Süleyman Demirel Üniversitesi
Fen Bilimleri Enstitüsü Dergisi
Cilt 25, Sayı 2, 283-296, 2021

Süleyman Demirel University
Journal of Natural and Applied Sciences
Volume 25, Issue 2, 283-296, 2021

DOI: 10.19113/sdufenbed.817282

Solving Capacitated Vehicle Routing Problem (CVRP) for the Environments with Circular Obstacles

Mehmet KARAKOÇ*[1], Aybars UĞUR[2]

[1]Alanya Hamdullah Emin Paşa University, Faculty of Engineering, Department of Computer Engineering, 07400, Antalya, Turkey
[2]Ege University, Faculty of Engineering, Department of Computer Engineering, 35100, İzmir, Turkey

(Alınış / Received: 28.10.2020, Kabul / Accepted: 16.05.2021, Online Yayınlanma / Published Online: 15.08.2021)

Keywords
Capacitated vehicle routing problem (CVRP),
Circular obstacles,
Meta-heuristic,
Genetic algorithms,
Local search,
Simulation

Abstract: In *capacitated vehicle routing problem* (CVRP), a fleet of vehicles with a certain capacity starts at a central depot and returns to this starting point after serving some customers by using the optimal set of routes with the minimum cost. However, in real-life, environments may include obstacles such as holes, machines, or trees of different sizes. In this research, a CVRP extension is proposed that is the problem of the classical one for the environments with various sized circular obstacles. To solve this problem, a hybrid *meta-heuristic* algorithm based on *genetic algorithms* improved by a *local search* was developed. Additionally, a *visual simulation tool* was designed to place obstacles and locations in the working space. The developed algorithm was tested for different customer-obstacle counts and obstacle sizes with various *obstacle occupancies* in the environment. The results obtained are presented and the problem's potential applications are discussed.

Kapasiteli Araç Rotalama Probleminin (KARP) Dairesel Engeller İçeren Ortamlar için Çözümü

Anahtar Kelimeler
Kapasiteli araç rotalama problemi (KARP),
Dairesel engeller,
Üst-sezgi,
Genetik algoritmalar,
Yerel arama,
Benzetim

Özet: *Kapasiteli araç rotalama probleminde* (KARP), belirli kapasitedeki bir araç filosu merkezî bir depodan harekete geçer ve en düşük maliyetli en uygun rota kümesini kullanarak birtakım müşterilere hizmet verip bu başlangıç noktasına geri döner. Gerçek hayatta ise ortamlar farklı büyüklüklerdeki delik, makine veya ağaç gibi engeller içerebilmektedir. Bu çalışmada, klasik KARP'nin çeşitli büyüklüklerdeki dairesel engeller içeren ortamlar için genişletilmiş bir biçimi önerilmektedir. Bu problemi çözmek için *yerel arama* ile iyileştirilmiş *genetik algoritmalar* tabanlı melez bir *üst-sezgisel* algoritma geliştirilmiştir. Ek olarak, çalışma uzayına engeller ve konumlar yerleştirmek için bir *görsel benzetim aracı* tasarlanmıştır. Geliştirilen algoritma ortam üzerinde çeşitli engel doluluklarıyla farklı müşteri-engel sayıları ve engel büyüklükleri için sınanmıştır. Elde edilen sonuçlar sunulmuş ve problemin potansiyel uygulamaları tartışılmıştır.

1. Introduction

Vehicle routing problem (VRP) is both a linear programming and a combinatorial optimization problem proposed by Dantzig and Ramser [1] in 1959. This well-studied problem is widely encountered in *computer networks*, *demand-sensitive transportation systems*, *logistics*, *operational research*, and *supply-chain management* with many applications for a number of problems including (i) cleaning of buildings/streets, (ii) fuel/mail/milk distribution, (iii) garbage/waste collection, (iv) pickup and delivery of objects/packages/parcels, (v) routing of personal services, salespeople, school buses or unmanned vehicles, and (vi) shipment of goods. VRP has three main components: (1) *locations* distributed homogeneously or heterogeneously, (2) *requests* that are homogeneous or heterogeneous, and (3) *vehicles* that are identical or not. In VRP, it is aimed to meet customer requests with the minimum cost while satisfying all operational constraints. In *capacitated VRP* (CVRP), on the other hand, each vehicle is identical with a limited capacity and all requests are known and met [2]. The optimal set of routes R to be used by m vehicles is determined that are at the capacity q and the velocity v (Figure 1).

* Corresponding author: mehmet.karakoc@alanyahep.edu.tr

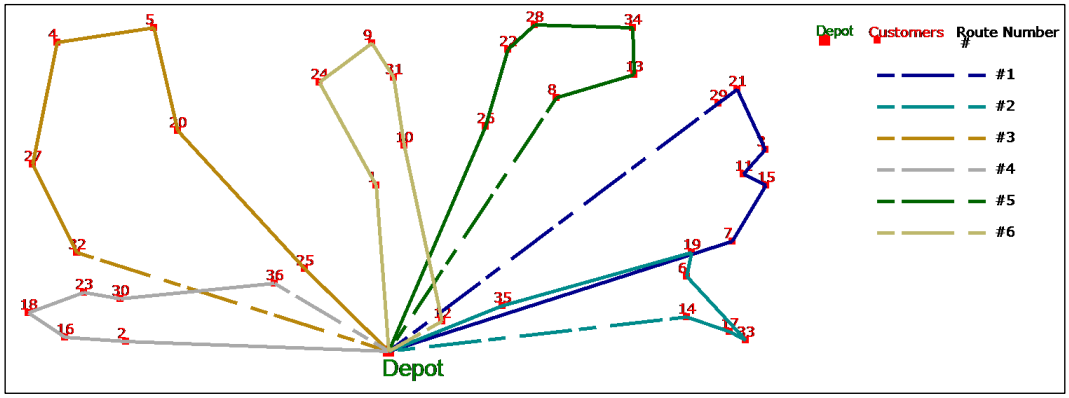


Figure 1. A simple CVRP solution for 36 customers

This vehicle fleet makes the deliveries through a central depot with the minimum cost *C* in order to meet all the requests of *n* customers distributed at different points on a geographical area.

In CVRP, depot-customer locations, the amounts of requests and vehicle capacity are known in advance, and the distance between any point pair is the cost between them. For the problem;

Four basic assumptions are as follows:

1. There is no request for the depot and it has enough goods stock to meet all requests
2. Each customer has only one request
3. Sufficient number of vehicles are available at the depot for transportation
4. The amounts of requests and vehicle capacity are defined as positive integers and vehicle capacity therefore may be utilized in full

Three hard-constraints are as follows:

1. Each vehicle starts at and returns to the depot after serving a subset of customers
2. Each customer is visited only once by one exact vehicle
3. The total load on any route cannot be over than the vehicle capacity

The output is the following:

1. The optimal set of routes at the minimum number while aiming the shortest or near-shortest total route length

In Figure 1, the dashed line on any route represents the last move between the last customer on that route and the depot. The solution consists of six routes in which six customers are assigned per each one.

In *traveling salesman problem* (TSP), a number of cities are given with the distances between all the city pairs and the shortest route is found that visits each city only once and ends at the starting point. In TSP, all locations have to be visited. When a traveling seller has the capacity to carry all requests, he/she can visit all cities in only one route. VRP is included between *bin packing problem* and TSP [3]. Therefore, it requires both assignment to vehicles and routing. In the *arc routing problem* (ARP), a postman has to visit all the roads of a city once and return to the starting point. In the *location routing problem*, the locations of depots are decided in addition to VRP. In VRP with time-windows, deliveries have time-intervals and each request is met within this time. In VRP with multiple routes, any vehicle may have more than one route. VRP with pickup and delivery requires the transportation of a set of goods from certain pickup locations to other delivery locations. In last-in-first-out (LIFO) VRP, on the other hand, any item to be delivered is the last item received. The periodic VRP consists of determining the optimal set of routes with the minimum cost for every day of a planning horizon given. Each customer is visited a required number of times while receiving the required amount of product every time. Moreover, in VRP with stochastic requests, a vehicle with a finite capacity leaves the depot with a full load and serves some customers whose requests are to be known when the vehicle arrives at them. In VRP with distance constraints, vehicles can travel up to the maximum distance allowed. VRP has many variations studied in literature.

Yücenur and Demirel [4] proposed a hybrid *meta-heuristic* method based on *genetic algorithms* (GA) and *ant colony optimization* (ACO) in solving multi-depot VRP. They used genetic clustering method for grouping customers into depots and ACO for routing. Wang and Lu [5] proposed a new hybrid GA with the optimal combination of cross-over and mutation probabilities. Yurtkuran and Emel [6] used a population-based hybrid algorithm and improved the solutions obtained with an iterative swap procedure as a new *local search* (LS). Fung et al. [7] developed many methods to transform a directional/non-directional ARP to VRP as its counterpart. Longo et al. [8] used the transformation into CVRP in solving capacitated ARP.

Luo and Chen [9] presented an improved solution method and its multi-phase model to solve the multi-depots VRP with/without time-windows in which depots are considered as the centroids of the clusters for all customers. They did clustering analyses to generate new clusters based on the best solution achieved by the preceding process and inherited the

improved path information to the new ones. Stanojević et al. [10] developed a new heuristic for CVRP that combines routes to generate better ones. Tlili et al. [11] studied the CVRP with distance constraints and proposed a hybrid swarm-based *meta-heuristic* combining the *particle swarm optimization* (PSO) and the variable neighborhood search. Marinakis et al. [12] introduced a new hybrid algorithm based on PSO to solve the VRP with stochastic requests. Moreover, Bortfeldt [13] considered the CVRP with 3D (*three-dimensional*) load and additional packing constraints, as its generalization where the requests are 3D and stackable, and introduced a hybrid algorithm including tree-search algorithms for assignment and tabu search for routing. Cacchiani et al. [14] addressed the periodic VRP as a generalization of VRP in which the number of routes per day cannot be over than the number of available vehicles and a few days of planning are taken into consideration for routing. Hà et al. [15] considered the generalized VRP with a flexible fleet size in which the number of vehicles is a decision variable. To minimize the daily routing cost, the appropriate fleet size may have been determined.

Patle et al. [16] presented the *mobile robot (MR) navigation* techniques used while considering the classical approaches such as *artificial potential field* (APF), *cell decomposition* and *roadmap approach*, and reactive approaches such as ACO, *artificial bee colony*, *bacterial foraging optimization*, *cuckoo search*, *firefly algorithm*, *fuzzy logic*, GA, *neural networks*, PSO and *shuffled frog leaping algorithm*. According to them, the reactive approaches are robust and widely used for MRs' *path planning* (PP). Patle et al. [17] illustrated the new dimension on *MR navigation* for the dynamic and static environments by the matrix-binary codes based GA and compared their controller with the other navigational controllers such as ACO, *artificial neural networks* and *fuzzy logic* for the same environmental conditions.

Nguyen and Le [18] developed and implemented a new PP method for MRs that could find the target with the near-shortest path length while avoiding some infinite loop traps of many obstacles in unknown environments, generate the MRs' trajectory both in dynamic and static environments by updates on the information of the onboard sensors. Sudhakara et al. [19] proposed an *enhanced APF* that generates the trajectories for wheeled MRs' mobility. The repulsive potential is built by repulsive function for discretizing outline of an arbitrarily shaped obstacle with its boundary points. Hassani et al. [20] proposed an algorithm based on a turning point strategy and free segments to solve the *robot PP problem* in a static environment with a *sliding mode* to control the stabilization of an *autonomous MR* to track a desired trajectory.

Sun et al. [21] presented a combinatorial approach for PP which aims to cover mission domains with different task-periods while guaranteeing both (i) *obstacle avoidance* and (ii) minimizing the number of robots used. The algorithm (i) deploys the sensors in the region to satisfy coverage requirements with the minimum cost, (ii) solves the TSP to obtain the closed path's frame, (iii) divides this path into the least segments while satisfying the coverage period constraints, and (iv) generates the closed route for each robot on the basis of this path's divided segments.

Bai et al. [22] designed a *PP algorithm* to minimize the time for a vehicle to travel between two given locations through the drift field while avoiding any obstacle and proposed an algorithm to assign the targets to the vehicles by using only local communication. Wang et al. [23] constructed a safety model of *obstacle avoidance* by analyzing a human driver's *obstacle avoidance* behavior. Then, based on this model, they improved the APF method and rebuilt the repulsive field range of obstacles, generated a collision-free path for *autonomous driving vehicles* based on the enhanced APF.

In this research, CVRP was addressed for the *environments with circular obstacles* (CVRP_EWCO) as its special case in real-life that has not been studied before as known from literature. For the solution, a hybrid *meta-heuristic* algorithm based on GA improved by a LS was developed. Additionally, a *visual simulation tool* was designed to place circular obstacles and depot-customer locations in the working space. Two main contributions to literature here are *as follows*:

1. defining the CVRP extension that will be able to be used on the enviroments (obstacles in different forms can be approximated by a circle shape simply) encountered frequently in daily life
2. conducting comprehensive experimental study by developing a solution method

Real-life environments may include buildings, chairs, columns, containers, furniture, holes, lakes, machines, pools, rows, shelves, tables, trees or worksites inside of any place such as airport, bank, campus, facility, factory, fairground, forest, harbor, indoor, land, marketplace, room, working-area or workshop. Such an environment with obstacles requires *obstacle avoidance* for MRs, traveling sellers, underwater vehicles and such vehicles or people. Moreover, for the problems including the navigation, PP and routing of these vehicles, some paths may not be passable. In adapting the problem to such an environment, a solution method is proposed with the following two steps:

1. *distance calculation*: the shortest distances between point pairs are found;
   a. determining the *visibility cases* between point pairs by finding the obstacles on the potential/candidate line/segment between each point pair
   b. finding new distances between point pairs having obstacles between them by using guide points
2. *CVRP solution*: the problem transformed into the classical CVRP is solved;
   a. determining the paths between point pairs by using the weighted adjacency matrices
   b. applying the developed algorithm to CVRP by using the new path and distance information

This paper's rest is organized as follows. In Section 2, the studied problem is defined and the solution method proposed is described in detail. In Section 3, the performance of the developed algorithm is shown. In Section 4, the results obtained are discussed and the final thoughts are summarized.

2. Material and Method

In this section, all the details regarding the studied problem and the solution method proposed are given.

2.1 Problem definition

In CVRP, cost matrices are known or the costs for the given locations in 2D (*two-dimensional*) environments are calculated by the *Euclidean distance*. However, in the problem proposed, there are obstacles between these locations, vehicles cannot pass through the obstacles, and therefore the paths between these locations have to be found by taking into consideration the obstacles before determining the routes. The obstacles can be either circular or be modelled with circles covering them which is one of the easiest approaches.

The CVRP_EWCO is the problem of the classical *capacitated vehicle routing* on 2D environments with various sized circles. Any obstacle having multiple edges in real-life may generally be defined as a circle approximately. Therefore, the obstacles having various shapes were easily modelled in the form of circles enclosing themselves with the given centers and radius. In this problem, circles are placed in the environment of the classical CVRP that represent the obstacles at regular spaces between each other in addition to the depot-customer locations and the vehicles can neither pass through these circles nor move in a curvilinear manner. Each vehicle has to move between all the point pairs along its route with the shortest path by taking into consideration all the obstacles in the environment. In the solution, it may cause to longer paths for vehicles not to move through the guides at certain distances to the obstacles. This solution therefore only and particularly intends for developing convenient routes inside of the floor in e.g. factories through the potential movement points (an example in Figure 2) to be defined on movable area with an algorithm. Guides were used for the following reasons:

- reducing the number of calculations (*complexity of the solution*)
- When a vehicle is not defined as a point, vehicles occupy a 2D space and the routes that are at certain distances to the obstacles need to be determined. Furthermore, there may be narrow spaces in the environment and/or it may be essential to use the paths at certain distances to the obstacles in avoiding dangerous situations (e.g. hot objects) regarding the vehicles and obstacles.

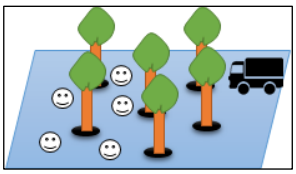


Figure 2. An environment with six (6) obstacles (*trees*) and five (5) customers (*faces*)

In this problem, it is assumed that the environment is known, 2D, limited, static and continuous; obstacles do not move; each location of the depot, a customer, or a guide is a point and the distance between any point pair is not known in advance. Each vehicle, defined as a point, can pass among the obstacles while moving on the continuous area outside the obstacles placed in the environment regularly. In the solution, each route includes the related locations that do not intersect with any obstacle. To illustrate this new problem, a sample is given in Figure 3 with its solution generated for one route in which "12 (3 × 4)" circular obstacles and "20 (4 × 5)" guides are placed in the environment regularly with the random locations of *one* depot and *four* customers.

In Figure 3, the locations of the depot, customers and guides are shown with a big square, little squares and little circles, respectively, with the related orders. The guides shown with arrows are placed in the environment regularly so that each guide pair on the same horizontal/vertical line can only see each other. Therefore, there is no intersection on the line between them with any obstacle (*visible to each other*). As seen, in the navigations between the locations that do not see each other directly, *when obstacles exist on the potential line between point pairs*, a new and passable path is determined on which the guides are visited to move to the target location from the current one.

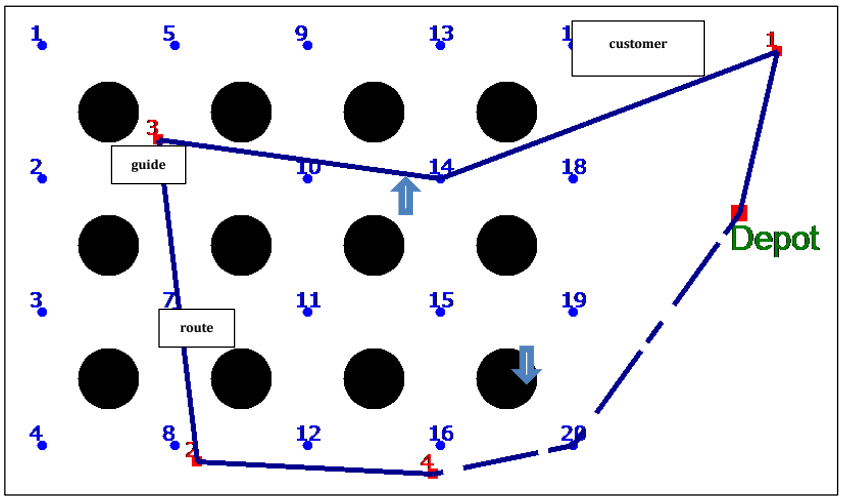


Figure 3. The shortest path in the environment with certain sized circular obstacles and guides for one route

2.2 Solving CVRP for the Environments with Circular Obstacles

In this subsection, the flow of the solution method proposed is shown and the way of obtaining the weighted adjacency matrices is described followed by the solution.

2.2.1 Generic solution method to the problem

When calculating the distance between any point pair, the intersections of the potential line between them with the obstacles are taken into consideration. After calculating the distances between all point pairs, the developed algorithm is applied to CVRP. The workflow of this method is given in Figure 4.

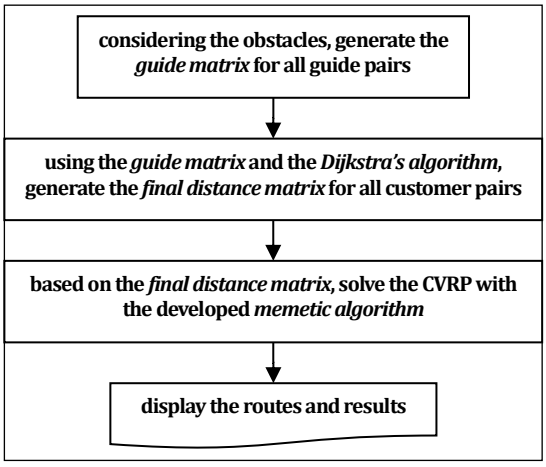


Figure 4. The workflow of the proposed method regarding the problem's solution

Each value of the *final distance matrix* including the distances between all customer pairs is generated in two steps: (1) the *guide matrix* is generated after calculating the distances between all guide pairs, and (2) based on this matrix, for each customer pair *who are not visible to each other*, the *straight-line distance matrix* is generated by determining the path between them and its total distance.

2.2.2 Creating straight-line distance matrices for customer pairs

On the CVRP_EWCO, the path between any customer pair *who are not visible to each other* is determined and the vehicles are allowed to move through the defined guides that are not so close to the obstacles. In Figure 5, a sample environment is given with *one* obstacle and *four* guides, and it is not possible to move between *C1 - C2* customer pair directly since an obstacle exists on the dashed line between them. The navigation between them can be done through the guide *G4*, a path that is longer than the *Euclidean distance*. This approach is employed since circular movements around obstacles may not be appropriate for each type of vehicle.

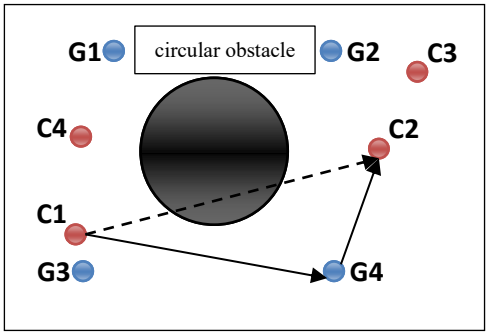


Figure 5. The shortest path between *C1 - C2* customer pair in the environment with a circular obstacle and *four* guides

To determine the intersections, the following cases between a circle and a line are considered: (i) no intersection, (ii) tangent line (*intersection at one point*), or (iii) secant (*intersection at two points*). When *one point intersection* becomes the case, the corresponding point is the tangent that belongs to the circle corresponding to the related obstacle. The ellipse that refers to the circle corresponding to the related obstacle is firstly located at the center and its semi-axes values (*half the width* (semi-major axis) and *half the height* (semi-minor axis)) are calculated. Quadratic parameters and discriminant are then

calculated. A *quadratic formula* and its *discriminant* are given in Equation (1) and Equation (2), respectively.

$$At^2 + Bt + C = 0 \quad (1)$$

$$B^2 - 4AC \quad (2)$$

The *formula* to find the values of t that satisfy the Equation (1) is given in Equation (3).

$$t = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A} \quad (3)$$

The number of real solutions to the Equation (3) depends on the discriminant as follows:

- discriminant < 0: There are no real solutions.
- discriminant = 0: There is one solution.
- discriminant > 0: There are two solutions.

While one solution is obtained if the *delta* (Δ: t in the Equation (3)) is zero, two solutions are obtained when it is positive. For the intersections, the roots are found from this value. The algorithm to determine the circular obstacles on the line between any point pair is given as pseudo-code in Table 1.

Table 1. Pseudo-code to determine the circular obstacles on the line between two points

Algorithm: determineObstaclesOnTheLine(Point p1, Point p2)
1. **for** *an obstacle* **in** *all the obstacles* in the environment **do**
2. **if** "the potential line between *p1 - p2* intersects with this *obstacle*" **then**
3. **return true** // no move between *p1 - p2* directly
4. **end if**
5. **end for**
6. **return false** // no intersection

In Table 1, when the result is *false*, the distance between *p1 - p2* becomes the *Euclidean distance* between them; otherwise, it is taken into consideration as "∞".

For the example in Figure 5, calculating the distances between all the guide pairs, the *guide matrix* of "4 × 4" (*four* guides) including the distances is generated only once. When calculating the distance between any customer pair *who are not visible to each other*, two rows and two columns are firstly added to the *guide matrix*; and then, the *straight-line distance matrix* of "6 × 6" including the distance between them and the distances between them and all the guide pairs is generated with the *visibility graph* approach. The matrix generated for *C1 - C2* customer pair is given in Table 2.

In Table 2, G_i and C_j refer to the guide location a and the customer location b with the i^{th} and j^{th} indices, respectively, and c_{ab} is the distance/cost between a and b. For each customer pair *who are not visible to each other*, a new *straight-line distance matrix* is generated by only changing two rows at the bottom and two columns at the right. Applying the *Dijkstra's shortest path algorithm* on it, the shortest path between the related customer pair passing through the guides and its total cost/distance are found.

Table 2. Straight-line distance matrix generated for *C1 - C2* customer pair

Location	G1	G2	G3	G4	C1	C2
G1	0	c_{G1G2}	c_{G1G3}	c_{G1G4}	c_{G1C1}	c_{G1C2}
G2	c_{G2G1}	0	c_{G2G3}	c_{G2G4}	c_{G2C1}	c_{G2C2}
G3	c_{G3G1}	c_{G3G2}	0	c_{G3G4}	c_{G3C1}	c_{G3C2}
G4	c_{G4G1}	c_{G4G2}	c_{G4G3}	0	c_{G4C1}	c_{G4C2}
C1	c_{C1G1}	c_{C1G2}	c_{C1G3}	c_{C1G4}	0	c_{C1C2}
C2	c_{C2G1}	c_{C2G2}	c_{C2G3}	c_{C2G4}	c_{C2C1}	0

2.2.3 Creating final distance matrix

Based on the four customers (*C1*, *C2*, *C3*, *C4*) in the example in Figure 5, the calculations are repeated for six customer pairs (*C1 - C2*, *C1 - C3*, *C1 - C4*, *C2 - C3*, *C2 - C4*, *C3 - C4*) and the upper triangle of the *final distance matrix* of "4 × 4" (Table 3) is completed that includes only the distances between all the customer pairs. The problem is thus transformed into the classical CVRP.

Table 3. Final distance matrix

Customer	C1	C2	C3	C4
C1	0	c_{C1C2}	c_{C1C3}	c_{C1C4}
C2	c_{C2C1}	0	c_{C2C3}	c_{C2C4}
C3	c_{C3C1}	c_{C3C2}	0	c_{C3C4}
C4	c_{C4C1}	c_{C4C2}	c_{C4C3}	0

The *final distance matrix* in Table 3 does not include any distance related to any guide. The distance obtained by using the *straight-line distance matrix* for any customer pair only forms the corresponding element in the *final distance matrix*. For the solution, the developed algorithm is applied on this matrix.

2.3 Optimization based on GA improved by a LS

VRP is a NP-hard problem since the complexity (*solution time*) increases exponentially with the increase of the problem size. In solving such complex and hard problems, GA is a robust optimization technique as a global search method from the field of *evolutionary computing*. Although the best one is not guaranteed, it provides a good solution set. To overcome this problem and prevent getting stuck at *local optimum*, it was hybridized with a LS. The workflow of the developed *memetic algorithm* (the third step in Figure 4) is given in Figure 6.

The details of the basic components of the *memetic algorithm* [24] example in Figure 6 are given in the following subsections. As a hybrid algorithm based on GA improved by a LS (*2-opt*), Uğur [25] had used it to solve the special 3D TSP in which all cities and solution paths were on the surfaces of a cuboid while it was used in this research for a completely different CVRP extension with circular obstacles in 2D environment.

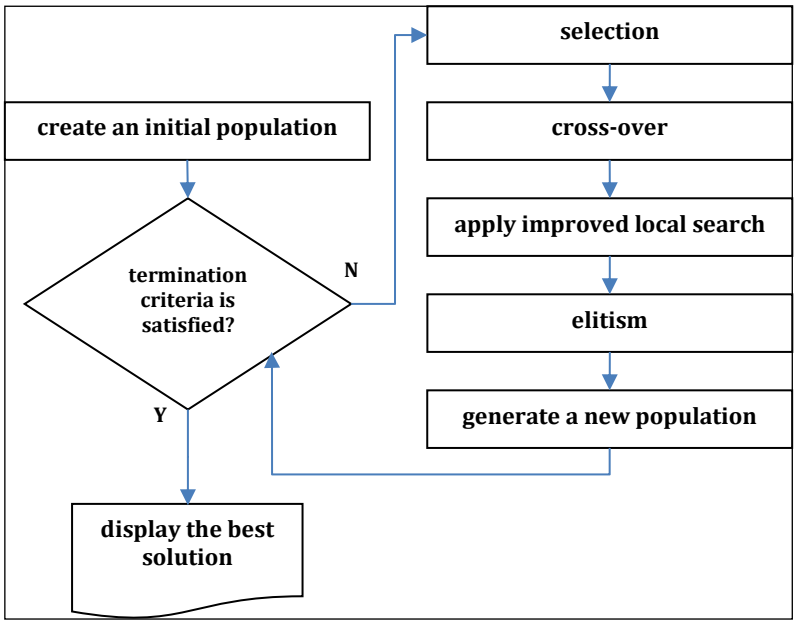


Figure 6. The workflow of the developed memetic algorithm

2.3.1 Chromosome representation

In the representation with one vector, each route separated from each other with the depot location 0 (*zero*) includes the orders of the customers on its own. Each customer takes place only once in each representation whose length may dynamically change. The representation of "{0 1 3 5 0 4 2 0 6}" consists of three routes given in Example 1:

Example 1:
Route#1: depot 1 3 5
Route#2: depot 4 2
Route#3: depot 6

In Example 1, the customers are clustered in three routes. The number of locations in the representation is the sum of the number of customers (n) and the number of routes (*zeros*). When each vehicle meets only one request, the number of locations in the representation becomes $2n$. The representation of an appropriate candidate solution requires each of its routes to satisfy the problem-specific constraints.

2.3.2 Initial population creation

The nearest neighbor algorithm was used to create an initial population and the genetic process was configured so that any unfeasible solution did not exist in the search space. After calculating the distances between all customer pairs, for each location, a certain number of locations that are the closest ones to this location are firstly determined; and then, pairs of consecutive points that are closer to each other can be generated.

2.3.3 Fitness evaluation

In cost calculation, it was aimed to minimize both (i) the number of routes and (ii) the total distance in the solution. The fitness of the i^{th} chromosome in the population is calculated with the multi-objective function F in Equation (4).

$$F(i) = \alpha \times m^i + \beta \times C^i \qquad (4)$$

where α is the coefficient for the number of routes (m) and β is the coefficient for the sum of the lengths of all routes (C).

2.3.4 Genetic operators

The single-point and double-point permutation cross-overs were applied to the parents selected and the offspring obtained were subjected to a LS.

2.3.5 Local search

LS is an effective technique that improves the current solution iteratively by using its neighborhoods. In 2-opt algorithm, a chromosome is broken from two points and recreated so that this range would be in reverse. This continues as long as the local minimum is not achieved. Using this algorithm, it is aimed to further shorten the total route length and better/fast converge to the solution while providing diversity, and to decrease the total distance with the location changes of inner-route and inter-routes. After the change in the sequence (*solution*), inter-routes loads are balanced so that the number of routes either becomes fixed or is decreased by one when two consecutive *zeros* exist in a new feasible chromosome as in Example 2:

Example 2:
Old : 0 1 3 5 0 *4 2 0* 6 → before 2-opt
New : 0 1 3 5 0 *0 2 4* 6 → 0 1 3 5 0 2 4 6

The improved LS (*2-opt algorithm*) is given as pseudo-code in Table 4 that was applied at each generation when it would become the case to improve the chromosome in preventing its slowness.

Table 4. Pseudo-code of the improved LS applied

Algorithm: applyImprovedLocalSearch(Chromosome chrome)
1. **while** "the improvement is the case for *chrome*" **do**
2. specify the best edge pair of "i, i + 1" and "j, j + 1" for *chrome*
3. **if** *dist*(i, i + 1) + *dist*(j, j + 1) > *dist*(i, j) + *dist*(i + 1, j + 1) **then**
4. exchange the edges and update the fitness of *chrome*
5. **end if**
6. **end while**

In Table 4, *i* and *j* refer to the genes in the *chrome* while *dist* is the matrix including the distances between all customer pairs.

2.3.6 Next generation creation

At each generation, *elitism* is firstly applied and the best two chromosomes are directly passed to the next one. Then, to maintain the population size (n), "$n - 2$" offspring are produced and the chromosomes obtained are exchanged with the parents starting with the worst one. To obtain better chromosomes, offspring who are not better than their parents are also given the chance to live.

2.3.7 Termination criteria

The algorithm execution is stopped when the maximum number of generations is reached.

3. Results

In this section, the details for data set and experimental setup are firstly given. Then, to show the performance of the developed algorithm, the experiments in the environments without obstacle are presented for benchmarking. Lastly, to show the applicability, the details in placing circular obstacles and the experiments conducted on the CVRP_EWCO are given.

3.1 Data set and experimental setup

The working space is considered as a land, map, or a scene that is in accordance with the dimensions (width/height) of the simulation tool of "554 × 554" unit^2 (Figure 7). Predetermined environment size allows to change the obstacle radius easily. When obstacles are small, their impacts may not be observed sufficiently. The scene is set as squared so that it can contain a certain number of obstacles.

The developed algorithm was tested as follows: (i) in the environments without obstacle by using a few CVRP instances from the VRPLIB library in literature, and (ii) on the CVRP_EWCO for different customer-obstacle counts and obstacle sizes with various *obstacle occupancies* where the obstacles are placed in the environment regularly. In the first experiments (Section 3.2), location data, the amounts of requests and vehicle capacity were used for each CVRP instance. These experiments were repeated with the same point sets. In other experiments (Section 3.4 and Section 3.5) on the CVRP_EWCO, random point sets were used while the amounts of requests and vehicle capacity were fixed. In these experiments, each customer request is 10 while the vehicle capacity is 250. For example, the number of routes becomes *four* for the solution of each problem including 100 customer locations since the total request is "1000 (100 × 10)". In the experiments in which the number of customers changes (Section 3.4), the number of routes in the solution may also change. These experiments were repeated with different point sets. For each run in each experiment, the number of routes and the total cost/distance, the number of generations in which the best value was found and the run-time at that moment, and the total run-time were saved. The GA parameters and their values used in all the experiments are given in Table 5.

Table 5. Test parameters and the related values

Parameter	Value
population size	100
maximum number of generations	1000
selection method	tournament
probabilities for cross-over and LS	0,80 – 0,05
α and β [26]	0,7 – 0,5

When using *the nearest neighbor algorithm* to create an initial population, the *nearest neighbor* value and the probability for applying this algorithm are 5 and 0,90, respectively.

For the developed algorithm, a computer graphics supported *visual simulation tool* was designed (a screenshot in Figure 7) and the experiments were conducted by using C# on Microsoft Visual Studio 2010 on a Windows 7 operating system installed Intel Pentium Dual CPU notebook at 2.13 GHz processor with 3GB main memory.

3.2 Testing the developed algorithm in the environments without obstacle

To show the statistical success of the developed algorithm, it was run on a few CVRP instances [27], 10 times per each one. The number of routes, the paths and the minimum/maximum distances were determined with average distance, average error (*the formula* in Equation (5)) and standard deviation, and vehicle occupancy (*the formula* in Equation (6)). The results obtained are given in Table 6.

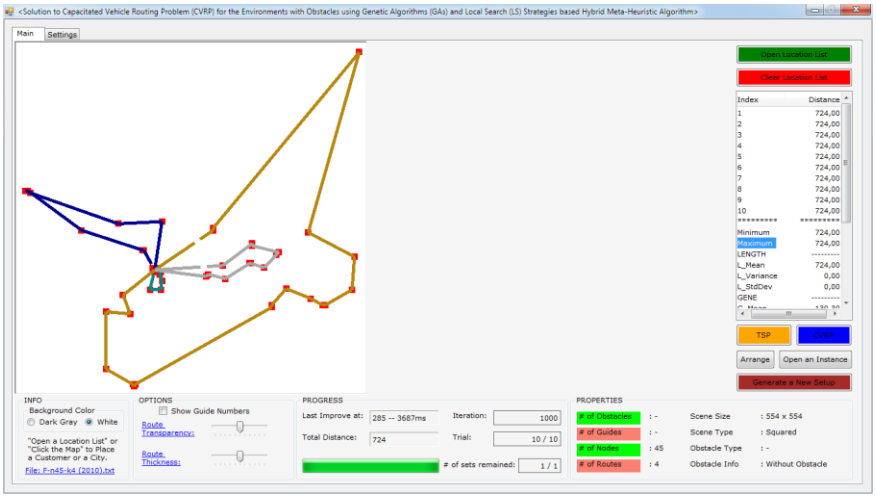


Figure 7. The computer graphics supported *visual simulation tool* designed

Table 6. Results regarding the classical CVRP's solution

Instance	best-known value [28]	n	m	q	(*)	(^)
att-n48-k4	40002	47	4	15	40002	0,73
A-n34-k5	778	33	5	100	778	0,92
A-n80-k10	1763	79	10	100	1763	0,94
B-n39-k5	549	38	5	100	549	0,88
E-n22-k4	375	21	4	6000	375	0,94
E-n23-k3	569	22	3	4500	569	0,75
E-n30-k3	534	29	3	4500	534	0,94
E-n51-k5	521	50	5	160	521	0,97
E-n101-k8	815	100	8	200	815	0,91
F-n45-k4	724	44	4	2010	724	0,90
F-n72-k4	237	71	4	30000	237	0,96

* (average distance obtained with the developed algorithm)
^ (vehicle occupancy)

$$error\ (\%) = \frac{mean - best\_known}{best\_known} \times 100 \qquad (5)$$

where *mean* is the average distance and *best_known* is the best-known value.

$$vehicle\ occupancy = Q \div (m \times q) \qquad (6)$$

where Q is the total load on vehicles (*total amount of the requests of all customers*), m is the number of vehicles, and q is the vehicle capacity.

In Table 6, n is the number of customers. As seen, in each run for each instance, the minimum number of routes/vehicles and the shortest total route length were found that have been reported in literature [28]. Since average errors and standard deviations are zero, the results obtained are fairly successful with the complete convergence provided.

3.3 Placement of circular obstacles

Before generating the depot-customer locations, the obstacles and guides to be distributed onto the environment homogeneously are generated at regular spaces in matrix form. The parameters considered here are *as follows*:

- W and H are the width and height of the working space
- R and C are the numbers of row and column of the matrix for the obstacles to be placed in this form
- *radius* and *diameter* are the radius and diameter of each obstacle

The algorithm to generate circular obstacles is given as pseudo-code in Table 7.

Table 7. Pseudo-code to generate circular obstacles

Algorithm: createCircularObstacles
1. $wR = W / (1 + R)$
2. **if** $wR <$ *diameter* **then**
3. **error:** "obstacles cannot be created with these inputs!"
4. **return**
5. **end if**
6. $hR = H / (1 + C)$
7. **if** $hR <$ *diameter* **then**
8. **error:** "obstacles cannot be created with these inputs!"
9. **return**
10. **end if**
11. clear the obstacle list
12. $x = -radius$
13. **for** $r = 1$ to R **do**
14. $x = x + wR$
15. $y = -radius$
16. **for** $c = 1$ to C **do**
17. $y = y + hR$
18. create the obstacle at "*x, y*" with *radius* and add it to the *list*
19. **end for**
20. **end for**

With the algorithm in Table 7, "$R \times C$" regular obstacles are generated and distributed onto the environment homogeneously (see Figure 3).

The obstacle radius is set so that it is not over than the maximum value in which the maximum number of obstacles can be placed in the environment without intersecting with each other. The gap between (i) each pair of consecutive obstacles on the same horizontal/vertical line, or (ii) an edge of the environment and the obstacles on the obstacle line that is the closest one to this edge depends on the environment size and obstacle count-size.

3.3.1 Creation of guides

Guides are placed in the environment so that each obstacle is seen from the *four* corners by the *four* guides that are the closest ones to this obstacle and the center of gravity of these guides becomes the center of the circle corresponding to this obstacle. The algorithm to generate guides is given as pseudo-code in Table 8.

Table 8. Pseudo-code to generate guides

Algorithm: createGuides
1. $rUpper = (R + 1) \times 2$
2. $cUpper = (C + 1) \times 2$
3. $wR = W / rUpper$
4. $hR = H / cUpper$
5. clear the guide list
6. $r = 1$
7. **while** $r < rUpper$ **do**
8. $x = r \times wR - 1$
9. $c = 1$
10. **while** $c < cUpper$ **do**
11. $y = c \times hR - 1$
12. create the guide at "x, y" and add it to the *list*
13. $c = c + 2$
14. **end while**
15. $r = r + 2$
16. **end while**

With the algorithm in Table 8, "$(R + 1) \times (C + 1)$" regular guides are generated for "$R \times C$" regular obstacles and distributed onto the environment homogeneously (see Figure 3).

3.3.2 Creation of depot-customer locations

As the depot-customer locations, the area outside the obstacles and guides is used. Therefore, firstly obstacles and guides, and random depot-customer locations are then generated and placed in the environment while taking into consideration the cases in which a point is both not enclosed by a circle and does not intersect with a guide.

3.3.3 Setup regarding the experimental study

In the experimental study on the CVRP_EWCO, obstacles and guides are placed in the environment regularly. Depot-customer locations are randomly generated with neither being placed on any obstacle nor overlapping with the guides. In the experiments, it was analyzed how the changes of customer-obstacle counts and obstacle size affected the paths and the total distances determined through the guides. However, the changes of the amounts of requests, vehicle capacity and the number of routes were not considered. In the experiments, to use the scene well, obstacles were placed in the environment so that there would occur enough space on it outside the obstacles and the gap between any obstacle pair was not so narrow. Additionally, in each experiment, the *obstacle occupancy* in the environment (*area covered by all the obstacles on it*) was also calculated.

3.4 Experiments for different customer counts

In these experiments, the number of customers is changed while the number of obstacles is "100 (10 × 10)" and the obstacle radius is 10 (*maximum value for 100 obstacles*). Firstly, 100 obstacles are placed in the environment, and the experiment is started with 100 customers and repeated 10 times. Then, for the same obstacle setup, 15 random locations are removed from the current set of customer locations and the next one is obtained. The experiments are conducted in this manner by reducing the number of customers until it becomes 10 for the related setup. This was not repeated for different customer placements. While the *obstacle occupancy* in the environment is 10,24%, the results with a sample are given in Figure 8, Graph 1 and Table 9.

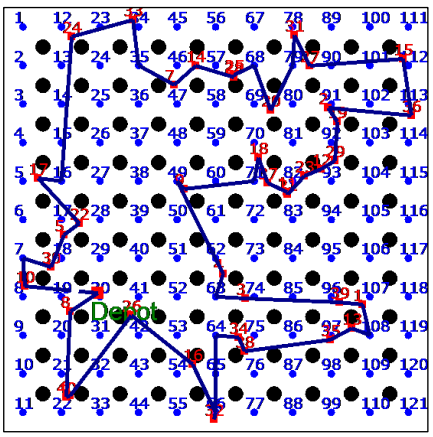
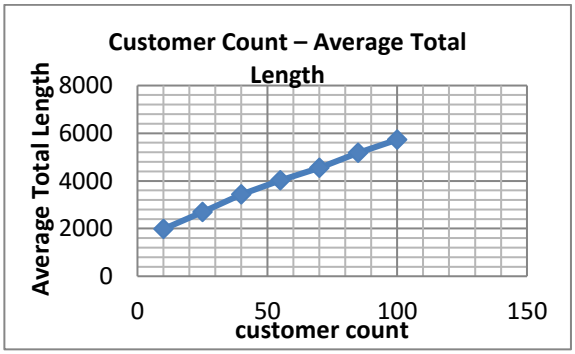


Figure 8. 25 customers, 100 obstacles and "121 (11 × 11)" guides in the environment for one route



Graph 1. Total route lengths for different customer counts

In Graph 1, for the same obstacle setup, with the increase of the number of customers in the environment, the complexity and the total distance increase linearly.

3.5 Experiments for various obstacle occupancies in the environment

In these experiments, while the number of customers was 100, both different obstacle counts and different obstacle sizes were used as the baseline.

3.5.1 Experiments for different obstacle counts

In these experiments, the number of obstacles is changed while the obstacle radius is 10 (*maximum value for 100 obstacles*). With the change of the number of obstacles, the locations of the obstacles and both the number and the locations of the guides change. Moreover, customer locations cannot be placed on a common area for all obstacle placements. Therefore, in each experiment, 100 customer locations are placed in the environment and the routes are determined. This was repeated 10 times for different customer placements. The results with a sample are given in Figure 9, Graph 2 and Table 10.

In Graph 2, with the increase in the number of obstacles, the complexity increases fairly. Moreover, the total distance may either increase or decrease. When the number of obstacles changes, the locations of them also change. Therefore, in these experiments, since customer locations cannot be a set of common points for each experiment, a certain set of customer locations was not used. The increase in the number of obstacles increases the number of guides and the path alternatives, and the total distance therefore might be decreased. For example, the increase of the number of guides has caused the total distance to decrease at the end of the graph.

3.5.2 Experiments for different obstacle sizes

In these experiments, the obstacle radius is changed while the number of obstacles is "100 (10 × 10)". Firstly, 100 customer locations are placed in the environment so that the largest sized 100 obstacles covering the largest area are placed in the environment. In each experiment, without changing the locations, the obstacle radius is reduced until it becomes the minimum. This was repeated 10 times for the same customer placement. The results with a sample are given in Figure 10, Graph 3 and Table 11.

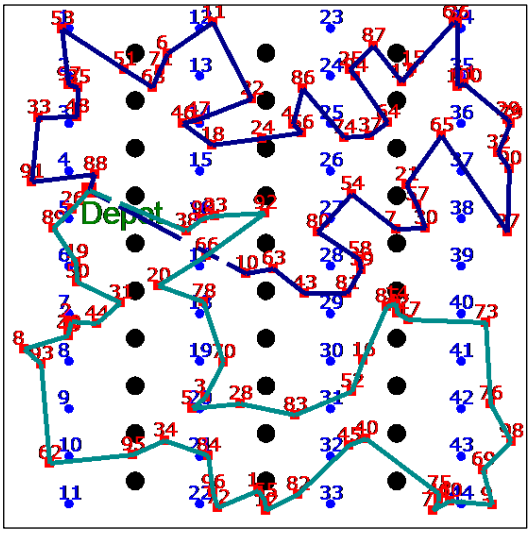


Figure 9. 100 customers, "30 (10 × 3)" obstacles and "44 (11 × 4)" guides in the environment for two routes (the vehicle capacity is 500)

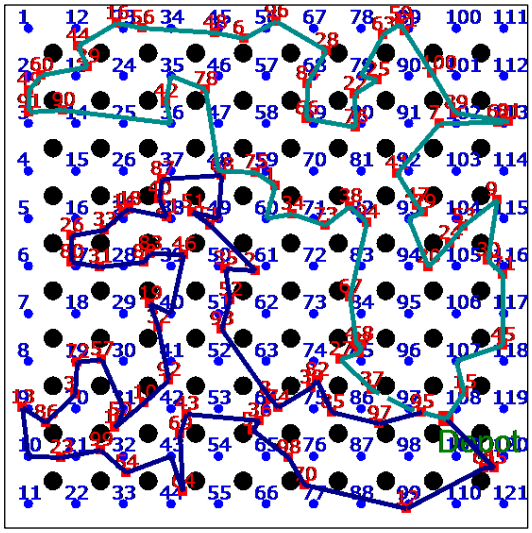
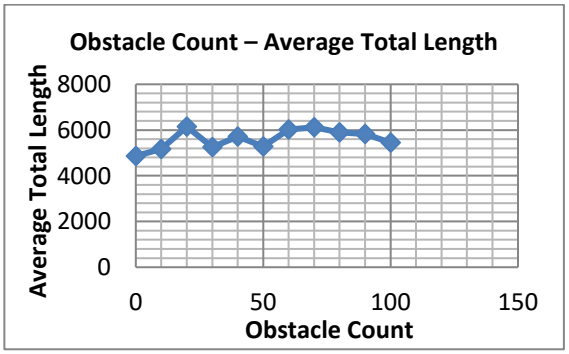
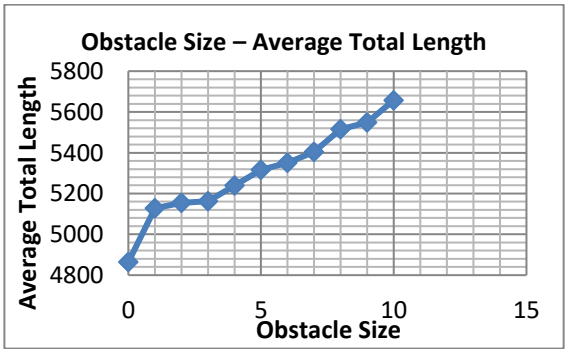


Figure 10. 100 customers, 100 obstacles and "121 (11 × 11)" guides in the environment for two routes (the vehicle capacity is 500 and the obstacle radius is 10)



Graph 2. Total route lengths for different obstacle counts



Graph 3. Total route lengths for different obstacle sizes

Table 9. The change of customer count with the fixed obstacle count-size ("100 (10 × 10)" / the obstacle radius is 10)

CC		10	25	40	55	70	85	100
RC		1	1	2	3	3	4	4
TL	Avg	1981	2696	3429	4015	4540,2	5172,7	5728,6
	SD	0	0	0	0	4,94	59,68	25,21
RT	Avg	1,4	49,4	355,4	585,7	1899,7	3731,5	5762,9
	SD	4,2	104,3	186,39	356,59	583,07	875,34	622,81

CC: customer count / RC: route count
TL: total length / RT: run-time during the genetic process (ms) / SD: standard deviation

Table 10. The change of obstacle count with the fixed customer count and obstacle size (100 / the obstacle radius is 10)

OC		X	1 x 10	2 x 10	3 x 10	4 x 10	5 x 10	6 x 10	7 x 10	8 x 10	9 x 10	10 x 10
		0	10	20	30	40	50	60	70	80	90	100
GC		0	22	33	44	55	66	77	88	99	110	121
OFR		0	1,02	2,05	3,07	4,09	5,12	6,14	7,17	8,19	9,21	10,24
TL	Avg	4863,9	5174,9	6144,5	5258,3	5715,8	5281,2	6018,7	6113,9	5898,2	5828,5	5447,9
	SD	34,57	33,27	56,91	130,05	29,42	52,32	47,6	48,17	39,81	26,65	91,03
RT	Avg	6608,8	6018	7020,8	5787,4	6506,8	6204	6094,8	6119,1	5915,2	6629	6959,8
	SD	732,48	902,24	1170,6	1070,84	1425,56	542,79	1150,55	1101,82	1201,11	576,71	433,17

OC: obstacle count / GC: guide count / OFR: obstacle occupancy (%)
TL: total length / RT: run-time during the genetic process (ms) / SD: standard deviation

Table 11. The change of obstacle size with the fixed customer-obstacle counts (100 / "100 (10 × 10)")

OR		0	1	2	3	4	5	6	7	8	9	10
OFR		0	0,10	0,41	0,92	1,64	2,56	3,68	5,02	6,55	8,29	10,24
TL	Avg	4863,9	5127,5	5154,3	5162,4	5239,9	5315,9	5351,4	5403,9	5515,1	5548,8	5657,4
	SD	34,57	25,17	27,63	17,52	24,01	21,98	15,08	20,67	35,57	26,6	41,44
RT	Avg	6608,8	62895,6	61126,7	63413,2	64833,9	60147	61283,4	59522,8	61381,7	59529,4	61816,1
	SD	732,48	3679,75	3016,79	2286,31	2301,07	4298,14	3201,89	2469,57	3169,54	3627,4	3998,45

OR: obstacle radius / OFR: obstacle occupancy (%)
TL: total length / RT: run-time during the genetic process (ms) / SD: standard deviation

In Graph 3, the increase of the obstacle radius increases the *obstacle occupancy* in the environment squarely. For different obstacle sizes, neither the centers of the obstacles nor the locations of the guides change. As the obstacles get larger, the total distance increases linearly. The passable points for the larger sized obstacles are also passable for the smaller sized ones. As the obstacles get smaller for the same set of customer locations, the total distance decreases linearly. While the obstacle radius is zero (*the environment without obstacle*), all customer pairs are *visible to each other*. Therefore, when it becomes 1 (*the environment with obstacles*), the total distance increases dramatically.

4. Discussion and Conclusion

In the experimental study on the CVRP_EWCO, depot-customer locations and obstacle count-size were set for different point sets and different obstacle setups. The optimal set of routes was determined with the minimum cost obtained within a small number of generations (100 on average). The appropriate solutions were able to be achieved within short run-times and the developed algorithm could converge to the best values fast with the successful results.

When the guides are away from the obstacles, longer paths (*total distances*) may become the case. On the other hand, obstacles may be neither identical nor placed in the environment regularly. Therefore, obstacle size, in most cases, may cause bigger differences for the total distances rather than the number of obstacles. The optimization of solution guarantees the length of each route in the solution would be the minimum or near-minimum. To further shorten the length of any route, it may be an alternative to move closer to the obstacles.

As observed, the increase of the number of customers increases the total distance. The increase of the number of customers/obstacles increases the number of calculations fairly since the cases between all customer pairs and all guide pairs are taken into consideration. With the increase of obstacle count-size; (i) the available free space in the environment is reduced, (ii) it might not be possible to reach at the customers through other customers around the

obstacles directly, and (iii) the total distance therefore may increase. On the other hand, depending on the different customer placements, the experiments for the change of the number of obstacles do not affect the outcomes very much (see Section 3.5.1).

In this research, CVRP was addressed for the *environments with circular obstacles* (CVRP_EWCO) as its special case in real-life. With the transformation of the problem into the classical CVRP, the navigations between depot-customer locations were done through the guides at certain distances to the obstacles placed in the environment regularly. Moreover, the optimal solutions could be provided to the problem with the developed *memetic algorithm*. Using the solution method proposed, it will be possible to adapt VRP to the working-areas that may include obstacles through which the vehicles cannot pass. Thus, this problem might be extended to apply in narrow and confined areas such as bus terminals, construction zones, factories or harbors. Furthermore, the solution method proposed can be adapted to solve other VRP types or similar problems.

Acknowledgment

This research is based on the PhD thesis titled "Developing a Solution Method to Capacity Constrained Vehicle Routing Problem for the Environments with Obstacles Using Metaheuristic Algorithms" in 2016 at the Department of Computer Engineering in Ege University, İzmir/Turkey.

Declaration of Ethical Code

In this study, we undertake that all the rules required to be followed within the scope of the "Higher Education Institutions Scientific Research and Publication Ethics Directive" are complied with, and that none of the actions stated under the heading "Actions Against Scientific Research and Publication Ethics" are not carried out.

References

[1] Dantzig, G. B., Ramser, J. H. 1959. The truck dispatching problem. Management Science, 6(1), 80-91.

[2] Boonsam, P., Suthikarnnarunai, N., Chitphaiboon, W. 2011. Assignment problem and vehicle routing problem for an improvement of cash distribution. World Congress on Engineering and Computer Science (WCECS), Vol II, October 19-21, San Francisco, USA.

[3] Qureshi, G., Bajaj, P. R., Puranik, P. V. 2012. Particle swarm optimization with genetic operators for vehicle routing problem. International Journal of Engineering Science and Technology (IJEST), 4(7), 3086-3090.

[4] Yücenur, G. N., Demirel, N. Ç. 2011. A hybrid algorithm with genetic algorithm and ant colony optimization for solving multi-depot vehicle routing problems. Journal of Engineering and Natural Sciences, Sigma 29(3), 340-350.

[5] Wang, C. H., Lu, J. Z. 2009. A hybrid genetic algorithm that optimizes capacitated vehicle routing problems. Expert Systems with Applications, 36(2), 2921-2936.

[6] Yurtkuran, A., Emel, E. 2010. A new hybrid electromagnetism-like algorithm for capacitated vehicle routing problems. Expert Systems with Applications, 37(4), 3427-3433.

[7] Fung, R. Y. K., Liu, R., Jiang, Z. 2013. A memetic algorithm for the open capacitated arc routing problem. Transportation Research Part E: Logistics and Transportation Review, 50, 53-67.

[8] Longo, H., de Aragão, M. P., Uchoa, E. 2006. Solving capacitated arc routing problems using a transformation to the CVRP. Computers & Operations Research, 33(6), 1823-1837.

[9] Luo, J., Chen, M. R. 2014. Improved shuffled frog leaping algorithm and its multi-phase model for multi-depot vehicle routing problem. Expert Systems with Applications, 41(5), 2535-2545.

[10] Stanojević, M., Stanojević, B., Vujošević, M. 2013. Enhanced savings calculation and its applications for solving capacitated vehicle routing problem. Applied Mathematics and Computation, 219(20), 10302-10312.

[11] Tlili, T., Faiz, S., Krichen, S. 2014. A hybrid metaheuristic for the distance-constrained capacitated vehicle routing problem. Procedia - Social and Behavioral Sciences, 109, 779-783, 2nd World Conference on Business, Economics and Management.

[12] Marinakis, Y., Iordanidou, G. R., Marinaki, M. 2013. Particle swarm optimization for the vehicle routing problem with stochastic demands. Applied Soft Computing, 13(4), 1693-1704.

[13] Bortfeldt, A. 2012. A hybrid algorithm for the capacitated vehicle routing problem with three-dimensional loading constraints. Computers & Operations Research, 39(9), 2248-2257.

[14] Cacchiani, V., Hemmelmayr, V. C., Tricoire, F. 2014. A set-covering based heuristic algorithm for the periodic vehicle routing problem. Discrete Applied Mathematics, 163(1), 53-64.

[15] Hà, M. H., Bostel, N., Langevin, A., Rousseau, L. M. 2014. An exact algorithm and a metaheuristic for the generalized vehicle routing problem with

flexible fleet size. Computers & Operations Research, 43, 9-19.

[16] Patle, B. K., Ganesh Babu L, Anish Pandey, Parhi, D. R. K., Jagadeesh, A. 2019. A review: On path planning strategies for navigation of mobile robot. Defence Technology, 15 (4), August, 582 - 606.

[17] Patle, B. K., Parhi, D. R. K., Jagadeesh, A., Kashyap, S. K. 2018. Matrix-Binary Codes based Genetic Algorithm for path planning of mobile robot. Computers & Electrical Engineering, 67, April, 708 - 728.

[18] Nguyen, H. T., Le, H. X. 2016. Path planning and Obstacle avoidance approaches for Mobile robot. IJCSI International Journal of Computer Science Issues, 13 (4), July.

[19] Sudhakara, P., Ganapathy, V., Priyadharshini, B., and Sundaran, K. 2018. Obstacle Avoidance and Navigation Planning of a Wheeled Mobile Robot using Amended Artificial Potential Field Method. Procedia Computer Science, 133, 998 - 1004. International Conference on Robotics and Smart Manufacturing (RoSMa2018).

[20] Hassani, I., Maalej, I., Rekik, C. 2018. Robot Path Planning with Avoiding Obstacles in Known Environment Using Free Segments and Turning Points Algorithm. Hindawi Mathematical Problems in Engineering, Article ID 2163278, 13 pages.

[21] Sun, G., Zhou, R., Di, B., Dong, Z., Wang, Y. 2019. A Novel Cooperative Path Planning for Multi-robot Persistent Coverage with Obstacles and Coverage Period Constraints. Sensors, 19, 1994.

[22] Bai, X., Yan, W., Cao, M., Xue, D. 2019. Distributed multi-vehicle task assignment in a time-invariant drift field with obstacles. IET Control Theory & Applications, 13 (17), Special Issue: Distributed Optimisation and Learning for Networked Systems.

[23] Wang, P., Gao, S., Li, L., Sun, B., Cheng, S. 2019. Obstacle Avoidance Path Planning Design for Autonomous Driving Vehicles Based on an Improved Artificial Potential Field Algorithm. Energies, 12, 2342.

[24] Moscato, P., Cotta, C. 2003. A gentle introduction to memetic algorithms. pp 105-144. Glover, F., Kochenberger, G. A., ed. Handbook of Metaheuristics, Springer US, 57, Boston MA, 560p.

[25] Uğur, A. 2008. Path planning on a cuboid using genetic algorithms. Information Sciences, 178(16), 3275-3287.

[26] Chand, P., Mohanty, J. R. 2013. Solving vehicle routing problem with proposed non-dominated sorting genetic algorithm and comparison with classical evolutionary algorithms. International Journal of Computer Applications (IJCA), 69(26), 34-41.

[27] Networking and Emerging Optimization. 2013. Capacitated VRP Instances | Vehicle Routing Problem. http://neo.lcc.uma.es/vrp/vrp-instances/capacitated-vrp-instances/ (Access Date: 03.10.2020).

[28] Computational Infrastructure for Operations Research. 2003. Vehicle Routing Data Sets. http://www.coin-or.org/SYMPHONY/branchandcut/VRP/data/index.htm.old (Access Date: 03.10.2020).