Engineering Journal of the University of Qatar, Vol. 12, 1999, PP. 75-89

# OPTIMUM CUTTING SCHEDULE OF REINFORCING STEEL BARS

**Ahmed B. Senouci and Aly N. El-Bahrawy**
Department of Civil Engineering,
Qatar University, Doha, QATAR

## ABSTRACT

Reinforcing steel bars are generally procured in standard lengths. The cutting of standard steel bars to the specified lengths usually yields an appreciable amount of steel waste. Minimization of steel waste problem is of interest because reinforced concrete structures are currently in extensive use. The generated steel waste depends directly upon the selected cutting schedule. A dynamic programming method is presented for the selection of cutting schedules of minimum steel waste. Two examples are presented to illustrate the computational process of the method and to compare its results with those obtained by El-Bahrawy (1995).

## INTRODUCTION

Reinforcing steel bars are fabricated to a standard length (often 12 meters), bundled, and shipped to construction sites. They are then cut to the required lengths as specified by the project requirement. The cutting process of standard steel bars (i.e., with standard lengths) usually yields an appreciable amount of steel waste which may represent a significant extra cost. The generated amount of steel waste depends directly upon the selected cutting schedule. The development of a computerized method for the selection of cutting schedules that minimize steel waste is definitely needed in order to reduce construction costs.

El-Bahrawy (1995) proposed a computerized method to minimize the cutting steel waste of reinforcing steel bars in the construction industry. The method is divided in two steps, namely the combination step and the optimal choice step. In the first step a heuristic approach is developed to generate sufficient number of feasible combinations

that is not exhaustive, while the second step finds the optimal number of repeated combinations to satisfy the requirements list. In the latter step a linear programming model is developed where the system variables are the number of times each combination has to be repeated, and the objective function is the summation of the waste lengths resulting from the chosen combinations. The proposed method presents two main shortcomings, the first is that the heuristic approach does not guarantee the generation of all feasible cutting combinations, while the second is that the linear programming model used does not guarantee an integer solution.

The present paper uses a dynamic programming approach to minimize the cutting waste length of reinforcing steel bars. Two examples are presented to illustrate the computational process of the method and to compare its results with those obtained by El-Bahrawy (1995).

## PROBLEM FORMULATION

The project requirement is usually given in the form of a table that specifies, for each bar diameter and length, the number of requested bar sections. The solution to a cutting problem consists of answering the following two questions (El-Bahrawy 1995): 1) How to cut a standard steel bar into smaller pieces to form a combination of different section lengths?, and 2) how many combinations should be used to satisfy the numbers requested from each bar section? The cutting problem can, therefore, be divided into two separate tasks. The first one determines all of the feasible cutting combinations for a standard steel bar. The second one computes the number of times each feasible cutting combination is to be repeated in order to satisfy the project requirement.

For example, the project requirement, for a specific bar diameter may consist of 40 bar sections with a length of 2.2 meters and 10 bar sections with a length of 5 meters. In Task One, two feasible cutting combinations can be selected, the first one yields 5 bar sections with a length of 2.2 meters and a waste length of 1 meter, and the second combination yields 3 bar sections with a length of 2.2 meters, 1 bar section with a length of 5 meters, and a waste length of 0.4 meter. In order to satisfy the project requirement using the selected combinations, the first cutting combination is repeated 2 times while the second combination is repeated 10 times. The total steel waste length is equal to 6 meters.

## OPTIMUM CUTTING SCHEDULE OF REINFORCING STEEL BARS

## DYNAMIC PROGRAMMING SOLUTION

The theory of dynamic programming is a powerful optimization technique based on implicit enumeration (Aris 1974, Denardo 1982, and Bertsekas 1987). The method deals effectively with problems that have a sequential type structure. It splits the problem into a sequence of stages in which lower-dimension optimization takes place. The basic dynamic programming principle includes the following steps:

1. Divide the whole problem into a family of problems of the same nature
2. Tie the optimal solutions of these problems through a recurrence equation (objective function).

The computational process is performed in two passes: a forward pass and a backward pass. In the forward pass, local optimizations (i.e., at each stage) are determined. In the backward pass, the global optimization (i.e., for the whole problem) is determined.

The dynamic programming solutions for the two tasks (i.e., the feasible cutting combinations and the optimal number of cutting combinations) are described in the following sections.

### Task One: Feasible Cutting Combinations

Task One is concerned with the determination of all feasible cutting combinations for a standard steel bar. Figure 1 presents a flowchart of the dynamic programming procedure in Task One.

**Forward procedure**

The forward procedure is designed to find all optimal and sub-optimal solutions at each stage for each of the associated states. A stage (i) represents the bar section length (i), where i=1,NSection. NSection represents the number of bar section lengths requested. A state (i,j) associated with a stage (i) represents the number of bar sections of length (i) that are selected, where j=0,….,Nbars(i). Nbars (i) represents the maximum number of bar sections of length (i) that can be cut from a standard steel bar.

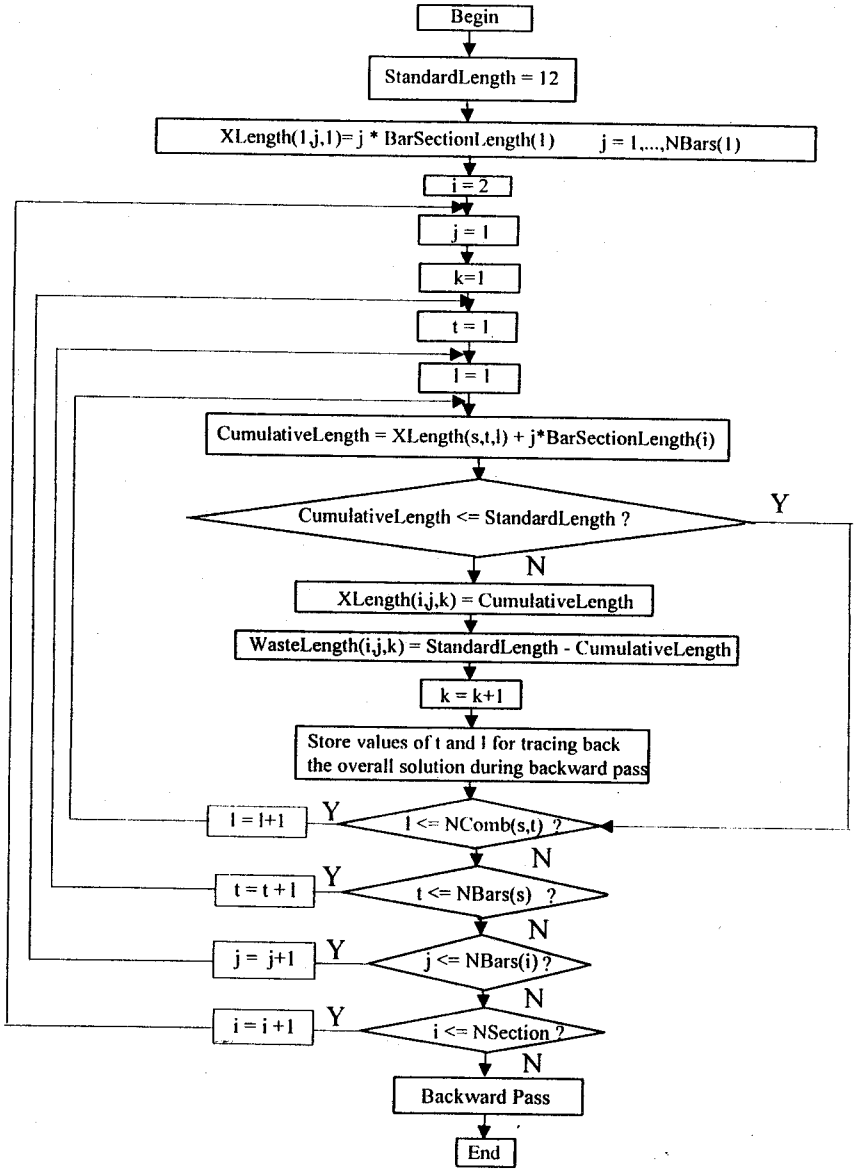


**Fig. 1. Task one flowchart**

## OPTIMUM CUTTING SCHEDULE OF REINFORCING STEEL BARS

To be able to define all optimal and sub-optimal solutions for each stage, it is necessary to introduce a steel length function. Let $M_i$ represent the number of optimal and sub-optimal solutions at state (i,j). Let XLength(i ,j ,k) denote the $k^{th}$ value of the steel length function at state (i,j), where i=1,….,NSection, j= 0,….,Nbars(i), and k =1,2,…,$M_i$.

The cumulative steel lengths XLength(i,j;k) are determined by considering every preceding state (s,t) associated with the preceding stage (s) that yields a cumulative steel length less or equal to the standard bar length (i.e. 12 meters). The cumulative steel length associated with the current state (i,j) is given by the following recurrent equation:

$$XLength(i, j, k) = XLength(s, t, l) + j * BarSectionLength(i) \qquad [1]$$

where XLength(s,t,*l*) is the $l^{th}$ cumulative steel length at the state (s,t) and BarSectionLength(i) is the section length (i).

The dynamic programming procedure begins with an evaluation of the cumulative steel lengths at the first stage (i.e., the first bar section length):

$$XLength(1, j, 1) = j * BarSectionLength(1) \qquad [2]$$

Then, the recurrent equation (Eq. 1) is applied for the states associated with the remaining stages. The state variable (t) as well as the variant (*l*) of the cumulative steel length Xlength(s,t,l) are stored for tracing back the optimum solution during the backward optimization procedure.

When the steel lengths are computed for the states associated with the last stage, the corresponding steel waste lengths are computed by subtracting the actual steel lengths from the standard length (i.e., 12 meters).

**Backward procedure**

After solving all of the local optimizations for the last stage (i.e., computing all of the steel waste lengths for the states associated with the last stage), the current state variable (i) as well as the variant (k) of cumulative steel length that yielded the minimum steel waste are selected. Then, starting from the last stage and tracing backward, the predecessor's state variable (s) and cumulative steel length variant (l)

that led to this minimum steel waste are identified. This process of examining and selecting the predecessor's state variable and cumulative steel waste variant propagates backwards until the first stage is reached.

## Task Two: Optimal Number of Cutting Combinations

Task Two computes the number of times each feasible cutting combination has to be repeated in order to satisfy the project requirement. Figure 2 presents a flowchart of the dynamic programming procedure in Task Two.

## Forward Procedure

In the dynamic programming formulation, a stage (i) represents a standard steel bar. A state (i,j), which is associated with stage (i), represents the selected cutting combination (j) determined in Task One.

The objective of the procedure is designed to find all optimal and sub-optimal solutions at each stage for each of the possible states. In order to achieve this objective two arrays are used: cumulative waste length Waste and cumulative number of bar section length TSL.

## Cumulative Waste Length Array:

The array stores the cumulative waste lengths generated at every state associated with each stage. The $k^{th}$ cumulative waste length for state (i,j) is denoted Waste(i,j,k). In this expression, i = 1, ....., N ; j = 1,...., Ncomb; k=1,...., M(i,j); where Ncomb is the total number of feasible cutting combinations; and M(i,j) is the number of optimal and sub-optimal solutions at state (i,j).

The cumulative waste lengths Waste(i,j,k) are determined by considering every state (s,t) associated with the preceding stage (s). The $k^{th}$ cumulative waste length associated with the current state (i,j) is given by the following recurrent equation:

$$\text{Waste}(i,j,k) = \text{Waste}(s,t,l) + \text{WastePerCutComb}(j) \qquad [3]$$

where Waste(s,t,*l*) is the $l^{th}$ cumulative Waste length at the state (s,t) and WastePerCutComb(j) is the steel waste generated by the cutting combination (j).

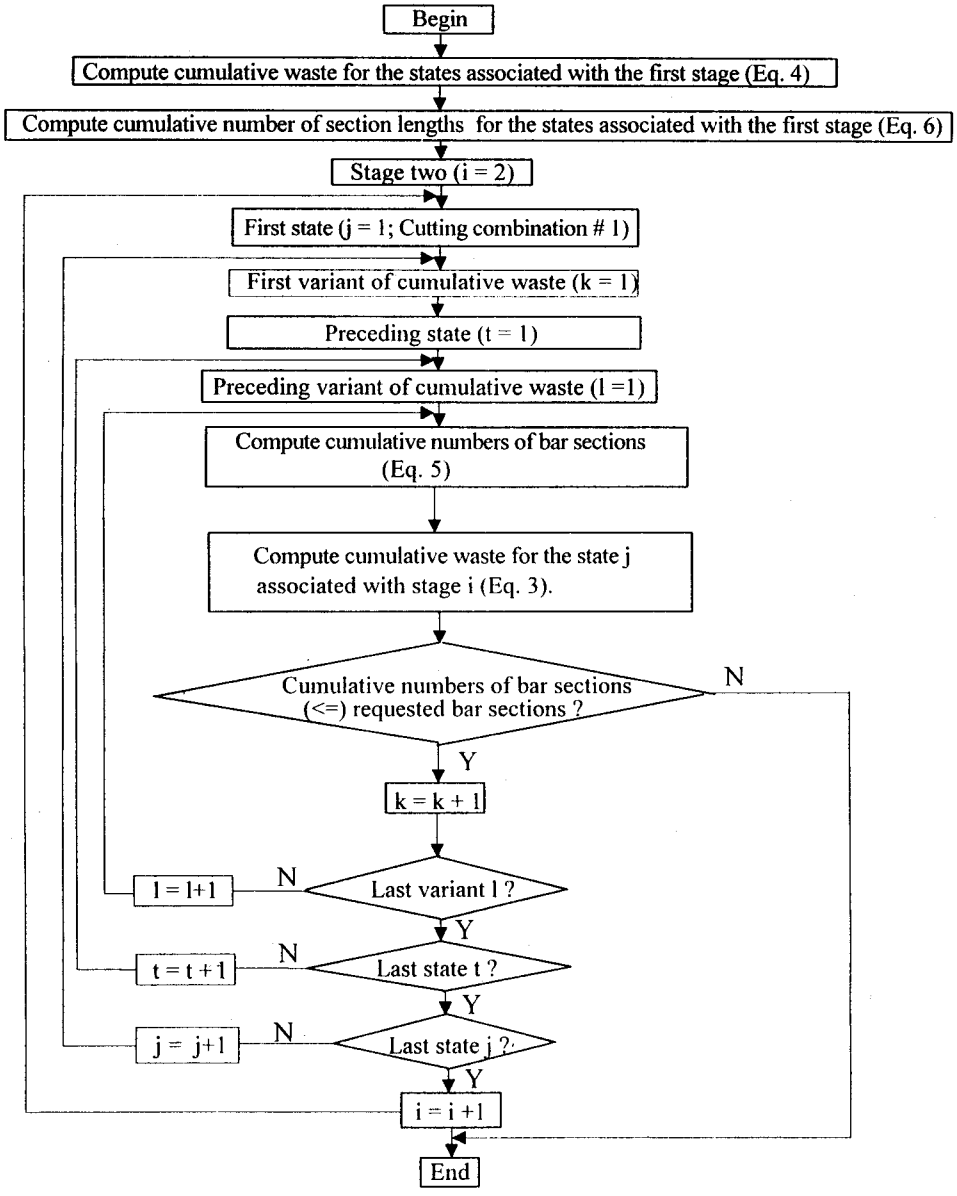


**Fig. 2. Task two flowchart**

The dynamic programming procedure begins with evaluating of the cumulative steel waste lengths at the first stage (i.e., the first standard length steel bar):

$$\text{Wast(i,j,I)} = \text{WastePerCutComb(j)} \quad [4]$$

Then, the recurrent equation (Eq. 3) is applied for the states associated with the remaining stages. It should be noted that the state variable (t) and the variant (*l*) of steel waste are stored for tracing back the optimum solution during the backward optimization procedure.

The computational process is continued until the requested number of bar sections is reached. Thus, it is necessary to keep track of the cumulative number of bar sections.

## Cumulative Number of Bar Sections:

The array stores the cumulative number of bar sections. TSL(i,j,k,n) denotes the cumulative number of bar section length n associated with the cumulative waste length Waste(i,j,k).

The cumulative number of bar section lengths TSL(i,j,k,n) are computed using the following equation:

$$\text{TSL}(i, j, k, n) = \text{TSL}(s, t, l, n) + \text{NumSectionPerCutComb}(j, n) \quad [5]$$

where TSL(s,t,*l*,n) is the cumulative number of bar section length and NumSectionPerCutComb(j,n) is the number of bar sections length (n) that are generated by the cutting combination (j).

The cumulative number of section lengths at the first stage is obtained using the following equation:

$$\text{TSL}(i, j, k, n) = \text{NumSectionPerCutComb}(j, n) \quad [6]$$

## Backward Procedure

This procedure is similar to the one performed in Task One.

## ILLUSTRATIVE EXAMPLE 1

The following example is used to illustrate the details of the dynamic programming solution. The requirement list, for a specific steel bar diameter, consists of 6 bar sections with a length of 2.5 meters, 5 bar sections with a length of 4.3 meters, and 3 bar sections with a length of 5 meters.

### Task One: Combinations of Section Lengths

Five states are associated with the first stage while three states are associated with the second and third stage (Figure 3). The forward procedure starts by calculating the cumulative steel lengths for the first stage. The cumulative steel length for the first state is equal to zero. The cumulative steel length for the second state, which is equal to 2.5, is computed using Eq. 2. The computation process is continued in the same manner for the remaining states associated with stage 1. Then, the cumulative steel lengths for the states associated with stage 2 are computed using Eq. 1. The preceding state variable (s) as well as the variant ($I$) of cumulative steel length are stored for tracing back the overall solution during the backward pass. The cumulative steel lengths for the states associated with the third stage are computed in a manner similar to the one used for the states associated with the second stage. Table 1 summarizes the computation results for the first, second, and third stages.

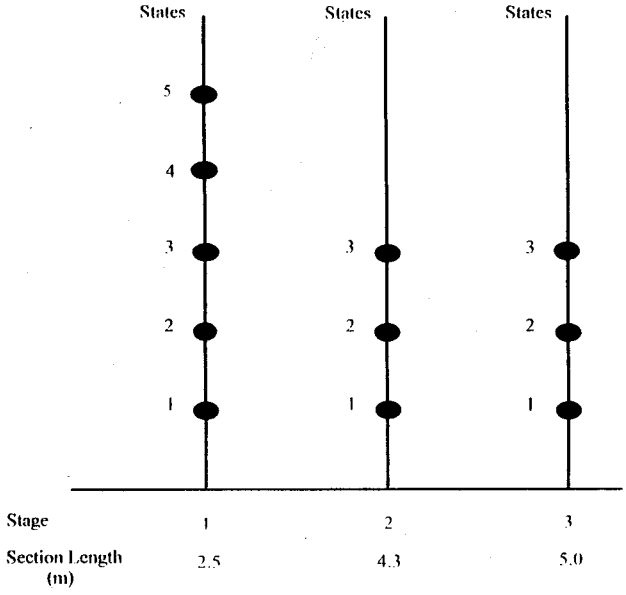


**Fig. 3. Stages and associated states for task one**

Table 1. Computation results for task one of example 1

| Current Stage | | | | Preceding Stage | | | | Current Stage | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Steel | Bar | | | Steel | Cum. | Cum. | Optimum Preceding | |
| Stage No. i | State No. j | Length Variable k | Section Length (m) | Stage No. s | State Variable t | Length Variable l | Steel Length (m) | Steel Length (m) | State t | Length Variable l |
| 1 | 1 | 1 | 0.0 | ----- | ----- | ----- | ----- | 0.0 | ----- | ----- |
| | 2 | 1 | 2.5 | ----- | ----- | ----- | ----- | 2.5 | ----- | ----- |
| | 3 | 1 | 5.0 | ----- | ----- | ----- | ----- | 5.0 | ----- | ----- |
| | 4 | 1 | 7.5 | ----- | ----- | ----- | ----- | 7.5 | ----- | ----- |
| | 5 | 1 | 10.0 | ----- | ----- | ----- | ----- | 10.0 | ----- | ----- |
| 2 | 1 | 1 | 0.0 | 1 | 1 | 1 | 0.0 | 0.0 | 1 | 1 |
| | 1 | 2 | 0.0 | 1 | 2 | 1 | 2.5 | 2.5 | 2 | 1 |
| | 1 | 3 | 0.0 | 1 | 3 | 1 | 5.0 | 5.0 | 3 | 1 |
| | 1 | 4 | 0.0 | 1 | 4 | 1 | 7.5 | 7.5 | 4 | 1 |
| | 1 | 5 | 0.0 | 1 | 5 | 1 | 10.0 | 10.0 | 5 | 1 |
| | 2 | 1 | 4.3 | 1 | 1 | 1 | 0.0 | 4.3 | 1 | 1 |
| | 2 | 2 | 4.3 | 1 | 2 | 1 | 2.5 | 6.8 | 2 | 1 |
| | 2 | 3 | 4.3 | 1 | 3 | 1 | 5.0 | 9.3 | 3 | 1 |
| | 2 | 4 | 4.3 | 1 | 4 | 1 | 7.5 | 11.8 | 4 | 1 |
| | 3 | 1 | 8.6 | 1 | 1 | 1 | 0.0 | 8.6 | 1 | 1 |
| | 3 | 2 | 8.6 | 1 | 2 | 1 | 2.5 | 11.1 | 2 | 1 |
| 3 | 1 | 1 | 0 | 2 | 1 | 1 | 0.0 | 0.0 | 1 | 1 |
| | 1 | 2 | 0 | 2 | 1 | 2 | 2.5 | 2.5 | 1 | 2 |
| | 1 | 3 | 0 | 2 | 1 | 3 | 5.0 | 5.0 | 1 | 3 |
| | 1 | 4 | 0 | 2 | 1 | 4 | 7.5 | 7.5 | 1 | 4 |
| | 1 | 5 | 0 | 2 | 1 | 5 | 10.0 | ***10.0*** | ***1*** | ***5*** |
| | 1 | 6 | 0 | 2 | 2 | 1 | 4.3 | 4.3 | 2 | 1 |
| | 1 | 7 | 0 | 2 | 2 | 2 | 6.8 | 6.8 | 2 | 2 |
| | 1 | 8 | 0 | 2 | 2 | 3 | 9.3 | ***9.3*** | ***2*** | ***3*** |
| | 1 | 9 | 0 | 2 | 2 | 4 | 11.8 | ***11.8*** | ***2*** | ***4*** |
| | 1 | 10 | 0 | 2 | 3 | 1 | 8.6 | 8.6 | 3 | 1 |
| | 1 | 11 | 0 | 2 | 3 | 2 | 11.1 | ***11.1*** | ***3*** | ***2*** |
| | 2 | 1 | 5 | 2 | 1 | 1 | 0.0 | 5.0 | 1 | 1 |
| | 2 | 2 | 5 | 2 | 1 | 2 | 2.5 | 7.5 | 1 | 2 |
| | 2 | 3 | 5 | 2 | 1 | 3 | 5.0 | ***10.0*** | ***1*** | ***3*** |
| | 2 | 4 | 5 | 2 | 2 | 1 | 4.3 | 8.3 | 2 | 1 |
| | 2 | 5 | 5 | 2 | 2 | 2 | 6.8 | ***11.8*** | ***2*** | ***2*** |
| | 3 | 1 | 10 | 2 | 1 | 1 | 0.0 | 5.0 | 1 | 1 |
| | 3 | 2 | 10 | 2 | 1 | 2 | 2.5 | 7.5 | 1 | 2 |
| | 3 | 3 | 10 | 2 | 1 | 3 | 5.0 | ***10.0*** | ***1*** | ***3*** |
| | 3 | 4 | 10 | 2 | 2 | 1 | 4.3 | 8.3 | 2 | 1 |
| | 3 | 5 | 10 | 2 | 2 | 2 | 6.8 | ***11.8*** | ***2*** | ***2*** |

When the cumulative steel lengths are computed for all the states associated with the last stage, the corresponding steel waste lengths are computed by substracting the cumulative steel lengths from the standard length (i.e., 12 meters).

The backward procedure starts with the selection of the state variable (i) as well as the variant (k) of cumulative steel length that yielded the minimum steel waste at the last stage. Then, starting from the last stage and tracing backwards, the predecessor's state variable (s) and cumulative steel length variant (*l*) that led to this minimum steel waste are identified. This process of examining and selecting the predecessor's state variable and cumulative steel waste variant propagates backwards until the first stage is reached. Table 2 summarizes the feasible cutting combinations.

## Task Two: Optimal Number of Cutting Combinations

In Task Two, all of the feasible combinations listed in Table 2 must be considered in the computation procedure in order to guarantee an optimum solution. However, the first two combinations are found, by inspection, to yield the optimum solution. Therefore, only these two combinations are considered herein in order to simplify the computational procedure.

As shown in Figure 4, two states are associated with each stage. The states represent the cutting combinations while the stages represent the standard steel bars. The forward procedure starts by calculating the cumulative steel waste lengths for the first stage. The cumulative waste and cumulative number of bar sections for the states associated with the first stage are computed using Eqs. 4 and 6, respectively. On the other hand, the cumulative waste and cumulative number of bar sections for the states associated with the second and third stages are computed using Eqs. 3 and 5, respectively. The preceding state variable (s) as well as the variant (l) are stored for tracing back the optimum solution in the backward procedure. Table 3 summarizes the computation results for the first, second, and third stages.

In the backward procedure, the state variable as well as the variant of cumulative steel waste that yielded the minimum steel waste are selected. Then, starting from the last stage and tracing backwards, the predecessor's state variable and cumulative waste variant that led to this minimum steel waste are identified. This process of examining and selecting the predecessor's state variable and cumulative steel waste variant propagates backwards until the first stage is reached. Table 4 summarizes the optimal cutting combinations obtained in Task Two.

Table 2. Feasible cutting combinations for task one of example 1

| Combination Number | Steel Length (m) | Steel Waste Length (m) | Number of Bar Sections | | |
|---|---|---|---|---|---|
| | | | No.1 (2.5 m) | No.2 (4.3 m) | No.3 (5 m) |
| 1 | 11.8 | 0.2 | 1 | 1 | 1 |
| 2 | 11.8 | 0.2 | 3 | 1 | 0 |
| 3 | 11.8 | 0.9 | 1 | 2 | 0 |
| 4 | 10.0 | 2.0 | 0 | 0 | 2 |
| 5 | 10.0 | 2.0 | 2 | 0 | 1 |
| 6 | 10.0 | 2.0 | 4 | 0 | 0 |
| 7 | 9.3 | 2.7 | 2 | 1 | 0 |
| 8 | 9.3 | 2.7 | 0 | 1 | 1 |
| 9 | 8.6 | 3.4 | 0 | 2 | 0 |

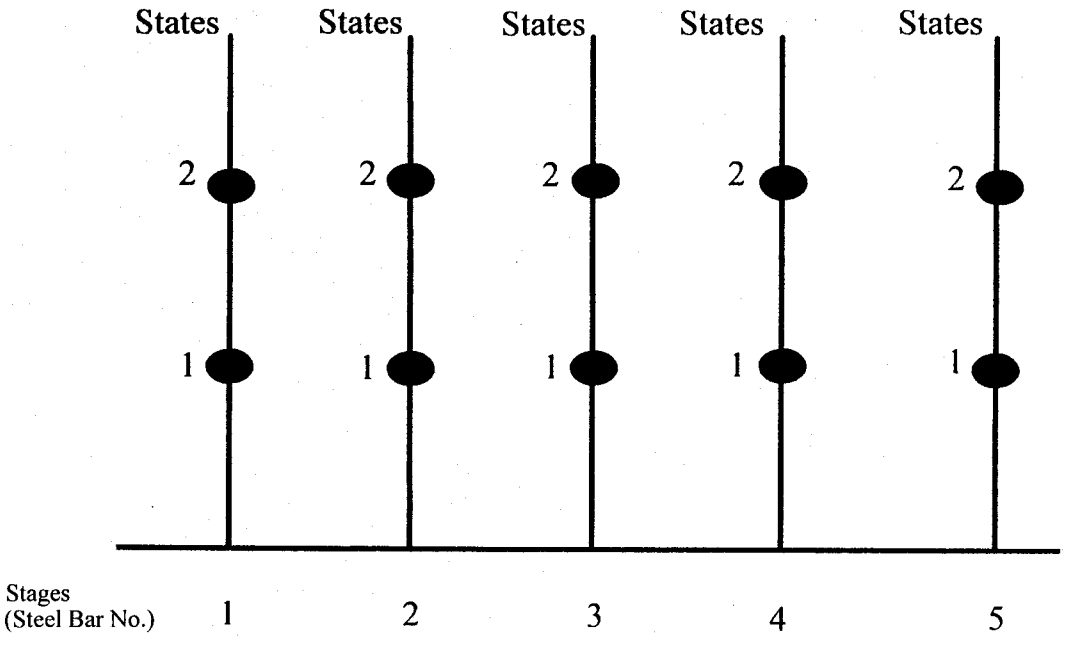


**Fig. 4. Stages and associated sates for task two**

Table 3. Computation results for task two of example 1

| Current Stage | | | | | | | Preceding Stage | | | | | | | | Current Stage | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Number of Sections | | | | | | | Number of Sections | | | | | Number of Sections | | | Optimum Preceding | | |
| Stage No. i | State No. j | Steel Waste Var. k | No.1 (2.5 m) | No.2 (4.3 m) | No.3 (5 m) | Waste Per Comb. m | Stage No. s | State Var. t | Steel Waste Var. l | No.1 (2.5 m) | No.2 (4.3 m) | No.3 (5 m) | Waste Per Comb. (m) | Cum. Steel Waste (m) | No.1 (2.5 m) | No.2 (4.3 m) | No.3 (5 m) | State t | Waste Variable l | Cum. Steel Waste (m) |
| 1 | 1 | 1 | 1 | 1 | 1 | 0.2 | ----- | ----- | ----- | ----- | ----- | ----- | ----- | ----- | 1 | 1 | 1 | ----- | ----- | 0.2 |
| | 2 | 1 | 3 | 1 | 0 | 0.2 | ----- | ----- | ----- | ----- | ----- | ----- | ----- | ----- | 3 | 1 | 0 | ----- | ----- | 0.2 |
| 2 | 1 | 1 | 1 | 1 | 1 | 0.2 | 1 | 1 | 1 | 1 | 1 | 1 | 0.2 | 0.2 | 2 | 2 | 2 | 1 | 1 | 0.4 |
| | 1 | 2 | 1 | 1 | 1 | 0.2 | 1 | 2 | 1 | 3 | 1 | 0 | 0.2 | 0.2 | 4 | 2 | 1 | 2 | 1 | 0.4 |
| | 2 | 1 | 3 | 1 | 0 | 0.2 | 1 | 1 | 1 | 1 | 1 | 1 | 0.2 | 0.2 | 4 | 2 | 1 | 1 | 1 | 0.4 |
| | 2 | 2 | 3 | 1 | 0 | 0.2 | 1 | 2 | 1 | 3 | 1 | 0 | 0.2 | 0.2 | 6 | 2 | 0 | 2 | 1 | 0.4 |
| 3 | 1 | 1 | 1 | 1 | 1 | 0.2 | 2 | 1 | 1 | 1 | 1 | 1 | 0.2 | 0.4 | 3 | 3 | 3 | 1 | 1 | 0.6 |
| | 1 | 1 | 1 | 1 | 1 | 0.2 | 2 | 1 | 2 | 1 | 1 | 1 | 0.2 | 0.4 | ***5*** | ***3*** | ***2*** | ***1*** | ***2*** | ***0.6*** |
| | 1 | 1 | 1 | 1 | 1 | 0.2 | 2 | 2 | 1 | 3 | 1 | 0 | 0.2 | 0.4 | 7 | 3 | 1 | 2 | 1 | 0.6 |
| | 1 | 1 | 1 | 1 | 1 | 0.2 | 2 | 2 | 2 | 3 | 1 | 0 | 0.2 | 0.4 | 9 | 3 | 0 | 2 | 2 | 0.6 |
| | 2 | 1 | 3 | 1 | 0 | 0.2 | 2 | 1 | 1 | 1 | 1 | 1 | 0.2 | 0.4 | ***5*** | ***3*** | ***2*** | ***1*** | ***1*** | ***0.6*** |
| | 2 | 2 | 3 | 1 | 0 | 0.2 | 2 | 1 | 2 | 1 | 1 | 1 | 0.2 | 0.4 | 7 | 3 | 1 | 1 | 2 | 0.6 |
| | 2 | 3 | 3 | 1 | 0 | 0.2 | 2 | 2 | 1 | 3 | 1 | 0 | 0.2 | 0.4 | 7 | 3 | 1 | 2 | 1 | 0.6 |
| | 2 | 4 | 3 | 1 | 0 | 0.2 | 2 | 2 | 2 | 3 | 1 | 0 | 0.2 | 0.4 | 9 | 3 | 0 | 2 | 2 | 0.6 |

Table 4. Optimal number of cutting combinations (Task two of example 1)

| Number of Cutting Combinations | | Number of Sections | | | Total Waste |
|---|---|---|---|---|---|
| #1 | #2 | 2.5 m | 4.3 m | 5.0 m | (m) |
| 2 | 1 | 5 | 3 | 2 | 0.6 |

## ILLUSTRATIVE EXAMPLE 2

The following example, which was reported by El-Bahrawy (1995), is given to compare the results of the dynamic programming solution with those of the linear programming.

A construction project requires, for a specific steel bar size, 50 bar sections with a length of 8.2 meters, 20 bar sections with a length of 5.3 meters, 30 bar sections with a length of 4.2 meters, and 60 bar sections with a length of 2.1 meters.

The first step in the dynamic programming solution of the problem is the enumeration of the feasible cutting combinations. Table 5 summarizes these cutting combinations. The second step is the determination of the optimal number of cutting combinations. Table 6 summarizes the solution found.

The final solution obtained is the same as the linear programming solution. However, the dynamic programming solution provides two optimal combinations instead of one. This additional combination gives the engineer an opportunity to choose between two optimal solutions.

Table 5. feasible cutting combination for task one of example 2

| Combination Number | Combinations | Waste Length (m) |
|---|---|---|
| 1 | 2 * 5.30 | 1.4 |
| 2 | 2.10 + 2 * 4.20 | 1.5 |
| 3 | 2.10 + 1 * 8.20 | 1.7 |
| 4 | 8.2 | 3.8 |
| 5 | 5.3 + 3 * 2.1 | 0.4 |
| 6 | 4.2 + 2.1 * 3 | 1.5 |
| 7 | 5 * 2.1 | 1.5 |
| 8 | 5.3 + 4.2 + 2.1 | 0.4 |

Table 6. Optimal number of cutting combinations (Task two of example 2)

| Solution | Number of Cutting Combinations | | | | Number of Sections | | | | Total Waste |
|---|---|---|---|---|---|---|---|---|---|
| Number | #1 | #2 | #3 | #4 | 2.1 m | 4.2 m | 5.3 m | 8.2 m | (m) |
| 1 | 10 | 15 | 46 | 4 | 61 | 30 | 20 | 50 | 132 |
| 2 | 10 | 15 | 45 | 5 | 60 | 30 | 20 | 50 | 132 |

## CONCLUSIONS

The present offers a dynamic programming solution to the problem of optimal cutting schedule of reinforcing steel bars. The solution is divided into two tasks. The first one (Task One) finds the feasible cutting combination while the second one (Task Two) finds the optimal number of cutting combinations. Two illustrative examples were given to demonstate the details of the dynamic programming solution and to compare the results with the linear programming solution presented by El-Bahrawy (1995). The proposed solution finds all possible optimum cutting combinations. The disadvantage, however, is the difficulty in solving large size problems. With the availability of more powerful computers, the latter problem can be easily overcome.

## REFERENCES

1. **Aris, R..** (1974). Discrete Dynamic Programming: an introduction to the optimization of staged processes. Blaisdell Publication Corporation, New York.

2. **Bertsekas, D. P.** (1987). Dynamic Programming: deterministic and stochastic models, Englewood Cliffs, New Jersey, USA.

3. **Denardo, E. V.** (1982). Dynamic Programming: models and applications, Englewood Cliffs, New Jersey.

4. **El-Bahrawy, A. N.** (1995). An algorithm to minimize unused lengths of steel bars and sections. Engineering Journal of University of Qatar, Vol. 8, pp.185-196.