2. Effect of "P" on the Percentage of Unsupplied Length.

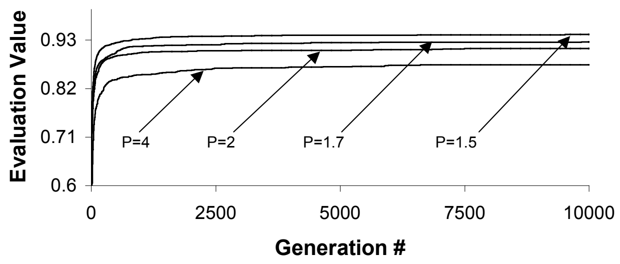

Fig. 3. Effect of "P" on the Performance of GA3.

Figure 2 shows the effect of increasing P on the percentage of the unsupplied length of the demand. At P = 1.5, the percentage of unsupplied length dropped considerably to less than 2%, so a value of $P \geq 1.5$ is generally recommended.

Examining the performance of the model in the recommended range of P, Figure 3 shows the effect of increasing P on the performance of the model. From Figure 4, the value of P was set to 1.5 for optimal performance.

The enhanced evaluation function offers the advantage of being able to work with several standard lengths. For each standard length, the efficient patterns are generated; then, the algorithm eliminates the patterns that have a total used length (C_i) that is less than or equal to the length of the next shorter standard length. The same pattern will automatically be generated using the shorter standard unit. Thus, less waste will be present in the pattern using the shorter standard unit than in the same pattern using the longer standard unit.

4. CASE STUDIES

To validate the GA model, several visits to local Fargo steel fabricators' workshops were made. The purpose of these visits was to have access to actual orders that were cut. The same orders were scheduled using GA3, and the amounts of waste were compared to see if a reduction of waste could have been achieved if GA3 schedule was followed. The GA model was also compared against an Integer Programming (IP) model. Comparison results are shown for three case studies listed below.

4.1 Case Study I

The type of section used was W 14 x 90, weighing 90 lbs/ft. Table 1 shows the available standard lengths. Table 2 shows the demanded lengths. Table 3 shows the waste comparison between the three schedules; the workshop schedule and those of the Integer Programming model and the Genetic Algorithm model.

From Table 2, the following figures of waste can be calculated:

Waste reduced by weight using GA1D = 1800 lb; and

percentage of waste reduced using GA1D = (1800 / 7211.43) x 100 = 25%.

Table 1. Case Study I: Available Standard Lengths

Serial	Standard Length (ft)
1	69
2	65
3	60
4	50
5	32

Table 2. Case study I: Demanded Lengths

Serial # (j)	Length (l_j)	Quantity Needed (d_j)
1	54'.00	1
2	53.66'	1
3	50.75'	5
4	38.66'	2
5	34.75'	1
6	30.25'	13
7	28.75'	3
8	26.583'	6
9	23.083'	1
10	22.583'	4
11	7.66'	3

Table 3. Case study I: Waste Comparison

	Total Demanded Length (ft)	Total Demanded Weight (lb)	Length Used (ft)	Weight Used (lb)	Waste (lb)	Waste %
Work Shop Schedule	1248.873	112398.57	1329	11961	7211.4	6.42
GA1D Schedule	1248.873	112398.57	1309	11781	5411.4	4.81

4.2 Case Study II

The type of section used was MC 12 x 35, weighing 35 lbs/ft. The available standard lengths were 40-ft units. Table 4 shows the demanded lengths. Table 4 shows the waste comparison between the two schedules.

From Table 5, the following figures of waste can be calculated:

Waste reduced by weight using GA1D = 52.5 lb; and percentage of waste reduced using GA1D = 52.5 / 350 x 100 = 15%.

In the previous case study, GA1D was able to reach a solution that would bundle 31.5 ft of unused length in one pattern. As this length is considerably large, it could still be used in another job and is not considered waste, thus achieving more waste reductions than the case of the workshop cutting schedule.

Table 4. Case study II: Demanded Lengths

Serial # (j)	Length (l_j)	Quantity Needed (d_j)
1	19'	4
2	15.5'	4
3	15'	4
4	12'	4
5	10'	4
6	8.5'	4

Table 5. Case study II: Waste Comparison

	Total Demanded Length (ft)	Total Demanded Weight (lb)	Length Used	Weight Used (lb)	Waste (lb)	Waste %
Work Shop Schedule	320	11200	330	11550	350	3.125
GA1D Schedule	320	11200	328.5	11498	298	2.66

4.3 Case Study III

The type of section used was RD 2.5, weighing 16.69 lbs/ft. The available standard lengths were 40 ft units. Table 6 shows the demanded lengths. Table 7 shows the waste comparison between the three schedules. From Table 6, the following figures of waste can be calculated:

Waste reduced by weight using GA1D = 528.17 lb; and percentage of waste reduced using GA1D = 528.17 / 669.1 x 100 = 79.0%.

Table 6. Case study III: Demanded Lengths

Serial # (j)	Length (l_j)	Quantity Needed (d_j)
1	12.146'	2
2	11.896'	2
3	11.729'	32
4	9.396'	6
5	9.0625'	4
6	7.229'	4
7	4.177'	28
8	3.0'	6

Table 7. Case study III: Waste Comparison

	Total Demanded Length (ft)	Total Demanded Weight (lb)	Length Used (ft)	Weight Used (lb)	Waste (lb)	Waste %
Work Shop Schedule	679.91	11347.70	720	12016	669.1	5.9
GA1D Schedule	679.91	11347.70	688.3	11488	140.9	1.24

In the previous case study, although the number of the standard 40-ft length units used to satisfy the demand in all cases was 18, the waste in the workshop schedule was scattered between the patterns. On the other hand, GA3 was able to reach a solution that would bundle 31.646 ft of unused length in one pattern that could still be used in another job.

The evaluation function explains the reason that GA3 will always attempt to bundle the waste in a single cutting pattern. In the denominator of the evaluation function, if it is possible for the algorithm to reduce the first term (total length of standard units used) by the length of an entire standard unit, which would only be possible by paying the price of increasing the second term (1.5 x unsupplied length); if the increase in the second term was less than the decrease in the first term, then the algorithm would recognize this case as a better solution. This case would only happen if the increase in the total length of the unsupplied units were less than two-thirds of the standard length. Thus, at least one-third of the length of that standard unit would be saved. Therefore, we might add to the advantages of our model that it will not only attempt to minimize the waste by reducing the total length of the standard units used, but it will also attempt to bundle as much waste as possible in one pattern, thus further reducing the amount of waste.

4.4 Summary of Case Studies

A summary of the three case studies is shown in Table 8. As shown in Table 8, a total of 2380.67 lb of wasted steel could have been saved using the proposed GA approach, which is about 28.92% of the total amount of generated waste. This amount of savings shows that the proposed model could help in reducing the amount of waste generated and shows the importance of using such techniques in planning for the cutting of one-dimensional stocks.

5. CONCLUSION

GA models are successful in dealing with complex combinatorial problems. These problems are characterized by having large solution spaces due to the existence of a huge number of combinations, or alternatives, to choose from; therefore, an optimal solution for these problems is normally a complex task to accomplish. For this reason, GA was proposed to solve the one-dimensional cutting stock problem (CSP) experienced in the construction industry. The GA model that was developed was successful in reducing the amount of cutting losses. A computer software with a user-friendly interface was also developed using Visual Basic programming language. The data entry for the software is simple, and no training was necessary.

The final model was validated by using actual industry orders that were obtained from local workshops. The output cutting schedules were compared with the actual cutting schedules used in cutting the workshops' steel sections. The case studies showed that the amount of wastes experienced in the actual cutting of the workshops' steel sections was reduced, on average, by 28.92% using the output of GA3. In some cases, GA3 was able to lump the wastes in bigger lengths, thus achieving more savings. Using optimization model described in cutting one-dimensional stocks substantially reduces the amount of construction waste currently experienced in the construction industry.

6. REFERENCES

[1] Apotheker, S. (1990). "Construction and Demolition Debris–The Invisible Waste Stream." Resource Recycling, v(9) n(12) pp. 66-74.

[2] Mills, T. H.; Showalter, E.; and Jarman, D. (1999). "Cost Effective Waste Management Plan." Cost Engineering, v(41) n(3) pp. 35-43.

[3] Gilmore, P. C. and Gomory, R. E. (1961). "A Linear Programming Approach to the Cutting-Stock Problem." Operartions Research Society of America, v(9) n(6) pp. 849-859.

[4] Johnston, R. W. (1980). "Cutting Schedules for the Paper Industry." 4th IFAC Conference Proceedings, Ghent, Belgium.

[5] Goulimis, C. (1990). "Optimal Solutions for the Cutting Stock Problem." European Journal of Operational Research, v(44) n(2) pp. 197-208.

[6] Haessler, R. W. (1975). "Controlling Cutting Patterns Changes in One-Dimensional Trim Problems." Operations Research, v(23) n(3) pp. 483-493.

[7] Lefrancois, P. and Gascon, A. (1995). "Solving a One-Dimensional Cutting-Stock Problem in a Small Manufacturing Firm: A Case Study." IIE Transactions, v(27) n(4) pp. 483-496.

[8] Vahrenkamp, R. (1996). "Random Search in the One-Dimensional Cutting Stock Problem." European Journal of Operational Research, v(95), n(1), pp. 191-200.

[9] Pierce, J. F. (1964). Some Large-Scale Production Scheduling Problems in the Paper Industry. Prentice Hall, Inc., Englewood Cliffs, NJ.

[10] Zheng D.; Ng T.; Kumaraswamy M.,(2004). "Applying a Genetic Algorithm-based Multiobjective approach for Time-Cost optimization" *Journal of Construction Engineering and Management,* 130(2), 168-176.

[11] Hegazy T., Elhakeem A., Elbeltagi E. (2004) "Distributed Scheduling Model for Infrastructure Networks" *Journal of Construction Engineering and Management,* 130(2), 160-167.

[12] Elazouni A.; Metwally F.(2005) "Finance Based Scheduling: Tool to Maximize Project Profit Using Improved Genetic Algorithms" *Journal of Construction Engineering and Management,* 131(4), 400-412.

Table 8. Summary of the Case Studies.

	Total Demanded Weight (lb)	Workshop Waste (lb)	Workshop Waste (%)	GA1D Waste (lb)	GA1D Waste (%)	Waste Reduction Using GA1D (lb)
Case I	112398.57	7211.43	6.42	5411.43	4.81	1800
Case II	11200	350	3.125	297.5	2.66	52.5
Case III	11347.7	669.1	5.9	140.93	1.24	528.17
Total	134946.27	8230.53	6.1	5849.86	4.33	2380.67

APPENDIX 1: Procedure for Generating the Efficient Feasible Cutting Patterns

Let L_s be the length of the standard bars; $l_1, l_2, \ldots, l_N$ be the lengths of demanded units in descending order; and $d_1, d_2, \ldots, d_N$ be their corresponding demands.

1. Set i = 1 (pattern number 1)

$A_{i1} = \min(< (L_s / l_1) >, d_1)$,

$A_{i2} = \min(< ((L_s - A_{i1}\, l_1) / l_2) >, d_2), \ldots,$

$$A_{iN} = \min(< ((L_s - \sum_{j=1}^{N-1} A_{ij}\ l_j) / l_N) >, d_N)$$

where < g > is the largest integer less than or equal to g.

2. Efficient pattern number i is [$A_{i1}, A_{i2}, \ldots, A_{iN}$], where

A_{ij} = the number of demanded units of length l_j that are present in pattern number i.

3. In pattern number i, if there is a j, $1 \le j \le N-1$, such that $A_{ij} > 0$, then

let k be the largest such j, and go to 4.

If there is no such j, then terminate the procedure. All efficient cutting

patterns have been generated.

4. Set i = i + 1 (next pattern)

Set $A_{i1} = A_{(i-1)1}, A_{i2} = A_{(i-1)2}, \ldots, A_{i(k-1)} = A_{(i-1)(k-1)}$

$A_{ik} = A_{(i-1)k} - 1$

$$A_{i(k+1)} = \min(< ((L_s - \sum_{j=1}^{k} A_{ij}\ l_j) / l_{k+1}) >, d_{k+1}), \ldots,$$

$$A_{iN} = \min(< ((L_s - \sum_{j=1}^{N-1} A_{ij}\ l_j) / l_N) >, d_N)$$

Goto 2.