Sigma J Eng Nat Sci, Vol. 42, No. 3, pp. 714–730, June, 2024

Sigma Journal of Engineering and Natural Sciences
Web page info: https://sigma.yildiz.edu.tr
DOI: 10.14744/sigma.2023.00028

Research Article

Modeling open vehicle routing problem with real life costs and solving via hybrid civilized genetic algorithm

Erhan TONBUL[1], Melis ALPASLAN TAKAN[2,*], Gamze TUNA BÜYÜKKÖSE[3], Nihal ERGİNEL[4]

[1]Institute of Science, Anadolu University, Eskisehir, 26470, Türkiye
[2]Department of Industrial Engineering, Bilecik Seyh Edebali University, Bilecik, 11230, Türkiye
[3]Department of Distance Education, Anadolu University, Eskisehir, 26470, Türkiye
[4]Department of Industrial Engineering, Eskisehir Technical University, Eskisehir, 26555, Türkiye

ARTICLE INFO

Article history
Received: 02 August 2022
Revised: 12 October 2022
Accepted: 08 February 2023

Keywords:
Open Vehicle Routing Problem; Real Life Transportation Costs; Hybrid Metaheuristic; Genetic Algorithm

ABSTRACT

Many companies prefer to use third party logistics firms to deliver their goods and as such planning the return of the vehicles to the depot is not required. This is called open vehicle routing problem (OVRP). In literature, the OVRP is handled with minimum distance as objective function like vehicle routing problem. But in the real world, the objective function achieves minimum many costs like standard routing cost, stopping by cost and the deviation cost. The standard routes are previously defined under free market conditions by third party logistic firms. The deviation from the standard route is required to arrive cities which are not on the standard route. The stop by cost occurs on the delivery points. In this paper mentioned three costs are considered in the objective function while many papers consider only distance related costs in the literature. This paper proposes a new mathematical model for the OVRP. In the constraints, the last points of the routes are researched in detail. The standard route costs are determined by considering the last point of the route. Because of the NP-hard structure of the OVRP, the proposed mathematical model is solved with a hybrid metaheuristic called Civilized Genetic Algorithm (CGA). CGA is developed by hybridizing a modified genetic algorithm and a local search algorithm. The application of this study is implemented for the delivery routing of a combi boiler producer in Turkey. The third party logistic firms may use this proposed model and the solution approach for the real life applications.

Cite this article as: Tonbul E, Alpaslan Takan M, Tuna Büyükköse G, Erginel N. Modeling open vehicle routing problem with real life costs and solving via hybrid civilized genetic algorithm. Sigma J Eng Nat Sci 2024;42(3):714–730.

INTRODUCTION

In the classical vehicle routing problem (VRP), which is a well-known NP-hard problem introduced by Dantzig and Ramser [1] to find the optimal routes of vehicles, all customers are served by homogeneous fleet of vehicles each of which has a fixed capacity. Due to the NP-hard problem,

***Corresponding author.**
*E-mail address: melis.alpaslan@bilecik.edu.tr

This paper was recommended for publication in revised form by Regional Editor Ahmet Selim Dalkilic

several heuristic algorithms and metaheuristic algorithms have been proposed to solve these problems, such as genetic algorithm (Ornbuki et al. [2], Awad et al. [3]), tabu search algorithm (Cordeau et al. [4], Brandão et al. [5]), simulated annealing algorithm (Baños et al. [6]), ant colony algorithm (Kahar and Zobaa [7]), ant colony optimization and firefly algorithm (Goel and Maini [8]), and monarch butterfly algorithm (Yi et al. [9]).

For many VRPs, each customer demand is served by a single vehicle. The routes start and end at a single depot. The objective function is finding the best set of routes which minimize the total routing cost or the total distance traveled. Nowadays, companies prefer to get their transport jobs done by another third party professional logistic firm and they are not concerned with the return of the vehicles, which is called Open Vehicle Routing Problem (OVRP). The OVRP is an important variant of the VRP. The main difference of this specific VRP from the classical VRP is that vehicles in use do not return to the starting point. Based on graph theory, each route in the VRP is a Hamiltonian cycle and each route in the OVRP is a Hamiltonian path.

Schrage [10] defines open and close trips as a real life routing problem, which is a variant of the VRP. Defining OVRP for the first time, Sarıklis and Powel [11] stated that the aim is to minimize the total travel and vehicle operating costs starting from the depot and the vehicles do not have to return to the depot after serving all customers. In literature, there are many applications of OVRP. For instance, newspaper home deliveries or school buses do not return to the starting point. Some companies, which do not have their own fleet, outsource the delivering services to the third party logistics firms.

Although there are many papers analyzing different solution methods in the literature, the majority does not differ in the mathematical modeling of OVRP. They consider the travelling costs as a total cost of transportation. Several researchers have focused on the solution of OVRP with heuristic and metaheuristic algorithms since OVRP is an NP hard problem. Sariklis and Powell [11] used a minimum spanning tree method with penalty procedure based on cluster first route second strategy. Brandao [5] solved OVRP with maximum route length constraints with tabu search. Fu et al. [12] studied a tabu search algorithm using a new initial solution method called farthest first heuristic. Tarantillis et al. [13] proposed an OVRP with multi-depot using an algorithm called list based threshold, which is considered the solution method. They proposed a single parameter metaheuristic for the solution of the problem. Repouissis et al. [14] studied an OVRP with time windows. They used a greedy look-ahead route construction heuristic algorithm as a solution method. Pisinger and Ropke [15] used adaptive large neighborhood search to solve OVRP. Li et al. [16] reviewed OVRP algorithms in the literature and adapted their own record-to-record travel algorithm for the OVRP. Pessoa et al. [17] introduced a branch-cut-and-price algorithm to solve capacitated OVRP. Derigs and Reuter [18] implemented the attribute-based hill climber heuristic to the OVRP. Flezsar et al. [19] proposed a variable neighborhood search algorithm for the OVRP. Zachariadis and Kiranoudis [20] proposed a local search metaheuristic which examines wide solution neighborhoods for the OVRP. They studied the problem with two different objective configurations. First objective configuration minimizes total number of vehicles primarily and total travelling cost secondarily, the second objective configuration minimizesonly the total travelling cost. Cao and Lai [21] researched OVRP with fuzzy demands. Stochastic simulation and improved differential evolution algorithm were used to solve the problem. MirHassani and Abolghasemi [22] used particle swarm optimization method. Li et al. [23] studied the heterogeneous fixed fleet OVRP in which customers were fulfilled by a fleet of fixed number of vehicles with different capacities. They proposed a multistart adaptive memory programming metaheuristic algorithm. Marinakis and Marinaki [24] studied a bumble bee mating optimization algorithm for OVRP. They analyzed the objective function as a hierarchical objective. First, the number of vehicles was minimized and then total distance travelled was minimized. Şevkli and Güler [25] proposed a variable neighborhood search based algorithm to solve OVRP. They also improved the results of the real- world company's solutions. Soto et al. [26] studied OVRP with multiple depots and developed a general multiple neighborhood search hybridized with a tabu search strategy. Brandao [27] studied an OVRP with time windows by using iterated local search algorithm. Hashemi and et al. [28] studied the another variation of open and VRP a combination of the multi-trip with time windows. Dasdemir et al. [29] studied a real life OVRP with overbooking, multiple objectives and unknown initial pickup points. Dutta et al. [30] proposed a model for OVRP which considers two conflicting realistic objectives: minimizing the operating costs and minimizing the carbon emission due to fuel consumption by the service vehicles. To solve the multi-objective problem Strength Pareto Evolutionary Algorithm (SPEA2) and Non-dominated Sorting-based Genetic Algorithm (NSGA-II) was studied. Sun et al. [31] presented an effective memetic algorithm for an open vehicle routing problem under uncertain travel times. Yu et al. [32] proposed a learning whale optimization algorithm for the two-dimensional loading open vehicle routing problem with time window to minimize the total distance. Hosseinabadi et al. [33] proposed a new meta-heuristic algorithm for OVRP. Ruiz et al. [34] proposed two mixed-integer formulations for the OVRP with split deliveries. They also used a cutting-plane method to improve the optimization performance.

Obviously, OVRP has been widely studied by many researchers, considering different characteristics but similar objectives to minimize the total travelling cost or total travelled distance in the literature. However, there are few papers with different objectives, such as minimizing the total number of used vehicles (MirHassani and

Abolghasemi [22]; Brandao [5]; Brandao [27]; Marinakis and Marinaki [24], Lopez-Sanchez et. al. [35]); minimizing overcapacity and over duration penalty (Tarantilis et al. [13]), and minimizing the cost of fuel emissions as a green OVRP (Niu et. al. [36], Li et al. [37]). Niu et al. [36] considered the objective function for optimizing the fuel emissions cost and solved a hybrid tabu search algorithm involving several neighborhood search strategies. Also, Vincent et al. [38] proposed OVRP with cross-docking. They proposed mixed integer mathematical model with the objective of minimizing total transportation cost incurred in the pickup and delivery process.

In competitive environment, third party logistic firms defined several standard routes and transportation prices. The standard price affects not only the distance cost but also competitors based on the rules of free market. We refer to this cost in this paper as "standard routing cost". If delivering points are on the standard route, the third party logistic firms demand extra cost for leaving the highway, entering the city and turning back to highway, which is called "stop by cost". In addition, if the delivering points are out of these routes, third party logistic firms apply the extra cost, called "deviation cost" for the fuel cost of extra distance. To the best of our knowledge, there is no study in the literature for modelling the above mentioned real life costs by now. Therefore, the main motivation of this study is to consider real life costs besides distance cost, and use this model in transportation operations carried out by third party logistic firms.

The main contributions of this paper are:

- To give an opportunity using the defined standard routes by logistic firms,
- To simulate the real life costs for OVRP,
- The real life costs include extra payments such as using highway, entering-exiting the customer city, deviation cost from the standard route.
- To apply this model by changing parameters without any modification, because this model reflects the practical life.

In this study, we present a mathematical model and an improved solution for a real life OVRP. The real life costs considered in this study belong to a combi boiler producer's logistic company in Eskisehir-Turkey. The company hires vehicles from third party logistic companies and route them for delivering their goods. What has motivated us to conduct this study is the absence of a mathematical model and a solution for real life costs which are standard routing cost, stop by cost, and deviation cost, which are explained in detail. In this paper, mentioned costs are modelled with a novel mathematical model and solved with a new hybrid metaheuristic called CGA., GA is modified and improved by preventing the incest relationships among individuals in the population. CGA is a hybridization of a modified genetic algorithm and a local search algorithm. CGA's results are proved to be moderately superior to HGA's.

The limitations of this study are: CPU times (3746) and city numbers (199). The problem parameters handled in this study can also be used for other types of VRPs such as: capacitated VRP, split-delivery VRP, VRP with time windows etc.

The rest of the paper is organized as follows: The components of proposed OVRP with real life costs are described below. The notation and mathematical model for OVRP with real costs are given. The improved genetic algorithm-CGA is explained. The computational experiments are given and the solutions of real life OVRP via CGA are presented. The conclusion is provided in the last Section.

MATERIALS AND METHODS

In many cases, manufacturing companies prefer the delivery of their finished products done by professional third party logistic firms. These manufacturing companies' logistic departments make the delivery plan of their products without having their own fleet but hiring one. They hire vehicles from a co-operating logistic firm and are not concerned about the return of the vehicles to the depot. This transportation form is called OVRP and it is the most proper model for the companies whose main task is not transportation.

In this study, OVRP with real costs in objective function is proposed. In real life, the cost includes not only the distance fuel cost but also several extra costs in free market condition. The studied problem is different from the classical OVRP due to the costs in the objective function. In the objective function, there are three types of costs which are referred to as standard routing cost, stop-by cost, and deviation cost. These three costs are described in detailed in the following sub-sections.

Standard Routing Cost

For a manufacturing company without a fleet of its own, vehicle hiring cost is the standard price for hiring a vehicle and it is defined by a third party logistic firm. Third party logistic firms tend to have standard routes for travelling from an origin (depot) to the city of the customer by following a sequence of cities, unless the origin and the customer's city is adjacent as in real life. Prices for different standard routes are defined according to free market conditions. Consequently, a manufacturing company willing to hire a vehicle has to comply with one of these predetermined routes and standard price, r_i, such that r_i is the standard price of a vehicle following a path beginning from the depot and stopping at i. The r_i price shows the basic hiring vehicle cost including fuel, driver cost, tollgates, etc., from depot to the end point of the route. Naturally, the manufacturing companies' delivery plan may start from an origin and end at a customer's city i. However, there still may be other customer's city on the standard route and/or out of the route but near the standard route related to r_i. In this case, the real route differs from the standard route between depot and i, , and other costs emerge, which are called "stop-by cost" and

"deviation cost". These costs are going to be discussed in detail in the following sub-sections.

In the objective function, this cost is described by the following equation:

$$min \sum_{i=1}^{N} r_i \left(y_{ik} - \sum_{j=1, i \neq j}^{N} x_{ijk} \right) \quad (1)$$

where, x_{ijk} shows the decision variable that if the vehicle k travels from city i to city j, it takes 1 otherwise 0. Also, y_{ik} represents other decision variable which if the vehicle k travels to customer's city i, it takes 1 otherwise 0. This means that if the customer' city i is the last point of the route, the vehicle will not travel to another customer's city j. In this case, $y_{ik} = 1$, $x_{ijk} = 0$. The standard routing cost of the customer i (r_i) will be the cost of the route that ends with customer's city i. If the customer's city i is not the last point of the route, the vehicle will travel to another customer's city j. In this case, $y_{ik} = 1$, $x_{ijk} = 1$. To find the standard cost of route, it is necessary to find the end city on route since the standard cost is defined according to the end city on route in competitive marketing conditions. What poses a challenge here is to find the end customer, as they define the standard route's cost.

Stop-by Cost

Total travelling cost also includes stop-by cost. This cost arises from entering-exiting the customer city other than the final city on the route. It means leaving the highway, entering the city, traveling to the customer city, and returning to the highway. The final node's stop by cost is excluded. The reason is that the hiring cost actually includes the cost of the final node and the cost of hiring a vehicle contains loading goods at 0 and unloading at j, and the cost of travelling to the final node j itself.

In the classical vehicle routing problem, when taking a route, there is no difference between following a particular path directly to the final customer city and going to the final customer city, by stopping-by at delivery points which are on the same particular route. However, in real life, those two cases do not have the same costs due to travelling to the exact location of the delivery point (rather than passing-by from that city's highway) and extra time spent going through potential traffic. Passing-by a node and stopping-by a node are not the same things because of the consequences of departing the highway. Thus, logistic firms demand a price of stopping-by if there are delivery points between *0* and *j*. In this study, the stop-by cost is described by Eq.2, and added to the objective function of the proposed model:

$$min \sum_{k=1}^{K} \left(stop_{cost} \left(\sum_{i=1}^{N} y_{ik} - v_k \right) \right) \quad (2)$$

where y_{ik} decision variable takes 1 value if vehicle k travels to i, otherwise takes 0 value; v_k decision variable takes 1 value if vehicle k is used for travel, otherwise takes 0; $stop_{cost}$ is a constant value for stopping to any customer city.

Deviation Cost

The deviation cost is the emerging cost in a route starting from *0* and stopping at *j*, when a vehicle is does not follow the standard route between *0* and j. In this case, a route between *0* and *j* contains customer cities which are not included in the third party logistic firm's standard route. Therefore, the vehicle following the route has to travel extra distances since the standard routes are most commonly the shortest ones. In rare cases, due to the condition of the road, the fastest route can be chosen instead of the shortest in real life.

When the hired vehicle needs to deliver the goods to a customer city that is not on the standard route but close to it, there will be a deviation from the standard route (Figure 1). This deviation has an extra cost and should be added to the hiring cost in order to attain the total travelling cost. It is simply calculated by subtracting the total distance of the standard route from the actual distance travelled by the hired vehicle. This difference is multiplied by the fuel cost per extra kilometer, which is a parameter defined by the logistic firm. In the objective function, this cost is described in the constraint set with the Eq. 3:

$$\min fuel_{cost} \sum_{k=1}^{K} \left(\sum_{i=0}^{N} \sum_{j=1}^{N} d_{ij} x_{ijk} - \sum_{i=1}^{N} d_{0i} \left(y_{ik} - \sum_{j=1}^{N} x_{ijk} \right) \right) \quad (3)$$

where, addition to below descriptions, d_{ij} represents the parameter that is the distance (kilometer) from customer i to j and $fuel_{cost}$ is the constant value that shows fuel cost per kilometer.

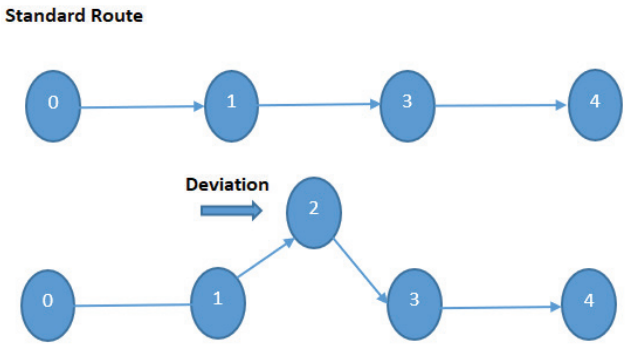

Figure 1. Representation of the Deviation Cost from Standard Route.

MATHEMATICAL MODEL FOR OVRP WITH REAL COSTS

In this study, standard routing cost is determined by considering the last customer city on the route. While composing the mathematical model of the problem, we added

a parameter r_i which indicates the cost of travelling from depot to the last point of the route. Every customer city has its own standard routing cost. However, standard routing cost is not a unique or constant standard routing cost for every city. For instance, in a competitive environment, while the cost of travelling from depot to city A is \$100, the cost of travelling from depot to city B is \$150. If a route ends with city A, the standard routing cost will be \$100 in that route. Similarly, if a route ends with city B, the standard routing cost will be \$150 in that route. The number of cities stopped on the route should be one less than the sum of the number of cities on the route because there is no stopping cost for the last city. Stop by cost is handled as a constant ($stop_{cost}$) in the model. If there is a deviation from the standard route, the fuel cost ($fuel_{cost}$) based on distance is added to objective function.

The mathematical model of the studied problem is given below:

Sets

i, j: customer city ($i, j = 0, \ldots, N$); 0 is depot location.

k: vehicles ($k = 1,2, \ldots, K$)

Decision variables

$$x_{ijk}: \begin{cases} 1, & if\ vehicle\ k\ travels\ from\ i\ to\ j \\ 0, & o/w \end{cases}$$

$$y_{ik}: \begin{cases} 1, & if\ vehicle\ k\ travels\ to\ i \\ 0, & o/w \end{cases}$$

$$v_k: \begin{cases} 1, & if\ vehicle\ k\ is\ used \\ 0, & o/w \end{cases}$$

where x_{ijk} is a binary decision variable that takes the value 1 if *j* is visited immediately after *i* by vehicle *k*, otherwise takes 0; y_{ik} is also binary decision variable which takes the value 1 if *i* is visited by vehicle *k*, otherwise 0; and v_k is a binary decision variable as well, which takes 1 if vehicle *k* is used.

α_i, α_j: positive variables used for subtour elimination

Parameters

q_i: demand of customer *i*

d_{ij}: distance from customer *i* to *j*

cap: capacity of vehicles

$stop_{cost}$: entry cost of vehicle to any customer city

$fuel_{cost}$: fuel cost per kilometer

r_i cost of travelling from depot to the last point of that route

$$min \sum_{k=1}^{K}\left(\sum_{i=1}^{N} r_i\left(y_{ik} - \sum_{j=1,,i\neq j}^{N} x_{ijk}\right) + stop_{cost}\left(\sum_{i=1}^{N} y_{ik} - v_k\right)\right) + fuel_{cost}\sum_{k=1}^{K}\left(\left(\left(\sum_{i=0}^{N}\sum_{j=1}^{N} d_{ij}x_{ijk}\right) - \sum_{i=1}^{N} d_{0i}\left(y_{ik} - \sum_{j=1}^{N} x_{ijk}\right)\right)\right) \quad (4)$$

subject to:

$$\sum_{k=1}^{K} y_{ik} = 1 \qquad i = 1, \ldots, N \quad (5)$$

$$\sum_{i=1}^{N} q_i y_{ik} \leq cap * v_k \qquad k = 1,2, \ldots., m \quad (6)$$

$$\sum_{i=0, i\neq j}^{N} x_{ijk} = y_{jk} \qquad j = 1, \ldots, N;\ k = 1,2, \ldots., m \quad (7)$$

$$\sum_{j=1}^{N} x_{ijk} \leq y_{ik} \qquad i = 1, \ldots, N;\ k = 1,2, \ldots., m;\ i \neq j \quad (8)$$

$$\sum_{j=1}^{N} x_{0jk} = v_k \qquad k = 1,2, \ldots., m \quad (9)$$

$$\alpha_i - \alpha_j + (N+1)\sum_{k=1}^{K} x_{ijk} \leq N \qquad i = 1, \ldots, N;\ j = 1, \ldots, N;\ i \neq j \quad (10)$$

where the first constraint (Eq.5) states that every customer demand must be satisfied, the second constraint (Eq.6) guarantees that the capacity of the vehicle cannot be exceeded. In this model, homogeneous fleet is used, so all vehicles have the same capacity. The third constraint (Eq.7) is the classical vehicle flow constraint defined in literature. The fourth constraint (Eq.8) is an open vehicle routing constraint which means that a vehicle does not have to return to the depot. The fifth constraint (Eq.9) indicates that if vehicle *k* is used, this vehicle travels from depot to any customer *j*. The sixth constraint (Eq.10) is the subtour elimination constraint [39].

SOLUTION METHODOLOGY

The use of metaheuristic algorithms for solving the OVRP is very common in the literature due to the NP-hard structure of the problem. They may not guarantee optimal solutions, but they provide sufficiently good solutions for the optimization problems. Well known metaheuristic algorithms are genetic algorithms, tabu search algorithm, simulated annealing algorithm, and hybrid algorithms, etc.

Table 1. Distance Matrix for Five Demand Points

	0	1	2	3	4	5
0	0	336	573	966	611	490
1	336	0	909	646	632	755
2	573	909	0	1310	589	256
3	966	646	1310	0	738	1054
4	611	632	589	738	0	333
5	490	755	256	1054	333	0

Table 2. Standard Route Costs and Demands Five Demand Points

Demand Points	Standard Cost	Demand
0	0	0
1	672	11
2	1146	6
3	1932	8
4	1222	7
5	980	7

Table 3. Distance Matrix for Ten Demand Points

	0	1	2	3	4	5	6	7	8	9	10
0	0	336	573	966	611	490	557	1034	883	901	770
1	336	0	909	646	632	755	893	755	1219	1237	1059
2	573	909	0	1310	589	256	292	1237	346	328	212
3	966	646	1310	0	738	1054	1429	397	1642	1571	1360
4	611	632	589	738	0	333	825	696	931	833	622
5	490	755	256	1054	333	0	544	981	598	535	315
6	557	893	292	1429	825	544	0	1466	342	505	476
7	1034	755	1237	397	696	981	1466	0	1579	1464	1253
8	883	1219	346	1642	931	598	342	1579	0	296	520
9	901	1237	328	1571	833	535	505	1464	296	0	246
10	770	1059	212	1360	622	315	476	1253	520	246	0

Table 4. Standard Route Costs and Demands for Ten Demand Points

Demand Points	Standard Cost	Demand
0	0	0
1	672	11
2	1146	6
3	1932	8
4	1222	7
5	980	7
6	1114	5
7	2068	4
8	1766	9
9	1802	12
10	1540	10

Table 5. The Results of Model for Five, Ten, and Fifteen Demand Points with GAMS

n	Minimum costs	CPU Time (s)
5	4116.5	0.20
10	9075.5	3.78
15	stopped	>28800.00

In this paper, the proposed model which includes the real life costs was analyzed in GAMS software for solving the small sized problems. The OVRP was solved with GAMS v24.8.2 software BARON v16.12.7 solver for five and ten demand points. The distance matrix, standard route costs and demands are given in Table 1-2 for five demand points, and Table 3-4 for ten demand points. The distance matrix represents the kilometers from depot "0" and to each other. Standard route costs show the competitive price ($) from depot to demand points. The vehicle capacity is 20, stop by cost is $36, and fuel cost per kilometer is $ 1.5 in these problems. The results are given in Table 5. This model is also tried for 15 demand points. GAMS runs for eight hours, then it is stopped because of the non-efficient solution time.

For the large sized problems, we have developed Civilized Genetic Algorithm (CGA) based on Hybrid Genetic Algorithm (HGA) described by Liu et al. [40]. We propose CGA by modifying HGA with prevention of incest relationship between individuals. We use test to show the efficiency of CGA. The short introduction of Genetic Algorithm is given below. HGA, which is the improved form of Genetic Algorithm, is briefly given. Also, the proposed CGA is introduced below in detail. The test problem results

of CGA are compared with HGA and Simulated Annealing Algorithm and discussed. After that, the proposed CGA is implemented to the real life problem. The solutions for the real life problem are given in the computational part.

Genetic Algorithm

Genetic algorithm was first described by Holland [41] in the literature. Genetic algorithms inspired biological evolution process, are generally successful in the area of complex solution spaces. Genetic algorithm is a population-based metaheuristic, so the algorithm searches from different points not from a single point of the solution space. When the structure of the problem is analyzed, chromosome representation of the genetic algorithm is taken as permutation. The fitness function measures how the solutions get better. If the fitness function is organized well, the algorithm is very effective. Mutation and crossover parameters also greatly affect the performance of the algorithm. This algorithm mimics a natural selection method in nature. Algorithm starts with a random generation of initial solutions. Each solution represents the individuals of the population.

For example, consider the traveling salesman problem with 10 customer cities. The optimal solution is 3-2-10-8-4-7-1-6-9-5. This is found by calculating the minimum total travelling distance between customer cities. The total travelling distance is the fitness function of an individual (chromosome). Each chromosome is separately created by genes, which are customer cities. This algorithm is based on gene changing by applying some operations between different individuals. These operations are selection, crossover, mutation, and swap operations. Operations work iteratively and generates better populations. Algorithm finishes when the maximum iteration number is reached. The flow of the genetic algorithm is given in Figure 2. There are three operations of genetic algorithm: selection, crossover, and mutation. These are introduced shortly as follows:

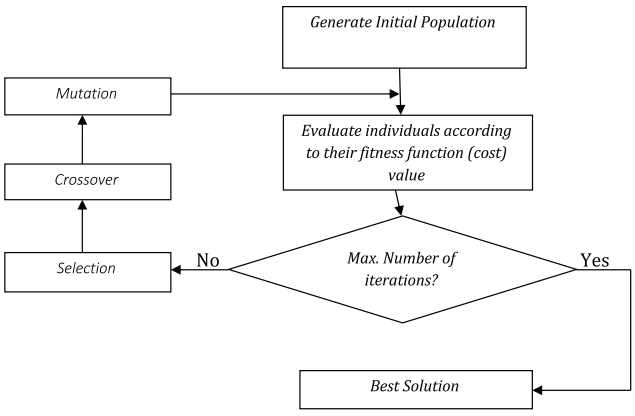

Figure 2. Genetic Algorithm's Flow Chart.

Selection Method

In the nature, it is more probabilistic that strong individuals transfer more genes to new populations. Weak individuals may continue their life, but they probably cannot transfer genes to new populations. While each individual in the population is a candidate for transferring genes to new generations in natural life, it is a systematic case where strong individuals are more likely to be selected.

In the genetic algorithm, selection is the process of selecting individuals from one population to transfer their genes to the next generation. When making this selection, individuals' probability of being selected is determined by using the fitness function value of the individuals. Therefore, it is very important to determine the fitness function well. The fitness function will take a value that represents the quality of a result.

Although there are several different methods for determining the probability of individuals being selected, the most popular methods are the tournament method and the roulette wheel method. In the tournament method, two of the randomly selected individuals will be competed to win the competition with a large number of matched function values (up to the number of individuals to be selected) and the winning individuals will be selected. Roulette wheel is a more widely used method. In roulette wheel method, the fitness function of each individual is f_i, the probability of selecting each individual is P_i. P_i is determined by the Eq. (11) as follows:

$$P_i = \frac{f_i}{\sum f_i} \tag{11}$$

This selection is done by n times. As can be seen, procedures which are similar to natural selection process, have the possibility of being selected, but are more advantageous for strong individuals.

Crossover Operation

It is the procedure in which the selected individuals are crossed to generate children. The expectation is that individuals form stronger individuals by the crossing process. A representation of crossover is very easy as seen in Figure 3. For the crossing process, each pair of matching chromosomes is randomly cut off, and the genes are mutually exchanged after the batch. This operator is a 1-point crossover operator. The two-point crossover operator likewise divides the chromosomes from two different random points and the middle part of the chromosomes are changed. Crossing occurs with a specific probability. In this way, it is considered that there may not be children of the two matching individuals. There are many different crossover operators. Which one or ones will be used is very important for the efficiency of the algorithm.

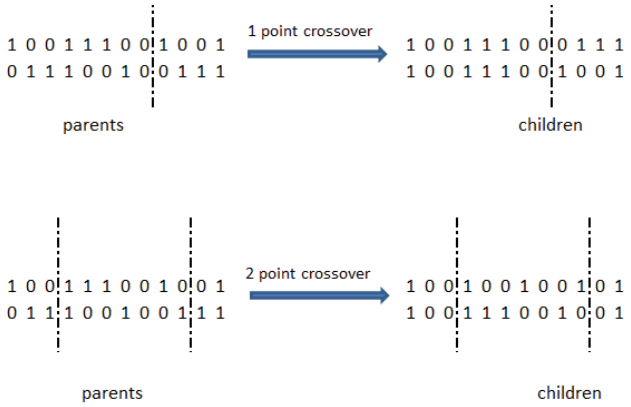

Figure 3. Crossover Operation

Mutation Operation

Mutations occur very rarely in nature. Mutation can be explained as the deterioration and differentiation of some genes due to some external factors. Also, mutation is rarely seen in the genetic algorithm. This procedure plays an important role in the search for different regions. In the mutation procedure, each gene has the possibility of deterioration. Deterioration means a binary representation which can be considered as a result of the conversion of the result from 1 to 0 or the random displacement of two genes in a permutation display. That is, a result in the form of 1-0-0-1-0-1-0-0-1 turns into 1-0-0-1-0-0-0-0-1 in the event of a mutation in the sixth gene. Although the likelihood of this event is very low, occasional mutations occur because there is this possibility for each gene in each individual. Since there are many iterations in the algorithm, mutation occurs occasionally. The probability of mutation of a gene (P_m) is another important parameter to be determined.

After the selection of individuals for crossover operation, a new population pool is generated. This pool includes old individuals and new individuals who are born with a crossover. The decision to be made is which individuals will create the new generation. This means that some individuals have to die. The tendency of strong individuals to survive and the tendency of weak individuals to disappear is obvious. Different displacement strategies may be involved. The most common strategies used are generational displacement and fixed displacement strategies. Correct determination of this strategy is the most critical issue in terms of the effectiveness of the algorithm. In generational displacement, children systematically replace with parents. In fixed displacement, n individuals are transferred to the new generation each time in the pool where children and other individuals are located. Since the basic logic of the algorithm is the survival of the stronger one, the selection method that must be created here must be dependent on the fitness function values. Running the roulette wheel in such a way to select the individuals to be transferred to the new generation is a good example of the displacement procedure. The remaining unselected individuals are removed from the population and their role in the algorithm is terminated.

Many studies improved GA during search process. Katoch et al. [42] gave a review on GA. They summarized used operators in encoding, schemes, crossover, mutation, and selection steps of GA. In selection step, roulette wheel, rank, boltzmann, tournament, and stochastic universal sampling are well-known selection techniques (Jebari [43]). In crossover step, the famous operators are single/two/k-point, uniform, partially matched, order, precedence preserving crossover, shuffle, reduced surrogate and cycle (Soon et al. [44]). In mutation step, the well-known mutation operators are displacement, simple inversion, and scramble mutation (Jebari [43]). Although various variants of GA introduced in literature, the most attractive improvements are Hybrid GAs. Sampling capability is mainly developed area. While El-Mihoub et al. [45] introduced the effect of probability of local search on the population size of GA, Espinoza et al. [46] presented self-adaptive hybrid GA for evaluating the effect of reducing the population size. Hedar and Fukushima [47] used simplex crossover, and Tan et al. [48] used simulated annealing instead of standard mutation operator. Guo et al. introduced [49] GA with column generation used to create initial solutions for GA. Luo et al. [50] proposed an algorithm combining GA and variable neighborhood search where solution space is decomposed into small multi-neighborhood spaces, and then search in each neighborhood space by GA in turn.

In this study, a new hybrid metaheuristic prevents the incest relationships among individuals in the population. CGA's results are showed that it is to be superior to HGA's.

Hybrid Genetic Algorithm

In this study, we modified HGA, which is described by Liu et al. [40]. Our modified algorithm is called "Civilized Genetic Algorithm". The HGA steps are given as follows:

Step 1. Generate initial population (with 30 chromosomes).

Step 2. Select 2 individuals by the tournament method.

Step 3. Crossover the selected individuals ($P_{crossover} = 1$).

Step 4. Select one of the crossovered individuals randomly and apply local search to its solution ($P_{mutation} = 0.25$) for the approximation of the result that the individual represents. Updated result is included in the population as a mutated new individual.

Step 5. Evaluate the fitness functions for all individuals. Select an individual with a poor value of fitness function randomly. Swap this individual with the new individual.

Step 6. Apply selection, crossover, mutation, and swap operations until the stopping criteria.

In the HGA, a comparatively small sized population is analyzed. The population is considered with 30 chromosomes. First three chromosomes of a population with 30 chromosomes are generated by using three different heuristics, and the rest of the chromosomes are generated randomly for diversification. Fitness function values of

individuals in the population are different in order to prevent loss of diversity feature in the algorithm. If there are individuals with the same fitness function value at the stage of starting the population, one of them is extracted from the population and another individual is randomly included in the population. This procedure is continued until the fitness function values of the individuals in the population are completely different from each other. At the displacement stage of the algorithm, the individuals who will participate in the population have different fitness function values from other individuals in the population.

The permutation encoding was chosen for the chromosome representation. In permutation encoding, every chromosome is a string of numbers, which represents a number in a sequence. The representation of a chromosome is carried out by ordering the numbers of the demand points in which order the vehicles will travel to these points (without any depot location). The result of this chromosome is obtained by placing the depot location at the relevant places of the order of demand points by considering the capacity constraints. Selection strategy is based on the tournament method. Three chromosomes are randomly selected to evaluate as a parent. Chromosome with the best fitness function value is taken from these three chromosomes as the first parent. The same procedure is done for the second parent among the remaining two chromosomes. Selected chromosomes are directly crossovered by using ordered crossover operator with the certain probability ($P_{crossover}$ = 1). After the crossover step, two new individuals are generated. One of the individuals is selected randomly for the mutation and the other one takes place outside of the population. Mutation is done by the probability of 0.25 ($P_{mutation}$ = 0.25) and a new result is obtained.

Instead of the classical mutation operators, local search procedures are used in the algorithm to be applied to the result obtained by 1-1 change (swap), 1-0 change (insert), and 2-opt movements are applied on the basis of "accepting the first development movement". Whenever an improvement is made, the local search procedure returns to the beginning. After the accepted movement, the corresponding result is updated. 1-1 exchange is analyzed in the neighborhood of this result. If there is no improvement, 1-0 exchange is analyzed. If there is still no improvement, 2-opt exchange is made in the neighborhood of the result. When there is no movement to improve the result, the result represents a local best and the mutation procedure ends. If the fitness function value of the solution obtained is different from the other individuals' fitness function values in the population and is greater than the worst fitness function value in the population, the chromosome which represents the resulting solution is chosen to be taken into the population, otherwise it is discarded in order not to enter the population at all.

The next stage continues with the local best result obtained. Depot location is removed from this local best result, and the result is converted into a chromosome. After this procedure, the result is added to the population again. In the displacement stage, individuals in the population are sorted in the descending order according to the fitness function value, and chromosomes are divided equally into two groups. A single individual from the group with the worse fitness function value (16th chromosome and after) is selected to be excluded from the population. Instead of that, new individuals are created with crossover and mutation operators, and the population size remains constant. The selection, crossover, mutation, and displacement phases are repeated until a termination criterion is achieved. When the terminating criterion of the algorithm is provided, the chromosome with the best fitness function value in the population is the best result obtained.

Proposed Civilized Genetic Algorithm

In this study, HGA is improved in two ways: prevention of incest relationships to strengthen diversity and two newly added crossover operators. Firstly, CGA's initial population is seeded with two individuals, which are generated via heuristic methods; nearest neighborhood and savings algorithm. The rest of the population is generated randomly. For diversification purposes, chromosomes with the same fitness function values are not accepted as suggested by HGA. Selection, crossover, and mutation procedures are implemented as described in HGA algorithm, except incest prevention method in selection procedure and two newly added crossover operators in CGA. Proposed incest prevention method and two newly added crossover operators are explained below in detail.

Prevention of Incest Relationships

In developed societies, it is known that individuals do not prefer consanguineous marriages. The reason for this is the possibility of the emergence of poorly-featured genes (genes that may be associated with various diseases or abnormal syndromes), similar to the possibility of these bad characteristics occurring in children of blood-binding parents during the crossover. Genetic algorithms also show a similar situation if the similar bad characteristics are paired. In a genetic algorithm with permutation chromosome structures, individuals with blood binding are likely to have similar gene sequences, even if they have different fitness function values.

In this case, the matching of chromosomes with similar gene sequences is not good in terms of investigating the different regions of the search space. Particularly, it is seen that the diversity of the search is weakened if the relatively good chromosomes (with incest relationship), which have bad sequences, are repeated and transformed to new generations. Liu et al. [40] proposed the hybrid genetic algorithm in the deep-rooted localization of the movement feature. Diversity is ensured by the fact that individuals in the population have different fitness function values. It is clear that this mechanism makes individuals different from each other. However, this mechanism does not have a feature

which analyzes the similarity of these different genes. In order to eliminate this deficiency, the blood ties of individuals were kept in memory in the IGA, recommended within the scope of this study, and the mating of individuals with blood ties was prevented.

The memory structure allows each individual to have six children (the number of children kept in memory is a parameter and can be chosen differently) and two parents can be involved. Since one of the two new individuals are sent out of the population after the crossover operation in the algorithm steps, the fraternal relations are not kept in memory. Thanks to this structure, the matching of individuals with similar genes is prevented and the diversity of the algorithm is strengthened.

Newly added crossover operators

In the research of Liu et al. [40], ordered crossover operator is used in the hybrid genetic algorithm only. This operator is very effective fat genetic algorithms with permutation representation. In this paper, two new crossover operators are used additionally: partially mapped crossover and 2-point crossover. In the proposed civilized genetic algorithm, these three crossover operators are used randomly to create different options on chromosomes, so they offer solution search space to have a stronger structure.

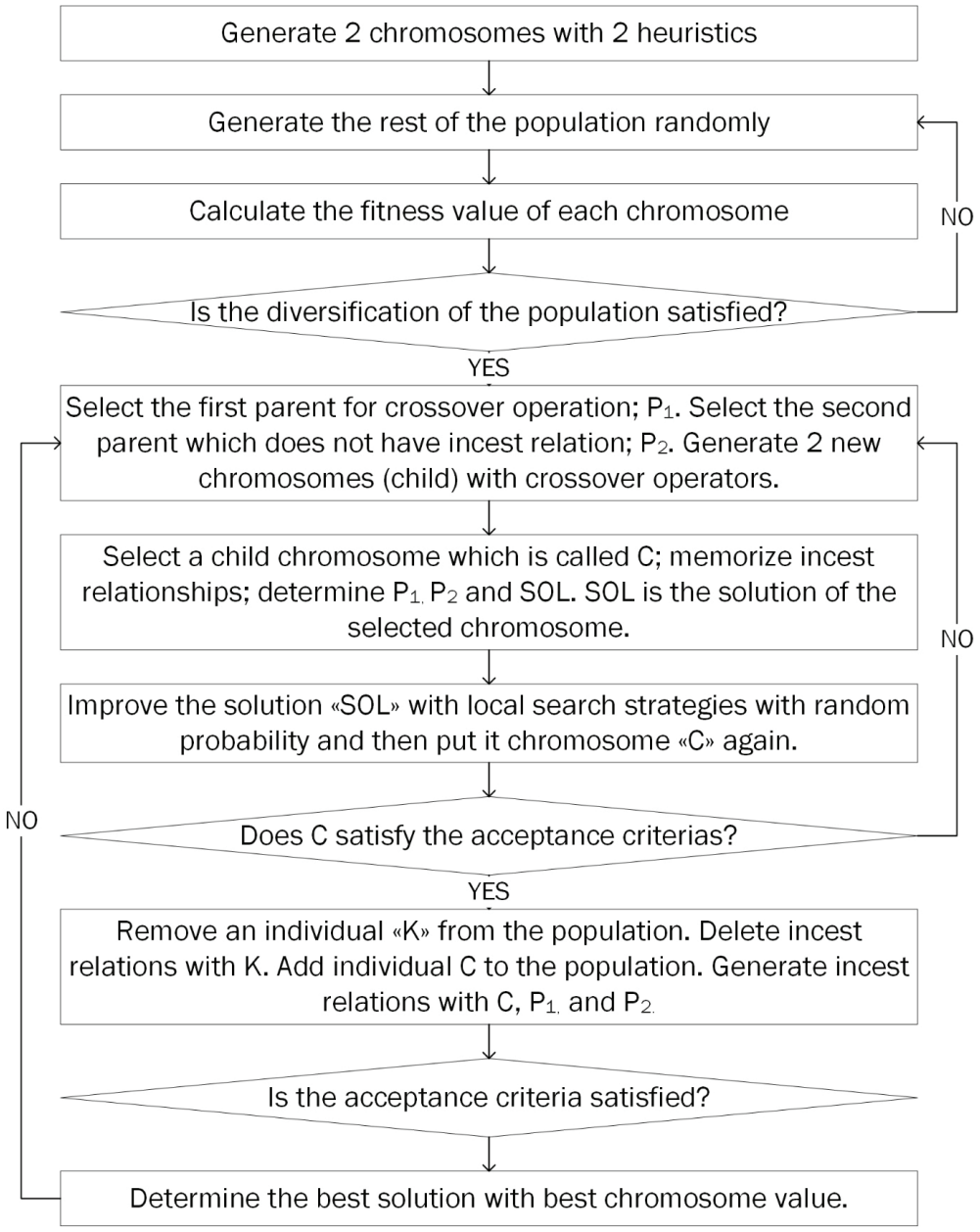

Figure 4. Proposed Civilized Genetic Algorithm Flowchart.

The flowchart of the developed algorithm is given below in Figure 4.

RESULTS ON COMPUTATIONAL EXPERIMENTS

In literature review, we encountered no appropriate test samples for the mathematical model proposed in this paper. In tables below, the objective of the sample problems is to minimize the total distance travelled. In these sample problems, routes originating from the depot ends in the depot. These well-known vehicle routing test problems are chosen to test the performance of the proposed algorithm. To make a just comparison, only for testing purposes, proposed algorithm's objective function is modified as minimizing the total distance, and the algorithm is made suitable to solve classical vehicle routing problems, in which routes originate from the depot and ending in the depot. According to the results, proposed algorithm could obtain the best known solutions for the majority of the test problems and improve the best known solutions in two instances. These results indicate that the proposed algorithm works effectively. After the algorithm is proved to have a good performance, it is altered to its original form regarding the constraints and the objective function defined in this study.

The proposed civilized genetic algorithm (CGA), a hybrid genetic algorithm (LJG-HGA) studied by Liu et al. [40], and classical simulated annealing algorithm (SA) were coded in Visual Studio 6.0 for the purposes of benchmarking.

Since there are not enough number of test instances in the literature for open vehicle routing problems, we decided to do the benchmarking of the algorithms on the most studied capacitated vehicle routing problems. The benchmarking is done between simulated annealing algorithm, the metaheuristic proposed by Liu et al. [40] (LJG-HGA), and this study's proposal, Civilized Genetic Algorithm (CGA). To do the benchmarking, two sets of test instances are selected. A set of 27 test problems (Set A) by Augerat et al. [51] (Table 6) and a set of 7 problems (Table 7), which does not include the loading time and time limit of the routes, out of 14 test problems by Christofides et al. [52] were utilized.

LJG-HGA and CGA were implemented with 10 different parameters for each test problem while simulated annealing was tried three times with different parameters for each test problem, due to the dramatic increase in CPU time. The comparisons are demonstrated in Table 6 and Table 7.

The superiorities that the proposed CGA have over LJG-HGA and SA:

In terms of the best solutions found:

- In 15 instances, CGA performed better than LJG-HGA in terms of the *best solution* the algorithm could obtain. Similarly, in 14 instances, CGA found *better best solutions* compared to SA.
- In 18 instances, best solutions did not differ between CGA and LJG-HGA.
- The number of test problems, in which the best solutions were identical, were 11 in comparison to CGA and SA.
- However, in one instance out of 34, LJG-HGA could obtain a better best solution than CGA and SA generated a better best solution than CGA in another instance.

In terms of average of the outcomes achieved:

- In 20 of 34 test problems, CGA outperformed LJG-HGA by means of the *average outcome* of the solutions after 10 times of running for each algorithm in each instance.
- In 10 test problems, the average did not differ for CGA and LJG-HGA.
- However, in four instances, LJH-HGA surpassed CGA in terms of the average.
- In all test problems except one, CGA's average was better than the SA.

In terms of improvements made:

- Both LJG-HGA and CGA improved the best known solution for the same two test instances while SA could improve one test instance's best known solution. In both of the test problems, for which the best known solution could be improved, CGA's improvement was better than the LJG-HGA. In A-n62-k8, where one of the instances the improvement took place, CGA improved the best known solution by %2.01, and in A-n80-k10, CGA's improvement was %1.42.

In terms of CPU time:

In 24 instances, CGA generated its best solution faster than the LJG-HGA. In terms of CPU time, LJG-HGA outperformed others in eight instances. SA was significantly slower than both LJG-HGA and CGA in all circumstances.

- In average, CGA's CPU times are less than SA's by 47.6%, and less than LJG-HGA's by 29.4%.
- The CPU times for CGA were six seconds for 38 nodes in A-n38-k5; 34 seconds for 55 nodes in A-n55-k9; 331 seconds for 80 nodes in A-n80-k10; and 3746 seconds for 199 nodes in vrpnc5. These CPU times are acceptable for the solution of the problem. CGA's performance was better than the others when compared with respect to CPU times, as well.

Overall, CGA's performance was found to be superior to LJG-HGA and SA in either finding better best solutions and in terms of averages, or the CPU time spent while finding solutions. In two test problems, best known solution is improved. It is observed that CGA performs better in relatively bigger scale problems. For the instances which has more complex solution spaces, the proposed algorithm, CGA, performs relatively slow in acceptable limits. Next section describes the results of the test problems.

RESULTS AND DISCUSSION

Following the demonstration of the efficiency of proposed CGA in the previous section, the real life problem

Table 6. Augerat et al. (Set A) VRP Test Instances

		SA				LJG-HGA				CGA				
Test Pr.		**SA_{best}**	**SA_{avg}**	**t_{best} (CPU)**	**%GAP**	**LJG-HGA_{best}**	**LJG-HGA_{avg}**	**t_{best} (CPU)**	**%GAP**	**CGA_{best}**	**CGA_{avg}**	**t_{best} (CPU)**	**%GAP**	**Best Known**
A-n32-k5	32	784*	799	56	%0	784*	784*	4	%0	784*	784*	3	%0	784
A-n33-k5	33	661*	661*	21	%0	661*	661*	2	%0	661*	661*	3	%0	661
A-n33-k6	33	742*	742.66	91	%0	742*	742*	3	%0	742*	742*	3	%0	742
A-n34-k5	34	778*	778.33	35	%0	778*	778*	7	%0	778*	778*	4	%0	778
A-n36-k5	36	799*	809.66	205	%0	803	806.1	21	%0.50	799*	799*†	8	%0	799
A-n37-k5	37	669*	675	104	%0	669*	669*	6	%0	669*	669*	5	%0	669
A-n37-k6	37	951	962	184	%0.21	949*	949*	6	%0	949*	949*	6	%0	949
A-n38-k5	38	730*	732.66	93	%0	730*	731.2	25	%0	730*	730*†	6	%0	730
A-n39-k5	39	825	829.66	48	%0.36	825	825	9	%0.36	822*	823†	15	%0	822
A-n39-k6	39	833	835	98	%0.24	831*	833	29	%0	831*	832.3†	114	%0	831
A-n44-k7	44	939	940	50	%0.21	937*	938.5	25	%0	937*	938.7	14	%0	937
A-n45-k6	45	953	961.33	75	%0.95	948	954.1	201	%0.42	948	951.3†	111	%0.42	944
A-n45-k7	45	1153	1172.33	220	%0.61	1146*	1146*	16	%0	1146*	1146*	9	%0	1146
A-n46-k7	46	917	940	192	%0.32	917	917.1	8	%0.32	914*	914*†	11	%0	914
A-n48-k7	48	1100	1100.33	60	%2.52	1074	1079.6	64	%0.09	1073*	1073.3†	26	%0	1073
A-n53-k7	53	1017	1026.33	347	%0.69	1017	1019.8	37	%0.69	1017	1017†	25	%0.69	1010
A-n54-k7	54	1167*	1171	589	%0	1167*	1169.3	62	%0	1167*	1167*†	12	%0	1167
A-n55-k9	55	1073*	1099	340	%0.46	1078	1082	54	%0.46	1075	1075†	34	%0.18	1073
A-n60-k9	60	1362	1371	74	%0.59	1354*	1358.1	275	%0	1354*	1357†	280	%0	1354
A-n62-k8	62	1315	1327.33	255	%2.10	1283	1284	77	-%0.38	1262	1281†	210	-%2.01	1288
A-n63-k9	63	1627	1645.33	355	%0.68	1627	1633.3	408	%0.68	1627	1633.7	145	%0.68	1616
A-n63-k10	63	1319	1327	157	%0.38	1320	1326	434	%0.45	1318	1324.2†	148	%0.30	1314
A-n64-k9	64	1429	1435	439	%2.00	1416	1424.2	159	%1.07	1415	1418.4†	126	%0.99	1401
A-n65-k9	65	1184	1198	428	%0.85	1181	1183.3	85	%0.59	1178	1178.1†	44	%0.34	1174
A-n69-k9	69	1170	1181.33	732	%0.94	1171	1173	240	%1.03	1168	1169.6†	59	%0.77	1159
A-n80-k10	80	1738	1751.33	940	-%1.44	1740	1746.6	742	-%1.30	1738	1745.8†	331	-%1.42	1763

Table 7. Christofides. Mingozzi and Toth VRP Test Instances

		SA			LJG-HGA				CGA				
Test Pr.	**N**	**SA_{best}**	**SA_{avg}**	**t_{best} (CPU)**	**% GAP**	**LJG-HGA_{best}**	**LJG-HGA_{avg}**	**t_{best} (CPU)**	**% GAP**	**CGA_{best}**	**CGA_{avg}**	**t_{best} (CPU)**	**% GAP**	**Best Known**
vrpnc1	50	524.61*	530.6555	223	%0	524.61*	524.61*	13	%0	524.61*	524.61*	16	%0	524.61
vrpnc2	75	859.0439	864.3108	898	%2.85	841.97	845.39	240	%0.80	835.89	838.65†	126	%0.07	835.26
vrpnc3	100	830.3448	832.17	669	%0.51	827.39	830.90	2050	%0.15	828.42	831.07	1918	%0.28	826.14
vrpnc4	150	1052.7374	1064.704	2622	%2.36	1043.37	1047.47	3653	%1.45	1041.73	1044.87†	1460	%1.29	1028.42
vrpnc5	199	1332.81	1352.619	3727	%3.22	1328.97	1339.87	3732	%2.91	1320.66	1342.34	3746	%2.27	1291.29
vrpnc11	120	1133.8194	1164.09	1626	%8.80	1042.11*	1042.73	267	%0	1042.11*	1042.11*†	81	%0	1042.11
vrpnc12	100	876.8677	884.1366	1790	%7.00	822.33	822.33	216	%0.33	821.46	822.12†	200	%0.23	819.56

*: Best known and/or the optimal solution found, †: CGA performed superior to SA, **BOLD:** Improved best known solution

Table 8. The Standard Costs and 26 Demands of Points

Demand Points	Standard Costs ($)	Demands (pallet)
Eskişehir (Depot)	0	0
Ankara	490	10
Sivas	1450	15
Erzurum	2350	25
Adana	1400	25
Kilis	1990	12
Amasya	1200	10
Samsun	1370	7
Erzincan	1960	6
Manisa	820	11
İzmir	875	22
Aydın	915	7
Konya	720	9
Antalya	880	13
Çankırı	800	4
Çorum	1050	3
Tokat	1350	2
Kahramanmaraş	1800	6
Gaziantep	1940	11
Afyonkarahisar	370	8
Burdur	700	5
Ordu	1460	3
Giresun	1610	4
Gümüşhane	1800	3
Bayburt	1880	2
Kütahya	200	5
Uşak	640	7

in Turkey, which is considered real life parameters, is presented in Table 8 in this section. Demand points are the cities in Turkey. Depot one of the combi boiler producer is in Eskisehir. This problem is solved with proposed CGA and results are given in Table 9 where the costs according to both minimum total distance and minimum total real life costs are there for presenting the difference between them.

The model introduced in this paper, with 26 demand points and one depot, was solved via proposed CGA algorithm. The vehicle capacity is 50, stop by cost is $30, and fuel cost per kilometer is $ 0.25 in these problems. Five routes were determined by two models. The routes are given in Table 10. Although the distance in the introduced model is longer than the model which consider the minimum total distance, the total cost of introduced model is less than the other. This is because prices of standard routes are determined under competitive conditions.

Table 9. Results of the Total Cost According to Minimum Total Distance and Minimum Real Life Costs

	Costs				
	Distance (km)	**Standard route costs**	**Deviation costs**	**Stop by costs**	**Total cost**
Considering minimum total distance	4862	8105	245.25	630	8980.25
Considering minimum real life costs	5235	7535	400.5	630	8565.5

Table 10. Routes according to minimum total distance and minimum real-life costs

	Routes
Considering minimum total distance	Eskişehir-Kütahya-Burdur-Antalya-Adana
	Eskişehir-Afyon-Konya-Kahramanmaraş-Gaziantep -Kilis
	Eskişehir-Samsun-Ordu-Giresun-Gümüşhane-Bayburt-Erzurum-Erzincan
	Eskişehir-Ankara-Çankırı-Çorum-Amasya-Tokat-Sivas
	Eskişehir-Uşak –Manisa-İzmir-Aydın
Considering minimum real life costs	Eskişehir-Samsun-Ordu-Giresun-Gümüşhane-Bayburt-Erzurum-Erzincan
	Eskişehir-Çankırı-Sivas-Kahramanmaraş-Gaziantep-Kilis
	Eskişehir-Kütahya-Afyon –Burdur-Antalya-Konya
	Eskişehir-Ankara-Çorum-Amasya-Tokat-Adana
	Eskişehir-Uşak-Manisa-İzmir-Aydın

CONCLUSIONS

Many VRPs have been considered in the literature and some OVPRs have also been studied, which consider the minimum distance as an objective function. However, in real life OVRP, there are different costs namely standard route cost, stop by cost, and deviation costs. When we examined the OVRP of combi boiler producer's goods in Eskisehir, we saw that the cost was based not only on the distance but also on the competitive price of other third party logistics firms. Some standard routes and prices are defined for main cities by considering highway and costs, road conditions, etc. If there are delivery points except these standard routes, the deviation cost is added to the cost. Also, if there are stop points on the standard route, stop by costs are occurred. Our motivation is to set a mathematical model and solve this problem for companies that hire vehicles from third party logistic firms and to route them by considering these real life costs. Moreover, no prior studies have examined these real life costs in the literature. For instance; Dasdemir et al. [29] considered the minimization of the total travelled distance, Dutta et al. [30] studied minimizing the operating costs and minimizing the carbon emission due to fuel consumption by the service vehicles, Hashemi et al. [28] considered the minimization of the total transportation costs, Niu et al. [36] studied on OVRP to minimize the total cost of fuel emissions cost and the driver wages, etc. Therefore, this study presents a new and precise model for companies that meet real life costs such as standard routing cost, stop by cost, and deviation cost.

In this paper, firstly, the mathematical model was set and solved with GAMS software for demand point 5 and 10. Secondly, due to the NP hard structure of model, the Civilized Genetic Algorithm was presented by improving the genetic algorithm. The efficiency of CGA was proved on test problems. Finally, the real life problem with 26 demand points was solved for delivering combi boiler producer in Turkey via proposed CGA. Results demonstrate that the total cost of model that takes real life costs into account is less than the model that takes minimum distance into account, because of competitive standard route prices.

In our future research, we intend to determine routes for close up vehicle routing problems and for delivery and pickup vehicle routing problems. Also, model parameters such as costs, capacity, etc. can be fuzzy numbers to cope with the inherent uncertainties of the model. Other metaheuristic algorithms such as bee colony, simulated annealing and tabu search can be used for the studied problem.

ACKNOWLEDGEMENTS

This work supported by the Anadolu University in Eskisehir, Turkey under projects no: 1505F515 at 2015-2018.

AUTHORSHIP CONTRIBUTIONS

Authors equally contributed to this work.

DATA AVAILABILITY STATEMENT

The authors confirm that the data that supports the findings of this study are available within the article. Raw data that support the finding of this study are available from the corresponding author, upon reasonable request.

CONFLICT OF INTEREST

The author declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

ETHICS

There are no ethical issues with the publication of this manuscript.

REFERENCES

[1] Dantzig GB, Ramser JH. The truck dispatching problem. Manag Sci 1959;6:80–91. [CrossRef]

[2] Ombuki B, Nakamura M, Maeda O. A hybrid search based on genetic algorithms and tabu search for vehicle routing. Proceedings of 6th IASTED International Conference on Artificial Intelligence and Soft Computing; 2002 May; Canada. 2002. pp. 176–181.

[3] Awad H, Elshaer R, AbdElmo'ez A, Nawara, G. An effective genetic algorithm for capacitated vehicle routing problem. Proceedings of the International Conference on Industrial Engineering and Operations Management; 2018 Mar 6-8; Bandung, Indonesia. IEOM Society International; 2018. pp. 374–384.

[4] Cordeau JF, Gendreau M, Laporte G. A tabu search heuristic for periodic and multi-depot vehicle routing problems. Netw Int J 1997;30:105–119. [CrossRef]

[5] Brandão J. A tabu search algorithm for the open vehicle routing problem. Eur J Oper Res 2004;157:552–564. [CrossRef]

[6] Baños R, Ortega J, Gil C, Fernández A, de Toro F. A simulated annealing-based parallel multi-objective approach to vehicle routing problems with time windows. Expert Syst Appl 2013;40:1696–1707. [CrossRef]

[7] Kahar NHA, Zobaa, AF. Application of mixed integer distributed ant colony optimization to the design of undamped single-tuned passive filters based harmonics mitigation. Swarm Evol Comput 2018;44:187–199. [CrossRef]

[8] Goel R, Maini R. Evolutionary ant colony algorithm using firefly based transition for solving vehicle routing problems: EAFA for VRPs. Int J Swarm Intell Res 2019;10:46–60. [CrossRef]

[9] Yi JH, Wang J, Wang GG. Using Monarch butterfly optimization to solve the emergency vehicle routing problem with relief materials in sudden disasters. Open Geosci 2019;11:391–413. [CrossRef]

[10] Schrage L. Formulation and structure of more complex/realistic routing and scheduling problems. Netw Int J 1981;11:229–232. [CrossRef]

[11] Sariklis D, Powell S. A heuristic method for the open vehicle routing problem. J Oper Res Soc 2000;51:564–573. [CrossRef]

[12] Fu Z, Eglese R, Li LY. Corrigendum to the paper: A new tabu search heuristic for the open vehicle routing problem. J Oper Res Soc 2006;57:1018. [CrossRef]

[13] Tarantilis CD, Ioannou G, Kiranoudis C, Prastacos G. Solving the open vehicle routeing problem via a single parameter metaheuristic algorithm. J Oper Res Soc 2005;56:588–596. [CrossRef]

[14] Repoussis PP, Tarantilis CD, Ioannou G. The open vehicle routing problem with time windows. J Oper Res Soc 2007;58:355–367. [CrossRef]

[15] Pisinger D, Ropke S. A general heuristic for vehicle routing problems. Comput Oper Res 2007;34:2403–2435. [CrossRef]

[16] Li F, Golden B, Wasil EA. The open vehicle routing problem: Algorithms, large-scale test problems, and computational results. Comput Oper Res 2007;34:2918–2930. [CrossRef]

[17] Pessoa A, De Aragao MP, Uchoa E. Robust Branch-Cut-and-Price Algorithms for Vehicle Routing Problems. In: Golden B, Raghavan S, Wasil E, editors. The vehicle routing problem: Latest advances and new challenges. 1st ed. Boston, MA: Springer; 2008. p. 297–325. [CrossRef]

[18] Derigs U, Reuter K. A simple and efficient tabu search heuristic for solving the open vehicle routing problem. J Oper Res Soc 2009;60:1658–1669. [CrossRef]

[19] Fleszar K, Osman IH, Hindi KS. A variable neighbourhood search algorithm for the open vehicle routing problem. Eur J Oper Res 2009;195:803–809. [CrossRef]

[20] Zachariadis EE, Kiranoudis CT. An open vehicle routing problem metaheuristic for examining wide solution neighborhoods. Comput Oper Res 2010;37:712–723. [CrossRef]

[21] Erbao C, Mingyong L. The open vehicle routing problem with fuzzy demands. Expert Syst Appl 2010;37:2405–2411. [CrossRef]

[22] MirHassani SA, Abolghasemi N. A particle swarm optimization algorithm for open vehicle routing problem. Expert Syst Appl 2011;38:11547–11551. [CrossRef]

[23] Li X, Leung SC, Tian P. A multistart adaptive memory-based tabu search algorithm for the heterogeneous fixed fleet open vehicle routing problem. Expert Syst Appl 2012;39:365–374. [CrossRef]

[24] Marinakis Y, Marinaki M. A bumble bees mating optimization algorithm for the open vehicle routing problem. Swarm Evol Comput 2014;15:80–94. [CrossRef]

[25] Şevkli AZ, Güler B. A multi-phase oscillated variable neighbourhood search algorithm for a real-world open vehicle routing problem. Appl Soft Comput 2017;58:128–144. [CrossRef]
[26] Soto M, Sevaux M, Rossi A, Reinholz A. Multiple neighborhood search, tabu search and ejection chains for the multi-depot open vehicle routing problem. Comput Ind Eng 2017;107:211–222. [CrossRef]
[27] Brandão J. Iterated local search algorithm with ejection chains for the open vehicle routing problem with time windows. Comput Ind Eng 2018;120:146-159. [CrossRef]
[28] Hashemi S, Salari M, Ranjbar M. Multi-trip open vehicle routing problem with time windows: A case study. Int J Ind Eng 2020;27.
[29] Dasdemir E, Testik MC, Öztürk DT, Şakar CT, Güleryüz G, Testik ÖM. A multi-objective open vehicle routing problem with overbooking: Exact and heuristic solution approaches for an employee transportation problem. Omega 2022;108:102587. [CrossRef]
[30] Dutta J, Barma PS, Mukherjee A, Kar S, De T. A hybrid multi-objective evolutionary algorithm for open vehicle routing problem through cluster primary-route secondary approach. IntJ Manag Sci Eng Manag 2022;17:132–146. [CrossRef]
[31] Sun L, Pan QK, Jing XL, Huang JP. A light-robust-optimization model and an effective memetic algorithm for an open vehicle routing problem under uncertain travel times. Memetic Comput 2021;13:149–167. [CrossRef]
[32] Yu NK, Jiang W, Hu R, Qian B, Wang L. learning whale optimization algorithm for open vehicle routing problem with loading constraints. Discrete Dyn Nat Soc 2021;2021:8016356. [CrossRef]
[33] Hosseinabadi AAR, Zolfagharian A, Alinezhad P. An Efficient Hybrid Meta-Heuristic Algorithm for Solving the Open Vehicle Routing Problem. In: Recent developments and the new direction in soft-computing foundations and applications. New York: Springer; 2021. p. 257–274. [CrossRef]
[34] Ruiz E, García-Calvillo I, Nucamendi-Guillén S. Open vehicle routing problem with split deliveries: Mathematical formulations and a cutting-plane method. Oper Res 2020;22:1017–1037. [CrossRef]
[35] López-Sánchez AD, Hernández-Díaz AG, Vigo D, Caballero R, Molina J. A multi-start algorithm for a balanced real-world open vehicle routing problem. Eur J Oper Res 2014;238:104–113. [CrossRef]
[36] Niu Y, Yang Z, Chen P, Xiao J. Optimizing the green open vehicle routing problem with time windows by minimizing comprehensive routing cost. J Clean Prod 2018;171:962–971. [CrossRef]
[37] Li Y, Soleimani H, Zohal M. An improved ant colony optimization algorithm for the multi-depot green vehicle routing problem with multiple objectives. J Clean Prod 2019;227:1161–1172. [CrossRef]
[38] Vincent FY, Jewpanya P, Redi AP. Open vehicle routing problem with cross-docking. Comput Ind Eng 2016;94:6–17. [CrossRef]
[39] Miller CE, Tucker AW, Zemlin RA. Integer programming formulation of traveling salesman problems. Journal ACM 1960;7:326–329. [CrossRef]
[40] Liu R, Jiang Z, Geng N. A hybrid genetic algorithm for the multi-depot open vehicle routing problem. OR Spect 2014;36:401–421. [CrossRef]
[41] Holland JH. Adaptation in Natural and Artificial Systems: An introductory Analysis with Applications to Biology, Control, and Artificial intelligence. Michigan: University of Michigan Press; 1975.
[42] Katoch S, Chauhan SS, Kumar V. A review on genetic algorithm: past, present, and future. Multimed Tools Appl 2021;80:8091–8126. [CrossRef]
[43] Jebari K, Madiafi M. Selection methods for genetic algorithms. Int J Emerg Sci 2013;3:333–344.
[44] Soon GK, Guan TT, On CK, Alfred R, Anthony P. A comparison on the performance of crossover techniques in video game. In Proceedings of the 2013 IEEE International Conference on Control System, Computing and Engineering; 2013 Nov 29 - Dec 1; Penang, Malaysi. IEEE; 2013. pp. 493–498. [CrossRef]
[45] Elmihoub T, Hopgood AA, Nolle L, Battersby A. Performance of hybrid genetic algorithms incorporating local search. In Proceedings of 18th European Simulation Multiconference; 2004 Mar; Erlangen, Germany. Gruner Duck; 2004. pp. 154–160. [CrossRef]
[46] Espinoza FP, Minsker BS, Goldberg DE. Performance evaluation and population reduction for a self adaptive hybrid genetic algorithm (SAHGA). In Proceedings of Genetic and Evolutionary Computation Conference; 2003 July; Heidelberg, Berlin. Springer; 2003. pp. 922–933. [CrossRef]
[47] Hedar AR, Fukushima M. Simplex coding genetic algorithm for the global optimization of nonlinear functions. In Proceedings of Multi-Objective Programming and Goal Programming; 2003; Heidelberg, Berlin. Springer; 2003. pp. 135–140. [CrossRef]
[48] Tan KC, Li Y, Murray-Smith DJ, Sharman KC. System identification and linearisation using genetic algorithms with simulated annealing. In Proceedings of the First International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications; 1995; Sheffield, UK. IEET; 1995. pp. 164–169. [CrossRef]
[49] Guo C, Wang C, Zuo X. A genetic algorithm based column generation method for multi-depot electric bus vehicle scheduling. In Proceedings of the Genetic and Evolutionary Computation Conference Companion; 2019 Jul 13-17; Prague, Czech Republic. 2019. pp. 367–368. [CrossRef]

[50] Luo Y, Pan Y, Li C, Tang H. A hybrid algorithm combining genetic algorithm and variable neighborhood search for process sequencing optimization of large-size problem. Int J Comput Integr Manuf 2020;33:962–981. [CrossRef]

[51] Augerat P, Belenguer JM, Benavent E, Corberan A, Naddef D, Rinaldi G. Computational results with a branch and cut code for the capacitated vehicle routing problem. Available at: https://www.osti.gov/etdeweb/servlets/purl/289002. Accessed on May 10, 2024.

[52] Christofides N, Mingozzi A, Toth P. The Vehicle Routing Problem. In: Christofides N, Mingozzi A, Toth P, Sandi C, editors. Combinatorial optimization. Chichester: Wiley; 1979. p. 315–338.