Computers & Operations Research ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem

K. Antony Arokia Durai Raj, Chandrasekharan Rajendran*

Department of Management Studies, Indian Institute of Technology Madras, Chennai 600036, India

ARTICLE INFO

Keywords:
Genetic algorithm
Two-stage transportation problems
Fixed-charge
Distribution center
Opening cost

ABSTRACT

Transportation of goods in a supply chain from plants to customers through distribution centers (DCs) is modeled as a two-stage distribution problem in the literature. In this paper we propose genetic algorithms to solve a two-stage transportation problem with two different scenarios. The first scenario considers the per-unit transportation cost and the fixed cost associated with a route, coupled with unlimited capacity at every DC. The second scenario considers the opening cost of a distribution center, per-unit transportation cost from a given plant to a given DC and the per-unit transportation cost from the DC to a customer. Subsequently, an attempt is made to represent the two-stage fixed-charge transportation problem (Scenario-1) as a single-stage fixed-charge transportation problem and solve the resulting problem using our genetic algorithm. Many benchmark problem instances are solved using the proposed genetic algorithms and performances of these algorithms are compared with the respective best existing algorithms for the two scenarios. The results from computational experiments show that the proposed algorithms yield better solutions than the respective best existing algorithms for the two scenarios under consideration.

1. Introduction

A supply chain refers to the flow of material through different facilities, starting with raw-materials and ending with finished products delivered to final consumers. A multi-stage distribution problem is a typical problem for firms with supply chain networks [15]. A two-stage transportation problem involves the freight transportation network from plants to customers through distribution centers (DCs) or warehouses. A firm may have constraints with respect to opening or operating a DC. These DCs are generally assumed to have capacitated or uncapacitated storage facilities. In this paper, a two-stage transportation problem with two different scenarios is considered. The first scenario considers the per-unit transportation cost and the fixed cost associated with every route, and unlimited capacity at every DC (or a warehouse). The second scenario reckons with the opening cost of a DC, per unit transportation cost from a plant to a DC and also from a DC to a customer.

The presence of fixed costs results in the objective function being a step function [1]. The fixed-charge transportation problem (FCTP) is NP-hard, hence heuristic methods have been proposed in the past to solve multi-stage transportation problems by considering different scenarios in the supply chain (e.g. [19,25,8,15,2]). The research on fixed-charge transportation problem (FCTP) has centered mostly on single-stage distribution problems and there is a need to develop efficient algorithms to solve two-stage distribution problems [15]. In this paper, two-stage transportation problems with two scenarios are considered, and a genetic-algorithm-based solution technique (genetic algorithm coupled with an improvement scheme, called TSGA) is proposed. Subsequently, the uncapacitated two-stage fixed-charge transportation problem (Scenario-1 problem) is represented as a single-stage FCTP. This single-stage transportation problem is solved using the proposed genetic algorithm, called SSGA (single-stage GA). The performance of the proposed methods and the best existing methods in the respective two scenarios are compared by making use of benchmark problem instances available in the literature.

2. Literature review

The fixed charge problem was first formulated by Hirsch and Dantzig [14] (see [17]). Balinski [3] formulated the fixed charge transportation problem, explained the special properties of FCTP, and proposed an approximate method to solve the FCTP. Many researchers developed solution techniques to solve the single-stage fixed-charge transportation problems in the past (e.g. [17,5,22,18,20]). All exact algorithms are computationally intractable in view of the problem being NP-hard [23].

* Corresponding author. Tel.: +91 44 2257 4559; fax: +91 44 2257 0509.
E-mail address: craj@iitm.ac.in (C. Rajendran).

0305-0548/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2011.09.020

Nomenclature

l	number of plants
m	number of DCs
n	number of customers
C_{ij}	cost per unit of transportation from plant i to DC j
F_{ij}	fixed charge associated with the shipment from plant i to DC j
T_{jk}	cost per unit of transportation from DC j to customer k.
G_{jk}	fixe dcharge associated with the shipment from DC j to customer k
S_i	supply available at plant i
W_j	capacity available at DC j (applicable only to Scenario-2; in case of Scenario-1, the capacity of a DC is assumed to be unlimited)
D_k	demand requirement at customer k
H_j	fixed-cost for operating (i.e., or opening) DC j (applicable only to Scenario-2; in case of Scenario-1, the cost does not exist)
U	an upper limit on total number of DCs to be opened

Variables

X_{ij}	quantity shipped from plant to DC i to DC j
Y_{jk}	amount of product shipped from DC j tocustomer k
Q_{ijk}	quantity shipped from plant through DC j to customer k

Many researchers attempted to solve the FCTP using metaheuristics such as tabu search or genetic algorithms to solve such hard optimization problems [28,23,7,6]. The use of genetic algorithms (GA) for solving linear and nonlinear transportation problems is well discussed (e.g. [16,28,11,10,7]). Vignaux and Michalewicz [28] introduced the permutation representation and the matrix representation of chromosomes. Another way of representing a chromosome is through the use of Prüfer numbers (see [7]).

Geoffrion and Graves [9] were the first to study the two-stage transportation problem, and they proposed a Bender's decomposition algorithm to solve the multi-commodity distribution problem. Many researchers attempted to solve multi-stage transportation problems using exact algorithms and heuristics (e.g. [19,13,27,24,25]). Syarif and Gen [25] proposed a hybrid genetic algorithm with a chromosome for each stage represented by Prüfer numbers.

Recently, Jawahar and Balaji [15] proposed a genetic algorithm to solve a two-stage transportation problem by considering the per-unit transportation cost, fixed cost associated with each route and uncapacitated DCs or warehouses. This type of two-stage transportation problem is referred to as Scenario-1 problem in our paper. Jawahar and Balaji considered a chromosome to represent a candidate solution from DCs to customers; the other part of the solution (i.e., distribution plan from plants to DCs) is obtained by applying the well-known Least Cost Method (LCM) that uses a cost called equivalent cost, per-unit transportation cost and the fixed cost associated with a route. The GA by Jawahar and Balaji appears to be the best existing method, as of then, for the two-stage transportation problem considering both per-unit transportation cost and fixed cost associated with a route. Later, Balaji and Jawahar [2] developed a simulated annealing (SA) based algorithm to solve Scenario-1 two-stage transportation problems. The performance of the SA is compared with a lower bound as well as with the solution obtained by solving the problem as a linear programming problem that uses the approximated per-unit transportation cost. The authors found that their simulated annealing algorithm produced solutions closer to lower bounds and better than the solutions obtained by solving problems using approximate methods. This SA appears to be the best existing method, up to now, for two-stage distribution problems (i.e., Scenario-1 two-stage transportation problem).

As for the two-stage transportation problem considering the opening cost of a DC, Gen et al. [8] proposed a genetic algorithm using a priority-based encoding procedure to solve the two-stage transportation problem. They dealt with a two-stage transportation problem by considering the per-unit transportation cost, cost of opening a facility and by limiting the number of DCs to be opened. This type of transportation problem is referred to as Scenario-2 problem in our paper. Such two-stage transportation problems are NP-hard [8]. It appears that their GA using the priority-based encoding is the best method to solve the two-stage transportation problem when the opening cost of a facility and per-unit transportation cost are considered.

As an outcome of the literature review, it appears that there is scope to develop efficient metaheuristics to solve the two-stage transportation problems by considering the fixed cost associated with each route or the cost of opening a facility (called Scenario-1 and Scenario-2, respectively, in this paper). In this work an attempt is made to develop a genetic algorithm, and compare the performance of the proposed genetic algorithm with the respective best existing algorithm in two scenarios by solving benchmark problem instances. For this purpose, the GA proposed by Jawahar and Balaji [15] and the SA by Balaji and Jawahar [2] are compared with the proposed GA (called TSGA) to solve the two-stage transportation problem by considering the per unit transportation cost, fixed cost associated with a route and uncapacitated capacity at every DC (Scenario-1). The genetic algorithm designed by Gen et al. [8] is compared with the performance of our proposed GA (TSGA) to solve the two-stage transportation problem by considering the per-unit transportation cost and the opening cost of a DC (Scenario-2). Subsequently we represent the two-stage fixed-charge transportation problem in Scenario-1 as a single-stage FCTP, and solve it using the single-stage genetic algorithm, called SSGA. The proposed and the best existing methods in the respective two scenarios are evaluated using the respective benchmark instances available in the literature.

3. Scenario-1: consideration of per-unit transportation cost and fixed cost along a route

Scenario-1 problem considers an uncapacitated two-stage fixed-charge transportation problem with the consideration of per-unit transportation cost and fixed cost associated with each route between facilities of a supply chain (see [15] for details).

The objective in the two-stage FCTP is to minimize total transportation cost, given by

$$Z = \sum_{i=1}^{l}\sum_{j=1}^{m} C_{ij}X_{ij} + \sum_{i=1}^{l}\sum_{j=1}^{m} F_{ij}\delta_{ij} + \sum_{j=1}^{m}\sum_{k=1}^{n} T_{jk}Y_{jk} + \sum_{j=1}^{m}\sum_{k=1}^{n} G_{jk}\lambda_{jk}, \quad (1)$$

Please cite this article as: Antony Arokia Durai Raj K, Rajendran C. A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem. Computers and Operations Research (2011), doi:10.1016/j.cor.2011.09.020

subject to

$$\sum_{j=1}^{m} X_{ij} \le S_i, \quad i = 1,2,\ldots,l, \tag{2}$$

$$\sum_{j=1}^{m} Y_{jk} = D_k, \quad k = 1,2,\ldots,n, \tag{3}$$

$$X_{ij} \le s_{ij}\delta_{ij} \quad \text{with} \tag{4}$$

$$s_{ij} = \min(S_i, W_j) \quad \forall i \text{ and } \forall j, \quad \text{and}$$

$$\delta_{ij} = \begin{cases} 0 & \text{if } X_{ij} = 0, \\ 1 & \text{otherwise,} \end{cases}$$

$$Y_{jk} \le r_{jk}\lambda_{jk} \quad \text{with} \tag{5}$$

$$r_{jk} = \min(W_j, D_k) \quad \forall j \quad \text{and} \quad \forall k \quad \text{and}$$

$$\lambda_{jk} = \begin{cases} 0 & \text{if } Y_{jk} = 0, \\ 1 & \text{otherwise,} \end{cases}$$

$$\sum_{i=1}^{l} S_i \ge \sum_{k=1}^{n} D_k,$$

$$\sum_{i=1}^{l} X_{ij} = \sum_{k=1}^{n} Y_{jk} \quad \forall j \quad \text{with} \tag{6}$$

$$X_{ij} \ge 0 \quad \text{and} \quad Y_{jk} \ge 0 \text{ (non-negativity constraints).} \tag{7}$$

In Scenario-1 problem, objective function (1) minimizes the total transportation cost involving the fixed costs and the per-unit transportation costs. Constraint (2) ensures that the quantity shipped out from a plant is less than or equal to the available capacity. Constraint (3) ensures that the total shipment received from a distribution center does not exceed the demand at each customer location. Constraint (4) ensures that if $X_{ij} > 0$, then $\delta_{ij} = 1$, and similarly constraint (5) ensures that if $Y_{jk} > 0$, then $\lambda_{jk} = 1$. Constraint (6) corresponds to the flow conservation in the system i.e., the total quantity shipped from plants to distribution centers is equal to the quantity shipped from distribution centers to customers. Constraint (7) ensures the non-negativity of decision variables.

4. Methodology of the proposed two-stage genetic algorithm (TSGA)

The proposed two-stage genetic algorithm comprises the following: a compact form of representing the chromosome, a new heuristic initialization procedure to generate an initial population, a two-phase improvement scheme and a mode of crossover operation. The overall procedure of the proposed two-stage genetic algorithm (TSGA) is presented below. The working mechanism of various steps in the proposed TSGA is presented in the subsequent sections.

Step 1: Generate chromosomes or solutions to initialize the population (see Section 4.2 for details).
Step 2: Evaluate all the chromosomes and choose the best distinct 20 chromosomes.
Step 3: Select 2 chromosomes for crossover, based on fitness values of chromosomes, using the roulette wheel selection (see Section 4.3).
Step 4: Perform the crossover operation to produce 1 offspring (see Section 4.4).
Step 5: Apply the proposed improvement scheme on the offspring (see Section 4.5).
Step 6: After repeating the above steps and thus generating 20 offspring, the best distinct 20 chromosomes among 20 parents and 20 offspring are chosen to survive into the next generation.
Step 7: Terminate the algorithm, when the TSGA has been executed for 10,000 generations or 3600 CPU-time seconds, whichever is earlier.

4.1. Representation of a chromosome

The chromosome is a candidate solution to the problem. The permutation representation and the matrix representation of chromosomes were introduced by Vignaux and Michalewicz [28] for solving the single-stage transportation problem. The matrix representation often results in the offspring having more than $(m+n-1)$ basic cells in the offspring after the crossover operation; in addition the matrix representation needs a string length of $(m \times n)$ to completely represent a solution. Though the permutation representation ensures that there are at most $(m+n-1)$ basic cells in the offspring after crossover operation, it needs a string length of $(m \times n)$ to completely represent a solution. The Prüfer number representation is known to be an efficient way (in terms of the string length required to represent a chromosome being $(m+n-2)$) to represent a chromosome in the transportation problem [7]. However, the use of Prüfer number representation may result in infeasibility after genetic operations and it is found to lack locality and inheritability [12]. The chromosome in priority-based encoding needs a string length of $(m+n)$ to completely represent a solution [8]. This procedure may decode into a solution with an incomplete shipment plan. In this paper we follow permutation representation in a compact form where we retain the basic cells and remove the non-basic cells, and hence we represent a chromosome with a string length of $(m+n-1)$ within the general framework of a single-stage transportation problem. Note that a chromosome in a permutation representation will always yield a unique transportation plan. Moreover, the permutation representation is better than the known representation such as matrix representation, Prüfer number representation and priority based encoding procedure because the permutation representation maintains feasibility even after the application of genetic operators like crossover and mutation.

The two-stage transportation problem involves two stages of shipment, namely, shipments from plants to DCs and shipments from DCs to customers. Let the order of the matrix pertaining to the transportation between DCs to customers be $(m \times n)$ and that from plants to DCs be $(l \times m)$, with the overall problem size indicated by $(l \times m \times n)$. Therefore we represent a chromosome in the form of two sub-chromosomes: S-C 1 represents a solution with respect to shipments from DCs to customers, and S-C 2 represents a solution with respect to shipments from plants to DCs. These two sub-chromosomes constitute the complete chromosome for the overall problem. As an example consider a $(3 \times 3 \times 4)$ two-stage transportation problem given in Tables 1a and 1b, and the transportation plan given in Table 2c: we have S-C 1 with basic cells (with respect to shipments between DC j and customer k) as {5–6–4–11} and thereafter we have S-C 2 with basic cells (with respect to shipments between plant i and DC j) as {1–2–8–5–6}. Note that in Tables 1a and 1b the capacity that is indicated at each distribution center is the available capacity and it is infinite, whereas in Table 2c the capacity that is indicated at each distribution center is finite and it corresponds to the utilized capacity. See Tables 2a and 2b to know about the transformation from the available infinite capacity to the utilized capacity at each distribution center, as indicated finally in Table 2c. In this paper, an element $(i, j)/(j, k)$ in a transport plan corresponds to the element $((i-1) \times m+j)/((j-1) \times n+k)$ in the permutation

Please cite this article as: Antony Arokia Durai Raj K, Rajendran C. A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem. Computers and Operations Research (2011), doi:10.1016/j.cor.2011.09.020

Table 1a
Two-stage transportation matrix with fixed costs shown inside the matrix.

		Distribution Center (DC) j			
		DC_1	DC_2	DC_3	Supply
Plant i	P_1	400	7000	5500	500
	P_2	1700	8000	3500	400
	P_3	800	7700	6000	300
	Available capacity	∞	∞	∞	

		Customer (Cu) k				
		Cu_1	Cu_2	Cu_3	Cu_4	Available capacity
Distribution Center j	DC_1	2300	5000	6000	7500	∞
	DC_2	3500	6000	8500	9000	∞
	DC_3	7000	4500	200	500	∞
	Demand	250	350	50	350	

Table 1b
Two-stage transportation matrix with variable costs shown inside the matrix.

		Distribution Center (DC) j			
		DC_1	DC_2	DC_3	Supply
Plant i	P_1	17	5	9	500
	P_2	25	60	10	400
	P_3	50	20	50	300
	Available capacity	∞	∞	∞	

		Customer (Cu) k				
		Cu_1	Cu_2	Cu_3	Cu_4	Available capacity
Distribution Center j	DC_1	18	25	60	10	∞
	DC_2	5	4	50	20	∞
	DC_3	15	24	80	90	∞
	Demand	250	350	50	350	

Table 2a
LCM with the use of $(C'_{jk})^{dc-cu}_{static}$: partial chromosome or partial solution is {5}.

		Customer (Cu) k				Available capacity	Utilized capacity
		Cu_1	Cu_2	Cu_3	Cu_4		
Distribution Center (DC) j	DC_1	27.2	39.3	180.0	31.4	∞	0
	DC_2	**19.0**	21.1	220.0	45.7	∞	250
	DC_3	43.0	36.9	84.0	91.4	∞	0
	Demand	~~250~~ 0	350	50	350		

Table 2b
LCM with the use of $(C'_{ij})^{p-dc}_{static}$: partial chromosome or partial solution is {1}.

		Distribution Center (DC) j			Supply	
		DC_1	DC_2	DC_3		
Plant i	P_1	**18.14**	19.00	119.00	~~500~~ 150	
	P_2	29.86	80.00	80.00	400	
	P_3	52.67	45.67	170.00	300	
	Capacity	~~350~~ 0	600	50		

Table 2c
Final allocation of cells (shipment plan) and the corresponding chromosome (in permutation representation).

S-C 2:
{1 – 2 – 8 – 5 – 6}

		Distribution Center (DC) j			
		DC_1	DC_2	DC_3	
Plant i	P_1	350	150		500
	P_2		150	50	400
	P_3		300		300
		350	600	50	

S-C 1:
{5 – 6 – 4 – 11}

		Customer (Cu) k				
		Cu_1	Cu_2	Cu_3	Cu_4	W_j^{act}
Distribution Center (DC) j	DC_1				350	350
	DC_2	250	350			600
	DC_3			50		50
		250	350	50	350	

Please cite this article as: Antony Arokia Durai Raj K, Rajendran C. A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem. Computers and Operations Research (2011), doi:10.1016/j.cor.2011.09.020

representation, respectively, for the transportation plan between plants and DCs/DCs to customers, and a cell enters into a permutation representation of a transport plan only when its X_{ij} or Y_{jk} equals the minimum of left-over supply and the left-over demand; in addition, a cell is called a basic cell only when $X_{ij}>0$ or $Y_{jk}>0$. See Appendix A4 for details of the procedure *MATRIX_PERM* that converts a transport plan in a matrix form to a compressed permutation form. Following this procedure, the resulting chromosome in compressed form will have S-C 1 with basic cells (with respect to shipments between DC j and customer k) as {4–5–6–11} and thereafter S-C 2 with basic cells (with respect to shipments between plant i and DC j) as {1–2–6–8–5}. Note that this procedure holds for both Scenarios 1 and 2 in terms of S-C 1 and S-C 2. Also note that the transport plan remains the same before and after the application of *MATRIX_PERM* procedure.

4.2. Initialization of population

The initial population is generated with the consideration of approximated costs (see Sections 4.2.1 and 4.2.2) in the place of actual fixed costs and actual variable transportation costs. We generate a total of 108 chromosomes: 2 chromosomes through the use of $(C'_{ij})^{p-dc}_{static}$ and $(C'_{jk})^{dc-cu}_{static}$ in the Least Cost Method (LCM) and Vogel's Approximation Method (VAM); 2 chromosomes through the use of $(C''_{ij})^{p-dc}_{static}$ and $(C''_{jk})^{dc-cu}_{static}$ in the LCM and VAM; 2 chromosomes through the use of $(C'_{ij})^{p-dc}_{dynamic}$ and $(C'_{jk})^{dc-cu}_{dynamic}$ in the Least Cost Method (LCM) and Vogel's Approximation Method (VAM); 2 chromosomes through the use of $(C''_{ij})^{p-dc}_{dynamic}$ and $(C''_{jk})^{dc-cu}_{dynamic}$ in the LCM and VAM; 50 chromosomes through the use of $(C'_{ij})^{p-dc}_{dynamic}$ and $(C'_{jk})^{dc-cu}_{dynamic}$ with a probabilistic choice between the Least Cost Method (LCM) and Vogel's Approximation Method (VAM); 50 chromosomes through the use of $(C''_{ij})^{p-dc}_{dynamic}$ and $(C''_{jk})^{dc-cu}_{dynamic}$ with a probabilistic choice between the LCM and VAM. We now describe below the generation of these initial chromosomes.

4.2.1. Generation of chromosomes using the static information of transportation

Let us define

$$(C'_{ij})^{p-dc}_{static}=\frac{F_{ij}}{\min(S_i,W_j)}+C_{ij}$$
(with respect to shipment from plant i to DC j), (8)

$$(C'_{jk})^{dc-cu}_{static}=\frac{G_{jk}}{\min(W_j,D_k)}+T_{jk}$$
(with respect to shipment from DC j to customer k) (9)

$$(C''_{ij})^{p-dc}_{static}=F_{ij}+C_{ij}\min(S_i,W_j)$$
(with respect to shipment from plant i to DC j), and (10)

$$(C''_{jk})^{dc-cu}_{static}=G_{jk}+T_{jk}\min(W_j,D_k)$$
(with respect to shipment from DC j to customer k). (11)

We consider $(C'_{ij})^{p-dc}_{static}$, $(C'_{jk})^{dc-cu}_{static}$, $(C''_{ij})^{p-dc}_{static}$, and $(C''_{jk})^{dc-cu}_{static}$ to be static in the sense that when plant i/DC j actually ships out/receives X_{ij}, and DC j/customer k correspondingly ships out/receives Y_{jk}, we do not use the left-over supply at plant i, left-over capacity at DC$_j$ and unsatisfied demand of customer k, in Eqs. (8)–(11). Instead, we replace the given C_{ij} and F_{ij} by $(C'_{ij})^{p-dc}_{static}$, and T_{jk} and G_{jk} by $(C'_{jk})^{dc-cu}_{static}$ in the corresponding cell of the two-stage transportation problem and use the well-known LCM for solving the resultant linear transportation problem. Hence, in the process, we do not compute $(C'_{ij})^{p-dc}_{static}$ and $(C'_{jk})^{dc-cu}_{static}$ by considering the actual left-over supply at plant i, left-over capacity at DC_j and unsatisfied demand of customer k, after every shipment. Similarly, we replace the given C_{ij} and F_{ij} by $(C'_{ij})^{p-dc}_{static}$, and T_{jk} and G_{jk} by $(C'_{jk})^{dc-cu}_{static}$ in every cell, and solve the resultant problem using the well-known VAM. These approaches are, respectively, called as LCM-$(C'_{ijk})^{p-dc-cu}_{static}$ and VAM-$(C'_{ijk})^{p-dc-cu}_{static}$ in this paper. See Taha [26] for the implementation of the LCM and VAM. Similar approaches are followed with the consideration of $(C''_{ij})^{p-dc}_{static}$ and $(C''_{jk})^{dc-cu}_{static}$, and the approaches are, respectively, called as LCM-$(C''_{ijk})^{p-dc-cu}_{static}$ and VAM-$(C''_{ijk})^{p-dc-cu}_{static}$ for the sake of simplicity and completeness. Of course, the final computation is done with the consideration of C_{ij}, F_{ij}, T_{jk}, and G_{jk} corresponding to the final shipments quantity in each cell (i.e., considering actual X_{ij} and Y_{jk}, and hence computing the total cost reckoning with fixed costs and variable costs).

As an example, consider the build-up of a chromosome (in the permutation or string-type representation) obtained by the use of LCM-$(C'_{jk})^{dc-cu}_{static}$ and $(C'_{ij})^{p-dc}_{static}$. See Tables 2a and 2c for the problem presented in Tables 1a and 1b. Consider the right-hand side of Table 2a, where the matrix with element $(C'_{jk})^{dc-cu}_{static}$ representing the approximated cost of transportation from DCs to customers. Note that DCs have unlimited capacities in Scenario-1 and hence we consider two types of capacities at DCs, namely, available capacity (initially assumed to uncapacitated) and the utilized capacity (i.e., actual storage utilized in a DC). Apply the LCM and allocate the maximum possible feasible shipment from the corresponding DC to a customer; update the utilized capacity and left-over demand; and then taboo the column/row from further consideration, if the demand is zero (see Table 2a).

Now let us consider the utilized capacity at DCs as the demand with respect to the distribution from plants to DCs and apply the LCM using $(C'_{ij})^{p-dc}_{static}$. Note that we need to consider the utilized capacities of DCs to calculate the approximated cost matrix using $(C'_{ij})^{p-dc}_{static}$ with respect to shipment from plants to DCs. Accordingly, the updated cost matrix is given in Table 2b. Apply the LCM and allocate the maximum possible feasible shipment from a plant to a DC; update correspondingly the left-over supply at the plant and the left-over capacity at the DC; and taboo the column/row from further consideration, if the left-over supply is zero. Proceeding likewise, the resulting final shipments are given in Table 2c.

4.2.2. Generation of chromosomes using the dynamic information of transportation

As opposed to the static computation of $(C'_{ij})^{p-dc}_{static}$, $(C''_{ij})^{p-dc}_{static}$, $(C'_{jk})^{dc-cu}_{static}$, and $(C''_{jk})^{dc-cu}_{static}$, we have also thought of $(C'_{ij})^{p-dc}_{dynamic}$, $(C''_{ij})^{p-dc}_{dynamic}$, $(C'_{jk})^{dc-cu}_{dynamic}$ and $(C''_{jk})^{dc-cu}_{dynamic}$. These computations are carried out with S'_i, W'_i, and D'_k denoting, respectively, the left-over supply at plant i, left-over capacity at DC_j and unsatisfied demand of customer k, as and when the shipment in the corresponding row or column takes place. Similar to the use of $(C'_{ij})^{p-dc}_{static}$ and $(C'_{jk})^{dc-cu}_{static}$, we use $(C'_{ij})^{p-dc}_{dynamic}$ and $(C'_{jk})^{dc-cu}_{dynamic}$ to obtain a solution to the two-stage transportation problem. Similarly we use $(C''_{ij})^{p-dc}_{dynamic}$ and $(C''_{jk})^{dc-cu}_{dynamic}$ to obtain a solution to the two-stage transportation problem. Note that we follow the permutation representation of chromosomes so that we have at most $(m+n-1)$ genes in S-C 1 and at most $(l+m-1)$ genes in S-C 2 in every chromosome. On the other hand, the matrix representation of chromosome is cumbersome (in view of the entire matrix of shipments being stored), and this representation may result in infeasibility when the crossover of two matrix solutions or chromosomes is done. We replace the terms $(C'_{ij})^{p-dc}$ and

Please cite this article as: Antony Arokia Durai Raj K, Rajendran C. A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem. Computers and Operations Research (2011), doi:10.1016/j.cor.2011.09.020

$(C'_{jk})^{dc-cu}$ by a single term $(C'_{ijk})^{p-dc-cu}$ for the sake of simplicity of presentation of expressions.

4.2.3. Generation of chromosomes with a probabilistic choice between LCM and VAM

Notations

NPD number of genes or basic cells, with $X_{ij} > 0$, already present in the chromosome.

NDC number of genes or basic cells, with $Y_{jk} > 0$, already present in the chromosome.

In addition to the 8 solutions generated through the use of LCM-$(C'_{ijk})^{p-dc-cu}_{static}$, LCM-$(C'_{ijk})^{p-dc-cu}_{dynamic}$, VAM-$(C'_{ijk})^{p-dc-cu}_{static}$, VAM-$(C'_{ijk})^{p-dc-cu}_{dynamic}$, LCM-$(C''_{ijk})^{p-dc-cu}_{static}$, LCM-$(C''_{ijk})^{p-dc-cu}_{dynamic}$, VAM-$(C''_{ijk})^{p-dc-cu}_{static}$, and VAM-$(C''_{ijk})^{p-dc-cu}_{dynamic}$, we also generate 50 solutions through probabilistic choice between the LCM and VAM to select a cell for shipment, coupled with the consideration of $(C'_{ij})^{p-dc}_{dynamic}$ and, $(C'_{jk})^{dc-cu}_{dynamic}$ respectively. Note that in the transportation problem, we first determine the cell that is to be chosen for shipment, and then determine how much of actual shipment is feasible with respect to that cell, depending upon the left-over supply and the left-over demand. The second part of this determination is related to feasibility; and the first part (i.e., choice of a cell) is done depending upon the criterion or method (either the LCM or VAM). In this part of the paper, we consider the LCM with a probability of 0.5 and the VAM with a probability of 0.5, and hence choose a cell accordingly. It means that the final solution consists of some cells (and the corresponding shipments) selected by the LCM and the remaining cells selected by VAM. This entire process is repeated 50 times to yield 50 solutions. Similarly, we generate 50 solutions through the probabilistic choice between LCM and VAM, coupled with the consideration of $(C''_{ij})^{p-dc}_{dynamic}$ and $(C''_{jk})^{dc-cu}_{dynamic}$, respectively. A total of 108 chromosomes are thus obtained and evaluated. The best 20 chromosomes are selected. Every such chromosome in the matrix plan is converted to a permutation form in terms of S-C 1 and S-C 2, with each sub-chromosome consisting of basic cells only.

We now do the following steps to expand every chromosome to a complete string with S-C 2 now consisting of $(l \times m)$ genes and S-C 1 now consisting of $(m \times n)$ genes. This expansion is necessary to perform the crossover operation presented in Section 4.4. Because of the string length being long and the associated computational effort and memory requirements for storing such strings, we set the initial population size to 20:

Step 1: Every chromosome is evaluated with respect to the total cost of transportation, (as $\{X_{ij}\}$ and $\{Y_{jk}\}$ are known for every chromosome), consisting of the sum of variable costs and fixed costs.

Step 2: We also form the solution (or chromosome) in the permutation representation for both S-C 1 and S-C 2 using *MATRIX_PERM*. See Appendix A4 for details.

Step 3: Now, we expand the chromosome in permutation representation so as to have $(l \times m)$ genes in S-C 2 and $(m \times n)$ genes in S-C 1. This means that we need to add $((l \times m)$–*NPD*$)$ cells to S-C 2 and $((m \times n)$–*NDC*$)$ cells to the S-C 1. For this purpose, sort the remaining $((l \times m)$–*NPD*$)$ cells in increasing order of $(C''_{ij})^{p-dc}_{static}$ and $((m \times n)$–*NDC*$)$ cells in increasing order of $(C''_{jk})^{dc-cu}_{static}$, and append this ordered set to the existing chromosome that has been formed in Step 2.

Step 4: S-C 2 and S-C 1 thus obtained have $(l \times m)$ and $(m \times n)$ genes, respectively. Denote such an expanded S-C 1 and S-C 2 with $(m \times n)$ and $(l \times m)$ genes, in general, by

$\{\text{CHR}\} = \{\{\text{chr}_2(p),\ p = 1,2,\ldots,(l \times m)\};$

$\{\text{chr}_1(p),\ p = 1,2,\ldots,(m \times n)\}\}$ or simply

$\{\text{CHR}\} = \{\{\text{chr}_2(p)\};\quad \{\text{chr}_1(p)\}\}.$

4.3. Fitness function

The objective function is to minimize the total transportation costs. The fitness level of each chromosome determines the probability of selection of the chromosomes. The parents (chromosomes) are selected for crossover (using the roulette-wheel selection) using the following fitness function:

$$F_k = \frac{1}{Z_k} \quad \text{where } k = 1,2,\ldots,N \text{ with } N \text{ denoting the population size.} \tag{12}$$

4.4. Crossover

The crossover operator is developed with the premise that offspring reproduced will be good, if the offspring inherits more genes (or characteristics) from the better parent. Hence the proposed crossover operator is designed to retain more gene characteristics from the better parent than from the other parent. The proposed crossover operator yields only one offspring after crossover operation. After having chosen two chromosomes for crossover, we term the better chromosome as the base chromosome, and perform an operation on this chosen chromosome with the objective of inheriting a fraction of the characteristics of the other chromosome called the donor chromosome. We now sample a random number u between [0, 1] corresponding to every gene in the base chromosome. If $u \le 0.95$, then the corresponding gene is inherited from the base chromosome into the offspring. Now the offspring thus obtained is filled (with respect to the unfilled genes) from the donor chromosome using the order-based crossover operator. The pseudo-code for the crossover operation is given below:

Step 1: Select the better parent among the two selected parents; set $p = 1$.

Step 2: For $p = 1$ to $(l \times m)$ (in the case of SC_2^*)/$(m \times n)$ (in the case of SC_1^*), do the following:

```
{
   generate a uniform random number u;
   if u ≤ 0.95
   then
      inherit the element in position p from SC1*/SC2* to
      the corresponding position in the offspring,
} /*end of 'for p=1 to (l × m)/m × n' */
```

Step 3: Inherit the unfilled cells from the donor parent one after other (sequentially) to the unfilled genes in the offspring.

We now explain the proposed crossover operation. Using the roulette-wheel selection, two distinct chromosomes are selected. Let Z_1 and Z_2 denote the total transportation cost (i.e., objective function value) for these two parent chromosomes 1 and 2, respectively. Considering the parent chromosomes one by one, we expand its first sub-chromosome S-C1 to a string length of $(m \times n)$, and its second sub-chromosome S-C 2 to a string length of $(l \times m)$. Considering the first sub-chromosome, namely, S-C1, of the two

Please cite this article as: Antony Arokia Durai Raj K, Rajendran C. A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem. Computers and Operations Research (2011), doi:10.1016/j.cor.2011.09.020

parent chromosomes, choose the better sub-chromosome, and call it as SC_1^*, and similarly, obtain SC_2^* by considering the second sub-chromosome of the two parent chromosomes. For example, let the transportation cost with respect to S-C 2 and S-C 1 for parent 1 be 28,200 monetary units (MU) and 33,350 MU, respectively; and the transportation cost of S-C 2 and S-C 1 for parent 2 be 48,800 MU and 27,350 MU, respectively. Hence, we select the expanded S-C 2 from parent 1 and the expanded S-C 1 from parent 2, and call them, respectively, SC_2^* and SC_1^*. In our TSGA, genes get inherited with the probability of 0.95 from SC_1^* and SC_2^*, and the remaining genes are filled from the other respective sub-chromosomes. The crossover operation is explained below.

(obtained from the crossover operation and then subjected to the improvement scheme proposed in this paper) are thus generated. Note that each offspring produced from parents is subjected to an improvement scheme (see Section 4.5).

It is worth mentioning that mutation operation is not invoked in TSGA in the conventional sense; in the proposed crossover operator the offspring inherits 95% of gene characteristics from the base parent chromosome, while inheriting the remaining 5% of its gene characteristics from the donor chromosome. These characteristics of the proposed TSGA, we believe, contribute to the effectiveness of proposed TSGA. In fact, computation results later show the proposed TSGA performs quite well in comparison

For parent 1, let assume that $Z_1=61{,}550$, Z(S-C 2)=28,200 and Z(S-C 1)=33,350. Let also assume that S-C 2 and S-C 1 are as follows:

S-C 2								
2	1	6	4	7	3	8	9	5

S-C 1											
11	6	9	4	5	1	3	7	10	2	8	12

For parent 2, let assume that $Z_2=76{,}150$, Z(S-C 2)=48,800 and Z(S-C 1)=27,350. Let also assume that S-C 2 and S-C 1 are as follows:

S-C 2								
6	1	2	8	5	3	9	4	7

S-C 1											
11	5	6	4	1	3	9	7	10	2	8	12

We therefore define SC_2^* from Parent 1 and SC_1^* from Parent 2 as follows:

SC_2^*								
2	1	6	4	7	3	8	9	5

SC_1^*											
11	5	6	4	1	3	9	7	10	2	8	12

First let us consider SC_2^*, sample 9 uniform random numbers (corresponding to 9 genes in SC_2^*). Let the sampled uniform random numbers be {0.2, 0.6, 0.56, 0.42, 0.89, 0.96, 0.14, 0.59, 0.965}; we retain the genes corresponding to those uniform random numbers less than or equal to 0.95 (this setting has been selected after pilot runs), as given below:

0.2	0.6	0.56	0.42	0.89	0.96	0.14	0.59	0.965

2	1	6	4	7	3	8	9	5

2	1	6	4	7		8	9	

We now inherit the unassigned cells 5 and 3 (in the same order as found in S-C 2 of Parent 2) into the offspring (this operation is similar to the conventional order-based crossover operation). We now have the offspring with its resultant S-C 2: {2–1–6–4–7–5–8–9–3}.

6	1	2	8	5	3	9	4	7

2	1	6	4	7	5	8	9	3

Similarly, repeat the above procedure by considering SC_1^*, and let the offspring with its resultant S-C 1 such as the one given below:

11	4	6	5	1	3	9	7	10	2	8	12

We now have the full offspring with its resultant S-C 2 and S-C 1, respectively, as follows:

S-C 2								
2	1	6	4	7	5	8	9	3

S-C 1											
11	4	6	5	1	3	9	7	10	2	8	12

We allocate the shipments by scanning offspring's S-C 1 (invoke *PERM_MATRIX* 1 procedure—see Appendix A1 for details) from the left to right; we then allocate the shipments by scanning offspring's S-C 2 from the left to right (invoke *PERM_MATRIX* 2 procedure—see Appendix A2 for details), with the offspring's S-C 1 reduced to {11–4–6–5} and thereafter the offspring's S-C 2 reduced to {2–6–4–8} (see Table 2d). Twenty offspring

to the best existing methods with respect to most benchmark problem instances.

4.5. Improvement scheme

The proposed improvement scheme attempts to unearth better solutions present in the neighborhood of the given

Please cite this article as: Antony Arokia Durai Raj K, Rajendran C. A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem. Computers and Operations Research (2011), doi:10.1016/j.cor.2011.09.020

Table 2d
Final shipment plan for the offspring chromosome (in permutation representation).

S-C 2:
{2 – 1 – 6 – 4 – 7 – 5 – 8 – 9 – 3}

Plant i \ Distribution Center (DC) j	DC_1	DC_2	DC_3	
P_1		500		500
P_2	350		50	400
P_3		100		300
	350	600	50	

S-C 1:
{11 – 4 – 6 – 5 – 1 – 3 – 9 – 7 – 10 – 2 – 8 – 12}

Distribution Center (DC) j \ Customer (Cu) k	Cu_1	Cu_2	Cu_3	Cu_4	
DC_1				350	350
DC_2	250	350			600
DC_3			50		50
	250	350	50	350	

solution. Every chromosome (with its S-C 1 of full string length $(m \times n)$ and S-C 2 of full string length $(l \times m)$) is subjected to the proposed improvement scheme that works in two phases, Phase I and Phase II. In Phase I the solution is improved by allocating the maximum permissible shipment quantity to each route and in Phase II the shipment quantity to be allocated is determined by adapting the stepping stone method (SSM)—see Taha [26] for details of the SSM. We first apply Phase I of the improvement scheme on S-C 1. The matrix solution (i.e., shipment plan from DCs to customers) obtained after Phase I is converted into permutation representation using *MATRIX_PERM* (see Appendix A4 for details). The resulting S-C 1 in permutation representation is subjected to Phase II of the improvement scheme. The matrix solution (i.e., shipment plan from DCs to customers) obtained after Phase II is converted into permutation representation using *MATRIX_PERM*. The utilized capacity with respect to the resulting S-C 1 (in terms of shipment from DCs) serves as the demand with respect to S-C 2 (in terms of shipment into DCs). We now invoke *PERM_MATRIX* 1 to determine W_j^{act} and then apply Phases I and II of the improvement scheme on S-C 2.

We now present the working mechanism of Phases I and II.

4.5.1. Formal procedure for phase I (PHASE 1):

In *PHASE* I we improve the seed solution by allocating the maximum possible feasible shipment to the current basic variable as well as to the non-basic variable. The formal procedure with respect to S-C 1 is now presented. Set *SS*= S-C 1. Let the shipment be denoted in general by Y_{jk}.

Step 0: Initialize *LBS*=*SS*; $Z_{LBS}=Z_{SS}$; and $p'=1$.
Step 1: Set $W'_j = W_j\ \forall j$ and $D'_k = D_k\ \forall k$; consider the cell corresponding to p' in $\{\text{chr}_1\ (p)\}$ in *SS*; let this cell be (j',k');
if $(Y_{j'k'} \neq \min\{W'_{j'}, D'_{k'}\})$
then
increase the shipment quantity of $Y_{j'k'}$ to y (where $y = \min\{W'_{j'}, D'_{k'}\}$) and update the corresponding left-over supply and demand, respectively; proceed to Step 2
else
increment p' by 1 and revert to the beginning of this step.
Step 2: Scan *SS* from left to right (excluding gene p' in $\{\text{chr}_1\ (p)\}$), and allocate every cell with the maximum possible feasible shipment considering the left-over supply and the left-over demand (with respect to *SS*); call the resultant solution *CS*.
Step 3: Evaluate *CS*, and call the objective function value Z_{CS}.
Step 4: If $(Z_{CS} < Z_{LBS})$
then
update *LBS*=*CS*; $Z_{LBS}=Z_{CS}$; and set $p'=p'+1$
else
set $p'=p'+1$.
Step 5: if ($p' \leq$ the number of cells as specified by the algorithm—see Section 5)
then
go back to Step 1
else
proceed to Step 6.
Step 6: If $(Z_{LBS} < Z_{SS})$
then
set *SS*=LBS and $Z_{SS}=Z_{LBS}$; and $p'=1$; revert to Step 1
else
seed solution *SS* and Z_{SS} remain the same, and we proceed to Step 7.
Step 7: Terminate the procedure; set $W_j^{act} = \sum_{k=1}^{n} Y_{jk}$.

As for the formal procedure of *PHASE* 1 with respect to S-C 2, the above procedure remains the same except for the following changes in Steps 1 and 2, and Step 7 denoting only the procedure termination. Set *SS*=S-C 2. Let the shipment in general be denoted by X_{ij}.

Step 1: Set $S'_i = S_i\ \forall i$ and $W_j^{act'} = W_j^{act}, \forall j$; consider the cell corresponding to p' in $\{\text{chr}_2\ (p)\}$ in *SS*; let this cell be (i',j');
if $(X_{i'j'} \neq \min\{S'_{i'}, W_{j'}^{act'}\})$
then
increase the shipment quantity of $X_{i'j'}$ to y (where $y = \min\{S'_{i'}, W_{j'}^{act'}\}$) and update the corresponding left-over supply and demand (with respect to *SS*), respectively; proceed to Step 2
else
increment p' by 1 and revert to the beginning of this step.
Step 2: Scan *SS* from the left to right (excluding gene p' in $\{\text{chr}_2\ (p)\}$), and allocate every cell with the maximum possible feasible shipment; call the resultant solution *CS*.

4.5.2. Formal procedure for phase II (PHASE 2):

The objective of *PHASE* 2 is to see if a non-basic cell can be made a basic cell and to see if such an introduction leads to an improved solution. The formal procedure pertaining to Phase II with respect to S-C 1 is now presented. Set *SS*= S-C 1. Let the shipment quantities be denoted in general by Y_{jk}:

Step 0: Invoke *PERM_MATRIX* 1 on S-C 1 obtained from Phase I to get the solution in the matrix form.
Initialize *LBS* =*SS* and $Z_{LBS}=Z_{SS}$.
Step 1: Set $W'_j = W_j\ \forall j$; and $D'_k = D_k\ \forall k$; let p' be the position of first non-basic cell (note that in this paper, we

use the term 'non-basic cell' to refer to a cell for which $Y_{jk}=0$); consider the cell corresponding to p' in {chr$_1$ (p')} in *SS* and let this cell be (j',k').

Step 2: Apply the stepping-stone method to determine the shipment quantity to be allocated as follows. Increment the chosen non-basic cell by 1 and scan S-C 1 from the left to right to find out the set of basic cells whose shipment consequently decreases by 1. Identify the least shipment in *SS* among such basic cells. Set this shipment quantity as the shipment for the chosen non-basic cell. Let the shipment quantity be y'; thereafter scan *SS* from the left to right (excluding gene p' in {chr$_1$ (p')}), and allocate every cell with the maximum possible feasible shipment; call the resultant solution *CS*.

Step 3: Evaluate *CS*; and call the objective function value Z_{CS}.

Step 4: If ($Z_{CS} < Z_{LBS}$)
then
update $LBS=CS$; $Z_{LBS}=Z_{CS}$; and set $p'=p'+1$
else
set $p'=p'+1$.

Step 5: If ($p' \leq$ number of cells as specified by the algorithm—see Section 5)
then
go back to Step 1
else
proceed to Step 6.

Step 6: If ($Z_{LBS} < Z_{SS}$)
then
set $SS=LBS$ and $Z_{SS}=Z_{LBS}$; set p' to the first non-basic gene position in *SS* and revert to Step 1
else
seed solution *SS* and Z_{SS} remain the same and we proceed to Step 7.

Step 7: Terminate the procedure; set $W_j^{act}=\sum_{k=1}^{n} Y_{jk}$.

As for the formal procedure of *PHASE* 2 with respect to S-C 2, the above procedure remains the same except for the following changes in Steps 0–2 as given below, with Step 7 denoting the termination condition only. Set *SS*=S-C 2. Let the shipment quantities in general be denoted by X_{ij}.

Step 0: Invoke *PERM_MATRIX* 2 on S-C 2 obtained from Phase I to get the solution in the matrix form. Initialize $LBS=SS$ and $Z_{LBS}=Z_{SS}$.

Step 1: Set $S_i'=S_i$ $\forall i$, and $W_j^{act\prime}=W_j^{act}$ $\forall j$; let p' be the position of first non-basic cell (note that in this paper, we use the term 'non-basic cell' to refer to any cell for which $X_{ij}=0$); consider the cell corresponding to p' in {chr$_2$ (p')} in *SS* and let this cell be (i',j').

Step 2: Apply the stepping-stone method and determine the shipment quantity to be allocated to the chosen cell. Let the shipment quantity be y'; scan *SS* from the left to right (excluding gene p' in {chr$_2$ (p')}), and allocate every cell with the maximum possible feasible shipment; call the resultant solution *CS*.

4.6. Survival

The best distinct 20 chromosomes among parents and offspring (obtained after improvement scheme) are selected for survival into the next generation.

4.7. Termination condition

Terminate the algorithm, when the TSGA has been executed for 10,000 generations or 3600 CPU-time seconds, whichever is earlier (in the case of Scenario-1).

5. Results and discussion for Scenario-1

The proposed TSGA is coded in C language, and executed on a personal computer. The processor used is an Intel Pentium-IV processor with a speed of 3 GHz and 1 GB RAM. The problem instances are solved by the proposed genetic algorithm. The GA is run with the following settings: improvement scheme, with the numbers of cells considered for improvement up to the {($m+n-1$)}th position in S-C 1, and {($l+m-1$)}th position in S-C 2 in a chromosome in order to restrict the computational effort in solving the two-stage transportation problem. Pilot runs have been carried out to arrive at the best parameter setting for genetic algorithms proposed in this study and the details of the pilot runs are presented in Appendix A5.

In order to check for the robustness of the proposed TSGA in terms of its consistent and good performance, more than one run is to be conducted. The method of antithetic (i.e., negatively correlated) sampling is a commonly used procedure for generating a negatively correlated pair of samples [4]. This is accomplished using random numbers that are antithetic to the original stream of random numbers. Such an approach leads to a good estimation of performance of the TSGA. The TSGA is run with a random number stream {u} (i.e., uniform random numbers) and the next run is done using {$1-u$} (i.e., antithetic random numbers). The former run is termed TSGA-RN and the later run is termed TSGA-ARN.

In order to evaluate the performance of the proposed genetic algorithm, a *t*-test for the significance of pairwise difference has been conducted with respect to the solutions obtained from runs TSGA-RN and TSGA-ARN with respect to Scenario-1 (see Table 3: columns 5 and 6). The null hypothesis states that there is no significant paired difference between the performance yielded by the execution of TSGA with the use of random numbers and those by TSGA with use of the corresponding antithetic random numbers. The paired *t* test has been conducted using SPSS-12 (with $\alpha=0.05$), showing no significant difference between solutions obtained by TSGA, with the use of a uniform random number stream {u} and with the use of the corresponding antithetic random number stream {$1-u$} (i.e., TSGA-RN and TSGA-ARN), and thereby showing the robustness and consistency of the performance of the proposed TSGA with respect to results indicated in Table 3.

The results by the proposed TSGA, the best existing algorithm and a lower bound on the objective function value (Z) for the benchmark problem instances are presented in Table 3. The best solution among existing algorithms (GA proposed by Jawahar and Balaji [15] and the SA by Balaji and Jawahar [2]) for every problem instance is considered for comparing the performance of the proposed TSGA. The lower bound values are obtained by solving the problem instances as a linear programming problem by treating (relaxing) the binary variables (δ_{ij} and λ_{jk}—see Section 3 for their details) as real variables bounded by 0 and 1, and using CPLEX. It is seen that the proposed genetic algorithm performs quite well in most problem instances. The computational effort required to solve the problem instances were not reported in the respective papers by Jawahar and Balaji [15], and Balaji and Jawahar [2]. For every problem instance, the number of solutions enumerated by the SA by Balaji and Jawahar [2] is computed based on parameters defined by Balaji and Jawahar [2], and

Please cite this article as: Antony Arokia Durai Raj K, Rajendran C. A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem. Computers and Operations Research (2011), doi:10.1016/j.cor.2011.09.020

Table 3
Performance evaluation of the proposed TSGA and the existing methods for Scenario-1.

Problem instance	Lower bound on Z (obtained by relaxation of binary variables)	Best solution[a]	Number of solutions enumerated by the SA by Balaji and Jawahar [2]	Proposed TSGA		Number of solutions enumerated by TSGA to get the best solution
				TSGA-RN	TSGA-ARN	
$2\times2\times3$	112,600.00	***112,600***	1444	***112,600***	***112,600***	248
$2\times2\times4$	237,750.00	***237,750***	1924	***237,750***	***237,750***	464
$2\times2\times5$	173,878.57	***180,450***	2404	***180,450***	***180,450***	590
$2\times2\times6$	157,050.00	***165,650***	2884	***165,650***	***165,650***	617
$2\times2\times7$	161,711.43	***162,490***	3364	***162,490***	162,850	679
$2\times3\times3$	56,020.00	***59,500***	2164	***59,500***	***59,500***	598
$2\times3\times4$	31,522.70	***32,150***	2884	***32,150***	***32,150***	678
$2\times3\times6$	64,724.17	69,970	4324	***67,380***	***67,380***	816
$2\times3\times8$	253,419.29	263,000	5764	***258,730***	***258,730***	10,857
$2\times4\times8$	74,737.50	***80,400***	7684	84,600	84,600	1509
$2\times5\times6$	73,195.00	94,565	7204	***80,865***	***80,865***	1249
$3\times2\times4$	45,470.00	***47,140***	2884	***47,140***	***47,140***	525
$3\times2\times5$	169,825.00	***178,950***	3604	***178,950***	***178,950***	604
$3\times3\times4$	52,591.67	***57,100***	4324	61,000	61,000	831
$3\times3\times5$	151,394.29	***152,800***	5404	156,900	171,760	1047
$3\times3\times6$	132,890.00	***132,890***	6484	***132,890***	***132,890***	848
$3\times3\times7$ (a)	93,111.09	***104,115***	7564	106,745	106,745	1072
$3\times3\times7$ (b)	279,514.41	***287,360***	7564	295,060	295,060	1188
$3\times4\times6$	71,462.50	***77,250***	8644	81,700	81,700	1217
$4\times3\times5$	118,450.00	***118,450***	7204	***118,450***	***118,450***	1014

A solution in bold and italics indicates the best solution with respect to that problem instance.
The problem instances were taken from Jawahar and Balaji [15].

[a] Best among solutions reported by Jawahar and Balaji [15] and Balaji and Jawahar [2].

presented in Table 3. The total number of solutions enumerated for solving the $(2\times3\times8)$ problem instance to obtain the best solution of 258,730 monetary units (mu) by the proposed TSGA is 10,857. For the same problem instance, the number of solutions enumerated by the proposed TSGA to obtain the solution (namely, 263,000 mu obtained by the SA) is 1064, whereas the SA enumerated 5764 solutions to obtain the solution of 263,000 mu. The performance of the proposed TSGA is appreciable considering the number of solutions enumerated to obtain the best solution for solving the benchmark problem instances. The proposed genetic algorithm is able to discover the best solutions in most of the problem instance (except in five problem instances) compared to the best solutions reported in the literature. The improvement in the total cost of transportation obtained by TSGA with respect to the best solutions obtained by existing methods for various problem instances varies from 2% to 15%. In addition, the solution obtained by the proposed TSGA is very close to the lower bound solution, the average percentage deviation from the lower bound solution is about 5%.

The t-test for the significance of pairwise difference has been conducted with respect to the solutions obtained by solving benchmark problem instances using the best existing algorithm and the proposed TSGA-RN (see Table 3: columns 3 and 5) and the result of t-test shows there is no significant difference between solutions obtained by best existing algorithm and GA. However, we believe that such an improvement of the proposed TSGA over the best existing algorithms is quite valuable in real-life situations and that we have obtained solutions that could serve as benchmarks for future research attempts. Nevertheless, we proceed to develop another heuristic methodology for this two-stage problem in Section 8.

6. Scenario-2: consideration of per unit transportation cost and fixed cost for opening or operating a DC

Scenario-2 problem considers a capacity constrained two-stage fixed-charge transportation problem with the consideration of per-unit transportation cost between facilities of a supply chain and the fixed cost associated with the opening a distribution center (refer to [8] for details).

The objective in the two-stage TP is to minimize the total transportation cost given by

$$Z=\sum_{i=1}^{l}\sum_{j=1}^{m}C_{ij}X_{ij}+\sum_{j=1}^{m}\sum_{k=1}^{n}T_{jk}Y_{jk}+\sum_{j=1}^{m}\sum_{k=1}^{n}H_j\gamma_j, \tag{13}$$

subject to

$$\sum_{j=1}^{m}X_{ij}\le S_i,\quad i=1,2,\ldots,l, \tag{14}$$

$$\sum_{k=1}^{n}Y_{jk}\le W_j\gamma_j,\quad j=1,2,\ldots,m, \tag{15}$$

$$\sum_{j=1}^{m}Y_{jk}=D_k,\quad k=1,2,\ldots,n, \tag{16}$$

$$\sum_{j=1}^{m}\gamma_j\le U, \tag{17}$$

$$\gamma_j=\begin{cases}0 & \text{if distribution center } j \text{ is closed,}\\ 1 & \text{otherwise,}\end{cases}$$

$$\sum_{i=1}^{l}X_{ij}=\sum_{k=1}^{n}Y_{jk}\quad \forall j, \tag{18}$$

$$\sum_{i=1}^{l}S_i\ge\sum_{k=1}^{n}D_k\quad \forall j \quad \text{with}$$

$$X_{ij}\ge0\quad\text{and}\quad Y_{jk}\ge0\ \text{(non-negativity constraints).} \tag{19}$$

In Scenario-2 problem, objective function (13) minimizes the total transportation cost involving the fixed costs and the per-unit transportation costs. Constraint (14) ensures that the quantity

Please cite this article as: Antony Arokia Durai Raj K, Rajendran C. A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem. Computers and Operations Research (2011), doi:10.1016/j.cor.2011.09.020

shipped out from a plant is less than or equal to the available capacity. Constraint (15) ensures that the quantity shipped out from a distribution center is less than or equal to the capacity at the distribution center, and it also ensures that if $Y_{jk}>0$, then $\gamma_j=1$. Constraint (17) ensures that the number of distribution centers opened is not greater than the allowed upper limit. Constraint (18) ensures the flow conservation in the system, i.e., the total quantity shipped from plants to distribution centers is equal to the quantity shipped from distribution centers to customers. Constraint (19) ensures the non-negativity of the decision variables.

The proposed GA (TSGA) for Scenario-1 is now modified to solve the two-stage transportation problem (Scenario-2) that considers the opening cost of a DC and the per-unit transportation cost from a given plant to a given DC, and the per-unit transportation cost from a DC to a customer. The modifications are the following:

1) Set $F_{ij}=0$, and $G_{jk}=0$, while computing values for $(C'_{ij})^{p-dc}_{static}$, $(C'_{jk})^{dc-cu}_{static}$, $(C''_{ij})^{p-dc}_{static}$, $(C''_{jk})^{dc-cu}_{static}$, $(C'_{ij})^{p-dc}_{dynamic}$, $(C''_{ij})^{p-dc}_{dynamic}$, $(C'_{jk})^{dc-cu}_{dynamic}$ and $(C''_{jk})^{dc-cu}_{dynamic}$, and
2) instead of initializing capacity available at DC to ∞ (i.e., $W_j=\infty\ \forall j=1, 2,\ldots,m$), consider the actual capacity available at each DC.

With these modifications, we use the proposed TSGA to solve the Scenario-2 transportation problem. The proposed TSGA has been executed for 10,000 generations or 36,000 CPU-time seconds, whichever is earlier.

7. Results and discussion for Scenario-2

The performance of the proposed GA is compared with the best existing algorithm. The proposed algorithm (TSGA) is modified (as indicated in Section 6) to solve the benchmark instances for Scenario-2. The TSGA is run with a random number stream $\{u\}$ (i.e., uniform random numbers) and the next run is done using $\{1-u\}$ (i.e., antithetic random numbers). The former run is termed TSGA-RN and the later run is termed TSGA-ARN. The best existing algorithm was proposed by Gen et al. [8]. The best values over 30 runs (i.e., 10 runs with three different population sizes for each problem) were reported in the paper by Gen et al. [8] and the results of performance evaluation are presented in Table 4. It is clear that the proposed TSGA performs better than the best existing method. In order to have more insight into the relative performance of the proposed algorithm and existing algorithm, we first note the number of solutions totally enumerated by the existing algorithm. Accordingly, we have executed our TSGA by keeping this total number of solutions as the termination condition for the given problem instance. We then note the best solution up to that number of solutions enumerated. The results are presented in Table 5. It is clear that our proposed algorithm continues to perform better than the existing algorithm even with this restriction on the number of solutions enumerated.

In order to evaluate the performance of proposed genetic algorithm, a t-test for the significance of pairwise difference has been conducted with respect to the solutions obtained from runs TSGA-RN and TSGA-ARN with respect to Scenario-2 (see Table 4: columns 4 and 5). The null hypothesis states that there is no significant paired difference between the performance yielded by the execution of TSGA with the use of random numbers and those by TSGA with use of the corresponding antithetic random numbers. The paired t test has been conducted using SPSS-12 (with $\alpha=0.05$) and it has shown no significant difference between solutions obtained by TSGA-RN and TSGA-ARN, thereby indicating the robustness and consistency of the performance of the proposed GA with respect to results indicated in Table 4. The proposed genetic algorithm is able to the discover same or better solutions in all the problem instances. The improvement in the total cost of transportation (i.e., objective function value) varies from 1% to 6%. The t-test for the significance of pairwise difference has been conducted with respect to the solutions obtained by solving benchmark problem instances using the best existing algorithm and the proposed TSGA-RN (see Table 4: columns 3 and 4), and the result do not indicate any significant difference between solutions obtained by the best existing algorithm and TSGA. Even though the results of t-test do not show any significance difference, we believe that the improvement of even 1% is quite important and valuable in real-life situations, and that we have obtained solutions that could serve as benchmarks for future research attempts.

Table 4
Performance evaluation of proposed genetic algorithm and the best existing method (GA by Gen et al. [8]) for Scenario-2.

Problem instance	Optimal solution (see [8])	Best existing method	Proposed TSGA	
			TSGA-RN	TSGA-ARN
$3\times4\times5$	***1089***	***1089***	***1089***	***1089***
$4\times5\times10$	***2283***	***2283***	***2283***	***2283***
$4\times5\times15$	***2527***	***2527***	***2527***	***2527***
$8\times10\times20$	***2886***	***2886***	***2886***	***2886***
$10\times20\times40$	–	2952	***2924***	***2924***
$15\times25\times50$	–	2962	***2877***	***2877***
$40\times70\times100$	–	3136	***2933***	2938

Note: A solution in bold and italics indicates the optimal/best solution with respect to that problem instance.
The problem instances were taken from Gen et al. [8].
The proposed GA has been executed over 36,000 CPU-time seconds or 10,000 generations, whichever is earlier.

Table 5
Performance of proposed GA and the best existing method (GA by Gen et al. [8]) with respect to the same number of solutions enumerated (as specified by Gen et al. [8]) for Scenario-2.

Problem instance	Limit on the total number of solutions enumerated	Best existing method	Proposed TSGA	
			TSGA-RN	TSGA-ARN
$3\times4\times5$	6000	***1089***	***1089***	***1089***
$4\times5\times10$	40,000	***2283***	***2283***	***2283***
$4\times5\times15$	90,000	***2527***	***2527***	***2527***
$8\times10\times20$	300,000	***2886***	***2886***	***2886***
$10\times20\times40$	1,500,000	2952	***2924***	***2924***
$15\times25\times50$	2,000,000	2962	2884	***2877***
$40\times70\times100$	2,000,000	3136	***3021***	3027

Note: A solution in bold and italics indicates the optimal/best solution with respect to that problem instance.
The problem instances were taken from Gen et al. [8].
The values (with limits on the total solutions enumerated) are as given by Gen et al. [8].

8. A new methodology to solve the Scenario-1 two-stage transporation problem and its incorporation in the proposed genetic algorithm

In this section a methodology is proposed to represent the uncapacitated two-stage FCTP (i.e., Scenario-1 problem), and then

Please cite this article as: Antony Arokia Durai Raj K, Rajendran C. A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem. Computers and Operations Research (2011), doi:10.1016/j.cor.2011.09.020

solve the resulting problem using the single-stage genetic algorithm (SSGA—see [21]). The Scenario-2 two-stage transportation problem is not considered in this attempt because the Scenario-2 two-stage transportation problem has capacitated storage at DCs, and hence it is not possible to adapt the Scenario-2 two stage transportation problem to be solved by our SSGA.

Table 6a
Transportation matrix with shipment Q_{ijk} in each cell.

Plant	Distribution center	Customer Cu_1	Cu_2	…	…	Cu_n	Supply
P_1	DC_1	Q_{111}	Q_{112}	…	…	Q_{11n}	S_1
:	:	:	:			:	
P_1	DC_m	Q_{1m1}	Q_{1m2}	…	…	Q_{1mn}	
P_2	DC_1	Q_{211}	Q_{212}	…	…	Q_{21n}	S_2
:	:	:	:			:	
P_2	DC_m	Q_{2m1}	Q_{2m2}	…	…	Q_{2mn}	:
:	:	:	:			:	
:	:	:	:			:	:
P_l	DC_1	Q_{l11}	Q_{l12}	…	…	Q_{l1n}	
:	:	:	:			:	S_l
P_l	DC_m	Q_{lm1}	Q_{lm2}	…	…	Q_{lmn}	
	Demand	D_1	D_2			D_n	

Table 6b
The resulting per-unit transportation cost matrix with element τ_{ij} equal to ($C_{ij}+T_{jk}$) after adding a dummy customer.

Plant	Distribution center	Customer Cu_1	Cu_2	Cu_3	Cu_4	Cu_5	Supply
P_1	DC_1	35	42	77	27	0	
P_1	DC_2	10	9	55	25	0	500
P_1	DC_3	24	33	89	99	0	
P_2	DC_1	43	50	85	35	0	
P_2	DC_2	65	64	110	80	0	400
P_2	DC_3	25	34	90	100	0	
P_3	DC_1	68	75	110	60	0	
P_3	DC_2	25	24	70	40	0	300
P_3	DC_3	65	74	130	140	0	
	Demand	250	350	50	350	200	

8.1. Methodology to solve Scenario-I problem using the SSGA

The representation of the uncapacitated two-stage FCTP is given in Table 6a. This representation is proposed to adapt the uncapacitated two-stage FCTP and then solve the resulting problem using the SSGA. In Table 6a, Q_{ijk} denotes the quantity shipped from plant i to customer k through DC j. The above adaption holds because the DC is unlimited in its capacity. Consider the following example to illustrate the steps essential to solve the adapted two-stage transportation problem (Scenario-1) using the SSGA. The problem instance has 3 plants, 3 uncapacitated DCs and 4 customers. Reconsider the cost matrices with per-unit transportation cost and fixed cost of transportation given in Tables 1a and 1b. Balancing the transportation matrix, and adding C_{ij} and T_{jk}, we get the resulting per-unit transportation matrix and fixed-cost matrix as given in Tables 6b and 6c, respectively. Note that the fixed cost associated with a route from a plant to a DC and the fixed cost associated with a route from a DC to a customer should not be summed up (as the summation done with respect to per-unit transportation cost) because it leads to duplication of fixed cost while computing the total cost of transportation (see Eq. (1) in Section 3). The string length required to represent a full chromosome is ($l\times m\times n$). The compact form proposed in this study will not require a string length of ($l\times m\times n$) but in the order of ($l+n-1$) because we store only the basic cells as the chromosome.

8.2. Procedure to convert permutation representation to matrix solution

The formal procedure to convert a permutation solution into a matrix solution is presented in Appendix A3 (see procedure *PERM_MATRIX* 3 for details). An example is given below to explain the mechanism to convert the chromosome in permutation representation to its corresponding matrix representation. Consider the following chromosome (in the permutation representation): {5, 1, 7, 27, 19, 34, 33, 35, 20, 30, 15, 45, 10, 40, 25, 28, 13, 3, 18, 43, 6, 16, 26, 4, 36, 8, 11, 38, 2, 31, 37, 12, 23, 17, 9, 21, 32, 39, 41, 42, 22, 29, 14, 24, 44} with respect to Tables 6b and 6c. Note that gene 1 corresponds to cell 5 (in permutation representation), which is cell (1,1,5) (in matrix representation). Allocate the maximum shipment possible (here it is 200) to cell 5, and then update correspondingly the left-over supply and left-over demand, respectively. Set {$alloc\_array_3$ (b), $b=1$}={5}, representing the corresponding shipment from a plant to a customer through a DC (in this example it corresponds to a shipment from plant 1 to customer 5 through DC 1). Repeat the above procedure

Table 6c
The resulting fixed-cost transportation matrix after adding a dummy customer.

Plant	Distribution center	Fixed cost from plant i to distribution center j	Fixed cost from distribution center j to customer k — Customer Cu_1	Cu_2	Cu_3	Cu_4	Cu_5	Supply
P_1	DC_1	400	2300	5000	6000	7500	0	
P_1	DC_2	7000	3500	6000	8500	9000	0	500
P_1	DC_3	5500	7000	4500	200	500	0	
P_2	DC_1	1700	2300	5000	6000	7500	0	
P_2	DC_2	8000	3500	6000	8500	9000	0	400
P_2	DC_3	3500	7000	4500	200	500	0	
P_3	DC_1	800	2300	5000	6000	7500	0	
P_3	DC_2	7700	3500	6000	8500	9000	0	300
P_3	DC_3	6000	7000	4500	200	500	0	
			250	350	50	350	200	

Please cite this article as: Antony Arokia Durai Raj K, Rajendran C. A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem. Computers and Operations Research (2011), doi:10.1016/j.cor.2011.09.020

until all the genes are considered, one by one from left to right. The final matrix representation corresponding to the above mentioned permutation representation is given in Table 7.

Hence, the final $\{alloc\_array_3\ (b)\}$ is {5–1–7–27–19–34–33}. The calculation of Z for the above matrix solution is given below:

$$Z = (250 \times 35) + (200 \times 0) + (50 \times 9) + (100 \times 35) + (300 \times 34) + (50 \times 110) + (250 \times 60) + 400 + 7000 + 1700 + 3500 + 800 + 2300 + 0 + 6000 + 7500 + 4500 + 6000 + 7500 = 90600 \text{ monetary units.}$$

The SSGA (see [21]) developed for solving the single-stage fixed-charge transportation problem is now applied to solve the modified uncapacitated two-stage fixed-charge transportation problem in this study. We now define the following:

pop_size: population size or number of chromosomes in the population;
par_pop: parent population, consisting of *pop_size* parent chromosomes; and
int_pop: intermediate population consisting offspring that are obtained from the crossover of parent chromosomes in *par_pop*.

The overall procedure of the SSGA is given below:

Step 1: Follow the procedure explained in Section 4.2 to generate chromosomes to initialize the population in SSGA, with the consideration of cost values (see Section 8.1, and Tables 6b and 6c as examples).
Step 2: Evaluate all the chromosomes in the initial population and choose the best distinct *pop_size* chromosomes from the initial population based on the objective function Z, and call it as *par_pop*.
Step 3: Apply the proposed improvement schemes on every chromosome with the application of Phase I on the complete chromosome followed by the application of Phase II on the complete chromosome (see Section 4.5 on Improvement Scheme for details).
Step 4: Select two chromosomes for crossover from *par_pop*, based on fitness values of chromosomes by following the roulette wheel selection.
Step 5: Perform the crossover operation to produce one offspring (see Section 4.4 for details). Note that the crossover operator discussed in Section 4.4 is for the two-stage transportation problem, which constitutes of two sub-chromosomes, whereas in case of single-stage transportation problems, there is only one chromosome, and hence the procedure remains the same otherwise.
Step 6: Apply the proposed improvement schemes on the offspring.
Step 7: After repeating Steps 4–6, and thus generating offspring in the *int_pop*, the best distinct *pop_size* chromosomes among *par_pop* and *int_pop* are chosen to survive into the next generation.
Step 8: Terminate the algorithm, based on the termination criterion for a given problem.

Table 7
Matrix representation corresponding to the given solution (in the permutation representation): {5, 1, 7, 27, 19, 34, 33, 35, 20, 30, 15, 45, 10, 40, 25, 28, 13, 3, 18, 43, 6, 16, 26, 4, 36, 8, 11, 38, 2, 31, 37, 12, 23, 17, 9, 21, 32, 39, 41, 42, 22, 29, 14, 24, 44}.

		Customer					Supply
Plant	Distribution center	Cu_1	Cu_2	Cu_3	Cu_4	Cu_5	
P_1	DC_1	250				200	
P_1	DC_2		50				500
P_1	DC_3						
P_2	DC_1				100		
P_2	DC_2						400
P_2	DC_3		300				
P_3	DC_1			50	250		
P_3	DC_2						300
P_3	DC_3						
	Demand	250	350	50	350	200	

9. Results and discussion for Scenario-I problem using SSGA

The SSGA (adapted to solve the Scenario-I problem in this work) is run with a random number stream $\{u\}$ (i.e., uniform random number, called SSGA-RN) and the next run is doing using $\{1-u\}$ (i.e., antithetic random numbers, called SSGA-ARN). The results with respect to the two-stage transportation problem instances (when the adapted uncapacitated two-stage transportation problem is solved using SSGA proposed for solving single-stage transportation problem) are given in Table 8. The proposed SSGA performs very well in all the problem instances, improving over the best existing algorithms (GA by Jawahar and Balaji [15]; and SA by Balaji and Jawahar [2]) and also our TSGA. The total number of solutions enumerated to solve benchmark problem instances using the SA by Balaji and Jawahar [2] is presented in Table 8. The results also show that the proposed SSGA performs very well in solving all problem instances. The performance of proposed SSGA is worthy of consideration with respect to the number of solutions enumerated to obtain the best solution for solving the benchmark problem instances considered in this study. The solution obtained by the proposed GA is close to the lower bound solution, the average percentage deviation from the lower bound solution being about 3%. Since columns 7 and 8 in Table 8 are identical, it is evident that there is no significant difference between solutions obtained by SSGA-RN and SSGA-ARN, thereby showing the robustness and consistency of the performance of the proposed SSGA with respect to results indicated in Table 8. Further, t-test for the significance of pairwise difference has been conducted with respect to the solutions obtained by solving benchmark problem instances using the best existing algorithms and the proposed SSGA-RN (see Table 8: columns 3 and 7), showing that there exists significant difference between solutions obtained by the best existing algorithms and SSGA; and thereby showing the performance of the proposed SSGA being superior to performance of the best existing algorithms with respect to results indicated in Table 8. Furthermore, t-test for the significance of pairwise difference has been conducted with respect to the solutions obtained by solving benchmark problem instances using the proposed TSGA-RN and SSGA-RN (see Table 8: columns 5 and 7), indicating a significant difference between solutions obtained by the TSGA and the SSGA, and thereby showing the superior performance of the proposed SSGA in solving the two-stage fixed-charge transportation problem.

10. Summary

In this paper, we have proposed a TSGA to solve two-stage transportation problem by considering two scenarios. The first scenario considers the per-unit transportation cost and the fixed cost associated with the route, coupled with the unlimited capacity at every DC (or warehouse). The proposed TSGA with

Please cite this article as: Antony Arokia Durai Raj K, Rajendran C. A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem. Computers and Operations Research (2011), doi:10.1016/j.cor.2011.09.020

Table 8
Performance evaluation of the TSGA,SSGA and the best existing methods.

Problem instance	Lower bound on Z (obtained by relaxing the binary variables)	Best solution[a]	Number of solutions enumerated by the SA by Balaji and Jawahar [2]	Proposed TSGA		Proposed SSGA		Number of solutions enumerated by SSGA to get the best solution
				TSGA-RN	TSGA-ARN	SSGA-RN	SSGA-ARN	
$2\times2\times3$	112,600.00	***112,600***	1444	***112,600***	***112,600***	***112,600***	***112,600***	637
$2\times2\times4$	237,750.00	***237,750***	1924	***237,750***	***237,750***	***237,750***	***237,750***	857
$2\times2\times5$	173,878.57	***180,450***	2404	***180,450***	***180,450***	***180,450***	***180,450***	1214
$2\times2\times6$	157,050.00	***165,650***	2884	***165,650***	***165,650***	***165,650***	***165,650***	1354
$2\times2\times7$	161,711.43	***162,490***	3364	***162,490***	162,850	***162,490***	***162,490***	1889
$2\times3\times3$	56,020.00	***59,500***	2164	***59,500***	***59,500***	***59,500***	***59,500***	1503
$2\times3\times4$	31,522.70	***32,150***	2884	***32,150***	***32,150***	***32,150***	***32,150***	1859
$2\times3\times6$	64,724.17	69,970	4324	67,380	67,380	***65,945***	***65,945***	2577
$2\times3\times8$	253,419.29	263,000	5764	***258,730***	***258,730***	***258,730***	***258,730***	5235
$2\times4\times8$	74,737.50	80,400	7684	84,600	84,600	***77,400***	***77,400***	5246
$2\times5\times6$	73,195.00	94,565	7204	80,865	80,865	***75,065***	***75,065***	3574
$3\times2\times4$	45,470.00	***47,140***	2884	***47,140***	***47,140***	***47,140***	***47,140***	1429
$3\times2\times5$	169,825.00	178,950	3604	178,950	178,950	***175,350***	***175,350***	2061
$3\times3\times4$	52,591.67	***57,100***	4324	61,000	61,000	***57,100***	***57,100***	3060
$3\times3\times5$	151,394.29	***152,800***	5404	156,900	171,760	***152,800***	***152,800***	4555
$3\times3\times6$	132,890.00	***132,890***	6484	***132,890***	***132,890***	***132,890***	***132,890***	2981
$3\times3\times7$ (a)	93,111.09	104,115	7564	106,745	106,745	***99,095***	***99,095***	7095
$3\times3\times7$ (b)	279,514.41	287,360	7564	295,060	295,060	***281,100***	***281,100***	7011
$3\times4\times6$	71,462.50	77,250	8644	81,700	81,700	***76,900***	***76,900***	7105
$4\times3\times5$	118,450.00	***118,450***	7204	***118,450***	***118,450***	***118,450***	***118,450***	4227

A solution in bold and italics indicates the best solution with respect to that problem instance.
The problem instances were taken from Jawahar and Balaji [15].

[a] Best among solutions reported by Jawahar and Balaji [15] and Balaji and Jawahar [2].

respect to Scenario-1 performs quite well in most of the problem instances. The second scenario considers the opening cost of a DC (with capacity constraints) and the per-unit transportation cost from a given plant to a given DC, and the per-unit transportation cost from the DC to a customer. The proposed TSGA is adapted to solve the two-stage transportation problem in Scenario-2. This adapted TSGA performs very well in all problem instances with respect to Scenario-2. A methodology is also proposed to represent the two-stage fixed-charge transportation problem into a single-stage FCTP, and then solve the resulting problem using the proposed SSGA. The SSGA performs well in all problem instances with respect to Scenario-1. Future researchers can attempt to solve the two-stage transportation problem considering the step fixed-charge between facilities in a supply chain. The interested readers are welcome to contact the authors to get the codes of the proposed algorithms.

Acknowledgment

The authors are grateful to the referees for the suggestions and comments to improve the earlier two versions of this paper.

Appendix A

A.1. Procedure to convert from full permutation representation to matrix representation (PERM_MATRIX 1)

We now present the pseudo-code to convert a chromosome of an arbitrary permutation of $(m\times n)$ to the solution in the matrix representation. This procedure is called *PERM_MATRIX* 1. First, the S-C 1 representing the solution from DC to customer is evaluated, and then the S-C 2 representing the solution from plant to DC is evaluated. Let the chromosome with permutation representation be denoted by$\{\text{chr}_1(p),p=1,2,\ldots,(m\times n)\}$.

Procedure initialization: Initialize $b=0$; $alloc\_array_1$ to a null array; $W_j^{act}=0\ \forall j$, $W'_j=W_j\ \forall j$, and $D'_k=D_k\ \forall k$.

```
Begin
    for p=1 to m×n, do the following:
    {
      set j = ⌊(chr1(p)−1) / n⌋ +1,   /* j: index for DC*/
      k=chr1(p)−(⌊(chr1(p)−1)/n⌋ × n),   /* k: index for
      customer*/
      sjk = min(W'j,D'k),
      Yjk = sjk,
      D'k = D'k−sjk,
      W'j = W'j−sjk,
      W^act_j = W^act_j + sjk;
      if (sjk > 0)
      then
        set b=b+1 and
        alloc_array1 (b)=chr1(p)
    } /*end of 'for p=1 to m×n' */
end
```

Note that at most $(m+n-1)$ genes or cells have $Y_{jk}>0$, with the remaining cells or genes having $Y_{jk}=0$.

An example is given below to explain the mechanism to convert the chromosome in permutation representation to the chromosome in matrix representation (with respect to the same problem given in Tables 1a and 1b). Consider the following chromosome (in the permutation representation): {11–5–6–9–4–1–3–7–10–2–8–12}. Note that gene 1 corresponds to cell 11 (in permutation representation), which is cell (3, 3) (in matrix representation). Allocate the maximum feasible shipment possible (here it is 50) to cell 11, and then update the correspondingly the left-over supply and left-over demand, respectively. See Table A1 for details. Set $\{alloc\_array_1\ (b),\ b=1\}=\{11\}$ and $W_3^{act}=50$.

Please cite this article as: Antony Arokia Durai Raj K, Rajendran C. A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem. Computers and Operations Research (2011), doi:10.1016/j.cor.2011.09.020

Table A1
Allocation to cell 11 (in permutation representation).

		Customer (Cu) k				
		Cu_1	Cu_2	Cu_3	Cu_4	
Distribution Center (DC) j	DC_1					∞
	DC_2					∞
	DC_3			50		∞
		250	350	~~50~~ 0	350	

Table A2
Matrix representation corresponding to the given solution (in the permutation representation) {11–5–6–9–4–1–3–7–10–2–8–12}.

		Customer (Cu) k				
		Cu_1	Cu_2	Cu_3	Cu_4	
Distribution Center (DC) j	DC_1				350	∞
	DC_2	250	350			∞
	DC_3			50		∞
		250	350	50	350	

Now, consider gene 2 corresponding to cell 5 as per permutation representation and to cell (2, 1) as per matrix representation. Allocate the maximum feasible shipment possible (here it is 250) to cell 5, and then update correspondingly left-over supply and left-over demand. Repeat the above procedure until all the genes are considered one by one from left to right. The final matrix representation corresponding to the above mentioned permutation representation is given in Table A2. Hence, the final *alloc_array*$_1$ is obtained as {11–5–6–4} and $W_j^{act} = \{350, 600, 50\}$.

The procedure to convert from the permutation representation to matrix representation with respect to the S-C 2 is called as *PERM_MATRIX* 2. Note that the utilized capacity (i.e., $W_j^{act\prime}$s) at a DC is considered instead of available capacity (infinite capacity or uncapacitated). The reason of evaluating the S-C 1 and then S-C 2 is to have a meaningful and feasible allocation/shipment from plants to DCs and from DCs to customers.

A.2. Procedure to convert a solution in full permutation representation to matrix representation (PERM_MATRIX 2)

The procedure to obtain a solution in the matrix form from a given permutation sequence of basic cells (called Procedure *PERM_MATRIX* 2) is explained below.

Let the solution with the permutation representation be denoted by $\{\text{chr}_2(p), p = 1, 2, \ldots, (l \times m)\}$.

Procedure initialization: Initialize $b=0$; *alloc_array*$_2$ to a null array; $S_i' = S_i\ \forall i$, and $W_j^{act\prime} = W_j^{act}\ \forall j$.

```
Begin
    for p=1 to (l × m), do the following:
    {
        set i = ⌊(chr2(p)−1) / m⌋ + 1,
        /* i: index for plant*/
        j=chr2(p)−(⌊(chr2(p)-1)/m⌋ × m),
        /* j: index for DC*/
        sij = min(Si',Wj^act'),
        Xij = sij,
        Wj^act' = Wj^act'−sij, and
        Si' = Si'−sij;
        if (sij > 0)
            then
                set b=b+1 and
                alloc_array2 (b)=chr2(p)
    } /*end of 'for p=1 to l × m' */
end
```

Note that at most $(l+m-1)$ cells have $X_{ij} > 0$, with the remaining cells having their X_{ij} equal to 0.

A.3. Formal procedure to convert full permutation representation to matrix representation (PERM_MATRIX 3)

The procedure to obtain a solution in the matrix form from a given permutation sequence of basic cells (called Procedure *PERM_ MATRIX* 3) is explained below with an example. Let the chromosome with permutation representation be denoted by {chr (p), $p=1, 2, \ldots, (l \times m \times n)$}.

Procedure initialization: Initialize $b=0$; *alloc_array*$_3$ to a null array; $S_i' = S_i\ \forall i$, and $D_k' = D_k\ \forall k$.

```
Begin
    for p=1 to (l × m × n), do the following:
    {
        set i = ⌊(chr(p)−1)/(m × n)⌋ + 1,
        /* i: index for plant*/
        j
        =⌊((chr(p)−1)−((i−1)×(m×n)))/n⌋+1,
        /* j: index for DC*/
        k=chr(p)−(⌊(chr(p)−1)/n⌋ × n),
        /* k: index for customer*/
        sik = min(Si',Dk')
        Qijk = sik
        Si' = Si'−sik, and
        Dk' = Dk'−sik
        if (sik > 0)
        then
            set b=b+1 and
            alloc_array3 (b)=chr(p)
    }/*end of 'for p=1 to (l × m × n)' */
end
```

In case of the modified (i.e., represented) two-stage transportation problem into a single-stage problem, note that at most $(l+n-1)$ genes or cells will have $Q_{ijk} > 0$, with the remaining cells or genes having their $Q_{ijk}=0$.

A.4. Procedure to convert from matrix representation to permutation representation (called MATRIX_PERM)

The procedure to obtain a permutation sequence of basic cells from a given transport matrix with basic cells (called Procedure *MATRIX_PERM*) is explained below.

Initialize two set of basic cells, called $\{BC\}_{p-dc}/\{BC\}_{dc-cu}$ to ϕ, a null set; a tabu set of basic cells called $\{TBC\}_{p-dc}/\{TBC\}_{dc-cu}$, to ϕ, another null set; set $W_j' = W_j\ \forall j$ and $D_k' = D_k\ \forall k$; $/S_i' = S_i\ \forall i$ and $W_j^{act'} = W_j^{act}\ \forall j$. Consider the solution in matrix representation, and scan from cell (1, 1) to cell $(m, n)/(l, m)$, to identify those cells for which $Y_{jk} \neq 0$ and $Y_{jk} = \min\{W_j', D_k'\}/X_{ij} \neq 0$ and $X_{ij} = \min\{S_i', W_j^{act'}\}$. Let one such cell be $(j', k')/(i', j')$. This cell is now appended to the set of basic cells (i.e., $\{BC\}_{p\text{-}dc}/\{BC\}_{dc\text{-}cu}$); add this cell also to the tabu set of basic cells (i.e., $\{TBC\}_{p-dc}/\{TBC\}_{dc-cu}$); set $W_{j'}' = W_{j'}' - Y_{j'k'}$; $D_{k'}' = D_{k'}' - Y_{j'k'}$; and $Y_{j'k'} = 0/S_{i'}' = S_{i'}' - X_{i'j'}$; $W_{j'}^{act'} = W_{j'}^{act'} - X_{i'j'}$; and $X_{i'j'} = 0$. Repeat the process of identifying a cell, excluding those in TBC, for which $Y_{jk} \neq 0$ and

Please cite this article as: Antony Arokia Durai Raj K, Rajendran C. A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem. Computers and Operations Research (2011), doi:10.1016/j.cor.2011.09.020

Table A3
Results of pilot runs concerning the population-size setting.

Generation number	$2 \times 3 \times 8$[a]			$3 \times 3 \times 7$ (b)[a]			$3 \times 4 \times 6$[a]		
	10	20	30	10	20	30	10	20	30
Initial population	262,380	259,830	266,580	295,060	296,860	295,060	82,600	81,700	82,600
100	259,380	258,730	258,730	295,060	295,060	295,060	82,600	81,700	81,700
200	259,380	258,730	258,730	295,060	295,060	295,060	82,600	81,700	81,700
300	259,380	258,730	258,730	295,060	295,060	295,060	82,600	81,700	81,700
400	259,380	258,730	258,730	295,060	295,060	295,060	82,600	81,700	81,700
500	259,380	258,730	258,730	295,060	295,060	295,060	82,600	81,700	81,700
600	259,380	258,730	258,730	295,060	295,060	295,060	81,700	81,700	81,700
700	259,380	258,730	258,730	295,060	295,060	295,060	81,700	81,700	81,700
800	259,380	258,730	258,730	295,060	295,060	295,060	81,700	81,700	81,700
900	259,380	258,730	258,730	295,060	295,060	295,060	81,700	81,700	81,700
1000	259,380	258,730	258,730	295,060	295,060	295,060	81,700	81,700	81,700
1100	259,380	258,730	258,730	295,060	295,060	295,060	81,700	81,700	81,700
1200	259,380	258,730	258,730	295,060	295,060	295,060	81,700	81,700	81,700
1300	259,380	258,730	258,730	295,060	295,060	295,060	81,700	81,700	81,700
1400	259,380	258,730	258,730	295,060	295,060	295,060	81,700	81,700	81,700
1500	259,380	258,730	258,730	295,060	295,060	295,060	81,700	81,700	81,700
1600	259,380	258,730	258,730	295,060	295,060	295,060	81,700	81,700	81,700
1700	259,380	258,730	258,730	295,060	295,060	295,060	81,700	81,700	81,700
1800	259,380	258,730	258,730	295,060	295,060	295,060	81,700	81,700	81,700
End of execution	259,380	258,730	258,730	295,060	295,060	295,060	81,700	81,700	81,700

[a] Problem instances taken from Jawahar and Balaji [15].

$Y_{jk} = \min\{W'_j, D'_k\} / X_{ij} \neq 0$ and $X_{ij} = \min\{S'_i, W_j^{act'}\}$, and proceed until all basic cells are identified and appended to $\{BC\}_{p-dc}/\{BC\}_{dc-cu}$, finally yielding S-C 2 and S-C 1 respectively.

A.5. Discussion on the pilot runs

Pilot runs were carried out to set the parameters such as the number of probabilistically generated initial solutions and the population size. The number of chromosomes generated using probabilistic choice between LCM and VAM is set at 100 based on the quality of the final solution obtained by having 100, 200, and 300 such generated chromosomes initially. Similarly, there is no improvement in the quality of the final solution by increasing the number of chromosomes in the population, and the population size is set at 20 by considering the quality of the final solution and the convergence of algorithm to obtain the final solution. As an example, to elaborate, the population size is set by conducting pilot runs with the population sizes set at 10, 20, and 30. The results of pilot runs are summarized in Table A3. Only the solutions concerning $2 \times 3 \times 8$, $3 \times 3 \times 7$ (b) and $3 \times 4 \times 6$ problem instances (see [15] for details) are presented, since there are no differences in the quality of the final solution and the convergence of algorithm for other problem instances. It is to be noted that increasing the population size increases the computation time, and that the best setting for the population size appears to be 20 and used for further experiments; in Table A3, we also provide the quality of solution yielded by the algorithm up to the end of a given generation, associated with a given population size to justify our choice.

References

[1] Adlakha V, Kowalski K. A simple algorithm for the source-induced fixed-charge transportation problem. Journal of the Operational Research Society 2004;55:1275–80.

[2] Balaji N, Jawahar N. A simulated annealing algorithm for a two-stage fixed charge distribution problem of a supply chain. International Journal of Operational Research 2010;7:192–215.

[3] Balinski ML. Fixed cost transportation problem. Naval Research Logistics Quarterly 1961;8:41–54.

[4] Banks JB, John C, Nelson BL. Discrete-event system simulation. 2nd ed. New Delhi, India: Prentice-Hall of India Private Limited; 1998.

[5] Diaby M. Successive linear approximation procedure for generalized fixed-charge transportation problems. Journal of Operational Research Society 1981;42:991–1001.

[6] Eckert C, Gottlieb J. Direct representation and variation operators for the fixed charge transportation problems. In: Proceedings of the seventh international conference on parallel problem solving from nature, 2002. p. 77–87.

[7] Gen M, Cheng R. Genetic algorithms and engineering optimisation. New York: Wiley; 2000.

[8] Gen M, Altiparmak F, Lin L. A genetic algorithm for two-stage transportation problem using priority-based encoding. OR Spectrum 2006;28:335–54.

[9] Geoffrion AM, Graves GW. Multicommodity distribution system design by benders decomposition. Management Science 1974;20:822–44.

[10] Gottlieb J, Eckert C. A comparison of two representations for the fixed charge transportation problem. In: Proceeding of the sixth international conference on parallel problem solving from nature, 2000. p. 345–54.

[11] Gottlieb J, Paulmann L. Genetic algorithms for the fixed charge transportation problem. In: Proceeding of the fifth IEEE international conference on evolutionary computations, 1998. p. 330–5.

[12] Gottlieb J, Julstrom BA, Raidl GR, Rothlauf F. Prüfer numbers: a poor representation of spanning trees for evolutionary search. In: Proceeding of the genetic and evolutionary computation conference, 2001. p. 343–53.

[13] Hindi KS, Basta T, Pienkosz K. Efficient solution of a multi-commodity, two-stage distribution problem with constraints on assignment of customers to distribution centers. International Transactions in Operational Research 1998;5:519–27.

[14] Hirsch WM, Dantzig GB. Notes on linear programming: Part XIX, the fixed charge problem. Rand Research Memorandum, no. 1383, Santa Monica, California, 1954.

[15] Jawahar N, Balaji AN. A genetic algorithm for two-stage distribution problem associated with a fixed charge. European Journal of Operational Research 2009;194:496–537.

[16] Michalewicz Z, Vignaux GA, Hobbs M. A non-standard genetic algorithm for the nonlinear transportation problem. ORSA Journal on Computing 1991; 3:307–16.

[17] Murty KG. Solving the fixed charge problem by ranking the extreme points. Operations Research 1968;16:268–79.

[18] Palekar US, Karwan MH, Zionts S. A branch-and-bound method for the fixed charge transportation problem. Management Science 1990;36:1092–105.

[19] Pirkul H, Jayaraman VA. Multi-commodity, multi-plant, capacitated location allocation problem: formulation and efficient heuristic solution. Computers and Operations Research 1998;25:869–78.

[20] Raj KAAD, Rajendran C. Fast heuristic algorithms to solve a single-stage fixed-charge transportation problem. International Journal of Operational Research 2009;6:304–29.

[21] Raj KAAD, Rajendran C. A genetic algorithm for the fixed-charge transportation model: single-stage problem. Technology Operation Management, in preparation.

[22] Sandrock K. A simple algorithm for solving small fixed-charge transportation problems. Journal of Operational Research Society 1988;39:467–75.

[23] Sun M, Aronson JE, Mckeown PG, Drinka D. A tabu search heuristic procedure for the fixed charge transportation problem. European Journal of Operational Research 1998;106:441–56.

Please cite this article as: Antony Arokia Durai Raj K, Rajendran C. A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem. Computers and Operations Research (2011), doi:10.1016/j.cor.2011.09.020

[24] Syarif A, Yun Y, Gen M. Study on multi-stage logistic chain network: a spanning tree-based genetic algorithm approach. Computers and Industrial Engineering 2002;43:299–314.
[25] Syarif A, Gen M. Double spanning tree-based genetic algorithm for two stage transportation problem. International Journal of Knowledge-based Intelligent Engineering Systems 2003;7:214–21.
[26] Taha HA. Operations research: an introduction.7th ed. New Jersey: Prentice-Hall; 2003.
[27] Tragantalerngsak S, Holt J, Ronnqvist M. An exact method for the two-echelon, single-source, capacitated facility location problem. European Journal of Operational Research 2000;123:473–89.
[28] Vignaux GA, Michalewicz Z. A genetic Algorithm for the linear transportation problem. IEEE Transactions on Systems, Man and Cybernetics 1991;21: 445–52.

Please cite this article as: Antony Arokia Durai Raj K, Rajendran C. A genetic algorithm for solving the fixed-charge transportation model: Two-stage problem. Computers and Operations Research (2011), doi:10.1016/j.cor.2011.09.020