# A COMPARATIVE STUDY OF INITIAL BASIC FEASIBLE SOLUTIONS TO TRANSPORTATION PROBLEMS

by >2/14

HARRY MAC SCOTT

B. S., Kansas State University, 1968

---

A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Computer Science

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1971

Approved by:

Paul S. Fisher
Major Professor

# TABLE OF CONTENTS

LIST OF TABLES

# CHAPTER I

## INTRODUCTION

Along with the growing popularity and wide-spread use of computers, transportation problems of linear programming has developed into an often used money saving tool of many varied types of businesses. For example a company with ten million dollars worth of transportation costs per year could very likely save 5% or five hundred thousand dollars in a year's time. On the other hand, a small company may save over a hundred dollars a day if the company could use the transportation method for each new day's delivery problems. The problem lies in the tremendous amount of time and expense required by the computer to solve the problem each time. Therefore, by the proper selection of one method of finding the initial solution the amount of time required by the computer could be as little as 1/3 to 1/7 the time required if another method was used.

A certain class of linear programming problems, known as transportation problems, deal with the distribution of a commodity from various sources of supply to various points of demand. The transportation problem has been used in varied types of practical applications such as regular time and overtime production analysis, product allocation, machine assignment, and product distribution, in an effort to minimize costs of distribution and/or allocation and optimize distribution efficiency.

Regular time and overtime production deals with the problem of more workers on the job getting paid regular wages or correspondingly fewer workers working overtime to produce the same amount of the product. The manufacturer estimates the demand for his product for a number of future time periods, and from these estimates various costs including the cost of producing the product on regular time, the cost of producing the product on overtime, and the cost of storing the product for each time period can be derived. The transportation method will provide a schedule of production over each of the time periods in such a way as to minimize the above costs plus any other costs the manufacturer would like to consider.

Product allocation is a type of linear programming problem easily solved by the transportation method. This type is illustrated by the following example. A large company or corporation has a given number of plants producing the same product, for example an automobile tire company with 4 or 5 plants throughout the United States. By utilizing the cost of producing the product at each plant and the cost of shipping the product from the plants to each warehouse, the transportation method can give the optimal amount to be produced at each plant and sent to the various warehouses for a required number of tires.

The transportation method will also handle machine assignment problems. An example of this type of problem is in classroom assignment at a university. The university building is a shop consisting of a number of classrooms (machines) which can handle a certain type and number of jobs, i.e. a class of students. At the beginning of a semester, the transportation model can allocate classes of students to various classrooms in such a way that a minimal amount of moving from classroom to classroom can be accomplished without lengthening the total time required for each classroom in a given day.

The original and most common use of the transportation method of linear programming problems is the product distribution problem. For instance a company has m warehouse containing $a_1, a_2, \ldots, a_m$ quantities of the product respectively. The sum of the products available at the warehouses will be equal to A. The company has n retail outlets with the number of products each can sell equal to $b_1, b_2, \ldots, b_n$ respectively. The sum of the products in demand at the retail outlets is equal to B. Assume that A is equal to B since the transportation method is only set up to handle problems with the supply equal to the demand. (Later a scheme to make A equal to B will be discussed). The cost of transporting an item from warehouse $a_i$ to retail outlet $b_j$ is equal to $c_{ij}$. The number of products to be sent from each warehouse $a_i$ to each retail outlet $b_j$ is equal to $x_{ij}$ where $x_{ij}$ must be greater than or equal to zero, because a retail outlet may not supply a warehouse as would be the case if $x_{ij}$ is negative. Since the entire supply of each warehouse, $a_i$, will be shipped, $x_{i1}+x_{i2}+\ldots x_{in}= a_i$ for all i=1,2,...,m warehouses. Now since the constraint of A equal to B is put on the system, each retail outlet will receive the exact amount it required. Thus, $x_{1j}+x_{2j}+\ldots+x_{mj}=b_j$ for each retail outlet, i.e. for j=1,2,...n. Therefore the total cost, C, of shipping A products to n retail outlets from m warehouses is equal to the sum of the cost of sending one product times the number of products sent from each warehouse to each retail outlet.

Represented mathematically:

$$C = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}x_{ij}$$

subject to: $x_{ij}=a_i$, for $i=1,2,\ldots,m$;

$x_{ij}=b_j$, for $j=1,2,\ldots,n$;

and $x_{ij}\geq 0$, for all i and all j.

(CHUNG, 241)

The object of the transportation model is to allocate the number of products to be sent from each warehouse to each retail outlet at the minimum cost, C, while satisfying the specified constraints.

The general method of solution for transportation problems is to first establish the problem matrix with each origin as a row and each destination as a column. The elements of the array are the costs of sending one item from the warehouse (origin), corresponding to the row, to the retail outlet (destination), corresponding to the column. The constraints on each row will be the total amount of the product available at the warehouse. The constraints on the columns will be the amount required at each destination. Second an initial solution must be obtained that satisfies all constraints on the matrix model. A matrix with m origins and n destinations will have m+n-1 $x_{ij}$'s, that is m+n-1 quantities of goods shipped from origin i to destination j. Now a test, (the particular method of testing is explained later when discussing the modified distribution type of transportation method), is performed on the matrix to see if there exists a solution that will provide a lower total cost to the company. If one exists, a new solution is found and this new solution is in turn tested for a possible better solution. Each solution which is tested, and the new solution formed is called an iteration. On a large problem thousands of iterations may be required before an optimal solution, an answer that can not be improved upon, is found.

In most cases the supply of the origins is unequal to the demand of the destinations. Since we make the assumption that the supply A, equals the demand B, certain additions to the original matrix are required. If the supply is greater than the demand, a dummy destination is inserted into the matrix with a demand equal to the difference between the original supply and demand. A cost of zero is used in all entries of that column. If supply is less than the demand, a dummy row is inserted with zero costs and a supply equal to the difference between the original supply and demand. This method will balance the supply and demand and will have no affect on the other origins and destinations. The transportation method will automatically allocate products to the dummy row or column and the final solution will give those warehouses or retail outlets which will be short on their orders. (METZGER, 31)

The modified-distribution method, abbreviated MODI method, is one of the more commonly used methods to solve transportation problems. Several different methods for determining initial solutions, discussed later, will supply the MODI method with an initial solution of m+n-1 $x_{ij}$'s that corresponds to quantities of goods shipped from origin i to destination j. These $x_{ij}$'s along with their corresponding costs per unit, $c_{ij}$'s, will be considered as assigned cells or elements of the matrix. The MODI method tries to find a unassigned cell that will improve the solution by replacing an assigned cell, thus giving an improved solution which may or may not be optimal. If not optimal, the process is repeated against the new solution.

The MODI method assigns an unknown value $R_i$ to each assigned row in the transportation matrix, and number $K_j$ to each assigned column in the matrix. A set of formulas are established by $R_i+K_j=C_{ij}$ for each assigned cell in the solution. Since there are m rows and n columns, there are m R's and n K's or m+n unknowns. As stated previously there are m+n-1 assigned cells in the solution, therefore there are m+n unknowns and only m+n-1 equations. In order to solve for the R's and K's, one value for either an R or K must be assumed so the system may be solved. The value given to the R or K after solving the equations has no significance since only the relative values of R and K are important, but most generally a value of zero is given to $R_1$. At times a solution may not have the right amount of $x_{ij}$'s, in other words there will be less than m+n-1 assigned cells in the solution. In order to find a value for all R's and K's additional cells must be assigned at random to bring the total number of assigned cells up to m+n-1. Each new cell is given a $x_{ij}$ equal to zero and has no affect on the final solution.

After all R's and K's are computed an evaluation of each unassigned cell is carried out by the formula $R_i+K_j-C_{ij}$ for each unassigned cell. If for any unassigned cell the value of the formula is negative, an improvement in the solution is possible. If all evaluations of unassigned cells are greater than or equal to zero, then the last solution is optimal. If there are negative evaluations of the above formula, the MODI method selects the unassigned cell with the largest negative evaluation and inserts it into the solution replacing a selected assigned cell. Then the MODI method establishes a closed path starting at the unassigned cell to be inserted and making right turns at assigned cells only until it returns to the unassigned cell. In other words, if searching down a column for an assigned cell, make a right angle turn and search along a row for the next assigned cell. There exists one such path for each unassigned cell although it may be very difficult to find. (HADLEY, 284) Starting with a plus at the unassigned cell, the MODI method alternates signs at each turn of the closed path until reaching the unassigned cell again. The smallest amount at a negative assigned cell will then be added or subtracted to each assigned cell in the closed path depending on the sign as determined above. Thus the MODI method inserts a new cell into the solution and new R and K values must then be calculated. The whole process is repeated until an optimal solution is found. (METZGER, 20)

The first step, establishing the problem matrix, is simple in nature and can not be altered to save much time in solving the complete problem. The actual solving of the transportation problem has several different methods but each seems to have specified advantages and therefore should be left up to the individual as to the best method for his needs. Finding the initial solution has many different methods, and has a great influence on the number of iterations required to find an optimal solution; and therefore, a great influence on the computer time involved in finding the best or least cost answer. Few comparisons have been made on the methods for finding initial solutions yet much time could have been saved if the proper method for finding the initial solution had been used to start the transportation problem. Five of the more common methods for finding a solution are the northwest corner method, the column minimization method, the row minimization method, the matrix minimization method, and Vogel's approximation method.

The northwest corner method, is by far the simplest method for obtaining the initial solution. Appendix A is the northwest corner method as coded in FORTRAN IV. This method gets its name because it starts with the cell in the upper left hand corner, $x_{11}$, and allocates the maximum feasible amount available at that point. In other words the minimum between the amount available at origin one and the amount required at destination one. Move one cell to the right if there is any remaining amount at the origin, otherwise move one cell down. If it is impossible to do either, then the solution has been formed. After each move allocate the maximum feasible amount and move again. (HILLIER, 180)

The column minimization method as shown in Appendix B is also implemented in FORTRAN IV. This method gets its name because each column is considered seperately until it is satisfied before moving to the next column. Starting in column one, select the lowest cost in the column. If more than one minimum cost is found, choose any one of the possibilities. Choose the maximum feasible amount between this column and the corresponding row and place it in $x_{r1}$. If $x_{r1}$ is equal to the demand of column one, move on to the next column. Otherwise remove the row from further consideration for the entire method and find the next lowest cost in that column, assign $x_{r'1}$ equal to the minimum of of the supply of row r' and the demand remaining after the last assignment,

$b_1-x_{r1}$. Continue in this manner until the requirements of the first destination is satisfied, then move on to the next column. Repeat the procedure for each column. If the supply of the row is equal to the remaining demand of the column, remove only the row from consideration, find the next lowest cost in the column, assign a zero to the corresponding $x_{r'1}$, then move on to the next column. (HADLEY, 306)

The row minimization method as coded in Appendix C in FORTRAN IV gets its name by considering each row seperately until all are satisfied. This technique is similar to the column minimization method. Starting in row one, select the lowest cost in the row. If more than one minimum cost is found, choose any one of the possibilities. Choose the maximum feasible amount between this row and the corresponding column and place it in $x_{1c}$. If $x_{1c}$ is equal to the supply of row one then move on to the next row. Otherwise remove the column from further consideration for the entire method and find the next lowest cost in the row. Assign $x_{1c'}$ equal to the minimum of the supply remaining of row one after the last assignment, $a_1-x_{1c}$, and the demand of the new column c'. Continue in this manner until the requirements of the first supply row are met, then move on to the next row. Repeat the procedure for each row. If the supply of the row is equal to the demand of the column selected, remove only the column from consideration, find the next lowest cost in the row, assign a zero to the corresponding $x_{1c'}$, then move on to the next row. (HADLEY, 307)

The matrix minimization method selects the lowest cost from the entire matrix. Appendix D has coded version of this technique in FORTRAN IV. Find the minimum cost in the entire matrix, say $c_{ij}$. Set $x_{ij}$ equal to the minimum of $a_i$ and $b_j$, in other words set $x_{ij}$ equal to the maximum feasible amount between the supply of row i and the demand of column j. Remove row i or column j depending on which requirement was satisfied. If the demand was satisfied reduce the supply of row i by the amount $b_j$, $a_i=a_i-b_j$. If the supply was satisfied reduce the demand by the amount $a_i$, $b_j=b_j-a_i$. Repeat the process until all the rows or all the columns are removed. If there is not a unique minimum cost, select any one of them. If the supply is equal to the demand at the point $c_{ij}$ remove only a row or column but not both.

Vogel's approximation method will usually yield a near optimal solution in a majority of distribution problems. The method, coded in FORTRAN IV in Appendix E, was developed by W. R. Vogel in late 1955. For each row compute $k_i$ equal to the lowest cost in the row subtracted from the next lowest cost in the row. For each column compute $l_j$ equal to the lowest cost in the column subtracted from the next lowest cost in the column. Select the largest value among all of the k's and l's and the corresponding row, if a k was selected, or column, if an l was selected, is now considered. From the row or column selected choose the lowest cost in that row or column. Now set $x_{ij}$ equal to the minimum of the supply of row i or the demand of column j. Remove the row or column from consideration depending on which requirement was satisfied and repeat the entire method for the remaining rows and columns. If a row and column are satisfied at the same time remove either a row or column but not both. (CHUNG, 246)

Each of the methods for finding the initial solution are simple to use although some take much longer than others. The speed and simplicity of the northwest corner method is usually out weighed by the near optimal solutions delivered by the other methods. Each does give an initial solution that is acceptable to any method of transportation problem solvers such as the "stepping" method (METZGER, 11), the primal-dual method (SIMONNARD, 301), or the modified distribution method commonly called the MODI method and used most often because of its simplicity of operation.

The MODI method was used in comparing the different methods for finding an initial solution to the transportation problems. Appendix F contains a FORTRAN II coded version of the MODI method used on an IBM 360/30. The program is capable of solving a problem of 200 origins and 900 destinations in 53,000 bytes of core memory on the 360/30.

Although transportation problems are usually simpler than linear programming problems, there are several difficulties still present in transportation models. A high number of origins and/or destinations require a considerable amount of data gathering. For example a company with 200 origins and 900 destinations require 180,000 different costs of shipping from each origin to each destination, plus 1100 quantities to be shipped or received corresponding to the 200 origins and 900 destinations respectively. These 181,100 values must be punched on cards or placed on tape for the computer to read and the computer must be programmed to handle such a large amount of data in matrix form. With a problem this large, the computer may require hours to obtain an optimal solution and if any errors in the data exist, thousands of dollars could be lost with the delay in transporting the products due to the time involved in rerunning the problem on the computer.

Although many authors describe the various methods for obtaining initial solutions, few offer any comparison of the relative advantages of any method over the other methods. Vogel's approximation method is often recommended for large distribution problems but no method is recommended for the smaller problems common to moderately sized businesses. A comparison among the five methods for finding initial solutions mentioned earlier has been made on four small transportation problems common to relatively small businesses.

All solutions were found on the same computer and this was the only job running so that the difference in time measurements are not due to other concurrent jobs. The same program, the MODI method, was used for all tests so that the number of iterations required and times required to find the optimal solution could be accurately compared.

For each computer run, the time required to find the initial solution, the time required to reach the optimal solution from the initial solution, and the number of iterations required to reach the optimal solution were printed out by the computer along with the final optimal solution. All times printed out were the actual time the computer was working on the problem minus any time the computer was waiting for input or output devices.

All data for two of the transportation models, the 5 origins by 119 destinations and the 12 origins by 50 destinations were supplied by companies wanting a solution to their problems. The data for the other two transportation models were derived to give a more general test of the different methods of finding initial solutions. All times will vary from computer to computer and from program to program but the relative sizes of the times should remain constant.

# CHAPTER II

## RESULTS

The northwest-corner method, the column minimization method, the row minimization method, the matrix minimization method, and Vogel's approximation method are the most commonly used methods of finding an initial solution to a transportation problem of linear programming. Four different transportation models were used in order to provide a basis for comparing the different methods of finding initial solutions.

TABLE I

Transportation Model

24 x 24

| Method | Time (Seconds) Init. Solution | Time (Seconds) Opt. Solution | Iterations Required | Total Time (Sec.) |
|---|---|---|---|---|
| Northwest Corner | .13 | 116.58 | 71 | 116.71 |
| Column Minimization | .50 | 54.96 | 39 | 55.46 |
| Row Minimization | .58 | 13.23 | 9 | 13.81 |
| Matrix Minimization | 1.68 | 58.19 | 39 | 59.87 |
| Vogel Approximation | 23.80 | 55.49 | 40 | 79.29 |

The first transportation model tested had 24 origins and 24 destinations. Since the total supply at the origins was less than the total demand a dummy origin was set up making the transportation model 25 origins and 24 destinations. The time required to establish an initial solution ranged from 0.13 seconds for the northwest corner method to 23.80 seconds for Vogel's approximation method. The time required to arrive at the optimal solution after the initial solution was established took the row minimization method 13.23 seconds and 9 iterations while it took the northwest corner method 116.58 seconds and 71 iterations to arrive at the same solution. For the 25 x 24 transportation model the row minimization method took a total of 13.81 seconds to solve the problem which is roughly 1/4 as long as the column minimization method and approximately 1/8 as long as the northwest corner method. Table I contains a summary of the comparative times for the 24 x 24 transportation model.

TABLE 2

Transportation Model

5 x 119

| Method | Time (Seconds) Init. Solution | Time (Seconds) Opt. Solution | Iterations Required | Total Time (Sec.) |
|---|---|---|---|---|
| Northwest Corner | .17 | 132.04 | 111 | 132.21 |
| Column Minimization | .47 | 38.70 | 35 | 39.17 |
| Row Minimization | 4.82 | 32.94 | 28 | 37.76 |
| Matrix Minimization | 6.95 | 64.66 | 55 | 71.61 |
| Vogel Approximation | 87.02 | 21.31 | 20 | 108.33 |

The second transportation model tested had 5 origins and 119 destinations. The time required to establish an initial solution ranged from 0.17 seconds for the northwest corner method to 87.02 seconds for Vogel's approximation method.

The time required to arrive at the optimal solution after the initial solution was established took Vogel's approximation method 21.31 seconds and 20 iterations while it took the northwest corner method 132.04 seconds and 111 iterations to arrive at the same solution. The row minimization method and the column minimization method took 37.76 seconds and 39.17 seconds respectively to solve the whole problem which are roughly 1/3 the time required for Vogel's approximation method or for the northwest corner method. A summary of the comparative times for all five methods is furnished in Table 2.

TABLE 3

Transportation Model

119 x 5

| Method | Time (Seconds) Init. Solution | Time (Seconds) Opt. Solution | Iterations Required | Total Time (Sec.) |
|---|---|---|---|---|
| Northwest Corner | .20 | 149.72 | 111 | 149.92 |
| Column Minimization | 4.17 | 39.98 | 28 | 44.15 |
| Row Minimization | .52 | 54.99 | 35 | 55.51 |
| Matrix Minimization | 7.65 | 26.07 | 17 | 33.73 |
| Vogel Approximation | 76.37 | 40.75 | 28 | 117.12 |

The third transportation model was a reverse of the matrix shape of model 2 with 119 origins and 5 destinations but with different data. The time required for the initial solutions was 0.20 seconds for northwest corner method and 76.37 for Vogel's method. The matrix minimization method had the lowest total time of 33.73 seconds and the northwest corner method took 149.92 seconds or almost 5 times as long to find the optimal solution. Table 3 gives a summary of the transportation model with 119 origins and 5 destinations.

TABLE 4

Transportation Model

12 x 50

| Method | Time (Seconds) Init. Solution | Time (Seconds) Opt. Solution | Iterations Required | Total Time (Sec.) |
|---|---|---|---|---|
| Northwest Corner | .13 | 83.36 | 78 | 83.49 |
| Column Minimization | .40 | 14.77 | 13 | 15.17 |
| Row Minimization | 1.06 | 90.60 | 86 | 91.66 |
| Matrix Minimization | 3.42 | 20.21 | 19 | 23.63 |
| Vogel Approximation | 30.45 | 64.10 | 56 | 94.55 |

The fourth transportation model had 12 origins and 50 destinations. Since the total supply at the origins was greater than the total demand of the destinations, a dummy demand was inserted making the model 12 x 51. Again the northwest corner method obtained the initial solution the fastest and Vogel's method was by far the slowest. The column minimization method found the optimal solution in 15.17 seconds which is 1/6 that of three of the other methods. Note in Table 4 the excessively long time required for row minimization compared to column or matrix minimization. Each different transportation model has properties that make one initiation method better than others and because of this, no particular method for finding initial solutions can be recommended for all models even if they are the same size of other models.

Although only four out of thousands or even millions of different transportation models were discussed, the results showed a variety in which methods of establishing initial solutions were best for each case. The results did show that even though the northwest corner method had fast initiation times, the great amount of time required to obtain the optimal solution makes the northwest corner method a poor choice for transportation problems. Vogel's approximation method, recommended by most authors, seems to require few iterations to obtain the optimal solution but requires more time to find the initial solution than some of the other methods required to find the optimal solution. Vogel's approximation method should probably be considered for large complex problems which were not used in this comparative study.

The remaining three initiation methods are competitive although column minimization probably should be used when there are a lot more origins than destinations and row minimization when there are a lot more destinations than origins. Matrix minimization seems to be slightly slower than either column or row minimization methods.

# CHAPTER 3

## SUMMARY

With the advent of modern high speed computers, many of the more successful businesses are the ones that have learned to utilize the computer to the fullest but most efficient means. The rising cost of transportation has caused the efficient distribution of products to be one of the major areas that a business looks at to cut costs and increase profits. Transportation problems of linear programming problems deal with the optimal method of distributing products from a number of warehouses to a number of retail outlets. Initial solutions, from which the optimal solution is derived, have a very great affect on the time required to find the optimal solution.

The northwest corner method, column minimization method, row minimization method, matrix minimization method, and Vogel's approximation method, five of the more common methods for finding initial solutions to transportation problems, were compared to determine the best method to use on small transportation models common to small to medium sized businesses. Only four different transportation models were used to test the different methods for finding initial solutions but the results were different enough to demonstrate that some methods will save a large amount of total time for finding an optimal solution over other methods.

The total time for finding the initial solutions for all four transportation models was 0.63 seconds for the northwest corner method and 217.64 seconds Vogel's approximation method which is over 11 times the time required for its nearest competitor. The time required to obtain the optimal solution after the initial solution has been established took 481.70 seconds for the northwest corner method while the other methods ranged from 148.41 to 191.76 seconds to obtain the same optimal solutions for all four transportation models. The column minimization method had the lowest total time of the four models with 153.95 seconds. Vogel's approximation method required 399.29 seconds while the northwest corner method required 482.33 seconds which is over three times the total time required by the column minimization method.

TABLE 5

Summary

| Method | Time (Seconds) Init. Solution | Time (Seconds) Opt. Solution | Iterations Required | Total Time (Sec.) |
|---|---|---|---|---|
| Northwest Corner | .63 | 481.70 | 372 | 482.33 |
| Column Minimization | 5.56 | 148.41 | 115 | 153.95 |
| Row Minimization | 6.98 | 191.76 | 158 | 198.74 |
| Matrix Minimization | 19.71 | 169.13 | 130 | 188.84 |
| Vogel Approximation | 217.84 | 181.65 | 144 | 399.29 |

Results indicate that the northwest corner method, although fast in obtaining initial solutions, probably should not be used in finding an initial solution to solve a transportation problem. Vogel's approximation method was comparable to the rest of the methods when considereing the time required to find the optimal solution after the initial had been established, but Vogel's method required too much time to find the initial solution to make it feasible for small problems. All five methods are compared in Table 5 with column minimization having the lowest times. Column minimization should probably be used if the model is to be run only once. If the model is small and will need to be run on the computer several times then comparisons should be made among column, row, and matrix minimization methods to determine the best method for the individual type of transportation model. If the problem is large, Vogel's approximation method should probably be considered in the comparison of methods to use.

## SELECTED BIBLIOGRAPHY

BARSOV, A.S. 1959. What Is Linear Programming?, Translated by M. B. P. Slater and D. A. Levine, D. C. Heath and Co. Boston 1965

CHUNG, An-min. 1963. Linear Programming, Charles E. Merrill Books. Columbus, Ohio 1966

di Roccaferrera, G. M. F. 1967. Introduction to Linear Programming Processes, South-Western Publishing Co., Cinncinnati, Ohio, 1967

HADLEY, G. 1962. Linear Programming, Addison-Wesley Publishing Co., Reading, Massachusetts, 1962

HILLIER, F. S. and Lieberman, G. J. 1967. Introduction to Operations Research, Holden-Day, San Francisco, California, 1967

METZGER, R. W., 1958. Elementary Mathematical Programming, John Wiley and Sons, New York, 1963

NAYLOR, T. H. and Byrne, E. T. 1963. Linear Programming, Wadsworth Publishing Co., Belmont, California, 1963

SIMONNARD, M. 1966. Linear Programming, Translated by W. S. Jewell, Prentice-Hall, Engelwood Cliffs, New Jersey, 1966

# APPENDIXES

# APPENDIX A

```
C       NORTHWEST-CORNER METHOD
        DO 50 I=1,NORIG
        DO 50 J=1,NDEST
     50 IBAS(I,J)=0
        NROW=1
        NCOL=1
     51 IF (IDEST(NCOL).LT.IORIG(NROW)) GO TO 52
        IBAS(NROW,NCOL)=IORIG(NROW)+K1
        IDEST(NCOL)=IDEST(NCOL)-IORIG(NROW)
        NROW=NROW+1
        IF (NROW.LE.NORIG) GO TO 51
     52 IBAS(NROW,NCOL)=IDEST(NCOL)+K1
        IORIG(NROW)=IORIG(NROW)-IDEST(NCOL)
        NCOL=NCOL+1
        IF (NCOL.LE.NDEST) GO TO 51
C       END OF NORTHWEST-CORNER METHOD
```

## APPENDIX B

```
C       COLUMN MINIMIZATION METHOD
        DO 50 I=1,NORIG
        DO 50 J=1,NDEST
     50 IBAS(I,J)=0
        DO 51 I=1,NORIG
     51 IU(I)=0
        DO 55 J=1,NDEST
     52 ISAV=10**7
        IROW=0
        DO 53 I=1,NORIG
        IF (IU(I).EQ.1) GO TO 53
        IF (ICOST(I,J).GE.ISAV) GO TO 53
        ISAV=ICOST(I,J)
        IROW=I
     53 CONTINUE
        IF (IDEST(J).LT.IORIG(IROW)) GO TO 54
        IU(IROW)=1
        IBAS(IROW,J)=IORIG(IROW)+K1
        IDEST(J)=IDEST(J)-IORIG(IROW)
        GO TO 52
     54 IBAS(IROW,J)=IDEST(J)+K1
        IORIG(IROW)=IORIG(IROW)-IDEST(J)
     55 CONTINUE
C       END OF COLUMN MINIMIZATION METHOD
```

# APPENDIX C

```
C       ROW MINIMIZATION METHOD
        DO 50 I=1,NORIG
        DO 50 J=1,NDEST
     50 IBAS(I,J)=0
        DO 51 J=1,NDEST
     51 IV(J)=0
        DO 55 I=1,NORIG
     52 ISAV=10**7
        ICOL=0
        DO 53 J=1,NDEST
        IF (IV(J).EQ.1) GO TO 53
        IF (ICOST(I,J).GE.ISAV) GO TO 53
        ISAV=ICOST(I,J)
        ICOL=J
     53 CONTINUE
        IF (IORIG(I).LT.IDEST(ICOL)) GO TO 54
        IV(ICOL)=1
        IBAS(I,ICOL)=IDEST(ICOL)+K1
        IORIG(I)=IORIG(I)-IDEST(ICOL)
        GO TO 52
     54 IBAS(I,ICOL)=IORIG+K1
        IDEST(ICOL)=IDEST(ICOL)-IORIG(I)
     55 CONTINUE
C       END OF ROW MINIMIZATION METHOD
```

# APPENDIX D

```
C       MATRIX MINIMIZATION METHOD
        DO 50 I=1,NORIG
        IU(I)=0
     50 IBAS(I,J)=0
        DO 51 J=1,NDEST
     51 IV(J)=0
        N=NORIG+NDEST-1
        DO 55 K=1,N
        ISAV=10**7
        DO 53 I=1,NORIG
        IF (IU(I).EQ.1) GO TO 53
        DO 52 J=1,NDEST
        IF (IV(J).EQ.1) GO TO 52
        IF (ICOST(I,J).GE.ISAV) GO TO 53
        ISAV=ICOST(I,J)
        NROW=1
        NCOL=J
     52 CONTINUE
     53 CONTINUE
        IF (IDEST(NCOL).LT.IORIG(NROW)) GO TO 54
        IU(NROW)=1
        IBAS(NROW,NCOL)=IORIG(NROW)+K1
        IDEST(NCOL)=IDEST(NCOL)-IORIG(NROW)
        GO TO 55
     54 IV(NCOL)=1
        IBAS(NROW,NCOL)=IDEST(NCOL)+K1
        IORIG(NROW)=IORIG(NROW)-IDEST(NCOL)
     55 CONTINUE
C       END OF MATRIX MINIMIZATION METHOD
```

# APPENDIX E

```
C       VOGEL'S APPROXIMATION METHOD
        DO 50 I=1,NORIG
        IU(I)=0
        DO 50 J=1,NDEST
     50 IBAS(I,J)=0
        DO 51 J=1,NDEST
     51 IV(J)=0
        N=NORIG+NDEST-1
        DO 60 K=1,N
        L=0
        DO 54 I=1,NORIG
        IF (IU(I).EQ.1) GO TO 54
        ISAV=10**7
        L=L+1
        IVOGEL(L,2)=I
        DO 52 J=1,NDEST
        IF (IV(J).EQ.1) GO TO 52
        IF (ICOST(I,J).GE.ISAV) GO TO 52
        IVOGEL(L,3)=J
        ISAV=ICOST(I,J)
     52 CONTINUE
        ISAV1=10**7
        DO 53 J=1,NDEST
        IF (IV(J).EQ.1) GO TO 53
        IF ((ICOST(I,J).GE.ISAV1).OR.(J.EQ.IVOGEL(L,3))) GO TO 53
        ISAV1=ICOST(I,J)
     53 CONTINUE
        IVOGEL(L,1)=ISAV1-ISAV
     54 CONTINUE
        DO 57 J=1,NDEST
        IF (IV(J).EQ.1) GO TO 57
        L=L+1
        ISAV=10**7
        IVOGEL(L,3)=J
        DO 55 I=1,NORIG
        IF (IU(I).EQ.1) GO TO 55
        IF (ICOST(I,J).GE.ISAV) GO TO 55
        IVOGEL(L,2)=I
        ISAV=ICOST(I,J)
     55 CONTINUE
        ISAV1=10**7
        DO 56 I=1,NORIG
        IF (IU(I).EQ.1) GO TO 56
        IF ((ICOST(I,J).GE.ISAV1).OR.(I.EQ.IVOGEL(L,2))) GO TO 56
        ISAV1=ICOST(I,J)
     56 CONTINUE
        IVOGEL(L,1)=ISAV1-ISAV
     57 CONTINUE
```

```
      ISAV=10**7
      ISAV=-ISAV
      DO 58 I=1,L
      IF (IVOGEL(I,1).LT.ISAV) GO TO 58
      LSAVE=I
   58 CONTINUE
      IF (IDEST(IVOGEL(LSAVE,3)).LT.IORIG(IVOGEL(LSAVE,2))) GO TO 59
      IBAS(IVOGEL(LSAVE,2),IVOGEL(LSAVE,3))=IORIG(IVOGEL(LSAVE,2))+K1
      IDEST(IVOGEL(LSAVE,3))=IDEST(IVOGEL(LSAVE,3))-IORIG(IVOGEL(LSAVE,
     12))
      IU(IVOGEL(LSAVE,2))=1
      GO TO 60
   59 IBAS(IVOGEL(LSAVE,2),IVOGEL(LSAVE,3))=IDEST(IVOGEL(LSAVE,3))+K1
      IORIG(IVOGEL(LSAVE,2))=IORIG(IVOGEL(LSAVE,2))-IDEST(IVOGEL(LSAVE,
     13))
      IV(IVOGEL(LSAVE,3))=1
   60 CONTINUE
C     END OF VOGEL'S APPROXIMATION METHOD
```

# APPENDIX F

```
C     MODIFIED DISTRIBUTION METHOD
      DIMENSION ICOST(201),IU(201),IV(900),IORIG(201),IDEST(900)
      DIMENSION IU1(201),IV1(900),INET(2210),INET1(201),INET2(900)
      DIMENSION X(18)
      DIMENSION IBAS(1100,3)
      EQUIVALENCE(IU(1),INET(1)),(IORIG(1),INET1(1)),(IDEST(1),INET2(1))
      NR=1
      NP=2
      NPR=3
      ND=7
      ICOUNT=1
  937 FORMAT(18A4)
      READ(NR,937) X
    1 READ (NR,2) NORIG,NDEST
    2 FORMAT (3I7)
    3 K1=10**7
      K2=10**8
      NDEST1=NDEST
      NORIG1=NORIG
      DO 4 I=1,NORIG
    4 READ (NR,1005) IORIG(I)
    5 FORMAT (8I10)
 1005 FORMAT (I8)
      DO 6 I=1,NDEST
    6 READ (NR,1005) IDEST(I)
    8 ISUMO=0
      ISUMD=0
      DO 9 I=1,NORIG
    9 ISUMO=ISUMO+IORIG(I)
      DO 10 I=1,NDEST
   10 ISUMD=ISUMD+IDEST(I)
      IF (ISUMO-ISUMD) 16,18,18
   16 NORIG=NORIG+1
      IORIG(NORIG)=ISUMD-ISUMO
   18 REWIND ND
      DO 7 I=1,NDEST
      READ (NR,5) (ICOST(J),J=1,NORIG1)
      ICOST(NORIG1+1)=999999
      WRITE (ND) (ICOST(J),J=1,NORIG)
    7 CONTINUE
   11 IF (ISUMO-ISUMD) 15,15,12
   12 NDEST=NDEST+1
      IDEST(NDEST)=ISUMO-ISUMD
   13 DO 14 I=1,NORIG
   14 ICOST(I)=999999
      WRITE (ND) (ICOST(I),I=1,NORIG)
   15 REWIND ND
   17 NROW=1
      NCOL=1
      READ (ND) (ICOST(I),I=1,NORIG)
```

```
      ISAV=K1
      DO 19 I=1,NORIG
   19 IU(I)=1
   21 IF (ICOST(NROW)-ISAV) 24,22,22
   22 NROW=NROW+1
   23 IF (NROW-NORIG) 21,21,26
   24 IF (IU(NROW)-1) 22,25,22
   25 ISAV=ICOST(NROW)
      NROW1=NROW
      GO TO 22
   26 IF (IORIG(NROW1)-IDEST(NCOL)) 27,29,37
   27 IBAS(ICOUNT,1)=IORIG(NROW1)+K1
      IBAS(ICOUNT,2)=NROW1
      IBAS(ICOUNT,3)=NCOL
      ICOUNT=ICOUNT+1
   28 IDEST(NCOL)=IDEST(NCOL)-IORIG(NROW1)
      IU(NROW1)=0
      NROW=1
      ISAV=K1
      GO TO 21
   29 IBAS(ICOUNT,1)=IORIG(NROW1)+K1
      IBAS(ICOUNT,2)=NROW1
      IBAS(ICOUNT,3)=NCOL
      ICOUNT=ICOUNT+1
      IU(NROW1)=0
      NROW=1
      ISAV=K1
      IF (NCOL-NDEST) 30,39,39
   30 IF (ICOST(NROW)-ISAV) 31,33,33
   31 IF (IU(NROW)-1) 33,32,33
   32 ISAV=ICOST(NROW)
      NROW1=NROW
   33 NROW=NROW+1
      IF (NROW-NORIG) 30,30,34
   34 IBAS(ICOUNT,1)=K1
      IBAS(ICOUNT,2)=NROW1
      IBAS(ICOUNT,3)=NCOL
      ICOUNT=ICOUNT+1
      NCOL=NCOL+1
   35 IF (NCOL-NDEST) 36,36,39
   36 READ (ND) (ICOST(I),I=1,NORIG)
      NROW=1
      ISAV=K1
      GO TO 21
   37 IBAS(ICOUNT,1)=IDEST(NCOL)+K1
      IBAS(ICOUNT,2)=NROW1
      IBAS(ICOUNT,3)=NCOL
      ICOUNT=ICOUNT+1
      IORIG(NROW1)=IORIG(NROW1)-IDEST(NCOL)
   38 NCOL=NCOL+1
      GO TO 35
   39 REWIND ND
      ICOUNT=ICOUNT-1
```

```
      ICP=1
      IBOT=1
      DO 1034 I=1,ICOUNT
      IF (IBAS(I,3)-ICP) 1035,1034,1035
 1035 ITOP=I-1
      IDIS=ITOP-IBOT
      IF (IDIS) 1036,1033,1036
 1036 CONTINUE
      DO 1037 J=1,IDIS
      ITOPE=ITOP-J
      DO 1038 K=IBOT,ITOPE
      KL=K+1
      IF (IBAS(K,2)-IBAS(KL,2)) 1038,1038,1039
 1039 IB11=IBAS(K,1)
      IB12=IBAS(K,2)
      IB13=IBAS(K,3)
      IBAS(K,1)=IBAS(KL,1)
      IBAS(K,2)=IBAS(KL,2)
      IBAS(K,3)=IBAS(KL,3)
      IBAS(KL,1)=IB11
      IBAS(KL,2)=IB12
      IBAS(KL,3)=IB13
 1038 CONTINUE
 1037 CONTINUE
 1033 ICP =IBAS(I,3)
      IBOT=I
 1034 CONTINUE
 1040 DO 40 I=1,NORIG
      IU(I)=0
   40 IU1(I)=0
      DO 41 I=1,NDEST
      IV(I)=0
   41 IV1(I)=0
      REWIND ND
   42 IV1(1)=1
      READ (ND) (ICOST(L),L=1,NORIG)
      NCO=1
      I=0
   46 I=I+1
      IF (I-ICOUNT) 1044,1044,1049
 1044 IF (IBAS(I,3)-NCO) 45,44,45
   44 J=IBAS(I,2)
      IF (IU1(J)-1) 1041,46,1041
 1041 IU(J)=ICOST(J)-IV(NCO)
      IU1(J)=1
      GO TO 46
   45 NCO= IBAS(I,3)
      READ (ND) (ICOST(L),L=1,NORIG)
      DO 47 L=I,ICOUNT
      IF (IBAS(L,3)-NCO) 1046,1047,1046
 1047 IF (IV1(NCO)-1) 1048,47,1048
 1048 K=IBAS(L,2)
      IF (IU1(K)-1) 47,48,47
   47 CONTINUE
```

```
      GO TO 1049
 1046 I=L
      GO TO 45
   48 IV(NCO)=ICOST(K)-IU(K)
      IV1(NCO)=1
      I=I-1
      GO TO 46
 1049 REWIND ND
      DO 1050 I=1,NDEST
      IF (IV1(I)-1) 42,1050,42
 1050 CONTINUE
      ISAV=0
      NCOL=1
   65 IF (NCOL-NDEST) 59,59,61
   59 READ (ND) (ICOST(K),K=1,NORIG)
      DO 63 J=1,NORIG
   60 IS1=IU(J)+IV(NCOL)-ICOST(J)
      IF (IS1-ISAV) 63,63,62
   62 ISAV=IS1
      NROW1=J
      NCOL1=NCOL
   63 CONTINUE
      NCOL=NCOL+1
      GO TO 65
   61 IF (ISAV) 66,123,66
   66 MNO=(NORIG+NDEST)*2
      DO 67 I=1,MNO
   67 INET(I)=0
      DO 68 I=1,NORIG
   68 INET1(I)=0
      DO 69 I=1,NDEST
   69 INET2(I)=0
      REWIND ND
      NCOL3=0
      I=1
   70 INET(I)=NROW1
      INET(I+1)=NCOL1
      NROW=NROW1
      NCOL=1
      I=I+2
   71 DO 74 J=1,ICOUNT
      IF (IBAS(J,3)-NCOL) 1073,1072,1073
 1073 NCOL=NCOL+1
 1072 IF (IBAS(J,2)-NROW) 74,72,74
   72 IF (NCOL-NCOL1) 73,74,73
   73 INET(I)=NROW
      INET(I+1)=NCOL
      I=I+2
      GO TO 77
   74 CONTINUE
      IF (NCOL-NDEST) 71,71,75
   75 WRITE (NPR,76) NROW1,NCOL1
```

```
   76 FORMAT (31H1UNABLE TO COMPLETE LOOP IN ROW,I6,6HCOLUMN,I6)
      GO TO 123
   77 INET2(NCOL)=1
      NROW3=0
   78 DO 79 J=1,ICOUNT
      IF (IBAS(J,3)-NCOL) 79,1079,79
 1079 IF (IBAS(J,2)-INET(I-2)) 1082,79,1082
 1082 IF (IBAS(J,2)-NROW3) 79,79,82
   79 CONTINUE
      NROW3=0
 1080 I=I-2
      IF (I) 80,80,81
   80 WRITE (NPR,76) NROW1,NCOL1
      GO TO 123
   81 NROW=INET(I)
      NCOL=INET(I+1)
      INET2(NCOL)=0
      NCOL3=NCOL
      GO TO 87
   82 NROW=IBAS(J,2)
      IF (INET1(NROW)) 1080,83,1080
   83 INET(I)=NROW
      INET(I+1)=NCOL
      I=I+2
   84 IF (NROW-NROW1) 86,85,86
   85 I=I-2
      GO TO 1080
   86 INET1(NROW)=1
   87 DO 89 J=1,ICOUNT
      IF (IBAS(J,2)-NROW) 89,1088,89
 1088 IF (IBAS(J,3)-NCOL3) 89,89,88
   88 IF (IBAS(J,3)-INET(I-1)) 90,89,90
   89 CONTINUE
      NCOL3=0
      GO TO 93
   90 NCOL=IBAS(J,3)
      IF (INET2(NCOL)) 93,91,93
   91 INET(I)=NROW
      INET(I+1)=NCOL
      I=I+2
      NCOL3=0
   92 IF (NCOL-NCOL1) 77,95,77
   93 I=I-2
      IF (I) 80,80,94
   94 NROW=INET(I)
      NCOL=INET(I+1)
      INET1(NROW)=0
      NROW3=NROW
      GO TO 78
   95 I=3
      ISAV=K2
```

```
  96 NROW=INET(I)
     NCOL=INET(I+1)
  97 DO 99 K=1,ICOUNT
     IF (IBAS(K,2)-NROW) 99,98,99
  98 IF (IBAS(K,3)-NCOL) 99,100,99
  99 CONTINUE
 100 IF (IBAS(K,1)-ISAV) 101,102,102
 101 ISAV=IBAS(K,1)
     NROW2=NROW
     NCOL2=NCOL
 102 IF (NCOL-NCOL1) 103,106,103
 103 I=I+4
     IF (I-MNO) 96,96,104
 104 WRITE (NPR,108) NROW1,NCOL1
 105 FORMAT (40H1BASIS LOOP HAS MORE ELEMENTS THAT BASIS,2I6)
     GO TO 123
 106 IF (ISAV-K2) 109,107,107
 107 WRITE (NPR,108) NROW1,NCOL1
 108 FORMAT (24H1NO ELEMENT LESS THAN K2,2I6)
     GO TO 123
 109 J=-1
     NROW=INET(1)
     NCOL=INET(2)
     ISAV=ISAV-K1
     I=3
 114 NROW=INET(I)
     NCOL=INET(I+1)
 115 DO 117 K=1,ICOUNT
     IF (IBAS(K,2)-NROW) 117,116,117
 116 IF (IBAS(K,3)-NCOL) 117,121,117
 117 CONTINUE
 121 IBAS(K,1)=IBAS(K,1)+J*ISAV
 122 J=-J
     I=I+2
     IF (NCOL-NCOL1) 114,1020,114
1020 IF (NCOL2-INET(2)) 1121,1130,1140
1130 IF (NROW2-INET(1)) 1121,1140,1140
1121 DO 1123 I=1,ICOUNT
     IF (IBAS(I,3)-NCOL2) 1123,1122,1123
1122 IF (IBAS(I,2)-NROW2) 1123,1124,1123
1123 CONTINUE
1124 IOLD=I
     DO 1127 I=IOLD,ICOUNT
     IF (IBAS(I,3)-INET(2)) 1127,1125,1128
1125 IF (IBAS(I,2)-INET(1)) 1127,1127,1128
1127 CONTINUE
1128 INEW=I-2
     IF (INEW-IOLD) 1136,1129,1129
1129 DO 1135 I=IOLD,INEW
     IBAS(I,1)=IBAS(I+1,1)
     IBAS(I,2)=IBAS(I+1,2)
1135 IBAS(I,3)=IBAS(I+1,3)
```

```
1136 IBAS(INEW+1,1)=ISAV+K1
     IBAS(INEW+1,2)=INET(1)
     IBAS(INEW+1,3)=INET(2)
     GO TO 1040
1140 DO 1142 I=1,ICOUNT
     IF (IBAS(I,3)-INET(2)) 1142,1141,1143
1141 IF (IBAS(I,2)-INET(1)) 1142,1143,1143
1142 CONTINUE
1143 INEW=I
     DO 1146 I=INEW,ICOUNT
     IF (IBAS(I,3)-NCOL2) 1146,1144,1146
1144 IF (IBAS(I,2)-NROW2) 1146,1147,1146
1146 CONTINUE
1147 IOLD=I-1
     IF (IOLD-INEW) 1149,1150,1150
1150 IX=IOLD
     IDIST=IOLD-INEW+1
     DO 1148 I=1,IDIST
     IX=IOLD-I+1
     IBAS(IX+1,1)=IBAS(IX,1)
     IBAS(IX+1,2)=IBAS(IX,2)
1148 IBAS(IX+1,3)=IBAS(IX,3)
1149 IBAS(INEW,1)=ISAV+K1
     IBAS(INEW,2)=INET(1)
     IBAS(INEW,3)=INET(2)
     GO TO 1040
 123 WRITE (NPR,124)
 124 FORMAT (23H1ORIGIN    SHADOW PRICE)
 125 FORMAT (1H ,I6,I14)
 126 WRITE (NPR,125) (I,IU(I),I=1,NORIG1)
 127 WRITE (NPR,128)
 128 FORMAT (23H1DESTIN    SHADOW PRICE)
 129 WRITE (NPR,125) (I,IV(I),I=1,NDEST1)
 130 WRITE (NPR,131)
 131 FORMAT (49H1ORIGIN   DESTIN  QUANTITY   UNIT COST TOTAL COST)
     REWIND ND
     J=0
     II19=0
     DO 136 K=1,ICOUNT
     IF (IBAS(K,3)-NDEST1) 1132,1132,136
1132 IF (IBAS(K,3)-J) 133,133,132
 132 I=IBAS(K,3)-J
     J=IBAS(K,3)
     DO 134 L=1,I
 134 READ (ND) (ICOST(M),M=1,NORIG)
 133 ITEM=IBAS(K,1)-K1
     MN=IBAS(K,2)
     IF (MN-NORIG1) 1133,1133,136
1133 ITEM1=ICOST(MN)
     ITEM2=ITEM*ITEM1
     WRITE (NPR,135) IBAS(K,2),IBAS(K,3),ITEM,ITEM1,ITEM2
 135 FORMAT (1H ,I6,I9,I10,I12,I12)
     II19=II19+ITEM2
 136 CONTINUE
```

```
      WRITE (NPR,137) II19
  137 FORMAT (12HOTOTAL COST=,I9)
      WRITE (NPR,138)
  138 FORMAT (22H1NORMAL END OF PROGRAM)
      STOP
C     END OF MODIFIED DISTRIBUTION METHOD
      END
```

# A COMPARATIVE STUDY OF INITIAL BASIC FEASIBLE SOLUTIONS TO TRANSPORTATION PROBLEMS

by

HARRY MAC SCOTT

B. S., Kansas State University, 1968

---

AN ABSTRACT OF A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Computer Science

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1971

An explanation of a certain class of linear programming problems known as transportation problems including several types of applications such as regular time and overtime production analysis, product allocation, machine assignment, and product distribution. A general description of how the transportation method works plus a complete description of the modified distribution method, commonly called the MODI method, are used to explain the transportation method.

Various methods of finding initial solutions include the northwest corner method, the column minimization method, the row minimization method, the matrix minimization method, and Vogel's approximation method. These methods are compared on four different transportation problems. A comparison is given on time to obtain initial solutions, time to obtain the optimal solutions after the initial solution is found, the number of iterations required, and the total time required to obtain an optimal solution.