Indian Journal of Science and Technology, *Vol 7(7), 949-954, July 2014*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Step Fixed Charge Transportation Problems via Genetic Algorithm

S. Molla-Alizadeh-Zavardehi[1], A. Mahmoodirad[2*] and M. Rahimian[2]

[1]Department of Industrial Engineering, Masjed Soleiman Branch, Islamic Azad University, Masjed Soleiman, Iran; saber.alizadeh@gmail.com
[2]Department of Mathematics, Masjed Soleiman Branch, Islamic Azad University, Masjed Soleiman, Iran; alimahmoodirad@yahoo.com

Abstract

In this paper, we consider the step fixed-charge transportation problem where is one of the most important problems in transportation research area. To tackle such an NP-hard problem, we present Genetic Algorithm (GA). Since crossover and mutation operators have significant role on the algorithm's quality, some crossover and mutation operators are tested in this work. For this aim, several problem sizes are generated at random and then through extensive computational experiments, appropriate GA parameter values were chosen. Besides, the efficiency and convergence of the proposed algorithms was evaluated by solution quality. The results showed that the GA was more robust and consistently outperformed Simulated Annealing (SA) for all instances.

Keywords: Genetic Algorithm, Step Fixed Charge Transportation Problem, Transportation Problem

1. Introduction

Fixed Charge Problems (FCP) arise in a large number of production and transportation systems. Such FCPs are typically modeled as 0-1 integer programming problems. A special case of the general FCP is Fixed Charge Transportation Problem (FCTP). The problem involves the distribution of a single commodity from a set of supply centers (sources) to a set of demand centers (destinations) such that the demand at each destination is satisfied without exceeding the supply at any source. The objective is to select a distribution scheme that has the least cost of transportation. Two kinds of costs are considered, a continuous cost which linearly increases with the amount transported between a source i and a destination j and a fixed charge which is incurred whenever a nonzero quantity is transported between source i and destination j . The fixed charge may represent toll charges on a highway; landing fees at an airport; setup costs in production systems or the cost of building roads in transportation systems. Depending on the specific applications, the importance of the fixed charge in the model will vary.

Fixed charge problems were first proposed by Hirsch and Dantzig[4]. Initial at tempts to solve the problem were mainly heuristic in nature. The best known heuristics are by Balinski[2]. Murty[9] developed the first exact algorithm; his algorithm employs a vertex ranking procedure and works best when the fixed charges are small in comparison to the continuous costs. Steinberg[11] provided an exact algorithm based on the branch and bound method. But, the exact branch and bound method is applicable to small problems only, since the effort to solve an FCTP grows substantially with the size of the problem as explained in Walker[13]. A good deal of effort has been devoted to finding approximate solutions to FCTPs. The heuristic methods try to reach the optimum through simplex like iterations. Cooper and Drebes[3], Denzler[4], Steinberg[11], and Walker[13] have developed heuristic adjacent extreme point

**Author for correspondence*

algorithms for the general FCTP. Since there is no guarantee that the solution obtained in the final iteration is a minimum (local or global), they experiment with substituting different combinations of variables into the basis to obtain better results. However, the criterion used to stop the simplex iterations (positive gradients between the location identified in the final iteration and its neighboring peaks), while sufficient for a Linear Program (LP), does not guarantee optimality for the FCTP. Adlakha and Kowalski[1] proposed a simple heuristic algorithm for solving small FCTP. However, it is stated that the proposed method is more time consuming than the algorithms for solving a regular transportation problem. Several heuristic methods were proposed for solving fixed cost transportation problem[5,11,12].

Step Fixed Charge Transportation Problem (SFCTP) is an extended version of the FCTP. The SFCTP in its representation first was founded by Kowalski and Lev[6]. In the SFCTP due to the step function structure of the objective function, Kowalski and Lev[6] were dealing with a "NP-hard" problem.

Since the problems with fixed charges are usually NP-hard (nondeterministic polynomial time) problem[7], the computational time to obtain exact solutions increases in a polynomial fashion and very quickly becomes extremely long as the dimensions of the problem increase[6]. In order to find the best solution, we proposed the genetic algorithm. It is known that the effectiveness of a GA highly depends on the great choice of the encoding scheme, and the selection, crossover, and mutation operators, as well as their parameters by which they are applied[10].

The rest of the paper is organized as follows: in Section 2, the SFCTP model is described. The GA developed to the SFCTP in section 3. In Section 4, the experimental design and comparisons are presented. Finally, the conclusion and future work are presented in Section 5.

2. Step Fixed Charge Transportation Problem

The following notations are used to define the mathematical model.

Set of indices:

I set of suppliers ($i = 1,2,\ldots,I$)
J set of customers ($j = 1,2,\ldots,J$)

Parameters:

S_i capacity of supplier i
D_j capacity of customer j
$c_{i,j}$ cost of transporting one unit of product from supplier i to customer j
$A_{i,j}$ a certain amount of transporting from supplier i to customer j
$k_{i,j,1}$ fixed charge of transporting one unit of product from supplier i to customer j
$k_{i,j,2}$ additional fixed cost when the transported units exceeds a certain amount $A_{i,j}$

Decision variables:

$x_{i,j}$ quantity of product shipped from plant i to customer j
$b_{i,j,1}$ binary variable equal to 1 if $x_{i,j} > 0$ and equal to 0 otherwise
$b_{i,j,2}$ binary variable equal to 1 if $x_{i,j} > A_{i,j}$ and equal to 0 otherwise

The mathematical model of the problem as follows[6]:

$$\text{Min} \quad \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}\, x_{ij} + \sum_{i=1}^{m}\sum_{j=1}^{n}\left(b_{ij,1}\, k_{ij,1} + b_{ij,2}\, k_{ij,2}\right)$$

s.t

$$\begin{aligned}
&\sum_{i=1}^{m} x_{ij} = S_i && i = 1,2,\ldots,m,\\
&\sum_{i=1}^{n} x_{ij} = D_j && j = 1,2,\ldots,n,\\
&x_{ij} \geq 0, && \forall i, j,\\
&b_{ij,1} = \{0,1\}, && \forall i, j,\\
&b_{ij,2} = \{0,1\}, && \forall i, j,
\end{aligned}$$

where $k_{i,j,1}$, $k_{i,j,2}$, $k_{i,j}$, $A_{i,j} \geq 0$.

Note that fixed cost associated with route (i, j) has two steps. It could have multiple steps, depending on the problem structure. Without loss of generality, we assume that the problem is balanced, that is:

$$\sum_{i=1}^{n} S_i = \sum_{j=1}^{m} D_j \qquad S_i, D_j, c_{ij}, k_{ij} \geq 0.$$

3. The Genetic Algorithm Proposed

Genetic Algorithms (GAs) arise in the 1970s by the work of Holland in 1975. They were intended to tackle industrial problems that were difficult to solve with the methods available at that time. Nowadays, GA is considered to be one of the typical metaheuristic methods for tackling various optimization problems. The idea behind GA comes from Darwin's 'survival of the fittest' concept, meaning that good parents produce better offspring.

GA employs a population of chromosomes each of them represents an encoded solution. A fitness value is allocated to each chromosome according to its performance in which the more desirable the chromosome, the higher the fitnedss value becomes[8]. By using genetic operators, each successive incremental improvement in a chromosome becomes the basis for the next generation. The process continues by a set of genetic operators until some stopping criterion is met. Four fundamental steps are mostly used in GAs: reproduction, roulette wheel, crossover and mutation (for more detailed see[8]).

3.1 Initialization

Each generated chromosome is considered as an individual solution to the problem. In the first generation chromosomes are generated as many as population size. The random method is applied for generating the initial population.

3.2 Selection Mechanism

As the total transportation cost including variable and fixed costs should be minimized in this problem, better solutions are those results with lower objective functions. The higher fitness value considered the better chromosome, so the applied function is formulated as follows:

$$\text{Fitness Value} = \frac{1}{\text{Objective Function}}$$

Since the Roulette-Wheel selection mechanism is deployed, the chromosomes with higher fitness values have more chance to be selected.

3.3 Genetic Operators

Reproduction: The P_r% of the chromosomes with higher fitness values are transferred to the next generation.

Crossover: Crossover combines the two selected chromosomes' features in order to create two better offsprings. The remaining, (1-P_r%), of the chromosomes in next generation going to be generated from crossover operation. We used the one-point, two- point and Uniform crossover operators in this paper.

Mutation: The mutation operator is an important process of any successful GA that reorganizes the structure of the genes so that the algorithm can escape from searching just in local optimum area. It can also be regarded as a simple local search technique.

Performing the crossover operation, offspring are going to be mutated with the probability of P_m. It means that a random number in range [0, 1] is generated for each of the offsprings. If this number was less than P_m then the mutation operation is going to be performed. We utilize the Swap, Big Swap, Inversion and Displacement mutation operators in this paper.

4. Experimental Design

4.1 Instances

Zavardehi et al.[8] generated random insta-nces to verify the effectiveness of their GA approach. We use the same datasets except step cost in this paper. To cover various types of problems, we considered several levels of influencing inputs. First, we generated random problem instances for m = 10, 15, 30, and 50 suppliers and n = 10, 15, 20, 30, 50, 100, and 200 customers, respectively. We considered both small-sized and large-sized problem instances, which was presented by the number of suppliers and customers. Seven different problem sizes, 10 ×10, 10 ×20, 15 ×15, 10 ×30, 50 ×50, 30 ×100 and 50 ×200 are considered for experimental study, which present different levels of difficulty for alternative solution methods. After specifying the size of problems in a given instance, considering the significant influence of the fixed costs to the solution for each size, four problem types (A–D) are employed. For a given problem size, problem types differ from each other by the range of fixed costs, which increases upon progressing from problem type A through problem type D. The variable costs range over the discrete values from 3 to 8. The problem sizes, types, suppliers/customers, and fixed costs ranges are shown in Table 1.

4.2 Parameter Setting

The performance of the GA is generally sensitive to the parameter setting which influences the search efficiency and the convergence quality. Twenty-eight test problems, with different sizes and specifications, are generated and solved to evaluate the performance of the presented algorithms.

The instances are implemented using MATLAB on a PC with dual core Duo 2 2.8 GHz and 4 GB of RAM. All algorithms ran 3 times and Due to having different scale of objective functions in each instance the relative

Table 1. Test problems characteristics

Problem size	Total Demand	Problem type	Range of variable costs				Range of first fixed costs			Range of second fixed costs		
			Aij	a^l	a^l-b^l	α and β	a^l	a^l-b^l	α and β	a^l	a^l-b^l	α and β
10×10	10,000	A	400	U(3, 7)	U(0, 1)	U(0.25, 1)	U(50, 200)	U(0, 25)	U(5, 25)	U(50, 200)	U(0, 25)	U(5, 25)
10×20	15,000	B	400	U(3, 7)	U(0, 1)	U(0.25, 1)	U(100, 400)	U(0, 50)	U(10, 50)	U(100, 400)	U(0, 50)	U(10, 50)
15×15	15,000	C	400	U(3, 7)	U(0, 1)	U(0.25, 1)	U(200, 800)	U(0, 100)	U(20, 100)	U(200, 800)	U(0, 100)	U(20, 100)
10×30	15,000	D	400	U(3, 7)	U(0, 1)	U(0.25, 1)	U(400, 1,600)	U(0, 200)	U(40, 200)	U(400, 1,600)	U(0, 200)	U(40, 200)
50×50	50,000											
30×100	30,000											
50×200	50,000											

percentage deviation (RPD) is used for each instance. The RPD is obtained by the following formula:

$$RPD = \frac{Algsol - Minsol}{Minsol} \times 100$$

where Algsol and Minsol are the obtained objective value and minimum objective value found from both proposed algorithms for each instance, respectively. After obtaining the results of the test problems in different trial, results of each trial are transformed into RPD measure.

Using the average of RPD measures of trials, the parameters and operators that have minimum RPD average are selected as the best ones. Therefore, the parameters of GA were set as follows: population size = 100, Crossover percentage = 0.8, Mutation probability = 0.15, Crossover operator = Two-point crossover, Mutation operator = Displacement Mutation.

4.3 Experimental Results

We set searching time to be identical for both algorithms which is equal to 1.5× (n + m) milliseconds. Hence, this criterion is affected by both n and m. The more the number of suppliers and customers, the more the rise of searching time increases. We generated 20 instances for each twenty eight problem type, summing to 28×20=560 instances which are different from the ones used for parameter setting to avoid bias in the results. For further comparison, the maximum generations is set to 1000. The best objective function and their convergence after 1000 generations are reported in Figure 1. The improvement of GA on SA is

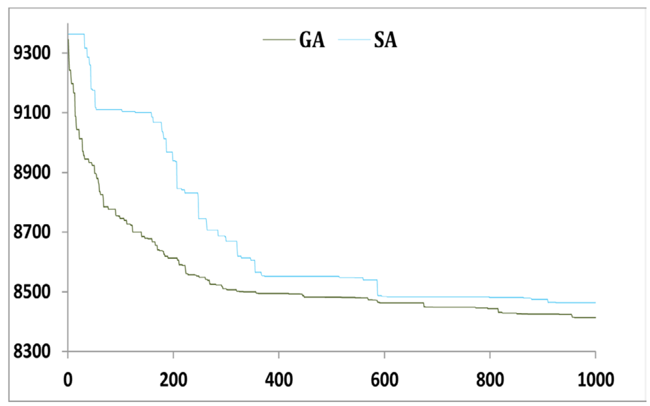

Figure 1. Evolution in 1000 generations.

obvious. From this figures, it is concluded that GA has a better convergence than SA on the instances.

Considering 20 instances for each of the 28 problem type, or 80 instances for each of the 7 problem sizes, for both algorithms, the instances have been run 5 times and hence, by using the RPD we deal with 400 data for each algorithm.

The averages of these data for each algorithm and each instance are shown in Figure 2.

In order to verify the statistical validity of the results, we have performed an Analysis Of Variance (ANOVA) to accurately analyze the results. The point that can be concluded from the results is that there is a clear statistically meaningful difference between performances of the algorithms. The means plot and LSD intervals (at the 95% confidence level) for two algorithms are shown in Figure 3.

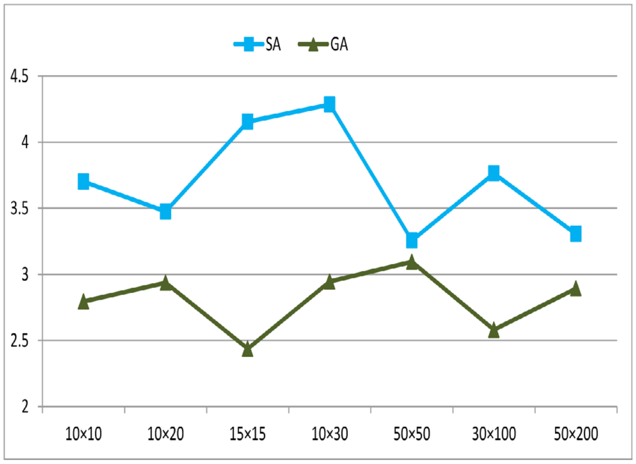

Figure 2. Means plot for the interaction between each algorithm and problem size.

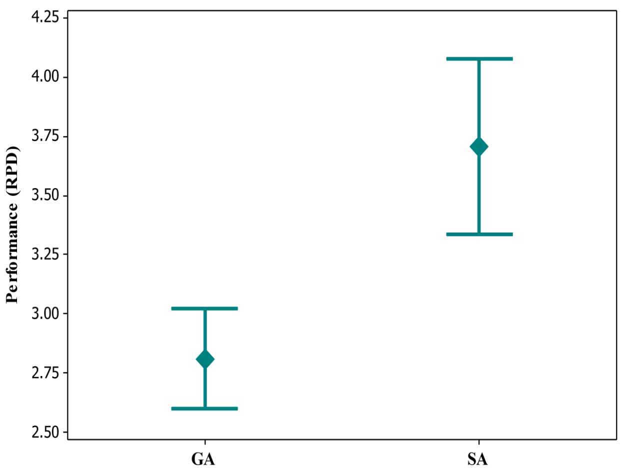

Figure 3. Means plot and LSD intervals for the GA and SA algorithms.

Since, we are to appraise the robustness of the algorithms in different circumstances, the effects of the problem sizes on the performance of both algorithms are analyzed. The reciprocal between the capability of the algorithms and the size of problems is illustrated in Figure 2. As can be seen from the result figure, not only is the overall performance of GA better than SA, but GA is more robust. Thus, GA has the capability to reduce the search space significantly and to obtain better solutions with less computational time than SA.

5. Conclusion and Future Works

In this paper, a real-world modeling of transportation problem has been investigated. We developed three crossover and four mutation operators for the problem from the GA literature. The robustness of the algorithm may be improved by fine-tuning the GA parameters and operators, relating the population size, reproduction percentage, mutation probability, crossover and mutation types. We solved the randomly generated problems by GA and also with SA to compare them. The obtained results show the proficiency of GA comparison with SA. Results showed that the GA proposed was capable of obtaining better solutions with a more reasonable computational time compared to the SA, for all sizes. The future work is to extend our approach to the case of multistage SFCTP, fuzzy costs, or other optimization objectives. Also, considering other well-known metaheuristics such as tabu search or new ones such as imperialist competitive algorithm is encouraged.

6. Acknowledgement

This study was supported under research project entitled "Addressing a fixed-charge solid transportation problem in multi-stage supply chain" by Islamic Azad University, Masjed-Soleiman Branch. The first author is grateful for this financial support.

7. Reference

1. Adlakha V, Kowalski K. A simple heuristic for solving small fixed-charge transportation problems Omega. Int J Manag Sci Eng Manag. 2003; 31:205–211.
2. Balinski ML. Fixed cost transportation problems. Naval Research Logistics. 1961. 8, 41–54.
3. Cooper L, Drebes C. An approximate solution method for the fixed charge problem. Nav Res Logist Q. 1967; 14:101–13.
4. Denzler DR. An approximate algorithm for the fixed charge problem. Nav Res Logist Q. 1964; 16:411–16.
5. Gottlieb J, Paulmann L. Genetic algorithms for the fixed charge transportation problem. Proceedings of IEEE International Conference on Evolutionary Computation, Anchorage. 1998; 330–35.
6. Kowalski K, Lev B. On step fixed-charge transportation problem. OMEGA, The International Journal of Management Science. 2008; 36(5):913–17.
7. Hirsch W, Dantzig GB. The Fixed Charge Problem. Nav Res Logist Q. 1968; 15:413–24.
8. Molla-Alizadeh-Zavardehi S, Hajiaghaei-Keshteli M, Tavakkoli-Moghaddam R. Solving a capacitated fixed-charge transportation problem by artificial immune and

genetic algorithms with a Prüfer number representation. Expert Systems with Applications. 2011; 38:10462–74.
9. Murty KG. Solving the Fixed Charge Transportation Problem by Extreme Point Ranking. Operations research Quarterly. 1968; 16:268–79.
10. Ruiz R, Maroto C. A genetic algorithm for hybrid flow shops with sequence dependent setup times and machine eligibility. European Journal of Operational Research. 2006; 169, 781–800.
11. Steinberg DI. (1970). The fixed charge problem. Nav Res Logist Q. 1970; 17:217–36.
12. Sun M, Aronson JE, Mckeown PG, Drinka D. A tabu search heuristic procedure for the fixe charge transportation problem. Eur J Oper Res. 1998; 106:441–45.
13. Walker WE. A heuristic adjacent extreme point algorithm for the fixed charge problem. Management Science. 1976; 22:587–96.