Hybrid Two-Stage Algorithm for Solving Transportation Problem

Saleem Z. Ramadan (Corresponding author) Department of Mechanical and Industrial Engineering Applied Science Private University PO box 166, Amman 11931, Jordan E-mail: s_ramadan@asu.edu.jo

Imad Z. Ramadan				
Finance and Banking Science Department				
Applied Science Private University				
PO box 166, Amman 11931, Jordan				
Tel: 962-796-177-911 E-mail: prof_imad67@yahoo.com				

Received: February 6, 2012Accepted: February 22, 2012Published: April 1, 2012doi:10.5539/mas.v6n4p12URL: http://dx.doi.org/10.5539/mas.v6n4p12

Abstract

In this paper a hybrid two-stage algorithm is proposed to find the optimal solution for transportation problem (TP). The proposed algorithm consists of two stages: the first stage uses genetic algorithm (GA) to find an improved nonartificial feasible solution for the problem and the second stage utilizes this solution as a starting point in the RSM algorithm to find the optimal solution for the problem. The algorithm utilizes big M method to handle \geq constraints and northwest corner method, minimum cost method, and Vogel's method are also used to generate the initial population for the GA. Performance of the algorithm is tested under different simulated scenarios and compared to both GA and revised simplex method (RSM). The results showed that the new hybrid algorithm performs competitively against GA and RSM. The proposed algorithm can be easily extended to cover different kinds of linear programming (LP) problems with minor changes such as inventory control, employment scheduling, personnel assignment and transshipment problems.

Keywords: Linear programming, Genetic algorithms, Transportation problem

1. Introduction

Matrix notation is widely used to express linear programming (LP) problems. In this notation, let X be an *n*-vector that represents the variables in the problem, A be an (*m by n*)-matrix that represents the constraints coefficients in the problem, and C be an *n*-vector that represents the objective function coefficients in the problem, then the LP problem with *m* constraints and *n* variables can be expressed in a matrix notation as

AX = b

with *m* constraints and *n* variables can be expressed in a maximize or minimize
$$z = CX$$

s.t.

 $X \ge 0$ where **b** is an *m*-vector that represents the right hand side values of the constraints in the LP problem. Another widely used notation for AX = b constraints is

$$\sum_{j=1}^{n} \mathbf{P}_{j} \mathbf{x}_{j} = \mathbf{b}$$

where \mathbf{P}_i is the *j*th column of **A**.

A basic solution of AX= b can be found by solving *m* equations with *m* variables after setting (*n*-*m*) variables equal to zeros under the condition of having a unique solution. Let X_B be the set of the basic variables, then **B** is said to form a basis if the *m*-vectors in **B** are linearly independent.

Using the combination formula, the number of possible basic solutions for any LP (feasible and infeasible) can be given by

$$C_n^m = \frac{n!}{m!(n-m)!} \tag{1}$$

Noting that the right hand side of the constraint in the LP is always positive, the feasibility condition can be given by

$$\mathbf{B}^{-1}\mathbf{b} \ge 0 \tag{2}$$

For a given X_B , **B**, and corresponding objective vector C_B , any simplex iteration can be expressed by two equations; the optimality condition equation,

$$z + \sum_{i=1}^{n} (z_i - c_i) x_i = C_B B^{-1} b_i$$

and the feasibility condition equations,

$$(X_B)_i + \sum_{j=1}^n (B^{-1}P_j)_i x_j = (B^{-1}b)_i$$

where

$$\mathbf{z}_{j} - \mathbf{c}_{j} = \mathbf{C}_{B}\mathbf{B}^{-1}\mathbf{P}_{j} - \mathbf{c}_{j}$$

and $(\mathbf{Y})_i$ represents the *i*th element of the vector \mathbf{Y} .

The rule of thumb in simplex method is to choose the entering variable for the maximization (minimization) problem, x_j , as the variable that will have the most negative (positive) value for $z_j - c_j$, and to choose the leaving variable, x_j , as the variable with the minimum ratio according to

$$\min_{i} \left\{ \frac{(B^{-1}b)_{i}}{(B^{-1}P_{j})_{i}} \left| ((B^{-1}P_{j}))_{i} > 0 \right\}$$
(3)

The transportation problem (TP) can be considered as a special case of LP that deals with the problem of finding the optimal shipping schedule that will minimize the total shipping cost between S sources and D destinations. The solution should guarantee that the supply and demand limits are met. An assumption is made that the shipping cost and the number of units shipped are proportional. Let $s = \{1, 2, ..., S\}$ be the set of sources and let $d = \{1, 2, ..., D\}$ be the set of destinations, then the TP can be formulated as

$$\min \operatorname{mize} \sum_{i=1}^{S} \sum_{j=1}^{D} c_{ij} x_{ij}$$
(4)

subject to
$$\sum_{i=1}^{s} x_{ij} \ge a_j \quad \forall j \in d$$
 (5)

$$\sum_{j=1}^{D} x_{ij} \le b_i \quad \forall i \in s$$
(6)

$$\mathbf{x}_{ii} \ge 0 \quad \forall i \in \mathbf{s}, j \in \mathbf{d}$$

where x_{ij} represents the amount transported from source *i* to destination *j*, c_{ij} the cost of transporting of one unit from source *i* to destination *j*, a_j the need of destination *j*, and b_i the capacity of source *i*. Constraint (5) ensures that the demand of each destination *j* will be satisfied and constraint (6) ensures that the capacity of each source *i* will not be exceeded.

Metaheuristic and exact algorithms to solve this problem have been extensively developed in literature due to the importance of this problem in real life issues. Lin (2010) used Differential Evolution to solve the TP where the demand and supply are both fuzzy quantities. Kakol et al. (2010) developed CABI (Conceptual Algorithm Based on Bee Intelligence) to solve the unbalanced TP where the behavior of bees was exploited to solve the problem. Chun and Yi (2009) used GA to solve a fixed charge TP associated with multiple transportation modes selection of a supply chain where three types of costs are considered. Kutcha and Chanas (1996) suggested a definition for the optimal solution of the TP with fuzzy cost coefficients and suggested an algorithm. Gen and Li (1998) exploit the spanning tree special data structure for the TP to develop a GA for solving the bicriteria TP. Kutcha and Chanas (1998) proposed an algorithm to solve the TP when the integrality condition imposed on the solution. Sun et al. (1998) used tabu search approach to solve the TP making use of the network based implementation of the Simplex method. Adlakha and Kowalski (2003) proposed a simple heuristic algorithm

for solving small TP using mixed integer programming formulation. Gen et al. (1998) used a hybrid GA to solve the bi-criteria TP exploiting the concept of spanning tree and Prüfer number for initialization. Gen et al. (1995) solved the TP using GA when the decision makers have an optimistic point of view.

Most of the literatures solve the TP using metaheuristic optimization techniques. While the metaheuristic optimization techniques generally got the benefit of solving large size problems at a reasonable computational cost, it does not guarantee the feasibility or the optimality of the solution obtained or even, in many cases, it cannot tell how close the solution obtained is to the optimal solution. In addition, the metaheuristic optimization techniques use stochastic elements, thus, generally speaking, the solution will not be the same when the algorithm is repeated for the same problem. Moreover, metaheuristic optimization often caught in a local optima trap where the algorithm converges to a local optima instead of the global optima and there is no way to know that. For these reasons, it is not wise to rely totally on the matheuristic optimization techniques, therefore, the exact algorithms that guarantee the optimal solution are still very important even though, in general, they have relatively higher computation time.

2. Objective

The aim of this paper is to exploit the good traits of both optimization techniques such that the fast computational time form the metaheuristic techniques and the ability to guarantee the optimal solution from the exact techniques will be exploited. A hybrid algorithm is proposed that will exploit the good traits of both techniques to solve the TP. The algorithm will utilize GA to produce an improved nonartificial feasible solution that will be used as a starting point for the RSM to produce the optimal solution within a reasonable computational time. Figure 1 is a schematic representation for the proposed algorithm.

3. Algorithms and Methods

Due to the special structure of the transportation problem, nonartificial initial basic solution for the problem can be generated by one of three methods: northwest-corner method, least-cost method, Vogel's approximation method. The three methods are different in the quality of the solution found. Generally speaking, North-west method is the worst and Vogel's approximation is the best. It is worthwhile to mention that none of the three methods can guarantee the optimal solution for the problem, therefore, they only can be used to generate a nonartificial starting point for an exact optimization method like simplex or revised simplex method.

3.1 Northwest-Corner Method

The northwest-corner method starts with a variable x_{11} corresponds to the northwest-corner cell of the tableau. The method goes in these steps

Step 1: Allocate as much as possible to the selected cell and adjust the corresponding quantities of supply and demand by subtracting the allocated quantity from them.

Step 2: Remove the row or the column that corresponds to any source or destination with zero quantity. If it happens that a supply and a demand zeros out simultaneously, remove the corresponding row or column of one of them arbitrary and leave the other one with zero quantity.

Step 3: If only one row or column left then stop. Otherwise, if a row has just been removed, move to the cell that is just below the current cell. If a column has just been removed, move to the cell that is just to the right of the current cell. Go to step 1.

3.2 Least-Cost Method

The least-cost method keeps track of the cheapest route. The method starts by allocating as much as possible to the cheapest shipping unit cost cell. In short the method goes in these steps:

Step 1: Allocate as much as possible to the selected cell and adjust the corresponding quantities of supply and demand by subtracting the allocated quantity from them.

Step 2: Remove the row or the column that corresponds to any source or destination with zero quantity. If it happens that a supply and a demand zeros out simultaneously, remove the corresponding row or column of one of them arbitrary and leave the other one with zero quantity.

Step 3: If only one row or column left then stop. Otherwise, look for the cheapest uncrossed unit cost cell. Go to step 1.

3.3 Vogel's Approximation Method (VAM)

VAM is an enhanced version of the least-cost method, the method goes in these steps:

Step 1: For each row (column), determine a penalty quantity by subtracting the cheapest shipping unit cost from the next cheapest shipping unit cost in the same row (column).

Step 2: Select the cell that has the cheapest shipping unit cost in the row (column) with the largest penalty. Allocate as much as possible to the selected cell and adjust the corresponding quantities of supply and demand by subtracting the allocated quantity from them.

Step 3: Remove the row or the column that corresponds to any source or destination with zero quantity. If it happens that a supply and a demand zeros out simultaneously, remove the corresponding row or column of one of them arbitrary and leave the other one with zero quantity.

Step 4: Do one of the following

- 1) If only one row or column left with zero supply or demand then stop.
- 2) If one row or column left with positive supply or demand then determine the basic variables in the row (column) by least-cost method and then stop.
- 3) If all the left rows and columns with zero supply or demand then determine the zero basic variables in the rows and columns by least-cost method and then stop.
- 4) Otherwise, go to step 1.

3.4 Genetic Algorithm

Evolutionary algorithms (EAs), as metaheuristic optimization techniques, are characterized with a population that evolves over time. The underlying principle for EAs can be traced back to Darwin's theory of evolution. The principle is easy; given a population of individuals, the environmental pressure will dictate which individual will survive by a process called natural selection. In short, the strong individual will adapt to the surroundings and survives and the weak one will not adapt to the surroundings and eventually dies out (Popov A., 2003; Goldberge E. G., 1989). GAs have certain traits that make them the most popular among the different EAs. GAs use both crossover and mutation operators which make their population more diverse and consequently more immune to be trapped in a local optima. In theory, the diversity helps the algorithm to be faster in reaching the global optima as it allows the algorithm to explore the solution space faster.

3.5 Overview of the Genetic Algorithm

The GA that will be used in this paper is as follows:

3.5.1 Chromosome Representation

The chromosome used in the GA contains the basic variables for the TP. The number of genes in each chromosome will be the same as the total number of sources and destinations. The gene value will assume an integer value from one to the total number of variables in the problem. The genes will construct a distinct set such that no repetition is allowed.

3.5.2 Initial Population Generation

Seeded initial population strategy will be adopted. Three feasible chromosomes, only, will be generated for the initial population corresponding to northwest corner method, minimum cost method, and Vogel's approximation method. The generation process of those chromosomes will guarantee the feasibility of them.

It is important here to emphasize that the intention of the GA step is *not* to solve the TP, but to produce an improved nonartificial starting point for the RSM in a reasonable computational time. This is the reason why a low number of initial population will be used.

3.5.3 Mating

To increase the diversity of the population at the beginning, a deterministic Best-worse mating strategy will be adopted for the first two thirds of the generations such that the best chromosome found in that generation will be mating with the worst chromosome found in the same generation. To exploit the information accumulated over the first two thirds of the generations about the promising solutions, a deterministic Best-second-Best mating strategy will be adapted for the last third of the generations such that the best chromosome found in that generation will be mating with the second best chromosome found in the same generation.

3.5.4 Crossover Operator

A random length-based crossover strategy will be used for crossover in which the number of genes from the Best chromosome that will be taken to the offspring will be a random number from the close set of 1 to the total number of genes in that parent. The rest of the genes in the offspring will be taken from the second part of the second parent

such that only genes that are not in the first part of the offspring will be taken from the second part of that parent. Those genes will be replaced by genes taken randomly from the *common* non-basic variables set of the two parents to ensure that no repetition of genes will happen, in case that some genes are repeated. The common non-basic variables set is the set generated by the intersection between the non-basic variables set of the first parent and the non-basic variables set of the second parent.

A linear function will be used to calculate the crossover rate, P_c as follows:

$$P_{\rm C} = 1 - \frac{i}{\rm Ng}$$

where N_g is the total number of generation.

This formula will set P_c almost 1 at the beginning. This will have a high crossover probability that will explore the solution space adequately and determine the promising areas of the optimal solution. The crossover rate will be reduced linearly as the number of generations *i* advanced until a value of zero is reached at the last generation Ng. The reason of using such a decreasing crossover rate function is to increase the diversity in the population to explore the sample space adequately at the beginning and then to reduce it, as the information about the optimal solution is accumulated over generations, so that the information accumulated about the optimal solution over generations will not be lost by high rate of crossover.

3.5.5 Mutation Operator

A random-position mutation strategy will be used to introduce random variations in the chromosomes. This strategy will select a gene to be mutated randomly from a chromosome and then replace this gene with a gene selected randomly from the non-basic variables set of that chromosome. A linear function will be used to calculate the mutation rate P_m as follows:

$$P_m = \frac{i}{Ng}$$

This will set the mutation rate about zero at the beginning where crossover rate is very high (and thus mutation is not needed) and 1, as its maximum value, when crossover rate is 0 and mutation is more needed. The reason of using such an increasing function for the mutation rate is to exploit the information about the optimal solution accumulated over generations, promising areas, in the chromosomes.

3.5.6 Fitness Function

The fitness function used in this GA represents the total shipping cost between S sources and D destinations and it can be given in a matrix format as $z = C_B B^{-1}b$. It is clear that all what is needed to calculate the fitness value for a certain chromosome is the set of this chromosome's genes, which contains the set of the current basic variables. From this chromosomes and from the original problem, the B^{-1} , **b**, and C_B vectors can be determined and the fitness value, z, can be calculated. Since the TP is a minimization problem, the smaller the z, the fitter the chromosome. This criterion will be used in the selection process, which will be discussed next.

3.5.7 Selection

The enlarged sample space strategy will be used in this GA in which the offspring and the parents will form a population pool and all the members in the pool will compete for their survival (evolutionary process) according to their fitness. This enlarged sample space strategy will increase the diversity in the selection process, which is believed to enhance the performance of the GA.

It is worthwhile to note that the process for generating the offspring by crossover and mutation strategies *does not guarantee* feasibility. A feasibility check using equation (2) must be done for the generated offspring before considering it in the pool and if it is not feasible the offspring must be excluded from the pool.

3.5.8 Termination Criterion

The GA will stop when certain number of generations N_g are reached. N_g depends on the size of the problem through the number of variables involved in that problem. As the size of the problem increases the number of variables involved in that problem also increases, therefore, the generation number will increase according to the following equation:

$Ng = round(10 + 0.001 * N_v)$

where N_v is the number of variables involved in the problem. For a TP with S = D, N_v can be calculated by:

 $N_v = (\text{problem size})^2 + 3 \times \text{problem size}$

where problem size is the number of destinations or sources in a the TP.

3.6 Revised Simplex Method Algorithm (RSM)

While the Simplex method calculates all the numbers in the tableau from which many are not needed, RSM calculates those numbers that are relevant only, hence, it is considered more efficient than Simplex method in terms of computational time. Also, the round-off errors and significant digit loss are problems that characterize the Simplex method especially when the entries of the matrix differs widely in their order of magnitude. This will cause round-off errors in the memory of a computer. The RSM can avoid such problems, and thus using the RSM is considered necessary in this respect.

The RSM will be used in the second step where an improved nonartificial feasible basic solution, generated from the GA step, will be used as the starting point for this step. The RSM algorithm can be summarized as follows:

- 1) Pick the entering variable x_j along with its associated vector p_j .
 - a. Evaluate $Y = c_B B^{-1}$
 - b. For all non-basic variables compute $z_j c_j = Y P_j c_j$
 - c. Choose the largest negative value (maximization problem). If there is no negative value then stop.
- 2) Determine leaving variable, x_r along with its associated vector p_r .
 - a. Evaluate the current right hand side by evaluating $x_B = B^{-1}b$
 - b. Evaluate the current constraint coefficients of entering variable $a_i = B^{-1}p_i$
 - c. Apply minimum ratio rule given by equation (3)
- 3) Find current **B** and evaluate B^{-1}
- 4) Go to step 1,

where p_i is the vector associated with x_i variable in the constraint matrix, z_j is the contribution of variable x_j in the objective function, c_j is the objective coefficient of the variable x_j c_B is the vector of coefficients of the basic variables in the objective function, and **B** is the basic variable matrix.

4. Illustration Example for Crossover and Mutation

Assume a company has three warehouses (S1, S2, and 33) and wants to satisfy the three customer's needs (D1, D2, and D3) with minimum transportation cost. Table 1 summarizes the cost of transportation along with the capacity and needs for each warehouse and customer respectively.

From equations 5 and 6 it is clear that there are six constraints. Since there are three demand constraints, three negative surplus variables and three positive surplus variables will be added for those three constraints. The addition is that each constraint will have extra two variables one with positive sign and one with negative sign. For the capacity constraints a slack variable will be added in each constraint. In the objective function six variables will be added with zero coefficients and three with coefficients M where M is a very large positive number. The resulting problem is formulated as

min imize
$$z = 32x_{11} + 40x_{12} + 120x_{13} + 60x_{21} + 68x_{22} + 104x_{23} + 200x_{31} + 80x_{32} + 60x_{33}$$

 $-0s_1 + Ms_2 - 0s_3 + Ms_4 - 0s_5 + Ms_6 + 0s_7 + 0s_8 + 0s_9$
subject to : $x_{11} + x_{21} + x_{31} - s_1 + s_2 = 30$
 $x_{12} + x_{22} + x_{32} - s_3 + s_4 = 35$
 $x_{13} + x_{23} + x_{33} - s_5 + s_6 = 30$
 $x_{11} + x_{12} + x_{13} + s_7 = 20$
 $x_{21} + x_{22} + x_{23} + s_8 = 30$
 $x_{31} + x_{32} + x_{33} + s_9 = 45$
 $x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33}, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9 \ge 0$

The number of basic variables will be 6 and the number of non basic variables will be 12. The variables will be given a sequential numbers starting from 1 and ending in 18 such that 1 represents x_{11} , 2 represents x_{12} , 3 represents x_{13} , 4 represents x_{21} , ..., 18 represents s_9 , based on which the chromosome representation for this problem will consist of six genes. The values of those gene are taken randomly from the closed set {1, 2, 3, ..., 18} without repetition. Suppose that the best chromosome found in generation 3 was [2 3 5 1 12 14] and the worst chromosome for that generation was [7 6 12 18 2 13]. This means that the *common* set of non basic variables for those two

chromosomes is $[4 \ 8 \ 9 \ 10 \ 11 \ 15 \ 16 \ 17]$. To do the crossover, a random number between 1 and 6 will be drawn, say 4. The first 4 genes from the Best chromosome will be used in the first 4 genes of the offspring, this means that the first part of the offspring will be $[2 \ 3 \ 5 \ 1 \ X \ X]$. The second part, the other two genes, will be taken from the second part of the worst chromosome such that no repetition is allowed in the offspring. This means that $[2 \ 13]$ should be taken from the worst chromosome but since this will create a repetition in the offspring (infeasible offspring since number 2 will be repeated, the only number that will be taken is 13 for gene number 6. The value for the remaining gene, gene number 5, will be selected randomly from the set of the *common* non basic variables, say 11. This will produce the complete offspring $[2 \ 3 \ 5 \ 1 \ 11 \ 13]$.

For this offspring, the non basic variable set is [4 6 7 8 9 10 12 14 15 16 17 18]. To do the mutation for this offspring, a random number between 1 and 6 will be selected to represent the gene number that will be mutated from the offspring. Suppose that gene number 4 was selected then a number from the non basic variable set will be selected randomly, say 7. This 7 will replace gene number 4 from the offspring. The mutated offspring will be [2 3 5 7 11 13]. A feasibility check using equation (2) for this offspring will be done to make sure that it is feasible and then added to the pool if it turns out that it is feasible or neglected otherwise.

5. Results and Discussion

10 different TPs were generated randomly to investigate the effectiveness of the proposed algorithm and solved using the RSM alone, GA alone, and the proposed hybrid algorithm for the sake of comparison. The machine used to solve the problems has the following specifications: Manufacturer HP, Model HPE-500f, Processor AMD phenon (tn) IIX6 1045T processor 2.70GHz, RAM 8.0 GB, system 64-bit operating system.

The maximum allowed time for solving the problems was set to 1 day (86400 seconds), if the optimal solution reached before this time, the program will terminate and will record the optimal solution along with the time the solution was reached, otherwise, the program will terminate and will indicate that the optimal solution was not reached. Table 2 shows the problem size along with the number of variables needed in RSM to solve the problem. It is clear that as the problem size increases the number of variables increases, therefore, the solution space increases and the time needed to reach the optimal solution, generally, increases because, according to equation (1), the number of possible basic solutions increases as well.

Table 3 shows the results for the ten problems solved using the RSM only. The table shows that RSM was able to reach the optimal solution, within the time limitation, for problems up to 251500 variables. RSM did not reach the optimal solution for the problem with 564750 variables so the algorithm was terminated at 86400 second. It is obvious from the table that the time taken for the RSM increases with increasing the problem size, which was expected.

Table 4 shows the results for the same ten problems solved using the GA used in step 1 alone. The table shows that GA was able to reach the optimal solutions within the time limitations for problems up to 22950 variables. From the table, it is clear that the times needed for the RSM to reach the optimal solutions for problems up to 22950 were less than the times needed by the GA. The GA was not able to find the optimal solutions for the problems beyond problem of 22950 variables within the time limitation with the given number of generations. This can be understood in the light of the large sample space (huge number of possible basic solutions that the GA has to search according to equation (1). To appreciate the huge number of possible basic solutions (feasible and infeasible) for problem of 130 variables, equation (1) can be used to calculate this number as,

number of possible basic solutions =
$$C_{130}^{20} = \frac{130!}{20!(130-20)!} = 1.6736E + 23$$

The number obtained is very huge and thus one can imagine how huge is the sample space for problem containing 40600 variables and beyond (can't be evaluated even by computers). The huge sample space and the relatively low generation number used in the GA restrict the quality of solution reached by the GA to what is presented in Table 4.

Table 5 shows the results for the same ten problems solved using the proposed hybrid algorithm. It is clear that the hybrid algorithm was able to reach the optimal solution for all the problems within the time limitation. Comparing the results in Table 5 with the results in Table 4, it is clear that the GA step, which provided improved nonartificial initial feasible solution to the RSM, helped the RSM to reach the optimal solutions with lower computational time since the same code for the RSM was used in both situations and the only difference is that in Table 5 the initial feasible solution was provided by the GA opposed to the default initial basic solution generated by the RSM in Table 4.

Moreover, Table 6 compares the time needed to reach the optimal solution for the RSM and the proposed hybrid algorithm. It is clear that the proposed hybrid algorithm performs better in problems with number of variables beyond 22950 variables with improvement in computational time up to 29.9% corresponds to the problem size of 40600 variables. The table also shows that the hybrid algorithm performs poor with problems below 1720 variables.

To understand the behavior of the proposed algorithm, it is beneficial to keep in mind the general behavior of GA. The general convergence behavior of GA toward the optimal solution is as shown in Figure 2. The algorithm has a high rate of improvement for the fitness values at the early generations and after that, the rate of improvement for the fitness value decreases.

This means that 1 generation at the beginning of the evolutionary process, generally, improves the fitness value more than 1 generation at the end. Exploiting this behavior, GA can be run for a few generations (the generations that have the highest improvement rate on the fitness) to improve the initial population founded by the northwest corner method, minimum cost method and Vogel's approximation method. This can exploit the high rate of improvement for the fitness value of the TP for the early generations. After that RSM will be used to find the optimal solution for the problem using this improved nonartificial initial solution as a starting point.

The performance of the hybrid algorithm was worse than the RSM for the small problems. This is because the time used by GA to generate an improved nonartificial starting point for the RSM was not justified by the time saved in RSM because of using this starting point. This is the reason why the hybrid algorithm performed worse than the RSM for the small size problems. As the size of the problem increases the time saved by using the improved initial solution in the RSM overcomes the time needed to produce this initial solution (time needed for GA part) and thus the overall time of reaching the optimal solution improved. This is very clear in the problems with sizes above 130 variables.

It is clear that beyond problem size 40600 variables, the improvement in time for the hybrid algorithm starts to decline. Unfortunately, problems with more than 564750 variables could not be solved due to the limitations in the Matlab version used, and thus it is unknown what will happen for problems with number of variables larger than 564750 variables.

Figure 3 shows the trend of improvement in the computational time with number of variables used in the problem. The solid line shows the actual data while the dashed line shows the trend in the data. If the same trend continues, it is clear that the hybrid algorithm will still perform better than RSM for problems up to 1,000,000 variables. This trend shows clearly that using the proposed hybrid algorithm will reduce the computational time for solving the TP problem when the size of the problem is as large as 1,000,000 variables.

6. Conclusions

This study investigated the benefit of using a hybrid two stage algorithm (GA-RSM) to solve the TP problem. In the first stage GA was used to produce an improved nonartificial initial basic variables set. The improved nonartificial initial basic variables set were used in the second stage as the starting point for the RSM.

The results, for large size problems, indicated that using GA to generate an improved nonartificial initial basic solution for the RSM can enhance the overall computational time needed to reach the optimal solution by the RSM only and GA only. The results also showed that the time spent in improving the initial solution for the RSM using GA is not justified when the number of variables involved in the problem is low. This is because of the overall computational time for RSM only. In short, the results show that the hybrid algorithm performs competitively against GA and RSM when high number of variables involved in the problem.

This hybrid algorithm can be easily extended to cover a variety of problems like inventory control, employment scheduling, personnel assignment and transshipment problems with minor changes.

Acknowledgment

The author is grateful to the Applied Science Private University, Amman, Jordan, for the financial support granted to this research (Grant No. DRGS-2011-28.)

References

Adlakhaa, V., & Kowalskib, K. (2003). A simple heuristic for solving small fixed-charge transportation problems. *Science direct*, 31, 205-211.

Chanas, S., & Kuchta, D. (1996). A concept of the optimal solution of the transportation problem with fuzzy cost coefficients. *Elsevier, Fuzzy Sets and Systems*, 82, 299-305. http://dx.doi.org/10.1016/0165-0114(95)00278-2

Chanas, S., & Kuchta, D. (1998). Fuzzy integer transportation problem. *Elsevier, Fuzzy Sets and Systems*, 98, 291-298. http://dx.doi.org/10.1016/S0165-0114(96)00380-6

Chun, F., & Yi, Z. (2009). A Genetic Algorithm of Two-stage Supply Chain Distribution Problem Associated with Fixed Charge and Multiple Transportation Modes. *IEEE Computer Society*, Fifth International Conference on Natural Computation, 76-80. http://dx.doi.org/10.1109/ICNC.2009.585

Gen, M., Ida, K., Li, Y., & Kubota, E. (1995). Solving Bicriteria Solid Transportation Problem with Fuzzy Numbers by A Genetic Algorithm. *Computers ind. Engng.*, 29, 537-541.

Gen, M., Ida, K., & Li, Y. (1998). Bicriteria Transportation Problem by Hybrid Genetic Algorithm. *Computers ind. Eng.*, 35, 363-366. http://dx.doi.org/10.1016/S0360-8352(98)00095-3

Gen, M., & Liy, Y. (1998). Spanning Tree-based Genetic Algorithm for Bicriteria Transportation Problem. *Elsevier, Computers ind. Eng.*, 35, 531-534. http://dx.doi.org/10.1016/S0360-8352(98)00151-X

Goldberge, E. G. (1989). Genetic Algorithms in Search Optimization and Machine Learning, Adison-Wesely.

Kakol, A., Grotowski, T., Koszalka, L., Kasprzak, A., & Burnham, K. (2010). Performance of Bee Behavior-based Algorithm for Solving Transportation Problem. *IEEE Computer Society*, Fifth International Conference on Systems, 83-87. http://dx.doi.org/10.1109/ICONS.2010.22

Lin, F. (2010). Using Differential Evolution for the Transportation Problem with Fuzzy Coefficients. *IEE Computer society*, International Conference on Technologies and Applications of Artificial Intelligence, 299-304. http://dx.doi.org/10.1109/TAAI.2010.56

Popov, A. (2003). Genetic Algorithms for Optimization. Technical University, Sifia.

Sun, M., Aronson, J., McKeown, P., & Drinka, D. (1998). A tabu search heuristic procedure for the fixed charge transportation problem. *Elsevier, European Journal of Operational Research*, 106, 441-456. http://dx.doi.org/10.1016/S0377-2217(97)00284-1

Table 1. Data for the illustration example

		Destinati	ons	
Sources	D1	D2	D3	Capacity
S1	32	40	120	20
S2	60	68	104	30
S3	200	80	60	45
Need	30	35	30	

Table 2. Problem size versus number of variables

Problem size	N_{v}
3 by 3	18
10 by 10	130
40 by 40	1720
60 by 60	3780
100 by 100	10300
150 by 150	22950
200 by 200	40600
300 by 300	90900
500 by 500	251500
750 by 750	564750

Problem size	Simplex method time	Optimal solution
3 by 3	0.0016	1.33E+04
10 by 10	0.0449	3.15E+04
40 by 40	2.1418	7.60E+06
60 by 60	7.1739	3.68E+07
100 by 100	42.8267	2.29E+08
150 by 150	164.9802	1.30E+09
200 by 200	427.7174	2.24E+12
300 by 300	1.7904E+03	3.65E+13
500 by 500	2.8044E+04	4.5623E+14
750 by 750	8.6400E+04	NO

Table 3. Time and optimal solutions for the ten problems using RSM

Table 4. Time and number of generations used to reach the optimal solution using GA method

Problem size	GA time	Number of generations	Optimal solution reached
3 by 3	0.0053	4	yes
10 by 10	0.32	75	yes
40 by 40	23.90	163	yes
60 by 60	59.60	236	yes
100 by 100	303.60	352	yes
150 by 150	1223.50	423	yes
200 by 200	86400.00	1323	No
300 by 300	86400.00	1023	No
500 by 500	86400.00	863	No
750 by 750	86400.00	593	No

Table 5. Time needed to reach the optimal solution using hybrid algorithm

Problem	Hybrid	GA step	Number of	Optimal
size	time	solution	gen	solution
3 by 3	0.0093	1.33E+04	10	1.33E+04
10 by 10	0.0549	3.41E+04	10	3.15E+04
40 by 40	1.69	8.61E+06	12	7.60E+06
60 by 60	5.61	4.57E+07	14	3.68E+07
100 by 100	32.102	2.83E+08	20	2.29E+08
150 by 150	120.81	1.58E+09	33	1.30E+09
200 by 200	299.74	2.68E+12	51	2.24E+12
300 by 300	1.44E+03	4.71E+13	101	3.65E+13
500 by 500	2.36E+04	5.93E+14	262	4.56E+14
750 by 750	8.43E+04	1.02E+17	575	7.89E+16

Problem	Simplex	Hybrid time	Improvement %
size	method time	<u>j</u>	I
3 by 3	0.0016	0.0093	-481.3%
10 by 10	0.0449	0.0549	-22.3%
40 by 40	2.1418	1.69	20.8%
60 by 60	7.1739	5.61	21.8%
100 by 100	42.82	32.10	25.0%
150 by 150	164.98	120.81	26.8%
200 by 200	427.71	299.74	29.9%
300 by 300	1.79E+03	1.44E+03	19.1%
500 by 500	2.80E+04	2.36E+04	15.6%
750 by 750	Optimal N/A	8.89E+04	No comparison

Table 6.	Comparison	between Hybrid	time and Revised	simplex method
	1	2		1











Figure 3. Improvement % in computational time vs. number of variables used in the TP