CHAPTER 3

REVISED SIMPLEX METHOD AND DATA STRUCTURES

3.1 INTRODUCTION

While solving a linear programming problem, a systematic search is made to find a non-negative vector $X$ which extremizes a linear objective function

$$Z = \sum_{j=1}^{n} c_j x_j$$  \hspace{1cm} (3.1)

such that it satisfies a set of linear constraints of the form

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i; \hspace{0.5cm} i = 1, 2, \ldots, m; \hspace{0.5cm} x_j \geq 0;$$  \hspace{1cm} (3.2)

The parameter $c_j$ is the contribution/cost coefficient associated with unit output of the activity $x_j$. The $b_i$'s are the limited available resources in the form of men, money, machine and material. The problem is to find that vector $X$ which satisfies the relationships (3.2) and extremizes the linear objective function (3.1).

To find a solution to the above problem, revised simplex method can be used. The revised simplex method uses the same basic principles of the regular simplex method. But at each iteration, the entries in the entire tableau are not calculated. The relevant information for moving from one basic feasible solution to another is directly generated from the original set of equations.
3.2 REVISED SIMPLEX PROCEDURE

The revised simplex method saves both storage and computational time compared to the simplex method. Unlike the original simplex method, only the inverse of the current basis is maintained to generate the next inverse. All other quantities except \( X_B \) are computed from their definitions as and when necessary. Eventhough \( X_B \) can be computed from its definition, it is more economical to transform it at each stage. When we have to determine the reduced cost coefficients, we simply determine

\[
(z_j - c_j) = c_B^{-1} p_j - c_j
\]

If all \( (z_j - c_j) \geq 0 \), then optimal solution is obtained.

3.2.1 Detailed algorithm

Again taking the standard LP model

Extremize \( Z = C X \)

Subject to \( A X \geq P_0; \ X \geq 0 \) \hspace{1cm} (3.3)

Where

\[
A = \begin{bmatrix}
a_{11} & a_{12} & \cdots & a_{1n} \\
a_{21} & a_{22} & \cdots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
a_{m1} & a_{m2} & \cdots & a_{mn} \\
\end{bmatrix}
\]

(\( m \times n \))
Let the columns corresponding to the matrix $A$ be denoted by $P_1, P_2, P_3, ..., P_n$ where

\[
P_0 = \begin{bmatrix}
  b_1 \\
  b_2 \\
  . \\
  \vdots \\
  b_m
\end{bmatrix}, \quad
X = \begin{bmatrix}
  x_1 \\
  x_2 \\
  . \\
  \vdots \\
  x_n
\end{bmatrix}
\]

\[
C = \begin{bmatrix}
  c_1 \\
  c_2 \\
  . \\
  \vdots \\
  c_n
\end{bmatrix}
\]

Let the vector $X$ be partitioned as

\[
X = \begin{bmatrix}
  X_B \\
  X_N
\end{bmatrix}
\]

Where $X_B$ corresponds to the basic variables and $X_N$ to the non-basic variables.
The revised simplex procedure solves repeatedly a set of linear algebraic equations of the form

\[ B X_B = P_0 \]  \hspace{1cm} (3.4)

and finds the value of the objective function

\[ Z = C_B X_B \]  \hspace{1cm} (3.5)

The set of equations (3.4) and (3.5) may be put in matrix notation as

\[
\begin{bmatrix}
1 & -C_B \\
0 & B
\end{bmatrix}
\begin{bmatrix}
Z \\
X_B
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
P_0
\end{bmatrix} \hspace{1cm} (3.6)
\]

Let

\[ M = \begin{bmatrix}
1 & -C_B \\
0 & B
\end{bmatrix} \]

Rewriting the equation (3.6) as

\[ M \times \begin{bmatrix}
Z \\
X_B
\end{bmatrix} = 
\begin{bmatrix}
0 \\
P_0
\end{bmatrix} \]

The solution vector in terms of the matrix M is

\[
\begin{bmatrix}
Z \\
X_B
\end{bmatrix} = M^{-1} \begin{bmatrix}
0 \\
P_0
\end{bmatrix} \hspace{1cm} (3.7)
\]
\[ M^{-1} \text{ exists if and only if the basis matrix } B \text{ is nonsingular. Hence} \]

\[
M^{-1} = \begin{bmatrix}
1 & CB^{-1} \\
0 & B^{-1}
\end{bmatrix}
\]  

(3.8)

The solution vector is given by

\[
\begin{bmatrix}
Z \\
X_B
\end{bmatrix} = \begin{bmatrix}
1 & CB^{-1} \\
0 & B^{-1}
\end{bmatrix} \begin{bmatrix}
0 \\
P_0
\end{bmatrix}
\]

\[
= \begin{bmatrix}
CB^{-1}P_0 \\
B^{-1}P_0
\end{bmatrix}
\]  

(3.9)

The solution is based mainly in identifying \( B^{-1} \) for the current iteration.

The basis matrix \( B \) is different from the previous or succeeding basis matrix by only one column and so is the \( M \) matrix. Let the matrices \( M_c \) and \( M_n \) correspond to the current and next iterations of the revised simplex procedure. The \( M_n^{-1} \) may be obtained from \( M_c^{-1} \) using linear algebra and this eliminates the computation for direct inversion.

The procedure for determining \( M_n^{-1} \) from \( M_c^{-1} \) is summarized below.

Let the identity matrix \( I_{m+1} \) be represented as

\[
I_{m+1} = (e_1, e_2, \ldots, e_i, \ldots, e_{m+1})
\]
where \( \mathbf{e}_i \) is a unit vector with a unit element at the \( i \)th place and the rest zero. Let \( x_j \) and \( x_r \) be the entering and leaving variables at the start of any iteration. The \( M_n^{-1} \) can then be computed using the relationship

\[
M_n^{-1} = E_0 M_c^{-1}
\]  
(3.10)

where the transformation matrix \( E_0 \) has \( m \) unit vectors and only one nonunit vector corresponding to the entering variable.

\[
\begin{bmatrix}
\mathbf{0} \\
\mathbf{0} \\
\vdots \\
1 \\
\mathbf{0} \\
\vdots \\
\mathbf{0}
\end{bmatrix}
\begin{bmatrix}
\alpha_{0j} \\
\alpha_{1j} \\
\vdots \\
\alpha_{rj} \\
\vdots \\
\alpha_{mj}
\end{bmatrix}
\begin{bmatrix}
-1/ \alpha_{rj} \\
-1/ \alpha_{rj} \\
\vdots \\
-1/ \alpha_{rj} \\
\vdots \\
-1/ \alpha_{rj}
\end{bmatrix}
\]

leaving vector \hspace{0.5cm} entering vector \hspace{0.5cm} multiplier

where \( \alpha_{0j} = z_j - c_j \)

and \( \alpha_{ij} = B_i^{-1} p_j \); \( i = 1, 2, \ldots, m \)

The entering vector is multiplied by the multiplier to generate the non-unit column \( (\eta) \) of the \( E_0 \) matrix.
The $E$ matrix now formed as

$$E_0 = (e_1, e_2, e_3, ..., e_r, n, e_{m+1})$$

where $e_1, e_2, ..., e_r, e_{m+1}$ are all unit vectors and $n$ alone is the non unit vector. Thus the $M^{-1}$ is constructed using $M_c^{-1}$ and the transformation matrix $E_0$.

The various steps involved in the revised simplex procedure are as follows:

**Step 1: Determination of the entering vector $P_j$.**

$$z_j - c_j = C_B B^{-1} p_j - c_j = (1, C_B B^{-1}) [-c_j]$$  \[ (3.11) \]

The most promising vector enters the basis. Otherwise, if all $(z_j - c_j) \geq 0$, then the optimal solution is attained.
Step 2: Determination of the leaving vector $P_r$. When the entering vector is $P_j$ and the current basis matrix is $B_c$, the leaving vector must correspond to

$$\theta = \min \left[ \frac{B_c^{-1}P_0}{B_c^{-1}P_j}; a_{kj} > 0 \right]$$

where $a_{kj} = B_c^{-1}P_j$ and

$$k = 1, 2, ..., n$$

If all $a_{kj} \leq 0$ then the problem has no bounded solution.

Step 3: Determination of the next basic solution. The $E_0$ matrix is constructed as explained above and the $M_n^{-1}$ is computed by using (3.10) and the solution vector is defined by (3.7). Thus $B_n^{-1}$ is expressed as a function of $B_c^{-1}$ and the processing is returned to step 1.

This procedure is repeated until the optimal solution is reached.

3.3 DATA STRUCTURE IN REVISED SIMPLEX

Usually large scale real life linear programming problems are of high sparsity in nature. If the constraint coefficient matrix is represented by sequential mapping then it consumes lot of additional memory for storing zero entries. Again checks were made for zero/nonzero entries before every multiplication or division operation which consumed additional machine time. Hence Revised simplex algorithm which was described earlier, exploits the concept
of Data structure using linked list to represent the matrices which are involved in this algorithm.

The different types of representation of matrices depending on their usage is described in chapter 2. Revised simplex algorithm employs the following matrices:

i. Constraint coefficient matrix $A$
ii. $M^{-1}$ inverse matrix and
iii. The transformation matrix $E_0$.

Depending on the usage of the matrices, they are represented in different manner.

3.3.1 Representation of $A$ Matrix

This matrix is used for the following purposes:

i. Determination of $z_j - c_j$ vector
ii. To find $B^{-1} P_j$ where $j = 1, 2, \ldots, n$.

The constraint coefficient matrix is split up into $n$ column vectors as $P_1, P_2, \ldots, P_n$. Then the vectors corresponding to the promising variables are used for post multiplication. Since $A$ matrix is always used for post multiplication it is represented using columnwise circularly linked list as explained in Chapter 2.

3.3.2 Representation of $M^{-1}$ Matrix

$M^{-1}$ matrix is used for the following purposes:

i. Determination of $z_j - c_j$ using first row of $M^{-1}$ inverse matrix
ii. To find $B^{-1}P$ and

iii. To update the new $M^{-1}$ matrix using the relationship

$$M_n^{-1} = E_0 M_0^{-1}$$

In the cases (i) and (ii) this matrix is used for premultiplication and in case (iii) it is used for postmultiplication. Hence the $M^{-1}$ matrix is represented using Multiply linked list.

### 3.3.3 Representation of the Transformation Matrix $E_0$

Since the transformation matrix contains all unit vectors except $n$ vector it is not represented physically. Only the nonunit vector $n$ is represented using an array.

### 3.4 NUMERICAL EXAMPLE

Max $Z = x_1 + 3x_2$

Subject to $x_1 \leq 5$

$x_1 + 2x_2 \leq 10$

$x_2 \leq 4$

$x_1, x_2 \geq 0$
The constraint coefficient matrix $A$ is given by

$$
A = \begin{bmatrix}
1 & 0 \\
1 & 2 \\
0 & 1
\end{bmatrix}
$$

The column wise representation of $A$ matrix is shown in Figure 3.1.

$M^{-1}$ matrix is given by

$$
M^{-1} = \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
$$

Multiply linked list representation of $M^{-1}$ matrix is shown in Figure 3.2.

The $M^{-1}$ matrix has $m + 1$ rowwise circular lists and $m$ columnwise circular lists. For the first column of $M^{-1}$ matrix circular list representation is not necessary since all the elements in that column are always zero except the first entry (it has the value 1 only). To start with, the first row has no nonzero entries since initially $C_B B^{-1} = 0$. Hence the right pointer of the first row head node points to itself.
Figure 3.1 Columnwise circulary linked list
Figure 3.2 Multiply linked list
Iteration 1:

Step 1: \((z_j - c_j) = [-1 -3]\)

\(z_2 - c_2\) is the most negative and hence \(x_2\) is the most promising variable.

\[
B^{-1}p_2 = \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix}, \quad B^{-1}p_0 = \begin{bmatrix} 5 \\ 10 \\ 4 \end{bmatrix}
\]

Using the minimum ratio test the leaving variable is \(S_3\).

\[
\eta_{\text{old}} = \begin{bmatrix} -3 \\ 0 \\ 2 \\ 1 \end{bmatrix}, \quad \eta_{\text{new}} = \begin{bmatrix} 3 \\ 0 \\ -2 \\ 1 \end{bmatrix}
\]

The \(M^{-1}\) is updated using the relationship

\[M^{-1}_{\text{next}} = E_0M^{-1}_{\text{current}}\]

In order to reduce the number of computations in the above matrix multiplication, the following procedure is adopted. Instead of forming the \(E_0\) matrix and performing the matrix multiplication, the \(\eta\) vector is used as described below. While entering the \(\eta\) vector in a particular column, it is to be verified whether any other column has already
Figure 3.3 Multiply linked list

HN: Head Node  RN: Row Number  CN: Column Number
VAL: Value of the element  RLINK/DLINK: Pointers
Iteration 2:

Step 1: \( z_j - c_j = \begin{pmatrix} -1 & 0 \end{pmatrix} \)

\( z_1 - c_1 \) is having negative value, i.e., \( x_1 \) is the promising variable to enter the basis.

\[
B^{-1} p_1 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, \quad B^{-1} p_0 = \begin{bmatrix} 10 \\ 2 \\ 4 \end{bmatrix}
\]

Using the minimum ratio test, \( s_2 \) is the leaving variable.

\[
\eta_{\text{old}} = \begin{bmatrix} -1 \\ 1 \\ 1 \\ 0 \end{bmatrix}; \quad \eta_{\text{new}} = \begin{bmatrix} 1 \\ -1 \\ 1 \\ 0 \end{bmatrix}
\]

The new \( M^{-1} \) matrix is updated using the relationship (3.12) and the \( M^{-1} \) matrix and the multiply linked list representation is shown in Figure 3.4.

\[
M^{-1} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & -1 & 2 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\]
Figure 3.4 Multiply linked list

HN: Head Node  RN: Row Number  CN: Column Number
VAL: Value of the element  RLINK/DLINK: Pointers
Iteration 3:

Step 1: \( z_j - c_j = (0 \ 0) \)

There are no more promising variables and therefore optimal solution is obtained. Thus the solution is given by

\[
\begin{bmatrix}
Z \\
X_B
\end{bmatrix} = M^{-1} 
\begin{bmatrix}
0 \\
0
\end{bmatrix}
\]

\[
\begin{align*}
Z &= 14 \\
s_1 &= 3 \\
x_1 &= 2 \\
x_2 &= 4
\end{align*}
\]

3.5 COMPARISON OF REVISED SIMPLEX AND REVISED SIMPLEX WITH DATA STRUCTURE

Algorithms, in general, may have different computational efficiencies. Comparisons based on the time taken for a few typical problems are furnished in Table 3.1. One of the drawbacks of the ordinary simplex procedure is that it occupies lot of memory to store zero entries in the constraint coefficient matrix and \( M^{-1} \) matrix. Again each and everytime whenever a multiply/divide operation is done, all entries are accessed and it has to be checked whether it is zero or not.
<table>
<thead>
<tr>
<th>Sl.No</th>
<th>No. of Variables</th>
<th>No. of Constraints</th>
<th>No. of Iterations</th>
<th>Time in Seconds Revised Simplex</th>
<th>Time in Seconds Revised Simplex with DS</th>
<th>Percentage Sparsity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>31</td>
<td>17</td>
<td>16</td>
<td>2.21</td>
<td>1.50</td>
<td>87</td>
</tr>
<tr>
<td>2</td>
<td>60</td>
<td>16</td>
<td>20</td>
<td>2.34</td>
<td>1.74</td>
<td>87</td>
</tr>
<tr>
<td>3</td>
<td>90</td>
<td>21</td>
<td>30</td>
<td>3.77</td>
<td>2.20</td>
<td>90</td>
</tr>
<tr>
<td>4</td>
<td>121</td>
<td>26</td>
<td>41</td>
<td>3.05</td>
<td>2.44</td>
<td>92</td>
</tr>
<tr>
<td>5</td>
<td>150</td>
<td>31</td>
<td>50</td>
<td>9.30</td>
<td>3.93</td>
<td>93</td>
</tr>
<tr>
<td>6</td>
<td>300</td>
<td>56</td>
<td>100</td>
<td>52.95</td>
<td>12.45</td>
<td>93</td>
</tr>
<tr>
<td>7</td>
<td>330</td>
<td>61</td>
<td>110</td>
<td>69.35</td>
<td>15.15</td>
<td>94</td>
</tr>
<tr>
<td>8</td>
<td>333</td>
<td>62</td>
<td>118</td>
<td>76.27</td>
<td>16.53</td>
<td>94</td>
</tr>
<tr>
<td>9</td>
<td>372</td>
<td>43</td>
<td>83</td>
<td>60.75</td>
<td>30.83</td>
<td>92</td>
</tr>
<tr>
<td>10</td>
<td>927</td>
<td>74</td>
<td>261</td>
<td>535.05</td>
<td>248.37</td>
<td>95</td>
</tr>
<tr>
<td>11</td>
<td>261</td>
<td>69</td>
<td>143</td>
<td>156.21</td>
<td>145.31</td>
<td>25</td>
</tr>
</tbody>
</table>
From this comparison the following inferences are made:

i. If the sparsity of the constraint matrix is high, then the computational savings in the revised simplex using data structure concept are proportional to the size of the problems due to the following reasons:

a. Memory requirement is considerably small for storing the constraint coefficient matrix and inverse matrix.

b. Unnecessary checking of zero/nonzero entries for every multiply/divide operation is avoided.

ii. If the sparsity of the constraint matrix is low, then the savings for all the different type of problems are marginal and that too due to the nature of the $M^{-1}$ matrix. I.e. to start with the $M^{-1}$ matrix will have only $(m + 1)$ nonzero entries instead of $(m + 1) \times (m + 1)$ entries. Then each time when a promising variable is entered the $M^{-1}$ matrix is filled with nonzero entries.

Graphs connecting the number of resource constraints and the time required to solve the linear programming problems listed in Table 3.1 by the revised simplex and the revised simplex with data structure concepts are shown in Figure 3.5. It may be observed from the graphs drawn using the computer results that the slope of the curve for the simplex algorithm is steeper than that of the simplex with data structure for any range of constraints. This shows that for a given problem, the computing time required per constraint of the simplex algorithm with data structure is less than that of the simplex algorithm.
Comparison of Revised Simplex, with and without Data Structure

Figure 3.5

No. of Constraints

R.S

R.S.D.S

Time in Seconds
3.6 CONCLUSION

In this chapter different type of data structure concepts are involved for representing matrices. Revised simplex algorithm is compared with the Revised simplex algorithm using data structure concepts. It has been observed that the Revised simplex using data structure concept is always better compared with the original Revised simplex method for any type of problems. In the next chapter Multiplex and Dual Multiplex algorithms using data structure concepts are discussed.