A genetic algorithm for the generalized transportation problem

W. Ho and P. Ji*

Department of Manufacturing Engineering, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong

Abstract: The Generalized Transportation Problem (GTP) is an extension of the linear Hitchcock transportation problem. However, it does not have the unimodularity property, which means the linear programming solution (like the Simplex Method) can not guarantee to be integer. This is a major difference between the GTP and the Hitchcock transportation problem. Although some special algorithms, such as the generalized stepping-stone method, have been developed, but they are based on the linear programming model and the integer solution requirement of the GTP is relaxed. This paper proposes a Genetic Algorithm (GA) to solve the GTP and a numerical example is presented to show the algorithm and its efficiency.

Keywords: generalized transportation problem, genetic algorithm, integer programming

*Author to whom all correspondence should be addressed.
1 INTRODUCTION

The transportation problem, being one of the famous linear programming models, was originally proposed by Hitchcock in 1941. Since then a great deal of the attention has been paid on the problem and many different versions of the problem have been proposed. The problem can be viewed as either linear or nonlinear problem depending on the objective of the model. Moreover, the problem can be classified into either planar or solid depending on what kind of constraints it subjects to [11].

In this paper, the problem discussed is called the generalized transportation problem [6, 15] or the machine loading problem [9], which was proposed by Ferguson and Dantzig in 1956 [10]. The generalized transportation problem (GTP) has been studying for a long time because it is found to be applicable to many situations, such as, machine loading; electrical power distribution; flow of funds in a company; scheduling of jobs on computers; and warehouse distribution.

Many researchers presented a number of methods to solve this problem. Lourie [15] solved the problem by using the stepping stone algorithm associated with the altering topology. Eisemann [9] proposed a generalized stepping stone algorithm, which is the extension of the loop technique of the stepping stone method for the Hitchcock transportation problem. Balas [5] adopted the ideas of underlying the usual stepping stone algorithm to specialize the dual method and the poly-(c) technique for the GTP. Balachandran and Thompson [1-4] studied an operator theory of parametric programming for the problem. Thompson and Sethi [17] developed a pivot and probe algorithm (PAPA) to solve an uncapacitated GTP with some side constraints. They took the advantage of the fact that the dual of an uncapacitated GTP has fewer variables than constraints, and used the probe operator to select only the most important constraints to consider. It was stated that the PAPA works better on a sparse than a dense problem, and it also performs well on fewer side constraints. Ji et al. [14] developed an algorithm for the dual form of the GTP from the idea of the revised simplex method. The algorithm is similar to the revised simplex method except the computation equations and procedure. It was found that the performance of the algorithm is much
better. Not only the starting solution and basis are much better than those of the revised simplex method, but also it does not require to develop the linear programming model and introduce slack or artificial variables. However, all of these algorithms are based on the linear programming model of the GTP and the integer solution requirement is relaxed.

In the recent years, the Genetic Algorithms (GA) have been applied successfully in a wide variety of optimization problems such as the quadratic assignment problem, the minimum spanning tree problem, and the traveling salesman problem [7-8, 11, 16]. The success is critical due to GA’s simplicity, ease of operation, and parallel and global perspective. The GTP, being an integer linear programming model, should also be solved using a GA, and this is the purpose of this paper.

The paper is organized as follows. Section 2 describes the mathematical model of the GTP. Section 3 provides a brief introduction to general genetic algorithms and a detailed genetic algorithm for the GTP. Section 4 shows a numerical example of the GTP solved by the GA. Section 5 shows how effective and efficient the GA is in tackling a large size GTP. Finally, Section 6 concludes the paper.

2 MATHEMATICAL MODEL

Suppose that there are m machines, which are required to produce n types of products. Each product can be produced on any machines. When machine i is assigned to produce product j, it requires $a_{ij} (> 0)$ hours per unit and costs $c_{ij} (> 0)$ dollars per unit. Besides, machine i has $t_i$ hours available, and the volume requirement for product j is $s_j$ units. The objective is to determine the quantity $x_{ij}$ of product j to be produced on machine i. To determine the quantity $x_{ij}$, an integer linear programming model can be formulated as follows:

\[
\text{MIN } \psi = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij} \quad \quad (1)
\]

SUBJECT TO

\[
\sum_{j=1}^{n} a_{ij} x_{ij} \leq t_i \quad \quad i = 1, 2, \ldots, m \quad (2)
\]
\[
\sum_{i=1}^{m} x_{ij} = s_j \quad j = 1, 2, \ldots, n \tag{3}
\]
\[
x_{ij} \geq 0 \text{ and integers} \quad i = 1, 2, \ldots, m; j = 1, 2, \ldots, n \tag{4}
\]

The objective function (1) is to minimize the total production cost. Constraint set (2) is due to the limited available time, which means that each machine must be operated within the fixed time period. Constraint set (3) is due to the product volume requirement, which means that the amount of each product must be produced with the product volume. Model IP1 is referred to as the Generalized Transportation Problem (GTP). Table 1 below shows a GTP tableau, which has four machines and five types of products. The northwest corner in cell \((i, j)\) indicates the unit time \(a_{ij}\), while the northeast corner represents the unit cost \(c_{ij}\). For example, in cell \((1, 1)\), the unit production time is 5 and the unit cost is 3.

The generalized transportation model is an extension of the linear Hitchcock transportation model indeed, where all \(a_{ij} = 1\). Also, it is more complex than the linear type due to the interacting nature of the constraints, and the integer solution requirement. Products compete with each other for allocation to economical and fast machines, but the capacity limitation forces some products to more expensive and slower machines. Furthermore, the linear Hitchcock transportation model possesses the unimodularity property, which means the optimal linear programming solution is automatically the optimal integer solution. However, the GTP does not have this property. Consequently, a linear programming method, like the simplex method, is not suitable for solving the generalized transportation model. So, this paper proposes to apply a Genetic Algorithm to solve the problem.

3 A GENETIC ALGORITHM

Genetic Algorithms (GA), invented by John Holland in the 1960s, are stochastic optimization techniques [7-8, 13, 16]. A Genetic Algorithm starts with an initial set of random solutions called
population. Each solution in the population is called a chromosome. The chromosomes evolve through successive iterations, called generations. During each generation, the chromosomes are evaluated by using some measures of fitness. The operations to generate new chromosomes, called offspring, are crossover and mutation. In the crossover phase, the GA attempts to exchange portions of two parents (two chromosomes in the population) to generate an offspring. The crossover operation speeds up the process to reach better solutions. In the mutation phase, the mutation operator maintains the diversity in the population to avoid being trapped in local optima. A new generation is formed by (1) selecting some of the parents and offspring according to their fitness values, and (2) rejecting others so to keep the population size constant. The fitter the chromosomes, the higher the probabilities of being selected. After the predetermined number of generations is performed, the algorithm converges to the best chromosome, which hopefully represents the optimal solution or may be a sub-optimal solution to the problem.

The genetic algorithm for the GTP is proposed in the following subsections. Here, a matrix is used to represent a chromosome in the GA, as shown in Figure 1.

### 3.1 Initialization

The basic idea of the initialization procedure to initialize a chromosome is:

1. (1) to select a random decision variable, say $x_{ij}$;
2. (2) to assign the available amount of units as many as possible to the decision variable $x_{ij}$; and
3. (3) to update the data such as the limited available time as well as the demand requirement so that the newly generated chromosome can satisfy all the constraint requirements.

The initialization procedure is listed as follows:

**Step 1:** Select a random number $k$ from set $\pi$, $\pi = \{1, 2, \ldots, mn\}$.

**Step 2:** Calculate the corresponding row $i$ and column $j$ by assigning: $i = \left[ 1 + \frac{k - 1}{n} \right]$ and $j = \left[ 1 + \frac{k}{n} \right]$.
Step 3: Assign the available amount of units to \( x_{ij} \): \( x_{ij} = \min \left( \frac{t_i}{a_{ij}}, s_j \right) \)

Step 4: Update \( t_i \) and \( s_j \): \( t_i = t_i - x_{ij} \times a_{ij} \) and \( s_j = s_j - x_{ij} \); and delete \( k \) from \( \pi \).

Step 5: Repeat Step 1 to Step 4, until \( \pi \) becomes empty.

The population size is denoted as \( psize \) and is predetermined by the GA user. The above procedure will be repeated \( psize \) times to generate \( psize \) chromosomes, that is, \( h = 1, 2, \ldots, psize \) in this initialization process.

3.2 Evaluation

In a GA, both parent and offspring chromosomes must be evaluated by some measures of fitness. For the GTP, the objective function of model IP1 is used for the evaluation purpose. In general, the evaluation function can be different from the objective. Let \( eval(X_h) \) be the fitness function for chromosome \( X_h \) in the GTP, then

\[
eval(X_h) = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}
\]

3.3 Selection

In the GA for the GTP, a roulette wheel approach [7, 12] is adopted in order to choose some chromosomes to undergo genetic operations. The approach is based on an observation that a roulette wheel has a section allocated for each chromosome in the population, and the size of each section is proportional to the fitness of the chromosome it corresponds to. It is true that the roulette wheel selection mechanism chooses chromosomes probabilistically, instead of deterministically. For example, although one chromosome has the highest fitness, there is no guarantee it must be selected. The only thing that is certain is that on the average a chromosome will be chosen with the probability proportional to its fitness. The selection procedure is listed as follows:

Step 1: Calculate the total fitness of the population:
\[ F = \sum_{h=1}^{\text{psize}} \text{eval}(X_h) \]

**Step 2:** Calculate the selection probability \( p_h \) for each chromosome \( X_h \):
\[ p_h = \frac{F - \text{eval}(X_h)}{F \times (\text{psize} - 1)}, \quad h = 1, 2, \ldots, \text{psize} \]

**Step 3:** Calculate the cumulative probability \( q_h \) for each chromosome \( X_h \):
\[ q_h = \sum_{j=1}^{h} p_j, \quad h = 1, 2, \ldots, \text{psize} \]

**Step 4:** Generate a random number \( r \) from the range \([0, 1]\).

**Step 5:** If \( q_{h-1} < r \leq q_h \), then chromosome \( X_h \) is selected.

### 3.4 Genetic operations

The genetic operators used in the algorithm for the GTP are the same as those used in the general GA. The genetic operators consist of two search mechanisms, called crossover and mutation.

The number of chromosomes selected for performing crossover and mutation operators depends on the crossover rate and the mutation rate, which are set by the GA user. Let \( \text{crossno} \) and \( \text{mut} \) denote the number of chromosomes selected to undergo the crossover and mutation, respectively, then \( \text{crossno} = \text{round}(\text{cr} \times \text{psize}) \) and \( \text{mut} = \text{round}(\text{mr} \times \text{psize}) \), where \( \text{cr} \) is the crossover rate, and \( \text{mr} \) is the mutation rate. It is because a pair of chromosomes is required to undergo the crossover operation, the number of pairs of chromosomes, denoted as \( \text{cross} \), is an integer, so

\[
\text{cross} = \begin{cases} 
\frac{\text{crossno}}{2} & \text{if } \text{crossno} \text{ is even} \\
\frac{\text{crossno} - 1}{2} & \text{otherwise}
\end{cases}
\]

#### 3.4.1 Crossover operation

**Step 1:** Implement the selection procedure in Section 3.3 (Step 1 to Step 5) twice to choose a pair of chromosomes, \( X_1 = (x^1_{ij}) \) and \( X_2 = (x^2_{ij}) \), from the population to perform the crossover
operation.

Step 2: Create two temporary matrices, \( D = (d_{ij}) \) and \( R = (r_{ij}) \), as follows:

\[
d_{ij} = \left( \frac{x_{ij}^1 + x_{ij}^2}{2} \right) \quad \text{and} \quad r_{ij} = (x_{ij}^1 + x_{ij}^2) \mod 2.
\]

Step 3: Divide matrix \( R \) into two matrices \( R_1 = (r_{ij}^1) \) and \( R_2 = (r_{ij}^2) \) such that \( R = R_1 + R_2 \) and

\[
\sum_{j=1}^{n} r_{ij}^1 = \sum_{j=1}^{n} r_{ij}^2 = \frac{1}{2} \sum_{j=1}^{n} r_{ij} \quad i = 1, 2, ..., m
\]
\[
\sum_{i=1}^{m} r_{ij}^1 = \sum_{i=1}^{m} r_{ij}^2 = \frac{1}{2} \sum_{i=1}^{m} r_{ij} \quad j = 1, 2, ..., n
\]

Step 4: Produce two offspring, \( X_1' \) and \( X_2' \), as follows:

\[
X_1' = D + R_1 \quad \text{and} \quad X_2' = D + R_2
\]

3.4.2 Mutation operation

Step 1: Select a matrix \( X \) (a chromosome) randomly as a parent for the mutation operation by following the selection procedure in Section 3.3.

Step 2: Extract a submatrix \( Y \) from the parent matrix by randomly selecting 2 rows and 2 columns.

Step 3: Reallocate the submatrix \( Y \). Use the initialization procedure at section 3.1 (Step 1 to Step 5) to assign new values to the submatrix such that all constraints are satisfied.

Step 4: Create an offspring by replacing the appropriate elements of the parent matrix with the new elements from the reallocated submatrix \( Y \).

3.5 GA procedure

The procedure of the GA for the GTP is listed as follows:

Step 1: Set the GA parameters, including the population size \( (p_{size}) \), the number of iterations \( (itno) \), the crossover rate \( (cr) \) and the mutation rate \( (mr) \).
Step 2: Generate $p_{size}$ initial chromosomes by following the initialization procedure in Section 3.1.

Step 3: Evaluate the fitness value $eval(X_h)$ for all chromosomes in the population.

Step 4: Follow the selection procedure in Section 3.3 to form cross pairs of chromosomes to perform the crossover operation in Section 3.4.1.

Step 5: Follow the selection procedure in Section 3.3 to select mut chromosomes to perform the mutation operation in Section 3.4.2.

Step 6: Compare all offspring, including the chromosomes generated from both the crossover and the mutation operations, with the chromosomes in the population by the fitness values ($eval(X_h)$). Retain the best $p_{size}$ chromosomes in the population.

Step 7: Determine the best chromosome at each iteration. Repeat Step 4 to Step 7 until $itno$ iterations are performed.

4 NUMERICAL EXAMPLE

The GTP example in Table 1 is used to illustrate how the GA works. An integer linear programming model for the problem can be formulated as follows:

$$\text{MIN } (3x_{11} + 6x_{12} + 6x_{13} + 7x_{14} + 4x_{15} + 6x_{21} + 5x_{22} + 6x_{23} + 15x_{24} + 8x_{25} + 5x_{31} + 3x_{32} + 10x_{33} +$$
$$6x_{34} + 7x_{35} + 4x_{41} + 4x_{42} + 7x_{43} + 8x_{44} + 9x_{45})$$

$$\text{SUBJECT TO}$$

$$5x_{11} + 2x_{12} + 2x_{13} + 5x_{14} + 6x_{15} \leq 48,000$$
$$2x_{21} + 7x_{22} + 8x_{23} + 7x_{24} + 6x_{25} \leq 48,000$$
$$4x_{31} + 2x_{32} + 2x_{33} + x_{34} + 6x_{35} \leq 48,000$$
$$3x_{41} + 2x_{42} + 4x_{43} + 5x_{44} + 2x_{45} \leq 48,000$$
$$x_{11} + x_{21} + x_{31} + x_{41} = 12,000$$
$$x_{12} + x_{22} + x_{32} + x_{42} = 8,000$$
$$x_{13} + x_{23} + x_{33} + x_{43} = 10,000$$
\[ x_{14} + x_{24} + x_{34} + x_{44} = 8,000 \]
\[ x_{15} + x_{25} + x_{35} + x_{45} = 6,000 \]
\[ x_{ij} \geq 0 \text{ and integer}; \quad i = 1, 2, 3, 4; j = 1, 2, 3, 4, 5. \]

(IP2)

The GA parameters are set as follows: the population size \( psize = 25 \), the iteration number \( itno = 1000 \), the crossover rate \( cr = 0.4 \), and the mutation rate \( mr = 0.3 \). Therefore, the number of pairs of chromosomes selected to perform the crossover operation, \( cross = 5 \). The mutation number, \( mut = 8 \). The appendix in this paper illustrates how the GA performs in the first iteration in detail. Figure 2 shows the minimum production cost obtained at each iteration. After 1,000 iterations, the best solution is shown in Table 2 with the production cost of 203,474. Although the GA cannot guarantee that the optimal solution can be found, the best solution determined by the GA for this specific GTP problem is near optimal. The error is only 0.1%. The optimal integer solution (in Table 3) can be obtained from a commercial package, like LINDO, by solving the original integer linear programming model, IP2.

From Figure 2, it can be noticed that the objective value drops sharply at the first 180 iterations. Since the population size \( psize \) is small, only 25, the GA can only produce some not-so-good chromosomes at beginning. Later, the GA generates good offspring quickly from those highly-fit parents. This phenomenon is called rapid convergence. When the objective gets closer to the best solution, the improvement rate becomes very slow. The curve levels off after 312 iterations, that is, the objective remains the same after that number of iterations.

5 A COMPUTATIONAL STUDY

In this section, a GTP with larger problem size is solved in order to examine the effectiveness and efficiency of the proposed GA. The problem size studied here is \( 20 \times 30 \) (i.e. 20 machines are available to produce 30 types of products). Therefore, there are totally 600 integer decision variables. In addition, the effect of the population size on the quality of the solution will be
studied. The proposed GA will be executed three times each with a unique population size including 25, 50, and 100, while the other GA parameters are kept constant. The iteration number \( itno = 5000 \), the crossover rate \( cr = 0.4 \), and the mutation rate \( mr = 0.3 \).

The performances as well as the results of the GAs are shown in Figure 3 and Table 4 respectively. In Figure 3, two phenomena can be noticed. First, all the curves drop significantly in the first 1,000 iterations, but they tend to be level off between the iteration number 4,000 – 5,000. Second, the curve of the larger population size is in a lower position. In the other words, the GA together with a larger population size can produce a better solution. However, it is found that the gap between the curves representing 50 and 100 population size is smaller than the gap between the curves representing 25 and 50 population size. It means that there is only a slight improvement even though the population size is doubled. Therefore, it is believed that there is no great contribution to the quality of the solution when the population size is too high. On the other hand, there is an adverse effect indeed. It is because more computational time will be required when the population size is higher.

Although the size of the problem being studied is large, the proposed GA is able to obtain a desirable solution in an acceptable computational time. According to Table 4, the best production cost found is 975,982. The error is only 3.2%.

6 CONCLUSIONS

The generalized transportation problem, as an extension of the linear Hitchcock transportation problem, is widely used in many situations. However, its mathematical model is a general integer linear programming formulation, and it does not have the unimodularity property. So, it is difficult to get the optimal integer solution. In this paper, a genetic algorithm was proposed to solve the generalized transportation problem. The procedure of the GA for the GTP was detailed in the paper. Finally, a numerical example was presented to illustrate the model of the GTP and the genetic algorithm.
ACKNOWLEDGMENTS

The authors wish to acknowledge the Hong Kong Polytechnic University for the financial support of the project (No. G-W031).

REFERENCES


APPENDIX

As mentioned before, there are seven steps in the GA for solving the GTP and finding the best production cost. The first iteration (generation) is described in detail in order to demonstrate how the genetic algorithm works. The steps are listed as follows:

Step 1: In this case, $psize = 25$, $itno = 1000$, $cr = 0.4$ and $mr = 0.3$. Therefore, $cross = 5$, and $mut = 8$.

Step 2: Generate 25 initial chromosomes by following the initialization procedure in Section 3.1. For instance, one initial chromosome is generated as follows:

Step 2.1: Select a random number $k$ from set $\pi$, $\pi = \{1, 2, ..., 20\}$.

For example, $k = 8$.

Step 2.2: Calculate the corresponding row $i$ and column $j$ by:

$$i = 1 + (8 - 1) / 5 = 2 \quad ; \quad j = 1 + (8 - 1) \text{ mod } 5 = 3$$

Step 2.3: Assign the available amount of units to $x_{23}$:

$$x_{23} = \min (t_2 / a_{23}, s_3) = \min (6000, 10000) = 6000$$

Step 2.4: Now, update $t_2$ and $s_3$ and delete $k = 8$ from set $\pi$:

$$t_2 = 48,000 - 6,000 \times 8 = 0; \quad s_3 = 10,000 - 6,000 = 4,000$$

Repeat the above procedure (Step 2.1 to Step 2.4), until $\pi$ becomes empty, and the initial chromosome is obtained as:

$$X_1 = \begin{bmatrix} 6400 & 8000 & 0 & 0 & 0 \\ 0 & 0 & 6000 & 0 & 0 \\ 0 & 0 & 0 & 8000 & 6000 \\ 5600 & 0 & 4000 & 0 & 0 \end{bmatrix}$$

Similarly, the remaining 24 chromosomes can be generated by following the above procedure.

Step 3: Evaluate the fitness value $eval(X_h)$ for all 25 chromosomes in the population. Here, the fitness value for $X_1$ is:

$$eval (X_1) = 6,400 \times 3 + 8,000 \times 6 + 6,000 \times 6 + 8,000 \times 6 + 6,000 \times 7 + 5,600 \times 4 + 4,000 \times 7 = 243,600$$
The table below shows the fitness values of all 25 chromosomes:

| eval(X_1)=243600 | eval(X_2)=271200 | eval(X_3)=253000 | eval(X_4)=235000 | eval(X_5)=276000 |
| eval(X_6)=272500 | eval(X_7)=276000 | eval(X_8)=204000 | eval(X_9)=306000 | eval(X_{10})=312800 |
| eval(X_{11})=248500 | eval(X_{12})=293200 | eval(X_{13})=250000 | eval(X_{14})=277000 | eval(X_{15})=266000 |
| eval(X_{16})=228800 | eval(X_{17})=291600 | eval(X_{18})=274400 | eval(X_{19})=331600 | eval(X_{20})=244400 |
| eval(X_{21})=266000 | eval(X_{22})=264400 | eval(X_{23})=331600 | eval(X_{24})=276000 | eval(X_{25})=295000 |

**Step 4:** 5 \((cross=5)\) pairs of chromosomes (or say 10 chromosomes) are selected to perform the crossover operation. In order to select the chromosomes, the procedure of the roulette wheel approach is listed as follows:

**Step 4.1:** Calculate the total fitness for the 25 chromosomes:

\[
F = \sum_{h=1}^{25} eval(X_h) = 6,788,200
\]

**Step 4.2:** Calculate the selection probability \(p_h\) for each chromosome:

\[
p_h = \frac{F - eval(X_h)}{F * (\text{size} - 1)}, \quad h = 1, 2, ..., 25
\]

The selection probability \(p_h\) for chromosome \(X_h\) is listed as follows:

| \(p_1\) = 0.0402 | \(p_2\) = 0.0400 | \(p_3\) = 0.0401 | \(p_4\) = 0.0402 | \(p_5\) = 0.0400 |
| \(p_6\) = 0.0400 | \(p_7\) = 0.0400 | \(p_8\) = 0.0404 | \(p_9\) = 0.0398 | \(p_{10}\) = 0.0397 |
| \(p_{11}\) = 0.0401 | \(p_{12}\) = 0.0399 | \(p_{13}\) = 0.0401 | \(p_{14}\) = 0.0400 | \(p_{15}\) = 0.0400 |
| \(p_{16}\) = 0.0403 | \(p_{17}\) = 0.0399 | \(p_{18}\) = 0.0400 | \(p_{19}\) = 0.0396 | \(p_{20}\) = 0.0402 |
| \(p_{21}\) = 0.0400 | \(p_{22}\) = 0.0400 | \(p_{23}\) = 0.0396 | \(p_{24}\) = 0.0400 | \(p_{25}\) = 0.0399 |

**Step 4.3:** Calculate the cumulative probability \(q_h\) for each chromosome:

\[
q_h = \sum_{j=1}^{h} p_j, \quad h = 1, 2, ..., 25
\]

The cumulative probability \(q_h\) for chromosome \(X_h\) is:

| \(q_1\) = 0.0402 | \(q_2\) = 0.0802 | \(q_3\) = 0.1203 | \(q_4\) = 0.1605 | \(q_5\) = 0.2005 |
| \(q_6\) = 0.2405 | \(q_7\) = 0.2805 | \(q_8\) = 0.3209 | \(q_9\) = 0.3607 | \(q_{10}\) = 0.4004 |
| \(q_{11}\) = 0.4405 | \(q_{12}\) = 0.4804 | \(q_{13}\) = 0.5205 | \(q_{14}\) = 0.5605 | \(q_{15}\) = 0.6005 |
| \(q_{16}\) = 0.6408 | \(q_{17}\) = 0.6807 | \(q_{18}\) = 0.7207 | \(q_{19}\) = 0.7603 | \(q_{20}\) = 0.8005 |
| \(q_{21}\) = 0.8405 | \(q_{22}\) = 0.8805 | \(q_{23}\) = 0.9202 | \(q_{24}\) = 0.9601 | \(q_{25}\) = 1.0000 |
Step 4.4: Generate a number \( r \), ranging from 0 to 1, 10 times randomly for the purpose of selecting 10 chromosomes from the population. If the first two random numbers are 0.0218 and 0.8782, for example, then \( X_1 \) and \( X_{22} \) will be selected as the first pair of chromosomes to perform the crossover operation (refer to Section 3.4.1). Here,

\[
X_1 = \begin{bmatrix} 6400 & 8000 & 0 & 0 & 0 \\ 0 & 0 & 6000 & 0 & 0 \\ 0 & 0 & 0 & 8000 & 6000 \\ 5600 & 0 & 4000 & 0 & 0 \end{bmatrix} \quad \text{and} \quad X_{22} = \begin{bmatrix} 0 & 0 & 10000 & 5600 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 8000 & 0 & 2400 & 6000 \end{bmatrix}.
\]

Create two temporary matrices, \( D = (d_{ij}) \) and \( R = (r_{ij}) \).

\[
D = \begin{bmatrix} 3200 & 4000 & 5000 & 2800 & 0 \\ 0 & 0 & 3000 & 0 & 0 \\ 6000 & 0 & 0 & 4000 & 3000 \\ 2800 & 4000 & 2000 & 1200 & 3000 \end{bmatrix} \quad \text{and} \quad R = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.
\]

If matrix \( R \) contains 1’s, then divide \( R \) into \( R_1 \) and \( R_2 \), whereas \( R_1 \) and \( R_2 \) must have the same total number of 1’s on each column. In this case, it is noticed that there is no 1 in matrix \( R \).

Therefore, two offspring \( X_1' \) and \( X_{22}' \) are the same:

\[
X_1' = X_{22}' = \begin{bmatrix} 3200 & 4000 & 5000 & 2800 & 0 \\ 0 & 0 & 3000 & 0 & 0 \\ 6000 & 0 & 0 & 4000 & 3000 \\ 2800 & 4000 & 2000 & 1200 & 3000 \end{bmatrix}.
\]

Step 5: 8 chromosomes are selected to perform the mutation operation. Assume \( X_1 \) is one of the chromosomes selected as a parent for mutation (refer to Section 3.4.2), then

\[
X_1 = \begin{bmatrix} 6400 & 8000 & 0 & 0 & 0 \\ 0 & 0 & 6000 & 0 & 0 \\ 0 & 0 & 0 & 8000 & 6000 \\ 5600 & 0 & 4000 & 0 & 0 \end{bmatrix}
\]

Randomly select rows 1 and 4, columns 1 and 3. The corresponding submatrix \( Y \) and the reallocated submatrix are:

<table>
<thead>
<tr>
<th>Corresponding submatrix ( Y )</th>
<th>Reallocated submatrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>\begin{bmatrix} 6400 &amp; 0 \ 5600 &amp; 4000 \end{bmatrix}</td>
<td>\begin{bmatrix} 4800 &amp; 4000 \ 7200 &amp; 0 \end{bmatrix}</td>
</tr>
</tbody>
</table>

Replacing the reallocated submatrix into \( X_1 \), the offspring after mutation is:
Step 6: Calculate the fitness values for all offspring (totally 18), and compare them with the parents in the population. In this case, the fitness values of the three offspring are: $X_1'' = 241,200$, and $X_1' = X_2' = 254,000$. So, $X_1'$, $X_1''$, and $X_2'$ will replace parents $X_{10}$, $X_{19}$ and $X_{23}$ because these three parents are the worst in the population.

Step 7: The best chromosome for the first iteration is $X_8$ because it has the smallest fitness value. Repeat Step 4 to Step 6 until 1,000 iterations are performed.
Table 1  A Generalized Transportation Problem.

<table>
<thead>
<tr>
<th>Product j</th>
<th>Machine 1</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_{11}</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{12}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{13}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{14}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{15}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t_j</td>
<td>48000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Machine 2</td>
<td>2</td>
<td>6</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{21}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{22}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{23}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{24}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{25}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Machine 3</td>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{31}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{32}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{33}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{34}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{35}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Machine 4</td>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{41}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{42}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{43}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{44}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x_{45}</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>s_j</td>
<td>12000</td>
<td>8000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2  The best chromosome after 1,000 iterations.

| x_{11} = 526 | x_{12} = 0 | x_{13} = 4,683 | x_{14} = 0 | x_{15} = 6,000 |
| x_{21} = 0   | x_{22} = 0 | x_{23} = 5,317 | x_{24} = 0 | x_{25} = 0   |
| x_{31} = 0   | x_{32} = 8,000 | x_{33} = 0    | x_{34} = 8,000 | x_{35} = 0 |
| x_{41} = 11,474 | x_{42} = 0    | x_{43} = 0    | x_{44} = 0    | x_{45} = 0   |

Total production cost = 203,474

Table 3  The optimal integer solution.

| x_{11} = 800 | x_{12} = 0 | x_{13} = 4,000 | x_{14} = 0 | x_{15} = 6,000 |
| x_{21} = 0   | x_{22} = 0 | x_{23} = 6,000 | x_{24} = 0 | x_{25} = 0   |
| x_{31} = 0   | x_{32} = 8,000 | x_{33} = 0    | x_{34} = 8,000 | x_{35} = 0 |
| x_{41} = 11,200 | x_{42} = 0    | x_{43} = 0    | x_{44} = 0    | x_{45} = 0   |

Total production cost = 203,200

Table 4  The results of the GAs.

<table>
<thead>
<tr>
<th>Population size</th>
<th>25</th>
<th>50</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best solution found</td>
<td>1,006,723</td>
<td>981,845</td>
<td>975,982</td>
</tr>
<tr>
<td>Quality</td>
<td>93.5%</td>
<td>96.2%</td>
<td>96.8%</td>
</tr>
<tr>
<td>Computational time</td>
<td>23 mins</td>
<td>45 mins</td>
<td>90 mins</td>
</tr>
</tbody>
</table>

Optimal production cost = 945,636
\[ X_h = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{ml} & x_{m2} & \cdots & x_{mn} \end{bmatrix} \]

**Figure 1** Representation of a chromosome.

**Figure 2** The best production cost at each iteration.
Figure 3  The performances of the GAs.
Author/s:
Ho, W; Ji, P

Title:
A genetic algorithm for the generalised transportation problem

Date:
2005-07-11

Citation:

Persistent Link:
http://hdl.handle.net/11343/120165

File Description:
Accepted version