Deterministic Operations Research

Some Examples

Ümit YÜCEER

November 29, 2006

Abstract

A summary of deterministic operations research models in linear programming, inventory theory, and dynamic programming.

1 Linear Programming

A mathematical model of the problem is developed basically by applying a scientific approach as described earlier. There are a number of activities to be performed and each unit of each activity consumes some amount of each type of a resource. Resources are available in limited quantities. A measure of performance (an effectiveness or ineffectiveness measure) is defined according to the objective(s) of the management on the activities of concern.

“Allocating the available amounts of resources to these activities in an effective manner” is transformed into a set of mathematical expressions that may result in a mathematical programming model of the type:

\[
\max \ f(x_1, x_2, \ldots, x_n)
\]

subject to
In the problem (1) through (5), \( x_j \) represents the level of activity to be employed (a decision variable) for \( j = 1, 2, \ldots, n \), and the function \( f(x_1, x_2, \ldots, x_n) \) is the measure of effectiveness based on the values of the decision variables \( x_j \) and is called the \textit{objective function}. The mathematical problem (1) through (5) then requires determining the value of each \( x_j \) in such a way that the objective function attains its maximum value without exceeding the available amounts of the resources, and satisfying the minimum requirements and conditions specified by the constraints (2) through (5).

Each constraint function \( g_i(x_1, x_2, \ldots, x_n) \) represents the usage of resource and the right hand side value \( b_i \) is the available amount of the resource \( i = 1, 2, \ldots, r \). Further, for \( i = r + 1, r + 2, \ldots, q \) \( g_i(x_1, x_2, \ldots, x_n) \) shows the consumption of a certain ingredient \( i \), and its corresponding right hand side value \( b_i \) represents the minimum requirement. There may also be some conditions imposed on the activities and they are represented by equalities (4). Any maximization problem can be converted to a minimization problem by simply multiplying the objective function by \((-1)\) and vice versa. In a similar fashion any constraint of the form (2) can be converted to the form (3) by simply multiplying by \((-1)\) and vice versa. An equality constraint can be expressed as two inequality constraints.

In particular, if the objective function and the constraint functions are linear, then such a mathematical programming problem is called a \textit{linear programming (LP)} problem. A typical LP model looks like as given below.

\[
\begin{align*}
g_i(x_1, x_2, \ldots, x_n) &\leq b_i \quad \text{for } i = 1, 2, \ldots, r \\
g_i(x_1, x_2, \ldots, x_n) &\geq b_i \quad \text{for } i = r + 1, r + 2, \ldots, q \\
g_i(x_1, x_2, \ldots, x_n) &= b_i \quad \text{for } i = q + 1, q + 2, \ldots, m \\
x_j &\geq 0 \quad \text{for all } j = 1, 2, \ldots, n
\end{align*}
\]
subject to

\[
\begin{align*}
    a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n & \leq b_1 \\
    a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n & \leq b_2 \\
    \vdots & \\
    a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n & \leq b_m \\
    a_{(m+1)1}x_1 + a_{(m+1)2}x_2 + \ldots + a_{(m+1)n}x_n & \geq b_{m+1} \\
    \vdots & \\
    a_{r1}x_1 + a_{r2}x_2 + \ldots + a_{rn}x_n & \geq b_r \\
    a_{(r+1)1}x_1 + a_{(r+1)2}x_2 + \ldots + a_{(r+1)n}x_n & = b_{r+1} \\
    \vdots & \\
    a_{s1}x_1 + a_{s2}x_2 + \ldots + a_{sn}x_n & = b_s \\
    x_j & \geq 0 \text{ for } j = 1, 2, \ldots, n
\end{align*}
\]

1.1 Basic Assumptions of Linear Programming Problems and Examples

In linear programming problems, the objective function and the constraint functions are linear. That implies that the relationships between the components of the real system can be expressed or approximated by linear expressions with a high degree of certainty. Further, the measure of performance is represented by a linear function of the decision variables. The linearity condition therefore requires satisfying the following basic assumptions.

1. Proportionality and Additivity: Each unit of an activity consumes a specified amount of a resource, and the increased level of activity increases the consumption of the resource in a direct proportion. Further, the total amount of resource consumed by all the activities is the
sum of resources consumed by each activity. The total effectiveness measure is the sum of effectiveness measures of each type of activity. For instance, the total profit is the sum of all profits generated by each activity.

2. Divisibility: The level of each activity is allowed to be any fractional real value.

3. Nonnegativity: The level of each type of activity cannot be negative. This is a natural restriction on the variables in most cases.

Some examples of linear programming problems will be presented to illustrate and clarify the concepts on mathematical and linear programming. More extensive examples can be seen in Operations Research or Management Science books, see for instance, Hillier and Lieberman [2].

1.1.1 Example 1. A production problem

A firm produces two products: A and B. Three types of resources are required for producing A and B. Each week, 30 kgs Raw Material, 21 hours of Labor time and 19 hours of Machine time are available for production. Each unit of A requires 5 kgs of the raw material, 3 hours of labor time and 2 hours of machine time. Each unit of B requires 3 kgs of the raw material, 4 hours of labor time, and 4 hours of machine time. Each unit of A yields a profit of $3 and each unit of B yields a profit of $5. The problem is then how much to produce each type of product so that the total profit is maximized without exceeding the available amounts of the resources.

Let $x_j$ denote the quantity of product $j \in \{A, B\}$ to be produced weekly, then the linear program is given as follows.

\[
\begin{align*}
\text{max} & \quad z = 3x_1 + 5x_2 & \quad \text{Total weekly profit} \\
\text{subject to} & \\
\end{align*}
\]
The optimal solution to this problem is given in the following sections.

### 1.1.2 Example 2. A minimization problem

Goldilocks needs to find at least 12 kg of gold and at least 12 kg of silver to pay the monthly rent. There are two mines in which Goldilocks can find gold and silver. Each day that Goldilocks spends in mine 1, she finds 2 kg of gold and 2 kg of silver. Each day that Goldilocks spends in mine 2, she finds 1 kg of gold and 3 kg of silver. Formulate an LP for Goldilocks to meet her requirements while spending as little time as possible in the mines.

Let $x_j$ represent the amount of time she spends in mine $j = 1, 2$.

\[
\begin{align*}
\text{min } z &= x_1 + x_2 & \text{Total time spent in the mines} \\
\text{subject to} & \\
2x_1 + x_2 & \geq 12 & \text{Gold} \\
2x_1 + 3x_2 & \geq 18 & \text{Silver} \\
x_1, x_2 & \geq 0
\end{align*}
\]

### 1.1.3 Example 3. A Diet Problem

Every individual has to consume daily a minimum amount of each type of vitamin for a healthy living. The school cafeteria serves three different types
of food: meat balls, fried chicken, salad. The vitamin contents and the cost (in terms of million TL) of each type food per serving are given in the table below. In addition, table gives the daily minimum intake of Vitamin A and B.

<table>
<thead>
<tr>
<th></th>
<th>meat ball</th>
<th>fried chicken</th>
<th>salad</th>
<th>minimum requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>17</td>
<td>12</td>
<td>25</td>
<td>110 mg</td>
</tr>
<tr>
<td>B</td>
<td>10</td>
<td>14</td>
<td>7</td>
<td>165 mg</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>6.75</td>
</tr>
<tr>
<td></td>
<td>3.25</td>
</tr>
</tbody>
</table>

How much of each type of food should be consumed at a minimum cost while satisfying the minimum daily intake requirements?

Let $x_j$ be the amount of food type $j = 1, 2, 3$ corresponding to meat balls, fried chicken, and salad respectively consumed daily (assuming a fraction of a serving allowed). Then the linear programming model is given as follows.

$$\min \ z = 5x_1 + 6.75x_2 + 3.25x_3 \quad \text{Total daily cost}$$

subject to

$$17x_1 + 12x_2 + 5x_3 \geq 110 \quad \text{Vitamin A requirement}$$

$$10x_1 + 14x_2 + 7x_3 \geq 137 \quad \text{Vitamin B requirement}$$

$$x_j \geq 0 \quad \text{for } j = 1, 2, 3$$

The optimal solution of this diet problem is to consume 1.23 servings of meat balls and 17.81 servings of salads daily at a minimum cost of 64.04 TL.
1.2 Solution Methodologies

1.2.1 Graphical Solution

If the linear programming problem contains only two variables, it is possible to draw the set of points satisfying all the constraints (feasible region) and finding the point(s) within the feasible region where the objective function attains its maximum (minimum).

Example (Example 1.1.1 continued) The feasible region determined by the inequalities (6) through (9) is given in the Figure 1. The isoprofit lines for the objective function is drawn first. For instance, for \( z = 15 \), the isoprofit line is the segment from \((0, 3)\) to \((5, 0)\) within the feasible region. Shifting the isoprofit line upwards or downwards shows whether the objective function value increases or decreases. Shifting the isoprofit line upwards in this case increases the \( z \) value and the maximum profit is obtained at the point \((2, 3.75)\) without leaving the feasible region. The maximum of the objective function occurs at the vertex (extreme point, corner point) \((2, 3.75)\) and the objective function value is \( z = 24.75 \).

\[
\max z = 3x_1 + 5x_2
\]

subject to

\[
\begin{align*}
5x_1 + 3x_2 & \leq 30 & (6) \\
x_1 + 2x_2 & \leq 10 & (7) \\
3x_1 + 4x_2 & \leq 21 & (8) \\
x_1, x_2 & \geq 0 & (9)
\end{align*}
\]

As can be observed from the figure, the optimal solution of a linear programming problem always occurs at a vertex (unique optima) or a number of adjacent vertices (case of multiple optima).

Exercise: Solve the Goldilocks problem (1.1.2) graphically.
Figure 1: Graphical solution of feasible region

1.2.2 Simplex method

This is a general method for solving linear programming problems algebraically. The simplex method was first developed by G.B. Dantzig just after WWII. Since then, it has proven to be an efficient way of solving linear programming problems. As observed from the graphical solution approach, simplex method searches the vertices of the feasible region until an optimal vertex is obtained. It starts from an easy to find vertex (usually the origin in case it is a vertex), and moves to an adjacent vertex if the objective function value is improved. The process is repeated until no further progress is possible, hence an optimal vertex is obtained. One can imagine that in an \( n \) dimensional hyperspace, the number of vertices is a very large number. That means in some cases the simplex method may take a very long time to find an optimum solution. Hypothetically, its worst–case behavior may take
centuries for a very simple problem. However, in solving real life problems, it has proven to be very efficient. There are several commercially available computer codes of simplex method such as LINDO, LINGO, CPLEX. Figure 2 displays the LINDO model and its solution of Example 1.1.1.

1.3 Mathematical Programming and Optimization

Various IE/OR problems are modeled as mathematical programming problems. An objective function is maximized or minimized subject to a set of constraints. In a linear programming problem the objective function and all the constraint are linear functions. If some of the variables in a linear programming problem takes only integer values, the problem becomes a mixed integer programming problem. If the objective function is a quadratic function but all constraints are linear functions, then such a problem is called a quadratic programming problem. A nonlinear programming problem is a mathematical programming problem in which at least one problem function is nonlinear.

An optimal or best solution is sought always under the given set of constraints. One should realize that these solutions are optimal only for the given set of constraints. The process of finding the values of variables which maximize or minimize a function subject to a set of constraints is called optimization. There are different types of optimization procedures or methods for different classes of problems. As the problems get more complex, the optimization procedures become more difficult. Usually an optimization process contains algorithms. An algorithm is a step by step iterative procedure of obtaining a solution to a mathematical programming problem. The simplex algorithm is an iterative procedure for finding an optimal solution to a linear programming problem. A solution method is exact if it is proven mathematically that it can find the optimum solution of a problem in a finite number of iterations. The number of iterations may be very large in
some cases. There are mathematical programming problems where there are no known methods or algorithms or the existing algorithms take very long. Based on some mathematical properties or structures of such problems, heuristic methods are proposed. A heuristic method does not guarantee an optimum solution but it finds a near optimum solution very quickly.

Consider the traveling salesman problem. A salesman wants to visit \( n \) cities one city at a time starting from a city (origin) and finally returning back to the origin. The distance between each pair of cities is given. The salesman wishes to find a sequence of \( n \) cities so that the total traveling distance is minimum without returning to any city visited before. If \( n \) is small, one can obtain the optimal sequence of cities traveled by listing all possible sequences (\( n! \)), and calculating the total distance traveled. However, if \( n \) is large, listing all \( n! \) possible sequences is prohibitive if not impossible. Thus heuristic methods have been found that find a near optimal sequence of cities very quickly.

The optimal solution for a mathematical programming problem is not necessarily the best possible solution of the modelled problem of interest. A model is an abstraction of the real system containing some simplifications. On the other hand, in the real life there are too many uncertainties that affect the system. Therefore a model should only serve as a guide for action in real life.

The Nobel Laureate in economics H. Simon states that optimization of a mathematical programming problem is basically a satisficing procedure in reality. More specifically, a satisfactory optimizing solution is obtained rather than the optimum solution. In real life satisficing is a practical and pragmatic way of obtaining a solution to the problem of concern.
1.4 Some Examples of Modeling and Formulation

Some interesting examples from the current literature will be presented for an illustration of application of mathematical programming models.

1.4.1 A Transportation Problem

There are $m$ plants manufacturing a particular product and there are $n$ cities where this product is consumed. The available supply ($s_i$) in each plant $i = 1, 2, \ldots, m$ will be shipped to the cities for consumption, and obviously the total amount of shipment from a plant cannot exceed its supply. On the other hand, the demand ($d_j$) in each city $j = 1, 2, \ldots, n$ must be satisfied, hence the amount of shipment to a city cannot be less than its demand. The cost of shipment per unit from each plant to each city ($c_{ij}$) for $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n$ is given. Then the problem is to determine how much to ship from each plant to each city so that the total cost of the shipment is minimum without exceeding the supply in each plant yet satisfying the demand in each city. A table can display all the pertinent information for a transportation problem and is called the transportation tableau 1. The transportation tableau gives the unit cost of shipment from each plant to each city and the supply in each plant and the demand in each city.

If the total supply ($\sum_{i=1}^{m} s_i$) is equal to the total demand ($\sum_{j=1}^{n} d_j$), then it is called a balanced transportation problem. If the total supply is not equal to the total demand, then introducing a dummy plant or a dummy city can convert the problem to a balanced transportation problem. The model presented here assumes a balanced transportation problem.

Let $x_{ij}$ denote the amount of shipment from plant $i = 1, 2, \ldots, m$ to city $j = 1, 2, \ldots, n$. Then the minimization of the total cost of all the shipments from all the plants to all the cities yields the model:
### Table 1: A transportation tableau

<table>
<thead>
<tr>
<th>$c_{ij}$</th>
<th>Cities (j)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Supply</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plants (i)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>9</td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>8</td>
<td>100</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>12</td>
<td>3</td>
<td>8</td>
<td>5</td>
<td>8</td>
<td>125</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>7</td>
<td>9</td>
<td>7</td>
<td>6</td>
<td>15</td>
<td>175</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>8</td>
<td>8</td>
<td>6</td>
<td>7</td>
<td>5</td>
<td>200</td>
</tr>
<tr>
<td>Demand</td>
<td></td>
<td>110</td>
<td>125</td>
<td>100</td>
<td>135</td>
<td>130</td>
<td>600</td>
</tr>
</tbody>
</table>

\[
\min \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}x_{ij} \tag{10}
\]

subject to

\[
\sum_{j=1}^{n} x_{ij} = s_i \quad \text{for } i = 1, 2, \ldots, m \tag{11}
\]

\[
\sum_{i=1}^{m} x_{ij} = d_j \quad \text{for } j = 1, 2, \ldots, n \tag{12}
\]

\[
x_{ij} \geq 0 \quad \text{for all } i = 1, 2, \ldots, m, j = 1, 2, \ldots, n \tag{13}
\]

The first set of constraints (11) represents that the total amount of shipment from each plant $i$ is $s_i$. The second set of constraints (12) represents that the total amount of shipment into a city $j$ is $d_j$.

#### 1.4.2 A Plant Location Problem

In the transportation problem of Subsubsection (1.4.1), the locations of plants are given. As an extension of the transportation problem, let’s now consider choosing the locations of plants from a set of given $m$ potential locations. When the plant in location $i = 1, 2, \ldots, m$ is opened, there is fixed cost ($f_i$)
of opening the plant. Then the problem becomes determining which plant locations to choose to supply all \( n \) cities and how much to ship from each plant (opened) to each city at a minimum cost without exceeding the available supply at the plants (opened) and yet satisfying the demand requirement in each city. In addition, no shipment from an unopened plant to any city is possible.

In addition to the variables \( x_{ij} \) of a transportation problem, now we need another set of variables to decide whether the plant in location \( i = 1, 2, \ldots, m \) should be opened. Let

\[
y_i = \begin{cases} 
1 & \text{if the plant in location } i \text{ is opened} \\
0 & \text{otherwise} 
\end{cases} \tag{14}
\]

The mathematical model is then given as follows.

\[
\min \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}x_{ij} + \sum_{i=1}^{m} f_iy_i \tag{15}
\]

subject to

\[
\sum_{i=1}^{m} x_{ij} = d_j \quad j = 1, 2, \ldots, n \tag{16}
\]

\[
\sum_{j=1}^{n} x_{ij} \leq y_is_i \quad i = 1, 2, \ldots, m \tag{17}
\]

\[
y_i \in \{0, 1\} \quad \text{for } i = 1, 2, \ldots, m \tag{18}
\]

\[
x_{ij} \geq 0 \quad \text{for all } i = 1, 2, \ldots, m, j = 1, 2, \ldots, n \tag{19}
\]

The objective function (15) is now the sum of the total cost of all the shipments from all the plants (opened) to all the cities and the cost of opening the plants. The first set of constraints (16) represent the demand requirement in each city. In view of the constraints (18), the second set of constraints (17) guarantee that the shipment from from an unopened plant
is zero, and the total shipment from an opened plant does not exceed its available supply.

2 An Inventory Problem

An item is consumed at a constant rate per unit time, and its annual demand is $D$ units. It is stocked to avoid shortages. There is a unit holding cost per unit per year. Whenever an order is placed, there is a fixed cost of ordering. Whenever the level of the item is depleted down to zero, an order is placed. Then the problem is simply to decide how many units to order whenever an order is placed so that the annual total cost of placing orders and carrying inventories is minimized.

2.1 Economic Order Quantity Model (EOQ)

The simplest deterministic model is the Economic Order Quantity (EOQ) model. This simplest model contains only the annual holding costs and the annual ordering costs. Balancing these two types of costs will determine an inventory policy based on the two factors:

1. When to order?
2. How much to order?

The following notation is standard in inventory theory and is borrowed from Winston [11].

$D = \text{The number of items (or units) demanded per year. Then, during any time interval of length } t \text{ years, an amount } (Dt) \text{ is demanded.}$

$K = \text{Setup cost for a production run of } (q) \text{ units or cost of placing an order regardless of the order quantity of } q \text{ units.}$

$h = \text{Holding cost if one unit is held in the stock for one year. In short, if } (I) \text{ units are held for } T \text{ years, a holding cost of } (ITh) \text{ is incurred.}$
Unit purchasing cost.

Subsequently, the number of orders per year for an order quantity \( q \) is given by \( D/q \) and the average inventory level in each period is \( q/2 \).

Annual ordering cost per year (AOC) = \( \frac{KD}{q} \) (20)

Annual holding cost per year (AHC) = \( \frac{hq}{2} \) (21)

Annual purchasing cost (APC) = \( pD \) (22)

The total cost function is the sum of all these costs together.

\[
TC = AOC + AHC + APC
\]

\[
TC(q) = \frac{KD}{q} + \frac{hq}{2} + pD
\]

The optimal value of \( q \) that minimizes \( TC(q) \) is obtained easily by taking the derivative of (23) and setting it to zero gives the optimal order quantity of Expression (24).

\[
TC'(q) = 0
\]

\[
= \frac{h}{2} - \frac{KD}{q^2} = 0
\]

\[
q^* = \sqrt{\frac{2KD}{h}}
\]

The equation (24) is often called the Economic Order Quantity (EOQ) formula. The Figure 3 displays the behavior of a simple inventory system and the usage of the item over time. This figure clearly illustrates the EOQ model with a reorder point.
It is quite obvious that at the end of first, second, third cycle, the time will be \( t_0, 2t_0D, 3t_0 \) and so on respectively. Hence the length of each cycle is constant and equal to \( t_0 = q/D \) in this model.

The close relationship between \( q \), and the annual total cost, the annual holding cost, and the annual ordering cost is depicted in the Figure 4. The convexity of the total cost function \( TC(q) \) is clearly shown in this figure, and consequently, it has a unique minimum for \( q > 0 \).

It is obvious in Figure 4 that the intersection point of Annual Holding and Annual Ordering Cost functions determines the optimal value of \( q \). At this order quantity, these two costs are equal and balanced.

Lead time is the length of the interval between time of placing an order and the time of receiving the shipment. If the lead time is constant, then the reorder point (ROP) simply reflects the demand during the lead time.

### 2.1.1 A Numerical Example

An oil company is drilling for oil in Alaska and concerned about its inventory of spare parts. It takes a month to receive the order. The demand for the special part A is assumed constant at a rate of 1225 per year. The order cost is $1000 per order. The holding cost is $20 per year. Determine the optimum order size \( q \) that minimizes the total inventory costs.

**Solution:**

\[ K = \$1000 \text{ per order} \]
\[ D = 1225 \text{ part per year} \]
\[ h = \$20 \text{ per unit per year} \]

\[ q^* = \sqrt{\frac{2 \times 1000 \times 1225}{20}} \]
\[ q^* = 350 \text{ parts} \]
Since it takes only one month to receive the shipment of 350 units, the order must be given when the inventory level drops down to 103 units or less. The ROP is \(103 \approx 1225/12\).

3 Heuristic Methods

3.1 Heuristics for Traveling Salesman Problem

Consider the following distance matrix.

<table>
<thead>
<tr>
<th></th>
<th>City 1</th>
<th>City 2</th>
<th>City 3</th>
<th>City 4</th>
<th>City 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>City 1</td>
<td>0</td>
<td>132</td>
<td>217</td>
<td>164</td>
<td>58</td>
</tr>
<tr>
<td>City 2</td>
<td>132</td>
<td>0</td>
<td>290</td>
<td>201</td>
<td>79</td>
</tr>
<tr>
<td>City 3</td>
<td>217</td>
<td>290</td>
<td>0</td>
<td>113</td>
<td>303</td>
</tr>
<tr>
<td>City 4</td>
<td>164</td>
<td>201</td>
<td>113</td>
<td>0</td>
<td>196</td>
</tr>
<tr>
<td>City 5</td>
<td>58</td>
<td>79</td>
<td>303</td>
<td>196</td>
<td>0</td>
</tr>
</tbody>
</table>

3.1.1 Nearest–Neighbor Heuristic (NNH)

Start with any city and visit the nearest city. Go to nearest unvisited city from the last visited city. Repeat this step until all cities are visited, then return the city of origin.

**Example:** Arbitrarily choose City 1.

The nearest city to City 1 is City 5, \(d_{15} = 58\).

The nearest unvisited city to City 5 is City 2, \(d_{52} = 79\).

The nearest unvisited city to City 2 is City 4, \(d_{24} = 201\).

The nearest unvisited city to City 4 is City 3, \(d_{43} = 113\).

The return trip to City 1 is \(d_{31} = 217\).

Therefore, the tour is 1–5–2–4–3–1 and the total distance traveled is 668.

If start from City 3, the tour is 3–4–1–5–2–3, and the distance traveled is \(113 + 164 + 58 + 79 + 290 = 704\).
3.1.2 Cheapest-Insertion Heuristic (CIH)

Begin at any city and find its nearest neighbor. Create a subtour joining these two cities. Next replace an arc in the subtour. If arc \((i, j)\) is replaced by arcs \((i, k)\) and \((k, j)\), then the length \(d_{ik} + d_{kj} - d_{ij}\) is added to the subtour. Repeat this step until all the cities are visited.

**Example:** Begin at City 1. The nearest city is City 5. Subtour is 1–5–1, and the distance traveled is 58 + 58 = 116.

<table>
<thead>
<tr>
<th>Arcs replaced</th>
<th>Arcs added to subtour</th>
<th>Added length</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,5)</td>
<td>(1,2),(2,5)</td>
<td>132+79-58=153</td>
</tr>
<tr>
<td>(1,5)</td>
<td>(1,3),(3,5)</td>
<td>217+303-58=462</td>
</tr>
<tr>
<td>(1,5)</td>
<td>(1,4),(4,5)</td>
<td>164+196-58=302</td>
</tr>
<tr>
<td>(5,1)</td>
<td>(5,2),(2,1)</td>
<td>79+132-58=153</td>
</tr>
<tr>
<td>(5,1)</td>
<td>(5,3),(3,1)</td>
<td>303+217-58=462</td>
</tr>
<tr>
<td>(5,1)</td>
<td>(5,4),(4,1)</td>
<td>196+164-58=302</td>
</tr>
</tbody>
</table>

The subtour: 1–5–2, and the total distance=58 + 79 + 132 = 269. Notice that 269 – 153 = 116 is added to the length of the subtour.

<table>
<thead>
<tr>
<th>Arcs replaced</th>
<th>Arcs added to subtour</th>
<th>Added length</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,2)</td>
<td>(1,3),(3,2)</td>
<td>217+290-132=375</td>
</tr>
<tr>
<td>(1,2)</td>
<td>(1,4),(4,2)</td>
<td>164+201-132=233</td>
</tr>
<tr>
<td>(2,5)</td>
<td>(2,3),(3,5)</td>
<td>290+303-79=514</td>
</tr>
<tr>
<td>(2,5)</td>
<td>(2,4),(4,5)</td>
<td>2201+196-79=318</td>
</tr>
<tr>
<td>(5,1)</td>
<td>(5,3),(3,1)</td>
<td>303+217-58=462</td>
</tr>
<tr>
<td>(5,1)</td>
<td>(5,4),(4,1)</td>
<td>196+164-58=308</td>
</tr>
</tbody>
</table>

The subtour 1–5–2–4–1 is obtained with total distance=164+201+79+58=502.

The distance added is 502-269=233.

<table>
<thead>
<tr>
<th>Arcs replaced</th>
<th>Arcs added to subtour</th>
<th>Added length</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,4)</td>
<td>(1,3),(3,4)</td>
<td>217+113-164=166</td>
</tr>
<tr>
<td>(4,2)</td>
<td>(4,3),(3,2)</td>
<td>113+290-201=202</td>
</tr>
<tr>
<td>(2,5)</td>
<td>(2,3),(3,5)</td>
<td>290+303-79=514</td>
</tr>
<tr>
<td>(5,1)</td>
<td>(5,3),(3,1)</td>
<td>303+217-58=462</td>
</tr>
</tbody>
</table>
The completer tour: 1–5–2–4–3–1 with a total distance=217+113+201+79+58=668 and the distance added is 668-502=166.

4 Dynamic Programming

Stage by stage solution method for solving problems with multi-stages.

4.1 Two problems

4.1.1 Match Puzzle

Suppose there are 30 matches on a table. I begin by picking up 1, 2, or 3 matches. Then my opponent must pick up 1, 2, or 3 matches. We continue in this fashion until the last match is picked up. The player who picks up the last match is the loser. How can I (the first player) be sure of winning the game?

4.1.2 Milk

I have a 9-litre cup and a 4-litre cup. I have to bring home exactly 6 liters of milk. How can I accomplish this goal?

References


Figure 2: The model and the solution of Example 1.1.1 by LINDO

max 3x1+5x2
st
5x1+3x2<30
3x1+4x2<21
2x1+4x2<19
end

LP OPTIMUM FOUND AT STEP 2

OBJECTIVE FUNCTION VALUE

1) 24.75000

VARIABLE VALUE REDUCED COST
X1 2.000000 0.000000
X2 3.750000 0.000000

ROW SLACK OR SURPLUS DUAL PRICES
2) 8.750000 0.000000
3) 0.000000 0.500000
4) 0.000000 0.750000

NO. ITERATIONS= 2
Figure 3: Behavior of a simple inventory system

Figure 4: Behavior of various cost components as a function of $q$