Engineering Journal of University of Qatar, Vol. 8, 1995, p. 185 - 196

AN ALGORITHM TO MINIMIZE UNUSED LENGTHS OF STEEL BARS AND SECTIONS

Aly N. El-Bahrawy
Associate Professor
Civil Engineering Department, University of Qatar
Doha, Qatar.

ABSTRACT

The reinforcement steel bars and other metal or wooden sections used in the construction industry are usually available in standard lengths and are cut in large numbers of specified lengths (pieces) according to the needs of the structures. The choice of how to cut these lengths is important to minimize the unused part of the standard length. If the size of the project is big, the losses may be large causing an increase in the total cost of the project. The engineers responsible for cutting the sections may use trial and error procedure to minimize the losses. The present paper demonstrates a simple algorithm for selecting the optimal cutting method to minimize the unused lengths. The solution of the problem is obtained through a two step procedure, where the first step generates possible feasible combinations made of several pieces, while the second uses a linear programming model to minimize the unused lengths, while satisfying the amounts requested from each piece. The paper demonstrates the developed algorithm through a series of examples ranging from simple to real life examples showing its ability to find an optimal solution to the problem. The concept has many applications and can produce significant savings in construction material.

INTRODUCTION

Construction materials are shipped to the construction site in huge amounts. Some of these materials are in the form of sections of different sizes produced in standard lengths. Project requirements are usually made through the request of great numbers of different specified lengths or pieces from each size. The problem of choosing the method of dividing the standard sections, like reinforcement bars, aluminum, steel and wooden sections, is important since it can have a considerable effect on the total cost of the project. The aim of this work is find an optimal solution to the problem, through a two step algorithm, by which the waste length is minimized. The scope of application of such an algorithm and its simplicity should be very appealing to companies in the construction business. Similar

concepts for optimal floor-planning in integrated circuit design are used by other researchers (1), to minimize the wastage areas.

DESCRIPTION OF THE PROBLEM

Assume that the project requirement from a certain diameter of reinforcement bars are given in the form of a table containing the length of each piece and the number requested. The solution of the problem can be tackled by answering two questions: how to cut a single standard length into smaller pieces to form a 'combination' of different lengths?, and how many combinations should be used to satisfy the different numbers requested from each piece?. In other words, what is required is to choose the feasible combinations, and the number of each combination to satisfy the list of requirements.

The first step of choosing a combination can not be handled as a typical permutations or combinations problem. The permutations are defined as follows: Given N distinct objects and a number M < = N, the number of configurations formed by placing M objects in line is denoted by P^N_M and is called permutations of N objects taken M at a time. In the latter definition two permutations are different if the order of placement is not the same. The number of permutations are computed as follows:

$$\overset{N}{\underset{M}{P}} = N(N-1)....(N-M+1)$$

On the other hand, combinations are denoted by C^N_M, and are formed by selecting the objects in all possible ways where the order of selection is immaterial. The number of combinations is computed as follows:

$$\overset{N}{\underset{M}{C}} = \frac{N!}{M!\ (N-M)!}$$

The use of combinations to find the number of ways the standard length can be divided into pieces is not helpful since the objects to choose from are not distinct, and the number M is not constant. More than one piece of the same length can be included in the combination, and the number of objects in a combination is conditional by the fact that the summation of the individual lengths should be less than the standard length. An exhaustive search to determine all possible feasible combinations could be used or combinations can be chosen at random since the number of feasible combinations can be very large. It will be shown later, however, that this effort is not justified since a solution can be found with a limited

number of feasible combinations. A simple heuristic approach is developed to generate such combinations.

The Suggested Solution

The solution to the problem is composed of two steps. The first is the search for a set of feasible combinations to be used in the second step which is the optimal choice of the number of these combinations needed to satisfy the requested number of pieces. The solution is applicable only for sections of a particular size. Several solutions will be needed if requests are made for several sizes.

Combination step

A heuristic approach is developed to generate a sufficient number of feasible combinations. The idea is simply to compile a large array of piece lengths in a certain order, and use this array to choose feasible combinations. This choice is carried out by using the array to add one length to the next until the accumulated length exceeds the maximum standard length. Then, the number of pieces in the combination will be obtained by excluding the last piece. This concept is presented in a flowchart-like diagram shown in figure (1). The following describes how the large array is compiled:

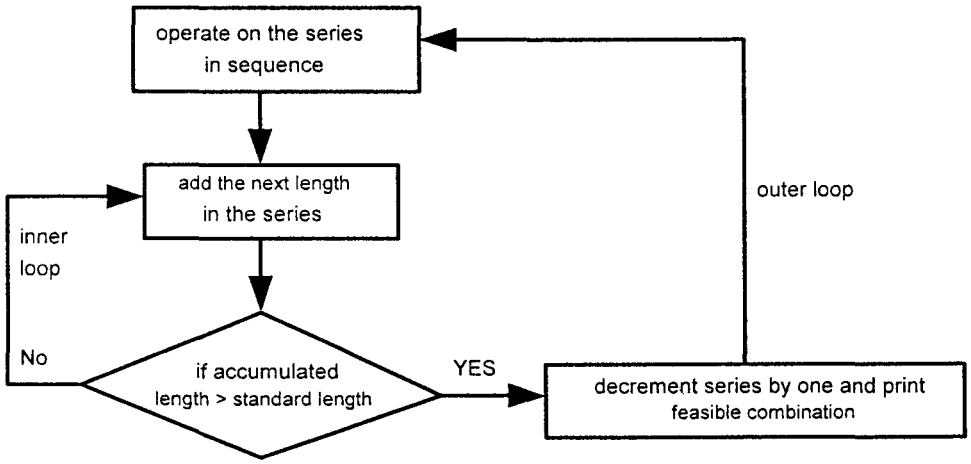

Fig. 1: Combination generating algorithm

- the first elements of the array contain the vector of requested piece lengths repeated four times, two times in ascending order, and two times in descending order. The idea behind this is to generate possible combinations from the set of pieces without repetition. The number of generated combinations will be limited.

- given the number of pieces and length of each piece, calculate the maximum number of similar pieces to make up a feasible combination

- complement the array with the series of similar lengths of pieces determined in the previous step, arranged in many different ways. Here, the number of generated combinations will be large.

The above method of compiling the array and the way it is compiled is very arbitrary, and the function of which is to generate sufficient number of feasible combinations. The method permits the generation of combinations from unrepeated and repeated pieces.

Theoretically speaking, the method can generate as much combinations as the elements in the array. However, the method can generate many similar combinations that have the same pieces and same accumulated length, arranged in different sequences. This extended list can then be processed by sorting the different combinations and using the query function of any database programs to extract the unique combinations. The resulting reduced list can then be inspected to exclude combinations with large waste lengths.

Optimal choice step

To find the optimal number of repeated combinations to satisfy the requirements list, a linear programming model is developed. The system variables are the number of times each combination has to be repeated. The objective function is the summation of the waste lengths resulting from repeating each combination. The system constraints are all in the form of equality constraints, where each equation calculates the number requested from a specified piece using the chosen combinations. The number of variables and the sequence of pieces in each combination required to formulate the linear program are the output of the previous step.

The linear programming model is expressed mathematically in the following form:

Minimize

$$Z = c_1x_1 + c_2x_2 + c_3x_3 + \ldots\ldots\ldots\ldots + c_nx_n$$

where c_i is the waste length from combination and x_i is the number of times combination i is repeated

subject to the set of constraints:

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \ldots\ldots\ldots\ldots + a_{1n}x_n = b_1$$

$$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \ldots\ldots\ldots\ldots + a_{2n}x_n = b_2$$

$$\vdots \qquad\qquad\qquad\qquad\qquad\qquad \vdots$$

$$a_{m1}x_1 + a_{m2}x_2 + a_{m3}x_3 + \ldots\ldots\ldots\ldots + a_{mn}x_n = b_m$$

where a_{ij} = number of similar pieces i in combination j
b_i = number requested from piece j
n = the number of feasible combinations
m = the number of different pieces requested

The model can be expressed in condensed form as

Minimize

$$Z = \sum_{i=1}^{n} c_i x_i$$

subject to the set of constraints,

$$| A | \underline{x} = \underline{b}$$

ILLUSTRATIVE EXAMPLES

Three different examples are cited here to demonstrate the developed algorithm. The first example is a simple two-dimensional problem chosen to illustrate the solution of the LP model in a graphical form. The second example is a four variable example to enable the search for all possible feasible combinations and to

study the merit of the exhaustive search for combinations. The third example is an actual request forming for a part of a large construction project to illustrate the application of the developed algorithm to real problems.

Example One

It is assumed that a simple project needs two different lengths of a certain section, 15 pieces of length 4.3 m and 30 pieces of length 2.5 m. The standard length of the section is 12 m. Two combinations, i.e. two design variables, are chosen. The first combination is 3 x 2.5 + 4.3 with a waste of 0.2 m, the second is 2.5 + 2 x 4.3 with a waste of 0.9 m. The linear programming model can be set as follows:

Minimize

$$Z = 0.2\, x_1 + 0.9\, x_2$$

subject to the constraints:

$$3\, x_1 + x_2 = 30$$
$$x_1 + 2\, x_2 = 15$$

The graphical representation of the feasible region and the cost surface is shown in figure (2). The optimal solution is $x_1^* = 9$ and $x_2^* = 3$, and the total waste is $Z^* = 4.5$ m. This means that the minimum waste is realized by dividing the standard lengths 9 times with the first combination, and 3 times with the second.

Example Two

In this example four pieces are requested with different lengths as follows:

Table 1: Requirements List for Example 2

No of pieces	length (m)
50	8.2
20	5.3
30	4.2
60	2.1

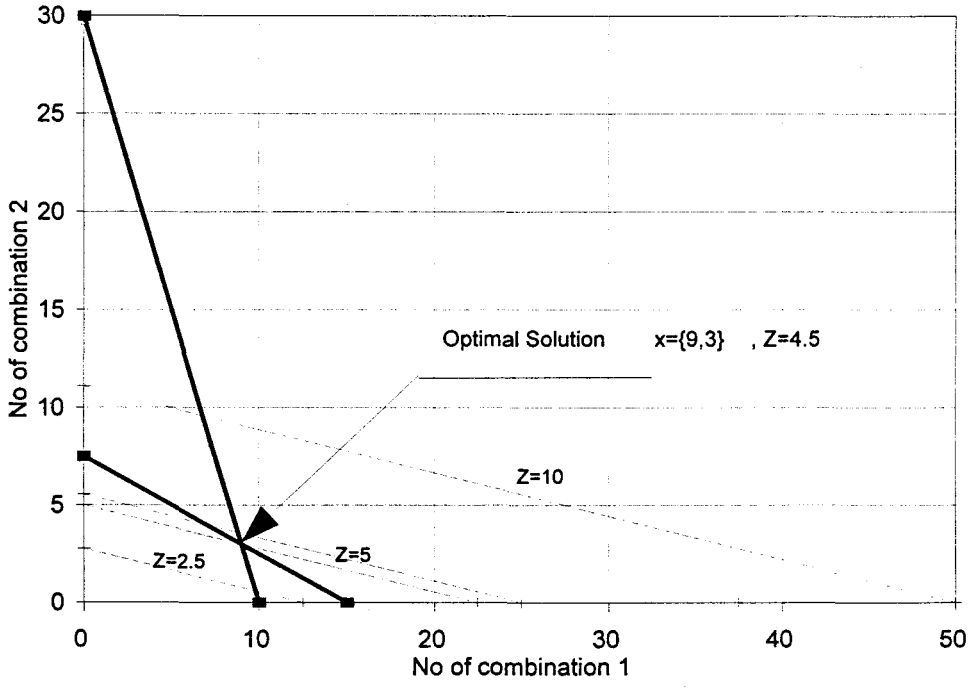

Fig. 2: Graphical solution for example 1

The number of combinations are found by arranging the array used for compiling the combinations, using the method described above, in all possible ways of similar pieces, in addition to the combination composed of the last three pieces. Using the equation for calculating the permutations $P^4\underline{4}$, the number of permutations will be $= 4x3x2x1 = 24$. The order of placing the vectors of similar pieces affects the choice of combinations. Table (2) shows the list of the permutations for arranging the series of similar pieces in a line.

This list was used to generate feasible combinations of repeated pieces. The generated combinations were more that 200, that were then screened to exclude similar and wasteful combinations. The resulting combinations were 8 in number as follows:

Table 2: List of Permutations for Example 2

8.20	5.30	4.20	2.10
8.20	5.30	2.10	4.20
8.20	4.20	5.30	2.10
8.20	4.20	2.10	5.30
8.20	2.10	4.20	5.30
8.20	2.10	5.30	4.20
5.30	8.20	4.20	2.10
5.30	8.20	2.10	4.20
5.30	4.20	8.20	2.10
5.30	4.20	2.10	8.20
5.30	2.10	4.20	8.20
5.30	2.10	8.20	4.20
4.20	8.20	5.30	2.10
4.20	8.20	2.10	5.30
4.20	5.30	8.20	2.10
4.20	5.30	2.10	8.20
4.20	2.10	5.30	8.20
4.20	2.10	8.20	5.30
2.10	8.20	5.30	4.20
2.10	8.20	4.20	5.30
2.10	5.30	8.20	4.20
2.10	5.30	4.20	8.20
2.10	4.20	5.30	8.20
2.10	4.20	8.20	5.30

Table 3: Feasible Combinations for Example 2

Serial	Combination	Waste length (m)
1	8.2	3.8
2	8.2 + 2.1	1.7
3	5.3*2	1.4
4	5.3 + 2.1*3	0.4
5	4.2 + 2.1*3	1.5
6	4.2*2 + 2.1	1.5
7	2.1*5	1.5
8	5.3 + 4.2 + 2.1	0.4

The objective function and the associated constraint matrix are give as follows:

Minimize

$$Z = 0.4x_1 + 0.4x_2 + 1.4x_3 + 1.5x_4 + 1.5x_5 + 1.5x_6 + 1.7x_7 + 3.8x_8$$

The equality constraints are:

$$
\begin{array}{llllllll}
 & & & & & x_7 & +x_8 & = 50 \\
x_1 + x_2 & +2x_3 & & & & & & = 20 \\
x_1 & & & +x_5 & +2x_6 & & & = 30 \\
x_1 + 3x_2 & & +5x_4 & +3x_5 & +x_6 & +x_7 & & = 60
\end{array}
$$

The solution to the LP problem resulted in the following:

$x^* = \{0,0,10,0,0,15,45,5\}$, and $Z^* = 132$ m. It can be shown here that only four out of eight combinations were selected to satisfy the requirements.

Example Three

This example is an actual requirements list for a huge construction project. At certain stage of execution, the request from three different diameters are as follows:

Table 4: Requirements List of Example 3

Length (m)	Number D=20 mm	Length (m)	Number D=16 mm	Length (m)	Number D=14 mm
8.27	14950	10.8	6770	9.00	22436
7.31	7480	9.00	2710	2.72	13302
5.71	4210	8.67	6770	1.47	14976
5.64	3270	2.74	13310	1.33	11635
5.62	4210	0.82	49910	1.27	13302
5.60	3270			1.22	26471
5.40	20210			.82	22160
5.00	4210			.77	63692
4.92	3270			.51	72215
4.88	7480			.36	16422
4.79	10480				
3.12	14950				
2.93	14950				
1.88	8410				
1.82	6540				
1.63	7480				
1.39	270				

The algorithm will be applied only on the third diameter for illustration. The same steps used for the previous example were used here. Only 16 combinations (variables) are used, and the number of constraints is 10. The combinations used are shown in the following table:

Table 5: Combinations for Example 3

Serial	Combination	Waste (m)
1	9.00+0.77*2+0.51*2	0.44
2	2.72*4	1.12
3	2.72*3+1.47*2	0.90
4	2.72*2+1.47*4	0.68
5	2.72+1.47*6	0.46
6	1.47*4+1.43*4	0.40
7	1.47*3+1.43*5	0.44
8	1.27*5+1.22*4	0.77
9	1.27*4+1.22*5	0.82
10	0.82*14	0.52
11	0.77*9+0.51*9	0.48
12	0.77*8+0.51*11	0.23
13	0.36*32	0.48
14	0.51*2+0.36*30	0.18
15	1.22*6+0.82*5	0.58
16	1.22*5+0.82*7	0.16

The result of the linear programming model is $z_* = 18062$, and the number of repeated combinations is $x_* = \{22436, 3186, 0, 0, 557, 2909, 0, 2660, 0, 0, 0, 1386, 0, 574, 0, 3166\}$. The solution obtained were rounded to the nearest integer. Very slight violation to the equality constraints resulted from the rounding of optimal values. However, the optimal waste length remained almost the same.

DISCUSSION

The choice of the number of feasible combinations to be used in the optimal choice step has been arbitrarily selected in the third example. The extensive search

for all possible combinations in the second example showed that the exhaustive search for feasible combinations is not justified.

In the first two examples, the solution of the linear programming problem produced integer optimal values of variables. However, the third example optimal values were fractional, and had been rounded to the nearest integer.

Two practical considerations are important here. The first is the choice of the combinations to form the terms in the equality constraints. Theoretically speaking, the greater the number of such combinations the greater the possibility of finding a feasible solution. However, some judgement should be exercised in the choice of the reduced set of combinations guided mainly by the requested number of pieces. The reduced number of combinations used in the constraint matrix has an implicit advantage of facilitating the process of cutting in the workshop. Some problems with regard to the feasibility of the solution may result from the choice of the reduced set of combinations. This can be simply investigated by changing the equality constraints into greater than or equal constraints temporarily, and use the values of slack variables to add or delete combinations to improve the choice made before.

The second consideration is the fractional solution obtained. In order to obtain an integer solution to the problem, integer programming techniques should be used. However, it was stated in (2) that: 'Solutions of integer programming problems are generally difficult, too time consuming nd expensive,Hence a practical approach is to treat all integer variable as continuous and solve the associated linear program by the simplex method. We may be fortunate to get some of the values of the variables as integers automatically, but when the simples method produces fractional solutions for some integer variable, they are generally rounded off to the nearest integer such that the constraints are not violated. This is very often used in practice, and generally produce a good integer solution close to the optimal integer solution, especially when the values of integer variables are large'. The rounded solution obtained for example No 3 proved the soundness of the quoted point of view.

CONCLUSIONS

The present paper suggests a useful algorithm for the optimal cutting of steel bars and sections used in construction to minimize the waste lengths. The algorithm is composed of two steps; the combination step is capable of generating sufficient feasible combinations, and the optimal choice step is an efficient linear programming model capable of producing the optimal numbers of repeated combinations to satisfy the requirements. The two step algorithm proposed show

that the essence of the problem is to find to number of repeated combinations and not to search for all possibilities of feasible combinations. The series of examples presented helped in clarifying the use and benefit of the algorithm and making it available to practicing engineers.

REFERENCES

1. **Al-Mohanadi, A.H.L., and Saleh, A., 1991.** "A Computer Aided Design Tool for Area Optimization Using Breadth First Search Technique", Engineering Journal of Qatar University, Vol. 4, pp. 25-33.

2. **Reklaitis, G.V., Ranindran, A., and Ragsdell, K.M., 1983.** "Engineering Optimization, Methods and Applications", John Wiley and Sons.